Description of data and problem you want to solve

The dataset that I will be using for this final project is the *"stock_data.csv"* dataset. This dataset is in a csv format and it consists of 5791 unique tweets reacting to the stock news. The question that I am interested in answering at the end of this analysis is "How do people react to various news regarding the economic change?" This is a set of labeled data that I retrieved from Kaggle.com: the first column shows the text from the tweets, whereas the second column shows the sentiment of the tweets (-1 being negative and 1 being positive).

Explanation of methods or tools used

Data Cleansing:

I started with some basic data cleansing like checking null values, tokenization, removing stop words and lemmatization.

Tokenization is done with the *RegexpTokenizer* function from the *nltk.tokenize* package. The tweets are tokenized based on whitespace only, by specifying the RegEx: *'\s+', gaps = True*. I figured removing the punctuations will not be a good idea in this dataset particularly, because some of the tweets contains the value of the stocks, which are in decimal or percentage forms. Therefore, removing punctuations such as "." or "%" might alter the nature of the tweets.

I used the *stopwords* package from *nltk.corpus* to remove stopwords, such as "be", "on", "my", and other words that has no impact on the sentiment analysis later on.

For the last phase of data cleansing, the *WordNetLemmatizer* from *nltk.stem* is used to return the words into their root forms. Finally, the tokenized words are joined again with whitespace in between.

Sentiment Analysis:

The most important function that is used in this phase of the analysis is the *sentiment* command from *TextBlob* package. The polarity of the tweets (ranging from -1 to 1: -1 being most negative

and 1 being most positive) is saved as *tweets_sentiment* and will be used in the next two phases: data visualization and results comparison.

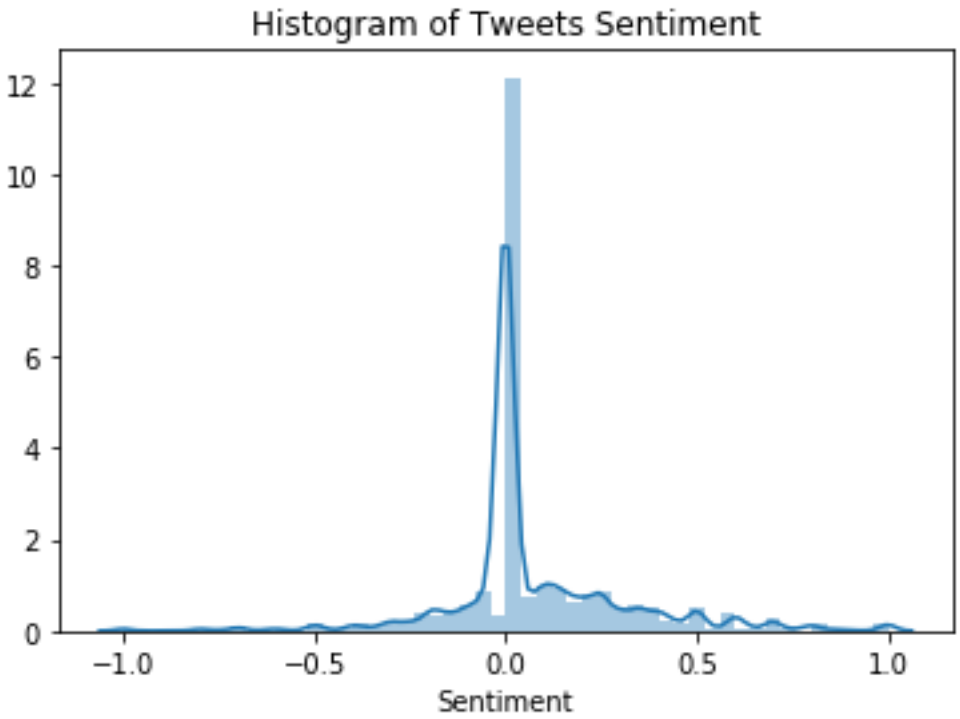
Data Visualization:

Two of the main packages in the Python data visualization library are used here: *matplotlib.pyplot* and *seaborn.* The *distplot* function from *seaborn* is used to create a histogram to visualize the frequency of occurrences of the sentiment polarity, whereas the *pie* function from *matplotlib.pyplot* is used to visualize the frequency of positive, negative, and neutral sentiments in the form of a pie chart.

