samlimish / **Project5**

0 stars 0 forks

Star

Unwatch

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

samlimish ... 2 minutes ago

View code

Capstone Project: Compare Consumer Sentiments of Apple, Google, and Android from Past to Present

Instructor: James Irving
Date: 7/23/21

Contents

Business Problem

For better or worse, people's perception of tech giants have changed over time. A company that consults these large companies' PR teams have hired me to find how the consumers' sentiments have changed. To gather the necessary information, I am going to go to Twitter, and I will compare the public's emotion towards these companies using vader sentiment analysis.

Data Collection

The old Twitter data were provided, but for the new Twitter data, I used tweepy and Twitter's developer API to collect 1,500 recent tweets for each of the three companies: Apple, Google, and Android.

```
# Apple
apple_search_words='#Apple OR Apple OR #iPhone OR iPhone \
-AppStore -pie -games -game -juice -Android -Google -"Big Apple" -gala -giveaway -vinegar -cider\
-filter:retweets -url:amazon'
date_since='2018-10-06'

apple_tweets = tw.Cursor(api.search,
                  tweet_mode='extended',
                  q=apple_search_words,
                  lang="en",
                  include_card_uri=False,
                  since=date_since).items(1500)
# tweets
```

```
tweet_list=[]
for tweet in apple_tweets:
    tweet_list.append(tweet._json)
# tweet_list[0]
```

```
# Google
google_search_words= '#Google OR Google \
-Apple -android\
-filter:retweets -url:amazon'
google_tweets = tw.Cursor(api.search,
                  tweet_mode='extended',
                  q=google_search_words,
                  lang="en",
                  include_card_uri=False,
                  since=date_since).items(1500)
```

```
for tweet in google_tweets:
    tweet_list.append(tweet._json)
```

```
# Android
android_search_words='#Android OR Android\
-Apple -Google -game -games -giveaway\
-filter:retweets -url:amazon'
android_tweets = tw.Cursor(api.search,
                  tweet_mode='extended',
                  q=android_search_words,
                  lang="en",
                  include_card_uri=False,
                  since=date_since).items(1500)
```

```
for tweet in android_tweets:
    tweet_list.append(tweet._json)
```

I also decided that it would be best to exclude tweets that contained more than one company names, and for Apple, since there were multiple related words/phrases, more filtering had to be done.

Data Cleaning

First, I had to create a text cleaner to help get rid of non asc-ii characters and other pieces of strings that I did not need.

cleaner

I noticed that many of the tweets did not contain any product or company names. Using regular expression, I looked for the company names in the text and gave values to the product or company column.

```
# Lump apple products as Apple instead of having different products
# This is still cleaning for the older tweets
brands2={'Google':'Google', 'Apple' : 'Apple', 'Android': 'Android', 'iPad':'Apple', 'i-pad':'Apple' ,'iPhone': 'Apple'}
for key, values in brands2.items():
    clean_old_df.loc[clean_old_df['tweet_text'].str.contains(key, case=False), 'product or company']= values
```
executed in 111ms, finished 12:15:42 2021-07-28

For rows that already contained a company name or product name, I decided to reduce the number of options into three: Apple, Google, and Android.

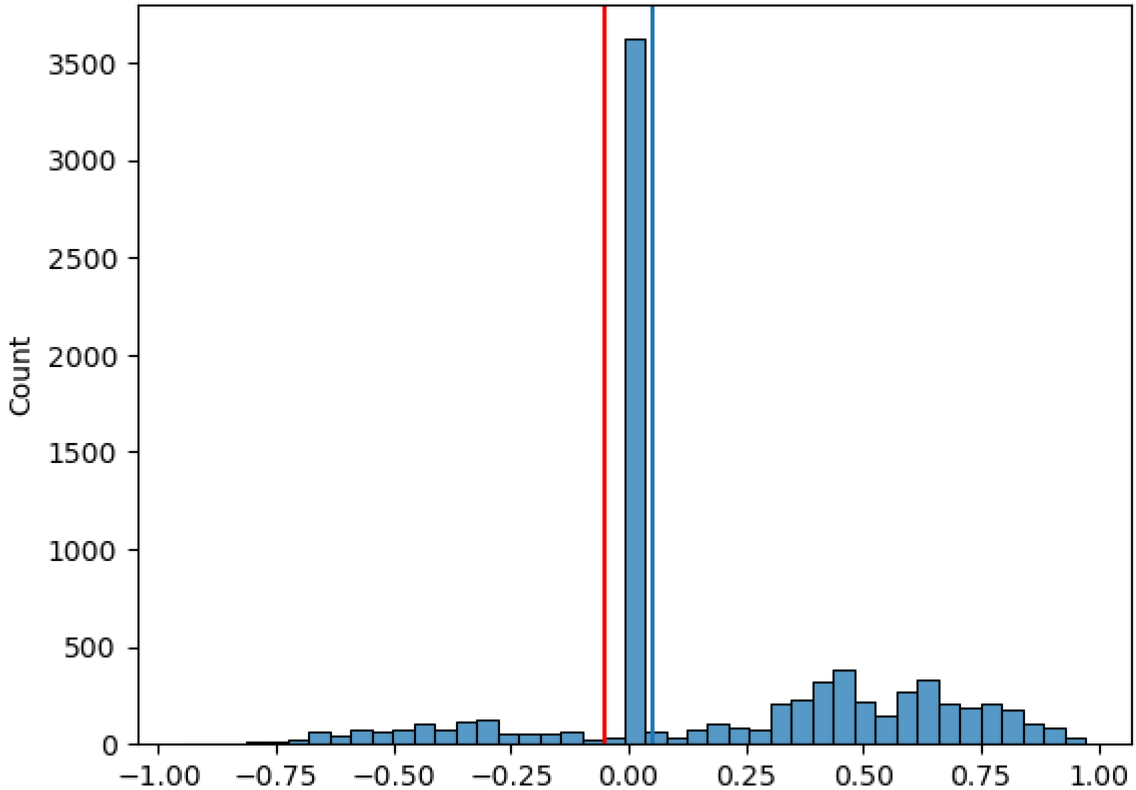
VADER Sentiment Analysis

From the VADER sentiment analysis, I collected the 'compound' values as they represented the most accurate sentiment depiction. If the values were bigger than or equal to .05, it was considered positive; if less than or equal to -.05, negative; and the rest were considered neutral.

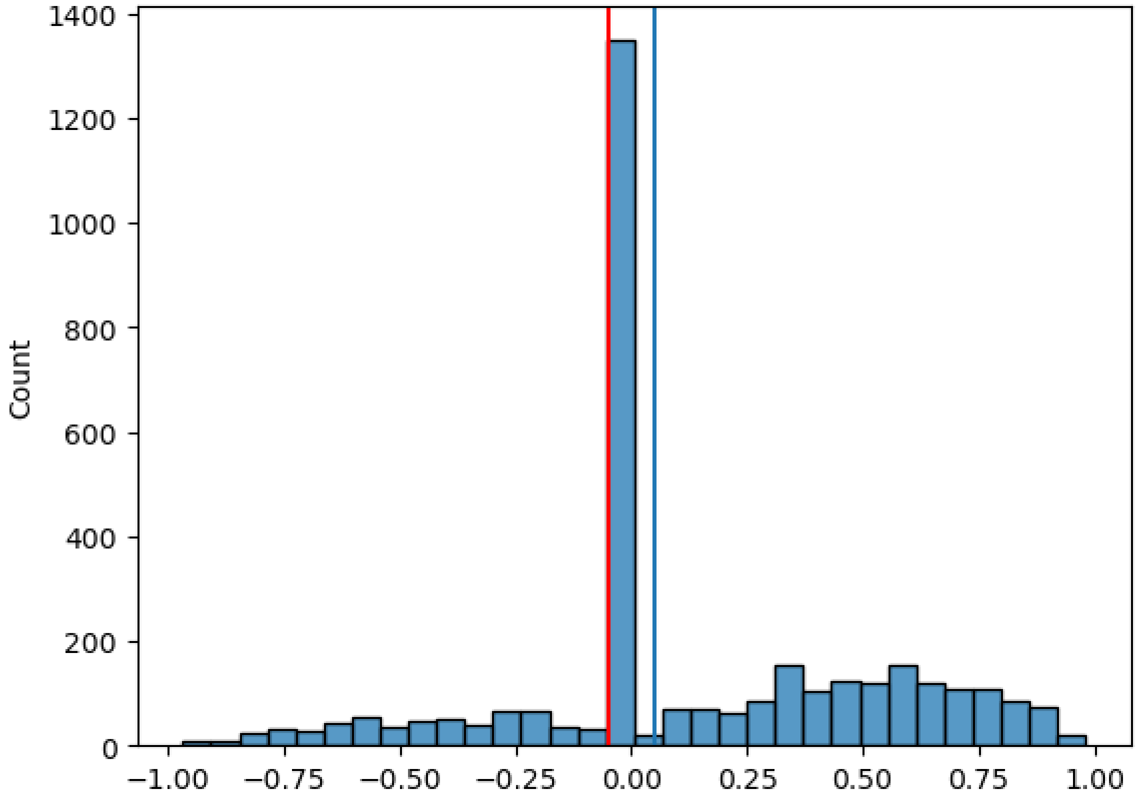
```
# Vader Sentiment Analysis
sid = SentimentIntensityAnalyzer()
old_tweet_sentiment=[]

for text in clean_old_df2['clean_text']:
    ss=sid.polarity_scores(text)
    old_tweet_sentiment.append(ss['compound'])
old_tweet_sentiment
```

The following shows the distribution of the sentiments.

