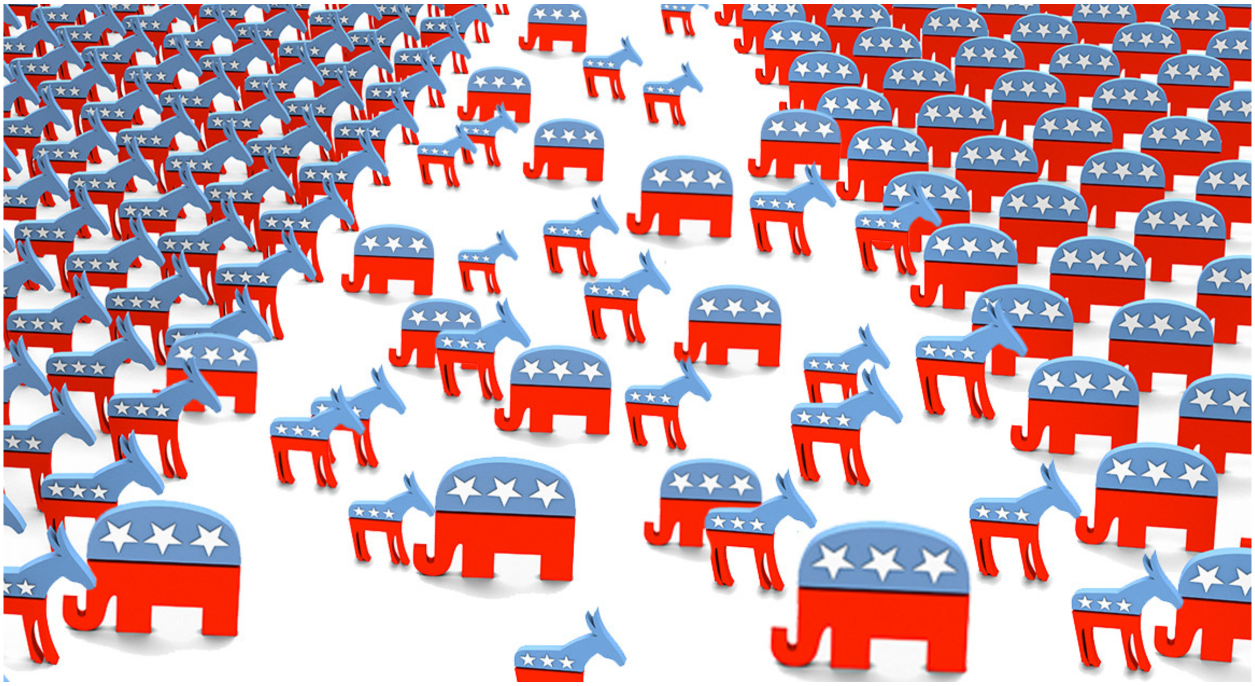
Springboard Data Science Career Track Capstone

Predicting Political Affiliations using Natural Language Processing



Steven Pereira

Introduction

Over the past few election cycles it seems that constituents have increasingly divergent views formed on party lines. Compromise has become a sign of weakness, with the adoption of contrarian positions becoming a more prevalent political strategy. The evolution of the American political sphere was the inspiration for this project. It isn't uncommon for people to assume political affiliation of their peers based on a handful of observed viewpoints. This project tests if Machine Learning can do the same

This project seeks to build a predictive model that can identify political affiliation using Natural Language Processing. The model will be trained on Youtube comments on videos regarding Covid-19 from the 3 major cable news networks: CNN, Fox, and MSNBC. The assumption of this project is that consumers of videos share the political affiliation of the network.

