Prediction of Airline Sentiments

Bo Liu

1. Introduction

Social media such as Facebook and Twitter play an important role in people's lives today. Users share product and service reviews on social media and consequently, many companies take advantage of the rich data scraped from these platforms to help making marketing decisions, or find opportunities to improve customer experience, which will eventually increase customer retention rate.

In this project, we work on some sentiment analysis on a few major U.S. airlines. The dataset is provided by figure eight website [1], with approximately 15,000 rows and 20 columns. Twitter data was scraped from February of 2015 and labels of neutral, positive and negative tweets are also provided in the dataset. We will leverage some of these features and sentiment labels to train a supervised classification model to predict the probability of reviews falling into 3 classes of sentiments.

Like any NLP problems, the first step is to clean the text field and format the features into table or matrix that machine learning algorithms can process. Steps include removing hashtags and mentions, removing punctuations, stop words and non-English words, expanding contractions, lemmatization and so on. Next we remove some NaN and blank text fields and get the data ready for modeling.

We will implement several algorithms that are frequently used in processing text fields, such as CountVectorizer, TF-IDF Vectorizer, N-Grams and Word2Vec from Gensim. Within each framework, a few classification models such as Random Forests and Logistic Regression will be fit and accuracy on the validation set will be reported and compared. In addition, we also use AFINN Lexicon and TextBlob as out of box algorithms to predict labels, which can be used as benchmark to compare our results. Finally, the best feature model combination will be chosen to apply to the independent test data and test accuracy will be reported.

2. Data Visualization

The raw data is provided by figure eight website [1] in which they collected "Tweets" for 6 major Airline Companies in February 2015, and 20 fields such as name of the user, time zone, number of retweets and so on. In this project, we only leverage a few columns from the original dataset as we believe fields such as user name are not beneficial in the prediction of sentiments.

Number of counts by sentiment and Airline Company are shown in Figure 2.1. A visual inspection shows that majority of the sentiments are negative.

Some sample tweets can be found in Table 2.2 to give the readers some idea of the texts that users input and the sentiment labels provided in the raw dataset. Some words with clear indications of customer sentiments are highlighted in bold. Notice that in some cases, the texts

themselves may not be clear to the readers to judge whether this is a positive or negative statement, however, by using emoticons in the texts greatly helps in identification of sentiments. There are papers that study sentiment via emoticons on social media [2], but we will not explore this option in this project, leaving the opportunity as a potential future work. In fact, after removing punctuations in the texts, all emoticons will be lost.

Figure 2.3 shows total number of retweets by sentiment, and it can be seen that there are significantly more retweets in the negative sentiment group than the other 2 sentiment groups. As a result, including the retweet column may add value in the classifications.

