UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA TEHNOLOGIA INFORMAȚIEI

Proiect Inteligență artificială
Complex Word Classification

Prof. Coord. Bogdan Alexe, Alexandra Diaconu

Buiciuc Ana-Maria, Grupa 351

Despre

Acest proiect are scopul de a identifica complexitatea unui cuvânt din limba engleză, problemă care a fost redusă la clasificare binară. Cuvintele complexe primesc eticheta 1, iar cele non-complexe primesc eticheta 0.

1. Submisie cu KNN

- Scor public: 0.60762 < baseline
- Scor privat: 0.59931
- Hiperparametru: numărul de vecini range(1,8,2)
- Caracteristici folosite:
    - Funcție pentru aflarea corpus-ului cuvântului corpus_feature(corpus)
    - Funcție pentru aflarea lungimii cuvântului length(word)
    - Funcție pentru aflarera numărului de vocale nr_vowels(word)
    - Funcție care verifică dacă cuvântul este titlu (prima literă este majusculă) is_title(word)
    - Funcție pentru căutarea cuvântului în Dale Chall is_dale_chall(word)
    - Funcție pentru aflarea numărului de silabe nr_syllables(word)
    - Funcție care împarte și returnează toate cuvintele din data frame get_all_tokens(df)
    - Funcție de frecvență a unui cuvânt word_frequency(word)
    - Funcție care reține feature-urile din Wordnet ale unui cuvânt synsets(word)
- Funcția get_word_structure_features(word) preia feature-urile unui cuvânt, feature-uri colectate cu funcțiile ajutătoare de mai sus (care nu au legătură cu Wordnet)
- Funcția get_wordnet_features(word) preia feature-urile Wordnet cuvântului, feature-uri colectate cu ajutorul funcției synsets.
- Timp de antrenare:

```
print(train_time)

0.2596266269683838
```

2. Submisie Naive Bayes

- Scor public: 0.76443
- Scor privat: 0.75205
- Parametri: default
- Caracteristici folosite:
    - Funcție pentru aflarea corpus-ului cuvântului corpus_feature(corpus)
    - Funcție pentru aflarea lungimii cuvântului length(word)
    - Funcție pentru aflarera numărului de vocale nr_vowels(word)
    - Funcție care verifică dacă cuvântul este titlu (prima literă este majusculă) is_title(word)
    - Funcție pentru căutarea cuvântului în Dale Chall is_dale_chall(word)

- Funcție pentru aflarea numărului de silabe nr_syllables(word)
- Funcție care împarte și returnează toate cuvintele din data frame get_all_tokens(df)
- Funcție de frecvență a unui cuvânt word_frequency(word)
- Funcție care verifică dacă cuvântul începe cu un prefix din lista făcută de mine has_prefixes(word)

Această listă conține prefixe științifice și medicale. Lista se găsește aici: https://unibucro0-my.sharepoint.com/:x:/g/personal/ana_buiciuc_s_unibuc_ro/Edz3cHbZbchApZi5QnVstqUBTNYMSxXbSGAZNBoylDBkOg?e=DWh7Mn.

Prefixele au fost preluate din următoarele surse:

- https://biolympiads.com/most-useful-biological-prefixes-and-suffixes-for-the-biology-olympiad/
- https://www.thoughtco.com/hydrocarbon-nomenclature-prefixes-608208
- https://chem.libretexts.org/Bookshelves/Analytical_Chemistry/Supplemental_Modules_(Analytical_Chemistry)/Quantifying_Nature/Units_of_Measure/Prefixes
- https://en.wikipedia.org/wiki/Category:Chemistry_suffixes
- https://ncela.ed.gov/files/uploads/43/13Sciencepreﬁxsuﬃx.pdf
- https://denalirx.com/drug-prefix-root-and-suffix/
    - Funcție care reține feature-urile din Wordnet ale unui cuvânt synsets(word)
    - Funcție care returnează numărul de sinonime no_of_synonims(word)
    - Funcție care returnează numărul de hipernime no_of_hypernyms(word)
- Funcția get_word_structure_features(word)preia feature-urile unui cuvânt, feature-uri colectate cu funcțiile ajutătoare de mai sus (care nu au legătură cu Wordnet)
- Funcția get_wordnet_features(word) preia feature-urile Wordnet cuvântului, feature-uri colectate cu ajutorul funcțiilor synsets, no_of_synonims, no_of_hypernyms.

- Timp de antrenare:

```
0.7381575367000988
Timpul de antrenare:  0.09257674217224121
```

- Acuratețe:

```
[231] model = GaussianNB()
      cv = KFold(n_splits = 10, random_state = 1, shuffle = True)
      print(cross_val_score(model, X_train, y_train, cv = cv, scoring='balanced_accuracy', n_jobs = -1).mean())
      model.fit(X_train,y_train)
      preds = model.predict(X_test)

      0.7381575367000988
```

- Matricea de confuzie:

```
[232] y_pred = cross_val_predict(model, X_train, y_train, cv=cv)
      conf_matrix = confusion_matrix(y_train, y_pred)
      print(conf_matrix)

      [[4232 2680]
       [ 104  646]]
```