Documentație proiect

Dialectul românesc se aseamănă foarte mult cu cel moldovenesc existând însă cateva diferențe cum ar fi influența limbii ruse și a celei ucrainiene (într-o masură mai mică) și apariția unor regionalismele la nivelul dialectului moldovenesc. Aceste aspecte reprezintă un factor de determinare a dialectului.

I. Citirea și procesarea datelor

Citirea datelor

Fișierele sunt deschise cu `codecs.open(nume_fisier, encoding='utf-8')`, parametrul encoding este destinat citirii caracterelor speciale. Datele sunt citite cu `np.genfromtxt(file_open, comments=None, delimiter='\t', dtype=None, names=('Id', 'Text/Label'))`, acestea sunt împărțite folosind delimitatorul '\t' în Id și Text pentru samples, repectiv Id și Label pentru labels.

Procesarea datelor

Formarea vocabularului

Datele sunt prelucrate folosind CountVectorizer, care creează un vocabular pe baza cuvintelor întalnite, apelând metoda `fit(training_data)` și apoi cu ajutorul metodei `transform(training_data)` frazele sunt codificate în funcție de numărul de apariții ale cuvintelor ce le compun. Pentru prima competiție am folosit și SnowballStemmer din nltk.stem.snowball pentru a aduce cuvintele la radacină. Acest lucru a îmbunătățit performanta modelului.

Pentru a treia competiție am modificat parametrul `token_pattern` din CountVectorize pentru a nu mai elimina caracterele de tipul {$&+,:;=?@#|<>.^*()%!-}. Acestea se găsesc în interiorul cuvintelor și, în lipsa acestui parametru, caracterele erau eliminate, ducând astfel la scăderea numărului de cuvinte și la realizarea unui vocabular greșit. Am folosit și parametrul `ngram_range(1,2)` care construiește un vocabular format atât din cuvinte independente, cât și din seturi de câte două cuvinte alaturate. Acest lucru a adus un plus în diferențierea dialectelor.

Normalizarea datelor

Etapa a doua o reprezintă normalizarea datelor pe care am făcut-o cu ajutorul funcției `sklearn.preprocessing.Normalizer` atât cu normalizare de tip l1 cât și de tip l2, după caz. S-a observat că, pentru modelul Complement Naive Bayes, normalizarea datelor folosind l2 a dat un `f1_score` mai mare indiferent dacă se folosea `ngrams_range(1,2)` sau nu. Pentru modelul Mașini cu Vectori Suport normalizarea datelor cu l1 aduce un plus de performanță pentru Kernel rbf și `ngrams_range(1,2)` și pentru Kernel linear fară `ngrams_range(1,2).` Normalizarea l2 este mai bună pentru celelalte variante.

II. Modelele abordate

Mașini cu Vectori Suport (SVM)

Detalii de implementare

Există două tipuri de abordare pentru a clasifica datele:

- ONE VS ALL – Sunt antrenați num_classes clasificatori, fiecare clasă fiind diferențiată de toate celelalte prin acest clasificator. Eticheta finală este dată de clasificatorul care a obținut scorul maxim și este asociată unui nou exemplu.
- ONE VS ONE – Sunt antrenați $\frac{\text{num\_classes} * (\text{num\_classes} - 1)}{2}$ clasificatori, câte unul pentru fiecare pereche de două clase. Eticheta finală este dată de clasificatorul care a obținut cele mai multe voturi și este asociată unui nou exemplu.

Definirea modelului

Am folosit `sklearn.svm.SVC(C, kernel)` unde am încercat varianta cu Kernel de tip rbf și `sklearn.svm.LinearSVC(C)` pentru varianta cu Kernel de tip linear. Implementarea folosită este cea de ONE VS ONE.

Parametrului de penalitate pentru eroare C, care se referă la cât de mult poate modelul să evite clasificarea greșită, i-au fost asociate valori din intervalul [1, 15]. Dacă valoarea lui C este prea mare poate apărea supra-învățarea (overfitting), iar daca este prea mică poate apărea sub-învățarea (underfitting).