Old Tweet

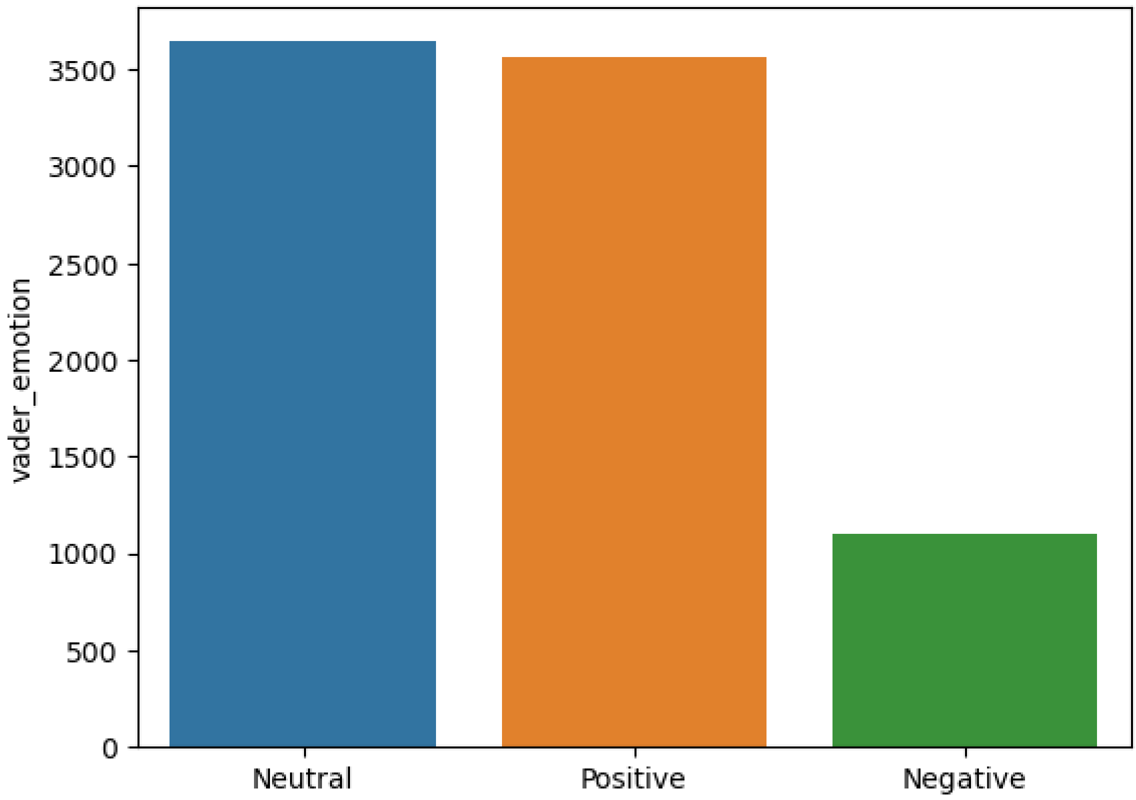


New Tweet

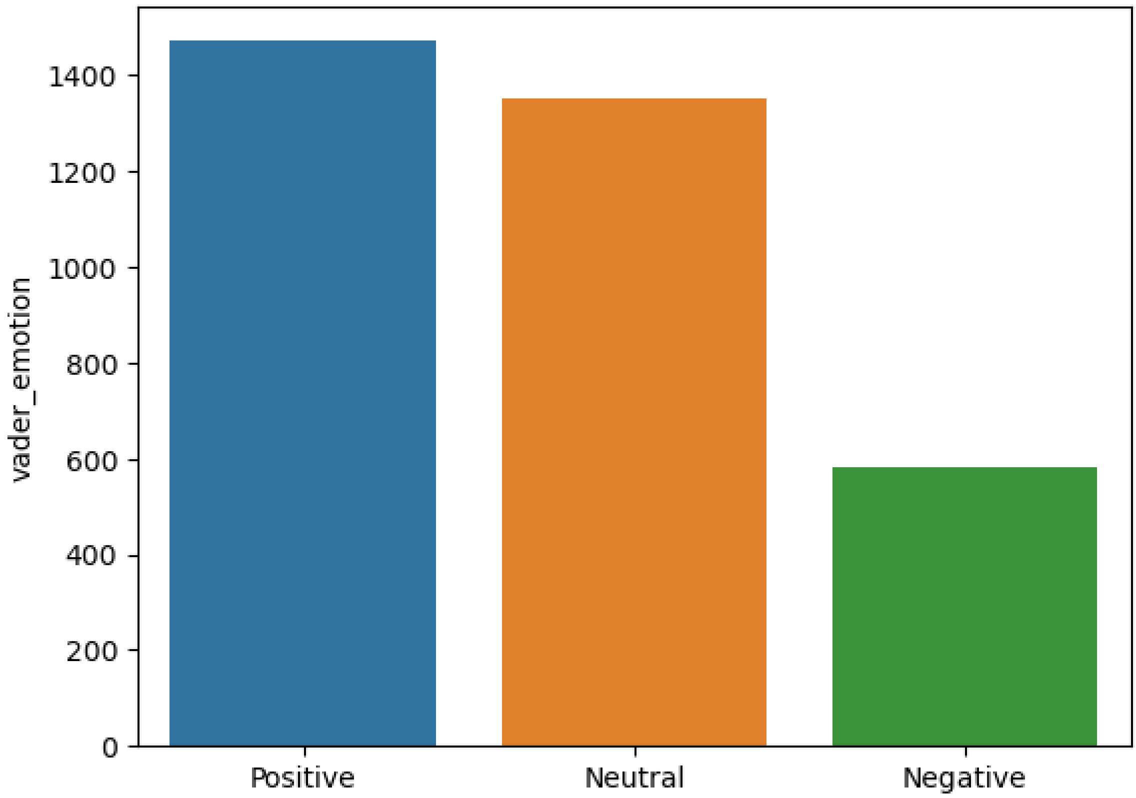

Count
1400
1200
1000
800
600
400
200
0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

From these distributions I noticed that 0 was the most common sentiment value and that the number of negative tweets were much lower those of positive and neutral tweets.

Old Tweet



New Tweet



When I compared the initially given emotion of the tweets to the emotion results from vader sentiment analysis, only slightly more than half of the tweets' sentiment had been correctly identified.

```
print(clean_old_df2['emotion_match'].value_counts())
clean_old_df2['emotion_match'].groupby(clean_old_df2['vader_emotion']).value_counts()
```

executed in 14ms, finished 16:12:48 2021-07-28

```
True     4509
False    3797
Name: emotion_match, dtype: int64

vader_emotion  emotion_match
Negative       False             880
               True              219
Neutral        True             2503
               False            1140
Positive       True             1787
               False            1777
Name: emotion_match, dtype: int64
```

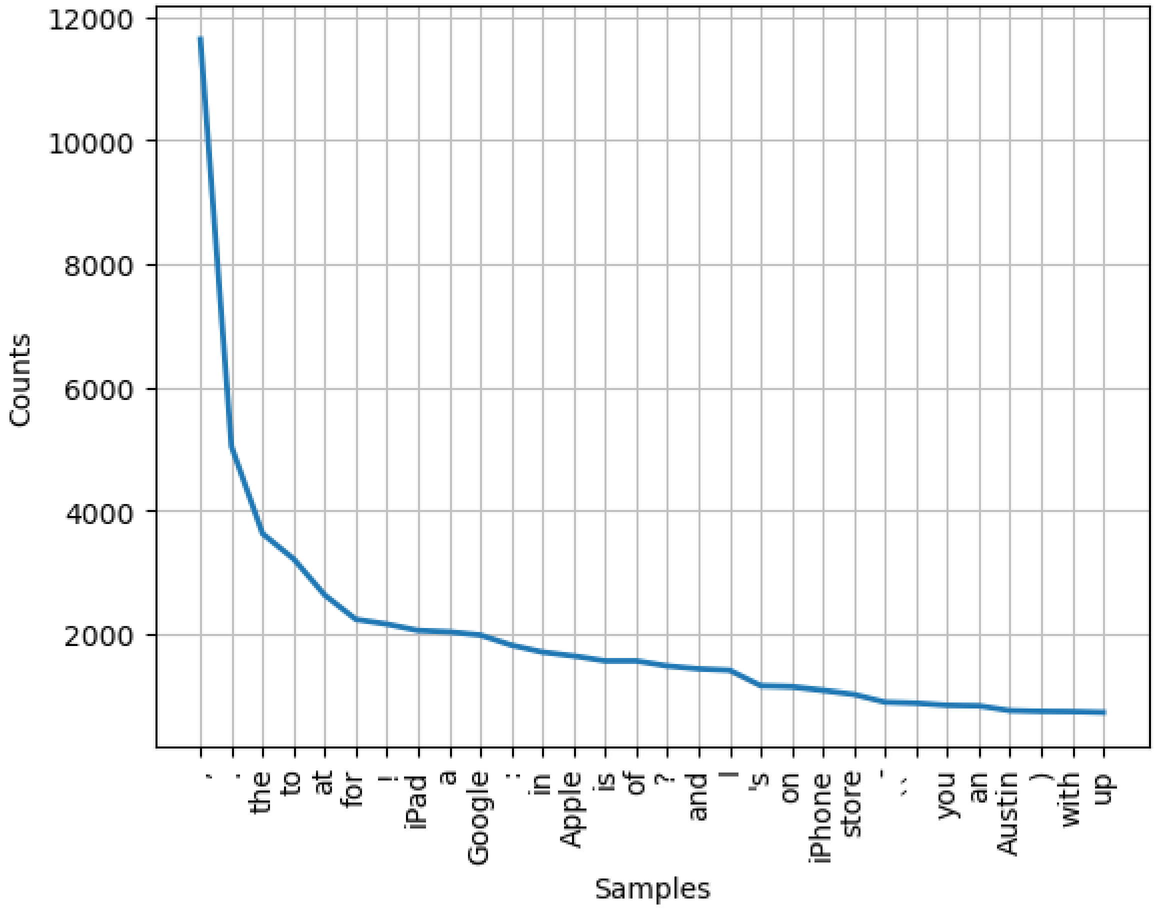
Tokenizing and Stop Words

For NLP analysis, it is important to have a token list. Token list is splitting a text file into words or characters.

```
# Create a list of all the common words in the old tweet
corpus_old = clean_old_df2['clean_text'].to_list()
token_list_old= word_tokenize(', '.join(corpus_old))
```

executed in 1.08s, finished 16:12:50 2021-07-28

Using this, I created a frequency distribution plot of the most commonly used words/characters



Then I created a list of common stopwords, added punctuations, as well as the company names and other common Twitter terms.

```
#Stop words
stop_words=stopwords.words('english')
stop_words.sort()
stop_words
```

