Airline Tweets Sentiment Analysis

😔 ☺️

Alamy

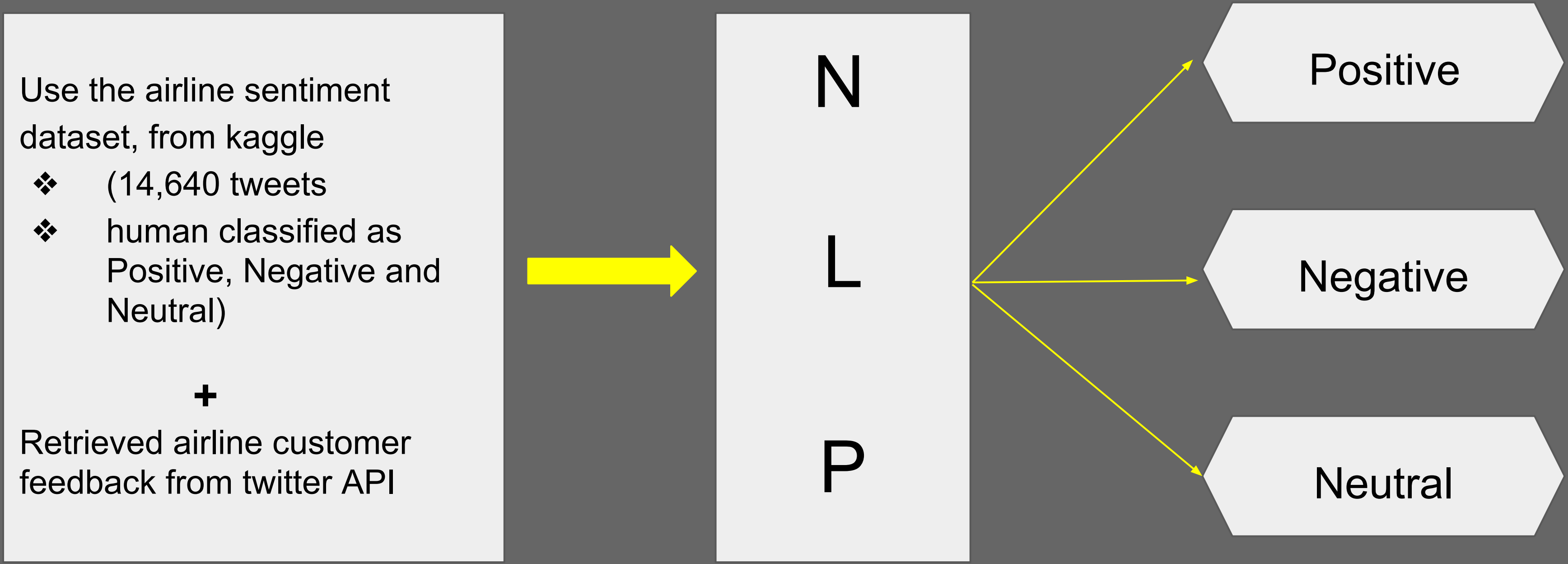
Can machine learning help airline companies gauge customer satisfaction?

😊 ☹️

By Sangita Gupta

Airline Sentiment Classification

Goal



Use the airline sentiment dataset, from kaggle

- (14,640 tweets
- human classified as Positive, Negative and Neutral)

