Final Report

This project sought out to answer a single question: Which single class represents the sentiment of the majority of tweets on Twitter? A solution was successfully discovered, and a model capable of predicting sentiments was created. Careful steps were constructed to get to this point.

The dataset used for this project was collected from Kaggle. The two csv files were fairly clean, which was the reason the dataset was picked initially. There was only one column with missing values (Tweet content), and these values were dropped. Duplicate values were also dropped. From here, exploratory data analysis was conducted.

The exploratory data analysis phase of the project was where the solution was revealed. A histogram was initialized to create a visual representation of the counts for each label (Figure 1). It is easy to see that the sentiment of the majority of tweets on Twitter from this large sample are represented by the negative class.

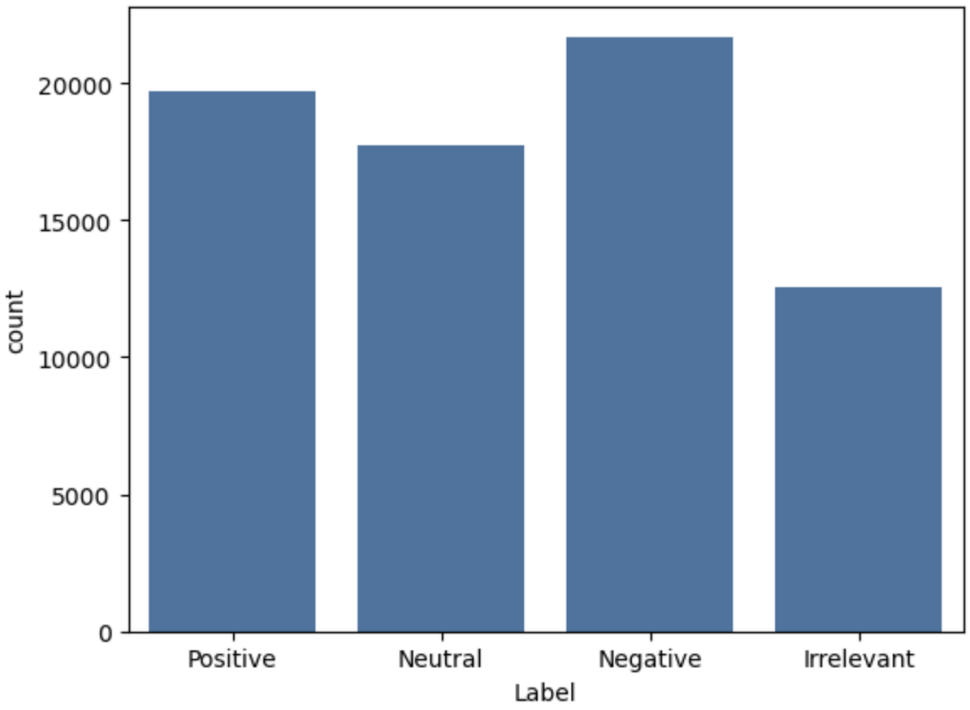

Figure 1

Final Report

The preprocessing step was fairly simple. A HTML remover was created using the BeautifulSoup library and applied to the Content column of the training and validation dataset. The same process was repeated to clean the data using regular expressions. This cleaning removed URLs, mentions and hashtags, and extra spaces.

Natural Language Processing (NLP) requires that text data be tokenized and stored with a corresponding frequency. The idea is that the most common words will give an idea to what the document is about. However, this approach is not always the most effective (ie. a document written by an artist might contain the word ‘painting’ a lot, but that may not be the main theme of that particular document). TF-IDF (Term Frequency-Inverse Document Frequency) tries to solve this issue by assigning weights to these tokens and penalizing those that occur too frequently in an effort to draw out more informative features.

The TfidfVectorizer from scikit-learn was fit and transformed on the X variable before performing the train/test split (since all these values contained text). Data was then split into a 80/20 train/test split. All these measures ensure efficiency in the modeling phase and prevent data leakage.

A multinomial Naive Bayes classifier was attempted initially. Since we have more than two types of outcomes, this seemed like a perfect choice. The classification report reaffirmed this idea (Figure 2). However, there was still room for improvement so other models were examined. Next, a logistic regression model was created. By default, the LogisticRegression algorithm from scikit-learn automatically detects the presence of more than two classes in the target feature, so results can be the main focus instead of messing with major adjustments to the model. Major improvements were seen (85% accuracy vs. the 72% seen by the Naive Bayes classifier).

However, the best performing model came from the Linear Support Vector Machine provided by scikit-learn. This is a popular model to use when doing NLP because text data tends to be linearly separable. While this model was the most computationally expensive out of the three, the performance was exceptional (Figure 3).

Final Report

Accuracy: 0.7229974881384315

	precision	recall	f1-score	support
Irrelevant	0.93	0.43	0.59	2477
Negative	0.66	0.90	0.76	4316
Neutral	0.82	0.63	0.71	3559
Positive	0.70	0.80	0.75	3980
accuracy			0.72	14332
macro avg	0.78	0.69	0.70	14332
weighted avg	0.76	0.72	0.71	14332

Figure 2

Accuracy: 0.8547306726207089

	precision	recall	f1-score	support
Irrelevant	0.88	0.80	0.84	2477
Negative	0.83	0.91	0.87	4316
Neutral	0.88	0.82	0.85	3559
Positive	0.85	0.86	0.85	3980
accuracy			0.85	14332
macro avg	0.86	0.85	0.85	14332
weighted avg	0.86	0.85	0.85	14332

Figure 3

To conclude, the Linear Support Vector Machine was the best performing model with almost 86% accuracy and a stellar classification report. In addition, we saw that the majority of tweets on Twitter from our sample corresponded to the negative class, potentially giving us a glimpse into the overall sentiment among users on the platform. The value these insights could bring are ever present. Maybe companies will scrap the web or use APIs to perform this analysis, and focus on specific instances relating to their company to use in unison with collected reviews. Twitter themselves could leverage this information when introducing paid subscription services like those they have offered in the past. In other words, proper analysis leads to a deeper understanding of a situation and ultimately better decision making.