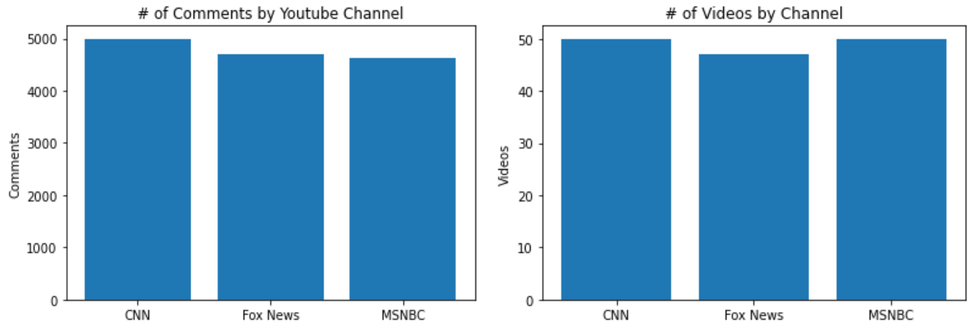
Audience

From the standpoint of the service provider, in this case Youtube, using comments to predict qualities of viewers would be valuable for their business model. Youtube would be able to refine channel and video recommendations in order to keep users engaged with the platform. Additionally, being able to predict qualities of users through comments can also improve ad targeting. Improving ad targeting and increasing user engagement would increase ad revenue. Being a free platform, advertising is a large part of Youtube's total revenue.

Data Wrangling

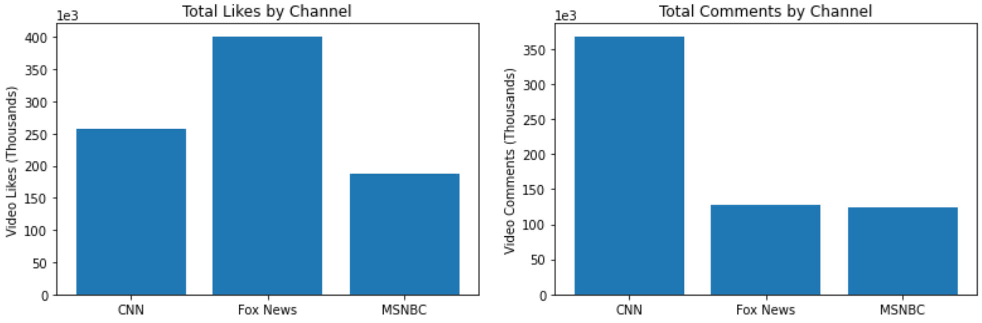
Using the Google API, comments were extracted from Covid-19 related Youtube videos from the 3 major cable news networks: CNN, Fox News, and MSNBC. The API limits requests to 50 videos per channel and 100 top rated comments per video. This results in a maximum of 15,000 comments from 150 videos from the 3 different news networks. Additionally, viewer engagement metrics pertaining to the associated video were gathered for each comment as well. This included total video views, likes, and comment count which includes comments not represented in this data set. The likes on each respective comment were included as well. The final dataset after removal of null values and outlier length comments was 14,329 comments from 147 videos.

Channel Representation



Video Engagement

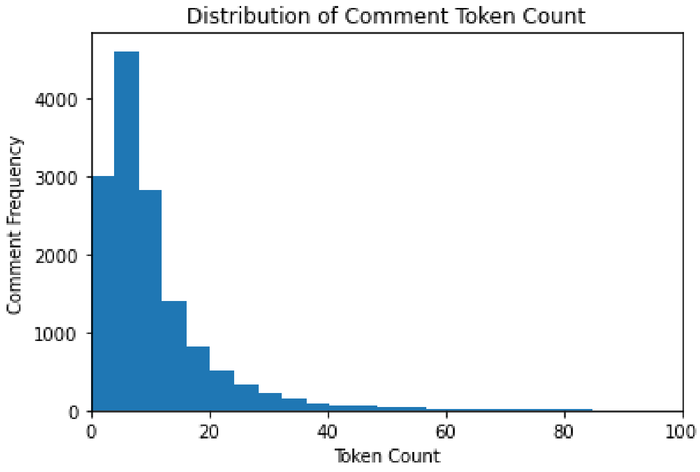
****Total Comments includes comments not represented in dataset****