Complement Naive Bayes (CNB)

Detalii de implementare

ComplementNB implementează Complement Naive Bayes. CNB este o adaptare a algoritmului standard Multinomial Naive Bayes (MNB), care este potrivită pentru seturile de date ce sunt dezechilibrate. CNB folosește statistici pentru a calcula ponderile modelului.

Definirea modelului

Alpha este parametrul de netezire și i-am asociat valori din intervalul [0,1].

III. Antrenarea și evaluarea modelelor

Antrenarea

Metodele folosite în cadrul ambelor abordari sunt: `fit(normalize_train_data, training_labels)` unde `normalize_train_data` sunt datele de antrenare normalizate, iar `training_labels` sunt etichetele pentru antrenare (0 pentru MD și 1 pentru RO), această metodă

returnează antrenarea modelului. Prezicerea dialectului se realizează apelând `predict(normalize_test_data)` unde `normalize_test_data` sunt datele de test.

Evaluarea

Calcularea acurateței este realizată cu funcția `sklearn.metrics.accuracy_score (predicted_labels_svm, test_labels)`, unde `predicted_labels_svm` sunt predincțiile rezultate anterior din `predict(normalize_test_data)`, iar test_labels sunt etichetele corecte. Acuratețea reprezintă scorul de clasificare a exactității.

Pentru a calcula f1_score am folosit `f1_score(test_labels,predicted_labels_svm)` care se calculează după formula $f1 = 2 * \frac{(\text{precision} * \text{recall})}{(\text{precision} + \text{recall})}$; unde precision reprezintă ce proporție din identificările pozitive, a fost într-adevăr corectă, iar recall reprezintă ce proporție din valoriile pozitive au fost identificate drept corecte.

IV. Submit-uri

Pe parcursul competiției am avut atât submit-uri cu valori rezultate din SVM, cât și din CNB. Pentru prima competiție am avut rezultate mai bune folosind SVM de tip linear cu normalizare de tipul l2. Cele mai bune rezultate fiind in jurul valorii de 0.66.

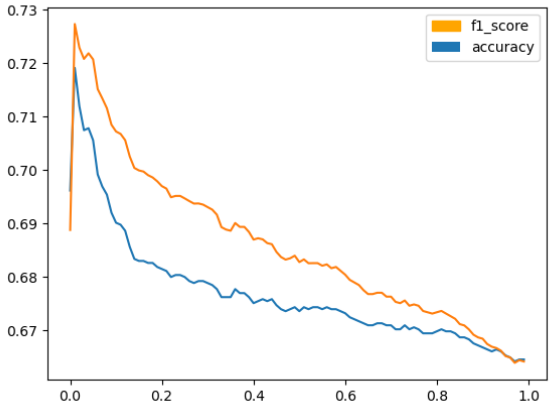
Pentru a doua competiție toate rezultatele erau undeva in zona lui 0.46 folosind SVM de ambele tipuri. Nu existau modificari majore nici dacă schimbam tipul de normalizare. Pentru acestea două nu am avut submit-uri cu CNB, dar am încercat să îmbunătățesc performanta SVM-ului folosind `ngrams_range`, însă acesta nu a adus rezultate foarte bune.

Pentru a treia competiție inițial am folosit SVM. Observând după desenarea câtorva grafice că pentru Kernel rbf acuratețea și `f1_score` sunt mai mari față de Kernel de tip linear. Primele submit-uri au pe validare `f1_score` = 0.66 utilizând Kernel rbf. Pe platformă am urcat rezultate pentru varianta cu și fara datele de validare adăugate. Am obținut scoruri mai bune pentru rezultatele generate de varianta cu adăugarea datelor de validare. Valorile pentru parametrul C au fost alese cu ajutorul graficelor generate în jurul punctelor `f1_score` maxime.