Figure 2.1

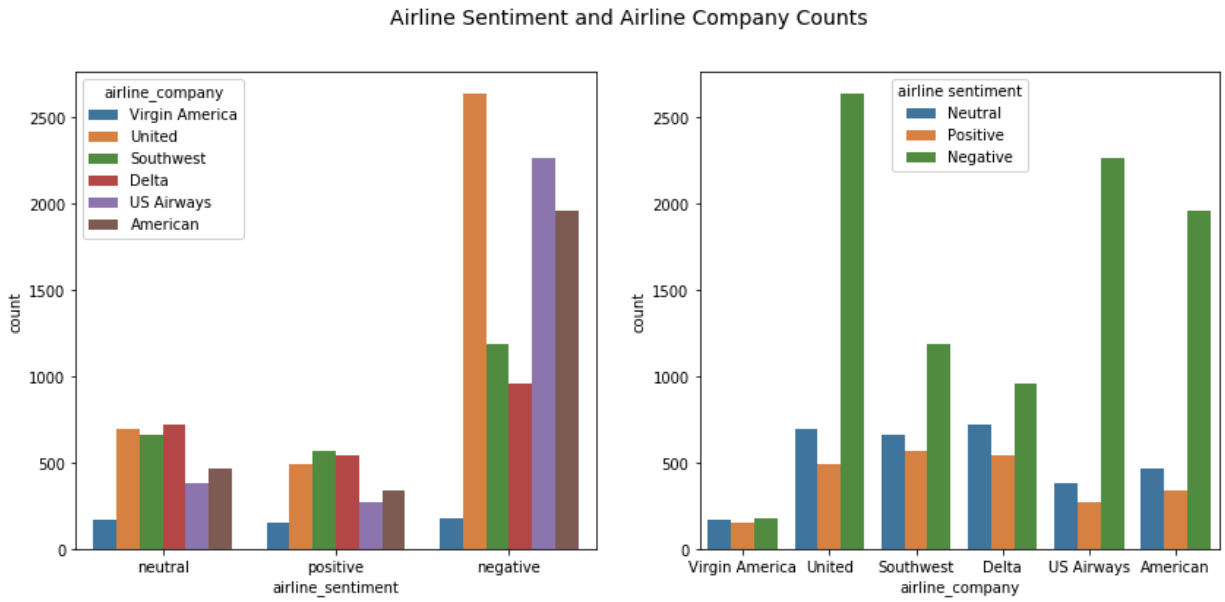
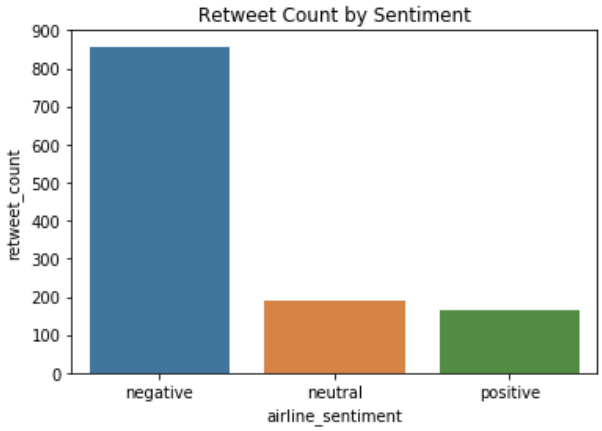


Table 2.2

Sentiment	*Text*	*Airline*
neutral	@VirginAmerica I didn't today... Must mean I need to take another trip!	Virgin America
negative	@VirginAmerica it's really aggressive to blast **obnoxious** "entertainment" in your guests' faces & they have little recourse	Virgin America
negative	@VirginAmerica and it's a really big **bad thing** about it	Virgin America
negative	@VirginAmerica **seriously would** pay $30 a flight for seats that didn't have this playing.	Virgin America
positive	@VirginAmerica yes, nearly every time I fly VX this ‰ЫПear worm‰Ыќ won‰ЫЄt go away **:)**	Virgin America
neutral	@VirginAmerica Really missed a prime opportunity for Men Without Hats parody, there. https://t.co/mWpG7grEZP	Virgin America
positive	@virginamerica Well, I didn't‰Ы_but NOW I DO! **:-D**	Virgin America

Figure 2.3



3. Text Cleaning

As shown in the sample tweets in Table 2.2, the text fields are messy and some of them don't convey a clear message of whether the customer is satisfied or unsatisfied. We take several steps to clean up the texts, such as removing hashtags and mentions, removing stop words, non-English words and punctuations, lemmatization, and so on. After all the steps, the sample texts look like the following (Table 3.1):

Table 3.1

Text
today must mean need take another trip
really aggressive blast obnoxious entertainment guest face little recourse
really big bad thing
seriously would pay flight seat playing
yes nearly every time fly vx ear worm win go away
really miss prime opportunity man without hat parody
well

4. Building ML Models

In this section, we explore a few commonly used NLP feature extraction techniques such as CountVectorizer and N-Grams, TF-IDF Vectorizer from Scikit-learn, and Word2Vec from Gensim. Within each framework, we fit classification models such as Random Forest, Multinomial Naïve Bayes, LightGBM and Logistic regression with regularizations. For each feature model combination, the same pre-defined 5-fold cross validation object is passed to RandomizedSearchCV in Scikit-learn on the training set to select the best combination of hyperparameters, the 5 validation accuracy scores will be recorded and average validation score will be compared among different feature model combinations.

