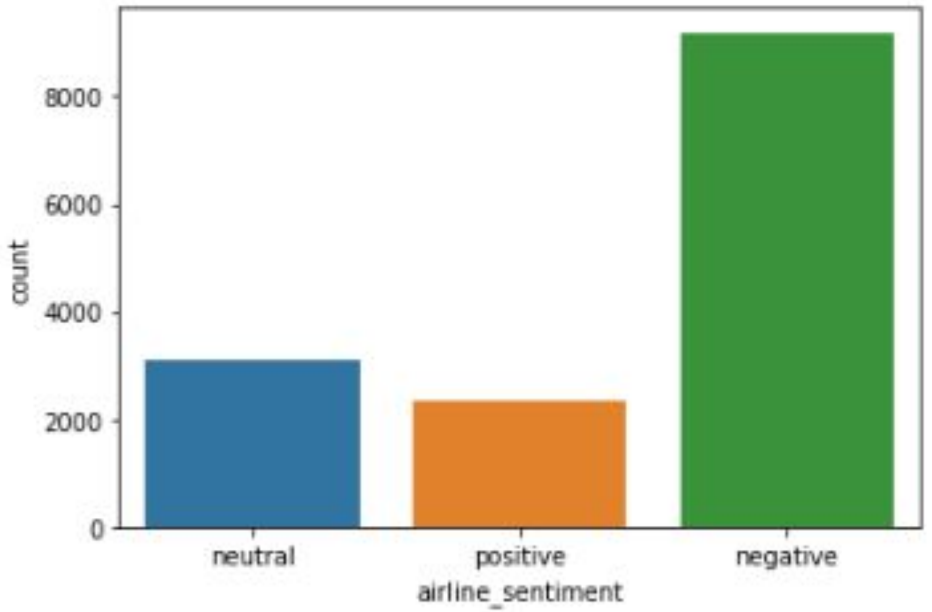
Sentiment Classifier

Define the Task

- Automatically classifying customer feedback is highly valuable to companies as it allows them to address/remedy negative feedback and response positively to positive feedback.
- An accurate classifier will help the company retain customers who had negative experiences with outreach programs while potentially increasing revenue from customers who had positive experiences by offering sales/discounts/rewards.
- I will be working with a dataset of tweets related to airlines that represent either positive, neutral, or negative sentiment towards said airline.
- I will be using F1 score to measure the effectiveness of models as it is a more robust measure of classification accuracy.

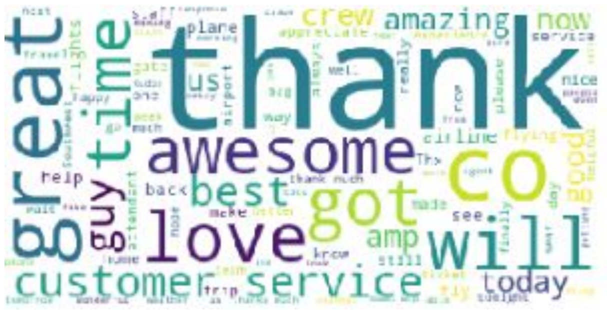
Explore the Data

- We see that the dataset is Unbalanced, having many more Negative examples.
- I removed the neutral examples From the data as it hurt the models And has limited upside.
- I tried balancing the dataset, but Increasing the number of positive Examples did not help the models.

