

BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM

VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR

IRÁNYÍTÁSTECHNIKA ÉS INFORMATIKA TANSZÉK

MÉLYTANULÓ ALGORITMUSOK ALKALMAZÁSA AGY-GÉP INTERFÉSZEK FEJLESZTÉSÉHEZ

TDK dolgozat

KÉSZÍTETTE

Rácz Melinda FIGCJ1

KONZULENS

Dr. Márton Gergely (MTA-TTK)

Dr. Harmati István (BME-IIT)

2018

TARTALOM

1. Motiváció és célok.....	3
2. Bevezető.....	5
3. Elméleti háttér	6
3.1. Alapfogalmak	6
3.2. Alapinformációk a felhasznált adatokról	8
3.2.1. A mérőeszköz leírása.....	8
3.2.2. Az adatok leírása.....	8
3.3. Konvolúciós hálók.....	9
3.3.1. Neurális hálók.....	9
3.3.2. Konvolúciós hálók	12
4. Mérések és eredmények	15
4.1. Alkalmazott eszközök	15
4.2. Aktivitások osztályozása adatkeretek alapján.....	15
4.2.1. Adatok előfeldolgozása	15
4.2.2. A használt hálók kiválasztása.....	19
4.2.3. Az egyes felvételekre kapott eredmények.....	22
4.3. Aktivitások detektálása időszeletek alapján	26
4.3.1. Adatok előfeldolgozása	26
4.3.2. A használt háló kiválasztása	27
4.3.3. Teszteredmények.....	29
5. Összegzés.....	31
6. Hivatkozások.....	32

1. MOTIVÁCIÓ ÉS CÉLOK

Az elektronikus eszközök tisztán gondolatokkal történő vezérlése lehetségessé vált úgynevezett agy-gép interfész rendszerek által, ezek segítségével egyrészt mozgássérült személyek végtagprotéziseket és tolószékeket irányíthatnak [21] [18], továbbá bezártság-szindrómában szenvedő betegek számára biztosíthatnak újfajta kommunikációs csatornákat [17]. Léteznek fejbőrre vagy agyfelszínre helyezett érzékelőkön alapuló interfészek, de a hasznos agyi jelek kinyerésének leghatékonyabb módja az idegszövetbe ültetett elektródok alkalmazása [16] [23]. Jelenleg azonban utóbbi módszeren alapuló rendszerek sem képesek azt a hatékonyságot biztosítani, mellyel például egy robotkéz egy természetes végtaghoz hasonló gyorsasággal, precizitással és ügyességgel vezérelhető lenne.



1. ÁBRA: MOZGÁSSÉRÜLT SZEMÉLY IDEGSZÖVETBE ÜLTETETT MULTIELEKTÓDRA ÉPÜLŐ AGY-GÉP INTERFÉSZ RENDSZER SEGÍTSÉGÉVEL EGY ROBOTKART VEZÉREL [18]

Egy szövetbe ültetett szenzor – jellemzően mikrotechnológiával kialakított multielektórod-rendszer [20] - minél több környező sejt aktivitását képes elkülöníteni a mért jelekből, a szenzor által biztosított felvételek információtartalma annál nagyobb lesz, és ez az elképzelt mozgások, vagy akár komplexebb gondolatok továbbítására alkalmas sávszélességet is meghatározza. Az elmúlt években különböző kutatócsoportok olyan sokcsatornás multielektórodokat fejlesztettek, melyek több száz, akár ezret meghaladó kontaktuspontot tartalmaznak, ezek segítségével pedig igen pontos képet kaphatunk az érzékelőket környező idegi aktivitásokról [1] [22] [19]. Probléma azonban, hogy az idegi jelek valós idejű feldolgozása nehezen tart lépést ezen új eszközök által szolgáltatott adatmennyiséggel. Valamint ugyan az egyes makroszkopikus agyi régiók többnyire elkülöníthetőek funkciók szerint, mikroszkopikus szinten minden felhasználónak az ujjlenyomathoz hasonlóan igen egyedi módon jelennek meg a gondolatai, ráadásul az idegsejtek holisztikusan kódolják az információt. Ezért az agy-gép interfészek felhasználói által kiadott gondolati utasítások dekódolásához személyre szabott, adaptív rendszerek kifejlesztése van szükség. Célunk, hogy a

napjainkban gyorsan fejlődő kép, beszéd és videofelvételek elemzésére kiválóan alkalmas tanulóalgoritmusok az agyi jelek klasszifikációjához is robusztus módon alkalmazhatóak legyenek. Ez a dolgozat arra tesz kísérletet, hogy sokcsatornás elektódrendszerrel felvett agyi jelekben található egysejt-aktivitásokat („tüskéket”) elkülönítse, mely folyamat kritikus eleme egy idegszövetbe ültetett multielektrodokon alapuló agy-gép interfésznek [24].

2. BEVEZETŐ

A dolgozat olyan mélytanuló (deep learning) algoritmusokat alkalmazó módszert mutat be, mely a Magyar Tudományos Akadémia Természettudományi Kutatóintézetének Kognitív Idegtudományi és Pszichológiai Kutatóintézetével együttműködésben idegszövetbe ültethető elektródák segítségével rögzített jelek automatikus feldolgozását célozza.

A használt elektródák 32 sorban és 4 oszlopban elhelyezkedő érzékelő elemeket tartalmaznak, így a feldolgozandó (altatásban levő rágcsálók hallókérgi aktivitását monitorozó) felvételek 128 csatornán tartalmaznak adatot. Egy elektróda közelében körülbelül 30–50 olyan idegsejt található, melynek aktivitási mintázata jól elkülöníthető. A Keras és TensorFlow keretrendszerek felhasználásával fejlesztett algoritmusnak képesnek kell lennie a felvétel rögzítését végző elektróda környezetében elhelyezkedő idegsejtek aktivitásának detektálására, illetve azok megkülönböztetésére aszerint, mely idegsejttől származnak.

Az egyes felvételekben található sejtaktivitások manuálisan felcímkézettek, így az adatsorok felhasználhatók az algoritmus felügyelt tanításához. Nehézséget jelent az egyes klaszterek (adott neuronhoz tartozó aktivitások) esetleges térbeli átlapolódása. Figyelembe véve, hogy a későbbi alkalmazhatóság szempontjából előnyös az algoritmus célhardveren történő futtathatósága (pl. esetleges későbbi on-line adatfeldolgozáshoz), az adatsorokon a lehető legkisebb előfeldolgozást célszerű végrehajtani. A módszernek továbbá a detektálandó elemek esetleges kisebb mértékű torzítása esetén is robusztusnak kell lennie. Ezért konvolúciós hálókat alkalmaztunk. A fenti követelményekkel összhangban jelentkezik a feladat megoldását megvalósító háló viszonylagos egyszerűsége – szoftver túlzott komplexitása (kis teljesítményű hardver esetén jelenthet problémát), illetve túlillesztés (a háló bonyolultságából következő elégtelen generalizációs képesség) elleni védelem.

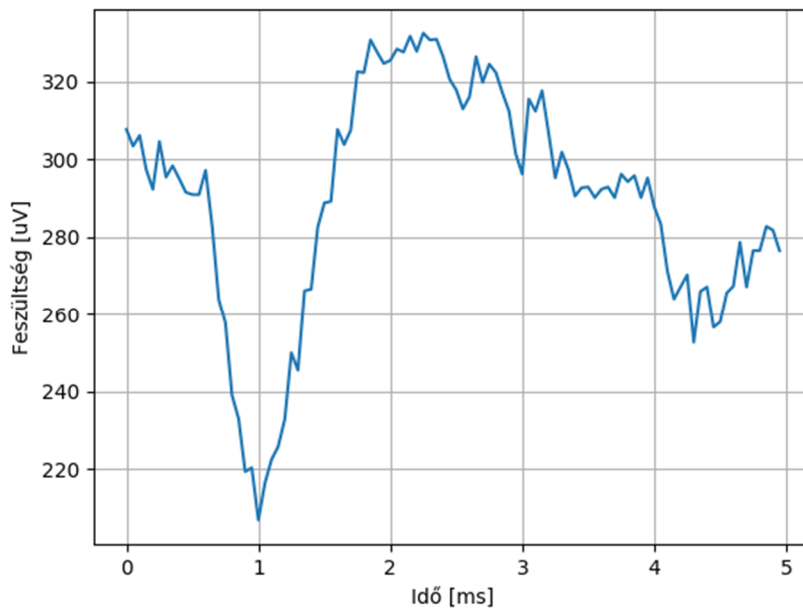
Az elért eredmények alapján a hálóval egy adatfolyamból például a 23 legnagyobb amplitúdót eredményező idegsejt jelei 85 %-ot meghaladó hatékonysággal detektálhatóak és különíthetőek el egymástól.

Kulcsszavak: tüskedetektálás, spike detection, mélytanuló algoritmus, deep learning, konvolúciós háló, neurális háló

3. ELMÉLETI HÁTTÉR

3.1. Alapfogalmak

Tüske: a detektálni/osztályozni kívánt idegi aktivitás; egy adott idegsejt (ezt egységnek vagy klaszternek is nevezik) akciós potenciáljának feleltethető meg. Egy tüske időbeli hossza hozzávetőlegesen 5 ms. Időbeni lefutását a 2. ábra mutatja.



2. ÁBRA: IDEGI AKTIVITÁS JELLEGGÖRBÉJE

Keret: az elektróda 128 csatornái értékeinek összessége egy megadott időpillanatban (az elektródáról készített „pillanatfelvétel”). Egy ilyen keretet mutat a 3. ábra (sárgával a nagyobb, késsel az alacsonyabb feszültségeket jelölve).¹

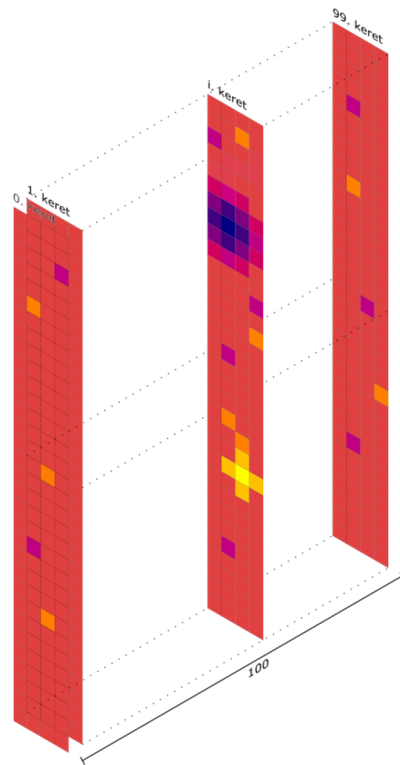


3. ÁBRA: KERET JELLEMZŐ KÉPE

Időszlet: egy időszakban rögzített keretek összessége (az elektródáról készített rövid „videó”).

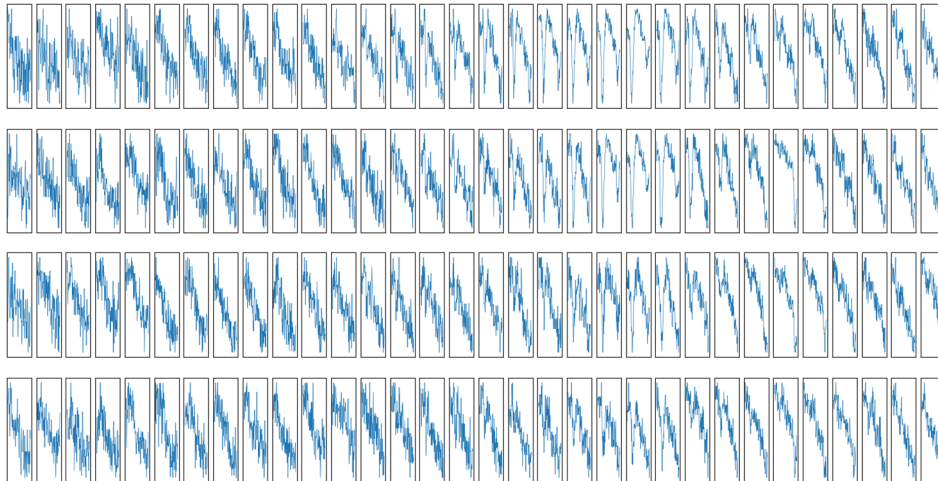
A 4. ábra egy időszletet szemléltet. Ha a bemeneten kisméretű szűrést végeznénk, a közepén látható pillanatfelvételen szereplő kék foltként láthatnánk egy aktivitás maximumát, a sárga folt pedig egy már lecsengő aktivitás képe lenne.

¹ A kép 90°-kal elforgatásra került a jobb helykihasználás érdekében: a 0. csatorna a bal felső, a 127. a jobb alsó.



4. ÁBRA: IDŐSZELET ILLUSZTRÁCIÓJA

Az 5. ábra is egy időszelést mutat, az egyes csatornáknak megfelelően mátrixba rendezve.²



5. ÁBRA: EGY IDŐSZELET CSATORNÁKRA BONTVA

Pozitív adat: azon adathalmaz, ami olyan kereteket/időszelleteket tartalmaz, amiben a detektálni/osztályozni kívánt adat megtalálható (pl. a 3. és az 5. ábra is ilyen adat – előbbin az aktivitás ugyan nem látható, mivel nem végeztünk alacsony frekvenciás szűrést, utóbbi jobb felső részén több csatornán is jól látható az aktivitásra jellemző mintázat).

² Az ábra értelmezése azonos a 3.-éval.

Negatív adat: azon adathalmaz, ami olyan kereteket/időszeleteket tartalmaz, amiben a detektálni/osztályozni kívánt adat nem található meg.

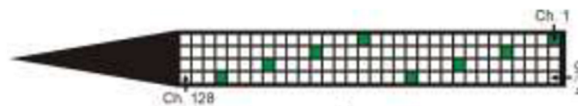
Címke: adott adathoz/adathalmazhoz társítható információ, arra vonatkozik, hogy az adat/adathalmazban található adatok pozitívak vagy negatívak.

3.2. Alapinformációk a felhasznált adatokról

3.2.1. A mérőeszköz leírása

Az adatok rögzítéséhez használt mérőtűk szilíciumból készültek, rajtuk egy 32 x 4 érzékelő elemből álló titánium-nitrid érzékelő mátrix található. A hasznos felület maximalizálása érdekében az érzékelő elemek porózusak.

A mérőtűk mérete 8 mm x 100 μ m x 50 μ m (hosszúság-szélesség-vastagság); az érzékelő elemek 20 μ m x 20 μ m-es négyzetek, impedanciájuk 1 kHz-en hozzávetőlegesen 50 k Ω . Az elektródák között 22,5 μ m szélességű szabad felület található. Egy ilyen elektróda sematikus képét mutatja a 6. ábra.



6. ÁBRA: AZ ADATOK RÖGZÍTÉSÉHEZ HASZNÁLT MÉRŐTŰ FELÉPÍTÉSE [1]

3.2.2. Az adatok leírása

A konvolúciós hálókkal feldolgozandó adatsorokat a Magyar Tudományos Akadémia Természettudományi Kutatóközpont Kognitív Idegtudományi és Pszichológiai Kutatóintézetének munkatársai készítették [1].

Az adatok altatásban levő rágcسالók hallókérgében történő idegi aktivitásnak feleltethetők meg. Egy felvétel természetesen több neuronhoz tartozó tüskét tartalmaz; ezeket a kutatók manuálisan szétválogatták, külön fájlokban listázva a tüskék maximumának időpontjait. E fájlkat az algoritmusok írásakor az adatok felcímkzésére használtuk fel.

Az adatrögzítés során használt mintavételi frekvencia 20 kHz. Az adatsorok sávszélessége 0,1–7500 Hz. Az adatok az őket tartalmazó .dat fájlban a rögzítés időpontjának, azon belül az elektródák csatornáinak sorrendjében helyezkednek el, kétbájtos, előjeles egész számként ábrázolva (LSB = 0,195 μ V).

Összesen 9 adatfájlt használtunk fel, amelyek egyenként 23–46 neuron aktivitásait tartalmazták.

3.3. Konvolúciós hálók

3.3.1. Neurális hálók

A neurális hálók olyan matematikai struktúrák, amelyek függvények közelítésére használhatók fel. Ezek a függvények lehetnek folytonos (pl. regresszió), vagy diszkrét (pl. a bemenet valahány kategóriába sorolása) értékűek.

3.3.1.1. A neurális hálók alapstruktúrája

Neurális hálónak nevezzük az alábbi matematikai struktúrákból (mesterséges neuronok) összeállított rendszereket [1]:

$$y = \sigma \sum_{i=0}^n w_i x_i,$$

ahol:

- x_i a neuron bemenetei (a 0. indexnek megfelelő elemet ofszetnek – bias – nevezik; gyakran elhagyják);
- w_i a neuron bemeneteit súlyozó tényezők (az ofszet súlya általában 1), a háló tanítása során ezeket a paramétereket hangoljuk – értékük inicializáláskor általában kis véletlenszám (hatással lehet a végeredményre is);
- $\sigma(\cdot)$ pedig az aktivációs függvény; szolgáltathat a bemenetével arányos (pl. identitás), vagy valamilyen telítődéssel rendelkező kimenetet (ez lehet sima – pl. tangens hiperbolicus) vagy nem sima – pl. egységugrás.

A neuronok rétegekbe szervezhetők: ezek tartalmazhatnak egy vagy több neuront, mindegyik az előző réteg kimeneteit kapja bemenetül; funkció szempontjából feloszthatók bemeneti, rejtett, illetve kimeneti rétegekre.

Az univerzális approximáció tétele szerint megfelelően nagy neuronszámú, 1 rejtett rétegű neurális hálóval tetszőleges mértékben közelíthető bármely kompakton folytonos függvény [4].

A mesterséges neuronok a „valódi” idegsejtek leegyszerűsített matematikai modelljeinek tekinthetők: az idegsejtek is rétegekbe szerveződnek, a szomszédaitól kapott jeleket adaptív módon súlyozzák és tüzelési mechanizmusuk is nagyban hasonlít az általunk alkalmazott aktivációs függvényekre.

3.3.1.2. Költségfüggvények és optimalizálók

A neurális hálók fontos összetevői a költségfüggvények és az optimalizálók. A neurális hálók statisztikai elven működnek: ha ismert a bemenet és az azokra adandó kimenet, a hálók súlyait úgy állítják, hogy a már látott bemenetekhez hasonló inputokra az elvárt kimenethez hasonló eredményt állítson elő.³ Természetesen, minél több bemenet/kimenet párt (ezeket tanító adatoknak nevezzük) „látott” már a háló (és minél többször) az eredmény annál biztosabban közelíti az elvárt kimenetet (feltételezve, hogy a tanító adatok megfelelően reprezentálják a megvalósítandó leképezést).

A költségfüggvények a háló által szolgáltatott kimenet és az elvárt kimenet eltéréseinek függvényei; azt fejezik ki, hogy a hiba milyen mértékben „büntetendő”, vagyis milyen mértékben szükséges a súlyok állítása a hiba függvényében.

A Keras függvénykönyvtár által (általunk is) használt költségfüggvények [5]:

- mean_squared_error: $u = \overline{(y - y_{elvárt})^2}$, $\overline{(\cdot)}$ a középérték operátor
- mean_absolute_error: $u = \overline{|y - y_{elvárt}|}$
- mean_absolute_percentage_error: $100 \cdot \overline{\left| \frac{y_{elvárt} - y}{\max(y_{elvárt}, \varepsilon)} \right|}$, $\varepsilon > 0 \Rightarrow \max(y_{elvárt}, \varepsilon) > 0$
- mean_squared_logarithmic_error: $u = \overline{(\ln \max(y, \varepsilon) - \ln \max(y_{elvárt}, \varepsilon))^2}$
- squared_hinge: $u = \overline{(\max(1 - y \cdot y_{elvárt}, 0))^2}$
- hinge: $u = \overline{\max(1 - y \cdot y_{elvárt}, 0)}$
- categorical_hinge: $\max(1 + \max(1 - y_{elvárt}) \cdot y - \sum y_{elvárt} y, 0)$, $\sum(\cdot)$ a mátrix összes elemének összegzése
- logcosh: $u = \overline{|\ln \text{ch}(y - y_{elvárt})|}$ (elméletileg; a gyakorlatban kicsit másképp implementált)
- categorical_crossentropy (a TensorFlow backend biztosítja)
- binary_crossentropy (a TensorFlow backend biztosítja)
- kullback_leibler_divergence: $u = \sum y_{elvárt} \ln \frac{\min(\max(y_{elvárt}, \varepsilon), 1)}{\min(\max(y, \varepsilon), 1)}$
- poisson: $u = \overline{y - y_{elvárt} \cdot \ln(y + \varepsilon)}$
- cosine_proximity: $u = -\sum \|y_{elvárt}\|_2 \cdot \|y\|_2$

³ Ez természetesen csak a felügyelt tanulás esetére vonatkozik, amiről ebben a dolgozatban is szó van.

A költségfüggvények mindegyike konvex, vagyis létezik minimuma, amit (mint egy völgy legmélyebb pontját) elérni igyekszünk a neuron súlyainak változtatása által.

A neuronok súlyainak állítása a hibavisszaterjesztés módszerével történik: kiszámítjuk a költségfüggvény gradiensét (az egyes súlyok szerinti parciális deriváltjait), majd a gradiens egyes tagjainak valamilyen függvényével módosítjuk az adott súly értékét⁴ – ezek a függvények az optimalizálók.

Az optimalizálóknak három fő csoportja létezik [3]:

- batch optimalizálók (determinisztikus, vagy off-line módszerek) – a tanítóadat-címke párok egészére hajtják végre a paraméterek frissítését;
- stochasztikus optimalizálók (on-line módszerek) – minden tanítóadat-címke pár után végrehajtják a paraméterek frissítését;
- a két szélsőség közé eső (minibatch) módszerek, amelyek a tanító adatok valamely kisebb-nagyobb halmazára hajtják végre a frissítést.

A Keras könyvtár optimalizálói:

- SGD (Stochastic Gradient Descent): on-line/minibatch módszer,
$$w_{k+1} = w_k + \eta \nabla_w \frac{1}{n} \sum_{i=1}^n u(f(x_i, w), y_i),$$
 ahol w_k a súlyok vektora az aktuális időpillanatban, x_i és y_i a tanítóadat-elvart kimenet párok az adott batchben, η pedig a bátorsági tényező (vagy tanulási arány, az optimalizáló hangolható paramétere), segítségével befolyásolható az algoritmus konvergenciája, illetve annak sebessége (kis érték esetén a konvergencia lassú, de stabil, nagy értéke esetén pedig lehet gyors, de nem feltétlenül stabil);
- Adagrad (adaptív gradiens) módszer [7]: hasonlít az SGD-re, de a bátorsági tényezőt folyamatosan csökkenti (a korábbi gradiensek négyzetösszegének gyökével osztja), így biztosítva, hogy a hiba értéke kezdetben gyorsan csökkenjen, később pedig ne oszcilláljon;
- RMSprop, Adadelta [6]: az Adagrad továbbfejlesztései, a korábbi gradiensek négyzetének összegzését exponenciálisan súlyozott mozgóátlaggal váltják fel (a nagyon régi értékeket az optimalizáló „elfelejti”), ez az ellen védi az algoritmust, hogy a költségfüggvény „laposabb” részein túl lassan konvergáljon;
- Adam (adaptív momentum) [8]: az RMSprop momentummal történő kiegészítése; a momentum olyan mennyiség, ami az eddig (valamilyen exponenciális súlyozással) összegzett

⁴ Szemléletesen a hibavisszaterjesztés azt jelenti, hogy meghatározzuk, az egyes súlyok milyen mértékben járulnak a kimenet hibájához, majd ennek megfelelően állítunk rajtuk.

negatív gradiensek irányába mutat (ahogy a newtoni mechanika momentuma a mozgó testek esetében), ez szintén a tanulási sebesség növelését segíti elő (főként kis, vagy zajos gradiensek esetén);

- Adamax [8]: az Adam továbbfejlesztett verziója, de a gradiens korábbi értékeit nem L^2 norma szerint összegzi, hanem végtelen norma szerint (a korábbi négyzetösszeg-képzés helyett a gradiens abszolút értékét veszi figyelembe exponenciális súlyozással);
- Nadam: az Adam Nesterov-momentummal történő kiegészítése; a Nesterov-momentum abban különbözik az egyszerű momentumtól, hogy a momentum képletében a gradiens pillanatnyi értékét a paraméterek előző pillanatbeli momentumával eltolta pontban számolja, így mintegy „jósolni” képes (a momentum értékének kiszámítása a gradiens tényleges kiszámítása előtt történik), a „mozgó testet” lelassítani, még mielőtt az újból egy „emelkedőhöz” érne.

3.3.2. Konvolúciós hálók

A neurális hálókat több területen, így a képfeldolgozás területén is sikerrel alkalmazzák. Előnyként jelentkezik például a képfeldolgozás során, hogy a priori ismeretekre, pl. a bemenet bonyolult előfeldolgozására, lényeges jellemzők a nyers adatból „kézzel” történő kinyerésére nincs szükség, mert a hálók képesek megtanulni azokat [14] [15].

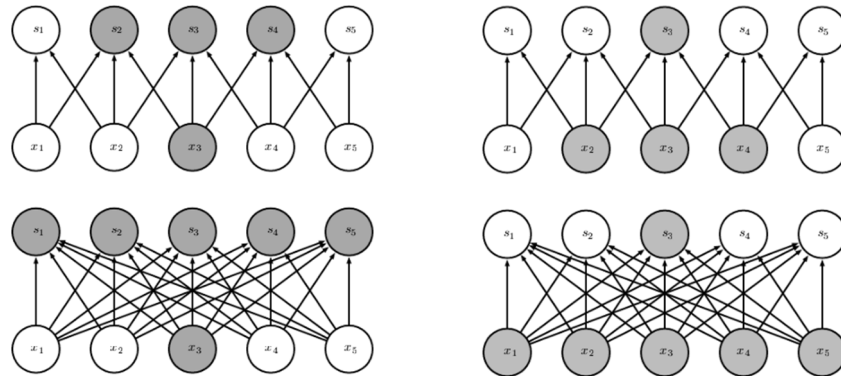
Azonban az általános neurális hálók rétegeinek bemenetein megjelenik az őket megelőző réteg összes kimenete, amelyekhez megfelelő súlyokat kell találni – sok neuront tartalmazó hálók esetén ez hátrányos lehet.

E probléma megoldására (vagy legalábbis enyhítésére) alkalmazzák a konvolúciós hálókat, amelyek a bemeneteken kapott adatok jellemzőinek lokális voltát használják ki: az esetek nagy részében elegendő, ha a szomszédos neuronok között van kapcsolat, viszont a számításokat nagymértékben leegyszerűsíti. A konvolúciós hálók működése nagymértékben hasonlít az elsődleges látókéregére (pontosabban az inspirációt létrehozásukra neurológiai vizsgálatok jelentették).

A konvolúciós hálók széleskörűen alkalmazottak képfeldolgozási célokra, olyan bonyolult mintázatok keresésére [9], felismerésére a bemenetben, mint akár az emberi arc, vagy olyan objektumok, amelyek többféle formát is felvehetnek (pl. azonos állatfajhoz tartozó különböző fajták, vagy formatervezett bútorok). Jó példa ilyen hálókra a Google képkeresője.

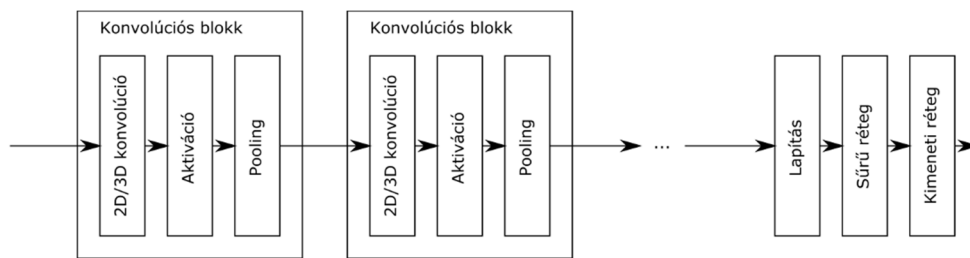
Joggal feltételezhetjük tehát, hogy a neurális hálók alkalmasak lehetnek neurális aktivitások detektálására (akár EEG jelből [12]) és osztályozására [11], mivel ez a feladat is visszavezethető mintaillesztésre.

A konvolúciós és a sűrű rétegek közötti különbséget (egy bemenet hatása a kimenetekre, egy kimenetre hány bemenet van hatással) a 7. ábra szemlélteti (felül a konvolúciós, alul a sűrű réteg látható).



7. ÁBRA: KONVOLÚCIÓS ÉS SŰRŰ RÉTEG ÖSSZEHOSSZOLÍTÁSA [3]

Az általunk alkalmazott konvolúciós háló szerkezetét a 8. ábra mutatja. Az egyes rétegek rövid leírása olvasható az alábbiakban.

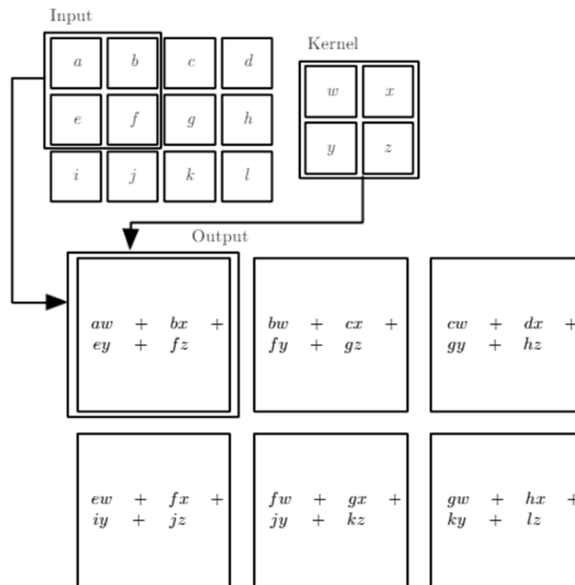


8. ÁBRA: KONVOLÚCIÓS HÁLÓ RÉTEGSTRUKTÚRÁJA

3.3.2.1. Konvolúciós réteg

Azt, hogy adott rétegbeli neuron az előző rétegnek hány kimenetét használja fel, a kernelmérettel definiáljuk.

A kernel lényegében egy súlyozó mátrix, vagy szűrő, ami a bemenet minden pontjára kiszámítja a „közvetlenül felette levő” neuron és annak valahány szomszédja lineáris kombinációját. Ez az érték kerül a réteg aktivációs függvényének bemenetére. A konvolúció műveletét szemlélteti a 9. ábra.



9. ÁBRA: KONVOLÚCIÓ 2D ADATOKON [3]

Egy adott konvolúciós réteghez több szűrő is tartozhat, amelyek a réteg bemenetének különböző jellemzőinek megtanulására lehetnek alkalmasak; ezek számát nevezzük a konvolúciós réteg mélységének.

3.3.2.2. Pooling, stride

A pooling és a stride mechanizmusok a háló egyszerűsítéséhez járulnak.

A pooling a konvolúciós réteg kimenetének speciális alulmintavételezése (kis környezetben található maximum- vagy átlagszámítás a kimenet). A pooling a bemenet kismértékű eltolásaira is érzéketlenné teszi a hálót.

A stride a konvolúciós réteg bemenetének alulmintavételezése (a konvolúció végrehajtásakor mekkorákat „ugrik” a szűrő – pl. kétdimenziós konvolúció esetén a (2, 3)-as stride azt jelenti, hogy a kép egyik irányában minden második, a másik irányban minden harmadik pozícióba eső képpontra számoljuk ki a konvolúciót). A stride abból a szempontból hatékonyabb az egyszerű alulmintavételezésnél, hogy az amúgy eldobott értékeket nem szükséges a hálónak kiszámítania.

3.3.2.3. Sűrű réteg

A konvolúciós háló kimeneti rétege (előfordul, hogy az utolsó előtti is) sűrű, vagyis minden neuronja kapcsolatban áll bemeneteivel; ez a réteg végzi a tulajdonképpeni detektálást/osztályozást. Többdimenziós konvolúciós rétegek előtt egy lapító réteg előzi meg.

4. MÉRÉSEK ÉS EREDMÉNYEK

4.1. Alkalmazott eszközök

A konvolúciós hálókat tanító/tesztelő scripteket Python nyelven implementáltuk JetBrains PyCharm fejlesztőkörnyezetet használva. Az összetettebb matematikai műveletek végrehajtásához a NumPy függvénykönyvtárat alkalmaztuk, a neurális hálók és a hozzájuk kapcsolódó műveleteket (optimalizáció, költségfüggvények, stb.) pedig a Keras függvénykönyvtár szolgáltatta.

Az osztályozást végző programokat egy Lenovo ThinkPad X230 Tableten írtuk meg és futtattuk. Miután a laptop az alaplagra integrált, relatíve egyszerű videokártyával rendelkezik, az algoritmusok a CPU-n futottak. A laptop CPU-ja Intel i5-3320M, órajel-frekvenciája 2,60 GHz. A rendelkezésre álló RAM 3,69 GB. A gépen 64 bites Windows 7 fut.

A detektálást végző programok implementálásához/futtatásához egy NVIDIA GeForce GTX1050 videokártyával, Intel i7-4770 (3,40 GHz órajel-frekvenciájú) processzorral, 16 GB RAM-mal és 64 bites Windows 7 operációs rendszerrel rendelkező asztali PC-t alkalmaztunk.

4.2. Aktivitások osztályozása adatkeretek alapján

Feladatunk az egyes neuronokhoz tartozó aktivitások osztályozása volt, mindehhez negatív mintákat ezúttal nem használtunk fel. A használt regisztrátumokban 23–46 klaszter aktivitásai szerepeltek, ez néhány százezernyi aktivitást jelent felvételenként.

4.2.1. Adatok előfeldolgozása

Az egyes aktivitások maximumaihoz tartozó kereteket időrendi sorrendben betöltöttük a memóriából, majd -1 és 1 közé normáltuk azokat az alábbi módon:

$$x' = mx + b,$$

ahol

$$m = \frac{1}{\frac{1}{2}(|\max x| + |\min x|)},$$

$$b = \frac{\max x + \min x}{2}.$$

Az adatokat ezután szétválasztottuk tanító- és tesztadatokra megadott arányban, tanító adatok gyanánt a legkésőbbi adatokat használtuk fel.

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.7789613604545593	0.12098155170679092	0.6322997808456421	0.5613467693328857	0.7812440395355225	0.793418288230896	0.7772493958473206
mean_absolute_error	0.6688225269317627	0.11869888007640839	0.4493056833744049	0.12497622519731522	0.5592543482780457	0.5546889901161194	0.6022446155548096
mean_absolute_percentage_error	0.5948259234428406	0.12516644597053528	0.3422103822231293	0.6011033058166504	0.5881681442260742	0.5893095135688782	0.6703442931175232
mean_squared_logarithmic_error	0.7721133828163147	0.13619935512542725	0.6770020723342896	0.3355526030063629	0.7957009673118591	0.7801027297973633	0.775537371635437
squared_hinge	0.769260048866272	0.13125357031822205	0.5773254632949829	0.39813581109046936	0.7310252785682678	0.7268403768539429	0.7422484159469604
hinge	0.6640669703483582	0.1230739951133728	0.38482025265693665	0.24386532604694366	0.4616701602935791	0.5470800995826721	0.6572189331054688
categorical_hinge	0.7757276296615601	0.15997716784477234	0.7133346199989319	0.7740156054496765	0.7791516184806824	0.7789613604545593	0.7873311638832092
logcosh	0.7759178280830383	0.08921437710523605	0.6574091911315918	0.21609282493591309	0.7880920767784119	0.7534715533256531	0.7854289412498474
categorical_crossentropy	0.7783907055854797	0.6159406304359436	0.6621647477149963	0.7873311638832092	0.768118679523468	0.776678740978241	0.8006467819213867
kullback_leibler_divergence	0.8078752160072327	0.6014837622642517	0.6621647477149963	0.7751569151878357	0.7633631229400635	0.775537371635437	0.7932280898094177
poisson	0.7951303124427795	0.12840022146701813	0.6614038348197937	0.7487159967422485	0.7827658653259277	0.7738253474235535	0.7884725332260132
cosine_proximity	0.8004565238952637	0.2451968789100647	0.6718660593032837	0.7780102491378784	0.7637435793876648	0.7833365201950073	0.7888529300689697

1. TÁBLÁZAT: EGYRÉTEGŰ, 32 SZŰRÖT TARTALMAZÓ HÁLÓ EREDMÉNYEI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.8057827949523926	0.12364466488361359	0.7359710931777954	0.5797983407974243	0.7985543012619019	0.7939889430999756	0.7783907055854797
mean_absolute_error	0.7000190019607544	0.12497622519731522	0.49495911598205566	0.24538710713386536	0.44664257764816284	0.5611565709114075	0.6307780146598816
mean_absolute_percentage_error	0.6507513523101807	0.03500095009803772	0.28932851552963257	0.6267833113670349	0.5891192555427551	0.6452349424362183	0.6830891966819763
mean_squared_logarithmic_error	0.7677382826805115	0.1285904496908188	0.7392048835754395	0.3465855121612549	0.7856191992759705	0.797793447971344	0.7953205108642578
squared_hinge	0.7702111601829529	0.12231310456991196	0.5465094447135925	0.41202205419540405	0.7542324662208557	0.7203728556632996	0.743960440158844
hinge	0.6186037659645081	0.133536234498024	0.4814532995223999	0.24291421473026276	0.6092828512191772	0.5716187953948975	0.6089023947715759
categorical_hinge	0.7595586776733398	0.21932661533355713	0.7536618113517761	0.7983641028404236	0.7743960618972778	0.7960814237594604	0.787140965461731
logcosh	0.7922769784927368	0.12402510643005371	0.7388244271278381	0.23492486774921417	0.7825756072998047	0.7987445592880249	0.7846680879592896
categorical_crossentropy	0.8025490045547485	0.61213618516922	0.7199923992156982	0.7761080265045166	0.7783907055854797	0.7780102491378784	0.797227334976196
kullback_leibler_divergence	0.7605097889900208	0.6157504320144653	0.7243674993515015	0.8074947595596313	0.7671675682067871	0.7842876315116882	0.7789613604545593
poisson	0.7721133828163147	0.13315579295158386	0.7152368426322937	0.7515693306922913	0.7833365201950073	0.7899942994117737	0.7888529300689697
cosine_proximity	0.7953205108642578	0.2606049180030823	0.7382537722587585	0.769260048866272	0.7757276296615601	0.7899942994117737	0.7884725332260132

2. TÁBLÁZAT: EGYRÉTEGŰ, 64 SZŰRÖT TARTALMAZÓ HÁLÓ EREDMÉNYEI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.785999596118927	0.12440555542707443	0.630207359790802	0.6050979495048523	0.7806733846664429	0.7574662566184998	0.7846680879592896
mean_absolute_error	0.5293893814086914	0.12497622519731522	0.2630777955055237	0.2451968789100647	0.5263458490371704	0.42876166105270386	0.12231310456991196
mean_absolute_percentage_error	0.5493627786636353	0.0983450636267662	0.30169299244880676	0.5031386613845825	0.4903937578201294	0.47974130511283875	0.46281149983406067
mean_squared_logarithmic_error	0.7747764587402344	0.11356286704540253	0.6587406992912292	0.3330796957015991	0.7732546925544739	0.7761080265045166	0.7907552123069763
squared_hinge	0.7814342975616455	0.12497622519731522	0.5484116673469543	0.44759368896484375	0.7133346199989319	0.690127432346344	0.7656458020210266
hinge	0.46433326601982117	0.1171770989894867	0.3924291431903839	0.12231310456991196	0.5592543482780457	0.4413163363933563	0.12497622519731522
categorical_hinge	0.7584173679351807	0.12878067791461945	0.5790374875068665	0.7605097889900208	0.7730644941329956	0.7715427279472351	0.7587977647781372
logcosh	0.7740156054496765	0.12478599697351456	0.6431424617767334	0.15731406211853027	0.7460528612136841	0.7709720134735107	0.7753471732139587
categorical_crossentropy	0.7806733846664429	0.6448544859886169	0.66596919298172	0.782956063747406	0.765075147151947	0.7553737759590149	0.7858093976974487
kullback_leibler_divergence	0.7768689393997192	0.6248810887336731	0.6279246807098389	0.7726840376853943	0.7734449505805969	0.7686893939971924	0.7958911657333374
poisson	0.7586075663566589	0.13524824380874634	0.625832200050354	0.7808635830879211	0.7820049524307251	0.7557542324066162	0.7774395942687988
cosine_proximity	0.7957009673118591	0.21590261161327362	0.6634963154792786	0.7521399855613708	0.7679284811019897	0.7858093976974487	0.7863800525665283

3. TÁBLÁZAT KÉTRÉTEGŰ, 16 SZŰRÖT TARTALMAZÓ HÁLÓ EREDMÉNYEI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.769260048866272	0.1221228837966919	0.7268403768539429	0.6005326509475708	0.7844778299331665	0.7839071750640869	0.7960814237594604
mean_absolute_error	0.6808065176010132	0.12497622519731522	0.3119649887084961	0.12497622519731522	0.5828419327735901	0.12231310456991196	0.12231310456991196
mean_absolute_percentage_error	0.5303404927253723	0.021495148539543152	0.42590832710266113	0.6496100425720215	0.6011033058166504	0.3711242079734802	0.6473273634910583
mean_squared_logarithmic_error	0.7962716221809387	0.12497622519731522	0.7298839688301086	0.3836789131164551	0.7960814237594604	0.7827658653259277	0.7924671769142151
squared_hinge	0.7331177592277527	0.12231310456991196	0.5518356561660767	0.4390336573123932	0.7683089375495911	0.7568955421447754	0.79665207862854
hinge	0.4715617299079895	0.11698687821626663	0.4858284294605255	0.24500665068626404	0.12497622519731522	0.5680045485496521	0.6041468381881714
categorical_hinge	0.7821951508522034	0.16397184133529663	0.7274110913276672	0.7814342975616455	0.7928476333618164	0.7559444308280945	0.7559444308280945
logcosh	0.7875214219093323	0.1196499913930893	0.733307957649231	0.19630968570709229	0.7842876315116882	0.7837169766426086	0.7702111601829529
categorical_crossentropy	0.7757276296615601	0.6505611538887024	0.7047745585441589	0.8055925369262695	0.7922769784927368	0.7869507074356079	0.7726840376853943
kullback_leibler_divergence	0.7707818150520325	0.6509416103363037	0.7239870429039001	0.787140965461731	0.7835267186164856	0.7827658653259277	0.7896138429641724
poisson	0.7991249561309814	0.13715046644210815	0.7127639055252075	0.8015978932380676	0.7882822751998901	0.769260048866272	0.7863800525665283
cosine_proximity	0.8173863291740417	0.21095682680606842	0.733307957649231	0.7846680879592896	0.7743960618972778	0.7842876315116882	0.8052120804786682

4. TÁBLÁZAT: KÉTRÉTEGŰ, 32 SZŰRÖT TARTALMAZÓ HÁLÓ EREDMÉNYEI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

		Háló által jelzett neuron																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Ténylegesen aktív neuron	1	417	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	2	0	131	0	0	0	0	0	0	0	0	0	69	0	0	0	0	0	0	0	0	0	0
	3	0	0	63	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	4	0	0	0	33	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	73	0	0
	5	2	0	0	0	207	0	0	0	1	0	0	0	0	0	2	1	0	1	0	0	0	0
	6	0	0	0	2	0	102	0	0	0	0	0	0	1	0	1	0	0	0	2	0	0	8
	7	2	0	0	0	2	0	517	2	129	1	0	0	0	0	0	2	1	0	0	0	0	1
	8	2	0	0	0	1	0	0	333	0	0	0	0	0	0	1	0	1	0	0	0	0	0
	9	1	0	0	0	0	0	129	1	335	0	0	1	6	0	0	1	0	0	0	0	0	0
	10	0	0	1	0	1	0	0	1	0	92	0	0	0	0	0	1	0	0	0	0	0	2
	11	0	0	0	0	1	0	0	0	0	0	103	0	0	0	0	0	0	0	0	0	0	0
	12	110	0	0	0	1	0	2	1	0	0	0	192	0	0	0	0	0	0	0	0	0	0
	13	8	1	0	2	2	0	10	1	31	0	2	2	154	0	20	17	1	21	9	1	1	3
	14	1	0	0	0	0	0	1	0	0	2	3	0	0	58	1	1	0	4	0	0	0	0
	15	0	0	0	0	2	0	0	0	0	0	0	0	0	0	107	0	0	2	0	0	0	0
	16	8	0	0	0	5	0	2	3	1	0	0	0	0	0	1	617	1	4	1	0	0	0
	17	0	0	0	1	0	0	1	0	2	0	0	0	0	0	4	3	171	0	0	0	101	0
	18	5	0	0	0	7	0	0	1	1	1	0	0	0	0	0	0	0	185	0	0	0	1
	19	2	0	0	0	0	1	0	0	1	0	0	0	0	0	1	1	0	3	101	0	0	2
	20	0	0	0	14	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	61	0	0
	21	3	9	0	0	1	0	0	2	0	1	2	0	2	4	15	2	1	2	0	1	278	0
	22	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	20
	23	1	0	0	0	1	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	20

5. TÁBLÁZAT: 4. TÁBLÁZAT RMSPROP/COSINE_PROXIMITY EREDMÉNYÉHEZ TARTOZÓ KONFÚZIÓS MÁTRIX

4.2.2. A használt hálók kiválasztása

Célunk olyan háló kiválasztása volt, ami relatíve gyorsan, relatíve pontosan osztályoz; ennek eléréséhez a háló alábbi paramétereit hangoltuk:

- rétegszám
- rétegenkénti szűrőszám
- stride
- pooling
- kernelméret
- optimalizáló függvény
- költségfüggvény

A konvolúciós és sűrű rétegek aktivációs függvényeként lineáris, a kimeneti sűrű réteg aktivációs függvényeként pedig softmax függvényt használtunk.

A feladatot megvalósító Python kódot az olvasó a Melléklet A 1.1. pontjában találja.

Kiindulás gyanánt az alábbi kernelméret, stride és pooling mellett kerestük az optimális rétegszámot, rétegenkénti szűrőszámot, optimalizálót és költségfüggvényt:






Kernelméret	(4, 4)
Stride	(2, 2)
Pooling	(2, 2)

A lehetséges rétegszám-szűrőszám kombinációk körét az alábbira szűkítettük le:

Rétegszám	Rétegenkénti szűrőszám
1	32
1	64
2	16
2	32

Az adatok első 95 %-át használtuk tanításra, utolsó 5 %-át tesztelésre. Minden mérésnél 30 tanítási ciklussal dolgoztunk.

Az egyes hálókra kapott eredményeket az 1–4. táblázat tartalmazza (az értékek 0 és 1 közötti számok lehetnek). Az egyes táblázatok sorai a 3.3.1.2 szakaszban már ismertetett költségfüggvényeknek, oszlopai pedig az ugyanitt tárgyalt optimalizálóknak feleltethetők meg; az egyes cellák az adott függvények együtt alkalmazásából adódnak. A színek értelmezése:

	$x < 0.5$
	$0.5 \leq x < 0.6$
	$0.6 \leq x < 0.7$
	$0.7 \leq x < 0.8$
	$0.8 \leq x$

Látható, hogy az SGD optimalizáló és a `mean_absolute_error`, a `mean_absolute_percentage_error` és a hinge költségfüggvények nem használhatók erre a feladatra: az esetek nagy részében az őket alkalmazó hálók nem, vagy jóval kisebb mértékben tanultak más optimalizálókat/költségeket használó hálóknál. Az Adagrad optimalizáló hatásfoka szintén elmarad a többitől, de a háló komplexitásának növekedésével javuló tendenciát mutatott: azokkal a költségfüggvényekkel együtt alkalmazva, amikkel megvalósult a tanulás, egyrétegű és 32 szűrős, valamint kétrétegű és 16 szűrős hálókkal az eredmények nagy része a 60–70 %-os, egyrétegű és 64 szűrős továbbá kétrétegű és 32 szűrős hálókkal a 70–80 %-os hatékonysági tartományba esett.

A táblázatokban szereplő értékek természetesen csak durva becslést jelentenek az egyes hálók jóságára; sokkal pontosabb képet kaphatunk hatásfokukról a konfúziós mátrixok vizsgálatával. Azonban az adathalmaz, amin a betanítás a tesztelés kezdeti fázisában zajlott, bár a legkevesebb féle aktivitást szolgáltatva (23 szeparált neuronnal), így is nagyméretű, nehezen olvasható konfúziós mátrixokat eredményezett. Az érdeklődő olvasó a tesztek során készített mátrixokat a Melléklet B dokumentumban találja, a fenti négy táblázat adataihoz tartozókat az 1.1. szakaszban.

Példaként egy mátrixot (az első négy táblázat legjobb értékéhez – 0.8173863291740417 – tartozót – 4. táblázat, `RMSprop/cosine_proximity`) itt is mutatunk (5. táblázat). Sorai azt jelölik, az egyes neuronokhoz hány aktivitás tartozik, oszlopai pedig azt, hogy az adott aktivitást mely neuron aktivitásaként azonosította a háló (a Melléklet mátrixainak értelmezése megegyezik jelen táblázatban szereplővel, viszont a fejlécek hiányoznak). Tökéletes identifikáció esetén csak a mátrix főátlójában szereplnének elemek. Elégtelen tanulás esetén az értékek teljesen szétszóródnak, vagy csak néhány oszlopra koncentrálódnak.

Az eredmények alapján így kiszűrhetjük azokat az optimalizálókat illetve költségfüggvényeket, amelyek a feladat megoldása során nem alkalmazhatók, a többieket a következő utáni lépésben felhasználtuk. Emellett úgy találtuk, hogy a bonyolultabb hálók alkalmazása „nem éri meg”: tanításuk a legegyszerűbb változatnál jóval több időt vett igénybe, viszont nem szolgáltatott annál lényegesen jobb eredményt (az Adagrad optimalizáló esetét kivéve), így a következő lépéseknél már csak az egyrétegű, 32 szűrős hálót használtuk.

Ezt követően a közepesen jó `RMSprop/mean_squared_error` párosítást használva a háló paramétereit kezdtük el hangolni. Tapasztalataink alapján a `stride` és a `pooling` növelése az eredmények romlásával járt együtt, így értéküket (1, 1)-re választottuk.

Az általunk kipróbált lehetséges kernelekre kapott eredményeket a 6. táblázat listázza.

Kernel	Találati arány
(1, 1)	74 %
(2, 2)	78 %
(4, 4)	77 %
(2, 1)	78 %
(4, 2)	81 %
(8, 4)	76 %
(4, 1)	75 %
(8, 2)	79 %
(16, 4)	78 %
(8, 1)	77 %
(16, 2)	81 %
(32, 4)	78 %
(16, 1)	77 %
(32, 2)	78 %
(32, 1)	79 %
(1, 2)	78 %
(2, 4)	77 %
(1, 4)	73 %

6. TÁBLÁZAT: TALÁLATI ARÁNY A KERNELMÉRET FÜGGVÉNYÉBEN

Ezután a két legjobb találati arányt eredményező kernelt használva végimentünk a két lépéssel ezelőtt kiválasztott optimalizálók és költségfüggvények összes párosításán. Tanító adatként ezúttal az adatok első 50 %-át használtuk, tesztadatként a másik felét. A kapott eredményeket a 7. táblázat és a 8. táblázat tartalmazza. A Melléklet B 1.2. szakaszában található az értékekhez tartozó konfúziós mátrixok.

A színek értelmezése:

	$0.5 \leq x < 0.6$
	$0.6 \leq x < 0.7$
	$0.7 \leq x < 0.8$
	$0.8 \leq x < 0.85$
	$0.85 \leq x$

Az eredmények láthatóan javultak az előző, taláalomra kiválasztott kernel használatával kapottakhoz képest: 80 % alatti találati arányt csupán 10 esetben kaptunk (ez nem éri el az összes eset 10 %-át), további 11 esetben a találati arány meghaladta a 85 %-ot is.

A 85 % feletti eredményeket szolgáltató kombinációkat választottuk ki a további használatra. Ezek adatai a 9. táblázatban láthatók.

4.2.3. Az egyes felvételekre kapott eredmények

Az előzőekben kiválasztott hálóknak megtanítottuk az egyes felvételekben szereplő aktivitásokat, a tanító- és tesztadatokat az előbbinek megfelelően osztottuk fel (fele-fele arányban), 30 tanítási ciklust alkalmaztunk.

A feladatot megvalósító kód a Melléklet A 1.2. pontjában található.

A hálók eredményeit az egyes felvételekre a 10. táblázat tartalmazza. A táblázat egyes mezőikhez tartozó konfúziós mátrixok a Melléklet B 1.3. szakaszában található.⁵

Mint az megfigyelhető, egy felvétel kivételével az osztályozás hatékonysága (legalább egy hálóra) 80 % feletti volt, 21 esetben pedig a 85 %-ot is meghaladta; a 2017_05_19__0_002 felvétel esetén minden érték 85 % fölé esett. A legjobb találati arány (0.8877900393154117) is ehhez a felvételhez kapcsolódik (4. háló: Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error), a hozzá tartozó konfúziós mátrix a 11. táblázatban látható.

A 2016_10_19__0_002 felvétel kiértékelésekor a találati arány 71,6 és 78,5 közé esett, továbbá négy esetben előfordult, hogy az osztályozást végző háló egyszerűen nem tanult (minden mintát egy neuron aktivitásának vett). Hasonló, bár valamivel jobb eredményeket produkáltak a hálók a 2017_04_05__0_002 felvétel esetén is: ténylegesen tanuló hálók esetén az eredmények 78,95 és 84,6 % közé estek.

Ennek oka nem tudjuk, mi lehet, mivel a felvétel sem a benne előforduló neuronok számát, sem minőségét tekintve nem tér el lényegesen a többitől. Figyelemre méltó a tény, hogy a négy nem tanuló háló mindkét esetben egyezett, viszont eredményeik nem: mindannyian egy adott neuron aktivitásaiként azonosították a felvételben szereplőket, viszont ez a neuron mindkét felvétel esetén minden háló esetén különbözött.

⁵ A felvételek címe a 10. táblázatban rövidített formában látható. A teljes címformátum: ARat_éééé_hh_nn__0_002.dat (a Mellékletben is így szerepel)

	RMSprop	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.8528566956520081	0.831015408039093	0.8040752410888672	0.8307490348815918	0.8170887231826782	0.8394056558609009
mean_squared_logarithmic_error	0.850307285785675	0.8434580564498901	0.7525922060012817	0.8285801410675049	0.8166701793670654	0.8321568965911865
squared_hinge	0.8471871018409729	0.6779931783676147	0.5960693359375	0.8262209892272949	0.821921169757843	0.8300641179084778
categorical_hinge	0.838663637638092	0.8328037858009338	0.827590823173523	0.8444473743438721	0.8310534358024597	0.7959133386611938
logcosh	0.821369469165802	0.8372176885604858	0.6360229253768921	0.8265634179115295	0.8493369817733765	0.8167272210121155
categorical_crossentropy	0.8487091064453125	0.833564817905426	0.842525839805603	0.8072335124015808	0.8213884830474854	0.8356766700744629
kullback_leibler_divergence	0.8395197987556458	0.8388729095458984	0.8215407133102417	0.8395769000053406	0.8349917531013489	0.8426399827003479
poisson	0.8341926336288452	0.8402047157287598	0.8351058959960938	0.8254789710044861	0.8359810709953308	0.8035425543785095
cosine_proximity	0.8335838317871094	0.8445234894752502	0.8090789914131165	0.8402618169784546	0.8528566956520081	0.8152052164077759

7. TÁBLÁZAT: (4, 2) SZŰRÖT ALKALMAZÓ HÁLÓ TALÁLATI ARÁNYAI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

	RMSprop	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.82745760679245	0.8159091472625732	0.8122943043708801	0.8392344117164612	0.8378835916519165	0.8190483450889587
mean_squared_logarithmic_error	0.8635299801826477	0.8396149277687073	0.7407393455505371	0.8330511450767517	0.8404901027679443	0.85957270860672
squared_hinge	0.8423736095428467	0.6327505111694336	0.6795342564582825	0.7887406945228577	0.833374559879303	0.8417838215827942
categorical_hinge	0.8079564571380615	0.8226822018623352	0.820189893245697	0.8603527545928955	0.8487091064453125	0.8536557555198669
logcosh	0.8279522657394409	0.8269439339637756	0.7073876261711121	0.8003462553024292	0.8244325518608093	0.8599532246589661
categorical_crossentropy	0.8510112166404724	0.8327657580375671	0.809231162071228	0.8402808308601379	0.8462167978286743	0.8187249302864075
kullback_leibler_divergence	0.8278191089630127	0.8337931036949158	0.8374269604682922	0.8442952036857605	0.8435912728309631	0.8146534562110901
poisson	0.838606595993042	0.8321188688278198	0.8228344321250916	0.8340023756027222	0.8395578265190125	0.8352200388908386
cosine_proximity	0.8150910139083862	0.8371606469154358	0.8633968234062195	0.8447327613830566	0.8503643274307251	0.8106580972671509

8. TÁBLÁZAT: (16, 2) SZŰRÖT ALKALMAZÓ HÁLÓ TALÁLATI ARÁNYAI A KÖLTSÉGFÜGGVÉNYEK ÉS OPTIMALIZÁLÓK FÜGGVÉNYÉBEN

Háló száma	Kernel	Optimalizáló	Költséggüggvény
1	(4, 2)	RMSprop	mean_squared_error
2	(4, 2)	RMSprop	mean_squared_logarithmic_error
3	(4, 2)	Adamax	cosine_proximity
4	(16, 2)	RMSprop	mean_squared_logarithmic_error
5	(16, 2)	RMSprop	categorical_crossentropy
6	(16, 2)	Adadelata	cosine_proximity
7	(16, 2)	Adam	categorical_hinge
8	(16, 2)	Adamax	cosine_proximity
9	(16, 2)	Nadam	mean_squared_logarithmic_error
10	(16, 2)	Nadam	categorical_hinge
11	(16, 2)	Nadam	logcosh

9. TÁBLÁZAT: A LEGJOBBNAK ÍTÉLT KÖLTSÉGFÜGGVÉNYEK

Háló	2016_10_12_0_002	2016_07_18_0_002	2016_10_19_0_002	2017_06_01_0_002	2016_07_27_0_002	2016_07_20_0_002	2017_05_19_0_002	2017_05_31_0_002	2017_04_05_0_002
1	0.8235003352165222	0.8400025613091918	0.7280001236984038	0.8265060680131179	0.8331145696189355	0.836475032567978	0.8779642621133706	0.8041868507862091	0.8459375198870561
2	0.8371606469154358	0.8383100105515909	0.7452219230481948	0.8414617049853834	0.8503144917708609	0.8502910196781158	0.8776392536789271	0.8251784443855286	0.8245160624434852
3	0.8356385827064514	0.8412027946401078	0.785184022909449	0.8486597264645906	0.8409427447207773	0.8484274506568908	0.8806746386245746	0.8180410265922546	0.8462410004246435
4	0.8366659879684448	0.8392684427307852	0.7172988114204499	0.8438330805142561	0.8275387780181769	0.8341978967189789	0.8877900393154117	0.8171871066093445	0.8447892980291942
5	0.8439527153968811	0.8411503598799481	0.7161536565669206	0.8466940859034617	0.8381548427006932	0.8442918419837951	0.8532982807942834	0.8544417560100556	0.7968325621474588
6	0.8501930832862854	0.8190481012026674	0.7505330914998366	0.830626215589257	0.8192433261659039	0.8040794491767883	0.8718412970405099	0.8291358649730682	0.7895436960459148
7	0.8423165678977966	0.8386974551296477	0.022572460704122832	0.8173214119190287	0.8208184067643033	0.846635353565216	0.8764146785455409	0.8242544054985046	0.07619158275358089
8	0.8500979542732239	0.8359707236886949	0.7468063895990876	0.8445605567814557	0.8341803820653844	0.8600735247135163	0.882880062367491	0.8254460871219635	0.8441216136787667
9	0.8233861327171326	0.8294073447241475	0.01847555726430304	0.8511500020239835	0.854220705353276	0.8549525320529938	0.8809822367443723	0.8311942458152771	0.01269101065304614
10	0.8131694793701172	0.8226982991780099	0.0040722240077607046	0.8147262150817888	0.8023426828790797	0.8177882134914398	0.8690555096080806	0.8210489451885223	0.0063176847178528185
11	0.8379406929016113	0.8425486802308857	0.0049903254041638295	0.8313117393359536	0.8347369112473297	0.8517001867294312	0.8753235685835172	0.8276382863521576	0.05896337650714157

10. TÁBLÁZAT: A HASZNÁLT HÁLÓK TALÁLATI ARÁNYAI AZ EGYES FELVÉTELEKRE

Háló által jelzett neuron

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	
1	45	7	39	2	16	42	8	4	5	0	10	8	15	0	0	9	2	0	0	1	29	6	3	0	4	4	10	14	21	5	22	2	134	20	26	6	
2	45	13464	37	8	45	39	23	2	26	54	36	10	38	9	0	5	16	1	1	3	65	15	15	2	0	9	18	31	39	12	35	1	169	80	96	5	
3	16	23	5439	6	9	50	14	0	4	0	15	3	27	1	1	16	13	0	2	2	38	9	6	5	3	5	20	25	16	9	32	0	210	42	23	21	
4	15	12	31	2054	10	47	3	2	14	0	12	4	14	0	0	6	13	0	0	0	20	5	1	5	6	3	8	4	10	6	13	5	108	26	22	2	
5	12	7	9	1	5913	19	11	1	9	0	3	97	19	1	0	12	6	0	0	0	33	10	2	4	0	2	6	16	20	4	11	2	38	26	10	3	
6	9	24	22	61	7	7530	9	0	4	0	6	1	26	4	0	4	12	0	0	0	21	1	6	3	9	8	2	16	18	5	13	5	104	22	24	5	
7	28	47	57	14	14	62	11525	4	18	0	12	4	43	4	0	24	6	0	5	1	45	5	7	0	17	13	7	22	33	18	41	2	207	94	63	10	
8	1	1	3	0	1	5	4	1154	0	0	0	0	4	0	3	0	1	0	0	0	3	1	1	0	0	2	0	0	0	0	0	1	14	4	5	1	
9	9	9	9	3	6	16	3	0	5480	0	0	7	39	3	0	1	6	0	0	1	39	5	1	0	3	0	3	8	17	4	8	16	41	13	23	1	
10	18	1962	23	13	26	29	37	3	20	3372	57	19	4	0	2	0	22	4	1	0	7	51	7	1	5	2	8	30	11	5	28	15	46	47	68	1	
11	76	66	230	30	28	128	88	24	17	4	7676	28	38	8	3	56	98	0	13	2	84	81	21	9	28	6	79	103	53	31	213	9	761	197	102	16	
12	13	4	7	0	124	6	2	1	2	0	1	1119	3	0	0	4	9	0	2	1	12	7	2	1	3	0	3	2	8	1	6	2	20	5	2	5	
13	3	0	3	1	1	2	3	1	2	0	2	0	2820	1	0	13	0	0	0	2	5	0	1	0	0	0	0	2	7	1	5	1	17	31	1	0	
14	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	568	0	0	0	0	1	0	0	0	0	1	2	1	1	0	0	3	2	0	0		
15	12	0	3	6	1	13	4	4	6	5	31	0	0	0	0	2102	1	9	1	0	0	0	1	0	0	5	0	0	9	4	1	255	3	21	7	8	0
16	4	10	12	7	2	5	6	1	3	0	4	4	23	1	0	840	8	0	1	0	3	8	0	0	4	1	3	4	7	1	3	0	25	64	11	0	
17	22	68	57	14	32	53	58	1	19	8	32	23	31	3	4	43	7835	0	3	2	83	11	11	2	52	5	30	44	51	7	36	26	181	49	63	3	
18	4	2	0	1	0	3	6	0	2	0	5	3	1	0	2	0	2	358	0	0	2	2	370	1	0	0	0	2	0	3	8	3	2	3	3	0	2
19	5	1	11	11	8	28	7	4	0	12	81	3	0	0	27	3	10	1	2047	0	0	4	0	0	2	0	0	3	2	1	6	1	2	13	3	2	
20	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	272	1	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	
21	5	8	8	0	9	16	9	0	2	0	4	2	24	0	0	0	98	0	1	0	6046	0	2	0	0	4	1	2	14	2	1	57	40	3	42	3	
22	5	9	6	1	3	7	3	3	1	0	2	1	0	0	0	4	0	0	0	4	905	3	1	3	0	3	5	3	3	3	0	20	11	4	0		
23	3	2	2	0	3	9	4	0	1	0	2	4	8	0	0	1	0	3	2	1	5	2	2320	0	1	0	0	1	2	0	3	0	15	6	1	0	
24	0	4	4	1	1	0	1	0	1	0	2	0	1	0	0	0	0	0	0	0	2	0	1	279	0	1	4	1	0	0	4	0	9	3	0	0	
25	13	9	18	1	8	11	7	1	5	5	5	4	12	4	0	6	20	2	0	0	0	7	1	0	3361	10	3	14	11	1	15	1	70	21	9	3	
26	0	0	1	0	1	0	4	0	2	0	1	0	242	0	0	49	0	0	0	0	1	1	0	0	2	88	0	1	0	0	1	2	0	4	0	1	
27	1	4	3	0	0	3	0	1	0	0	0	4	0	0	0	1	0	0	0	2	6	1	0	0	0	0	682	9	3	2	4	0	28	3	0	2	
28	4	37	93	6	19	19	18	3	15	3	13	18	18	1	0	21	74	0	4	0	25	8	3	2	5	1	36	4516	27	0	10	3	31	27	13	0	
29	7	13	17	2	17	20	8	4	4	0	1	0	4	3	0	5	5	0	1	0	19	12	6	0	6	2	8	3	7914	3	6	2	52	17	13	1	
30	6	13	8	1	0	18	2	6	6	0	3	0	9	0	0	1	4	0	0	0	12	2	6	0	1	8	4	6	7	2391	12	0	58	6	6	2	
31	24	16	43	0	14	52	11	12	3	0	14	1	18	3	5	9	14	0	2	1	36	7	4	0	3	1	25	18	20	14	8178	3	237	25	25	9	
32	2	4	0	0	0	8	2	0	0	0	0	0	8	0	0	0	78	0	0	0	2039	0	0	0	1	2	3	0	3	0	0	1332	9	1	20	1	
33	13	15	34	1	11	41	2	3	12	0	7	1	26	1	0	4	1	1	2	4	25	1	10	0	0	4	25	8	17	1	30	1	14977	22	22	9	
34	18	10	184	10	11	29	21	5	6	0	17	11	22	4	0	42	31	0	1	1	20	11	4	1	12	3	8	20	15	6	33	4	136	2613	16	4	
35	13	9	9	2	11	20	2	1	2	0	5	3	11	2	0	6	7	0	0	2	19	4	2	0	3	6	5	7	10	4	8	13	77	20	9561	3	
36	0	0	5	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3	0	0	0	0	1	0	0	0	0	0	12	0	2	986		

Ténylegesen aktív neuron

11. TÁBLAZAT: A 4. HÁLÓ ÉS A 2017_05_19__0_002 FÁJL MÉRÉSI EREDMÉNYÉHEZ TARTOZÓ KONFÚZIÓS MÁTRIX

4.3. Aktivítások detektálása időszeletek alapján

4.3.1. Adatok előfeldolgozása

Ebben a kísérletben 100 keretből álló időszeletekkel tanítottuk be a hálót:

$$f_s = 2 \cdot 10^4 \text{ Hz} \Rightarrow T_s = 5 \cdot 10^{-5} \text{ s}$$
$$T_{spike} = 5 \cdot 10^{-3} \text{ s}$$
$$N = \frac{T_{spike}}{T_s} = 100,$$

ahol

- f_s a mintavételi frekvencia,
- T_s a mintavételi periódusidő,
- T_{spike} a tüskék időbeni hossza,
- N az alkalmazandó adatkeretek száma egy időszeleten belül.

Kihasnálva azt, hogy a tüskék tényleges hossza valamivel kisebb a választott mintaszámnál (nagyjából 20 mintával) és hogy némi bizonytalanságot vigyünk a rendszerbe (megnehezítsük az aktivitások megtanulását azáltal, hogy maximumuk nem mindig ugyanott helyezkedik el), az aktivitásokat számított kezdetükhöz képest egy 0 és 20 közötti véletlen egész számmal eltoltuk az időszeleten belül.

Ezúttal is felhasználtuk a felvételben szereplő összes aktivitást, viszont egyrészt nem különböztettük meg őket neuronok szerint, másrészt negatív adatokat vettünk melléjük a pozitív minták között harmad- és kétharmadúton. Azért használtunk lényegesen több negatív adatot, mert az aktivitások a valóságban igen ritkán jelentkeznek: még az általunk használt mennyiség is felülreprezentálja a neuronok tüzelését a nyugalmi állapothoz képest.

Az adatok első felét tanításra, a másik felét tesztelésre használtuk. Ezúttal azonban nem töltöttük be az összes adatot a memóriába, hanem először 1000 időszeletből álló csomagokon betanítottuk a hálót, majd szintén 1000 időszeletből álló csomagokon teszteltük, az egyes csomagokat közvetlenül felhasználás előtt betöltve.

4.3.2. A használt háló kiválasztása

Az alkalmazott hálót az előző módszerhez hasonlóan választottuk ki. A kiindulásképpen használt háló paraméterei az alábbiak voltak:

Rétegszám	1
Szűrőszám	1
Kernelméret	(4, 4, 4)
Stride	(4, 4, 4)
Pooling	(2, 2, 2)

Az utolsó réteget kivéve ez esetben is lineáris függvényt használtunk aktivációs függvény gyanánt, a kimeneti réteg aktivációjaként is meghagytuk a softmaxot.

A kísérletek során végig ezt a hálót használtuk, mivel relatíve egyszerű, a számítások nem túlzottan időigényesek és alkalmazásával relatíve jó eredményeket kaptunk.

A tanítást és tesztelést végző programkód a Melléklet A 2.1. pontjában található.

Betöltöttünk egy 250 pozitív és 500 negatív időszeltekből álló csomagot, ennek felét tanító, másik felét tesztadatnak használtuk. A tanító adatokat normálás után 30-szor tanítottuk meg a hálónak, a lehetséges költségfüggvényeket és optimalizálókat felhasználva. Az egyes kombinációk találati arányai együtt normált adatok esetén a 12. táblázatban, külön-külön normált adatokra a 13. táblázatban láthatók. E két táblázat színekódjai a következők:

	Nem tanult
	$x < 0.7$
	$0.7 \leq x < 0.75$
	$0.75 \leq x$

Látható, hogy az adatok normálásának módja nagy hatást gyakorol az egyes költségfüggvény-optimalizáló kombinációk hatékonyságára: például a hinge és categorical_hinge költségeket használó hálóok együtt normált adatok esetén többségében nem is tanultak, míg külön-külön normált időszeltek esetén a 75 % feletti eredmények több mint felét eredményezték.

Az egyes találati arányokhoz tartozó konfúziós mátrixok a Melléklet B 2.1. szakaszában található. Példaként álljon itt a legjobb eredményhez (0.7694370150566101) tartozó mátrix (külön-külön normált adatok, Adadelta/categorical_hinge), amely a 14. táblázatban látható. A Mellékletben szereplő, fejléc nélküli mátrixok értelmezése megegyezik az alábbiéval.

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.6568364500999451	0.667560338973999	0.6595174074172974	0.7104557752609253	0.6729222536087036	0.7694370150566101	0.7050938606262207
mean_absolute_error	0.6809651255607605	0.667560338973999	0.6621983647346497	0.6729222536087036	0.667560338973999	0.667560338973999	0.667560338973999
mean_absolute_percentage_error	0.667560338973999	0.667560338973999	0.6729222536087036	0.667560338973999	0.667560338973999	0.667560338973999	0.667560338973999
mean_squared_logarithmic_error	0.6621983647346497	0.667560338973999	0.7292225360870361	0.7319034934043884	0.7345844507217407	0.6836460828781128	0.7453083395957947
squared_hinge	0.7453083395957947	0.667560338973999	0.7319034934043884	0.7399463653564453	0.7158176898956299	0.7426273226737976	0.6890080571174622
hinge	0.6729222536087036	0.667560338973999	0.667560338973999	0.667560338973999	0.667560338973999	0.6756032109260559	0.667560338973999
categorical_hinge	0.6648793816566467	0.667560338973999	0.667560338973999	0.667560338973999	0.667560338973999	0.6756032109260559	0.667560338973999
logcosh	0.7024128437042236	0.667560338973999	0.6836460828781128	0.7587131261825562	0.7131367325782776	0.7667560577392578	0.6916890144348145
categorical_crossentropy	0.7050938606262207	0.7104557752609253	0.6863270998001099	0.7104557752609253	0.697050929069519	0.6836460828781128	0.6890080571174622
binary_crossentropy	0.7453083395957947	0.6916890144348145	0.7640750408172607	0.7292225360870361	0.7184986472129822	0.6997318863868713	0.7613940834999084
kullback_leibler_divergence	0.6997318863868713	0.667560338973999	0.6756032109260559	0.7158176898956299	0.747989296913147	0.7238605618476868	0.6916890144348145
poisson	0.7613940834999084	0.6729222536087036	0.6541554927825928	0.7184986472129822	0.6782841682434082	0.707774817943573	0.7050938606262207
cosine_proximity	0.7399463653564453	0.6595174074172974	0.667560338973999	0.7694370150566101	0.707774817943573	0.6782841682434082	0.7104557752609253

12. TÁBLÁZAT: DETEKTÁLÁSI ARÁNY AZ EGYES OPTIMALIZÁLÓK ÉS KÖLTSÉGFÜGGVÉNYEK ESETÉN (KÖZÖSEN NORMÁLT ADATOK FELHASZNÁLÁSÁVAL)

	RMSprop	SGD	Adagrad	Adadelta	Adam	Adamax	Nadam
mean_squared_error	0.7506702542304993	0.6809651255607605	0.6997318863868713	0.707774817943573	0.6836460828781128	0.6997318863868713	0.7024128437042236
mean_absolute_error	0.6997318863868713	0.7131367325782776	0.7587131261825562	0.7694370150566101	0.7104557752609253	0.6756032109260559	0.667560338973999
mean_absolute_percentage_error	0.6595174074172974	0.6595174074172974	0.7265415787696838	0.7131367325782776	0.6863270998001099	0.7613940834999084	0.7453083395957947
mean_squared_logarithmic_error	0.6997318863868713	0.747989296913147	0.7319034934043884	0.7050938606262207	0.7265415787696838	0.6890080571174622	0.7050938606262207
squared_hinge	0.6809651255607605	0.697050929069519	0.697050929069519	0.7050938606262207	0.7131367325782776	0.7024128437042236	0.7050938606262207
hinge	0.7667560577392578	0.7667560577392578	0.7694370150566101	0.7667560577392578	0.7694370150566101	0.6756032109260559	0.7694370150566101
categorical_hinge	0.7667560577392578	0.7265415787696838	0.7453083395957947	0.7694370150566101	0.7667560577392578	0.7560321688652039	0.7694370150566101
logcosh	0.6916890144348145	0.7131367325782776	0.707774817943573	0.7721179723739624	0.6997318863868713	0.6997318863868713	0.7050938606262207
categorical_crossentropy	0.7050938606262207	0.7050938606262207	0.6756032109260559	0.7024128437042236	0.6997318863868713	0.7319034934043884	0.7104557752609253
binary_crossentropy	0.7024128437042236	0.7050938606262207	0.7292225360870361	0.7024128437042236	0.7104557752609253	0.7104557752609253	0.7158176898956299
kullback_leibler_divergence	0.6943699717521667	0.707774817943573	0.6729222536087036	0.7024128437042236	0.7587131261825562	0.7050938606262207	0.7613940834999084
poisson	0.6916890144348145	0.7050938606262207	0.697050929069519	0.707774817943573	0.697050929069519	0.7104557752609253	0.7158176898956299
cosine_proximity	0.7050938606262207	0.7587131261825562	0.7533512115478516	0.6863270998001099	0.7211796045303345	0.6890080571174622	0.7050938606262207

13. TÁBLÁZAT: DETEKTÁLÁSI ARÁNY AZ EGYES OPTIMALIZÁLÓK ÉS KÖLTSÉGFÜGGVÉNYEK ESETÉN (EGYENKÉNT NORMÁLT ADATOK FELHASZNÁLÁSÁVAL)

		Jelzett aktivitás	
		Igen	Nem
Tényleges aktivitás	Nem	124	0
	Igen	86	163

14. TÁBLÁZAT: KONFÚZIÓS MÁTRIX KÜLÖN NORMÁLT ADATOK ÉS ADADELTA/CATEGORICAL_HINGE KOMBINÁCIÓ ESETÉN

4.3.3. Teszteredmények

A tanítás során legjobb eredményt szolgáltató kombinációval rendelkező hálónak megtanítottuk a 2016_10_12__0_002 felvétel aktivitásait (az összes aktivitást felhasználtuk, felét tanító-, felét tesztadat gyanánt). Összesen 60-60 adatcsomagot kaptunk. Minden adatcsomaggal 3 tanítási ciklust eszközöltünk.

A tesztet megvalósító algoritmus programkódja a Melléklet A 2.2. szakaszában, a teszt eredményeit jelentő konfúziós mátrixok a Melléklet B 2.2. pontjában található meg.

A teszt során meglepetést szerzett, hogy a háló az aktivitásokat pontosan jelezte, vagyis hamis negatív eredményt nem kaptunk, az összes hiba a hamis pozitívak köréből került ki.

A konfúziós mátrixok ez esetben kis méretűek voltak; a legjobb, legrosszabb és „átlagos” eredményt itt közöljük.

A legjobb eredmény (találati arány: 0,803333) konfúziós mátrixa a 15. táblázatban látható. A negatív adatok 58,156 %-át találta el helyesen a detektor.

		Jelzett aktivitás	
		Igen	Nem
Tényleges aktivitás	Nem	1000	0
	Igen	590	1410

15. TÁBLÁZAT: LEGJOBB DETEKTÁLÁSI TESZTEREDMÉNY KONFÚZIÓS MÁTRIXA

A legrosszabb eredmény (találati arány: 0,758667) konfúziós mátrixa a 16. táblázatban látható. A helyesen jelzett negatív adatok aránya itt 43,2602 %.

		Jelzett aktivitás	
		Igen	Nem
Tényleges aktivitás	Nem	1000	0
	Igen	724	1276

16. TÁBLÁZAT: LEGROSSZABB DETEKTÁLÁSI TESZTEREDMÉNY KONFÚZIÓS MÁTRIXA

Az átlagos (a 60 adatcsomag átlagaként számolt) eredmény (találati arány: 0,780933) konfúziós mátrixa a 17. táblázatban látható. A helyesen jelzett negatív adatok aránya 51,0575 %, vagyis jó közelítéssel a háló a negatív adatok felét detektálja helyesen, míg a pozitív adatokat 100 %-os hatékonysággal képes jelezni.

		Jelzett aktivitás	
		Igen	Nem
Tényleges aktivitás	Nem	1000	0
	Igen	657,2	1342,8

17. TÁBLÁZAT: ÁTLAGOS DETEKTÁLÁSI TESZTEREDMÉNY KONFÚZIÓS MÁTRIXA

5. ÖSSZEGZÉS

Jelen dolgozatban konvolúciós neurális hálókat alkalmaztunk idegsejtek aktivitásának detektálására illetve osztályozására.

A detektálás határfoka az aktivitások tekintetében 100, míg a hamis pozitívak aránya közelítőleg 50 %; ezen az arányon a továbbiakban érdemes lenne megkísérelni a javítást, viszont már ez a detektor is felhasználható egy olyan osztályozó algoritmus bemeneti fokozataként, ami a negatív adatokat képes elszeparálni a tényleges aktivitásoktól.

Bár a rendelkezésre álló idő erre nem volt elegendő, később érdemes lenne a detektálást az összes adatfájltra lefuttatni (különösen azokra, amikre az osztályozó algoritmus is érdekes – különösen jó, vagy éppen átlag alatti – kimenetet produkált). Különösen hasznos volna az algoritmus tesztelése olyan időszakokon, amik az egyes aktivitásoknak csak egy részét tartalmazzák, ezzel fontos lépést téve az esetleges valós idejű alkalmazások felé.

Az általunk használt (minden egyes adatfájl esetében működőképes) osztályozó algoritmusok az egyes aktivitások megkülönböztetésére 71,6–88,78 %-os hatékonysággal képesek a felvétel bizonyos (egyelőre nem tisztázott) jellemzőitől függően.

A későbbiekben hasznos lenne a negatív adatok kategóriájának az algoritmushoz vétele, amivel mintegy kiegészítve a detektáló algoritmust (pontosan kompenzálva annak hibáját) az osztályozó annak kimeneti fokozata lehetne.

A kapott eredmények nem tekinthetők rossznak annak fényében, hogy mindkét háló csupán egy konvolúciós réteggel rendelkezik, így relatíve egyszerű hálók, amiknek a tanítása sem vesz igénybe túlzottan sok időt.

6. HIVATKOZÁSOK

- [1] Richárd Fiáth, Bogdan Cristian Raducanu, Silke Musa, Alexandru Andrei, Carolina Mora Lopez, Chris van Hoof, Patrick Ruther, Arno Aarts, Domonkos Horváth, and István Ulbert. 2018. „A Silicon-Based Neural Probe with Densely-Packed Low-Impedance Titanium Nitride Microelectrodes for Ultrahigh-Resolution in Vivo Recordings.” *Biosensors & Bioelectronics* 106 (May): 86–92. <https://doi.org/10.1016/j.bios.2018.01.060>
- [2] Lantos Béla. 2002. „Fuzzy Systems and Genetic Algorithms.” Műegyetemi Kiadó
- [3] Ian Goodfellow, Yoshua Bengio, Aaron Courville. 2016. „Deep Learning.” MIT Press
- [4] Szemenyei Márton. 2017. „Tanuló látás” (a Számítógépes látórendszerek című tárgy oktatási anyaga)
- [5] Losses – Keras Documentation; forráskód: <https://github.com/keras-team/keras/blob/master/keras/losses.py>
- [6] Matthew D. Zeiler. 2012. „ADADELTA: An Adaptive Learning Rate Method.” <https://arxiv.org/abs/1212.5701>
- [7] Duchi, J., Hazan, E., & Singer, Y. 2011. „Adaptive Subgradient Methods for Online Learning and Stochastic Optimization.” *Journal of Machine Learning Research*, 12, 2121–2159. <https://dl.acm.org/citation.cfm?id=2021068>
- [8] Diederik P. Kingma, Jimmy Ba. 2014. „Adam: A Method for Stochastic Optimization.” *International Conference on Learning Representations*, 1–13. <https://arxiv.org/abs/1412.6980v5>
- [9] Thomas Hossler. 2011. „Where is Waldo? A Deep Learning Approach to Template Matching.” *CVPR '11 Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, 1793–1800. <https://dl.acm.org/citation.cfm?id=2191919>
- [10] Yann LeCun, Yoshua Bengio. 1998. „Convolutional Networks for Images, Speech, and Time Series.” *The handbook of brain theory and neural networks*, MIT Press Cambridge, MA, USA, 1998, 255–258. <https://dl.acm.org/citation.cfm?id=303704>
- [11] Pedro Corrêa Pereira Vasco de Lacerda. 2016. „A Deep Learning Assessment of Spike Detection with Multi-electrode Arrays”. <https://fenix.tecnico.ulisboa.pt/downloadFile/1689244997255503/ExtendedAbstract-PedroLacerda.pdf>
- [12] Khalajzadeh H., Mansouri M., Teshnehlab M. 2014. „Face Recognition Using Convolutional Neural Network and Simple Logistic Classifier.” In: Snášel V., Krömer P., Köppen M., Schaefer G. (eds) *Soft Computing in Industrial Applications. Advances in Intelligent Systems and Computing*, vol 223. Springer, Cham. https://doi.org/10.1007/978-3-319-00930-8_18
- [13] Alexander Rosenberg Johansen, Jing Jin, Tomasz Maszczyk, Justin Dauwels, Sydney S. Cas, M. Brandon Westover. 2016. „Epileptiform Spike Detection via Convolutional Neural Networks.” *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* <https://doi.org/10.1109/ICASSP.2016.7471776>

- [14] Robin Tibor Schirrmeister, Jost Tobias Springenberg, Lukas Dominique Josef Fiederer, Martin Glasstetter, Katharina Eggenberger, Michael Tangermann, Frank Hutter, Wolfram Burgard, Tonio Ball. 2017. „Deep Learning With Convolutional Neural Networks for EEG Decoding and Visualization.” *Human Brain Mapping*, Nov; 38 (11): 5391–5420. <https://doi.org/10.1002/hbm.23730>
- [15] Ben Athiwaratkun, Keegan Kang. 2015. „Feature Representation In Convolutional Neural Networks.” <https://arxiv.org/abs/1507.02313>
- [16] Chaudhary, Ujwal, Niels Birbaumer, and Ander Ramos-Murguialday. “Brain-Computer Interfaces for Communication and Rehabilitation.” *Nature Reviews. Neurology* 12 (9): 513–25. <https://doi.org/10.1038/nrneurol.2016.113>.
- [17] Chaudhary, Ujwal, Bin Xia, Stefano Silvoni, Leonardo G Cohen, and Niels Birbaumer. 2017. “Brain–Computer Interface–Based Communication in the Completely Locked-In State.” *PLOS Biology* 15 (1): e1002593. <https://doi.org/10.1371/journal.pbio.1002593>.
- [18] Hochberg, Leigh R, Daniel Bacher, Beata Jarosiewicz, Nicolas Y Masse, John D Simeral, Joern Vogel, Sami Haddadin, et al. 2012. “Reach and Grasp by People with Tetraplegia Using a Neurally Controlled Robotic Arm.” *Nature* 485 (7398): 372–75. <https://doi.org/10.1038/nature11076>.
- [19] Jun, James J, Nicholas A Steinmetz, Joshua H Siegle, Daniel J Denman, Marius Bauza, Brian Barbarits, Albert K Lee, et al. 2017. “Fully Integrated Silicon Probes for High-Density Recording of Neural Activity.” *Nature* 551 (November): 232. <http://dx.doi.org/10.1038/nature24636>.
- [20] Maynard, E M, C T Nordhausen, and R A Normann. 1997. “The Utah Intracortical Electrode Array: A Recording Structure for Potential Brain-Computer Interfaces.” *Electroencephalography and Clinical Neurophysiology* 102 (3): 228–39.
- [21] Nicolas-Alonso, Luis Fernando, and Jaime Gomez-Gil. 2012. “Brain Computer Interfaces, a Review.” *Sensors (Basel, Switzerland)* 12 (2): 1211–79. <https://doi.org/10.3390/s120201211>.
- [22] Raducanu, Bogdan C, Refet F Yazicioglu, Carolina M Lopez, Marco Ballini, Jan Putzeys, Shiwei Wang, Alexandru Andrei, et al. 2017. “Time Multiplexed Active Neural Probe with 1356 Parallel Recording Sites.” *Sensors (Basel, Switzerland)* 17 (10). <https://doi.org/10.3390/s17102388>.
- [23] Schwartz, Andrew B, X Tracy Cui, Douglas J. Weber, and Daniel W Moran. 2006. “Brain-Controlled Interfaces: Movement Restoration with Neural Prosthetics.” *Neuron* 52 (1): 205–20. <https://doi.org/https://doi.org/10.1016/j.neuron.2006.09.019>.
- [24] Tam, W, R So, C Guan, and Z Yang. 2015. “EC-PC Spike Detection for High Performance Brain-Computer Interface.” In 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 5142–45. <https://doi.org/10.1109/EMBC.2015.7319549>.

