AdvancedTrack_RACOONS_report

April 10, 2022

1 Data Hacks 2022 - Alex, Prabina, & Atharva

1.1 Final Report - Advanced Level

Read in Files, Import

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from wordcloud  import WordCloud
```

[2]:
```
data = pd.read_csv('data/trainset.csv')
data.head()
```

[2]:

	Sentence	Sentiment
0	According to the Finnish-Russian Chamber of Co…	neutral
1	The Swedish buyout firm has sold its remaining…	neutral
2	$SPY wouldn't be surprised to see a green close	positive
3	Shell's $70 Billion BG Deal Meets Shareholder …	negative
4	SSH COMMUNICATIONS SECURITY CORP STOCK EXCHANG…	negative

1.1.1 Intro

Our project is going to include the following: 1. Data Cleaning 2. Exploratory Analysis, where we explore the data with visualizations 3. Feature Engineering 4. Results + Analysis

1.1.2 Data Cleaning

[3]:
```
import nltk
from nltk.stem import WordNetLemmatizer
from nltk.corpus  import stopwords
import re
```

[4]:
```
def cleaning(df):
    lowered=df.lower() # lowering the sentences
    removed = re.sub(r'[^a-z]',' ',lowered)  # replacing the non alphabets with␣
 ↪space
    splitted=removed.split(' ')   # splitting the sentences by spaces to␣
 ↪lemmatize
```

```
    df = [WordNetLemmatizer().lemmatize(word) for word in splitted
        if word not in stopwords.words('english')]  # lemmatizing and removing␣
↪stopwords
    df =' '.join(df) # joining back the words of list
    return(removed)
```

[5]:
```
data['Sentence'] = data['Sentence'].apply(cleaning)
data.head()
```

[5]:

	Sentence	Sentiment
0	according to the finnish russian chamber of co…	neutral
1	the swedish buyout firm has sold its remaining…	neutral
2	spy wouldn t be surprised to see a green close	positive
3	shell s billion bg deal meets shareholder …	negative
4	ssh communications security corp stock exchang…	negative

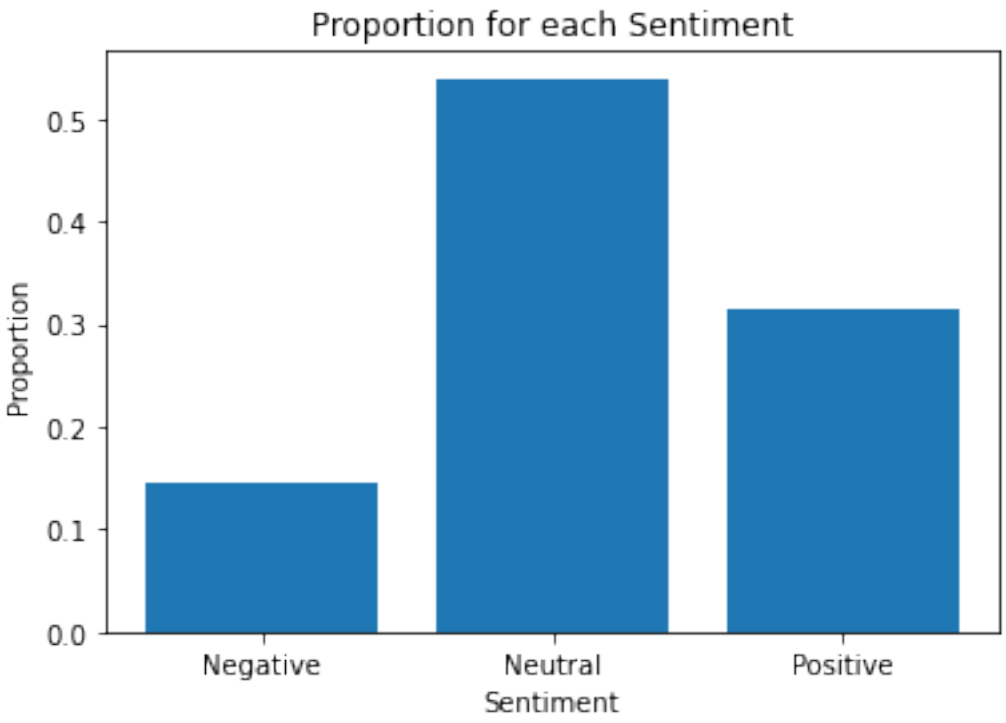
1.1.3 Visualizations

See what proportion of responses are negative, positive, and neutral.

[6]:
```
proportion_neg = data.loc[data['Sentiment'] == 'negative'].shape[0] / data.
↪shape[0]
proportion_neutral = data.loc[data['Sentiment'] == 'neutral'].shape[0] / data.
↪shape[0]
proportion_positive = data.loc[data['Sentiment'] == 'positive'].shape[0] / data.
↪shape[0]
props = [proportion_neg, proportion_neutral, proportion_positive]