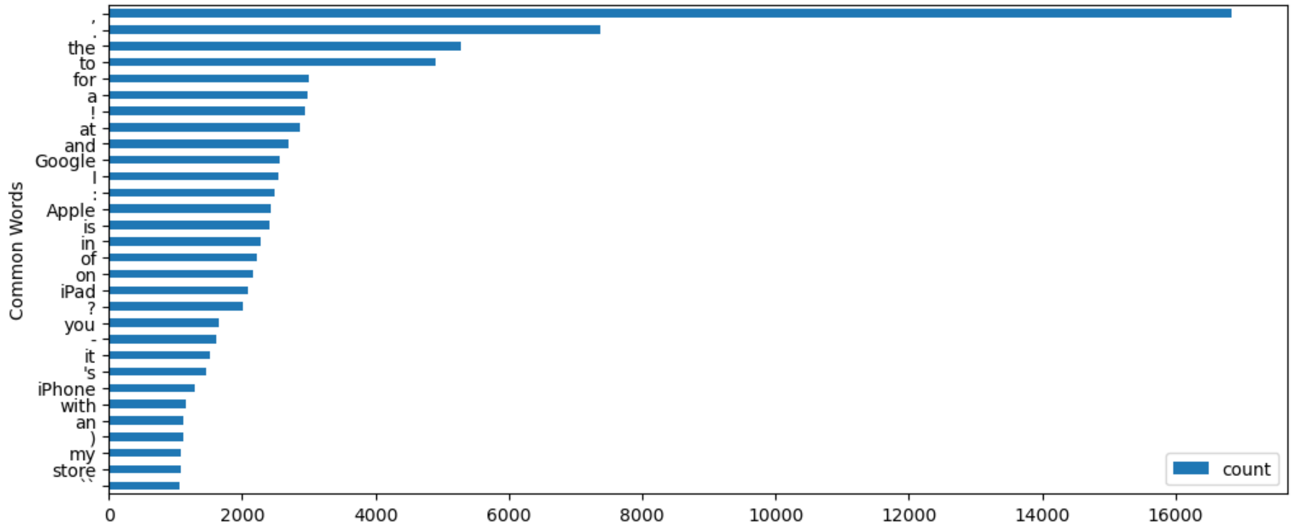
executed in 14ms, finished 16:12:53 2021-07-28

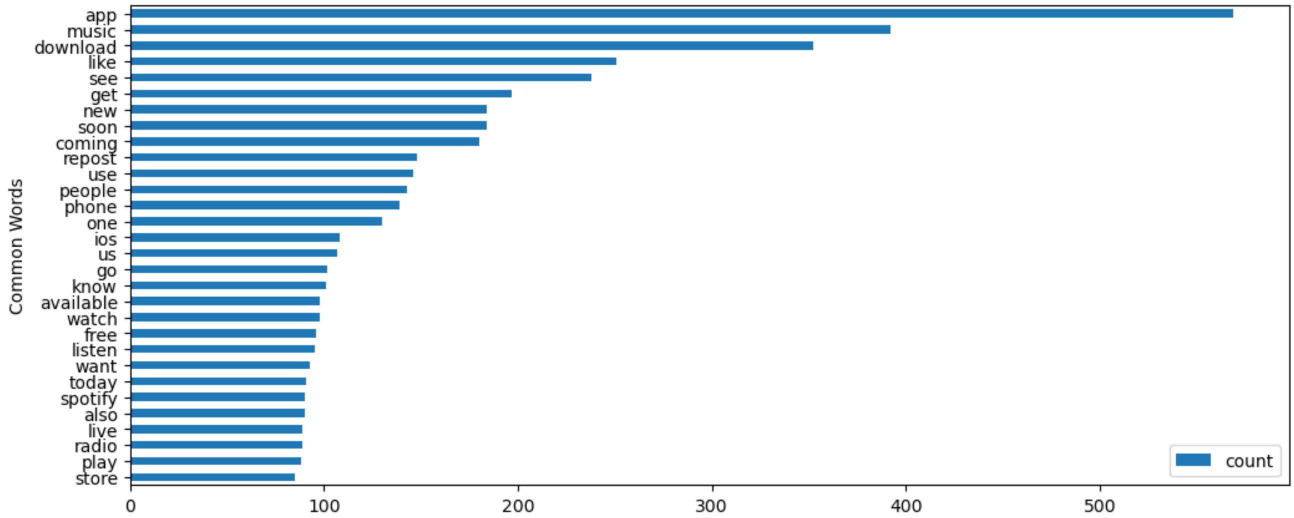
...

```
# First I am including punctuation marks, common twitter phrases such as rt and mention;
# Quotation marks were also included since they were not part of the string.punctuation
# Finally, posessives such as 's and the company names and products were included in the stop words
stop_words.extend(string.punctuation)
stop_words.extend(['RT', 'mention','SXSW','link'])
stop_words.extend(['“','”','...',"''",'’','``'])
stop_words.extend(["'s","n't"])
stop_words.extend(['apple','google','android','apple','ipad','i-pad','iphone'])
```

executed in 14ms, finished 16:12:53 2021-07-28

First, I created a horizontal frequency distribution plot that used a token list from both the old and new tweets, then I created the same frequency distribution plot that used the stop words.



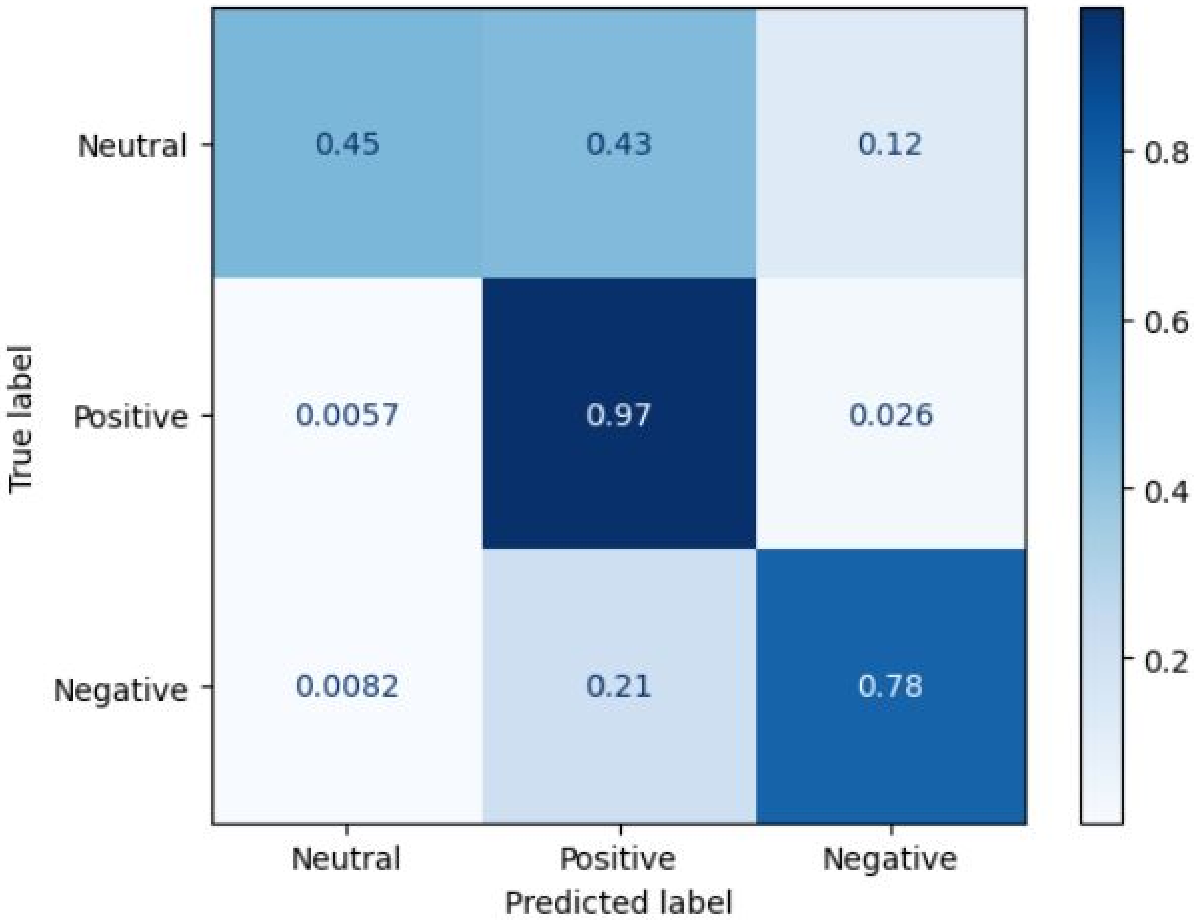


Data Modeling

For modeling, I chose Random Forest and Naive Bayesian models. I chose them because they were the best fit for doing NLP analysis. The random forest models for both the old and new tweets were created without any parameters at first, then it was pipelined, then refined again using GridSearch CV. In general, the recall score for the negative tweets were much lower than those of its counterparts. This is becaus, as mentioned before, the total number of negative tweets were much smaller than those of positive tweets or neutral tweets.

Random Forest

```
              precision    recall  f1-score   support

     Neutral       0.91      0.45      0.60       343
    Positive       0.73      0.97      0.83      1050
    Negative       0.93      0.78      0.85      1098

    accuracy                           0.81      2491
   macro avg       0.85      0.73      0.76      2491
weighted avg       0.84      0.81      0.81      2491

Training Score = 1.00
Test Score = 0.81
```



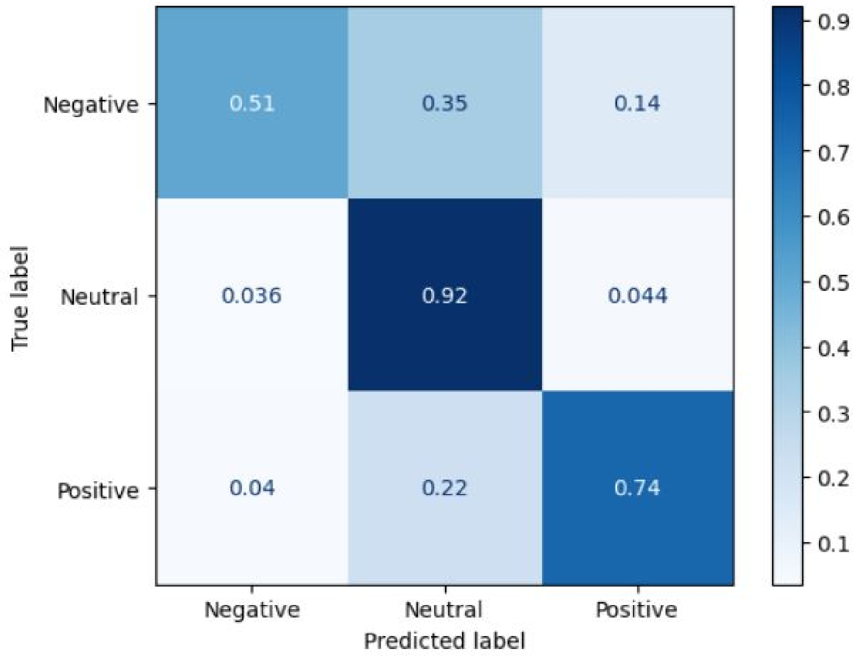
Old Tweet Random Forest with grid search