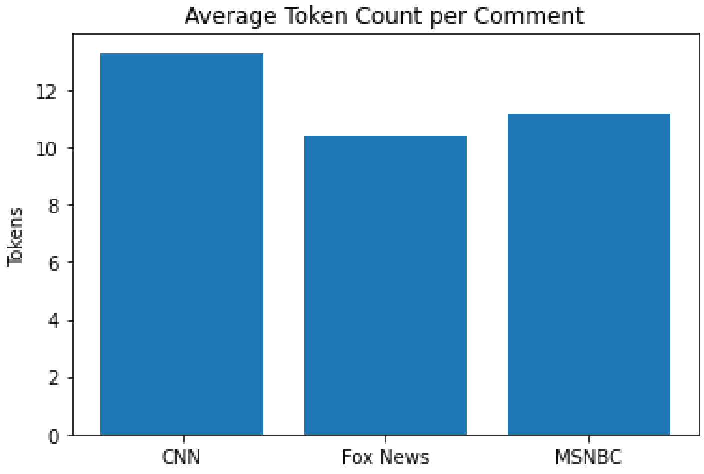


Exploratory Data Analysis

Similar to the preprocessing for modeling, the comments had to be processed for EDA. For this step, each comment was converted to lowercase, tokenized, stripped of non-alpha characters, and had English stopwords removed. EDA was conducted by looking at top word counts and highest TF-IDF weights. Word counts and TF-IDF weights were then stratified by News Network. Additionally Bi-Grams and Tri-grams were examined looking at counts and TF-IDF. Below are some interesting distributions. Not wanting to place potentially partisan assumptions while interpreting distributions, I invite the reader to examine the distributions and consider differences between networks.

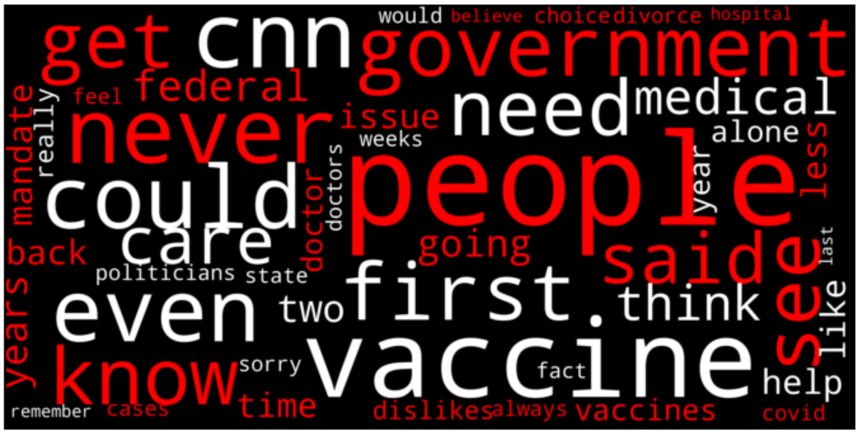
Token Count Distribution



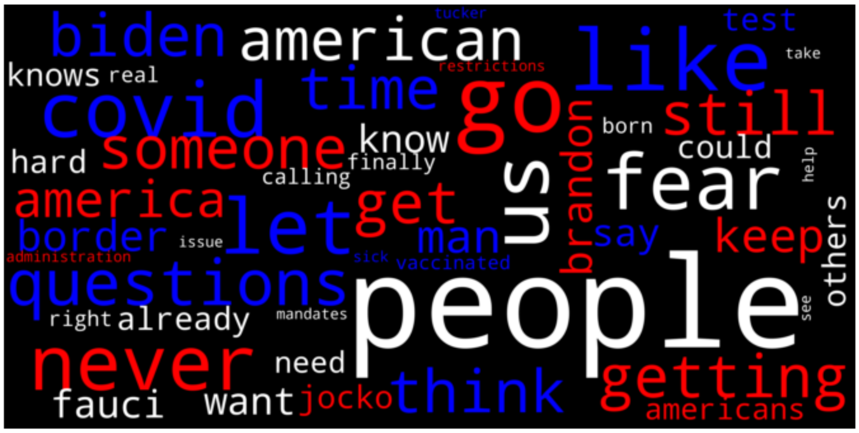


Token Counts

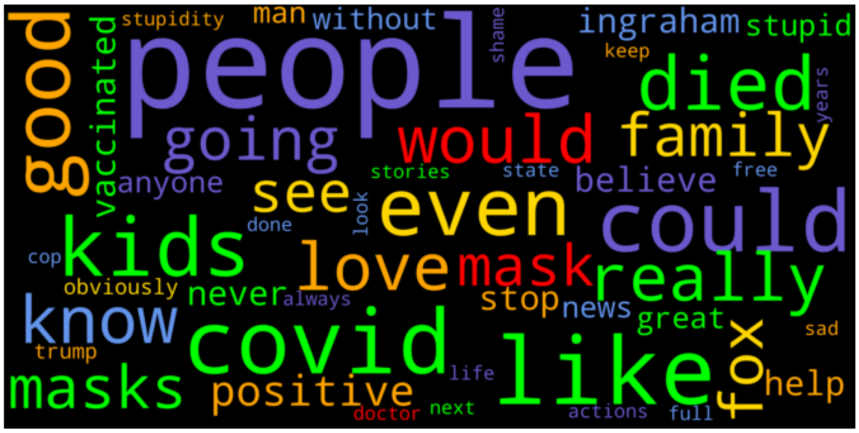
Most Frequent CNN Tokens (Most Liked Comments)



