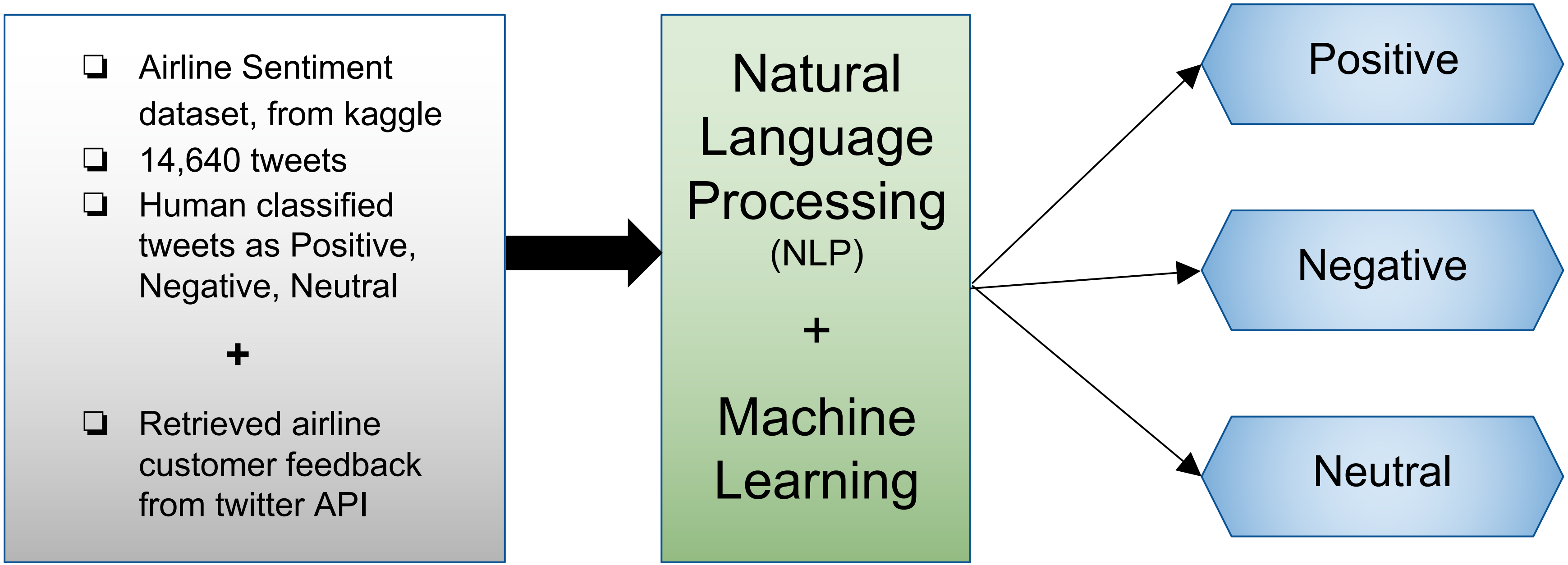

Airline Tweets
Sentiment Analysis
American
Alamy

Can machine learning help airline companies gauge customer satisfaction?

😊😞

By Sangita Gupta

Airline Sentiment Classification



Conversion of tweets into airline sentiment

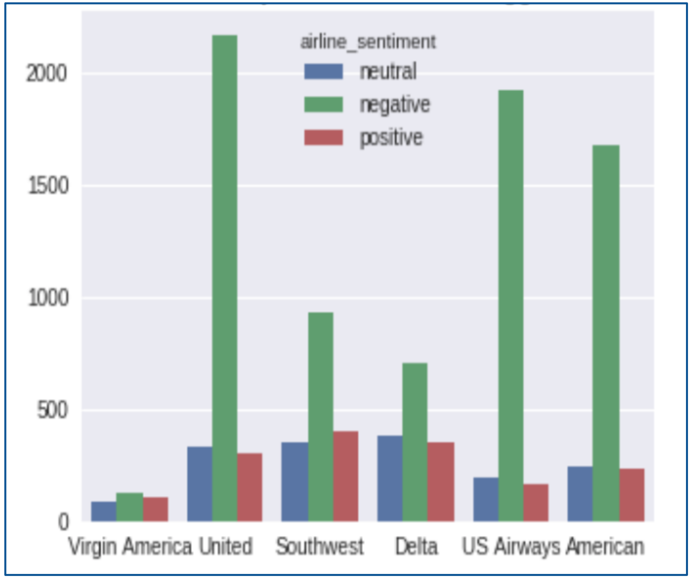
I ❤️ flying @VirginAmerica. ☺👍

👍👍✈️✈️💗 When are you guys going to start flying to Paris? @VirginAmerica: @LizaUtter You're welcome.”