4.1 CountVectorizer

CountVectorizer is one of the Bag-of-Words models which creates a document-term matrix with each row representing a document and each column addressing a token. The values in the matrix are frequencies of tokens / words that it shows up in the document. This count value is also known as weight.

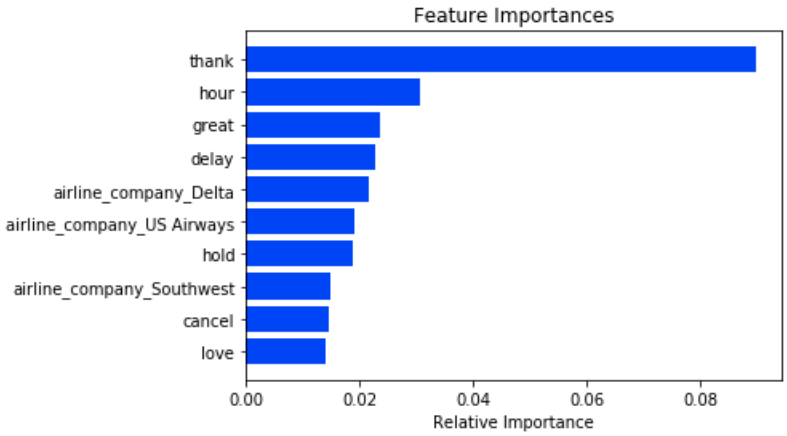
Here we fit 4 models based on the feature matrix created by CountVectorizer and also adding in the retweet count column and airline company columns coded using one-hot encoding.

4.1.1 Random Forest

We asked the model to select 5 combinations of parameters from n = 100, 200, 300, 400 and max depth ranging from 1 to 20 using 5-fold CV. The optimal parameter combinations chosen by randomized search cross validation is n = 400 and max depth = 18. The average training score is 0.643, which is quite close to the average validation score 0.638.

Top 10 features selected by the model on the training set are shown in Figure 4.1.1. Terms with clear indication of sentiments are among the top features, for example, words such as “thank” and “great” are showing positive sentiments, and “delay”, “cancel” are most likely associated with negative sentiments.

Figure 4.1.1



Confusion matrix for the entire training data is shown in Table 4.1.2. We can see that except for negative sentiment, the recall rates are quite low, 0.04 and 0.02 for neutral and positive, respectively, indicating the model has trouble teasing out the neutral and positive sentiments from the vast majority of negative sentiments.

Table 4.1.2

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	109	0	2,332
Positive	5	29	1,833
Negative	9	0	7,323

4.1.2 Naïve Bayes

We use the same pre-defined 5-fold cross validation indices to fit 5 Naïve Bayes models. The average training score is 0.84, which is higher than the average validation score 0.76. This indicates some degrees of overfitting. Later on we will compare the average validation score with other models to select the best performing one. Confusion matrix for the entire training data is shown in Table 4.1.3. The performance shows much improvement compared to Random Forest in the last section.

Table 4.1.3

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	1,424	122	895
Positive	104	1,378	385
Negative	252	126	6,954

4.1.3 LightGBM

The best combination of parameters selected by randomized search cross validation is n = 400, learning rate = 0.5 and max depth = 1. The average training score is 0.79, which is quite close to the average validation score 0.76.

Top 10 features selected by the model on the entire training set are shown in Figure 4.1.4. Similar to Random Forest, terms such as “thank”, “cancel”, “delay” and “bad” appear on the top important words list. Confusion matrix can be found in Table 4.1.5.

Figure 4.1.4

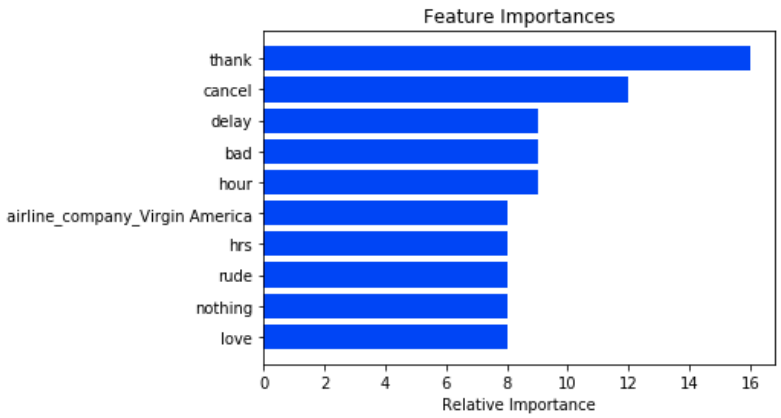


Table 4.1.5

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	1,128	177	1,136
Positive	201	1,199	467
Negative	310	163	6,859

