Documentație proiect Inteligență Artificială

Proiect Kaggle – Identificarea dialectului sursă a traducerii

Scopul competiției este de a prezice dialectul nativ al unui text pe baza traducerii acestuia în diferite limbi.

Submisia 1 – SVM (Support Vector Machines) + CountVectorizer (63,798%)

- Citim datele folosind pandas
- Codificăm etichetele din string in int, astfel **Ireland** va avea eticheta 0, **England** va avea eticheta 1, iar **Scotland** va avea eticheta 2 și vom aplica dicționarul *label2id* peste toate etichetele din train
- Preprocesarea datelor:
  - Extragerea informațiilor necesare din text
  - Eliminarea semnelor de punctuație
  - Impărțirea in cuvinte (Tokenizare)
- Aplicăm funcția de preprocesare intregului set de date
- Impărțim datele in train, validare și test, în ordinarea în care apar acestea
  - 20% date de test din total
  - 15% date de validare
- Permutăm indicii pentru a amesteca datele, pentru că există șanse ca datele originale sa fie ordonate într-un mod ce nu reflectă realitatea (de exemplu în ordinea etichetelor)
- Count Vectorizer, cu parametrii:
  - tokenizer = lambda (data e deja procesat, nu mai e nevoie de tokenizer aici)
  - preprocesor = lambda (data e deja procesat, nu mai e nevoie de tokenizer aici)
  - max_features = 100000
  - Facem antrenarea pe datele de antrenare vectorizate
- Antrenarea SVM
  - SVC (Linear Support Vector Classification)
    - Parametru de regularizare $C = 0.1$: puterea regularizării este invers proporțională cu C (trebuie să fie valoare strict pozitivă)
  - Facem antrenarea pe datele de antrenare vectorizate
  - Antrenarea a durat 15.71 secunde
  - Am obținut o acuratețe pe datele de test de 68.005%

Submisia 2 – SVM (Support Vector Machines) + Funcție de featurizre (62,391%)

- Citim datele
- Codificăm etichetele din string in int, astfel **Ireland** va avea eticheta 0, **England** va avea eticheta 1, iar **Scotland** va avea eticheta 2 și vom aplica dicționarul *label2id* peste toate etichetele din train

- Preprocesarea datelor:
  - Extragerea informațiilor necesare din text
  - Eliminarea semnelor de punctuație
  - Impărțirea in cuvinte (Tokenizare)
- Aplicăm funcția de preprocesare intregului set de date
- Impărțim datele in train, validare și test, în ordinarea în care apar acestea
  - 20% date de test din total
  - 15% date de validare
- Permutăm indicii pentru a amesteca datele, pentru că există șanse ca datele originale sa fie ordonate într-un mod ce nu reflectă realitatea (de exemplu în ordinea etichetelor)
- Bag of Words
  - vom număra numărul de apariții al tuturo cuvintelor din datele noastre
  - pentru o evaluare justă, nu ar fi indicat să includem și cuvintele din datele de test
  - construim funcții:
    - funcție care să returneze cele mai frecvente cuvinte;
    - funcție care construiesc dicționare (garantează o ordine pentru cuvintele caracteristice)
    - funcție de *featurize* – pentru un text preprocesat dat și un dicționar care mapează pentru fiecare poziție ce cuvânt îi corespunde, returnează un vector care reprezintă frecvențele fiecărui cuvânt
      1. numărăm toate cuvintele din text
      2. prealocăm un array care va reprezenta caracteristicile noastre
      3. umplem array-ul cu valorile obținute din counter: fiecare poziție din array trebuie să reprezinte frecvența aceluiași cuvânt din toate textele
    - funcție de *featurize* pentru mai multe texte (featurize_multi) – pentru un set de texte preprocesate și un dicționar care mapează pentru fiecare poziție ce cuvânt îi corespunde, returnează matricea trăsăturilor tuturor textelor
- Transformăm datele în format vectorial
- Facem experimente pe împărțire în train-valid-test
- Antrenarea SVM
  - SVC (Linear Support Vector Classification)
    - Parametru de regularizare $C = 0.25$: puterea regularizării este invers proporțională cu C (trebuie să fie valoare strict pozitivă)
  - Facem antrenarea pe datele de antrenare featurizate
  - Antrenarea a durat 12.39 secunde
  - Am obținut o acuratețe pe datele de test de 62.391%

- Reantrenăm modelul pe toate datele vectorizate prin funcția *featurize_multi* și obținem o acuratețe pe datele deja văzute de model de 64.625%
- Procesăm datele pentru a vectoriza predicții
- În urma a 5 fold cross-validation s-au obținut următoarele valori de acuratețe: 32.896%, 26.425%, 51.527%, 27.231%, 33.497%