@VirginAmerica awaiting my return phone call, just would prefer to use your online self-service option :(

@VirginAmerica plz help me win my bid upgrade for my flight 2/27 LAX--->SEA!!! 🍷👍💺✈️

NLP
+
Machine
Learning



Natural Language Processing

A field of data science that can analyze text and make predictions using that text as data

Raw Tweets

Clean Tweets: remove urls, @users, numbers, punctuation

Data dimensionality reduction methods

Remove Stopwords: I, you, we, so, to, ...

Apply Lemmatization:

am, are, is ⇒ be
car, cars, car's, cars' ⇒ car

OR

Apply Stemming:

caresses, ponies, cats ⇒ caress, poni, cat

Processed Tweets

- **Raw Tweet:**
  - @VirginAmerica Applied for Status Match on Feb 1. Got confirmation email same day. Still no news though. You guys have dropped ball Late Flightly 😥
- **Cleaned and Emoji Encoded Tweet:**
  - applied for status match on feb got confirmation email same day still no news though you guys have dropped ball late flightly emoji_34
- **Stop Words Removed Tweet:**
  - applied status match feb got confirmation email day news guys dropped ball late flightly emoji_34
- **Stemming Applied to Tweet:**
  - appli statu match feb got confirm email day news guy drop ball late flightli emoji_34
- **Lemmatization Applied to Tweet:**
  - apply status match feb get confirmation email day news guy drop ball late flightly emoji_34

Handling Emojis

Extract emojis from tweets to see how they relate to the classified sentiment.

emojis	❤ ☺ 👍	😡	😢	💜 ✈	🍷 👍 💺 ✈	💕 💕	😁	❤	👏	😂 💗	🍸	😒	👎	👍 👍 ✈ ✈ 💗	😊 😀 😃 😁	😎
airline_sentiment	positive	negative	negative	positive	neutral	positive	negative	positive	positive	positive	positive	negative	negative	neutral	neutral	positive

They look like good sentiment predictors. They are symbols so we have to encode them.

Users tend to group emojis together.

`'i ❤️ flying ☺️👍'`

So first separate the emojis into individual symbols.

`'i ❤️ flying ☺️ 👍'`

Now that they are separate, let's encode them as features...