+

Retrieved airline customer feedback from twitter API

N

L

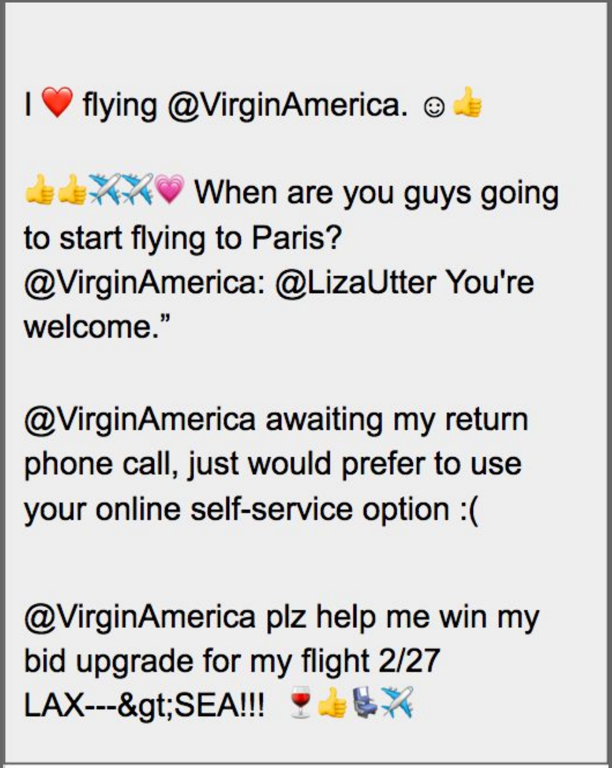
P

Positive

Negative

Neutral

Conversion of tweets into airline sentiment



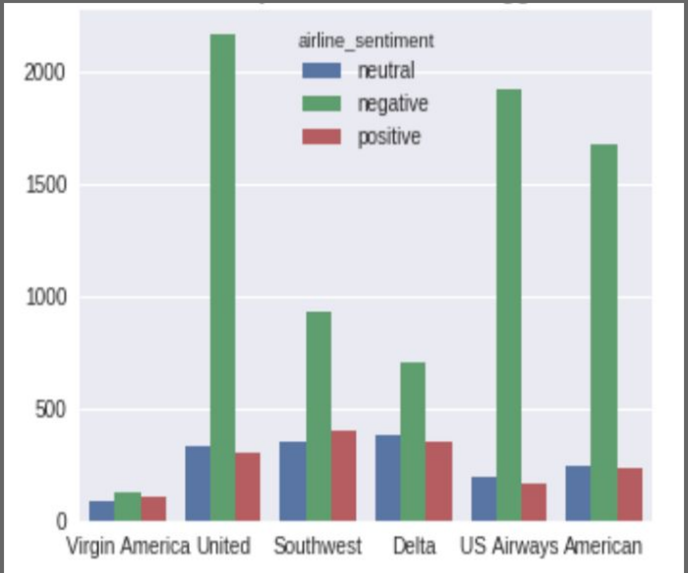
I ❤️ flying @VirginAmerica. ☺👍

👍👍✈️✈️💗 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---&gt;SEA!!! 🍷👍💺✈️

NLP



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

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

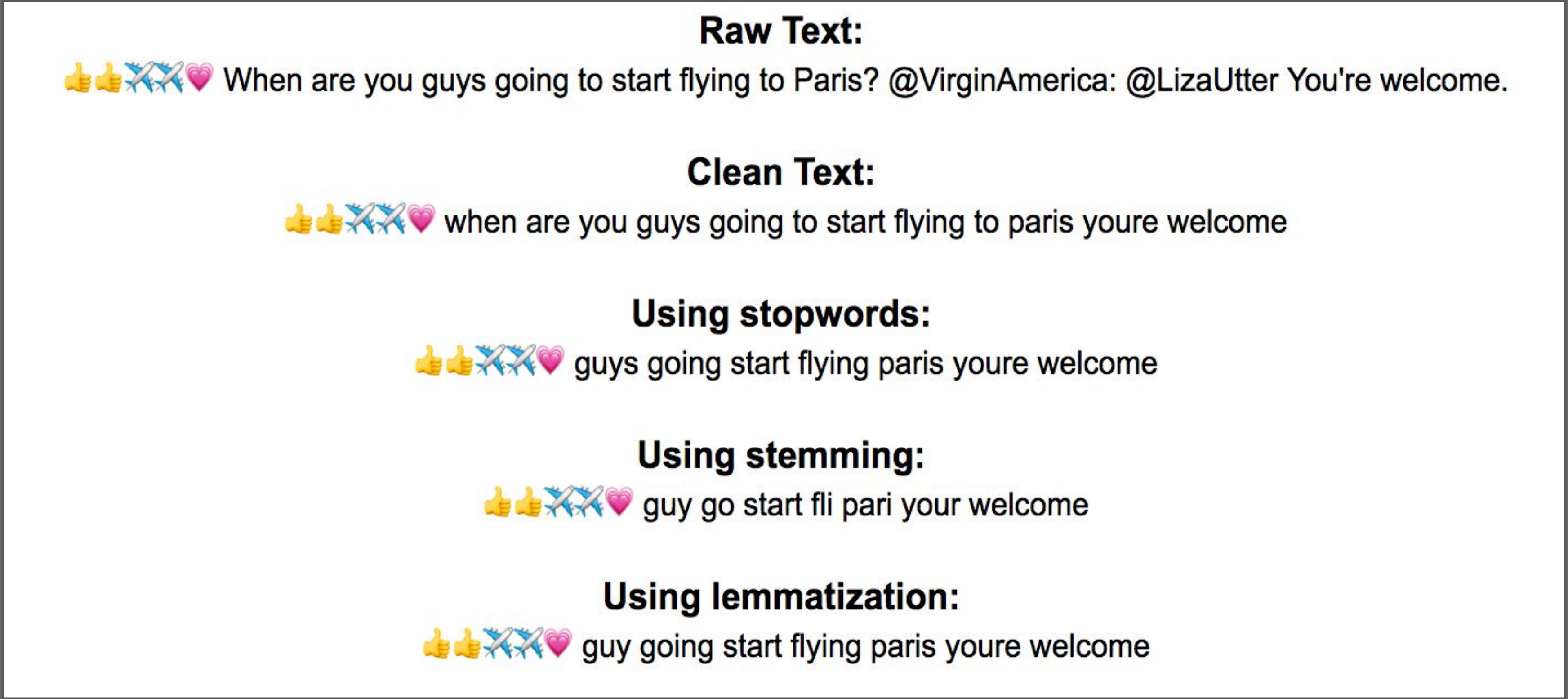
Lemmatization:

