Automatic Sentiment Analysis of Tweets about the COVID-19 Pandemic

Problem

For the past two years, the COVID-19 pandemic has wracked the globe. In these trying times, many people have turned to Twitter to express their fears, hopes, and frustrations regarding the pandemic. Their tweets provide a possible metric for assessing public attitudes toward the pandemic and the measures put in place to combat it. Given the large volume of tweets produced each day, however, it is not feasible to read and analyze each one by hand. The goal of this project therefore was to develop a method for automatically assigning sentiment scores to Tweets about COVID-19 pandemic.

Approach

Sentiment classification requires a pre-labelled dataset for training. For this project, I sourced data from the Corona Virus Tweets NLP - Text Classification dataset, a collection of 44995 tweets about the COVID-19 pandemic collected in early March 2020. This dataset contains the original text of each tweet, its date, the location of its sender (where known), and a manually assigned sentiment score ranging from extremely negative (0) to extremely positive (4). The two relevant data points for the current project are the original text and the sentiment score.

Before processing the data, I preformed exploratory data analysis to test the accuracy of the pre-assigned sentiment scores. To do so, I assigned my own labels to a random sample of 100 Tweets and calculated the mean difference in sentiment score between the pre-assigned labels and my own. This resulted in a mean difference of 1.08. A hypothesis test using 10000 bootstrap resamples showed that such outcome was highly unlikely ($p < .0001$) if my labels were incompatible (i.e., had a mean difference of 2 or more, which corresponds to a change from 'negative' to 'positive' or vice-versa) with the pre-assigned ones.

Preprocessing entailed several steps. First, I removed non-alphabetic characters, diacritics, hashtags, handles, links, and proper names from each Tweet since these elements do not contribute significantly to sentiment. Then, I lemmatized the remaining words and converted them into word vectors using the GloVe Twitter-25 embedding. Finally, I created a document vector for each Tweet by taking an TfIdf-weighted sum of its constituent word vectors.

In the modelling stage of the project, I used Bayesian optimization to tune three simple classifiers (k-nearest neighbors, naive bayes, and decision tree) and two ensemble ones (random forest and ada boosting) on the basis of mean difference in sentiment score (i.e., Mean Absolute Error). MAE made the most sense as a scoring metric because I was concerned with the overall performance of each classifier than its recall or precision. MAE also measures the severity of errors unlike other holistic metrics such as Matthew's Correlation Coefficient. A classifier that labeled all extremely positive Tweets as extremely negative, for example, would receive the same

MCC score as a classifier that simply downgraded all extremely positive Tweets to positive. I also plotted the confusion matrix and distribution of predicted labels for each classifier in order to determine where they struggled. These visualizations along with mean difference in sentiment score informed the final model selection.

Findings

The results of this process were as follows:

Classifier	**Mean Absolute Error**	**Comments**
K-Nearest Neighbors	.96	Occasionally confused 'Positive' and 'Negative' Labels; frequently demoted 'Extremely Positive' to 'Positive'
Naïve Bayes	1.1	Overpredicted 'Neutral' label at the expense of 'Positive' and 'Negative'
Decision Tree	1.08	Similar to KNN
Random Forest	.92	Similar to KNN
Ada Boosting	.96	Similar to KNN

Random Forest performed the best at classifying Tweets about the COVID-19 pandemic, achieving a Mean Absolute Error in sentiment score of .92. As the confusion matrix (fig 1) shows, it occasionally confused Positive and Negative Tweets and frequently demoted Extremely Positive Tweets to merely Positive. Nevertheless, it hews closely enough to the original distribution of sentiments to be useful (fig 2).

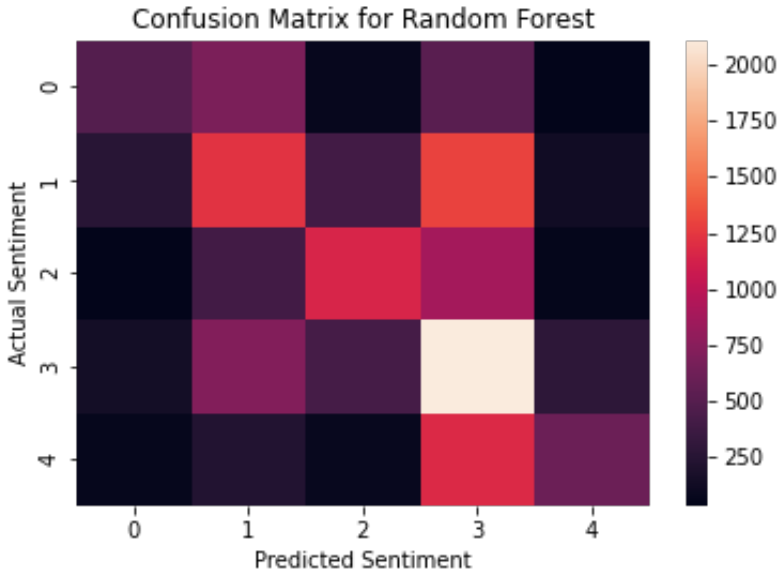

Fig 1: 0 = Extremely Negative, 1=Negative, 2=Neutral, 3= Positive, 4=Extremely Positive

