Projekt 2 - Eksploracja Danych w Pythonie
Krystian Nowakowski
Styczeń 21, 2021

1. Wstęp

Zadanie polega na napisaniu klasyfikatora przewidującego ogólną ocenę na podstawie tekstowej opinii. Zbiór uczący znajduje się w pliku "reviews_train.csv".
Do wykonania zadania wykorzystano takie biblioteki jak:

- *scikit-learn* - algorytmy klasyfikacji, metryki
- *nltk* - badanie polaryzacji opinii
- *pandas* - przetwarzanie danych
- *pickle* - zapisywanie i wczytywanie modeli

Całość implementacji dostępna jest w jupyter notebook *"Projekt II - Eksploracja danych w Pythonie"* natomiast w celu przetestowania rozwiązania na nowy zbiorze danych należy uruchomić skrypt *"test_model.sh"* podając jako parametr pliku z danymi w formacie csv. Przykład: "*test_model.sh test_dataset.csv"*.

2. Przetwarzanie danych i wybór atrybutów

Z początkowego zbioru danych wybrano następujące atrybuty:

- *reviewText* oraz *summary* - na podstawie tekstów obliczono dwie osobne przewidywane oceny. Do tego celu użyto biblioteki *ntlk* i metody bag-of-words. Poszczególne wyrazy zostały wyznaczone za pomocą RegexpTokenizer.
- *unixReviewTime* - czas udostępnienia recenzji
- *helpful* - atrybut rozbito na trzy osobne kolumny.
    - *helpful* - liczba oznaczeń jako pomocna
    - *nothelpful* - liczba oznaczeń jako niepomocna
    - *helpful_diff* - różnica helpful-nothelpful

Zrezygnowano z użycia pozostałych atrybutów ponieważ ich wykorzystanie nie przyniosło znaczących korzyści.

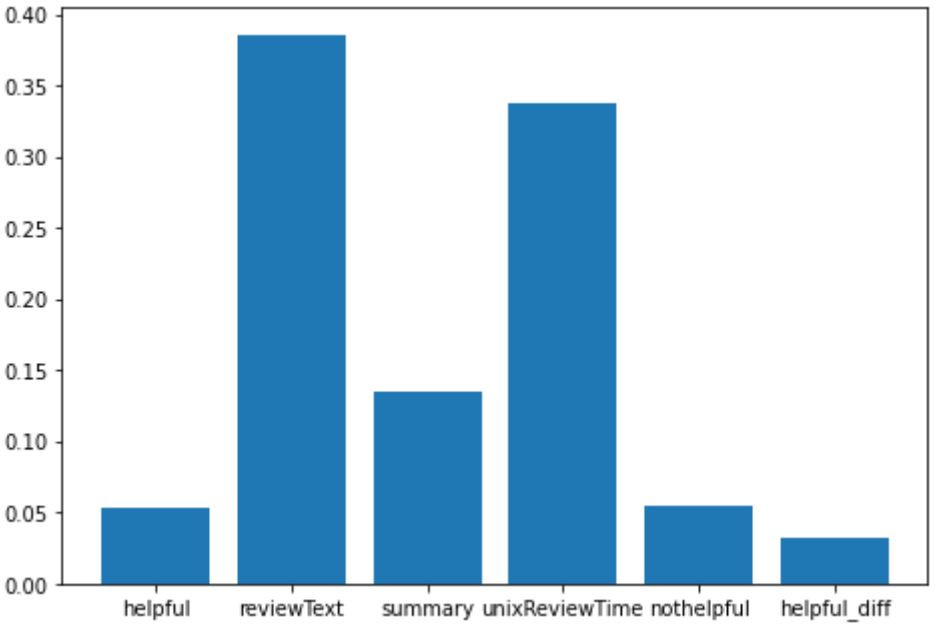
Oba pomocnicze modele wykorzystywały klasyfikator MultinomialNB i uzyskały trafność ponad 58%.

3. Model klasyfikatora

Jako głowny klasyfikator do oceny tekstowych opinii wybrano DecisionTreeClassifier. Potrzebował on niewiele czasu na wyuczenie, osiągał satysfakcjonujące wyniki oraz udostępniał prostą metodę dającą wgląd do stopnia wykorzystywania atrybutów.

Zbiór danych podzielono na zbiór uczący i testowy w proporcjach odpowiednio 80% i 20%. Osiągnięto w ten sposób trafność na poziomie 59,7%.

Najbardziej istotnymi atrybutami dla modelu były *reviewText* oraz *unixReviewTime*.



4. Wnioski

Dzięki zastosowaniu trzech modeli uczenia maszynowego uzyskano poprawę o 9% względem najprostszego algorytmu wybierającego klasę większościową.
Dalszą poprawę mogłoby przynieść zastosowanie lepszej metody tokenizacji, wykorzystanie lematyzacja i stemmingu lub sprawdzenie większej ilośc kosztownych klasyfikatorów jakich jak np.: RandomForestClassifier.