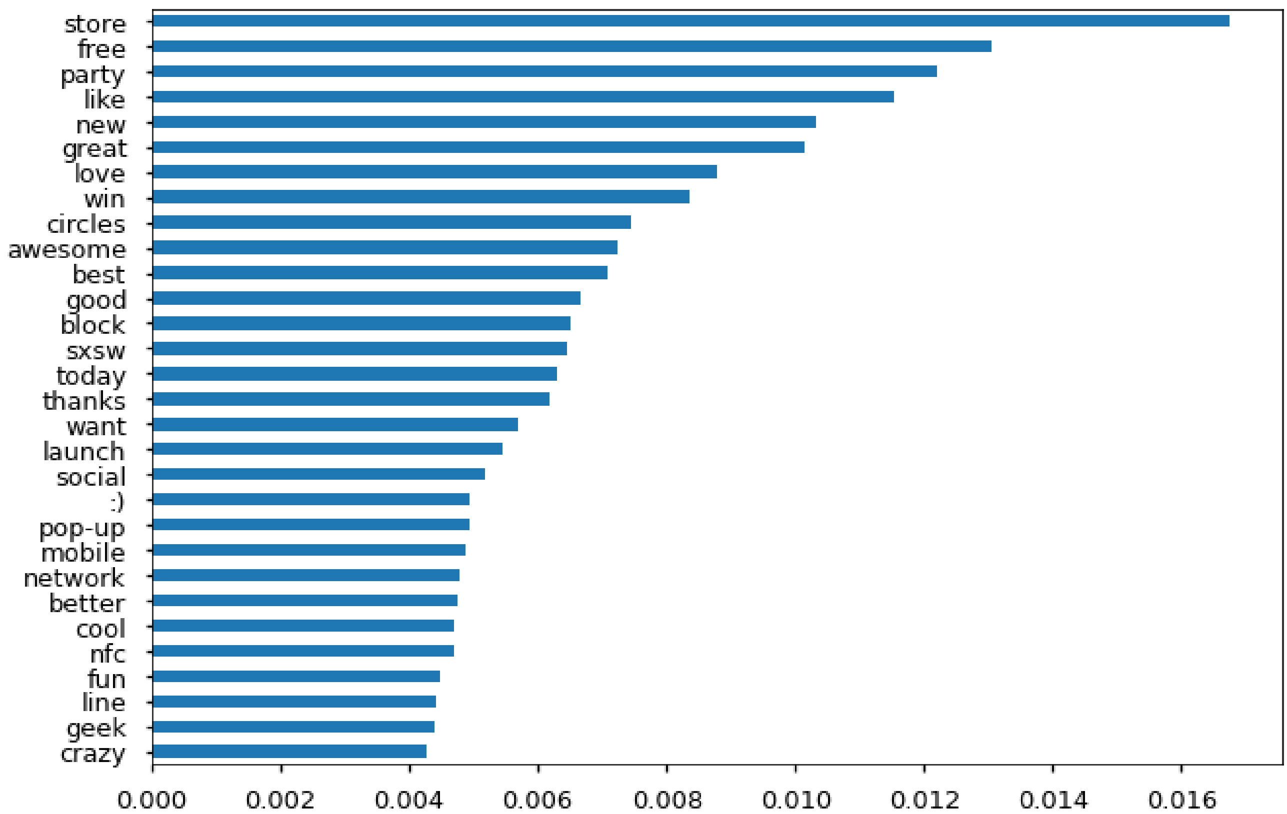
Along with the confusion matrix of the models, I found the important features that the random forest model used to determine the texts' sentiment. In this case, the important features are words that helped the model determine sentiments. Each of the graph shows 30 most important features/words. Words such as like, free, love, and best were near the top for both old and new tweets.

```
1 model_score(gs.best_estimator_, old_X_test, old_y_test, X_train= old_X_train, y_train= old_y_train)
```

executed in 767ms, finished 12:17:39 2021-07-28

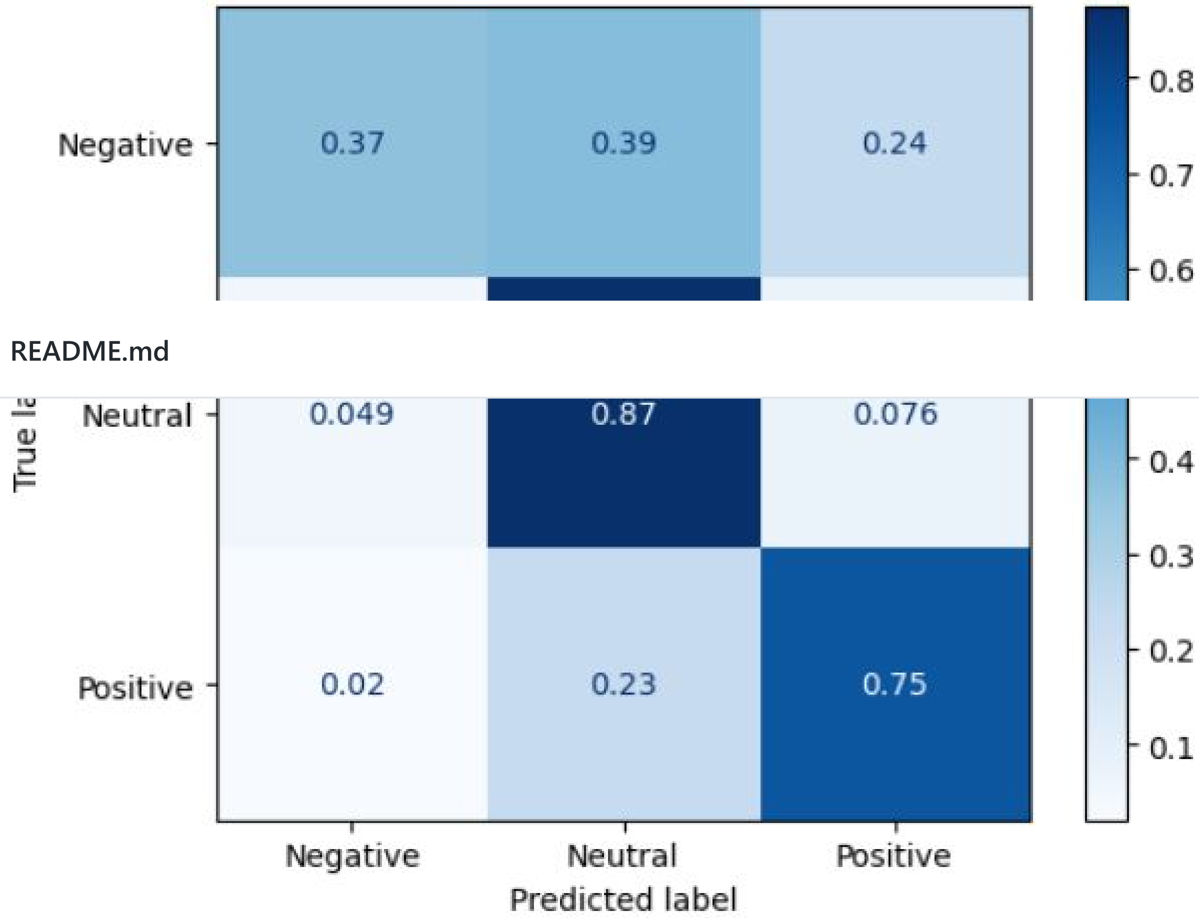
```
              precision    recall  f1-score   support

    Negative       0.68      0.51      0.58       338
     Neutral       0.74      0.92      0.82      1091
    Positive       0.89      0.74      0.81      1063

    accuracy                           0.79      2492
   macro avg       0.77      0.72      0.74      2492
weighted avg       0.80      0.79      0.78      2492

Training Score = 0.95
Test Score = 0.79
```



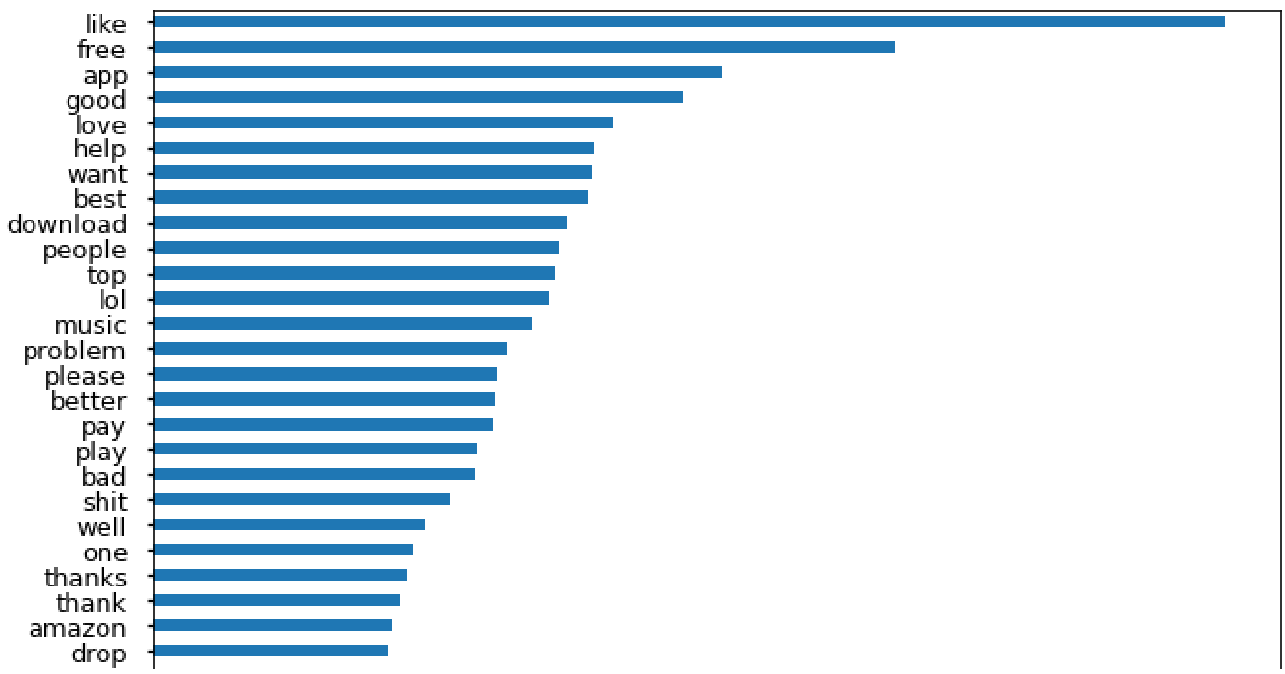


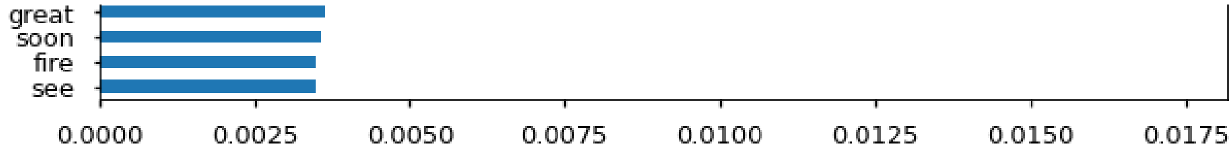
New Tweet Random Forest with grid search

```
              precision    recall  f1-score   support

    Negative       0.69      0.37      0.48       175
     Neutral       0.68      0.87      0.76       406
    Positive       0.82      0.75      0.78       441

    accuracy                           0.73      1022
   macro avg       0.73      0.66      0.68      1022
weighted avg       0.74      0.73      0.72      1022

Training Score = 0.99
Test Score = 0.73
```