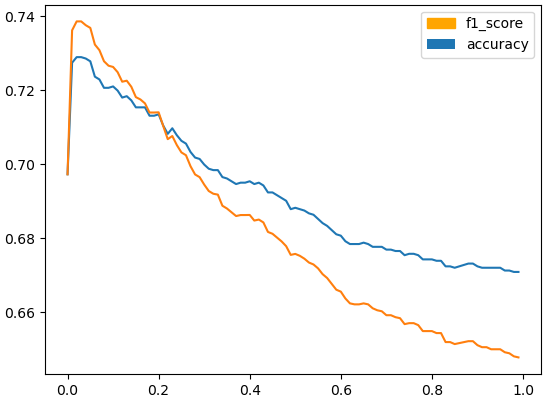
Modelul CNB a avut o performanță mai bună în a treia competiție. Majoritatea rezultatelor fiind de peste 0.66, indiferent dacă se folosea `ngrams_range` sau nu. Valorile cele mai bune au fost pentru alpha intre 0.01 și 0.04. Acestea având un `f1_score` mai mare de 0.71.

Exemple pentru Complement Naive Bayes

ngram_range(1,2)



Figură 1. ComplementNB cu alpha care ia valorile dintre 0 și 1 cu diferență de 0.01 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.727 pentru alpha = 0.01

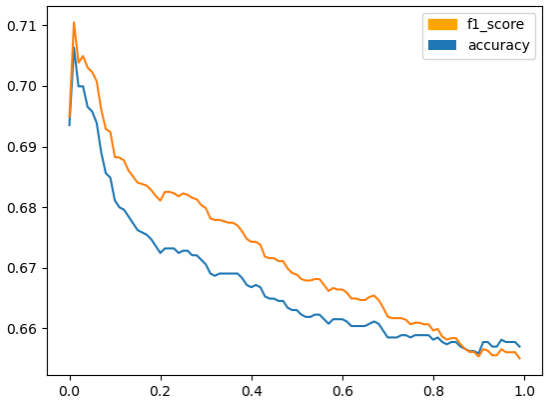


Figură 2. ComplementNB cu alpha care ia valorile dintre 0 și 1 cu diferență de 0.01 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.738 pentru alpha = 0.02

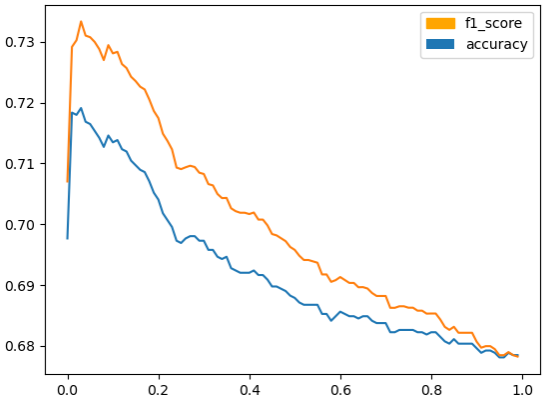
Matricea de confuzie pentru alpha=0.01		
Actual / Predicted	0	1
0	915	386
1	360	995

Matricea de confuzie pentru alpha=0.02		
Actual / Predicted	0	1
0	919	382
1	338	1017

fara ngram_range(1,2)



Figură 3. ComplementNB cu alpha care ia valorile dintre 0 și 1 cu diferență de 0.01 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.71 pentru alpha = 0.01



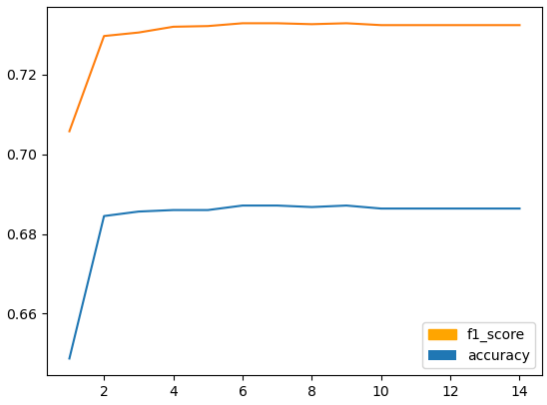
Figură 4. ComplementNB cu alpha care ia valorile dintre 0 și 1 cu diferență de 0.01 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.733 pentru alpha = 0.03

