Tudorache Alexandru-Theodor

Grupa 242

Proiect Inteligență Artificială

- documentație –

Pentru rezolvarea cerinței acestui proiect am dezvoltat 3 modele de inteligență artificială cu ajutorul cărora am reușit să obțin diferite scoruri de acuratețe pentru datele de validare:

1. Un model Multinominal Naive-Bayes dezvoltat folosind biblioteca scikit-learn.naive\_bayes
2. Un model bazat pe o rețea de perceptroni dezvoltată folosind MLPClassifier din biblioteca scikit-learn
3. Un model bazat pe o rețea neuronală convoluțională dezvoltată folosind biblioteca Tensorflow

Pentru citirea datelor (atât cele de antrenare, cât și cele de validare și testare) am folosit clasa Image din biblioteca PIL și metoda *open* a acestei clase. Pentru toate cele 3 modele am nevoie de matrici bidimensionale având mărimea de nr\_imagini \* dimensiune\_imagini. Am observant că dimensiunea imaginilor este de 32\*32 = 1024, iar numărul de imagini din fiecare set de date putea fi numărat cu ajutorul fișierelor de tip txt (train.txt, validation.txt, test.txt). Astfel, am reușit să creez 3 matrici cu dimensiunile 30001x1024 (pentru train), 5000x1024 (pentru validation), respectiv 5000x1024 (pentru test).

Mai departe voi descrie pașii efectuați pentru dezvoltarea fiecărui model încercat:

1. Multinominal Naive-Bayes

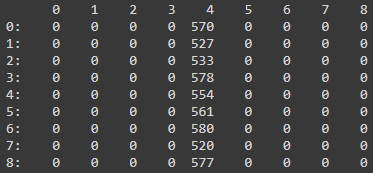
Înainte de toate, am observant că valorile pixelilor din mulțimile date (train, validation, test) sunt cuprinse între 0 și 255 (inclusiv) și am definit o funcție de discretizare a mulțimilor noastre care primește ca parametru, pe lângă matricea cu datele imaginilor, și intervalele împărțite între 0 și 255, obținute folosind funcția*linspace* din Numpy. Funcția a fost definită cu ajutorul funcției *digitize* din Numpy. După definire, am apelat funcția cu valori cuprinse între 1 și 15 pentru numărul de intervale folosite. Pentru fiecare valoare returnată de funcția de discretizare am creat un model folosind clasa *MultinominalNB* din sklearn.naive\_bayes, am antrenat modelul cu metodă *fit* a clasei, iar cu ajutorul metodelor *predict* și *score* ale aceleiași clase am obținut etichetele prezise de model pentru mulțimea de validare, respectiv acuratețea cu care aceste etichete au fost prezise. Cu ajutorul mulțimilor de etichete am reprezentat matricea de confuzie pentru fiecare model în parte.

Rezultatele sunt urmatoarele:

* Nr. Intervale = 1:

Acuratețe  = 0.1108

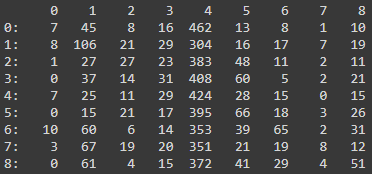
Matrice de confuzie:



* Nr. Intervale = 2:

Acuratețe = 0.157

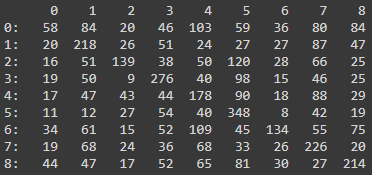
Matrice de confuzie:



* Nr. Intervale = 3:

Acuratețe = 0.3582

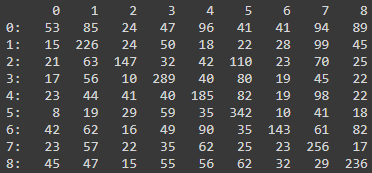
Matrice de confuzie:



* Nr. Intervale = 4:

Acuratețe = 0.3754

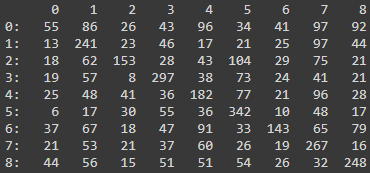
Matrice de confuzie:



* Nr. Intervale = 5:

