NLP-Part_2

November 13, 2024

0.1 Part 2)

- For the same person from step 1), use the Wikipedia API to access the whole content of that person's Wikipedia page.
- The goal of part 2) is to produce the capability to:
  1. For that Wikipedia page determine the sentiment of the entire page
  2. Print out the Wikipedia article
  3. Collect the Wikipedia pages from the 10 nearest neighbors in Step 1)
  4. Determine the nearness ranking of these 10 to your main subject based on their entire Wikipedia page
  5. Compare the nearest ranking from Step 1) with the Wikipedia page nearness ranking

[13]:
```
!pip install wikipedia
!pip install -U textblob
!python -m textblob.download_corpora

import wikipedia
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.decomposition import TruncatedSVD
import matplotlib.pyplot as plt
from textblob import TextBlob
```

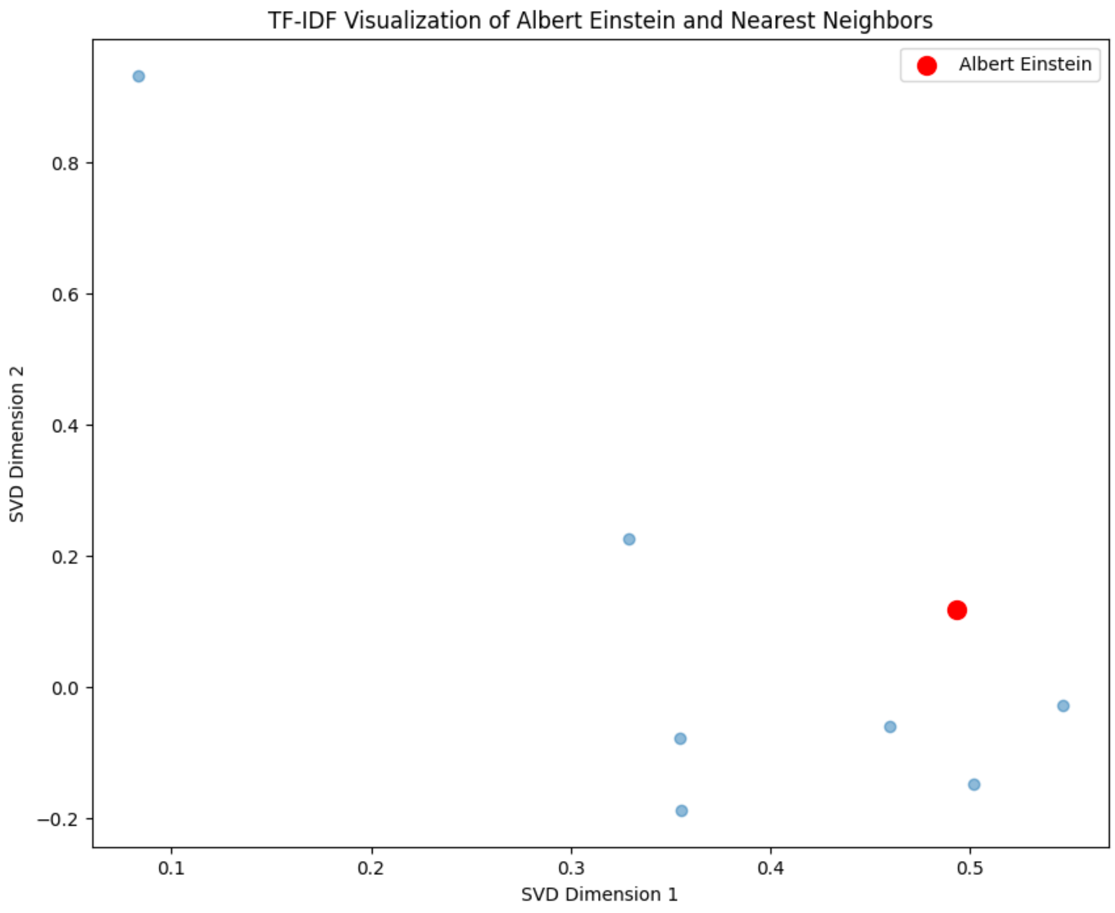
```
Requirement already satisfied: wikipedia in /usr/local/lib/python3.12/site-
packages (1.4.0)
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.12/site-
packages (from wikipedia) (4.12.3)
Requirement already satisfied: requests<3.0.0,>=2.0.0 in
/usr/local/lib/python3.12/site-packages (from wikipedia) (2.32.3)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.12/site-packages (from requests<3.0.0,>=2.0.0->wikipedia)
(3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/site-
packages (from requests<3.0.0,>=2.0.0->wikipedia) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.12/site-packages (from requests<3.0.0,>=2.0.0->wikipedia)
(2.2.3)
```

```
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.12/site-packages (from requests<3.0.0,>=2.0.0->wikipedia)
(2024.8.30)
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.12/site-
packages (from beautifulsoup4->wikipedia) (2.6)
WARNING: Running pip as the 'root' user can result in broken permissions

and conflicting behaviour with the system package manager, possibly rendering

your system unusable.It is recommended to use a virtual environment instead:

https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know

what you are doing and want to suppress this warning.

Requirement already satisfied: textblob in /usr/local/lib/python3.12/site-
packages (0.18.0.post0)
Requirement already satisfied: nltk>=3.8 in /usr/local/lib/python3.12/site-
packages (from textblob) (3.9.1)
Requirement already satisfied: click in /usr/local/lib/python3.12/site-packages
(from nltk>=3.8->textblob) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.12/site-packages
(from nltk>=3.8->textblob) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in
/usr/local/lib/python3.12/site-packages (from nltk>=3.8->textblob) (2024.11.6)
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/site-packages
(from nltk>=3.8->textblob) (4.67.0)
WARNING: Running pip as the 'root' user can result in broken permissions

and conflicting behaviour with the system package manager, possibly rendering

your system unusable.It is recommended to use a virtual environment instead:

https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know

what you are doing and want to suppress this warning.

[nltk_data] Downloading package brown to /root/nltk_data…
[nltk_data]   Package brown is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package conll2000 to /root/nltk_data…
[nltk_data]   Package conll2000 is already up-to-date!
[nltk_data] Downloading package movie_reviews to /root/nltk_data…
[nltk_data]   Package movie_reviews is already up-to-date!
Finished.
```

```
[14]: def get_wikipedia_content(person_name):
          try:
              page = wikipedia.page(person_name)
              return page.content
          except wikipedia.exceptions.PageError:
