Applied Machine Learning Homework 4

Due 12/15/21 11:59PM EST

Q1: Natural Language Processing

We will train a supervised training model to predict if a tweet has a positive or negative sentiment.

Dataset loading & dev/test splits

1.1) Load the twitter dataset from NLTK library

In [1]:
```
import nltk
nltk.download('twitter_samples')
from nltk.corpus import twitter_samples
```

```
[nltk_data] Downloading package twitter_samples to /root/nltk_data...
[nltk_data]   Package twitter_samples is already up-to-date!
```

1.2) Load the positive & negative tweets

In [2]:
```
all_positive_tweets = twitter_samples.strings('positive_tweets.json')
all_negative_tweets = twitter_samples.strings('negative_tweets.json')
```

1.3) Create a development & test split (80/20 ratio):

In [3]:
```
#code here
import pandas as pd
from sklearn.model_selection import train_test_split
df = pd.DataFrame({'review':all_positive_tweets + all_negative_tweets})
y = ['positive']*5000 + ['negative']*5000
dev_text, test_text, dev_y, test_y = train_test_split(df, y, test_size = 0.2, random_state = 0)
```

Data preprocessing

We will do some data preprocessing before we tokenize the data. We will remove `#` symbol, hyperlinks, stop words & punctuations from the data. You can use the `re` package in python to find and replace these strings.

1.4) Replace the `#` symbol with '' in every tweet

In [4]:
```
#code here
dev_text['review'] = dev_text['review'].apply(lambda x: x.replace("#", ''))
test_text['review'] = test_text['review'].apply(lambda x: x.replace("#", ''))
dev_text[['#' in review for review in dev_text.review]].head()
```

Out[4]:

	review

1.5) Replace hyperlinks with '' in every tweet

In [5]:
```
#code here
import re
dev_text['review'] = dev_text['review'].apply(lambda x: re.sub(r'http\S+', '', x))
test_text['review'] = test_text['review'].apply(lambda x: re.sub(r'http\S+', '', x))
dev_text[['http' in review for review in dev_text.review]].head()
```

Out[5]:

	review

1.6) Remove all stop words

In [6]:
```
#code here
from sklearn.feature_extraction.text import ENGLISH_STOP_WORDS

stop_words = set(ENGLISH_STOP_WORDS)

def remove_stop_words(sentence):
  return (" ").join([word for word in sentence.split(' ') if word.lower() not in stop_words])

dev_text['review'] = dev_text['review'].apply(lambda x: remove_stop_words(x))
test_text['review'] = test_text['review'].apply(lambda x: remove_stop_words(x))
dev_text.head()
```

Out[6]:

	review
7389	@RafaelAllmark I've supporting start ignored :...
9275	@KirkHerbstreit Braxton gone pro! :(feel bad...
2995	@WilliamHC3 Hi! like @imPastel concert? Let kn...
5316	@BOYBANDSFTCARA sure!!! Sorry :(x
356	Good luck "LizaMinnelli upcoming UK appearance...

1.7) Remove all punctuations

In [7]:
```
#code here
import string
punct = set(string.punctuation)

def remove_punct(sentence):
  return ''.join(ch for ch in sentence if ch not in punct)

dev_text['review'] = dev_text['review'].apply(lambda x: remove_punct(x))
test_text['review'] = test_text['review'].apply(lambda x: remove_punct(x))
dev_text.head()
```

Out[7]:

	review
7389	RafaelAllmark Ive supporting start ignored le...
9275	KirkHerbstreit Braxton gone pro feel bad him...
2995	WilliamHC3 Hi like imPastel concert Let know c...
5316	BOYBANDSFTCARA sure Sorry x
356	Good luck LizaMinnelli upcoming UK appearances...

1.8) Apply stemming on the development & test datasets using Porter algorithm

In [8]:
```
#code here
import nltk
nltk.download('punkt')

from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.stem import PorterStemmer

porter = PorterStemmer()

def stem_sentence(sentence):
  token_words = word_tokenize(sentence)
  stem_sentence = [porter.stem(word) for word in token_words]
  return " ".join(stem_sentence)

dev_text['review'] = dev_text['review'].apply(lambda x: stem_sentence(x))
test_text['review'] = test_text['review'].apply(lambda x: stem_sentence(x))
dev_text.head()
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[8]:

	review
7389	rafaelallmark ive support start ignor let toda...
9275	kirkherbstreit braxton gone pro feel bad him u...
2995	williamhc3 Hi like impastel concert let know c...
5316	boybandsftcara sure sorri x
356	good luck lizaminnelli upcom UK appear garyjho...

Model training

1.9) Create bag of words features for each tweet in the development dataset

In [9]:
```
#code here
from sklearn.feature_extraction.text import CountVectorizer
vector = CountVectorizer()
dev_text_bow = vector.fit_transform(dev_text['review'])
test_text_bow = vector.transform(test_text['review'])
dev_text_bow
```

Out[9]:
```
<8000x14660 sparse matrix of type '<class 'numpy.int64'>'
        with 47397 stored elements in Compressed Sparse Row format>
```

In [10]:
```
vector.get_feature_names_out()[:50]
```

Out[10]:
```
array(['00128835', '009', '00kouhey00', '0100', '0115am', '0116am',
       '01282', '0129ann', '01482', '02', '02079', '02392441234', '0272',
       '0345', '0388', '0702am', '071', '0717am', '0717pm', '0724pm',
       '0725pm', '0732pm', '0734am', '0878', '089624641747', '0lliebudz',
       '0plate', '10', '100', '1000', '10000', '100k',
       '100reasonstovisitmombasa', '100time', '101', '1010', '1017',
       '1017nell', '1031genfmsbi', '1033', '103k', '1057darwin', '1059',
       '1093', '10am', '10x12', '11', '110', '1100', '1100d'],
      dtype=object)
```

1.10) Train a supervised learning model of choice on the development dataset

In [11]:
```
#code here
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(dev_text_bow, dev_y)
```

Out[11]:
```
LogisticRegression()
```

1.11) Create TF-IDF features for each tweet in the development dataset

In [15]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
#code here
tfidf_vector = TfidfVectorizer()
dev_text_tfidf = tfidf_vector.fit_transform(dev_text['review'])
test_text_tfidf = tfidf_vector.transform(test_text['review'])
```

1.12) Train the same supervised learning algorithm on the development dataset with TF-IDF features

In [16]:
```
#code here
log_reg_tfidf = LogisticRegression()
log_reg_tfidf.fit(dev_text_tfidf, dev_y)
```

Out[16]:
```
LogisticRegression()
```

1.13) Compare the performance of the two models on the test dataset

In [19]:
```
#code here
print('----Logistic Regression Test Mean Accuracy----')
print('Bag of Words: ' + str(log_reg.score(test_text_bow, test_y)))
print('TFIDF: ' + str(log_reg_tfidf.score(test_text_tfidf, test_y)))
```

```
----Logistic Regression Test Mean Accuracy----
Bag of Words: 0.7485
TFIDF: 0.753
```

In [ ]:
```

```