Acuratețe = 0.3856

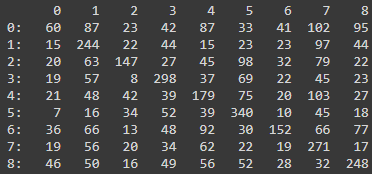
Matrice de confuzie:



* Nr. Intervale = 6:

Acuratețe = 0.3878

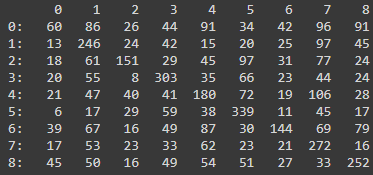
Matrice de confuzie:



* Nr. Intervale = 7:

Acuratețe = 0.3894

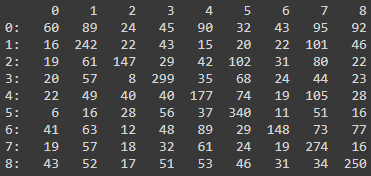
Matrice de confuzie:



* Nr. Intervale = 8:

Acuratețe = 0.3874

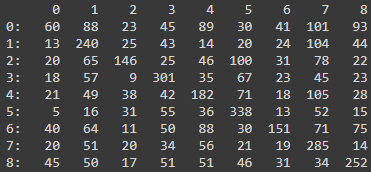
Matrice de confuzie:



* Nr. Intervale = 9:

Acuratețe = 0.391

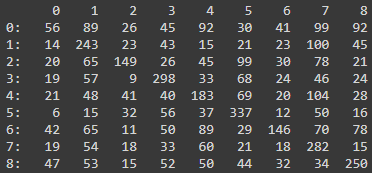
Matrice de confuzie:



* Nr. Intervale = 10:

Acuratețe = 0.3888

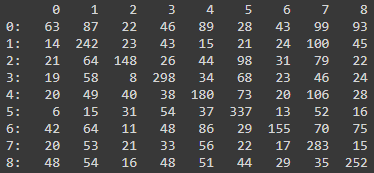
Matrice de confuzie:



* Nr. Intervale = 11:

Acuratețe = 0.3916

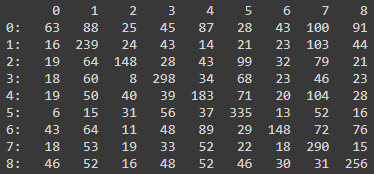
Matrice de confuzie:



* Nr. Intervale = 12:

Acuratețe = 0.392

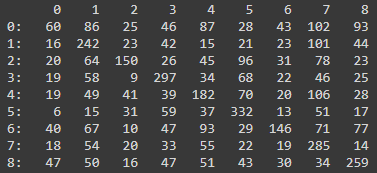
Matrice de confuzie:



* Nr. Intervale = 13:

Acuratețe = 0.3906

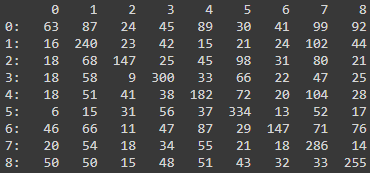
Matrice de confuzie:



* Nr. Intervale = 14:

Acuratețe = 0.3908

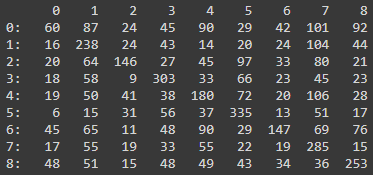
Matrice de confuzie:



* Nr. Intervale = 15:

Acuratețe = 0.3894

Matrice de confuzie:



Am observant că modelul cu 12 intervale obține cea mai bună acuratețe și am trimis o încercare pe datele de testare pe site-ul competiției și am obținut o acuratețe de 0.36986.

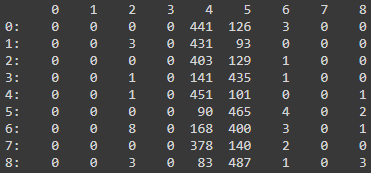
1. MLPClassifier

Înainte de toate, am normalizat datele de antrenare și datele de validare folosind clasa *StandardScaler* din sklearn.preprocessing și metodele *fit\_transform*(pentru mulțimea de antrenare), respectiv *transform*(pentru mulțimea de validare) ale aceleiași clase. Apoi, am instanțiat un model MLPClassifier cu diferiți hiperparametri, am antrenat modelul pe mulțimea train folosind metoda *fit* și am extras etichetele prezise și acuratețea cu care au fost prezise folosind metoda *predict*, respectiv funcția *accuracy\_score* (din sklearn.metrics). Am reprezentat matricea de confuzie pentru fiecare model încercat. Rezultatele arată astfel:

