

Modelul Bag-of-Words. Clasificatorul SVM.

În acest laborator vom clasifica texte folosind modelul Bag-of-Words („sac de cuvinte”) pentru reprezentarea textelor și clasificatorul SVM predat la curs. Revedeți slide-urile de la curs (cursul 5) pentru a vă reaminti în detaliu în ce constă clasificatorul SVM.

1. Modelul Bag-of-Words (BOW)

Modelul Bag-of-words este o reprezentare simplificată a textelor folosită în procesarea limbajului natural și în regăsirea informației. Conform acestui model, reprezentăm un text numărând de câte ori apare un cuvânt dintr-un anumit dicționar în textul respectiv. Prin folosirea unei asemenea reprezentări, informația legată de ordinea cuvintelor, gramatică, topică, sensul cuvintelor se pierde.

Exemplu¹: considerăm textele 1 și 2 de mai jos în limba engleză.

```
(1) John likes to watch movies. Mary likes movies too.  
(2) John also likes to watch football games
```

Eliminând semnele de punctuație obținem listele de cuvinte pentru cele două texte:

```
"John", "likes", "to", "watch", "movies", "Mary", "likes", "movies", "too"  
"John", "also", "likes", "to", "watch", "football", "games"
```

Considerăm dicționarul D format din reuniunea tuturor cuvintelor din cele 2 texte:

```
D = {"John", "likes", "to", "watch", "movies", "Mary", "too", "also",  
"football", "games"}
```

Reprezentăm fiecare text numărând de câte ori apare fiecare cuvânt din dicționarul D în fiecare text. Obținem reprezentările bag-of-words următoare:

```
BoW1 = {"John":1, "likes":2, "to":1, "watch":1, "movies":2, "Mary":1,  
"too":1, "also":0, "football":0, "games":0};
```

```
BoW2 = {"John":1, "likes":1, "to":1, "watch":1, "movies":0, "Mary":0,  
"too":0, "also":1, "football":1, "games":1};
```

Pentru dicționarul D fixat, reprezentările se pot scrie sub forma de vectori de frecvențe:

```
v1 = [1, 2, 1, 1, 2, 1, 1, 0, 0, 0];  
v2 = [1, 1, 1, 1, 0, 0, 0, 1, 1, 1];
```

Suma elementelor fiecărui vector reprezintă numărul de cuvinte din text. De obicei, pentru probleme de clasificare în care încercăm să discriminăm între texte de diferite lungimi se

¹ https://en.wikipedia.org/wiki/Bag-of-words_model

folosesc reprezentări normalizate. Spre exemplu, folosind norma L_1 (suma absolută a elementelor unui vector) obținem vectorii normalizați L_1 :

$$v1_{L1} = \frac{v1}{\|v1\|_1} = \left[\frac{1}{9}, \frac{2}{9}, \frac{1}{9}, \frac{1}{9}, \frac{2}{9}, \frac{1}{9}, \frac{1}{9}, 0, 0, 0 \right]$$

$$v2_{L1} = \frac{v2}{\|v2\|_1} = \left[\frac{1}{7}, \frac{1}{7}, \frac{1}{7}, \frac{1}{7}, 0, 0, 0, \frac{1}{7}, \frac{1}{7}, \frac{1}{7} \right]$$

Folosind norma L_2 (norma Euclidiană) obținem vectorii normalizați L_2 :

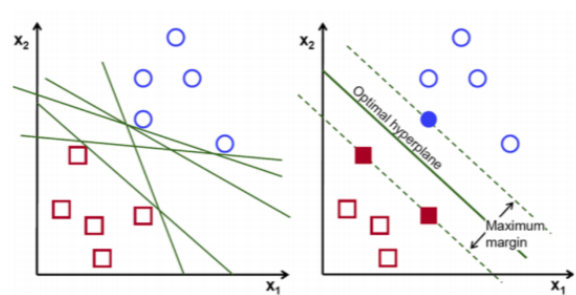
$$v1_{L2} = \frac{v1}{\|v1\|_2} = \left[\frac{1}{\sqrt{13}}, \frac{2}{\sqrt{13}}, \frac{1}{\sqrt{13}}, \frac{1}{\sqrt{13}}, \frac{2}{\sqrt{13}}, \frac{1}{\sqrt{13}}, \frac{1}{\sqrt{13}}, 0, 0, 0 \right]$$

$$v2_{L2} = \frac{v2}{\|v2\|_2} = \left[\frac{1}{\sqrt{7}}, \frac{1}{\sqrt{7}}, \frac{1}{\sqrt{7}}, \frac{1}{\sqrt{7}}, 0, 0, 0, \frac{1}{\sqrt{7}}, \frac{1}{\sqrt{7}}, \frac{1}{\sqrt{7}} \right]$$

2. Clasificatorul SVM

La curs a fost prezentat clasificatorul SVM, clasificator care încearcă să separe clase folosind un hiperplan de margine maximă (figura alăturată prezintă cazul binar, cu două clase).

Pentru implementarea acestui algoritm vom folosi biblioteca **scikit-learn**. Aceasta este dezvoltată în Python, fiind integrată cu NumPy și pune la dispoziție o serie de algoritmi optimizați pentru probleme de clasificare, regresie și clusterizare. Instalarea bibliotecii se face prin comanda sistem: **pip install -U scikit-learn**.

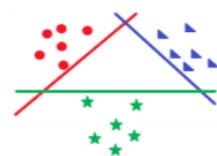


2.1 Detalii de implementare

Există două abordări pentru a clasifica datele aparținând mai multor clase: (1) one vs all și (2) one vs one. Vom exemplifica aceste două abordări pentru datele din figura alăturată, considerând exemplele din $n = 3$ clase.