README.md

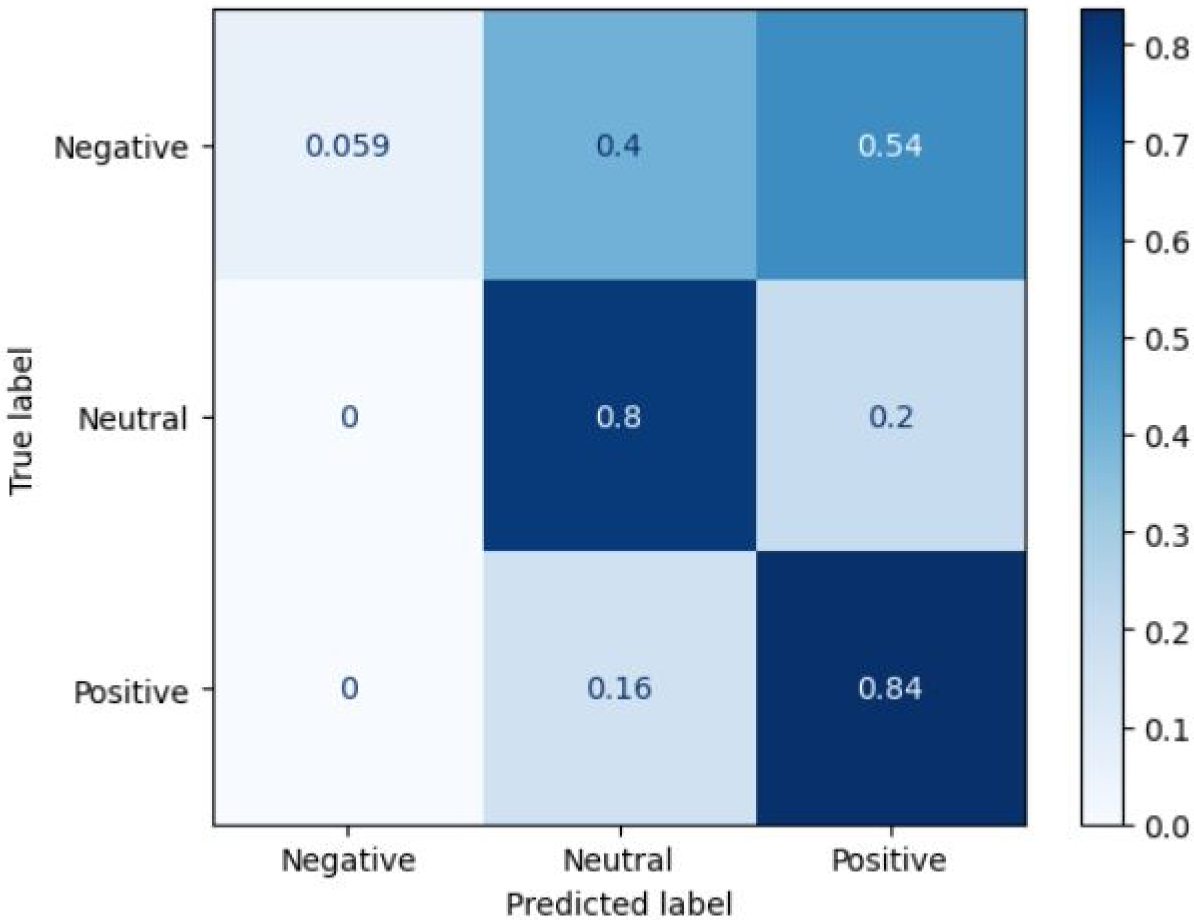




Naive Bayesian

Comapred to the random forest models, the navie bayesian models did not perform as well, especially before performing the grid search.

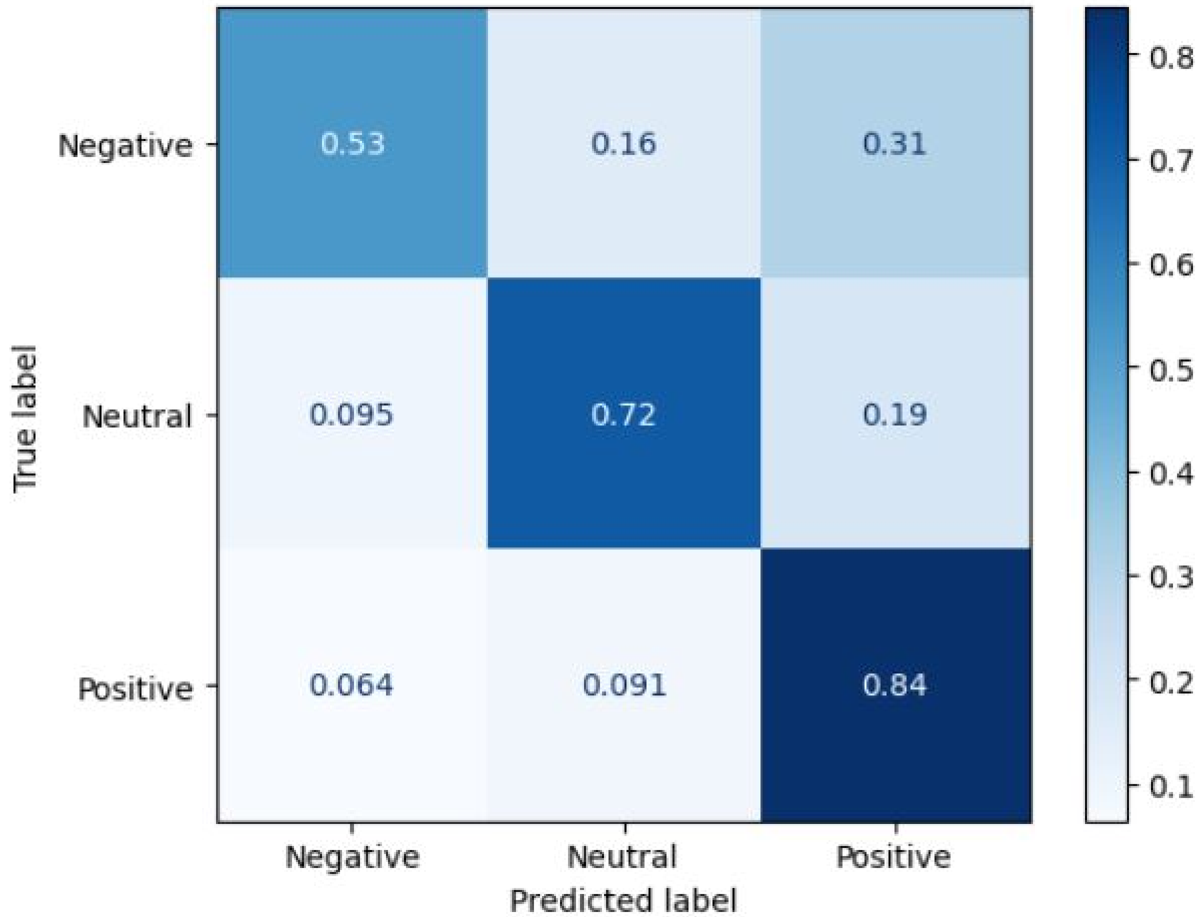
```
              precision    recall  f1-score   support

    Negative       1.00      0.06      0.11       338
     Neutral       0.74      0.80      0.77      1091
    Positive       0.69      0.84      0.76      1063

    accuracy                           0.72      2492
   macro avg       0.81      0.57      0.55      2492
weighted avg       0.75      0.72      0.67      2492

Training Score = 0.81
Test Score = 0.72
```



As can be seen, the recall score for the negative tweets are extremely low. This however improves after performing the grid search.

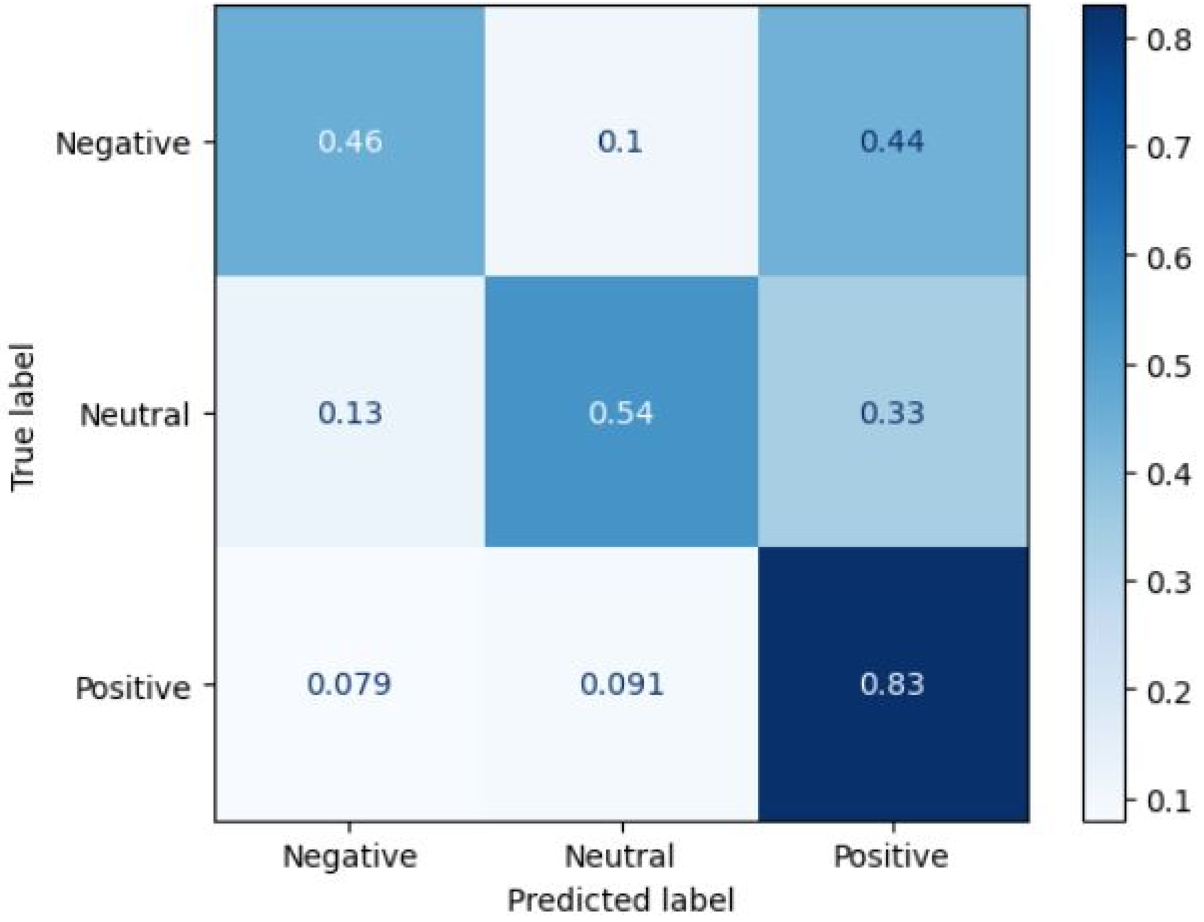
Old Naive Bayesian with grid search

```
              precision    recall  f1-score   support

    Negative       0.51      0.53      0.52       338
     Neutral       0.84      0.72      0.77      1091
    Positive       0.74      0.84      0.79      1063

    accuracy                           0.75      2492
   macro avg       0.70      0.70      0.70      2492
weighted avg       0.75      0.75      0.75      2492

Training Score = 0.95
Test Score = 0.75
```