* Funcție de activare=’tanh’, Hidden Layer Sizes = (1), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 200 :

Acuratete = 0.1844

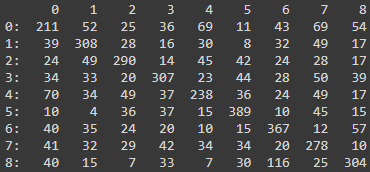
Matrice de confuzie:



* Funcție de activare=’tanh’, Hidden Layer Sizes = (10), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 200 :

Acuratete = 0.5384

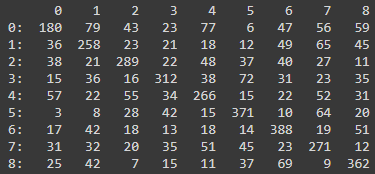
Matrice de confuzie:



* Funcție de activare=’tanh’, Hidden Layer Sizes = (10), Learning Rate = 0.00001, Momentum = 0, Nr. Maxim de Iterații = 200 :

Acuratete = 0.5394

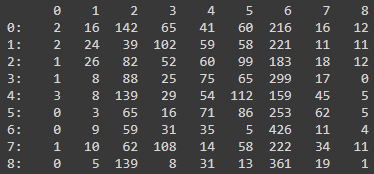
Matrice de confuzie:



* Funcție de activare=’tanh’, Hidden Layer Sizes = (10), Learning Rate = 10, Momentum = 0, Nr. Maxim de Iterații = 200 :

Acuratete = 0.1468

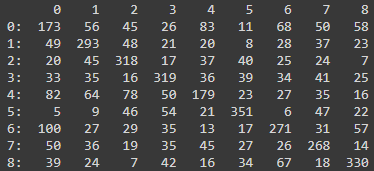
Matrice de confuzie:



* Funcție de activare=’tanh’, Hidden Layer Sizes = (10), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 20 :

Acuratete = 0.5004

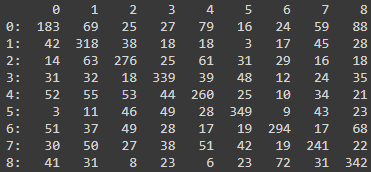
Matrice de confuzie:



* Funcție de activare=’tanh’, Hidden Layer Sizes = (10, 10), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 2000 :

Acuratete = 0.5204

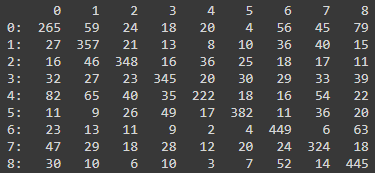
Matrice de confuzie:



* Funcție de activare=’relu’, Hidden Layer Sizes = (10, 10), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 2000 :

Acuratete = 0.6274

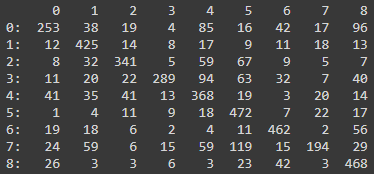
Matrice de confuzie:



* Funcție de activare=’relu’, Hidden Layer Sizes = (100, 100), Learning Rate = 0.01, Momentum = 0, Nr. Maxim de Iterații = 2000 :

Acuratete = 0.6544

Matrice de confuzie:



1. Rețea neuronală convoluțională

Înainte de toate, am normalizat datele de antrenare și datele de validare folosind clasa *StandardScaler* din sklearn.preprocessing și metoda *transform* a clasei, pentru ambele mulțimi de date(atât train, cât și validation).

Am creat o stivă cu mai multe straturi pentru model:

- 1 strat de input cu mărimea 32\*32\*1 deoarece imaginile sunt alb-negru cu mărimea de 32x32.

- 1 strat convoluțional 2D cu 16 filtre, mărimea kernel-ului 3x3, funcția de activare ‘relu’

- 1 strat ce realizează operația de max pooling pentru date 2D cu pool size de 2\*2.

- 1 strat convoluțional 2D cu 32 filtre, mărimea kernel-ului 3x3, funcția de activare ‘relu’

- 1 strat ce realizează operația de max pooling pentru date 2D cu pool size de 2\*2.