MELLÉKLET A

Használt kódok

1. AKTIVITÁSOK OSZTÁLYOZÁSÁHOZ HASZNÁLT KÓDOK

1.1. Költségek és optimalizálók teljesítményének kiértékelése ismert háló esetén

```
import sys
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Flatten
from keras.layers import Conv2D, MaxPooling2D
from sklearn.metrics import confusion_matrix
import time

# 0. DEKLARÁCIÓK, DEFINÍCIÓK #####
# Program paraméterei
fn = "ARat_2016_10_12_0_002.dat" # Ebben a fájlban keresünk mintákat
ofn = "Eredmény.txt" # Ebbe a fájlba írjuk a futtatás eredményét
n_cluster = 23 # Ennyi klaszter adatait használjuk fel
n_category = n_cluster + 1 # Tanítási kategóriák száma (klaszterek száma + 1 kategória a negatív adatoknak)
training_rate = 0.5 # Tanításhoz az adatok ekkora hányadát használjuk fel

# Fix konstansok, dimenziók
n_row = 32 # Ennyi sorból áll egy minta
n_column = 4 # Ennyi oszlopból áll egy minta
n_channel = n_row*n_column # Ennyi csatorna van egy mintában
n_frame = 2*n_channel # Egy minta ennyi bájtot tartalmaz (kétbájtos előjeles egészek)
n_labeled_data = n_channel + 1 # Ennyi oszlopa van a mintákat és a címkéket tartalmazó mátrixoknak

# Felhasznált adatok, paraméterek, dimenziók
N_spike = np.zeros((0, 1), dtype=np.int64) # Egy klaszterhez ennyi pozitív minta tartozik
N_train = np.zeros((0, 1), dtype=np.int64) # Tanító pozitív adatok száma klaszterenként
N_test = np.zeros((0, 1), dtype=np.int64) # Tesztadatok száma klaszterenként (pozitív minták)

n_true = 0 # Pozitív (tűskéket tartalmazó) minták száma
n_train = 0 # Tanító adatok száma
n_test = 0 # Tesztadatok száma

index_true = [] # Pozitív minták indexei

data_true = np.zeros((0, n_labeled_data), dtype=np.int64) # Pozitív minták és a hozzájuk tartozó címkék tömbje
data_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok tömbje
data_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok tömbje
label_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok címkéinek tömbje
label_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok címkéinek tömbje

# 1. ADATOK BETÖLTÉSE, ELŐFELDOLGOZÁS #####
start0 = time.time() # Időmérés
np.set_printoptions(threshold=sys.maxsize, linewidth=sys.maxsize) # A kimenet ne tördelje a sorokat feleslegesen

# 1.1. Adatok betöltése #####
# 1.1.1. Pozitív adatok előállításá #####

# Végigmegyünk az összes neuronon
for i_cluster in range(n_cluster):
    print(str(i_cluster) + ". neuron adatainak beolvasása")
    start = time.time()

    # Indexek beolvasása ev2 fájlból
    evfn = "ARat" + str(i_cluster) + ".ev2"
    index_true_temp = np.loadtxt(evfn, dtype='int', skiprows=0, usecols=None)

    # Pozitív minták száma
    n_spike = index_true_temp.shape[0] # Az adott klaszterre
    N_spike = np.vstack([N_spike, n_spike]) # A többihez vesszük

    # Pozitív minták indexei
    index_true_temp = index_true_temp[0:n_spike, 5] # Utolsó oszlop tartalmazza a tűske maximumához tartozó minta indexét
    index_true_temp = np.array(index_true_temp, dtype=np.int64)
    index_true_temp = index_true_temp*n_frame # A dat fájlban ekkora ugrásonként következik új minta
    index_true = np.hstack([index_true, index_true_temp]) # A többihez vesszük

    # Minták beolvasása dat fájlból
    data_true_temp = np.zeros([n_spike, n_labeled_data])
    with open(fn, "rb") as f:
        f.seek(0) # Fájl elejére ugrás
        for j in range(n_spike):
            f.seek(index_true_temp[j]) # Minta elejére ugrás
            for i in range(n_channel):
                data_true_temp[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True) # Minta betöltése
    data_true_temp[0:n_spike, n_channel] = i_cluster # Klaszterhez tartozó minták felcímkézése
    data_true = np.vstack([data_true, data_true_temp]) # A többihez vesszük

    # Beolvasás időtartamának kiírása
    end = time.time()
    elapsed = end - start
    print(elapsed)

n_true = index_true.shape[0]
index_true = np.array(index_true, dtype=np.int64)
index_true = np.sort(index_true)
```

```

# 1.2. Előfeldolgozás
print("Előfeldolgozás")
start = time.time()

# Adatsorok normálása
minimum = data_true[0:n_true, 0:n_channel].min()
maximum = data_true[0:n_true, 0:n_channel].max()
eltolas = (maximum + minimum)/2
skalafaktor = (np.abs(maximum) + np.abs(minimum))/2
data_true[0:n_true, 0:n_channel] = (data_true[0:n_true, 0:n_channel] - eltolas)/skalafaktor

# Tanító- és tesztadatok szétválogatása az egyes klaszterekre
N_train = np.ndarray.round(training_rate*N_spike)
N_test = N_spike - N_train
N_spike_cum_end = np.cumsum(N_spike)
N_spike_cum_start = (N_spike_cum_end - N_spike.transpose()).flatten()

for i_cluster in range(n_cluster):
    data_train_temp = data_true[N_spike_cum_start[i_cluster]:int(N_spike_cum_start[i_cluster] + N_train[i_cluster]), 0:n_labeled_data]
    data_test_temp = data_true[int(N_spike_cum_end[i_cluster] - N_test[i_cluster]):N_spike_cum_end[i_cluster], 0:n_labeled_data]
    data_train = np.vstack([data_train, data_train_temp])
    data_test = np.vstack([data_test, data_test_temp])

# Cimkék és adatok szétvágása külön mátrixokra
n_train = data_train.shape[0]
n_test_all = data_test.shape[0]

label_train = data_train[0:n_train, n_channel]
label_test = data_test[0:n_test_all, n_channel]
data_train = data_train[0:n_train, 0:n_channel]
data_test = data_test[0:n_test_all, 0:n_channel]

label_test0 = label_test # Cimkék félretétele a konfúziós mátrixhoz

# Átalakítás a háló számára
data_train = data_train.reshape(n_train, n_row, n_column, 1)
data_test = data_test.reshape(n_test_all, n_row, n_column, 1)
label_train = keras.utils.to_categorical(label_train, num_classes=n_category)
label_test = keras.utils.to_categorical(label_test, num_classes=n_category)

end1 = time.time() # Adatok betöltésével/előfeldolgozásával eltelt idő

# 2. KONVOLÚCIÓS HÁLÓ #####
print("Háló felépítése")
start1 = time.time()

# 2.1. Háló felépítése
n_reteg = 0 # A bemenő konvolúciós rétegen felüli konvolúciós rétegek száma
depth = 32 # Konvolúciós rétegek szűrőszáma

stride = (2, 2)
kernel = (4, 4)
pooling = (2, 2)

# Lehetséges optimalizálók listája
n_optimizer = 7
switcher_optimizer = {
    0: "RMSprop",
    1: "SGD",
    2: "Adagrad",
    3: "Adadelta",
    4: "Adam",
    5: "Adamax",
    6: "Nadam"
}

# Lehetséges költségfüggvények listája
n_loss = 12
switcher_loss = {
    0: "mean_squared_error",
    1: "mean_absolute_error",
    2: "mean_absolute_percentage_error",
    3: "mean_squared_logarithmic_error",
    4: "squared_hinge",
    5: "hinge",
    6: "categorical_hinge",
    7: "logcosh",
    8: "categorical_crossentropy",
    9: "kullback_leibler_divergence",
    10: "poisson",
    11: "cosine_proximity"
}

# Végigmenve az optimalizálókon és a költségeken adott felvétel neuronjainak aktivitására betanítjuk a hálót
for i_optimizer in range(n_optimizer):
    optimizer = switcher_optimizer.get(i_optimizer) # Adott ciklushoz tartozó optimalizáló

    with open(ofn, 'a') as file:
        file.write(optimizer)
        file.write('\n')

    for i_loss in range(n_loss):
        loss = switcher_loss.get(i_loss) # Adott ciklushoz tartozó költségfüggvény

        halo = Sequential()

        halo.add(Conv2D(depth, strides=stride, kernel_size=kernel, activation='linear', input_shape=(n_row, n_column, 1), padding='same'))
        halo.add(MaxPooling2D(pooling, padding='same'))

        k_reteg = 0
        i_reteg = 0
        for i_reteg in range(n_reteg):
            halo.add(Conv2D((i_reteg + 2)*depth, kernel, activation='linear', padding='same'))

```



```

        halo.add(MaxPooling2D(pool_size=pooling, padding='same'))
        k_reteg = i_reteg

halo.add(Flatten())
halo.add(Dense((i_reteg + 2)*depth, activation='linear'))
halo.add(Dense(n_category, activation='softmax'))

# 2.2. Fordítás, tanítás, eredmények
halo.compile(loss=loss, optimizer=optimizer, metrics=['accuracy'])
halo.fit(data_train, label_train, batch_size=round(n_train/1000), epochs=30)
print(optimizer)
print(loss)

# Konfúziós mátrix
label_predicted = halo.predict(data_test)
label_predicted0 = np.zeros(n_test_all)

for i in range(n_test_all):
    label_predicted0[i] = np.argmax(label_predicted[i, 0:n_category])
cm = confusion_matrix(label_test0, label_predicted0)
print(cm)

# Tesztadatok kiértékelése
score = halo.evaluate(data_test, label_test, batch_size=n_test_all)
print(score)

# Eredmények fájlba írása
with open(ofn, 'a') as file:
    file.write(loss)
    file.write('\n')
    file.write(str(cm))
    file.write('\n')
    file.write(str(score))
    file.write('\n')
    file.write('\n')

# Időmérés vége
end0 = time.time()
elapsed0 = end0 - start0
elapsed1 = end1 - start0
elapsed2 = end0 - start1

# Fájlba írás
print("Teljes eltelt idő: " + str(elapsed0))
print("Ebből adatok előállítás: " + str(elapsed1))
print("Háló tanítása: " + str(elapsed2))

```

```

with open(ofn, 'a') as file:
    file.write("Teljes eltelt idő: " + str(elapsed0))
    file.write('\n')
    file.write("Ebből adatok előállítás: " + str(elapsed1))
    file.write('\n')
    file.write("Háló tanítása: " + str(elapsed2))
    file.write('\n')
    file.write('\n')

```

1.2. Adott kernelméret/optimalizáló/költség kombinációk kiértékelése

```

import sys
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Flatten
from keras.layers import Conv2D, MaxPooling2D
from sklearn.metrics import confusion_matrix
import time

# 0. DEKLARÁCIÓK, DEFINÍCIÓK #####
# Program paraméterei
fn = "ARat_2016_07_27_0_002.dat" # Ebben a fájlban keresünk mintákat
ofn = "Eredmény.txt"
n_cluster = 37 # Ennyi klaszter adatait használjuk fel
n_category = n_cluster # Tanítási kategóriák száma (Klaszterek száma + 1 kategória a negatív adatoknak)
training_rate = 0.5 # Tanításhoz az adatok ekkora hányadát használjuk fel

# Fix konstansok, dimenziók
n_row = 32 # Ennyi sorból áll egy minta
n_column = 4 # Ennyi oszlopból áll egy minta
n_channel = n_row*n_column # Ennyi csatorna van egy mintában
n_frame = 2*n_channel # Egy minta ennyi bájtot tartalmaz (kétbájtos előjeles egészek)
n_labeled_data = n_channel + 1 # Ennyi oszlopa van a mintákat és a címkéket tartalmazó mátrixoknak

# Felhasznált adatok, paraméterek, dimenziók
N_spike = np.zeros((0, 1), dtype=np.int64) # Egy klaszterhez ennyi pozitív minta tartozik
N_train = np.zeros((0, 1), dtype=np.int64) # Tanító pozitív adatok száma klaszterenként
N_test = np.zeros((0, 1), dtype=np.int64) # Tesztadatok száma klaszterenként (pozitív minták)

n_true = 0 # Pozitív (tüskéket tartalmazó) minták száma
n_train = 0 # Tanító adatok száma
n_test = 0 # Tesztadatok száma

index_true = [] # Pozitív minták indexei

data_true = np.zeros((0, n_labeled_data), dtype=np.int64) # Pozitív minták és a hozzájuk tartozó címkék tömbje
data_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok tömbje
data_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok tömbje
label_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok címkéinek tömbje
label_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok címkéinek tömbje

```

```

# 1. ADATOK BETÖLTÉSE, ELŐFELDOLGOZÁS #####
start0 = time.time() # Időmérés
np.set_printoptions(threshold=sys.maxsize, linewidth=sys.maxsize) # A kimenet ne tördelje a sorokat feleslegesen

# 1.1. Adatok betöltése #####
# 1.1.1. Pozitív adatok előállításá /#####
# Végigmegyünk az összes neuronon
for i_cluster in range(n_cluster):
    print(str(i_cluster) + ". neuron adatainak beolvasása")
    start = time.time()

    # Indexek beolvasása ev2 fájlból
    evfn = "Arat" + str(i_cluster) + ".ev2"
    index_true_temp = np.loadtxt(evfn, dtype='int', skiprows=0, usecols=None)

    # Pozitív minták száma
    n_spike = index_true_temp.shape[0] # Az adott klaszterre
    n_spike = np.vstack([N_spike, n_spike]) # A többihez vesszük

    # Pozitív minták indexei
    index_true_temp = index_true_temp[0:n_spike, 5] # Utolsó oszlop tartalmazza a túske maximumához tartozó minta indexét
    index_true_temp = np.array(index_true_temp, dtype=np.int64)
    index_true_temp = index_true_temp*n_frame # A dat fájlban ekkora ugrásonként következnek új minta
    index_true = np.hstack([index_true, index_true_temp]) # A többihez vesszük

    # Minták beolvasása dat fájlból
    data_true_temp = np.zeros([n_spike, n_labeled_data])
    with open(fn, "rb") as f:
        f.seek(0) # Fájl elejére ugrás
        for j in range(n_spike):
            f.seek(index_true_temp[j]) # Minta elejére ugrás
            for i in range(n_channel):
                data_true_temp[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True) # Minta betöltése
    data_true_temp[0:n_spike, n_channel] = i_cluster # Klaszterhez tartozó minták felcímkézése
    data_true = np.vstack([data_true, data_true_temp]) # A többihez vesszük

    # Beolvasás időtartamának kiírása
    end = time.time()
    elapsed = end - start
    print(elapsed)

n_true = index_true.shape[0]
index_true = np.array(index_true, dtype=np.int64)
index_true = np.sort(index_true)

# 1.2. Előfeldolgozás #####
print("Előfeldolgozás")
start = time.time()

# Adatsorok normálása
minimum = data_true[0:n_true, 0:n_channel].min()
maximum = data_true[0:n_true, 0:n_channel].max()
eltolas = (maximum + minimum)/2
skalafaktor = (np.abs(maximum) + np.abs(minimum))/2
data_true[0:n_true, 0:n_channel] = (data_true[0:n_true, 0:n_channel] - eltolas)/skalafaktor

# Tanító- és tesztadatok szétválogatása az egyes klaszterekre
N_train = np.ndarray.round(training_rate*N_spike)
N_test = N_spike - N_train
N_spike_cum_end = np.cumsum(N_spike)
N_spike_cum_start = (N_spike_cum_end - N_spike.transpose()).flatten()

for i_cluster in range(n_cluster):
    data_train_temp = data_true[N_spike_cum_start[i_cluster]:int(N_spike_cum_start[i_cluster] + N_train[i_cluster]), 0:n_labeled_data]
    data_test_temp = data_true[int(N_spike_cum_end[i_cluster] - N_test[i_cluster]):N_spike_cum_end[i_cluster], 0:n_labeled_data]
    data_train = np.vstack([data_train, data_train_temp])
    data_test = np.vstack([data_test, data_test_temp])

# Címkek és adatok szétvágása külön mátrixokra
n_train = data_train.shape[0]
n_test_all = data_test.shape[0]

label_train = data_train[0:n_train, n_channel]
label_test = data_test[0:n_test_all, n_channel]
data_train = data_train[0:n_train, 0:n_channel]
data_test = data_test[0:n_test_all, 0:n_channel]

label_test0 = label_test # Címkek félretétele a konfúziós mátrixhoz

# Átalakítás a háló számára
data_train = data_train.reshape(n_train, n_row, n_column, 1)
data_test = data_test.reshape(n_test_all, n_row, n_column, 1)
label_train = keras.utils.to_categorical(label_train, num_classes=n_category)
label_test = keras.utils.to_categorical(label_test, num_classes=n_category)

endl = time.time() # Adatok betöltésével/előfeldolgozásával eltelt idő

# 2. KONVOLÚCIÓS HÁLÓ #####
print("Háló felépítése")
start1 = time.time()

# 2.1. Háló felépítése #####
n_reteg = 0 # A bemenő konvolúciós rétegen felüli konvolúciós rétegek száma
depth = 32 # Konvolúciós rétegek szűrőszáma

stride = (1, 1)
pooling = (1, 1)

n_ciklus = 11 # Lehetséges kombinációk száma

```

```

# Adott kombinációkhoz tartozó kernelek
switcher_kernel = {
    0: (4, 2),
    1: (4, 2),
    2: (4, 2),
    3: (16, 2),
    4: (16, 2),
    5: (16, 2),
    6: (16, 2),
    7: (16, 2),
    8: (16, 2),
    9: (16, 2),
    10: (16, 2)
}

# Adott kombinációkhoz tartozó optimalizálók
switcher_optimizer = {
    0: "RMSprop",
    1: "RMSprop",
    2: "Adamax",
    3: "RMSprop",
    4: "RMSprop",
    5: "Adadelta",
    6: "Adam",
    7: "Adamax",
    8: "Nadam",
    9: "Nadam",
    10: "Nadam"
}

# Adott kombinációkhoz tartozó költségfüggvények
switcher_loss = {
    0: "mean_squared_error",
    1: "mean_squared_logarithmic_error",
    2: "cosine_proximity",
    3: "mean_squared_logarithmic_error",
    4: "categorical_crossentropy",
    5: "cosine_proximity",
    6: "categorical_hinge",
    7: "cosine_proximity",
    8: "mean_squared_logarithmic_error",
    9: "categorical_hinge",
    10: "logcosh"
}

for i_ciklus in range(n_ciklus):
    # Adott kombináció paramétereinek betöltése
    kernel = switcher_kernel.get(i_ciklus)
    optimizer = switcher_optimizer.get(i_ciklus)
    loss = switcher_loss.get(i_ciklus)

    # Paraméterek fájlba írása
    with open(ofn, 'a') as file:
        file.write(optimizer)
        file.write('\n')
        file.write("Stride: " + str(stride))
        file.write('\n')
        file.write("Kernel: " + str(kernel))
        file.write('\n')
        file.write("Pooling: " + str(pooling))
        file.write('\n')

    halo = Sequential()

    halo.add(Conv2D(depth, strides=stride, kernel_size=kernel, activation='linear', input_shape=(n_row, n_column, 1), padding='same'))
    halo.add(MaxPooling2D(pooling, padding='same'))

    k_reteg = 0
    i_reteg = 0
    for i_reteg in range(n_reteg):
        halo.add(Conv2D((i_reteg + 2)*depth, kernel, activation='linear', padding='same'))
        halo.add(MaxPooling2D(pool_size=pooling, padding='same'))
        k_reteg = i_reteg

    halo.add(Flatten())
    halo.add(Dense((i_reteg + 2)*depth, activation='linear'))
    halo.add(Dense(n_category, activation='softmax'))

    # 2.2. Fordítás, tanítás, eredmények ~~~~~
    halo.compile(loss=loss, optimizer=optimizer, metrics=['accuracy'])
    halo.fit(data_train, label_train, batch_size=round(n_train/1000), epochs=30)
    print(optimizer)
    print(loss)

    # Konfúziós mátrix
    label_predicted = halo.predict(data_test)
    label_predicted0 = np.zeros(n_test_all)

    for i in range(n_test_all):
        label_predicted0[i] = np.argmax(label_predicted[i], 0:n_category)
    cm = confusion_matrix(label_test0, label_predicted0)
    print(cm)

    # Tesztadatok kiértékelése
    score = halo.evaluate(data_test, label_test, batch_size=int(round(n_test_all/10)))
    print(score)

```

```

# Eredmények fájlba írása
with open(ofn, 'a') as file:
    file.write(loss)
    file.write('\n')
    file.write(str(cm))
    file.write('\n')
    file.write(str(score))
    file.write('\n')
    file.write('\n')

# Időmérés vége
end0 = time.time()
elapsed0 = end0 - start0
elapsed1 = end1 - start0
elapsed2 = end0 - start1

# Fájlba írás
print("Teljes eltelt idő: " + str(elapsed0))
print("Ebből adatok előállítás: " + str(elapsed1))
print("Háló tanítása: " + str(elapsed2))

with open(ofn, 'a') as file:
    file.write("Teljes eltelt idő: " + str(elapsed0))
    file.write('\n')
    file.write("Ebből adatok előállítás: " + str(elapsed1))
    file.write('\n')
    file.write("Háló tanítása: " + str(elapsed2))
    file.write('\n')
    file.write('\n')

```

2. AKTIVITÁSOK DETEKTÁLÁSÁHOZ HASZNÁLT KÓDOK

2.1. Költségek és optimalizálók teljesítményének kiértékelése ismert háló esetén

```

import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Flatten
from keras.layers import Conv3D, MaxPooling3D
from keras.models import load_model
from sklearn.metrics import confusion_matrix
import time

# DEKLARÁCIÓK, DEFINÍCIÓK #####
# Program paramétereit

start = time.time()

fn = "ARat_2016_10_12_0_002.dat" # Ebben a fájlban keresünk mintákat
ofn = "Eredmény.txt"
n_cluster = 24 # Ennyi klaszter adatait használjuk fel
n_category = 2 # Tanítási kategóriák száma
batch_size = 250 # Ennyi pozitív adatot olvasunk be egyszerre
training_rate = 0.5 # Tanításhoz az adatok ekkora hányadát használjuk fel

# Fix konstansok, dimenziók
n_row = 32 # Ennyi sorból áll egy keret
n_column = 4 # Ennyi oszlopból áll egy keret
n_frame = 100 # Ennyi keretet olvasunk be egy időszület gyanánt
n_channel = n_row*n_column # Ennyi minta van egy keretben
n_byte = 2*n_channel # Egy keret ennyi bajtot tartalmaz (kétbájtos előjeles egészek)
n_sample = n_frame*n_channel # Ennyi mintát olvasunk be egy időszület gyanánt
n_labeled_data = n_sample + 1 # Ennyi oszlopa van a mintákat és a címkéket tartalmazó mátrixoknak

# Felhasznált adatok, paraméterek, dimenziók
n_true = 0 # Pozitív (tüskéket tartalmazó) minták száma
n_false = 0 # Negatív (tüskéket nem tartalmazó) minták száma
n_all = 0 # Összes minta száma
n_train = 0 # Tanító adatok száma
n_test = 0 # Tesztadatok száma
n_batch = 0 # Ennyi adatcsomaggal dolgozunk

index_true = [] # Pozitív minták indexei
index_false = [] # Negatív minták indexei

data_true = np.zeros((0, n_labeled_data), dtype=np.int64) # Pozitív minták és a hozzájuk tartozó címkék tömbje
data_false = np.zeros((0, n_labeled_data), dtype=np.int64) # Negatív minták és a hozzájuk tartozó címkék tömbje
data_all = np.zeros((0, n_labeled_data), dtype=np.int64) # Összes mintát tartalmazó tömb
data_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok tömbje
data_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok tömbje
label_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok címkéinek tömbje
label_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok címkéinek tömbje

# ADATOK INDEXEINEK ELŐÁLLÍTÁSA #####

# Pozitív minták indexei
for i_cluster in range(n_cluster):
    # Indexek beolvasása ev2 fájlból
    evfn = "Arat" + str(i_cluster) + ".ev2"
    # evfn = "Arat_2016_10_12_0_002_48.ev2"
    index_true_temp = np.loadtxt(evfn, dtype='int', skiprows=0, usecols=None)
    n_spike = index_true_temp.shape[0] # Pozitív minták száma az adott klaszterre

    index_true_temp = index_true_temp[0:n_spike, 5] # Utolsó oszlop tartalmazza a tüske maximumához tartozó minta indexét
    index_true_temp = np.array(index_true_temp, dtype=np.int64)

    index_true = np.hstack([index_true, index_true_temp]) # A többihez vesszük

```

```

n_true = index_true.shape[0]

for i in range(n_true):
    index_true[i] = index_true[i] - 20 # Véletlenszerű kismértékű eltolás

index_true = index_true * n_byte # A dat fájlban ekkora ugrásonként következik új minta
index_true = np.array(index_true, dtype=np.int64)
index_true = np.sort(index_true)

# Negatív minták a pozitívak között harmad- és kétharmadúton, lefelé egész számra kerekítve
# Mivel az első keret előtt is keresünk mintákat, az indexelést nullától kezdjük, az utolsó indexet levágva
index_true_offset = np.zeros(n_true)
index_true_offset[1:n_true] = index_true[0:n_true-1]

# Az előbb számított indexekhez képesti eltolások
offset1 = np.round((index_true - index_true_offset)/3)
offset1 = offset1.astype(np.int64)
offset2 = 2*offset1

# Az eltolt indexek összefűzése, majd sorbarendezése
index_false = np.hstack([index_true_offset + offset1, index_true_offset + offset2])
index_false = index_false.astype(np.int64)
index_false = np.sort(index_false)
n_false = index_false.shape[0]

# ADATOK BETÖLTÉSE, ELŐFELDOLGOZÁS, HÁLÓ TANÍTÁSA #####
# Mindez ciklikusan, mintacsomagonként, hogy a memóriában elférjenek az adatok
n_batch = 1 # Adatcsomagok száma
print(n_batch)

n_true_batch = batch_size
n_false_batch = 2*batch_size

n_true_train = int(round(n_true_batch * training_rate))
n_false_train = int(round(n_false_batch * training_rate))

for k in range(n_batch):
    print(k)
    # Minták beolvasása dat fájlból #####
    data_true = np.zeros((n_true_batch, n_labeled_data), dtype=np.int64)
    data_false = np.zeros((n_false_batch, n_labeled_data), dtype=np.int64)
    data_train = np.zeros((0, n_labeled_data), dtype=np.int64)

    with open(fn, "rb") as f:
        f.seek(0) # Fájl elejére ugrás
        for j in range(n_true_batch):
            f.seek(index_true[j] + k*n_true_batch) # Minta elejére ugrás
            for i in range(n_sample):
                data_true[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True) # Minta betöltése
    data_true[0:n_true_batch, n_sample] = 0 # Klaszterhez tartozó minták felcímkézése

    with open(fn, "rb") as f:
        f.seek(0)
        for j in range(n_false_batch):
            f.seek(index_false[j] + k*n_false_batch)
            for i in range(n_sample):
                data_false[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True)
    data_false[0:n_false_batch, n_sample] = 1

    # Adatok közös mátrixba vétele
    data_all = np.vstack([data_true, data_false])
    n_all = n_true_batch + n_false_batch

    # Előfeldolgozás #####
    # Adatsorok normálása
    data_all = data_all.astype(dtype=float)

    # Közös norma az összes időszeletre
    # minimum = data_all[0:n_all, 0:n_sample].min()
    # maximum = data_all[0:n_all, 0:n_sample].max()
    # eltolas = (maximum + minimum) / 2
    # skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2
    # data_all[0:n_all, 0:n_sample] = data_all[0:n_all, 0:n_sample] / skalafaktor

    # Külön minden időszeletre
    minimum = data_all[0:n_all, 0:n_sample].min(axis=1)
    maximum = data_all[0:n_all, 0:n_sample].max(axis=1)
    minimum = minimum.reshape((n_all, 1))
    maximum = maximum.reshape((n_all, 1))
    eltolas = (maximum + minimum) / 2
    skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2

    data_all[0:n_all, 0:n_sample] = np.subtract(data_all[0:n_all, 0:n_sample], eltolas)
    data_all[0:n_all, 0:n_sample] = np.divide(data_all[0:n_all, 0:n_sample], skalafaktor)

    # Pozitív adatok szétosztása tanító és tesztadatokra
    data_train_temp = data_all[0:n_true_train, 0:n_labeled_data]
    data_test_temp = data_all[n_true_train+1:n_true_batch, 0:n_labeled_data]

    data_train = np.vstack([data_train, data_train_temp])
    data_test = np.vstack([data_test, data_test_temp])

    # Negatív adatok szétosztása tanító és tesztadatokra
    data_train_temp = data_all[n_true_batch+1:n_true_batch+n_false_train, 0:n_labeled_data]
    data_test_temp = data_all[n_true_batch+n_false_train+1:n_all, 0:n_labeled_data]

    data_train = np.vstack([data_train, data_train_temp])
    data_test = np.vstack([data_test, data_test_temp])

    # Címkek és adatok szétvágása külön mátrixokra
    n_train = data_train.shape[0]

```

```

label_train = data_train[0:n_train, n_sample]
data_train = data_train[0:n_train, 0:n_sample]

# Átalakítás a háló számára
data_train = data_train.reshape(n_train, n_frame, n_row, n_column, 1)
label_train = keras.utils.to_categorical(label_train, num_classes=n_category)

# Tesztadatok feldolgozása #####
# Tesztadatok szétvágása adatokra és címkékre
n_test_all = data_test.shape[0]

label_test = data_test[0:n_test_all, n_sample]
data_test = data_test[0:n_test_all, 0:n_sample]

label_test0 = label_test # Címkek félretétele a konfúziós mátrixhoz

# Átalakítás a háló számára
data_test = data_test.reshape(n_test_all, n_frame, n_row, n_column, 1)
label_test = keras.utils.to_categorical(label_test, num_classes=n_category)

# KONVOLÚCIÓS HÁLÓ #####
n_reteg = 0 # Háló bemeneti konvolúciós rétegen felüli konvolúciós rétegek száma
depth = 1 # Rétegenkénti szűrőszám

stride = (4, 4, 4)
kernel = (4, 4, 4)
pooling = (2, 2, 2)

# Lehetséges optimalizálók listája
n_optimizer = 7
switcher_optimizer = {
    0: "RMSprop",
    1: "SGD",
    2: "Adagrad",
    3: "Adadelta",
    4: "Adam",
    5: "Adamax",
    6: "Nadam"
}

# Lehetséges költségfüggvények listája
n_loss = 13
switcher_loss = {
    0: "mean_squared_error",
    1: "mean_absolute_error",
    2: "mean_absolute_percentage_error",
    3: "mean_squared_logarithmic_error",
    4: "squared_hinge",
    5: "hinge",
    6: "categorical_hinge",
    7: "logcosh",
    8: "categorical_crossentropy",
    9: "binary_crossentropy",
    10: "kullback_leibler_divergence",
    11: "poisson",
    12: "cosine_proximity"
}

# A költségek és optimalizálók minden kombinációján végigmegy
for i_optimizer in range(n_optimizer):
    for i_loss in range(n_loss):

        optimizer = switcher_optimizer.get(i_optimizer)
        loss = switcher_loss.get(i_loss)

        halo = Sequential()

        halo.add(Conv3D(depth, strides=stride, kernel_size=kernel, activation='linear',
            input_shape=(n_frame, n_row, n_column, 1), padding='same', data_format='channels_last'))
        halo.add(MaxPooling3D(pooling, padding='same'))

        k_reteg = 0
        i_reteg = 0
        for i_reteg in range(n_reteg):
            halo.add(Conv3D((i_reteg + 2) * depth, kernel, activation='linear', padding='same'))
            halo.add(MaxPooling3D(pool_size=pooling, padding='same'))
            k_reteg = i_reteg

        halo.add(Flatten())
        halo.add(Dense((i_reteg + 2) * depth, activation='linear'))
        halo.add(Dense(n_category, activation='softmax'))

        halo.compile(loss=loss, optimizer=optimizer, metrics=['accuracy'])
        # Háló tanítása
        halo.fit(data_train, label_train, batch_size=round(n_train/100), epochs=30)

        # TESZT A TANÍTÁSI FOLYAMAT UTÁN #####
        # Konfúziós mátrix #####
        label_predicted = halo.predict(data_test)
        label_predicted0 = np.zeros(n_test_all)

        for i in range(n_test_all):
            label_predicted0[i] = np.argmax(label_predicted[i, 0:n_category])
        cm = confusion_matrix(label_test0, label_predicted0)
        print(cm)

        # 4.3. Tesztadatok kiértékelése #####
        score = halo.evaluate(data_test, label_test, batch_size=n_test_all)
        print(score)

```

```

        with open(ofn, 'a') as file:
            file.write(optimizer)
            file.write('\n')
            file.write(loss)
            file.write('\n')
            file.write(str(cm))
            file.write('\n')
            file.write(str(score))
            file.write('\n')
            file.write('\n')

end = time.time()
elapsed = end - start
print("Eltelt idő: " + str(elapsed))
with open(ofn, 'a') as file:
    file.write(str(cm))
    file.write('\n')
    file.write(str(score))
    file.write('\n')
    file.write("Eltelt idő: " + str(elapsed))
    file.write('\n')

```

2.2. Aktivitás detektálása

```

import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Flatten
from keras.layers import Conv3D, MaxPooling3D
from keras.models import load_model
from sklearn.metrics import confusion_matrix
import time

# 0. DEKLARÁCIÓK, DEFINÍCIÓK #####
# Program paraméterei

start = time.time()

fn = "ARat_2016_10_12_0_002.dat" # Ebben a fájlban keresünk mintákat
ofn = "Eredmény.txt" # Ebbe a fájlba mentjük az eredményeket
n_cluster = 24 # Ennyi klaszter adatait használjuk fel
n_category = 2 # Tanítási kategóriák száma
batch_size = 1000 # Ennyi pozitív adatot olvasunk be egyszerre
training_rate = 0.5 # Tanításhoz az adatok ekkora hányadát használjuk fel

# Fix konstansok, dimenziók
n_row = 32 # Ennyi sorból áll egy keret
n_column = 4 # Ennyi oszlopból áll egy keret
n_frame = 100 # Ennyi keretet olvasunk be egy időszelét gyanánt
n_channel = n_row*n_column # Ennyi minta van egy keretben
n_byte = 2*n_channel # Egy keret ennyi bajtot tartalmaz (kétbájtos előjeles egészek)
n_sample = n_frame*n_channel # Ennyi mintát olvasunk be egy időszelét gyanánt
n_labeled_data = n_sample + 1 # Ennyi oszlopa van a mintákat és a címkéket tartalmazó mátrixoknak

# Felhasznált adatok, paraméterek, dimenziók
n_true = 0 # Pozitív (tüskéket tartalmazó) minták száma
n_false = 0 # Negatív (tüskéket nem tartalmazó) minták száma
n_all = 0 # Összes minta száma
n_train = 0 # Tanító adatok száma
n_test = 0 # Tesztadatok száma
n_batch = 0 # Ennyi adatcsomaggal dolgozunk
n_true_train = 0 # Pozitív tanító adatok száma
n_true_test = 0 # Pozitív tesztadatok száma
n_false_train = 0 # Negatív tanító adatok száma
n_false_test = 0 # Negatív tesztadatok száma

index_true = [] # Pozitív minták indexei
index_false = [] # Negatív minták indexei
index_train_true = [] # Pozitív tanító adatok indexei
index_test_true = [] # Pozitív tesztadatok indexei
index_train_false = [] # Negatív tanító adatok indexei
index_test_false = [] # Negatív tesztadatok indexei

data_true = np.zeros((0, n_labeled_data), dtype=np.int64) # Pozitív minták és a hozzájuk tartozó címkék tömbje
data_false = np.zeros((0, n_labeled_data), dtype=np.int64) # Negatív minták és a hozzájuk tartozó címkék tömbje
data_all = np.zeros((0, n_labeled_data), dtype=np.int64) # Összes mintát tartalmazó tömb
data_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok tömbje
data_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok tömbje
label_train = np.zeros((0, n_labeled_data), dtype=np.int64) # Tanító adatok címkéinek tömbje
label_test = np.zeros((0, n_labeled_data), dtype=np.int64) # Tesztadatok címkéinek tömbje

# 1. KONVOLÚCIÓS HÁLÓ #####
n_reteg = 0 # Háló bemeneti konvolúciós rétegen felüli konvolúciós rétegek száma
depth = 1 # Rétegenkénti szűrőszám

stride = (4, 4, 4)
kernel = (4, 4, 4)
pooling = (2, 2, 2)

halo = Sequential()

halo.add(Conv3D(depth, strides=stride, kernel_size=kernel, activation='linear', input_shape=(n_frame, n_row, n_column, 1),
padding='same', data_format='channels_last'))
halo.add(MaxPooling3D(pooling, padding='same'))

k_reteg = 0
i_reteg = 0
for i_reteg in range(n_reteg):
    halo.add(Conv3D((i_reteg + 2)*depth, kernel, activation='linear', padding='same'))
    halo.add(MaxPooling3D(pool_size=pooling, padding='same'))
    k_reteg = i_reteg

```

```

halo.add(Flatten())
halo.add(Dense((i_reteg + 2)*depth, activation='linear'))
halo.add(Dense(n_category, activation='softmax'))

halo.compile(loss='categorical_hinge', optimizer='Adadelta', metrics=['accuracy'])

# 2. ADATOK INDEXEINEK ELŐÁLLÍTÁSA #####

# Pozitív minták indexei
for i_cluster in range(n_cluster):

    # Indexek beolvasása ev2 fájlból
    evfn = "Arat" + str(i_cluster) + ".ev2"
    # evfn = "Arat_2016_10_12_0_002_48.ev2"
    index_true_temp = np.loadtxt(evfn, dtype='int', skiprows=0, usecols=None)
    n_spike = index_true_temp.shape[0] # Pozitív minták száma az adott klaszterre

    index_true_temp = index_true_temp[0:n_spike, 5] # Utolsó oszlop tartalmazza a túske maximumához tartozó minta indexét
    index_true_temp = np.array(index_true_temp, dtype=np.int64)

    index_true = np.hstack([index_true, index_true_temp]) # A többihez vesszük

n_true = index_true.shape[0]

for i in range(n_true):
    index_true[i] = index_true[i] - 20 - 2*np.random.randint(0, 5) # Véletlenszerű kismértékű eltolás

index_true = index_true * n_byte # A dat fájlban ekkora ugrásonként következik új minta
index_true = np.array(index_true, dtype=np.int64)
index_true = np.sort(index_true)

# Negatív minták a pozitívak között harmad- és kétharmadúton, lefelé egész számra kerekítve
# Mivel az első keret előtt is keressük mintákat, az indexelést nullától kezdjük, az utolsó indexet levágvva
index_true_offset = np.zeros(n_true)
index_true_offset[1:n_true] = index_true[0:n_true-1]

# Az előbb számított indexekhez képesti eltolások
offset1 = np.round((index_true - index_true_offset)/3)
offset1 = offset1.astype(np.int64)
offset2 = 2*offset1

# Az eltolt indexek összefűzése, majd sorbarendezése
index_false = np.hstack([index_true_offset + offset1, index_true_offset + offset2])
index_false = index_false.astype(np.int64)
index_false = np.sort(index_false)
n_false = index_false.shape[0]

# Indexek szétválasztása tanító- és tesztadatok indexeire
index_train_true = index_true[0:int(round(training_rate*n_true))]
index_train_false = index_false[0:int(round(training_rate*n_false))]
n_train_true = index_train_true.shape[0]
n_train_false = index_train_false.shape[0]

index_test_true = index_true[int(round(training_rate*n_true))+1:n_true]
index_test_false = index_false[int(round(training_rate*n_false))+1:n_false]
n_test_true = index_test_true.shape[0]
n_test_false = index_test_false.shape[0]

# 3. ADATOK BETÖLTÉSE, ELŐFELDOLGOZÁS, HÁLÓ TANÍTÁSA #####
# Mindez ciklikusan, mintacsomagonként, hogy a memóriában elférjenek az adatok
n_batch = int(np.floor(min(n_train_true, n_train_false)/batch_size)) # Adatcsomagok száma
print(n_batch)

n_true_batch = batch_size
n_false_batch = 2*batch_size

n_train_true = int(round(n_true_batch * training_rate))
n_train_false = int(round(n_false_batch * training_rate))

for k in range(n_batch):
    print(k)
    # 3.1. Minták beolvasása dat fájlból #####
    data_true = np.zeros((n_true_batch, n_labeled_data), dtype=np.int64)
    data_false = np.zeros((n_false_batch, n_labeled_data), dtype=np.int64)

    with open(fn, "rb") as f:
        f.seek(0) # Fájl elejére ugrás
        for j in range(n_true_batch):
            f.seek(index_train_true[j + k*n_true_batch]) # Minta elejére ugrás
            for i in range(n_sample):
                data_true[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True) # Minta betöltése
            data_true[0:n_true_batch, n_sample] = 0 # Klaszterhez tartozó minták felcímkézése

    with open(fn, "rb") as f:
        f.seek(0)
        for j in range(n_false_batch):
            f.seek(index_train_false[j + k*n_false_batch])
            for i in range(n_sample):
                data_false[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True)
            data_false[0:n_false_batch, n_sample] = 1

    # Adatok közös mátrixba vétele
    data_all = np.vstack([data_true, data_false])
    n_all = n_true_batch + n_false_batch

    # 3.2. Előfeldolgozás #####
    # Adatsorok normálása
    data_all = data_all.astype(dtype=float)

    # Közös norma az összes időszelre
    # minimum = data_all[0:n_all, 0:n_sample].min()
    # maximum = data_all[0:n_all, 0:n_sample].max()
    # eltolas = (maximum + minimum) / 2

```



```

# skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2
# data_all[0:n_all, 0:n_sample] = (data_all[0:n_all, 0:n_sample] - eltolas) / skalafaktor

# Külön minden időszületre
minimum = data_all[0:n_all, 0:n_sample].min(axis=1)
maximum = data_all[0:n_all, 0:n_sample].max(axis=1)
minimum = minimum.reshape((n_all, 1))
maximum = maximum.reshape((n_all, 1))
eltolas = (maximum + minimum) / 2
skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2

data_all[0:n_all, 0:n_sample] = np.subtract(data_all[0:n_all, 0:n_sample], eltolas)
data_all[0:n_all, 0:n_sample] = np.divide(data_all[0:n_all, 0:n_sample], skalafaktor)

# Címkek és adatok szétvágása külön mátrixokra
label_train = data_all[0:n_all, n_sample]
data_train = data_all[0:n_all, 0:n_sample]

# Átalakítás a háló számára
data_train = data_train.reshape(n_all, n_frame, n_row, n_column, 1)
label_train = keras.utils.to_categorical(label_train, num_classes=n_category)

# Háló tanítása
halo.fit(data_train, label_train, batch_size=round(n_all/100), epochs=3)

# 4. TESZT A TANÍTÁSI FOLYAMAT UTÁN #####
n_batch = int(np.floor((min(n_true_test, n_false_test)/batch_size))) # Adatcsomagok száma
print(n_batch)

n_true_batch = batch_size
n_false_batch = 2*batch_size

for k in range(n_batch):
    print(k)
    # 3.1. Minták beolvasása dat fájlból #####
    data_true = np.zeros((n_true_batch, n_labeled_data), dtype=np.int64)
    data_false = np.zeros((n_false_batch, n_labeled_data), dtype=np.int64)

    with open(fn, "rb") as f:
        f.seek(0) # Fájl elejére ugrás
        for j in range(n_true_batch):
            f.seek(index_test_true[j] + k*n_true_batch) # Minta elejére ugrás
            for i in range(n_sample):
                data_true[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True) # Minta betöltése
    data_true[0:n_true_batch, n_sample] = 0 # Klaszterhez tartozó minták felcímkézése

    with open(fn, "rb") as f:
        f.seek(0)
        for j in range(n_false_batch):
            f.seek(index_test_false[j] + k*n_false_batch)
            for i in range(n_sample):
                data_false[j, i] = int.from_bytes(f.read(2), byteorder='little', signed=True)
    data_false[0:n_false_batch, n_sample] = 1

    # Adatok közös mátrixba vétele
    data_all = np.vstack([data_true, data_false])
    n_all = n_true_batch + n_false_batch

    # 3.2. Előfeldolgozás #####
    # Adatsorok normálása
    data_all = data_all.astype(dtype=float)

    # Közös norma az összes időszületre
    # minimum = data_all[0:n_all, 0:n_sample].min()
    # maximum = data_all[0:n_all, 0:n_sample].max()
    # eltolas = (maximum + minimum) / 2
    # skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2
    # data_all[0:n_all, 0:n_sample] = (data_all[0:n_all, 0:n_sample] - eltolas) / skalafaktor

    # Külön minden időszületre
    minimum = data_all[0:n_all, 0:n_sample].min(axis=1)
    maximum = data_all[0:n_all, 0:n_sample].max(axis=1)
    minimum = minimum.reshape((n_all, 1))
    maximum = maximum.reshape((n_all, 1))
    eltolas = (maximum + minimum) / 2
    skalafaktor = (np.abs(maximum) + np.abs(minimum)) / 2

    data_all[0:n_all, 0:n_sample] = np.subtract(data_all[0:n_all, 0:n_sample], eltolas)
    data_all[0:n_all, 0:n_sample] = np.divide(data_all[0:n_all, 0:n_sample], skalafaktor)

    # Címkek és adatok szétvágása külön mátrixokra
    label_test = data_all[0:n_all, n_sample]
    label_test0 = label_test # Címkek félretétele a konfúziós mátrixhoz
    data_test = data_all[0:n_all, 0:n_sample]

    # Átalakítás a háló számára
    data_test = data_test.reshape(n_all, n_frame, n_row, n_column, 1)
    label_test = keras.utils.to_categorical(label_test, num_classes=n_category)

    # 4.2. Konfúziós mátrix #####
    label_predicted = halo.predict(data_test)
    label_predicted0 = np.zeros(n_all)

    for i in range(n_all):
        label_predicted0[i] = np.argmax(label_predicted[i, 0:n_category])
    cm = confusion_matrix(label_test0, label_predicted0)
    print(cm)

    # 4.3. Tesztadatok kiértékelése #####
    score = halo.evaluate(data_test, label_test, batch_size=n_all)
    print(score)

```

```
# Fájlbá írás
with open(ofn, 'a') as file:
    file.write(str(cm))
    file.write('\n')
    file.write(str(score))
    file.write('\n')

end = time.time()
elapsed = end - start
print("Eltelt idő: " + str(elapsed))
with open(ofn, 'a') as file:
    file.write("Eltelt idő: " + str(elapsed))
    file.write('\n')
```

MELLÉKLET B

Teszteredmények

SGD

mean_squared_error

Table with 17 columns and 33 rows of numerical data for mean_squared_error.

0.12098155170679092

mean_absolute_error

Table with 17 columns and 33 rows of numerical data for mean_absolute_error.

0.11869888007640839

mean_absolute_percentage_error

Table with 17 columns and 33 rows of numerical data for mean_absolute_percentage_error.

0.12516644597053528

mean_squared_logarithmic_error

Table with 17 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.13619935512542725

squared_hinge

Table with 17 columns and 33 rows of numerical data for squared_hinge.

0.13125357031822205

hinge

Table with 17 columns and 33 rows of numerical data for hinge.

0.1230739951133728

categorical_hinge

101	38	11	0	14	0	6	130	21	8	0	33	2	0	0	11	0	32	0	1	8	0	2	0
35	7	6	0	14	0	12	7	9	8	0	39	2	2	0	15	1	17	0	0	25	0	1	0
16	0	2	2	5	4	5	0	12	3	2	0	1	0	0	3	7	0	0	0	2	0	0	0
11	1	0	7	3	1	29	2	15	2	0	0	3	2	5	2	1	3	3	0	16	0	0	1
8	36	7	2	19	0	45	11	12	0	0	6	1	0	0	52	0	12	0	2	1	0	0	0
2	1	0	0	0	63	4	1	1	0	1	1	0	0	1	23	3	4	1	0	10	1	0	0
35	3	3	26	3	3	183	65	151	4	16	47	51	3	3	25	1	10	11	2	12	0	0	0
91	19	1	4	27	0	3	68	24	2	0	4	2	0	1	64	2	2	0	3	21	0	0	0
10	9	4	22	3	7	157	34	81	6	14	22	43	0	1	25	3	6	11	3	6	1	6	0
25	1	0	1	4	0	0	14	6	3	0	5	4	0	0	30	3	2	0	0	0	0	0	0
9	4	0	0	8	0	6	25	7	2	0	4	4	0	0	21	1	7	0	2	4	0	0	0
81	24	7	0	29	2	9	32	14	9	1	43	7	3	0	12	0	11	0	0	19	1	2	0
27	24	1	4	8	4	84	21	21	7	1	13	16	1	1	26	1	17	5	3	6	0	0	0
8	5	0	0	7	4	3	2	3	1	1	1	0	4	1	13	11	1	5	1	0	0	0	0
13	7	1	8	5	0	12	9	6	3	5	15	1	0	1	2	3	2	0	1	17	0	0	0
64	36	7	17	21	6	67	68	62	2	2	7	35	0	1	100	61	15	9	9	54	0	0	0
1	5	2	0	3	11	38	3	5	19	3	3	10	27	29	41	7	2	0	0	58	11	0	0
14	20	6	0	13	0	33	20	18	1	0	2	4	0	0	44	4	19	0	1	2	0	0	0
22	1	0	0	2	0	0	36	1	11	0	2	1	0	0	16	3	8	0	0	9	1	0	0
4	0	0	0	1	6	27	0	0	1	2	0	11	0	7	2	9	0	2	0	4	0	0	0
23	5	3	1	16	0	35	8	38	11	1	20	5	5	9	13	24	18	0	1	83	4	0	0
4	1	0	0	0	0	0	4	4	1	0	1	0	0	2	0	2	0	0	3	3	0	0	0
3	3	0	4	1	0	0	7	0	0	0	5	0	0	0	2	0	0	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0.15997716784477234

logcosh

0	0	0	0	0	0	0	220	0	0	0	0	0	0	0	39	122	37	0	0	0	0	0	0
0	0	0	0	0	0	0	116	0	0	0	0	0	0	0	16	29	39	0	0	0	0	0	0
0	0	0	0	0	0	0	33	0	0	0	0	0	0	0	8	5	18	0	0	0	0	0	0
0	0	0	0	0	0	0	82	0	0	0	0	0	0	0	3	2	20	0	0	0	0	0	0
0	0	0	0	0	0	0	120	0	0	0	0	0	0	0	20	58	16	0	0	0	0	0	0
0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	4	1	13	0	0	0	0	0	0
0	0	0	0	0	0	0	415	0	0	0	0	0	0	0	40	66	136	0	0	0	0	0	0
0	0	0	0	0	0	0	214	0	0	0	0	0	0	0	27	65	32	0	0	0	0	0	0
0	0	0	0	0	0	0	273	0	0	0	0	0	0	0	26	21	154	0	0	0	0	0	0
0	0	0	0	0	0	0	41	0	0	0	0	0	0	0	8	28	21	0	0	0	0	0	0
0	0	0	0	0	0	0	59	0	0	0	0	0	0	0	10	28	7	0	0	0	0	0	0
0	0	0	0	0	0	0	158	0	0	0	0	0	0	0	23	79	46	0	0	0	0	0	0
0	0	0	0	0	0	0	193	0	0	0	0	0	0	0	14	35	49	0	0	0	0	0	0
0	0	0	0	0	0	0	46	0	0	0	0	0	0	0	2	9	14	0	0	0	0	0	0
0	0	0	0	0	0	0	75	0	0	0	0	0	0	0	7	5	24	0	0	0	0	0	0
0	0	0	0	0	0	0	470	0	0	0	0	0	0	0	29	32	112	0	0	0	0	0	0
0	0	0	0	0	0	0	171	0	0	0	0	0	0	0	16	8	88	0	0	0	0	0	0
0	0	0	0	0	0	0	115	0	0	0	0	0	0	0	21	48	17	0	0	0	0	0	0
0	0	0	0	0	0	0	44	0	0	0	0	0	0	0	10	15	44	0	0	0	0	0	0
0	0	0	0	0	0	0	38	0	0	0	0	0	0	0	9	0	29	0	0	0	0	0	0
0	0	0	0	0	0	0	187	0	0	0	0	0	0	0	10	21	105	0	0	0	0	0	0
0	0	0	0	0	0	0	8	0	0	0	0	0	0	0	1	2	11	0	0	0	0	0	0
0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	1	5	5	0	0	0	0	0	0

0.08921437710523605

categorical_crossentropy

397	0	0	0	0	1	0	3	3	0	1	0	7	0	0	1	2	0	3	0	0	0	0	0
34	30	0	0	0	0	2	0	0	0	0	127	0	0	0	1	0	0	0	0	5	0	1	0
0	0	2	3	1	0	0	34	1	0	2	0	0	0	4	6	3	5	0	0	3	0	0	0
1	1	3	20	2	4	0	16	13	0	3	0	28	0	0	6	0	0	0	0	10	0	0	0
5	0	0	0	175	0	0	5	0	0	0	0	0	0	1	1	0	27	0	0	0	0	0	0
0	0	0	0	0	96	0	5	0	0	0	0	0	1	0	0	3	0	8	0	4	0	0	0
1	0	1	0	3	0	568	2	70	0	0	2	5	0	1	4	0	0	0	0	0	0	0	0
23	0	0	0	3	0	4	281	0	4	1	0	0	0	2	14	0	6	0	0	0	0	0	0
1	0	0	0	3	0	296	4	152	0	0	2	15	0	0	1	0	0	0	0	0	0	0	0
5	0	0	0	2	0	0	37	1	44	0	0	0	0	0	5	0	4	0	0	0	0	0	0
10	0	0	0	3	0	2	54	0	1	13	0	0	0	0	9	0	11	1	0	0	0	0	0
101	20	0	0	3	0	5	0	0	0	1	168	1	0	3	1	0	0	1	0	2	0	0	0
13	0	0	3	39	0	40	34	40	0	4	4	69	0	3	21	1	16	2	0	2	0	0	0
3	0	0	0	0	1	1	7	0	3	1	0	4	28	0	10	2	4	6	0	1	0	0	0
4	0	1	0	0	0	1	10	0	0	1	0	0	0	89	1	3	0	0	0	1	0	0	0
12	1	0	0	6	0	6	19	1	0	0	0	0	0	2	587	0	9	0	0	0	0	0	0
0	2	0	1	0	0	1	2	1	0	1	0	3	0	1	2	117	0	0	0	152	0	0	0
8	0	0	0	69	0	2	9	0	2	0	0	0	0	0	0	0	111	0	0	0	0	0	0
7	0	0	0	0	0	0	11	0	0	4	0	0	0	0	5	0	21	63	1	0	0	1	0
0	0	7	17	0	0	0	1	0	0	0	0	26	1	0	8	0	1	1	0	14	0	0	0
21	22	0	0	1	0	0	12	0	1	7	6	2	0	11	8	2	1	1	0	228	0	0	0
0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	1	0	9	7	0	0	0	0	0
8	0	0	0	0	1	0	0	8	0	5	0	0	0	0	0	0	3	1	0	0	0	0	0

0.6159406304359436

kullback_leibler_divergence

397	0	0	0	1	0	2	1	1	1	0	8	0	0	2	2	0	3	0	0	0	0	0	0
54	32	0	0	0	0	2	0	0	1	0	105	0	0	0	2	0	0	0	0	4	0	0	0
0	0	1	3	3	0	0	21	2	0	7	0	0	0	2	12	3	8	0	0	2	0	0	0
1	1	1	20	2	8	0	11	11	0	3	0	28	0	0	8	0	0	0	0	13	0	0	0
8	0	0	0	183	0	0	4	0	0	0	0	0	0	1	3	0	15	0	0	0	0	0	0
0	0	0	1	0	96	0	5	0	0	0	0	0	2	0	0	2	1	7	0	3	0	0	0
0	0	0	0	4	0	478	2	159	0	0	2	7	0	1	4	0	0	0	0	0	0	0	0
25	0	0	0	3	0	4	271	0	4	1	0	0											

Adagrad

mean_squared_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_squared_error.

0.6322997808456421

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_error.

0.4493056833744049

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_percentage_error.

0.3422103822231293

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.6770020723342896

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.5773254632949829

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.38482025265693665

Adadelta

mean_squared_error

Table with 17 columns and 33 rows of numerical data for mean_squared_error.

0.5613467693328857

mean_absolute_error

Table with 17 columns and 33 rows of numerical data for mean_absolute_error.

0.12497622519731522

mean_absolute_percentage_error

Table with 17 columns and 33 rows of numerical data for mean_absolute_percentage_error.

0.6011033058166504

mean_squared_logarithmic_error

Table with 17 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.335526030063629

squared_hinge

Table with 17 columns and 33 rows of numerical data for squared_hinge.

0.39813581109046936

hinge

Table with 17 columns and 33 rows of numerical data for hinge.

0.24386532604694366

Adam

mean_squared_error

Table with 20 columns and 20 rows of numerical data for mean_squared_error.

0.7812440395355225

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_error.

0.5592543482780457

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_percentage_error.

0.5881681442260742

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.7957009673118591

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.7310252785682678

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.4616701602935791

categoryal_hinge

Table with 20 columns and 31 rows of numerical data for the 'categoryal_hinge' metric.

0.7791516184806824

logcosh

Table with 20 columns and 31 rows of numerical data for the 'logcosh' metric.

0.7880920767784119

categoryal_crossentropy

Table with 20 columns and 31 rows of numerical data for the 'categoryal_crossentropy' metric.

0.768118679523468

kullback_leibler_divergence

Table with 20 columns and 31 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7633631229400635

poisson

Table with 20 columns and 31 rows of numerical data for the 'poisson' metric.

0.7827658653259277

cosine_proximity

Table with 20 columns and 31 rows of numerical data for the 'cosine_proximity' metric.

0.7637435793876648

Adamax

mean_squared_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_squared_error.

0.793418288230896

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_absolute_error.

0.5546889901161194

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_absolute_percentage_error.

0.5893095135688782

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.7801027297973633

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.7268403768539429

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.5470800995826721

Nadam

mean_squared_error

Table with 20 columns and 33 rows of numerical data for mean_squared_error.

0.7772493958473206

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for mean_absolute_error.

0.6022446155548096

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for mean_absolute_percentage_error.

0.6703442931175232

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.775537371635437

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.7422484159469604

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.6572189331054688

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7873311638832092

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7854289412498474

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.8006467819213867

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7932280898094177

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7884725332260132

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7888529300689697

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7595586776733398

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7922769784927368

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.8025490045547485

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7605097889900208

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7721133828163147

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7953205108642578

SGD

mean_squared_error

Table with 20 columns and 30 rows of numerical data for mean_squared_error.

0.12364466488361359

mean_absolute_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_error.

0.12497622519731522

mean_absolute_percentage_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_percentage_error.

0.03500095009803772

mean_squared_logarithmic_error

Table with 20 columns and 30 rows of numerical data for mean_squared_logarithmic_error.

0.1285904496908188

squared_hinge

Table with 20 columns and 30 rows of numerical data for squared_hinge.

0.12231310456991196

hinge

Table with 20 columns and 30 rows of numerical data for hinge.

0.133536234498024

0.21932661533355713

categorical_hinge

135	33	1	5	34	1	6	46	7	7	26	8	0	2	35	41	8	4	5	1	10	0	3
32	7	3	1	23	2	5	2	8	4	12	3	16	3	9	32	9	4	0	0	24	0	1
0	4	5	0	0	0	0	17	8	2	0	1	0	0	2	13	2	4	1	2	2	0	1
4	1	2	8	1	16	7	3	3	0	5	2	4	2	22	3	0	6	2	13	0	0	0
16	1	0	0	64	0	27	23	0	1	12	7	0	2	6	4	0	10	28	7	1	0	5
2	3	0	0	0	78	2	0	0	2	0	0	16	1	0	3	5	1	1	0	2	1	0
15	2	0	4	2	5	178	24	181	11	1	17	22	5	15	125	6	23	12	1	0	5	3
21	36	1	8	67	0	8	70	24	4	4	16	0	0	5	55	0	1	2	8	8	0	0
11	0	2	4	0	9	86	20	155	6	0	22	21	4	26	53	9	13	17	0	9	7	0
8	6	0	1	10	0	0	1	29	4	2	15	0	0	9	4	1	2	2	4	0	0	0
24	20	0	0	2	1	15	0	1	0	1	4	0	0	0	24	0	9	1	1	0	0	1
77	12	2	5	34	2	4	14	10	7	11	18	6	5	7	30	25	5	1	1	29	0	1
10	3	1	0	3	2	66	22	28	3	14	8	20	6	0	49	3	20	16	9	7	1	0
7	6	5	0	4	10	1	2	3	0	0	3	5	4	0	7	1	0	4	0	9	0	0
14	0	0	0	2	0	20	0	10	1	1	0	0	1	20	19	7	0	0	6	10	0	0
11	8	0	15	22	5	65	28	25	24	18	25	10	7	19	260	11	4	14	59	7	2	4
8	2	5	10	9	11	11	2	33	5	0	4	29	22	2	21	33	4	0	0	72	0	0
12	3	0	2	51	1	17	35	4	0	10	8	0	1	1	17	1	9	23	5	1	0	0
24	10	0	0	12	1	2	9	4	1	3	3	1	1	5	5	0	11	13	2	5	0	1
3	1	1	1	1	7	12	5	2	1	1	1	11	8	0	9	3	0	1	1	7	0	0
15	5	2	0	9	5	26	1	46	5	7	6	19	4	4	62	29	2	0	4	70	0	2
1	0	0	0	0	0	1	3	0	0	0	0	0	0	1	0	2	0	7	4	0	0	0
4	2	0	0	3	0	0	0	5	2	3	4	0	0	0	2	1	0	0	0	0	0	0

0.21932661533355713

logcosh

0	0	0	0	0	0	418	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	200	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	63	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	99	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	214	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	112	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	652	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	338	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	456	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	104	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	302	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	284	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	66	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	111	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	639	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	265	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	201	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	113	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	74	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	311	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0.12402510643005371

categorical_crossentropy

401	0	0	0	2	0	2	2	0	1	0	6	0	0	2	1	0	1	0	0	0	0	0
53	32	0	0	0	0	2	0	0	0	0	109	0	0	0	1	0	0	0	0	2	0	1
1	0	5	2	0	0	29	1	1	2	0	0	5	6	5	4	0	0	1	2	0	0	0
2	1	2	21	2	5	0	7	23	0	2	0	24	1	0	7	0	0	0	0	9	0	1
7	0	0	0	186	0	0	8	0	0	0	0	0	1	1	0	11	0	0	0	0	0	0
0	0	1	0	0	95	0	5	0	0	0	0	1	0	0	3	0	9	0	3	0	0	0
1	0	0	0	3	0	534	1	106	0	0	2	5	0	1	4	0	0	0	0	0	0	0
13	0	0	0	0	0	3	306	1	2	0	0	0	2	8	0	3	0	0	0	0	0	0
2	0	0	0	3	0	258	4	191	0	1	0	15	0	0	0	0	0	0	0	0	0	0
4	0	0	0	2	0	0	39	1	49	0	0	0	0	2	0	1	0	0	0	0	0	0
13	0	0	0	2	0	2	50	0	0	13	0	0	0	12	0	11	0	0	0	0	0	1
135	20	0	0	2	0	6	1	0	0	0	134	0	0	4	1	0	1	0	0	2	0	0
15	0	0	4	34	0	30	25	73	0	6	4	59	0	2	24	1	12	1	0	1	0	0
2	0	0	1	0	2	1	7	0	7	0	0	5	21	0	10	3	5	4	1	1	0	1
2	1	0	0	2	0	1	8	0	0	1	0	0	0	89	0	5	0	0	0	2	0	0
13	1	0	0	6	4	6	22	3	1	1	0	0	0	2	570	0	12	1	0	0	1	0
0	3	0	1	0	0	1	2	2	0	1	0	4	0	1	2	117	0	0	2	147	0	0
9	0	0	0	79	0	2	9	0	2	0	0	0	0	0	3	0	97	0	0	0	0	0
6	0	0	0	0	0	0	9	1	0	3	0	0	0	0	5	0	23	66	0	0	0	0
0	0	9	14	0	0	0	0	2	0	0	0	34	0	0	4	0	0	0	1	12	0	0
24	25	0	0	1	0	0	10	1	2	6	3	1	0	8	9	1	1	0	0	231	0	0
1	0	0	0	0	0	0	3	0	0	0	0	0	0	0	2	0	8	8	0	0	0	0
8	0	0	0	1	0	0	8	0	5	0	0	0	0	0	0	0	3	1	0	0	0	0

0.61213618516922

kullback_leibler_divergence

395	0	0	0	2	0	6	3	0	1	0	6	0	0	2	1	0	1	0	0	0	0	1
35	38	0	0	0	0	2	0	0	2	0	119	0	0	1	2	0	0	0	0	0	0	1
0	0	9	2	1	0	0	22	1	0	3	0	0	0	3	7	2	9	0	1	2	2	0
1	1	2	16	2	5	0	9	11	0	2	0	35	1	0	8	0	0	0	0	13	0	1
6	0	0	0	175	0	0	3	0	0	0	0	0	0	1	2	0	27	0	0	0	0	0
0	0	0	0	0	95	0	3	0	0	0	0	1	2	0	2	1	0	12	0	1	0	0
0	0	0	0	2	0	583	1	58	0	0	2	5	0	1	4	0	0	0	1	0	0	0
14	0	0	0	0	0	4	300	0	1	1	0	0	0	2	13	0	3	0	0	0	0	0
1	0	0	0	1	0	313	4	129	0	1	0	24	0	1	0	0	0	0	0	0	0	0
6	0	0	0	2	0	0	39	0	39	0	0	1	0	0	4	0	6	0	0	0	0	1
5	0	0	0	2	0	2	41	0	1	22	1	0	0	0	15	0	12	2	0	0	0	1
104	20	0	0	2	0	6	1	0	0	1	164	1	0	2	2	0	0	1	0	2	0	0
14	0	2	3	29	1	43	24	51	0	7	3	73	0	3	22	1	11					

Adagrad

mean_squared_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_squared_error.

0.7359710931777954

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_error.

0.49495911598205566

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_percentage_error.

0.28932851552963257

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.7392048835754395

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.5465094447135925

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.4814532955223999

Adadelta

mean_squared_error

Table with 18 columns and 33 rows of numerical data for Adadelta mean_squared_error.

0.5797983407974243

mean_absolute_error

Table with 18 columns and 33 rows of numerical data for Adadelta mean_absolute_error.

0.24538710713386536

mean_absolute_percentage_error

Table with 18 columns and 33 rows of numerical data for Adadelta mean_absolute_percentage_error.

0.6267833113670349

mean_squared_logarithmic_error

Table with 18 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.3465855121612549

squared_hinge

Table with 18 columns and 33 rows of numerical data for squared_hinge.

0.41202205419540405

hinge

Table with 18 columns and 33 rows of numerical data for hinge.

0.24291421473026276

Adam

mean_squared_error

Table with 28 columns and 33 rows of numerical data for mean_squared_error.

0.7985543012619019

mean_absolute_error

Table with 28 columns and 33 rows of numerical data for mean_absolute_error.

0.44664257764816284

mean_absolute_percentage_error

Table with 28 columns and 33 rows of numerical data for mean_absolute_percentage_error.

0.5891192555427551

mean_squared_logarithmic_error

Table with 28 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.7856191992759705

squared_hinge

Table with 28 columns and 33 rows of numerical data for squared_hinge.

0.7542324662208557

hinge

Table with 28 columns and 33 rows of numerical data for hinge.

0.6092828512191772

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7743960618972778

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7825756072998047

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7783907055854797

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7671675682067871

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.783365201950073

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7757276296615601

Adamax

mean_squared_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_squared_error.

0.7939889430999756

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_absolute_error.

0.5611565709114075

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for Adamax mean_absolute_percentage_error.

0.6452349424362183

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.797793447971344

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.7203728556632996

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.5716187953948975

categoryal_hinge

Table with 20 columns and 31 rows of numerical data for the 'categoryal_hinge' metric.

0.7960814237594604

logcosh

Table with 20 columns and 31 rows of numerical data for the 'logcosh' metric.

0.7987445592880249

categoryal_crossentropy

Table with 20 columns and 31 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7780102491378784

kullback_leibler_divergence

Table with 20 columns and 31 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7842876315116882

poisson

Table with 20 columns and 31 rows of numerical data for the 'poisson' metric.

0.7899942994117737

cosine_proximity

Table with 20 columns and 31 rows of numerical data for the 'cosine_proximity' metric.

0.7899942994117737

Nadam

mean_squared_error

Table with 20 columns and 20 rows of numerical data for the Nadam mean_squared_error metric.

0.7783907055854797

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for the Nadam mean_absolute_error metric.

0.6307780146598816

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for the Nadam mean_absolute_percentage_error metric.

0.6830891966819763

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for the mean_squared_logarithmic_error metric.

0.7953205108642578

squared_hinge

Table with 20 columns and 20 rows of numerical data for the squared_hinge metric.

0.743960440158844

hinge

Table with 20 columns and 20 rows of numerical data for the hinge metric.

0.6089023947715759

category_hinge

Table with 20 columns and 20 rows of numerical data for category_hinge.

0.787140965461731

logcosh

Table with 20 columns and 20 rows of numerical data for logcosh.

0.7846680879592896

category_crossentropy

Table with 20 columns and 20 rows of numerical data for category_crossentropy.

0.7972227334976196

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for kullback_leibler_divergence.

0.7789613604545593

poisson

Table with 20 columns and 20 rows of numerical data for poisson.

0.7888529300689697

cosine_proximity

Table with 20 columns and 20 rows of numerical data for cosine_proximity.

0.788472532260132

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7584173679351807

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7740156054496765

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7806733846664429

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.77686893997192

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7586075663566589

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7957009673118591

SGD

mean_squared_error

Table with 20 columns and 20 rows of numerical data for mean_squared_error.

0.12440555542707443

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_error.

0.12497622519731522

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_percentage_error.

0.0983450636267662

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.11356286704540253

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.12497622519731522

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.1171770989894867

categoryal_hinge

Table with 21 columns and 32 rows of numerical data for the 'categoryal_hinge' metric.

0.12878067791461945

logcosh

Table with 21 columns and 32 rows of numerical data for the 'logcosh' metric.

0.12478599697351456

categoryal_crossentropy

Table with 21 columns and 32 rows of numerical data for the 'categoryal_crossentropy' metric.

0.6448544859886169

kullback_leibler_divergence

Table with 21 columns and 32 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.6248810887336731

poisson

Table with 21 columns and 32 rows of numerical data for the 'poisson' metric.

0.13524824380874634

cosine_proximity

Table with 21 columns and 32 rows of numerical data for the 'cosine_proximity' metric.

0.21590261161327362

Adagrad

mean_squared_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_squared_error.

0.630207359790802

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_error.

0.2630777955055237

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for Adagrad mean_absolute_percentage_error.

0.30169299244880676

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.6587406992912292

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.5484116673469543

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.3924291431903839

categoryal_hinge

Table with 20 columns and 31 rows of numerical data for the categoryal_hinge metric.

0.5790374875068665

logcosh

Table with 20 columns and 31 rows of numerical data for the logcosh metric.

0.6431424617767334

categoryal_crossentropy

Table with 20 columns and 31 rows of numerical data for the categoryal_crossentropy metric.

0.66596919298172

kullback_leibler_divergence

Table with 20 columns and 31 rows of numerical data for the kullback_leibler_divergence metric.

0.6279246807098389

poisson

Table with 20 columns and 31 rows of numerical data for the poisson metric.

0.625832200050354

cosine_proximity

Table with 20 columns and 31 rows of numerical data for the cosine_proximity metric.

0.6634963154792786

Adadelta

mean_squared_error

Table with 16 columns and 37 rows of numerical data, representing mean_squared_error for Adadelta.

0.6050979495048523

mean_absolute_error

Table with 16 columns and 37 rows of numerical data, representing mean_absolute_error for Adadelta.

0.2451968789100647

mean_absolute_percentage_error

Table with 16 columns and 37 rows of numerical data, representing mean_absolute_percentage_error for Adadelta.

0.5031386613845825

mean_squared_logarithmic_error

Table with 16 columns and 37 rows of numerical data, representing mean_squared_logarithmic_error.

0.3330796957015991

squared_hinge

Table with 16 columns and 37 rows of numerical data, representing squared_hinge.

0.44759368896484375

hinge

Table with 16 columns and 37 rows of numerical data, representing hinge.

0.12231310456991196

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7605097889900208

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.15731406211853027

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.782956063747406

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7726840376853943

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7808635830879211

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7521399855613708

Adam

mean_squared_error

Table with 20 columns and 20 rows of numerical data for mean_squared_error.

0.7806733846664429

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_error.

0.5263458490371704

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_percentage_error.

0.4903937578201294

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.7732546925544739

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.7133346199989319

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.5592543482780457

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for 'categoryal_hinge'.

0.7730644941329956

logcosh

Table with 20 columns and 20 rows of numerical data for 'logcosh'.

0.7460528612136841

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for 'categoryal_crossentropy'.

0.765075147151947

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for 'kullback_leibler_divergence'.

0.7734449505805969

poisson

Table with 20 columns and 20 rows of numerical data for 'poisson'.

0.7820049524307251

cosine_proximity

Table with 20 columns and 20 rows of numerical data for 'cosine_proximity'.

0.7679284811019897

Adamax

mean_squared_error

Table with 20 columns and 33 rows of numerical data for Adamax mean_squared_error.

0.7574662566184998

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for Adamax mean_absolute_error.

0.42876166105270386

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for Adamax mean_absolute_percentage_error.

0.47974130511283875

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.7761080265045166

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.690127432346344

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.4413163363933563

categoryal_hinge

Table with 20 columns and 31 rows of numerical data for the 'categoryal_hinge' metric.

0.7715427279472351

logcosh

Table with 20 columns and 31 rows of numerical data for the 'logcosh' metric.

0.7709720134735107

categoryal_crossentropy

Table with 20 columns and 31 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7553737759590149

kullback_leibler_divergence

Table with 20 columns and 31 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.76868939971924

poisson

Table with 20 columns and 31 rows of numerical data for the 'poisson' metric.

0.7557542324066162

cosine_proximity

Table with 20 columns and 31 rows of numerical data for the 'cosine_proximity' metric.

0.7858093976974487

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for 'categoryal_hinge'.

0.7821951508522034

logcosh

Table with 20 columns and 20 rows of numerical data for 'logcosh'.

0.7875214219093323

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for 'categoryal_crossentropy'.

0.7757276296615601

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for 'kullback_leibler_divergence'.

0.7707818150520325

poisson

Table with 20 columns and 20 rows of numerical data for 'poisson'.

0.7991249561309814

cosine_proximity

Table with 20 columns and 20 rows of numerical data for 'cosine_proximity'.

0.8173863291740417

SGD

mean_squared_error

Table with 20 columns and 30 rows of numerical data for mean_squared_error.

0.1221228837966919

mean_absolute_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_error.

0.12497622519731522

mean_absolute_percentage_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_percentage_error.

0.021495148539543152

mean_squared_logarithmic_error

Table with 20 columns and 30 rows of numerical data for mean_squared_logarithmic_error.

0.12497622519731522

squared_hinge

Table with 20 columns and 30 rows of numerical data for squared_hinge.

0.12231310456991196

hinge

Table with 20 columns and 30 rows of numerical data for hinge.

0.11698687821626663

categoryal_hinge

Table with 20 columns and 32 rows of numerical data for the 'categoryal_hinge' metric.

0.16397184133529663

logcosh

Table with 20 columns and 32 rows of numerical data for the 'logcosh' metric.

0.1196499913930893

categoryal_crossentropy

Table with 20 columns and 40 rows of numerical data for the 'categoryal_crossentropy' metric.

0.6505611538887024

kullback_leibler_divergence

Table with 20 columns and 32 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.6509416103363037

poisson

Table with 20 columns and 32 rows of numerical data for the 'poisson' metric.

0.13715046644210815

cosine_proximity

Table with 20 columns and 32 rows of numerical data for the 'cosine_proximity' metric.

0.21095682680606842

Adagrad

mean_squared_error

Table with 20 columns and 33 rows of numerical data for mean_squared_error.

0.7268403768539429

mean_absolute_error

Table with 20 columns and 33 rows of numerical data for mean_absolute_error.

0.3119649887084961

mean_absolute_percentage_error

Table with 20 columns and 33 rows of numerical data for mean_absolute_percentage_error.

0.42590832710266113

mean_squared_logarithmic_error

Table with 20 columns and 33 rows of numerical data for mean_squared_logarithmic_error.

0.7298839688301086

squared_hinge

Table with 20 columns and 33 rows of numerical data for squared_hinge.

0.5518356561660767

hinge

Table with 20 columns and 33 rows of numerical data for hinge.

0.4858284294605255

Adadelta

mean_squared_error

Table with 20 columns and 30 rows of numerical data for mean_squared_error.

0.6005326509475708

mean_absolute_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_error.

0.12497622519731522

mean_absolute_percentage_error

Table with 20 columns and 30 rows of numerical data for mean_absolute_percentage_error.

0.6496100425720215

mean_squared_logarithmic_error

Table with 20 columns and 30 rows of numerical data for mean_squared_logarithmic_error.

0.3836789131164551

squared_hinge

Table with 20 columns and 30 rows of numerical data for squared_hinge.

0.4390336573123932

hinge

Table with 20 columns and 30 rows of numerical data for hinge.

0.24500665068626404

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for 'categoryal_hinge'.

0.7814342975616455

logcosh

Table with 20 columns and 20 rows of numerical data for 'logcosh'.

0.19630968570709229

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for 'categoryal_crossentropy'.

0.8055925369262695

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for 'kullback_leibler_divergence'.

0.787140965461731

poisson

Table with 20 columns and 20 rows of numerical data for 'poisson'.

0.8015978932380676

cosine_proximity

Table with 20 columns and 20 rows of numerical data for 'cosine_proximity'.

0.7846680879592896

Adam

mean_squared_error

Table with 20 columns and 20 rows of numerical data for mean_squared_error.

0.7844778299331665

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_error.

0.5828419327735901

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_percentage_error.

0.6011033058166504

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.7960814237594604

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.7683089375495911

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.12497622519731522

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7928476333618164

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7922769784927368

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7922769784927368

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7835267186164856

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7882822751998901

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.7743960618972778

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the categoryal_hinge metric.

0.7559444308280945

logcosh

Table with 20 columns and 20 rows of numerical data for the logcosh metric.

0.7837169766426086

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the categoryal_crossentropy metric.

0.7869507074356079

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the kullback_leibler_divergence metric.

0.7827658653259277

poisson

Table with 20 columns and 20 rows of numerical data for the poisson metric.

0.769260048866272

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the cosine_proximity metric.

0.7842876315116882

Nadam

mean_squared_error

Table with 20 columns and 20 rows of numerical data for mean_squared_error.

0.7960814237594604

mean_absolute_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_error.

0.12231310456991196

mean_absolute_percentage_error

Table with 20 columns and 20 rows of numerical data for mean_absolute_percentage_error.

0.6473273634910583

mean_squared_logarithmic_error

Table with 20 columns and 20 rows of numerical data for mean_squared_logarithmic_error.

0.7924671769142151

squared_hinge

Table with 20 columns and 20 rows of numerical data for squared_hinge.

0.79665207862854

hinge

Table with 20 columns and 20 rows of numerical data for hinge.

0.6041468381881714

categoryal_hinge

Table with 20 columns and 20 rows of numerical data for the 'categoryal_hinge' metric.

0.7559444308280945

logcosh

Table with 20 columns and 20 rows of numerical data for the 'logcosh' metric.

0.7702111601829529

categoryal_crossentropy

Table with 20 columns and 20 rows of numerical data for the 'categoryal_crossentropy' metric.

0.7726840376853943

kullback_leibler_divergence

Table with 20 columns and 20 rows of numerical data for the 'kullback_leibler_divergence' metric.

0.7896138429641724

poisson

Table with 20 columns and 20 rows of numerical data for the 'poisson' metric.

0.7863800525665283

cosine_proximity

Table with 20 columns and 20 rows of numerical data for the 'cosine_proximity' metric.

