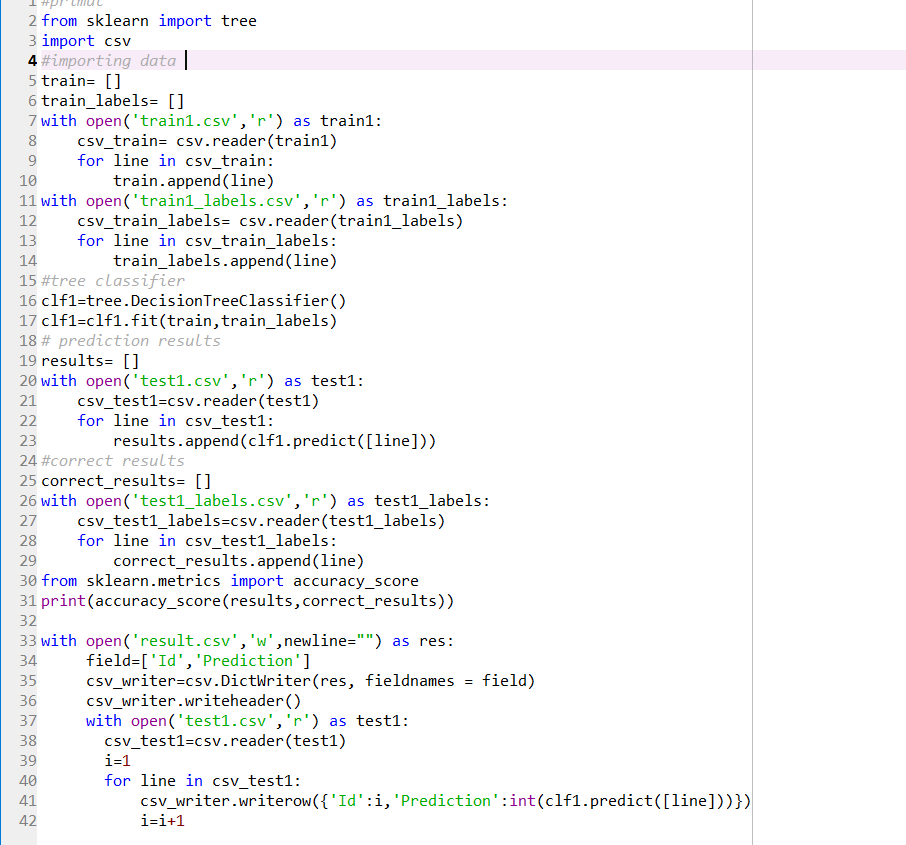
Black-Box Classification

Classify samples without knowing what they are

Rosca Maria Alexandra

Grupa 311

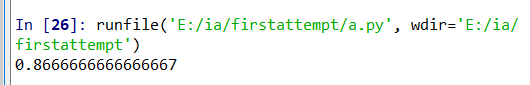
* Decision Tree Classifier

Dupa o scurta documentare, am hotarat sa incerc clasificatorul DecisionTreeClassifier() din biblioteca scikit-learn

Nota:

1. Fisierul „test1.csv” contine primele 300 de data din fisierul „train\_samples.csv”, la fel si fisierul „test1\_labels.csv”
2. Fisierul „train1.csv” contine liniile ramase, la fel si „train1\_labels.csv”

Rezultatul->o acuratete de doar 86.667% pe cele 300 de date separate de mine pentru testare si doar 77.7% pe datele de test din concurs

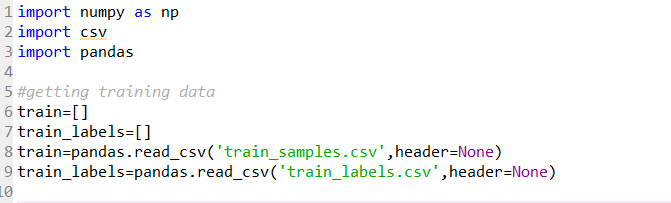


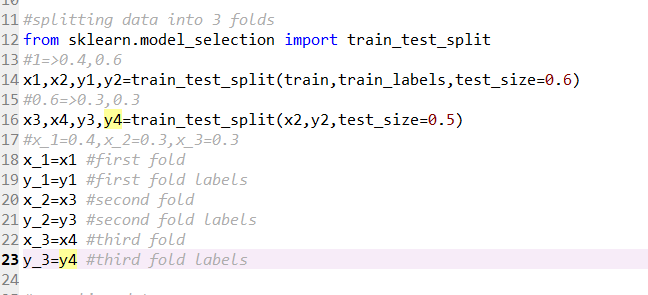
* Support Vector Machine

Dupa incercarea mai multor clasificatori (Decision Tree 77.7%, Kneighbours-81.3% etc.) am constatat ca o acuratete putin mai buna pe datele noastre obtinem folosind Svm- clasificator folosit pentru unul dintre rezultatele pe care le-am ales pentru nota finala.