4.1.4 Logistic Regression

In this model, we searched over L1, L2 penalty, and regularization strength parameter C in the list of [0.005, 0.01, 0.1, 1, 10, 100, 200]. It turned out L1 penalty with C = 10 is the best combination. The validation score is by far the highest, 0.788. Confusion matrix on the entire training data is shown in Table 4.1.6.

Table 4.1.6

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	1,680	142	619
Positive	289	1,423	255
Negative	302	114	6,916

4.2 TF-IDF Vectorizer

TF-IDF is another Bag-of-Words model which stands for “term frequency-inverse document frequency”, meaning the weight assigned to each token not only depends on its frequency in a document but also how recurrent that term is in the entire corpora. Words that frequently appears in the entire corpora will be assigned a lower weight as it does not pertain to a specific document, and hence does not have sufficient power to distinguish that document from the rest.

4.2.1 Random Forest

Results of this classifier is very similar to the Random Forest performance within CountVectorizer framework. Best n = 400 and max depth = 16. Average validation score and training score are 0.633 and 0.638 respectively. Top 10 features are almost identical and the confusion matrix is also very similar. Table 4.2.1 shows the results.

Table 4.2.1

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	49	0	2,392
Positive	1	17	1,849
Negative	4	0	7,328

4.2.2 Naïve Bayes

The average training score of this model is 0.74, which is slightly higher than the average validation score 0.70. Overfitting is not as serious as in the CountVectorizer case. The confusion matrix can be found in Table 4.2.2.

Table 4.2.2

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	807	40	1,594
Positive	79	709	1,079
Negative	60	8	7,264

4.2.3 LightGBM

The best combination of parameters selected by randomized search cross validation is n = 100, learning rate = 0.5 and max depth = 3. Average validation score and training score are 0.761 and 0.807 respectively.

Top 10 features selected by the model on the training set are shown in Figure 4.2.3. Similar to Random Forest, terms such as “thank”, “cancel”, “delay” and “bad” appear on the top important words list. Confusion matrix in Table 4.2.4 shows reasonably good recall and precision tradeoff.

Figure 4.2.3

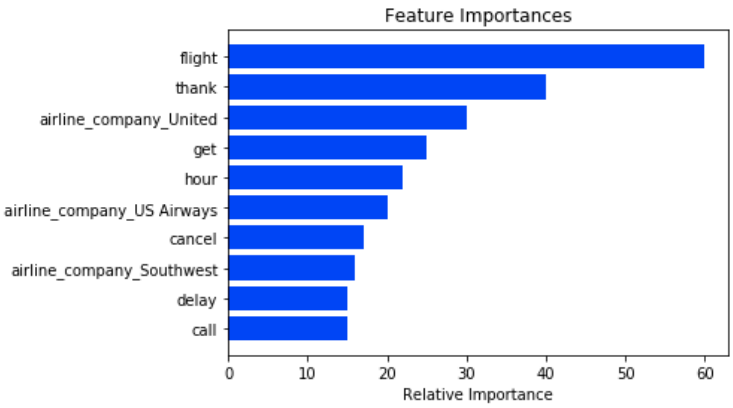


Table 4.2.4

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	1,170	165	1,106
Positive	174	1,245	448
Negative	247	119	6,966

4.2.4 Logistic Regression

L1 penalty with C = 100 is the best combination of hyperparameters. Table 4.2.5 shows the confusion matrix of the entire training data. Its average validation accuracy is 0.764, which is by far the second highest.