Most Frequent FOX Tokens (Most Liked Comments)

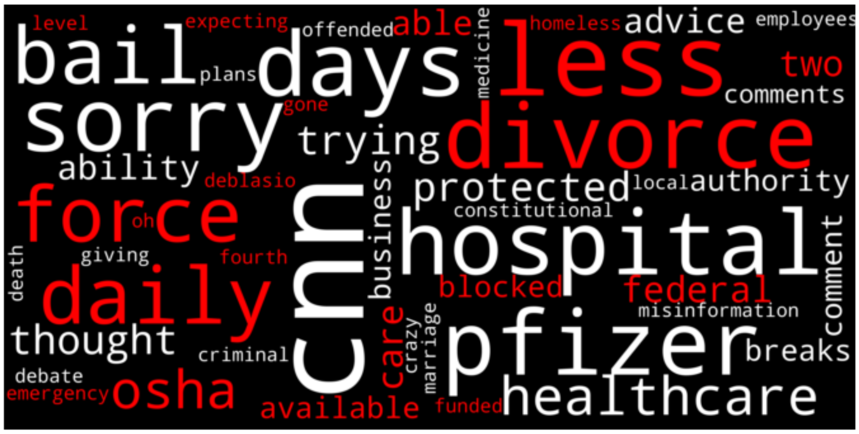


Most Frequent MSNBC Tokens (Most Liked Comments)

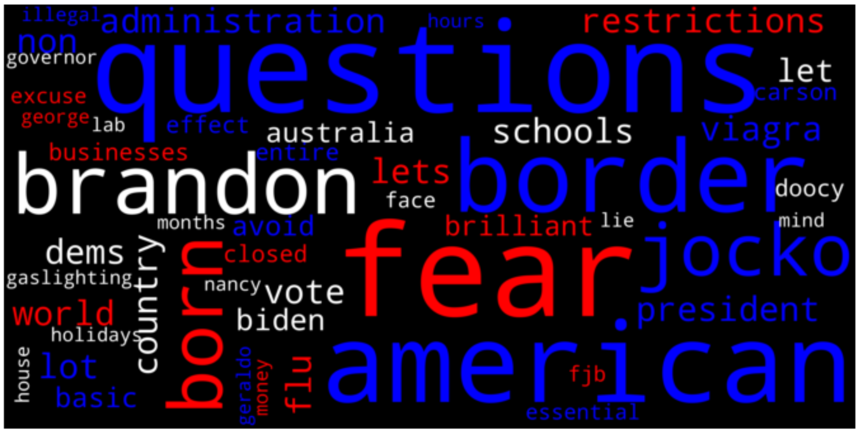


TF-IDF Weights

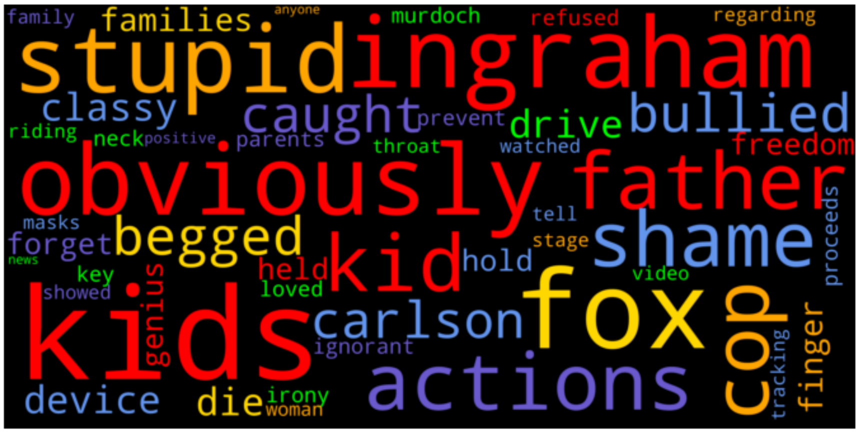
Top CNN Tokens by TFIDF (Most liked Comments)