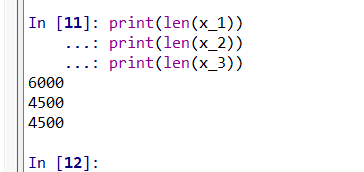
Mai jos voi ilustra impartirea datelor in 3 parti, clasificatorul antrenat pe fiecare dintre ele si acuratetea acestuia pe celelalte doua ramase prin accuracy\_score si confusion\_matrix.

1. Citirea datelor

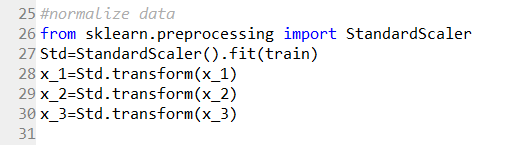


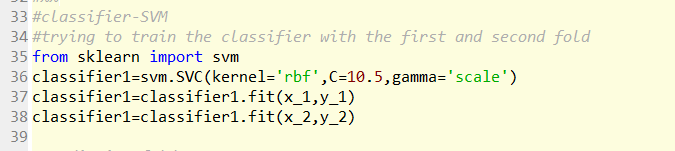
1. Impartirea acestora

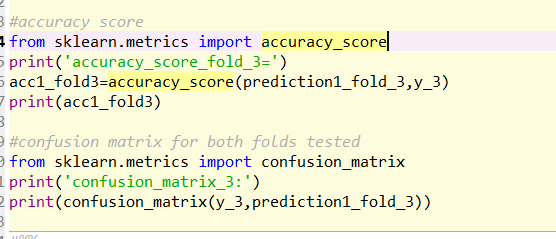
-am impartit cele 15000 de date din fisierul „train\_samples.csv” in 3 parti a cate 40%, 30% si respectiv 30%