plt.bar(['Negative', 'Neutral', 'Positive'], props)
plt.xlabel('Sentiment')
plt.ylabel('Proportion')
plt.title('Proportion for each Sentiment')
```

[6]:
```
Text(0.5, 1.0, 'Proportion for each Sentiment')
```



Clearly there is an imbalance towards 'Neutral' Sentinement, with it consisting of more than 50% of the total training data set.

This is potentially due to the fact that if a model is not sure about how to rate a sentence, it rates it neutral.

1.1.4 Query data by whether sentence is alpha-numeric, PRABINA TODO

1.1.5 Query data by length of sentence

We wondered if the length of the sentence has anything to do with how a sentence is rated as negative, neutral, or positive.

[7]:
```
def sentence_len(col):
    return len(col)
data['len_of_sentence'] = data['Sentence'].apply(sentence_len)
data
```

[7]:

	Sentence	Sentiment	\
0	according to the finnish russian chamber of co…	neutral	
1	the swedish buyout firm has sold its remaining…	neutral	
2	spy wouldn t be surprised to see a green close	positive	
3	shell s billion bg deal meets shareholder …	negative	
4	ssh communications security corp stock exchang…	negative	
…	…	…	

```
4377  investments in product development stood at   …   neutral
4378  hsbc says unit to book      million charge on …  negative
4379  rising costs have forced packaging producer hu…  negative
4380  in the building and home improvement trade   s…   neutral
4381  helsinki afx   kci konecranes said it has won …  positive

      len_of_sentence
0                 128
1                 135
2                  47
3                  56
4                 190
…                 …
4377               72
4378               56
4379              107
4380               88
4381              145