Table 4.2.5

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	1,155	121	1,165
Positive	187	1,108	572
Negative	174	88	7,070

4.3 CountVectorizer and N-Grams

An N-Gram is simply a sequence of N words. For example, “stand” and “up” are 1-Grams, whereas “stand up” is a 2-Gram. When combining N-Grams with CountVectorizer, we obtain a document-term matrix whose values are frequencies of each of the up to N-Grams. This dramatically increased the width of the matrix, and due to the long computation times of fitting Random Forest and other machine learning models, we only fit a Multinomial Naïve Bayes model in this case.

4.3.1 Naïve Bayes

The average training and validation score on the 5-Fold cross validation data are 0.90 and 0.75, respectively, showing some degrees of overfitting. Confusion matrix on the entire training set is shown in Table 4.3.1.

Table 4.3.1

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	1,820	45	576
Positive	35	1,524	308
Negative	45	15	7,272

4.4 Word2Vec

As stated in Wikipedia, Word2vec is a group of related models that are used to produce word embeddings and they are often trained using neural networks. Each unique word in the corpus is assigned a corresponding vector and word vectors are positioned in the vector space such that words that share common contexts in the corpus are located close to one another in the space.

In this project, we will not train our own neural network model to create our data specific word embeddings, instead, we will use a pre-trained Word2Vec model in Gensim to generate our

embeddings. For the sake of computation time, we choose 100 dimensions vectors to represent each unique word in our training samples. Then we average the word embedding vectors based on which words appears in one document / row to get the aggregated vector for that document / row.

We only include these 100 features in the model without other columns such as retweet counts and etc.. As these features do not have explicit interpretations, we will not plot the top features like what we did in other frameworks. Also note that Naïve Bayes in Scikit-learn only works on positive feature values, we will not fit this model here as the 100 features contain real numbers with negative values.

The models work in similar ways as previous sections. The confusion matrix presented in the following sections are based on the entire training data for each model.

4.4.1 Random Forest

Average validation score and training score are 0.71 and 0.79 respectively, showing slight overfitting.

Table 4.4.1

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	1,106	72	1,263
Positive	127	825	915
Negative	155	29	7,148

4.4.2 LightGBM

Average validation score and training score are 0.71 and 0.74 respectively.

Table 4.4.2

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	932	156	1,353
Positive	200	778	889
Negative	313	148	6,871

4.4.3 Logistic Regression

Average validation score and training score are 0.729 and 0.731 respectively.