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

Stemming:

caresses, ponies, cats ⇒
caress, poni, cat

Processed Tweets



Raw Text:

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

Clean Text:

👍👍✈️✈️💗 when are you guys going to start flying to paris youre welcome

Using stopwords:

👍👍✈️✈️💗 guys going start flying paris youre welcome

Using stemming:

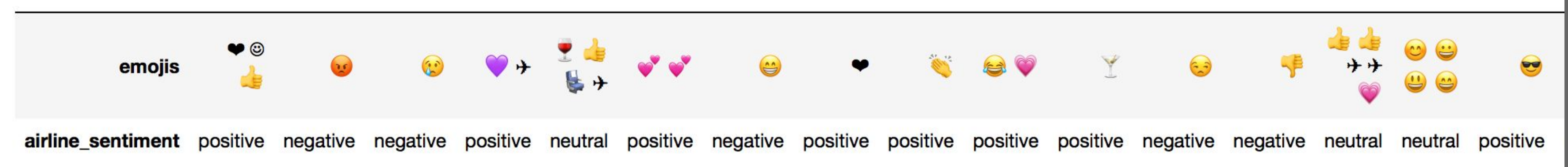
👍👍✈️✈️💗 guy go start fli pari your welcome

Using lemmatization:

👍👍✈️✈️💗 guy going start flying paris youre welcome

Handling Emojis

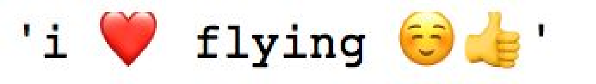
Extract emojis from tweets to see how it relates to the classified tweet sentiment:



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

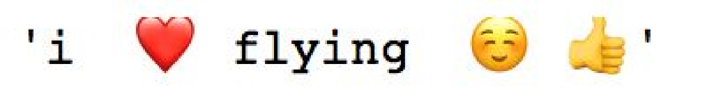
They look like good sentiment predictors. They are symbols so we have to encode them.