0.8052120804786682

1.2. Költséggüggvény/optimalizáló kombinációk eredményei a kiválasztott szűrőkkel rendelkező hálók esetén (4, 2) szűrő eredményei

RMSprop

mean_squared_error

4121	0	0	0	7	0	0	1	0	0	2	40	0	0	1	3	0	3	3	0	0	0	0
0	1319	0	0	0	0	0	0	0	0	0	681	0	0	0	0	0	0	0	0	0	0	0
0	3	603	0	0	0	0	1	3	0	1	0	1	0	1	22	0	7	1	0	0	0	0
0	0	0	242	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	817	4	0	0
14	1	1	0	2111	0	2	1	0	0	1	0	0	0	1	1	0	7	0	0	0	0	0
0	0	0	2	0	1157	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	11	0
38	0	0	0	7	0	3532	10	2893	6	8	14	14	1	8	37	0	3	3	0	0	0	1
38	0	1	0	13	0	14	3257	4	0	0	0	1	2	1	27	0	13	2	0	0	1	3
9	0	0	0	2	0	465	8	4156	4	6	16	36	3	2	18	1	1	4	0	4	1	1
17	0	0	0	4	0	1	4	1	903	0	3	0	15	1	3	0	5	1	0	0	1	18
4	0	0	0	2	0	1	0	0	1	1021	0	0	0	1	2	0	2	1	0	2	0	0
243	0	0	0	2	0	2	0	0	0	1	2807	0	0	1	1	0	1	0	0	0	0	1
7	9	3	2	6	0	5	1	33	1	3	2	2711	4	5	61	0	42	8	0	1	3	3
1	3	0	0	0	0	1	0	0	2	1	0	8	680	1	3	0	5	0	1	0	0	1
6	0	0	1	4	0	3	4	1	0	0	0	7	0	1072	5	5	6	0	0	1	0	0
48	0	0	0	15	0	21	12	9	6	2	14	8	1	7	6266	0	23	1	0	1	0	1
0	0	0	4	0	0	1	0	1	0	0	0	3	0	1	5	2188	0	1	0	622	0	0
22	0	0	0	217	0	16	2	1	1	3	5	2	1	0	7	0	1727	2	0	0	2	0
8	0	0	0	2	1	4	2	9	2	1	4	16	2	1	15	0	25	1026	0	2	10	3
0	0	1	92	0	0	0	0	0	0	0	0	1	4	0	1	0	4	2	651	1	0	0
1	271	2	1	0	0	0	1	1	1	0	1	35	18	6	17	0	10	6	14	2839	1	0
0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0	2	2	0	0	217	0
7	0	0	0	3	0	0	1	1	7	0	0	1	1	1	8	0	4	0	0	0	0	221

0.8528566956520081

mean_squared_logarithmic_error

4160	0	0	0	1	0	0	2	0	0	2	12	0	0	1	0	0	0	3	0	0	0	0
0	1210	0	0	0	0	0	0	0	2	0	787	1	0	0	0	0	0	0	0	0	1	0
0	3	627	0	0	0	0	1	2	0	0	0	1	0	0	3	0	4	2	0	0	0	0
0	0	0	77	0	0	0	0	0	0	0	0	6	1	0	1	1	0	4	969	7	0	0
22	0	1	0	2063	0	4	7	0	4	2	0	3	0	2	6	0	21	0	0	0	3	2
0	0	1	0	0	1132	0	0	0	0	0	0	15	0	0	0	1	0	2	0	0	21	0
42	0	0	0	4	0	3950	11	2487	10	13	7	16	2	6	17	1	2	5	0	0	0	2
36	0	1	0	6	0	10	3298	0	1	2	0	0	1	0	14	0	2	3	0	0	1	2
10	0	0	0	2	0	685	9	3954	5	5	13	29	1	1	8	1	1	10	0	1	1	1
20	0	0	0	3	0	1	2	0	0	929	0	2	1	8	0	1	0	0	1	0	0	9
4	0	1	0	1	0	1	0	0	1	1022	0	1	0	0	2	0	2	1	0	1	0	0
691	0	0	0	2	0	3	1	1	0	0	2354	1	0	2	1	0	1	1	0	0	0	1
7	7	3	0	3	0	7	1	31	1	5	1	2780	3	2	21	0	17	15	0	0	2	4
0	1	0	0	0	0	1	0	0	2	2	0	11	685	1	1	0	1	0	0	1	1	1
6	0	0	0	4	0	7	4	5	4	3	0	11	0	981	3	76	5	4	0	1	0	1
55	0	1	0	11	0	28	30	12	10	5	12	17	2	6	6223	0	15	5	0	1	0	2
0	0	0	1	0	0	2	0	0	0	1	0	7	0	1	3	2574	0	3	0	234	0	0
27	0	0	0	106	0	16	8	1	6	5	1	7	2	0	9	0	1812	5	0	0	2	1
8	0	0	0	2	0	5	4	6	4	1	4	19	1	0	5	1	7	1052	0	0	11	3
0	0	1	16	0	0	0	0	0	0	0	0	18	8	0	1	3	3	2	704	1	0	0
2	155	4	0	0	0	0	2	4	4	3	1	278	48	2	6	1	7	21	11	2674	0	2
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	8	0	0	214	0
8	0	0	0	1	0	1	1	1	17	0	0	3	1	1	3	0	0	0	0	0	0	218

0.850307285785675

squared_hinge

4120	0	0	0	2	0	0	2	0	0	1	46	0	0	2	0	0	2	3	0	0	0	3
0	1459	0	0	0	0	0	0	0	1	0	540	0	0	0	0	0	0	0	0	0	1	0
0	3	621	1	0	0	0	1	2	0	0	0	0	0	0	2	0	7	2	0	0	0	4
0	0	0	581	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	480	2	0	0
20	1	1	0	2083	0	3	4	0	2	2	0	1	0	1	1	0	12	0	0	0	3	6
0	0	0	1	0	1113	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	49	6
35	0	0	0	4	0	3581	8	2888	3	1	12	11	0	10	10	0	1	1	0	0	1	9
36	0	1	0	9	0	13	3274	2	1	0	0	1	1	1	9	0	4	2	0	0	1	22
5	0	0	0	2	0	535	9	4105	4	2	15	19	1	9	5	1	2	4	0	3	3	13
14	0	0	0	4	0	2	1	1	725	0	2	0	5	1	0	0	1	0	0	0	0	221
7	0	1	0	3	0	6	1	1	1	1003	0	2	1	1	2	0	4	1	0	2	0	1
309	1	0	0	0	0	3	0	0	0	0	2742	0	0	2	0	0	1	0	0	0	0	1
7	8	3	3	5	0	6	1	40	0	2	1	2693	4	13	15	0	31	12	0	0	7	59
0	1	0	0	0	0	1	0	0	1	0	0	7	680	1	0	0	4	0	1	0	0	11
4	0	0	1	5	0	2	3	1	0	0	0	1	0	1087	1	1	4	0	0	1	0	4
53	2	0	3	14	0	34	39	15	7	3	20	9	4	15	6103	0	59	2	0	1	1	51
1	0	0	7	0	0	2	0	0	0	0	0	7	0	3	2	2061	0	1	0	742	0	0
23	0	0	0	157	0	17	5	1	1	3	5	2	1	0	3	0	1775	1	0	0	2	12
9	0	0	0	2	0	5	4	7	1	1	4	15	1	1	6	0	18	991	0	3	33	32
0	0	1	239	0	0	0	0	0	0	0	0	1	5	0	1	0	3	2	503	1	0	1
1	302	1	9	0	0	0	0	1	2	1	1	41	26	13	3	0	10	12	10	2776	1	15
0	0	0	0	0	0	0	0	1	0	0	0	1	0	2	0	0	1	4	0	0	212	4
7	0	0	0	1	0	0	0	1	1	0	0	0	1	1	2	0	0	0	0	0	0	241

0.8471871018409729

categorical_hinge

4155	0	0	0	5	0	10	0	0	0	2	7	0	0	0	1	0	0	1	0	0	0	0
0	1457	0	0	0	0	0	0	0	0	0	544	0	0	0	0	0	0	0	0	0	0	0
0	3	610	1	0	0	0	1	2	0	0	0	0	0	0	19	0	6	1	0	0	0	0
0	0	0	688	0	0	0	0	0	0	0	0	1	0	0	1	0	2	1	370	3	0	0
17	0	1	0	2102	0	5	1	0	0	1	0	0	0	0	6	0	7	0	0	0	0	0
0	0	0	2	0	1151	0	0	0	0	3	0	0	2	0	0	1	2	10	0	0	0	0
42	0	0	0	7	0	3935	13	2496	2	5	6	9	0	8	47	1	3	1	0	0	0	0
40	0	1	0	13	0	20	3254	0	0	1	0	0	1	1	34	0	10	2	0	0	0	0
10	0	0	0	3	0	677	8	3924	3	11	15	20	0	4	46	1	10	2	0	3	0	0
21	0	0	4	7	0	4	9	2	877	2	2	9	17	3	6	0	12	2	0	0	0	0
5	0	0	0	3																		

logcosh

4164	0	0	0	5	0	0	1	0	0	1	6	0	0	0	0	3	1	0	0	0	0
0	1206	0	0	0	0	0	0	0	1	0	792	0	0	0	1	0	0	0	0	0	0
0	3	616	0	0	0	0	1	4	0	0	0	0	0	4	0	13	1	0	0	0	1
0	0	0	287	0	0	0	0	0	0	0	0	2	0	0	1	0	17	1	742	16	0
18	0	1	0	2085	0	2	3	0	1	1	0	0	0	0	1	0	27	0	0	0	1
0	0	1	1	0	1148	0	0	0	0	0	0	2	0	0	2	0	2	0	0	15	1
50	0	0	0	6	0	5876	15	580	5	0	4	3	0	8	19	1	2	2	0	0	4
38	0	1	0	12	0	13	3284	0	0	0	0	1	1	11	0	11	0	0	0	1	4
13	0	0	0	2	0	2492	11	2152	3	1	11	17	0	1	14	0	9	2	0	4	4
19	0	0	0	4	0	3	5	0	831	0	1	0	4	1	2	0	5	0	0	0	102
9	0	1	0	2	0	11	1	0	1	989	0	0	0	1	1	0	17	1	0	2	1
1319	0	0	0	2	0	4	0	0	0	0	1727	0	0	2	0	0	3	1	0	0	1
12	7	4	0	5	0	13	1	38	1	1	0	2630	3	2	29	0	139	9	1	2	10
0	1	0	0	0	0	1	0	0	1	0	0	9	636	1	1	0	42	1	1	7	6
7	0	0	0	6	0	7	4	2	0	0	0	1	0	1067	2	2	9	0	1	6	1
74	0	1	0	14	0	36	40	7	3	2	4	5	1	10	6060	0	150	2	0	2	24
1	0	0	3	0	0	2	0	0	0	0	0	1	0	1	1	1531	3	0	0	1283	0
27	0	0	0	92	0	15	3	0	2	2	1	0	0	0	1	0	1862	0	0	0	2
13	0	0	0	2	0	9	4	7	2	1	1	12	1	1	7	0	77	966	0	5	9
0	0	2	116	0	0	0	0	0	0	0	0	2	5	0	2	0	16	2	596	13	3
1	121	3	0	0	1	0	4	2	1	0	0	19	8	3	4	0	31	4	6	3013	4
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	6	3	0	0	1
7	0	0	0	1	0	1	2	0	1	0	0	0	0	1	1	2	0	0	0	0	234

0.821369469165802

categorical_crossentropy

4143	0	1	0	2	0	0	6	0	0	2	19	0	0	2	1	0	4	0	0	0	1
0	1336	0	0	0	0	1	1	1	2	1	653	2	0	0	0	0	1	1	0	0	1
0	3	614	0	0	0	0	2	6	0	0	0	0	0	0	6	0	11	1	0	0	0
0	0	0	57	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1004	3	0
22	0	1	0	2094	0	5	7	0	1	1	0	0	0	2	0	0	6	0	0	0	1
0	0	1	1	0	1157	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	2
33	0	0	0	6	0	5795	15	687	4	6	6	2	0	7	5	1	2	2	0	0	4
28	0	1	0	5	0	8	3320	1	0	0	0	0	1	1	6	0	5	0	0	0	1
7	0	0	0	2	0	2204	10	2472	3	5	10	4	1	2	5	1	4	1	0	1	4
20	0	0	0	3	0	4	6	1	853	1	1	0	9	3	2	0	4	0	0	0	70
4	0	1	0	1	0	2	1	0	0	1022	0	0	0	1	0	0	2	1	0	1	1
848	0	0	1	3	0	16	3	3	0	1	2175	0	0	3	1	0	2	1	0	0	2
15	7	3	2	9	0	10	4	120	0	5	1	2588	5	6	18	1	83	7	0	1	23
0	2	0	0	2	0	1	0	0	2	0	0	6	684	1	0	0	5	0	2	0	2
5	0	0	0	3	0	2	4	0	0	0	0	1	0	1069	1	22	5	0	0	1	0
79	1	1	0	17	0	42	58	12	6	8	11	7	4	1	14	6118	0	46	0	1	12
0	0	0	1	0	0	2	0	0	0	1	0	0	0	3	1	2475	2	0	1	340	0
28	0	0	0	267	0	15	9	0	1	3	2	1	1	0	3	0	1671	1	0	0	5
16	0	0	0	2	0	9	13	12	3	2	2	13	1	1	6	0	48	985	0	4	12
0	0	1	23	0	0	0	0	0	0	0	0	1	4	0	1	1	2	0	723	0	1
3	271	1	0	1	0	0	3	15	1	0	0	35	22	7	4	0	12	9	17	2819	5
1	0	0	0	0	0	1	1	0	0	0	0	1	0	1	0	0	9	6	0	0	203
8	0	0	0	2	0	0	2	1	2	0	0	0	0	1	1	2	0	0	0	0	236

0.8487091064453125

kullback_leibler_divergence

4150	0	0	0	4	0	2	9	0	0	3	2	3	0	2	0	0	6	0	0	0	0
0	1333	0	0	1	0	6	1	3	2	3	618	24	0	2	3	0	4	0	0	0	1
0	3	622	0	0	0	0	1	3	0	0	0	0	0	0	3	0	10	1	0	0	0
0	0	0	311	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	744	8	0
17	0	1	0	2098	0	5	8	0	2	1	0	1	0	0	0	0	6	0	0	0	1
0	0	0	2	0	1152	0	0	0	0	0	0	5	0	0	0	1	1	0	0	11	0
24	0	0	0	5	0	5931	13	557	4	5	3	14	0	9	4	1	2	2	0	0	1
25	0	1	0	5	0	5	3322	0	0	0	1	1	1	1	8	0	6	0	0	1	1
4	0	0	0	3	0	2534	10	2116	4	3	5	40	0	3	5	1	3	1	0	2	2
19	0	0	0	3	0	3	5	0	898	1	1	0	10	1	2	0	3	0	0	0	31
4	0	1	0	1	0	3	1	0	0	1020	0	2	0	0	1	1	0	1	0	0	0
1586	0	0	0	1	5	0	31	5	1	0	4	1367	25	1	4	10	0	16	1	0	0
7	6	2	1	6	0	10	2	31	1	2	0	2768	2	2	13	0	45	4	0	2	5
0	1	0	0	1	0	1	0	0	2	0	0	9	684	1	0	0	6	0	1	0	1
4	0	0	0	1	4	0	2	3	1	1	0	2	0	1056	1	32	6	0	0	2	0
54	1	1	0	15	0	43	44	7	6	7	2	22	2	10	6138	0	72	1	1	1	8
0	0	0	0	2	0	0	2	0	0	0	0	1	0	1	1	2439	1	0	0	379	0
26	0	0	0	160	0	17	8	0	2	3	1	4	1	0	2	0	1780	1	0	0	2
13	0	0	0	2	0	9	8	10	3	1	1	25	0	3	10	1	57	925	0	8	8
0	0	0	109	0	0	0	0	0	0	0	0	1	5	0	1	2	3	0	635	1	0
1	78	1	2	0	0	0	4	3	1	0	0	134	23	10	2	0	10	4	10	2939	3
0	0	0	0	0	0	1	1	0	1	0	0	1	0	1	0	0	6	2	0	0	212
8	0	0	0	1	0	1	1	0	6	0	0	2	1	1	2	0	2	0	0	0	230

0.8395197987556458

poisson

4170	0	1	0	1	0	0	3	0	0	2	2	0	0	1	0	0	1	0	0	0	0
0	981	0	0	0	0	4	1	1	2	3	998	2	0	2	2	0	1	3	0	0	1
0	3	627	0	0	0	0	1	2	0	0	0	0	0	0	2	0	7	1	0	0	0
0	0	0	746	0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	300	15
23	0	1	0	2091	0	4	5	0	1	1	0	0	0	7	1	0	5	0	0	0	1
1	0	0	2	0	1146	0	0	0	0	0	0	1	0	0	0	1	1	1	0	1	18
51	0	0	0	6	0	5672	14	788	5	7	1	0	1	10	8	1	2	3	0	1	5
34	0	1	0	7	0	6	3313	0	1	0	0	0	1	1	8	0	4	0	0	0	1
18	0	0	0	2	0	2122	11	2531	4	6	6	6	2	3	6	1	3	4	0	6	5
23	0	0	0	4	0	3	3	0	879	0	1	1	24	3	1	0	2	1	0	0	32
5	0	1	0	1	0	2	3	0	1	1019	0	0	0	1	0	0	1	1	0	2	0
1712	0	0	1	2	0	14	3	0	0	4	1313	0	0	2	4	0	2	1	0	0	1
45	7	4	4	11	0	13	5	66	4	29	0	2541	13	16	28	2	82	12	0	8	18
0	0	0	0	1	0	1	0	0	2	2	0	2	694	1	0	0	3	0	0	0	1
4	0	0	0	2	0	1	3	0	0	0	0	0	0	10							

Adagrad

mean_squared_error

4128	0	1	0	8	0	10	4	0	1	2	14	1	0	5	2	0	4	1	0	0	0	0
1	1115	0	0	0	0	1	2	1	1	0	875	3	0	0	0	0	1	1	0	0	0	0
0	3	617	0	1	0	0	1	4	1	1	0	1	0	2	5	0	5	2	0	0	0	0
2	0	0	394	0	0	0	2	0	0	0	0	2	0	0	2	1	2	1	646	14	0	0
22	1	1	0	2092	0	5	2	0	2	1	0	1	0	0	2	0	11	0	0	0	0	0
13	0	0	1	0	1151	0	0	0	0	0	0	1	0	0	2	0	1	3	0	0	0	0
50	0	0	0	5	0	4791	10	1656	4	1	5	21	0	7	19	2	3	1	0	0	0	0
53	0	2	0	16	0	28	3239	2	3	0	0	3	0	4	20	0	3	3	0	1	0	0
17	0	0	0	2	1	1137	13	3490	6	3	11	29	1	5	11	1	4	1	0	5	0	0
17	0	0	0	3	0	4	6	1	929	1	2	2	4	0	2	1	5	0	0	0	0	0
10	0	1	0	4	0	13	1	0	2	994	0	1	0	1	3	0	6	0	0	1	0	0
1360	4	0	0	3	0	6	2	2	0	0	1660	12	0	3	1	0	4	1	0	1	0	0
31	8	2	3	10	0	18	4	112	1	4	0	2589	6	5	39	1	69	3	1	4	0	0
2	2	0	0	0	0	1	0	0	3	1	0	4	682	1	1	1	4	0	0	5	0	0
13	0	0	0	5	0	6	0	0	2	2	0	1	0	1066	6	10	4	0	0	0	0	0
102	0	1	0	23	0	35	31	10	13	6	3	9	0	10	6156	0	33	1	0	2	0	0
1	0	0	2	0	0	2	0	0	0	0	0	0	0	3	3	2331	0	0	0	484	0	0
33	0	0	0	301	0	19	6	0	5	3	1	2	0	1	7	0	1627	3	0	0	0	0
14	0	0	0	2	0	5	22	13	6	4	4	11	0	1	23	0	65	960	3	0	0	0
1	0	1	108	1	0	0	0	1	0	0	0	1	7	1	1	3	4	1	615	12	0	0
8	86	1	3	2	0	0	7	0	1	1	0	24	12	11	4	0	5	3	4	3053	0	0
4	0	0	0	1	38	0	9	2	3	1	0	6	0	1	30	0	46	84	0	0	0	0
44	0	0	0	4	0	4	33	2	78	0	0	1	2	2	22	1	60	2	0	0	0	0

0.831015408039093

mean_squared_logarithmic_error

4123	0	1	0	8	0	10	4	0	1	2	17	2	0	4	2	0	3	2	0	0	0	2
0	1248	0	0	0	0	1	1	3	1	1	742	2	0	0	0	0	1	1	0	0	0	0
0	3	620	1	1	0	0	1	2	1	0	0	0	0	1	5	0	5	2	0	0	0	1
2	0	0	359	0	0	0	5	0	0	0	0	1	0	0	1	0	4	1	681	12	0	0
20	1	1	0	2086	0	5	2	0	2	1	0	4	0	0	1	0	17	0	0	0	0	0
11	0	0	1	0	1145	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	12	0
46	0	1	0	4	0	4644	14	1808	7	3	4	7	0	9	18	2	3	2	0	0	1	2
40	0	1	0	14	0	22	3263	1	3	0	0	2	0	4	16	0	4	4	0	1	1	1
12	0	0	0	2	0	1095	11	3539	8	4	12	15	1	5	14	1	6	3	0	3	1	5
11	0	0	0	3	0	2	3	1	904	0	2	2	1	0	1	1	2	0	0	0	0	44
8	0	1	0	4	0	11	1	2	1	995	0	0	1	1	2	0	7	1	0	2	0	0
1227	1	0	0	3	0	8	2	5	0	1	1788	8	1	3	2	0	4	4	0	1	0	1
18	9	2	3	7	0	19	6	147	3	4	0	2561	5	4	35	2	66	5	1	6	5	2
1	2	0	0	0	0	1	0	0	2	2	0	3	683	1	1	0	3	0	0	5	0	3
9	0	0	0	5	0	5	0	0	2	1	0	2	0	1072	3	12	3	0	0	0	0	1
91	0	1	0	18	0	37	46	12	13	7	3	7	0	14	6146	0	37	1	0	1	0	1
1	0	0	3	0	0	2	1	0	0	0	0	1	0	3	3	2350	0	0	0	462	0	0
32	0	0	0	214	0	19	8	1	2	4	0	1	0	0	6	0	1717	1	0	0	2	1
12	0	0	0	2	0	6	15	7	4	3	4	9	0	1	13	0	40	973	0	1	34	9
1	0	1	65	1	0	0	0	0	0	0	0	1	6	0	2	3	5	2	660	10	0	0
6	100	1	2	1	0	0	9	0	1	3	0	6	14	11	5	0	5	5	4	3051	0	1
2	0	0	0	0	0	0	1	1	1	0	0	1	0	1	2	0	3	3	0	0	209	1
14	0	0	0	1	0	3	7	0	19	0	0	0	0	1	7	0	6	0	0	0	0	197

0.8434580564498901

squared_hinge

4141	1	0	0	2	0	12	3	0	0	0	18	1	0	0	3	0	0	0	0	0	0	0
2	1110	0	0	0	0	0	0	1	0	0	886	2	0	0	0	0	0	0	0	0	0	0
23	5	0	0	4	20	0	238	14	0	0	6	16	0	0	315	0	0	0	0	2	0	0
65	0	0	0	2	45	0	62	46	0	0	0	133	0	0	304	277	0	0	0	132	0	0
41	1	0	0	2072	0	4	8	1	0	0	0	4	0	0	8	0	0	0	0	1	0	0
8	0	0	0	0	1158	0	2	0	0	0	0	1	0	0	3	0	0	0	0	0	0	0
42	0	0	0	4	0	4334	7	2154	0	0	5	15	0	0	12	2	0	0	0	0	0	0
54	0	0	0	23	0	19	3238	10	0	0	0	2	0	0	29	1	0	0	0	1	0	0
16	0	0	0	4	2	903	11	3745	0	0	12	20	0	0	19	1	0	0	0	4	0	0
28	0	0	0	21	3	4	514	6	0	0	2	9	0	0	382	2	0	0	0	6	0	0
129	0	0	0	31	4	12	403	4	0	0	0	4	12	0	0	415	6	0	0	0	29	0
1304	12	0	0	2	0	8	0	4	0	0	1721	4	0	0	1	0	0	0	0	3	0	0
28	9	0	0	47	1	18	11	120	0	0	1	2592	0	0	72	1	0	0	0	10	0	0
11	3	0	0	4	86	1	82	9	0	0	0	80	0	0	120	29	0	0	0	282	0	0
88	1	0	0	15	0	13	27	6	0	0	0	8	0	0	96	769	0	0	0	92	0	0
85	1	0	0	31	0	33	43	16	0	0	4	8	0	0	6209	2	0	0	0	3	0	0
0	0	0	0	0	0	2	1	0	0	0	0	4	0	0	4	2211	0	0	0	604	0	0
113	1	0	0	1485	0	21	93	3	0	0	0	4	0	0	285	0	0	0	0	3	0	0
52	1	0	0	3	255	6	272	18	0	0	4	22	0	0	460	3	0	0	0	37	0	0
46	0	0	0	3	22	0	25	47	0	0	0	127	0	0	181	217	0	0	0	89	0	0
9	68	0	0	2	2	0	9	0	0	0	0	18	0	0	9	3	0	0	0	3105	0	0
4	0	0	0	3	78	0	21	1	0	0	0	4	0	0	111	1	0	0	0	2	0	0
54	0	0	0	13	0	3	104	3	0	0	0	2	0	0	73	2	0	0	0	1	0	0

0.6779931783676147

categorical_hinge

4128	0	0	0	8	0	15	4	0	1	1	18	1	0	2	2	0	0	1	0	0	0	0
0	1198	0	0	0	0	0	2	1	1	0	792	3	0	1	1	0	1	1	0	0	0	0
0	3	625	1	1	0	0	1	0	1	0	0	0	0	0	7	0	3	1	0	0	0	0
8	0	0	680	0	0	0	4	2	0	0	0	2	2	0	3	3	7	3	330	22	0	0
20	1	1	0	2086	0	9	3	0	1	1	0	2	0	1	2	0	13	0	0	0	0	0
0	0	1	1	0	1035	0	0	0	0	1	0	39	3	0	4	17	2	58	2	9	0	0
51	0	1	0	5	0	5101	19	1312	7	3	5	15	1	10	35	1	3	5	0	1	0	0
39	0	1	0	17	0	25	3265	1	3	0	0	2	0	3	19	0	2	0	0	0	0	0
21	0	0	0	2	0	1409	14	3183	8	6	11	27	3	5	30	4	4	5	0	5	0	0
14	0	0	0	3	0	5	6	0	935	1	2	2	3	1	2	0	3	0	0	0	0	0
12	0	1	0	4	0	12																

categorical_crossentropy

4113	0	2	0	4	0	14	7	0	2	4	17	1	0	5	4	0	2	2	0	0	0	4
3	1070	0	0	0	1	4	4	4	1	1	901	3	0	0	5	0	1	3	0	0	0	0
0	1	613	1	0	0	0	1	5	1	1	0	0	0	3	6	0	6	3	0	0	0	2
1	0	0	334	0	0	0	4	0	0	0	0	1	0	0	1	0	2	1	715	7	0	0
27	1	1	0	2063	0	8	5	0	3	1	0	3	0	3	2	0	23	0	0	0	0	0
16	0	0	0	0	1140	0	0	0	0	0	0	1	0	0	1	0	1	2	0	0	11	0
39	0	1	0	3	0	5507	7	977	4	0	4	4	0	5	15	2	2	2	0	0	0	3
43	0	2	0	11	0	23	3253	1	2	0	0	0	4	24	0	4	4	0	0	0	6	6
13	0	0	0	2	2	1886	9	2767	6	2	5	14	1	4	12	1	5	2	0	2	0	4
11	0	0	0	3	0	5	2	0	890	1	2	1	3	1	1	0	3	0	0	0	0	54
10	0	1	0	4	0	13	1	0	2	989	0	0	2	4	0	8	1	0	0	2	0	0
1454	11	0	0	2	0	15	3	6	0	1	1539	5	1	3	11	0	3	2	0	2	0	1
31	8	2	3	6	0	35	6	143	2	5	1	2507	7	4	41	2	89	6	1	3	3	5
2	0	0	0	0	0	1	0	0	2	0	0	3	678	1	1	1	6	1	1	8	0	2
9	0	0	0	4	0	6	0	0	2	1	0	2	0	1060	4	23	3	0	0	0	0	1
93	0	1	0	18	0	41	42	8	10	6	3	6	0	10	6171	1	22	1	0	1	0	1
1	0	0	3	0	0	2	0	0	0	1	0	2	0	4	3	2327	1	1	0	481	0	0
37	0	0	0	220	0	20	9	0	2	3	0	2	0	0	9	0	1702	2	0	0	1	1
15	0	0	0	2	0	8	12	6	4	2	4	7	0	2	17	0	55	977	0	1	14	7
2	0	1	69	0	0	0	0	0	0	0	0	1	5	0	3	3	5	2	658	8	0	0
11	83	1	3	2	0	0	7	1	1	0	0	14	9	10	8	1	7	6	5	3055	0	1
3	0	0	0	0	0	0	1	1	1	0	0	2	0	1	1	0	6	11	0	0	197	1
13	0	0	0	1	0	3	3	0	16	0	0	0	1	1	8	0	6	0	0	0	0	203

0.833564817905426

kullback_leibler_divergence

4101	0	2	0	5	0	13	8	0	1	3	29	0	0	5	4	0	4	2	0	0	0	4
1	1173	0	0	0	1	1	2	2	1	1	811	4	0	0	2	0	1	1	0	0	0	0
0	2	615	1	0	0	0	1	4	1	0	0	0	0	1	10	0	4	1	0	0	0	2
2	0	0	351	0	0	0	4	0	0	0	0	1	0	0	1	0	3	1	692	11	0	0
27	1	1	0	2059	0	6	5	0	2	0	0	4	0	4	4	0	27	0	0	0	0	0
16	0	0	0	0	1142	0	0	0	0	0	0	1	0	0	1	0	1	2	0	0	9	0
38	0	1	0	3	0	5078	10	1378	4	1	5	16	0	8	23	2	3	2	0	0	0	3
42	0	2	0	11	0	18	3266	1	3	0	0	0	0	3	24	0	2	3	0	1	0	1
12	0	0	0	2	2	1465	10	3175	6	3	8	18	1	5	19	1	4	0	0	3	0	3
11	0	0	0	3	0	4	4	0	904	1	2	1	2	1	0	3	0	0	0	0	0	40
12	0	1	0	4	0	11	3	1	1	984	0	0	2	3	8	0	5	0	0	2	0	0
1110	20	0	0	3	0	11	3	7	0	1	1876	9	1	3	10	0	2	1	0	1	0	1
26	9	3	2	9	1	24	8	130	2	5	3	2517	7	4	54	1	86	5	1	6	3	4
2	2	0	0	0	0	1	0	0	2	0	0	4	676	1	2	0	5	1	0	9	0	2
8	0	0	0	2	0	6	1	0	1	1	0	1	0	1075	5	11	3	0	0	1	0	0
84	1	1	0	16	0	35	38	8	10	5	3	6	0	13	6201	0	11	1	0	2	0	0
0	0	0	3	0	0	2	0	0	0	1	0	1	1	3	2	2222	2	0	1	588	0	0
40	1	0	0	262	0	17	11	0	3	3	1	2	0	0	14	0	1650	1	0	0	1	2
14	1	0	0	2	1	8	12	7	4	1	4	10	0	2	21	0	51	963	0	1	26	5
2	0	1	94	1	0	0	0	1	0	0	0	1	5	2	3	1	3	2	629	12	0	0
9	105	1	0	1	1	0	7	0	0	0	0	15	9	11	8	1	6	3	5	3042	0	1
1	0	0	0	0	0	0	2	1	1	0	0	1	0	1	2	0	6	13	0	0	196	1
12	0	0	0	1	0	2	5	1	18	0	0	0	0	1	12	1	5	0	0	0	0	197

0.8388729095458984

poisson

4113	0	2	0	5	0	9	6	0	1	4	21	1	1	5	3	0	2	2	0	0	1	5
2	1131	0	0	0	1	1	2	2	1	2	855	2	0	0	0	0	1	1	0	0	0	0
0	3	622	1	0	0	0	1	2	1	0	0	0	0	1	3	0	5	2	0	0	0	2
1	0	0	351	0	0	0	4	0	0	0	0	1	0	0	1	1	2	1	697	7	0	0
25	1	1	0	2076	0	6	6	0	4	2	0	3	0	3	2	0	10	0	0	0	1	0
11	0	0	0	0	1144	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	12	2
40	0	1	0	4	0	4622	13	1831	4	5	4	13	0	8	16	1	4	3	0	0	1	5
35	0	3	0	10	0	19	3280	0	1	1	0	1	0	3	15	0	2	3	0	0	1	3
16	0	0	0	2	1	1153	11	3476	4	6	8	23	1	5	11	1	5	3	0	4	1	6
10	0	0	0	3	0	2	3	1	894	1	3	1	4	0	1	0	2	0	0	0	0	52
6	0	1	0	3	0	6	1	2	2	1009	0	0	0	1	1	0	3	0	0	2	0	0
1188	4	0	0	2	0	11	1	4	0	1	1821	8	0	3	9	0	3	2	0	1	0	1
37	9	2	1	8	1	13	6	115	2	5	1	2577	7	3	37	0	61	6	1	3	3	12
2	2	0	0	0	0	1	0	0	2	0	0	3	683	1	2	1	4	0	0	4	0	2
8	0	0	0	3	0	4	0	0	1	2	0	2	0	1081	4	7	2	0	0	0	0	1
87	0	1	0	16	0	36	36	7	13	8	4	6	0	13	6178	1	25	0	0	1	0	3
1	0	0	3	0	0	2	0	1	0	1	0	2	0	3	2	2367	0	0	1	443	0	0
39	0	0	0	240	0	18	11	1	6	4	1	2	0	1	7	0	1673	1	0	0	2	2
14	0	0	0	2	0	4	7	7	2	5	4	11	0	1	17	0	41	982	0	1	25	10
2	0	1	86	0	0	0	0	0	0	0	0	1	5	1	1	3	4	2	642	9	0	0
11	124	1	3	2	0	0	7	0	1	2	0	9	10	12	5	0	5	4	5	3023	0	1
2	0	0	0	0	0	0	1	1	1	0	0	1	0	1	1	0	2	8	0	0	206	1
10	0	0	0	2	0	2	2	1	18	0	0	0	0	1	1	5	0	2	0	0	0	211

0.8402047157287598

cosine_proximity

4114	0	2	0	8	0	10	5	0	2	3	22	1	0	4	3	0	2	2	0	0	0	3
0	1243	0	0	0	0	0	0	1	1	0	752	2	0	0	0	0	1	1	0	0	0	0
0	3	621	1	0	0	0	1	4	1	0	0	0	0	2	3	0	5	2	0	0	0	0
1	0	0	298	0	0	0	2	0	0	0	0	1	0	0	1	0	2	1	749	11	0	0
19	1	1	0	2092	0	4	2	0	2	1	0	3	0	0	1	0	14	0	0	0	0	0
13	0	0	1	0	1143	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	11	1
43	0	1	0	5	0	4548	11	1907	5	1	5	9	0	9	19	1	5	2	0	0	1	3
43	0	1	0	18	0	20	3257	0	2	0	0	3	0	3	18	0	4	4	0	1	0	3
12	0	0	0	2	1	1002	9	3637	7	4	14	14	2	4	13	1	7	1	0	3	1	3
11	0	0	0	3	0	3	2	1	906	0	2	1	6	1	1	0	2	0	0	0	0	38
11	0	1	0	4	0	11	1	2	1	987	0	0	1	1	3	0	9	1	0	2	0	

mean_squared_logarithmic_error

3980	10	1	0	13	0	28	14	0	6	3	79	2	0	6	26	0	9	3	0	1	0	0
88	1004	0	0	1	0	5	3	0	0	1	884	5	0	2	3	0	2	0	0	3	0	0
5	4	389	7	7	0	5	70	4	7	20	2	10	3	14	58	0	29	3	2	4	0	0
3	1	9	676	0	6	8	6	64	0	1	0	20	3	1	11	18	1	12	181	45	0	0
58	1	0	0	1957	0	6	7	0	1	2	0	2	0	10	13	0	83	0	0	0	0	0
0	0	1	9	0	972	0	3	1	0	0	0	7	0	0	7	39	1	44	13	75	0	0
16	2	0	1	4	0	5968	14	520	3	1	10	17	1	3	10	3	2	0	0	0	0	0
75	0	1	1	14	1	40	3165	5	4	8	0	1	0	6	42	0	12	2	0	0	0	0
8	3	2	1	2	0	3075	26	1521	9	2	17	36	2	4	15	3	2	2	4	3	0	0
15	0	1	0	6	3	6	53	1	864	5	2	0	2	0	6	0	13	0	0	0	0	0
54	0	1	2	7	0	15	44	1	5	799	1	0	2	6	44	2	49	2	0	3	0	0
1332	265	0	0	8	1	16	10	1	1	1	1390	13	0	2	9	0	1	3	0	6	0	0
27	20	34	6	60	0	155	16	240	12	10	5	2120	21	3	50	0	61	10	27	32	0	1
1	4	4	8	0	0	2	7	2	49	4	0	5	564	1	2	4	10	3	2	35	0	0
14	1	0	0	3	0	7	7	1	0	6	1	0	0	1006	16	44	3	0	0	6	0	0
111	4	2	1	25	3	56	81	11	10	39	5	10	2	13	6000	4	51	1	0	6	0	0
0	0	5	6	0	0	7	6	4	0	1	0	8	3	23	5	1882	1	0	4	871	0	0
74	0	1	0	319	0	20	28	0	7	6	3	8	0	5	56	0	1479	1	0	1	0	0
10	1	0	3	2	3	7	93	13	7	24	6	6	0	1	25	1	75	852	0	4	0	0
2	1	11	413	0	2	5	10	91	1	1	0	21	11	2	3	13	2	2	133	33	0	0
25	67	8	3	0	11	0	21	9	2	5	3	22	26	27	17	100	6	18	19	2836	0	0
4	0	6	1	0	1	0	25	0	2	0	0	3	0	1	9	0	42	126	3	2	0	0
63	0	0	1	1	0	5	46	4	94	4	0	0	1	1	13	0	22	0	0	0	0	0

0.7525922060012817

squared_hinge

4122	0	0	0	9	0	19	10	5	2	1	0	0	0	0	10	0	2	1	0	0	0	0
1956	0	0	0	2	0	0	2	9	1	1	0	6	0	0	4	0	1	0	0	19	0	0
31	0	0	1	25	2	0	136	4	45	45	0	17	0	0	138	0	188	3	0	8	0	0
3	0	0	29	0	45	0	6	158	79	43	0	316	0	0	16	0	52	18	0	301	0	0
180	0	0	1	1787	0	8	25	3	0	1	0	8	0	0	37	0	88	0	0	2	0	0
2	0	0	1	0	985	0	1	0	0	1	0	20	0	0	14	0	2	30	0	116	0	0
31	0	0	0	0	0	3213	3	3298	0	3	0	16	0	0	6	0	1	0	0	4	0	0
124	0	0	0	11	0	14	3151	31	0	1	0	2	0	0	37	0	4	0	0	2	0	0
23	0	0	0	2	4	832	4	3812	1	7	0	36	0	0	7	0	4	0	0	5	0	0
89	0	0	0	6	1	4	239	18	348	20	0	4	0	0	78	0	161	7	0	2	0	0
110	0	0	0	11	0	9	66	6	4	639	0	1	0	0	61	0	88	21	0	21	0	0
3016	0	0	0	4	0	7	4	9	2	1	0	5	0	0	4	0	2	3	0	2	0	0
42	0	0	16	52	12	15	6	146	29	68	0	2227	0	0	35	0	229	3	0	30	0	0
10	0	0	3	0	0	1	8	6	164	51	0	31	0	0	18	0	123	63	0	229	0	0
97	0	0	0	27	0	5	17	4	0	32	0	3	0	0	38	0	11	0	0	881	0	0
127	0	0	1	37	7	19	62	37	3	84	0	11	0	0	5945	0	89	3	0	10	0	0
0	0	0	0	0	0	1	1	5	0	4	0	9	0	0	5	0	2	0	0	2799	0	0
175	0	0	1	416	1	20	33	5	3	12	0	2	0	0	62	0	1275	2	0	1	0	0
36	0	0	0	2	8	2	52	21	20	26	0	9	0	0	49	0	157	748	0	3	0	0
0	0	0	14	0	43	0	2	178	38	8	0	220	0	0	8	0	24	4	0	218	0	0
63	0	0	1	3	10	0	9	2	14	31	0	16	0	0	18	0	5	4	0	3049	0	0
3	0	0	1	7	5	0	8	0	23	0	0	4	0	0	18	0	47	106	0	3	0	0
90	0	0	0	2	0	4	40	3	21	7	0	0	0	0	23	0	59	6	0	0	0	0

0.596093359375

categorical_hinge

4056	0	0	0	5	0	31	2	0	0	1	83	0	0	0	2	0	0	1	0	0	0	0
0	996	0	0	0	0	2	0	0	0	0	1003	0	0	0	0	0	0	0	0	0	0	0
0	3	594	0	0	0	0	1	14	0	0	0	0	0	0	26	0	3	2	0	0	0	0
1	0	2	138	0	0	0	1	0	0	0	0	2	1	0	6	0	1	1	910	3	0	0
19	1	1	0	2052	0	21	7	0	1	0	0	0	0	1	10	0	27	0	0	0	0	0
1	0	1	0	0	1162	0	0	1	1	0	0	0	0	0	0	0	1	5	0	0	0	0
24	0	0	0	2	0	5829	11	674	2	0	10	6	0	2	12	1	1	1	0	0	0	0
35	0	1	0	11	0	21	3281	1	0	0	1	2	1	1	19	0	2	1	0	0	0	0
5	0	0	0	2	0	2633	7	2034	3	0	16	16	0	1	14	0	2	3	0	1	0	0
18	0	0	0	3	0	7	6	1	925	0	3	1	3	0	4	0	5	1	0	0	0	0
9	0	1	0	4	2	14	13	2	1	946	1	1	1	1	20	0	17	2	0	2	0	0
356	0	0	0	1	0	15	0	0	0	0	2684	0	0	2	1	0	0	0	0	0	0	0
7	8	2	3	5	0	39	3	115	0	1	5	2619	1	3	51	0	43	4	0	1	0	0
0	7	0	0	0	0	1	0	2	3	0	0	9	668	1	3	0	8	3	0	2	0	0
8	0	1	0	5	0	18	4	2	1	0	0	2	0	1061	6	1	5	1	0	0	0	0
59	0	1	0	13	0	45	29	7	4	0	17	5	0	6	6236	0	12	1	0	0	0	0
0	0	0	3	0	3	2	3	1	0	0	0	57	0	2	8	2236	1	9	1	500	0	0
26	0	0	0	147	0	28	11	0	2	1	4	1	0	0	11	0	1774	3	0	0	0	0
10	0	0	0	2	1	11	20	19	3	0	5	14	1	1	30	0	23	992	0	1	0	0
1	0	1	64	0	0	0	0	0	0	0	0	2	5	0	10	0	5	2	664	3	0	0
3	381	3	4	0	2	0	7	65	2	0	6	100	17	4	44	0	11	14	10	2552	0	0
1	0	0	0	0	70	2	4	5	0	0	0	5	0	1	21	0	5	111	0	0	0	0
22	0	0	0	3	0	4	26	2	93	0	0	3	1	2	36	0	60	3	0	0	0	0

0.82759823173523

logcosh

3820	29	0	0	15	0	33	18	1	7	5	131	3	0	9	91	0	11	4	0	4	0	0
139	834	0	0	2	0	10	3	0	0	0	950	4	0	3	30	0	2	1	0	23	0	0
4	6	0	0	19	1	6	146	11	56	5	1	40	0	18	203	5	96	0	1	25	0	0
9	4	1	33	5	5	17	7	231	28	7	1	292	0	7	106	70	14	21	24	184	0	0
85	1	0	0	1651	0	3	9	0	0	1	2	6	0	8	104	1	269	0	0	0	0	0
1	0	0	1	1	834	0	3	0	0	0	1	4	0	0	140	50	3	56	0	78	0	0
18	3	0	0	6	0	4607	11	1804	1	1	19	50	0	6	43	1	5	0	0	0	0	0
166	0	0	0	11	1	40	2888	5	5	9	0	1	4	0	230	2	14	1	0	0	0	0
17	5	0	1	5	2	2068	25	2461	5	0	22	43	0	14	52	9	4	1	0	3	0	0
75	1	0	1	11	0	7	206	5	481	6	2	0	0	0	157	0	24	1				

kullback_leibler_divergence

4164	0	0	0	1	0	0	6	0	0	2	2	0	0	1	0	0	1	4	0	0	0	0
0	909	0	0	0	0	2	2	1	2	0	1073	2	0	0	1	0	1	6	0	0	1	1
2	3	573	1	0	0	0	4	10	0	0	0	0	0	2	19	0	17	8	3	0	0	1
0	0	0	37	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1024	2	0	0	0
44	0	1	0	2006	0	7	22	0	4	1	0	1	0	0	6	5	0	40	0	0	1	2
2	0	0	1	0	1149	0	0	0	0	0	0	0	0	0	0	0	1	2	1	0	16	0
52	0	0	0	3	0	4923	17	1529	7	1	2	1	0	7	15	1	5	8	0	0	1	3
31	0	1	0	2	0	3	3328	0	0	0	0	0	0	6	0	3	2	0	0	1	0	4
20	0	0	0	2	0	1515	13	3137	4	1	7	6	1	2	9	1	4	8	0	2	1	4
25	0	0	0	3	0	1	6	1	911	0	1	0	5	1	2	0	3	1	0	0	0	17
10	0	1	0	1	0	5	9	1	1	995	0	1	0	1	1	0	7	1	0	2	0	1
1848	0	0	0	2	0	11	5	3	0	1	1176	0	0	2	3	0	2	4	0	0	0	2
54	7	3	1	3	0	11	11	80	7	1	0	2554	6	4	26	0	91	30	2	1	7	11
2	1	0	0	0	0	1	0	0	2	0	0	4	684	1	0	0	6	3	2	0	0	1
12	0	0	0	1	0	2	4	0	1	0	0	0	0	0	1078	1	9	5	1	0	0	0
129	0	1	0	9	0	38	63	6	8	3	3	5	1	9	6105	0	43	6	0	1	1	4
2	0	0	4	0	0	2	0	0	0	0	0	1	0	2	1	2427	0	3	2	382	0	0
34	0	0	0	37	0	16	12	0	1	3	0	1	1	0	5	0	1893	2	0	0	1	2
17	0	0	0	1	0	4	11	6	4	1	1	7	1	1	3	0	20	1047	0	0	6	3
0	0	0	10	0	0	0	0	0	0	0	0	1	4	0	1	0	2	2	737	0	0	0
11	91	1	1	0	0	0	11	7	2	0	0	38	35	10	4	0	15	63	23	2911	1	1
1	0	0	0	0	0	0	2	1	1	2	0	0	1	0	1	0	6	0	0	211	0	0
9	0	0	0	1	0	0	5	1	7	0	0	0	1	1	2	0	2	0	0	0	0	226

0.8215407133102417

poisson

4111	0	1	0	4	0	11	9	0	0	3	27	0	0	5	3	0	3	2	0	0	0	2
2	1196	0	0	0	1	5	2	1	1	1	784	1	0	0	3	0	1	2	0	1	0	0
1	3	594	1	0	0	0	1	6	0	1	0	0	0	3	19	0	6	4	3	0	0	1
0	0	0	143	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	915	4	0	0
22	1	1	0	2073	0	6	6	0	1	2	0	1	0	4	5	0	18	0	0	0	0	0
0	0	0	1	0	1155	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	13	0
34	0	1	0	4	0	5597	6	882	2	0	3	4	0	5	28	2	2	1	0	0	1	3
28	0	2	0	10	0	22	3289	0	0	0	0	1	3	14	0	2	4	0	1	1	0	0
18	0	0	0	2	0	2018	10	2636	2	5	6	8	1	4	18	1	3	0	0	3	1	1
15	0	0	0	3	0	3	3	1	909	1	1	0	5	2	3	0	3	0	0	0	0	28
6	0	1	0	3	0	10	2	0	1	1000	0	0	1	1	3	0	6	1	0	2	0	0
1406	7	0	0	2	0	18	2	7	0	1	1590	3	0	3	13	0	2	4	0	0	0	1
30	8	3	3	8	0	30	6	128	3	3	1	2473	6	7	87	1	87	8	1	6	6	5
2	0	0	0	0	0	1	0	0	2	1	0	2	686	1	2	0	4	0	0	5	0	1
5	0	0	0	2	0	3	2	0	1	0	0	2	0	1080	4	12	3	0	0	0	0	1
76	0	1	0	17	0	36	27	4	9	6	3	3	1	8	6228	1	13	1	0	1	0	0
1	0	0	4	0	0	2	0	1	0	0	0	0	0	2	2	2309	0	1	1	503	0	0
29	0	0	0	257	0	20	11	0	4	4	1	2	0	1	8	0	1668	2	0	0	1	0
10	0	0	0	2	0	7	9	8	2	1	3	6	0	1	20	1	45	993	0	3	17	5
0	0	1	34	0	0	0	0	0	0	0	0	1	4	0	1	1	3	2	707	3	0	0
4	65	1	1	2	0	0	8	2	0	0	0	8	17	11	10	0	5	8	8	3075	0	0
2	0	0	0	1	0	0	0	1	0	0	0	0	0	1	4	0	4	18	0	0	194	0
12	0	0	0	2	0	3	4	0	29	0	0	0	1	1	10	0	5	0	0	0	0	188

0.8351058959960938

cosine_proximity

4176	0	0	0	1	0	1	1	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
4	600	0	0	0	1	3	3	2	2	2	1363	8	0	2	4	0	4	1	0	1	1	0
14	3	541	1	0	0	0	6	16	0	0	0	0	0	6	34	0	15	5	0	0	0	2
2	0	0	371	0	0	0	1	0	0	0	0	1	0	0	1	3	7	1	664	15	0	0
25	0	1	0	2090	0	3	6	0	1	1	0	0	0	0	0	0	12	0	0	0	0	1
0	0	0	1	0	1152	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	17	0
62	0	0	0	5	0	5200	19	1232	6	1	0	8	0	9	21	1	4	1	0	0	2	4
39	0	1	0	8	0	13	3297	0	0	0	0	1	1	11	0	4	1	0	0	1	0	0
32	0	0	0	2	0	1539	12	3074	5	3	2	21	3	5	9	4	6	1	0	6	4	9
22	0	0	0	4	0	2	4	0	911	0	0	0	1	1	2	0	1	0	0	0	0	29
13	0	1	0	3	0	9	4	2	1	988	0	0	1	1	3	0	8	1	0	2	0	0
2810	0	0	0	2	0	7	0	0	0	0	233	1	0	1	3	0	1	1	0	0	0	0
33	5	2	3	7	1	12	3	48	7	1	1	2630	3	8	28	5	80	5	0	8	6	14
2	0	0	0	0	1	1	0	0	3	0	0	5	675	1	1	2	11	0	1	3	0	1
15	0	0	0	5	0	4	4	0	1	0	0	0	0	1049	2	30	4	0	0	1	0	0
113	0	0	0	14	0	35	50	5	8	1	0	6	0	8	6148	1	42	1	0	1	0	2
1	0	0	1	0	0	2	0	0	0	0	0	0	0	1	1	2556	0	0	0	264	0	0
33	0	0	0	183	0	17	11	0	2	3	0	0	0	5	0	1751	0	0	1	1	1	1
22	0	0	0	2	0	6	16	9	3	1	1	9	0	2	17	0	48	945	0	8	34	10
3	0	1	142	0	0	0	0	0	0	0	0	1	5	0	1	3	12	2	567	19	0	1
7	0	1	0	1	5	0	7	1	1	0	0	16	15	6	3	2	10	4	5	3140	0	1
4	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0	4	2	0	0	210	1
13	0	0	0	1	0	0	3	1	6	0	0	0	1	1	3	0	3	0	0	1	0	222

0.8090789914131165

Adam

mean_squared_error

4147	0	0	0	7	0	6	5	0	0	2	5	0	0	1	0	0	4	4	0	0	0	0
0	1046	0	0	0	0	3	1	3	2	1	923	9	0	0	0	0	2	8	0	0	1	2
0	3	553	1	0	1	0	5	24	0	1	0	0	1	0	0	12	0	20	15	0	0	6
0	0	0	295	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	764	3	0	0
14	0	1	0	2096	0	3	6	0	2	1	0	0	0	0	0	15	0	0	0	0	1	1
0	0	0	1	0	1150	0	0	1	0	0	0	2	0	0	0	0	1	1	0	0	16	0
37	0	0	0	6	0	5483	9	1015	5	0	1	2	0	4	2	1	3	0	0	1	5	0
29	0	1	0	7	0	16	3311	0	0	0	0	0	1	0	6	0	3	2	0	0	1	0
8	0	0	0	2	0	1907	11	2743	4	1	9	31	0	1	3	0	2	7	0	0	1	7
17	0	0	0	3	0	2	4	0	876	0	1	0	0	1	0	0	1	0	0	0	0	72
4	0	0	0	3	0	7	5	2	1	1003	0	0	0	1	2	0	5	1	0	2	0	

squared_hinge

4132	1	1	0	6	0	12	6	0	0	3	0	0	2	3	2	0	9	4	0	0	0	0
63	1870	0	0	1	1	6	4	13	2	5	0	7	0	1	10	0	15	3	0	0	0	0
0	1	616	0	0	0	0	2	5	0	0	0	0	1	1	5	0	10	2	0	0	0	0
1	0	0	1013	0	0	0	2	0	0	0	0	4	6	1	1	0	10	1	0	27	0	0
16	0	1	0	2062	0	5	6	1	4	2	0	0	0	2	0	0	41	0	0	0	0	0
0	0	0	0	0	1163	0	0	0	0	0	0	1	0	0	0	0	1	7	0	0	0	0
30	0	0	0	4	0	4881	15	1623	5	5	0	0	0	6	2	1	2	1	0	0	0	0
29	0	1	0	7	0	10	3314	1	0	0	0	0	1	1	9	0	3	1	0	0	0	0
11	0	0	0	2	0	1169	10	3521	4	5	0	4	0	1	4	1	1	1	0	3	0	0
17	0	0	0	3	0	4	5	0	931	1	0	0	13	1	0	0	2	0	0	0	0	0
4	0	0	0	1	0	5	1	0	0	1019	0	0	0	1	0	2	1	0	2	0	0	0
2924	56	0	0	2	0	28	3	8	0	3	0	1	0	3	9	0	21	1	0	0	0	0
7	6	3	1	3	0	9	4	86	7	12	0	2587	12	7	21	2	128	10	0	5	0	0
0	0	0	0	0	0	1	0	0	3	0	0	1	693	1	0	0	7	0	0	1	0	0
3	0	0	0	4	0	6	4	1	1	1	0	3	0	1068	1	15	5	1	0	2	0	0
60	0	1	0	13	0	36	44	21	11	8	0	7	5	13	6116	0	95	3	0	2	0	0
0	0	0	1	0	0	2	0	0	0	0	0	0	0	1	2492	1	0	0	328	0	0	0
24	0	0	0	68	0	19	6	1	2	3	0	1	1	0	3	0	1877	3	0	0	0	0
13	0	0	0	2	0	8	8	9	6	2	0	11	3	1	7	0	43	1012	0	8	0	0
0	0	2	660	0	0	0	0	0	0	0	0	6	21	0	3	3	13	2	0	47	0	0
1	78	2	1	0	0	0	3	5	1	0	0	15	32	6	2	0	13	6	0	3060	0	0
0	0	0	0	0	9	1	1	0	2	0	0	5	0	1	2	0	21	183	0	0	0	0
14	0	0	0	1	0	3	10	2	146	0	0	2	8	2	6	0	58	1	0	2	0	0

0.8262209892272949

categorical_hinge

4157	0	0	0	6	0	6	2	0	0	1	6	0	0	1	0	0	1	0	0	0	0	1
0	1236	0	0	0	0	0	0	0	2	0	763	0	0	0	0	0	0	0	0	0	0	0
0	3	586	1	0	0	0	15	5	0	0	0	0	0	3	10	0	13	3	0	0	0	4
1	0	0	383	0	0	0	3	0	0	0	0	3	1	0	1	0	23	1	639	9	0	2
18	0	1	0	2092	0	9	2	0	2	0	0	0	0	2	1	0	12	0	0	0	0	1
0	0	0	1	0	1151	0	1	0	0	0	0	2	0	0	0	2	7	0	0	0	0	8
47	0	0	0	6	0	5057	18	1388	7	1	7	5	0	11	15	0	3	1	0	0	0	9
37	0	1	0	14	0	21	3275	0	1	0	0	1	0	1	11	0	2	1	0	0	0	12
11	0	0	0	2	0	1292	17	3332	6	3	15	17	0	11	7	1	7	1	0	2	0	13
16	0	0	0	4	0	4	3	0	861	0	2	0	0	2	0	0	0	0	0	0	0	85
9	0	1	0	4	0	13	6	0	3	978	0	0	0	1	3	0	13	1	0	2	0	3
1026	0	0	0	2	0	6	1	0	0	0	2016	0	0	2	1	0	4	0	0	0	0	1
7	8	2	3	6	0	10	6	35	4	1	1	2669	2	15	19	0	67	4	0	0	0	51
0	1	0	0	0	0	1	0	0	8	0	0	6	656	1	0	0	17	0	0	0	0	17
6	0	0	0	6	0	13	5	0	1	0	0	0	0	1077	1	1	4	0	0	0	0	1
70	0	1	0	17	0	38	48	12	9	2	6	11	0	20	6066	0	83	1	0	1	0	50
0	0	0	3	0	0	2	1	0	0	0	1	0	0	4	3	2491	1	1	0	318	0	1
26	0	0	0	180	0	23	6	0	3	2	1	1	0	1	4	0	1755	1	0	0	0	5
10	0	0	0	2	0	8	25	6	3	1	4	20	1	3	18	0	43	935	0	3	0	51
1	0	1	119	0	0	0	2	0	0	0	0	5	5	0	2	2	27	2	576	6	0	9
2	183	1	1	0	0	0	13	3	2	0	0	89	31	23	6	0	38	11	6	2807	0	9
0	0	0	0	0	3	1	17	0	0	0	0	5	0	4	2	0	33	146	0	0	0	14
8	0	0	0	1	0	3	3	0	7	0	0	0	0	1	2	0	1	0	0	0	0	229

0.844473743438721

logcosh

4165	0	0	0	7	0	0	1	0	0	2	1	0	0	1	0	0	2	2	0	0	0	0
0	906	0	0	2	1	1	0	1	2	1	1083	1	0	0	0	0	0	3	0	0	0	0
1	3	595	0	0	1	0	3	7	0	0	0	0	0	0	17	0	11	4	0	0	0	1
0	0	0	288	0	0	0	0	0	0	0	0	1	0	0	1	2	2	1	760	11	0	0
12	0	1	0	2120	0	0	0	0	1	0	0	0	0	0	0	5	0	0	0	0	0	1
0	0	0	1	0	1156	0	0	0	0	0	0	0	0	0	0	1	2	0	0	11	1	
52	0	0	0	7	0	4926	11	1524	11	5	2	2	0	6	19	2	2	1	0	1	1	3
38	0	1	0	23	0	12	3265	2	5	0	0	0	1	1	13	0	6	1	0	0	1	8
20	0	0	0	2	0	1237	9	3413	6	3	6	7	0	1	9	8	3	1	0	2	2	8
18	0	0	0	4	0	1	1	0	943	0	1	0	0	0	0	0	0	0	0	0	0	9
6	0	0	0	4	0	5	4	0	1	1008	0	0	0	1	2	0	3	1	0	2	0	0
1902	0	0	0	2	0	4	0	0	0	1	1144	0	0	2	0	0	2	1	0	0	0	1
16	7	3	1	11	0	8	3	69	14	2	0	2566	1	6	44	5	99	12	0	10	4	29
1	0	0	0	1	0	1	0	0	16	0	0	6	629	1	1	7	20	3	1	18	0	2
6	0	0	0	9	0	4	4	0	3	0	0	1	0	900	2	180	3	1	0	1	0	1
73	0	0	0	25	0	33	24	6	16	2	2	3	0	8	6162	1	61	1	0	1	1	16
0	0	0	0	2	0	2	0	0	0	0	0	0	0	1	1	2586	1	0	0	233	0	0
26	0	0	0	473	0	15	3	0	6	3	1	0	0	0	1	0	1476	1	0	0	2	1
11	0	0	0	2	1	8	5	7	7	1	1	11	0	1	10	1	34	988	0	5	27	13
0	0	1	98	0	0	0	0	0	0	0	0	1	5	0	1	5	5	2	630	9	0	0
1	7	1	1	2	0	0	1	2	4	0	0	14	7	6	4	4	15	8	5	3139	0	4
0	0	0	0	0	0	1	0	0	1	0	0	1	0	2	0	0	4	2	0	0	214	0
8	0	0	0	3	0	0	0	0	15	0	0	0	0	1	2	0	0	0	0	0	0	226

0.8265634179115295

categorical_crossentropy

4167	0	0	0	1	0	0	4	0	0	2	1	0	0	1	0	0	5	0	0	0	0	0
0	950	0	0	0	1	2	2	1	2	1	1036	2	0	1	2	0	1	0	0	0	0	0
2	3	541	1	0	0	0	2	8	0	0	0	0	1	0	14	29	0	39	1	1	0	1
0	0	0	73	0	0	0	2	0	0	0	0	1	0	0	1	0	0	0	986	3	0	0
28	0	1	0	2073	0	3	11	0	1	1	0	0	0	7	3	0	10	0	0	0	0	2
0	0	0	2	0	1158	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	9	1
65	0	0	0	6	0	4717	16	1712	5	5	2	5	0	13	21	0	4	1	0	0	0	3
34	0	1	0	3	0	1	3323	0	0	0	0	0	0	1	8	0	5	0	0	0	0	1
34	0	0	0	2	0	1360	14	3273	3	3	6	13	0	5	11	0	9	1	0	1	0	2
42	0	0	0	4	0	3	12	2	802	0	1	4	7	2	1	10	0	0	0	0	0	85
7	0	0	0	1	0	2	5	0	0	1009	0	0	0	3	1	0	6	1	0	2	0	0
2042	0	0																				

poisson

4161	0	0	0	1	0	0	4	0	0	1	8	0	0	1	0	0	5	0	0	0	0	0
0	1007	0	0	1	0	0	0	1	2	0	983	2	0	0	1	0	0	3	0	0	0	1
0	3	548	0	0	0	0	2	6	0	0	0	1	0	0	19	0	60	4	0	0	0	0
2	0	0	343	0	0	0	1	0	0	0	0	3	0	0	1	0	6	2	701	7	0	0
24	0	1	0	2101	0	0	5	0	1	1	0	0	0	0	1	0	5	0	0	0	0	1
0	0	0	0	0	1142	0	0	0	0	0	0	1	0	0	0	0	2	7	0	0	20	0
66	0	0	0	7	0	3426	13	2986	7	4	7	14	0	7	23	1	7	4	0	0	0	3
38	0	1	0	10	0	4	3295	1	1	0	0	0	1	1	7	0	16	1	0	0	0	1
25	0	0	0	2	0	525	13	4100	5	2	12	19	1	2	10	0	13	6	0	0	1	1
27	0	0	0	3	0	1	5	1	914	0	1	0	4	1	2	0	7	1	0	0	0	10
9	0	1	0	2	0	2	6	0	1	993	0	3	0	1	1	0	15	1	0	2	0	0
1048	0	0	0	2	0	3	2	1	0	1	1993	0	0	2	2	0	3	1	0	0	0	1
32	7	2	0	9	0	5	6	29	3	1	3	2674	1	2	26	0	88	15	0	0	3	4
0	1	0	0	2	0	1	0	0	2	0	0	9	669	1	0	0	20	0	1	0	0	1
9	0	0	0	6	0	1	4	0	1	0	0	5	0	1062	2	13	11	1	0	0	0	0
91	0	0	0	20	0	21	31	14	4	2	11	8	0	8	6124	0	99	0	0	1	0	1
1	0	0	1	0	0	2	0	0	0	0	0	2	0	1	2	2575	1	3	0	238	0	0
29	0	0	0	187	0	8	5	0	2	2	1	1	0	0	2	0	1768	2	0	0	1	0
14	0	0	0	2	0	4	7	4	3	0	1	12	1	1	6	0	60	1003	0	1	11	3
5	0	1	146	0	0	0	1	0	0	0	0	6	4	0	3	3	8	2	576	1	0	1
13	254	2	6	5	0	0	12	3	1	0	0	195	25	11	23	0	57	102	7	2507	1	1
1	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	0	3	28	0	0	189	0
9	0	0	0	2	0	0	3	1	7	0	0	2	1	1	3	0	7	0	0	0	1	218

0.8254789710044861

cosine_proximity

4155	0	0	0	7	0	0	4	0	0	2	8	0	0	1	0	0	3	1	0	0	0	0
0	1156	0	0	0	0	1	1	0	2	1	839	0	0	0	0	0	1	0	0	0	0	0
0	3	585	0	0	3	0	8	6	0	8	0	0	0	2	16	0	11	1	0	0	0	0
0	0	0	32	0	1	0	3	0	0	1	0	1	0	0	1	0	2	1	1022	2	0	0
12	0	0	0	2111	0	3	4	0	1	2	0	0	0	2	0	0	5	0	0	0	0	0
0	0	0	1	0	1158	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	11	0
40	0	0	0	6	0	5562	18	902	4	13	6	5	0	7	8	1	1	1	0	0	0	1
29	0	1	0	9	0	6	3319	0	0	0	0	0	1	1	6	0	4	0	0	0	1	0
7	0	0	0	3	0	1907	12	2751	3	11	12	18	0	1	6	0	3	1	0	0	1	1
18	0	0	0	4	0	4	8	0	901	1	2	0	7	2	3	0	10	1	0	0	0	16
4	0	0	0	2	0	1	0	0	0	1025	0	0	0	1	1	0	2	0	0	1	0	0
953	0	0	0	2	0	6	2	0	0	1	2086	0	0	2	2	0	3	1	0	0	0	1
7	7	3	3	7	0	10	8	45	0	9	1	2701	4	6	19	0	71	5	0	0	3	1
0	1	0	0	0	1	1	2	0	0	3	0	9	676	1	0	0	12	0	0	0	0	1
6	0	0	0	6	0	6	5	0	0	0	0	3	0	1080	2	2	5	0	0	0	0	0
66	0	1	0	18	0	41	91	10	4	13	6	6	1	14	6107	0	55	0	0	1	0	1
0	0	0	3	0	1	2	4	0	0	2	0	5	1	3	2	2467	0	1	0	335	0	0
27	0	0	0	336	0	18	15	0	3	6	1	0	0	5	0	1595	1	0	0	1	0	0
8	0	0	0	2	1	8	20	9	2	3	4	17	2	3	7	0	74	939	0	2	28	4
0	0	1	7	1	1	0	0	0	0	0	0	1	5	0	1	0	4	2	733	1	0	0
2	240	1	1	1	12	0	26	5	1	14	0	141	56	25	13	0	50	6	18	2612	1	0
0	0	0	0	0	1	1	3	0	0	0	0	1	0	2	0	0	8	1	0	0	208	0
8	0	0	0	4	0	4	8	0	10	1	0	3	1	1	2	0	7	0	0	0	0	206

0.8402618169784546

Adamax

mean_squared_error

4175	0	0	0	1	0	0	1	0	0	1	1	0	0	1	0	0	0	1	0	0	0	0
0	797	0	0	0	0	2	0	1	2	0	1194	1	0	0	0	0	1	2	0	0	1	0
5	3	595	0	0	0	0	2	8	0	0	0	0	0	18	0	6	5	0	0	0	0	1
2	0	0	511	0	0	0	2	0	0	0	0	1	1	0	1	0	1	1	539	7	0	0
23	0	1	0	2089	0	4	5	0	3	0	0	0	0	0	0	11	0	0	0	1	3	3
1	0	0	2	0	1107	0	0	0	0	0	0	2	0	0	0	0	0	1	0	1	26	32
59	0	0	0	5	0	5417	17	1036	6	0	2	1	0	7	14	1	2	1	0	0	1	6
37	0	1	0	8	0	9	3303	0	0	0	0	0	1	1	9	0	3	2	0	0	0	3
21	0	0	0	2	0	1686	12	2983	5	1	6	3	0	2	5	0	1	1	0	3	1	5
19	0	0	0	3	0	3	3	0	883	0	1	0	0	1	1	0	0	0	0	0	0	63
12	0	1	0	4	0	10	4	1	1	986	0	0	1	1	4	0	7	1	0	2	0	2
2025	0	0	0	2	0	4	1	0	0	0	1022	0	0	2	1	0	0	1	0	0	0	1
40	6	3	1	5	0	9	5	101	6	1	1	2566	3	11	34	0	46	17	0	3	2	50
2	0	0	0	0	0	1	0	0	3	0	0	5	683	1	1	0	4	0	0	1	0	6
6	0	0	0	5	0	4	4	0	1	0	0	0	0	1085	1	1	4	0	0	2	0	2
90	0	1	0	15	0	37	42	10	8	2	2	6	1	11	6158	0	27	1	0	1	0	23
1	0	0	8	0	0	2	2	0	0	0	0	1	0	2	2	1714	1	1	0	1092	0	0
30	0	0	0	156	0	17	8	0	4	3	1	0	0	0	5	0	1778	1	0	0	1	4
15	0	0	0	2	0	9	9	5	4	1	1	9	1	1	9	0	18	1016	0	3	7	23
7	0	1	166	0	0	0	0	0	0	0	0	1	6	0	1	0	5	2	563	3	0	2
9	18	1	3	0	0	0	4	5	1	0	0	30	21	13	7	0	10	18	3	3077	0	5
1	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0	2	10	0	0	203	5
8	0	0	0	1	0	0	3	1	3	0	0	0	0	1	1	0	0	0	0	0	0	236

0.8170887231826782

mean_squared_logarithmic_error

4169	0	0	0	3	0	0	4	0	0	1	3	0	0	0	0	0	1	0	0	0	0	0
0	1154	0	0	0	0	1	2	1	2	0	831	5	0	0	1	0	1	2	0	0	1	0
0	3	582	1	0	0	0	7	8	0	0	0	0	0	0	34	0	4	2	0	0	0	2
0	0	0	384	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	668	10	0	0
20	0	1	0	2091	0	4	9	0	2	0	0	1	0	0	3	0	6	0	0	0	1	2
1	0	0	2	0	1146	0	1	0	0	0	0	3	0	0	0	0	0	1	0	0	15	3
47	0	0	0	6	0	5658	19	797	5	0	2	5	0	6	23	1	1	1	0	0	1	3
32	0	1	0	6	0	4	3323	0	0	0	0	0	0	9	0	0	1	1	0	0	0	0
17	0	0	0	2	0	2062	18	2584	2	1	6	20	0	1	12	0	1	1	0	5	1	4
19	0	0	0	4	0	3	16	0	830	0	1	1	9	1	2	0	1	0	0	0	0	90
9	0	1	0	4	0	11	23	0	1	973	0	1	0	1	5	0	5	1	0	2		

categorical_hinge

4150	0	0	0	6	0	9	1	0	0	3	6	0	0	1	2	0	0	2	0	0	0	1
0	1187	0	0	0	0	0	0	0	2	1	808	0	0	1	1	0	1	0	0	0	0	0
0	3	616	1	0	0	0	1	2	0	1	0	0	0	1	12	0	4	2	0	0	0	0
0	0	1	570	0	0	0	1	0	0	1	0	1	0	0	1	0	3	1	484	3	0	0
17	0	1	0	2087	0	8	2	0	4	2	0	1	0	1	3	0	12	0	0	0	1	1
0	0	0	1	0	1148	0	0	0	0	0	0	1	0	0	0	1	3	0	0	0	18	0
42	0	0	0	3	0	3267	13	3182	11	5	4	5	0	6	28	1	2	1	0	0	1	4
39	0	1	0	13	0	20	3269	1	4	1	0	1	0	1	19	0	3	2	0	0	1	2
15	0	0	0	2	0	620	10	4010	13	7	10	15	1	5	10	1	3	3	0	1	1	10
13	0	0	0	3	0	2	2	0	944	0	2	0	0	0	1	0	0	0	0	0	0	10
6	0	1	0	3	0	13	1	2	2	1001	0	0	0	1	2	0	2	1	0	2	0	0
1243	0	0	1	3	0	13	1	3	0	1	1775	2	0	3	8	0	4	1	0	0	0	1
10	7	4	5	6	0	9	1	82	17	28	0	2550	4	14	74	0	49	11	1	0	6	32
0	1	0	1	0	0	1	0	0	20	3	0	4	666	1	2	0	4	1	0	0	0	3
6	0	0	0	6	0	7	3	0	2	0	0	0	0	1079	2	4	4	1	0	0	0	1
63	0	0	0	14	0	31	19	10	12	3	4	3	0	9	6242	0	23	1	0	0	0	1
0	0	0	5	0	0	1	0	1	0	2	0	0	0	3	3	2570	0	1	0	240	0	0
28	0	0	0	211	0	21	8	1	8	4	1	1	0	0	7	0	1714	2	0	0	1	1
9	0	0	0	2	0	4	4	6	8	4	4	8	0	1	25	0	29	1006	0	1	13	9
0	0	1	210	0	0	0	0	0	0	1	0	1	5	0	3	3	6	2	525	0	0	0
1	92	7	11	0	1	0	4	1	4	20	0	43	32	30	28	0	15	20	12	2897	3	4
0	0	0	0	0	0	0	1	1	1	1	0	1	0	2	4	0	3	5	0	0	206	1
8	0	0	1	2	0	1	2	0	30	0	0	0	1	1	6	0	1	0	0	0	0	202

0.8310534358024597

logcosh

4159	0	0	0	4	0	0	4	0	0	2	4	0	0	2	2	0	1	3	0	0	0	0
0	1523	0	0	0	0	1	1	1	2	1	471	0	0	0	1	0	0	0	0	0	0	0
0	3	622	0	0	0	1	3	0	1	0	0	0	0	7	0	4	2	0	0	0	0	0
0	0	1	186	0	0	0	1	0	0	1	0	1	1	0	1	0	1	1	859	13	0	0
20	0	1	0	2095	0	1	7	0	2	3	0	0	0	0	2	0	7	0	0	0	1	1
1	0	1	1	0	1142	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	2
45	0	0	0	6	0	4702	20	1714	11	21	4	3	0	8	28	1	2	3	0	0	2	5
32	0	1	0	9	0	5	3312	0	0	0	0	0	1	1	10	0	3	2	0	0	1	0
8	0	0	0	3	0	1103	13	3513	8	23	12	14	2	4	13	2	2	4	0	3	4	6
13	0	0	0	4	0	0	4	0	940	1	2	0	0	0	0	0	0	0	0	0	0	13
7	0	1	0	2	0	1	2	0	1	1016	0	0	0	1	1	0	2	1	0	2	0	0
1260	0	0	0	2	0	10	3	0	0	1	1771	1	0	2	5	0	1	2	0	0	0	1
14	7	5	0	8	0	8	9	41	15	54	0	2588	3	11	49	1	50	16	0	1	15	15
1	1	0	0	0	0	1	0	0	6	5	0	3	684	1	1	0	3	0	0	0	0	1
7	0	0	0	5	0	3	5	0	3	2	0	0	0	1074	2	6	5	1	0	0	0	2
61	0	1	0	19	0	32	63	7	15	13	3	4	0	9	6175	0	22	1	0	1	0	9
0	0	0	2	0	0	2	0	0	0	2	0	1	0	1	2	2338	0	1	0	477	0	0
27	0	0	0	292	0	16	11	0	6	8	1	1	0	0	6	0	1635	1	0	0	2	2
8	0	0	0	2	0	4	13	3	5	3	4	8	0	1	11	0	14	996	0	4	52	5
0	0	2	40	1	0	0	0	0	0	0	0	1	6	0	1	3	3	2	694	3	0	1
3	89	5	1	0	0	0	9	2	2	4	0	18	23	8	4	0	7	7	4	3036	1	2
0	0	0	0	0	0	1	1	0	1	0	0	1	0	1	0	0	2	0	0	0	218	0
7	0	0	0	2	0	0	5	0	13	0	0	0	1	1	3	0	0	0	0	0	0	223

0.8493369817733765

categorical_crossentropy

4168	0	1	0	1	0	0	5	0	0	1	2	0	0	1	0	0	1	1	0	0	0	0
0	1001	0	0	0	1	4	2	5	2	1	963	2	0	0	15	0	1	4	0	0	0	0
0	3	596	0	0	0	0	1	6	0	0	0	0	0	31	0	5	1	0	0	0	0	0
2	0	0	77	0	0	0	2	0	0	0	0	1	0	0	1	0	1	968	14	0	0	0
48	0	1	0	1999	0	10	19	0	2	1	0	2	0	6	20	0	30	0	0	1	0	1
1	0	1	0	0	1156	0	0	0	0	0	0	0	0	1	0	1	2	0	0	10	0	
43	0	0	0	3	0	5868	15	609	2	1	0	0	0	6	23	1	2	1	0	0	0	1
30	0	1	0	2	0	7	3323	0	0	0	0	0	0	1	8	0	5	0	0	0	0	0
16	0	0	0	1	0	2409	12	2265	2	1	6	3	0	1	15	1	3	0	0	1	0	1
26	0	0	0	3	0	5	13	2	868	1	1	0	6	1	5	0	7	0	0	0	0	39
8	0	1	0	1	0	10	8	0	1	993	0	0	0	1	6	0	5	1	0	2	0	0
2009	0	0	0	2	0	17	5	2	0	1	1006	0	0	2	13	0	0	1	0	0	0	1
44	6	3	0	2	0	14	8	101	0	1	1	2544	1	3	102	0	56	9	1	4	2	8
4	0	0	0	0	0	1	1	1	0	0	0	9	673	1	5	0	7	0	1	3	0	1
13	0	0	0	1	0	7	4	0	1	0	0	0	0	1052	6	24	5	0	0	2	0	0
74	0	1	0	4	0	33	27	5	2	1	0	2	0	5	6265	0	14	0	0	1	0	1
1	0	0	2	0	0	2	0	0	0	0	0	1	0	1	3	2250	1	0	0	565	0	0
34	0	0	0	32	0	17	12	0	1	3	1	2	0	0	9	0	1893	1	0	0	1	2
22	0	0	0	2	0	10	13	12	1	1	1	12	1	1	24	0	34	978	0	4	10	7
5	0	1	16	0	0	0	0	0	0	0	0	1	4	0	3	0	3	2	719	3	0	0
7	61	1	0	0	0	0	4	3	1	0	0	30	11	4	35	0	10	6	7	3043	0	2
1	0	0	0	0	0	1	1	0	0	0	0	1	0	1	2	0	3	4	0	0	211	0
9	0	0	0	1	0	2	4	0	4	0	0	0	1	1	7	0	1	0	0	0	0	225

0.8213884830474854

kullback_leibler_divergence

4166	0	0	0	1	0	0	5	0	0	2	2	0	0	1	0	0	4	0	0	0	0	0
0	923	0	0	0	0	2	1	2	2	1	1050	2	0	1	12	0	1	3	0	0	1	0
1	3	585	0	0	0	0	1	7	0	0	0	0	0	2	31	0	12	1	0	0	0	0
0	0	0	410	0	0	0	2	0	0	0	0	1	1	0	1	0	1	1	640	9	0	0
32	0	1	0	2077	0	5	7	0	2	1	0	0	0	4	4	0	7	0	0	0	0	0
1	0	0	1	0	1150	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	17	0
54	0	0	0	6	0	5435	17	1001	5	6	3	1	0	8	34	1	2	1	0	0	0	1
39	0	1	0	4	0	7	3306	0	0	0	0	0	1	1	11	0	6	0	0	0	1	0
22	0	0	0	2	0	1701	10	2940	4	5	7	5	0	3	26	1	7	1	0	2	0	1
28	0	0	0	3	0	3	4	1	915	0	1	0	6	2	0	3	0	0	0	0	9	0
5	0	1	0	1	0	4	1	0	1	1013	0	1	0	1	1	0	5	1	0	2	0	0
1756</																						

cosine_proximity

4154	0	0	0	5	0	1	3	0	0	2	6	1	0	2	0	0	3	4	0	0	0	0
0	1349	0	0	0	0	1	0	1	2	1	643	1	0	1	0	0	0	2	0	0	0	0
0	3	617	1	0	0	0	1	3	0	0	0	1	0	1	3	0	9	4	0	0	0	0
0	0	0	394	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	666	3	0	0
18	0	2	0	2092	0	3	4	0	2	2	0	2	0	0	1	0	12	0	0	0	1	1
2	0	0	2	0	1149	0	0	2	0	0	0	1	0	0	0	0	1	1	0	0	13	1
42	0	0	0	6	0	5264	16	1185	6	11	4	7	0	8	18	1	2	2	0	0	1	2
35	0	1	0	10	0	12	3294	0	0	2	0	1	1	1	9	0	5	4	0	0	1	1
10	0	0	0	2	0	1545	8	3097	4	9	11	22	2	4	8	1	3	6	0	1	1	3
13	0	0	0	3	0	2	4	0	931	1	2	0	0	1	1	0	3	0	0	0	0	16
6	0	1	0	2	0	4	0	0	1	1018	0	0	1	1	0	2	1	0	0	0	0	0
1324	0	0	1	2	0	15	3	2	0	1	1689	6	0	2	5	0	5	3	0	0	0	1
8	7	2	0	8	0	9	1	39	5	15	0	2683	2	10	19	0	72	14	1	1	5	9
0	1	0	0	1	0	1	0	0	5	4	0	11	660	1	0	0	13	1	1	6	0	2
4	0	0	0	6	0	4	3	0	0	2	0	1	0	1084	1	4	5	1	0	0	0	0
62	0	1	0	16	0	39	31	6	9	11	3	9	1	13	6173	0	53	3	0	1	1	3
0	0	0	3	0	0	2	0	0	0	2	0	2	0	2	1	2540	1	1	0	272	0	0
26	0	0	0	221	0	18	6	0	2	5	1	1	0	0	3	0	1721	2	0	0	2	0
8	0	0	0	2	0	6	5	4	4	2	4	13	0	1	4	0	26	1037	0	2	10	5
0	0	1	138	0	0	0	0	0	0	0	0	1	4	0	1	3	4	2	602	1	0	0
2	43	1	7	1	0	0	4	2	2	7	0	185	20	25	5	1	15	50	9	2844	1	1
0	0	0	0	0	0	1	0	0	1	0	0	1	0	1	0	0	3	6	0	0	212	0
7	0	0	0	2	0	1	3	1	8	0	0	0	1	2	3	0	4	0	0	0	0	223

0.8528566956520081

Nadam

mean_squared_error

4152	0	2	0	6	0	3	2	0	0	2	4	1	0	1	1	0	3	4	0	0	0	0
0	1523	0	0	1	0	2	1	1	0	1	465	1	0	0	0	0	1	4	0	0	1	0
0	3	602	0	0	0	0	1	9	0	1	0	1	0	0	12	0	6	8	0	0	0	0
0	0	0	131	0	0	0	0	0	0	0	0	4	0	0	1	0	0	1	916	13	0	0
16	0	2	0	2019	0	5	7	1	1	1	0	2	0	3	1	0	81	0	0	0	1	0
0	0	0	1	0	1114	0	0	0	0	0	0	23	0	0	0	0	1	3	0	0	30	0
35	0	0	0	5	0	4516	4	1990	2	6	3	4	0	8	1	0	1	0	0	0	0	0
32	0	1	0	7	0	20	3290	4	0	0	4	0	1	10	0	4	4	0	0	0	0	0
5	0	0	0	2	0	958	6	3721	3	1	12	21	0	1	0	0	1	3	0	2	1	0
19	0	0	0	4	0	3	18	7	811	1	1	1	3	2	6	1	11	2	0	0	1	86
4	0	0	0	2	0	3	0	0	1021	0	0	0	1	1	0	2	1	0	2	0	0	0
1581	0	0	0	2	0	14	3	0	0	1	1441	3	0	2	5	0	3	3	0	0	0	1
4	7	1	0	3	0	10	2	127	0	2	0	2674	1	3	18	0	36	12	0	7	2	1
1	1	0	0	0	0	1	2	7	0	1	0	25	620	1	2	0	11	2	1	31	0	1
5	0	0	0	1	5	0	4	4	5	0	1	0	5	0	1062	2	11	5	2	0	3	0
51	0	1	0	13	0	40	43	34	2	4	3	33	2	7	6142	0	40	12	0	5	1	2
0	0	0	2	0	0	2	1	0	0	0	0	2	0	1	1	2274	1	1	0	541	0	0
23	0	0	0	64	0	18	7	2	1	3	1	6	0	0	4	0	1875	2	0	0	2	0
11	0	0	0	2	0	5	4	14	3	1	1	17	0	0	4	0	16	1021	0	6	25	3
0	0	1	61	0	0	0	0	0	0	0	0	10	4	0	1	0	3	2	669	6	0	0
1	16	1	0	0	0	0	6	10	0	0	0	98	8	2	3	0	5	34	4	3036	1	0
0	0	0	0	0	0	1	0	0	0	0	0	2	0	1	0	0	0	36	0	0	185	0
7	0	0	0	2	0	3	6	2	2	0	0	2	1	1	2	0	5	0	0	1	0	221

0.8394056558609009

mean_squared_logarithmic_error

4169	0	0	0	2	0	0	2	0	0	2	4	0	0	0	0	0	2	0	0	0	0	0
0	1128	0	0	0	0	1	0	2	2	0	864	0	0	0	1	0	0	2	0	0	1	0
0	3	529	1	0	0	0	5	20	0	1	0	0	0	0	73	0	2	7	1	0	0	1
0	0	0	238	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	822	3	0	0
21	0	1	0	2088	0	6	8	0	1	1	0	0	0	0	5	0	9	0	0	0	0	0
0	0	0	1	0	1156	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	12	0
38	0	0	0	5	0	5245	14	1238	3	1	5	0	0	5	19	0	1	0	0	0	0	1
35	0	0	0	8	0	10	3304	1	0	0	0	1	0	0	12	0	3	3	0	0	0	0
10	0	0	0	2	0	1662	10	3017	4	2	9	3	0	1	11	1	1	1	0	1	1	1
22	0	0	0	3	0	3	6	1	910	1	1	0	2	1	3	0	1	1	0	0	1	21
6	0	0	0	2	0	8	4	1	0	1007	0	0	0	1	3	0	2	1	0	2	0	0
1498	0	0	0	2	0	10	0	2	0	1	1543	0	0	0	1	0	1	0	0	0	0	1
10	7	1	0	6	0	16	3	144	1	9	2	2496	4	1	152	0	29	7	0	2	2	18
2	1	0	0	0	0	1	0	0	2	2	0	4	679	1	7	0	6	0	1	0	0	1
8	0	0	0	3	0	9	4	8	1	0	0	5	0	1011	11	38	5	2	0	3	0	1
60	0	1	0	16	0	37	27	5	4	2	4	1	1	5	6260	0	10	1	0	0	0	1
1	0	0	0	3	0	2	0	0	0	0	0	1	0	0	4	2338	0	1	0	475	0	1
28	0	0	0	252	0	21	11	1	3	3	1	1	1	0	23	0	1658	2	0	0	1	2
13	0	0	0	2	0	8	7	16	2	1	1	12	2	0	33	0	15	983	0	1	28	9
0	0	1	90	1	0	0	0	0	0	0	0	1	6	0	1	0	2	2	651	2	0	0
1	83	2	2	0	0	0	5	25	1	2	2	61	33	1	80	0	8	17	16	2882	0	4
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	1	3	0	0	218	0
8	0	0	0	1	0	1	3	1	5	0	0	0	1	1	5	0	0	0	0	0	0	229

0.8321568965911865

squared_hinge

4159	0	0	0	6	0	0	5	0	0	2	2	0	0	2	1	0	1	3	0	0	0	0
0	1303	0	0	2	0	1	1	0	1	0	690	0	0	1	0	0	0	2	0	0	0	0
0	3	609	0	0	0	0	4	3	0	0	0	0	0	3	10	0	9	2	0	0	0	0
0	0	0	1049	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1	0	12	0	0
16	0	1	0	2111	0	1	2	0	1	0	0	0	0	0	0	0	8	0	0	0	0	0
3	0	0	1	0	1159	0	0	0	0	0	0	0	0	1	0	0	1	7	0	0	0	0
48	0	0	0	7	0	3957	13	2519	4	1	2	0	0	9	12	0	2	1	0	0	0	0
30	0	1	0	13	0	9	3304	1	0	0	0	0	1	1	10	0	3	4	0	0	0	0
12	0	0	0	3	0	657	7	4024	3	1	8	4	0	2	6	1	2	4	0	3	0	0
19	0	0	0	4	0	3	2	943	0	1	0	0	1	0	0	0	1	0	0	0	0	0
5	0	0	0	4	0	4	2	0	1	1013	0	0	0	1	2	0	2	1	0	2	0	0

logcosh

4159	0	0	0	6	0	0	3	0	0	1	7	0	0	1	0	0	2	2	0	0	0	0	0
0	767	0	0	1	1	0	1	0	1	0	1229	0	0	0	0	0	0	1	0	0	0	0	0
8	3	558	0	0	0	0	8	8	0	0	0	0	0	1	19	0	28	8	1	0	0	1	0
0	0	0	80	0	0	0	1	0	0	0	0	1	0	0	1	0	7	1	966	9	0	0	0
17	0	1	0	2109	0	1	3	0	2	0	0	0	0	0	0	0	6	0	0	0	0	1	0
0	0	0	1	0	1149	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	18	1	1
51	0	0	0	7	0	5316	19	1143	6	2	4	4	0	8	9	0	2	1	0	0	0	3	0
32	0	0	0	11	0	4	3319	0	0	0	0	1	0	6	0	2	2	0	0	0	0	0	0
21	0	0	0	2	0	1707	12	2957	3	1	9	7	0	1	6	0	3	1	0	3	2	2	0
20	0	0	0	4	0	1	10	0	900	0	1	2	4	1	1	0	3	1	0	0	0	29	0
11	0	0	0	4	0	5	21	0	1	970	0	1	0	1	3	0	16	2	0	2	0	0	0
1143	0	0	0	2	0	3	1	0	0	0	1907	0	0	1	0	0	1	0	0	0	0	1	0
14	6	2	0	7	0	9	7	48	1	1	2	2628	3	2	36	0	114	12	0	2	5	11	0
0	1	0	0	1	0	1	1	0	1	0	0	7	671	1	1	0	20	1	0	0	0	1	0
6	0	0	0	7	0	5	5	0	0	0	0	5	0	1075	2	1	4	1	0	3	0	1	0
79	0	1	0	18	0	40	93	9	8	2	4	5	1	9	6053	0	104	1	0	1	1	6	0
1	0	0	4	0	0	2	2	0	0	0	0	1	0	1	1	1547	3	1	1	1262	0	0	0
27	0	0	0	289	0	16	10	0	3	2	1	1	0	0	3	0	1652	2	0	0	1	1	0
11	0	0	0	2	0	7	17	7	1	1	3	9	1	1	4	0	36	1001	0	4	22	6	0
0	0	1	22	0	0	0	0	0	0	0	0	1	5	0	1	0	10	2	712	3	0	0	0
1	62	1	0	4	0	0	8	2	1	0	6	40	22	6	13	0	48	35	9	2963	1	3	0
1	0	0	0	0	0	1	2	0	0	0	0	1	0	1	0	0	1	6	0	212	0	0	0
8	0	0	0	2	0	0	5	1	4	0	0	0	1	1	3	0	7	0	0	0	0	223	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

0.8167272210121155

categorical_crossentropy

4164	0	2	0	1	0	0	2	0	0	2	2	0	0	1	3	0	3	0	0	0	0	0	1
0	929	0	0	0	1	6	0	16	2	1	1021	3	0	0	3	18	0	0	1	0	0	0	0
0	3	577	1	0	0	0	1	8	0	0	0	1	0	0	46	0	4	2	0	0	0	0	0
0	0	0	225	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	833	5	0	0	0
34	0	2	0	2029	0	11	7	0	3	2	0	2	0	6	26	0	9	0	0	0	0	9	0
0	0	0	2	0	1147	0	0	1	0	0	0	4	0	0	1	2	1	2	0	1	8	3	0
35	0	0	0	1	0	5746	4	742	2	2	3	3	0	7	24	1	1	0	0	0	0	4	0
40	0	2	0	4	0	18	3282	3	0	0	0	1	1	0	15	0	7	1	0	0	0	3	0
7	0	0	0	1	0	2118	5	2559	4	1	8	13	0	1	17	0	1	0	0	0	0	2	0
21	0	0	0	2	0	3	4	1	869	0	1	0	12	1	5	0	0	0	0	0	0	58	0
5	0	1	0	1	0	4	7	0	1	1001	0	0	0	1	9	0	3	1	0	2	0	1	0
1378	0	0	0	1	0	24	1	3	0	1	1629	1	1	2	16	0	1	0	0	0	0	1	0
18	6	2	0	3	0	9	1	111	0	4	0	2633	4	7	69	0	14	5	0	1	1	22	0
1	0	0	0	0	0	1	0	2	1	0	0	4	693	1	3	0	0	0	0	0	0	1	0
9	0	1	0	2	0	4	4	4	1	0	0	7	0	1038	8	27	5	1	0	3	0	1	0
52	0	1	0	5	0	32	11	11	3	2	2	4	0	4	6296	0	6	0	0	1	0	5	0
1	0	0	5	0	0	2	0	1	0	0	0	1	0	1	2	2459	0	0	0	354	0	0	0
40	0	0	0	107	0	21	10	1	4	3	1	5	2	1	28	0	1773	1	0	0	0	11	0
17	0	0	0	2	0	8	7	20	3	2	3	19	5	1	36	0	21	965	0	4	4	16	0
0	0	0	73	0	0	0	0	0	0	0	0	2	5	0	2	0	1	1	672	1	0	0	0
4	80	1	3	0	0	0	3	39	1	1	0	131	49	7	57	0	5	8	14	2818	0	4	0
2	0	0	0	0	0	1	1	0	0	0	0	2	0	1	3	0	1	25	0	0	187	2	0
9	0	0	0	1	0	0	0	1	4	0	0	0	1	1	5	0	0	0	0	0	0	233	0

0.8356766700744629

kullback_leibler_divergence

4147	0	2	0	1	0	1	7	0	0	2	5	1	0	3	2	0	4	2	0	0	0	4	0
0	1423	0	0	0	1	6	1	4	1	1	542	3	0	3	11	0	1	3	0	0	0	1	0
0	3	614	0	0	0	0	1	3	0	0	0	1	0	0	13	0	6	2	0	0	0	0	0
0	0	0	400	0	0	0	1	1	0	0	0	3	0	0	1	0	0	2	648	10	0	0	0
23	0	2	0	2051	0	8	11	0	2	3	0	4	0	10	10	0	12	0	0	0	0	4	0
0	0	0	1	0	1134	0	0	3	0	0	1	16	0	0	0	0	1	4	0	0	12	0	0
22	0	0	0	3	0	5447	9	1048	3	6	4	5	0	9	10	0	1	3	0	0	0	5	0
26	0	1	0	4	0	13	3310	2	1	1	0	0	0	1	8	0	2	3	0	0	0	5	0
2	0	0	0	1	0	1901	8	2765	4	5	7	24	0	6	4	0	1	2	0	1	0	6	0
20	0	0	0	3	0	3	3	0	846	1	1	3	5	3	2	0	1	0	0	0	0	86	0
4	0	1	0	1	0	1	1	0	0	1019	0	0	0	3	2	0	1	1	0	2	0	1	0
1183	1	0	0	3	0	23	4	4	0	4	1804	5	0	8	12	0	2	1	0	0	0	5	0
8	7	2	0	3	0	9	2	70	0	6	0	2698	1	27	26	0	20	9	0	0	1	21	0
0	1	0	0	1	0	1	0	0	2	2	0	13	674	1	1	0	4	0	1	5	0	1	0
4	0	0	0	0	0	1	3	0	0	0	0	0	0	1101	1	0	3	1	0	1	0	0	0
50	0	1	0	8	0	38	26	11	5	5	3	9	1	26	6219	0	17	1	0	1	0	14	0
0	0	0	8	0	0	2	0	0	0	0	0	14	0	12	2	1933	0	1	0	854	0	0	0
30	0	0	0	105	0	20	8	0	2	4	1	4	1	5	7	0	1811	1	0	0	1	8	0
11	0	0	0	2	0	7	6	7	1	1	1	16	1	4	6	0	15	1034	0	4	3	14	0
0	0	1	144	0	0	0	0	0	0	0	0	3	4	1	1	0	1	2	598	1	0	1	0
1	145	1	2	0	0	0	5	9	1	2	0	112	16	53	10	0	7	17	5	2835	0	4	0
0	0	0	0	0	0	1	2	0	0	0	0	2	0	4	0	0	25	0	0	188	3	0	0
7	0	0	0	1	0	0	0	2	3	0	0	0	0	1	2	0	0	0	0	0	0	239	0

0.8426399827003479

poisson

4168	0	1	0	1	0	0	5	0	0	2	1	0	0	1	0	0	2	0	0	0	0	0	0
0	622	0	0	1	0	0	1	1	1	1	1369	1	0	0	2	0	0	1	0	0	1	0	0
0	2	622	0	0	0	0	1	1	0	0	0	0	0	0	9	0	7	1	0	0	0	0	0
3	0	0	73	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	980	8	0	0	0
41	0	1	0	2060	0	5	18	0	1	1	0	0	0	2	3	0	7	0	0	0	0	1	0
1	0	1	1	0	1135	0	1	0	0	0	0	3	0	0	1	1	2	0	1	1	24	0	0
81	0	0	0	4	0	5105	18	1313	2	8	2	0	0	8	29	0	2	2	0	0	0	1	0
32	0	1																					

(16, 2) szűrő eredményei

RMSprop

mean_squared_error

Table with 22 columns and 22 rows of numerical data for RMSprop mean_squared_error. Values range from 0 to 212.

0.82745760679245

mean_squared_logarithmic_error

Table with 22 columns and 22 rows of numerical data for RMSprop mean_squared_logarithmic_error. Values range from 0 to 227.

0.8635299801826477

squared_hinge

Table with 22 columns and 22 rows of numerical data for squared_hinge. Values range from 0 to 216.

0.8423736095428467

categorical_hinge

Table with 22 columns and 22 rows of numerical data for categorical_hinge. Values range from 0 to 204.

0.8079564571380615

logcosh

Table with 22 columns and 22 rows of numerical data for logcosh. Values range from 0 to 224.