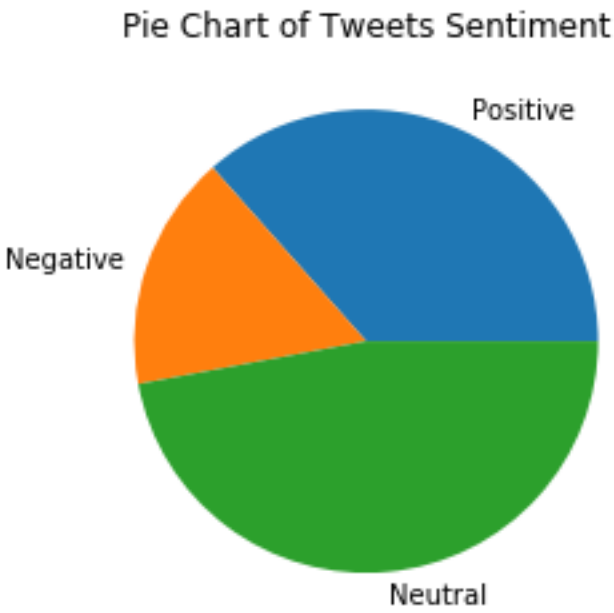
Results comparison

This dataset is a labeled data, which means the original sentiment of each tweet is already given. In my program, these "original sentiment" is saved as *original_sentiment.* Therefore, in this phase of my sentiment analysis, I compared the *original_sentiment* values to the sentiment values that I have found (which is saved as *tweets_sentiment*) to see how efficient the *sentiment* function is in analyzing the sentiment of tweets regarding the stock market change. I used the *mean_squared_error* function from the *sklean.metrics* package to do this.

Summary



Based on this histogram, the sentiment with around 0.0 polarity (neutral) has the most frequency. This can be due to a couple of reasons. One of the explanations is that most of the tweets are neutral: maybe the tweet was just informing about the price of the stock of a certain company or asking a simple question without any implication of sentiments. Another explanation that makes more sense is that the computer could not understand the human language well enough. For example, one of the tweets says the following: “Coronavirus scare: Global markets suffer record meltdown as global virus alarm grows”, which to human understanding, indicates a rather negative sentiment. However, the computer deemed it as a neutral sentiment, adding to the frequency of sentiment with 0.0 polarity hence.



This pie chart implies that most of the tweets are neutral, which is already explained above. We could also see that positive tweets have a higher frequency than negative tweets.

```
In [178]: mean_squared_error(original_sentiment, tweets_sentiment)
Out[178]: 0.9483855399480924
```

The *mean_squared_error* function produced an error rate of 0.9484, after comparing the sentiment values generated by the *textblob* package and the original sentiment values. This shows that the computer did a poor job in understanding and analyzing the sentiments of tweets regarding the stock market change.

Conclusion

In conclusion, disregarding the mean squared error, we see that neutral tweets have the highest frequency, whereas negative tweets have the lowest frequency. Sentiment analysis has always been a challenge for computer programs, it is a tremendously difficult task even for human beings. That said, the predictions might not be completely precise and accurate. However, it is still worth the effort to see how computers analyze human texts and hopefully in the future, more improvements will be made to yield better predictions and greater benefits.

Reference

Chaudhary, Y. (2020, June 5). *Stock-Market Sentiment Dataset.* Kaggle.com. Retrieved from: https://www.kaggle.com/yash612/stockmarket-sentiment-dataset

Koenig, R. (2019, July 5). *The Basics of EDA (with candy).* Towards Data Science. Retrieved from: https://towardsdatascience.com/the-basics-of-eda-with-candy-83b2e8ad9e63

Koenig, R. (2019, July 29). *NLP for Beginners: Cleaning & Preprocessing Text Data.* Towards Data Science. Retrieved from: https://towardsdatascience.com/nlp-for-beginners-cleaning-preprocessing-text-data-ae8e306bef0f