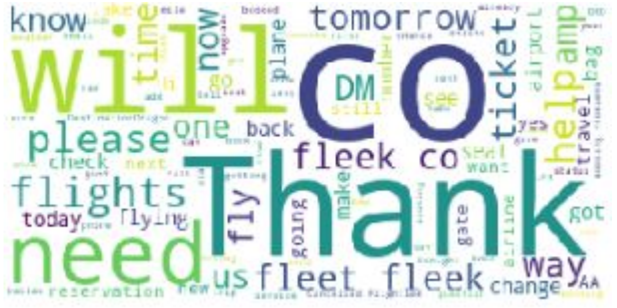


Common Words by Sentiment

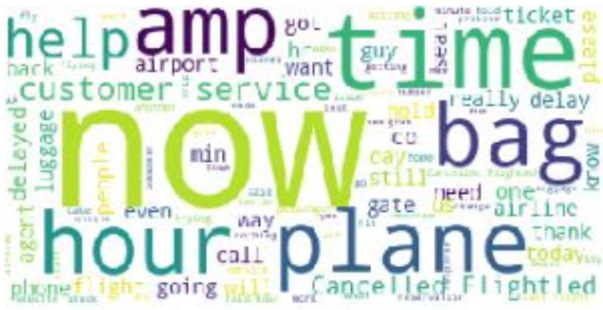
Positive



Neutral

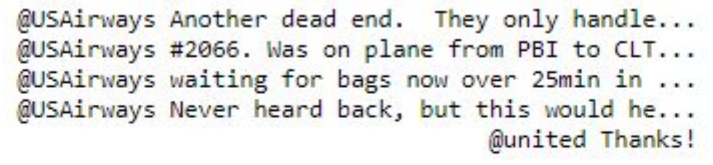


Negative



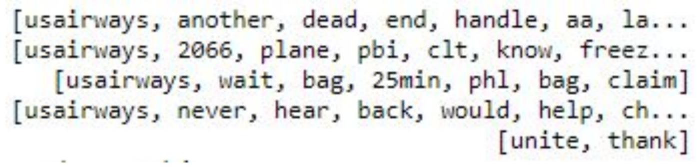
Text Preprocessing - Step 1

Before



```
@USAirways Another dead end.  They only handle...
@USAirways #2066. Was on plane from PBI to CLT...
@USAirways waiting for bags now over 25min in ...
@USAirways Never heard back, but this would he...
                                 @united Thanks!
```

After

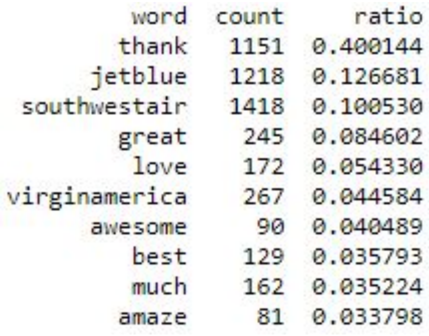


```
[usairways, another, dead, end, handle, aa, la...
[usairways, 2066, plane, pbi, clt, know, freez...
   [usairways, wait, bag, 25min, phl, bag, claim]
[usairways, never, hear, back, would, help, ch...
                                   [unite, thank]
```

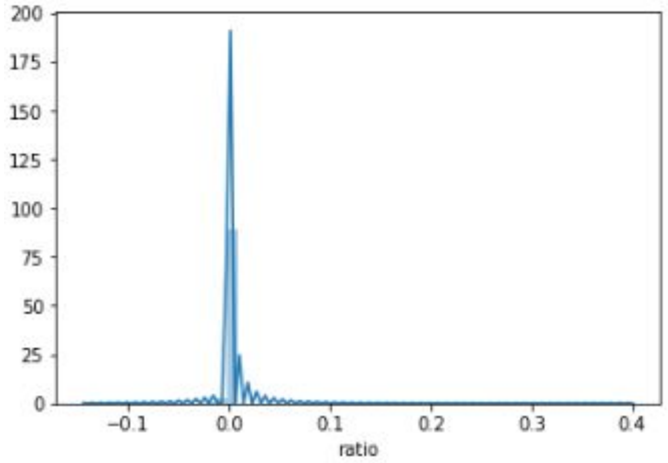
The first step in preparing the text is to remove punctuation, normalize the case by making everything lower case, removing extremely common words such as “is” or “a”, and lemmatizing the text, which means reducing words to their base form (e.g. “changed” to “change”).

