Solution - Preprocessing

November 23, 2021

Table of Contents

Instructions

Get data from kaggle.com

Load a dataframe

Basic pre-processing

0.1 Instructions

1. Load this data set from kaggle - kaggle datasets download -d gpreda/pfizer-vaccine-tweets
2. Determine the shape of the dataframe
3. Review the data types
4. Drop the id column
5. Check for null values
6. Perform the following pre-processing on the 'text' column.
    - (new column1) change all text to lowercase
    - (new column2) use new column1 and remove contractions.

    - (new column3) use new column2 and string the data back together
    - (new column4) use new column3 and tokenize into sentences
    - (new column5) use new column3, again, and tokenize into words

    - (new column6) use new column5 and special characters
    - (new column7) use new column6 and remove stop words
    - (new column8) use new column7 and perform stemming
    - (new column9) use new column8 and perform lemmanization
    - add columns tweet length and tweet word count

0.1.1 Get data from kaggle.com

[ ]:
```
#from google.colab import drive
#drive.mount('/content/drive')
```

[ ]:
```
#from google.colab import files

## Upload your kaggle json file (API Token)
#files.upload()
```

```
#!mkdir ~/.kaggle

#!cp kaggle.json ~/.kaggle/

#!chmod 600 ~/.kaggle/kaggle.json
```

[ ]:
```
#!kaggle datasets download -d gpreda/pfizer-vaccine-tweets
```

[ ]:
```
#!ls
```

[ ]:
```
#!mkdir data

#!unzip zip file name -d data
```

[ ]:
```
#!ls -l data/
```

0.1.2 Load a dataframe

[ ]:
```
# Imports
import pandas as pd

# What other imports are required?
#!pip install contractions
#!pip install pyspellchecker

import contractions
import string
import re

import nltk
#nltk.download('punkt')
from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize

#nltk.download('stopwords')
from nltk.corpus import stopwords
stop_words = set(stopwords.words('english'))

#nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
```

[ ]:
```
pfz = pd.read_csv('/Users/jimcody/Documents/2021Python/nlp/data/
 ↪vaccination_tweets.csv')
#pfz.head()
```

[ ]:
```
#pfz.shape
```

[ ]:
```
#pfz.info()
```

0.1.3 Basic pre-processing

[ ]:
```
# Drop columns
drop_columns = {'id'}
pfz = pfz.drop(columns = drop_columns)
#pfz.shape
```

[ ]:
```
# Change text to lowercase
pfz['lower'] = pfz['text'].str.lower()
#pfz.head()
```

[ ]:
```
# Remove contractions
pfz['remove_ctr'] = pfz['lower'].apply(lambda x: [contractions.fix(word) for␣
 ↪word in x.split()])
#pfz.head()
```

[ ]:
```
# Change no_contract back to a string
pfz["review_new"] = [' '.join(map(str, l)) for l in pfz['remove_ctr']]
#pfz.head()
```

[ ]:
```
# Create tokenized sentences
pfz['tokenized_sent'] = pfz['review_new'].apply(sent_tokenize)
#pfz.head()
```

[ ]:
```
# Create tokenized words
pfz['tokenized_word'] = pfz['review_new'].apply(word_tokenize)
#pfz.head()
```

[ ]:
```
print(string.punctuation)
```

[ ]:
```
# Remove special characters
punc = string.punctuation
pfz['no_punc'] = pfz['tokenized_word'].apply(lambda x: [word for word in x if␣
 ↪word not in punc])
pfz.head()
```

[ ]:
```
pfz['no_stopwords'] = pfz['no_punc'].apply(lambda x: [word for word in x if␣
 ↪word not in stop_words])
pfz.head()
```

[ ]:
```
pfz['pos_tags'] = pfz['no_stopwords'].apply(nltk.tag.pos_tag)
pfz.head()
```

[ ]:
```
def get_wordnet_pos(tag):
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('N'):
        return wordnet.NOUN
    elif tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN
```

[ ]:
```
pfz['wordnet_pos'] = pfz['pos_tags'].apply(lambda x: [(word,␣
 ↪get_wordnet_pos(pos_tag)) for (word, pos_tag) in x])
pfz.head()
```

[ ]:
```
wnl = WordNetLemmatizer()
pfz['lemmatized'] = pfz['wordnet_pos'].apply(lambda x: [wnl.lemmatize(word,␣
 ↪tag) for word, tag in x])
pfz.head()
```

[ ]:
```
pfz['review_len'] = pfz['text'].astype(str).apply(len)
pfz['word_count'] = pfz['text'].apply(lambda x: len(str(x).split()))
```

[ ]:
```
pfz.head()
```