NLP - EXP - 8

Atharva Prashant Pawar (9427) - [Batch - D]

@pip Installations

```
!pip install gensim
!pip install textblob
!pip install spacy
!python -m spacy download en_core_web_sm
```

```
Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (1.23.5)
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (2.31.0)
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-packages (from spacy) (1.10
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.1.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy) (67.7.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (23.2)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy) (3.3.0)
Requirement already satisfied: typing-extensions>=4.2.0 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,<3
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy) (3.4
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spacy) (0
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spa
Requirement already satisfied: click<9.0.0,>=7.1.1 in /usr/local/lib/python3.10/dist-packages (from typer<0.10.0,>=0.3.0->spacy)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy) (2.1.3)
2023-10-12 14:53:23.463879: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use avail
To enable the following instructions: AVX2 AVX512F FMA, in other operations, rebuild TensorFlow with the appropriate compiler fla
2023-10-12 14:53:25.957863: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
2023-10-12 14:53:29.846155: I tensorflow/compiler/xla/stream_executor/cuda/cuda_gpu_executor.cc:995] successful NUMA node read fr
2023-10-12 14:53:29.846775: I tensorflow/compiler/xla/stream_executor/cuda/cuda_gpu_executor.cc:995] successful NUMA node read fr
2023-10-12 14:53:29.847011: I tensorflow/compiler/xla/stream_executor/cuda/cuda_gpu_executor.cc:995] successful NUMA node read fr
Collecting en-core-web-sm==3.6.0
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.6.0/en_core_web_sm-3.6.0-py3-none-any.
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.8/12.8 MB 40.2 MB/s eta 0:00:00
Requirement already satisfied: spacy<3.7.0,>=3.6.0 in /usr/local/lib/python3.10/dist-packages (from en-core-web-sm==3.6.0) (3.6.1
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->er
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-cor
Requirement already satisfied: thinc<8.2.0,>=8.1.8 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-c
Requirement already satisfied: typer<0.10.0,>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core
Requirement already satisfied: pathy>=0.10.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-web-sm
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-
Requirement already satisfied: numpy>=1.15.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-web-sm
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-c
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-web-sm==3.6.0
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-web-sm==3
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-core-web-
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy<3.7.0,>=3.6.0->en-c
Requirement already satisfied: typing-extensions>=4.2.0 in /usr/local/lib/python3.10/dist-packages (from pydantic!=1.8,!=1.8.1,<3
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0-
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy<3.7.0
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spacy<3.7
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spa
Requirement already satisfied: click<9.0.0,>=7.1.1 in /usr/local/lib/python3.10/dist-packages (from typer<0.10.0,>=0.3.0->spacy<3
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy<3.7.0,>=3.6.0->en-c
✓ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
```

@Imports

```
import pandas as pd
import numpy as np
import re
# from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

from sklearn.feature_extraction.text import CountVectorizer
```

1. Bag of words: Extracts features from the text

```
corpus = [
    "This is the first document.",
    "This document is the second document.",
    "And this is the third one.",
    "Is this the first document?"
]

vectorizer = CountVectorizer()

# Fit and transform the text data into a bag of words
X = vectorizer.fit_transform(corpus)

# Get the feature names (words)
feature_names = vectorizer.get_feature_names_out()

# Convert the bag of words matrix to a dense array
X_dense = X.toarray()

# Display the feature names and the BoW matrix
print("Feature names:")
print(feature_names)
print("\nBag of Words matrix:")
print(X_dense)
```

```
Feature names:
['and' 'document' 'first' 'is' 'one' 'second' 'the' 'third' 'this']

Bag of Words matrix:
[[0 1 1 1 0 0 1 0 1]
 [0 2 0 1 0 1 1 0 1]
 [1 0 0 1 1 0 1 1 1]
 [0 1 1 1 0 0 1 0 1]]
```

TF-IDF (Term Frequency-Inverse Document Frequency)

2. TF-IDF: Information retrieval, keyword extraction

```
from sklearn.feature_extraction.text import TfidfVectorizer

# Sample text data
corpus = [
    "This is the first document.",
    "This document is the second document.",
    "And this is the third one.",
    "Is this the first document?"
]

# Initialize the TfidfVectorizer with options
tfidf_vectorizer = TfidfVectorizer()

# Fit and transform the text data into TF-IDF features
tfidf_matrix = tfidf_vectorizer.fit_transform(corpus)

# Get the feature names (words)
feature_names = tfidf_vectorizer.get_feature_names_out()

# Convert the TF-IDF matrix to a dense array
tfidf_dense = tfidf_matrix.toarray()

# Display the feature names and the TF-IDF matrix
print("Feature names:")
print(feature_names)
print("\nTF-IDF matrix:")
print(tfidf_dense)
```

```
Feature names:
['and' 'document' 'first' 'is' 'one' 'second' 'the' 'third' 'this']

TF-IDF matrix:
[[0.         0.46979139 0.58028582 0.38408524 0.         0.
  0.38408524 0.         0.38408524]
 [0.         0.6876236  0.         0.28108867 0.         0.53864762
  0.28108867 0.         0.28108867]
 [0.51184851 0.         0.         0.26710379 0.51184851 0.
```

```
 0.26710379 0.51184851 0.26710379]
[0.         0.46979139 0.58028582 0.38408524 0.         0.
 0.38408524 0.         0.38408524]]
```

