SPRAWOZDANIE

Zajęcia: Nauka o danych II

Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 9 Data 30.06.2025 Temat: Implementacja zaawansowanych metod analizy tekstu w Pythonie	Imię Nazwisko Hubert Mentel Informatyka II stopień, niestacjonarne, 2 semestr, gr.1a

1. Zadanie:

1. Załaduj korpus tekstów (np. recenzje filmów).
2. Przeprowadź tokenizację i wektoryzację tekstów.
3. Zastosuj Word2Vec i porównaj podobieństwo semantyczne między słowami.
4. Wykonaj analizę tematów przy pomocy LDA.
5. Przedstaw dane w 2D przy pomocy SVD.

Pliki dostępne są pod linkiem:
https://github.com/HubiPX/NOD/tree/master/NOD2/Zadanie%209

2. Opis programu opracowanego (kody źródłowe, zrzuty ekranu)

[4]:
```
texts = []
with open("Reviews.csv", encoding="utf-8") as f:
    for i, line in enumerate(f):
        if i >= 1000:
            break
        if line.strip():
            texts.append(line.strip())
```

[5]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(max_features=1000)
tfidf_matrix = vectorizer.fit_transform(texts)
print(tfidf_matrix.toarray())
feature_names = vectorizer.get_feature_names_out()
for idx, word in enumerate(feature_names):
    print(f"{idx}: {word}")
```

```
19: actual
20: actually
21: add
22: added
23: addicted
24: adding
25: addition
26: adult
27: advertised
28: after
29: aftertaste
30: again
31: agave
32: ago
33: ahmad
34: all
35: allergic
36: allergies
37: almost
```

[6]:
```
from gensim.models import Word2Vec
from nltk.tokenize import word_tokenize
import nltk

nltk.download('punkt')
sentences = [doc.lower().split() for doc in texts]
model = Word2Vec(sentences, vector_size=100, window=5, min_count=1)  # min_count=1 żeby działało nawet na małym zbiorze
print("Podobieństwo 'food' i 'good':", model.wv.similarity('food', 'good'))  # wartości od -1 do 1
print("Najbardziej podobne do 'food':")
print(model.wv.most_similar('food'))
```

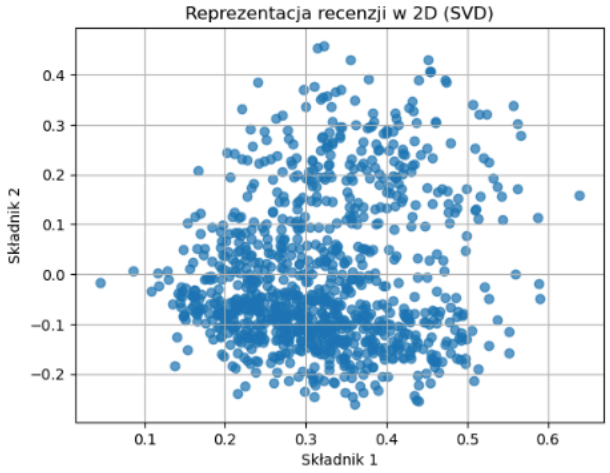
```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\48664\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
Podobieństwo 'food' i 'good': 0.99985147
Najbardziej podobne do 'food':
[('just', 0.999893307685852), ('in', 0.9998893737792969), ('and', 0.9998839497566223), ('also', 0.9998823404312134), ('from', 0.9998819231987), ('our',
0.9998814463615417), ('for', 0.9998805522918701), ('was', 0.9998801350593567), ('or', 0.999879777431488), ('but', 0.9998793601989746)]
```

[ ]:

[7]:
```
from sklearn.decomposition import TruncatedSVD
import matplotlib.pyplot as plt

# Redukcja wymiarowości do 2D
svd = TruncatedSVD(n_components=2)
X_2d = svd.fit_transform(tfidf_matrix)

# Rysowanie wykresu
plt.scatter(X_2d[:, 0], X_2d[:, 1], alpha=0.7)
plt.title("Reprezentacja recenzji w 2D (SVD)")
plt.xlabel("Składnik 1")
plt.ylabel("Składnik 2")
plt.grid(True)
plt.show()
```



[ ]:

6. Wnioski

Uzyskane wyniki wskazują, że na niewielkim podzbiorze 1000 recenzji zastosowane techniki reprezentacji tekstu jedynie częściowo oddają sens semantyczny danych: model Word2Vec, zbudowany na tak małej próbce, zawyża podobieństwo między często współwystępującymi słowami („food” i „good” ≈ 0,999) i zwraca w większości mało-informacyjne wyrazy funkcyjne jako najbliższe „food”, co sugeruje potrzebę szerszego korpusu oraz usunięcia stop-słów; z kolei wektor TF-IDF zredukowany do 2 wymiarów metodą SVD tworzy chmurę punktów bez wyraźnych klastrów, co potwierdza, że w tak niskim wymiarze nie ujawnia się struktura tematyczna recenzji, a także że dla skuteczniejszej analizy (np. klasteryzacji lub klasyfikacji sentymentu) warto zwiększyć liczbę cech, zastosować lepsze odfiltrowanie rzadkich słów oraz rozważyć bardziej zaawansowane modele, jak doc2vec lub transformery, wymagające jednak znacznie większej liczby przykładów.