0.8279522657394409

categorical_crossentropy

4136	0	2	0	2	0	1	9	0	0	2	12	3	0	1	6	0	5	1	0	0	0	1
0	1559	0	0	0	0	1	0	1	2	0	426	0	0	0	8	0	1	2	0	0	1	0
0	3	607	0	0	0	0	1	2	0	0	0	0	0	23	0	6	1	0	0	0	0	0
0	0	0	169	0	0	0	0	0	0	0	0	1	1	0	2	0	1	1	870	21	0	0
19	0	2	0	2077	0	3	2	0	2	1	0	1	0	3	17	0	11	0	0	0	0	2
0	0	0	1	0	1148	0	0	0	0	0	0	1	0	0	0	1	1	0	1	19	0	0
25	0	0	0	5	0	4035	15	2399	4	1	6	6	0	8	60	0	2	4	0	3	0	2
26	0	2	0	7	0	9	3300	1	1	0	0	1	1	21	0	6	1	0	0	0	1	1
3	0	0	0	2	0	824	10	3808	4	1	11	11	1	2	47	1	4	2	0	4	1	1
18	0	0	0	4	0	1	4	1	910	0	2	0	7	1	4	0	4	1	0	0	0	20
4	1	1	0	1	0	7	6	2	1	988	0	2	0	1	10	0	9	1	0	2	0	1
611	1	0	1	2	0	16	6	6	0	1	2378	3	0	2	25	0	5	1	0	0	0	1
7	7	3	0	6	0	6	2	55	1	1	2	2609	5	3	122	0	54	8	0	10	4	5
0	1	0	0	1	0	1	0	0	1	0	0	5	684	1	4	0	5	0	1	2	0	1
4	0	0	0	2	0	1	4	0	0	0	0	1	0	1078	7	7	6	0	0	5	0	0
41	0	0	0	8	0	11	16	2	2	1	1	3	1	6	6322	0	18	0	0	2	0	1
0	0	0	1	0	0	2	0	0	0	0	0	0	0	0	1992	1	0	0	0	826	0	0
26	0	0	0	111	0	14	6	0	3	2	1	3	1	0	14	0	1822	1	0	0	2	2
6	0	0	0	2	0	4	7	6	3	0	4	11	2	1	21	0	31	1007	0	7	16	5
0	0	1	60	0	0	0	0	0	0	0	0	1	5	1	2	0	3	2	662	17	0	3
1	159	1	0	0	0	0	1	0	1	0	0	12	12	2	19	0	8	5	4	2997	0	3
0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	1	0	0	219	0
7	0	0	0	1	0	0	0	1	9	0	0	0	1	1	9	0	1	0	0	2	0	223

0.8510112166404724

kullback_leibler_divergence

4166	0	0	0	1	0	0	5	0	0	2	2	0	0	1	0	0	1	3	0	0	0	0
0	1153	0	0	0	0	6	2	3	2	2	815	2	0	1	7	0	1	2	0	0	4	1
0	3	622	0	0	0	1	2	0	0	0	0	0	0	8	0	6	1	0	0	0	0	0
0	0	0	13	0	0	0	0	0	0	0	0	1	0	0	0	1	1045	5	0	0	0	0
31	0	1	0	2030	0	9	16	0	2	3	0	2	0	4	11	0	27	0	0	0	2	2
0	0	0	1	0	1155	0	0	0	0	0	0	0	0	0	0	1	4	0	0	0	11	0
33	0	0	0	3	0	5982	12	498	3	7	1	1	0	7	20	1	2	3	0	0	1	1
30	0	1	0	2	0	9	3323	0	0	0	0	1	0	6	0	3	1	0	0	1	0	0
8	0	0	0	0	0	2567	9	2103	4	5	8	5	2	1	11	0	3	5	0	0	5	1
25	0	0	0	2	0	4	8	1	882	2	1	0	16	1	4	0	4	1	0	0	1	25
4	0	1	0	1	0	2	1	0	0	1022	0	0	0	1	0	0	2	1	0	2	0	0
1584	0	0	0	2	0	25	3	3	0	3	1412	2	0	2	9	0	3	6	0	0	2	3
21	7	3	0	3	0	11	2	111	0	6	0	2596	5	2	38	0	64	16	2	2	16	5
2	1	0	0	0	0	1	1	1	0	0	0	4	689	1	1	0	2	1	1	0	1	1
7	0	1	1	1	0	2	6	5	0	2	0	1	0	1070	3	8	5	1	0	1	1	0
65	0	1	0	6	0	35	27	3	3	10	3	6	1	7	6239	0	24	1	0	1	2	1
1	0	0	2	0	0	2	0	0	0	1	0	1	0	2	3	2261	0	2	1	550	0	0
30	0	0	0	35	0	19	12	0	2	5	1	2	2	0	6	0	1888	2	0	0	2	2
13	0	0	0	2	0	8	8	5	2	1	1	10	1	0	8	0	16	935	0	1	120	2
0	0	1	1	0	0	0	0	0	0	0	0	1	4	0	1	0	1	1	746	1	0	0
4	184	2	2	0	0	5	10	0	9	0	71	36	4	15	0	14	49	24	2789	5	2	
0	0	0	0	0	0	1	1	0	0	0	0	1	0	1	0	0	2	5	0	0	214	0
9	0	0	0	1	0	1	4	0	6	1	0	1	1	1	7	0	1	0	0	0	1	221

0.8278191089630127

poisson

4165	0	0	0	1	0	0	7	0	0	2	2	0	0	1	1	0	1	1	0	0	0	0
0	1405	0	0	1	0	5	2	2	2	3	564	3	0	1	4	0	4	3	0	0	1	1
0	3	616	0	0	0	1	2	0	0	0	0	0	0	5	0	15	1	0	0	0	0	0
0	0	0	328	0	1	0	1	0	0	0	0	1	1	0	1	0	1	1	725	6	0	0
23	0	2	0	2074	0	5	9	0	2	2	0	1	0	4	3	0	13	0	0	0	1	1
0	0	0	0	0	1151	0	0	0	0	0	0	0	0	0	0	1	5	0	0	0	15	0
38	0	0	0	4	1	5387	18	1077	5	8	1	5	0	7	11	1	4	6	0	0	0	2
26	0	1	0	6	0	4	3330	0	0	0	0	1	1	3	0	2	1	0	0	0	1	1
11	0	0	0	2	1	1744	12	2902	4	7	6	13	2	2	7	1	6	9	0	3	2	3
22	0	0	0	3	0	2	6	0	915	1	1	0	4	1	1	0	1	1	0	0	0	19
4	0	0	0	1	0	2	1	0	0	1023	0	0	0	1	0	0	3	1	0	0	0	1
1874	0	0	0	2	0	17	5	3	0	4	1127	4	0	2	11	0	6	1	0	0	0	3
25	7	4	0	4	2	9	6	71	7	11	0	2585	4	5	24	0	93	20	0	1	9	23
0	1	0	0	1	1	1	1	0	2	0	0	5	686	1	0	0	5	0	1	0	0	2
5	0	0	0	1	0	1	5	1	0	0	0	1	0	1089	1	4	5	1	0	1	0	0
68	0	1	0	11	1	37	49	10	6	11	2	6	1	9	6157	0	59	1	0	1	0	5
0	0	0	5	0	0	2	1	0	0	1	0	1	1	3	2	2315	2	2	0	491	0	0
29	0	0	0	82	0	14	11	0	3	3	0	2	2	0	4	0	1852	1	0	0	1	4
12	0	0	0	2	0	4	16	3	3	2	1	9	1	1	5	0	29	1022	0	2	15	6
0	0	1	122	0	1	0	0	0	0	0	0	1	6	0	1	0	3	2	618	1	0	1
6	149	4	4	0	4	0	10	3	1	5	0	43	28	7	9	0	13	45	8	2883	1	2
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	1	5	0	0	216	0
8	0	0	0	1	0	0	1	1	4	1	0	1	1	1	2	0	2	0	0	0	0	232

0.838606595993042

cosine_proximity

4162	0	0	0	4	0	0	4	0	0	2	6	1	0	0	0	0	2	0	0	0	0	0
0	1013	0	0	0	0	0	0	2	2	0	978	1	0	2	0	1	1	0	0	0	0	1
3	3	594	0	0	0	5	10	0	0	0	0	1	0	0	19	0	5	1	0	0	0	2
0	0	0	210	0	0	0	0	0	0	0	0	1	0	0	0	0	1	847	7	0	0	0
20	0	2	0	2095	0	3	5	0	4	1	0	0	0	0	0	0	9	0	0	0	0	1
6	0	0	1	0	1148	0	0	2	0	0	0	2	0	0	0	0	1	0	1	9	2	0
46	0	0	0	6	0	3155	19	3310	7	1	3	7	0	8	7	0	1	1	0	1	0	3
32	0	0	0	7	0	1	3321	0	0	0	0	1	1	9	0	1	1	0	0	1	2	2
9	0	0	0	2	0	375	13	4295	3	1	11	10	0	1	7	0	1	1	0	3	1	4
17	0	0	0	3	0	1	4	2	919	0	2	0	4	1	0	0	0	0	0	0	0	24
7	0	1	0	2	0	6	5	2	1	1002	0	2	0	1	1	0	3	1	0	2	0	1
1468	0	0	0	2	0</																	

mean_squared_logarithmic_error

4115	0	2	0	8	0	8	5	0	3	2	22	1	0	5	5	0	3	1	0	0	1	0
0	1245	0	0	0	0	1	2	1	1	1	743	3	0	0	1	0	1	2	0	0	0	0
0	3	613	1	1	0	0	2	4	0	1	0	0	0	1	11	0	4	2	0	0	0	0
1	0	0	402	0	0	0	3	3	0	0	0	2	1	1	3	2	2	2	633	11	0	0
23	1	1	0	2068	0	5	2	0	3	1	0	5	0	3	2	0	25	0	0	0	1	0
8	0	0	0	0	1144	0	0	0	0	0	0	0	0	0	3	0	3	0	0	0	11	0
44	0	1	0	5	0	4936	16	1500	7	0	4	14	0	6	29	3	2	5	1	1	1	0
44	0	2	0	15	0	17	3259	1	3	0	0	4	3	3	19	0	2	4	0	1	0	0
13	0	0	0	2	2	1386	13	3233	8	4	14	18	2	4	21	2	5	5	1	4	0	0
12	0	0	0	3	0	3	10	1	932	2	2	1	5	1	1	1	3	0	0	0	0	0
10	0	1	0	4	0	11	3	1	1	991	0	1	2	1	4	0	6	0	0	1	0	0
1030	3	1	0	4	0	9	2	1	0	1	1984	9	1	3	4	0	2	3	0	2	0	0
29	9	3	3	6	1	24	7	109	3	8	1	2563	6	3	46	1	69	8	1	7	3	0
1	2	0	0	0	0	1	0	0	4	2	0	2	681	1	1	1	3	0	0	8	0	0
8	0	0	0	4	0	3	2	0	0	1	0	2	0	1066	11	15	3	0	0	0	0	0
85	0	1	0	22	2	35	43	5	12	8	4	8	1	13	6162	0	30	2	0	2	0	0
1	0	0	3	0	1	2	0	0	0	1	0	0	0	3	2	2276	1	1	0	535	0	0
33	1	0	0	301	0	20	13	0	4	3	0	2	0	0	9	0	1618	2	0	0	2	0
11	0	0	0	2	3	6	26	8	5	2	5	9	0	1	17	0	38	956	0	1	43	0
1	0	1	100	1	0	0	3	0	0	0	1	7	0	3	3	3	2	622	10	0	0	0
6	84	1	1	1	0	0	7	4	0	2	1	12	13	11	6	2	6	5	3	3059	1	0
0	0	0	0	0	0	0	2	1	0	1	0	1	0	2	2	0	4	7	0	0	206	0
33	0	0	0	3	0	3	44	1	81	1	0	1	0	1	24	0	59	2	0	0	2	0

0.8396149277687073

squared_hinge

4137	0	0	0	0	0	12	5	0	0	0	15	1	0	0	8	0	0	2	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	1998	1	0	0	1	0	0	1	0	0	0	0
42	0	0	0	0	0	0	73	9	0	0	12	21	0	0	421	0	0	5	59	1	0	0
1	0	0	0	0	0	0	3	8	0	0	0	12	0	0	9	8	0	2	986	37	0	0
329	0	0	0	0	0	18	113	2	0	0	1	95	0	0	1577	1	0	0	0	4	0	0
65	0	0	0	0	0	0	99	254	0	0	0	18	0	0	27	44	0	156	470	39	0	0
38	0	0	0	0	0	4260	9	2243	0	0	5	7	0	0	8	3	0	2	0	0	0	0
48	0	0	0	0	0	22	3265	6	0	0	0	2	0	0	29	1	0	3	0	1	0	0
12	0	0	0	0	0	865	10	3796	0	0	15	18	0	0	13	2	0	4	0	2	0	0
27	0	0	0	0	0	4	403	8	0	0	2	7	0	0	421	1	0	71	32	1	0	0
76	0	0	0	0	0	13	287	5	0	0	1	7	0	0	422	2	0	24	184	16	0	0
1099	0	0	0	0	0	7	2	0	0	0	1943	3	0	0	1	0	0	3	0	1	0	0
36	0	0	0	0	0	18	11	134	0	0	9	2551	0	0	100	2	0	17	22	10	0	0
4	0	0	0	0	0	1	9	3	0	0	0	15	0	0	49	5	0	77	392	152	0	0
32	0	0	0	0	0	10	17	5	0	0	0	4	0	0	72	890	0	0	31	54	0	0
85	0	0	0	0	0	33	45	23	0	0	4	11	0	0	6217	6	0	4	3	4	0	0
0	0	0	0	0	0	2	1	0	0	0	0	2	0	0	4	2234	0	0	4	579	0	0
182	0	0	0	0	0	22	235	8	0	0	1	22	0	0	1515	0	0	5	17	1	0	0
12	0	0	0	0	0	4	21	11	0	0	5	8	0	0	34	0	0	1034	4	0	0	0
0	0	0	0	0	0	0	6	0	0	0	0	19	0	0	13	3	0	3	684	29	0	0
8	0	0	0	0	0	8	0	0	0	11	23	0	0	0	11	1	0	6	20	3137	0	0
0	0	0	0	0	0	4	1	0	0	0	2	0	0	0	12	1	0	205	0	0	0	0
38	0	0	0	0	0	3	87	5	0	0	0	0	0	0	111	1	0	4	6	0	0	0

0.6327505111694336

categorical_hinge

4117	0	0	0	9	0	15	6	0	1	2	19	2	0	5	3	0	1	1	0	0	0	0
1	1017	0	0	0	0	1	2	0	1	1	973	2	0	0	1	0	1	1	0	0	0	0
2	3	617	1	1	0	0	1	0	0	0	0	0	0	1	13	0	3	1	0	0	0	0
8	0	1	669	0	0	0	5	4	0	0	0	2	0	1	2	5	5	2	344	18	0	0
26	1	1	0	2062	0	12	3	1	3	0	0	2	0	2	1	0	26	0	0	0	0	0
5	0	1	1	0	1059	0	0	0	0	5	0	22	0	0	2	14	3	45	0	15	0	0
52	0	1	0	4	0	5134	17	1292	7	2	5	14	1	8	28	1	2	6	0	1	0	0
52	0	1	0	16	0	19	3254	2	4	0	0	4	0	3	19	0	2	0	0	1	0	0
20	0	0	0	2	1	1635	16	2967	11	6	12	18	2	6	21	5	4	6	1	4	0	0
13	0	0	0	3	0	3	9	0	940	0	2	2	0	1	0	2	0	0	0	0	0	0
16	0	1	0	5	0	14	5	0	1	975	0	1	3	3	5	0	6	1	0	1	0	0
1228	7	0	0	5	0	6	2	0	0	0	1792	7	1	3	3	0	2	2	0	1	0	0
43	9	3	3	9	1	38	6	126	3	8	0	2531	6	5	37	1	68	6	0	7	0	0
2	3	0	0	0	0	1	0	0	7	0	0	4	676	1	0	1	4	1	0	7	0	0
12	0	0	0	4	0	10	3	0	1	0	0	2	0	1055	9	17	2	0	0	0	0	0
113	0	1	0	23	0	33	42	5	16	6	4	9	1	8	6132	1	39	1	0	1	0	0
1	0	0	3	0	0	2	0	1	0	1	0	4	1	4	2	2328	1	0	0	478	0	0
36	0	0	0	333	0	24	14	0	6	3	1	2	0	0	9	0	1577	3	0	0	0	0
21	1	0	0	2	5	8	37	9	6	0	4	12	0	1	21	0	50	955	0	1	0	0
9	0	1	347	1	0	0	1	17	0	0	0	2	12	1	3	3	4	3	334	19	0	0
5	104	1	4	1	3	0	7	5	1	3	0	11	8	7	6	0	7	2	0	3050	0	0
5	0	0	0	0	12	0	5	2	1	0	0	4	0	1	7	0	52	136	0	0	0	0
42	0	0	0	1	0	3	36	2	110	0	0	1	1	1	18	0	40	0	0	0	0	0

0.82268220118623352

logcosh

4119	0	2	0	8	0	10	4	0	3	1	22	1	0	5	3	0	2	1	0	0	0	0
1	1329	0	0	0	0	1	1	1	1	1	662	2	0	0	1	0	0	1	0	0	0	0
0	3	613	1	1	0	0	1	4	1	0	0	1	0	0	13	0	4	1	0	0	0	0
1	0	0	404	0	0	0	2	2	0	0	0	1	0	1	1	0	4	1	639	10	0	0
20	1	1	0	2077	0	4	2	0	3	1	0	4	0	2	1	0	24	0	0	0	0	0
6	0	0	1	0	1157	0	0	0	0	0	0	1	0	0	4	0	1	2	0	0	0	0
43	0	1	0	5	1	4009	10	2447	5	1	5	10	1	9	20	1	2	4	1	0	0	0
41	0	1	0	18	0	17	3262	2	4	0	0	2	1	4	21	0	2	2	0	0	0	0
17	0	0	0	2	2	795	11	3844	7	3	8	14	2	4	13	2	4	6	0	3	0	0
12	0	0	0	3	0	4	10	1	934	0	3	2	3	0	1	1	3	0	0	0	0	0
15	0	1	0	4	0	11	5	2	1	970	0	1	4	3	6	0						

kullback_leibler_divergence

4070	0	2	0	9	0	18	9	0	3	2	31	2	0	5	13	0	7	3	0	0	2	5
2	1126	0	0	0	1	1	4	5	1	1	849	3	0	0	5	0	1	2	0	0	0	0
0	1	617	1	1	0	0	0	0	1	0	0	0	0	2	10	0	5	1	0	0	0	0
0	0	0	299	0	0	0	1	6	0	0	0	2	0	1	2	1	4	1	741	8	0	0
22	1	1	0	2063	0	6	3	1	3	0	0	4	0	5	2	0	28	0	0	0	1	0
19	0	0	1	0	1134	0	0	1	0	0	0	2	0	0	2	0	1	1	0	0	11	0
25	0	1	1	4	0	4793	6	1689	4	0	4	12	0	7	19	2	3	1	1	0	1	2
37	0	3	0	18	0	18	3232	7	6	0	0	2	1	3	31	0	6	6	0	1	0	6
9	0	0	0	2	2	1312	8	3333	4	4	7	21	1	7	14	2	4	3	21	0	2	2
8	0	0	0	3	0	5	5	0	918	0	4	1	1	1	2	1	3	0	0	0	0	25
15	0	1	0	4	0	12	6	1	1	967	0	1	1	3	4	0	16	2	0	1	1	1
998	19	0	0	6	0	15	3	7	0	1	1975	5	1	3	17	0	2	2	0	4	0	1
25	8	2	3	7	1	30	5	150	1	4	3	2503	9	6	41	1	91	5	1	6	3	5
2	2	0	0	0	0	1	0	1	3	0	0	3	679	1	1	1	5	1	0	6	0	1
6	0	0	0	2	0	3	2	0	0	1	0	2	0	1073	7	16	2	0	0	1	0	0
82	0	1	0	22	1	34	34	13	11	5	4	7	1	10	6174	1	30	2	0	2	0	1
0	0	0	3	0	0	2	0	0	0	0	0	2	1	5	2	2171	3	0	0	637	0	0
32	0	0	0	310	0	19	7	0	2	4	1	2	2	1	9	0	1614	2	0	0	2	1
10	0	0	0	2	0	9	17	12	3	0	5	11	0	1	24	0	49	959	0	1	26	4
0	0	1	73	0	0	0	0	6	0	0	0	1	4	0	4	2	7	1	651	7	0	0
10	79	1	1	1	0	0	5	4	1	2	0	15	9	9	7	1	7	4	3	3066	0	0
0	0	0	0	1	0	0	1	1	1	0	0	1	1	0	6	8	0	0	0	204	0	0
12	0	0	0	1	0	2	3	1	18	0	0	0	1	1	8	0	4	0	0	0	0	204

0.8337931036949158

poisson

4078	0	2	0	9	0	16	10	0	3	3	25	2	0	5	12	0	6	2	0	0	2	6
2	1135	0	0	0	0	1	4	4	1	1	840	5	0	0	5	0	1	2	0	0	0	0
0	1	604	1	1	0	0	0	4	0	1	0	0	0	1	25	0	2	1	0	0	0	2
0	0	0	361	0	0	0	3	5	0	0	0	2	0	1	1	1	4	1	677	10	0	0
22	1	1	0	2071	0	3	3	0	3	1	0	4	0	5	3	0	23	0	0	0	0	0
9	0	0	1	0	1147	0	0	0	0	0	0	1	0	0	2	0	1	1	0	0	10	0
31	0	1	0	4	0	4992	11	1461	3	0	4	18	0	6	30	2	3	3	1	0	2	3
34	0	3	0	19	0	19	3248	4	3	0	0	2	0	4	30	0	2	5	0	0	0	4
11	0	0	0	1	2	1484	10	3147	3	3	7	28	1	3	20	2	5	4	0	3	0	3
11	0	0	0	3	0	4	5	1	910	0	2	2	1	1	2	1	4	0	0	0	0	30
9	0	1	0	5	0	11	3	1	1	987	0	1	1	3	4	0	7	0	0	1	1	1
1242	19	1	0	6	0	14	2	7	0	1	1727	7	1	3	18	0	2	4	0	4	0	1
26	8	2	3	8	1	29	5	102	1	5	2	2549	5	3	52	1	93	4	1	5	3	2
1	2	0	0	0	0	1	1	0	2	2	0	4	679	1	1	1	4	1	0	6	0	1
8	0	0	0	3	0	3	2	0	1	1	0	2	0	1056	7	31	1	0	0	0	0	0
85	0	1	0	21	2	29	38	8	9	6	3	8	1	11	6175	2	31	3	0	2	0	0
0	0	0	3	0	0	2	1	0	0	0	0	4	0	3	3	2308	2	0	0	500	0	0
37	0	0	0	410	0	19	14	0	4	4	0	2	1	1	13	0	1498	1	0	0	1	3
9	1	0	0	2	0	8	21	7	2	2	4	10	0	1	26	0	47	952	0	2	35	4
1	0	1	99	0	0	0	0	4	0	0	0	1	5	0	3	2	4	1	627	9	0	0
10	59	1	0	2	2	0	6	4	1	2	0	14	9	9	7	3	6	6	5	3079	0	0
0	0	0	0	1	0	0	2	1	1	0	0	1	0	1	2	0	7	8	0	0	201	0
11	0	0	0	1	0	3	4	0	16	0	0	0	0	1	9	0	4	0	0	0	0	206

0.8321188688278198

cosine_proximity

4137	0	0	0	7	0	7	2	0	1	0	14	1	0	4	3	0	1	1	0	0	0	3
0	1108	0	0	0	0	1	2	1	1	1	882	2	0	0	1	0	1	1	0	0	0	0
0	3	606	1	1	0	0	2	4	0	1	0	0	0	1	13	0	5	2	0	0	0	4
0	0	0	316	0	0	0	2	1	0	0	0	2	0	1	0	0	3	1	733	7	0	0
26	1	1	0	2064	0	5	1	0	3	1	0	1	0	2	3	0	31	0	0	0	1	0
9	0	0	1	0	1144	0	0	1	0	0	0	1	0	0	2	0	1	1	0	0	11	1
42	0	1	0	4	0	4724	15	1727	5	0	5	6	0	6	28	2	2	4	0	0	1	3
48	0	2	0	16	0	18	3244	3	5	0	0	1	1	3	24	0	4	4	0	1	1	2
16	0	0	0	2	1	1116	12	3521	5	4	11	12	2	2	17	2	5	4	0	3	1	1
12	0	0	0	3	0	3	3	1	923	1	2	0	2	1	1	1	3	0	0	0	0	21
15	0	1	0	4	0	12	5	1	1	977	0	0	3	1	6	0	7	2	0	1	0	1
1315	2	0	0	2	0	7	2	1	0	0	1719	1	0	3	3	0	1	1	0	1	0	1
29	9	2	3	7	1	19	7	128	4	3	0	2518	4	4	55	0	99	6	1	3	3	5
1	3	0	0	0	0	1	0	0	2	1	0	2	684	1	1	1	6	1	0	2	0	1
7	0	0	0	5	0	3	3	1	0	1	0	2	0	1070	9	11	2	0	0	0	0	1
82	0	0	0	24	0	34	38	9	12	2	3	5	1	9	6183	0	28	2	0	1	0	2
0	0	0	7	0	0	2	1	0	0	3	0	5	1	3	2	2270	1	1	1	529	0	0
34	0	0	0	212	0	21	9	0	3	4	0	1	0	0	8	0	0	0	0	0	2	0
11	0	0	0	2	0	6	20	9	4	0	5	10	0	1	24	0	31	967	0	1	37	5
0	0	1	83	1	0	0	0	0	0	0	0	1	6	0	2	2	3	2	652	4	0	0
10	90	1	4	1	1	0	7	0	0	2	0	11	17	9	7	0	7	6	5	3047	0	0
1	0	0	0	0	0	0	0	1	1	1	0	0	0	1	2	0	5	2	0	0	211	0
13	0	0	0	1	0	2	4	1	15	0	0	0	0	1	8	0	5	0	0	0	0	205

0.8371606469154358

Adadelta

mean_squared_error

4075	0	0	0	10	0	19	7	0	3	3	41	1	0	5	8	0	7	2	0	0	0	0
14	1144	0	0	0	0	1	2	2	1	1	827	4	0	0	2	0	2	1	0	0	0	0
7	2	586	1	1	0	0	13	4	1	1	2	0	0	1	15	0	8	1	0	0	0	0
1	0	0	355	0	0	0	4	5	0	0	0	2	1	1	3	2	2	1	680	9	0	0
24	1	0	0	2028	0	6	2	0	2	0	0	7	0	5	1	0	64	0	0	0	0	0
1	0	3	11	0	1034	0	1	0	0	10	0	25	0	0	2	27	5	23	7	23	0	0
31	0	0	0	2	0	4645	13	1825	2	0	6	21	0	5	15	4	4	0	1	1	0	0
45	0	1	0	18	0	15	3235	8	10	0	0	3	0	4	33	0	4	0	0	1	0	0
12	0	0	0	1	1	1325	10	3316	4	2	11	19	1	4	13	4	6	0	5	3	0	0
11	0	0	0	3	0	5	13	1	921	0	3	2	4	0	1	2	11	0	0	0	0	0
18	0	1	0	4	0	12	8	1	1	959	0	1	3	3	6	0						

squared_hinge

4052	0	0	0	13	0	33	22	0	13	2	0	1	0	0	24	0	14	1	0	6	0	0
1919	0	0	0	1	0	12	2	3	0	1	0	10	0	0	9	0	2	0	0	42	0	0
30	0	0	14	7	1	2	149	3	3	27	0	12	0	0	323	0	66	1	0	5	0	0
0	0	0	734	0	11	13	3	151	1	20	0	26	0	0	10	51	2	8	0	36	0	0
43	0	0	0	1953	0	10	5	0	0	3	0	4	0	0	3	2	116	0	0	1	0	0
0	0	0	16	0	969	0	1	0	0	0	0	2	0	0	6	102	2	19	0	55	0	0
19	0	0	1	3	0	5626	1	896	0	3	0	12	0	0	1	5	3	2	0	3	0	0
76	0	0	0	14	1	37	3142	11	18	12	0	3	0	0	42	2	15	3	0	1	0	0
12	0	0	11	1	0	2671	4	1995	6	3	0	19	0	0	3	4	4	1	0	3	0	0
18	0	0	0	5	1	4	46	10	833	8	0	1	0	0	8	1	42	0	0	0	0	0
31	0	0	19	7	0	16	58	3	5	753	0	2	0	0	30	8	87	9	0	9	0	0
2986	0	0	0	5	0	22	3	2	2	2	0	14	0	0	13	0	1	1	0	8	0	0
24	0	0	33	31	1	106	6	175	20	35	0	2244	0	0	23	1	175	12	0	24	0	0
1	0	0	224	0	0	4	4	23	196	25	0	13	0	0	10	14	92	33	0	68	0	0
14	0	0	1	9	0	10	13	0	0	59	0	1	0	0	24	877	8	0	0	99	0	0
97	0	0	8	48	4	36	86	19	13	125	0	8	0	0	5753	19	202	5	0	12	0	0
0	0	0	8	0	0	2	1	6	0	9	0	6	0	0	2	2377	1	0	0	414	0	0
44	0	0	0	390	0	26	17	0	6	6	0	6	0	0	27	0	1482	2	0	2	0	0
18	0	0	4	2	7	8	65	18	5	25	0	7	0	0	28	1	79	862	0	4	0	0
0	0	0	467	0	6	5	0	167	0	12	0	25	0	0	3	37	6	2	0	27	0	0
32	0	0	29	1	8	0	9	5	1	41	0	20	0	0	13	101	7	16	0	2942	0	0
2	0	0	7	0	1	2	1	1	1	0	0	2	0	0	10	4	53	139	0	3	0	0
26	0	0	1	2	0	5	57	4	121	1	0	0	0	0	10	1	27	0	0	0	0	0

0.6795342564582825

categorical_hinge

4143	0	0	0	6	0	12	1	0	0	1	16	0	0	1	0	0	0	1	0	0	0	0
0	864	0	0	0	0	0	0	0	0	0	1136	0	0	0	0	0	1	0	0	0	0	0
0	3	623	1	0	0	1	1	4	0	1	0	0	0	0	1	0	5	1	2	0	0	0
1	0	0	175	0	0	1	0	0	0	0	0	1	0	0	0	0	3	1	877	7	0	0
17	1	1	0	2098	0	9	1	0	0	1	0	0	0	0	0	0	12	0	0	0	0	0
0	0	0	1	0	1165	0	0	0	0	0	0	0	0	0	0	0	2	4	0	0	0	0
79	0	0	0	3	1	5068	16	1357	5	3	5	7	0	4	15	1	5	5	1	0	0	0
39	0	1	0	12	1	25	3281	1	0	1	0	0	1	1	9	0	2	1	0	2	0	0
49	0	0	0	2	5	1714	26	2851	2	5	14	30	1	4	10	1	9	10	0	4	0	0
20	0	0	0	4	0	4	24	1	892	3	2	1	12	1	1	1	10	1	0	0	0	0
10	0	1	0	3	0	14	4	0	0	995	0	0	1	1	1	0	5	1	0	1	0	0
1237	0	0	0	2	0	10	1	0	0	0	1804	0	0	1	0	0	3	1	0	0	0	0
49	7	5	3	11	2	31	7	36	0	8	3	2573	5	5	22	0	128	9	1	5	0	0
0	2	0	0	0	0	1	0	0	2	3	0	4	677	1	0	0	12	2	1	2	0	0
9	0	0	0	6	0	10	5	0	1	2	0	0	0	0	1067	1	7	4	1	0	0	
128	0	1	0	22	2	44	54	6	7	12	4	5	1	10	6038	0	95	1	4	1	0	0
1	0	0	2	0	0	2	0	0	0	1	0	0	0	1	1	2402	0	2	0	414	0	0
30	0	0	0	251	0	23	10	0	1	3	1	0	1	0	3	0	1683	2	0	0	0	0
16	0	0	0	2	4	7	45	4	1	2	4	8	1	1	6	0	50	979	0	3	0	0
3	0	1	63	0	0	0	0	0	0	0	0	1	5	0	1	1	4	2	672	4	0	0
4	88	2	1	0	1	0	8	0	1	1	2	6	11	7	4	0	13	6	10	3060	0	0
4	0	0	0	1	101	0	14	1	1	0	0	3	0	1	1	0	35	63	0	0	0	0
28	0	0	0	3	0	4	33	0	41	1	0	2	2	2	10	0	125	4	0	0	0	0

0.820189893245697

logcosh

3645	19	0	0	15	1	28	23	1	14	6	330	10	0	18	49	0	15	6	0	1	0	0
114	741	0	0	2	0	3	2	0	2	2	1081	18	0	6	9	0	2	5	0	14	0	0
16	4	0	17	21	0	5	121	19	14	42	3	75	0	30	163	0	96	0	0	17	0	0
0	8	0	442	0	10	19	1	208	0	18	0	187	0	2	30	42	2	12	0	85	0	0
61	1	0	0	1931	0	2	8	0	0	1	3	6	0	16	15	0	96	0	0	0	0	0
0	1	0	3	0	963	0	3	0	0	0	0	5	0	0	21	115	3	17	0	41	0	0
15	4	0	2	3	0	5071	28	1321	4	7	22	50	0	10	22	3	10	3	0	0	0	0
138	0	0	0	13	2	32	3075	8	11	5	2	2	0	6	54	0	25	3	0	1	0	0
14	12	0	10	3	1	2216	19	2281	10	11	20	80	0	10	33	6	9	2	0	0	0	0
51	1	0	0	6	1	5	87	5	746	3	2	2	0	0	17	0	50	1	0	0	0	0
120	1	0	0	9	1	14	88	3	4	576	5	3	1	16	76	6	89	12	0	13	0	0
1035	195	0	0	9	4	10	12	1	5	8	1687	38	0	12	19	0	6	6	0	12	0	0
26	35	0	37	40	4	68	7	203	16	34	13	2249	0	8	65	0	77	8	0	20	0	0
2	10	0	75	0	0	2	5	3	284	20	0	30	1	2	23	20	69	34	0	127	0	0
9	8	0	0	3	0	4	7	0	0	2	2	0	0	1001	15	58	4	0	0	2	0	0
110	6	0	1	34	3	37	79	15	8	58	11	14	0	22	5899	16	110	6	0	6	0	0
2	4	0	3	0	1	3	1	13	0	3	0	22	0	26	8	2229	1	1	0	509	0	0
68	0	0	0	429	0	11	24	0	4	3	4	13	0	7	63	0	1380	1	0	1	0	0
21	2	0	1	2	10	5	78	12	4	29	8	13	0	0	25	4	113	800	0	6	0	0
0	4	0	248	1	8	14	2	206	0	10	0	139	0	2	21	30	7	4	0	61	0	0
28	144	0	15	1	19	1	9	15	1	22	8	26	0	59	27	370	7	9	0	2464	0	0
4	0	0	0	0	2	0	3	0	1	0	1	3	0	3	11	4	59	131	0	3	0	0
79	0	0	0	1	0	3	51	0	84	0	0	0	0	1	14	0	21	1	0	0	0	0

0.7073876261711121

categorical_crossentropy

4171	0	0	0	1	0	0	5	0	0	2	0	0	0	1	0	0	1	0	0	0	0	0
0	865	0	0	0	0	6	2	5	2	1	1105	1	0	0	10	0	1	2	0	0	1	0
0	3	600	1	0	0	1	4	0	0	0	0	0	0	0	25	0	7	1	0	0	0	1
0	0	0	125	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	932	6	0	0
36	0	1	0	2053	0	6	14	0	2	1	0	0	0	5	8	0	11	0	0	0	0	3
0	0	0	1	0	1157	0	10	1	0	0	0	0	0	0	0	0	1	0	0	0	11	1
45	0	0	0	3	0	5269	16	1195	4	0	1	0	0	7	26	1	2	2	0	0	0	4
32	0	1	0	4	0	6	3321	0	0	0	0	0	1	1	7	0	2	1	0	0	0	1
13	0	0	0	2	0	1744	12	2930	4	1	6	1	0	1	15	0	3	0	0	2	0	3
25	0	0	0	3	0	2	5	1	903	0	1	0	0	1	3	0	1	0	0	0		

poisson

4132	0	0	0	5	0	9	4	0	1	2	12	1	0	5	2	0	3	2	0	0	1	2
7	853	0	0	0	0	6	3	2	1	2	1117	2	0	0	5	0	1	2	0	0	0	0
0	3	616	1	0	0	0	0	4	0	1	0	0	0	8	0	7	1	0	0	0	0	2
0	0	0	512	0	0	0	3	1	0	0	0	2	0	0	2	0	3	1	535	7	0	0
27	1	1	0	2060	0	3	0	0	2	1	0	2	0	4	3	0	35	0	0	0	0	1
26	0	0	1	0	1131	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	11	0
41	0	1	1	3	0	5104	6	1357	3	0	4	11	0	6	25	2	5	1	0	0	2	3
54	0	2	0	16	0	19	3228	1	1	0	1	2	3	25	0	14	2	0	0	3	6	
12	0	0	0	2	2	1636	7	3011	4	3	9	15	1	4	13	3	4	5	0	3	0	3
10	0	0	0	3	0	3	3	1	905	0	3	0	2	1	2	1	7	0	0	0	0	36
17	0	1	0	3	0	11	5	1	1	954	0	1	1	3	10	0	23	1	0	2	2	1
1685	3	0	0	3	0	16	1	4	0	1	1326	2	0	3	9	0	3	1	0	1	0	1
64	6	4	3	8	0	27	4	81	3	2	1	2501	5	6	43	1	129	5	1	5	7	4
2	2	0	0	0	0	1	0	0	2	0	0	2	674	1	1	1	11	0	0	9	0	1
7	0	0	0	1	0	1	1	0	1	0	0	2	0	1078	2	18	4	0	0	0	0	0
116	0	1	0	18	0	32	25	6	8	2	2	5	1	9	6172	1	34	1	0	1	0	1
1	0	0	5	0	0	2	0	0	0	0	0	0	0	5	2	2245	2	1	0	563	0	0
35	0	0	0	208	0	17	5	0	1	2	0	2	1	0	7	0	1726	2	0	0	2	0
12	0	0	0	1	0	7	8	8	3	1	4	10	0	1	11	1	63	960	0	2	36	5
2	0	1	153	0	0	0	0	0	0	0	0	1	5	0	3	1	6	2	573	9	0	1
10	60	1	2	1	2	0	10	2	0	0	1	14	9	9	14	1	14	5	3	3067	0	0
2	0	0	0	1	0	0	0	1	1	0	0	0	0	1	1	0	6	2	0	0	210	0
12	0	0	0	1	0	3	2	0	11	0	0	0	1	1	6	0	7	0	0	0	0	211

0.8228344321250916

cosine_proximity

4136	0	0	0	7	0	1	7	0	1	2	14	1	0	2	4	0	1	4	0	0	0	1
0	1455	0	0	0	0	1	0	0	2	1	539	1	0	0	0	0	1	1	0	0	0	0
0	3	614	1	0	0	0	1	7	0	0	0	0	0	11	0	4	2	0	0	0	0	0
0	0	0	283	0	0	0	0	0	0	0	0	1	0	0	0	0	1	779	2	0	0	0
17	1	1	0	2101	0	2	4	0	4	0	0	0	0	1	0	7	0	0	0	1	1	1
0	0	0	1	0	1155	0	0	0	0	0	0	2	0	0	0	1	1	0	0	0	12	0
34	0	0	0	6	0	5156	17	1287	14	3	8	9	0	8	23	1	2	3	0	1	1	2
27	0	1	0	10	0	8	3312	0	0	0	1	1	1	10	0	2	3	0	0	1	0	0
5	0	0	0	2	0	1539	12	3114	10	4	14	14	1	4	10	0	2	1	0	2	0	3
13	0	0	0	3	0	0	3	1	951	0	2	0	0	0	0	0	0	0	0	0	0	4
6	0	1	0	4	0	4	3	2	1	1004	0	1	0	1	3	0	4	1	0	2	0	0
726	1	0	0	2	0	8	3	1	0	1	2301	3	0	2	4	0	3	3	0	0	0	1
6	7	4	1	8	0	9	4	48	10	2	1	2707	3	7	29	0	45	8	0	0	5	6
0	1	0	0	1	0	1	0	0	12	0	0	6	681	1	1	0	2	0	0	0	0	1
4	0	0	0	5	0	1	4	0	3	0	0	0	0	1086	1	4	5	1	0	1	0	0
56	0	1	0	17	0	29	43	9	16	4	8	7	0	11	6201	0	26	1	1	1	0	4
0	0	0	2	0	0	2	0	0	0	0	2	0	2	2	2548	1	1	0	0	266	0	0
28	0	0	0	299	0	16	10	1	7	3	1	1	0	1	6	0	1631	1	0	0	2	1
8	0	0	0	2	0	9	13	5	8	1	4	15	0	3	10	0	17	1011	0	1	19	7
0	0	1	91	0	0	0	0	0	0	0	0	1	5	0	1	2	3	2	650	1	0	0
1	134	1	3	0	0	0	8	9	2	0	0	126	26	15	6	1	8	14	12	2856	1	2
0	0	0	0	1	0	0	2	1	1	0	0	0	0	1	0	0	2	2	0	0	215	0
7	0	0	0	2	0	0	3	0	25	0	0	0	0	1	3	0	1	0	0	0	0	213

0.8633968234062195

Adam

mean_squared_error

4143	0	0	0	5	0	0	6	0	0	2	14	2	0	1	0	0	4	4	0	0	0	0
0	1439	0	0	0	1	1	0	1	2	0	553	1	0	0	0	0	1	2	0	0	0	0
0	3	542	0	0	0	0	4	8	0	0	0	1	0	4	2	0	71	8	0	0	0	0
0	0	0	181	0	0	0	1	0	0	0	0	1	1	0	0	0	5	1	868	8	0	0
18	0	1	0	2094	0	2	4	0	2	1	0	0	0	0	0	17	0	0	0	0	1	1
0	0	0	1	0	1149	0	0	4	0	0	0	0	0	0	1	4	0	0	1	11	0	0
33	0	0	0	6	0	3961	19	2506	5	5	8	10	1	8	2	0	4	5	0	0	1	1
30	0	0	0	8	0	0	3326	0	0	0	1	1	0	5	0	3	2	0	0	1	0	0
4	0	0	0	3	0	798	13	3830	4	2	14	35	3	2	2	1	14	7	0	1	3	1
17	0	0	0	3	0	2	6	0	925	1	2	0	6	1	0	0	5	0	0	0	0	9
6	0	0	0	3	0	3	2	0	1	1010	0	1	0	2	1	0	7	1	0	2	0	0
642	0	0	0	2	0	6	2	1	0	1	2392	3	0	3	0	0	4	2	0	0	0	1
6	7	1	0	3	0	6	5	72	2	1	2	2566	9	8	9	0	194	14	1	2	1	1
0	1	0	0	0	0	1	0	0	2	0	0	3	684	1	0	0	14	0	0	0	0	1
4	0	0	0	4	0	1	5	1	1	0	0	4	0	1066	1	17	7	0	0	4	0	0
62	0	1	0	17	0	33	80	30	10	6	16	26	1	29	5519	0	588	4	2	4	1	6
0	0	0	2	0	0	2	0	0	0	0	0	1	1	3	0	2383	4	1	0	429	0	0
23	0	0	0	126	0	14	8	0	1	3	1	2	2	0	1	0	1824	1	0	0	2	0
8	0	0	0	2	0	4	9	3	3	1	4	11	1	1	2	0	50	1023	0	0	9	2
0	0	1	39	0	0	0	0	0	0	0	0	1	6	1	0	0	9	2	696	2	0	0
1	85	1	1	1	0	0	8	3	0	0	1	58	35	10	2	0	33	12	8	2966	0	0
0	0	0	0	0	0	0	1	1	0	0	0	1	0	2	0	0	8	23	0	0	189	0
7	0	0	0	1	0	0	6	1	7	0	0	0	1	2	2	0	24	0	0	0	1	203

0.8392344117164612

mean_squared_logarithmic_error

4153	0	0	0	6	0	0	3	0	0	2	5	1	0	1	7	0	0	3	0	0	0	0
0	1312	0	0	0	1	1	0	1	2	1	671	2	0	0	5	0	1	4	0	0	0	0
0	3	578	0	0	0	0	1	1	0	2	0	0	0	0	53	0	3	1	0	0	0	1
0	0	0	150	0	0	0	0	0	0	0	0	2	0	0	4	0	3	1	895	11	0	0
17	1	1	0	2080	0	2	5	0	2	2	0	4	0	0	5	0	19	0	0	0	1	1
0	0	0	0	0	1150	0	0	0	0	0	0	0	0	0	0	1	2	0	0	0	19	0
43	0	0	0	6	0	4590	17	1806	6	16	3	17	0	8	51	0	2	3	0	0	1	6
33	0	1	0	8	0	6	3297	0	0	0	0	0	1	1	19	0	2	5	0	0	1	3
11	0	0	0	2	0	1025	10	3580	4	8	9	34	0	1	35	0	3	5	0	2	3	5
18	0	0	0	3	0	2	3	1	913	1	1	0	0	1	5	0	1	1	0	0	1	26
6	0	0	0	1	0	1	1	0	0	1021	0	0	0	1	2	0	2	1	0	1	0	0

categorical_hinge

4132	0	0	0	7	0	12	4	0	0	2	17	0	0	1	3	0	0	3	0	0	0	0
0	1540	0	0	0	0	0	0	0	0	0	461	0	0	0	0	0	0	0	0	0	0	0
0	3	576	0	0	0	0	1	7	0	0	0	1	0	0	37	0	7	9	1	0	0	1
0	0	0	194	0	0	0	1	0	0	0	0	2	0	0	2	0	2	4	851	10	0	0
17	1	1	0	2071	0	13	1	0	2	0	0	0	0	0	5	0	27	0	0	0	2	0
1	0	0	0	0	1142	0	0	1	0	0	0	0	0	0	0	1	4	1	0	0	21	1
35	0	0	0	4	0	5291	13	1185	5	0	7	4	0	6	17	1	2	2	0	0	1	2
35	0	1	0	10	0	25	3285	0	1	0	0	1	1	12	0	2	3	0	0	1	0	0
10	0	0	0	2	0	1567	12	3077	3	1	15	19	0	1	10	0	1	8	0	3	5	3
15	0	0	0	3	0	5	2	0	939	0	2	0	0	0	2	0	0	0	0	0	0	9
7	0	1	0	4	0	14	3	2	2	979	0	0	0	1	10	0	8	4	0	1	1	0
742	1	0	0	2	0	10	1	0	0	0	2297	0	0	2	2	0	1	0	0	0	0	1
8	9	3	1	3	0	13	2	45	3	3	1	2674	1	3	41	0	47	18	0	0	12	23
0	6	0	0	0	0	1	0	0	5	0	0	7	673	1	1	0	5	4	1	1	0	2
7	0	0	0	6	0	14	3	0	3	0	0	0	0	1067	4	4	5	1	0	1	0	0
58	0	1	0	12	0	38	21	5	9	1	9	8	0	6	6242	0	17	2	0	1	3	2
0	0	0	5	0	0	2	0	0	0	0	0	2	0	1	4	2194	0	1	0	616	1	0
28	0	0	0	98	0	25	9	1	5	2	1	1	0	0	7	0	1826	3	0	0	2	0
9	0	0	0	2	0	7	4	4	5	1	4	7	0	1	12	0	14	1026	0	1	32	4
0	0	1	49	0	0	0	0	0	0	0	0	2	6	0	1	0	5	7	680	4	0	2
1	248	1	0	0	0	0	4	1	2	0	0	18	13	6	9	0	7	15	7	2885	6	2
0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	0	1	1	0	0	217	1
8	0	0	0	1	0	4	3	0	16	0	0	0	0	1	6	0	2	0	0	0	0	214

0.8603527545928955

logcosh

4169	0	0	0	6	0	0	4	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
1	646	0	0	2	1	1	0	0	2	1	1339	3	0	2	1	0	0	2	0	0	0	0
4	3	577	0	1	0	0	3	5	3	3	0	1	0	5	25	0	11	2	0	0	0	0
0	0	0	335	0	0	0	0	0	0	0	0	6	6	0	0	10	1	2	691	15	0	0
17	0	1	0	2117	0	0	0	0	2	0	0	0	0	0	0	3	0	0	0	0	0	0
0	0	0	1	0	1157	0	0	0	0	0	0	3	0	0	0	1	1	0	0	0	9	0
74	0	0	0	7	0	5634	17	760	29	7	1	9	1	9	22	0	2	1	0	1	0	1
37	0	1	0	13	0	4	3313	0	0	0	0	1	1	5	0	1	1	0	0	0	0	0
32	0	0	0	2	0	2208	12	2394	19	3	3	34	3	5	9	5	2	1	0	5	0	0
20	0	0	0	4	0	1	3	0	943	0	1	0	4	0	0	0	0	0	0	0	0	1
7	0	0	0	5	0	3	2	0	1	1012	0	0	1	1	1	0	1	1	0	2	0	0
2489	0	0	0	2	0	3	2	0	0	1	559	0	0	2	1	0	0	0	0	0	0	0
30	6	3	1	12	0	8	5	34	18	2	1	2658	10	18	33	8	43	6	0	12	1	1
0	0	0	0	1	0	1	0	0	5	0	0	3	694	1	0	0	2	0	0	0	0	0
6	0	0	0	7	0	2	4	0	1	0	0	0	0	1085	1	7	1	0	0	1	0	0
87	0	0	0	28	0	33	48	5	12	2	0	7	1	14	6172	0	21	1	1	1	0	2
1	0	0	1	0	0	2	0	0	0	0	0	0	0	1	1	2551	0	0	0	269	0	0
31	0	0	0	929	0	17	10	0	7	3	0	0	2	1	6	0	999	2	0	0	1	0
18	0	0	0	2	1	9	21	5	11	1	1	13	4	2	14	3	21	995	0	6	4	2
0	0	1	96	0	1	0	0	0	0	0	0	9	14	0	1	13	3	2	609	8	0	0
3	2	1	0	1	0	0	4	1	3	0	0	26	35	12	3	0	5	5	4	3120	0	0
1	0	0	0	0	0	1	4	0	1	0	0	2	5	5	1	0	8	50	0	0	147	0
10	0	0	0	5	0	1	8	0	38	0	0	1	1	2	6	0	1	0	0	1	0	181

0.8003462553024292

categorical_crossentropy

4163	0	0	0	1	0	0	3	0	0	2	6	0	0	1	2	0	3	0	0	0	0	0
0	1257	0	0	1	1	3	0	1	2	2	706	2	0	0	22	0	0	2	0	0	1	1
0	3	584	0	0	0	1	3	0	2	0	0	1	0	0	40	0	8	1	0	0	0	0
0	0	0	176	0	0	0	0	1	0	1	0	1	0	0	1	0	1	1	869	15	0	0
29	0	1	0	2076	0	5	3	0	1	1	0	0	3	10	0	10	0	0	0	0	0	1
0	0	0	1	0	1157	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	10	0
41	0	0	0	5	0	5579	10	868	2	9	2	3	0	7	40	1	2	4	0	0	0	2
36	0	2	0	6	0	14	3279	1	0	3	0	0	1	1	23	0	9	1	0	0	0	1
12	0	0	0	2	0	2142	10	2499	3	7	7	12	0	1	30	0	5	2	0	2	1	2
31	0	0	0	3	0	4	4	1	879	1	1	0	11	1	7	0	8	0	0	0	0	26
5	0	1	0	1	0	2	0	0	0	1021	0	0	0	1	1	0	2	1	0	2	0	0
1342	0	0	0	2	0	17	1	1	0	1	1667	2	0	2	18	0	2	1	0	0	0	3
23	7	3	1	7	0	9	2	43	0	13	1	2631	3	4	85	0	58	8	0	3	4	5
1	0	0	0	0	0	1	0	0	1	0	0	9	682	1	3	0	5	0	1	2	0	1
10	0	0	0	3	0	2	4	0	0	1	0	1	0	1073	6	8	4	1	0	2	0	0
52	0	0	0	8	0	23	12	6	1	2	2	2	0	6	6299	0	19	1	0	1	0	1
1	0	0	4	0	0	2	0	0	0	0	0	1	0	0	3	2262	0	0	0	553	0	0
30	0	0	0	106	0	14	4	0	2	3	1	0	1	0	10	0	1834	2	0	0	1	0
14	0	0	0	2	0	8	7	7	2	2	1	11	1	1	25	0	27	959	0	7	54	5
0	0	1	57	0	0	0	0	0	0	0	0	2	5	0	2	0	2	2	678	8	0	0
3	113	1	0	0	0	0	1	2	0	4	0	62	10	8	32	0	10	6	4	2968	1	0
0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	1	3	0	0	218	0
9	0	0	0	2	0	0	0	1	2	1	0	0	1	1	11	0	1	0	0	1	0	225

0.8402808308601379

kullback_leibler_divergence

4151	0	0	0	1	0	0	7	0	0	2	7	1	0	1	2	0	1	5	0	0	0	3
0	1253	0	0	0	0	1	0	1	2	0	712	2	0	0	4	0	1	24	0	0	1	0
0	3	586	0	0	0	1	5	0	0	0	0	1	0	0	25	0	8	11	0	0	1	2
0	0	0	122	0	0	0	0	1	0	0	0	3	1	1	1	0	0	19	896	21	0	1
27	0	1	0	2068	0	6	7	0	2	1	0	2	0	5	6	0	10	0	0	0	1	4
0	0	1	0	0	1154	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	15	0
41	0	0	0	5	0	5019	15	1396	8	4	5	10	1	8	31	0	2	23	0	0	2	5
27	0	2	0	6	0	5	3317	0	1	0	0	0	1	1	8	0	2	3	0	0	1	3
8	0	0	0	2	0	1363	10	3197	6	2	11	41	1	2	17	0	4	62	0	0	5	6
19	0	0	0	2	0	1	2	0	921	0	1	0	4	1	1	0	0	1	0	0	0	24
6	0	1	0	2	0	2	3	0	2	1007	0	0	0	1	3	0	6	1	0	1	1	1
1074																						

cosine_proximity

4154	0	0	0	7	0	0	6	0	0	2	7	0	0	1	2	0	0	2	0	0	0	0
0	1107	0	0	0	0	1	0	1	1	1	888	0	0	0	1	0	1	0	0	0	0	0
0	3	602	0	0	0	0	2	3	0	3	0	0	0	17	0	12	1	0	0	0	0	0
0	0	0	302	0	0	0	0	0	0	1	0	2	1	0	1	0	5	1	748	5	0	0
15	0	1	0	2097	0	2	5	0	2	1	0	0	0	0	1	0	14	0	0	0	1	1
0	0	0	1	0	1156	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	13	0
41	0	0	0	6	0	5471	19	998	4	7	4	7	0	8	5	0	2	1	0	0	1	1
29	0	0	0	8	0	6	3320	0	0	0	0	0	1	0	8	0	3	1	0	0	1	0
10	0	0	0	2	0	1846	13	2801	4	6	12	23	1	1	9	1	5	0	0	0	2	1
17	0	0	0	3	0	3	9	1	886	2	2	0	8	2	4	0	3	0	0	0	0	37
5	0	0	0	2	0	3	1	0	0	1021	0	0	0	1	1	0	2	0	0	1	0	0
1115	0	0	0	2	0	8	3	1	0	1	1922	0	0	3	1	0	2	0	0	0	0	1
6	7	3	1	6	0	9	5	60	0	11	1	2651	7	9	33	0	77	9	0	0	7	8
0	1	0	0	1	0	1	1	0	0	2	0	8	684	1	0	0	7	0	0	0	0	1
4	0	0	0	6	0	3	5	1	0	3	0	0	0	1082	1	5	5	0	0	0	0	0
57	0	1	0	18	0	39	51	9	2	16	6	8	1	9	6152	0	59	1	0	1	0	5
0	0	0	2	0	0	2	1	0	0	2	0	1	1	2	2	2456	0	0	0	357	0	0
26	0	0	0	206	0	16	10	0	2	4	1	0	1	0	3	0	1736	0	0	0	2	1
8	0	0	0	2	0	9	31	7	2	6	4	11	2	1	20	0	35	869	0	0	121	5
0	0	1	121	0	0	0	0	0	0	0	0	1	5	0	1	2	5	2	617	2	0	0
2	85	2	3	1	0	0	14	4	0	6	0	94	43	29	24	0	24	5	11	2876	2	0
0	0	0	0	0	0	1	2	0	0	0	0	1	0	1	1	0	0	1	0	0	218	0
8	0	0	0	2	0	1	7	1	4	1	0	0	1	1	5	0	4	0	0	0	0	220

0.8447327613830566

Adamax

mean_squared_error

4164	0	0	0	2	0	0	7	0	0	2	5	0	0	0	0	0	1	0	0	0	0	0
0	1088	0	0	0	0	1	2	3	5	2	883	5	0	1	2	0	2	4	0	1	1	1
1	3	563	0	0	0	0	6	7	2	7	0	0	1	1	39	0	10	1	0	0	0	2
0	0	0	418	0	0	0	2	0	0	1	0	2	3	0	1	1	3	1	619	15	0	0
23	0	1	0	2081	0	2	10	0	6	3	0	0	0	0	1	0	12	0	0	0	0	1
0	0	0	1	0	1157	0	0	0	1	0	0	0	0	0	0	1	2	0	0	0	10	0
47	0	0	0	6	0	5119	20	1304	14	14	2	3	2	7	26	1	2	2	0	1	1	4
30	0	1	0	7	0	0	3327	0	0	0	0	0	1	0	8	0	1	1	0	0	1	0
13	0	0	0	2	0	1478	16	3164	12	8	7	9	2	1	10	3	2	1	0	5	1	3
15	0	0	0	3	0	1	3	0	946	0	1	0	2	0	0	0	0	0	0	0	0	6
6	0	0	0	1	0	2	2	0	1	1019	0	0	0	0	1	0	2	1	0	2	0	0
1798	0	0	0	2	0	6	8	0	1	1	1229	0	0	2	5	0	3	1	0	0	0	3
9	7	3	2	4	0	7	9	56	24	35	0	2570	11	5	53	4	78	6	0	9	2	16
0	0	0	0	0	0	1	0	0	5	2	0	2	692	1	0	0	1	0	0	1	0	2
7	0	0	0	5	0	2	6	0	4	4	0	0	0	1042	1	36	5	1	0	2	0	0
64	0	1	0	13	0	27	60	9	12	11	2	2	1	8	6184	0	32	1	0	1	0	7
0	0	0	1	0	0	2	0	0	0	2	0	0	0	1	1	2444	0	0	0	375	0	0
29	0	0	0	149	0	17	11	0	7	4	1	1	1	0	6	0	1778	0	0	0	2	2
11	0	0	0	2	0	9	50	8	10	3	1	12	4	1	14	0	31	924	0	9	28	16
0	0	1	145	0	0	0	1	0	0	0	0	3	14	0	1	4	3	2	574	8	0	1
2	6	1	2	0	0	0	8	1	3	7	0	12	28	5	3	0	10	4	3	3128	0	2
0	0	0	0	0	0	1	1	0	0	0	0	1	0	1	0	0	6	2	0	0	210	3
7	0	0	0	2	0	0	3	0	18	0	0	0	1	1	3	0	1	0	0	0	0	219

0.8378835916519165

mean_squared_logarithmic_error

4171	0	0	0	2	0	0	1	0	0	1	5	0	0	1	0	0	0	0	0	0	0	0
0	1355	0	0	0	0	0	0	1	1	0	643	0	0	0	0	0	1	0	0	0	0	0
0	3	624	0	0	0	0	1	2	0	0	0	0	0	6	0	6	1	0	0	0	0	0
1	0	0	258	0	0	0	0	0	0	0	1	1	0	1	0	1	1	800	2	0	0	0
23	1	1	0	2069	0	3	5	0	2	1	0	0	0	1	0	32	0	0	0	1	1	1
0	0	1	1	0	1157	0	0	0	0	0	0	0	0	0	0	1	1	0	0	11	0	0
44	0	0	0	6	0	3645	18	2801	6	6	6	4	0	7	22	1	2	3	0	1	1	2
37	0	1	0	9	0	5	3308	0	0	0	0	0	1	0	10	0	4	1	0	0	1	0
12	0	0	0	2	0	551	13	4115	3	4	12	4	1	2	12	0	2	1	0	1	0	2
19	0	0	0	3	0	1	3	1	919	0	1	0	3	0	1	0	4	0	0	0	0	22
9	0	1	0	2	0	2	4	0	1	1007	0	0	0	1	1	0	6	1	0	2	0	0
1196	0	0	0	2	0	5	2	0	0	1	1845	0	0	2	1	0	3	1	0	0	0	1
15	7	3	2	3	0	7	4	92	1	6	1	2563	6	7	57	0	106	9	1	4	4	12
0	1	0	0	0	0	1	0	0	1	0	0	4	686	1	1	0	11	0	0	0	0	1
6	0	0	1	5	0	2	5	1	1	0	0	0	0	1081	2	1	7	0	0	2	0	1
71	0	1	0	12	0	27	33	13	8	4	5	4	1	7	6191	0	52	0	0	1	0	5
0	0	0	6	0	0	2	0	0	0	0	0	1	1	1	2	2249	2	1	0	561	0	0
29	0	0	0	83	0	16	6	0	3	3	1	1	1	0	5	0	1857	0	0	0	1	2
11	0	0	0	2	0	4	10	8	2	1	3	8	3	1	13	0	51	994	0	3	12	7
0	0	1	39	0	0	0	0	0	0	0	1	5	0	1	0	3	1	705	1	0	0	0
5	165	2	3	0	0	0	3	2	1	0	0	25	26	3	5	0	15	5	13	2949	0	3
0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	0	0	8	9	0	0	203	1
8	0	0	0	1	0	0	3	1	6	0	0	0	1	1	3	0	5	0	0	0	0	226

0.8404901027679443

squared_hinge

4159	0	0	0	5	0	1	4	0	0	2	7	0	0	1	1	0	1	0	0	0	0	0
0	1172	0	0	0	0	1	0	1	2	2	821	1	0	0	0	1	0	0	0	0	0	0
0	3	551	1	0	0	0	1	6	0	6	0	0	0	5	53	0	10	3	3	0	0	1
0	0	0	2	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1059	2	0	0	0
18	0	1	0	2107	0	2	1	0	2	1	0	0	0	0	0	0	7	0	0	0	0	1
0	0	0	1	0	1163	0	0	0	0	0	0	1	0	0	0	1	5	0	0	0	0	1
38	0	0	0	6	0	5076	15	1403	4	4	3	0	0	6	16	1	1	1	0	0	0	1
34	0	1	0	10	0	11	3305	1	0	0	0	1	1	10	0	2	1	1	0	0	0	0
11	0	0	0	3	0	1316	12	3360	3	5	8	3	0	1	9	0	2	0	0	3	0	1
18	0	0	0	4	0	4	3	0	913	2	1	0	6	1	2	0	1	0	0	0	0	22
6	0	0	0	1	0	5	1	2	0	1016	0	0	0	1	1	0	2	0	0	2	0	0

Nadam

mean_squared_error

4173	0	0	0	1	0	0	3	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0
0	774	0	0	0	1	1	0	1	2	0	1215	2	0	2	1	0	0	2	0	0	0	0
0	2	625	0	0	0	0	1	1	0	0	0	0	0	0	13	0	0	1	0	0	0	0
0	0	5	234	0	0	0	1	0	0	0	0	1	0	0	3	0	0	1	812	9	0	0
30	0	2	0	2034	0	3	13	0	4	2	0	0	0	2	39	0	11	0	0	0	0	0
0	0	2	1	0	1140	0	1	2	0	0	3	0	0	4	1	0	2	0	4	12	0	0
50	0	0	0	5	0	5445	18	1001	5	1	2	1	0	6	36	1	1	2	0	0	0	1
36	0	1	0	5	0	5	3308	1	1	0	0	1	0	17	0	1	1	0	0	0	0	0
18	0	0	0	2	0	1736	15	2922	4	1	6	6	0	1	19	0	1	1	0	4	0	1
20	0	0	0	1	0	3	2	0	942	0	1	0	0	1	4	0	0	0	0	0	0	3
13	0	1	0	2	0	7	6	2	2	984	0	1	1	2	11	0	1	2	0	2	0	0
2180	0	0	0	1	0	3	0	0	0	0	870	0	0	2	3	0	0	0	0	0	0	0
14	6	4	1	3	0	11	7	65	4	2	1	2558	2	5	180	3	15	10	2	10	2	5
2	0	0	0	0	0	1	0	0	6	0	0	7	678	1	6	0	1	0	3	2	0	0
8	0	3	0	3	0	4	5	0	2	0	0	2	0	1051	6	27	3	1	0	0	0	0
69	0	1	0	7	0	24	28	4	7	0	0	1	1	6	6262	0	19	1	2	2	0	1
0	0	0	2	0	0	2	0	0	0	0	0	0	2	4	2378	0	0	0	0	438	0	0
42	0	2	2	233	0	22	23	1	10	3	0	3	0	1	92	0	1569	2	0	0	1	2
12	0	0	0	2	0	7	12	7	4	1	1	9	0	2	35	1	6	1025	0	3	3	3
0	0	4	74	0	0	0	0	0	0	0	0	1	5	0	2	0	2	1	663	5	0	0
1	13	12	1	0	0	0	6	1	0	0	0	26	20	8	38	0	4	7	3	3085	0	0
1	0	0	0	0	0	1	0	0	0	0	0	1	0	1	8	0	1	63	0	0	149	0
12	0	0	0	0	0	3	6	1	31	0	0	0	1	1	19	0	0	0	0	0	0	181

0.8190483450889587

mean_squared_logarithmic_error

4035	0	2	0	10	0	13	20	4	0	4	15	5	3	3	28	0	26	8	0	1	0	4
0	1788	0	0	0	0	0	1	3	2	0	195	3	0	1	2	0	2	2	0	0	1	1
0	3	623	0	0	0	1	4	0	0	0	0	0	0	5	0	6	1	0	0	0	0	0
0	0	0	85	0	0	0	0	0	0	0	0	2	0	0	0	1	1	962	15	0	0	0
10	1	2	0	2100	0	5	4	1	2	2	0	2	0	0	2	0	9	0	0	0	0	0
0	0	1	1	0	1140	0	0	0	0	0	2	0	0	0	0	2	0	0	0	26	0	0
17	0	0	0	6	0	4550	12	1949	4	4	6	7	0	8	9	0	1	1	0	0	0	1
22	0	1	0	11	0	13	3309	2	0	0	0	1	1	11	0	3	2	0	0	0	0	0
1	0	0	0	2	0	1071	7	3604	3	2	10	20	0	2	7	0	1	2	0	3	1	1
12	0	0	0	4	0	1	3	1	930	0	2	0	5	1	4	0	5	1	0	0	0	8
4	0	1	0	2	0	2	2	0	0	1019	0	1	0	0	1	0	2	1	0	2	0	0
512	3	0	2	2	0	13	14	21	0	1	2397	32	2	2	22	0	30	3	0	1	0	2
2	7	3	1	4	0	8	2	57	0	4	1	2712	1	5	31	1	54	5	0	5	3	4
0	2	0	0	0	0	1	1	0	1	0	12	651	1	2	0	11	1	1	19	1	2	2
4	0	0	0	7	0	2	4	2	0	1	0	5	0	1062	1	18	4	1	0	4	0	0
35	0	1	0	16	0	34	28	17	4	7	9	2	7	6192	0	64	0	3	4	0	5	5
0	0	0	3	0	0	2	0	0	0	0	0	0	1	2	2194	0	2	0	622	0	0	0
22	0	0	0	277	0	19	10	2	3	3	1	2	2	0	5	0	1659	0	0	0	2	1
5	0	0	0	2	0	8	4	12	2	1	4	16	1	1	10	0	58	953	0	11	40	5
0	0	1	14	1	0	0	0	0	0	0	1	4	0	1	1	2	2	723	7	0	0	0
1	113	9	0	0	0	0	3	1	0	0	1	25	3	8	7	0	9	4	6	3033	0	2
0	0	0	0	0	0	1	0	0	0	0	0	3	0	1	0	0	13	3	0	0	204	0
7	0	0	0	2	0	2	3	1	8	0	0	0	1	1	4	0	7	0	0	1	1	217

0.85957270860672

squared_hinge

4148	0	0	0	4	0	0	4	0	1	2	18	1	0	0	0	0	0	3	0	0	0	0
0	962	0	0	0	0	0	0	1	2	0	1034	1	0	0	0	0	0	1	0	0	0	0
0	3	633	0	0	0	1	1	0	1	0	0	0	0	1	0	2	1	0	0	0	0	0
0	0	0	97	0	0	0	0	0	0	0	0	1	3	0	0	0	1	953	11	0	0	0
21	1	2	0	2092	0	1	7	0	5	2	0	0	0	1	0	8	0	0	0	0	0	0
0	0	0	1	0	1157	0	1	0	0	0	6	0	0	0	0	0	7	0	0	0	0	0
47	0	0	0	6	0	4729	18	1712	10	12	6	1	1	8	21	0	2	2	0	0	0	0
32	0	1	0	8	0	2	3317	1	0	1	0	0	1	1	9	0	2	2	0	0	0	0
10	0	0	0	2	0	1200	15	3457	8	8	13	6	2	1	9	0	2	1	0	3	0	0
13	0	0	0	3	0	1	3	0	952	2	2	0	0	0	1	0	0	0	0	0	0	0
6	0	1	0	1	0	3	0	0	0	1022	0	1	0	0	1	0	0	1	0	0	0	0
895	0	0	0	2	0	3	1	0	1	0	2156	0	0	1	0	0	0	0	0	0	0	0
26	7	4	0	7	0	8	6	74	10	29	3	2637	4	8	27	0	45	9	2	4	0	0
1	0	0	0	0	0	1	0	0	6	6	0	4	683	1	0	0	1	0	0	4	0	0
7	0	0	0	6	0	3	7	1	4	8	0	5	0	1047	2	19	3	1	0	2	0	0
63	0	1	0	12	0	31	47	9	12	29	14	10	0	8	6182	0	15	1	0	1	0	0
1	0	0	2	0	0	2	2	0	0	2	0	0	0	2	1	2457	0	0	0	357	0	0
28	0	0	0	213	0	17	12	2	5	11	3	4	2	0	9	0	1699	3	0	0	0	0
10	0	0	0	2	0	7	13	8	11	3	5	12	1	0	11	0	14	1033	0	3	0	0
0	0	1	18	0	0	0	0	0	0	0	0	4	5	0	1	0	3	2	719	4	0	0
3	48	7	0	0	1	0	9	2	2	13	2	20	21	6	4	0	5	8	8	3066	0	0
0	0	0	0	0	5	1	7	0	9	2	0	5	2	2	4	0	13	175	0	0	0	0
16	0	1	0	1	0	3	28	1	159	2	0	4	0	1	19	0	19	0	0	1	0	0

0.8417838215827942

categorical_hinge

3999	0	2	0	15	0	28	21	0	4	3	61	2	1	8	27	0	6	4	0	0	0	0
0	1813	0	0	0	0	3	0	0	0	0	182	0	0	1	1	0	0	1	0	0	0	0
0	3	603	0	1	0	1	3	5	1	0	0	0	0	8	13	0	4	1	0	0	0	0
0	0	0	412	0	1	0	1	3	0	0	0	3	6	8	6	10	3	2	548	63	0	0
7	0	1	0	2114	0	9	0	0	2	0	0	0	0	1	0	6	0	0	0	0	0	0
0	0	0	0	0	1154	0	0	0	0	0	0	0	0	0	0	1	1	0	2	14	0	0
19	0	0	0	5	0	4842	6	1666	3	0	4	11	1	6	8	0	0	1	0	2	1	0
20	0	1	0	13	0	31	3281	0	8	0	13	0	2	2	1	14	0	2	1	0	0	1
0	0	0	0	2	0	1618	7	3044	4	1	9	32	2	2	6	1	2	0	0	5	2	0
9	0	0	0	3	0	4	2	2	948	0	4	1	2	1	1	0	0	0	0	0	0	0
6	0	1	0	4	0	14	7	3	1	985	0	1	2	6	2	1	0	2				

categorical_crossentropy

4146	0	1	0	4	0	1	10	0	0	3	3	0	0	3	2	0	1	3	0	0	0	4
0	1410	0	0	0	1	5	1	4	2	5	560	3	0	1	4	0	4	1	0	0	0	0
0	3	570	0	0	0	0	3	10	0	2	0	0	0	4	27	0	20	4	0	0	0	0
0	0	0	144	0	0	0	1	1	0	1	0	0	1	0	1	6	0	1	896	13	0	0
17	0	2	0	2087	0	8	8	0	1	3	0	0	0	4	3	1	6	0	0	0	0	0
0	0	0	1	0	1162	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	7	0
31	0	0	0	4	0	5409	16	1071	5	6	1	1	0	7	14	4	2	2	0	1	0	1
23	0	1	0	6	0	6	3329	1	0	0	0	1	1	4	0	2	1	0	0	0	0	2
5	0	0	0	2	2	1851	12	2810	3	11	5	3	0	2	8	10	7	4	0	1	0	1
20	0	0	0	3	0	2	6	1	903	1	1	0	5	3	2	0	0	0	0	0	0	30
4	0	0	0	1	0	3	2	0	0	1022	0	0	0	1	0	1	1	0	2	0	0	0
1471	0	0	0	5	0	23	20	7	0	31	1444	4	0	16	11	0	22	3	0	0	0	2
14	7	1	0	4	1	14	7	266	5	40	1	2369	7	11	37	9	76	9	0	6	5	21
0	0	0	0	1	0	2	3	0	0	1	0	2	687	1	1	2	4	0	0	1	0	1
2	0	0	0	1	0	3	5	0	0	0	0	1	0	1083	2	12	3	1	0	2	0	0
61	1	0	0	12	1	38	42	10	4	12	2	3	0	9	6199	1	24	1	0	3	1	11
1	0	0	6	0	2	2	5	1	0	0	4	1	9	1	1536	1	0	0	1256	0	0	0
29	0	0	0	210	0	17	11	0	3	4	1	2	2	0	7	0	1717	2	0	0	1	2
14	0	0	0	2	0	6	32	21	3	2	1	8	1	3	10	0	21	938	0	6	56	9
0	0	0	36	0	0	0	0	0	0	0	1	4	0	1	11	3	2	694	5	0	0	0
1	128	2	0	0	7	0	8	49	2	3	0	19	28	10	2	8	13	11	5	2926	2	1
0	0	0	0	0	0	1	1	0	0	0	0	1	0	1	0	0	3	0	0	0	218	0
8	0	0	0	2	0	2	0	0	7	1	0	0	1	1	2	0	1	0	0	0	0	230

0.8187249302864075

kullback_leibler_divergence

4156	0	2	0	1	0	0	6	0	0	1	3	1	0	2	3	0	5	1	0	0	0	0
0	1016	0	0	0	1	4	0	11	2	0	939	7	0	4	12	0	2	3	0	0	0	0
0	3	569	2	0	0	0	1	9	0	0	0	1	0	0	34	0	20	2	0	0	0	0
0	0	0	109	0	0	0	0	0	0	0	0	0	0	0	1	0	1	950	5	0	0	0
22	0	2	0	2089	0	6	3	0	1	0	0	0	0	2	4	1	8	0	0	1	0	1
0	0	0	1	0	1152	0	0	0	0	0	0	1	0	0	1	2	3	0	1	11	0	0
38	0	0	0	6	0	3730	8	2754	4	0	1	3	1	6	17	1	2	2	0	1	0	1
31	0	1	0	7	0	12	3299	4	1	0	0	1	1	12	0	7	0	0	0	0	1	1
8	0	0	0	2	0	1010	9	3679	1	1	6	4	0	2	6	1	4	1	0	1	0	2
21	0	0	1	5	0	4	5	6	851	0	1	9	17	6	4	0	18	3	0	0	0	26
7	1	1	2	4	0	9	7	5	2	924	0	2	2	10	12	0	43	2	1	2	0	1
1604	0	0	1	4	0	11	3	7	0	1	1400	2	0	3	13	0	8	1	0	0	0	1
9	6	2	3	8	0	7	2	118	2	1	0	2585	13	3	36	2	96	7	0	4	0	6
0	0	0	0	2	0	1	1	0	0	0	0	4	684	1	0	0	6	0	2	5	0	1
4	0	0	2	1	0	2	3	1	0	0	1	0	1052	1	37	6	0	0	4	0	1	1
49	0	0	0	9	0	28	15	14	6	1	5	0	7	6255	0	39	0	0	2	0	4	4
0	0	0	2	0	0	2	0	0	0	0	0	1	1	2345	2	0	1	472	0	0	0	0
27	0	0	0	197	0	15	5	0	2	1	0	1	2	0	5	0	1750	2	0	0	0	1
11	0	0	1	2	0	4	4	16	1	0	1	11	3	2	15	0	54	993	0	9	2	4
0	0	0	34	0	0	0	0	0	0	0	1	4	0	0	1	0	0	717	0	0	0	0
3	37	0	4	0	0	0	3	9	1	0	0	57	22	8	7	0	13	5	10	3045	0	1
0	0	0	0	0	0	0	1	1	0	0	0	2	0	1	1	0	12	12	0	0	195	0
7	0	0	0	1	0	0	0	1	8	0	0	0	1	1	5	0	5	0	0	2	0	224

0.8146534562110901

poisson

4161	0	2	0	1	0	0	2	0	0	2	5	0	0	4	1	0	3	0	0	0	0	0
0	1094	0	0	0	0	6	0	9	2	1	862	5	0	10	6	0	1	3	0	0	1	1
0	2	616	0	0	0	0	1	2	0	0	0	1	0	1	12	0	7	0	0	0	0	0
0	0	0	365	0	0	0	0	0	0	0	2	1	1	1	0	1	1	676	18	0	0	0
25	0	2	0	2071	0	7	5	0	1	1	0	2	0	12	4	0	9	0	0	0	0	1
0	0	0	1	0	1151	0	0	0	0	0	0	4	0	1	0	0	1	0	0	13	0	0
39	0	0	0	5	0	5382	9	1106	3	0	1	6	0	13	7	0	2	1	0	0	0	1
36	0	2	0	7	0	11	3293	3	1	0	0	0	1	6	10	0	5	0	0	0	0	2
7	0	0	0	2	0	1768	9	2901	4	1	5	17	0	11	5	0	4	0	0	1	0	2
25	0	0	0	3	0	3	2	0	899	0	1	0	4	6	4	1	4	0	0	0	0	25
5	0	1	0	1	0	6	3	1	0	1006	0	1	0	4	2	0	4	1	0	2	0	0
1443	0	0	0	2	0	18	2	6	0	1	1560	1	0	11	10	0	2	0	0	0	0	3
17	7	1	0	4	0	8	2	74	3	1	1	2643	2	35	27	0	70	4	0	3	1	7
0	0	0	0	1	0	1	1	2	0	0	0	9	685	1	0	0	6	0	0	0	0	1
2	0	0	0	0	0	1	1	0	0	0	0	0	0	1106	1	1	3	0	0	0	0	0
54	0	1	0	7	0	37	19	11	4	2	1	6	1	26	6229	0	33	0	0	1	0	3
0	0	0	6	0	0	2	0	0	0	0	0	0	0	9	1	1958	2	0	0	848	0	0
29	0	0	0	119	0	17	7	0	2	3	1	2	1	4	4	0	1816	1	0	0	1	1
13	0	1	0	2	0	10	7	15	6	1	1	15	0	4	16	0	40	959	0	10	24	9
0	0	1	110	0	0	0	0	0	0	0	1	6	5	1	0	3	2	622	6	0	0	0
4	91	2	0	0	0	0	2	1	0	0	0	56	27	70	13	0	10	5	4	2939	0	1
0	0	0	0	0	0	1	0	0	1	0	0	1	0	3	0	0	1	2	0	0	216	0
8	0	0	0	2	0	0	0	1	6	0	0	1	1	3	5	0	0	0	0	0	0	228

0.8352200388908386

cosine_proximity

4141	0	0	0	1	0	5	13	0	0	2	3	0	0	1	3	0	2	5	0	0	0	5
0	1292	0	0	0	1	3	3	2	2	1	688	1	0	0	1	0	5	1	0	0	0	1
0	3	606	0	0	0	0	5	7	0	0	0	0	0	12	0	9	1	0	0	0	0	0
0	0	0	97	0	0	0	1	0	0	0	0	1	0	0	1	0	3	1	959	3	0	0
25	0	1	0	2034	0	11	8	0	3	0	0	0	0	2	6	0	47	0	0	0	2	1
0	0	0	1	0	1157	0	0	0	0	0	0	0	0	0	0	1	1	0	0	11	1	1
33	0	0	0	5	0	5976	18	501	6	0	3	3	0	5	18	1	2	0	0	1	3	3
24	0	1	0	9	0	13	3314	0	0	0	0	0	1	1	9	0	1	3	0	0	1	0
8	0	0	0	2	0	2792	13	1880	5	1	11	8	0	1	10	0	3	0	0	1	1	1
13	0	0	0	3	0	2	3	0	922	0	2	0	4	0	1	0	0	0	0	0	0	27
6	0	0	0	4	1	12	11	3	2	963	0	0	0	2	8	0	20	2	0	2	0	1

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

4160	0	0	0	3	0	0	5	0	0	2	7	0	0	1	0	0	1	2	0	0	0	0
0	1019	0	0	0	0	0	1	0	2	1	975	0	0	0	0	0	1	1	0	0	1	0
3	3	569	2	0	0	0	14	4	0	1	0	1	0	0	18	0	13	12	0	0	0	3
1	0	0	511	0	0	0	2	0	0	0	0	1	0	0	1	0	0	1	541	8	0	0
21	0	1	0	2079	0	2	16	0	2	1	0	0	0	1	1	0	15	0	0	0	0	1
1	0	0	2	0	1147	0	0	0	0	0	0	3	0	0	0	1	1	3	0	1	12	1
74	0	0	0	6	0	3394	31	2965	7	11	4	9	1	8	34	2	7	5	0	2	1	14
30	0	0	0	5	0	0	3331	0	0	0	0	1	0	7	0	1	2	0	0	0	0	0
22	0	0	0	2	0	461	27	4128	4	7	11	18	4	2	18	2	5	10	0	3	1	12
18	0	0	0	3	0	1	6	0	907	0	2	0	3	1	1	0	1	1	0	0	0	33
5	0	0	0	2	0	1	8	0	1	1010	0	0	1	2	0	3	1	0	2	0	1	1
1053	0	0	0	2	0	2	2	0	0	1	1991	0	0	2	0	0	3	2	0	0	0	1
30	7	2	1	5	0	7	20	26	3	4	2	2622	4	2	48	0	69	18	0	4	2	34
1	1	0	0	0	0	1	0	0	2	0	0	5	686	1	0	0	7	0	1	0	0	2
7	0	0	0	6	0	1	5	0	1	0	0	2	0	1068	3	11	5	1	0	4	0	1
65	0	0	0	13	0	22	64	10	9	2	4	4	1	8	6181	0	40	1	0	1	0	10
0	0	0	2	0	0	2	3	0	0	0	0	0	0	1	3	2334	0	1	0	480	0	0
29	0	0	0	100	0	10	12	0	2	3	1	1	1	0	4	0	1841	1	0	0	1	2
9	0	0	0	2	0	4	15	5	3	1	4	9	0	0	5	0	23	1036	0	6	3	8
1	0	0	208	0	0	0	0	0	0	0	0	1	5	0	1	0	5	2	533	1	0	0
6	91	0	3	0	0	0	9	1	2	1	0	21	23	6	7	0	9	13	7	3024	0	2
0	0	0	0	0	0	0	2	1	0	0	0	1	0	0	0	2	21	0	0	195	2	2
7	0	0	0	1	0	0	3	0	4	0	0	0	1	1	2	0	0	0	0	0	0	236

0.8371606469154358

Kernel: (4, 2), Adamax, cosine_proximity

4161	0	0	0	3	0	1	8	0	0	1	3	0	0	0	1	0	1	2	0	0	0	0
0	1082	0	0	0	0	3	6	1	2	0	874	19	0	1	8	0	1	3	0	0	1	0
0	3	564	1	0	0	0	12	6	0	0	0	1	0	2	40	0	8	6	0	0	0	0
0	0	0	382	0	0	0	1	0	0	0	0	1	1	0	1	0	1	1	667	11	0	0
20	0	1	0	2077	0	4	16	0	2	2	0	2	0	0	2	0	13	0	0	0	1	0
0	0	0	2	0	1153	0	1	0	0	0	0	0	0	0	1	0	2	0	0	13	0	
42	0	0	0	6	0	5635	22	828	4	0	1	3	0	7	22	1	1	1	0	0	1	1
26	0	1	0	4	0	4	3332	0	0	0	0	0	0	8	0	1	1	0	0	0	0	
13	0	0	0	2	0	2175	19	2470	3	2	6	22	0	1	16	2	1	1	0	3	1	0
18	0	0	0	4	0	3	13	1	921	0	1	0	2	1	2	0	4	1	0	0	6	6
8	0	1	0	2	0	10	21	0	1	980	0	1	0	1	4	0	5	1	0	2	0	0
1844	0	0	1	2	0	14	7	0	0	1	1150	18	0	2	15	0	3	1	0	0	0	1
8	7	1	1	5	0	13	18	34	1	1	0	2703	1	4	57	0	41	9	1	2	3	0
0	0	0	0	0	0	1	4	0	1	0	0	10	676	1	3	0	8	0	1	1	0	1
7	0	0	0	5	0	7	6	0	1	0	0	1	0	1049	4	29	5	1	0	0	0	0
57	0	0	0	12	0	35	65	4	7	2	1	4	1	8	6215	0	22	1	0	1	0	0
0	0	0	2	0	0	2	4	0	0	0	0	0	0	1	3	2536	0	1	0	277	0	0
27	0	0	0	119	0	19	14	0	3	3	119	0	2	0	0	6	0	1811	2	0	0	2
10	0	0	0	2	1	8	29	7	2	1	2	16	1	1	16	0	20	1010	0	0	6	1
0	0	1	151	0	0	0	0	0	0	0	0	2	5	0	1	3	5	2	584	3	0	0
1	10	1	2	1	0	0	21	2	2	0	0	82	19	7	17	0	12	6	6	3036	0	0
0	0	0	0	0	0	1	5	0	1	0	0	1	0	1	0	0	0	0	0	209	0	0
7	0	0	0	2	0	4	20	0	13	0	0	3	1	1	7	0	10	0	0	1	186	0

0.8356385827064514

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

4161	0	1	0	7	0	0	1	0	0	2	6	0	0	1	0	0	0	2	0	0	0	0
0	1153	0	0	1	0	1	0	0	2	1	839	1	0	1	0	0	1	1	0	0	0	0
0	3	629	0	0	0	0	1	1	0	0	0	0	0	0	2	0	6	1	0	0	0	0
0	0	0	368	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	694	1	0	0
15	0	2	0	2114	0	0	0	0	2	1	0	0	0	0	0	5	0	0	0	0	0	1
0	0	1	1	0	1158	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	10	0
46	0	0	0	7	0	4179	14	2248	4	14	7	10	1	9	24	0	3	3	0	0	1	5
42	0	1	0	18	0	9	3263	0	1	3	0	0	1	1	15	0	13	3	0	0	1	6
9	0	1	0	2	0	734	7	3896	3	12	14	15	2	6	12	0	11	4	0	1	2	6
19	0	0	0	3	0	2	3	0	902	1	1	0	8	1	1	0	3	0	0	0	0	33
7	0	0	0	2	0	1	0	0	0	1024	0	0	0	0	0	0	2	0	0	1	0	0
1193	0	0	0	3	0	2	1	0	1	0	1	1853	0	0	2	0	0	3	0	0	0	1
9	7	6	0	13	0	7	1	36	0	21	1	2589	11	7	44	0	136	7	2	0	4	9
0	1	0	0	1	0	1	0	0	1	0	0	4	684	1	0	0	12	0	1	0	0	1
6	0	1	0	6	0	1	4	0	0	0	0	0	0	1088	2	1	6	0	0	0	0	0
64	0	1	0	23	0	27	16	9	6	9	5	4	1	12	6196	0	55	0	0	1	0	6
0	0	0	11	0	0	2	0	0	0	1	0	5	2	4	2	2403	2	1	1	392	0	0
27	0	0	0	357	0	12	3	0	2	4	1	0	2	0	6	0	1592	0	0	0	1	1
10	0	0	0	2	0	5	3	9	2	3	3	13	2	1	17	0	50	992	0	0	12	9
0	0	2	124	0	0	0	0	0	0	0	1	6	0	1	0	3	2	618	0	0	0	0
4	123	43	24	10	0	0	2	0	2	7	1	129	74	20	16	0	60	13	21	2673	1	2
0	0	0	0	0	0	1	0	0	0	0	0	1	0	2	0	8	1	0	0	211	1	1
8	0	0	0	2	0	0	1	1	4	0	0	0	0	1	1	3	0	4	0	0	0	230

0.836659879684448

Kernel: (16, 2), RMSprop, categorical_crossentropy

4146	0	1	0	2	0	0	7	0	0	2	6	4	0	4	2	0	5	1	0	0	0	1
0	1358	0	0	0	1	3	0	1	2	0	603	24	0	3	2	0	1	3	0	0	0	0
0	3	615	0	0	0	0	1	2	0	0	0	1	0	2	9	0	8	1	1	0	0	0
0	0	0	112	0	0	0	0	0	0	0	3	0	1	1	1	1	1	939	7	0	0	0
20	0	1	0	2080	0	4	4	0	1	1	0	5	0	10	3	0	10	0	0	0	0	1
0	0	0	1	0	1150	0	0	0	0	0	0	6	0	0	0	1	1	0	0	0	13	0
39	0	0	0	5	0	5681	9	731	5	0	1	52	0	14	27	1	2	3	0	0	0	5
31	0	2	0	7	0	13	3298	1	1	0	2	1	1	9	0	7	0	0	0	2	2	2
7	0	0	0	2	0	2278	8	2255	3	1	7	140	0	10	10	3	4	3	0	1	1	4
18	0	0	0	3	0	2	3	2	901	0	1	11	4	4	2	0	4	0	0	0	0	22
7	0	1	0	2	0	4	5	1	1	998	0	3	0	4	1	0						

Kernel: (16, 2), Adam, categorical_hinge

4149	0	0	0	4	0	13	1	0	0	2	5	1	0	1	0	0	2	3	0	0	0	0
0	1449	0	0	0	0	3	0	0	0	1	548	0	0	0	0	0	0	0	0	0	0	0
0	3	571	3	0	0	1	1	19	1	4	0	0	0	7	5	0	18	10	0	0	0	0
0	0	0	465	0	0	0	1	0	0	1	0	0	1	0	0	3	3	2	576	13	0	0
23	1	1	0	2073	0	15	1	0	4	3	0	0	0	0	0	0	19	0	0	0	0	0
0	0	0	1	0	1158	0	0	0	0	0	0	1	0	0	0	2	2	8	0	0	0	0
35	0	0	0	2	0	5678	8	825	5	4	4	7	0	3	0	1	1	2	0	0	0	0
39	0	1	0	11	0	30	3268	1	6	2	0	0	2	1	7	0	5	4	0	0	0	0
8	0	0	0	3	0	2267	8	2392	4	6	11	24	3	1	0	2	3	2	0	3	0	0
16	0	0	0	4	0	4	2	1	945	1	2	0	1	0	0	1	0	0	0	0	0	0
6	0	1	0	1	0	13	0	0	0	0	1009	0	0	1	1	0	2	1	0	2	0	0
1363	0	0	1	2	0	17	0	0	0	1	1664	3	0	2	0	0	4	2	0	0	0	0
7	7	2	3	4	0	26	1	42	3	12	0	2703	6	2	8	1	72	7	1	3	0	0
0	1	0	0	0	0	1	0	0	3	3	0	3	690	1	0	0	4	1	0	0	0	0
6	0	0	0	6	0	13	3	3	2	2	0	1	0	1048	1	22	6	1	0	1	0	0
80	0	1	0	16	2	47	41	22	14	26	3	32	4	8	5984	2	139	12	1	1	0	0
0	0	0	2	0	0	2	0	0	0	2	0	0	0	0	1	2563	1	2	0	253	0	0
29	0	0	0	95	0	25	7	0	4	5	0	2	1	0	4	0	1834	2	0	0	0	0
9	0	0	0	2	0	8	3	5	5	1	4	15	1	2	4	1	40	1030	0	3	0	0
0	0	1	172	0	1	0	0	0	0	0	0	2	8	0	1	3	5	2	556	6	0	0
2	60	1	3	0	3	0	5	2	3	3	0	35	31	7	3	0	14	6	3	3044	0	0
0	0	0	0	0	8	0	3	1	2	0	0	7	0	3	1	3	61	136	0	0	0	0
15	0	0	0	3	0	4	8	1	150	0	0	3	1	1	3	1	60	4	0	1	0	0