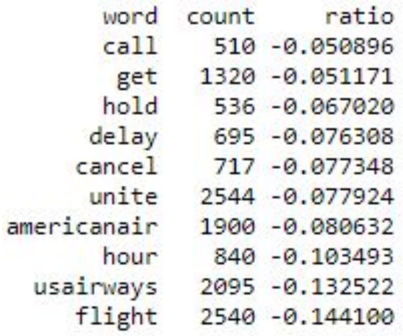
Text Preprocessing - Step 2

The second step was to find words that have high sentiment predictive power. I did this by comparing the percent of positive tweets a word was in as compared with the percent of negative tweets. We can see that many words had a ratio of 0, meaning they would likely have little predictive power. I removed words with little predictive power and also very rare words (appearing <5 times) which will help with mis-spellings.



```
         word  count     ratio
        thank   1151  0.400144
      jetblue   1218  0.126681
  southwestair   1418  0.100530
        great    245  0.084602
         love    172  0.054330
 virginamerica    267  0.044584
      awesome     90  0.040489
         best    129  0.035793
         much    162  0.035224
        amaze     81  0.033798
```



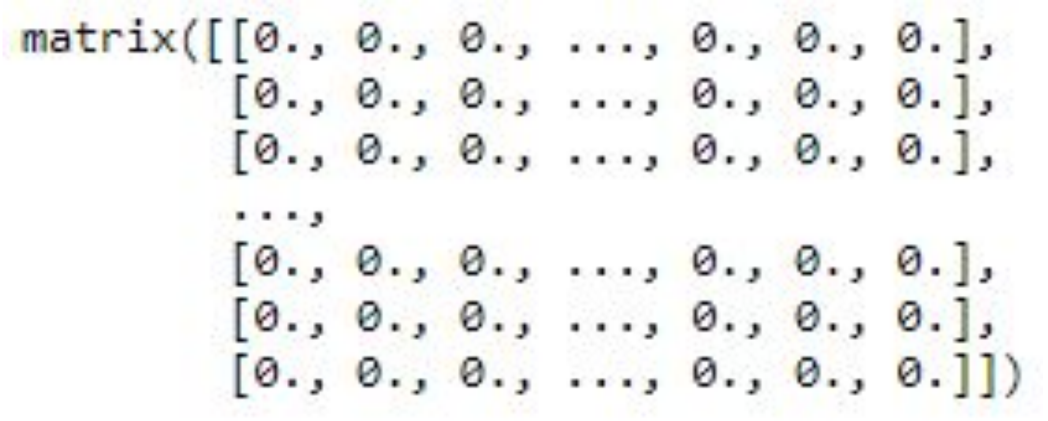


```
        word  count     ratio
        call    510 -0.050896
         get   1320 -0.051171
        hold    536 -0.067020
       delay    695 -0.076308
      cancel    717 -0.077348
       unite   2544 -0.077924
 americanair   1900 -0.080632
        hour    840 -0.103493
   usairways   2095 -0.132522
      flight   2540 -0.144100
```

Feature Extraction

The final step in preparing text for a machine learning model is to convert it into a document term matrix, which represents how much a word is represent in each tweet. I will use TFIDF to represent our features, which includes how frequently a term appears in a document, but also gives a penalty for how common a word is.