Encode Emojis

```
{'❤': 'EMOJI_1', '☺': 'EMOJI_2', '👍': 'EMOJI_3', '😡': 'EMOJI_4', '😢': 'EMOJI_5', '💜': 'EMOJI_6', '✈': 'EMOJI_7', '🍷': 'EMOJI_8', '💺': 'EMOJI_9', '😊': 'EMOJI_10', '😍': 'EMOJI_11', '👌': 'EMOJI_12', '💕': 'EMOJI_13', '🌞': 'EMOJI_14', '😃': 'EMOJI_15', '😩': 'EMOJI_16', '😭': 'EMOJI_17', '😎': 'EMOJI_18', '🙊': 'EMOJI_19', '😁': 'EMOJI_20', '❄': 'EMOJI_21', '👏': 'EMOJI_22', '😂': 'EMOJI_23', '💗': 'EMOJI_24', '🍸': 'EMOJI_25', '😒': 'EMOJI_26', '👎': 'EMOJI_27', '😀': 'EMOJI_28', '😄': 'EMOJI_29', '😘': 'EMOJI_30', '👸': 'EMOJI_31', '🎀': 'EMOJI_32', '🌏': 'EMOJI_33', '😥': 'EMOJI_34', '😉': 'EMOJI_35', '✨': 'EMOJI_36', '😱': 'EMOJI_37', '🎉': 'EMOJI_38', '🙌': 'EMOJI_39', '💤': 'EMOJI_40', '😞': 'EMOJI_41', '♥': 'EMOJI_42', '👋': 'EMOJI_43', '✌': 'EMOJI_44', '🙏': 'EMOJI_45', '😈': 'EMOJI_46', '😔': 'EMOJI_47', '🙅': 'EMOJI_48', '🆖': 'EMOJI_49', '💩': 'EMOJI_50', '✔': 'EMOJI_51', '🌴': 'EMOJI_52', '✅': 'EMOJI_53', '❌': 'EMOJI_54', '👠': 'EMOJI_55', '😜': 'EMOJI_56', '😻': 'EMOJI_57', '😕': 'EMOJI_58', '👿': 'EMOJI_59', '😤': 'EMOJI_60', '💪': 'EMOJI_61', '😫': 'EMOJI_62', '💔': 'EMOJI_63', '😪': 'EMOJI_64', '😣': 'EMOJI_65', '😬': 'EMOJI_66', '💁': 'EMOJI_67', '😋': 'EMOJI_68', '😏': 'EMOJI_69', '😖': 'EMOJI_70', '🌟': 'EMOJI_71', '📱': 'EMOJI_72', '🍻': 'EMOJI_73', '💖': 'EMOJI_74', '😅': 'EMOJI_75', '💝': 'EMOJI_76', '🚫': 'EMOJI_77', '↔': 'EMOJI_78', '😷': 'EMOJI_79', '⭐': 'EMOJI_80', '🎵': 'EMOJI_81', '❗': 'EMOJI_82', '🐴': 'EMOJI_83', '😆': 'EMOJI_84', '😑': 'EMOJI_85', '🐩': 'EMOJI_86', '⤴': 'EMOJI_87', '☀': 'EMOJI_88', '👊': 'EMOJI_89', '💯': 'EMOJI_90', '😠': 'EMOJI_91', '☕': 'EMOJI_92', '📲': 'EMOJI_93', '👺': 'EMOJI_94', '🙈': 'EMOJI_95', '💘': 'EMOJI_96', '💙': 'EMOJI_97', '👉': 'EMOJI_98', '🚪': 'EMOJI_99', '😳': 'EMOJI_100', '😵': 'EMOJI_101', '🚶': 'EMOJI_102', '🔵': 'EMOJI_103', '😐': 'EMOJI_104', '👀': 'EMOJI_105', '🍅': 'EMOJI_106', '🆘': 'EMOJI_107', '⛄': 'EMOJI_108', '😓': 'EMOJI_109', '🎲': 'EMOJI_110', '⌚': 'EMOJI_111', '🐳': 'EMOJI_112', '⤵': 'EMOJI_113', '😮': 'EMOJI_114', '😲': 'EMOJI_115', '😦': 'EMOJI_116', '➡': 'EMOJI_117', '🌺': 'EMOJI_118', '🌊': 'EMOJI_119', '❓': 'EMOJI_120', '😇': 'EMOJI_121', '🤣': 'EMOJI_122', '🏻': 'EMOJI_123', '🏼': 'EMOJI_124', '🌲': 'EMOJI_125', '🎁': 'EMOJI_126', '🎄': 'EMOJI_127', '🎅': 'EMOJI_128', '▶': 'EMOJI_129', '🌎': 'EMOJI_130', '🛬': 'EMOJI_131', '👇': 'EMOJI_132', '😯': 'EMOJI_133', '🤦': 'EMOJI_134', '♀': 'EMOJI_135', '☝': 'EMOJI_136', '®': 'EMOJI_137', '🔥': 'EMOJI_138', '💃': 'EMOJI_139', '🤞': 'EMOJI_140', '🏽': 'EMOJI_141', '🛬': 'EMOJI_142', '🛫': 'EMOJI_143', '🗽': 'EMOJI_144', '🏾': 'EMOJI_145', '🎊': 'EMOJI_146', '🥂': 'EMOJI_147', '🙃': 'EMOJI_148', '🤤': 'EMOJI_149', '👩': 'EMOJI_150', '🏖': 'EMOJI_151', '💛': 'EMOJI_152', '☃': 'EMOJI_153', '🤾': 'EMOJI_154', '♂': 'EMOJI_155', '🏀': 'EMOJI_156', '🙂': 'EMOJI_157', '🎼': 'EMOJI_158', '🤷': 'EMOJI_159', '🦋': 'EMOJI_160', '😴': 'EMOJI_161', '🥃': 'EMOJI_162', '🤗': 'EMOJI_163', '✡': 'EMOJI_164', '🤔': 'EMOJI_165', '🌍': 'EMOJI_166', '☁': 'EMOJI_167', '👙': 'EMOJI_168', '🤓': 'EMOJI_169', '🎶': 'EMOJI_170'}
```

Transform text into numerical data

Divide tweets into words and create a document term matrix. The columns are unique words in the corpus of tweets. The rows are tweets represented by a sequence of ‘0s’ and ‘1s’, depending on whether the word is present or not.