Users clump tweets together.



```
'i ❤️ flying ☺️👍'
```

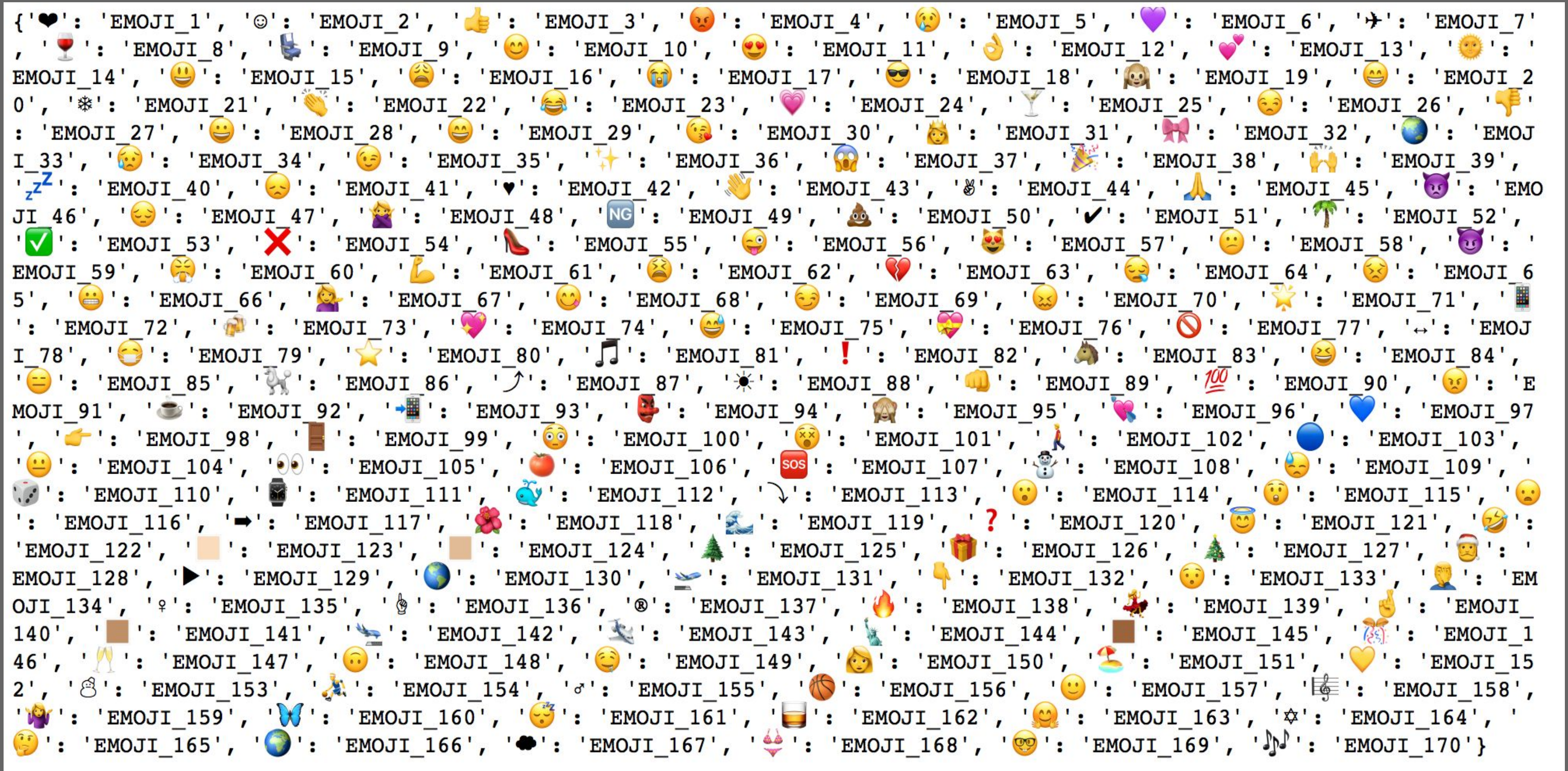
So first 'unclump' the emojis into individual symbols.



```
'i  ❤️  flying  ☺️  👍'
```

So now that they are separate, let's encode them!