0.8423165678977966

Kernel: (16, 2), Adamax, cosine_proximity

4155	0	0	0	6	0	0	4	0	0	2	8	1	0	0	0	0	4	0	0	0	0	1
0	1293	0	0	1	0	1	0	0	2	1	701	0	0	1	0	0	1	0	0	0	0	0
0	3	613	0	0	0	0	1	4	0	0	0	0	0	7	0	0	12	1	1	0	0	1
0	0	0	95	0	0	0	0	0	0	0	0	1	0	0	0	0	3	1	963	3	0	0
16	0	1	0	2102	0	1	1	0	4	1	0	0	0	0	0	0	12	0	0	0	1	1
0	0	0	1	0	1157	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	12	0
44	0	0	0	7	0	4549	18	1889	11	7	6	5	0	9	19	0	4	1	0	0	1	5
33	0	1	0	10	0	4	3309	0	0	0	0	0	1	1	10	0	4	1	0	0	1	2
7	0	0	0	2	0	939	13	3702	7	6	14	9	1	5	9	2	9	2	0	2	3	5
14	0	0	0	3	0	1	3	0	944	0	2	0	0	0	0	1	0	0	0	0	9	0
6	0	1	0	3	0	3	2	0	1	1011	0	1	0	1	0	0	5	1	0	1	0	0
1137	1	0	0	2	0	4	3	1	0	1	1897	0	0	3	2	0	6	1	0	0	0	1
7	7	2	1	5	0	7	3	57	13	10	0	2589	3	14	25	1	129	8	0	1	4	24
0	1	0	0	1	0	1	0	0	9	0	0	5	673	1	0	0	14	0	1	0	0	1
4	0	0	0	5	0	2	4	0	3	0	0	0	0	1083	1	6	6	0	0	0	1	
64	0	1	0	16	0	31	32	11	13	7	5	6	1	15	6127	0	93	0	0	2	0	11
0	0	0	2	0	0	2	0	0	0	0	0	0	3	2	2527	2	0	0	0	288	0	0
25	0	0	0	172	0	15	7	0	5	3	1	0	0	0	1	0	1776	0	0	0	2	1
8	0	0	0	2	0	7	10	6	7	2	4	13	0	2	10	0	67	949	0	1	36	9
0	0	1	20	0	0	0	0	0	0	0	1	6	0	1	2	4	2	718	1	0	1	1
5	57	4	2	1	0	0	6	1	3	1	0	53	26	26	8	1	34	5	18	2971	1	2
0	0	0	0	0	0	1	1	0	0	0	0	1	0	2	0	0	4	2	0	0	214	0
7	0	0	0	1	0	0	2	0	11	0	0	0	1	1	2	0	2	0	0	0	0	228

0.8500979542732239

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

4153	0	0	0	7	0	1	3	0	0	2	4	1	0	0	0	1	4	0	0	0	0	1
0	1395	0	0	1	0	1	0	1	2	1	595	1	0	0	1	0	1	0	0	0	1	0
0	3	592	0	0	0	0	1	1	0	0	0	0	0	0	26	0	18	1	0	0	0	1
0	0	0	119	0	0	0	0	0	0	0	0	1	0	0	1	0	3	1	936	5	0	0
17	0	0	0	2109	0	2	0	0	0	1	0	0	0	0	0	0	10	0	0	0	0	1
1	0	0	1	0	1122	0	0	0	0	0	8	0	0	1	1	4	1	1	1	27	4	4
40	0	0	0	7	0	4468	10	1999	5	1	3	3	0	8	27	0	2	0	0	0	0	2
31	0	0	0	17	0	12	3287	3	2	0	0	0	1	0	12	0	7	2	0	0	0	3
10	0	0	0	2	0	1097	10	3576	3	1	8	8	0	1	14	0	4	0	0	2	0	1
17	0	0	0	3	0	0	1	3	893	0	1	1	7	1	3	0	7	1	0	0	0	39
6	0	0	0	4	0	6	2	2	1	1005	0	0	0	1	2	0	5	1	0	2	0	0
1544	0	0	0	2	0	6	1	2	0	0	1496	0	0	2	0	0	5	0	0	0	0	1
6	7	1	0	9	0	7	2	70	0	4	0	2620	4	2	63	0	96	6	0	4	2	7
0	1	0	0	2	0	1	0	0	2	0	0	7	679	1	1	0	11	0	0	1	0	1
5	0	0	2	8	0	2	4	2	0	1	0	4	0	1056	8	4	14	0	0	5	0	0
60	0	0	1	22	0	27	24	7	3	4	3	4	1	6	6202	0	68	0	0	1	0	2
0	0	0	4	0	0	2	0	0	0	0	1	0	1	2	1736	3	0	0	0	1077	0	0
24	0	0	0	288	0	17	4	0	1	3	1	0	1	0	4	0	1662	0	0	0	2	1
11	0	0	0	2	0	8	4	10	2	2	1	17	1	1	21	0	58	956	0	3	29	7
0	0	1	31	0	0	0	0	0	0	0	0	1	5	0	1	0	3	1	712	2	0	0
1	68	1	2	15	0	0	3	1	1	0	0	36	27	7	12	0	24	5	11	3008	0	3
0	0	0	0	1	0	0	1	1	0	0	0	0	0	1	0	0	11	1	0	0	209	0
7	0	0	0	2	0	1	2	1	5	0	0	0	0	1	1	4	0	0	0	0	0	223

0.8233861327171326

Kernel: (16, 2), Nadam, categorical_hinge

4133	0	1	0	4	0	20	9	0	0	1	8	0	0	0	1	0	1	2	0	0	0	1
0	1101	16	0	0	0	3	0	0	2	0	879	0	0	0	0	0	0	0	0	0	0	0
4	3	591	3	0	0	0	26	7	0	0	1	0	0	1	0	1	4	0	0	0	1	0
0	0	15	591	0	0	0	1	0	0	0	0	0	0	0	0	0	1	449	8	0	0	0
22	1	2	0	2047	0	16	19	0	1	2	0	1	0	0	2	0	26	0	0	0	0	1
0	0	1	1	0	1158	0	1	0	0	0	0	0	0	0	0	1	0	8	0	0	0	2
33	0	1	0	2	0	4240	18	2265	2	0	3	1	0	2	3	1	1	1	0	0	0	2
27	0	1	0	5	0	17	3316	1	0	0	0	1	0	6	0	1	1	0	0	0	1	1
12	0	1	0	2	0	1164	16	3517	3	0	12	3	0	1	2	1	0	0	2	0	0	1
17	0	0	0	4	0	5	18	0	872	0	1	0	1	0	0	0	0	0	0	0	0	59
9	0	3	0	1	0	15	14	0	0	987	0	1	0	1	1	0	3	1	0			

Kernel: (4, 2), Adamax, cosine_proximity

2256	2	0	4	8	6	1	2	1	0	2	7	1	1	2	7	0	5	6	2	4	0	1	2	2	3	7	4	1	5	1	0	0	0	2	8	0	5	2	0	1	1	0	2	3	2		
11	17175	2	17	48	19	1	35	7	2	37	12	12	1	9	18	0	315	42	16	22	9	10	4	33	54	6	53	16	28	7	14	0	7	13	3	0	3	25	0	13	5	5	2	5	7		
7	18	2168	4	16	2	0	2	0	1	16	1	8	1	3	5	0	10	10	9	2	0	3	1	2	1	2	5	8	9	6	5	0	0	9	0	0	2	3	1	1	0	2	4	1	3		
35	97	8	6524	116	67	6	12	19	3	55	24	55	32	10	33	3	178	80	83	51	16	135	6	44	119	22	26	47	93	8	7	0	7	100	12	2	11	471	3	42	18	11	0	6	21		
117	115	18	84	15678	82	10	42	20	5	83	50	63	39	24	85	8	230	104	197	62	29	83	21	43	138	70	69	79	122	9	10	3	1	71	35	4	66	41	4	69	21	26	8	21	43		
11	25	6	10	22	3049	0	17	2	2	18	9	5	2	11	12	0	26	15	11	6	11	9	0	17	20	3	21	15	27	3	12	0	0	9	1	2	1	6	1	7	4	1	0	4	6		
1	25	2	22	56	10	5817	0	17	2	5	13	4	23	1	7	15	0	45	22	48	6	53	14	4	23	26	1	12	44	38	1	8	0	8	22	0	16	4	12	2	12	3	3	1	2	3	
1	6	3	0	7	2	2	10452	3	4	8	5	1	0	3	9	0	5	13	0	8	0	0	0	2	252	17	3	22	2	7	0	8	0	3	6	7	0	2	6	0	2	0	2	4	12		
2	12	0	3	4	3	1	7	3253	0	8	3	1	0	2	12	0	11	33	1	1	0	0	3	3	4676	6	6	13	2	18	1	3	0	5	1	3	0	8	4	0	7	3	0	2	0	8	
2	0	0	0	0	5	3	7	1	1888	0	0	0	0	1	3	0	9	6	1	3	2	1	1	7	0	0	6	0	3	0	12	0	0	2	0	0	1	1	1	1	0	1	0	1	7		
14	20	0	11	17	16	0	8	10	3	5792	7	6	0	4	6	0	49	93	4	8	2	6	1	8	28	5	22	6	15	2	13	0	0	14	1	0	1	15	0	6	1	1	0	1	7		
5	10	2	16	19	7	0	5	3	0	8	2766	23	3	8	16	1	22	22	5	4	1	6	15	2	7	8	10	4	15	0	2	0	1	9	9	1	1	8	1	2	4	3	1	9	7		
82	54	8	147	264	52	7	10	17	2	50	460	6329	34	22	48	6	129	83	86	2	11	61	870	38	90	33	36	57	116	10	10	4	1	98	9	2	4	43	2	20	19	15	4	8	22		
41	13	1	8	79	17	0	7	0	2	15	10	16	2395	4	6	1	27	14	32	6	3	8	4	9	22	6	4	13	22	2	2	0	1	21	15	1	5	7	0	18	3	5	3	4	17		
4	5	1	0	5	2	0	2	0	1	7	2	5	2	4292	4	6	8	2	1	1	1	2	3	11	10	1	5	3	2	0	2	0	0	4	2	0	0	5	0	2	6	2	7	1	0	0	
21	20	1	11	30	15	1	9	9	0	18	14	10	3	17	10755	1	31	31	6	2	2	3	1	8	26	27	21	1	14	3	9	0	0	37	2	0	5	10	0	6	4	1	9	47	16		
5	1	0	3	12	3	1	1	2	0	5	3	2	2	1	5	409	7	5	3	1	0	7	0	2	7	4	0	3	6	0	0	17	5	5	1	0	1	0	0	5	0	0	0	1	0		
24	129	10	65	132	50	14	32	15	6	57	27	62	17	14	47	4	14212	65	71	41	32	21	6	47	117	16	59	47	38	23	24	0	6	73	10	6	9	27	4	41	4	9	3	4	29		
58	34	5	46	177	51	5	34	38	6	305	25	17	9	20	64	2	47	12900	60	26	15	70	13	27	67	16	43	37	80	8	26	1	2	96	10	2	13	42	2	29	15	9	1	13	33		
47	100	23	138	298	61	11	31	35	4	67	31	30	0	23	71	9	227	86	10834	36	16	32	8	41	73	28	28	79	181	25	12	2	3	118	13	3	41	72	0	31	33	17	5	5	36		
26	26	4	12	64	23	2	10	4	1	27	3	4	7	7	31	0	42	15	16	6584	4	12	0	7	39	17	28	17	23	0	4	0	2	77	10	0	12	14	3	5	6	4	3	2	4	8	
7	33	3	33	22	393	2	10	16	2	15	23	6	34	0	16	24	1	40	55	87	13	6185	11	6	18	47	5	115	306	43	3	11	0	5	32	0	1	4	20	1	28	0	4	0	4	8	
11	26	1	65	66	25	6	4	5	4	21	4	26	16	4	16	1	63	27	58	18	2	1841	5	32	37	5	9	36	37	5	5	1	1	52	1	1	12	26	3	17	11	9	1	1	6		
7	11	1	25	40	7	3	11	6	3	11	67	1719	1	1	12	1	26	24	12	0	2	1	1764	7	21	1	11	8	12	0	2	0	5	8	4	0	1	3	0	15	5	1	1	0	17		
6	16	3	5	17	13	5	14	3	10	20	4	1	1	4	14	0	42	10	6	12	1	9	1	4651	12	6	9	7	18	0	6	0	1	10	3	0	3	8	1	4	0	2	5	2	8		
43	17	0	17	76	29	1	16	2367	3	39	20	5	7	13	34	0	22	53	8	10	6	1	8	20	6742	15	32	8	45	2	4	0	1	4	11	1	6	24	0	14	6	2	1	10	16		
23	24	5	13	42	19	1	7	2	0	2	1	8	6	3	37	0	35	1	4	16	3	10	0	6	16	4435	13	6	18	1	1	1	6	19	12	0	34	5	1	16	5	3	7	2	14		
5	5	0	0	1	27	0	20	0	6	4	5	4	0	4	15	0	1	14	0	0	13	1	1	4	13	1	1	10552	7	2	0	4	0	6	0	1	0	0	0	3	0	0	0	0	1	0	1
28	89	15	27	83	102	4	36	6	6	47	31	62	7	19	63	0	74	78	81	17	34	16	5	56	89	17	167	8183	98	14	11	1	6	46	6	1	13	38	2	36	1	8	5	8	22		
67	115	10	125	400	31	25	59	20	33	32	48	145	26	40	146	3	548	258	282	90	95	71	20	127	303	57	144	128	39646	15	21	1	22	152	18	6	14	77	6	132	11	32	19	24	67		
26	511	4	118	349	33	23	24	9	4	31	18	41	24	7	31	3	2166	87	137	20	12	137	13	30	64	13	21	90	60	2884	3	0	11	123	7	1	31	13	8	43	3	24	1	7	20		
0	0	0	1	2	2	0	4	7	1	17	0	0	0	2	2	0	3	4	0	2	1	0	0	2	2	1	5	0	1	0	0	1	5	0	0	0	0	1	0	0	0	0	0	0	0	0	
0	5	0	6	3	0	2	0	1	0	0	0	0	2	3	0	303	9	4	1	1	0	1	2	0	0	0	2	3	4	1	0	67	0	11	3	0	1	0	0	5	0	1	0	1			
0	12	1	2	5	0	2	4	3	0	0	0	0	1	5	8	4	27	4	0	12	2	1	1	5	6	6	2	16	1	1	0	2163	0	4	0	5	0	2	1899	3	1	6	1	5			
39	61	7	67	181	67	15	17	10	2	76	18	52	23	7	66	1	146	61	101	118	16	54	13	42	84	31	42	60	80	13	15	0	5	9691	6	1	11	74	5	21	4	16	1	2	16		
60	8	2	10	55	22	0	9	11	0	10	18	0	61	5	30	1	24	36	5	7	5	10	3	2	21	11	11	16	27	0	4	0	1	9	4390	0	5	7	1	3	5	0	2	7	9		
0	1	0	1	1	0	9	2	2	1	2	0	0	0	0	0	0	4	2	2	1	6	1	0	2	5	1	3	1	1	3	0	2	0	0	0	0	0	0	1	1	0	0	0	0	0		
20	12	4	27	187	9	3	7	1	2	30	3	11	26	1	11	0	61	27	74	9	4	47	1	7	34	22	2	27	26	2	4	0	3	49	5	3	1320	9	2	12	8	93	2	5	7		
9	26	7	366	22	19	0	16	5	3	9	4	6	10	11	15	0	34	12	11	18	6	9	3	10	17	3	24	9	19	1	7	0	7	13	2	1	2	6831	0	8	2	2	5	1	6		
0	3	0	3	5	4	0	0	1	3	0	0	2	1	0	4	0	22	5	5	13	1	11	0	1	4	2	0	8	3	0	1	0	1	0	0	0	0	1	520	3	0	1	0	0	1		
36	61	3	72	131	45	6	15	30	2	35	39	19	36	12	46	7	226	19																													

Kernel: (16, 2), RMSprop, categorical_crossentropy

2079	3	1	9	39	15	1	2	0	0	7	10	12	8	4	15	1	18	14	14	9	1	5	4	2	11	8	1	6	14	1	0	0	0	7	13	0	2	7	0	8	8	2	2	3	3	
6	17108	8	22	56	21	5	18	2	2	22	12	42	1	11	45	0	268	30	57	20	7	28	4	30	55	8	27	53	22	3	13	0	5	29	2	0	0	24	1	29	5	8	2	1	11	
4	8	2230	1	7	2	1	1	0	1	0	1	9	1	5	5	0	4	5	9	1	0	3	1	4	2	2	3	11	5	1	4	0	0	5	5	0	0	0	3	1	1	0	2	1	4	3
9	82	11	6212	112	196	13	10	1	1	44	21	141	12	13	59	2	153	82	165	46	18	179	3	44	134	24	24	134	91	10	7	0	5	116	13	0	3	357	4	92	33	7	1	8	26	
50	78	22	72	15654	167	12	32	2	3	48	39	172	18	28	116	8	176	110	281	52	11	119	8	30	135	61	48	119	101	8	10	4	0	81	32	2	19	27	6	120	26	30	8	23	34	
1	17	5	6	20	3115	0	7	0	1	12	6	5	1	10	13	2	29	9	14	6	9	6	0	19	15	5	14	21	20	2	8	0	0	11	1	2	1	5	1	5	4	3	0	4	4	
0	16	2	6	31	10	6047	13	0	4	7	1	22	0	7	17	0	20	10	46	3	34	3	1	5	19	1	12	37	22	0	3	0	8	11	0	7	0	8	2	9	0	2	2	2	3	
2	28	9	10	22	18	8	10276	1	6	8	8	13	2	9	24	1	25	18	20	7	0	7	1	172	41	4	23	20	9	0	10	0	3	10	8	0	0	20	2	14	5	3	2	5	15	
2	16	0	11	21	11	9	2	1868	1	8	2	10	0	5	22	0	33	36	17	10	7	0	8	3	5875	6	9	23	41	5	3	0	4	4	3	2	2	9	0	15	6	1	2	0	11	
1	0	1	0	0	9	6	2	0	1877	0	0	1	0	1	6	0	10	5	7	4	2	2	1	5	1	0	7	0	1	0	10	0	0	4	0	0	1	1	1	1	0	1	0	1	0	
6	28	2	13	36	38	2	6	1	3	5660	5	18	0	2	12	0	54	91	27	8	3	16	2	14	32	5	16	29	13	1	13	0	0	27	2	0	0	14	1	12	1	3	0	1	6	
3	10	5	18	23	11	0	3	0	0	8	2702	112	4	9	21	1	13	10	6	2	3	10	10	2	6	7	4	10	12	1	2	0	0	5	7	1	0	5	0	6	5	3	1	5	5	
16	34	6	95	144	70	8	7	2	1	23	152	7655	9	20	55	4	72	76	58	1	12	70	403	20	60	24	29	72	64	7	6	3	0	68	9	0	3	18	1	30	27	14	4	7	16	
14	9	1	10	89	29	0	2	0	1	10	10	46	2331	4	11	0	23	20	41	5	3	31	1	2	25	5	5	34	27	1	1	1	0	16	15	1	0	4	2	26	2	6	3	6	16	
0	3	1	0	4	1	1	1	0	2	4	3	6	0	0	4333	4	0	5	3	3	1	0	1	2	5	10	0	2	4	1	0	2	0	0	1	0	0	6	0	4	2	2	1	1	0	
5	6	2	3	14	13	1	0	0	0	2	10	11	0	17	11011	1	11	11	2	2	2	4	1	8	16	11	10	5	6	0	7	0	0	14	2	0	1	4	2	5	4	3	6	16	8	
1	1	0	2	15	9	1	0	0	0	0	3	5	0	10	9	362	5	9	7	1	0	13	0	2	8	2	0	2	4	0	0	41	4	6	1	0	0	0	2	3	0	5	1	1		
6	154	12	49	173	77	19	24	3	3	48	26	134	17	19	66	0	13889	64	195	32	20	47	5	34	119	17	42	100	32	24	18	0	4	92	9	0	0	20	3	104	10	4	3	4	28	
22	47	5	41	180	97	13	25	8	6	183	30	62	5	21	95	1	56	12708	112	31	15	107	17	34	109	16	38	111	77	12	29	1	0	132	9	0	5	28	3	41	13	14	2	13	25	
26	61	23	57	186	63	8	25	7	3	30	20	61	1	21	79	2	96	60	11556	27	7	45	5	29	67	21	15	85	98	13	9	2	0	85	10	1	18	23	0	35	30	12	4	6	32	
5	25	4	19	62	24	2	7	0	0	16	6	11	3	5	41	0	37	19	35	6555	3	23	0	3	26	17	15	26	23	0	4	0	2	122	10	0	5	7	4	18	5	2	3	2	5	
2	32	2	22	38	451	19	12	1	8	19	4	45	0	13	32	3	39	37	119	16	6086	18	4	18	41	7	55	400	39	3	7	0	3	23	0	1	2	13	1	40	3	4	0	3	5	
4	16	1	48	52	67	7	2	0	1	14	2	47	5	5	26	1	45	33	72	14	2	1885	3	20	29	3	7	56	27	6	4	1	1	45	2	1	3	16	2	23	9	6	1	2	7	
1	6	0	10	25	18	1	7	0	2	7	26	2550	1	1	14	0	19	20	5	0	3	3	1057	3	16	1	5	17	6	2	0	1	1	5	3	0	0	2	0	14	6	2	1	0	16	
5	21	1	15	24	29	7	16	0	4	11	3	7	3	5	15	1	37	15	13	11	1	21	2	4588	11	6	6	15	22	1	1	0	0	16	1	0	0	5	1	8	10	0	3	2	10	
21	23	0	25	94	92	2	9	416	0	32	27	32	3	17	45	1	39	57	35	25	5	13	6	20	8456	17	22	34	58	8	9	4	1	8	10	0	4	24	0	26	14	6	0	7	12	
15	18	1	10	61	24	3	0	0	0	2	1	22	5	3	64	0	28	1	9	7	0	21	0	4	16	4403	7	14	15	3	1	2	1	18	14	0	10	5	1	38	4	4	9	2	14	
5	15	1	2	16	87	0	11	0	7	7	20	1	9	27	0	2	13	17	7	14	2	1	4	23	1	10302	69	4	0	4	0	6	2	1	0	0	1	18	1	5	1	1	6	0	1	6
5	39	10	16	68	155	7	21	0	5	16	18	66	3	20	60	0	41	47	86	17	29	16	1	30	59	15	91	8627	49	3	10	1	5	24	4	1	0	20	2	33	4	4	4	8	18	
28	142	20	125	403	61	41	54	1	30	27	43	381	18	54	239	3	479	289	569	61	92	132	8	82	296	57	118	277	38899	19	19	2	19	161	12	2	1	45	4	227	21	27	16	24	83	
3	611	6	104	404	157	32	19	3	6	21	8	140	8	10	71	3	1746	107	264	13	12	123	9	22	73	20	16	334	53	2561	2	0	2	153	6	1	12	8	3	86	17	6	1	6	23	
0	1	1	1	0	5	0	2	3	1	5	1	2	0	2	4	0	6	4	0	2	0	0	0	6	3	1	4	0	1	1	4	0	0	1	2	1	1	0	0	0	0	0	0	0	0	0
0	6	0	7	3	1	1	0	1	0	0	0	1	3	0	297	8	4	1	1	1	1	1	4	0	0	0	2	2	4	1	0	71	1	8	3	0	0	0	0	0	0	0	0	0	1	
0	17	2	4	6	1	1	2	1	0	0	1	0	2	5	9	3	23	2	2	11	1	3	2	1	4	7	5	5	21	2	1	0	1443	4	3	0	0	0	8	2599	7	0	5	2	6	
10	59	12	44	152	95	21	18	1	1	42	13	113	12	17	115	0	102	60	144	71	19	70	5	33	86	22	32	104	72	7	12	0	5	9746	5	1	3	34	1	35	5	12	6	3	18	
23	3	2	5	83	26	0	8	0	0	5	14	14	35	9	35	0	19	40	28	6	4	4	4	35	6	7	18	27	1	0	1	0	14	4365	0	4	0	43	5	0	1	8	7			
0	1	0	1	2	0	17	0	1	0	2	0	1	0	0	0	1	1	2	1	3	1	0	0	2	4	0	2	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	25	8	40	275	123	7	7	1	1	15	3	34	18	2	30	1	73	35	149	11	3	90	1	15	50	29	2	93	31	1	4	1	1	64	6	1	737	13	1	54	10	129	3	7	8	
4	20	8	438	28	35	0	7	1	2	10	4	25	6	9	28	0	37	15	23	19	8	33	1	14	18	3	17	22	25	1	6	0	7	24	2	0	1	6648	0	13	8	7	5	2	7	
0	2	0	3	3	3	0	0	0	1	1	0	2	1	0	2	0	10	4	4	10	0	7	0	0	3	1	0	9	2	0	1	0	0	0	0	0	0	0	555	4	0	0	0	0	0	
11	46	6	41	108	69	5	10	9	2	24	25	28	17	5	54	6	124	20	29	36	12	32																								

Kernel: (16, 2), Adam, categorical_hinge

2267	2	0	4	4	3	0	0	0	0	3	5	1	0	2	7	0	3	4	0	4	0	1	3	1	2	5	5	1	8	2	1	0	0	2	13	0	0	3	1	5	1	1	2	1	2		
23	16315	21	46	53	38	4	49	1	2	73	21	58	3	12	50	0	422	65	132	20	18	29	4	69	95	11	57	123	81	41	16	0	5	26	3	0	2	49	2	49	9	4	3	5	14		
7	6	2225	3	3	2	1	3	0	1	14	1	6	0	3	8	0	6	9	4	2	0	0	1	3	1	2	4	5	9	2	3	0	0	4	0	0	1	2	1	2	0	2	2	3			
67	52	15	6359	59	45	5	14	1	4	59	25	89	25	11	43	0	137	76	224	46	16	135	4	34	110	19	26	103	137	9	6	0	6	72	13	0	4	517	11	67	25	15	3	7	23		
286	85	47	118	14468	93	9	39	3	4	91	56	200	46	34	132	0	241	132	566	56	34	80	6	33	136	61	132	236	22	10	0	0	102	43	5	26	63	8	170	62	42	14	32	50			
12	12	9	11	13	2970	1	19	0	5	27	7	12	2	10	18	0	19	19	21	7	19	7	0	18	23	2	25	49	35	2	13	0	0	9	1	1	0	11	2	10	5	1	0	6	6		
21	12	3	26	52	6	5692	16	0	5	19	4	39	2	8	22	0	27	31	90	5	51	17	1	30	22	1	11	82	52	2	5	0	8	24	0	13	2	14	3	23	3	1	2	4	2		
7	2	3	2	9	3	2	10423	1	7	12	2	1	0	6	12	0	3	11	3	4	0	0	2	270	21	1	20	4	8	1	5	0	3	2	9	0	0	6	0	3	0	0	3	4	14		
3	8	2	3	12	2	4	5	1542	2	21	2	7	3	4	14	0	19	48	5	1	5	2	1	3	6230	4	14	10	74	3	9	0	4	1	6	2	6	0	18	6	0	3	0	9	2		
2	0	1	0	0	4	2	7	0	3	1902	0	0	0	1	2	0	3	3	1	3	3	1	1	8	0	0	3	0	4	0	9	0	0	0	0	0	0	1	19	0	11	1	0	0	1	5	
16	9	1	10	13	12	0	7	0	3	5786	7	14	0	3	12	0	39	89	16	6	2	7	3	11	27	5	22	16	23	3	11	0	0	10	2	0	1	19	0	11	1	0	0	1	9	9	
14	5	5	17	13	9	0	5	1	0	8	2663	100	4	12	23	0	7	27	5	3	1	2	20	1	7	5	11	8	25	1	1	0	1	6	14	1	1	9	1	7	5	4	1	9	9		
74	22	14	89	71	29	4	13	4	2	41	193	7879	12	23	55	0	51	72	60	2	13	18	236	26	57	24	38	44	110	7	10	0	0	37	10	0	2	40	4	32	15	7	7	8	20		
81	12	8	12	58	17	1	6	0	1	14	7	49	2194	4	13	0	25	16	69	7	3	8	3	11	22	4	4	27	50	4	2	0	1	19	24	1	2	16	5	51	4	9	3	4	18		
4	2	3	1	1	0	0	2	0	2	11	1	7	1	4312	9	0	2	2	1	1	2	2	1	4	3	0	4	2	5	0	1	0	0	1	3	0	0	7	0	4	8	1	8	1	0	0	
29	4	1	6	9	12	1	7	0	1	7	6	14	0	13	10914	0	14	16	14	1	2	2	2	3	25	16	23	3	19	1	3	0	0	6	3	0	0	1	7	0	7	5	2	10	36	12	
26	6	0	12	20	6	1	1	2	1	8	7	10	18	42	10	0	42	5	54	1	0	50	0	4	8	5	0	6	14	0	0	0	5	29	1	0	2	3	3	1	29	1	98	3	1		
69	76	19	87	107	55	21	35	0	9	84	30	160	16	16	74	0	13496	79	292	37	49	20	2	46	122	17	51	121	135	65	22	0	4	69	14	9	4	53	18	101	14	7	8	5	31		
89	23	10	53	110	42	6	26	5	14	310	24	66	11	24	102	0	37	12773	100	22	21	53	15	27	57	13	52	74	143	4	29	0	0	85	13	0	4	42	3	54	13	9	3	10	28		
54	51	31	64	85	36	6	34	4	4	46	17	66	0	27	72	0	115	78	11498	23	18	5	4	24	66	19	24	74	196	24	12	0	0	63	14	2	7	65	3	44	31	9	8	7	34		
46	23	4	18	39	20	2	13	1	1	43	1	7	6	8	40	0	45	21	39	6430	7	10	0	8	39	27	28	26	39	2	4	0	2	99	14	0	7	18	11	33	9	1	3	2	5		
10	7	4	25	8	286	5	16	0	20	26	4	41	0	17	28	0	14	46	107	11	6167	1	1	10	45	3	107	544	42	0	7	0	3	13	0	0	2	18	1	40	0	1	1	4	5		
24	18	6	86	42	21	8	3	0	3	31	4	47	12	5	24	0	52	36	130	14	5	1695	2	23	36	8	8	54	57	5	5	0	1	42	1	1	3	43	11	22	19	6	1	2	7		
9	1	1	11	8	8	1	10	0	4	9	16	3118	0	1	8	0	8	14	4	0	3	0	0	545	5	20	0	12	3	10	1	1	0	1	2	5	0	0	1	0	19	4	0	1	0	13	
16	9	4	13	20	16	6	23	1	15	22	4	10	3	5	18	0	35	11	14	11	0	13	2	4548	23	7	8	23	29	1	5	0	0	8	3	0	1	11	2	10	5	2	5	2	9		
50	12	2	23	63	20	1	18	253	4	57	26	29	11	23	37	0	36	83	35	11	7	4	9	17	8584	11	38	30	97	3	15	0	1	3	12	0	2	44	0	40	24	4	2	4	14		
30	14	8	14	38	19	4	6	0	0	1	1	20	6	5	73	0	28	2	25	12	2	26	0	6	23	4309	13	12	32	8	1	0	2	17	14	1	12	6	1	54	10	1	9	2	16		
4	0	0	0	0	24	0	26	0	10	2	0	5	0	5	17	0	1	10	2	0	15	0	0	2	8	1	10513	55	3	0	3	0	6	0	1	0	0	0	0	5	0	0	0	1	1	1	1
46	27	22	24	36	73	3	30	0	10	32	16	102	5	25	75	0	36	76	135	13	36	13	1	36	69	18	124	8372	115	7	10	0	5	30	7	0	3	39	2	48	3	5	4	8	17		
86	44	13	58	128	15	17	60	1	35	31	24	224	3	39	161	0	220	180	302	57	105	24	7	47	176	35	133	137	40871	29	17	0	19	51	19	3	6	53	12	149	12	6	19	15	68		
67	190	9	146	233	24	22	25	2	6	33	13	100	18	7	54	0	1218	89	390	16	14	179	6	23	57	11	15	202	168	3640	1	0	3	92	7	0	11	29	17	87	11	18	5	7	20		
0	0	0	1	1	1	0	1	1	3	38	0	1	0	2	5	0	1	9	0	1	4	0	0	8	3	0	8	2	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
2	7	1	35	16	0	3	0	0	2	2	0	29	8	2	0	0	62	4	18	1	1	58	4	1	0	0	2	12	7	11	0	0	0	100	4	0	2	3	2	0	13	3	26	0	1		
0	5	0	2	1	1	1	3	1	0	0	0	0	0	6	5	0	6	2	1	8	1	0	1	3	3	6	1	16	0	1	0	0	1424	0	4	0	4	0	3	2695	1	0	6	1	8		
94	46	17	75	107	63	15	18	0	3	83	16	109	16	12	128	0	129	73	260	104	24	56	7	40	84	29	39	101	152	22	10	0	5	9271	7	0	4	107	21	39	8	16	7	3	18		
47	4	2	16	14	24	1	8	1	0	5	6	3	13	4	42	0	12	17	18	4	6	2	4	1	11	6	9	19	30	0	3	0	1	12	4536	1	3	8	4	5	6	0	3	4	8		
0	0	2	2	2	0	15	1	0	1	4	0	1	0	0	0	0	5	4	5	1	8	0	0	2	7	1	2	4	6	0	4	0	0	1	0	754	0	0	0	0	0	0	0	0	0		
53	9	9	38	177	25	4	6	1	2	34	3	23	24	1	18	0	51	27	301	9	2	80	0	23	30	20	1	82	45	10	3	0	1	69	6	3	736	23	2	32	14	200	7	7	8		
15	14	9	312	14	14	0	17	0	3	11	4	16	5	9	22	0	21	13	21	15	9	2	10	0	10	12	3	21	16	33	2	8	0	7	10	2	0	1	6880	3	11	4	1	5	2	6	
1	2	0	3	3	2	0	0	0	1	3	0	2	1	0	4	0	8	9	8	9	1	3	0	0	0	0	2	2	0	9	2	0	1	0	0	0	0	0	0	544	6	0	1	0	0	1	
30	27	3	29	30	26	4	13	4	2	26	2																																				

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

2339	0	0	1	2	2	0	0	0	0	2	1	1	0	2	0	0	0	2	1	0	0	1	1	0	3	0	0	1	0	0	0	1	3	0	0	1	1	1	0	0	2	1	0			
28	15354	18	99	170	77	18	36	1	4	106	44	121	5	35	46	0	673	100	393	27	20	58	5	26	88	31	41	139	78	53	15	0	5	33	1	1	2	27	3	93	7	12	5	14	11	
7	5	2207	1	6	1	5	2	0	0	10	1	10	0	6	4	0	2	11	24	1	0	2	1	2	0	2	4	5	3	2	4	0	0	2	0	0	1	1	2	1	1	0	1	10	4	2
58	37	12	6075	97	107	16	12	1	3	66	44	208	9	30	39	0	96	99	438	54	17	142	2	15	128	39	20	87	112	7	7	1	5	47	12	0	2	330	16	145	41	4	15	10	13	
233	35	26	76	15033	99	21	26	2	5	80	62	226	11	47	95	5	117	155	641	58	22	84	11	15	125	106	50	98	133	3	11	2	0	46	28	1	18	29	9	186	42	15	32	46	27	
12	4	6	7	16	3096	2	17	0	3	26	13	12	0	12	9	1	18	12	25	7	14	6	0	10	18	6	6	10	20	1	11	0	0	5	1	1	3	1	9	5	1	1	6	5		
3	6	2	5	27	7	6036	11	0	4	14	2	30	0	9	11	0	12	18	69	4	29	1	2	3	25	1	11	27	29	0	3	0	8	4	0	6	1	5	2	12	1	1	6	5	1	
8	3	3	1	18	2	10543	1	8	16	10	11	1	9	8	0	8	22	28	9	0	1	2	66	34	3	5	7	14	1	12	0	3	0	0	0	0	4	1	8	0	1	1	4	7		
3	8	0	2	11	2	3	4	1684	1	12	2	4	0	6	9	0	14	38	14	2	3	1	3	1	6200	7	6	6	35	2	3	0	5	0	1	0	6	6	0	7	6	1	3	2	0	
2	0	1	0	0	5	0	4	0	3	5852	4	14	1	7	9	0	25	89	23	4	2	6	2	6	28	5	6	8	16	0	7	0	0	0	0	0	0	1	1	1	0	3	0	1		
18	7	0	7	15	11	0	5	0	3	7	2861	42	1	8	13	0	1	13	7	2	1	1	10	1	5	10	2	3	7	0	0	0	0	0	0	4	1	1	3	3	5	4	2	1	8	2
14	3	2	4	14	3	0	1	1	0	7	36	342	7906	2	29	41	1	28	57	77	1	7	22	324	10	69	25	24	26	70	1	6	1	0	13	9	1	0	19	0	24	10	8	14	10	8
84	8	6	38	72	27	6	9	3	1	13	8	60	2184	10	6	0	11	23	75	4	3	29	3	4	26	14	5	19	33	1	1	0	0	13	4	1	2	6	3	46	5	7	6	7	11	
123	4	5	13	73	21	2	4	0	1	13	8	60	2184	10	6	0	11	23	75	4	3	29	3	4	26	14	5	19	33	1	1	0	0	13	4	1	2	6	3	46	5	7	6	7	11	
3	0	0	0	0	0	0	2	0	1	2	2	3	0	4382	1	0	2	1	1	0	0	0	0	0	9	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	1	4	0	0	0
37	3	3	5	13	18	3	5	0	1	10	14	27	0	24	10762	1	7	22	15	3	2	1	1	7	33	23	11	3	12	0	8	0	0	5	2	0	0	5	2	14	6	0	9	137	3	
7	1	0	2	10	3	2	1	2	0	0	4	7	0	53	6	220	3	13	8	1	0	8	0	0	7	5	0	2	5	0	0	27	4	1	1	0	0	0	1	1	7	1	120	2	0	
59	40	18	86	186	84	44	30	2	13	88	56	230	19	30	62	0	13156	90	463	38	35	54	5	14	115	23	33	99	92	40	24	0	4	66	11	2	3	22	12	235	13	7	22	10	14	
61	16	5	33	102	45	11	22	10	8	222	37	71	5	31	60	0	27	13162	115	31	11	51	12	7	82	19	22	37	86	1	27	0	0	42	6	0	1	21	6	39	6	6	11	23	9	
48	21	17	23	95	34	9	22	6	4	31	27	43	0	31	61	0	51	65	12037	16	4	10	11	14	59	22	11	34	108	5	9	1	0	15	10	1	6	15	2	29	11	4	16	12	14	
44	12	3	12	46	21	6	7	0	3	32	7	8	1	11	37	0	36	21	85	6532	3	12	0	3	42	33	15	11	29	0	4	0	2	29	7	0	3	7	17	33	4	1	17	3	2	
12	6	4	16	24	465	19	14	1	14	28	7	64	0	24	21	0	21	54	179	14	6202	13	4	13	54	7	42	218	40	1	6	1	3	16	0	1	0	12	0	56	2	3	3	4	2	
23	5	1	54	49	48	8	5	0	4	20	8	50	6	8	14	0	32	50	170	15	1	1798	2	11	34	7	8	36	38	5	4	0	1	24	1	1	0	11	3	32	14	2	9	7	4	
5	0	0	4	10	4	1	8	0	0	10	56	2962	0	3	8	0	3	16	7	0	1	0	711	2	25	1	3	2	4	1	0	0	1	0	3	0	0	2	0	15	3	1	2	0	3	
13	11	1	11	44	27	15	21	1	14	28	13	19	0	10	10	0	37	30	37	15	0	20	2	4420	42	11	7	13	42	1	7	0	0	4	2	0	0	8	2	12	4	1	7	6	5	
60	9	1	7	92	28	3	8	400	0	32	28	28	2	23	32	1	15	60	63	16	3	4	4	8	8636	23	14	12	46	2	7	0	0	3	5	0	4	16	0	23	18	1	4	12	6	
26	3	1	7	20	11	1	3	0	0	2	2	18	1	8	32	0	7	2	10	4	0	6	0	0	10	4603	8	5	14	1	1	0	0	3	9	0	1	4	2	34	2	1	11	5	5	
19	1	1	2	6	144	0	28	0	14	20	14	39	0	15	27	0	2	38	42	6	44	1	2	2	37	3	0	11	0	6	1	1	0	0	1	28	1	2	4	2	2	2	10	10		
42	16	11	23	55	120	13	25	1	14	37	38	135	3	32	50	0	27	81	262	19	44	20	2	20	76	23	94	8226	65	5	11	1	5	16	6	0	3	24	1	78	5	4	6	9	10	
117	52	21	68	263	43	33	54	2	30	28	73	388	8	61	147	0	170	280	725	54	99	58	8	32	255	73	113	185	39640	11	18	1	15	65	10	2	4	38	8	306	11	14	43	45	40	
47	107	9	146	440	69	49	22	3	8	28	40	215	13	20	37	2	1271	164	641	14	19	165	9	10	70	24	14	205	127	3004	3	0	2	73	5	0	12	12	6	120	12	11	9	16	12	
0	0	0	0	0	0	0	1	4	1	27	1	2	1	2	2	0	3	5	0	1	0	0	1	2	1	3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	4	0	8	6	0	2	0	0	0	3	0	2	13	0	277	7	5	5	3	1	2	4	0	0	0	1	6	3	0	0	63	0	5	2	0	1	0	0	6	0	12	0	1	1	2	
0	5	2	3	1	0	2	3	1	0	0	1	0	0	9	6	1	14	3	2	11	0	1	2	1	6	7	4	5	19	1	1	0	1226	0	2	0	4	0	7	2857	3	0	8	1	2	
73	34	15	82	175	102	43	21	1	10	90	32	188	15	34	87	0	104	121	623	146	29	84	6	28	101	48	32	106	121	9	12	0	5	8623	3	1	3	53	11	51	14	8	54	28	12	
94	4	3	14	103	46	4	10	2	1	16	41	20	167	32	32	0	32	70	33	10	10	3	3	66	16	6	26	54	2	6	1	0	18	3765	0	5	14	1	110	8	0	10	28	6		
0	0	1	1	2	0	38	0	0	0	3	0	2	0	0	0	0	1	6	1	6	1	0	1	12	1	2	2	4	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	6	6	31	233	36	10	5	1	2	31	6	33	15	4	11	0	39	53	378	9	3	67	0	5	39	51	2	34	37	1	3	0	1	21	5	0	698	11	3	65	16	167	18	22	5	
15	17	8	654	25	46	3	20	1	4	14	8	35	5	20	24	0	24	15	66	23	8	24	1	12	21	6	19	26	44	3	10	0	7	16	2	0	1	6312	2	25	11	3	7	3	1	
1	1	0	1	1	3	0	0	0	1	3	0	4	0	0	2	0	2	7	6	5	1	2	1	0	1	1	0	7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	17	4	25	64	34	6	14	8	2	29	42	29	6	13	37	3	63	21	52	29																										

Kernel: (16, 2), Nadam, logcosh

2188	2	1	5	19	5	1	1	3	0	5	7	2	6	4	6	0	6	9	7	2	0	6	3	1	11	9	1	5	13	2	0	0	0	4	12	0	4	9	0	0	5	1	2	1	1	
5	17017	10	16	54	13	1	28	2	3	19	17	35	3	15	21	0	286	41	54	12	13	44	9	19	79	34	34	18	53	32	14	0	6	30	2	0	4	21	1	27	4	12	3	2	10	
5	6	2270	1	4	0	0	1	4	1	4	1	9	0	5	2	0	1	3	2	1	0	4	1	2	1	2	1	3	7	2	3	0	0	2	0	0	1	3	0	0	1	0	0	2	3	
20	74	17	5476	122	54	6	11	6	4	45	27	152	33	18	27	0	130	76	214	29	34	626	13	26	147	92	21	65	172	28	6	2	6	126	13	0	11	603	5	94	29	30	0	5	23	
78	71	28	47	15587	63	3	32	3	3	50	44	166	31	28	67	5	123	99	363	28	39	185	18	25	180	161	47	61	170	12	9	5	1	47	37	1	38	43	5	86	23	39	6	12	33	
5	26	9	5	41	2859	1	17	1	5	22	11	23	2	13	9	0	27	20	29	3	38	42	1	13	24	21	13	13	50	11	14	0	0	27	2	1	2	12	1	8	3	6	0	2	7	
1	24	5	10	130	4	5473	16	0	7	14	3	58	3	10	12	0	35	24	100	2	73	156	5	10	35	30	12	38	73	6	3	0	8	26	0	5	2	11	2	16	2	4	1	2	2	
4	10	14	2	21	2	2	10398	1	6	7	12	20	1	9	8	0	13	11	7	7	0	6	3	189	36	7	15	5	21	0	10	0	3	4	3	0	6	1	9	0	0	0	2	14		
1	9	2	1	9	1	1	3	2022	2	4	2	2	0	3	7	0	10	33	8	1	2	2	0	2	5914	7	7	2	24	3	4	0	4	0	2	0	6	6	0	8	3	1	2	1	2	3
2	0	1	0	0	0	4	0	1905	0	0	2	0	2	1	0	6	3	2	2	1	2	2	4	2	1	2	2	5	0	13	0	0	1	0	0	0	1	0	0	0	1	0	0	2	0	3
11	30	2	17	34	12	0	6	4	3	5617	11	20	1	5	8	0	44	111	25	4	2	50	2	7	53	11	12	9	33	6	8	0	0	19	3	0	2	17	2	12	1	2	0	1	6	
4	7	3	7	20	1	0	0	1	0	7	2800	68	4	9	11	0	4	10	5	2	2	11	13	2	11	15	4	1	13	2	2	0	0	4	4	0	1	4	1	3	5	4	1	1	4	
30	30	6	26	110	18	4	4	8	1	24	160	7699	9	23	33	1	46	50	33	1	11	64	623	21	78	48	24	29	118	8	7	3	0	29	9	0	6	29	1	11	8	10	3	5	14	
15	7	3	2	61	9	0	3	0	2	7	13	25	2426	4	2	0	11	9	56	0	5	26	4	6	35	22	4	14	30	3	1	0	0	9	11	0	3	6	1	22	2	13	2	2	13	
0	0	4	0	1	0	0	1	0	1	3	2	9	1	4357	1	0	2	1	0	0	1	2	4	0	11	2	0	1	4	1	2	0	0	2	2	0	0	2	0	2	0	1	2	0	0	0
20	17	7	6	72	23	1	8	7	1	6	18	41	13	21	10461	1	23	27	45	1	1	57	5	12	40	76	12	4	62	5	9	1	0	39	6	0	9	6	1	15	8	9	9	31	21	
1	1	0	2	12	4	1	1	2	0	0	3	6	3	4	4	370	3	5	6	1	0	26	1	1	10	6	0	7	0	0	42	5	1	1	0	1	0	0	0	4	0	0	0	1	1	
15	106	19	47	206	32	3	26	4	9	41	29	159	19	19	38	1	13468	64	236	19	48	204	6	21	163	58	41	47	148	117	22	0	4	106	12	0	11	27	2	102	9	11	3	2	25	
33	38	7	32	184	40	4	29	17	8	180	34	78	14	25	57	1	43	12774	84	15	16	217	24	27	108	28	21	28	133	17	22	1	0	96	11	0	8	34	1	43	9	17	2	10	29	
29	64	33	30	155	17	4	24	8	4	36	27	81	0	28	52	1	87	73	11535	12	18	51	9	21	66	47	16	43	179	30	10	5	0	106	13	1	22	32	1	36	13	10	4	2	29	
14	34	4	17	149	23	2	9	2	2	33	5	6	18	12	29	0	52	31	61	5731	11	84	0	11	57	212	20	15	69	5	4	0	4	365	14	0	11	16	4	39	3	13	3	3	4	
5	24	5	11	47	68	2	13	1	12	21	4	43	0	16	17	1	32	45	124	10	6699	37	7	13	45	34	43	103	71	4	8	1	3	38	0	0	3	16	1	49	0	5	0	3	6	
5	15	1	29	38	13	2	2	1	4	11	3	32	9	4	10	0	23	24	63	7	5	2087	5	11	32	26	7	24	40	6	3	0	1	26	1	0	1	13	1	19	3	9	1	1	5	
1	5	0	2	16	4	0	3	4	2	7	25	2112	1	1	4	0	7	10	3	0	1	0	1587	3	24	3	4	2	11	1	0	0	2	1	4	0	0	2	0	12	2	1	0	0	10	
7	19	8	3	38	5	5	14	1	10	15	4	12	1	8	4	0	44	8	11	7	0	59	3	4510	23	20	6	10	54	1	5	1	0	17	2	0	3	9	2	7	1	5	3	1	7	
22	11	1	0	56	24	0	10	756	0	27	18	21	3	14	28	0	15	52	32	7	3	3	8	11	8433	32	20	10	55	4	4	1	0	3	8	0	4	26	0	16	5	5	1	3	7	
17	5	2	2	17	5	0	3	0	0	1	7	2	3	15	0	4	0	3	2	0	3	1	1	8	4721	4	0	11	1	0	0	1	2	11	0	0	4	2	2	11	2	1	6	1	5	
7	10	8	1	11	45	0	23	0	15	8	12	29	1	14	26	0	1	43	16	3	139	1	7	4	23	15	10170	20	11	0	12	0	6	3	1	0	0	0	23	0	2	1	2	7		
16	58	22	16	111	66	5	25	1	11	34	30	123	6	27	48	0	41	80	178	11	137	100	8	27	85	65	105	7899	133	38	12	4	5	59	8	0	6	35	2	70	1	22	4	5	19	
39	80	20	30	224	10	3	51	3	22	14	40	212	16	41	89	0	145	172	283	26	85	88	14	42	242	136	93	89	40998	22	15	1	19	59	11	0	6	41	4	126	6	18	8	13	55	
5	251	9	65	391	18	13	20	4	5	20	12	94	20	9	24	2	995	67	216	10	21	396	11	17	77	60	18	68	149	3985	3	2	2	2	79	6	0	18	6	3	54	6	36	1	2	15
0	0	0	0	2	0	0	2	3	0	18	0	1	1	2	2	0	4	8	0	1	1	0	0	3	5	1	3	0	1	0	1655	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
0	4	0	6	3	0	1	0	0	0	0	0	1	2	3	0	251	4	4	2	1	2	5	3	0	0	0	3	5	1	0	122	0	6	4	0	1	0	0	0	0	5	0	0	0	1	
0	15	3	0	5	0	1	2	1	1	0	0	0	0	6	6	1	25	3	1	3	1	1	3	1	6	10	4	2	24	5	1	0	1557	2	4	0	5	0	5	2509	1	1	3	0	3	
13	59	18	35	242	39	10	20	6	2	52	19	117	38	16	54	0	80	69	165	28	30	249	13	26	93	81	31	47	142	26	12	0	5	9455	6	0	16	49	1	30	4	22	1	2	15	
32	4	3	5	45	15	1	8	3	1	4	20	3	74	5	21	0	20	24	9	3	7	24	6	4	43	13	7	12	42	1	2	0	0	8	4407	0	6	8	0	10	4	7	2	3	7	
0	2	1	1	5	0	16	1	0	0	2	0	8	0	0	0	5	3	10	0	0	12	10	0	2	15	4	1	1	9	0	4	0	0	1	0	718	0	0	1	2	0	0	1	0	0	5
7	8	8	14	178	7	2	6	1	2	11	3	22	20	1	8	0	27	27	100	4	6	115	3	3	35	86	2	14	31	2	3	1	1	21	6	1	1227	11	0	24	6	153	3	4	4	0
6	23	12	209	20	10	0	16	2	3	7	5	19	10	10	18	0	24	14	22	8	11	62	3	11	20	12	18	4	40	8	9	0	7	28	3	0	2	6880	1	12	4	7	4	0	7	
0	1	1	2	2	1	0	0	0	1	2	0	4	2	0	2	0	10	4	3	5	0	21	1	0	4	4	0	8	4	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
21	41	8	15	120	16	3	11	12	2	26	29	39	27	9	34	6	94																													

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

1976	0	0	1	2	1	0	3	1	0	0	0	0	3	1	0	0	0	0	7	0	0	1	1	0	0	0	1	50	0	0	0	0	14	0	6	1	1	0	0	0	0	3
2	845	9	9	8	26	12	9	9	4	0	3	10	10	3	1	1	26	0	376	7	81	2	3	2	1	2	0	11	3	1	33	25	30	22	13	13	16	0	0	0	0	
7	0	2456	1	9	26	10	11	7	12	0	1	2	9	12	0	10	55	0	272	45	26	11	0	6	4	0	8	12	0	2	53	16	700	20	11	3	5	0	2	1		
16	5	2	3886	12	26	7	48	19	8	0	0	61	19	8	8	4	8	0	1131	35	39	7	57	1	0	5	18	25	3	0	13	19	12	26	160	7	4	0	6	0		
5	1	3	4	2111	11	3	9	58	4	0	133	7	1	6	0	1	7	0	27	7	3	3	8	1	0	0	1	0	0	0	16	15	5	24	4	0	1	0	3	1		
16	4	17	34	19	15917	109	20	7	15	0	3	22	48	31	2	40	30	1	265	43	46	18	48	2	1	0	12	17	0	4	31	42	52	83	31	18	11	0	27	2		
10	18	23	24	16	219	7726	19	33	46	0	10	49	68	13	7	14	40	0	690	118	104	14	12	4	5	1	31	17	3	10	77	76	33	31	16	52	10	0	16	4		
59	10	21	140	30	43	10	9717	30	49	0	8	87	46	18	26	9	33	0	701	44	94	19	137	2	2	1	7	3	1	3	42	41	22	75	214	16	1	2	0	2		
16	16	52	23	270	37	32	23	4809	28	0	42	35	45	22	11	1	42	1	90	142	61	21	5	4	6	1	14	40	3	4	98	43	27	49	24	13	13	0	1	1		
7	9	25	6	30	41	26	22	31	7842	0	4	7	37	26	6	12	21	0	122	53	48	32	2	6	4	4	10	9	1	18	24	19	33	6340	21	41	5	0	2	2		
0	0	1	1	0	0	3	0	1	102	0	1	4	0	2	1	2	0	4	82	15	0	1	1	1	0	2	0	0	0	0	9	0	5	1	0	0	0	170	0	0		
7	9	10	7	1177	18	19	12	137	38	1	1236	24	23	3	3	2	14	0	61	36	13	5	7	1	2	0	5	10	1	4	49	30	4	59	9	9	3	0	1	2		
23	7	7	19	36	80	14	40	21	23	0	6	8445	50	12	3	9	29	0	2227	55	49	10	4309	2	1	0	14	15	1	1	45	70	17	87	84	39	3	0	4	0		
13	20	41	50	25	89	53	26	64	81	0	18	110	19921	37	5	14	72	2	2189	140	174	38	34	13	2	1	29	79	6	12	135	84	56	94	33	109	30	3	4	3		
16	0	0	0	0	1	1	1	0	0	0	0	0	4440	0	1	2	0	0	0	0	1	5	0	0	0	1	0	0	0	0	0	64	0	2	0	0	1	0	0	37		
2	9	8	83	6	14	16	148	17	44	0	1	18	27	10	2801	2	9	0	53	9	36	208	5	0	0	1	133	0	1	19	22	11	1	15	9	4	0	1	0	0		
0	0	0	0	0	12	4	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	
12	29	26	13	24	34	17	21	22	24	0	2	26	29	6	3	7	2642	0	845	82	106	7	8	2	12	3	14	27	2	4	105	31	28	49	31	19	7	0	3	1		
0	1	3	6	2	5	4	8	8	12	0	1	7	9	0	2	2	6	272	85	40	9	3	1	1	0	5	3	11	3	5	28	11	1	7	5	3	26	0	0	0		
14	4	11	18	7	33	7	7	10	8	0	0	22	16	7	0	1	9	0	5207	33	18	3	18	0	2	0	4	10	0	1	26	24	18	32	13	17	4	0	4	0		
5	1	25	27	21	38	19	26	21	22	8	2	51	45	8	4	10	27	2	915	2461	60	4	5	3	9	1	8	14	1	4	68	40	42	39	32	19	11	5	2	0		
14	18	28	11	23	65	27	32	35	42	0	6	15	41	12	5	6	64	1	583	53	7491	17	8	9	18	3	5	20	0	3	74	43	66	68	29	10	66	2	2	2		
2	0	1	0	2	4	1	24	0	2	0	0	6	8	141	1	1	0	3	1	0	3219	1	0	0	0	4	0	0	0	0	3	3	7	1	1	1	1	0	1	0		
3	0	0	0	2	0	0	0	1	0	0	58	0	1	0	1	0	0	15	0	2	1	3947	0	0	0	0	0	1	0	0	0	3	0	8	3	0	0	0	0	0	0	
2	0	13	1	2	4	0	4	3	2	0	0	2	2	4	0	2	1	0	82	18	26	2	2	807	1	0	1	2	0	0	1	6	8	3	7	1	2	0	0	0		
0	2	2	0	0	1	2	2	1	0	0	0	1	2	0	0	0	195	0	1	3	3	1	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0		
1	0	0	1	0	0	0	1	6	0	0	1	2	0	0	0	3	0	87	0	2	1	1	0	0	0	46	0	8	0	1	1	36	1	1	2	5	0	0	0	0	0	
11	4	2	64	11	23	20	10	11	7	0	1	28	28	11	32	3	12	0	893	32	18	12	6	2	0	1	4721	13	1	4	16	37	7	19	119	15	4	0	0	5		
6	0	1	5	2	1	2	2	1	2	0	0	9	2	3	0	0	3	0	88	3	5	3	9	0	0	2	2	1046	0	3	1	4	2	6	5	3	1	0	0	1		
4	23	0	10	7	13	15	3	6	5	0	5	16	19	8	1	1	10	1	54	40	16	2	1	0	1	2	3	7	311	9	15	15	3	4	9	13	3	0	0			
1	3	3	11	2	16	9	5	6	12	0	5	24	12	2	1	0	0	0	161	15	18	5	2	0	0	6	2	70	1	736	15	19	5	6	5	18	0	0	2	1		
22	22	235	51	91	124	110	76	96	155	0	19	112	113	36	9	15	205	2	3131	380	422	30	25	6	41	17	49	92	3	17	7862	125	74	135	90	50	79	0	10	6		
12	1	4	4	12	16	8	3	4	6	0	0	5	9	13	0	4	5	0	103	16	9	7	5	0	3	16	1	9	0	0	17	3650	21	29	10	23	1	0	2	1		
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	475	0	0	0	0	0	0	0	0	0		
1	0	1	0	5	5	1	2	2	34	0	0	0	4	0	1	2	0	1	1	1	1	1	0	0	0	0	0	0	0	2	2	4978	0	0	0	0	0	0	0	0		
6	0	8	21	7	25	4	24	3	6	0	0	14	11	7	5	2	15	1	55	22	22	7	11	1	2	2	3	6	0	1	15	4	9	25	2699	0	1	0	4	1		
3	1	0	0	2	3	5	2	1	3	0	1	5	1	2	0	0	6	0	196	15	14	1	3	0	0	0	2	9	0	0	6	38	9	12	3	477	3	0	0	2		
1	0	1	0	2	0	0	0	1	0	0	0	1	0	0	1	0	0	2	1	7	1	2	0	0	0	0	0	0	0	0	6	0	5	5	1	0	494	0	0	0		
4	1	16	3	3	15	16	7	12	32	0	1	7	17	5	3	3	22	0	52	1782	32	8	1	2	1	3	1	3	3	1	52	9	7	9	7	6	1	119	1	0		
0	0	1	15	1	8	1	0	1	1	0	0	2	0	2	0	1	3	0	16	7	3	0	0	0	1	2	1	1	0	1	5	0	1	1	4	0	0	106	0	0		
5	0	1	0	0	1	0	0	0	0	0	0	0	0	282	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	0	0	0	1	0	0	0	0	269		

0.7452219230481948

Kernel: (4, 2), Adamax, cosine_proximity

2019	0	0	1	1	2	0	1	1	0	0	0	0	2	1	0	0	0	0	0	0	0	1	0	0	0	1	31	0	0	0	3	0	1	1	0	0	0	0	0	7
10	856	11	11	5	44	18	12	18	5	0	3	12	11	4	5	1	37	2	211	13	60	2	3	1	0	1	2	36	4	6	99	15	15	15	15	7	57	0	1	0
11	2	2726	4	7	37	11	21	14	9	0	0	7	8	13	2	11	68	0	146	70	13	12	0	6	0	1	6	16	0	4	126	7	424	11	14	2	7	2	6	1
22	5	4	4497	11	33	7	78	21	6	1	0	57	17	9	18	5	10	4	370	55	12	9	48	0	0	1	43	38	2	10	26	12	7	19	224	4	12	0	8	0
9	0	3	3	2096	16	2																																		

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

2052	0	0	1	1	1	0	2	0	0	0	0	0	1	1	0	0	0	0	3	0	0	1	0	0	0	0	1	3	0	0	0	2	0	2	1	0	0	0	0	1	
14	362	4	41	8	31	16	14	9	9	0	3	8	6	2	5	1	39	1	628	22	77	2	1	1	0	1	52	66	3	2	64	11	8	15	59	3	38	0	1	1	
19	0	2147	20	17	32	13	23	14	15	0	1	6	13	10	1	8	54	0	492	212	27	14	0	9	0	1	30	57	0	3	194	13	269	22	73	2	7	0	6	1	
25	0	4	4129	13	21	5	61	8	3	0	1	7	5	7	7	4	6	0	493	36	7	8	20	0	0	0	71	10	1	0	4	5	3	20	717	0	2	0	2	0	
10	0	0	8	2176	9	0	14	46	3	0	93	8	1	7	0	1	3	2	15	13	2	4	2	1	0	0	7	7	0	0	13	6	3	15	12	0	1	0	0	1	
37	1	11	89	23	15591	68	43	13	4	0	5	25	34	32	4	32	21	1	444	112	22	23	27	2	0	0	49	36	0	4	49	29	14	68	117	5	16	0	34	3	
34	5	14	96	21	302	6775	46	47	32	0	19	64	51	13	15	24	38	3	931	349	61	20	6	4	0	1	370	33	1	7	110	50	8	27	33	9	13	1	22	4	
83	2	8	63	28	30	7	10244	13	9	0	3	6	11	13	16	8	10	0	234	38	25	20	46	3	0	2	19	2	1	0	32	19	7	43	710	1	5	1	0	3	
36	1	23	57	431	35	23	46	4528	20	0	56	24	23	23	16	1	24	3	56	236	25	23	2	3	0	0	65	91	1	3	92	22	10	36	96	8	22	1	1	2	2
34	3	19	25	41	60	22	77	50	8117	0	15	7	42	34	18	11	26	1	224	141	48	38	1	4	0	6	70	53	0	24	57	17	13	5471	138	27	7	1	2	4	
0	0	0	1	0	0	0	0	0	120	0	2	3	0	3	1	1	0	4	77	2	0	0	2	0	1	0	1	0	0	6	0	1	1	2	0	0	180	0	1	0	1
16	1	3	13	1398	17	11	22	95	20	1	1089	16	18	5	5	1	7	1	53	54	2	5	1	0	0	0	30	24	0	0	39	14	2	45	31	3	6	1	1	1	
48	0	5	62	42	89	12	106	21	15	0	4	9041	29	16	4	9	33	0	3009	126	16	13	1732	2	0	1	25	21	0	2	63	41	6	73	1160	13	11	1	2	4	
37	14	23	230	43	81	44	63	82	84	0	34	133	17302	45	11	12	63	5	3943	467	118	41	12	16	0	1	223	202	1	28	183	54	23	71	127	48	41	2	2	0	
68	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25
6	0	1	42	5	9	4	185	10	10	0	2	2	10	10	2541	3	1	0	25	4	4	208	1	0	0	0	617	0	0	2	6	2	0	6	27	0	0	0	0	0	
0	0	0	0	0	24	4	0	2	0	0	0	0	0	3	0	2696	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	
35	7	16	46	33	33	15	29	21	14	0	6	21	21	7	4	7	2009	1	1214	154	71	8	3	2	1	1	62	102	0	4	172	26	14	35	106	8	6	1	7	1	
0	0	2	10	3	3	3	11	7	5	0	0	1	4	0	2	2	2	318	71	59	3	3	0	1	0	1	8	13	1	0	16	5	0	5	12	0	24	0	0	0	
20	0	6	32	6	25	4	12	8	7	0	2	11	11	6	1	1	9	0	5198	67	3	3	9	0	0	0	18	13	0	0	39	8	8	20	46	4	6	0	2	3	
18	0	13	62	25	31	12	32	17	8	15	2	21	29	8	4	10	21	4	917	2457	23	3	2	2	0	0	43	47	0	1	67	17	13	34	112	5	13	12	3	2	
37	20	24	75	31	79	33	74	52	54	0	9	17	50	15	11	5	101	2	737	162	6552	21	5	12	3	1	73	88	0	8	215	30	39	51	119	3	200	3	3	3	
5	0	0	0	0	1	1	34	0	0	0	0	0	2	8	131	1	1	0	2	0	0	3229	1	0	0	0	16	0	0	0	0	0	1	3	2	0	1	0	0	0	
7	0	0	2	4	1	0	8	0	1	0	0	256	0	1	0	1	0	0	70	0	0	1	3643	0	0	0	1	1	0	0	0	2	0	10	37	0	0	0	0	0	
4	0	13	9	3	4	0	7	4	1	0	0	1	1	3	0	2	3	0	88	32	15	2	1	770	0	0	6	11	0	1	1	7	3	3	15	0	1	0	0	0	
1	1	4	1	0	0	2	2	2	0	0	0	0	1	2	0	0	184	0	4	3	3	1	0	0	0	0	1	0	0	0	5	0	0	0	0	0	1	1	0		
6	0	0	1	0	0	0	2	1	1	0	0	1	1	0	0	0	2	0	97	4	0	1	1	0	0	31	7	13	0	3	1	21	0	1	11	2	0	0	0	0	
16	0	0	23	9	8	5	13	3	0	0	0	2	4	10	20	2	4	0	210	12	0	12	1	1	0	0	5626	7	0	0	5	9	2	14	181	1	2	0	0	1	
54	0	0	7	1	1	1	3	1	2	0	0	7	1	3	1	0	1	0	100	4	1	3	4	1	0	0	7	988	0	2	1	0	0	5	20	0	3	0	0	1	
8	19	0	27	7	12	10	8	9	4	0	7	13	15	6	4	1	8	3	86	73	6	2	1	0	0	0	43	42	160	10	12	6	2	2	40	5	3	0	0	1	
6	3	1	38	2	12	7	10	5	7	0	5	11	5	4	4	0	1	0	155	29	10	6	2	0	0	1	72	130	1	621	10	10	1	2	22	3	2	0	1	0	
69	7	97	171	101	118	69	96	98	93	0	26	70	86	37	18	13	176	13	2993	847	185	31	10	5	0	7	190	303	0	20	7512	58	22	100	360	12	106	0	9	9	
58	1	3	6	12	17	5	12	6	3	0	1	6	7	23	3	5	3	1	287	53	8	6	2	0	0	13	14	19	0	0	21	3357	13	20	29	7	1	0	4	3	
0	0	9	0	0	1	0	1	0	0	0	0	0	0	1	0	1	0	0	8	7	0	0	0	0	0	1	1	0	0	0	0	443	2	2	0	1	0	1	0	0	
6	0	1	0	0	7	7	1	2	4	54	1	0	0	0	6	0	1	2	0	8	5	1	4	1	0	0	1	0	5	0	0	0	0	0	4907	19	1	1	0	0	0
8	0	2	8	6	13	2	15	2	0	0	0	1	6	6	4	1	6	1	10	13	4	7	1	1	0	0	6	4	0	0	8	0	1	7	2902	1	2	0	0	1	
17	1	0	0	2	3	4	3	0	2	0	1	3	1	2	0	0	5	1	316	35	11	1	2	0	0	0	6	10	0	0	9	32	6	10	8	322	6	0	0	6	
2	0	0	0	2	0	1	1	0	0	0	0	0	2	0	0	1	0	0	0	4	5	1	2	0	0	0	1	0	0	0	13	0	2	4	6	0	483	0	0	1	
7	1	6	6	4	11	7	10	10	20	4	1	2	10	5	6	3	19	1	73	1782	12	7	0	1	0	0	14	13	1	0	41	6	4	9	19	1	0	149	1	1	
0	0	0	50	2	11	0	0	0	0	0	1	1	0	2	0	1	1	0	16	12	0	0	0	0	0	2	0	0	0	1	2	0	1	3	1	0	0	78	0		
44	0	1	0	0	0	0	0	0	0	0	0	0	0	340	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	185	

0.7172988114204499

Kernel: (16, 2), RMSprop, categorical_crossentropy

1999	0	0	2	2	1	0	3	1	0	0	0	0	3	0	0	0	0	0	11	1	1	0	1	0	0	0	1	41	0	0	0	1	0	3	1	0	0	0	0	1
4	452	2	19	5	19	8	10	7	5	0	0	9	12	2	0	1	22	0	878	8	31	1	2	1	0	0	2	33	5	0	29	7	4	8	16	2	24	0	0	0
11	0	1413	13	5	18	7	12	14	11	0	0	6	22	6	0	6	55	0	1369	203	15	5	0	7	0	0	10	36	0	1	369	4	158	11	24	1	11	0	1	1
18	0	3	4137	7	22	3	65	7	5	0	0	41	19	4	2	1	6	0	955	55	4	7	36	0	0	0	14	35	1	0	10	3	3	17	217	4	4	0	0	0
7	1	0	13	2081	10	0	12	86	3	0	14																													

ARat_2017_06_01_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

1184	1	55	0	3	2	0	0	0	2	1	0	2	0	1	0	2	4	0	3	0	128	0	0	3	1	3	2	0	1	0	0	1	3	0	3	1	0	0	2	0	
1	1931	0	3	1	2	0	0	1	7	1	0	0	0	1	0	1	2	1	4	0	7	1	0	1	0	0	1	1	1	1	1	0	6	2	0	1	0	0	0	5	2
6	5	2344	5	3	22	0	0	2	22	1	0	6	0	7	0	3	12	3	8	0	24	2	0	8	2	9	1	0	9	3	3	4	5	0	9	1	0	0	7	102	
0	1	2	2859	1	8	0	0	0	9	2	0	4	0	2	0	3	31	2	6	0	11	1	0	8	0	2	0	0	6	3	0	1	6	0	2	1	0	0	6	8	
0	0	3	3	1337	3	0	0	0	9	0	0	1	0	1	0	0	2	0	0	7	0	0	0	0	5	0	0	1	3	291	1	2	0	4	6	0	0	3	3		
3	34	32	9	13	14683	0	0	4	64	3	0	22	1	9	0	13	38	703	43	0	96	12	0	19	5	19	1	1	16	6	9	42	32	0	17	19	0	0	7	44	
0	8	22	0	1	4	0	0	67	27	0	0	72	1	0	0	0	1	1	30	0	16	3	0	2	26	2	0	25	0	16	3	0	1	0	0	85	0	0	0	4	
1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	162	0		
0	2	2	0	1	4	0	0	1272	3	0	0	0	0	3	0	0	2	0	2	0	4	0	0	3	7	3	0	0	2	1	0	2	0	0	1	0	0	0	2	1	
1	6	17	1	3	21	0	0	6	6029	0	0	6	0	4	0	3	11	6	18	0	31	7	0	52	3	25	1	1	6	3	11	3	13	0	3	2	0	0	7	12	
2	0	0	0	1	3	0	0	0	3	984	0	0	1336	0	0	0	0	9	0	0	0	0	0	0	0	0	1	0	0	0	4	0	0	0	0	0	0	0	3	1	
0	0	1	0	0	5	0	0	1	1	93	0	0	72	0	0	0	0	0	0	0	8	164	0	1	1	0	0	3	0	0	42	0	0	1	0	0	0	119	1		
2	7	19	1	8	11	0	0	3	22	1	0	4196	0	6	0	0	4	3	10	0	11	0	0	3	3	0	1	5	2	9	0	8	0	7	7	0	0	4	3		
0	0	7	3	2	11	0	0	0	8	45	0	2	1427	2	0	3	11	3	4	0	4	2282	0	1	0	0	1	0	9	1	1	5	3	0	2	0	0	0	12	4	
0	0	3	2	4	1	0	0	1	2	0	0	2	0	2197	0	4	0	0	2	0	5	0	0	0	7	1	0	0	0	0	2	1	0	0	1	2	0	0	0	4	
2	41	45	1	0	48	0	0	109	50	2	0	13	1	6	0	0	5	2	70	0	52	76	0	69	67	11	0	64	0	37	14	8	3	0	6	101	0	0	2	34	
3	1	6	2	0	5	0	0	0	8	0	0	2	0	0	0	1350	6	1	1	0	35	0	0	0	0	1	4	0	4	1	0	2	3	0	3	0	0	1	2		
1	7	18	12	9	34	0	0	5	41	0	0	16	0	7	0	1	9898	4	30	0	20	3	0	9	2	7	2	2	18	5	8	1	6	0	10	1	0	0	14	13	
3	7	6	0	2	3839	0	0	2	17	1	0	5	0	4	0	1	11	2546	12	0	11	4	0	3	1	14	0	0	10	1	5	10	3	0	3	2	0	0	0	6	
10	35	87	20	10	172	2	0	25	116	5	0	17	4	16	0	10	48	29	9374	0	203	150	1	88	10	154	0	2	37	22	28	616	28	0	12	23	0	0	50	125	
0	2	27	0	0	3	0	0	2	2	0	0	2	0	1	0	0	1	0	1	0	168	1	0	0	4	1	0	0	2	0	1	0	0	0	0	0	0	1	64		
9	4	44	2	0	35	0	0	0	33	2	0	8	2	3	0	7	17	2	13	0	6449	10	0	6	0	14	3	1	8	1	1	19	7	0	6	3	0	0	17	53	
2	16	65	6	3	141	2	0	6	76	7	0	5	6	1	0	4	11	16	62	0	108	7380	0	99	6	8	2	5	32	5	2	53	16	0	7	6	0	0	31	74	
0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	136	0	0	0	0	0	0	0	0	0	0		
8	19	59	14	2	93	3	0	10	226	3	0	10	1	10	0	4	28	16	56	0	56	37	0	5006	12	28	2	2	28	16	13	29	43	0	11	12	0	0	29	75	
2	0	7	3	1	3	0	0	1	10	1	0	1	0	4	0	0	2	0	3	0	3	1	0	2	1113	2	0	0	2	0	3	3	0	0	6	1	0	0	0	5	
0	0	0	3	2	8	0	0	2	12	1	0	4	0	1	0	1	9	3	147	0	8	1	0	6	1	0	2	2	2	4	182	4	0	2	5	0	0	7	1		
3	0	3	0	1	2	0	0	0	0	0	0	1	0	1	0	2	2	0	2	0	15	1	0	3	0	0	0	306	0	1	1	1	2	0	0	2	1	0	0	2	
2	2	3	0	2	2	0	0	0	11	2	0	0	1	2	0	1	1	2	5	0	3	4	0	9	0	0	0	1414	5	0	2	1	1	1	0	3	0	0	1	0	
1	5	10	2	2	21	0	0	1	22	3	0	4	0	6	0	2	19	8	19	0	30	6	0	10	0	18	0	0	4181	1	7	1	7	0	4	5	0	0	12	12	
2	4	22	3	6	15	0	0	2	27	0	0	3	0	3	0	0	9	2	25	0	20	16	0	23	4	8	0	2	10	532	11	3	2	0	4	6	0	0	1	20	
1	5	9	4	333	13	0	0	1	20	0	0	2	0	4	0	3	10	6	6	0	10	1	0	9	1	9	1	0	1	4	1298	2	5	0	8	6	0	0	4	11	
2	19	57	9	2	144	0	0	6	63	16	0	14	0	12	0	9	39	22	1366	0	116	32	1	52	8	203	3	1	23	5	5	8791	18	0	10	10	0	0	215	44	
0	2	2	4	1	3	0	0	0	11	0	0	1	0	1	0	0	6	4	1	0	5	0	0	7	0	5	0	0	2	0	0	1	3418	0	1	0	0	0	3	1	
0	62	31	0	0	41	0	0	99	52	0	0	31	0	59	0	0	4	0	18	0	3	26	0	19	75	0	0	90	26	17	21	3	3	0	3	107	0	0	1	38	
3	1	22	4	7	15	0	0	1	16	0	0	4	0	7	0	2	12	1	19	0	15	5	0	4	4	8	0	2	14	2	10	7	8	0	1932	11	0	0	2	13	
2	2	7	1	2	4	0	0	1	10	0	0	2	0	0	0	2	2	3	0	10	1	0	4	1	5	0	0	0	3	3	4	3	0	2	1061	0	0	1	5		
0	1	0	0	0	1	0	0	0	2	0	0	1	0	0	0	0	1	0	0	0	63	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0		
0	1	1	0	0	6	0	0	0	3	0	0	0	0	0	0	3	0	2	0	0	0	0	0	0	0	2	0	0	1	0	0	4	1	0	1	0	0	641	2		
1	8	25	3	1	46	0	0	3	26	2	0	7	0	1	0	5	13	3	125	0	27	9	1	10	1	10	0	0	20	1	2	210	14	0	7	1	0	0	7046	19	
1	2	62	3	2	16	0	0	3	22	0	0	6	0	5	0	3	14	2	10	0	20	5	0	8	2	10	0	0	9	3	2	9	4	0	2	3	0	0	5	2144	

0.8265060680131179

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

1249	0	10	0	4	2	1	0	2	2	1	0	4	0	2	0	85	3	0	3	0	23	0	0	3	2	2	1	0	1	1	0	0	1	0	3	3	0	0	0	0
2	1910	0	3	1	4	0	0	2	9	1	0	5	0	2	1	1	8	0	4	0	5	2	0	4	0	0	1	9	1	1	0	2	2	0	0	0	0	4	1	
21	4	2229	6	5	34	2	0	3	34	0	1	12	0	9	3	12	16	3	19	0	23	8	1	63	2	9	1	2	18	9	5	2	3	0	18	4	0	0	7	50
1	1	2	2863	3	4	0	0	1	14	2	0	5	0	2	0	3	39	2	5	0	5	0	1	11	0	1	0	0	4	3	0	1	3	0	2	1	0	0	4	2
0	0	1	0	1580	1	1	0	0	7	0	0	2	0	1	0	1	4	0	0	2	0	0	2	1	0	3	0	0	1	1	61	0	1	0	3	8	0	1	2	1
8	28	12	10	21	14891	4	0	6	71	3	1	35	1	10	8	36	58	442	45	0	58	13																		

Kernel: (4, 2), Adamax, cosine_proximity

1302	1	30	0	3	5	0	0	1	2	1	0	4	0	1	0	11	6	1	6	0	16	0	0	1	2	1	2	0	0	0	0	2	1	0	3	3	0	1	2	0	
1	1901	1	3	1	7	1	0	1	4	1	0	6	0	0	1	1	12	1	6	4	2	1	0	2	0	0	1	2	1	2	2	9	2	0	1	0	0	0	5	3	
15	3	2340	5	4	27	2	0	2	24	0	1	11	0	5	1	5	14	3	17	4	14	5	0	9	2	4	1	0	4	10	4	9	2	0	12	2	0	0	9	68	
0	1	2	2896	1	3	0	0	0	6	2	0	4	0	1	0	1	34	2	5	0	4	0	0	6	0	1	0	0	0	2	1	1	3	0	2	0	0	0	4	3	
0	0	2	1	1419	2	1	0	0	9	0	0	3	0	1	0	0	5	0	1	1	2	1	2	0	0	1	0	0	1	220	0	1	0	4	3	0	1	3	1		
7	28	26	14	15	14598	3	0	2	52	5	0	33	0	6	8	14	58	836	53	2	59	16	1	9	7	10	3	3	6	10	50	19	3	21	7	0	3	9	17		
0	0	0	0	0	0	410	0	0	0	0	0	4	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	164	0	
0	3	5	0	1	8	0	0	1251	7	0	0	3	0	3	0	0	4	0	1	0	2	1	0	2	4	3	0	0	7	2	2	2	0	0	0	0	0	2	3	1	
3	7	15	5	5	31	2	0	4	6014	0	0	11	0	2	0	3	20	8	37	2	15	9	0	49	1	14	1	2	4	3	7	3	9	5	4	1	0	3	5	8	
2	1	0	0	0	1	0	0	0	1	1232	1	0	1114	0	0	0	1	8	0	0	0	39	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	1	3	0
0	0	0	0	0	1	0	0	0	0	8	476	0	1	0	1	0	1	0	0	1	4	7	0	0	0	0	0	1	0	0	4	0	0	0	0	0	0	0	1	7	0
3	1	12	3	7	7	4	0	0	13	1	0	4254	0	1	1	0	6	3	10	2	4	0	0	2	1	0	0	1	1	5	0	2	0	6	2	0	0	6	0	0	
0	0	3	7	2	18	2	0	0	8	87	2	2	1827	1	1	3	17	3	6	1	1	1818	1	2	0	0	1	0	8	0	1	6	4	1	2	0	0	1	14	3	
0	4	3	3	6	1	0	0	2	3	0	0	4	0	2188	0	3	4	0	0	1	1	1	0	0	4	0	0	0	0	3	1	0	1	3	1	0	0	0	0	4	
1	0	3	0	0	6	8	0	0	4	1	1	14	0	3	865	0	5	1	11	0	2	0	0	1	0	1	0	0	3	0	3	0	1	0	0	0	0	2	3		
11	1	3	2	0	3	0	0	0	7	0	0	1	0	0	0	1372	10	2	1	0	7	0	0	2	0	0	1	0	1	1	2	3	0	1	0	0	0	4	1		
1	2	10	11	5	16	0	0	3	22	0	1	16	0	2	1	1	10040	4	18	2	8	1	0	1	4	1	1	2	3	2	1	1	1	0	8	1	0	3	8	3	
1	4	6	2	3	3589	1	0	1	14	1	0	8	1	4	3	3	17	2800	15	0	4	3	1	1	1	8	0	0	4	1	3	20	2	2	3	1	0	1	0	1	
13	15	67	20	10	165	5	0	13	88	5	3	26	4	11	11	13	78	36	9739	4	49	94	0	46	8	40	0	2	9	20	31	710	18	4	21	10	0	6	83	52	
0	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	1	262	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
53	6	49	9	3	107	3	0	1	60	3	2	18	1	4	5	14	32	4	42	12	6128	14	0	24	0	10	4	1	9	9	4	33	8	3	13	3	0	0	28	60	
5	13	56	14	3	401	3	0	4	98	8	3	10	8	1	5	13	29	24	182	0	42	6839	1	216	4	4	2	3	24	30	7	87	12	6	20	1	0	2	52	31	
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	
11	12	78	17	3	147	3	0	10	247	3	0	17	1	9	3	6	43	22	76	1	26	54	2	4832	10	12	5	2	12	38	25	57	30	1	22	11	0	2	54	57	
2	0	7	4	1	6	0	0	0	18	1	0	2	0	5	0	0	3	0	3	0	1	2	1	1	1096	1	0	2	1	0	5	2	0	0	10	0	0	0	0	5	
0	1	4	4	5	11	1	0	2	17	0	2	3	0	1	5	4	10	5	408	2	5	0	0	4	0	0	1	1	3	581	3	1	4	5	0	4	24	1	1		
6	0	2	0	1	2	0	0	0	0	0	0	3	0	1	2	6	2	0	3	0	3	0	0	2	0	0	0	307	0	1	1	4	0	0	2	1	0	0	0	2	
2	2	3	0	2	4	0	0	0	12	2	0	1	2	2	0	1	2	3	0	1	2	0	6	0	0	0	0	1419	1	0	2	2	0	3	0	0	0	2	0	2	
4	4	18	6	2	38	4	0	1	54	2	4	9	0	6	1	2	34	12	83	3	23	15	0	26	0	9	0	1	3976	5	11	3	8	3	9	3	3	2	19	16	
3	2	14	4	9	20	0	0	2	34	0	0	6	0	2	0	0	11	2	25	0	5	18	12	10	1	2	0	2	4	547	14	9	2	0	6	6	0	0	6	7	
1	6	4	4	4	457	13	0	0	1	26	0	0	3	0	4	0	2	18	6	4	0	4	1	2	3	1	5	1	0	0	2	4	0	11	0	0	2	3	3		
6	15	49	13	3	124	0	0	4	54	11	19	18	0	12	9	12	61	25	1368	5	34	20	0	32	5	35	1	1	8	6	7	8996	13	0	12	3	0	7	311	18	
1	1	7	3	2	8	0	0	0	23	1	1	4	0	1	0	0	10	8	1	0	5	3	0	11	0	4	1	0	1	1	3	2	3361	1	4	0	0	1	7	3	
0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	1	1	819	0	0	0	0	0	0	
3	2	14	4	7	15	0	0	0	21	0	0	5	0	5	0	2	16	2	15	0	3	5	0	1	4	4	0	2	6	6	6	2	0	1986	7	0	0	2	4		
2	3	14	1	6	17	0	0	2	15	0	0	5	0	0	0	2	6	2	11	0	9	3	1	7	1	4	0	0	3	3	11	3	1	0	11	992	0	2	3	4	
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	54	0	0	0	
0	1	0	0	0	3	0	0	0	1	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	1	0	0	580	77	0	
0	7	11	5	1	33	1	0	2	23	2	0	10	0	1	4	3	26	3	98	3	7	1	0	3	0	3	0	1	4	3	2	168	5	1	8	0	0	78	7120	10	
2	2	126	6	3	38	2	0	3	29	0	0	8	0	6	0	3	19	2	21	2	10	8	0	17	2	5	0	1	7	7	8	18	3	0	9	4	0	1	6	1999	

0.8486597264645906

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

1242	1	18	0	1	1	0	0	2	2	1	0	4	0	1	1	27	2	0	2	0	77	0	0	3	2	3	6	0	0	1	1	1	1	0	2	3	0	0	2	1	
1	1910	3	2	1	4	0	0	3	9	1	0	3	0	0	2	3	7	0	4	0	10	2	0	3	0	0	1	2	0	2	1	2	2	0	1	0	0	0	4	2	
17	4	2307	12	2	18	0	0	6	26	0	1	9	0	7	3	9	10	1	11	0	37	5	0	14	1	10	1	0	4	9	2	5	3	0	11	2	0	0	10	81	
0	0	2	2942	0	1	0	0	1	4	2	0	1	0	1	0	2	5	1	3	0	4	0	0	6	0	1	0	0	0	0	0	1	3	0	2	0	0	0	0	3	0
0	0	6	5	1231	12	0	0	3	15	0	0	6	0	1	0	2	3	0	0	1	7	1	1	4	0	5	0	0	6	350	0	2	0	8	10	0	1	1	4		
8	25	30	24	7	14892	0	0	10	79	4	3	29	0	6	15	27	35	414	47	3	98	22	1	31	6	19	7	1	5	8	6										