              return None

      def clean_text(text):
          text = text.lower()
          return text

      def analyze_wikipedia_content(person_name, nearest_neighbors):
          # Fetch Wikipedia content for the main person
          main_content = get_wikipedia_content(person_name)

          if main_content is None:
              print(f"No Wikipedia page found for {person_name}")
              return None, None, None, None

          # Clean the main content
          main_content = clean_text(main_content)

          # Collect and clean Wikipedia pages from the nearest neighbors
          neighbor_contents = []
          for neighbor in nearest_neighbors:
              content = get_wikipedia_content(neighbor)
              if content is not None:
                  neighbor_contents.append(clean_text(content))

          if len(neighbor_contents) < 2:
              print("Not enough data to perform analysis. Skipping ranking.")
              return None, None, None, None

          # Create Bag of Words and TF-IDF representations
          bow_vectorizer = CountVectorizer(stop_words='english')
          bow_matrix = bow_vectorizer.fit_transform([main_content] +␣
        ↪neighbor_contents)

          tfidf_vectorizer = TfidfVectorizer(stop_words='english')
          tfidf_matrix = tfidf_vectorizer.fit_transform([main_content] +␣
        ↪neighbor_contents)

          # Calculate cosine similarity
          bow_similarity = cosine_similarity(bow_matrix[0].reshape(1, -1),␣
        ↪bow_matrix[1:])
          tfidf_similarity = cosine_similarity(tfidf_matrix[0].reshape(1, -1),␣
        ↪tfidf_matrix[1:])
```

```
    # Calculate rankings
    bow_ranking = np.argsort(bow_similarity[0])[::-1]
    tfidf_ranking = np.argsort(tfidf_similarity[0])[::-1]

    # Create ranked lists of neighbors
    bow_wikipedia_ranking = [nearest_neighbors[i] for i in bow_ranking]
    tfidf_wikipedia_ranking = [nearest_neighbors[i] for i in tfidf_ranking]

    return main_content, bow_wikipedia_ranking, tfidf_wikipedia_ranking,␣
  ↪tfidf_matrix

# Test the function
person_name = "Albert Einstein"  # Use a well-known person as an example
nearest_neighbors = ["Marie Curie", "Isaac Newton", "Galileo Galilei", "Stephen␣
  ↪Hawking", "Richard Feynman", "Nikola Tesla", "Charles Darwin", "Aristotle",␣
  ↪"Archimedes", "Leonardo da Vinci"]

main_content, bow_wikipedia_ranking, tfidf_wikipedia_ranking, tfidf_matrix =␣
  ↪analyze_wikipedia_content(person_name, nearest_neighbors)

if main_content is not None:
    print(f"\nBoW Ranking:")
    for i, neighbor in enumerate(bow_wikipedia_ranking):
        print(f"{i+1}. {neighbor}")

    print("\nTF-IDF Ranking:")
    for i, neighbor in enumerate(tfidf_wikipedia_ranking):
        print(f"{i+1}. {neighbor}")

    # Visualization
    svd = TruncatedSVD(n_components=2, random_state=42)
    X_svd = svd.fit_transform(tfidf_matrix)

    plt.figure(figsize=(10, 8))
    plt.scatter(X_svd[1:, 0], X_svd[1:, 1], alpha=0.5)
    plt.scatter(X_svd[0, 0], X_svd[0, 1], color='red', s=100, label=person_name)
    plt.title(f"TF-IDF Visualization of {person_name} and Nearest Neighbors")
    plt.xlabel("SVD Dimension 1")
    plt.ylabel("SVD Dimension 2")
    plt.legend()
    plt.show()
else:
    print(f"No Wikipedia page found for {person_name}. Unable to perform␣
  ↪analysis.")
```

```
BoW Ranking:
1. Marie Curie
2. Stephen Hawking
3. Galileo Galilei
4. Richard Feynman
5. Charles Darwin
6. Nikola Tesla
7. Isaac Newton

TF-IDF Ranking:
1. Marie Curie
2. Richard Feynman
3. Stephen Hawking
4. Galileo Galilei
5. Charles Darwin
6. Nikola Tesla
7. Isaac Newton
```


TF-IDF Visualization of Albert Einstein and Nearest Neighbors
● Albert Einstein
SVD Dimension 2
SVD Dimension 1