Top FOX Tokens by TFIDF (Most liked Comments)



Top MSNBC Tokens by TFIDF (Most liked Comments)



Bi-gram Counts

CNN	FOX	MSNBC
'god bless', 56	'go brandon', 166	'let go', 67
'fully vaccinated', 54	'let go', 163	'go brandon', 59
'many people', 53	'god bless', 78	'get vaccinated', 39
'let go', 53	'gon na', 36	'gon na', 37
'go brandon', 48	'fox news', 29	'fully vaccinated', 32
'gon na', 40	'southern border', 27	'wear mask', 27
'get vaccinated', 39	'peter doocy', 26	'public health', 24
'got covid', 39	'thank god', 25	'health care', 23
'natural immunity', 35	'got covid', 22	'south africa', 23
'two years', 32	'side effects', 21	'fox news', 22
'wear mask', 31	'people need', 21	'got covid', 22
'mental health', 30	'natural immunity', 20	'natural immunity', 21
'thank god', 29	'federal solution', 20	'go back', 21
'health care', 28	'joe biden', 20	'looks like', 21
'even though', 27	'white house', 19	'new variant', 20
'healthcare workers', 27	'lets go', 19	'many people', 20
'long term', 26	'american people', 18	'two years', 19
'new york', 26	'ben carson', 18	'anthony fauci', 19
'know know', 25	'new york', 17	'good news', 19
'know lying', 24	'years ago', 17	'wearing mask', 18

Bi-grams TF-IDF

CNN	FOX	MSNBC
'lying know', 0.07067932899858004	'ben carson', 0.0887200120593771	'blah blah', 0.05706452815024745
'bari weiss', 0.04573368346966943	'dr oz', 0.07393334338281425	'cbd oil', 0.052309150804393495
'community schools', 0.04157607588151767	'thank tucker', 0.05914667470625141	'diet plan', 0.04755377345853954
'sorry loss', 0.04157607588151767	'dr carson', 0.05421778514739712	'brian williams', 0.042798396112685586
'know know', 0.03836123142780878	'hard evidence', 0.04928889558854284	'hahaha hahaha', 0.042798396112685586
'thank chris', 0.037418468293365904	'mike huckabee', 0.04928889558854284	'plan strictly', 0.042798396112685586
'watch cnn', 0.037418468293365904	'peter doocy', 0.047296768586740943	'say name', 0.042798396112685586
'chris cuomo', 0.03326086070521413	'jim jordan', 0.04436000602968855	'butter clarified', 0.03804301876683163
'chris thank', 0.03326086070521413	'monoclonal antibodies', 0.04436000602968855	'convince forward', 0.03804301876683163
'media training', 0.03326086070521413	'already knows', 0.03943111647083427	'get job', 0.03804301876683163
'prior infection', 0.03326086070521413	'evidence needed', 0.03943111647083427	'let convince', 0.03804301876683163

Preprocessing

Sample Weights

In EDA, looking at the most liked comments provided some interesting insight. In order to capture this effect, 2 separate weighting schemes were tested. The first weight is the ratio of observation comment likes to video views. This weight will normalize comment likes by the video traffic and captures the proportion of viewers which liked the comment. The second weight is a ratio of observation comment likes to total observation video comment likes. This weight captures the comments rank among other comments within their respective videos.

Sentiment Analysis

Each comment was analyzed using two sentiment analysis APIs: TextBlob and Vader for NLTK. The sentiment analysis provides a polarity score on a scale of Positive to Negative using lexicons. These 2 ratings were added as features to each comment as a means to provide context for the comment.

Comment Preprocessing

Each comment was processed for modeling by converting to lowercase, tokenizing, stripping of non- alpha characters, and lemmatization. Scikit vectorizors have built in tokenizers so tokens were joined back into a single string after processing.

Train Test Split

To avoid data leakages with sample weights, data was split on video id. Comments for each respective video were added back into the training and test sets.