Matricea de confuzie pentru alpha=0.01		
Actual / Predicted	0	1
0	919	382
1	398	957

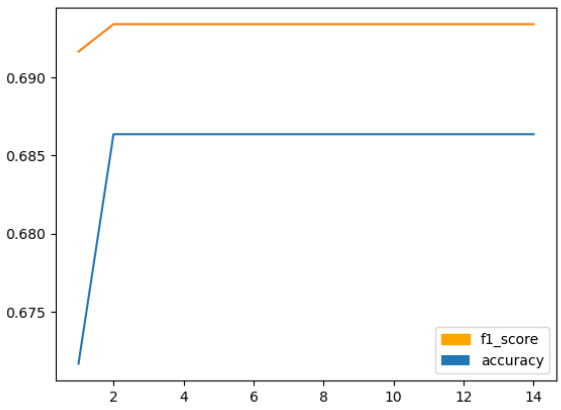
Matricea de confuzie pentru alpha=0.03		
Actual / Predicted	0	1
0	884	417
1	329	1026

Exemple pentru Mașini cu Vectori Suport

ngram_range(1,2)



Figură 5. Mașini cu Vectori Suport cu Kernel='rbf' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.732 pentru C = 5

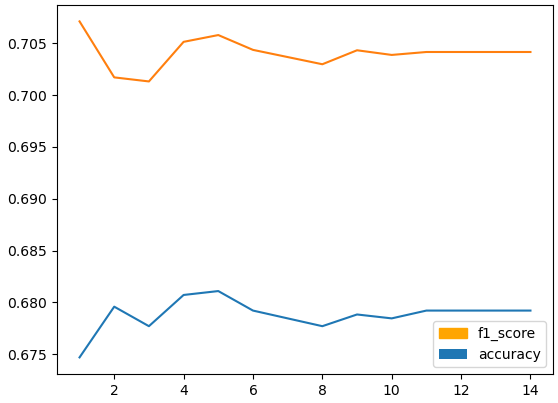


Figură 6. Mașini cu Vectori Suport cu Kernel='rbf' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.69 pentru C = 1

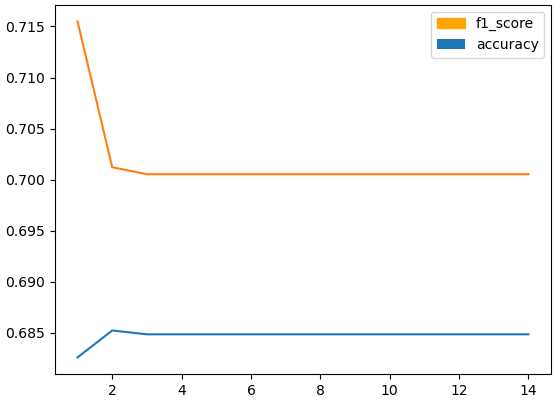
Matricea de confuzie pentru C = 5		
Actual / Predicted	0	1
0	682	619
1	215	1140

Matricea de confuzie pentru C = 1		
Actual / Predicted	0	1
0	806	495
1	377	978

fara ngram_range(1,2)



Figură 7. Mașini cu Vectori Suport cu Kernel='rbf' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.708 pentru C = 1

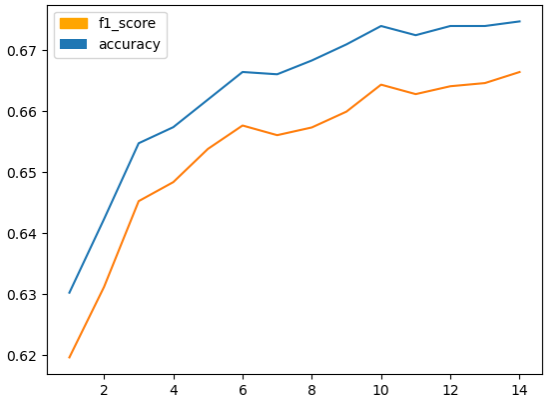


Figură 8. Mașini cu Vectori Suport cu Kernel='rbf' si C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.715 pentru C = 1

