Dobre Bogdan-Mihai
Grupa 512

Raport Cercetare Proiect Text Mining

Aceasta este raportul de cercetare pentru proiectul 2 de l materia Information Retrieval and Text Mining.

Am incercat 5 combinatii de modele si metode de text vectorization care pareau promitatoare. Fiecare astfel de model este intr-un fisier distinct cu un nume sugestiv si fiecare fisier are 2 functii principale: train_and_predict si n_fold_cross_validation. O sa descriu cele 2 functii mai jos:

a) Prima functie are rolul de a antrena modelul pe datele de antrenare, de a prezice etichetele pentru datele de testare si de a afisa acuratetea si matricea de confuzie curenta pentru genurile muzicale.
b) A doua functie are rolul de a face un 5 fold cross validation folosind datele de antrenare, antrenand modelul alea pentru subsetul de antrenare de la acel pas si prezicand etichetele pentru subsetul de testare de la acel pas. De asemenea la fiecare pas afiseaza acuratetea curenta si matricea de confuzie curenta.

O sa detaliez cele 5 combinatii de modele si metode de text vectorization mai jos. Este important de mentionat ca atunci cand ma refer la datele de antrenare in lista de mai jos, ma refer la datele de antrenare din functia respectiva: in train_and_predict se folosesc toate datele de antrenare iar in n_fold_cross_validation se foloseste doar subsetul curent din de date de antrenare. De asemenea, cand ma refer la datele de testare, pentru train_and_predict ma refer la datele de testare originale din fisierul specific in timp ce in n_fold_cross_validation ma refer la subsetul de testare de la pasul curent. Urmeaza descrierile celor 5 combinatii:

1) Prima combinatie pe care am incercat-o este utilizarea stopwords pentru vectorizare si utilizarea unui xgboost ca model. A fost facut un

top al celor mai frecvente 20 de cuvinte de tip stopword din toate cantecele din setul de date de antrenare. Fiecarui cantec i-a fost asociat propriul top al cuvintelor de tip stopwords si un vector construit dupa urmatoarea regula: daca un cuvant din topul exemplului curent este pe pozitia i in acest top si pe pozitia j in topul global, atunci in vectorul asociat exemplului se pune valoarea j la pozitia i. Dupa constructia vectorului asociat fiecarui cantec se aplica modelul xgboost pe acestea si se calculeaza acuratetea in cele 2 functii. Acuratetea a variat la fiecare rulare intre 0.26 si 0.28. Am decis sa incerc sa folosesc alte metode de vectorizare dupa observarea rezultatului acestei combinatii.

2) A doua combinatie pe care am incercat-o este utilizarea word2vec si xgboost. Un word2vec creat folosind datele de antrenare este aplicat pe setul de date de antrenare si apoi se aplica un model de xgboost pe datele de antrenare. Dupa acea, se prezic etichetele datelor de testare. Se calculeaza acuratetea in cele 2 functii. Acuratetea a variat la fiecare rulare intre 0.28 si 0.30. Am decis sa incerc si tfidf in loc de word2vec pentru a observa daca acuratetea creste.
3) A treia combinatie pe care am incercat-o este utilizarea tfidf si xgboost. Un tfidf antrenat pe datele de antrenare este folosit pentru a vectoriza exemplele. Dupa se foloseste un xgboost antrenat pe datele de antrenare pentru a se prezice etichetele datelor de testare. Se calculeaza acuratetea pentru cele 2 functii. Acuratetea a variat la fiecare rulare intre 0.38 si 0.40, crescand fata de modelul precedent. Am decis sa incerc sa folosesc si alt model in afara de xgboost pentru a vedea daca acuratetea creste.
4) A patra combinatie pe care am incercat-o este utilizarea tfidf si neural networks. Un tfidf antrenat pe datele de antrenare este folosit pentru a vectoriza exemplele. Dupa se foloseste un neural network antrenat pe datele de antrenare pentru a se prezice etichetele datelor de testare. Se calculeaza acuratetea pentru cele 2 functii. Acuratetea a variat la fiecare rulare intre 0.32 si 0.34, scazand fata de modelul precedent. Am decis sa incerc sa folosesc si alta metoda de vectorizare impreuna cu modelul de neural netowrk pentru a vedea daca pot obtine o acuratete mai buna.

5) A cincea combinatie pe care am incercat-o este utilizarea unui layer de vectorizare de la tensorflow hub si neural networks. Un layer de tensorflow hub in modelul de neural network este folosit pentru a vectoriza exemplele. Dupa se folosesc restul de neural network layers pentru antrenarea pe datele de antrenare si pentru a se prezice etichetele datelor de testare. In afara de layerul de vectorizare mai sunt 2 layere: un dense layer cu 128 de neuroni si inca unul cu 64 de neuroni, ambele avand functia de activare ReLU. Dupa acestea mai exista un dense layer cu 10 neuroni (numarul de clase). Se folosesc optimizatorul Adam si functia de pierdere Sparse Categorical Crossentropy. Se calculeaza acuratetea pentru cele 2 functii. Acuratetea a variat la fiecare rulare intre 0.39 si 0.41, obtinand astfel cea mai buna performanta.

In concluzie, putem observa ca ultima combinatie, cea care foloseste un layer de tensorflow hub pentru vectorizare si 3 layere dense pentru clasificare, are cea mai mare acuratete din cele incercate pana acum. Putem considera o medie pentru aceasta acuratete ca fiind 0.40 si ca aceasta combinatie este cea mai buna din cele incercate. Privind matricea de confuzie, am observat ca mereu cantecele din genul rock sunt clasificate gresit in proportie vizibil mai mare decat celelalte. Drept urmare, o directie interesanta de cercetare ar fi sa se gaseasca cauza acestui fenomen si posibile solutii.