Tokenize Text

Corpus of tweets are separated into words

Corpus is now like a “bag of words”

Document Term Matrix

The “bag of words” is used to make a table

Columns are the unique words in the corpus

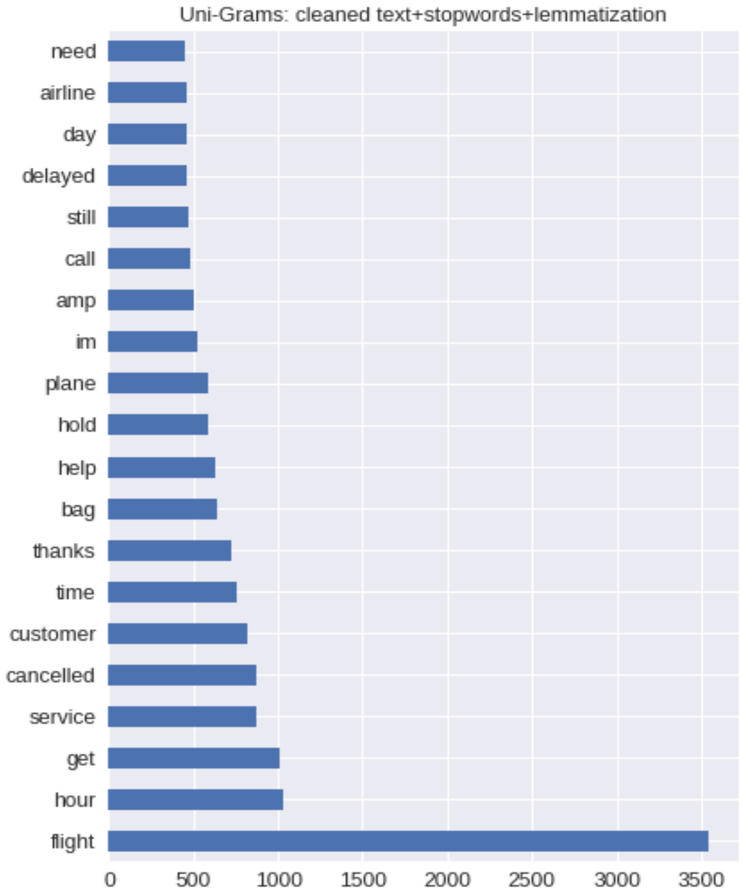
Rows are the individual tweets represented in numerical form

Vectorized Text

Each tweet is now represented by a series of ‘0s’ and ‘1s’ in the document term matrix

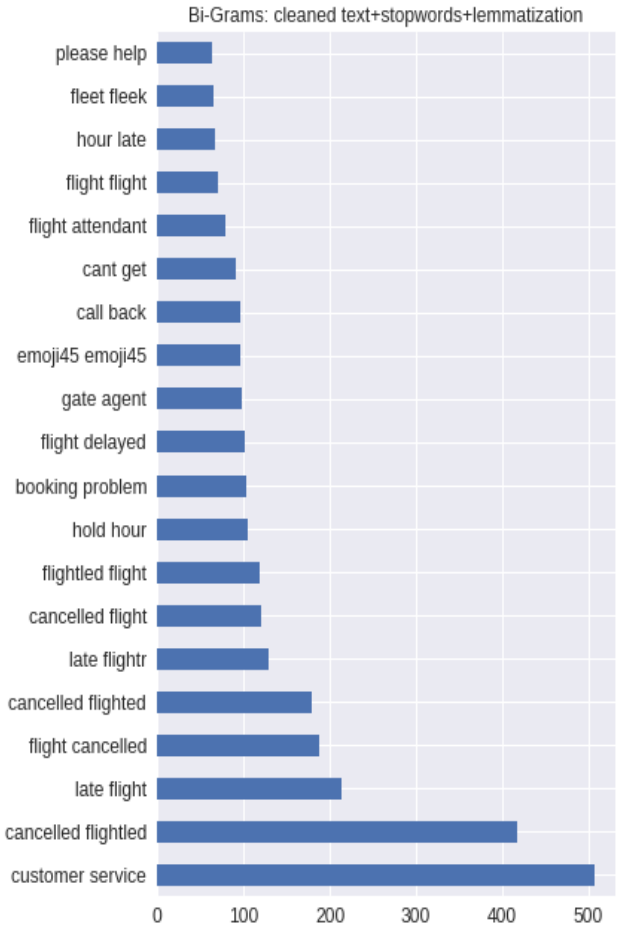
If the word exists in the tweet its value is ‘1’ if not, its ‘0’