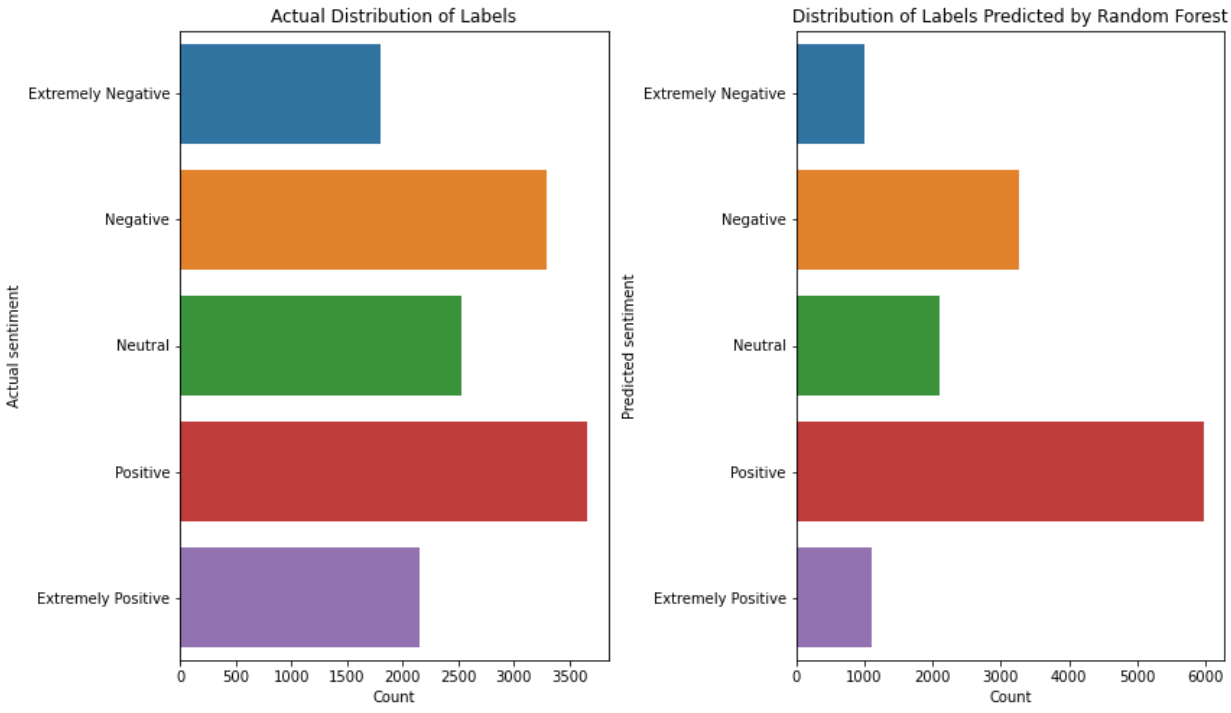

Fig 2: Comparison of the Actual Distribution of Sentiment Labels with Those Predicted by Random Forest

Recommendations

Based on my results, I would make the following recommendations regarding the automatic classification of Tweets about the COVID-19 pandemic:

1. Because the five sentiment classes form a sequence, use MAE to evaluate model performance.
2. Pay close attention to misclassifications. The types of error a model makes are as important as the number of errors it makes.
3. For best performance, employ a Random Forest Classifier with 782 estimators, each with a max depth of 25 and a maximum of 1 feature.

Ideas for Future Research

There are three avenues for further research on this topic, each corresponding to a different step in the Data Science pipeline.

The first avenue concerns training data. The Tweets I used for this project came with pre-assigned sentiment labels, but as I discovered during Exploratory Data Analysis, these labels were occasionally at odds with my own interpretations of the data. A potentially more robust approach would be to use unsupervised learning to group the Tweets into five clusters and then assign appropriate sentiment labels to each cluster based on its features. This would allow the five classes emerge from the data itself and would help us avoid the subjectivity inherent in manually assigned labels.

Second, it might be helpful to explore different methods for transforming Tweets into document vectors. While GloVe embeddings can capture the semantic relationships between individual words, they have a harder time capturing the syntactic relationship between words. This limitation proves especially problematic for sentiment analysis because sentiment often emerges from the interplay of different words in a sentence. Consider, for example, the difference between the phrases ‘bad’, ‘not bad’ (i.e., good), and ‘not too bad’ (i.e., OK). An algorithm that encodes syntactic and pragmatic relationships between words, such as BERT, can better model such semantic nuances.

Third, Deep Learning models have the potential to outperform Random Forest models on some tasks. Therefore, it would be worthwhile to test how various deep learning algorithms perform on the dataset.