Proiect ISIA

Determinarea persoanelor care au fost aprobate pentru un credit bancar,folosind algoritmul “Support Vector Machine”

Codita Vlad-Alexandru

Grupa:424A

1) **Ce problema rezolvam?**
   Trebuie determinat cererea a cator persoane din cele 690 exemple dintr-un anumit set de date, a fost aprobata pentru un credit bancar.In cele 15 dimensiuni alea bazei de date,6 sunt continue si restul de 9 sunt categorice

   O initiala problema a bazei de date este faptul ca aceasta contine 37 esantioane cu date lipsa. Aceasta problema am rezolvat-o imputand valori in cele 37 esantioane.Am ales sa nu le elimin deoarece reprezentau aproximativ 5% din setul de date.

   O alta problema a setului de date este multitudinea de dimensiuni categorice pe care a trebuit sa le transform folosind functia **get_dummies** din libraria **pandas** in valori binare,astfel marind numarul de coloane a setului de date.

2) **Librarii**
   Am ales sa utilizez libraria **pandas** pentru a citi datele din fisierul csv.

   Pentru a inlocui **missing data-ul** din baza de date din ‘?’ in ‘N/A’ a fost nevoie sa folosesc libraria **numpy.**

Pentru a implementa algoritmul **Support Vector Machine** am utilizat libraria **scikit-learn** din care am important urmatoarele functii:

-**svm** pentru a implementa algoritmul

-**train_test_split** din **sklearn.model.selection** pentru a imparte setul de date in Train/Test

-**accuracy_score** din **sklearn.metrics** pentru a masura acuratetea

-**MinMaxScaler** din **sklearn.preprocessing** pentru a normaliza setul de date in functie de (-1,1)

3) **Seturi de date de Train si Test**

Intrucat in baza de date nu era implementat acest lucru,am ales sa impart setul de date in **75%** pentru **Train** si restul de **25%** pentru **Test**.

Acest lucru a fost posibil datorita functiei **train_test_split** din **sklearn.model.selection**.

4) **Metrica**

Metrica folosita pentru a masura rezultatele algoritmului a fost **acuratetea**. Aceasta a fost calculata cu functia **accuracy_score**.

Am variat functia cost din urmatorul interval: c = [0.00001 , 0.03125 , 0.125 , 0.5 , 2 , 8 , 32 , 128]. Am introdus prima valorea 10^-5 pentru a vedea daca algoritmul invata cu adevarat.

Cost	Acuratete
0.00001	**52.90697674418605%**
0.03125	**82.55813953488372%**
0.083	**82.55813953488372%**
0.084	**83.13953488372093%**
0.125	**83.13953488372093%**
0.5	**83.13953488372093%**
2	**83.13953488372093%**
8	**83.13953488372093%**
32	**83.13953488372093%**
128	**83.13953488372093%**

Observam ca pe masura ce functia cost creste,acuratetea creste si ea ceea ce era de asteptat avand in vedere considerentele teoretice alea SVM-ului.

O problema de care m-am lovit a fost faptul ca de la o anumita valoare cost,programul atinge un prag peste care acuratetea nu mai creste. Dupa mai multe teste,am observat ca aceasta acuratete devine maxima la o valoare cost aproximativ egala cu 0.0835, aceasta fiind valoarea cost optima. Astfel,pentru a mari acuratetea avem nevoie de mai multe date in setul de date.