New Naive Bayesian with grid search

```
              precision    recall  f1-score   support

    Negative       0.48      0.46      0.47       175
     Neutral       0.79      0.54      0.64       406
    Positive       0.63      0.83      0.72       441

    accuracy                           0.65      1022
   macro avg       0.63      0.61      0.61      1022
weighted avg       0.67      0.65      0.64      1022

Training Score = 0.98
Test Score = 0.65
```

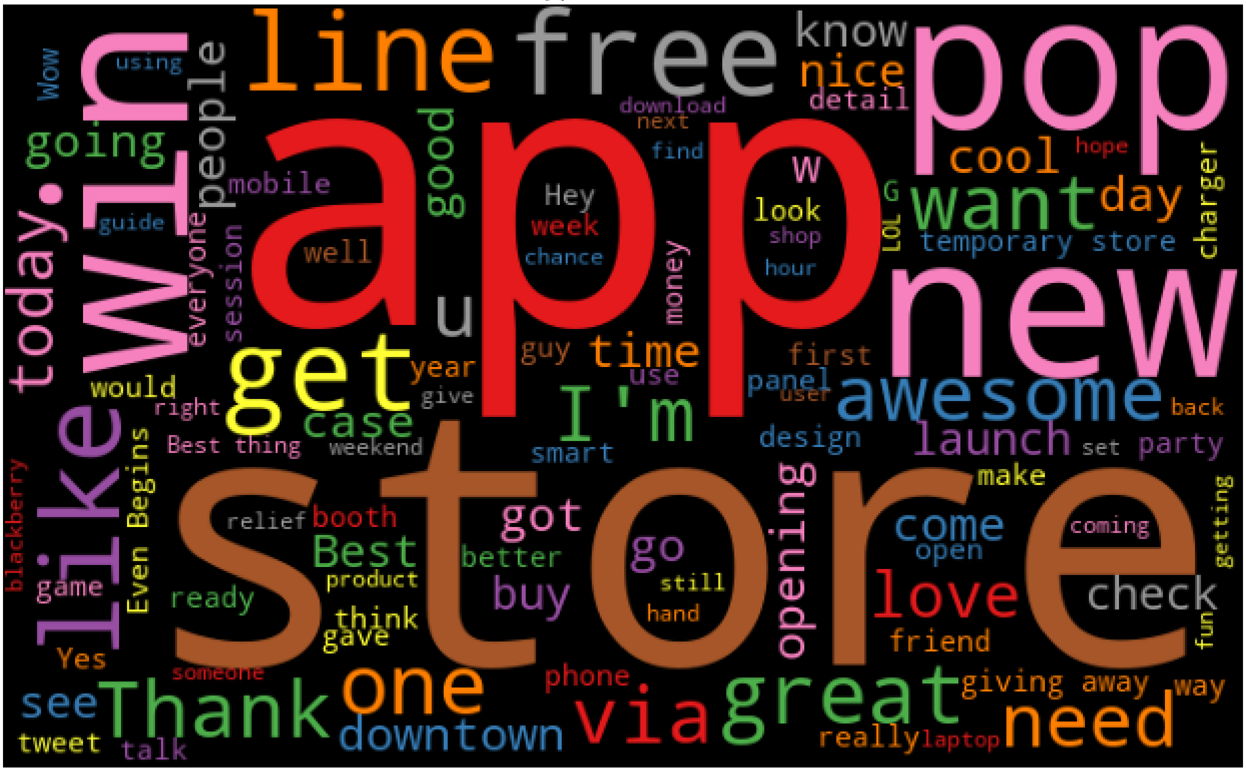


While the recall scores for the naive bayesian models with grid search are decent, the recall score for the neutral tweets from the new tweets are significantly lower than those of the random forest even though the number of neutral tweets were large.

Word Cloud

Using the tokens and stop words from above, I created some word clouds. The first two images of word clouds show common words from the tweets that had positive sentiment towards Apple in the old and new tweets.

Old Tweet Apple Positive Word Cloud



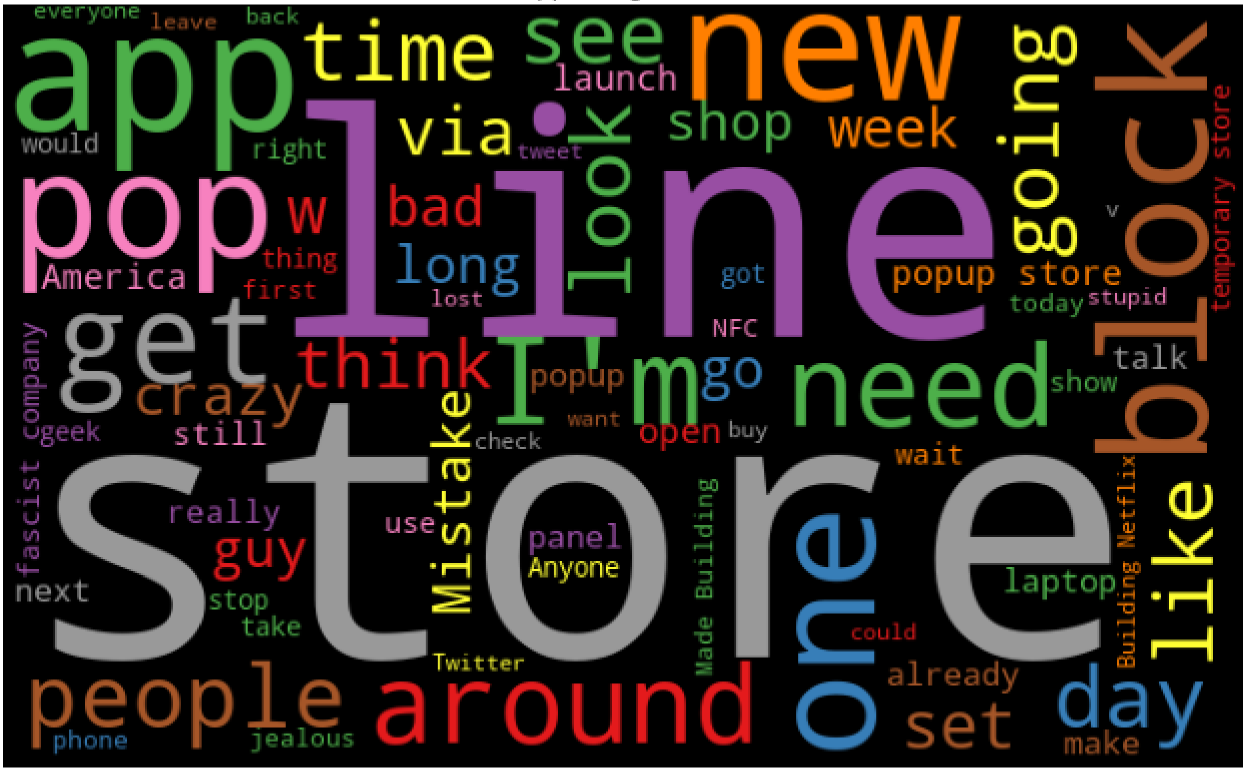
New Tweet Apple Positive Word Cloud

These are some of the most commonly appearing words in the word clouds that show how and why the sentiment towards the company has changed. These words and reason(s) for the sentiment change will be provided after each set of word cloud.