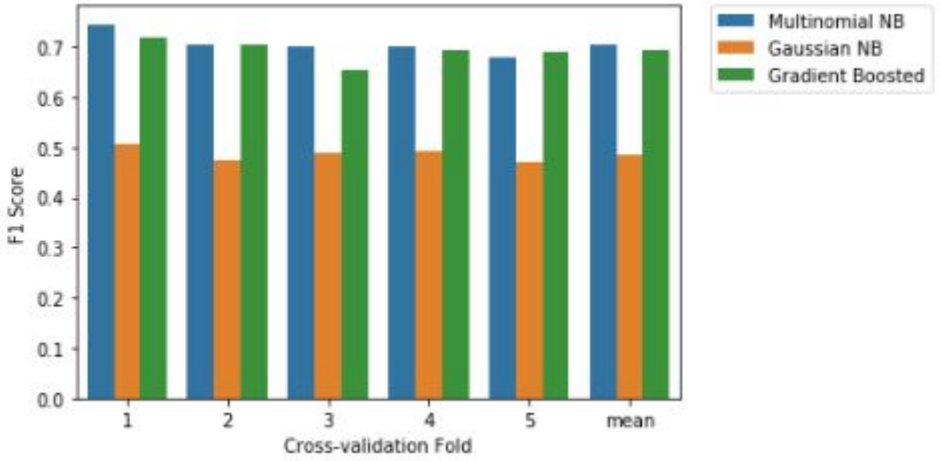
The matrix contains mostly 0's as it Lists the frequency of every word in The vocabulary for every tweet, and Most tweets will contain only a small Subset of the vocabulary.



```
matrix([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]])
```

Machine Learning Models

- The highest performing model is Multinomial Naive Bayes, which Even outperformed the Gradient Boosted model.
- Gaussian Naive Bayes performs Extremely poorly on this task, Indicating that the underlying Assumptions of normality are Likely not met.



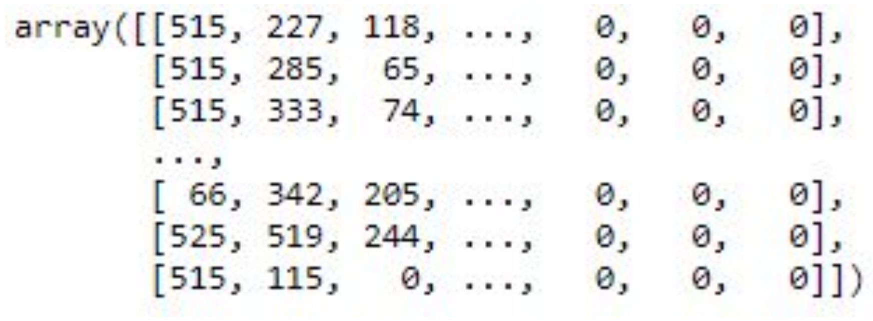
RNN Preprocessing

RNNs require that each word in the vocabulary be mapped to a unique number, and each series have an equal length input.

As such, I created a dictionary to map each new word to a new unique identifier, and padded all sequences with 0s so that the sequences all have matching length.

I will be building 3 RNN models:

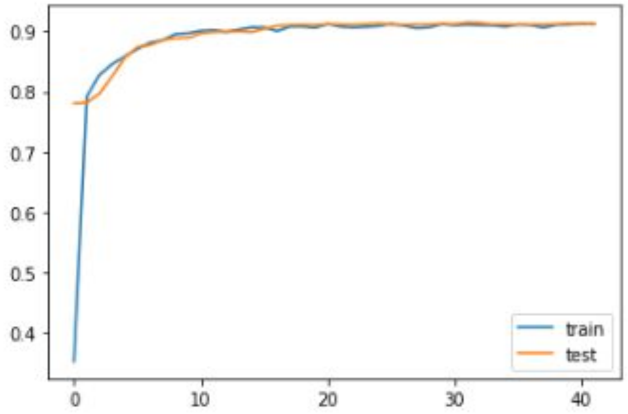
- With fully prepared tokens
- With rare and less informative tokens
- With stopwords