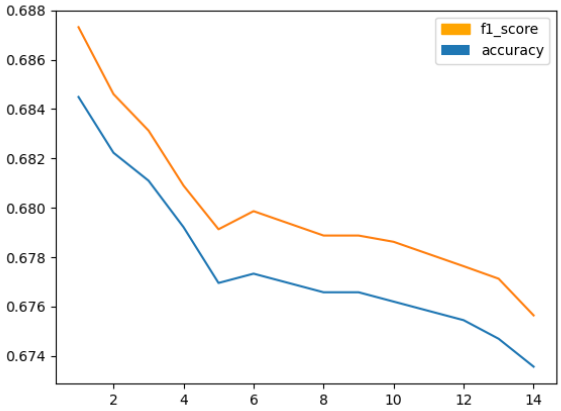
Matricea de confuzie pentru C = 1		
Actual / Predicted	0	1
0	749	552
1	312	1043

Matricea de confuzie pentru C = 1		
Actual / Predicted	0	1
0	753	548
1	295	1060

ngram_range(1,2)



Figură 9. Mașini cu Vectori Suport cu Kernel='linear' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.663 pentru C = 14

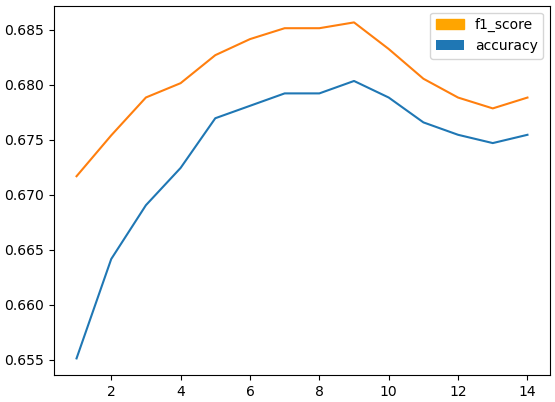


Figură 10. Mașini cu Vectori Suport cu Kernel='linear' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.68 pentru C = 1

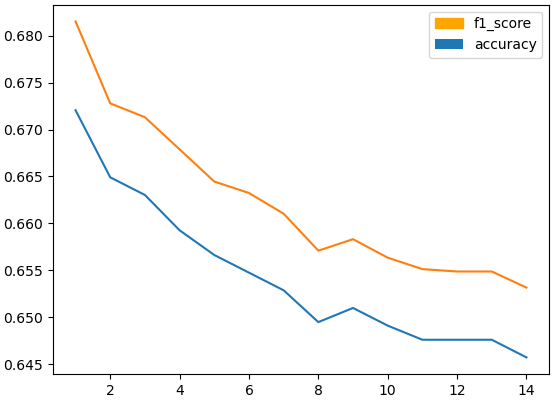
Matricea de confuzie pentru C = 14		
Actual / Predicted	0	1
0	929	372
1	492	863

Matricea de confuzie pentru C = 1		
Actual / Predicted	0	1
0	886	415
1	452	903

fara ngram_range(1,2)



Figură 11. Mașini cu Vectori Suport cu Kernel='linear' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l1
valoarea maximă a lui f1_score este 0.685 pentru C = 9



Figură 12. Mașini cu Vectori Suport cu Kernel='linear' și C care ia valorile dintre 1 și 15 cu diferență de 1 și cu normalizare de tipul l2
valoarea maximă a lui f1_score este 0.681 pentru C = 1

Matricea de confuzie pentru C = 9		
Actual / Predicted	0	1
0	881	420
1	429	926

Matricea de confuzie pentru C = 1		
Actual / Predicted	0	1
0	853	448
1	423	932