3. Word2Vec: Semantic analysis task

```
from textblob import TextBlob

text = "I love this product! It's amazing."

blob = TextBlob(text)
sentiment_score = blob.sentiment.polarity  # Ranges from -1 (negative) to 1 (positive)

# Classify sentiment as positive, negative, or neutral
if sentiment_score > 0:
    sentiment_label = "Positive"
elif sentiment_score < 0:
    sentiment_label = "Negative"
else:
    sentiment_label = "Neutral"

print(f"Accuracy: {round(sentiment_score * 100, 2)}%")
print(f"Sentiment: {sentiment_label}")
```

```
Accuracy: 61.25%
Sentiment: Positive
```

```
import gensim.downloader as api

# Download the pre-trained Word2Vec model (may take some time)
word2vec_model = api.load("word2vec-google-news-300")
```

```
[==================================================] 100.0% 1662.8/1662.8MB downloaded
```

4. GloVe: Word analogy, named entity recognition tasks

```
# Install wget (if not already installed)
!pip install wget

# Use wget to download the GloVe embeddings zip file
import wget

# Specify the URL of the GloVe embeddings file
glove_url = "https://nlp.stanford.edu/data/glove.6B.zip"

# Specify the destination path to save the file
destination_path = "/content/glove.6B.zip"  # You can change this path if needed

# Download the file
wget.download(glove_url, destination_path)

# Verify that the file has been downloaded
import os
if os.path.exists(destination_path):
    print("GloVe embeddings file downloaded successfully.")
else:
    print("Download failed. Please check the URL or your internet connection.")
```

```
Collecting wget
  Downloading wget-3.2.zip (10 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: wget
  Building wheel for wget (setup.py) ... done
  Created wheel for wget: filename=wget-3.2-py3-none-any.whl size=9655 sha256=4913b9eacb787d4c1eb7bb9c7f0f52bb40f47f71a504f17acf57d
  Stored in directory: /root/.cache/pip/wheels/8b/f1/7f/5c94f0a7a505ca1c81cd1d9208ae2064675d97582078e6c769
Successfully built wget
Installing collected packages: wget
Successfully installed wget-3.2
GloVe embeddings file downloaded successfully.
```

```
# Unzip the downloaded file
import zipfile

# Specify the destination directory for unzipping
```

```
unzip_destination_dir = "/content/glove"

# Unzip the file
with zipfile.ZipFile(destination_path, 'r') as zip_ref:
    zip_ref.extractall(unzip_destination_dir)
```

Word Analogy (e.g., "king - man + woman = queen"):

```
# Now you can load the GloVe word vectors
import numpy as np

# Load pre-trained GloVe word vectors
glove_file = '/content/glove/glove.6B.100d.txt'  # Update with the path to the unzipped GloVe file
word_vectors = {}
with open(glove_file, 'r', encoding='utf-8') as f:
    for line in f:
        values = line.split()
        word = values[0]
        vector = np.asarray(values[1:], dtype='float32')
        word_vectors[word] = vector

# Define a function to perform word analogy (your existing code)
def word_analogy(word1, word2, word3):
    try:
        vec1 = word_vectors[word1]
        vec2 = word_vectors[word2]
        vec3 = word_vectors[word3]

        analogy_vec = vec2 - vec1 + vec3

        # Find the most similar word to the analogy vector
        analogy_word = None
        min_distance = float('inf')
        for word, vec in word_vectors.items():
            distance = np.linalg.norm(vec - analogy_vec)
            if distance < min_distance:
                min_distance = distance
                analogy_word = word

        return analogy_word
    except KeyError:
        return "One or more words not found in the vocabulary."

# Example word analogy
word1 = "king"
word2 = "man"
word3 = "woman"
analogy_result = word_analogy(word1, word2, word3)
print(f"{word1} - {word2} + {word3} = {analogy_result}")
```

```
king - man + woman = woman
```

NER : Named Entity Recognition - tasks

```
import spacy

# Load the spaCy model for English
nlp = spacy.load("en_core_web_sm")

# Text to analyze
text = "Apple Inc. is an American multinational technology company headquartered in Cupertino, California. " \
       "It was founded by Steve Jobs, Steve Wozniak, and Ronald Wayne on April 1, 1976."

# Process the text with spaCy
doc = nlp(text)

# Extract named entities and their labels
for ent in doc.ents:
    print(f"Entity: {ent.text}, Label: {ent.label_}")
```

```
Entity: Apple Inc., Label: ORG
Entity: American, Label: NORP
Entity: Cupertino, Label: GPE
Entity: California, Label: GPE
Entity: Steve Jobs, Label: PERSON
Entity: Steve Wozniak, Label: PERSON
Entity: Ronald Wayne, Label: PERSON
Entity: April 1, 1976, Label: DATE
```