Detecting Airline Twitter Sentiment

Concept and Approach:

Labeling a tweet as negative or positive is a pretty easy task when you're a human.
But unfortunately, that's hard to scale when you have thousands (or more) of rows of tweets to classify for a training set.

I set out to put Vader (along with other ML tools) to the test in identifying negative tweets about 6 major US airlines. I trained a Logistic Regression on Vader's scores and tested them on sentiment labels provided in the dataset. I then mapped out the negative tweets to a set of 10 main topics.

Data

The Kaggle dataset I used contains over 14,485 tweets scrapped in 2015 about the 6 major US airlines: United, US Airways, American, Delta, Southwest and Virgin America.

Algorithms

Text pre-processing and cleaning

1. Using RegEx, removed remove links ('http', 'https', 'www', as well as Twitter's link shortner 't.co')
2. Emotion: I removed non-emotional punctuation such as periods, commas, but kept exclamation points.
3. I kept upper case words that also signify strong emotions
4. Removed emojis, digitals and HTML special characters, such as new lines, quotes and ampersands, among others.
5. Removed stop words that included airline names, as well as common industry words such as 'flight' and 'flights'.

Tokenization

1. *Used nltk's MWETokenizer*
2. In addition to removing stop words, I used MWETokenizer to keep colons and parenthesis as one to identify smiley or frowney faces

Lemmatization

1. *Used nltk's WordNetLemmatizer*
2. I also used a stemmer at first, but lost a lot of interpretability, so decided against it

Sentiment Analysis

1. Used vaderSentiment
2. Added key industry words like 'wait', 'refund' and others that Vader was missing to its lexicon, along with an estimate for a sentiment score for each word
3. Arrived at the following overall sentiment scores for each airline:

```
Delta:  {'neg': 0.151, 'neu': 0.643, 'pos': 0.206, 'compound': 1.0}
United:  {'neg': 0.218, 'neu': 0.614, 'pos': 0.168, 'compound': -1.0
}
Southwest:  {'neg': 0.179, 'neu': 0.616, 'pos': 0.205, 'compound': 1
.0}
US Airways:  {'neg': 0.248, 'neu': 0.608, 'pos': 0.144, 'compound':
-1.0}
Virgin America:  {'neg': 0.114, 'neu': 0.695, 'pos': 0.191, 'compoun
d': 0.9999}
American:  {'neg': 0.225, 'neu': 0.619, 'pos': 0.156, 'compound': -1
.0}
```

United, US Airways and **American** have an overall sentiment score of -1, or negative. **Delta**, **Southwest**, **Virgin America** have an overall sentiment score of 1, or positive.

4. Then, used standard score thresholds to map out the sentiment to each tweet in the dataset:

positive sentiment for compound scores >= 0.05
neutral sentiment for compound scores > -0.05 and < 0.05
negative sentiment for compound scores <= -0.05

Logistic Regression

1. Split tweets into 70% train and 30% test with X train being the converted 0,1,2 labels from Vader and test being, and y test being the original sentiment labels from the dataset
2. Set up a GridSearch for the Logistic Regression across a few parameters.
3. CountVectorizer with stop_words='english', max_df = 0.90, min_df = 0.01 resulted in:

Best model:

LogisticRegression(C=10, max_iter=10000, multi_class='multinomial')

```
              precision    recall  f1-score   support

           0       0.35      0.54      0.42       911
           1       0.88      0.51      0.64      2726
           2       0.36      0.69      0.47       708

    accuracy                           0.54      4345
   macro avg       0.53      0.58      0.51      4345
weighted avg       0.68      0.54      0.57      4345
```

4. TfidfVectorizer with stop_words='english', max_df = 0.90, min_df = 0.01 resulted in:

   Best model:

   LogisticRegression(C=1, class_weight='balanced', max_iter=10000,
   multi_class='multinomial')
)

```
              precision    recall  f1-score   support

           0       0.33      0.66      0.44       911
           1       0.89      0.45      0.60      2726
           2       0.41      0.65      0.50       708

    accuracy                           0.53      4345
   macro avg       0.54      0.59      0.51      4345
weighted avg       0.69      0.53      0.55      4345
```

5. I went with the TfidfVectorizer for the topic modeling:
   - A slightly higher negative precision of 89% - people are known for venting on twitter, so negative precision is my main comparison metric since I want to capture the most true negative sentiment
   - More of true Neutral sentiment is captured
   - Less of Negative sentiment is mislabeled as Positive.

```
Delta:  {'neg': 0.151, 'neu': 0.643, 'pos': 0.206, 'compound': 1.0}
```

```
United:  {'neg': 0.218, 'neu': 0.614, 'pos': 0.168, 'compound': -1.0
}
Southwest:  {'neg': 0.179, 'neu': 0.616, 'pos': 0.205, 'compound': 1
.0}
US Airways:  {'neg': 0.248, 'neu': 0.608, 'pos': 0.144, 'compound':
-1.0}
Virgin America:  {'neg': 0.114, 'neu': 0.695, 'pos': 0.191, 'compoun
d': 0.9999}
American:  {'neg': 0.225, 'neu': 0.619, 'pos': 0.156, 'compound': -1
.0}
```