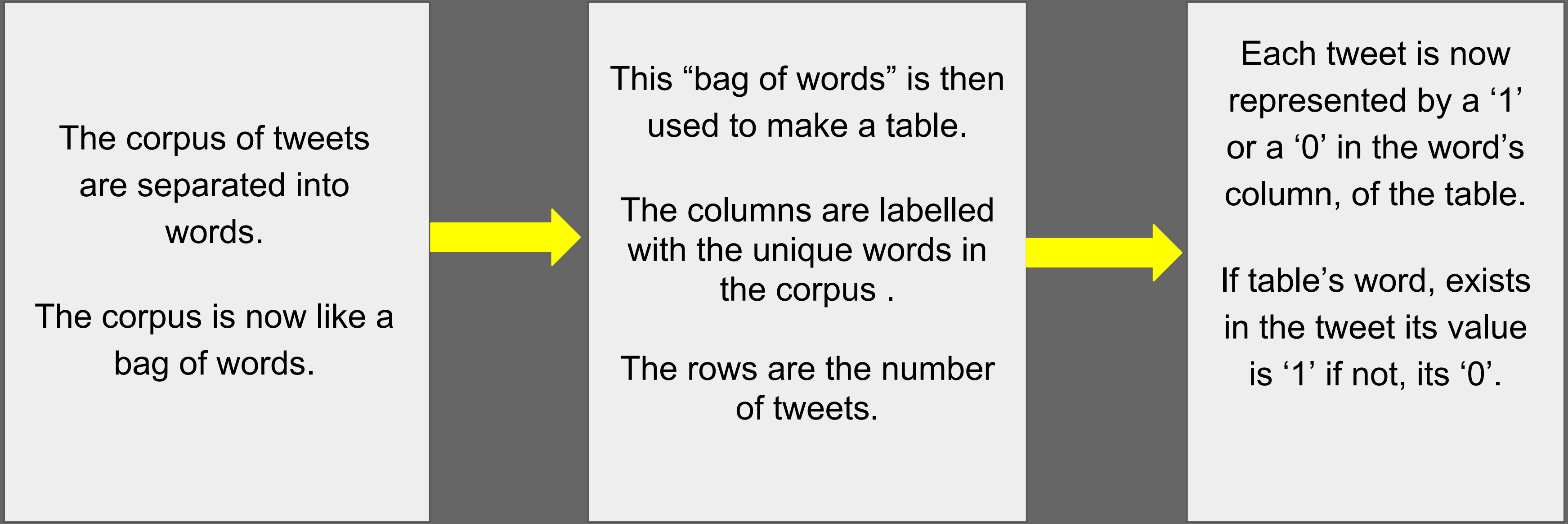
Encoding Emojis



```
{'❤': '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_2
0', '❄': 'EMOJI_21', '👏': 'EMOJI_22', '😂': 'EMOJI_23', '💗': 'EMOJI_24', '🍸': 'EMOJI_25', '😒': 'EMOJI_26', '👎'
: 'EMOJI_27', '😀': 'EMOJI_28', '😄': 'EMOJI_29', '😘': 'EMOJI_30', '👸': 'EMOJI_31', '🎀': 'EMOJI_32', '🌏': 'EMOJ
I_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', '👿': 'EMO
JI_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_6
5', '😬': 'EMOJI_66', '💁': 'EMOJI_67', '😋': 'EMOJI_68', '😏': 'EMOJI_69', '😖': 'EMOJI_70', '🌟': 'EMOJI_71', '📱
': 'EMOJI_72', '🍻': 'EMOJI_73', '💖': 'EMOJI_74', '😅': 'EMOJI_75', '💝': 'EMOJI_76', '🚫': 'EMOJI_77', '↔': 'EMOJ
I_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', '😠': 'E
MOJI_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', '🤦': 'EM
OJI_134', '♀': 'EMOJI_135', '☝': 'EMOJI_136', '®': 'EMOJI_137', '🔥': 'EMOJI_138', '💃': 'EMOJI_139', '🤞': 'EMOJI_
140', '🏽': 'EMOJI_141', '🛬': 'EMOJI_142', '🛩': 'EMOJI_143', '🗽': 'EMOJI_144', '🏾': 'EMOJI_145', '🎊': 'EMOJI_1
46', '🥂': 'EMOJI_147', '🙃': 'EMOJI_148', '🤤': 'EMOJI_149', '👩': 'EMOJI_150', '🏖': 'EMOJI_151', '💛': 'EMOJI_15
2', '☃': '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'}
```

Tokenization

Dividing tweets into words and creating a document term matrix, where the sentence becomes a group of words



The corpus of tweets are separated into words.

The corpus is now like a bag of words.

This “bag of words” is then used to make a table.