1. Normalizarea datelor standard , folosind media de pe tot fisierul cu date

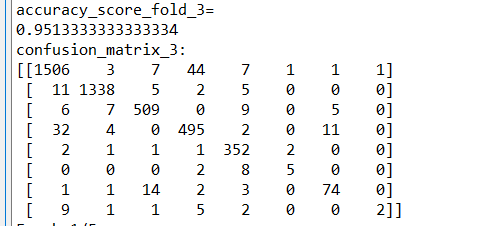


1. Antrenarea unui clasificator pe prima parte din date
2. Acuratete si matrice de confuzie

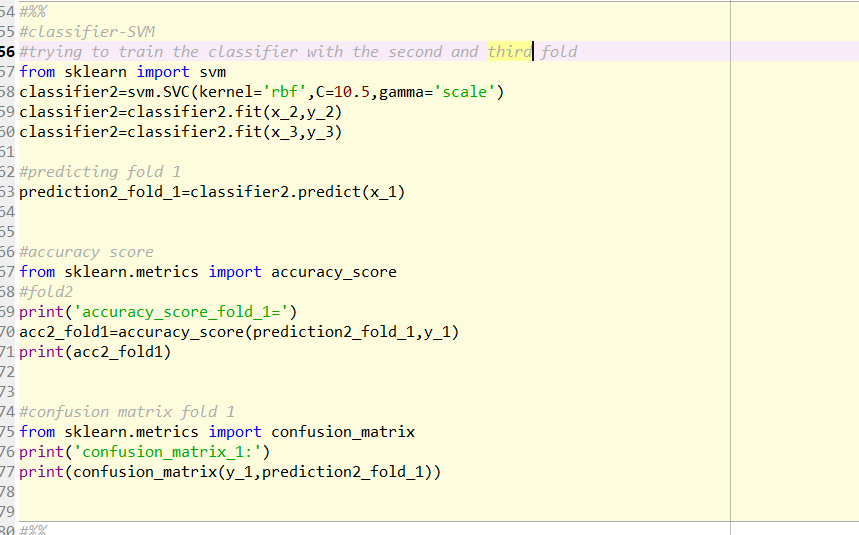


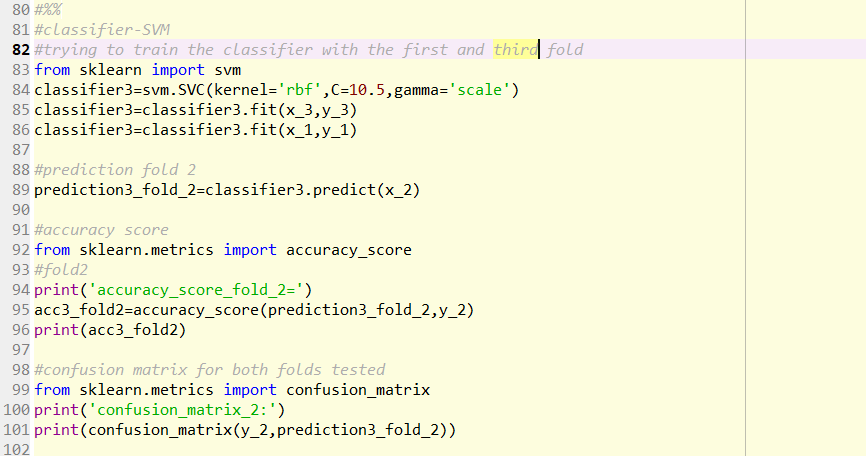
Rezultate:

Acuratete si matrice de confuzie

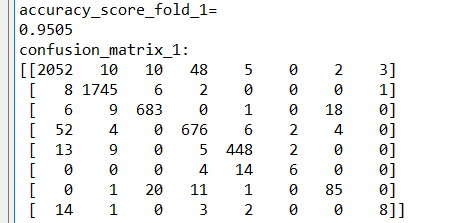


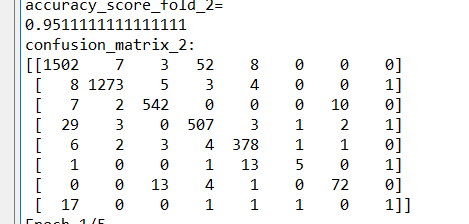
1. Am repetat pasii 4 si 5 pentru celalalte 2 parti de date ramase (antrenare+test

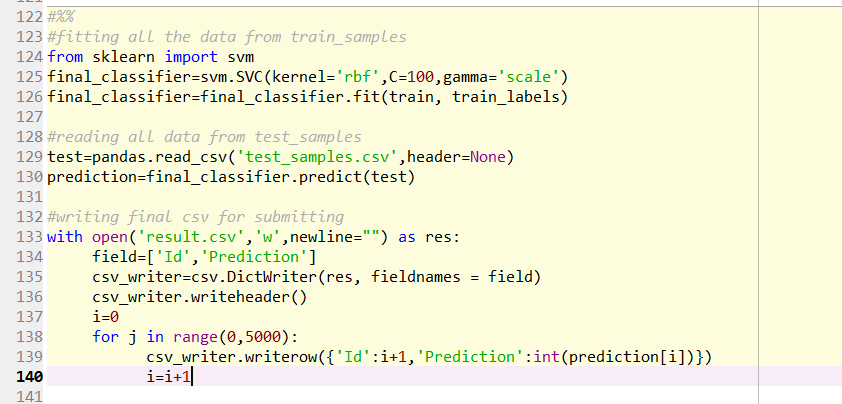




Rezultate: 95.0% pe fold 1 si 95.1% pe fold 2





1. Am antrenat un clasificator pe intregul set de date disponibil.
2. Alegerea parametrilor potriviti

Kernel- poate fi „linear”, „rbf”, „poly”

-reprezinta tipul de hiperplan folosit pentru separarea datelor( linear-dreapta)

-pentru submisia finala am ales „rbf”

Gamma- cu cat este mai mare cu atat modelul incearca sa clasifice datele mai bine, sa se plieze exact pe datele de antrenare

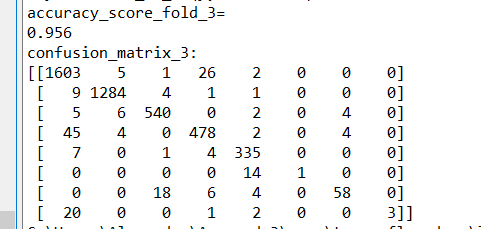
-am ales sa ii las valoarea defalut, „scale”

C-parametru de penalitate, cu cat este mai mare cu atat calsificarea este mai generala

-am ales C=10.5

Acesti parametri au fost obtinuti in urma testarii pe datele finale si alegerii acelora care au rezultate mai bune.