Word Features: uni-grams



- Words in Natural Language Processing become features of the text dataset.
- These features are known as predictors in machine learning and are used to form predictions about an output label, also referred to as target.
- In NLP, single words or multiple successive words may be used as features, also referred to as n-grams.
  - An n-gram is a contiguous sequence of n words
- The graph to the left shows the top 20 “1-grams”, also known as “uni-grams”, for tweets in the dataset.
  - flight, hour, get, service cancelled, customer, time, thanks, bag, help, hold, plane, im, amp, call, still, delayed, day, airline, need

Word Features: bi-grams and tri-grams

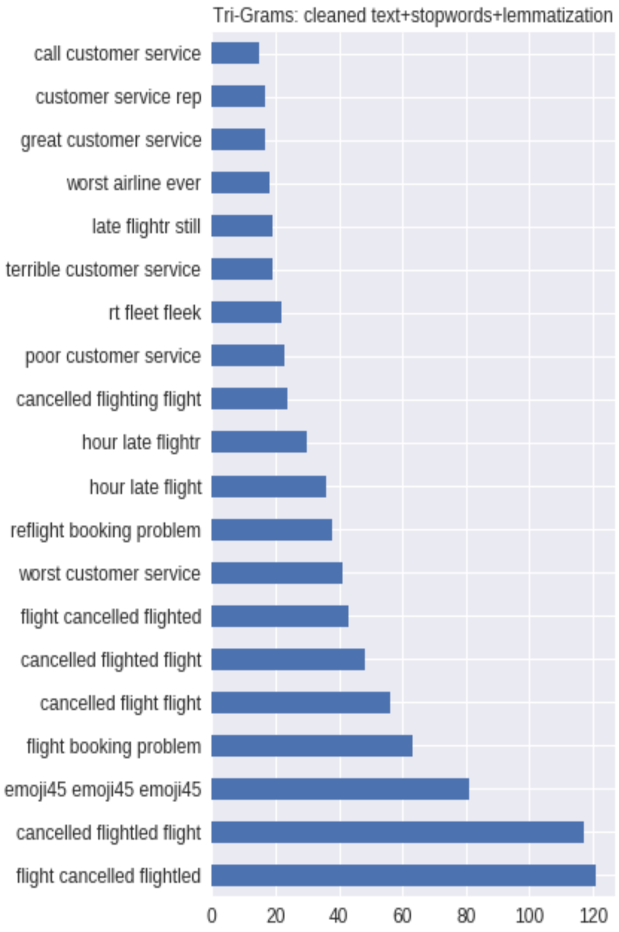


Top bi-grams:

- Customer service
- late flight
- flight cancelled
- hold hour
- booking problem
- flight delayed
- gate agent
- emoji45 emoji45
- hour late
- please help

Top tri-grams:

- Flight cancelled flightled
- cancelled flightled flight
- emoji45 emoji45 emoji_45
- flight booking problems
- cancelled flight flight
- worst customer service
- great customer service
- reflight booking problem
- hour late flight
- call customer service



Word Frequency

Model Selection

Naive Bayes

Learning Mechanism

- Naive Bayes models the joint distribution (X,Y) and then predicts the probability P(Y|X)
  - X is set of input features
  - Y is the output labels
- It is thus called a generative model.

Model Assumptions

- Assumes that every word in a sentence is independent from the other words.
- So for Naive Bayes the following sentences would all be the same.

"this was a fun party"

"this party was fun"

"party fun was this"

Logistic Regression

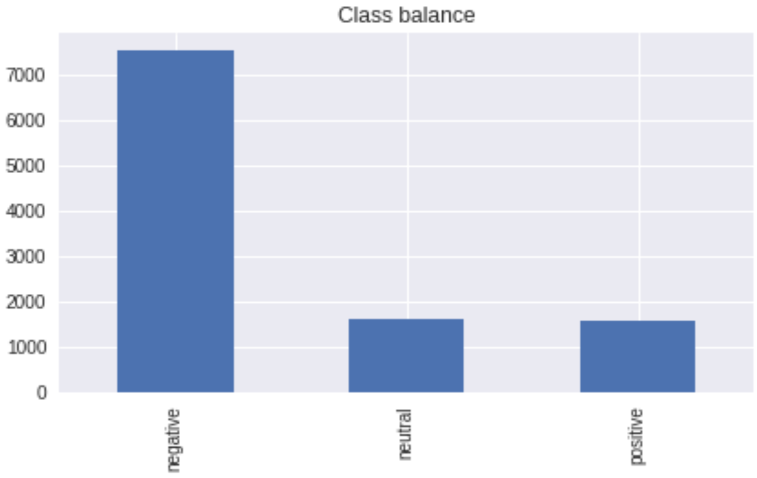
Learning Mechanisms

- Logistic Regression directly models the P(Y|X) from learning the input to the output mapping, by minimizing the prediction error.
- It is thus called a discriminative model, since it discriminates based on the error.