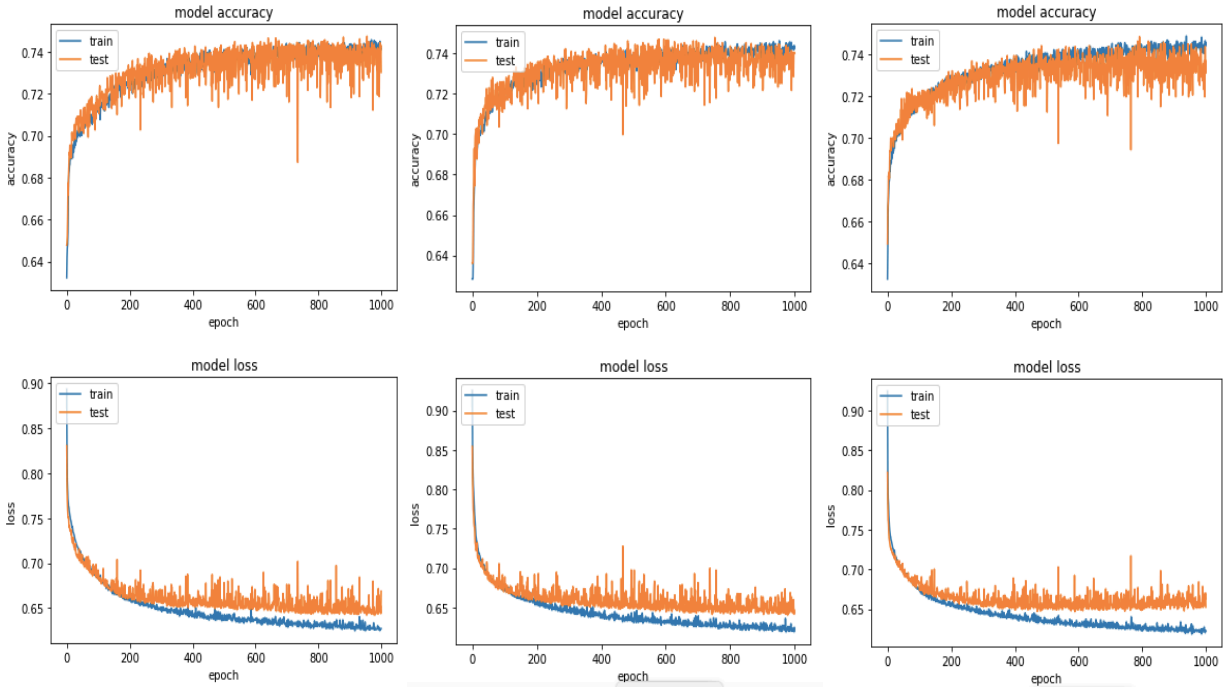
Table 4.4.3

Confusion Matrix	Predicted		
Actual	Neutral	Positive	Negative
Neutral	844	136	1,461
Positive	237	806	824
Negative	309	142	6,881

4.4.4 Neural Network

Out of curiosity we also fit 3 neural network models under the Word2Vec framework. In the first model we used 2 hidden dense layers with rectified linear unit activation function in both layers and 10 nodes and 5 nodes in each layer. In the second model we inserted a dense layer having 8 nodes in between the previous 2 hidden layers to make total 3 hidden dense layers. In the third model we simply changed the nodes to be 12 and 8 in the first model, and all other structures remain the same. The output layer uses SoftMax activation function as this is a classification task. We ran 1000 epochs in each model and the history plot for each model are shown in Figure 4.4.4. Average validation score in each of the 3 models are 0.731, 0.732 and 0.729.

Figure 4.4.4



4.5 Afinn and TextBlob Lexicon

The AFINN Lexicon is a list of English terms manually rated for valence with an integer between -5 (negative) and +5 (positive) by Finn Årup Nielsen between 2009 and 2011. We feed the text field in the training dataset directly to Afinn score function and if the score is negative, we classify the review as negative, neutral if the score turns out to be 0, and positive if the Afinn score is positive. Its overall accuracy on the training set is 0.504 and precision and recall for each of the 3 sentiments are shown in Table 4.5.1.

Table 4.5.1

Afinn	Precision	Recall
Neutral	0.333	0.492
Positive	0.342	0.832
Negative	0.891	0.425

Similarly, we use the sentiment scores obtained from TextBlob to predict 3 sentiments in our training set, and the results are comparable with what we got using Afinn Lexicon. Overall training accuracy is 0.423 and precision and recall are shown in Table 4.5.2.

Table 4.5.2

TextBlob	Precision	Recall
Neutral	0.292	0.612
Positive	0.302	0.651
Negative	0.885	0.303

These 2 algorithms are used as bench mark to get an idea of how well the models we built are performing as a comparison. It looks like all of the models we fit have higher scores than these 2 out of the box models.

4.6 Summary of model accuracy

Table 4.6.1 shows the accuracy for all of the models we trained. Logistic regression within CountVectorizer framework has the highest validation scores of all.

Table 4.6.1

Model	**Accuracy**
Count+Random Forest	0.638488
Count+Naïve Bayes	0.760481
Count+LightGBM	0.764433
Count+Logistic	**0.787973**
TF-IDF+Random Forest	0.633247
TF-IDF+Naïve Bayes	0.701804
TF-IDF+LightGBM	0.760653
TF-IDF+Logistic	0.763918
2-Gram+Naïve Bayes	0.746821
Word2Vec+Random Forest	0.706014
Word2Vec+LightGBM	0.708247
Word2Vec+Logistic	0.728522

4.7 Evaluation on Test set

We evaluated CountVectorizer + Logistic Regression model on the independent test set. The test accuracy is 0.769495 and confusion matrix, classification report are shown in the following Tables 4.7.1 and 4.7.2. Note that 0 represents Neutral, 1 means Positive and 2 means Negative sentiment.