Kernel: (16, 2), RMSprop, categorical_crossentropy

1245	1	51	1	3	6	0	1	2	3	1	1	2	0	2	1	6	5	0	4	0	30	2	0	4	3	6	3	0	0	3	1	9	4	0	2	3	0	0	3	0	
1	1881	6	2	0	9	0	0	4	9	1	0	3	0	2	2	1	7	1	10	2	4	2	0	2	0	1	1	0	1	6	1	17	2	0	0	0	1	0	4	2	
10	2	2337	9	3	21	1	0	3	18	0	1	8	0	6	0	3	11	3	16	5	19	6	0	9	1	16	1	0	4	12	2	19	5	0	3	2	0	0	11	71	
0	0	2	2937	0	1	0	0	2	4	1	1	2	0	1	0	2	9	2	2	0	2	0	0	1	0	4	0	0	0	0	1	3	0	2	0	0	0	5	1		
0	0	6	4	1419	4	1	0	0	10	0	1	2	0	0	0	5	0	1	2	2	0	0	4	1	0	5	0	0	0	2	198	2	2	0	3	5	0	1	2	3	
5	17	43	17	11	13922	2	2	5	67	3	4	21	0	9	16	11	52	1348	87	5	53	12	1	19	5	38	3	1	10	17	10	112	30	1	11	9	1	4	10	25	
0	0	0	0	0	0	392	0	0	1	0	0	4	0	0	17	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	169	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	
0	0	2	0	1	0	0	0	1277	2	0	0	2	0	3	0	0	2	0	2	0	0	0	1	7	4	0	0	0	0	7	1	4	1	0	0	0	0	0	1	0	
3	2	16	6	3	17	1	0	7	6027	0	0	7	0	3	2	3	15	5	27	2	12	2	0	51	3	35	2	0	5	6	6	6	12	5	2	2	0	0	8	9	
2	0	0	0	1	3	0	0	0	6	1140	4	1	1094	0	0	0	9	3	0	0	127	0	0	0	0	0	1	0	0	0	10	1	0	0	0	0	0	0	6	0	
0	0	0	0	0	1	0	0	0	0	504	0	0	0	1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	3	0	
1	0	13	8	5	9	4	0	1	15	1	0	4228	0	6	2	0	6	6	12	2	3	0	3	4	1	7	0	0	1	3	4	0	3	0	3	2	0	0	6	0	
0	0	4	12	0	12	1	0	1	12	122	2	2	924	1	0	3	14	3	10	1	3	2651	1	5	1	0	1	1	7	0	1	23	4	1	1	0	0	0	25	4	
0	1	2	3	7	1	0	0	2	3	0	0	1	0	2202	0	1	3	0	2	1	1	0	0	0	4	3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	2
7	0	3	0	0	2	2	1	0	3	0	0	6	0	3	891	0	1	1	6	0	1	2	0	1	1	5	0	0	0	3	0	3	2	0	0	0	0	0	1	1	
0	2	8	3	0	5	0	0	0	10	0	0	1	0	3	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	
2	3	17	31	6	19	0	0	8	40	0	0	14	0	4	2	1	9937	5	27	6	8	1	1	3	3	12	2	1	3	4	2	10	2	0	2	1	0	4	15	8	
2	5	11	3	2	2821	1	0	2	16	1	0	6	0	4	4	1	15	3536	19	0	6	1	1	3	2	17	1	0	4	7	2	25	3	0	3	1	0	1	0	3	
9	13	56	34	4	95	1	0	12	88	4	4	13	1	9	13	11	51	24	8412	4	58	60	0	45	4	534	0	0	11	32	15	1730	23	2	5	11	0	7	95	39	
0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	257	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	
25	3	57	12	4	51	0	3	2	57	2	5	11	2	5	9	10	25	5	37	13	6175	15	1	19	1	29	5	0	8	28	4	59	8	2	8	2	0	0	25	52	
4	8	35	18	3	107	1	1	5	69	10	3	6	2	1	10	5	18	31	166	0	49	7250	0	92	5	14	4	0	17	39	1	170	17	5	2	3	0	2	62	28	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	129	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	
9	3	70	27	2	68	1	0	13	246	3	1	11	1	10	5	6	34	23	60	2	18	33	3	4925	6	69	4	1	12	44	10	78	40	0	6	9	0	1	67	40	
1	0	3	5	1	2	0	0	1	16	1	0	0	0	8	0	0	1	0	1	0	0	1	1	1	1117	9	0	0	1	0	3	0	0	0	3	0	0	0	0	0	3
0	0	1	4	3	4	0	0	3	5	0	1	2	0	0	5	1	6	2	36	3	1	0	0	3	0	0	0	1	1	0	113	2	1	1	0	0	2	6	0		
3	0	3	0	1	1	0	0	0	0	0	0	0	0	1	3	3	2	0	1	0	4	1	0	1	0	0	0	316	0	1	3	1	5	0	0	0	0	0	1	1	
2	2	3	2	1	7	0	1	1	30	2	0	1	0	2	0	0	1	10	0	1	10	0	1	4	0	75	0	2	0	1320	1	0	0	4	1	0	2	0	0	2	
2	3	24	5	2	21	3	1	2	52	2	4	5	0	5	0	2	27	10	60	3	28	11	0	23	2	41	1	0	3992	13	8	11	10	1	7	5	0	0	20	13	
3	0	13	6	5	9	0	0	2	22	0	1	4	0	4	0	0	9	2	24	0	3	8	23	10	2	10	0	1	0	594	6	6	2	0	4	3	0	0	5	4	
1	3	19	7	481	13	0	0	2	29	0	0	5	0	4	0	2	15	6	13	2	2	2	1	6	0	14	1	0	0	7	1121	5	4	0	7	4	0	1	4	6	
0	3	30	18	1	57	0	2	7	36	9	19	11	0	9	6	7	28	16	585	7	29	10	0	29	4	436	2	0	10	10	2	9674	14	0	4	3	0	6	219	14	
0	1	4	6	1	4	0	0	0	19	0	0	3	0	1	0	0	7	6	1	0	3	2	0	8	0	5	0	0	1	0	0	0	3398	2	0	0	0	1	5	1	
0	0	1	0	0	1	0	0	0	0	0	0	1	0	0	4	0	0	0	1	1	0	0	0	0	0	0	0	0	0	4	0	0	816	0	0	0	0	0	0	0	
3	1	35	16	8	23	0	0	1	32	0	0	6	0	10	0	3	22	3	53	0	4	9	2	9	5	19	1	3	5	2	6	21	5	0	1820	8	0	0	9	7	
1	0	9	5	3	8	0	1	2	13	0	0	3	0	1	0	0	8	2	12	0	1	6	1	6	1	12	1	0	2	2	6	5	1	0	3	1023	0	1	1	4	
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	17	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	1	0	0	0	3	0	0	1	3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	0	0	0	0	565	88	0		
0	3	11	8	1	18	0	1	6	19	1	1	6	0	1	4	2	13	3	80	4	9	2	0	5	0	17	0	0	3	5	0	278	8	0	2	0	0	46	7085	5	
1	1	123	10	3	24	2	0	3	29	0	0	6	0	8	2	3	14	3	21	2	10	5	0	12	3	18	1	0	8	14	2	22	5	0	2	4	0	1	7	2008	

0.8466940859034617

Kernel: (16, 2), Adadelata, cosine_proximity

1142	1	34	0	2	8	0	0	3	4	1	0	4	0	1	0	15	9	0	7	0	121	10	0	3	4	2	0	0	1	3	7	2	4	0	3	4	0	0	2	11	
1	1922	0	3	1	4	0	0	3	5	1	0	3	1	0	2	0	9	0	2	0	6	1	0	1	0	0	0	2	1	2	1	5	2	0	0	0	0	0	4	3	
11	6	2006	9	0	26	0	0	3	24	2	0	10	0	4	2	4	18	2	14	0	32	18	0	6	4	6	0	0	6	10	13	14	4	0	5	3	0	0	9	367	
1	0	1	2923	0	1	0	0	0	3	2	0	1	0	0	0	1	25	2	4	0	2	0	0	3	0	1	0	0	1	1	0	0	3	0	2	0	0	0	0	3	5
0	0	3	4	1158	5	0	0	0	8	0	0	4	1	0	0	7	0	1	0	5	2	0	0	0	2	0	0	1	3	458	1	3	0	3	7	0	0	5	4		
3	44	18	22	9	14660	0	0	11	63	6	0	30	4	5	11	13	71	568	68	0	68	43	0	14	4	15	0														

Kernel: (16, 2), Adam, categorical_hinge

1327	1	17	0	4	2	0	0	1	2	1	0	2	0	0	1	14	3	0	2	0	19	0	0	1	1	3	0	0	0	0	1	1	0	0	2	0	0	3	0		
1	1933	0	2	2	5	0	0	1	8	1	0	1	0	2	1	7	0	0	0	5	1	0	1	0	0	0	0	0	1	4	1	0	0	0	0	0	4	1			
89	9	2253	9	12	27	1	0	2	36	0	1	7	0	9	3	11	11	0	12	0	19	7	0	13	3	13	0	0	12	6	5	12	4	0	11	2	0	0	15	24	
3	2	2893	2	5	0	0	0	0	11	1	1	2	0	3	0	2	28	0	6	0	3	0	0	6	0	2	0	0	0	0	0	1	3	0	2	0	0	0	6	1	
1	0	0	1	1590	2	1	0	0	8	0	1	0	0	2	0	1	2	0	0	0	4	0	0	0	0	3	0	0	1	2	54	0	1	0	3	5	0	0	3	0	
32	39	26	14	27	15238	2	0	5	103	5	1	20	0	13	14	43	49	0	42	0	68	16	0	17	5	22	0	2	23	4	8	79	23	4	31	28	0	0	12	4	
0	0	1	0	0	408	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	166	0	
1	2	1	0	3	3	0	0	1264	8	0	0	3	0	4	0	0	2	0	0	1	0	0	1	11	3	0	0	0	3	2	0	2	0	0	0	0	0	0	3	0	
7	7	9	0	10	18	1	0	4	6081	0	0	3	0	5	2	4	16	0	16	0	19	5	0	40	2	24	0	3	5	1	2	2	6	5	2	2	0	0	9	2	
2	1	0	0	1	6	0	0	0	1	1201	3	0	1150	0	0	1	0	0	0	0	35	0	0	0	0	0	1	0	0	0	3	0	0	0	0	0	0	0	3	0	
0	0	0	0	0	1	0	0	0	0	1	490	0	0	1	0	0	0	0	0	4	2	0	0	1	0	0	0	1	0	0	5	0	0	0	0	0	0	0	7	0	
17	4	23	8	24	22	4	0	0	56	1	0	4075	0	8	20	1	10	0	16	0	12	2	0	7	1	6	0	3	6	2	4	0	3	0	11	2	0	0	9	2	
3	2	3	6	3	16	2	0	1	11	76	4	2	1903	3	1	6	15	0	4	0	3	1724	0	4	0	0	0	10	1	0	12	3	1	2	0	0	0	0	31	1	
0	0	1	0	10	1	0	0	2	1	0	0	2	0	2209	0	3	3	0	0	0	2	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
1	1	4	0	0	6	4	0	0	5	1	1	1	0	3	883	0	1	0	6	0	2	2	0	1	0	9	0	0	0	1	0	3	0	0	0	1	0	0	2	1	
19	2	2	3	1	4	0	0	0	8	0	0	1	0	0	0	1366	6	0	2	0	7	0	0	1	0	0	0	3	1	0	4	2	0	4	0	0	0	5	0		
10	6	8	18	15	23	1	0	4	47	0	1	9	1	9	5	3	9954	0	20	0	11	0	0	7	1	7	0	2	9	0	2	0	1	1	8	3	0	0	14	4	
9	9	2	2	9	6329	1	0	2	30	2	0	8	1	7	4	5	15	0	10	0	8	5	0	6	1	13	0	0	9	0	2	25	2	1	8	3	0	0	1	0	
57	46	45	22	30	193	5	0	15	114	6	3	15	5	18	28	30	51	0	8480	0	116	95	0	75	5	300	0	3	44	17	43	1328	20	3	18	16	0	0	266	17	
0	2	12	0	2	2	2	0	1	2	0	0	1	0	0	0	3	0	0	0	203	0	0	0	1	0	0	1	0	0	2	0	0	0	0	0	0	0	3	46		
146	13	35	8	9	56	3	0	1	69	3	5	7	1	4	13	24	29	0	21	0	6127	12	0	30	1	21	0	1	22	2	2	42	6	1	8	4	0	0	35	18	
25	38	35	10	4	170	3	0	8	131	8	5	3	10	4	15	21	14	0	87	0	68	7104	0	154	2	10	0	6	56	11	18	105	9	8	11	15	0	0	84	11	
0	0	1	0	2	0	0	0	0	2	0	0	0	1	0	0	2	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
34	21	39	16	4	104	3	0	10	407	3	0	10	1	11	4	21	32	0	45	0	31	30	0	4803	12	38	0	16	26	14	41	45	27	1	17	21	0	0	66	8	
4	1	1	3	2	4	0	0	1	22	1	0	1	0	21	0	2	4	0	3	0	0	1	0	1	1090	5	0	0	3	0	2	0	0	0	5	2	0	0	0	0	0
0	2	0	3	6	11	0	0	2	14	0	2	2	0	1	6	2	10	0	67	0	4	0	0	3	0	0	2	2	2	201	2	1	1	3	0	0	0	29	0		
96	0	1	0	3	2	0	0	0	2	0	0	1	0	1	2	173	4	0	5	0	24	0	0	12	0	2	0	0	3	2	0	14	0	0	3	1	0	0	1	0	
3	3	3	0	5	3	0	0	0	12	2	0	0	1	2	0	1	0	0	4	0	0	1	0	5	0	0	0	1427	3	0	0	1	1	0	1	0	0	1	0		
9	5	9	6	7	28	2	0	0	50	2	4	3	0	6	1	2	25	0	19	0	12	2	0	10	0	27	0	0	4130	1	4	7	6	1	7	9	0	0	22	3	
21	4	7	5	16	23	0	0	3	48	0	1	4	0	5	0	14	8	0	22	0	17	21	0	22	3	6	0	2	13	473	14	5	2	0	6	10	0	0	8	2	
11	5	0	4	984	14	0	0	1	28	0	0	2	0	5	0	5	15	0	4	0	4	0	0	3	0	9	0	0	1	1	662	2	4	0	12	4	0	0	6	1	
25	31	26	13	13	119	0	0	12	73	9	21	12	0	17	12	40	46	0	729	0	55	21	0	44	7	289	0	1	19	6	15	9006	12	0	14	7	0	0	620	3	
4	1	3	3	3	9	0	0	0	32	1	1	4	0	2	1	2	10	0	1	0	5	0	0	22	0	5	0	0	2	0	0	1	3351	2	2	0	0	0	12	0	
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	822	0	0	0	0	0	
18	2	6	4	11	10	0	0	0	28	0	0	3	0	9	0	6	16	0	15	0	6	2	0	2	4	9	0	2	11	1	10	5	4	0	1945	17	0	0	4	1	
5	3	4	1	8	8	0	0	4	10	0	0	2	0	4	0	3	2	0	2	0	4	1	0	2	0	10	0	0	3	2	5	2	0	0	3	1050	0	0	6	0	
0	1	0	0	1	1	0	0	0	3	0	0	1	0	0	0	1	0	0	0	0	60	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0		
0	1	0	2	0	4	0	0	1	3	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	0	0	0	2	1	0	1	0	1	0	0	0	650	0		
4	8	5	4	1	25	1	0	4	29	2	1	5	0	1	4	5	18	0	55	0	11	2	0	3	0	10	0	5	0	3	127	5	1	5	3	0	0	7299	1		
37	10	330	7	11	50	2	0	6	55	1	0	6	2	11	2	6	20	0	40	0	52	18	0	57	6	15	0	1	35	8	20	20	4	1	15	14	0	0	24	1491	

0.8173214119190287

Kernel: (16, 2), Adamax, cosine_proximity

1189	1	14	0	5	10	0	0	1	1	1	0	4	0	2	1	27	5	0	7	0	116	4	0	2	2	1	0	0	1	2	0	1	1	0	5	3	0	0	2	0
1	1934	0	2	1	4	0	0	1	2	2	0	4	0	0	2	1	6	0	5	0	6	1	0	2	0	0	0	1	0	1	1	2	2	0	0	0	0	3	1	
21	10	2197	8	9	40	0	0	2	17	1	1	12	0	8	4	9	14	3	36	5	48	15	0	15	5	5	1	0	6	13	6	10	3	0	17	2	0	0	10	85
2	2	2	2907	0	5	0	0	0	2	2	0	5	0	3	0	1	24	2	5	0	4	0	0	6	0	1	0	0	1	2	0	0	3	0	2	0	0	0	3	1
0	0	1	4	1508	8	0	0	0	4	0	1	5	0	1	0	1	5	0	1	1	0	1	1	0	1	0	0	0	1	128	0	1	0	4	3	0	0	3	1	
2	32	9	12	17	14961	0	0	3	28	5	2	34	0	5	15	18	50	552	38	3	68	11	1	12	8	10	0	3	8	5	2</									

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

1147	0	21	0	3	8	0	1	2	1	1	1	2	0	2	1	39	3	0	3	0	139	2	0	4	3	3	1	0	0	1	2	1	1	0	4	8	0	0	2	2	
1	1925	0	4	0	4	0	0	3	3	1	0	2	0	1	1	1	6	0	2	2	9	3	0	3	0	0	0	1	1	4	0	2	1	0	0	1	0	0	4	0	
14	6	2142	10	3	48	1	0	3	22	1	1	7	0	6	2	7	11	1	11	5	50	17	0	47	2	7	1	0	7	13	5	6	2	0	10	6	0	0	9	155	
0	0	1	2936	1	3	0	0	1	4	2	0	0	0	1	0	1	12	1	4	0	3	0	0	5	0	1	0	0	1	0	0	3	0	2	2	0	0	0	0		
0	0	2	3	1403	3	1	1	0	3	0	1	0	0	0	0	1	1	0	1	2	1	0	1	0	0	2	0	0	0	3	231	0	1	0	4	15	0	1	1	3	
4	37	14	14	14	15231	2	4	5	35	7	6	16	2	9	7	22	38	168	31	15	77	19	1	21	6	15	4	2	5	18	9	34	16	2	16	70	0	3	4	16	
0	0	0	0	0	0	328	0	0	1	0	0	3	1	0	81	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	2	0	159	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	8	0		
0	2	1	0	0	2	0	0	1285	0	0	0	1	0	3	0	0	1	0	0	0	0	2	0	6	1	0	0	0	7	0	0	1	0	0	4	0	0	0	1	0	
2	6	15	7	4	52	1	0	15	5856	3	2	2	2	3	1	5	11	1	24	9	41	24	0	110	3	18	1	5	3	12	10	1	6	5	5	26	0	0	10	11	
2	0	0	0	0	0	0	0	0	0	1714	3	0	678	0	0	0	2	0	0	0	7	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	2	507	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	6	14	11	7	54	4	0	3	40	1	0	4062	0	2	7	1	9	2	21	3	23	5	1	20	2	2	0	0	3	3	12	0	2	0	8	9	0	0	8	12	
0	0	1	4	0	9	1	0	0	2	252	3	1	2841	1	0	3	6	1	1	1	3	704	1	0	0	0	0	5	0	1	1	0	2	0	0	0	6	2	0	0	
0	1	2	3	8	2	0	0	9	2	0	0	3	0	2176	0	2	2	0	1	2	1	0	0	7	0	0	0	0	0	1	0	0	1	2	14	0	0	0	2	0	
0	0	2	0	0	10	3	1	0	4	1	1	0	0	3	877	0	1	1	8	0	6	3	0	4	0	1	0	0	6	0	1	0	0	0	1	0	0	1	4	0	
1	1	3	3	0	5	0	0	0	6	0	0	1	0	1	0	1382	6	1	1	0	15	4	0	1	0	0	0	1	1	0	0	2	0	3	2	0	0	1	0	0	
1	4	13	34	10	34	0	0	15	39	2	1	10	0	2	2	1	9840	3	31	10	16	13	0	22	3	6	2	3	7	11	8	2	2	0	6	22	0	4	11	14	
1	5	3	3	3	5517	0	0	2	11	2	0	6	3	5	4	2	12	856	13	0	12	9	1	8	0	10	1	0	4	5	3	8	0	0	3	17	0	0	0	0	
11	25	44	35	9	281	1	0	16	86	10	11	10	9	12	19	20	42	9	8909	10	275	284	0	169	14	93	0	2	18	108	32	705	15	3	18	38	0	4	93	89	
0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	265	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
7	3	16	7	1	33	0	2	4	19	4	5	4	1	3	5	13	15	0	12	30	6469	11	0	9	1	11	0	0	3	7	2	12	7	1	8	15	0	0	10	29	
2	7	10	11	3	134	1	0	4	25	13	10	3	94	1	7	6	8	7	26	2	106	7552	0	92	6	7	1	1	13	13	3	23	7	5	7	16	0	2	19	16	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	45	0	0	0	0	0	0	0	0	0		
8	12	36	22	2	128	1	0	14	131	5	2	7	2	9	5	7	22	6	34	4	50	58	2	5097	11	16	5	4	15	43	15	18	18	0	16	54	0	0	32	50	
1	0	3	4	1	2	0	0	2	8	1	0	1	0	4	0	0	2	0	0	0	2	1	2	1122	1	0	0	0	3	0	0	0	0	5	11	0	0	0	3	0	
0	1	1	4	4	16	0	0	4	11	0	3	2	0	1	5	3	8	1	205	4	7	1	0	6	0	1	5	1	256	2	1	2	19	0	4	16	4	0	0		
4	0	3	0	1	3	0	0	0	0	0	0	0	0	1	1	14	2	0	1	0	10	1	0	4	0	0	0	298	0	1	2	1	2	0	0	1	1	0	0	1	
2	2	2	1	0	2	0	1	1	3	2	0	4	2	0	1	0	1	0	1	0	1	1	0	5	0	0	0	0	1443	1	0	0	1	0	0	1	0	0	0	0	
3	6	9	6	2	42	1	1	7	35	2	4	4	1	3	1	3	21	3	32	4	51	28	0	66	4	17	2	1	3958	20	10	4	5	1	6	25	0	0	14	17	
2	1	6	5	6	10	0	0	2	19	0	0	2	1	1	0	0	6	1	13	0	7	31	2	15	1	3	0	2	0	607	8	3	0	0	4	16	0	0	1	10	
1	4	4	4	4	401	14	0	0	3	16	0	0	2	0	4	0	2	11	2	3	3	2	4	1	5	1	7	1	0	0	3	1236	1	1	0	8	31	0	1	3	8
3	15	32	23	4	203	0	2	13	45	11	26	8	2	14	9	22	34	12	1155	11	120	93	0	101	8	103	3	4	12	40	11	8728	14	0	17	25	0	7	348	39	
0	2	3	5	2	13	0	0	5	18	1	1	3	0	1	0	0	6	3	1	2	4	6	0	36	0	5	1	0	1	5	0	0	3339	2	1	3	0	1	5	4	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	1	0	824	0	0	0	0	0	0	0	
2	2	10	4	7	15	0	0	1	15	0	0	2	0	4	0	2	11	1	12	0	4	17	1	6	5	6	0	3	7	3	7	4	4	0	1918	62	0	0	1	15	
0	0	0	1	2	2	0	1	0	0	0	0	1	0	0	0	0	0	0	0	2	1	1	0	0	2	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	
0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	8	58	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	1	0	0	0	4	0	0	2	0	4	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	1	0	1	0	0	552	98	0	0	
0	7	7	13	1	52	0	1	8	21	10	4	5	0	1	4	8	15	1	80	7	34	24	0	14	2	5	1	1	8	13	4	129	9	2	9	7	0	29	7100	11	
1	2	40	7	4	42	1	0	4	24	1	0	3	1	5	2	3	15	0	9	5	27	10	0	28	4	7	1	0	6	11	4	10	3	0	3	11	0	1	5	2077	

0.8511500020239835

Kernel: (16, 2), Nadam, categorical_hinge

1291	1	37	0	3	4	0	1	4	1	1	0	3	0	1	0	21	3	0	3	0	7	0	0	1	2	5	0	0	1	0	4	0	0	0	0	11	3	0	0	0	0
1	1944	0	2	1	3	0	0	2	3	1	0	4	0	0	0	0	4	0	0	0	4	2	0	1	0	0	0	1	0	0	1	4	1	0	2	0	0	0	3	1	0
36	7	2373	8	5	19	0	0	3	17	1	0	11	0	6	0	9	11	0	7	0	17	10	0	4	1	11	0	0	5	0	5	9	1	0	18	3	0	0	0	10	31
1	2	2	2920	1	4	0	0	0	4	2	0	4	0	1	0	2	16	0	5	0	2	1	0	3	0	2	0	0	1	0	0	1	3	0	2	1	0	0	4	1	0
0	0	3	3	1274	6	0	0	4	0	0	5	0	0	0	0	2	0	0	0	2	2	0	0	0	8	0	0	1	0	360	0	1	0	5	4	0	0	3	2	0	
6	46	39	16	15	15235	0	1	9	66	5	0	28	4	6	7	24	50	0	57	0	48	54																			

Kernel: (16, 2), Nadam, logcosh

1235	1	24	0	6	14	2	1	1	2	1	1	2	0	2	0	19	7	0	3	0	58	3	1	4	3	3	0	0	1	1	0	0	3	0	3	3	0	0	2	2	
1	1908	0	3	1	10	0	0	0	9	1	0	2	0	0	2	1	10	0	4	0	7	6	0	3	2	0	0	2	0	3	1	2	2	0	1	0	0	0	0	3	1
14	4	2179	10	4	45	0	0	2	21	0	1	6	0	9	1	4	21	2	12	1	19	13	0	29	15	9	0	0	7	8	3	9	2	0	17	2	0	0	16	153	
1	0	1	2930	0	0	0	0	0	3	1	1	1	0	1	0	1	20	0	2	0	2	0	0	5	2	1	0	0	1	1	0	0	3	0	2	1	0	0	3	2	
0	0	2	4	1511	6	0	0	0	5	0	1	0	0	1	0	0	3	0	1	0	3	2	0	1	2	1	0	0	0	3	126	0	1	0	3	5	0	0	3	1	
2	21	7	21	14	15256	0	0	4	44	3	1	13	0	7	3	16	59	271	24	0	61	11	0	14	15	14	0	2	5	7	6	40	24	2	19	13	0	0	11	9	
0	0	0	0	0	1	0	0	0	2	0	0	135	0	0	250	0	2	0	5	0	2	1	0	0	0	0	0	0	0	1	0	0	18	0	0	0	0	0	0	0	
1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	161	0	
0	1	2	0	2	10	0	0	1246	4	0	0	3	0	3	0	0	2	0	0	0	1	3	0	4	17	3	0	0	1	2	2	3	2	0	0	2	0	0	2	2	2
1	3	12	7	6	43	0	0	6	5922	0	0	4	0	3	3	5	17	4	15	0	24	15	0	115	8	22	0	8	6	6	7	2	14	5	7	4	0	0	9	9	
2	0	0	1	1	3	0	0	0	1	1018	3	0	1326	0	0	0	0	6	0	0	0	42	0	0	0	0	1	0	0	2	0	0	0	0	0	0	0	0	2	0	
0	0	0	0	0	1	0	0	0	0	2	487	0	0	0	0	0	0	0	0	0	5	5	0	0	1	0	0	1	0	0	4	0	0	0	0	0	0	0	7	0	
2	1	16	11	12	40	0	0	0	30	1	0	4140	0	3	3	1	14	3	7	0	8	1	0	11	9	3	0	0	3	1	3	0	5	0	16	3	0	0	7	5	
0	0	2	8	1	17	0	0	0	5	52	1	2	1722	1	1	3	13	1	2	0	3	1983	0	1	2	0	0	0	4	1	0	5	3	0	2	0	0	0	16	2	
0	1	1	3	6	1	0	0	1	1	0	0	2	0	2184	0	1	5	0	0	0	2	1	0	0	24	0	0	0	0	0	0	0	0	1	1	2	0	0	1	3	
0	0	1	3	0	10	0	0	0	4	0	1	2	0	3	851	0	7	1	23	0	5	2	0	7	2	5	0	0	0	2	0	4	2	0	0	0	0	2	2		
2	2	3	4	0	6	0	1	0	8	0	0	1	0	2	0	1375	8	1	1	0	12	1	0	2	0	0	0	1	1	0	0	2	0	0	0	0	0	2	2		
1	1	5	22	6	15	1	0	3	26	0	0	9	0	2	2	1	10010	3	13	0	7	6	0	2	10	7	0	3	7	6	5	0	2	0	9	3	0	0	10	7	
0	7	0	3	3	5131	1	0	1	12	1	0	3	0	5	4	2	18	1281	6	0	7	3	0	4	3	10	0	0	5	1	1	7	2	0	4	1	0	0	0	3	
14	20	34	30	14	248	0	0	10	101	5	3	15	4	15	14	16	84	16	9032	17	122	212	1	167	49	140	0	2	20	61	17	662	22	2	36	13	0	0	235	76	
0	2	14	0	0	4	0	0	1	2	0	0	2	0	0	0	3	0	1	0	177	1	0	1	3	0	0	0	0	0	1	1	0	0	0	0	0	0	2	68		
23	6	32	13	3	65	0	0	2	37	3	1	8	1	4	4	10	34	0	19	0	6294	15	1	37	7	13	0	2	4	7	1	18	9	2	17	4	0	0	26	57	
2	13	19	16	3	186	0	0	4	49	7	2	4	6	1	8	8	43	11	58	0	52	7373	1	180	19	7	1	4	15	12	1	49	13	3	22	5	0	0	48	18	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	6	24	24	3	129	0	0	5	134	3	1	7	1	9	3	6	36	12	28	1	17	41	0	5169	30	16	0	12	13	28	5	19	37	0	20	12	0	0	57	44	
0	0	0	3	1	1	0	0	0	3	1	0	0	0	2	0	0	2	0	0	0	1	0	1	0	1	1160	0	0	1	0	0	0	0	0	3	0	0	0	0	0	
0	1	0	4	6	15	0	0	2	11	0	2	3	0	1	2	4	10	3	125	0	8	1	0	6	9	2436	0	1	2	4	1	188	3	1	2	8	0	0	0	31	1
16	0	3	0	1	5	37	121	0	2	0	0	0	0	2	1	92	3	0	2	1	20	2	15	14	1	0	0	0	1	2	2	5	0	0	2	1	0	0	0	1	
2	0	2	1	1	3	0	0	0	4	2	0	0	1	2	0	1	1	1	0	0	2	0	5	0	0	0	0	1446	0	0	1	1	0	1	0	1	0	0	1	0	
4	4	12	7	3	36	1	0	0	31	2	4	5	0	3	1	2	43	4	36	0	31	21	0	68	13	17	0	2	3986	5	8	5	9	1	7	4	0	0	23	21	
2	0	3	4	9	23	0	0	2	22	0	0	3	0	2	0	0	7	2	7	1	5	33	0	26	14	3	1	3	2	582	6	1	2	0	6	5	0	0	4	5	
1	3	4	4	662	23	0	0	1	21	0	0	2	0	5	0	2	16	3	4	0	4	10	0	9	5	5	0	0	1	3	967	2	5	0	12	6	0	0	4	3	
3	14	30	17	6	162	0	0	3	52	7	22	12	1	12	7	17	56	19	1130	5	73	56	1	93	45	137	0	4	10	16	3	8489	17	0	21	5	0	0	750	22	
0	1	2	3	1	7	0	0	0	5	0	1	3	0	1	0	0	9	4	1	0	3	2	0	8	2	5	0	0	1	0	0	0	3410	1	3	0	0	0	5	1	
0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	1	0	0	1	0	0	0	2	0	0	0	1	1	819	0	0	0	0	0	0	
2	1	5	4	7	19	0	0	0	7	0	0	1	0	5	0	2	12	1	6	0	5	16	0	8	18	6	0	4	7	3	4	3	4	0	1988	8	0	0	1	4	
0	0	1	1	3	8	0	0	0	7	0	0	1	0	0	0	2	0	0	1	3	4	0	7	4	3	0	0	2	2	1	1	1	0	4	1084	0	0	3	1		
0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	2	0	0	0	63	0	0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
0	1	0	2	0	6	0	0	1	3	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0	0	0	0	0	2	1	0	1	0	0	0	0	648	0		
0	4	8	11	1	37	0	0	4	23	2	0	5	0	1	1	5	24	2	57	5	14	7	0	9	6	8	0	1	5	1	0	90	7	2	10	2	0	0	7287	8	
1	2	44	8	4	38	0	0	2	24	0	0	4	1	7	2	3	19	0	12	0	15	13	0	30	17	7	0	1	8	8	2	11	5	0	10	5	0	0	10	2064	

0.8313117393359536

Arat 2016_07_27_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

4493	2	0	0	5	4	3	0	0	13	6	10	77	1	0	5	6	3	4	2	0	9	10	0	8	0	2	11	2626	0	32	3	6	0	110	0	0	
0	4155	1	0	3	1	1	0	1	16	6	2	8	0	0	5	1	3	2	1	1	7	2	0	3	0	3	20	7	0	1	2	31	0	1	0	0	
1	0	177	0	2	0	1	0	0	23	0	4	2	4	0	0	81	28	1	21	0	0	2	2	0	0	0	0	1	0	2	5	1	5	2	0	0	0
0	6	0	0	0	0	343	0	0	34	3	0	6	0	0	0	1	2	3	626	0	1	0	0	1	0	0	5	4	0	3	0	1	0	0	0	0	0
3	11	3	0	3223	3	18	0	1	140	6	10	42	16	0	13	5	46	8	14	2	4	21	0	16	1	2	13	23	2	19	1130	14	0	4	0	0	
11	12	0	0	8	7152	37	0	0	91	21	2	34	1	2	3	0	4	7	0	3	8	3	10	5	0	9	10	17	0	20	2	17	127				

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

5348	0	0	0	1	1	2	0	0	0	3	12	31	0	0	3	0	0	0	2	6	6	0	4	0	0	9	1998	0	4	1	5	0	14	0	1		
0	4106	0	0	2	1	2	1	1	6	4	3	6	0	0	8	1	7	1	1	2	12	3	0	5	0	5	19	9	1	1	2	75	0	0	0	0	
1	0	201	0	5	1	4	0	0	6	0	12	1	5	0	1	58	39	0	7	0	0	2	2	0	0	2	2	2	4	3	4	2	0	1			
0	3	0	964	0	0	7	0	0	2	1	0	0	0	0	0	1	1	0	52	1	0	0	1	0	0	2	1	0	0	2	0	0	0	0	0		
8	12	0	1	2779	5	22	2	0	51	6	29	43	8	1	22	4	66	9	11	9	6	26	3	22	0	4	17	59	10	12	1543	21	0	1	0	1	
13	9	0	0	5	7170	172	0	0	21	10	4	25	1	2	4	0	2	7	1	8	9	3	14	7	0	15	10	24	3	15	3	20	39	1	0	2	
36	31	1	20	37	58	8107	2	3	183	24	43	116	23	4	37	5	227	18	404	40	11	67	10	40	0	14	24	111	38	39	24	46	0	2	0	6	
2	1	0	0	1	0	11	921	0	5	4	27	24	1	1	1	28	4	8	1	3	4	0	2	2	1	3	16	0	5	5	3	0	0	0	0		
1	3	0	0	1	1	1	0	957	10	4	6	10	0	0	2	1	1	0	0	2	0	7	2	6	0	0	2	3	0	0	3	3	0	0	0	0	
37	39	0	3	39	12	61	2	7	8625	14	114	145	53	3	147	17	133	26	32	27	11	62	6	36	2	18	30	126	31	165	43	73	0	5	0	9	
2	17	0	1	0	0	3	0	0	6	7790	4	9	1	1	1	0	1	5	0	4	1	3	0	4	0	1	3	6	0	1	1	10	0	0	0	0	
5	20	0	3	10	0	18	3	2	20	5	10494	23	3	0	2	2	25	7	10	22	2	34	0	17	0	1	15	39	0	3	45	28	0	1	0	4	
59	54	1	5	45	7	67	2	6	87	33	61	14645	15	2	22	10	49	25	22	36	13	61	2	68	1	18	47	134	20	29	30	70	0	2	0	5	
2	2	0	0	5	19	91	1	1	80	0	14	14	1044	1	12	2	138	14	8	18	3	3	10	5	0	5	1	3	116	7	8	21	15	6	0	92	
5	0	0	0	0	0	0	0	0	1	1	0	2	0	0	1000	2	0	0	0	0	0	0	0	0	0	1	864	1	0	1	1	0	0	0	0	0	
3	0	0	0	3	0	1	0	0	12	0	2	1	0	0	2432	0	6	0	0	8	0	4	0	2	0	1	1	3	6	5	1	4	0	0	0	1	
3	6	3	0	3	1	1	0	0	39	0	14	6	3	0	8	1161	12	3	5	1	0	8	1	5	0	1	1	15	2	4	6	5	0	0	0	0	
18	29	4	1	55	46	103	4	2	273	5	149	185	35	1	41	18	3733	21	91	44	19	78	11	43	0	21	17	156	72	48	47	38	0	3	0	4	
10	2	0	0	1	3	6	11	0	0	19	9	11	43	2	17	18	4	18	7738	0	5	1	23	3	19	0	9	4	18	6	9	2	30	0	0	0	4
5	12	0	20	8	4	70	0	0	25	3	15	20	3	0	3	4	74	10	1312	10	2	12	0	14	0	2	10	17	8	11	8	13	0	0	0	2	
5	4	0	0	6	5	7	0	0	8	3	8	10	6	0	10	1	29	6	8	3622	1	8	3	8	0	1	8	27	61	12	12	18	0	2	0	6	
32	8	0	0	0	0	10	0	1	0	2	9	29	5	0	2	1	8	2	0	5	156	8	3	3	0	3	6	1332	3	4	4	4	14	0	1	0	0
0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	9	0	0	0	0	
2	24	0	0	6	12	14	0	1	76	3	11	20	1	4	5	1	12	11	0	8	2	2	6360	13	0	17	3	3	5	7	9	90	4	0	0	1	
7	5	0	0	2	0	0	0	0	2	3	7	6	0	0	5	5	0	0	2	0	5	0	13	0	8644	0	0	26	13	0	3	4	5	0	0	0	1
0	1	0	0	0	0	0	5	0	0	2	0	0	0	0	0	0	0	1	0	0	0	1	1	664	0	1	2	0	1	0	0	0	0	0	0	0	
16	28	0	0	4	60	51	3	3	32	12	20	22	3	1	11	3	33	14	8	13	6	17	11	15	0	4946	11	17	17	13	22	15	10	4	0	0	
5	9	0	1	3	0	0	0	2	4	0	8	6	0	19	15	2	1	1	0	6	1	12	0	3	0	3	7643	28	1	4	3	15	0	0	0	1	
799	31	1	2	14	2	22	0	0	6	14	31	115	7	2	12	4	13	13	2	21	118	26	1	17	0	3	27	10714	9	6	13	48	0	7	0	3	
2	4	0	0	9	4	11	1	0	9	3	15	10	12	0	7	2	35	4	4	71	2	4	3	6	0	0	5	10	906	6	2	11	0	0	0	4	
11	16	1	0	9	5	33	4	1	553	13	17	59	10	1	58	4	33	16	15	12	1	21	7	16	0	8	9	31	18	4839	11	25	0	2	0	2	
10	10	0	1	1592	11	12	1	3	57	6	33	39	17	0	20	5	45	9	9	25	2	22	2	29	1	6	11	54	12	17	4185	26	3	0	0	4	
16	152	0	7	11	7	50	0	2	53	28	36	46	7	9	18	3	48	17	8	22	9	26	76	28	1	8	35	47	21	17	20	11077	0	0	0	4	
9	3	0	4	2	2936	19	2	0	2	4	5	1	0	4	2	0	1	4	0	4	0	0	6	5	1	7	2	2	0	5	5	4	1749	2	0	1	
2339	0	0	0	0	3	5	0	0	0	2	8	4	0	2	9	0	3	1	1	2	0	2	3	2	0	3	1	8	3	9	0	5	10	2248	0	0	0
0	3	0	0	0	0	1	0	0	2	43	0	2	0	0	0	0	2	0	0	0	0	3	0	0	0	2	128	0	1	0	0	0	0	0	0	0	
1	3	0	0	0	0	0	0	0	12	0	0	3	16	0	2	0	3	1	1	1	0	0	0	4	1	0	1	3	3	0	1	1	0	0	0	704	

0.8503144917708609

Kernel: (4, 2), Adamax, cosine_proximity

4282	1	0	0	1	1	1	0	1	0	4	15	47	1	0	2	0	0	5	0	2	11	5	0	5	0	1	9	3036	0	3	1	5	0	12	0	0	
0	4052	0	1	2	1	5	0	1	13	6	10	18	0	0	4	1	6	4	0	4	13	2	0	5	0	10	22	8	2	1	3	86	0	0	2	2	
1	0	256	0	1	0	1	0	0	6	0	11	1	4	0	0	48	25	0	2	0	0	0	1	2	0	0	0	1	0	0	2	1	2	0	0	0	0
0	3	0	949	0	0	29	0	0	5	1	0	1	0	0	1	1	2	34	1	1	0	1	0	1	0	1	4	2	0	0	1	1	0	0	0	0	
7	13	0	1	1895	4	32	2	6	130	6	49	90	15	0	13	4	77	10	7	10	7	23	1	24	0	3	11	49	13	15	2275	19	0	0	0	2	
12	11	0	0	6	7179	59	4	0	60	24	9	42	14	3	3	0	4	9	0	12	12	2	13	7	0	17	10	22	8	15	5	24	30	1	0	2	
28	27	1	31	26	44	8031	3	7	412	19	54	197	75	3	11	7	158	21	262	29	16	54	8	48	0	13	24	90	34	38	29	46	0	2	0	3	
1	1	0	0	1	0	5	895	0	18	4	38	38	0	2	0	0	28	3	7	2	3	4	0	3	0	1	2	17	0	4	5	3	0	0	0	0	
0	3	0	0	1	0	1	0	969	8	2	4	13	0	0	0	1	0	0	0	2	0	5	1	6	0	0	0	2	0	0	1	3	0	0	0	0	
19	33	1	3	13	2	36	2	7	9097	14	119	152	50	2	50	13	47	19	15	19	19	38	0	36	1	4	28	78	12	114	40	60	0	2	0	8	
2	9	0	1	1	0	3	0	1	13	7773	5	10	4	2	0	0	1	9	0	6	1	3	0	1	0	0	2	4	0	3	1	7	0	0	11	2	
2	10	0	4	1	0	4	1	3	25	3	10647	22	2	0	0	2	4	4	6	9	0	19	0	9	0	0	7	13	1	3	38	19	0	1	0	4	
40	36	0	6	14	2	32	2	10	123	28	68	14876	19	3	11	9	34	21	10	30	9	42	1	66	1	9	33	108	5	25	28	45	0	2	3	2	
2	2	1	0	3	17	26	1	1	102	0	17	18	1303	1																							

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

4219	0	0	0	1	0	1	0	0	0	4	7	55	1	0	2	0	0	0	0	3	11	1	0	0	0	1	8	3110	0	3	1	3	0	20	0	0	
0	4136	0	0	1	1	0	0	1	3	6	4	17	0	0	0	1	7	1	2	5	19	1	0	3	0	5	24	7	1	2	1	36	0	0	0	0	
1	0	237	0	1	0	1	0	0	3	0	9	4	2	0	0	54	36	0	8	0	0	0	1	0	0	0	1	1	1	1	1	3	0	0	0		
0	3	0	969	0	0	4	0	0	0	1	0	0	0	0	0	1	1	0	56	1	0	0	0	0	0	2	0	0	0	1	0	0	0	0	0		
4	13	1	1	2684	7	17	2	1	14	7	24	94	5	0	3	3	111	4	11	9	7	5	0	10	0	2	15	60	7	11	1664	16	0	1	0	0	
12	9	0	2	8	7326	18	0	0	5	25	4	45	2	0	2	0	2	5	1	9	11	0	4	2	0	14	12	27	5	29	3	15	18	2	0	2	
27	34	1	75	45	109	7230	2	3	91	28	33	311	20	0	6	12	315	13	1006	46	16	11	2	21	0	21	36	120	65	73	30	42	0	4	0	3	
1	0	0	0	3	2	4	859	0	4	6	25	57	1	1	0	0	58	3	10	4	3	2	0	2	0	1	3	21	1	6	5	3	0	0	0		
0	3	0	0	2	0	2	0	883	8	4	5	81	0	0	0	2	5	1	3	3	1	2	0	3	0	0	4	5	0	2	3	4	0	0	0	0	
24	46	0	6	51	36	71	3	7	7723	25	117	392	40	0	32	43	390	20	103	38	21	15	0	16	1	20	37	139	49	534	70	77	0	5	0	2	
2	9	0	1	0	0	1	0	0	3	7819	3	11	0	0	0	0	1	2	0	3	1	1	0	0	0	0	1	6	0	2	1	8	0	0	0	0	
2	20	0	2	10	1	6	2	1	13	4	10476	75	1	0	0	2	35	3	26	23	6	5	0	3	0	3	20	55	0	4	43	18	0	1	0	3	
37	36	1	5	15	3	29	0	4	19	28	38	15182	5	0	8	5	14	7	16	24	12	8	0	6	0	10	33	111	7	29	29	30	0	2	0	0	
2	3	1	0	6	23	55	2	1	38	1	13	42	972	1	2	3	291	9	16	26	6	0	6	2	0	4	1	1	159	8	10	16	14	8	0	19	
3	0	0	0	0	0	0	0	0	1	2	0	9	0	355	1	0	0	0	0	0	0	1	0	0	0	3	1502	1	0	5	1	0	0	0	0	0	
1	4	0	0	12	2	4	0	0	28	0	2	15	0	0	2249	1	29	1	7	35	0	3	0	2	0	3	6	8	39	31	5	7	0	1	0	1	
1	6	3	0	1	1	1	0	0	17	0	14	24	0	0	2	1189	20	0	11	1	2	0	0	2	0	1	2	7	0	3	8	1	0	0	0	0	
10	28	3	1	52	56	62	2	1	106	8	119	398	18	1	14	30	3891	12	135	46	19	7	3	20	0	20	24	135	63	59	46	24	0	1	0	1	
12	3	0	1	4	5	11	1	0	30	18	17	218	8	10	9	10	114	7378	1	18	5	7	1	5	0	17	9	35	24	42	4	24	0	1	0	3	
0	8	0	20	9	3	35	0	1	8	5	13	35	2	0	0	3	72	5	1401	6	0	2	0	4	0	3	10	16	5	14	10	7	0	0	0	0	
4	5	1	0	4	5	2	0	1	3	3	8	29	1	0	2	2	36	0	10	3670	5	1	0	1	0	1	6	20	57	7	11	9	0	0	0	1	
15	7	0	1	0	0	4	0	0	0	3	6	55	1	0	1	2	10	1	1	5	256	2	0	2	0	3	6	1251	3	4	3	9	0	0	0	0	
0	4	0	0	0	0	3	0	2	13	6	6	28	0	0	3	2	32	0	1	8	2	2829	0	5	0	0	9	11	2	5	12	7	0	0	0	1	
3	36	0	2	29	48	62	1	2	174	13	19	171	2	4	2	13	136	14	14	54	26	1	5408	4	1	45	3	3	10	110	24	287	4	0	0	2	2
11	9	0	1	12	7	5	0	2	20	21	22	271	5	3	5	0	20	8	6	41	5	6	0	8109	0	4	43	60	20	17	14	8	0	0	0	3	
0	1	0	0	0	0	0	39	0	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	626	0	0	3	0	4	1	2	0	0	0	0	
13	24	0	0	6	61	19	2	2	15	16	14	46	2	0	1	4	29	10	18	16	7	1	2	4	0	5023	17	19	13	14	22	12	6	3	0	0	
3	10	1	1	2	0	0	0	3	3	0	7	35	0	2	5	1	0	1	0	8	2	6	0	1	1	0	7644	29	1	12	3	9	0	5	0	1	
336	26	1	2	11	1	6	0	0	2	13	26	191	1	1	6	10	19	6	12	21	245	5	1	2	0	3	28	11043	4	6	27	35	0	11	0	2	
1	4	0	1	9	5	1	1	0	6	3	14	38	3	0	0	3	45	1	6	75	3	1	1	0	0	1	5	10	905	6	2	8	0	2	0	2	
9	12	0	1	15	9	30	2	1	267	14	15	125	5	0	11	11	52	7	17	12	1	2	1	5	0	8	7	44	18	5120	17	20	0	2	0	1	
5	11	1	1	1495	17	3	0	3	21	7	35	74	8	0	5	9	69	5	11	26	6	4	1	8	1	5	14	46	7	14	4349	17	0	0	0	1	
11	211	2	8	28	19	24	0	2	32	39	46	186	6	1	2	12	106	19	21	59	31	7	8	15	0	15	46	68	31	37	25	10790	0	1	0	1	
8	4	0	4	2	3386	4	2	0	1	4	3	10	0	1	2	0	0	2	0	3	0	0	2	1	1	8	5	1	0	7	5	3	1318	4	0	0	
1872	0	0	0	0	4	3	0	0	0	3	6	9	0	2	4	1	7	0	1	2	0	0	2	0	0	2	1	34	2	15	1	4	7	2691	0	0	0
0	3	0	0	0	0	1	0	0	2	62	0	4	0	0	0	0	1	0	0	0	0	0	1	0	0	0	113	0	0	0	0	0	0	0	0	0	
1	4	0	0	0	1	2	0	0	13	0	1	12	33	0	1	0	10	0	2	2	0	0	4	0	0	1	5	6	1	3	0	0	0	0	0	659	

0.8275387780181769

Kernel: (16, 2), RMSprop, categorical_crossentropy

4093	0	0	0	0	1	17	0	0	9	2	6	29	1	0	3	0	1	0	2	2	2	8	0	14	0	2	7	3188	0	11	2	5	0	46	0	0	
0	4108	0	4	2	1	4	1	1	12	2	5	3	2	0	6	1	5	0	2	3	9	7	0	6	0	0	11	11	0	2	2	71	0	0	1	2	
1	0	263	0	1	0	7	0	0	9	0	8	0	10	0	0	28	20	0	7	0	0	0	1	3	0	0	0	0	0	2	1	1	3	0	0	0	
0	0	0	1022	0	0	5	0	0	1	0	0	0	0	0	0	1	0	0	9	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
5	13	1	4	1563	2	77	3	0	129	3	25	31	41	0	12	1	66	3	13	6	3	21	1	27	0	1	12	46	6	12	2661	21	0	3	0	1	
14	11	0	1	7	7103	146	4	0	81	18	6	20	10	0	4	0	5	5	4	7	7	3	9	7	0	34	10	18	1	16	7	28	29	2	0	2	
19	21	1	44	8	21	8914	2	1	172	12	21	39	31	0	11	2	49	9	263	4	2	18	3	36	0	6	16	45	3	19	25	28	0	3	0	3	
2	0	0	0	1	0	15	946	1	23	3	19	12	0	1	0	0	17	2	10	1	2	3	0	3	0	1	2	11	0	4	3	3	0	0	0	0	
0	3	0	0	2	0	4	1	913	38	2	5	8	1	0	0	1	3	0	12	1	1	5	0	7	0	0	4	3	0	4	2	5	0	0	1	0	
13	39	0	7	11	1	146	2	7	9134	10	78	63	89	0	47	11	56	12	33	13	3	35	0	35	1	5	16	70	9	96	44	59	0	6	0	2	
2	19	0	4	1	1	13	0	1	18	7736	5	7	6	0	0	0	1	7	2	2	1	3	0	6	1	1	1	11	0	2	1	10	0	2	11	0	0
4	15	1	5	7	0	47	4	2	87	4	10374	15	11	0	1	5	32	5	37	16	3	21	0	30	0	3	10	45	0	3	46	25	0	1	1	3	
46	62	1	7	32	7	247	8	4	440	37	63	13826	100	0	16	12	123	17	83	29	17	41	1	114	1	27	42	177	12	43	43	57	0	12	2	4	
0	2	0	0	2	11	142	1	1	96	0	9	5	1305	1	2																						

Kernel: (16, 2), Adadelta, cosine_proximity

2769	0	0	0	2	0	0	0	1	0	5	15	43	1	0	2	0	0	2	0	2	11	3	0	11	0	0	9	4555	0	8	1	5	0	6	0	0	
0	4046	0	3	3	0	0	1	1	4	5	10	11	0	0	3	1	2	3	1	4	18	2	0	9	1	6	24	3	1	2	5	115	0	0	0	0	
1	0	155	0	3	1	3	0	0	8	0	17	1	3	0	0	98	51	0	9	0	0	2	2	0	0	0	1	1	2	2	2	2	0	0	1		
0	4	0	888	0	0	33	0	0	8	1	0	4	0	0	0	1	1	0	82	1	1	1	0	3	0	0	3	3	0	3	1	0	0	0	0		
4	15	0	1	2508	4	11	3	8	55	6	53	72	5	0	12	6	81	10	8	15	7	32	1	33	0	2	17	56	4	18	1740	23	0	0	3		
10	11	0	8	20	7169	14	6	0	28	18	14	45	4	0	3	1	4	8	0	12	13	2	13	11	0	8	10	25	0	23	6	27	103	1	0	2	
19	36	1	192	72	84	7297	15	4	273	24	77	221	16	0	12	7	268	15	556	79	26	117	13	65	0	15	27	148	17	47	32	66	1	2	0	7	
0	1	0	1	2	0	3	906	0	7	4	60	29	0	0	0	0	15	4	4	4	2	6	0	3	0	0	3	19	0	5	4	3	0	0	0	0	
0	3	0	0	1	1	0	0	957	7	3	5	16	0	0	0	1	0	2	0	4	0	5	0	9	0	0	4	4	0	0	1	3	0	0	0	0	
11	44	0	13	35	4	30	9	17	8426	19	164	228	18	0	67	22	141	23	30	47	31	81	1	56	1	5	35	128	7	314	55	79	0	2	0	10	
1	13	0	3	1	0	0	0	7	7787	5	10	0	0	0	0	9	0	0	0	8	1	3	0	7	0	0	3	3	1	3	1	8	0	0	0	1	
1	13	0	4	1	0	0	8	2	9	3	10622	29	0	0	0	2	3	2	2	11	1	17	0	25	0	0	18	29	1	3	30	23	0	1	0	3	
18	39	0	8	31	2	27	9	12	65	34	77	14817	4	0	10	8	33	17	13	32	11	54	1	118	1	5	47	149	2	28	28	49	0	1	0	3	
5	2	0	0	10	17	44	5	2	83	2	25	27	1123	0	3	3	151	11	9	33	5	2	7	14	0	3	2	3	59	7	9	23	15	2	0	55	
3	0	0	0	0	0	0	0	0	1	0	0	5	0	0	1	0	0	17	0	0	0	2	0	2	0	2	1848	0	0	2	1	0	0	0	0		
1	3	0	0	2	1	3	0	1	38	0	2	7	0	0	2330	0	9	2	1	24	0	8	0	13	0	1	7	5	5	24	4	4	0	0	1		
1	5	7	0	1	0	1	0	1	34	0	17	17	1	0	3	1158	17	2	9	1	2	4	1	7	0	0	3	8	0	5	7	5	0	0	0		
11	29	3	2	64	20	77	33	17	298	9	251	241	21	0	18	33	3571	24	123	56	23	82	3	55	0	10	17	123	36	66	51	40	0	1	0	7	
6	4	0	1	1	2	7	3	0	39	6	23	76	0	0	12	3	45	7642	0	7	1	23	3	54	0	4	10	22	2	17	4	24	1	0	0	3	
0	15	0	135	10	2	93	1	3	31	4	30	28	2	0	1	6	92	9	1105	15	1	24	0	19	0	1	9	24	2	15	7	11	0	0	0	2	
1	4	0	1	7	4	4	2	1	8	2	16	23	1	0	7	1	25	8	8	3651	4	8	1	17	0	1	10	25	27	8	9	17	0	0	0	4	
4	10	0	0	0	0	1	0	1	2	2	14	48	1	0	1	1	10	1	0	8	254	9	0	9	1	0	7	1243	3	3	5	13	0	0	0	0	
0	5	0	0	0	0	0	0	2	2	3	2	2	0	0	1	1	0	0	0	1	0	0	0	0	0	2929	0	22	0	0	0	2	6	0	0	2	
1	27	1	0	18	8	12	6	9	141	7	30	69	2	0	3	2	43	21	1	28	8	10	5962	28	1	19	3	5	0	32	16	208	5	0	0	1	
4	4	0	0	3	0	0	0	0	1	6	11	4	0	0	2	0	0	2	1	3	0	0	0	8680	0	0	21	7	0	2	4	2	0	0	1		
0	1	0	0	0	0	0	25	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	650	0	0	0	0	0	1	0	0	0	0	0		
13	48	0	1	60	87	34	8	5	109	13	86	55	5	0	2	7	118	24	27	32	15	29	7	52	0	4453	36	19	11	19	29	20	14	3	0	0	
2	12	0	2	1	0	0	0	2	3	0	9	10	0	0	6	1	0	4	0	5	1	3	0	8	1	1	7695	6	0	9	1	13	0	0	0	1	
96	27	0	1	12	1	5	1	0	6	12	61	145	0	0	7	8	16	12	3	34	259	27	0	45	1	3	33	11198	3	8	29	46	0	2	0	2	
1	2	0	0	13	5	8	3	3	10	6	22	29	10	0	5	3	53	6	5	144	4	5	3	10	0	0	5	12	762	9	2	16	0	0	0	6	
7	10	0	2	8	3	21	5	6	479	13	21	74	2	0	22	8	26	11	9	13	1	26	3	32	0	2	8	37	8	4961	15	25	0	2	0	1	
3	10	0	0	1550	8	3	2	7	49	7	67	59	6	0	10	7	58	7	6	29	7	30	1	41	2	3	17	40	5	24	4194	25	1	0	0	1	
3	201	0	10	20	9	27	3	10	64	39	59	109	7	0	9	5	55	36	8	56	25	28	27	57	2	9	38	54	6	37	32	10857	0	0	0	7	
9	5	0	1	6	2416	0	2	0	2	5	12	6	0	0	1	0	2	3	0	3	0	0	5	8	1	3	6	3	0	8	4	6	2273	0	0	1	
2804	0	0	0	2	6	9	0	0	1	2	11	14	1	0	15	1	14	3	1	2	1	0	3	8	0	1	3	151	0	50	2	7	14	1547	0	0	0
0	3	0	0	0	0	0	0	1	1	65	0	2	0	0	0	0	1	1	0	0	0	0	0	0	0	1	110	0	0	0	0	0	0	0	0	0	
1	5	0	0	0	0	0	0	0	13	0	1	10	6	0	0	0	4	2	3	1	0	0	0	4	0	0	1	5	4	1	3	1	0	0	0	696	

0.8192433261659039

Kernel: (16, 2), Adam, categorical_hinge

3253	2	0	0	1	0	1	0	1	0	5	19	32	0	0	2	0	0	8	0	2	0	8	0	7	0	0	7	4069	0	17	0	4	0	12	0	1
0	4104	0	0	1	2	1	2	1	5	4	11	8	0	0	0	1	1	3	4	3	0	2	0	4	0	4	26	16	3	1	2	75	0	0	0	0
1	0	223	0	3	0	2	0	0	5	0	17	2	2	0	0	69	16	0	10	0	0	2	2	0	0	0	1	2	1	3	1	2	1	0	0	0
0	3	0	896	0	0	16	0	0	5	1	0	2	0	0	0	1	0	0	103	1	0	1	0	1	0	3	3	0	0	1	1	0	0	0	0	0
5	23	0	1	2041	16	38	4	25	80	6	57	123	9	0	14	11	38	13	9	14	0	46	2	31	0	3	18	82	27	22	2018	36	0	1	0	0
12	11	0	2	2	7170	14	8	0	15	19	14	40	2	0	3	0	0	12	0	12	0	5	15	12	0	3	11	37	6	19	6	29	137	1	0	2
20	37	1	75	41	87	7636	17	8	156	15	66	175	10	0	11	6	52	18	761	43	0	152	8	53	0	9	25	135	89	56	25	56	1	2	0	5
0	1	0	1	1	0	3	940	0	5	2	40	24	0	0	0	3	3	15	2	0	0	8	0	3	0	0	2	22	0	3	3	4	0	0	0	0
0	3	0	0	1	1	2	0	973	5	2	3	7	0	0	0	1	0	2	0	2	0	5	0	7	0	0	5	4	0	1	0	2	0	0	0	0
16	47	0	9	27	15	75	15	21	8271	12	175	229	18	0	49	19	32	35	65	36	0	119	1	57	1	6	33	147	78	395	49	89	0	3	0	9
1	15	0	2	0	0	5	1	1	9	7774	5	11	1	0	0	0	0	9	1	7	0	2	0	3	0	0	3	6	2	4	1	9	0	0	0	3
1	13	0	2	0	0	1	0	3	6	3	10680	17	0	0	0	2	0	4	9	5	0	17	0	10	0	0	8	18	2	3	35	20	0	1	0	3
30	47	1	6	12	5	57	11	21	66	24	79	14677	5	0	9	10	5	29	32	30	0	106	1	105	1	8	41	192	13	41	28	58	0	1	0	2
3	4	0	0	4	20	88	5	2	66	0	23	28	987	0	3	2	42	15	29																	

Kernel: (16, 2), Adamax, cosine_proximity

3943	0	0	0	1	0	1	0	1	0	3	10	37	1	0	2	0	0	0	0	2	13	5	0	4	0	1	12	3394	0	4	1	5	0	11	0	0	
0	4167	0	1	3	1	0	0	1	3	2	3	6	0	0	1	1	1	0	0	1	19	2	0	3	0	2	23	3	3	1	2	34	0	0	1	0	
1	0	227	0	8	0	4	0	0	12	0	15	2	2	0	0	48	21	0	6	0	0	3	2	0	0	0	2	2	0	4	3	0	0	0	0		
0	4	0	995	0	0	13	0	0	5	0	0	0	0	0	0	1	0	0	13	1	1	1	0	1	0	0	2	1	0	0	1	0	0	0	0		
4	15	0	2	3170	4	19	2	6	64	5	32	40	3	0	10	2	12	5	5	2	13	24	0	22	0	2	14	44	13	9	1254	16	0	0	0		
12	14	0	0	30	7211	29	2	0	41	11	7	35	2	1	3	0	0	6	0	6	15	3	13	8	0	10	10	20	3	16	5	28	75	1	0	2	
22	52	1	57	91	69	8322	4	6	296	13	51	113	6	1	13	4	46	12	207	21	23	56	2	45	0	12	40	94	61	33	28	45	0	2	0	3	
0	1	0	1	4	0	6	898	0	18	4	35	32	0	1	0	0	11	3	9	2	4	8	0	3	1	1	5	22	0	5	5	4	0	0	1	1	
0	3	0	0	2	1	2	0	973	5	2	3	8	0	0	0	1	0	0	1	0	0	5	0	7	0	0	4	3	0	0	1	3	0	0	0	0	
14	57	0	8	51	7	56	3	12	8956	11	115	145	14	0	61	12	20	18	10	18	32	50	1	40	1	9	38	91	43	130	48	69	0	2	0	11	
2	30	0	2	1	1	6	0	1	16	7715	4	17	2	2	0	0	0	7	1	8	1	2	0	4	0	0	3	17	4	6	1	8	0	0	12	2	
2	29	0	6	15	0	10	11	4	29	4	10486	39	0	0	1	2	7	6	7	10	8	25	0	14	0	0	16	44	1	6	48	28	0	1	0	4	
39	59	1	7	78	2	70	4	24	135	22	63	14673	5	2	13	7	15	11	15	21	24	49	1	75	0	8	39	151	22	29	31	53	0	2	0	3	
2	5	0	0	24	20	94	1	2	147	0	21	18	854	1	2	2	64	10	3	16	7	4	8	8	0	8	1	2	246	7	12	24	15	5	0	128	
3	0	0	0	0	0	0	0	0	1	0	0	6	0	534	1	0	0	2	0	0	0	4	0	0	1	1	1327	1	0	1	1	0	0	0	1	0	
1	4	0	0	7	1	7	0	0	34	0	2	6	0	0	2360	0	3	0	0	14	1	5	0	8	0	1	5	2	14	12	4	4	0	0	0	1	
1	9	5	0	4	0	2	0	0	82	0	15	13	1	0	3	1123	5	2	7	1	2	9	0	4	0	1	3	7	4	4	6	4	0	0	0	0	
10	44	7	3	227	43	244	16	9	723	4	223	235	13	1	25	23	2656	16	90	38	37	99	4	44	0	25	20	167	189	56	59	48	0	1	1	15	
6	5	0	1	7	5	18	3	0	71	6	14	85	2	16	16	3	24	7547	0	6	4	31	3	39	1	7	4	26	19	15	5	48	0	0	4	4	
1	21	0	102	11	3	169	1	2	68	3	23	23	1	0	2	3	37	7	1104	4	3	17	0	16	0	1	8	22	14	13	9	8	0	0	0	1	
4	8	0	1	13	5	10	1	1	12	1	21	18	2	0	5	1	14	5	6	3495	3	7	2	11	0	1	7	29	173	9	15	22	0	0	0	3	
12	10	0	1	2	0	8	0	1	6	2	4	30	0	0	1	1	1	1	1	4	334	8	0	4	0	3	7	1185	6	2	4	13	0	0	0	0	
0	6	0	0	0	0	0	0	2	4	1	1	1	0	0	3	0	0	0	0	0	0	2944	0	12	0	0	5	1	0	0	4	5	0	0	0	2	
3	45	0	2	36	23	73	1	7	172	1	16	63	0	4	6	2	4	10	1	9	14	5	5856	20	1	29	2	1	9	18	16	273	3	0	1	1	
4	9	0	0	4	0	0	0	0	2	5	13	12	0	2	3	0	0	1	0	6	1	4	0	0	8635	0	0	31	12	1	2	7	3	0	0	1	
0	1	0	0	0	0	0	11	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	1	661	0	0	2	0	1	0	0	0	0	0	0	0	
14	66	0	2	23	75	45	3	3	81	10	23	42	1	0	1	4	30	11	9	12	16	20	6	16	0	4766	22	17	41	12	27	19	17	5	1	1	
4	12	0	3	2	0	0	0	2	2	0	7	14	0	6	7	1	0	2	0	8	4	10	0	3	1	0	7658	15	1	8	3	13	0	0	10	0	
258	33	1	2	26	2	18	0	0	14	6	37	101	1	1	10	6	4	10	1	18	362	26	1	15	0	2	22	11026	10	12	28	46	0	2	0	2	
1	4	0	1	14	2	7	0	1	14	3	16	13	1	0	3	2	10	2	2	39	3	3	2	6	0	0	5	10	971	6	2	15	0	0	0	4	
8	17	0	4	21	5	34	4	5	793	9	17	58	2	0	28	3	12	10	7	10	2	14	2	22	0	4	10	24	20	4674	12	25	0	2	0	3	
5	14	0	1	2169	4	4	0	4	64	3	38	30	2	0	7	4	12	5	6	13	13	18	1	24	1	2	15	29	10	13	3745	20	2	0	0	1	
7	348	0	8	37	9	28	1	8	63	21	43	76	2	1	7	3	17	20	3	14	29	29	14	36	2	4	39	48	22	25	23	10917	0	0	1	4	
6	4	0	3	9	2371	5	2	0	4	2	6	4	0	2	1	0	0	2	0	3	1	0	3	5	1	4	4	2	0	4	5	4	2332	2	0	0	
2341	0	0	0	2	5	11	0	0	7	2	7	6	0	2	13	2	7	1	1	2	1	1	2	3	0	2	1	46	1	33	0	9	14	2151	0	0	0
0	3	0	0	0	0	1	0	1	1	3	0	3	0	0	0	0	0	0	0	0	0	1	2	0	0	0	71	0	1	0	0	0	0	0	100	0	
1	6	0	0	0	0	0	0	0	19	0	2	3	4	0	1	0	1	1	1	1	1	0	0	4	0	0	5	4	1	1	0	0	0	0	1	705	

0.8341803820653844

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

4205	2	0	0	1	2	2	0	0	5	16	51	0	0	3	2	7	6	0	4	3	6	0	11	0	1	11	3069	0	11	1	9	0	22	0	1	
0	4062	0	3	1	1	0	1	2	4	7	3	4	0	0	5	0	3	0	0	1	10	2	0	4	0	3	22	6	0	1	2	133	0	0	0	4
1	0	235	0	1	0	3	0	0	5	0	8	0	1	0	0	27	64	1	2	0	0	0	3	2	0	0	0	1	1	1	3	4	0	0	0	2
0	0	0	1010	0	0	4	0	0	1	1	0	0	0	0	1	2	12	0	0	0	0	0	1	0	0	2	1	0	0	1	2	0	0	0	0	0
2	13	2	2	2459	3	24	2	3	43	7	25	26	14	0	12	2	154	8	7	5	5	19	0	20	0	1	19	30	5	11	1850	35	0	1	0	4
13	7	0	1	10	7069	23	2	0	24	20	8	21	3	2	3	0	19	12	0	9	6	3	14	8	0	13	9	18	3	16	2	51	225	1	0	4
28	28	1	20	28	66	8256	2	5	180	35	37	85	39	3	12	8	390	26	185	18	10	29	9	38	0	12	27	56	22	46	29	105	3	3	0	10
1	0	0	0	2	0	6	951	0	3	5	16	14	1	1	0	0	42	5	2	2	2	3	0	3	2	0	3	10	0	3	4	4	0	0	0	0
0	3	0	0	1	1	2	0	986	4	4	4	2	0	0	1	1	0	0	1	0	5	0	3	0	0	2	1	0	0	1	4	0	0	0	0	0
12	44	0	5	28	13	72	4	11	8555	18	98	84	77	2	71	15	369	25	20	17	6	52	3	35	1	7	24	67	11	211	60	116	0	2	0	18
0	8	0	1	0	0	1	0	1	5	7818	2	4	0	1	0	0	2	6	0	3	2	3	0	4	0	0	3	1	0	2	0	8	0	0	0	0
1	19	2	2	11	0	15	5	2	21	5	10423	11	5	0	1	4	108	7	9	18	3	26	0	17	0	0	21	18	0	2	48	48	0	2	0	9
33	52	1	5	55	5	62	5	40	152	43	67	14177	52	7	14	11	302	60	15	35	6	49	1	96	2	26	44	99	14	52	46	114	0	5	0	6
2	2	0	0	4	13	35	1	1	54	1	10	8	1144	2	3	2																				

Kernel: (16, 2), Nadam, categorical_hinge

3923	0	1	0	0	2	0	1	0	4	8	32	0	0	1	2	0	4	0	1	0	3	0	3	0	1	7	3430	0	7	2	5	0	13	0	1		
0	3875	0	1	2	11	4	0	2	35	8	15	18	0	0	1	2	4	5	2	3	0	2	0	14	0	15	19	24	2	1	5	214	0	0	0	0	
1	0	0	0	3	3	4	0	1	17	0	15	5	4	0	0	175	116	0	2	0	0	1	2	0	0	0	2	0	1	4	5	3	1	0	0		
0	3	0	0	922	0	0	17	0	0	15	2	0	0	0	1	3	0	57	0	0	1	0	2	0	0	4	7	0	0	2	1	0	0	0	0		
4	9	0	1	2090	18	13	0	10	128	9	28	126	1	0	6	10	128	13	3	5	0	18	0	40	0	2	20	68	9	17	2010	25	0	1	0	1	
12	3	0	0	2	7395	9	0	0	28	14	6	33	1	0	1	0	1	8	1	2	0	0	8	0	6	10	35	0	15	7	6	12	2	0	2		
35	22	0	31	60	161	6986	0	7	668	23	44	307	8	0	5	14	512	26	408	23	0	49	1	68	0	18	31	171	54	34	28	51	0	3	0	3	
1	0	1	0	10	19	84	0	1	79	5	125	112	0	0	0	3	498	8	42	3	0	5	0	4	3	2	4	49	0	12	11	4	0	0	0		
0	3	0	0	1	1	0	0	955	12	2	4	19	0	0	0	1	0	2	0	2	0	4	0	7	0	0	3	3	0	1	2	4	0	0	0	0	
15	31	0	2	18	12	15	0	12	9058	15	96	197	6	0	9	24	74	26	15	10	0	38	1	53	0	3	24	142	13	131	48	59	0	4	0	2	
1	5	0	1	0	0	5	0	1	22	7793	1	8	0	0	0	0	0	4	1	3	0	2	0	7	0	0	2	11	0	3	0	5	0	0	0	0	
2	9	0	1	3	1	3	0	3	37	5	10509	37	0	0	0	8	18	16	10	6	0	23	0	8	0	1	6	52	1	4	71	26	0	0	0	3	
33	30	0	6	12	10	14	0	14	100	42	55	14892	3	0	7	11	30	23	10	9	0	30	1	112	0	7	30	169	9	21	26	44	0	2	0	1	
3	1	0	0	8	29	37	0	4	301	1	17	40	697	0	0	3	338	28	5	13	0	2	4	13	0	6	1	5	119	2	10	21	10	6	0	37	
3	0	0	0	0	0	0	0	1	6	1	0	12	0	0	1	2	0	27	0	0	0	3	0	1	0	7	1817	2	0	1	0	0	0	0	0		
2	2	0	0	5	3	8	0	1	277	0	2	13	0	0	1973	2	42	7	3	30	0	9	0	18	0	1	8	6	25	44	5	8	0	2	0	0	
1	5	0	0	0	0	0	0	1	50	0	9	17	0	0	2	1179	12	2	2	1	0	5	0	6	0	1	2	7	1	3	6	5	0	0	0	0	
10	21	0	1	53	49	46	0	9	564	13	137	277	7	0	8	40	3632	32	48	14	0	42	1	49	0	14	16	165	49	24	52	38	0	1	0	3	
6	3	0	1	1	3	7	0	0	38	10	8	59	0	0	3	5	25	7755	0	4	0	15	0	43	0	3	4	21	1	7	2	19	0	1	0	1	
1	10	0	37	7	6	82	0	2	116	4	22	36	2	0	0	8	145	9	1106	3	0	8	0	17	0	2	9	30	6	8	10	11	0	0	0	0	
1	2	0	1	5	7	6	0	1	24	5	15	44	2	0	2	2	63	19	7	3406	0	6	1	17	0	1	10	45	161	8	16	25	0	1	0	2	
20	9	0	0	1	1	0	0	1	5	4	6	40	0	0	1	2	8	3	1	3	0	7	0	4	0	1	4	1503	8	2	5	12	0	0	0	0	
0	1	0	0	0	0	0	0	2	7	3	1	3	0	0	1	1	0	1	0	1	0	2930	0	16	0	1	5	2	0	2	5	9	0	0	0	0	
3	18	4	0	0	27	118	83	0	7	962	10	16	120	2	0	4	6	148	32	3	24	0	8	4525	30	0	25	2	8	20	31	21	462	7	0	0	1
6	3	0	0	0	0	0	0	0	6	6	20	9	0	0	0	0	0	4	1	1	0	3	0	8650	0	0	17	22	0	2	5	2	0	0	0	1	
0	2	0	0	0	2	0	0	0	3	2	0	3	0	0	0	0	1	2	0	1	0	0	1	2	646	0	0	8	0	2	0	4	0	0	0	0	
16	19	1	1	12	120	22	0	4	160	11	26	44	3	0	1	7	72	25	10	12	0	10	3	56	0	4679	13	40	12	11	27	17	4	3	0	0	
2	12	0	0	0	0	0	0	5	12	0	18	37	0	0	2	1	1	14	0	2	0	9	0	11	1	5	7595	45	0	7	3	12	0	1	0	1	
281	26	0	0	4	2	1	0	0	10	11	31	100	0	0	4	9	16	17	3	11	0	24	0	24	0	4	19	11401	8	8	33	50	0	4	0	2	
1	1	0	0	6	8	2	0	1	31	5	14	34	2	0	0	5	99	7	5	26	0	3	1	7	0	1	5	14	856	5	4	14	0	1	0	4	
13	5	0	1	9	6	10	0	3	1145	12	16	70	2	0	7	12	44	17	4	6	0	14	1	25	0	5	9	28	7	4352	12	24	0	2	0	0	
5	8	0	0	1	1106	28	4	0	7	116	4	35	103	0	0	4	12	95	11	4	10	0	15	0	39	1	4	14	49	5	17	4554	29	0	0	0	
5	88	2	6	15	54	68	0	6	169	36	43	115	1	0	4	12	95	58	12	27	0	15	1	65	0	18	27	73	55	27	26	10780	0	2	0	4	
8	3	0	1	0	3854	0	0	0	3	4	5	5	0	0	2	0	2	1	0	3	0	0	1	8	0	3	6	0	0	5	4	3	865	5	0	0	
2294	0	6	0	2	7	1	0	0	2	1	8	12	0	0	1	1	8	2	1	1	0	1	2	2	0	1	3	71	0	22	1	6	7	2210	0	0	
0	3	0	0	0	0	1	0	2	3	60	0	4	0	0	0	0	1	2	0	0	0	2	0	0	0	1	108	0	0	0	0	0	0	0	0		
1	4	0	0	0	1	0	0	0	62	2	1	8	12	0	0	0	12	3	2	1	0	0	0	5	0	0	6	5	0	5	1	0	0	0	0	630	

0.8023426828790797

Kernel: (16, 2), Nadam, logcosh

3030	0	0	0	1	0	1	0	0	3	4	32	1	0	2	0	0	0	0	1	4	5	0	5	0	1	10	4326	0	15	1	3	0	6	0	0	
0	4143	0	1	3	1	0	2	1	10	4	3	15	0	1	4	1	3	0	0	1	15	4	0	3	0	1	16	9	0	1	3	38	0	0	0	1
1	0	0	173	0	7	0	7	0	14	0	14	5	4	0	0	73	51	0	2	0	0	1	2	0	0	0	2	0	2	3	2	1	0	0	0	0
0	3	0	1001	1	0	14	0	0	4	0	1	0	0	0	1	1	0	5	1	0	0	0	0	0	0	4	2	0	0	1	0	0	0	0	0	
3	13	0	1	3222	3	18	4	4	70	3	20	60	4	0	7	2	33	5	5	3	4	25	0	19	0	1	15	55	3	15	1182	14	0	0	0	0
10	8	0	0	14	7234	66	31	0	29	17	5	39	1	2	3	0	2	6	0	6	10	3	7	7	0	12	10	25	4	21	3	14	26	1	1	2
18	34	1	23	45	48	8357	12	3	302	15	40	234	22	4	12	5	168	15	58	16	18	50	4	33	0	13	27	122	10	85	20	32	0	2	0	3
0	0	0	0	1	0	3	990	0	5	2	11	24	0	1	0	1	6	3	2	1	2	3	0	2	1	0	2	15	0	6	2	2	0	0	0	0
0	3	0	0	3	0	1	0	965	9	2	5	12	0	1	0	1	1	0	0	1	0	0	0	0	0	0	2	3	0	1	1	3	0	0	0	0
8	41	0	6	45	6	56	12	7	8808	12	96	194	26	2	53	13	65	18	2	15	12	50	2	26	1	4	19	109	3	343	41	55	0	1	0	2
1	22	0	1	2	1	5	1	1	13	7761	3	16	0	0	0	0	1	5	0	0	3	1	3	0	0	2	13	0	4	1	10	0	0	0	0	2
3	21	0	4	23	1	11	42	2	36	4	10401	57	3	1	0	2	11	7	6	8	6	28	0	25	0	1	20	72	0	7	36	21	0	0	0	4
24	44	0	6	41	3	27	4	6	89	17	42	14963	5	4	10	10	17	3	11	13	41	1	65	1	9	37	154	3	39	22	30	0	1	0	1	
2	4	0	0	8	18	102	6	1	225	0	14	37	973	2	3	3	171	13	1	16	6	4	7	6												

ARat_2016_07_20_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

6662	5	0	4	3	0	4	1	2	0	1918	1	23	1	8	15	0	4	72	18	38	21	3	8	2	2	5	0	0	0	27	15	2	0	0	69	
14	2561	0	13	13	4	3	8	4	0	1	13	11	0	3	31	0	14	36	38	79	1404	1	6	8	4	2	17	0	14	46	74	2	1	2	1616	
0	0	0	0	0	0	0	0	0	0	1	0	261	0	47	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0		
30	16	0	5855	9	1	23	1	9	0	21	0	55	8	6	0	0	4	2	103	54	24	20	1	5	1	23	8	0	0	18	28	4	0	4	52	
18	10	0	7	5157	0	1	1	1	0	9	0	28	0	3	0	0	2	2	59	9	143	2	0	6	5	5	9	0	0	8	9	4	0	4	27	
4	11	0	1	3	3609	1	0	0	0	0	8	3	1	0	44	0	3	21	25	22	25	121	5	6	3	2	33	0	7	51	28	19	0	10	4	
11	11	0	12	5	0	1753	0	6	0	3	0	18	0	3	0	0	3	2	30	16	5	11	0	2	1	4	4	0	0	16	8	3	0	2	16	
10	2	0	1	0	0	2	1951	25	0	3	0	4	0	5	0	0	2	2	1	29	7	0	0	1	0	0	0	0	2	3	0	0	0	35		
4	5	0	7	3	0	3	2	827	0	3	0	8	0	16	0	0	0	3	10	35	13	2	0	2	0	2	2	0	1	14	10	2	0	2	11	
5	0	0	0	2	0	7	3	222	0	0	0	4	0	12	0	0	2	1	13	4	4	2	0	1	1	2	1	0	1	5	7	0	0	1	3	
480	0	0	0	1	0	0	0	0	0	3694	0	5	2	1	0	0	0	2	3	1	0	2	5	0	0	0	0	0	0	0	0	0	0	0	5	
1047	0	0	9	3	0	4	1	0	0	11	4040	1	0	1	29	0	12	2321	22	12	20	0	10	7	0	0	0	0	5	11	6	4	0	0	9	
16	1	0	5	2	0	0	2	0	0	22	0	6739	0	12	0	0	2	0	70	21	12	4	5	4	0	27	0	0	1	48	28	1	0	0	57	
33	23	0	7	0	29	1	2	5	0	19	1	14	3603	1	11	0	2	15	114	56	6	2	3	0	3	1	40	0	0	14	23	2	3	5	10	
2	1	0	1	0	0	0	0	10	0	1	0	1	0	1436	0	0	4	0	9	3	2	0	0	0	0	5	0	0	0	5	2	0	0	0	11	
20	7	0	6	11	0	12	1	1	0	3	6	4	0	11	13175	0	17	2	74	13	24	0	21	5	0	0	872	0	7	19	12	1	0	1	0	
3	0	0	2	0	0	0	1	0	0	1	0	491	0	3	0	0	1	0	17	3	2	0	2	0	0	2	0	0	0	9	2	0	0	0	5	
16	6	0	16	10	0	3	3	0	0	8	0	4	3	2	0	0	7146	1	47	47	23	0	0	3	0	15	8	0	0	12	24	0	0	0	86	
4951	67	0	58	28	0	44	19	45	0	173	98	116	7	41	91	0	36	7186	420	227	128	30	40	44	9	32	5	0	10	144	105	19	0	8	124	
36	37	0	22	10	0	3	6	9	0	10	2	87	12	57	2	0	27	14	11085	106	45	19	5	8	1	36	5	0	5	59	56	3	1	4	115	
15	4	0	17	7	0	4	3	2	0	4	2	31	2	8	2	0	3	2	33	6199	38	3	5	5	1	16	6	0	7	29	25	2	0	5	83	
6	68	0	12	11	0	1	10	0	0	1	7	8	0	2	5	0	31	7	7	36	7156	0	11	6	0	7	0	0	14	35	47	0	0	1	264	
38	60	0	9	4	57	13	4	24	0	62	0	20	48	5	10	0	1	19	206	70	35	7515	1	2	24	4	106	0	0	133	95	46	2	50	34	
6	0	0	1	5	0	0	1	1	0	33	2	1	0	0	7	0	8	1	4	10	6	0	4434	2	0	598	8	0	2	4	3	0	0	0	4	
14	2	0	18	4	0	4	3	8	0	1	0	18	2	2	0	0	1	0	25	11	16	8	0	4719	0	7	0	0	0	4	7	1	0	0	26	
83	206	0	25	45	1	41	7	77	0	5	3	93	16	19	14	0	11	51	520	216	70	48	5	13	2753	33	70	0	6	106	178	4	1	13	66	
0	0	0	0	1	0	1	0	1	0	17	0	8	1	2	0	0	0	0	5	7	4	1	68	1	0	2	6	0	0	2	6	0	0	0	25	
8	83	0	12	9	0	7	2	4	0	40	0	24	25	15	33	0	9	1	188	55	19	19	8	5	2	15	11086	0	0	63	32	15	0	4	30	
0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	130	2	0	0	0	8	0	
0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	1	0	0	556	2	1	0	0	11	3	
24	11	0	3	8	0	1	10	13	0	8	1	34	3	18	1	0	3	1	11	55	33	9	4	4	0	46	6	0	1	8077	53	3	0	1	115	
31	39	0	24	10	0	3	16	12	0	8	1	57	3	3	2	0	10	8	64	108	89	13	2	16	3	18	8	0	8	95	8109	7	0	6	171	
5	1	0	1	2	0	1	0	0	0	2	0	4	3	0	2	0	0	0	10	6	2	0	0	2	0	5	7	0	0	4	8	2380	0	0	4	0
0	0	0	1	0	0	0	0	0	0	13	0	0	1031	0	0	0	0	0	14	2	0	7	0	1	2	0	0	0	0	1	0	708	0	1	0	
0	6	0	2	2	1	1	0	1	0	0	1	3	0	0	10	0	3	20	23	28	11	5	1	1	0	1	7	0	3	19	12	0	0	1145	1	
1	51	0	0	0	0	0	0	0	0	5	1	2	0	0	0	0	1	0	3	15	14	0	0	0	0	3	0	0	4	0	0	0	0	0	7215	

0.836475032567978

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

8319	2	0	4	3	0	2	1	1	1	106	15	19	1	4	2	2	2	311	18	15	14	5	7	2	3	2	0	0	0	9	8	0	0	0	55	
17	3145	2	13	23	6	4	10	4	3	0	17	25	0	0	21	1	11	93	147	21	1325	5	5	8	29	0	17	0	11	24	68	4	0	3	981	
0	0	0	0	0	0	0	0	0	0	1	0	196	0	11	0	0	0	0	229	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	10	0	5921	12	0	36	1	8	0	2	0	45	10	1	0	0	1	6	97	32	18	19	3	3	13	12	7	0	0	12	19	3	0	1	47	
21	5	0	8	5304	0	3	1	2	0	2	0	18	0	2	0	0	3	38	7	46	8	1	1	12	2	10	0	0	5	7	1	0	0	22		
3	10	0	1	3	3447	2	0	0	0	0	8	4	2	0	36	1	3	34	30	4	23	353	7	5	5	0	34	0	7	16	5	18	0	8	1	
15	4	0	12	3	0	1787	0	7	0	0	0	14	0	3	0	0	0	7	25	12	5	13	0	2	7	3	2	0	0	5	5	1	0	0	13	
17	5	0	1	1	0	4	1985	11	0	0	1	9	1	0	0	0	2	2	8	8	0	0	0	0	0	0	0	0	1	5	0	0	0	0	24	
9	5	0	6	1	0	2	2	849	3	0	0	8	0	1	0	0	0	4	33	22	8	4	0	2	4	2	0	0	5	6	0	0	2	9	0	
3	0	0	0	2	0	2	0	41	234	0	1	1	0	0	0	0	0	8	0	0	2	0	0	0	1	0	0	0	2	3	0	0	1	2	0	
2640	0	0	0	0	3	0	0	0	0	1522	6	5	3	0	0	1	0	5	3	0	1	1	7	0	0	0	0	0	0	0	0	0	0	0	4	0
662	3	0	6	2	0	1	0	0	0	2	4752	1	0	1	11	1	5	2097	6	1	10	0	4	3	0	0	0	0	2	5	5	0	0	0	5	0
34	1	0	2	5	0	0	2	0	0	3	0	6724	0	3	0	0	0	3	176	12	9	4	3	2	1	18	0	0	1	22	20	1	0	0	33	
54	7	0	3	0	33	2	3	1	0	1	2	11	3623	0	6	0	1	16	193	9	4	7	3	0	9	1	37	0	0	4	8	3	2	0	5	
7	2	0	1	0	0	1	0	31	0	0	2	1	1359	0	0	1	0	54	3	3	0	0	0	0	5	0	0	0	8	2	0	0	0	10	0	
24	10	0	10	19	0	18	1	1	0	0	15	3	0	6	12255	6	16	34	93	10	23	0	19	5	13	0	1696	0	5							

