Homework 1

Yu-Chieh Kuo B07611039[†]

[†]Department of Information Management, National Taiwan University

Usage

```
cd B07611039
cp -R /PA1-data .
mkdir out
mkdir output
python -V # Python 3.7.12 in my environment
pip install sklearn
# Requirement: sklearn, numpy, os
python3 pa1.py
```

Result and Explanation

Cosine similarity

The cosine similarity is 0.19991361508978656.

Details

The equirements of this assignment are sklearn, numpy and os. Check these packages are installed before execution.

First, I read data into a list and split lines. Then I use the functions in sklearn package to transform TFIDF vectors. Note that in the statement, it's mentioned to lowercase words and filter out English stopwords, I, therefore, use the option in TFidfVectorizer to achieve the requirement. Code is represented as below.

```
TFIDF_vectorizer = TfidfVectorizer(stop_words='english', lowercase=True)
TFIDF_vectors = TFIDF_vectorizer.fit_transform(data)
TFIDF_vectors
# <1095x19130 sparse matrix of type '<class 'numpy.float64'>'
#   with 200585 stored elements in Compressed Sparse Row format>
```

After constructing TFIDF vectors, I write doc1.txt and doc3.txt. Note that these two documents should represent the amount of terms and all terms with **non-zero** tf-idf among TFIDV vectors. Consequently, the data should be clean. I used to used if-condition to remove all terms with **0** tf-idf, but it doesn't make sense too much. My classmate tells me that it's available in python to check whether the value is zero and remove terms automatically. Code is represented as below.

```
print((TFIDF_vectors.toarray()[0] != 0).sum(), file = f)
print("t_index,tf-idf", file = f)
index = np.arange(TFIDF_vectors.shape[1])[(TFIDF_vectors.toarray()[0] != 0)]
tfidf = (TFIDF_vectors.toarray()[0])[(TFIDF_vectors.toarray()[0] != 0)]
for t_index, tfidf_value in zip(index, tfidf):
    print(f"{t_index},{tfidf_value}", file = f)
```

It follows the same process to make doc1.txt and doc3.txt. The only difference is the index of TDIFD vectors. To doc1.txt, the index is 0 and 2 otherwise.

Finally, use function to calculate cosine similarity directly.

```
cosine_similarity(TFIDF_vectors, TFIDF_vectors)[0][1]
```

The result is about 0.19991361508978656.