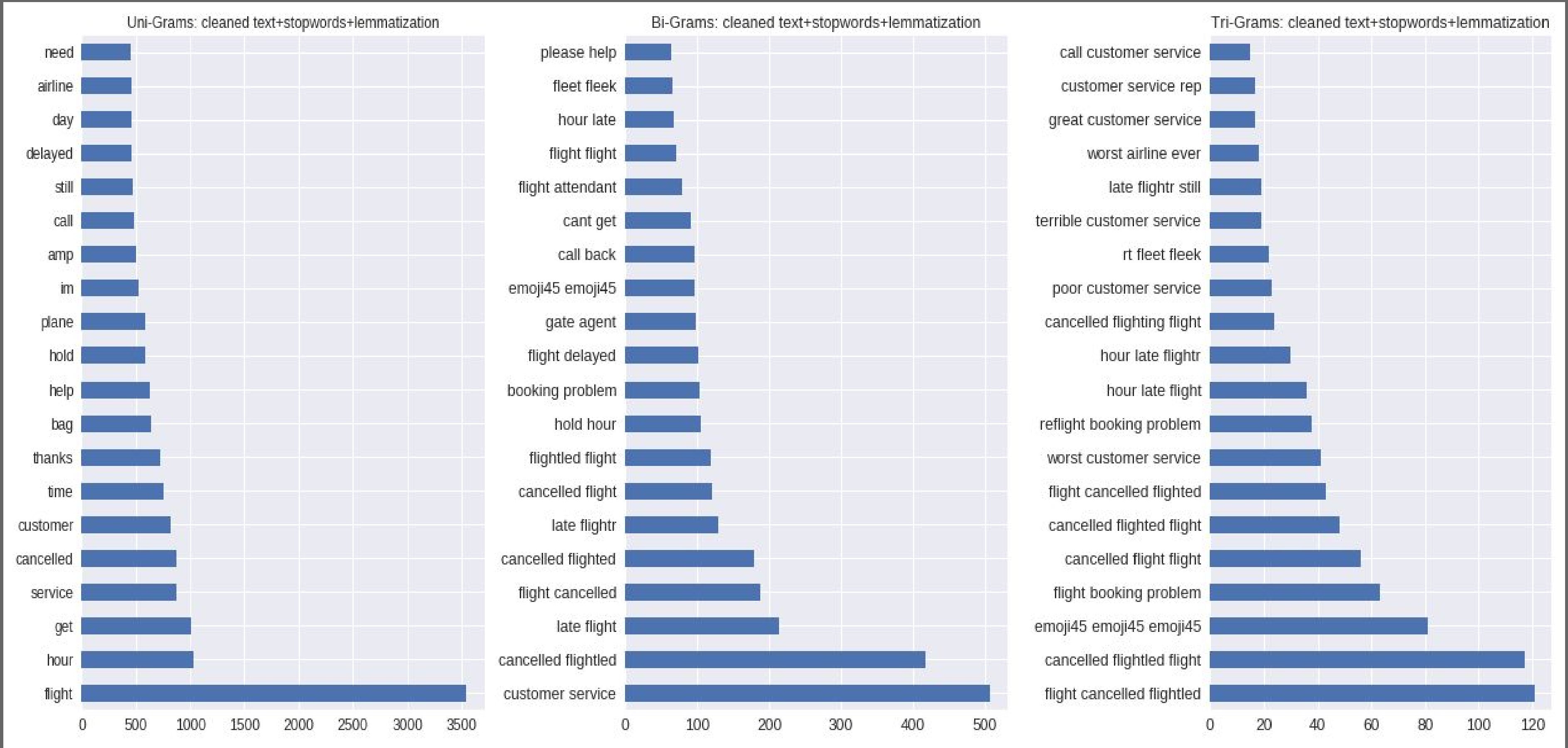
The columns are labelled with the unique words in the corpus .

The rows are the number of tweets.