Table 4.7.1

Confusion Matrix	**Predicted**		
Actual	Neutral	Positive	Negative
Neutral	326	52	237
Positive	72	303	110
Negative	155	45	1,611

Table 4.7.2

	precision	recall	f1-score	support
0	0.59	0.53	0.56	615
1	0.76	0.62	0.68	485
2	0.82	0.89	0.85	1811
micro avg	0.77	0.77	0.77	2911
macro avg	0.72	0.68	0.70	2911
weighted avg	0.76	0.77	0.76	2911

We checked Some examples of mis-classified reviews, below are a few reviews that are classified as positive, whereas the labels given in the dataset are negative. Looking at these reviews, many have the word “thank” but actually expressing dissatisfaction of the service. Interestingly, we believe the last review in these examples should be a positive one just as our model had predicted, but the actual label provided in the dataset is negative.

@JetBlue the free wifi makes up for the television not working... It's staticy #ithelpsabit
@united thanks for the link, now finally arrived in Brussels, 9 h after schedule...
@united thank you for dishonoring my upgrade and putting me in a seat I didn't want, all while not even notifying me. Great 1K service _ЩФ_
@USAirways Forget reservations. Thank you to the great leadership at your company, I've Cancelled Flighted my flight. Once again, thank you.

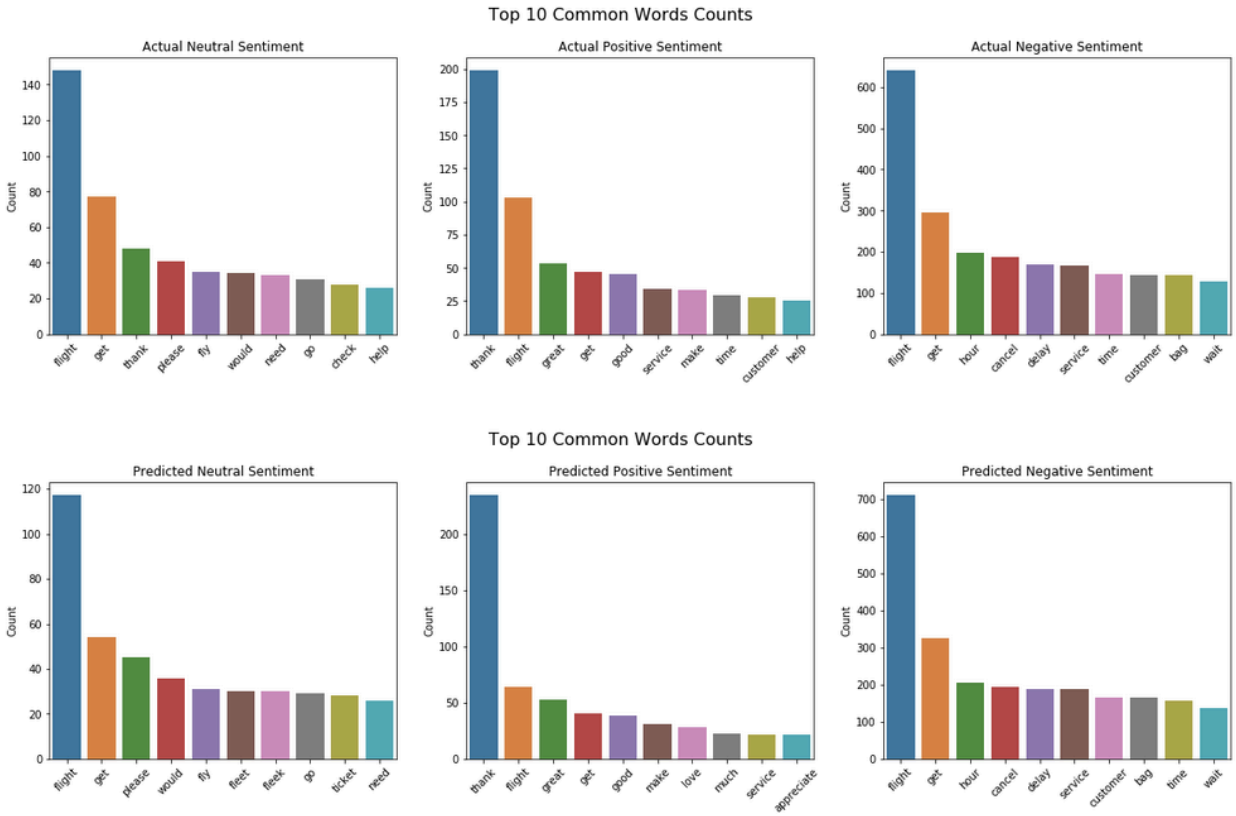
Below are some examples where the actual label is positive and our model classifies them as negative. In the first review below, the customer first expressed his/her thanks to 2 airline associates at baggage claim, but then said he/she has a complaint. Our model classifies this as a negative sentiment because it sees the word “complaint”, however, the label in the raw dataset is positive as it sees the first part of the review. Such reviews with 2 sentiment at the same time are very challenging for machine learning models.