Countvectorizer

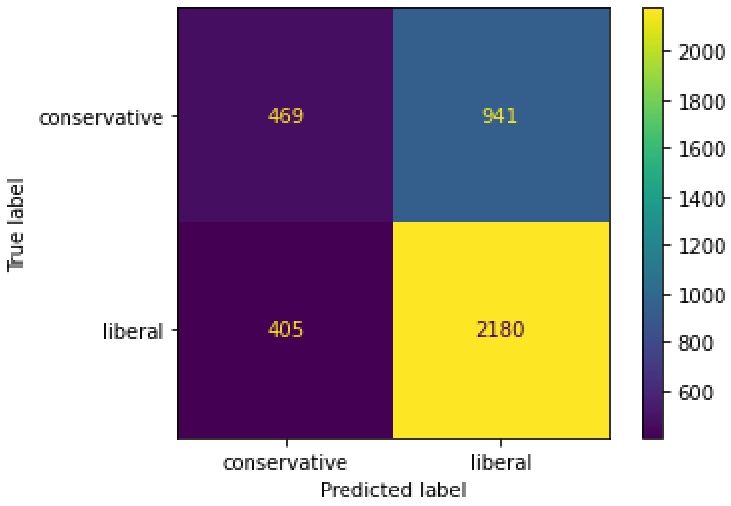
To avoid data leakage, countvectorizer was trained on the training set to create the feature vocabulary. This vocabulary was used to transform the training and test set.

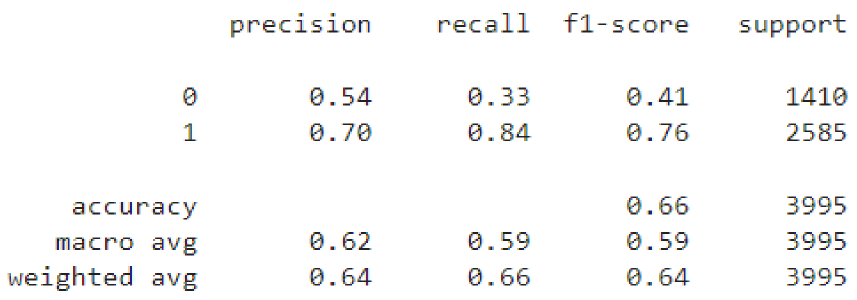
Modeling

A few classification models were tested for performance in predicting political affiliation. The training set consisted of 33% conservative video comments, and the test set consisted of 35% conservative video comments. Each model was trained on 7 sets with unique feature sets. An unweighted count vector, 2 weighted count vectors, and each of the weighted count vectors with sentiment analysis features added.

A Logistic Regression, Random Forest, Multinomial Naive Bayes, and Support Vector Machines were all trained on each of the 7 training sets. Each model used the default parameters. The best performing model and feature set combinations are highlighted below.

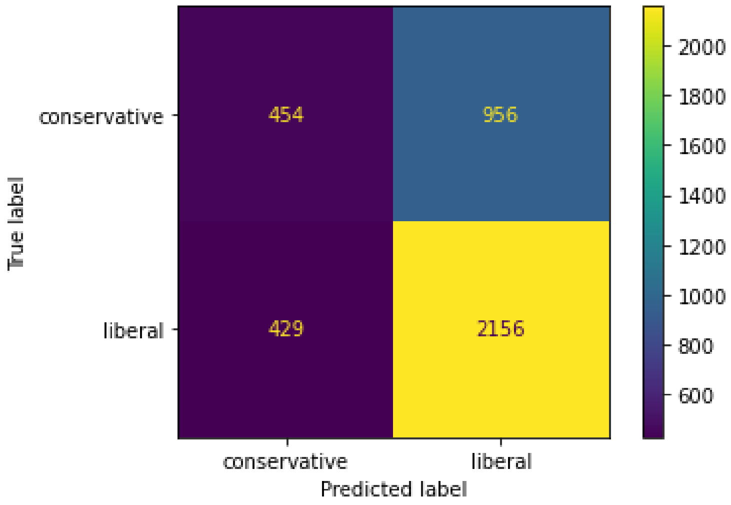
Logistic Regression + Count Vector

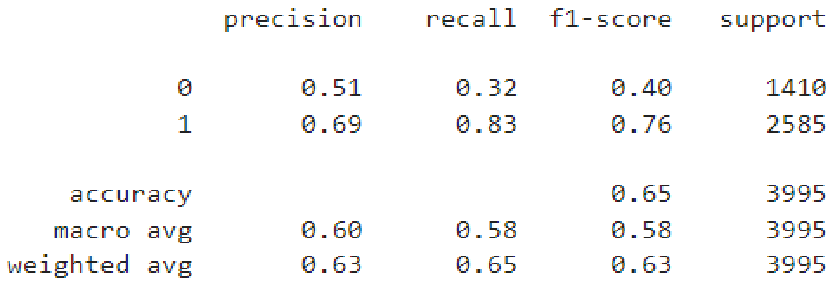




```
              precision    recall  f1-score   support

           0       0.54      0.33      0.41      1410
           1       0.70      0.84      0.76      2585

    accuracy                           0.66      3995
   macro avg       0.62      0.59      0.59      3995
weighted avg       0.64      0.66      0.64      3995
```