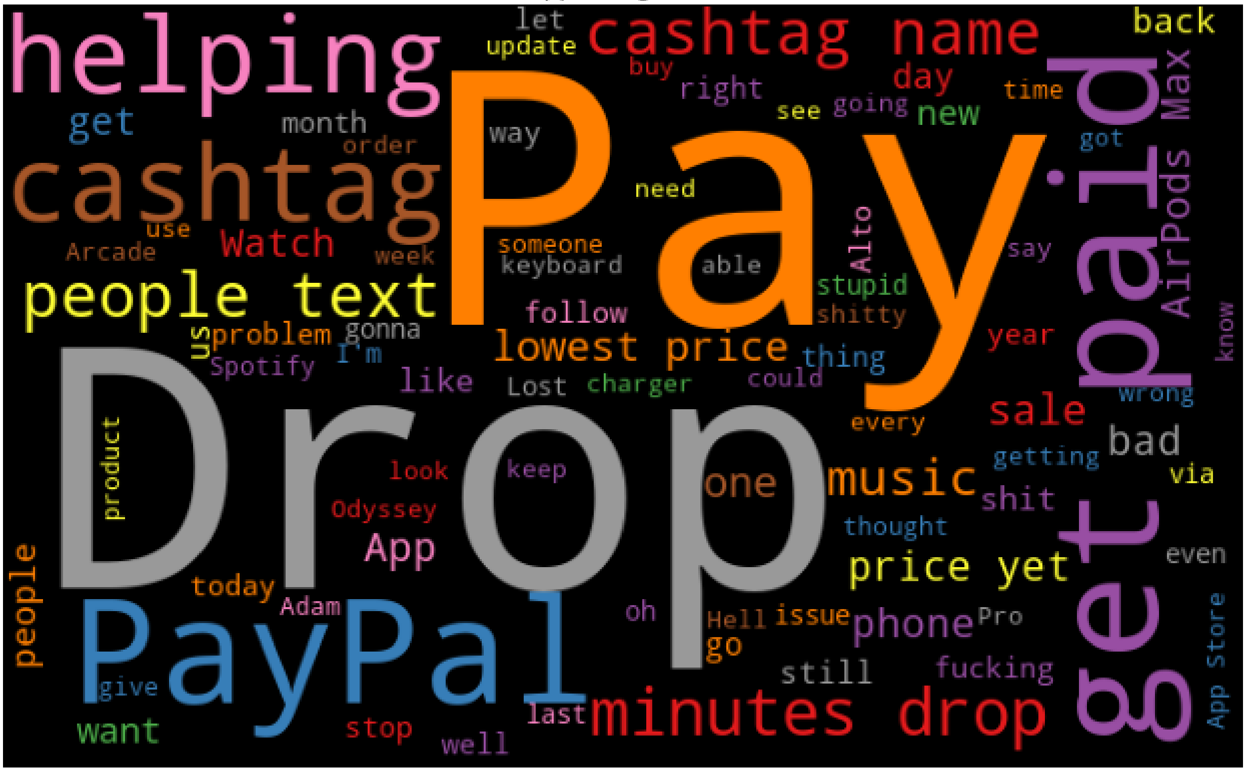
- Old Tweet: store, app, line, opening, win, great, Thank, want
- New Tweet: music, like, phone, Spotify, want, Watch

- Words associated with positive sentiment seems to change from one's experience at the Apple store or using their product apps such as Apple Music or Spotify.

Old Tweet Apple Negative Word Cloud



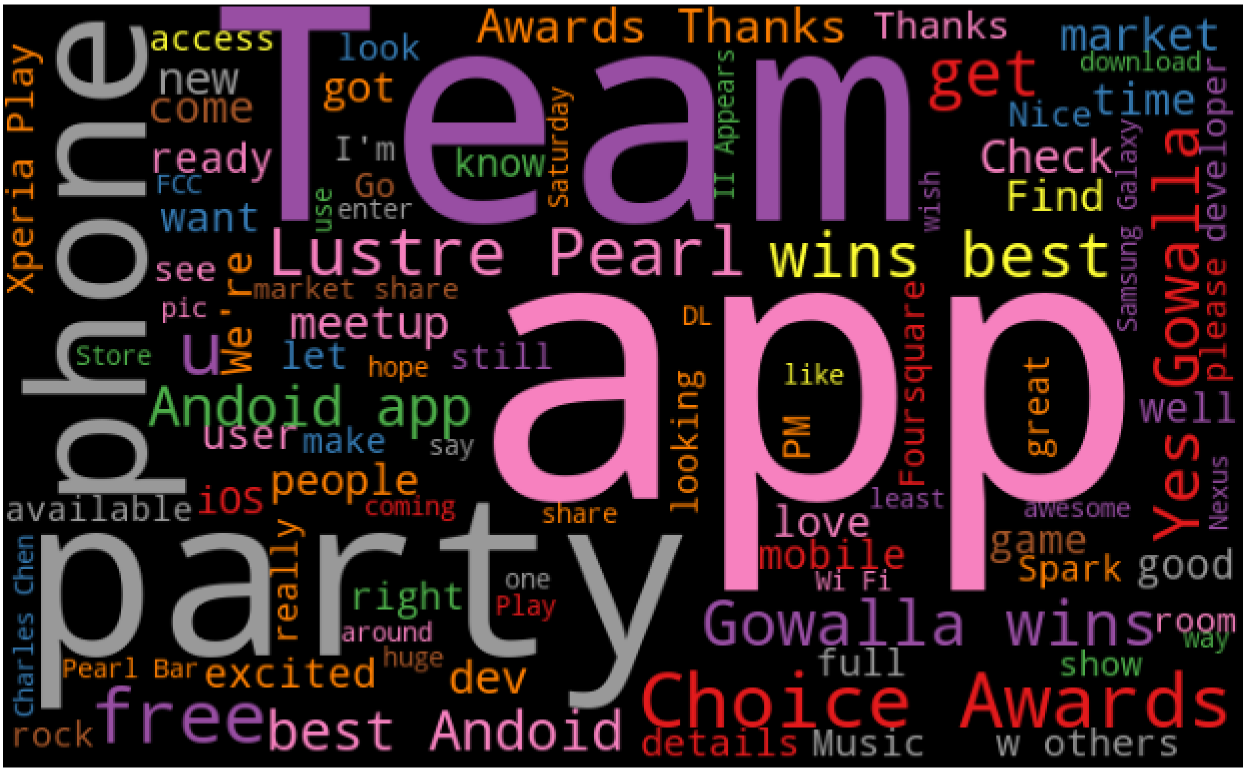
New Tweet Apple Negative Word Cloud



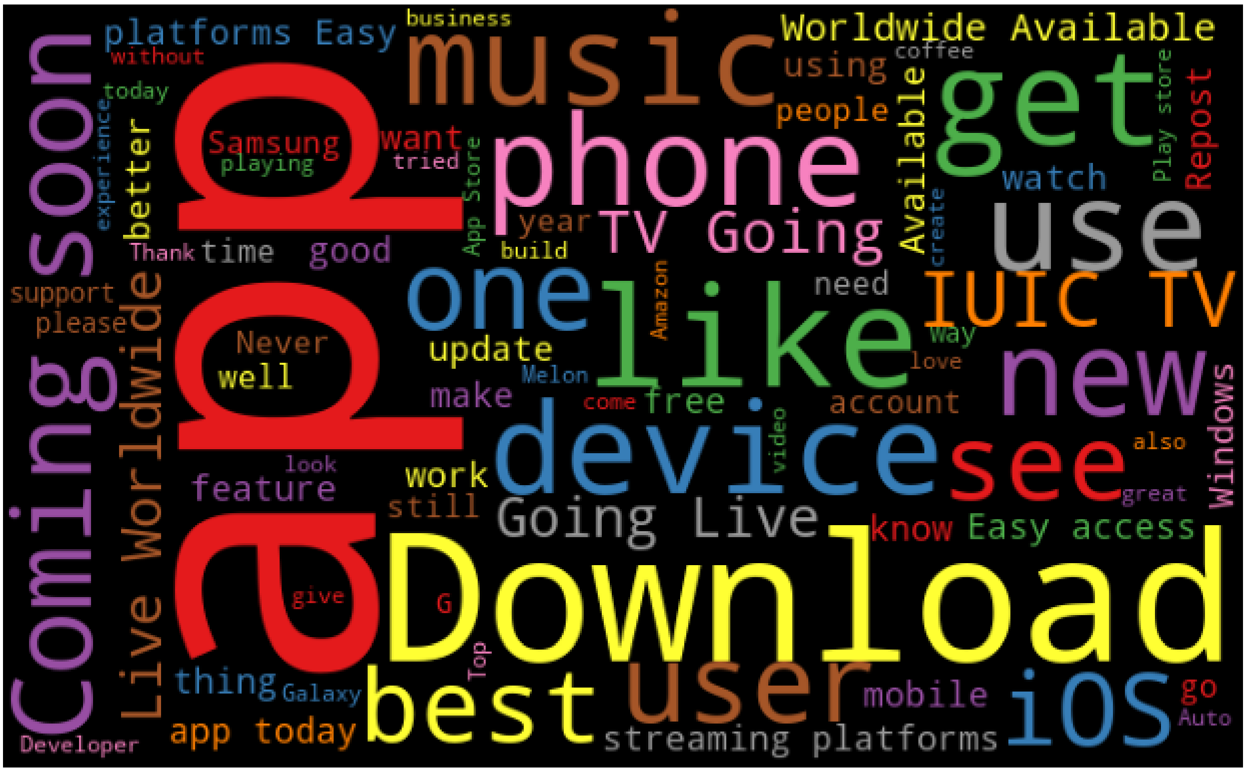
- Old Tweet: store, line, block, need, crazy
- New Tweet: price yet, people text, problem, stop, minutes drop
- As with before, the negative sentiment towards Apple has shifted from the Apple experience to people's discontent towards Apple's products and prices.

- Cashtag, get paid, cashtag name are all from advertisements that use an app called Cashpay. PayPal also seems to be associated with advertisements.

Old Tweet Android Positive Word Cloud



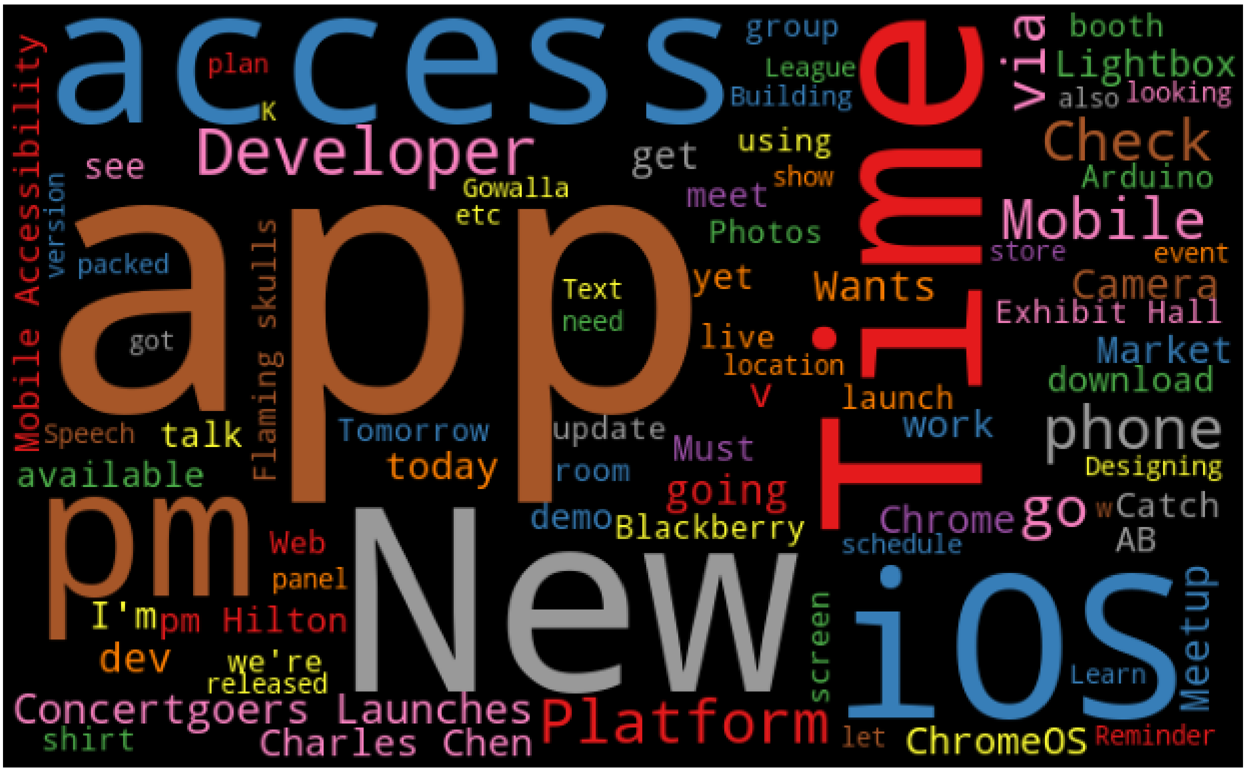
New Tweet Android Positive Word Cloud



- Old Tweet: Team, App, wins best, phone, best Android
- New Tweet: app, music phone, device, Easy access
- Like the tweets about Apple, people seem to use their phones for music apps more.