[4382 rows x 3 columns]
```

[8]:
```
data['len_of_sentence'].median()
```

[8]:
```
107.0
```

The *median* length of the 'Sentence' column is 107. Mean was not used as it is susceptible to outliers.

[9]:
```
bottom_50 = data.loc[data['len_of_sentence'] <= 107]
# bottom_50
```

Find how many sentences are placed under negative, neutral, and positive for sentences that have length lower than the median (107).

[10]:
```
bottom_sentiments = [bottom_50.loc[bottom_50['Sentiment']=='negative'].shape[0]
 ↪/ bottom_50.shape[0],
bottom_50.loc[bottom_50['Sentiment']=='neutral'].shape[0] / bottom_50.shape[0],
bottom_50.loc[bottom_50['Sentiment']=='positive'].shape[0] / bottom_50.shape[0]]
bottom_sentiments
```

[10]:
```
[0.16914749661705006, 0.487595850248083, 0.3432566531348669]
```

[11]:
```
top_50 = data.loc[data['len_of_sentence'] > 107]
# top_50
```

Find how many sentences are placed under negative, neutral, and positive for sentences that have length greater than the median (107).

[12]:
```
top_sentiments = [top_50.loc[top_50['Sentiment']=='negative'].shape[0] / top_50.
↪shape[0],
top_50.loc[top_50['Sentiment']=='neutral'].shape[0] / top_50.shape[0],
top_50.loc[top_50['Sentiment']=='positive'].shape[0] / top_50.shape[0]]
top_sentiments
```

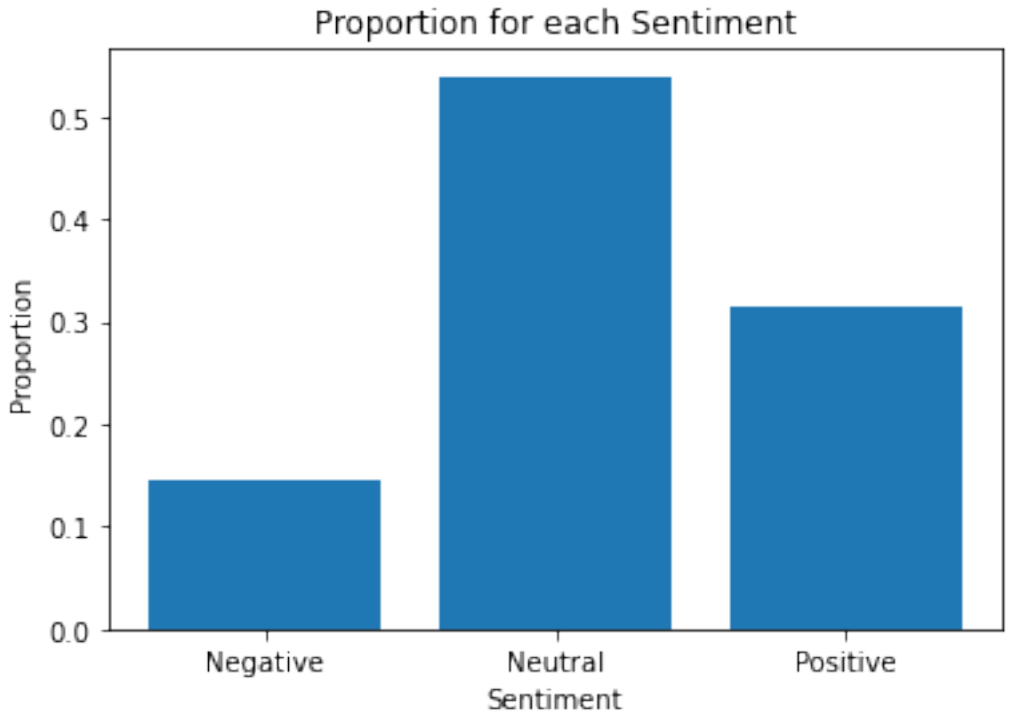
[12]:
```
[0.12055427251732101, 0.5921478060046189, 0.28729792147806005]
```

Now let's plot both the bottom 50th percentile (from df bottom_50) and the upper 50th percentile (from df top_50) and see if there are any clear deviations from our original graph.