Cu acesti parametrii am obtinut(avand pe x\_1 si x\_2 ca train set):



Pe datele totale am obtinut o acuratete de 83.8%

* Retele neurale

Dupa putin documentatie am gasit biblioteca tensorflow care ne pune la dispozitie implementarea unei retele neurale intr-un mod facil.

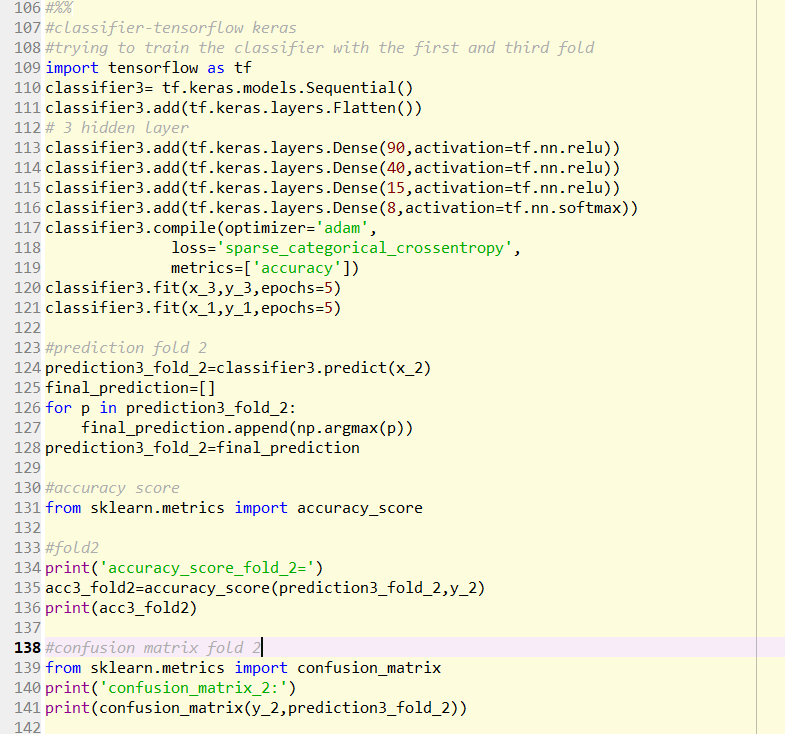
Dupa multiple incercari, am ales ca reteaua mea sa aiba, 3 straturi ascunse(90, 40 si respectiv 15 neuroni), sa foloseasca ca functie de activare „relu” si ca functie de iesire „softmax” , functie ce returneaza probabilitatea ca un set de date sa apartina uneia dintre cele 8 clase.Ca si vector de predictii finale, aplicam argmax pe vectorul returnat de softmax pentru a returna clasa cu probabilitatea cea mai mare.

Am repetat pasii de la svm pentru noul model (3 fold separation, training, testing,accuracy, confusion\_matrix).





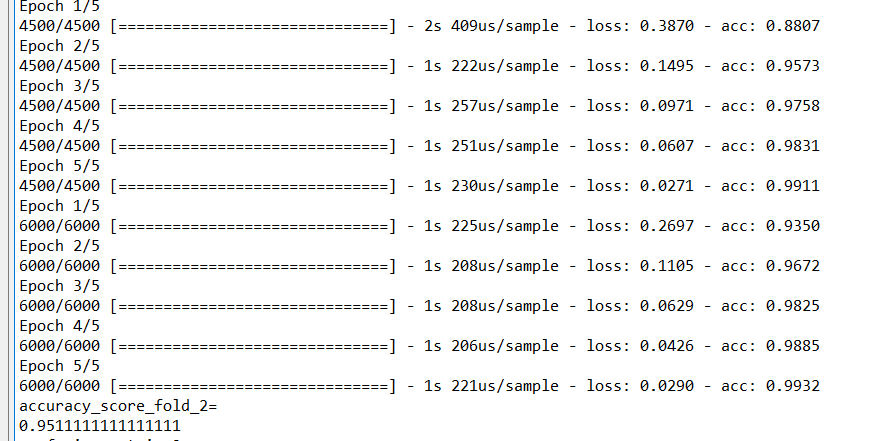






Rezultate:

83.1% pe datele finale



95.11% pe datele alese de mine-fold 2