Topic Modeling

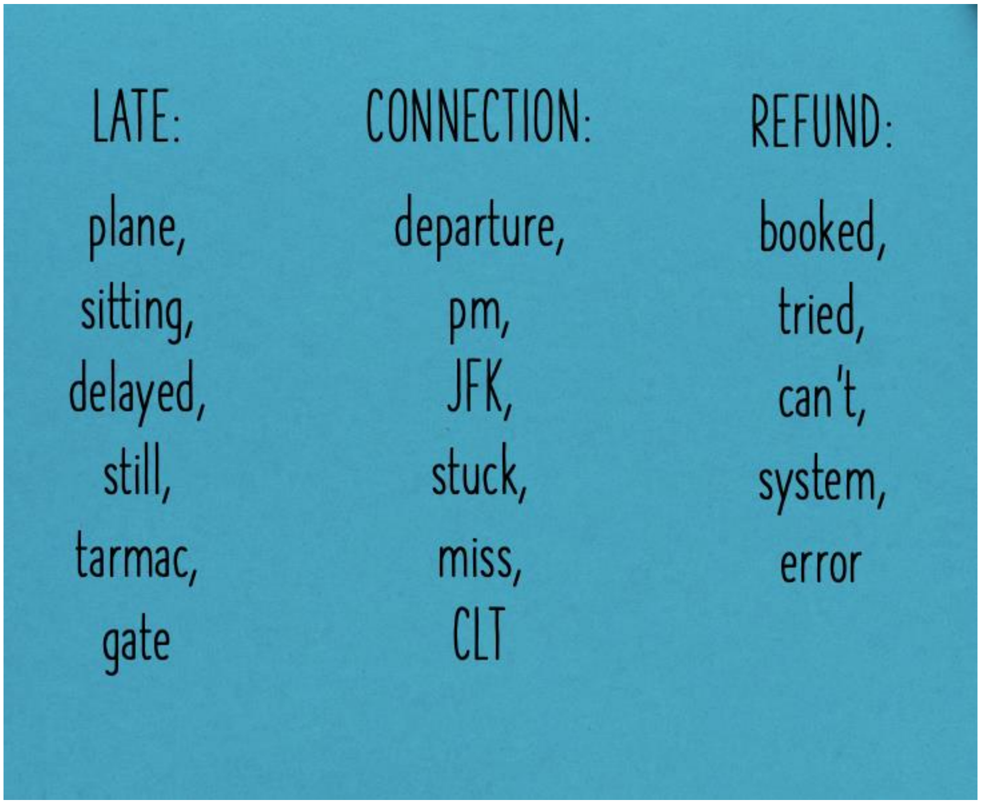
1. I tried NMF and CorEx, and went with CorEx as the final algorithm as it provided a more coherent and interpretable list of topics
2. CorEx topics and my corresponding interpretations after cross-reference with the dataset:

1. Issues with booking, changing flights/reservations
2. Being stuck in the airport or on the plane
3. Wrong information, lack of clarity from support
4. Refunds for weather delays/flight cancelation
5. Gate agents, waiting at the gate
6. Technical issues on flight/with plane
7. Late arrivals/luggage claim
8. Flight delays at the airport, hotel/food vouchers
9. Problems reaching customer support
10. Issues with connecting flights when flying home

1: problems, booking, online, number, change, plane, phone, website, reservation, sitting
2: airport, going, pay, people, seat, stuck
3: leave, line, service, tell, terrible, thanks, told, wrong
4: disappointed, getting, passenger, poor, refund, sorry, travel, weather, work
5: agent, answer, bad, baggage, boarding, day, flt, horrible, hour, know
6: experience, got, issue, like, new, really, ridiculous, right, said, seriously
7: check, email, fail, finally, let, luggage, make, pm, trip
8: ago, jfk, morning, pilot, response, voucher, way
9: called, guy, missed, stop, today, unacceptable, update
10: customer, dfw, flying, home, hotel, left, minute, problem, rebook, rude

BONUS: Pulling Up Similar Sentiment Words at random

1. I used gensim's Word2Vec to be able to pull a list of similar sentiment words for key words at random. A few examples:



Tools

- **Text pre-processing and EDA**: SQL, Pandas, Numpy, SQL, NLTK: stopwords, , MWETokenizer, WordNetLemmatizer, RegEx

- **Modeling:** vaderSentiment, sklearn: CountVectorizer, TfidfVectorizer, NMF; Logistic Regression, CorEx, gensim: Word2Vec

- **Visualizations:** Matplotlib, Seaborn