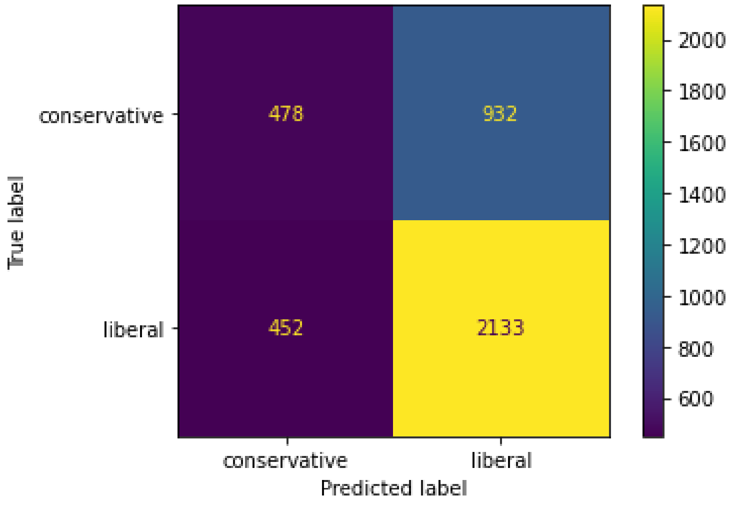
Random Forest + Weighted(1) Count Vector

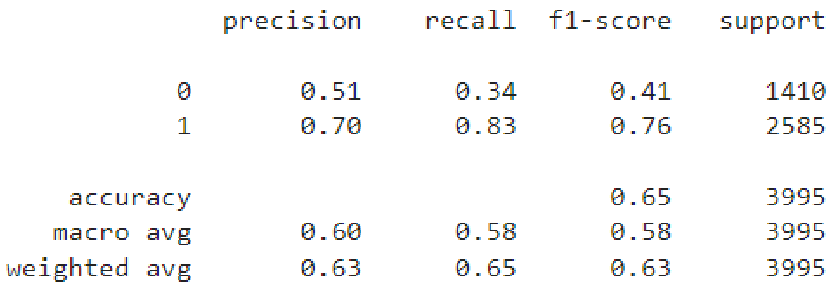




```
              precision    recall  f1-score   support

           0       0.51      0.32      0.40      1410
           1       0.69      0.83      0.76      2585

    accuracy                           0.65      3995
   macro avg       0.60      0.58      0.58      3995
weighted avg       0.63      0.65      0.63      3995
```

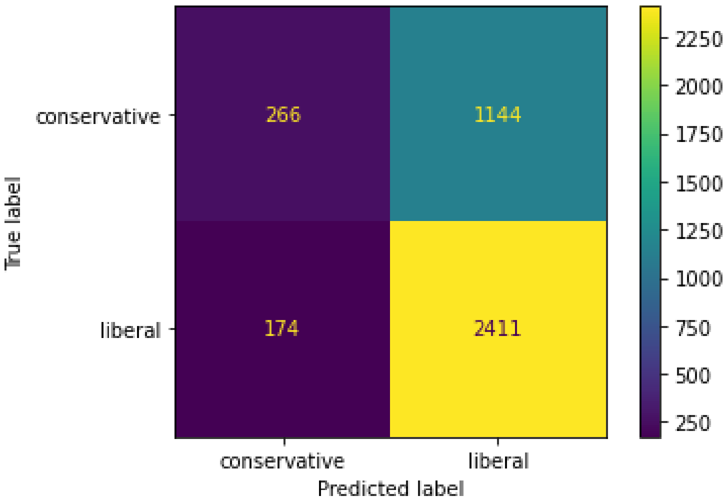
Multinomial Naive Bayes + Weighted(1) Count Vector with Sentiment Polarity