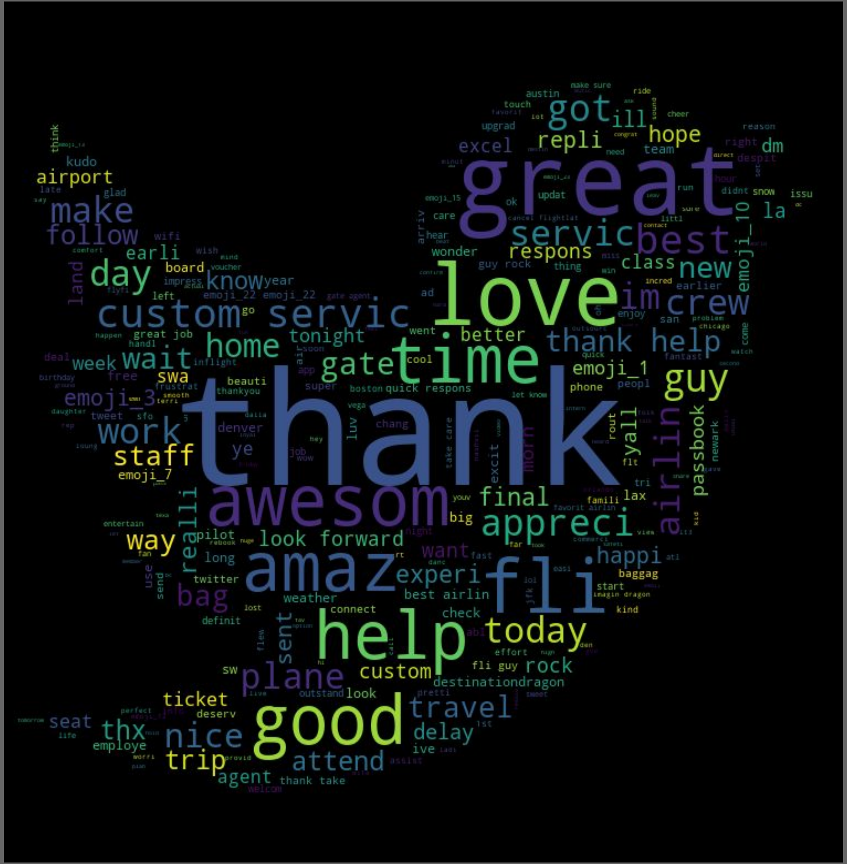
Each tweet is now represented by a ‘1’ or a ‘0’ in the word’s column, of the table.

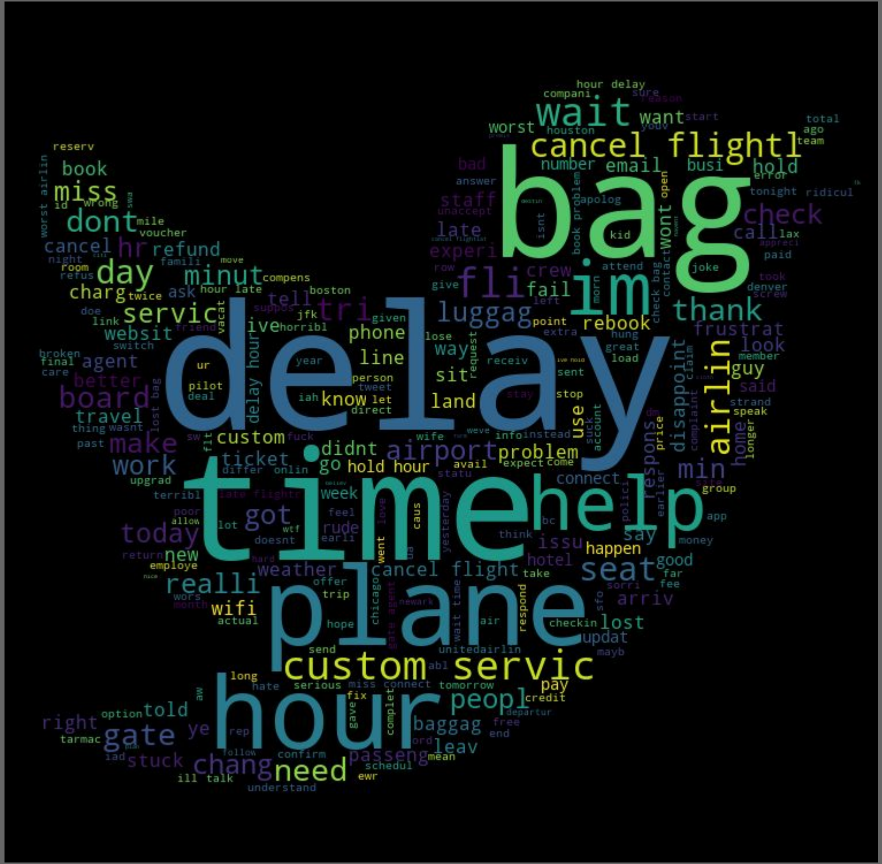
If table’s word, exists in the tweet its value is ‘1’ if not, its ‘0’.

Word Features



Word Frequency





Model Selection

Naive Bayes

Learning Mechanism

Naive Bayes models the joint distribution (X,Y) and then predicting the probability P(y| x) where X is the set of inputs features and Y is the output labels.

It's called a generative model.

Model Assumptions

Assumes that every word in a sentence is **independent** of the other ones.

