***Documentatie pentru proiectul***

***de la Inteligenta Artificiala***

***Student: Dima Cătălin-Ștefan***

***Grupa: 242***

**O scurta prezentare a proiectului**

Trebuie sa antrenam o masina pentru ca aceasta, la primirea unei propozitii sa poata sa recunoasca daca este in limba romana sau in moldoveaneasca.

Antrenarea masinii se face prin intermediul a doua fisiere .txt. Unul dintre acestea contine pe prima coloana ID-ul propozitiei si pe a doua contine propozitia in sine. Al doilea fisier are prima coloana identica cu cea din primul document, iar a doua coloana contine 0 in cazul in care propozitia este in molodveneasca / 1 in cazul in care propozitia este in romana.

Dupa antrenare urmeaza pasul in care masina noastra da predictii pentru propozitiile dintr-un fisier de tipul primului document .txt din antrenare si trebuie sa genereze raspunsul intr-un .csv file ce va fi in stilul celui de-al doilea document din antrenare.

**Support Vector Machines (SVM)**

Metoda de rezolvare aleasa de mine pentru acest proiect este Support Vector Machines (SVM) din sklearn.

Un model SVM este o reprezentare a exemplelor ca puncte in spatiu, mapate astfel incat exemplele sa fie ampartite printr-o fereastra cat mai larga.

Inainte de aplicarea acestei metode este nevoie sa implementam un “Bag of Words”.

L-am implementat prin intermediul unei clase cu doua functii:

* “build\_vocabulary” - il vom folosi pentru a ne crea vocabularul tuturor propozitiilor citite pe care ne vom antrena.
* “get\_features” - vom crea vectorul de aparitii a cuvintelor din vocabular in propozitiile carora trebuie sa prezicem rezultatul

Dupa implementarea de *“*Bag Of Words”:

1. Cream vocabular-ul pentru propozitiile de antrenare
2. Generam vectorul de aparitii in vocabular pentru pro-pozitiile de antrenare, dar si cele carora trebuie sa le prezicem rezultatul
3. Cei doi vectori rezultate trebuie sa le normalizam prin intermediul functiei “normalize\_data”
4. Prin intermediul datelor nomalizate vom putea sa an- trenam masina prin intermediul functiei “fit” de matri- cea normalizata a propozitiilor de antrenare si respec-tivele valori atribuite acestora ( In acest caz, i-am dat valoarea 100 hiperparametrului C deoarece imi oferea cea mai buna acuratete la final)

*“svm\_model = svm.SVC(C=100, kernel='linear')”* )

1. Dupa antrenare masina noastra este pregatita sa faca predictii pe propozitiile dorite prin intermediul functiei “predict” ce primeste ca parametru matricea normali- zata de valori pentru propozitiile carora trebuie sa le prezicem un rezultat.
2. Afisarea se face intr-un document .csv prin interme- diul folosirii functiilor din modulul csv in Python.

**Teste pentru algoritm**

Accuracy: 65.17319277108435

F1 score: 0.6615440907427735

Confusion Matrix:

[827, 474]

[451, 904]

**Kernel:**

* Linear (acuratete ok 0,65, timp ok)
* Rbf (e posibil sa aiba acuratete mai mare, timp mai mare de rulare)
* Polinomial (acuratete slaba 0,53)

**Parametru C:**

* 100 -- 0.65
* 50 -- 0.63
* 40 -- 0.62

“I recently read two articles that found similar conclusions: when dealing with Linear classifiers (like linear SVMs) the L2 norm performs very well.”