Model Assumptions

- It assumes non-collinearity of features.
- It splits the feature space linearly, so it deals fairly well even if some features are correlated.

Modeling



- Data is highly unbalanced.

Baseline Accuracy = 0.70

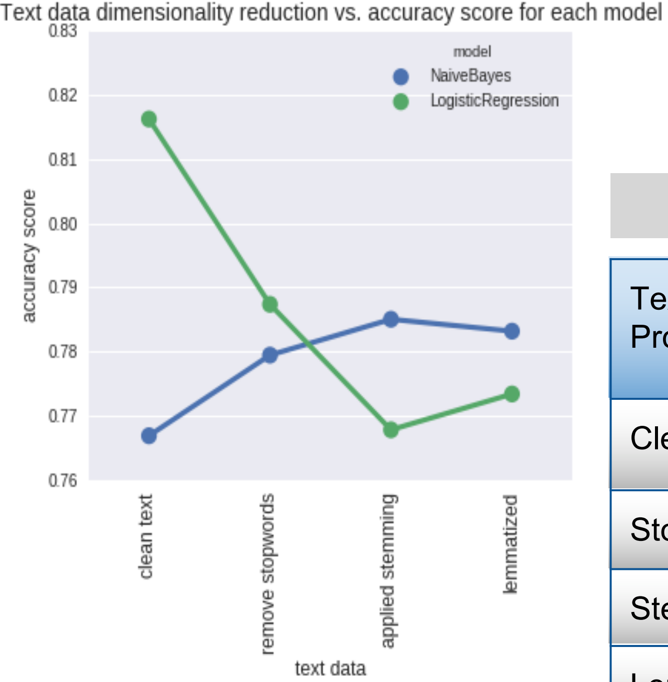
- We will be right 70% of the time if we always guess the most common class

Insights we want to gain

- ❑ What effects do ***emojis*** have on model predictions?
- ❑ What effects do different ***dimensionality reductions methods*** have on model predictions?
- ❑ What effects ***tuning*** the model's hyper parameters have on model predictions?