- 1 strat convoluțional 2D cu 64 filtre, mărimea kernel-ului 3x3, funcția de activare ‘relu’

- 1 strat ce realizează dropout pentru input (pentru a reduce overfitting-ul)

- 1 strat ce realizează flatten pentru input

- 1 strat dens (cu 10 unități și funcție de activare *softmax)* pentru output

Am avut două abordări asupra straturilor convoluționale 2D, o abordare cu bias regulizer L2(0.001) și una fără bias regulizer.

Am compilat modelul folosind un optimizer Adam cu diferite learning rate-uri și am antrenat modelul cu diferite batch size-uri și număr de epoci.

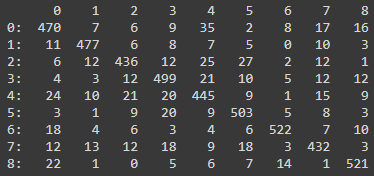
Rezultatele arată astfel:

* Model cu bias regulizer L2(0.001) pe straturile convolutionale 2D, learning rate = 0.0003, batch size = 256, nr. Epoci = 32

Loss pe mulțimea de antrenare = 0.3577; Acuratețe pe mulțimea de antrenare = 0.8722

Loss pe mulțimea de validare = 0.4079; Acuratețe pe mulțimea de validare = 0.8610

Matrice de confuzie:

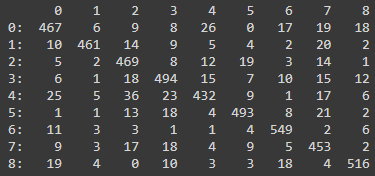


* Model cu bias regulizer L2(0.001) pe straturile convolutionale 2D, learning rate = 0.0003, batch size = 64, nr. Epoci = 32

Loss pe mulțimea de antrenare = 0.2770; Acuratețe pe mulțimea de antrenare = 0.9002

Loss pe mulțimea de validare = 0.3890; Acuratețe pe mulțimea de validare = 0.8668

Matrice de confuzie:

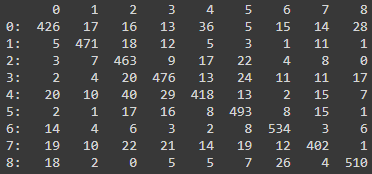


* Model fara bias regulizer pe straturile convolutionale 2D, learning rate = 0.003, batch size = 256, nr. Epoci = 8

Loss pe mulțimea de antrenare = 0.4159; Acuratețe pe mulțimea de antrenare = 0.8538

Loss pe mulțimea de validare = 0.4606; Acuratețe pe mulțimea de validare = 0.8386

Matrice de confuzie:

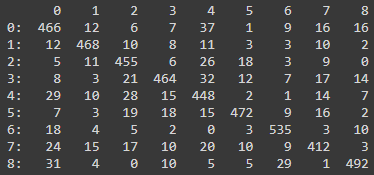


* Model fara bias regulizer pe straturile convolutionale 2D, learning rate = 0.003, batch size = 128, nr. Epoci = 8

Loss pe mulțimea de antrenare = 0.4132; Acuratețe pe mulțimea de antrenare = 0.8494

Loss pe mulțimea de validare = 0.4507; Acuratețe pe mulțimea de validare = 0.8424

Matrice de confuzie:

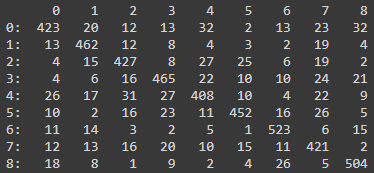


* Model fara bias regulizer pe straturile convolutionale 2D, learning rate = 0.003, batch size = 64, nr. Epoci = 8

Loss pe mulțimea de antrenare = 0.4305; Acuratețe pe mulțimea de antrenare = 0.8475

Loss pe mulțimea de validare = 0.5111; Acuratețe pe mulțimea de validare = 0.8170

Matrice de confuzie:



Am observant că pe modelul cu bias regulizer L2(0.001) pe straturile convoluționale 2D, learning rate = 0.0003, batch size = 64, nr. Epoci = 32 se obține cea mai bună acuratețe, am trimis o încercare pe datele de testare pe site-ul competiției și am obținut o acuratețe de 0.85680.