It is interesting in the last review the customer took a United flight and says united suck, so this should be a negative sentiment, rather than positive. Hence, we believe the original label is wrong.

@USAirways please thank Mellie at CAE, Tammy in baggage claim at CLT 4 #excellent customer service 2day, BUT I have a complaint.
@JetBlue Touchdown JFK! Well done pilots of JetBlue Flight 226! #JetBlueRocks
@united Brian at SFO customer service deserves a raise, gave me extra meal voucher and a good joke to cheer me up after flight delay. #FTW
@united you suck. @SouthwestAir you're the best.

Finally, the top 10 most common words in the actual and predicted sentiment are plotted in the following Figure 4.7.3.

Table 4.7.3



Our predictions are pretty close to the actual sentiments. For example, there are 7 words overlapping in the actual and predicted positive sentiment common words, and we can see words such as “thank”, “good”, “appreciate” that best captures positive sentiments are selected by the model.

5. Conclusions and Future work

5.1 Conclusions

In this project we trained some models to classify airline sentiments into 3 classes: neutral, positive and negative, on 6 major U.S. airlines. This is a typical sentiment analysis work that are widely applicable in all businesses, such as restaurant reviews, online shopping reviews and so on. To achieve this, we first cleaned the text field and formatted the features into table or matrix that machine learning algorithms can process. This involves removing hashtags and mentions and punctuations etc. using regular expressions, removing non-English words and stop words and lemmatization using NLTK and Spacy.

We then used a few Bag-of-words approaches to create feature matrix in preparation for fitting classification models. Within each framework, a few classifiers are fit and the best combination of vectorizer and model are chosen. It turned out that CountVectorizer combined with logistic regression with L1 penalty gives the best validation scores 0.788. When applying this best model to the independent test set, we got 0.769 accuracy.

5.2 Future work

We realized that the precision and recall rate for all of the models are not ideal, somewhere between 0.53 to 0.89. There may be room to improve both the precision and recall rates. Specifically, below are some thoughts for future work:

1. Explore emoticons in data processing and modeling [2].
2. There are many mis-spellings in the documents, explore ways to correct mis-spellings.
3. Try a few more combinations of hyperparameters in the Word2Vec model in Gensim, such as tweaking window size and total number of features to keep. It is also a good idea to train Word2Vec model using the airline dataset using neural network (NN), rather than using pre-trained models in Gensim.
4. Building more sophisticated NN models such as Recurrent NN using more curated feature matrices.
5. The model often makes predictions based on a single word in the sentence that it recognizes as positive or negative, but lacks overall understanding of the sentences like what humans can do. In a future refinement, we can train the model to understand the contexts of the reviews by using some topic modelling like Latent Dirichlet Allocation or Probabilistic Latent Semantic Analysis.

Reference

[1] https://www.figure-eight.com/data-for-everyone/
[2] https://arxiv.org/ftp/arxiv/papers/1511/1511.02556.pdf
[3] https://towardsdatascience.com/a-practitioners-guide-to-natural-language-processing-part-i-processing-understanding-text-9f4abfd13e72