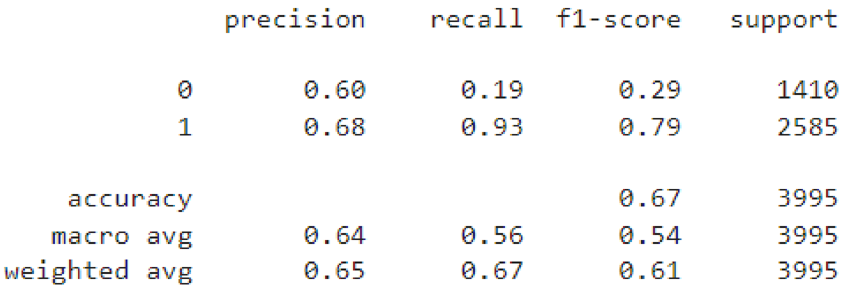




```
              precision    recall  f1-score   support

           0       0.51      0.34      0.41      1410
           1       0.70      0.83      0.76      2585

    accuracy                           0.65      3995
   macro avg       0.60      0.58      0.58      3995
weighted avg       0.63      0.65      0.63      3995
```

Support Vector Machines - Weighted(2) Count Vector

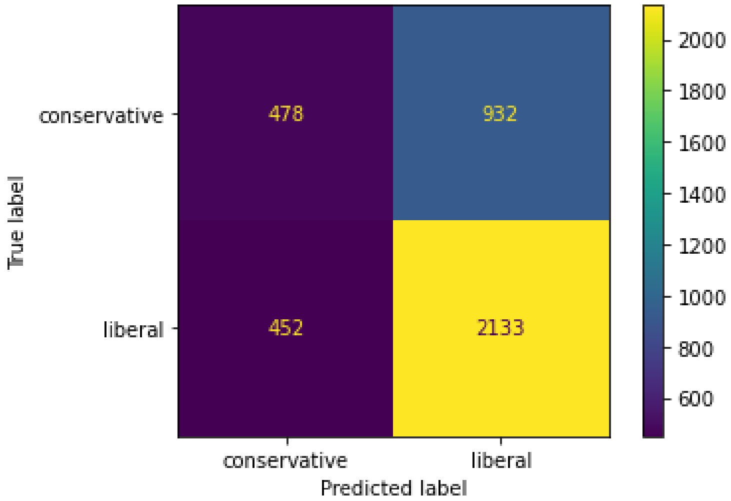


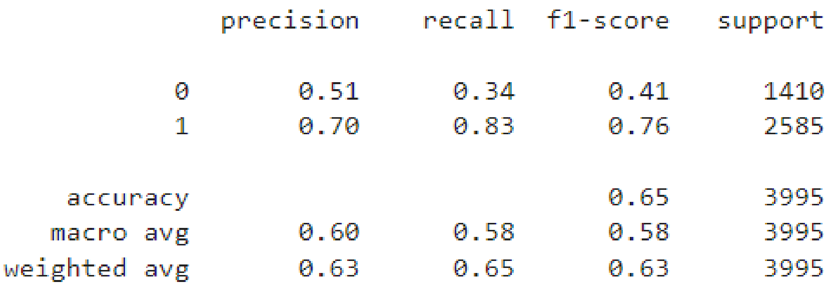


```
              precision    recall  f1-score   support

           0       0.60      0.19      0.29      1410
           1       0.68      0.93      0.79      2585

    accuracy                           0.67      3995
   macro avg       0.64      0.56      0.54      3995
weighted avg       0.65      0.67      0.61      3995
```

Model Selection and Conclusions

Multinomial Naive Bayes + Weighted(1) Count Vector with Sentiment Polarity





```
              precision    recall  f1-score   support

           0       0.51      0.34      0.41      1410
           1       0.70      0.83      0.76      2585

    accuracy                           0.65      3995
   macro avg       0.60      0.58      0.58      3995
weighted avg       0.63      0.65      0.63      3995
```

Out of the four models Multinomial Naive Bayes, Support Vector Machines, Logistic Regression, and Random Forest Classifier, Multinomial Naive Bayes was chosen to be the best performing model. The first sample weight, comment likes to video views, led to the best performing model. The TextBlob polarity rating as a feature also led to the best performing model.   All the models had similar performance, with accuracies between 65 - 67%. The chosen model, however, had the higher F1 scores despite having a lower accuracy. So overall this model was better at identifying both classes all around. For example SVM had one of the highest accuracies but had a low F1 score for the Conservative classification.