Here is the original graph again, for reference.

[13]:
```
plt.bar(['Negative', 'Neutral', 'Positive'], props)
plt.xlabel('Sentiment')
plt.ylabel('Proportion')
plt.title('Proportion for each Sentiment')
```

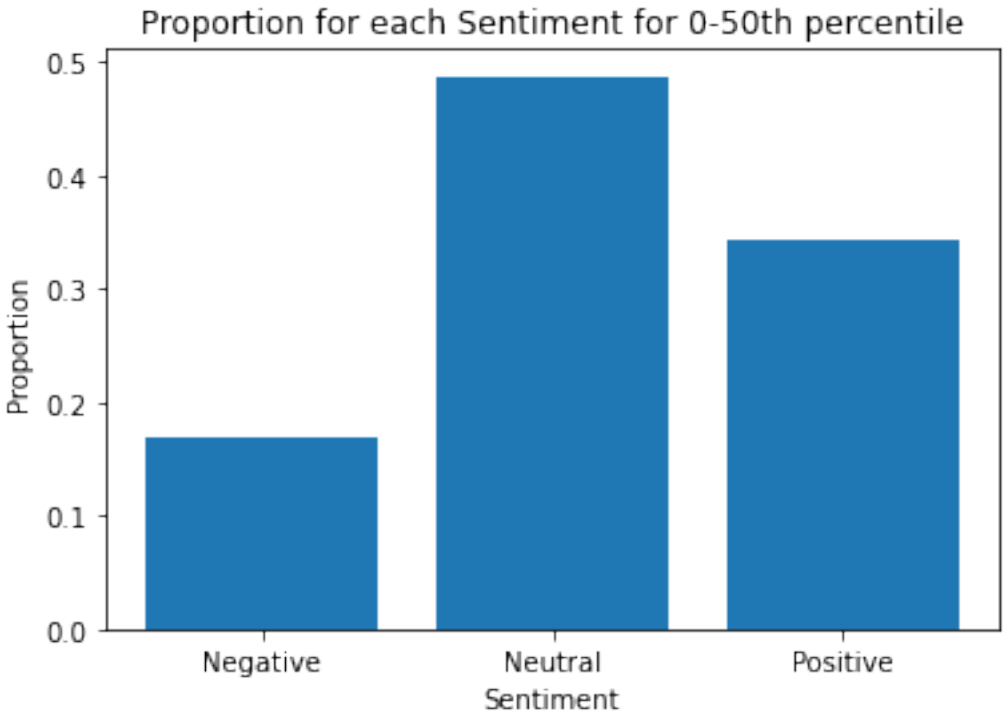
[13]:
```
Text(0.5, 1.0, 'Proportion for each Sentiment')
```



1.1.6 Here is the graph for the bottom 50th percentile

[14]:
```
plt.bar(['Negative', 'Neutral', 'Positive'], bottom_sentiments)
plt.xlabel('Sentiment')
plt.ylabel('Proportion')
plt.title('Proportion for each Sentiment for 0-50th percentile')
```

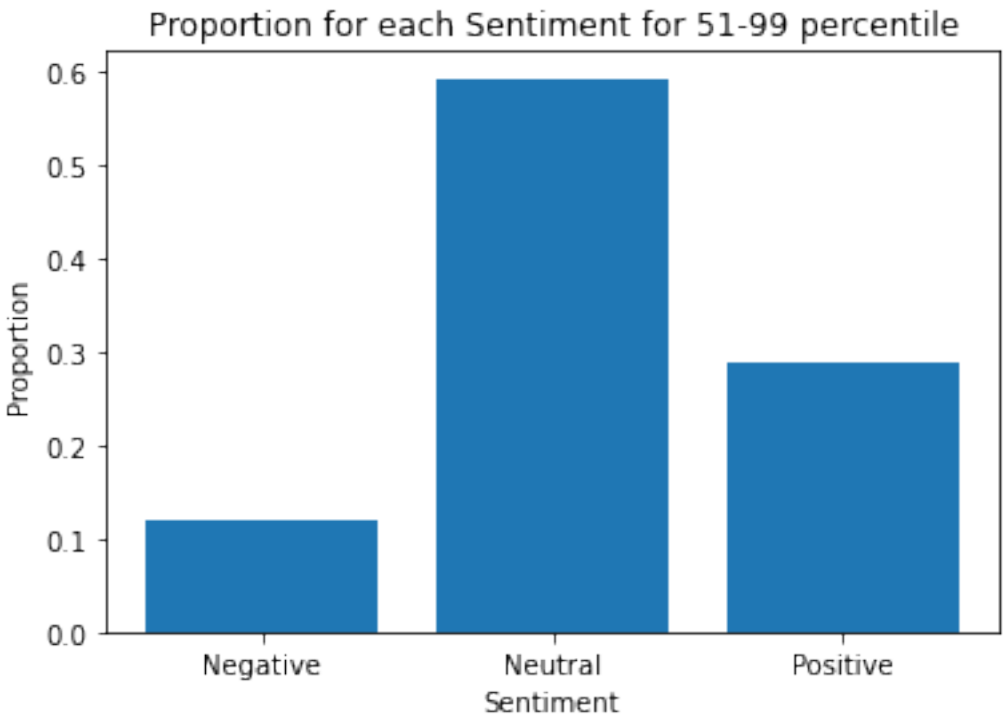
[14]: Text(0.5, 1.0, 'Proportion for each Sentiment for 0-50th percentile')



1.1.7 Here is the graph for the top 50th percentile

[15]:
```
plt.bar(['Negative', 'Neutral', 'Positive'], top_sentiments)
plt.xlabel('Sentiment')
plt.ylabel('Proportion')
plt.title('Proportion for each Sentiment for 51-99 percentile')
```

[15]: Text(0.5, 1.0, 'Proportion for each Sentiment for 51-99 percentile')



It seems that there is no correlation between shorter/longer sentences, and those sentences' ratings.

1.1.8 Word Cloud

[16]:
```
data.head()
```

[16]:

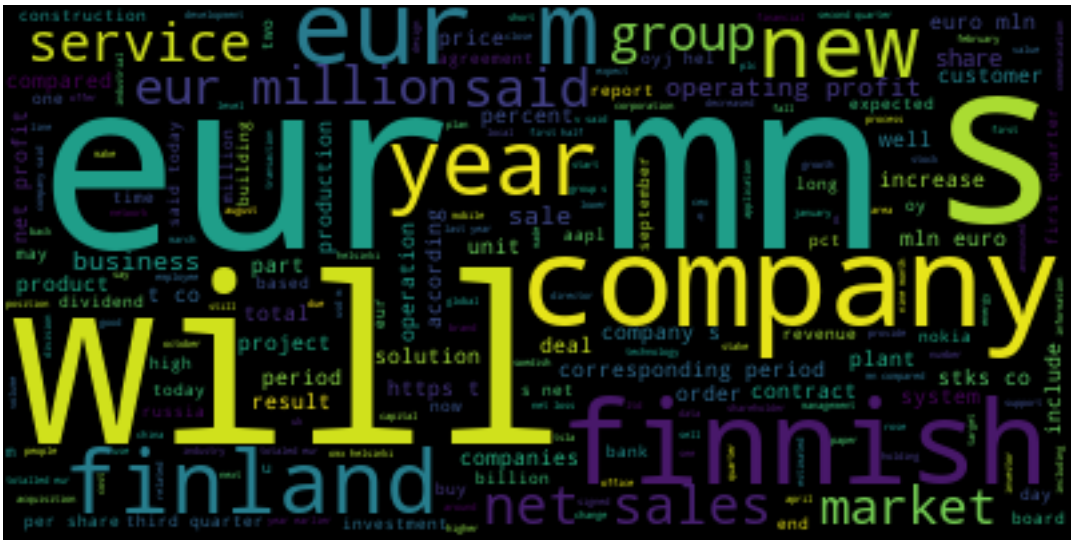
	Sentence	Sentiment	\
0	according to the finnish russian chamber of co…	neutral	
1	the swedish buyout firm has sold its remaining…	neutral	
2	spy wouldn t be surprised to see a green close	positive	
3	shell s billion bg deal meets shareholder …	negative	
4	ssh communications security corp stock exchang…	negative	

	len_of_sentence
0	128
1	135
2	47
3	56
4	190

[17]:
```
corpus=' '.join(data['Sentence'])
word_cloud = WordCloud().generate(corpus)
```

```
image = word_cloud.to_image()
image
```

[17]:



This word cloud shows the most frequent words from our train dataset. It seems like eur, mn, and year are some of the most common words. Considering our dataset included sentences from financial market, eur for europian, mn for million, and year is reasonable.

1.1.9 Analysis/ Results

We cleaned the data by turning every word to lower case, removing non-alphabetical words, and removing stop words, and lemmatizing by the words. We then made a Document Term Matrix by using tfidf vectorizer to vectorize each words. It converted text into numerical representation. We then created a testing dataset using 10% of our training dataset shuffled. We then performed label encoding by converting labels into numeric form so that it's machine-readable. We then used PCA for dimensionlaity reduction because our number of columns were higher than number of rows. After trying it on the test data, we predicted in the given test data. Our accuracy turned out to be 0.7266514806378133 and the F1 score to be 0.7489698056078457.

1.1.10 Conclusion/ future improvements

In our training dataset, the proportion of sentiments were imbalanced. In the future, we will pick a training dataset that has around the same number of neutrals, positives, and sentences.

We ran into another issue when predicting the sentiment on test data. We had initially used SVC (Support Vector Machines) to train our model but we had to switch to Logistic Regression because we ran into issues.