Kernel: (4, 2), Adamax, cosine_proximity

8271	3	0	19	6	0	3	1	3	1	92	25	22	1	4	16	0	2	269	8	26	17	0	13	7	13	2	0	0	0	26	29	0	0	0	54	
16	2616	0	41	25	3	4	12	6	2	0	17	33	2	2	22	1	7	102	46	41	1839	1	5	9	42	1	15	0	12	47	138	3	0	5	928	
0	0	0	0	0	0	0	0	0	0	1	0	432	0	1	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0			
25	1	0	6175	2	0	9	1	1	0	2	0	29	2	2	0	0	0	1	3	24	16	0	0	3	9	11	2	0	0	8	21	0	0	1	37	
14	1	0	26	5279	0	2	0	1	0	2	0	17	1	2	0	0	1	3	19	9	51	6	0	6	40	3	5	0	0	5	13	0	0	3	20	
5	9	0	5	6	3558	6	0	0	1	0	8	4	7	0	40	0	2	42	27	5	30	126	4	6	26	4	44	0	6	20	42	20	0	16	1	
14	2	0	88	4	0	1704	0	6	1	0	0	14	3	3	0	1	1	5	12	12	5	5	0	3	24	6	2	0	0	5	10	0	0	2	13	
12	1	0	4	2	0	4	1967	20	2	0	3	7	1	0	0	0	0	1	0	12	9	0	0	1	0	0	0	0	2	13	0	0	0	24		
7	3	0	17	2	0	2	3	851	3	0	0	10	1	5	0	1	0	5	2	22	12	1	0	2	9	1	0	0	4	12	1	0	2	9		
3	0	0	0	2	0	2	0	35	245	0	1	1	0	0	0	0	0	0	2	1	0	0	0	1	0	1	0	0	2	4	0	0	1	2		
2368	0	0	0	0	0	0	0	0	0	1796	8	6	2	1	1	0	0	9	0	1	1	0	5	0	0	0	0	0	0	0	0	0	0	3	0	
599	2	0	14	3	0	1	0	0	0	2	4903	1	0	1	18	0	7	1968	3	6	11	0	9	9	2	0	0	0	2	11	8	1	0	0	4	
33	1	0	13	7	0	0	2	0	1	2	0	6799	0	3	0	8	0	4	33	13	9	0	2	6	1	30	0	0	1	35	43	0	0	0	33	
55	6	0	23	0	22	1	3	2	0	4	1	15	3723	0	6	0	1	17	43	13	6	0	2	0	23	2	40	0	0	5	23	1	3	4	4	
4	1	0	3	0	0	0	0	19	0	0	0	6	0	1418	0	0	1	1	11	3	2	0	0	1	0	5	0	0	0	6	2	0	0	0	10	
16	2	0	22	19	0	12	1	1	0	0	15	4	0	9	12683	0	16	16	48	27	23	0	21	7	30	1	1294	0	6	26	23	0	0	3	0	
5	0	0	5	0	0	0	2	0	0	0	0	492	0	2	0	5	1	0	9	1	1	0	2	1	0	3	0	0	5	6	0	0	0	4	4	
34	34	0	237	32	0	7	7	33	0	2	0	6	41	1	1	1	6526	5	114	49	56	0	1	13	35	24	27	0	0	31	74	0	1	4	87	
3421	19	0	212	28	0	28	13	19	2	2	223	89	13	24	49	3	24	9203	122	102	120	3	38	58	98	26	2	0	7	104	152	4	0	8	89	
55	43	0	84	19	0	5	9	23	4	1	3	125	35	48	2	12	17	37	10877	60	52	11	3	22	48	43	13	0	6	42	98	3	1	4	82	
21	5	0	55	9	0	3	6	6	1	1	2	57	6	6	1	0	2	17	24	6042	52	3	3	9	26	19	3	0	9	27	73	1	0	3	71	
8	52	0	21	20	0	2	10	2	2	0	12	15	0	2	5	0	20	11	4	20	7260	0	10	17	9	6	0	0	13	34	88	1	0	2	107	
73	56	0	158	4	51	39	2	33	2	22	0	26	91	4	10	1	1	58	245	32	44	7072	1	4	340	2	99	0	0	24	126	35	2	26	14	
13	1	0	8	6	0	0	1	0	0	10	5	1	0	0	5	0	6	3	1	11	8	0	4083	3	7	910	5	0	2	40	7	0	0	0	5	
7	0	0	53	4	0	3	3	4	0	0	0	7	2	1	0	0	0	2	3	6	5	0	0	4760	0	5	0	0	5	9	0	0	0	22		
55	25	0	42	20	0	10	6	7	1	0	5	57	8	1	4	1	3	31	56	59	46	5	3	12	4100	23	7	0	1	43	126	1	0	4	37	
0	0	0	4	4	0	1	2	0	0	4	0	4	0	2	0	0	0	2	3	3	0	0	57	1	0	5334	0	0	0	5	12	0	0	0	13	
28	46	0	96	19	0	16	2	3	0	9	0	28	37	12	38	0	8	8	116	61	18	8	8	8	74	18	10995	0	0	50	67	10	0	6	24	
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	128	2	1	0	0	9	0	
0	1	0	3	0	0	0	1	0	0	0	0	1	1	0	0	0	1	3	0	1	3	0	0	0	0	1	0	0	524	2	3	0	0	33	1	
28	6	0	31	8	0	2	12	17	1	1	1	51	5	17	0	2	3	6	3	46	32	4	2	6	23	41	0	0	8012	101	3	0	0	0	93	
27	14	0	46	8	0	3	11	9	1	2	1	44	6	2	0	1	1	14	38	34	66	4	1	17	23	14	5	0	5	46	8386	2	0	4	109	
8	2	0	7	2	0	1	0	0	0	0	0	7	3	0	3	0	0	3	14	9	2	0	0	4	0	6	9	0	0	3	11	2353	0	0	2	0
1	0	0	3	0	0	0	0	0	0	12	0	4	1294	0	0	0	0	0	9	2	0	1	0	1	2	0	0	0	0	1	0	451	0	0	0	
0	4	0	18	3	2	0	0	2	0	0	1	4	0	0	8	0	3	34	16	13	12	1	1	1	8	2	7	0	2	15	29	0	0	1120	1	
9	135	0	2	1	0	0	0	0	0	1	1	18	1	2	0	0	1	0	2	15	24	0	0	0	1	8	0	0	4	7	1	0	0	7082	0	

0.8484274506568908

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

8370	2	0	5	3	0	0	1	0	1	287	33	23	0	4	11	1	1	96	4	12	13	0	10	3	3	4	0	0	0	7	1	0	0	0	38	
28	1832	0	11	30	8	5	14	6	7	0	22	49	0	7	38	1	16	122	85	34	2822	1	8	9	85	2	43	1	14	43	38	1	0	3	658	
0	0	0	0	0	0	0	0	0	1	0	0	329	0	38	0	0	0	68	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
45	7	0	5973	14	1	22	1	7	1	5	1	55	3	4	0	0	4	3	39	39	22	0	2	12	28	26	10	0	1	12	11	0	0	3	34	
28	1	0	8	5323	0	2	1	0	0	2	0	20	0	2	0	0	3	21	9	38	1	0	3	23	5	17	0	0	4	1	0	0	0	17		
4	1	0	1	5	3784	2	0	0	0	0	11	7	0	0	48	1	4	30	13	8	25	6	7	6	13	2	35	0	7	18	3	13	0	15	1	
20	0	0	13	6	0	1777	0	5	1	0	0	20	0	3	0	0	1	6	17	10	4	1	0	2	24	8	5	0	0	5	3	0	0	2	12	
15	0	0	0	0	0	2	2032	2	0	0	1	7	0	0	0	0	0	1	0	3	3	0	0	0	0	0	0	0	0	0	0	0	0	0	19	
10	2	0	6	1	0	4	4	861	7	0	0	15	0	8	0	0	0	4	4	20	9	1	0	2	13	2	3	0	1	3	0	0	0	1	6	
3	0	0	0	2	0	2	0	8	275	0	1	1	0	0	0	0	1	0	1	0	0	0	1	1	2	0	0	0	0	2	1	0	0	0	2	
1809	0	0	0	0	3	0	0	0	0	2360	11	5	2	1	0	0	0	2	0	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0	2
605	0	0	5	3	0	0	1	0	0	2	6179	1	0	1	16	1	5	729	3	1	8	0	11	5	1	0	0	0	2	4	0	0	0	0	2	
36	1	0	3	5	0	0	3	0	0	4	0	6873	0	6	0	0	1	0	46	11	6	0	4	4	1	20	0	0	0	27	5	0	0	0	23	
60	14	0	7	4	34	5	7	4	0	6	4	20	3469	1	17	0	1	23	206	15	12	0	3	1	56	1	54	0	0	4	10	2	1	5	2	
4	1	0	1	0	0	0	10	0	0	0	3	0	1439	1	0	3	1	5	3	1	0	0	1	0	6	1	0	0	6	0	0	0	0	7	0	
13	0	0	6	13	0	9	1	2	0	0	21	3	0	7	13695	0	13	4	14	7	16	0	26	6	10	1	431	0	5	18	4	0	0	0	0	
6	0	0	1	0	0	0	1	0	0</																											

Kernel: (16, 2), RMSprop, categorical_crossentropy

7929	2	0	4	3	0	9	1	0	1	356	5	35	1	4	19	2	0	368	3	27	26	11	7	6	10	7	11	0	0	14	11	0	0	1	60	
7	3193	0	11	14	0	3	8	2	1	0	8	21	0	0	38	0	10	74	34	24	1408	0	8	8	36	1	51	1	10	15	53	0	0	2	1002	
0	0	260	0	0	0	0	0	0	0	1	0	154	0	0	0	0	0	16	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0	0		
39	11	0	5948	6	0	35	1	1	0	5	0	45	6	3	0	9	3	7	19	41	26	9	2	15	21	23	34	0	0	9	19	0	0	3	45	
23	10	0	5	5217	0	4	1	0	0	4	0	25	2	2	0	1	1	3	15	10	71	2	0	9	32	6	39	1	0	8	8	0	1	9	20	
2	39	0	3	3	3530	6	1	0	0	0	3	8	1	0	43	0	3	50	17	24	37	81	7	8	17	3	108	1	8	18	18	13	0	17	1	
10	11	0	11	2	0	1789	0	0	0	0	0	15	0	2	0	6	1	3	10	11	5	6	0	2	8	7	22	0	0	2	5	0	0	4	13	
12	2	0	1	0	0	6	1972	8	0	0	1	8	0	0	0	1	0	3	2	19	8	0	0	0	3	2	0	0	2	5	0	0	0	0	30	
6	6	0	7	1	0	9	2	845	0	0	0	10	0	1	0	2	0	6	3	21	10	1	0	4	9	4	15	0	0	5	5	0	0	6	9	
4	1	0	0	2	0	5	0	52	209	0	0	1	0	0	0	0	0	2	3	3	2	2	0	1	2	2	1	0	1	2	3	0	0	1	4	
1683	1	0	0	0	4	0	0	0	0	2469	0	5	5	1	2	0	0	11	1	1	2	2	3	0	0	0	7	0	0	0	0	0	0	0	0	4
710	4	0	6	2	0	2	2	0	0	2	3213	1	0	1	32	0	5	3536	6	6	22	0	5	10	2	0	0	3	5	6	0	0	0	0	4	
19	6	0	4	3	0	0	2	0	0	5	0	6868	2	1	0	9	1	1	22	15	14	1	3	4	3	16	2	0	2	22	19	0	0	0	35	
40	22	0	6	0	28	1	3	0	0	7	1	13	3722	0	12	0	1	13	57	14	3	0	2	1	19	2	53	1	0	4	9	2	1	4	7	
2	3	1	1	0	0	2	0	12	0	0	0	6	0	0	1423	0	0	3	1	6	3	2	1	0	2	0	4	3	0	0	9	1	0	0	1	7
5	2	0	4	8	0	8	1	1	0	0	6	1	0	3	13067	0	9	10	10	7	14	0	14	7	7	0	1126	0	5	5	0	0	0	0	6	
2	1	0	1	0	0	0	1	0	0	0	0	504	0	0	0	12	0	0	4	1	2	0	0	1	0	3	0	0	0	4	2	0	0	0	0	
43	34	0	25	10	0	8	10	4	0	1	0	6	3	1	0	11	6840	4	69	27	34	7	0	14	21	34	118	0	0	15	33	0	0	1	110	
3296	40	0	51	22	0	54	14	6	2	15	45	109	7	14	69	25	20	9639	108	101	123	34	33	50	69	38	43	1	7	79	76	0	0	8	107	
52	107	0	30	11	0	9	8	8	3	3	2	137	42	22	3	72	15	35	10780	71	43	33	7	21	54	42	88	4	3	44	45	1	0	5	87	
16	11	0	16	5	0	4	5	0	1	3	1	42	8	3	3	5	6	14	28	6111	39	2	6	11	16	26	28	0	7	20	32	0	0	5	89	
4	164	0	12	8	0	3	11	1	1	1	6	13	0	2	8	1	28	7	5	16	7211	0	12	11	11	6	5	0	14	17	26	0	0	1	148	
68	163	0	29	3	80	55	3	5	0	18	0	24	62	3	8	7	2	53	186	45	49	6104	0	6	162	3	1288	2	0	26	82	22	0	126	13	
16	0	0	2	4	0	0	1	0	0	8	4	3	0	0	10	0	7	5	3	6	8	0	4339	3	5	657	53	0	2	1	0	0	0	0	4	
7	0	0	5	3	0	2	4	3	0	0	0	8	3	1	0	1	0	2	2	4	9	0	1	4807	2	4	3	0	0	6	3	0	0	0	21	
54	75	0	13	14	0	24	6	3	1	1	5	54	6	1	11	2	4	38	50	70	50	14	2	12	4043	29	82	0	3	32	57	1	0	1	41	
0	0	0	0	0	0	1	0	0	0	4	0	13	0	1	0	0	0	2	3	1	1	1	61	1	1	5347	2	0	0	0	0	0	0	0	12	
25	17	0	8	7	0	3	1	3	0	5	0	24	9	6	12	0	5	4	23	28	12	8	6	5	10	18	11510	2	0	19	15	4	0	2	22	
0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	12	114	1	0	0	11	0		
0	3	0	2	0	0	0	0	0	0	0	0	1	0	0	0	0	2	2	1	1	1	0	0	0	0	1	0	1	526	0	3	0	0	35	0	
31	30	0	8	7	0	5	13	9	0	2	1	56	4	5	1	24	3	11	4	72	52	6	2	5	22	61	65	0	4	7874	58	0	1	5	116	
35	54	0	31	9	0	24	12	15	0	2	0	93	11	1	15	4	45	47	76	106	25	3	22	64	27	90	0	6	47	7899	1	0	10	169		
10	7	0	4	2	0	9	0	1	0	0	0	10	6	0	3	0	0	11	12	10	3	3	0	8	0	7	50	0	0	3	17	2265	0	4	4	
0	0	0	0	0	0	0	0	0	0	11	0	1	1580	0	0	0	0	8	0	2	5	0	0	2	0	1	0	0	0	0	0	171	0	0		
0	11	0	5	1	0	4	1	0	0	0	1	4	0	0	9	0	3	37	20	32	10	0	1	2	14	2	23	0	2	13	10	0	0	1101	1	
4	149	0	0	0	0	0	0	0	0	2	1	7	0	1	0	1	0	1	12	10	0	0	0	0	8	0	0	0	0	1	0	0	0	0	7118	

0.8442918419837951

Kernel: (16, 2), Adadelta, cosine_proximity

7996	1	0	15	2	0	3	1	2	0	640	21	20	0	5	10	0	2	60	8	12	8	0	5	18	2	0	0	0	0	0	0	7	12	0	0	0	83
23	1868	0	18	23	3	6	7	2	0	0	16	32	0	1	18	0	8	26	72	4	176	0	5	19	8	2	9	0	0	4	43	5	0	6	3639		
0	0	0	0	0	0	0	0	0	0	1	0	436	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
26	3	0	6128	7	1	11	1	0	0	8	0	42	2	2	1	0	1	0	18	9	9	0	1	18	2	12	0	0	0	2	12	0	0	1	68		
27	15	0	22	5261	0	2	1	0	0	6	0	20	0	1	0	0	0	1	31	4	20	1	0	32	4	4	0	0	0	1	8	4	0	2	62		
15	24	0	3	3	3714	9	0	0	0	1	11	10	0	0	40	0	3	18	59	0	8	32	6	11	4	1	38	0	0	10	13	18	0	9	10		
18	22	0	90	4	0	1630	0	3	0	1	0	17	2	1	0	0	0	5	29	4	2	4	0	50	9	5	5	0	0	2	10	7	0	1	24		
13	0	0	1	1	0	3	1957	26	0	0	1	8	1	0	0	0	0	3	7	3	4	0	0	4	0	0	0	0	0	0	3	0	0	0	50		
11	21	0	34	2	0	18	3	733	0	0	0	16	0	4	0	0	0	5	43	13	6	1	0	24	2	1	5	0	0	1	15	4	0	4	21		
5	1	0	3	2	0	11	0	243	0	0	1	4	0	0	0	0	1	0	16	1	2	1	0	1	0	1	0	0	0	5	0	0	0	2	3		
1274	0	0	0	0	0	0	0	0	0	2898	3	5	1	1	0	0	0	4	1	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	10	
854	1	0	11	1	0	0	1	0	0	11	5941	4	0	1	16	0	4	682	6	2	4	0	6	11	0	0	0	0	0	3	3	2	0	0	0	21	
28	0	0	5	5	0	0	1	0	0	12	0	6866	0	2	0	0	0	0	42	10	4	0	4	10	0	23	0	0	0	5	12	0	0	0	50		
53	17	0	7	1	24	5	3	1	0	17	3	16	3376	0	9	0	1	11	393	4	2	0	4	5	5	1	44	0	0	3	15	2	0	2	24		
8	5	0	3	4	0	0	0	28	0	1	0	11	0	1267	1	0	3	0	132	1	1	0	0	3	0	5	0	0	0	2	1	0	0	1	16		
41	10	0	25	11	0	10	1	1	0	3	11	15	0	7	13112	0	13	2	87	3	14	0	23	18	6	0	875	0	0	9	13	3	0	4	8</		

Kernel: (16, 2), Adam, categorical_hinge

8170	2	0	9	3	0	0	2	2	1	375	14	20	0	4	22	0	1	178	3	23	7	0	12	4	5	0	0	0	0	12	24	0	0	0	40	
38	3284	0	18	31	6	4	15	2	3	3	16	64	0	9	35	0	14	112	81	125	956	1	8	9	50	1	37	0	0	43	202	11	1	18	846	
0	0	0	0	0	0	0	0	0	0	1	0	436	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
42	2	0	6045	5	0	7	1	1	0	5	0	42	4	4	0	0	2	1	10	60	14	0	3	4	13	16	3	0	0	8	49	0	0	6	38	
20	5	0	24	5291	0	0	1	0	0	5	0	16	0	1	0	0	1	1	15	9	48	1	0	6	20	7	2	0	0	5	17	3	0	9	22	
7	9	0	7	4	3664	1	1	0	2	1	9	17	3	0	46	0	3	31	25	2	17	51	7	3	8	2	59	0	0	16	35	25	0	13	2	
19	9	0	132	5	0	1610	0	3	0	0	0	16	2	3	0	0	1	10	12	19	4	6	0	2	24	7	6	0	0	5	28	5	0	4	13	
9	1	0	1	0	0	1	2026	2	2	0	0	2	0	0	0	0	1	0	8	3	0	0	0	0	0	0	0	0	0	11	0	0	0	18		
9	4	0	18	3	0	4	4	736	9	0	0	16	0	7	0	0	0	5	16	40	9	1	0	2	32	2	3	0	0	4	46	3	0	6	8	
4	1	0	0	3	0	1	0	13	259	0	1	3	0	0	0	0	0	0	3	2	0	0	0	1	1	1	0	0	0	7	0	0	1	2		
1689	0	0	0	0	0	0	0	0	0	2487	6	5	1	0	0	0	0	3	0	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	2	
781	1	0	6	2	0	0	2	0	0	6	4876	3	0	1	24	0	6	1822	3	7	4	0	13	7	0	0	0	0	7	10	0	0	0	4		
29	0	0	9	6	0	0	5	0	1	12	0	6869	0	1	0	0	0	0	17	16	5	0	4	3	0	28	0	0	24	29	0	0	0	21		
57	9	0	11	1	31	1	4	2	0	18	3	20	3609	0	10	0	1	15	91	21	3	0	6	0	27	1	44	0	0	4	42	3	4	6	4	
5	2	0	3	2	0	0	0	9	0	1	0	12	0	1415	2	0	2	1	7	4	2	0	0	1	0	5	0	0	6	4	0	0	1	9		
27	3	0	21	14	0	6	1	1	0	2	9	11	0	6	13016	0	16	4	20	24	15	0	33	5	9	0	1048	0	0	14	16	1	0	3	0	
5	0	0	1	0	0	0	2	0	0	1	0	509	0	1	0	0	1	0	4	0	0	2	1	0	3	0	0	0	5	4	0	0	0	4		
40	84	0	70	24	0	4	9	9	0	3	0	10	26	4	1	0	6676	3	140	39	51	0	2	9	9	19	33	0	0	24	111	0	0	3	80	
4379	15	0	93	29	1	6	17	6	5	21	250	114	4	24	96	0	22	8405	104	134	82	4	57	46	49	19	2	0	0	86	142	3	0	12	78	
87	17	0	66	25	0	5	11	1	4	7	3	256	8	35	4	0	16	29	10805	84	39	4	9	15	19	42	32	0	0	34	131	4	0	7	88	
30	4	0	30	9	0	3	6	0	1	5	2	55	4	5	2	0	7	6	18	6157	29	1	8	3	11	16	3	0	0	16	65	0	0	7	60	
19	127	0	19	29	0	1	14	1	2	2	15	40	0	4	10	0	29	26	25	78	6912	0	15	15	17	10	1	0	0	40	147	1	0	15	139	
64	72	0	146	5	67	17	4	32	1	40	0	34	73	4	16	0	1	58	242	37	25	7027	3	2	207	2	108	0	0	19	267	41	4	68	11	
3	0	0	1	2	0	0	2	1	0	36	0	2	0	0	0	0	7	2	0	8	1	0	4635	1	2	425	2	0	0	4	5	0	0	2	2	
14	0	0	136	6	0	1	3	5	0	1	0	20	2	1	0	0	0	0	8	14	9	0	3	4612	4	6	1	0	0	10	24	1	0	0	20	
85	40	0	33	26	2	8	7	4	1	4	3	78	3	1	9	0	4	35	103	102	35	4	7	8	3901	22	23	0	0	41	170	2	0	6	32	
0	0	0	2	1	0	1	3	0	0	16	0	7	1	1	0	0	1	1	0	7	1	1	200	1	0	5180	0	0	0	5	13	0	0	1	8	
10	25	0	89	18	0	5	3	3	0	33	0	28	22	8	49	0	6	9	57	109	15	2	12	7	23	18	11122	0	0	31	76	6	0	4	23	
0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0	136	0		
0	0	0	3	2	1	0	1	0	0	0	1	6	0	0	0	0	2	6	4	4	1	0	0	0	2	2	0	0	2	17	0	0	522	3		
49	13	0	11	14	0	3	24	17	1	9	1	78	3	16	1	0	4	10	11	164	33	2	8	5	27	48	13	0	0	7716	183	3	0	3	87	
39	6	0	24	7	0	0	13	1	1	5	0	64	2	2	3	0	3	11	21	51	35	1	3	13	21	11	3	0	0	31	8470	2	0	5	96	
8	4	0	6	2	0	1	0	0	0	1	0	7	3	0	3	0	0	5	7	12	3	0	0	3	0	5	13	0	0	3	27	2333	0	0	3	0
0	0	0	2	0	0	0	0	0	0	15	0	6	1224	0	0	0	0	7	2	0	1	0	0	2	0	2	0	0	0	1	0	519	0	0	0	
2	4	0	27	3	2	1	2	1	0	0	1	7	0	0	8	0	2	38	20	41	6	2	1	7	2	13	0	0	12	63	0	1	1039	1		
15	199	0	1	1	0	0	2	0	0	6	1	36	1	1	1	0	1	1	9	26	16	0	0	1	1	11	0	0	4	22	3	0	0	6956		

0.846635353565216

Kernel: (16, 2), Adamax, cosine_proximity

8277	1	0	3	3	0	7	0	0	0	272	24	16	0	3	16	0	6	174	3	14	12	0	9	3	2	3	0	0	0	24	16	0	0	0	45	
21	3286	0	21	19	3	10	9	2	3	1	18	34	4	0	29	2	16	109	26	23	996	1	6	9	28	1	7	1	13	43	117	4	0	1	1180	
0	0	0	0	0	0	0	0	0	0	1	0	434	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
48	8	0	6064	5	0	19	1	1	0	5	1	40	5	2	0	1	4	4	11	26	17	2	2	3	8	16	4	0	0	12	30	0	0	1	45	
30	11	0	18	5223	0	3	1	0	0	2	0	21	1	2	0	1	3	12	6	74	2	0	4	39	6	5	0	0	12	19	7	0	4	23		
7	20	0	3	4	3602	14	0	0	1	0	8	8	3	0	54	1	5	41	10	5	21	99	6	5	12	3	34	2	5	26	35	18	0	16	2	
18	6	0	17	4	0	1781	0	2	0	0	0	16	0	3	0	3	2	6	11	7	5	5	0	2	15	5	2	0	0	5	15	1	0	0	14	
16	1	0	3	0	0	15	1960	4	3	0	1	5	0	0	0	0	0	5	0	12	8	0	0	3	0	1	0	0	2	16	0	0	0	30		
8	7	0	22	2	0	11	4	785	1	0	0	10	2	3	0	4	0	10	6	24	12	2	0	2	14	2	2	0	1	11	28	2	0	3	9	
4	0	0	0	2	0	2	0	23	244	0	1	1	0	0	0	0	1	1	3	2	2	1	0	2	0	0	1	2	6	0	0	1	3	0		
1856	0	0	0	0	1	0	0	0	0	2313	8	5	2	0	0	0	9	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	3	0
729	2	0	6	2	0	1	0	0	0	4	5067	1	0	1	20	0	9	1694	1	4	9	0	8	6	1	0	0	0	2	7	7	1	0	0	3	0
32	2	0	6	5	0	0	1	0	0	10	0	6846	0	1	0	3	5	2	22	11	6	0	4	3	0	25	0	0	1	31	30	0	0	0	33	
55	23	0	5	0	26	2	3	1	0	9	1	18	3706	0	19	0	1	23	44	8	3	1	4	0	18	1	37	0	0	4	22	2	3	3	6	
5	5	0	1	0	0	0	0	12	0	1	0	9	0	1400	2	0	5	1	10	3	3	2	0	1	6	1	0	0	10	3	0	0	2	11		
18	6	0	6	10	0	16	1	1	0	1	14	2	0	4	13834	0	17	10	17	8	19	0	24	5	7	0	255	0	6	24	20	0	0	0	0	
7	0	0	3	0	0	0	2	0	0	0	0	500																								

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

8097	1	0	9	7	0	6	1	0	1	210	65	44	1	4	5	1	2	361	5	16	15	4	7	3	5	5	0	0	0	5	7	0	0	0	46	
16	2870	1	25	26	8	4	13	0	7	1	22	63	1	0	24	7	9	118	102	25	1795	1	8	8	42	0	23	0	8	20	62	4	0	1	729	
0	0	0	0	0	0	0	0	0	0	1	0	381	0	2	0	3	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
31	9	0	6115	10	0	13	1	1	0	2	0	42	3	2	0	10	3	3	12	22	15	4	2	9	13	13	4	0	0	5	10	0	0	31		
14	0	0	11	5354	0	3	1	0	0	2	0	18	0	2	0	1	0	4	31	6	25	2	0	4	19	1	7	0	0	2	2	1	0	19		
3	3	0	3	4	3782	2	0	0	0	0	8	5	2	0	40	0	2	32	25	6	21	48	7	5	5	1	28	0	4	10	5	16	0	2	1	
11	4	0	32	5	0	1765	0	0	0	0	0	22	0	2	0	8	1	6	20	11	5	6	0	2	17	3	4	0	0	4	4	0	0	13		
19	1	0	3	0	0	8	1985	1	0	0	1	15	1	0	0	0	2	2	13	6	1	0	0	1	1	0	0	1	2	0	0	0	0	22		
10	5	0	27	1	0	7	3	750	25	0	0	13	2	6	0	8	0	10	31	23	11	1	0	3	24	2	3	0	0	4	4	4	0	1	9	
4	0	0	0	3	0	1	0	5	273	0	1	2	0	0	0	0	1	0	2	1	0	1	1	1	0	1	0	0	1	2	0	0	0	3		
2276	0	1	0	0	4	0	0	0	0	1873	18	6	6	1	0	0	0	5	2	0	1	0	4	0	0	1	0	0	0	0	0	0	0	0	3	
437	0	0	14	3	0	4	0	0	0	2	5977	2	0	1	7	0	9	1093	2	3	6	1	3	7	1	0	0	0	1	2	5	1	1	0	3	
21	2	0	7	5	0	0	1	0	0	3	0	6933	0	2	0	0	1	2	31	11	0	1	3	5	0	18	0	0	0	15	7	0	0	11		
43	8	0	9	1	38	2	5	0	0	4	3	22	3735	0	8	0	1	19	65	13	3	2	2	0	19	2	26	0	0	4	8	2	2	0	2	
4	3	0	3	0	0	1	0	7	1	0	0	5	1	1397	0	3	1	1	40	3	3	0	0	1	1	5	1	0	0	5	1	0	0	6		
19	1	0	42	37	0	49	1	1	0	0	38	7	0	7	12248	0	17	39	110	11	23	0	23	9	22	1	1587	0	5	15	10	0	0	3	0	
2	0	0	0	0	0	0	0	0	0	0	0	522	0	0	0	3	0	0	11	0	0	0	0	1	0	2	0	0	0	1	0	0	0	2		
49	64	1	61	117	0	22	5	3	0	2	0	5	33	3	2	8	6572	14	157	35	52	7	0	15	27	32	60	0	0	16	35	0	0	0	86	
3359	9	4	87	27	1	34	14	1	8	9	542	142	6	10	15	14	19	9356	109	63	95	13	38	48	49	27	3	0	4	55	60	3	0	2	79	
53	38	0	33	13	0	4	5	2	4	2	5	231	21	14	1	77	15	36	11056	43	26	17	6	19	15	29	6	0	0	19	26	2	2	2	65	
31	6	0	46	10	0	5	5	2	2	6	3	89	7	2	1	12	3	31	67	5998	43	4	8	11	21	19	9	0	3	17	31	3	0	1	67	
11	87	0	18	22	0	5	10	2	4	1	20	41	0	2	3	2	18	13	21	16	7269	2	16	18	8	4	3	0	10	20	33	0	0	0	74	
71	51	0	67	17	95	57	2	2	2	26	1	27	59	3	8	4	1	76	455	27	31	7244	1	3	145	1	119	0	0	17	33	37	1	5	9	
20	0	0	7	7	0	0	1	0	0	9	13	3	1	0	1	0	7	4	5	4	4	0	4612	3	5	429	4	0	0	0	0	0	0	2	0	
11	0	0	14	5	0	4	3	1	0	0	0	11	0	1	0	1	0	2	7	2	2	1	4807	1	4	0	0	0	2	1	0	1	0	18		
73	27	0	29	31	2	21	6	2	3	0	6	83	8	0	4	22	2	40	132	61	46	21	3	12	4001	20	19	0	0	36	53	2	1	0	33	
0	0	0	1	8	0	1	0	0	0	7	0	21	2	1	0	0	1	5	3	5	1	134	1	2	5242	0	0	0	0	0	5	0	0	0	11	
39	24	1	86	27	0	63	0	2	0	10	0	36	56	11	17	1	7	23	222	36	16	10	12	14	24	14	10974	0	0	31	25	9	0	2	21	
0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	121	2	0	0	0	16	0		
0	2	0	3	0	0	0	1	0	0	0	1	3	0	0	0	0	2	6	2	1	5	0	1	1	0	1	0	438	2	6	0	0	102	1		
67	22	0	18	14	2	1	15	7	4	5	4	135	12	7	0	57	3	23	16	63	43	10	6	10	25	44	9	0	0	7781	65	3	0	0	86	
44	50	0	68	25	0	10	16	9	3	4	1	160	9	1	0	55	1	53	130	66	107	36	3	24	83	14	16	0	2	54	7770	5	0	3	122	
8	0	0	4	4	0	1	0	0	0	0	0	5	3	0	2	0	0	5	13	6	2	0	0	4	0	6	10	0	0	3	6	2365	0	0	2	0
0	0	0	2	0	1	0	0	0	0	9	0	2	943	0	0	0	0	9	0	0	0	5	0	1	1	0	0	0	0	1	0	807	0	0	0	
0	11	0	15	7	3	3	0	1	3	0	1	5	0	0	9	2	2	55	41	22	13	4	2	4	26	1	14	0	2	12	20	0	1	1027	1	
11	208	0	0	4	0	0	0	0	0	5	1	57	0	1	0	1	1	1	4	15	21	0	0	0	9	0	0	0	1	0	0	0	0	0	6975	

0.8549525320529938

Kernel: (16, 2), Nadam, categorical_hinge

7956	2	0	15	3	0	4	2	0	0	528	20	22	1	0	39	0	6	152	5	15	23	0	10	11	8	2	1	0	0	18	24	0	0	0	66	
19	1306	0	20	10	6	3	14	0	0	1	14	55	0	0	20	0	19	53	12	13	2916	1	6	10	7	3	13	0	10	17	86	2	0	4	1403	
0	0	0	0	0	0	0	0	0	0	1	0	436	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
25	9	0	6040	4	0	15	1	0	0	8	1	54	4	0	0	0	5	0	13	21	37	13	2	18	7	21	2	0	0	5	35	0	0	1	44	
19	4	0	15	4827	0	0	1	0	0	5	0	33	0	0	0	3	0	6	7	486	4	0	23	16	8	2	0	0	10	14	7	0	8	31		
10	4	0	2	3	3700	0	0	0	0	0	8	15	1	0	38	0	4	23	13	1	31	97	5	6	5	4	37	0	7	15	17	9	0	13	2	
15	16	0	29	3	0	1727	0	0	3	1	0	20	2	0	0	0	3	5	11	12	17	7	0	5	8	8	3	0	0	3	24	5	0	3	15	
10	0	0	4	0	0	1	2015	0	0	0	1	9	0	0	0	0	0	1	3	5	0	0	2	0	2	0	0	0	6	0	0	0	0	26		
11	45	0	49	1	0	13	5	0	327	0	0	21	0	0	0	0	0	9	10	25	78	20	0	19	26	4	4	0	0	6	91	1	0	213	9	
5	0	0	3	2	0	1	1	0	226	0	1	6	0	0	0	2	4	10	3	6	1	0	2	3	2	0	0	1	1	18	0	0	2	3		
1388	0	0	0	0	0	0	0	0	0	2775	4	6	2	0	0	0	6	1	0	2	0	10	0	0	0	0	0	0	0	0	0	0	0	0	7	0
844	0	0	7	1	0	1	2	0	0	5	4480	5	0	0	36	0	10	2110	6	2	21	0	15	11	0	0	0	0	3	12	5	2	0	0	7	0
21	0	0	4	0	0	0	2	0	0	18	0	6903	0	0	0	1	1	14	10	14	0	3	4	0	21	0	0	0	17	14	0	0	0	0	32	
43	20	0	23	1	29	3	6	0	0	17	1	23	3573	0	7	0	1	14	93	14	26	6	2	2	19	2	52	0	0	4	50	4	2	2	9	
18	19	0	4	3	1	1	0	0	718	2	0	40	1	0	2	7	4	128	4	18	19	0	2	8	9	6	0	1	14	4	0	0	448	12		
10	4	0	16	13	0	15	2	0	1	2	12	19	0	0	12606	0	21	0	35	6	36	0	26	12	9	0	1434	0	7	20	16	0	0	1	2	
2	0	0	2	0	0	0	1	0																												

Kernel: (16, 2), Nadam, logcosh

8062	3	0	10	3	0	13	2	0	0	316	25	37	0	5	5	0	5	276	3	14	29	3	13	2	3	5	0	0	0	23	14	0	0	0	62
10	2902	0	20	13	1	12	14	2	2	1	16	22	0	1	23	0	12	60	28	12	1342	0	7	8	9	1	13	4	0	33	67	1	0	1	1406
0	0	419	0	0	0	0	0	0	0	1	0	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
28	4	0	6122	5	0	31	1	1	0	3	1	43	1	1	0	0	2	3	7	15	17	0	1	3	0	25	3	1	0	7	14	0	0	0	46
27	10	0	18	5242	0	13	1	1	0	5	0	26	0	3	0	0	3	6	20	7	58	1	0	4	11	6	12	0	0	11	12	0	0	6	26
7	14	0	14	8	3384	34	2	4	1	0	7	6	3	5	54	0	5	58	19	9	36	187	5	6	7	4	84	7	0	37	25	28	0	7	3
12	3	0	18	2	0	1828	0	1	0	0	0	18	0	3	0	0	1	3	8	8	5	3	0	2	1	4	4	0	0	5	4	0	0	12	
9	0	0	3	0	0	19	2013	1	0	0	0	2	0	0	0	0	1	1	6	4	0	0	0	0	1	0	0	0	1	2	0	0	0	22	
8	2	0	23	1	0	30	3	819	1	0	0	15	0	8	0	0	0	7	2	18	11	1	0	2	1	3	2	1	0	4	7	4	0	3	11
4	0	0	1	2	0	8	1	37	230	0	1	1	0	0	0	0	1	0	2	2	1	1	0	1	0	2	0	0	2	4	0	0	0	2	
1893	0	0	0	2	0	0	0	0	0	2262	7	5	3	1	0	0	0	12	0	0	3	0	7	0	0	1	0	0	0	0	0	0	0	5	
585	2	0	13	2	0	2	3	0	0	2	5599	3	0	1	11	0	12	1292	3	1	15	0	16	5	0	0	0	0	4	6	2	0	0	6	
21	1	0	4	4	0	0	2	0	0	7	0	6874	0	3	0	0	2	3	27	10	8	0	3	2	0	25	0	0	0	29	21	0	0	33	
44	31	0	16	0	26	11	6	1	0	10	2	16	3584	0	10	0	2	26	124	16	10	0	2	0	5	2	49	0	0	6	30	7	3	6	
2	1	0	2	0	0	1	0	9	0	0	0	4	0	1438	0	0	3	1	5	3	1	0	0	1	0	0	0	0	6	1	0	0	0	9	
16	5	0	14	13	0	26	1	2	0	0	22	1	0	12	13255	0	21	22	19	5	26	0	25	5	4	1	790	4	0	24	12	0	0	0	
4	0	0	1	0	0	0	2	0	0	0	0	514	0	1	0	0	1	0	6	0	0	0	1	1	0	3	0	0	4	2	0	0	0	4	
16	18	2	53	7	0	27	7	0	0	5	0	3	4	3	1	0	7017	1	85	12	32	0	0	6	1	22	17	0	0	4	44	0	0	96	
3794	36	0	108	16	0	78	18	3	1	14	459	108	7	28	39	0	31	8844	88	47	128	6	44	33	16	40	1	7	0	95	96	4	0	3	113
56	73	0	48	10	0	20	10	7	3	4	3	318	11	44	3	0	20	32	10802	45	51	8	5	14	2	42	20	7	0	51	84	8	0	0	86
36	8	0	64	7	0	39	8	2	1	4	2	95	3	8	3	0	9	53	32	5862	50	2	9	6	7	23	14	5	0	40	61	3	0	6	101
4	86	0	17	10	0	9	12	1	2	1	13	17	0	2	5	0	26	6	3	9	7231	0	13	10	2	10	0	5	0	28	42	0	0	1	188
69	87	2	124	14	59	310	4	59	0	33	2	25	67	15	16	0	1	102	328	30	51	6454	3	2	81	4	407	3	0	54	145	54	1	55	36
5	0	0	1	1	0	0	1	0	0	6	4	1	0	0	2	0	7	3	2	0	4	0	4244	1	0	847	5	0	0	2	3	0	0	2	
16	1	0	58	5	0	23	4	7	0	0	0	19	4	1	0	0	0	5	7	5	16	1	0	4670	0	7	2	0	0	12	11	2	0	25	
64	94	0	59	22	2	113	8	14	2	2	5	82	9	7	11	0	9	87	112	70	66	15	6	9	3549	29	50	3	0	70	156	2	1	4	67
0	0	0	0	0	0	3	0	0	0	0	0	11	0	0	0	0	0	1	1	4	0	0	63	1	0	5348	0	0	0	2	6	0	0	11	
34	45	0	44	13	0	76	2	2	0	15	0	30	15	10	21	0	8	14	60	26	16	8	9	6	6	17	11201	3	0	54	39	11	0	1	27
0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	118	0	2	1	0	0	17	0
0	3	0	3	0	0	0	1	0	0	0	1	2	0	0	0	0	3	6	2	1	3	0	0	0	0	2	1	418	0	2	7	0	0	122	2
29	7	0	11	5	0	7	17	7	1	2	1	56	2	9	1	0	5	10	3	36	30	0	7	3	1	53	3	0	0	8082	63	1	1	0	104
30	29	0	45	10	0	47	18	7	0	3	1	95	3	2	0	0	3	33	24	32	82	8	3	15	8	16	9	7	0	62	8185	4	0	2	161
9	1	0	4	2	0	6	0	0	0	0	0	7	1	0	2	0	0	3	4	5	2	0	1	4	0	6	11	0	0	4	7	2368	0	0	2
0	0	4	2	0	0	1	0	0	0	11	0	0	0	0	0	0	0	10	2	0	1	0	0	1	0	0	0	0	0	1	0	563	0	1	
0	12	0	28	4	0	14	0	0	0	0	2	5	0	3	9	0	4	45	26	16	13	0	2	2	2	2	19	2	0	14	26	0	0	1055	2
2	61	0	2	0	0	0	0	0	0	4	1	19	0	0	0	0	1	0	0	11	12	0	0	0	0	12	0	0	0	1	1	0	0	0	7188

0.8517001867294312

ARat_2017_05_19_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

5339	18	37	1	8	32	5	2	3	0	12	17	19	0	0	10	2	0	0	0	24	3	6	0	1	0	5	6	14	6	25	15	91	8	7	8
37	13938	20	0	22	28	9	1	11	3	18	9	32	0	0	0	8	1	0	0	58	5	18	0	0	0	3	8	25	13	31	13	103	17	20	3
22	46	5455	7	7	43	16	0	4	0	24	12	33	0	2	20	17	0	1	0	49	4	13	2	2	0	7	25	10	12	37	5	164	26	10	30
20	20	38	2020	8	40	3	2	16	0	32	15	20	0	2	6	13	0	0	0	23	4	2	6	2	0	2	4	8	12	24	8	101	17	11	2
15	26	13	1	5725	21	11	2	13	0	6	239	29	0	0	7	5	0	0	0	37	8	3	1	1	0	3	16	24	7	15	5	43	21	7	3
17	40	21	23	3	7575	15	0	2	0	11	7	28	0	0	3	9	0	0	0	25	2	11	1	2	0	0	8	12	7	21	17	100	9	6	6
30	85	51	10	14	56	11587	1	13	0	16	13	60	0	1	23	7	0	2	0	56	0	18	0	7	0	2	15	38	24	49	12	176	40	36	10
3	2	3	1	1	5	4	1140	0	0	0	1	5	0	0	6	0	3	1	0	0	4	1	2	0	0	0	0	0	1	1	2	19	1	2	1
16	52	11	1	4	20	6	0	5445	0	2	7	43	0	0	1	3	0	0	0	28	5	1	0	1	0	1	7	8	8	9	35	46	1	11	2
19	4178	13	4	16	21	25	2	14	1368	48	20	5	0	5	1	19	10	0	0	4	10	10	0	4	0	0	15	9	6	35	19	27	11	25	1
81	127	186	16	18	88	81	14	17	1	8387	38	59	0	6	34	69	5	0	0	83	15	33	3	17	0	20	39	47	57	220	35	483	64	37	21
12	8	3	0	73	5	4	0	1	0	4	1192	5	0	0	2	7	0	2	0	8	4	4	0	1	0	0	0	9	2	5	4	15	2	0	5
2	7	2	1	0	2	1	0	2	0	3	0	2839	0	0	4	0	0	0	0	11	0	3	0	1	0	0	0	5	1	3	3	12	22	1	0
22	5	23	3	17	3	3	0	7	0	21	0	5	0	0	47	8	0	23	0	0	97	0	28	31	0	45	71	3	5	67	0	15	33	1	0
10	2	3	0	0	7	2	0	5	2	21	0	1	0	0	2301	0	4	1	0	0	1	0	0	3	0	0	2	2	2	127	3	8	2	3	0
4	14	10	3	2	5	7	1	3	0	6	5	79	0	0	829	5	0	0	4	6	0	0	2	0	2	2	8	3	3	0	24	30	8</		

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

5387	21	26	2	6	26	14	2	1	0	7	11	18	0	0	9	1	0	0	3	28	3	4	1	0	0	5	9	14	6	27	4	55	15	12	7	
34	14002	11	0	23	21	11	1	9	3	5	8	36	2	0	0	5	0	0	3	50	3	14	0	0	0	6	8	23	11	30	7	64	32	29	3	
28	70	5198	7	12	43	41	1	4	1	19	10	38	1	2	36	23	0	2	6	48	3	13	7	2	0	16	45	24	16	75	0	188	67	32	27	
25	30	32	1988	8	27	13	2	16	0	17	4	20	0	1	12	11	0	0	0	24	2	1	6	2	0	8	4	10	12	33	6	79	69	17	2	
15	26	7	0	5762	20	16	1	13	0	3	195	30	0	0	13	6	0	0	2	37	5	2	1	1	0	4	17	28	6	17	2	35	33	8	2	
29	54	22	61	3	7409	41	0	5	0	4	3	35	3	0	6	14	0	1	4	33	1	9	4	4	0	1	19	21	9	32	11	92	27	19	5	
23	69	21	6	5	37	11837	1	12	0	3	4	49	1	0	24	1	0	5	2	42	0	10	0	0	0	5	8	31	19	45	5	93	48	37	9	
4	4	2	0	1	4	4	1145	0	0	0	1	4	0	4	2	1	0	0	2	5	1	2	0	0	0	1	0	1	1	2	1	11	3	2	1	
18	56	7	3	3	13	9	0	5419	0	0	8	66	0	0	3	1	0	0	1	35	2	1	0	0	0	2	8	16	4	12	25	32	13	15	2	
18	4023	7	6	16	16	34	2	13	1571	25	17	4	0	2	2	14	3	0	1	7	3	8	0	1	0	3	17	10	5	32	15	13	27	28	1	
93	245	168	23	22	92	160	21	18	1	7521	34	59	6	5	83	97	2	11	7	91	11	32	14	12	0	42	123	62	74	375	15	453	302	107	20	
15	14	3	0	74	6	5	0	1	0	1	1164	6	0	0	5	8	0	1	1	9	0	4	0	1	0	3	3	10	2	7	4	15	9	1	5	
2	3	0	1	0	2	3	0	1	0	0	0	2856	1	0	6	0	0	0	5	6	0	1	0	0	0	0	0	4	1	3	1	4	24	1	0	
2	0	1	0	0	1	2	0	0	0	0	0	556	0	0	0	2	0	0	0	0	0	1	0	0	0	1	3	1	1	0	0	3	6	0	0	
13	3	2	2	1	7	4	2	6	5	18	0	0	0	0	2151	1	4	1	0	0	0	0	0	2	0	0	6	2	2	261	3	5	8	3	0	
4	15	7	3	1	4	9	0	2	0	2	4	41	1	0	854	4	0	0	3	4	4	0	0	1	0	2	2	9	1	6	0	21	53	8	0	
35	128	36	10	32	45	76	2	19	1	25	28	46	3	6	64	7645	0	2	3	211	3	17	5	15	0	21	59	64	13	69	28	117	76	55	4	
4	2	0	0	0	2	7	0	2	0	4	1	0	0	2	0	1	265	0	0	2	0	0	477	0	0	0	0	1	0	3	8	3	1	1	2	0
6	3	2	8	7	20	17	4	0	15	98	2	1	0	29	3	13	3	2002	0	0	0	0	2	0	0	0	4	2	2	17	1	1	31	3	2	
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	273	0	0	0	0	0	0	1	0	0	0	1	0	2	0	0	
7	10	5	0	4	8	12	0	1	0	2	5	24	0	0	1	32	0	1	1	6131	0	2	0	0	0	0	1	8	2	4	102	15	3	18	4	
10	25	5	2	7	8	6	4	1	1	5	6	0	0	0	4	6	0	0	0	8	817	7	1	4	0	5	15	5	7	5	0	19	24	6	0	
1	2	1	0	0	6	2	0	1	0	0	3	8	0	0	1	0	3	1	1	3	0	2359	0	0	0	0	0	0	0	1	2	2	3	1	0	
2	5	2	1	1	0	2	0	2	0	5	0	2	0	0	4	1	0	0	0	4	0	1	264	0	0	3	3	2	0	4	0	4	7	0	0	
66	20	16	5	10	11	32	1	12	7	13	16	21	8	1	31	41	2	1	7	1	4	6	2	3004	0	5	26	15	6	89	1	67	93	16	2	
0	0	1	0	0	0	4	0	1	0	1	0	322	0	0	49	1	0	0	12	1	1	0	0	0	0	0	1	0	0	1	2	0	4	0	0	
2	5	1	0	0	3	0	1	0	0	0	5	0	0	0	4	0	0	0	1	6	0	0	0	0	0	674	10	6	3	9	0	24	3	0	2	
8	63	21	5	12	13	45	2	19	2	13	14	22	1	0	36	52	0	2	5	30	2	3	4	2	0	17	4518	41	1	57	3	24	27	9	0	
15	29	13	1	11	14	10	3	5	0	1	3	8	1	0	4	2	0	0	2	23	5	10	0	3	0	0	1	7946	4	8	1	31	13	6	2	
6	15	6	1	0	13	3	5	6	1	0	0	9	0	0	2	2	0	0	6	13	0	7	0	0	0	1	4	5	2450	12	0	23	4	3	1	
23	23	15	0	6	33	12	9	2	0	4	3	26	2	4	4	3	0	2	2	31	0	6	0	0	0	7	10	16	12	8411	7	131	20	15	8	
2	7	1	0	0	6	2	0	0	0	0	8	0	0	0	35	0	0	0	0	1672	0	0	1	0	0	1	1	3	0	0	1761	2	1	9	1	
44	68	32	1	9	42	18	6	12	0	9	2	39	1	0	8	2	1	1	6	55	1	19	0	0	0	27	17	31	17	74	2	14721	32	24	10	
28	34	105	8	4	22	33	4	5	0	12	11	43	1	0	63	16	0	0	4	22	6	8	1	7	0	3	15	14	11	51	8	76	2701	11	2	
21	42	6	2	14	18	10	1	5	0	5	3	25	2	0	6	10	0	0	4	31	0	3	0	2	0	6	8	23	4	15	29	47	31	9470	4	
0	2	4	0	1	1	0	0	0	0	2	0	0	0	0	0	0	0	0	1	3	0	1	0	0	0	0	0	1	0	0	9	0	1	987		

0.8776392536789271

Kernel: (4, 2), Adamax, cosine_proximity

5156	32	70	8	9	30	10	3	7	0	32	16	29	0	0	15	4	0	0	2	37	5	6	3	5	6	9	16	25	5	45	3	80	23	26	7
22	13816	25	1	25	25	7	1	19	29	48	6	35	7	0	0	10	0	1	2	69	5	13	1	0	13	8	14	33	11	34	3	72	46	48	5
13	36	5529	8	3	28	10	0	6	0	42	4	30	2	0	23	18	0	1	3	45	2	6	8	1	7	17	27	17	4	40	0	97	39	15	24
14	21	30	2135	3	15	1	0	12	0	47	3	19	1	0	2	13	0	0	0	23	2	1	7	3	4	7	5	9	6	13	3	38	23	19	2
13	25	22	3	5642	20	8	2	18	0	22	225	31	2	0	15	19	0	0	0	38	10	2	10	0	6	12	21	33	6	18	2	31	40	8	3
18	51	36	167	1	7331	10	0	5	0	44	3	29	2	0	0	10	0	1	1	34	1	9	8	5	10	5	15	19	4	26	4	75	32	20	5
27	100	79	20	22	57	11258	5	21	0	108	11	52	5	1	28	25	0	5	1	66	5	14	2	34	14	9	30	50	19	66	2	134	102	71	9
2	3	6	1	1	4	4	1131	0	0	1	0	4	0	4	3	5	0	0	0	6	2	2	0	0	3	2	0	1	0	2	0	13	5	3	1
7	24	9	5	3	13	3	0	5501	0	10	0	41	2	0	0	1	0	0	1	49	1	1	0	4	0	2	7	17	4	10	11	25	9	12	2
13	3049	13	11	11	17	20	3	18	2428	112	15	6	0	4	1	25	2	0	0	8	8	6	1	4	2	4	23	10	5	31	15	12	33	33	1
41	86	155	19	9	63	40	11	16	1	9000	12	40	6	1	29	46	0	3	2	81	9	16	8	15	9	34	26	43	22	184	9	224	70	52	19
12	12	8	3	79	6	2	1	3	1	9	1123	5	0	0	5	11	0	1	1	19	2	4	3	3	0	5	5	14	1	7	2	12	11	2	5
0	3	1	1	0	2	1	0	3	0	4	0	2846	1	0	12	0	0	2	5	0	1	0	1	0	0	0	2	4	1	4	4	25	1	0	0
2	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	2	0	0	0	0	1	2	1	1	0	0	2	4	0	0
10	3	3	10	1	13	1	2	7	3	70	0	0	0	0	0	16	1	2	0	0	1	0	0	5	1	0	6	3	1	579	3	8	17	6	0
2	12	12	6	1	4	4	0	3	0	11	1	44	1	0	855	5	0	0	0	4	5	0	0	2	1										

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

5250	7	39	2	16	42	8	4	5	0	10	8	15	0	0	9	2	0	0	1	29	6	3	0	4	4	10	14	21	5	22	2	134	20	26	6	
45	13464	37	8	45	39	23	2	26	54	36	10	38	9	0	5	16	1	1	3	65	15	15	2	0	9	18	31	39	12	35	1	169	80	96	5	
16	23	5439	6	9	50	14	0	4	0	15	3	27	1	1	16	13	0	2	2	38	9	6	5	3	5	20	25	16	9	32	0	210	42	23	21	
15	12	31	2054	10	47	3	2	14	0	12	4	14	0	0	6	13	0	0	0	20	5	1	5	6	3	8	4	10	6	13	5	108	26	22	2	
12	7	9	1	5913	19	11	1	9	0	3	97	19	1	0	12	6	0	0	0	33	10	2	4	0	2	6	16	20	4	11	2	38	26	10	3	
9	24	22	61	7	7530	9	0	4	0	6	1	26	4	0	4	12	0	0	0	21	1	6	3	9	8	2	16	18	5	13	5	104	22	24	5	
28	47	57	14	14	62	11525	4	18	0	12	4	43	4	0	24	6	0	5	1	45	5	7	0	17	13	7	22	33	18	41	2	207	94	63	10	
1	1	3	0	1	5	4	1154	0	0	0	0	4	0	3	0	1	0	0	0	3	1	1	0	0	2	0	0	0	0	1	14	4	5	1		
9	9	3	3	6	16	3	0	5480	0	0	0	7	39	3	0	1	6	0	0	1	39	5	1	0	3	0	3	8	17	4	8	16	41	13	23	1
18	1962	23	13	26	29	37	3	20	3372	57	19	4	0	2	0	22	4	1	0	7	51	7	1	5	2	8	30	11	5	28	15	46	47	68	1	
76	66	230	30	28	128	83	24	17	4	7676	28	38	8	3	56	98	0	13	2	84	81	21	9	28	6	79	103	53	31	213	9	761	197	102	16	
13	4	7	0	124	6	2	1	2	0	1	1119	3	0	0	4	9	0	2	1	12	7	2	1	3	0	3	2	8	1	6	2	20	5	2	5	
3	0	3	1	1	2	3	1	2	0	2	0	2820	1	0	13	0	0	0	2	5	0	1	0	0	0	0	2	7	1	5	1	17	31	1	0	
1	0	0	0	1	0	0	0	0	0	0	0	568	0	0	0	2	0	0	0	0	1	0	0	0	0	0	1	2	1	1	0	0	3	2	0	0
12	0	3	6	1	13	4	4	6	5	31	0	0	0	0	2102	1	9	1	0	0	1	0	0	5	0	0	9	4	1	255	3	21	7	8	0	
4	10	12	7	2	5	6	1	3	0	4	4	23	1	0	840	8	0	1	0	3	8	0	0	4	1	3	4	7	1	3	0	25	64	11	0	
22	68	57	14	32	53	58	1	19	8	32	23	31	3	4	43	7836	0	3	2	83	11	11	2	52	5	30	44	51	7	36	26	181	49	63	3	
4	2	0	1	0	3	6	0	2	0	5	3	1	0	2	0	2	358	0	0	2	2	2	370	1	0	0	2	0	3	8	3	2	3	3	0	0
5	1	11	11	8	28	7	4	0	12	81	3	0	0	27	3	10	1	2047	0	0	4	0	0	2	0	0	3	2	1	6	1	2	13	3	2	
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	272	1	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	
5	8	8	0	9	16	9	0	2	0	4	2	24	0	0	0	98	0	1	0	6046	0	2	0	0	4	1	2	14	2	1	57	40	3	42	3	
5	9	6	1	3	7	3	3	1	0	2	1	0	0	0	0	4	0	0	0	4	905	3	1	3	0	3	5	3	3	3	0	20	11	4	0	
3	2	2	0	3	9	4	0	1	0	2	4	8	0	0	1	0	3	2	1	5	2	2320	0	1	0	0	1	2	0	3	0	15	6	1	0	
0	4	4	1	1	0	1	0	1	0	2	0	1	0	0	0	0	0	0	2	0	1	279	0	1	4	1	0	0	4	0	9	3	0	0		
13	9	18	1	8	11	7	1	5	5	5	4	12	4	0	6	20	2	0	0	0	7	1	0	3361	10	3	14	11	1	15	1	70	21	9	3	
0	0	1	0	1	0	4	0	2	0	1	0	242	0	0	49	0	0	0	0	1	1	0	0	2	88	0	1	0	0	1	2	0	4	0	1	
1	4	3	0	0	3	0	1	0	0	0	0	4	0	0	1	0	0	0	2	6	1	0	0	0	682	9	3	2	4	0	28	3	0	2		
4	37	93	6	19	19	18	3	15	3	13	18	18	1	0	21	74	0	4	0	25	8	3	2	5	1	36	4516	27	0	10	3	31	27	13	0	
7	13	17	2	17	20	8	4	4	0	1	0	4	3	0	5	5	0	1	0	19	12	6	0	6	2	8	3	7914	3	6	2	52	17	13	1	
6	13	8	1	0	18	2	6	6	0	3	0	9	0	0	1	4	0	0	0	12	2	6	0	1	8	4	6	7	2391	12	0	58	6	6	2	
24	16	43	0	14	52	11	12	3	0	14	1	18	3	5	9	14	0	2	1	36	7	4	0	3	1	25	18	20	14	8178	3	237	25	25	9	
2	4	0	0	0	8	2	0	0	0	0	0	8	0	0	0	78	0	0	0	2039	0	0	0	1	2	3	0	3	0	0	1332	9	1	20	1	
13	15	34	1	11	41	2	3	12	0	7	1	26	1	0	4	1	1	2	4	25	1	10	0	0	4	25	8	17	1	30	1	14977	22	22	9	
18	10	184	10	11	29	21	5	6	0	17	11	22	4	0	42	31	0	1	1	20	11	4	1	12	3	8	20	15	6	33	4	136	2613	16	4	
13	9	9	2	11	20	2	1	2	0	5	3	11	2	0	6	7	0	0	2	19	4	2	0	3	6	5	7	10	4	8	13	77	20	9561	3	
0	0	5	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	3	0	0	0	0	1	0	0	1	0	0	0	0	12	0	2	986

0.8877900393154117

Kernel: (16, 2), RMSprop, categorical_crossentropy

5318	27	56	2	13	20	4	2	1	0	10	8	11	0	0	8	25	0	0	2	14	6	8	2	2	0	4	7	18	5	30	3	95	9	8	6
24	14092	38	0	16	14	2	1	6	2	7	3	15	2	0	3	27	0	0	0	19	4	16	0	0	1	4	8	17	8	22	0	72	15	13	3
13	29	5744	1	4	10	5	0	0	0	8	1	15	1	0	4	48	0	0	1	21	1	7	2	1	0	4	15	7	8	31	0	87	14	5	18
23	28	91	1973	12	13	7	0	9	0	21	5	10	1	1	7	68	0	0	0	16	3	4	6	3	0	3	6	10	9	26	4	87	19	14	2
11	41	28	1	5870	18	7	1	4	0	2	70	18	1	0	13	63	0	0	0	27	10	3	2	0	0	1	15	19	5	14	2	33	18	8	2
37	74	117	139	8	6942	22	0	2	0	11	6	25	3	0	3	167	0	0	0	38	0	15	0	13	3	0	36	29	13	62	7	161	21	20	7
38	152	153	15	23	38	11302	0	8	0	15	12	41	4	1	20	100	0	0	1	53	1	21	1	22	1	2	22	45	21	63	4	166	49	48	10
5	3	13	1	1	5	4	1125	0	0	0	1	3	0	4	0	9	0	0	0	2	1	2	0	1	0	0	0	0	1	3	1	17	3	3	1
21	71	50	6	6	11	9	0	5311	0	3	6	37	2	0	1	49	0	0	1	32	3	5	0	3	0	0	14	21	5	17	9	54	6	20	1
13	4558	29	3	13	7	9	0	5	1103	25	11	1	0	3	0	36	2	0	0	4	5	8	0	6	1	0	8	10	3	29	12	15	7	18	0
83	183	805	27	23	69	75	15	13	1	7013	22	38	9	4	64	671	1	0	2	84	17	45	17	32	1	23	57	60	36	285	8	477	77	51	13
13	21	25	0	126	4	3	0	1	0	2	1062	3	0	0	4	39	0	1	1	11	3	6	0	3	0	1	1	13	1	9	2	17	1	0	4
3	9	9	1	0	2	2	0	1	0	2	0	2804	1	0	11	2	1	0	0	9	0	3	0	1	0	0	1	7	0	5	1	27	21	2	0
0	2	2	1	1	1	0	0	0	0	0	0	560	0	0	3	0	0	0	0	0	2	0	0	0	0	1	6	1	1	0	0	2	0	0	0
9	3	10	3	1	2	3	0	4	4	18	0	0	0	0	0	30	1	0	0	0	1	0	0	4	0	0	5	2	1	421	2	11	4	3	0
4	23	70	4	2	4	7	0	3	0	3	1	26	1	0	714	87	0	0	0	6	5	0	0	6	0										

Kernel: (16, 2), Adadelta, cosine_proximity

5096	51	27	4	35	44	21	7	9	0	9	13	35	2	0	5	14	0	0	0	41	11	7	2	1	0	22	27	33	4	46	8	103	14	25	8	
14	13876	7	0	29	24	10	2	21	27	9	5	46	5	0	1	37	0	0	0	74	5	12	0	1	0	11	17	38	9	32	10	62	24	41	5	
21	85	4958	10	22	62	34	9	11	3	32	17	43	1	2	27	81	0	2	0	65	18	15	7	7	0	57	81	43	12	76	3	204	33	24	40	
16	24	16	2066	10	38	4	1	17	1	13	4	21	0	1	3	23	0	0	0	19	3	1	8	3	0	18	4	20	3	25	6	73	19	20	1	
8	19	5	3	5855	17	10	2	11	0	2	137	34	1	0	5	16	1	0	0	34	13	1	4	1	0	5	15	32	6	16	2	25	18	7	2	
15	47	7	198	4	7352	17	1	10	0	7	3	33	1	0	3	17	0	0	0	32	2	8	5	5	0	8	14	33	5	32	8	88	13	11	2	
18	75	16	14	20	52	11692	1	18	1	6	7	71	1	2	13	21	1	5	0	59	0	14	0	5	0	8	20	46	16	44	5	107	35	49	10	
2	2	2	1	2	4	5	1143	0	0	0	0	5	0	5	2	3	0	0	0	5	1	2	0	0	0	2	0	1	0	1	2	12	4	2	1	
4	28	4	1	3	18	5	0	5468	0	0	1	31	0	0	1	10	0	0	0	41	4	1	0	4	0	1	12	44	3	9	35	32	2	11	1	
11	3517	6	8	17	17	22	2	16	2065	23	12	8	0	4	1	28	6	0	0	8	18	9	0	4	0	6	14	14	5	28	18	24	4	28	1	
60	172	103	37	44	98	95	20	22	6	7467	34	46	7	6	47	332	9	3	0	105	104	32	16	31	0	117	176	156	36	358	19	478	73	73	19	
11	14	2	1	127	5	2	1	5	1	2	1085	6	0	0	4	17	0	1	0	14	4	3	0	4	0	9	5	17	1	8	4	14	3	2	5	
1	6	1	1	1	2	3	0	10	0	3	0	2797	1	0	24	2	1	0	0	13	0	1	0	0	0	1	2	9	0	6	3	12	24	1	0	
1	1	0	1	7	1	1	0	2	0	0	0	1	545	0	0	3	0	0	0	0	3	0	0	0	0	2	4	2	1	1	0	3	4	0	0	
10	2	0	3	2	10	3	7	6	3	17	0	1	0	2012	0	13	1	1	0	0	3	0	0	2	0	0	8	5	0	384	3	7	3	6	0	
3	15	9	7	1	5	7	0	4	0	3	2	35	1	0	830	23	0	1	0	4	4	0	1	4	0	4	7	13	1	5	0	22	42	11	1	
20	84	16	11	37	44	48	5	28	1	13	13	49	3	4	24	7915	0	0	0	123	6	11	1	33	0	47	57	100	3	40	44	108	29	43	3	
3	2	0	0	0	2	3	1	2	0	1	2	1	0	1	0	0	358	0	0	3	0	0	389	0	0	0	1	1	4	6	4	1	1	2	0	0
4	2	2	9	5	15	6	6	0	16	94	7	1	0	22	0	23	7	2032	0	0	1	0	0	2	0	0	4	8	1	17	1	9	1	2	1	
1	2	0	0	1	2	0	0	2	0	0	0	1	0	0	209	0	0	0	0	0	0	1	0	0	0	11	41	0	0	0	1	4	3	0	0	
1	8	2	0	4	14	3	0	3	0	1	2	12	0	0	1	104	0	1	0	6059	0	2	0	0	0	0	1	11	1	2	140	8	1	21	1	
5	19	4	2	2	7	4	2	3	2	1	3	0	0	2	15	0	0	0	7	867	5	0	4	0	6	8	5	3	3	0	22	8	4	0	0	
2	9	0	0	3	7	3	0	1	0	0	2	8	0	0	2	0	4	1	0	5	0	2336	0	1	0	0	4	3	0	3	1	3	2	0	1	
0	5	1	1	3	0	1	1	1	0	4	0	3	0	0	0	0	0	0	3	0	1	274	0	0	9	1	2	0	4	0	3	2	0	0	0	
14	11	9	4	12	10	12	1	7	11	3	3	25	25	0	8	43	2	0	0	1	8	3	1	3295	0	6	19	15	1	40	3	46	11	6	3	
0	1	0	0	0	0	2	0	2	0	1	0	352	0	0	33	0	0	0	0	1	0	0	0	2	0	0	1	0	0	1	2	0	3	0	0	
0	4	0	0	1	2	0	1	0	0	0	4	0	0	0	2	0	0	0	0	7	2	0	1	0	0	688	10	7	2	5	0	19	2	0	2	
3	49	16	4	21	13	13	3	27	4	14	11	16	0	0	23	65	0	0	0	25	4	5	2	2	0	53	4572	67	0	16	5	21	10	9	0	
0	18	8	1	11	17	3	3	6	0	1	1	11	1	0	6	5	0	0	0	34	6	6	0	3	0	6	4	7971	1	4	1	24	8	13	2	
5	18	3	1	0	17	3	6	6	0	8	0	18	0	0	3	6	1	0	0	18	0	11	0	1	0	5	11	15	2378	23	0	30	4	6	1	
13	22	13	1	11	45	14	10	4	0	5	1	23	3	3	3	9	0	2	0	36	4	5	0	0	0	26	15	26	7	8349	10	153	12	12	10	
0	3	0	0	0	7	2	0	0	0	0	6	0	0	0	64	0	0	0	0	1643	0	0	0	0	0	3	0	1	0	0	1778	0	0	5	1	
21	87	42	4	28	50	22	9	15	1	18	7	52	1	0	7	13	2	3	0	69	10	22	1	1	0	66	23	59	16	71	6	14546	27	16	16	
23	58	79	28	33	28	32	13	15	2	27	27	43	6	2	65	110	0	0	0	22	29	11	7	21	0	11	57	32	10	140	8	149	2220	18	3	
12	46	4	2	17	30	7	2	17	1	10	2	42	3	0	11	50	0	1	0	40	5	6	0	6	0	6	15	24	6	21	45	87	35	9290	4	
0	3	2	0	1	2	0	0	0	0	1	0	2	0	0	0	0	0	0	0	7	0	1	0	0	0	0	1	0	0	0	0	6	0	2	985	

0.8718412970405099

Kernel: (16, 2), Adam, categorical_hinge

5026	40	99	2	9	31	23	4	6	0	18	16	22	0	0	9	13	1	0	0	28	10	11	7	4	0	11	42	14	6	47	9	168	23	15	10
20	13858	20	0	23	29	17	1	21	21	21	6	42	5	0	0	24	1	0	0	59	4	13	2	1	0	12	20	19	10	29	11	108	34	21	2
15	51	5366	2	5	38	28	1	7	1	26	1	35	1	2	18	19	0	1	0	53	6	7	7	2	0	15	50	15	12	36	1	200	32	11	41
17	29	51	1743	6	40	10	3	22	0	78	5	22	1	1	9	25	0	0	0	22	10	5	24	7	0	15	9	14	12	51	7	165	62	15	1
14	27	23	0	5561	24	25	2	13	0	12	230	28	2	0	16	14	1	0	0	37	26	2	8	8	0	12	41	40	7	29	2	59	34	6	4
20	45	32	54	2	7359	27	2	9	0	20	1	30	2	0	3	13	1	1	0	25	3	12	9	9	0	6	28	20	7	47	8	150	20	13	3
17	59	34	5	3	40	11853	1	15	0	8	1	48	1	0	13	7	0	3	0	44	0	7	0	7	0	5	21	22	12	30	5	144	15	29	3
3	2	4	0	1	4	5	1124	0	0	3	0	5	0	5	1	3	0	1	0	5	1	2	0	1	0	2	1	0	2	1	2	23	6	2	0
8	26	12	0	3	14	6	0	5487	0	4	1	34	0	0	2	4	0	0	0	40	1	4	0	3	0	2	12	11	2	8	34	39	6	8	3
10	3221	24	3	11	22	32	3	17	2276	46	14	6	0	5	2	15	9	0	0	6	30	11	0	4	0	4	24	10	5	34	19	34	19	24	4
48	119	212	7	12	71	90	13	19	1	8251	14	35	6	2	52	90	7	3	0	88	37	31	21	27	0	55	138	36	36	240	19	462	96	39	24
10	15	10	0	77	6	6	2	4	1	6	1083	6	0	1	5	13	0	1	0	16	9	6	4	4	0	6	17	15	1	9	3	27	4	2	8
2	9	3	1	0	3	5	0	8	0	4	0	2756	1	0	42	0	0	0	0	11	0	3	1	1	0	0	2	6	2	7	3	19	35	1	0
1	0	1	0	1	1	2	0	0	0	0	0	550	0	0	2	0	0	0	0	0	7	0	0	2	0	2	4	1	1	0	0	7	1	0	0
10	2	6	1	0	2	4	2	6	4	28	0	0	0	0	1983	1	7	1	1	0	8	1	0	3	0	0	8	2	0	401	3	17	5	6	0
3	13	13	0	2	5	8	0	4	0	6	1	25	1	0																					

Kernel: (16, 2), Adamax, cosine_proximity

5101	22	99	2	16	48	11	4	9	0	14	17	31	0	0	7	3	0	0	2	34	6	3	2	2	6	7	22	17	11	40	6	132	22	16	12	
31	13714	59	0	38	33	15	1	19	25	23	7	34	8	0	0	18	0	0	1	75	7	10	2	0	14	8	29	32	16	30	6	113	42	36	8	
10	21	5656	3	6	34	9	0	4	0	10	1	26	1	1	5	6	0	1	2	37	3	5	4	2	10	10	29	10	11	20	1	98	31	7	31	
16	18	78	1865	12	93	3	2	16	0	45	6	17	0	1	6	21	0	0	0	21	4	1	11	5	6	11	8	11	21	25	6	100	36	14	2	
10	10	19	0	5909	22	10	1	11	0	4	85	21	1	0	8	8	0	0	0	34	8	2	0	0	11	5	22	19	7	14	2	35	19	6	4	
7	25	34	21	7	7599	11	0	4	0	10	0	24	2	0	0	11	0	0	0	19	2	3	5	4	9	2	17	13	10	18	7	84	20	6	7	
23	64	111	8	15	60	11531	1	21	1	15	5	46	4	1	14	8	0	5	1	57	0	7	1	9	20	6	31	30	28	44	4	162	71	37	11	
2	2	7	1	3	5	4	1123	0	0	1	0	4	0	5	2	3	0	0	0	5	1	1	0	0	3	1	0	0	5	1	21	4	2	2		
3	23	14	0	5	18	3	0	5490	0	1	5	44	1	0	1	4	0	0	1	37	4	1	0	2	0	2	8	11	6	8	26	40	4	10	2	
14	2675	48	5	22	29	30	2	17	2776	56	15	4	1	4	0	26	4	0	0	8	10	4	1	4	3	3	35	11	9	30	16	29	19	33	1	
52	99	442	9	28	95	73	13	18	2	8030	17	47	7	1	27	106	0	1	2	86	19	17	19	28	14	36	126	45	95	223	19	435	102	45	23	
10	7	14	0	123	7	2	1	3	1	4	1106	6	0	0	2	8	0	1	1	15	2	2	0	1	0	4	9	9	3	7	4	14	5	1	5	
0	2	3	1	0	3	1	0	4	0	3	0	2834	1	0	14	0	0	0	2	7	0	0	0	0	2	0	2	6	1	4	1	9	24	1	0	
1	0	0	0	3	1	0	0	0	0	0	0	559	0	0	0	1	0	0	0	0	2	0	0	0	1	2	4	1	1	0	0	3	4	0	0	
8	2	10	2	1	15	3	2	6	4	41	0	0	0	0	1978	1	12	1	0	0	1	0	0	4	1	0	10	2	5	379	3	10	7	4	0	
3	11	18	2	4	6	6	0	4	0	6	3	46	1	0	818	10	0	0	1	5	5	0	0	4	2	2	9	7	2	4	0	21	57	8	0	
20	84	126	7	40	51	49	0	20	1	34	14	37	3	3	28	7766	0	1	2	156	5	9	8	35	13	20	87	44	21	39	29	126	47	32	6	
4	2	2	1	0	3	6	0	2	0	8	3	1	0	2	0	5	365	0	0	2	1	347	1	0	0	0	4	0	13	6	3	2	3	2	0	0
6	2	35	7	7	26	14	3	0	23	473	5	0	0	25	3	28	3	1572	0	0	1	0	0	2	1	0	11	3	6	16	1	1	18	4	2	
0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	3	1	0	0	
3	7	11	0	6	15	6	0	2	0	3	4	24	0	0	0	50	0	1	1	6106	0	2	0	0	4	0	2	4	3	2	107	22	1	13	4	
6	16	23	1	5	8	3	1	1	1	7	2	0	0	0	2	7	0	0	0	6	842	3	1	5	0	3	14	3	10	4	0	21	14	4	0	
3	6	5	0	4	10	5	0	1	0	1	3	12	0	0	1	1	3	1	1	4	0	2284	0	1	0	0	6	2	11	6	2	19	6	1	2	
0	3	7	0	1	0	1	0	1	0	4	0	1	0	0	0	0	0	0	0	3	0	0	0	0	1	0	0	0	4	0	4	1	0	0	0	
13	8	43	0	14	11	12	1	8	7	6	4	12	15	0	7	31	3	0	0	0	2	1	2	3276	14	4	21	12	10	31	1	58	24	4	3	
0	0	1	0	1	0	2	0	2	0	1	0	222	0	0	32	0	0	0	0	1	0	0	0	2	130	0	1	0	0	1	2	0	3	0	0	
2	4	4	0	1	3	0	1	0	0	1	0	4	0	0	1	0	0	0	0	6	1	0	0	0	1	664	27	4	3	8	0	19	3	0	2	
5	38	55	4	17	14	15	1	18	1	10	11	15	1	0	30	37	0	0	0	26	3	3	2	2	8	14	4663	19	0	13	3	20	18	7	0	
10	21	25	1	26	22	9	2	9	0	7	3	6	1	0	4	6	1	0	0	40	10	7	0	3	7	6	16	7820	11	15	2	59	16	5	5	
2	9	7	0	0	15	0	1	6	1	0	0	11	0	0	0	0	0	0	1	8	1	3	0	0	7	1	4	2	2488	5	0	21	3	1	1	
12	22	71	0	9	49	8	6	4	0	18	1	23	3	2	1	7	0	0	1	33	1	3	0	0	8	16	17	11	29	8240	7	195	24	13	13	
0	3	3	0	0	8	2	0	0	0	0	0	6	0	0	0	50	0	0	0	1733	0	0	0	0	4	3	1	0	1	0	1684	8	0	6	1	
10	36	76	0	11	51	4	2	14	1	14	2	31	1	0	2	1	0	1	2	38	2	8	0	0	11	27	17	17	21	41	2	14843	25	6	14	
18	19	143	9	14	28	27	4	8	0	23	10	32	4	0	68	34	0	0	1	22	8	3	4	14	5	6	32	11	22	45	10	111	2581	10	3	
19	37	57	2	22	35	13	4	10	0	23	4	21	2	0	11	63	0	0	2	37	3	3	0	4	4	22	7	21	20	16	19	37	118	51	9154	10
0	1	4	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	0	0	0	1	0	0	0	0	0	6	0	0	997		

0.882880062367491

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

5133	48	51	3	15	38	24	2	8	0	18	13	30	0	0	7	15	1	0	2	51	3	8	2	1	2	9	24	20	7	35	4	98	15	26	11
15	14004	19	0	41	24	15	1	15	3	11	4	35	3	0	0	6	1	0	2	46	5	14	0	0	2	9	15	18	5	23	5	58	20	29	6
13	45	5506	1	14	24	21	0	5	1	26	3	34	1	1	11	33	0	1	3	57	2	7	2	3	4	16	41	9	10	41	1	95	27	15	32
14	42	62	1807	14	37	14	2	13	0	63	12	24	0	2	11	64	0	0	0	35	6	1	6	6	2	15	10	8	8	43	5	79	55	19	2
11	10	6	0	5975	17	9	0	8	0	6	102	27	0	0	5	5	0	0	0	33	6	3	0	0	1	1	12	11	4	10	2	16	18	7	2
13	57	39	37	12	7297	41	0	5	0	16	11	38	3	0	1	41	1	2	1	62	2	10	2	7	4	4	41	20	6	42	9	97	18	31	11
16	70	44	5	15	30	11842	1	11	0	5	5	48	2	1	15	5	0	5	1	49	1	9	0	7	6	4	19	21	4	52	3	73	35	38	10
3	4	3	0	4	4	1140	0	0	0	2	3	0	4	0	0	4	0	0	0	5	1	2	0	0	0	1	1	0	3	2	1	11	3	3	1
6	35	10	0	6	12	5	0	5475	0	3	3	47	2	0	0	12	0	0	1	40	2	2	0	3	0	3	10	6	3	5	22	24	4	31	2
10	4157	10	2	19	13	35	1	16	1427	42	13	2	0	3	0	17	7	1	0	6	10	6	0	4	2	2	28	8	2	29	15	10	20	26	1
53	177	215	8	23	71	111	12	23	2	8178	28	44	6	2	41	175	5	6	2	106	23	22	5	29	4	37	154	37	24	253	18	332	97	55	23
9	14	6	0	118	6	3	0	2	0	2	1140	5	0	0	2	8	0	1	1	10	3	3	0	2	0	3	4	5	1	6	4	10	2	2	5
1	3	2	1	0	0	5	0	1	0	3	0	2858	1	0	4	0	1	0	2	6	0	1	0	0	0	0	1	4	0	4	1	4	21	1	0
1	1	1	0	7	1	0	0	0	0	0	0	561	0	0	1	0	0	0	0	0	1	0	0	0	0	1	3	1	1	0	0	2	1	0	0
10	3	2	1	1	4	4	0	5	3	32	1	0	0	0	15	1	0	0	0	0	1	0	0	5	0	0	13	2	0	310	3	3	5	5	0
3	20	19	1	3	5	9	0	2	0	5	7	60	1	0	794	15	0	1	0	3															

Kernel: (16, 2), Nadam, categorical_hinge

5141	48	30	7	6	38	12	7	15	0	13	9	25	1	0	29	10	0	0	2	27	4	3	0	8	0	0	48	13	11	22	2	126	36	21	10	
22	14049	14	1	15	24	6	1	16	21	14	2	40	4	0	1	15	0	1	0	50	2	4	0	0	0	0	15	22	12	11	2	46	25	15	4	
23	81	5180	22	5	54	19	6	8	6	33	1	34	1	2	69	44	0	4	7	45	5	6	0	8	0	0	115	10	23	42	1	160	50	17	24	
15	22	22	2122	5	20	4	2	19	0	19	1	20	0	0	14	15	0	0	2	19	4	1	0	4	0	0	6	5	13	9	2	63	36	15	2	
15	37	14	5	5635	17	11	4	14	0	8	188	41	8	0	26	27	0	0	2	33	13	2	0	7	0	0	39	25	8	16	0	51	51	7	3	
13	53	15	212	6	7324	19	1	16	1	10	0	39	5	0	14	18	0	0	2	20	2	9	0	5	0	0	26	9	15	7	6	89	28	12	5	
23	125	35	17	12	54	11453	5	21	2	19	1	57	4	3	39	24	0	5	2	53	0	10	0	32	0	0	89	35	32	43	2	143	56	46	10	
2	4	1	1	1	5	4	1138	0	0	0	0	4	0	5	3	4	0	2	0	5	1	2	0	0	0	0	0	0	4	0	12	7	3	1		
3	31	7	1	3	11	4	0	5513	0	3	0	60	3	0	3	0	0	0	1	26	4	1	0	2	0	0	17	7	4	3	16	24	19	7	1	
11	3207	9	10	9	13	14	1	16	2471	30	7	7	1	4	2	12	0	0	0	7	4	4	0	3	0	0	19	9	5	15	16	11	7	20	0	
59	176	150	58	16	74	69	18	18	4	8041	15	39	4	8	139	123	0	6	18	88	16	24	0	25	0	0	283	37	81	189	10	371	177	48	17	
13	21	8	4	95	5	3	1	4	5	2	1053	6	0	1	18	18	0	2	3	12	5	4	0	7	0	0	21	9	3	7	2	18	21	1	5	
1	4	2	1	0	2	1	0	4	0	3	0	2827	1	0	25	0	0	0	0	3	0	0	0	1	0	0	1	4	1	5	1	6	30	1	1	
1	0	0	0	0	0	0	0	0	0	0	0	569	0	1	2	0	0	0	0	0	2	0	0	0	0	0	2	1	1	0	0	4	0	0	0	
10	1	1	16	0	14	1	3	5	7	24	0	0	0	0	2244	2	9	1	0	0	1	1	0	0	3	0	0	11	2	4	125	2	4	16	5	0
1	14	11	7	2	5	5	0	3	0	3	0	39	2	0	879	4	0	0	0	3	3	0	0	3	0	0	6	6	0	3	0	17	42	7	0	
21	116	38	21	12	43	43	2	27	7	29	9	49	4	7	79	7897	0	1	12	75	7	10	0	40	0	0	102	27	15	30	28	109	61	39	3	
4	5	0	0	0	2	5	0	2	3	6	1	1	0	2	0	3	172	0	0	3	0	0	547	0	0	0	5	0	12	7	3	1	2	2	0	
3	3	5	24	1	13	4	1	0	23	138	0	1	0	29	7	19	2	1969	1	0	0	0	2	0	0	18	2	3	6	1	1	20	1	1		
0	2	0	0	0	2	0	0	0	0	0	0	0	1	0	11	0	0	0	258	0	0	0	0	0	0	0	1	0	0	0	1	2	1	0	0	
3	15	5	0	2	16	5	1	13	0	2	3	36	0	0	1	156	0	1	0	6052	0	2	0	0	0	0	4	5	2	3	36	17	4	16	3	
6	30	7	3	0	7	3	3	1	2	5	1	0	2	0	6	12	0	0	0	6	835	5	0	5	0	0	21	2	8	3	0	17	20	3	0	
2	22	1	2	1	7	3	0	1	0	2	2	10	0	0	2	1	2	1	1	6	0	2304	0	1	0	0	5	2	1	4	0	12	4	1	1	
0	15	18	4	1	0	1	0	1	0	29	0	3	0	0	93	14	0	0	16	4	2	1	0	1	0	0	52	1	1	5	0	12	45	0	0	
12	14	15	7	3	10	7	1	9	13	6	1	26	16	0	17	24	1	0	3	0	2	2	0	3332	0	0	32	9	5	9	2	54	19	4	3	
0	0	0	0	0	0	2	0	1	0	0	0	328	0	0	60	0	0	0	0	0	0	0	0	1	0	0	1	0	2	1	2	0	3	0	0	
3	12	6	1	0	5	1	3	0	0	9	0	4	1	0	5	4	0	0	305	7	4	0	0	0	0	0	291	7	6	13	0	64	4	2	2	
5	55	25	10	11	12	10	2	20	6	3	1	26	1	0	32	38	0	0	5	22	2	3	0	1	0	0	4727	12	2	5	2	19	11	5	0	
12	38	20	16	13	22	6	7	10	0	6	5	14	5	0	16	18	0	1	1	37	12	7	0	6	0	0	30	7756	12	8	0	75	16	4	2	
3	17	1	2	0	14	1	2	7	1	7	0	13	0	0	6	3	0	0	0	12	1	5	0	1	0	0	8	4	2458	3	0	21	6	1	1	
33	44	22	12	4	60	12	12	7	2	21	3	26	4	12	13	20	0	2	5	37	3	4	0	3	0	0	43	13	35	8140	4	177	35	29	10	
2	4	0	0	0	9	2	0	1	0	0	0	13	0	0	0	72	0	0	1	2695	0	0	0	0	0	0	0	1	0	0	705	1	1	5	1	
18	102	46	11	2	54	9	7	22	0	21	1	47	1	0	11	2	0	2	9	47	2	10	0	1	0	0	21	10	27	60	4	14721	39	13	11	
16	37	112	20	4	21	25	7	8	1	19	4	39	3	1	109	35	0	0	5	20	5	5	0	16	0	0	61	12	14	29	5	80	2601	13	2	
20	114	16	10	13	29	9	3	15	8	28	2	34	5	0	24	52	0	2	4	40	5	4	0	3	0	0	22	15	11	11	30	93	27	9195	3	
0	3	6	0	0	2	0	0	0	0	1	0	2	0	0	1	0	0	0	0	5	0	0	0	0	0	0	2	0	0	0	0	9	1	2	979	

0.8690555096080806

Kernel: (16, 2), Nadam, logcosh

5440	5	51	1	9	14	2	2	5	0	10	3	13	0	0	7	1	0	0	2	15	4	3	0	1	0	4	12	12	2	26	6	60	5	3	6	
58	13782	42	0	24	23	16	1	33	13	37	4	47	0	0	1	27	0	1	2	71	13	11	0	0	0	5	20	11	6	31	6	111	35	18	5	
19	26	5715	0	4	13	5	0	8	0	10	0	29	0	0	15	16	0	2	2	36	2	6	5	0	0	11	27	9	1	21	0	85	13	5	20	
41	28	124	1618	12	28	9	2	19	0	112	3	25	0	1	33	95	0	0	0	23	20	1	29	6	0	20	18	10	9	28	5	90	59	11	2	
14	23	18	0	5949	14	8	0	16	0	6	46	27	0	0	11	14	0	1	0	30	10	2	2	0	2	19	20	4	11	2	34	18	2	2		
42	50	81	45	11	7113	25	0	18	0	32	2	40	0	0	19	75	0	1	2	47	12	10	6	15	0	4	61	23	7	43	10	140	25	13	9	
45	67	129	4	18	38	11460	1	26	0	27	4	69	0	1	41	24	0	8	2	56	3	11	2	18	0	7	39	22	10	45	4	154	78	30	9	
5	3	8	0	2	4	4	1126	0	0	2	0	5	0	4	1	9	0	1	0	4	1	2	0	1	0	1	2	0	1	2	1	16	2	1	1	
13	6	10	0	3	10	3	0	5594	0	1	0	38	0	0	0	2	0	0	1	25	3	1	0	2	0	2	5	4	0	1	19	22	3	5	1	
23	3144	25	2	17	14	32	1	32	2290	70	8	6	0	2	5	48	2	1	0	9	32	6	0	4	0	2	38	7	2	35	16	23	23	24	1	
76	93	451	7	17	59	66	11	24	2	8061	11	45	0	1	102	233	0	11	2	79	42	17	19	27	0	45	152	41	14	199	17	369	65	29	14	
13	14	17	0	149	5	2	1	4	0	2	1018	6	0	0	11	32	0	0	2	1	9	12	4	2	3	0	5	13	9	1	7	4	20	6	1	4
4	1	4	1	1	1	0	0	4	0	3	0	2851	0	0	6	0	0	0	0	12	0	1	0	0	0	0	1	5	0	3	1	4	22	0	0	
9	4	43	4	38	3	2	0	5	0	4	0	4	0	0	22	6	0	36	5	0	130	0	59	48	0	26	88	2	1	8	0	12	24	0	0	
11	1	13	0	1	2	3	0	6	5	52	0	0	0	2013	10	24	1	1	0	0	3	0	0	4	0	0	13	4	0	324	3	10	5	3	0	
4	14	22	0	1	4	5	0	3	0	3	0	73	0	0	828																					