- From the wording in the old tweets, there is a rivalry between the users of Android phones and iPhone users, which we can no longer observe in the new tweets.

Old Tweet Android Neutral Word Cloud



New Tweet Android Neutral Word Cloud

- Old Tweet: app, New, Time, iOS, access, Platform, ChromeOS
- New Tweet: app, music, Download, RADIO APP, iOS
- Both old and new tweets seem to focus on Android apps, and judging form the word iOS, these tweets seem to either compare the type of apps or show the availability in

both the Google Play Store and the iOS App Store.

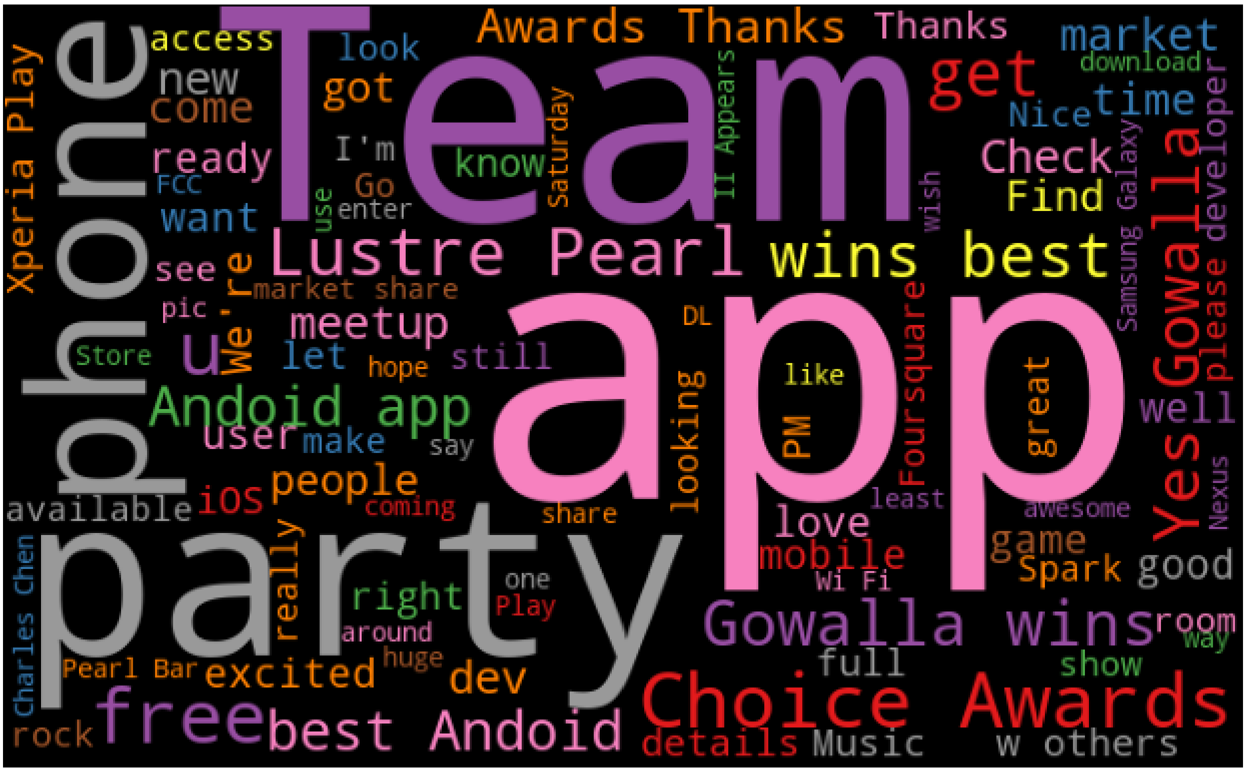
Old Tweet Google Neutral Word Cloud

New Tweet Android Neutral Word Cloud

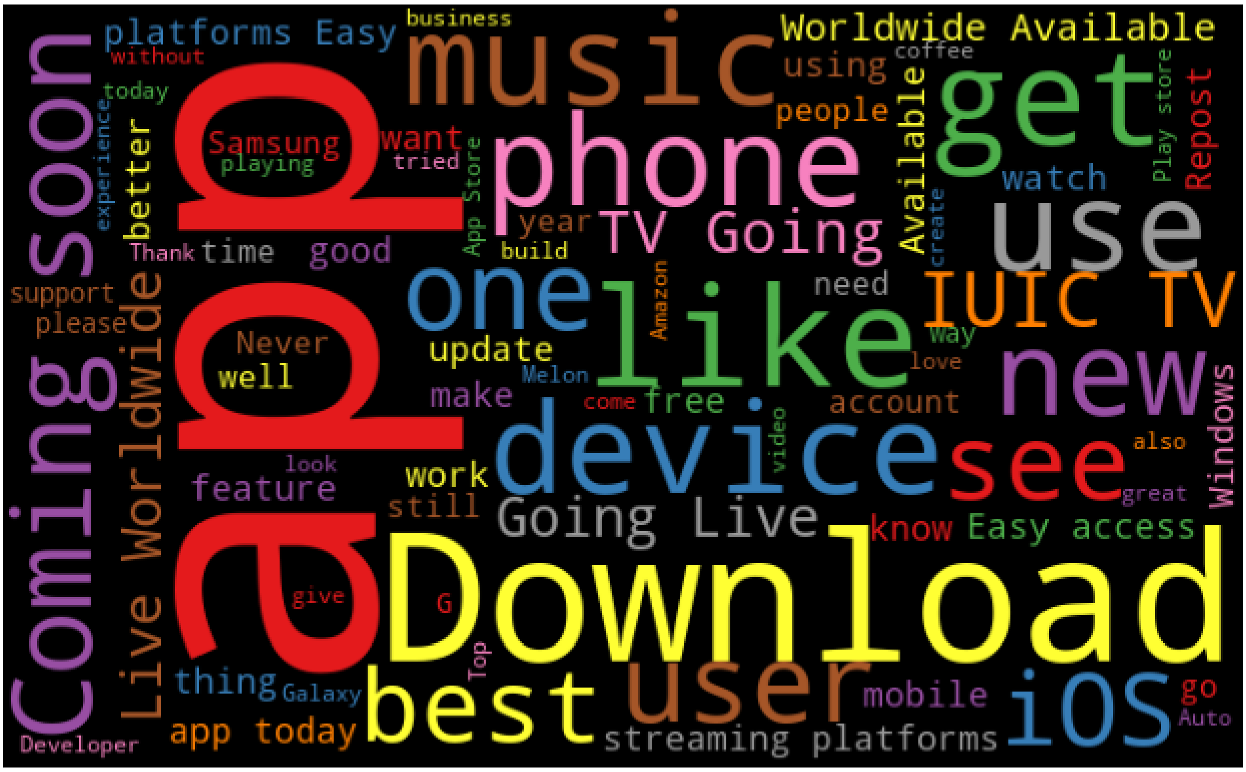
- Old Tweet: New Social, map, great, party, Social Network
- New Tweet: like, know, use, see, help, search, find
- Overall, the number of words related to positive sentiment towards Google seems to have decreased.

- Similar to the tweets with neutral sentiment, new tweets seem to be more related to the Google's search engine rather than Google as a company.

Old Tweet Android Positive Word Cloud



New Tweet Android Positive Word Cloud



- Old Tweet: Social Network, Major New, Possibly Today, Launch
- New Tweet: Search, New, use, map, soft detection, translate
- Neutral sentiment towards Google seems to have shifted from texts relating to social media to general uses of the Google search engine.

Model Recommendation

For both of the tweets, the Random Forest models after the gridsearch performed the best. The random forest models before the pipeline and the models after performing the grid search performed similarly; however, because the low recall score for the negative tweets were improved to that of the original random forest models while maintaining the relatively high recall scores for the neutral and positive tweets, I recommend the **Random Forest models using gridsearch**. As mentioned before, the number of tweets with negative sentiments were only a third of the size of either the positive or neutral neutral tweets which led to the poor recall rate of negative tweets for all models.

Conclusion

Future Work

There are many things that I would like to incorporate into this project that I did not have the necessary skills or time to perform, but given the chance I would like to do these for the future:

1. Collect similar amounts of tweets for all three sentiments
    - This would help improve the recall scores for the models and maybe give an insight as to why the naive bayesian models did not perform as well before gridsearch.
2. Create a time line of sentiments and events.
    - By collecting tweets from the past throughout to the present, I would be able to show how the sentiment trends gradually changes.
    - Incorporating events from the past could also help understand the sentiment trends and highlight specific actions that the company had taken that negatively or positively impacted the public's sentiment.

Final thoughts

This project was an adventure from the start as I had to learn how to collect my own data. Another aspect that I had not experienced before was the lengthy text cleaning that came with using tweets. And while the old tweet data were provided, I had to use the vader sentiment analysis as I had noticed much of the tweets did not correctly represent the tweet's sentiment. The aspect that most intrigued me about the Twitter sentiment towards these large companies is the obviously smaller number of negative emotion towards the companies and the overwhelming number of neutral sentiments towards them as well. I had initially expected to see a large increase in negative sentiment towards companies such as Apple as they have become less consumer friendly in their product designs. And as mentioned above in the future works section, I believe that creating a time line of the sentiment changes and events could help paint a clearer picture of how much these companies have changed in their policies and in the eyes of the consumers.

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- **Jupyter Notebook** 99.9%
- **Python** 0.1%