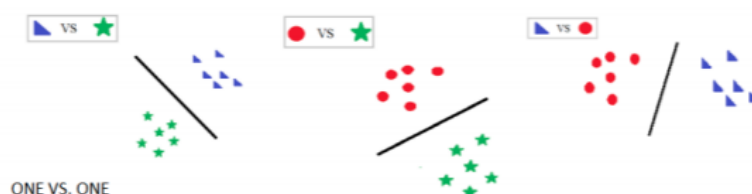


Abordarea *one vs all*: sunt antrenați n clasificatori, câte unul corespunzător fiecărei clase, care să o diferențieze pe aceasta de toate celelalte (toate celelalte exemple sunt privite ca aparținând aceleiași clase). Eticheta finală pentru un exemplu nou va fi dată de clasificatorul care a obținut scorul maxim.



Abordarea *one vs one* (implicită în scikit-learn): sunt antrenați $n \times (n-1)/2$ clasificatori câte unul corespunzător fiecărei perechi de câte două clase. Eticheta finală pentru un exemplu nou va fi cea care obține cele mai multe voturi pe baza acestor clasificatori.

2.2



Implementarea în scikit-learn

1. *Definirea modelului* se realizează folosind comanda:
`svm_model = sklearn.svm.SVC(C, kernel, ...)`
2. *Antrenarea modelului* se realizează folosind comanda:
`svm_model.fit(train_data, train_labels)`
3. *Predicția modelului* pentru exemple de testare se realizează folosind comanda:
`predicted_labels = svm_model.predict(test_data)`

unde:

parametrul C (float, default = 1.0) reprezintă parametrul de penalitate pentru eroarea făcută de model pe exemplele de antrenare. Valoarea lui C influențează alegerea hiperplanului optim de separare. Pentru valori mari ale lui C va fi ales un hiperplan cu o margine mai mică dar cu acuratețe cât mai mare a clasificării pe exemplele din mulțimea de antrenare. Pentru valori mai mici ale lui C va fi ales un hiperplan cu o margine mai mare, chiar dacă acesta duce la clasificarea greșită a unor exemple din mulțimea de antrenare.

parametrul kernel (string, default = 'linear') reprezintă tipul de funcție nucleu ales pentru modelarea suprafeței de decizie a clasificatorului. Implicit, este ales kernelul liniar, care definește un hiperplan. Alte valori definesc suprafețe neliniare: rbf, poly, etc.

parametrul train_data reprezintă mulțimea exemplilor de antrenare, stocată sub forma unei matrice, pe fiecare linie se află un exemplu de antrenare având drept componente caracteristicile extrase.

parametrul train_data reprezintă etichetele corespunzătoare fiecărui exemplu de antrenare din matricea, sub forma unui vector coloană.

parametrul test_data reprezintă mulțimea exemplilor de testare, stocate sub forma unei matrice, pe fiecare linie se află un exemplu de testare având drept componente caracteristicile extrase.

3. Clasificarea textelor folosind modelul BOW și clasificatorul SVM

Problema pe care o aveți de rezolvat în acest laborator este o problemă cu aplicabilitate practică și anume a clasificării emailurilor în spam sau inbox (non-spam). Veți clasifica emailuri scrise în limba engleză în cele două clase (eticheta 0 - inbox și eticheta 1 - spam). Veți lucra în cadrul acestui laborator cu un set de date ce conține 5172 de emailuri (3672 emailuri din inbox și 1500 emailuri de tip spam). În rezolvarea problemei de la acest

laborator veți folosi modelul Bag-of-Words pentru reprezentarea textelor și clasificatorul SVM.

Codul de la care porniți în acest laborator citește emailurile și etichetele lor și împarte cele 5172 de emailuri în trei submulțimi: aproximativ 3500 de exemple de antrenare, aproximativ 600 de exemple de validare și aproximativ 1000 de exemple de testare. Este important să remarcați că aveți etichete pentru toate aceste exemple și puteți evalua astfel performanța unui clasificator. Codul de la care porniți elimină semnele de punctuație reprezentând inițial textele ca liste de cuvinte (tokens) apoi calculează cele mai frecvente 1000 de cuvinte care apar în cele ~3500 exemple de antrenare.

Realizați următoarele:

1. Scrieți codul Python care calculează reprezentarea BOW pentru o matrice de exemple în funcție de cuvintele cele mai frecvente din texte.
2. Extrageți reprezentările BOW normalizate L_2 pentru exemplele de antrenare, validare, testare.
3. Normalizați datele voastre vectorizate reprezentare în funcție de norma L_2 . Pentru fiecare exemplu de text (antrenare, validare, testare) reprezentarea BOW a sa va fi dată de un vector linie de 1000 de componente. Reprezentarea BOW a matricei de exemple de antrenare va avea astfel dimensiunile 3500×1000 . Reprezentarea normalizată are proprietatea că fiecare linie are norma L_2 egală cu 1. Comparați doi clasificatori SVM pe datele ne-normalizate pe datele normalizate. Ce observați diferit?
4. Antrenați mai mulți clasificatori SVM liniari pe exemplele de antrenare folosind diverse valori pentru C . Spre exemplu puteți încerca valori din mulțimea $\{0.001, 0.01, 0.1, 1, 10, 100\}$. Alegeți cel mai bun clasificator SVM dintre cei antrenați, cel care are acuratețea cea mai mare pe mulțimea de exemple de validare.
5. Care este acuratețea clasificatorului vostru pe mulțimea de exemple de testare?
6. Încercați același clasificator cu metode diferite de procesare a datelor care: folosesc primele 100, 200, 300, 500 de cuvinte, care separă și după '\n', care elimină stop words, care păstrează punctuația. Comparați acuratețea pe setul de validare și pe cel de testare.