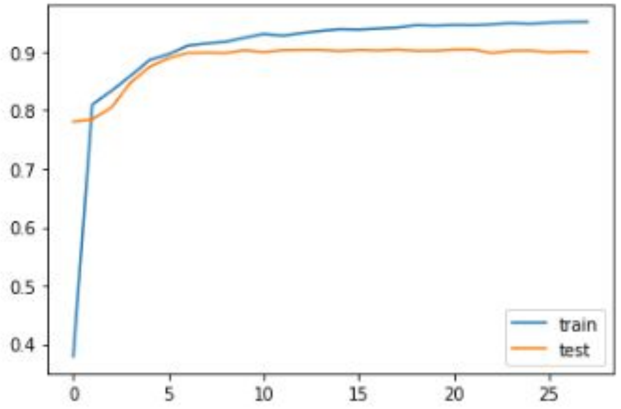
```
array([[515, 227, 118, ...,   0,   0,   0],
       [515, 285,  65, ...,   0,   0,   0],
       [515, 333,  74, ...,   0,   0,   0],
       ...,
       [ 66, 342, 205, ...,   0,   0,   0],
       [525, 519, 244, ...,   0,   0,   0],
       [515, 115,   0, ...,   0,   0,   0]])
```

RNN Performance

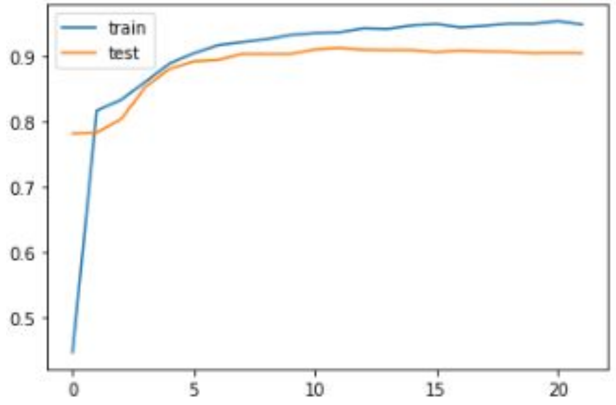
Prepared Tokens



Including Rare Words



Including Stopwords

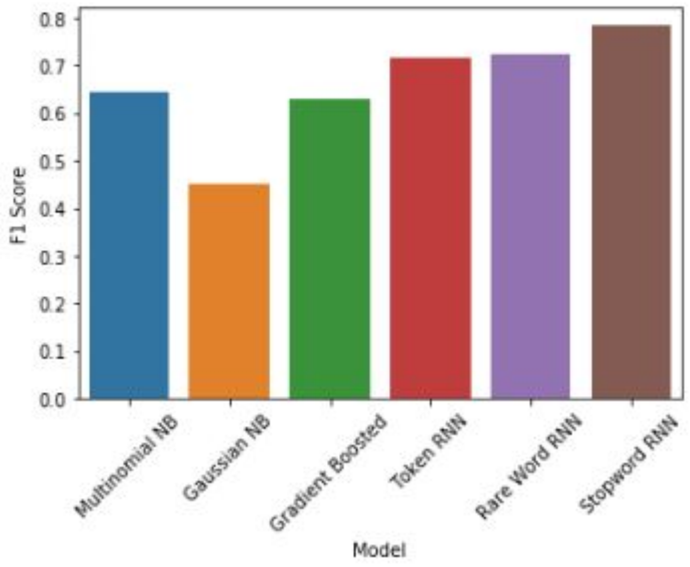


All three models have similar performance, achieving a maximum of just around 91% accuracy.

Overall Performance

On the test set, we see that the traditional Machine Learning models do not perform as well as they did In validation which is a signal of overfit.

The RNN models were trained on accuracy, so cannot Compare their validation and test scores. However, All 3 RNN's outperformed all 3 Machine Learning Models. The RNN with stopwords outperformed all Other models by .04 F1 score, a massive improvement.



Potential Improvements

Other inclusions that could continue to improve the models:

- Including N-grams as features could help both the traditional and RNN models, but would require a massive increase in computational power and training time.
- Including a chi squared distribution to automatically select relevant and effective features.
- RNN models are best used on very large datasets, so collecting more data or increasing the size of the dataset would likely have a large benefit on the RNN models.