ARat_2017_05_31_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

8553	0	91	2	71	108	104	3	2	108	42	0	8	135	1	6	9	23	266	43	14	3	26	1	0	0	16	0	4	63	0	0	86	1
1	5996	1	0	0	0	0	1	0	2	1	8	0	0	1	0	2	1	0	1	1	0	0	0	0	0	17	0	211	1	4	3	0	1
36	0	5381	3	39	16	52	1	3	45	13	0	5	65	1	13	2	39	113	16	20	0	11	0	0	5	0	11	16	0	0	40	1	
16	0	15	625	3	11	32	0	2	10	11	0	5	12	3	0	1	1	31	11	0	0	2	0	0	18	0	15	7	0	0	8	0	
1	0	3	3	5204	7	3	0	2	1	2	0	3	10	0	0	0	2	27	0	0	0	1	0	0	0	0	0	1	0	0	5	0	
7	0	8	1	12	4109	14	1	2	16	0	0	3	19	0	6	0	4	46	267	0	1	2	0	0	0	0	1	5	0	0	31	0	
7	0	8	0	12	15	3731	0	1	21	6	0	4	11	2	0	0	1	53	5	0	0	4	0	0	0	0	1	4	0	0	10	0	
0	15	4	3	0	3	0	907	0	1	134	7	2	1	1	1	3	7	0	1	1	3	0	1	0	0	2	0	3	0	0	2	0	1
22	1	6	0	974	20	12	0	1715	14	13	0	4	11	1	2	1	4	56	9	1	1	5	1	0	0	3	0	1	7	0	0	10	1
14	0	14	1	16	19	23	0	11	7300	4	0	1	41	0	0	0	5	106	10	3	0	3	0	0	0	2	0	3	13	0	0	48	0
34	0	122	11	45	49	51	8	2	62	7249	0	13	106	1	0	2	58	161	42	0	5	10	0	0	0	7	0	1	41	0	0	47	0
1	8	0	1	0	0	0	0	0	10	0	1127	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	1	0	0	0	0	1
5	0	10	4	10	4	10	0	0	2	6	0	0	1951	21	0	3	0	1	39	3	1	0	1	0	0	1	0	1	5	0	0	7	0
1	0	5	2	19	1	8	1	0	2	0	0	1	5329	0	0	0	0	27	1	0	0	2	2	0	0	1	0	0	2	0	0	33	0
10	0	10	1	1	10	1784	0	4	7	11	0	0	9	1641	0	0	7	49	3	0	1	10	2	0	0	5	0	0	4	0	0	10	0
6	16	312	1	0	6	2	3	0	3	2	0	2	1	5	781	0	6	2	1	2	2	0	1	0	0	6	0	9	1	1	0	2	0
4	5	2	1	0	2	1	0	0	2	0	1	1	0	0	0	0	434	2	2	1	2	0	1	0	0	3	0	5	171	0	0	0	1
10	0	77	3	11	121	22	0	0	21	6	0	9	36	0	3	0	3304	53	387	52	4	1	1	0	0	1	13	0	0	11	0	0	
20	0	26	5	60	32	42	0	0	50	12	0	7	127	0	2	0	8	24778	12	4	0	4	0	0	0	0	4	21	0	0	155	0	
17	0	33	4	11	4446	22	0	0	16	2	0	8	45	0	10	3	22	100	3015	9	3	2	0	0	0	2	0	0	8	0	0	37	0
26	35	120	6	1	40	15	1	2	6	5	20	3	14	4	13	10	229	13	650	3203	42	1	0	0	12	0	4	9	3	5	5	4	
15	9	1	2	0	6	4	5	2	0	0	5	0	4	2	0	3	12	1	1	5	4	1022	0	0	9	0	1	0	16	1	0	0	0
15	0	1	0	11	44	14	0	0	27	24	0	5	2	0	1	0	1	92	0	0	0	4627	3	0	0	0	3	4	0	0	829	0	
2	0	0	0	3	3	4	0	2	5	4	0	4	3070	1	0	1	2	39	0	0	0	12	568	0	0	7	0	0	2	0	0	18	0
1	1	1	0	0	0	0	89	0	0	123	2	0	0	0	0	0	106	0	11	0	151	0	0	0	0	0	0	0	0	0	0	0	0
28	2	2	0	5	0	1	0	3	5	0	4	2	0	0	0	32	0	0	7	0	1	0	0	0	0	0	2	25	1	7	0	0	0
102	0	49	7	21	57	78	2	10	53	49	0	17	451	20	0	9	17	194	8	2	10	15	27	0	0	6000	0	28	29	1	0	71	2
39	1	2	0	0	0	0	1	1	0	0	0	27	0	9	0	52	0	0	0	7	11	0	0	0	0	5	0	0	0	0	0	0	0
216	0	128	22	10	50	93	0	44	56	44	0	37	65	70	3	7	37	6433	88	0	6	73	27	0	0	167	0	5193	57	0	3	67	1
3	0	2	1	18	15	8	0	0	16	2	0	5	25	0	3	4	0	47	1	1	0	1	0	0	0	0	0	0	0	0	15	5	
8	260	3	0	0	0	1	4	0	1	0	5	1	0	2	8	17	0	0	0	9	6	0	0	0	0	14	0	0	1	2069	5	0	2
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	2	6	2	0	0	0	8	0	0	1	2	0	0	0	0	19	0	0	0	4	1	0	0	0	0	1	0	0	0	6487	0
0	3	1	0	0	0	0	0	0	0	1	1	0	0	0	4	1	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	477	0

0.8041868507862091

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

8436	0	129	12	67	51	89	3	0	191	40	0	40	137	0	3	0	27	288	29	13	3	36	1	0	0	38	0	25	68	0	0	62	1
0	5946	3	2	0	0	0	1	0	2	1	6	8	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	265	0	3	0	0	1
26	0	5423	19	39	5	30	1	1	62	8	0	10	53	0	3	0	35	127	7	15	0	15	0	0	0	7	0	21	16	0	0	24	0
8	0	9	725	1	4	5	0	0	10	6	0	15	8	1	0	0	1	22	0	0	0	2	0	0	0	6	0	8	4	0	0	4	0
2	0	3	3	5225	0	1	0	1	0	1	0	4	12	0	0	0	1	20	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
15	0	21	18	31	3487	20	0	0	38	2	0	9	32	0	4	0	15	85	666	1	44	9	0	1	0	11	1	7	6	0	0	32	0
9	0	15	2	20	8	3682	0	0	35	5	0	7	19	1	0	0	1	65	4	0	0	6	0	0	0	1	0	3	4	0	0	9	0
0	16	7	6	0	0	1	844	0	1	186	6	5	2	1	1	0	8	0	2	1	1	0	0	1	0	6	0	4	0	1	2	0	1
23	2	10	7	1469	10	10	0	1160	57	11	0	13	14	2	0	0	6	54	8	2	0	5	2	0	0	11	0	4	8	0	0	7	0
8	0	13	2	17	4	5	0	1	7436	1	17	2	2	30	0	0	0	4	68	2	0	0	1	0	0	4	6	0	0	0	30	0	0
35	0	133	54	48	12	40	0	1	96	7207	0	31	108	0	0	1	48	169	22	0	3	8	0	0	0	23	0	10	40	0	0	38	0
0	8	0	3	0	0	0	1	0	11	0	1126	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	5	4	6	1	3	0	0	4	2	0	1996	15	0	1	0	0	34	0	1	0	0	0	0	0	0	2	3	0	0	6	0	0
2	0	6	2	14	1	6	1	0	7	0	0	2	5332	0	0	0	0	29	0	1	0	3	2	0	0	0	0	1	2	0	0	26	0
17	0	18	8	1	5	2040	0	3	16	12	0	13	14	1306	0	0	7	57	5	0	0	14	2	0	0	20	0	5	6	0	0	10	0
5	17	413	14	0	0	2	3	0	6	1	0	4	0	1	661	0	5	2	0	5	1	0	0	1	0	15	0	14	1	1	0	1	0
6	7	4	20	0	1	0	0	0	3	1	1	5	1	0	0	347	2	2	0	2	0	2	0	0	0	6	0	12	218	0	0	0	1
8	0	129	10	15	25	21	0	0	38	8	0	15	36	0	1	0	3439	67	177	110	5	5	0	1	0	5	0	5	17	0	0	10	0
14	0	23	6	65	6	20	0	0	86	7	0	11	117	0	1	0	6	24876	2	4	0	3	0	0	0	0	10	18	0	0	94	0	0
36	0	46	30	31	2786	34	0	0	70	9	0	21	57	0	6	0	32	150	4387	17	29	12	0	0	5	0	0	18	0	0	39	0	0
24	37	162	48	1	12	8	2	1	17	6	21	17	12	0	11	2	224	14	287	3466	61	2	0	4	0	32	0	11	9	1	3	3	3
17	9	3	5	0</																													

Kernel: (4, 2), Adamax, cosine_proximity

8022	0	207	3	131	109	70	6	1	156	49	0	9	110	4	9	12	46	394	40	23	0	46	3	0	0	20	0	135	108	0	0	75	1
0	5897	1	0	0	0	0	1	0	2	1	2	0	0	0	0	0	0	0	0	1	0	0	1	0	0	4	0	339	1	0	2	0	1
18	0	5397	3	50	8	16	2	0	45	8	0	3	40	0	8	0	55	144	3	21	0	22	3	0	0	0	62	12	0	0	26	1	
12	0	23	587	10	9	11	0	0	9	8	0	4	6	2	0	1	5	37	6	0	0	3	1	0	0	7	0	84	10	0	0	4	0
1	0	3	2	5231	1	0	0	0	1	0	0	1	6	0	0	0	0	26	0	0	0	0	0	0	0	0	0	1	0	0	0	2	0
7	0	6	1	26	4075	8	0	0	24	1	26	0	5	15	0	8	0	13	80	232	0	0	9	1	0	1	1	5	5	0	0	32	0
14	0	27	0	29	16	3607	0	0	35	7	0	4	14	4	0	0	1	95	6	1	0	10	0	0	0	0	7	6	0	0	13	0	
0	11	6	1	0	2	0	930	0	2	108	5	0	1	1	1	0	14	1	0	1	0	0	1	0	0	2	0	12	1	0	2	0	1
13	1	7	1	1782	15	6	0	924	22	6	0	3	7	1	1	0	4	57	4	1	0	6	2	0	6	2	0	6	10	0	0	8	0
11	0	13	1	20	9	5	0	2	7329	1	0	1	26	0	0	0	6	122	5	5	0	5	1	0	1	1	0	14	13	0	0	46	0
24	0	157	7	70	34	33	31	1	76	7133	0	15	72	0	0	0	103	196	19	0	0	18	3	0	0	4	1	26	63	0	0	41	0
0	13	0	2	1	0	0	1	0	19	0	1112	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	
5	0	9	3	16	2	3	0	0	5	6	0	1911	15	0	3	0	2	62	1	2	0	5	0	0	0	2	23	5	0	0	5	0	
1	0	15	2	31	1	7	2	0	6	1	0	1	5251	0	0	0	1	55	0	2	0	5	11	0	0	1	5	4	0	0	35	0	
12	0	12	1	3	8	1272	1	3	10	11	0	1	2	2084	0	0	7	80	5	0	0	19	7	0	0	10	0	16	6	0	0	9	0
4	18	308	0	0	2	1	4	0	2	2	0	2	0	0	777	0	6	2	1	2	1	0	0	0	0	5	0	35	0	0	0	1	0
2	7	4	0	0	1	0	0	0	1	1	1	0	0	0	0	359	2	4	1	2	0	2	0	0	0	2	25	225	0	0	0	2	
2	0	83	3	16	72	8	0	0	20	5	0	8	24	0	2	0	3544	73	147	103	0	8	2	0	0	1	1	3	11	0	0	11	0
5	0	15	4	60	4	2	0	0	31	6	0	3	72	0	1	0	6	25027	1	2	0	7	3	0	0	0	6	12	0	0	102	0	
17	0	40	5	29	4639	9	0	0	36	5	0	7	41	0	10	1	38	159	2690	15	1	10	3	0	0	0	5	16	0	0	39	0	
10	35	134	5	2	29	5	2	0	14	4	15	3	4	3	14	5	340	15	360	3434	4	3	2	0	0	10	25	13	2	3	3		
41	10	6	5	0	13	2	2	3	1	4	0	2	4	0	18	24	3	2	16	45	835	0	0	0	0	17	15	6	1	46	4	0	0
4	0	0	0	13	13	3	0	0	22	19	0	3	0	0	1	0	0	89	0	0	0	5326	4	0	0	0	9	1	0	0	196	0	
2	0	7	1	6	2	3	0	1	5	5	0	2	2027	3	0	0	3	71	0	0	0	12	1557	0	0	1	0	19	4	0	0	16	0
0	1	2	0	0	0	0	8	0	1	335	1	0	0	0	0	0	128	0	1	0	7	0	0	0	1	0	0	0	0	0	0	0	
7	2	3	1	11	0	0	0	1	2	0	1	0	0	0	0	23	0	0	3	1	1	0	0	0	45	1	0	4	21	0	0	0	0
74	1	91	16	43	34	36	6	7	67	51	0	16	196	26	1	15	31	242	9	4	2	24	272	0	0	5633	1	325	43	0	0	60	3
3	3	0	0	0	0	0	1	0	0	0	0	6	0	0	2	0	1	0	0	3	0	0	0	0	0	0	0	0	2	0	0	1	0
51	0	50	0	18	19	23	0	20	31	36	0	13	7	46	2	0	32	4807	30	0	2	71	41	0	1	40	0	7586	30	0	0	40	1
3	0	3	2	25	9	3	1	0	17	2	0	5	16	0	4	0	1	74	2	2	0	3	1	0	0	0	2	3008	0	0	15	5	
7	745	4	0	0	0	0	6	0	1	0	5	2	0	1	9	7	0	0	0	11	0	0	0	0	3	0	17	1	1590	5	0	2	0
0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	2	10	2	0	0	0	4	0	0	1	1	0	0	0	0	34	0	0	0	4	2	0	0	0	1	0	0	0	384	0	0
0	4	1	0	0	0	0	0	0	2	1	1	0	0	0	8	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	472

0.8180410265922546

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

8945	0	58	5	44	34	50	2	0	141	42	0	6	100	0	3	0	5	223	14	6	0	27	1	0	0	8	0	5	24	0	0	46	0
21	5550	4	3	0	0	1	2	0	2	3	7	6	0	0	3	1	1	0	0	1	2	0	1	0	0	43	1	590	2	5	2	0	2
77	0	5286	28	37	7	29	1	1	113	22	0	7	57	1	6	0	20	122	5	39	0	17	3	0	0	5	0	34	9	0	0	21	0
18	0	7	704	1	5	8	0	0	28	9	0	2	11	1	0	1	0	23	0	0	0	3	0	0	6	0	7	3	0	0	2	0	
3	0	3	3	5206	0	3	0	1	5	2	0	2	12	0	0	0	0	31	0	0	0	1	0	0	0	0	0	1	0	0	2	0	
66	0	10	5	24	3972	14	0	0	77	5	0	4	24	0	8	1	7	73	207	3	0	8	2	1	0	6	1	4	5	0	0	28	0
23	0	17	0	16	8	3657	0	1	50	18	0	4	17	0	0	0	1	60	4	0	0	5	1	0	0	1	0	2	3	0	0	8	0
3	8	4	4	0	1	0	816	0	1	242	6	0	2	1	1	0	2	0	0	0	0	0	1	1	0	6	0	3	0	0	0	0	1
37	0	6	3	1009	8	11	0	1665	40	15	0	4	10	0	2	1	1	55	5	1	0	3	3	0	0	2	0	2	6	0	0	6	0
23	0	5	5	15	4	3	0	5	7445	3	1	1	33	0	0	1	1	57	2	1	0	2	0	0	2	0	2	5	0	0	21	0	
49	0	52	24	37	14	22	3	1	75	7561	0	11	79	0	0	0	10	128	8	0	0	6	1	0	0	5	0	4	10	0	0	27	0
1	10	0	5	0	0	0	1	0	15	0	1117	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0
7	0	9	5	10	2	4	0	0	8	11	0	1948	18	0	3	0	0	39	2	2	0	3	0	0	0	1	1	5	3	0	0	4	0
1	0	5	3	13	0	6	0	0	10	2	0	1	5336	0	0	0	0	28	0	0	0	3	5	0	0	0	2	1	0	0	21	0	
38	0	17	8	1	6	2328	0	3	24	19	0	2	9	1012	0	0	3	58	4	0	0	11	6	0	0	14	0	7	3	0	0	6	0
25	11	328	8	0	0	1	5	0	24	5	0	2	0	0	732	0	4	1	0	1	1	1	0	1	0	6	0	16	0	1	0	0	0
19	4	3	17	0	3	1	0	0	6	1	1	2	2	0	0	330	1	3	1	2	0	1	0	0	8	0	10	225	0	0	0	1	0
51	0	196	11	14	66	22	0	0	106	37	0	11	38	0	2	1	3124	72	238	127	0	4	1	0	1	0	3	0	3	0	8	0	0
44	0	22	6	53	10	29	0	0	168	23	0	9	137	0	2	0	4	24739	5	0	0	5	1	0	0	1	0	13	12	0	0	86	0
119	0	45	16	24	4506	32	1	0	132	18	0	9	56	0	9	2	18	146	2612	16	0	8	1	0	0	2	0	0	11	0	0	32	0
179	26	173	32	0	26	10	4	1	45	14	19	3	12	0	18	6	166	14	383	3299	7	2	0	5	0	26	1	14	7	2	3	3	1
202	8	5	7	0	4	4	4	2	13	0	6	1	2	0	8	6	0	1	7	17	760	0</											

Kernel: (16, 2), RMSprop, categorical_crossentropy

8374	0	103	5	45	48	71	1	9	531	18	0	6	75	0	5	0	53	208	29	12	0	30	1	0	0	45	0	22	43	0	0	55	0	
0	5953	1	0	0	0	0	0	0	2	1	2	0	0	0	0	0	1	0	0	1	0	0	1	0	0	16	0	272	0	0	2	0	1	
29	0	5269	4	34	8	27	0	7	152	2	0	3	28	0	38	0	105	88	12	40	0	18	1	0	0	11	0	44	9	0	0	18	0	
13	0	17	581	1	5	7	0	2	84	7	0	2	3	1	0	0	5	20	5	1	0	3	1	0	0	38	0	37	4	0	0	2	0	
3	0	7	4	5179	1	5	0	2	15	1	0	2	11	0	0	0	3	30	0	1	0	2	0	0	0	0	0	0	3	0	0	6	0	
14	0	6	1	16	3347	11	1	0	63	2	0	4	13	0	6	0	39	50	941	1	0	7	2	0	0	6	0	2	4	0	0	19	0	
14	0	20	0	13	7	3638	0	9	89	2	0	2	8	1	0	0	4	53	8	1	0	7	1	0	0	2	0	5	5	0	0	7	0	
3	17	6	1	0	0	0	817	0	7	98	7	0	3	1	2	0	75	0	1	3	0	0	1	2	0	23	0	30	0	1	4	0	1	
14	1	2	1	405	7	6	0	2325	61	2	0	2	4	0	1	0	4	33	4	1	0	5	1	0	4	1	1	4	4	0	0	6	0	
6	0	1	1	7	1	2	1	2	7569	0	2	0	8	0	1	0	6	12	1	4	0	1	0	0	1	0	2	1	0	0	8	0	0	
74	0	198	8	50	15	65	6	7	781	6118	0	11	81	0	4	0	263	155	25	5	1	17	7	1	0	124	0	35	39	0	0	36	1	
0	9	0	0	0	0	0	0	0	29	0	1111	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
6	0	18	5	11	2	10	0	0	28	4	0	1867	16	0	3	0	5	41	5	7	0	5	0	0	8	0	34	5	0	0	5	0	0	
4	0	24	3	17	2	9	1	1	42	1	0	2	2	0	2	0	6	37	1	4	0	2	28	0	0	11	0	7	3	0	0	22	0	
27	0	17	1	1	2	1770	0	9	50	7	0	0	3	1537	0	0	16	41	10	0	0	15	8	0	0	30	0	22	6	0	1	6	0	
5	16	241	0	0	0	1	1	0	14	1	1	0	0	1	835	0	7	1	1	7	1	0	0	1	0	16	0	23	0	0	0	0	0	
7	8	5	0	0	0	0	0	0	14	0	1	0	0	0	0	315	5	2	3	2	0	1	0	1	3	13	0	25	235	0	0	0	1	
3	0	27	3	8	18	7	0	0	42	0	0	0	16	0	3	0	3785	28	90	92	0	3	1	1	0	5	0	2	6	0	0	7	0	
42	0	76	8	52	26	47	0	1	312	6	0	7	101	0	5	0	36	24421	18	14	0	10	3	0	0	15	0	37	26	0	0	106	0	
33	0	23	7	19	2266	25	0	3	109	3	0	6	29	0	9	0	88	94	5039	17	1	5	0	0	0	5	0	1	8	0	1	24	0	
14	30	78	6	0	12	3	0	1	45	3	13	0	5	0	19	0	491	9	202	3513	3	2	1	2	0	21	0	13	6	1	3	2		
28	9	3	2	0	6	1	2	2	12	0	2	1	1	0	7	2	5	1	14	81	812	0	0	1	1	95	1	1	0	35	1	0	0	
21	0	10	0	8	26	9	0	3	64	14	0	3	1	0	4	0	7	70	0	0	0	5326	13	0	0	7	0	16	5	0	0	96	0	
5	0	12	1	2	3	7	0	9	36	1	0	3	1480	0	0	0	4	39	0	0	0	8	2080	0	0	29	0	21	3	0	0	4	0	
0	1	1	0	0	0	0	0	0	6	1	0	0	0	0	0	0	4	0	0	0	1	0	0	470	0	1	0	0	0	0	0	0	0	
7	1	1	1	3	0	0	0	1	8	0	0	0	0	0	11	0	0	0	1	1	1	0	0	0	79	4	0	4	3	1	0	0	0	
62	2	42	6	9	17	23	1	13	213	16	0	5	40	5	0	0	28	124	5	3	0	11	48	0	0	6560	0	63	16	0	0	16	1	
9	2	1	0	0	0	0	0	0	0	0	1	3	0	1	0	0	0	0	0	7	2	0	0	0	0	7	121	3	1	4	0	0	0	
172	0	110	14	1	21	45	0	51	344	27	0	13	10	36	5	0	77	1715	65	1	1	59	47	0	0	252	0	9866	38	0	1	26	0	
6	0	5	1	10	7	7	0	0	64	1	0	5	17	0	2	1	10	42	3	3	0	1	1	0	0	0	0	0	0	0	8	5	0	
9	604	2	1	0	1	0	3	0	2	0	4	0	0	0	9	2	0	0	0	17	0	0	0	0	0	27	1	12	1	1717	2	0	2	
0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	384	0	0	0
0	0	7	3	6	7	0	0	0	51	0	0	1	4	0	0	0	2	23	0	1	0	8	22	0	0	4	0	3	2	0	0	6389	0	0
0	4	1	0	0	0	0	0	0	1	0	1	0	0	0	10	1	0	0	0	0	0	0	0	0	1	0	0	0	4	0	0	0	468	0

0.8544417560100556

Kernel: (16, 2), Adadelta, cosine_proximity

8398	0	118	10	82	124	149	7	3	184	58	0	6	91	1	11	0	28	249	48	18	1	41	1	0	0	29	0	25	58	0	0	49	0	
2	5984	0	0	0	0	0	1	0	2	1	3	0	0	1	0	0	0	0	1	1	0	0	0	0	0	12	0	242	1	1	1	0	0	0
32	0	5354	11	47	11	39	1	3	68	18	0	3	50	1	24	0	56	114	9	19	0	20	1	0	0	2	0	28	12	1	0	23	0	
10	0	9	692	4	7	22	1	1	18	8	0	3	5	1	0	1	1	19	5	0	0	3	0	0	0	10	0	14	3	0	0	2	0	
1	0	3	2	5236	1	2	0	1	0	1	0	1	11	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
6	0	6	2	24	4172	14	0	1	20	1	0	2	15	0	11	0	7	44	191	0	2	5	0	0	0	1	1	1	5	0	0	24	0	
7	0	14	1	14	19	3702	0	1	34	9	0	2	13	3	0	0	1	51	3	0	0	7	0	0	0	0	4	3	0	0	8	0	0	
0	15	4	4	0	3	1	884	0	2	158	3	0	2	2	2	0	11	0	1	1	0	0	0	0	3	0	5	1	0	1	0	0	0	
13	1	7	1	1184	19	11	0	1568	7	14	0	2	8	2	1	0	1	33	3	1	0	3	1	0	0	2	0	2	5	0	0	6	0	
11	0	7	1	18	12	7	0	15	7403	5	0	12	0	28	0	3	69	8	5	0	5	1	0	0	8	1	0	8	7	0	0	21	0	
27	0	62	9	47	37	59	20	1	82	7426	0	7	72	0	1	0	59	119	27	0	0	11	1	0	0	7	1	4	22	0	0	26	0	
0	11	0	2	1	0	1	1	0	46	1	1088	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	14	12	13	8	20	0	0	9	11	0	8	1875	17	1	3	0	2	37	9	4	0	0	0	0	4	5	15	9	0	0	4	0	0
0	1	11	3	21	1	8	2	0	9	2	0	1	5309	0	0	0	0	29	1	2	0	7	5	0	0	1	0	2	3	0	0	19	0	
8	0	11	1	1	11	1692	0	3	13	11	0	0	7	1746	0	0	2	33	4	0	0	11	3	0	0	7	0	7	4	0	0	4	0	0
6	19	289	8	0	3	2	2	0	7	3	1	1	1	1	802	0	6	1	0	0	1	1	1	0	0	4	0	13	0	1	0	0	0	
5	8	5	9	0	5	0	0	0	5	1	1	0	0	0	0	372	3	2	3	1	0	0	0	0	0	4	0	11	206	0	0	0	0	0
4	0	104	5	15	149	20	0	0	23	20	0	6	29	0	4	23	3178	54	414	91	3	6	0	0	0	1	0	2	11	0	0	8	0	0
19	0	41	8	96	37	63	0	1	108	27	0	6	123	1	4	0	8	24647	18	3	0	10	2	0	0	2	0	30	28	0	0	87	0	0
17	0	32	9	22	4991	24	1	0	31	6	0	5	37	0	11	0	21	98	2455	10	0	6	1	0	0	2	0	1	10	0	0	25	0	0
29	36	133	11	2	41	11	3	2	17	13	14	2	7	5	23	3	214	12	782	3092	15	2	1	0	0	10	1	5	7	2</				

Kernel: (16, 2), Adam, categorical_hinge

8575	0	103	2	59	49	51	1	4	144	73	0	5	71	0	16	0	14	330	25	19	0	51	2	0	0	19	0	72	42	0	0	58	4	
3	5580	1	0	0	0	0	0	0	4	1	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	651	1	1	0	0	1	
24	0	5278	7	43	3	20	0	2	67	23	0	5	47	1	14	0	37	168	3	17	0	24	6	0	0	2	0	126	7	0	0	21	2	
18	0	13	563	3	5	14	0	2	30	11	0	5	5	1	0	1	2	32	6	0	0	3	1	0	0	17	0	97	6	0	0	3	1	
2	0	3	2	5199	2	0	0	4	0	1	0	2	10	0	0	0	0	40	0	0	0	2	0	0	0	0	0	1	0	0	0	7	0	
42	0	10	4	24	3604	11	0	1	43	6	0	4	16	0	14	0	16	99	584	3	1	14	5	0	0	2	0	15	8	0	0	28	1	
31	0	26	0	17	14	3577	0	3	52	15	0	5	10	0	0	0	1	92	5	1	0	12	1	0	0	0	0	18	4	0	0	12	0	
10	17	10	0	0	0	1	533	0	9	481	2	0	1	1	2	0	9	1	0	0	0	1	0	0	6	0	16	0	0	2	0	1		
22	1	10	0	826	11	8	0	1888	5	18	0	4	9	0	1	0	0	55	4	1	0	10	2	0	0	4	4	7	0	0	7	1		
11	0	19	1	15	0	3	0	25	7339	3	0	1	22	0	2	0	1	131	2	3	0	10	1	0	0	0	0	11	7	0	0	29	1	
32	0	81	5	44	18	18	0	1	77	7486	0	7	49	0	2	0	13	182	5	0	0	24	6	0	0	4	0	26	15	0	0	31	1	
2	11	0	1	0	0	0	0	0	59	0	1077	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
11	0	11	3	12	2	3	0	0	6	9	0	1874	13	0	3	0	1	60	1	5	0	7	0	0	0	2	0	52	6	0	0	4	0	
5	0	10	2	21	1	7	0	0	12	4	0	2	5214	0	1	0	0	71	0	2	0	15	20	0	0	4	2	0	0	0	28	1		
33	0	22	2	1	9	2176	0	5	22	16	0	4	3	1093	1	0	4	89	4	0	0	21	8	0	0	10	0	43	4	0	0	9	0	
6	19	340	0	0	0	2	0	0	5	3	1	2	0	0	734	0	3	4	0	0	1	2	2	0	0	1	0	46	0	0	0	1	1	
14	9	4	0	0	0	1	0	1	3	1	1	1	0	0	1	0	198	1	4	3	1	1	0	0	10	0	30	353	0	0	0	4	0	
16	0	196	5	16	42	17	0	0	38	45	0	7	27	0	5	0	3341	110	169	68	0	13	3	0	0	2	0	8	12	0	0	7	0	
8	0	6	1	46	5	2	0	1	24	12	0	3	55	0	3	0	0	25122	1	0	0	7	1	0	0	0	9	10	0	0	0	53	0	
83	0	52	7	28	3162	22	0	1	68	18	0	8	40	0	22	0	26	206	3959	20	1	19	2	0	0	0	0	13	29	0	0	29	0	
77	39	212	11	0	19	6	2	3	31	15	14	3	7	1	40	1	235	18	434	3252	3	3	1	0	0	14	0	32	12	1	3	3	9	
187	9	16	4	0	4	2	4	2	3	1	7	0	1	0	35	2	1	2	7	19	769	0	0	0	0	15	0	3	1	32	0	0	0	0
4	0	1	0	9	10	2	0	0	19	10	0	2	0	0	1	0	0	85	0	0	0	5488	3	0	0	0	0	9	2	0	0	58	0	
4	0	4	0	3	1	3	0	3	7	9	0	2	1953	2	0	0	0	85	0	1	0	15	1617	0	0	3	0	23	1	0	0	10	1	
1	1	4	0	0	1	0	16	0	3	414	0	0	0	0	0	0	40	0	3	0	2	0	0	0	0	0	0	0	0	0	0	0	0	
36	2	1	3	5	0	0	0	2	18	0	0	0	0	1	0	8	0	0	3	1	1	0	0	0	2	0	7	36	1	0	0	0	0	
182	0	44	8	9	17	35	0	14	115	57	0	9	77	15	3	0	16	245	2	5	0	29	146	0	0	2	5861	0	363	37	0	0	35	5
90	2	4	0	0	0	0	0	0	0	0	1	39	0	0	4	0	0	0	0	8	1	0	0	0	0	3	0	1	0	8	0	0	1	
93	0	33	0	1	7	23	0	32	42	44	0	12	5	21	3	0	16	4375	21	0	0	67	37	0	0	34	0	8082	29	0	1	18	1	
8	0	8	2	19	6	3	0	0	23	9	0	4	18	0	2	0	1	70	1	2	0	4	2	0	0	0	6	2993	0	0	10	12		
42	485	4	1	0	0	0	2	0	2	0	4	1	0	0	14	1	0	0	0	4	0	0	0	0	0	13	0	39	1	1795	6	0	2	
2	4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	378	0	1	
0	0	0	0	7	3	0	0	0	17	1	0	1	1	0	0	0	0	83	0	0	0	21	5	0	0	0	2	3	0	0	6389	0	0	
0	2	0	0	0	0	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	484	

0.8242544054985046

Kernel: (16, 2), Adamax, cosine_proximity

8556	0	99	14	99	70	65	4	1	98	99	0	12	100	0	16	0	33	288	47	13	1	43	2	0	0	14	0	13	36	0	0	65	1	
2	5899	1	5	0	0	0	0	0	1	5	1	2	0	0	0	0	0	0	0	2	1	0	0	1	0	0	17	0	309	1	2	3	0	1
31	0	5365	25	52	8	21	0	0	31	30	0	7	51	1	16	0	72	132	9	19	0	20	2	0	0	1	0	25	7	0	0	21	1	
10	0	8	716	4	7	9	0	2	6	13	0	6	5	1	0	0	1	25	3	0	1	3	0	0	0	4	0	8	2	0	0	5	0	
1	0	1	2	5233	0	1	0	1	0	1	0	2	10	0	0	0	0	17	0	0	0	1	0	0	0	0	0	0	1	0	0	4	0	
9	0	6	7	31	3662	10	0	1	15	5	0	4	17	0	11	0	18	73	636	0	3	9	2	0	0	1	1	2	2	0	0	30	0	
16	0	23	1	25	12	3633	0	0	23	27	0	5	14	3	0	0	2	77	9	1	0	8	1	0	0	1	0	3	3	0	0	9	0	
0	10	2	4	0	0	0	652	0	1	401	4	0	1	1	0	14	0	0	1	0	0	0	0	0	0	3	0	4	0	0	3	0	1	
17	1	5	2	1557	9	8	0	1170	9	19	0	2	9	0	2	0	2	50	6	1	0	6	1	0	5	2	0	2	3	0	0	7	0	
20	0	24	14	26	10	10	0	8	7240	23	1	2	31	0	8	2	0	6	124	14	6	0	13	1	0	0	3	0	9	9	0	0	41	0
10	0	26	8	45	14	9	2	0	21	7720	0	8	42	0	0	0	27	132	18	0	0	7	0	0	0	2	1	1	5	0	0	29	0	
0	15	0	2	2	0	1	0	0	22	1	1105	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	
5	0	6	5	14	2	5	0	0	2	11	0	1954	18	0	3	0	0	40	4	2	0	1	0	0	0	0	4	3	0	0	6	0	0	
1	0	9	3	23	1	5	1	0	4	6	0	1	5305	0	0	0	0	35	1	1	0	10	6	0	0	0	2	2	0	0	21	0		
22	0	14	9	3	6	1657	1	4	11	25	0	3	6	1676	0	0	7	71	11	0	1	20	7	0	0	10	0	4	3	0	0	8	0	
5	15	255	13	0	3	1	3	0	1	7	0	2	0	0	833	0	7	2	1	0	1	1	0	0	0	6	0	15	0	1	0	1	0	
8	10	5	23	0	4	1	0	0	2	4	1	4	2	1	0	250	4	3	5	1	1	1	0	0	1	7	0	16	285	0	0	0	2	
3	0	53	6	15	35	9	0	0	16	23	0	8	27	0	3	0	3591	63	223	42	3	6	1	0	0	2	8	0	0	0	9	0	0	
14	0	19	6	79	9	13	0	0	25	25	0	6	109	0	3	0	8	24941	9	4	0	9	3	0	0	0	4	11	0	0	0	72	0	
23	0	30	11	38	2980	13	0	1	12	11	0	8	44	0	13	0	36	135	4397	12	2	6	2	0	0	1	0	0	4	0	0	36	0	
28	29	109	21	3	19	3	1	3	10	15	11	5	7	1	22	0	332	14	615	3208	13	2	1	0	0	8	1	7	4	0	3	3	3	
30	9	0	11	0	3	2																												

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

8579	0	99	2	99	58	92	0	2	175	33	0	10	74	0	2	0	49	271	30	8	0	37	3	0	0	24	0	26	59	0	0	57	0
1	6025	2	0	0	0	0	0	0	2	1	2	0	0	0	0	0	0	0	0	0	0	0	1	0	0	16	0	202	0	0	0	0	1
34	0	5367	3	47	4	31	1	0	54	4	0	5	43	0	2	0	113	131	3	14	0	17	2	0	0	3	0	31	17	0	0	21	0
15	2	27	616	4	5	16	0	0	22	6	0	15	4	1	0	0	8	23	5	0	0	3	1	0	0	16	0	41	5	0	0	4	0
0	0	0	2	5251	0	0	0	0	1	0	0	1	6	0	0	0	0	11	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0
15	0	15	1	35	3726	16	0	0	36	1	0	5	18	0	5	0	36	79	507	0	0	14	4	0	0	6	0	5	8	0	0	23	0
11	0	23	0	20	7	3676	0	0	39	3	0	3	11	0	0	0	2	71	6	0	0	7	1	0	0	0	3	4	0	0	9	0	
15	22	21	1	0	0	2	603	0	3	312	7	1	2	1	1	2	74	0	2	0	2	0	1	5	0	10	0	11	0	0	4	0	1
14	3	8	0	1641	5	10	0	1081	30	6	0	1	9	0	0	0	5	42	5	1	0	7	1	0	4	6	0	3	7	0	0	6	0
9	0	11	1	20	2	4	0	3	7463	0	0	1	12	0	0	0	5	60	3	1	0	7	0	0	0	1	0	5	6	0	0	23	0
63	0	247	20	81	18	69	1	1	148	6761	0	21	130	0	0	0	232	178	26	0	1	13	9	0	0	16	0	18	33	0	0	41	0
0	11	1	1	1	0	0	0	0	48	0	1081	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	1
7	0	14	5	16	2	8	0	0	6	3	0	1928	18	0	1	0	4	34	2	1	0	5	1	0	0	3	1	16	6	0	0	4	0
3	1	14	2	24	1	8	0	0	18	0	0	1	5271	0	0	0	2	40	1	1	0	5	12	0	0	1	0	2	6	0	0	24	0
19	0	20	1	1	7	2363	0	4	22	10	0	3	2	995	0	0	10	61	6	0	0	13	9	0	0	12	0	9	6	0	0	6	0
11	18	424	2	0	0	1	1	0	7	2	0	2	1	0	657	0	12	3	0	2	1	0	1	1	0	12	0	15	0	0	0	0	0
10	8	4	2	0	3	0	0	0	2	1	0	0	0	0	0	373	4	2	2	0	0	1	0	0	1	6	0	13	208	0	0	0	1
5	0	49	3	16	21	15	0	0	24	3	0	7	23	0	1	0	3777	55	89	29	0	5	2	0	0	0	3	13	0	0	7	0	
16	0	35	5	97	5	13	0	0	97	6	97	13	94	0	1	0	9	24860	4	3	0	9	2	0	1	0	0	20	26	0	0	53	0
34	0	34	4	47	3044	24	0	1	54	4	0	9	41	0	6	1	100	146	4196	2	1	16	3	0	0	1	0	2	17	0	0	28	0
38	46	117	8	3	16	8	0	2	25	4	11	3	7	0	12	1	1021	12	306	2792	17	2	1	3	0	19	1	8	9	0	3	3	
52	15	8	2	0	2	4	2	2	6	0	2	1	1	0	3	3	5	2	9	12	930	0	0	3	0	42	0	3	3	13	1	0	0
12	0	7	0	13	9	8	0	0	26	24	0	2	1	0	1	0	1	86	0	0	0	5365	8	0	0	0	0	9	5	0	0	126	0
3	1	5	0	3	3	4	0	1	10	1	0	2	1962	1	0	1	3	51	0	0	0	9	1666	0	0	3	0	8	4	0	0	6	0
0	1	1	0	0	0	0	0	0	0	8	0	0	0	0	0	0	4	0	0	0	0	0	0	0	470	0	1	0	0	0	0	0	
7	1	3	2	10	0	0	0	2	3	0	0	0	0	0	0	12	0	0	1	0	1	0	0	0	70	2	0	3	9	1	0	0	0
107	3	74	8	16	13	40	0	7	82	34	0	9	154	8	0	1	33	184	3	1	0	21	188	0	1	6206	0	96	15	0	0	25	0
6	5	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	1	0	0	1	2	0	0	0	0	1	136	0	1	1	0	0	0
141	0	105	7	12	14	46	0	25	54	34	0	23	15	33	1	0	49	2929	38	0	1	63	43	0	2	125	0	9173	43	0	0	21	0
6	0	4	1	20	9	9	0	0	27	3	0	6	16	0	3	6	3	48	5	1	0	2	1	0	0	0	1	3022	0	0	7	3	
22	1242	4	1	0	0	0	2	0	2	0	3	0	0	0	4	2	1	0	0	10	1	0	0	0	0	42	1	4	1	1067	5	0	2
0	6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	379	0	0
0	0	1	2	12	2	0	0	0	19	0	0	1	3	0	0	0	2	69	0	0	0	5	13	0	0	3	0	3	4	0	0	6394	0
0	3	3	0	0	0	0	0	0	2	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	475

0.8311942458152771

Kernel: (16, 2), Nadam, categorical_hinge

8500	0	158	0	66	74	131	0	4	227	51	0	8	45	7	11	1	26	250	31	20	4	36	1	5	0	18	0	13	42	0	0	59	1
8	5830	11	0	0	0	0	0	3	13	1	0	1	0	2	3	1	0	0	2	0	0	0	0	0	0	38	0	301	3	36	0	0	0
24	1	5439	0	41	8	28	0	2	65	15	0	3	24	3	28	0	63	110	1	28	1	19	1	5	0	1	5	8	0	0	23	1	
28	1	163	0	6	9	144	9	0	191	9	0	15	4	30	1	1	3	22	22	2	10	0	1	0	0	74	0	85	3	0	0	5	1
2	0	4	0	5229	2	0	0	3	1	0	0	2	3	0	0	0	0	22	0	0	1	2	0	0	0	0	0	0	1	0	0	3	0
17	0	12	0	21	3889	20	0	0	41	2	0	5	12	0	11	1	14	54	401	2	20	6	1	1	0	4	0	2	3	0	0	16	0
10	0	19	0	17	12	3686	0	1	49	4	0	4	4	10	0	0	0	54	1	2	0	8	0	2	0	0	2	2	0	0	9	0	
3	17	18	0	0	0	1	0	1	13	362	0	3	0	8	2	0	13	0	3	0	1	0	0	0	432	0	15	0	3	1	0	0	207
10	2	12	0	961	14	11	0	1790	17	10	0	1	2	3	1	0	0	39	1	2	0	5	1	0	0	1	0	2	3	0	0	7	0
11	0	29	0	16	1	2	0	15	7461	3	0	0	7	0	0	0	4	58	2	7	3	0	4	0	0	0	0	1	2	0	0	18	0
51	0	82	0	54	20	80	0	1	163	7188	0	11	30	4	3	0	46	133	25	0	0	18	0	131	0	11	0	6	27	0	0	34	9
0	7	0	0	1	0	0	0	46	1093	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	2	0	0	0	0
12	0	21	0	16	2	6	0	0	11	8	0	1920	12	1	3	1	1	34	2	5	1	5	0	0	0	4	0	7	8	0	0	5	0
23	0	40	0	31	2	15	0	0	39	5	0	3	5092	4	1	0	2	51	4	7	4	6	32	0	0	24	0	5	10	0	0	37	0
10	0	21	0	1	3	1047	0	3	22	10	0	1	1	2374	0	0	3	37	2	0	0	14	7	1	0	8	0	4	3	0	0	6	0
4	12	336	0	0	3	1	0	0	9	1	0	1	0	3	771	0	5	3	1	7	1	0	1	5	0	2	0	4	0	2	0	0	1
6	3	7	0	0	1	3	0	0	9	1	0	2	0	0	0	417	2	2	3	1	0	0	0	0	0	3	0	13	168	0	0	0	0
8	0	113	0	12	67	23	0	0	41	15	0	6	18	0	3	0	3519	47	179	48	3	8	3	10	0	3	2	9	0	0	8	2	
26	0	54	0	72	29	45	0	0	142	22	0	12	81	3	6	0	12	24710	22	14	2	15	8	0	0	3	0	17	17	0	0	57	0
47	0	63	0	23	4064	49	0	0	100	9	0	9	22	1	14	5	31	108	3164	16	26	8	1	4	0	1	0	2	17	0	0	30	1
41	33	192	0	1	26	15	0	4	48	10	0	5	4	11	17	4	309	9	619	3056	51	2	0	13	0	9	0	3	14	1	0	3	1
23	7	11	0	0	2	8	0	3																									

Kernel: (16, 2), Nadam, logcosh

8604	0	103	4	59	77	58	0	7	182	25	0	6	63	1	3	0	26	250	12	12	0	46	1	0	0	6	0	11	165	0	0	67	1	
1	6040	1	0	0	0	0	0	0	2	1	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	5	0	197	1	1	0	0	1	
40	0	5278	8	49	9	18	0	6	83	6	0	5	53	1	15	0	87	117	2	47	0	17	5	0	0	3	0	38	29	0	0	30	1	
16	0	9	661	3	8	9	0	3	20	8	0	1	7	1	1	0	1	22	1	0	0	3	0	0	0	6	0	30	22	1	0	6	0	
2	0	1	2	5232	1	1	0	0	2	0	0	1	11	0	0	0	1	7	0	0	0	1	0	0	0	0	0	0	7	0	0	6	0	
13	0	10	3	19	4165	14	0	1	37	3	0	4	13	0	9	0	14	55	122	5	0	12	0	0	1	0	2	24	0	0	29	0		
21	0	26	0	18	14	3602	0	6	58	6	0	3	14	3	0	0	2	69	3	1	0	8	1	0	0	0	9	20	0	0	12	0		
8	21	10	3	0	2	1	830	3	3	160	5	1	1	1	2	2	24	0	0	2	1	0	1	0	1	0	11	3	1	5	0	1		
17	1	1	0	803	12	3	0	1980	14	3	0	1	6	0	0	0	1	20	2	1	0	4	1	0	0	1	0	1	14	0	0	8	1	
10	0	2	2	16	3	2	0	16	7466	0	0	1	24	0	0	0	3	30	1	2	0	4	0	0	0	1	0	1	20	0	0	33	0	
92	0	200	11	60	37	35	1	10	251	6746	0	14	82	0	6	1	143	160	13	1	1	19	4	0	0	9	1	17	163	0	0	50	0	
0	13	0	2	1	0	0	1	2	71	0	1059	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	
11	0	13	7	16	5	5	0	0	15	5	0	1884	16	0	2	0	2	46	2	5	0	4	0	0	3	0	25	11	0	0	8	0		
7	0	10	2	20	1	5	0	0	17	2	0	1	5245	0	0	0	1	52	1	1	0	11	1	0	0	3	0	5	14	0	0	38	0	
26	1	15	3	2	10	1516	0	10	22	12	0	3	6	1801	0	0	9	61	2	0	1	19	8	0	0	14	0	16	11	0	0	11	0	
10	19	332	2	0	2	1	1	0	9	2	0	3	0	0	751	0	6	0	0	4	0	3	0	0	0	5	0	16	3	2	0	2	0	
3	8	2	1	0	2	0	0	0	1	0	1	0	0	0	0	327	2	2	1	2	0	1	0	0	0	2	0	6	278	0	0	0	2	
9	0	63	3	12	91	14	0	0	34	9	0	7	23	0	2	0	3613	57	64	99	0	3	1	0	0	2	0	3	30	0	0	8	0	
25	0	26	5	104	25	18	0	0	173	9	0	4	73	0	2	0	8	24624	8	7	0	14	1	0	0	0	20	92	0	0	131	0		
40	0	43	7	21	5547	16	0	2	69	5	0	6	31	0	10	0	41	119	1745	24	1	14	1	0	0	1	0	1	41	0	1	29	0	
43	49	108	10	0	30	4	0	3	24	6	15	2	6	1	17	4	367	12	205	3528	10	2	0	0	0	10	0	8	25	2	3	3	4	
61	10	4	3	0	5	2	5	3	4	0	5	0	1	2	0	32	6	2	1	4	22	877	0	0	0	0	11	1	2	3	61	1	0	0
5	0	1	0	8	12	5	0	7	23	12	0	2	0	0	1	0	1	61	0	0	0	5288	1	0	0	0	1	11	0	0	264	0		
6	0	1	0	3	3	3	0	6	10	2	0	2	2453	1	0	1	3	61	0	0	0	17	1120	0	0	8	0	10	9	0	0	27	1	
2	2	1	0	0	1	0	53	0	1	195	3	0	0	0	0	0	141	0	0	0	84	0	0	0	0	0	0	2	0	0	0	0		
28	2	2	2	11	0	0	0	4	10	0	0	0	0	0	0	18	1	1	3	2	2	0	0	0	0	0	5	36	0	0	0	0		
201	2	63	27	16	37	27	1	17	107	37	0	12	233	14	2	1	26	194	6	4	0	32	141	0	0	5805	0	144	113	0	0	65	2	
11	3	0	0	0	0	0	0	1	0	0	0	4	0	0	0	2	0	0	0	1	1	0	0	0	0	0	135	1	0	3	0	0	0	
179	0	72	8	7	37	35	0	59	100	33	0	13	14	38	3	0	39	2949	33	0	1	82	34	0	0	74	0	9019	116	0	1	50	1	
2	0	1	1	7	1	1	0	0	14	0	0	3	9	0	2	1	0	20	0	1	0	1	1	0	0	0	0	3126	0	0	6	6		
6	400	2	0	0	0	0	2	0	2	0	3	0	0	0	2	1	0	0	0	7	0	0	0	0	1	0	2	2	1982	2	0	2	0	
0	5	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	2	5	3	0	0	0	11	0	0	0	1	0	0	0	0	22	0	0	0	4	0	0	0	0	0	1	8	0	0	6476	0	
0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	483	0	

0.8276382863521576

ARat_2017_04_05_0_002.dat

Kernel: (4, 2), RMSprop, mean_squared_error

3189	28	25	18	25	8	8	191	25	5	114	4	7	19	127	29	27	0	0	9	32	14	41	1	29	244	10	26	19	21	1	11	0	
3	5518	5	15	19	4	2	26	23	2	29	0	3	4	25	4	3	0	0	5	9	27	17	0	6	96	6	0	2	6	0	28	0	
0	1	4500	4	2	2	0	6	5	0	8	0	1	2	1	1	0	0	0	3	1	10	0	0	2	11	5	0	1	0	0	6	0	
1	19	3	4928	9	378	0	11	7	0	30	0	3	1	15	4	1	0	0	6	0	21	23	0	1	43	4	2	0	7	0	6	0	
12	58	34	50	10648	27	9	94	44	9	73	1	11	13	123	15	10	0	0	45	31	70	54	0	25	295	7	16	10	33	0	25	3	
0	16	8	1298	7	1541	0	3	6	1	34	0	0	1	4	2	0	0	0	2	2	16	17	0	0	29	3	0	2	4	0	3	0	
3	12	6	21	13	8	4017	11	10	1	50	2	4	5	28	18	15	0	0	3	57	160	19	0	6	50	7	1	30	10	1	0	5	
11	85	31	59	70	26	8	9400	61	7	192	2	7	55	266	39	45	0	0	30	88	69	99	0	28	825	38	10	26	60	0	20	0	
0	7	4	12	9	7	0	8	5625	3	10	0	5	3	7	1	0	0	0	3	2	23	7	0	4	221	3	0	1	4	0	10	0	
0	1	0	0	23	1	0	0	1	1214	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	2	5	0	0	
1	23	19	19	26	32	8	26	14	3	10284	0	8	3	26	7	4	0	0	28	12	74	54	0	9	122	3	1	1	7	1	14	0	
4	23	11	13	21	6	11	33	23	1	35	1258	4	45	28	11	11	0	0	2482	25	19	21	0	8	66	1	9	13	10	0	0	0	
1	5	17	4	2	4	1	18	15	6	11	0	5145	0	10	3	1	0	0	8	6	13	7	0	0	40	4	2	4	0	1	93	0	
4	19	2	16	21	4	3	41	14	0	32	7	3	3029	26	11	5	0	0	58	17	15	17	0	11	122	6	1	1	10	4	9	0	
6	60	32	59	37	22	10	167	67	3	168	0	16	24	9197	16	15	0	0	45	39	70	56	0	35	530	17	9	44	39	2	24	1	
2	28	7	20	9	6	1	8	17	3	34	0	4	12	16	3867	3	0	0	8	4	26	16	0	2	157	10	5	2	12	0	6	0	
1	22	23	48	28	7	7	58	45	1	37	2	9	27	44	19	5035	0	0	4	15	35	38	0	16	69	155	7	9	14	0	10	0	
1	0	0	0	2	0	1	6	0	0	8	1	0	1	7	0	0	1125	3	0	1	0	1	2	0	1	0	0	192	8	0	30	1	1
10	1	0	0	1	1	4	28	1	0	6	4	0	86	20	2	10	5	1649	0	1	0	3	0	4	3	0	3	19	0	50	1	0	
5	8	8	21	13	4	4	22	16	5	41	26	7	10	33	1	3	0	0	4541	6	50	11	0	5	72	1	7	1	4	4	16	0	
4	17	8	32	36	17	13	28	19	2	71	1	5	6	26	24	10	0	0	11	7096	2096	62	0	5	77	11	4	13	16	1	5	1	
0	8	9	13</																														

Kernel: (4, 2), RMSprop, mean_squared_logarithmic_error

3326	30	14	23	55	10	5	120	36	6	67	11	19	34	100	27	25	0	0	14	40	13	38	1	44	106	5	18	23	15	0	82	0
2	5510	3	24	37	7	0	15	28	2	24	1	9	6	22	2	2	0	0	12	8	24	15	0	7	39	1	0	3	2	0	82	0
0	1	4436	8	16	8	0	7	13	0	10	0	12	3	3	2	0	0	0	7	5	8	1	0	4	12	6	1	1	0	0	8	0
1	12	2	4870	19	482	0	2	8	0	20	0	12	1	12	3	1	0	0	6	1	18	23	0	2	14	2	1	0	3	0	8	0
5	21	12	24	11441	21	3	11	35	6	28	5	22	1	19	4	3	0	0	37	12	30	25	0	18	23	2	1	0	9	0	27	0
0	12	4	1187	15	1704	0	0	7	1	13	0	2	1	4	1	0	0	0	2	2	12	15	0	0	8	1	0	2	3	0	3	0
7	18	7	46	76	9	3470	13	40	5	62	14	7	4	53	26	17	0	0	5	189	316	37	0	21	44	6	5	63	9	0	3	1
19	98	25	105	196	37	2	9061	135	9	194	11	40	109	375	36	45	0	0	52	114	71	100	0	50	516	31	7	55	52	0	112	0
0	6	2	13	13	8	0	5	5760	6	7	2	14	4	5	2	0	0	0	3	2	15	6	0	6	79	2	0	2	3	0	14	0
0	1	0	0	22	1	0	0	0	0	1218	0	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	4	0
2	27	16	29	68	40	1	23	34	3	10189	3	24	8	34	7	5	0	0	48	15	68	61	0	20	75	0	1	1	4	0	23	0
2	19	3	12	33	6	1	9	24	1	18	1978	13	10	9	5	6	0	0	1961	25	14	15	0	7	7	1	0	3	7	0	3	0
0	2	0	2	1	1	0	4	5	4	1	0	5366	0	1	1	0	0	0	4	2	1	4	0	0	2	0	0	0	0	0	20	0
4	15	2	22	55	4	1	17	21	1	23	37	13	3037	23	10	2	0	0	94	21	8	13	0	11	39	5	2	1	6	1	20	0
7	58	23	95	114	24	3	118	134	4	151	4	79	32	9147	16	12	0	0	56	59	66	56	0	63	295	16	12	50	30	0	86	0
3	28	6	29	37	10	0	9	31	3	31	0	11	16	16	3838	3	0	0	14	8	29	17	0	13	96	7	4	2	12	0	12	0
2	24	12	91	60	10	0	51	71	2	35	5	26	32	57	22	4971	0	0	11	19	33	38	0	26	34	72	5	17	12	0	47	0
2	0	0	1	3	0	0	2	0	0	7	1	1	1	8	0	0	1182	4	0	2	0	1	1	0	1	0	144	9	0	21	1	0
7	2	0	0	4	1	3	10	1	0	1	8	0	43	10	2	8	2	1763	0	0	0	2	0	3	0	0	1	10	0	28	3	0
2	6	5	19	21	3	2	5	18	7	20	159	19	4	9	1	2	0	0	4565	6	28	8	0	3	14	0	0	1	2	0	16	0
4	20	7	49	68	21	1	16	25	3	66	3	12	9	26	16	10	0	0	17	7916	1268	63	0	12	33	9	3	13	16	1	10	0
0	11	6	16	30	12	1	4	18	11	24	0	19	1	8	1	4	0	1	21	1869	4798	22	0	7	21	6	1	3	2	0	25	0
2	31	21	25	61	12	0	15	23	2	61	2	25	9	23	9	3	0	0	31	14	63	7147	0	21	93	12	3	4	5	0	36	0
8	1	0	2	5	2	4	7	3	0	2	4	1	1	6	614	22	1	2	0	0	0	1	507	3	3	0	2	2	1	6	11	0
1	12	4	5	41	7	3	16	16	3	19	1	27	3	7	9	2	0	0	16	10	32	17	0	5045	29	1	0	2	2	0	7	0
6	34	7	69	78	18	1	37	443	3	90	1	21	21	51	7	9	0	0	43	23	74	60	0	20	6927	8	6	7	21	0	48	0
0	2	5	11	2	4	0	2	7	2	11	0	5	5	4	3	2	0	0	2	1	13	7	0	4	16	1455	0	0	1	0	7	0
3	9	9	95	32	3	1	46	41	2	43	3	12	18	69	4	4	0	0	8	11	8	18	0	19	91	4	1891	18	6	0	41	0
19	87	26	153	191	43	6	100	200	10	203	7	93	58	328	36	40	0	0	75	62	81	105	0	58	400	19	25	13395	31	0	113	1
5	54	8	53	63	7	0	49	37	4	62	4	18	43	51	13	14	0	1	29	106	66	47	0	26	99	6	3	5	5671	0	47	3
55	8	1	41	6136	22	25	65	40	6	60	15	22	16	145	34	35	1	3	2	49	0	22	0	41	26	2	39	237	17	7850	35	13
1	27	3	16	22	4	0	11	16	7	28	0	55	6	11	7	4	0	0	11	22	19	11	0	14	38	2	0	3	5	0	3452	0
3	10	1	5	14	7	0	14	4	0	16	2	0	8	4	6	7	0	0	2	5	0	6	0	12	13	0	4	11	1148	0	7	120

0.8245160624434852

Kernel: (4, 2), Adamax, cosine_proximity

3813	20	16	9	17	10	10	74	13	2	21	4	10	4	51	19	29	0	0	8	17	15	24	1	21	37	3	15	18	14	0	12	0
9	5599	3	12	24	18	2	23	9	1	13	1	3	1	25	2	4	0	0	6	4	31	22	0	3	18	0	0	2	6	0	46	0
2	2	4500	3	2	8	1	8	3	0	6	0	10	1	2	1	0	0	0	4	0	10	0	0	2	2	2	0	1	0	0	2	0
3	14	2	3852	14	1488	0	8	7	0	13	0	12	0	12	2	1	0	0	15	0	36	21	0	2	8	1	2	0	8	0	2	0
41	53	29	29	10982	45	9	68	47	9	39	5	25	3	77	10	22	0	0	50	14	78	50	0	28	59	1	9	12	31	0	20	0
0	8	5	494	11	2408	0	2	4	1	5	0	2	1	5	0	1	0	0	3	2	18	13	0	0	4	1	0	3	6	0	2	0
10	10	4	10	13	11	4058	10	7	1	23	6	6	0	22	8	18	0	0	3	49	229	14	0	7	8	2	1	37	5	1	0	0
50	93	30	56	91	49	10	9876	72	8	104	3	20	21	282	25	61	0	0	52	82	105	93	0	39	262	15	11	48	61	0	38	0
0	14	7	16	11	12	0	15	5678	6	11	1	22	2	11	3	6	0	0	9	1	35	9	0	7	84	1	0	5	9	0	4	0
0	1	0	0	29	1	1	0	0	1203	0	0	5	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	8	0	0	0
27	56	20	27	53	68	10	64	22	3	9927	2	24	3	92	9	11	0	0	49	7	101	77	0	23	91	1	8	14	20	1	19	0
11	18	7	5	18	7	8	21	12	1	15	1080	6	1	15	5	10	0	0	2849	17	28	16	0	10	6	1	6	9	8	0	2	0
1	3	10	2	0	4	0	5	1	5	3	0	5271	0	8	2	1	0	0	6	5	5	5	0	0	5	0	1	4	0	1	73	0
45	32	8	20	24	11	4	80	24	0	30	36	10	2617	82	12	24	0	1	213	14	22	21	0	18	74	4	14	12	29	5	22	0
44	61	33	58	58	39	18	169	75	3	104	0	46	13	9404	11	23	0	0	60	32	96	55	0	52	173	11	16	80	41	1	33	1
7	37	7	21	18	16	1	22	24	3	25	1	11	9	27	3810	7	0	0	17	6	53	21	0	12	89	4	5	12	14	0	6	0
2	18	15	22	24	17	5	52	33	1	19	0	19	4	40	8	5351	0	0	9	10	36	29	0	10	14	18	5	8	8	0	8	0
12	0	0	0	3	0	1	5	0	0	7	2	1	0	6	0	0	1142	3	0	2	0	1	2	0	1	0	164	13	0	25	1	1
45	2	0	1	1	2	5	22	1	0	4	15	0	37	15	1	13	3	1634	4	1	0	2	1	9	1	0	9	28	0	55	1	0
7	9	5	8	12	12	5	10	10	2	14	24	23	1	12	0	3	0	0	4704	4	43	10	0	2	10	0	2	2	4	4	3	0
17	17	8	22	43	23	18	24	13	2	45	2	11	3	27	17	16	0	0	12	5872	3388	65	0	6	9	4	3	25	19	1	5	0
1	11	9	9	14	11	41	5	6	8	10	0	30	0	5	1	4	0	0	12	531	6186	23	0	5	3	1	1	4	1	0	10	0
9	39	28	12	36</																												

Kernel: (16, 2), RMSprop, mean_squared_logarithmic_error

3849	4	22	6	12	8	6	58	3	5	20	1	8	1	52	18	18	0	0	5	40	16	47	1	20	17	8	15	18	5	3	21	0	
31	4536	8	33	44	20	2	89	10	6	57	1	3	7	116	12	9	0	0	19	80	91	110	0	20	50	10	5	32	8	1	477	0	
0	1	4535	1	2	2	0	3	0	1	2	0	3	0	2	0	0	0	0	2	4	0	0	0	2	3	0	1	0	0	8	0		
3	3	5	4147	11	1184	0	9	1	0	21	0	9	0	17	3	1	0	0	9	2	40	32	0	2	3	4	1	1	7	0	8	0	
44	16	40	37	10844	39	5	83	13	28	54	1	19	2	104	11	10	0	0	48	73	95	82	0	32	43	14	9	22	22	2	53	0	
3	4	10	620	10	2254	0	3	2	1	15	0	0	0	4	3	0	0	0	2	2	22	25	0	0	1	4	0	4	6	0	4	0	
13	5	7	13	14	10	3700	9	2	5	21	2	5	3	21	5	12	0	0	2	148	481	27	0	9	4	6	1	41	1	3	3	0	
67	34	38	49	71	36	4	9806	20	9	122	1	21	18	341	27	41	0	1	33	166	131	170	0	44	156	37	11	78	44	5	76	0	
1	8	13	23	17	11	0	43	5313	12	32	1	19	4	59	6	9	0	1	15	30	82	28	0	19	152	12	1	21	13	6	28	0	
0	1	0	0	6	1	0	0	0	1236	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	
34	10	33	17	42	47	6	54	5	5	10063	0	14	1	64	8	5	0	0	36	36	121	104	0	20	31	3	4	17	12	3	34	0	
20	10	14	8	22	8	6	33	4	1	25	458	8	6	25	7	13	0	0	3349	35	36	36	0	13	4	3	9	18	6	0	15	0	
1	1	11	3	0	1	0	7	1	6	2	0	5246	0	6	1	1	0	0	4	5	9	7	0	0	0	3	0	4	0	1	101	0	
55	10	9	23	33	10	0	101	8	2	41	4	7	2577	104	15	16	0	1	133	47	27	59	0	27	44	12	13	28	20	37	45	0	
57	21	46	46	51	30	7	162	30	11	119	0	39	11	9388	8	13	0	0	43	122	141	101	0	57	95	18	9	86	23	7	68	1	
10	10	7	19	10	12	1	25	8	4	28	0	9	8	40	3835	4	0	0	11	16	81	30	0	12	35	18	7	20	9	0	16	0	
3	7	32	34	31	13	3	65	10	1	29	0	22	6	70	15	4997	0	0	7	42	46	59	0	25	6	183	8	25	14	2	30	0	
12	0	1	0	1	0	0	2	0	0	6	1	1	1	4	0	0	1257	3	0	2	0	3	1	0	0	0	44	14	0	38	1	0	
41	1	0	1	0	2	3	17	0	0	5	1	0	20	20	1	13	4	1667	0	1	0	3	1	6	0	0	0	2	22	0	79	2	0
9	2	13	11	12	9	4	17	3	12	33	3	13	1	25	1	1	0	0	4617	19	59	16	0	7	11	3	5	5	3	3	28	0	
11	2	7	16	24	16	2	13	2	3	24	0	6	1	14	7	5	0	0	6	6759	2691	68	0	6	1	9	1	6	6	2	9	0	
0	0	8	7	10	8	5	3	2	12	5	0	10	0	3	1	1	0	0	2	916	5889	21	0	2	1	6	0	1	0	1	28	0	
3	5	31	7	21	12	1	15	1	3	22	0	15	1	11	5	3	0	0	14	12	69	7439	0	14	12	10	2	4	4	0	17	0	
15	0	0	1	1	2	4	5	0	0	2	0	0	1	5	182	15	0	1	0	0	0	1	952	0	1	0	1	7	0	18	7	0	
7	4	12	2	21	8	3	17	2	8	21	0	23	1	7	8	5	0	0	13	11	56	25	0	5057	6	4	0	4	2	0	10	0	
41	23	21	72	56	41	3	140	77	12	137	1	15	13	239	11	19	1	0	49	179	190	170	0	37	6305	28	15	92	33	8	105	0	
1	0	7	5	1	1	0	2	0	2	7	0	2	1	3	1	0	0	0	0	1	13	9	0	2	1	1505	0	0	0	0	7	0	
19	5	15	30	13	5	1	67	6	7	33	1	6	5	102	2	1	0	0	8	40	17	40	0	18	39	9	1948	35	3	31	0		
55	31	36	36	88	51	12	108	33	13	110	1	41	17	201	21	39	1	1	61	61	118	148	0	41	63	23	7	14461	25	0	60	2	
35	14	17	26	34	12	0	62	7	4	55	0	7	13	52	10	12	0	0	24	810	141	77	0	32	25	12	5	8	5035	5	54	6	
154	6	2	26	471	27	39	75	4	7	43	7	12	4	121	22	27	0	1	2	63	1	38	0	27	24	6	19	184	5	13605	25	16	
13	6	4	13	12	7	1	27	3	8	20	0	27	2	7	6	0	0	0	9	24	30	23	0	14	15	7	1	6	7	1	3485	0	
12	4	1	4	9	9	2	21	2	0	14	1	0	4	10	5	9	0	0	2	29	0	9	0	11	3	1	4	11	991	3	9	249	

0.8447892980291942

Kernel: (16, 2), RMSprop, categorical_crossentropy

3347	34	9	18	58	8	2	43	20	3	28	4	7	21	68	28	25	0	0	31	12	22	63	0	48	318	6	12	13	28	0	31	0	
4	5600	4	9	34	6	0	4	5	1	6	0	3	5	7	2	2	0	0	26	1	19	17	0	6	73	3	0	2	8	0	40	0	
0	2	4384	5	14	4	0	7	11	1	9	0	9	4	8	3	1	0	0	14	1	14	4	0	7	45	7	1	1	6	0	10	0	
1	23	2	4461	34	782	0	1	14	0	12	0	4	1	8	4	1	0	0	32	0	28	26	0	4	61	5	0	0	11	0	8	0	
4	24	13	13	11395	15	1	8	22	6	16	3	7	4	13	4	4	0	0	67	1	40	31	0	22	91	2	1	0	10	0	28	0	
0	20	5	861	23	1957	0	1	6	1	8	1	0	2	5	4	0	0	0	10	1	19	19	0	0	39	3	0	2	9	0	3	0	
10	24	3	26	56	12	3314	15	28	1	48	9	4	6	67	28	23	0	0	10	88	442	55	0	21	171	8	4	68	19	4	8	1	
34	154	25	73	249	41	0	6607	91	7	124	3	18	106	697	53	66	0	0	143	60	103	177	0	48	2433	51	19	55	120	0	100	0	
0	12	1	13	13	8	0	2	5538	4	4	0	3	3	4	1	0	0	0	15	1	19	6	0	7	299	2	0	1	8	0	15	0	
0	1	0	0	27	1	0	0	1	1213	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	5	0	
6	67	16	31	101	53	2	21	28	3	9484	3	11	13	74	12	7	0	0	88	5	96	94	0	34	494	6	6	10	27	0	37	0	
2	13	1	1	24	2	0	3	7	1	6	465	3	4	0	5	4	0	0	3581	5	22	10	0	7	18	2	0	1	5	0	0	0	
0	9	4	4	6	3	0	4	13	6	4	0	5159	0	7	2	1	0	0	29	3	11	7	0	1	29	4	1	2	0	1	111	0	
5	22	2	10	30	5	0	4	14	0	14	17	1	2834	17	11	3	0	0	281	6	19	15	0	11	147	6	1	0	13	1	19	0	
13	74	20	49	121	23	0	41	74	4	86	1	23	25	8686	16	15	0	0	122	16	94	80	0	54	979	16	10	48	58	0	61	1	
3	30	2	14	37	7	0	2	16	3	11	1	5	14	9	3795	3	0	0	20	2	35	21	0	11	207	11	1	4	12	0	9	0	
2	33	11	32	49	9	0	20	48	2	15	1	13	23	42	19	5004	0	0	31	11	33	39	0	26	91	170	6	5	24	0	26	0	
5	0	0	0	3	0	0	0	0	0	6	2	1	1	8	0	0	999	4	0	1	0	4	0	0	4	0	0	300	12	0	39	2	1
7	2	0	0	2	1	3	6	1	0	1	5	0	48	7	2	8	0	0	1768	4	0	0	4	0	3	3	0	0	10	0	23	4	0
0	5	1	1	7	2	0	1	3	9	2	2	1	1	2	0	0	0	0	4852	1	19	3	0	3	16	0	0	0	0	0	14	0	
13	33	7	30	83	17	1	17	23	3	60	2	6	12	28	18	11	0	0	41	4509	4361	76	0	26	235	12	6	15	57	1	14	0	
0	17	6	7	32	9	0	4	13	9	17	0	11	3	10	1	3	0	1	35	300	6277	23	0	13	101	7	2	1	9	0	31	0	
0	40	18	9																														

Kernel: (16, 2), Adamax, cosine_proximity

3528	24	20	11	31	7	8	109	7	1	44	1	10	16	110	39	18	0	0	18	30	13	42	3	60	57	2	20	26	28	3	21	0	
8	5499	6	16	30	9	1	21	8	0	20	0	2	6	39	7	4	0	0	17	10	32	25	0	16	23	0	0	5	12	2	69	0	
0	2	4507	3	4	3	1	7	1	0	5	0	8	1	3	2	0	0	0	4	2	3	0	0	3	3	1	0	1	4	0	4	0	
1	16	2	4863	14	421	0	8	11	0	29	0	13	0	20	9	1	0	0	21	2	34	28	0	4	11	1	2	0	9	0	3	0	
22	35	27	36	11009	25	8	49	25	1	50	3	25	7	97	17	7	0	0	60	23	58	55	0	47	65	2	5	11	45	0	29	2	
0	10	7	1196	16	1655	0	3	6	1	20	0	1	7	2	0	0	0	0	5	2	20	21	0	1	8	1	0	3	9	1	3	0	
5	11	6	14	22	8	3756	10	5	0	29	3	6	5	31	12	9	0	0	5	146	374	21	0	22	10	3	2	53	3	1	1	0	
27	88	30	67	91	36	7	9391	56	7	158	0	23	56	425	47	40	0	0	77	128	95	126	0	64	331	20	13	67	129	1	57	0	
0	15	10	19	21	10	0	14	5529	6	26	0	23	8	27	10	4	0	0	20	8	43	12	0	15	117	5	0	7	19	2	9	0	
0	2	0	0	44	1	1	0	0	0	1127	0	0	5	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	67	0	0	0
6	32	23	21	51	40	7	35	11	2	10115	1	21	6	77	13	6	0	0	55	18	84	73	0	26	43	1	3	10	24	3	22	0	
3	14	7	6	20	3	5	12	4	1	15	313	7	9	13	7	5	0	0	3648	18	21	18	0	17	6	1	1	4	11	0	3	0	
1	3	8	3	0	3	0	4	1	5	4	0	5254	0	8	2	1	0	0	4	4	4	5	0	3	3	0	1	4	0	0	96	0	
11	16	5	16	24	5	1	28	10	0	29	6	12	2931	38	13	4	0	2	182	21	16	22	0	18	35	3	5	5	31	2	17	0	
21	48	30	55	64	25	11	125	54	2	127	0	48	22	9407	23	12	0	0	70	53	86	62	0	108	167	10	9	56	71	4	39	1	
3	20	6	17	10	7	1	7	11	3	19	0	11	10	23	3993	2	0	0	15	3	30	20	0	14	33	4	1	4	12	0	6	0	
2	20	21	58	36	9	8	63	54	1	29	1	21	42	81	26	4943	0	0	22	35	40	44	0	61	22	49	12	21	37	0	27	0	
6	0	1	0	3	0	1	4	0	0	6	1	1	8	0	0	0	1233	4	0	2	0	1	2	3	2	0	71	11	0	29	1	1	
20	2	0	1	1	2	3	10	0	0	5	8	0	117	10	2	5	5	1618	13	1	0	3	0	18	1	0	0	12	0	54	1	0	0
2	5	5	7	9	4	3	9	2	0	13	5	23	1	14	0	1	0	0	4763	2	29	12	0	5	10	0	1	1	4	8	7	0	0
4	13	6	23	48	18	5	18	7	3	51	0	7	4	22	19	4	0	0	19	7038	2252	74	0	13	6	4	1	18	30	2	8	0	
0	11	10	8	20	10	16	5	4	7	24	0	28	1	9	2	1	0	0	23	1209	5491	24	0	7	4	4	0	3	7	1	13	0	
3	29	26	13	39	11	3	17	8	2	34	0	21	6	22	10	3	0	0	26	5	51	7322	0	25	35	6	1	6	15	0	14	0	
5	2	0	1	2	1	4	4	0	2	2	0	1	7	521	15	0	1	1	1	1	0	2	615	8	2	0	2	2	3	11	6	0	0
3	11	7	2	23	7	2	12	3	1	19	0	27	2	4	10	0	0	0	15	4	40	20	0	5100	9	1	0	2	10	0	3	0	
14	43	13	68	59	26	3	74	135	2	124	0	26	24	147	16	16	0	0	69	38	102	93	0	47	6836	9	8	43	62	0	36	0	
0	3	8	16	4	3	0	8	6	2	11	0	6	10	7	7	5	0	0	6	1	18	13	0	7	36	1377	1	0	7	0	9	0	
7	8	13	39	18	2	1	58	23	1	34	1	9	17	107	6	1	0	0	12	14	12	24	0	35	68	3	1929	25	20	1	21	0	
29	72	31	63	118	43	14	118	83	3	158	0	59	33	346	43	45	0	0	94	46	88	115	0	91	224	15	16	13907	55	0	43	13	
9	40	8	22	30	7	0	27	7	2	38	1	13	16	40	12	5	0	0	30	60	61	42	0	30	21	1	1	4	6031	1	20	15	
92	10	3	36	1585	21	45	83	22	0	55	11	18	14	200	47	32	4	2	6	38	0	31	0	110	53	2	38	189	31	12216	38	31	
6	38	4	22	22	7	1	22	12	5	30	0	31	6	43	12	9	0	0	15	26	20	18	0	35	36	2	2	7	26	1	3337	0	
3	8	1	2	8	6	1	9	2	0	15	1	0	4	1	6	4	0	0	2	5	0	7	0	10	3	0	1	6	1073	0	2	249	0

0.8441216136787667

Kernel: (16, 2), Nadam, mean_squared_logarithmic_error

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4306	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5887	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4572	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5523	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11845	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2999	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4573	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11657	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5979	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1249	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10829	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4192	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5421	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3508	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10810	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4285	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5785	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1392	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1912	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4945	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9717	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6942	0	0	0	0	0	0
0	0	0	0	0																														

2. AKTIVITÁSOK DETEKTÁLÁSA

A konfúziós mátrixok első sora a valós pozitív, második sora a negatív adatoknak felel meg; bal oszlopa a háló által pozitívnak, jobb oszlopa pedig negatívnak ítélt adatoknak.

2.1. Költségfüggvény/optimalizáló kombinációk eredményei

Együtt normált adatok esetén

RMSprop

mean_squared_error

7	117
11	238

0.6568364500999451

mean_absolute_error

32	92
27	222

0.6809651255607605

mean_absolute_percentage_error

30	94
30	219

0.667560338973999

mean_squared_logarithmic_error

36	88
38	211

0.6621983647346497

squared_hinge

93	31
64	185

0.7453083395957947

hinge

32	92
30	219

0.6729222536087036

categorical_hinge

15	109
16	233

0.6648793816566467

logcosh

54	70
41	208

0.7024128437042236

categorical_crossentropy

51	73
37	212

0.7050938606262207

binary_crossentropy

92	32
63	186

0.7453083395957947

kullback_leibler_divergence

50	74
38	211

0.6997318863868713

poisson

120	4
85	164

0.7613940834999084

cosine_proximity

105	19
78	171

0.7399463653564453

SGD

mean_squared_error

0	124
0	249

0.667560338973999

mean_absolute_error

0	124
0	249

0.667560338973999

mean_absolute_percentage_error

0	124
0	249

0.667560338973999

mean_squared_logarithmic_error

0	124
0	249

0.667560338973999

squared_hinge

0	124
0	249

0.667560338973999

hinge

0	124
0	249

0.667560338973999

categorical_hinge

0	124
0	249

0.667560338973999

logcosh

0	124
0	249

0.667560338973999

categorical_crossentropy

71	53
55	194

0.7104557752609253

binary_crossentropy

33	91
24	225

0.6916890144348145

kullback_leibler_divergence

0	124
0	249

0.667560338973999

poisson

19	105
17	232

0.6729222536087036

cosine_proximity

0	124
3	246

0.6595174074172974

Adagrad

mean_squared_error

6	118
9	240

0.6595174074172974

mean_absolute_error

31	93
33	216

0.6621983647346497

mean_absolute_percentage_error

15	109
13	236

0.6729222536087036

mean_squared_logarithmic_error

74	50
51	198

0.7292225360870361

squared_hinge

76	48
52	197

0.7319034934043884

hinge

0	124
0	249

0.667560338973999

categorical_hinge

0	124
0	249

0.667560338973999

logcosh

35	89
29	220

0.6836460828781128

categorical_crossentropy

47	77
40	209

0.6863270998001099

binary_crossentropy

116	8
80	169

0.7640750408172607

kullback_leibler_divergence

23	101
20	229

0.6756032109260559

poisson

0	124
5	244

0.6541554927825928

cosine_proximity

0	124
0	249

0.667560338973999

Adadelta

mean_squared_error

56	68
40	209

0.667560338973999

mean_absolute_error

30	94
28	221

0.6729222536087036

mean_absolute_percentage_error

0	124
0	249

0.667560338973999

mean_squared_logarithmic_error

91	33
67	182

0.7319034934043884

squared_hinge

98	26
71	178

0.7399463653564453

hinge

0	124
0	249

0.667560338973999

categorical_hinge

0	124
0	249

0.667560338973999

logcosh

124	0
94	159

0.7587131261825562

categorical_crossentropy

60	64
44	205

0.7104557752609253

binary_crossentropy

90	34
67	182

0.7292225360870361

kullback_leibler_divergence

71	53
53	196

0.7158176898956299

poisson

73	51
54	195

0.7184986472129822

cosine_proximity

124	0
86	163

0.7694370150566101

Adam

mean_squared_error

33	91
31	218

0.6729222536087036

mean_absolute_error

0	124
0	249

0.667560338973999

mean_absolute_percentage_error

32	92
32	217

0.667560338973999

mean_squared_logarithmic_error

87	37
62	187

0.7345844507217407

squared_hinge

68	56
50	199

0.7158176898956299

hinge

18	106
18	231

0.667560338973999

categorical_hinge

0	124
0	249

0.667560338973999

logcosh

59	65
42	207

0.7131367325782776

categorical_crossentropy

50	74
39	210

0.697050929069519

binary_crossentropy

62	62
43	206

0.7184986472129822

kullback_leibler_divergence

95	29
65	184

0.747989296913147

poisson

19	105
15	234

0.6782841682434082

cosine_proximity

57	67
42	207

0.707774817943573

Adamax

mean_squared_error

124	0
86	163

0.7694370150566101

mean_absolute_error

0	124
0	249

0.667560338973999

mean_absolute_percentage_error

0	124
0	249

0.667560338973999

mean_squared_logarithmic_error

34	90
28	221

0.6836460828781128

squared_hinge

95	29
67	182

0.7426273226737976

hinge

15	109
12	237

0.6756032109260559

categorical_hinge

3	121
0	249

0.6756032109260559

logcosh

119	5
82	167

0.7667560577392578

categorical_crossentropy

41	83
35	214

0.6836460828781128

binary_crossentropy

46	78
34	215

0.6997318863868713

kullback_leibler_divergence

73	51
52	197

0.7238605618476868

poisson

50	74
35	214

0.707774817943573

cosine_proximity

20	104
16	233

0.6782841682434082

Nadam

mean_squared_error

51	73
37	212

0.7050938606262207

mean_absolute_error

27	97
27	222

0.667560338973999

mean_absolute_percentage_error

0	124
0	249

0.667560338973999

mean_squared_logarithmic_error

104	20
75	174

0.7453083395957947

squared_hinge

41	83
33	216

0.6890080571174622

hinge

0	124
0	249

0.667560338973999

categorical_hinge

0	124
0	249

0.667560338973999

logcosh

43	81
34	215

0.6916890144348145

categorical_crossentropy

44	80
36	213

0.6890080571174622

binary_crossentropy

118	6
83	166

0.7613940834999084

kullback_leibler_divergence

48	76
39	210

0.6916890144348145

poisson

51	73
37	212

0.7050938606262207

cosine_proximity

59	65
43	206

0.7104557752609253

Külön normált adatok esetén

RMSprop

mean_squared_error

84	40
53	196

0.7506702542304993

mean_absolute_error

52	72
40	209

0.6997318863868713

mean_absolute_percentage_error

65	59
68	181

0.6595174074172974

mean_squared_logarithmic_error

57	67
45	204

0.6997318863868713

squared_hinge

53	71
48	201

0.6809651255607605

hinge

123	1
86	163

0.7667560577392578

categorical_hinge

123	1
86	163

0.7667560577392578

logcosh

45	79
36	213

0.6916890144348145

categorical_crossentropy

50	74
36	213

0.7050938606262207

binary_crossentropy

40	84
27	222

0.7024128437042236

kullback_leibler_divergence

45	79
35	214

0.6943699717521667

poisson

35	89
26	223

0.6916890144348145

cosine_proximity

48	76
34	215

0.7050938606262207

SGD

mean_squared_error

31	93
26	223

0.6809651255607605

mean_absolute_error

42	82
25	224

0.7131367325782776

mean_absolute_percentage_error

0	124
3	246

0.6595174074172974

mean_squared_logarithmic_error

96	28
66	183

0.747989296913147

squared_hinge

52	72
41	208

0.697050929069519

hinge

123	1
86	163

0.7667560577392578

categorical_hinge

75	49
53	196

0.7265415787696838

logcosh

57	67
40	209

0.7131367325782776

categorical_crossentropy

54	70
40	209

0.7050938606262207

binary_crossentropy

52	72
38	211

0.7050938606262207

kullback_leibler_divergence

55	69
40	209

0.707774817943573

poisson

47	77
33	216

0.7050938606262207

cosine_proximity

124	0
90	159

0.7587131261825562

Adagrad

mean_squared_error

52	72
40	209

0.6997318863868713

mean_absolute_error

114	10
80	169

0.7587131261825562

mean_absolute_percentage_error

46	78
24	225

0.7265415787696838

mean_squared_logarithmic_error

79	45
55	194

0.7319034934043884

squared_hinge

45	79
34	215

0.697050929069519

hinge

124	0
86	163

0.7694370150566101

categorical_hinge

83	41
54	195

0.7453083395957947

logcosh

55	69
40	209

0.707774817943573

categorical_crossentropy

28	96
25	224

0.6756032109260559

binary_crossentropy

66	58
43	206

0.7292225360870361

kullback_leibler_divergence

19	105
17	232

0.6729222536087036

poisson

55	69
44	205

0.697050929069519

cosine_proximity

104	20
72	177

0.7533512115478516

Adadelta

mean_squared_error

49	75
34	215

0.707774817943573

mean_absolute_error

124	0
86	163

0.7694370150566101

mean_absolute_percentage_error

42	82
25	224

0.7131367325782776

mean_squared_logarithmic_error

53	71
39	210

0.7050938606262207

squared_hinge

50	74
36	213

0.7050938606262207

hinge

123	1
86	163

0.7667560577392578

categorical_hinge

124	0
86	163

0.7694370150566101

logcosh

124	0
85	164

0.7721179723739624

categorical_crossentropy

51	73
38	211

0.7024128437042236

binary_crossentropy

33	91
20	229

0.7024128437042236

kullback_leibler_divergence

46	78
33	216

0.7024128437042236

poisson

53	71
38	211

0.707774817943573

cosine_proximity

46	78
39	210

0.6863270998001099

Adam

mean_squared_error

36	88
30	219

0.6836460828781128

mean_absolute_error

55	69
39	210

0.7104557752609253

mean_absolute_percentage_error

9	115
2	247

0.6863270998001099

mean_squared_logarithmic_error

67	57
45	204

0.7265415787696838

squared_hinge

55	69
38	211

0.7131367325782776

hinge

124	0
86	163

0.7694370150566101

categorical_hinge

124	0
87	162

0.7667560577392578

logcosh

55	69
43	206

0.6997318863868713

categorical_crossentropy

55	69
43	206

0.6997318863868713

binary_crossentropy

48	76
32	217

0.7104557752609253

kullback_leibler_divergence

77	47
43	206

0.7587131261825562

poisson

45	79
34	215

0.697050929069519

cosine_proximity

81	43
61	188

0.7211796045303345

Adamax

mean_squared_error

49	75
37	212

0.6997318863868713

mean_absolute_error

39	85
36	213

0.6756032109260559

mean_absolute_percentage_error

124	0
89	160

0.7613940834999084

mean_squared_logarithmic_error

44	80
36	213

0.6890080571174622

squared_hinge

42	82
29	220

0.7024128437042236

hinge

38	86
35	214

0.6756032109260559

categorical_hinge

117	7
84	165

0.7560321688652039

logcosh

48	76
36	213

0.6997318863868713

categorical_crossentropy

83	41
59	190

0.7319034934043884

binary_crossentropy

55	69
39	210

0.7104557752609253

kullback_leibler_divergence

46	78
32	217

0.7050938606262207

poisson

52	72
36	213

0.7104557752609253

cosine_proximity

44	80
34	213

0.6890080571174622

Nadam

mean_squared_error

49	75
36	213

0.7024128437042236

mean_absolute_error

0	124
0	249

0.667560338973999

mean_absolute_percentage_error

104	20
75	174

0.7453083395957947

mean_squared_logarithmic_error

52	72
38	211

0.7050938606262207

squared_hinge

54	70
40	209

0.7050938606262207

hinge

124	0
86	163

0.7694370150566101

categorical_hinge

124	0
86	163

0.7694370150566101

logcosh

55	69
41	208

0.7050938606262207

categorical_crossentropy

54	70
38	211

0.7104557752609253

binary_crossentropy

49	75
31	218

0.7158176898956299

kullback_leibler_divergence

81	43
46	203

0.7613940834999084

poisson

54	70
36	213

0.7158176898956299

cosine_proximity

54	70
40	209

0.7050938606262207

2.2. A tesztadatokra a legjobb hálóval kapott eredmények

Az egyes konfúziós mátrixok a tesztadatok egyes, 3000 időszeléből álló csomagjainak felelnek meg.

1000	0
683	1317
0.7723333239555359	
1000	0
674	1326
0.7753333449363708	
1000	0
682	1318
0.7726666927337646	
1000	0
690	1310
0.7699999809265137	
1000	0
688	1312
0.7706666588783264	
1000	0
646	1354
0.7846666574478149	
1000	0
652	1348
0.7826666831970215	
1000	0
714	1286
0.7620000243186951	
1000	0
672	1328
0.7760000228881836	
1000	0
662	1338
0.7793333530426025	
1000	0
678	1322
0.7739999890327454	
1000	0
680	1320
0.7733333110809326	
1000	0
638	1362
0.7873333096504211	
1000	0
624	1376
0.7919999957084656	
1000	0
662	1338
0.7793333530426025	
1000	0
650	1350
0.7833333611488342	
1000	0
620	1380
0.7933333516120911	
1000	0
622	1378
0.7926666736602783	
1000	0
718	1282
0.7606666684150696	
1000	0
692	1308
0.7693333625793457	
1000	0
660	1340
0.7799999713897705	
1000	0
626	1374
0.7913333177566528	
1000	0
648	1352
0.7839999794960022	
1000	0
624	1376
0.7919999957084656	
1000	0
664	1336
0.7786666750907898	
1000	0
610	1390
0.79666668176651	
1000	0
724	1276
0.7586666941642761	
1000	0
686	1314
0.7713333368301392	

1000	0
652	1348
0.7826666831970215	
1000	0
632	1368
0.7893333435058594	
1000	0
674	1326
0.7753333449363708	
1000	0
686	1314
0.7713333368301392	
1000	0
690	1310
0.7699999809265137	
1000	0
626	1374
0.7913333177566528	
1000	0
650	1350
0.7833333611488342	
1000	0
632	1368
0.7893333435058594	
1000	0
664	1336
0.7786666750907898	
1000	0
650	1350
0.7833333611488342	
1000	0
642	1358
0.7860000133514404	
1000	0
716	1284
0.7613333463668823	
1000	0
590	1410
0.8033333420753479	
1000	0
658	1342
0.7806666493415833	
1000	0
672	1328
0.7760000228881836	
1000	0
614	1386
0.7953333258628845	
1000	0
642	1358
0.7860000133514404	
1000	0
612	1388
0.796000038146973	
1000	0
630	1370
0.7900000214576721	
1000	0
610	1390
0.79666668176651	
1000	0
608	1392
0.7973333597183228	
1000	0
668	1332
0.7773333191871643	
1000	0
658	1342
0.7806666493415833	
1000	0
710	1290
0.7633333206176758	
1000	0
654	1346
0.7820000052452087	
1000	0
666	1334
0.77799997138977	
1000	0
668	1332
0.7773333191871643	
1000	0
648	1352
0.7839999794960022	

1000	0
652	1348
0.7826666831970215	
1000	0
637	1363
0.7876666784286499	
1000	0
660	1340
0.7799999713897705	
1000	0
672	1328
0.7760000228881836	