Model Accuracy Effects: using emojis as sentiment predictors

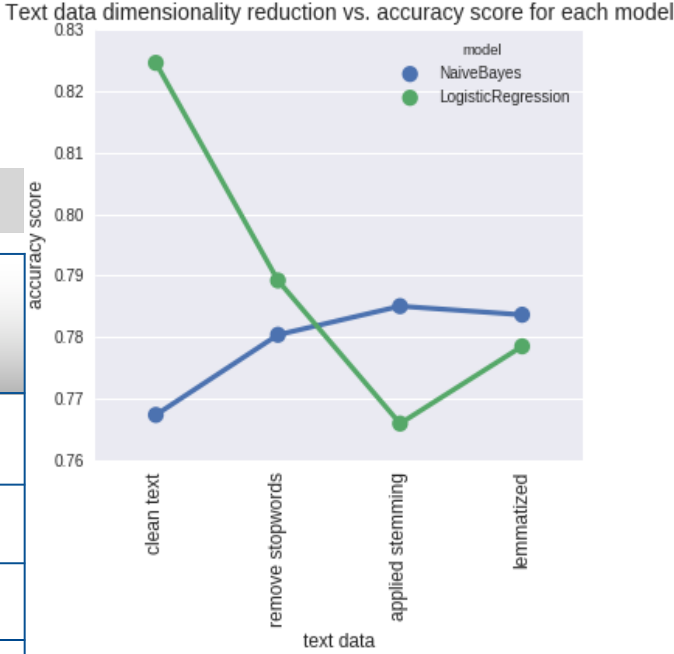
Emoji Not Encoded



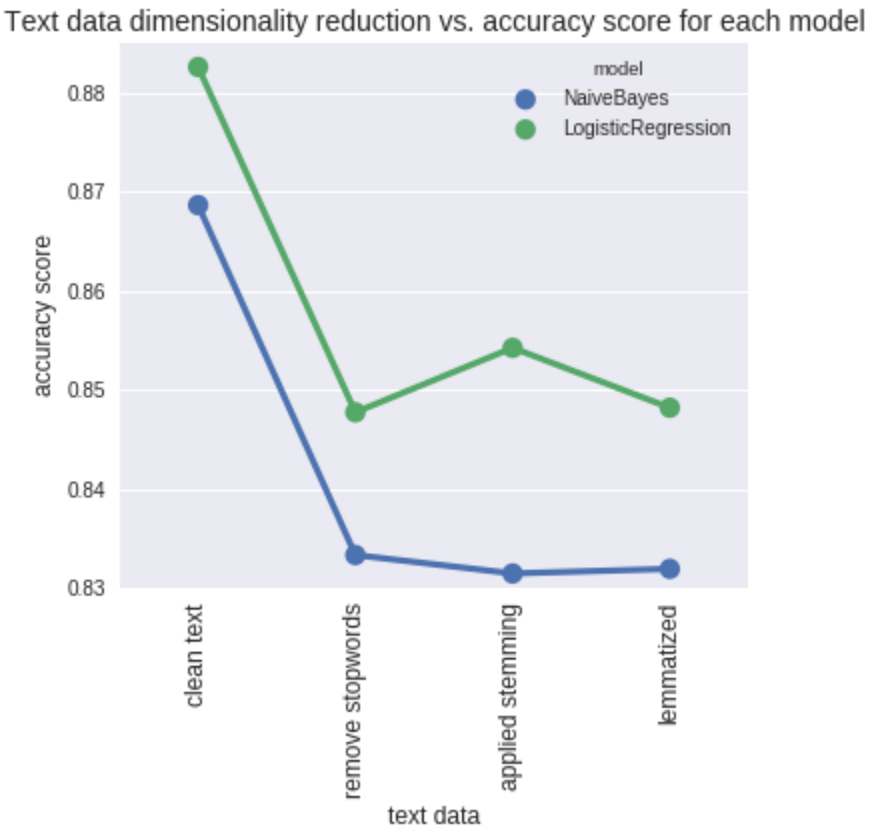
Model accuracy effects

Text Processing	Naive Bayes	Logistic Regression
Clean Text	+ 0.047%	+ 0.836%
Stop Words	+ 0.093%	+ 0.186%
Stemming	+ 0%	- 0.186%
Lemmatization	+ 0.046%	+ 0.51%

Emoji Encoded



Model Accuracy Effects: tuning model hyper-parameters



Parameters Tuned:

- Countvectorizer
  - min_df = 5, max_df = 0.95, ngram_range = (1,2)
- Logistic Regression model
  - C = 1.0
- Naive Bayes model
  - Alpha = 0 .10000000000000001

Model Accuracy Effects using tuned models

Text Processing	Naive Bayes	Logistic Regression
Clean Text	+ 10.12%	+ 5.803%
Stop Words	+ 5.292%	+ 5.850%
Stemming	+ 4.642%	+ 8.82%
Lemmatization	+ 4.828%	+ 6.9642%

Best tuned model results

Accuracy of tuned models

Text Processing (Emoji Encoded)	Naive Bayes	Logistic Regression
Clean text	86.7%	88.3%
Stop Words	83.3%	84.8%
Stemming	83.1%	85.4%
Lemmatization	83.1%	84.8%

No data dimensionality reduction:

- ❑ Logistic Regression achieves an accuracy of 88.3%.
- ❑ Naïve Bayes achieves an accuracy of 86.7%.

With data dimensionality reduction:

- ❑ Logistic Regression achieves an accuracy of about 85%.
- ❑ Naïve Bayes achieves an accuracy of about 83%.

Model performance compared to baseline accuracy:

- ❑ Baseline accuracy was 70%, if always guessing the most common class, which in this case is the negative class.
- ❑ Logistic Regression achieves 18% better accuracy without dimensionality reduction and 15% better with.
- ❑ Naive Bayes achieves 16.7% better accuracy without dimensionality reduction and 13% better with.
- ❑ The machine learning models outperform the baseline accuracy by 13 to 18 percent.

Conclusions about Models

- **Using encoded emojis as sentiment predictors:**
  - Logistic Regression model accuracy seems to improve.
    - However, when using stemming accuracy actually decreases.
  - Naive Bayes model accuracy seems to marginally show improvement. For the most part it seem to be unaffected.
- **Tuning model hyper-parameters:**
  - Logistic Regression and Naive Bayes have significant accuracy gains by tuning hyper-parameters.
  - Naive Bayes has the highest accuracy gain when using data that is only cleaned and emoji encoded.
  - Logistic Regression has the highest accuracy gain when using data with stemming applied.
- **Applying data dimensionality reduction methods:**
  - Logistic Regression model accuracy seems to be about 3% worse than without dimensionality reduction.
  - Naïve Bayes model accuracy seems to be about 4% worse than without dimensionality reduction.

Classify airline tweets from Twitter

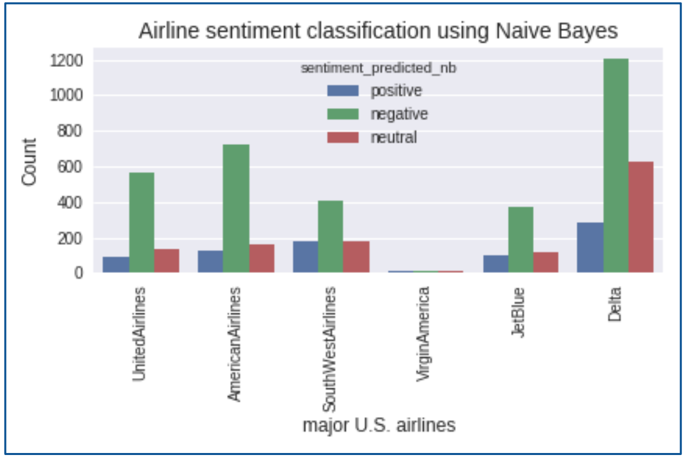
Airline customer tweets

TWITTER API

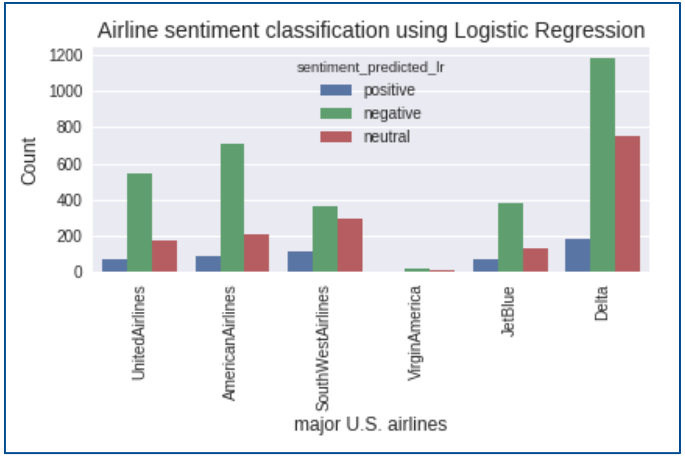
Pull tweets for major US airlines using hashtags

Process tweets using Natural Language Processing Techniques

Classify using Naive Bayes



Classify using Logistic Regression



Airline Sentiment Classification

Tweet: Kudos to the #unitedairlines staff for helping us with our crazy reservations. Nice send-off from CHS

Sentiment Prediction NB: positive **Sentiment Prediction LR:** positive

Tweet: @united YESSSSSSSSSSSSSSSSSSSSSSSSSS! I don't know how today can get any better, this is amazing! #ThankYou #UnitedAirlines #FlyingTheTahitiSkies

Sentiment Prediction NB: positive **Sentiment Prediction LR:** positive

Tweet: A peek inside Classified, where CEOs and celebrities dine in a hidden restaurant at Newark Liberty International Airport. https://t.co/bpmhavKTSh #TableReady #speakeasy #UnitedAirlines #exclusive

Sentiment Prediction NB: neutral **Sentiment Prediction LR:** neutral

Tweet: @united That's too bad. 😞 #YYJ would love some more #UnitedAirlines service choices✈️

Sentiment Prediction NB: positive **Sentiment Prediction LR:** negative

Tweet: Montreal, Canada to Phoenix, Arizona for only $271 CAD roundtrip with United. #UnitedAirlines #Montreal https://t.co/3PErNhmXZ0

Sentiment Prediction NB: negative **Sentiment Prediction LR:** neutral

Tweet: I am really pissed off with @united   Luggage missing since Jan 3 and no one from the airline has made any attempt to explain the problem #unitedairlines does not care

Sentiment Prediction NB: negative **Sentiment Prediction LR:** negative

Next Step: Dealing with class imbalance

Challenges in predicting sentiment from tweets

- Machines learn best with precise, unambiguous and structured data. Tweets however, are not generally precise, often ambiguous and the language used is far from structured.
- Highly imbalanced dataset in favor of the negative class.

To improve model prediction accuracy, I will explore techniques of dealing with class imbalance.

Next Step - Dealing with Class imbalance in the dataset

- Resample the most frequent class to have a similar corpus size as the other classes.
- Tune the penalty hyper-parameter of Logistic Regression.
- Evaluate other algorithms which deal well with imbalanced datasets.

Questions?

To test out my airline sentiment classifiers go to:

http://34.212.204.117:5000/predict-sentiment-interface