So for NB the following 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 output mapping, by minimizing the prediction.

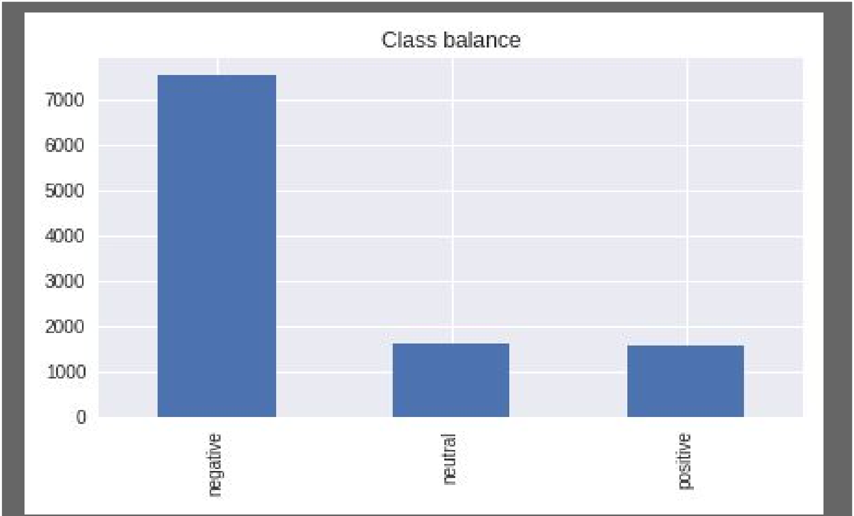
Its called a discriminative model, since it discriminates based on the error.

Model Assumptions

It assumes non-collinearity of features.

It splits the feature space linearly, so it works OK even if some of the variables are correlated.

Modeling



- Data is highly unbalanced.

- **Baseline Accuracy:**
.7 or 70% if we always guess the most frequent class.

Questions we want to Answer?

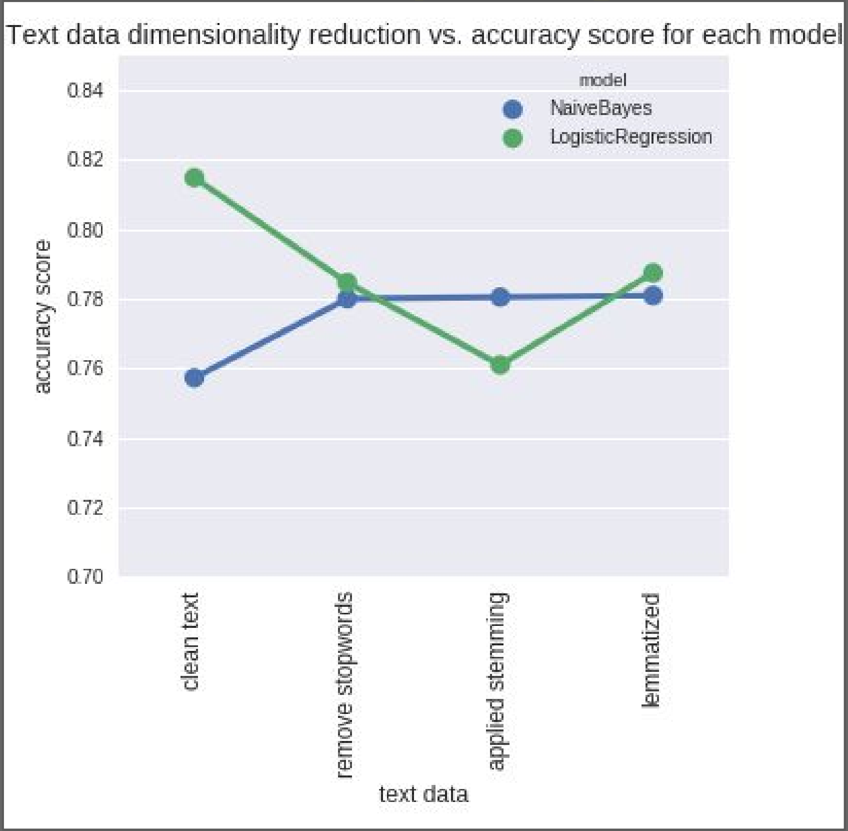
- What effects do ***emojis*** have on model predictions?

- What effects do different ***dimensionality reductions methods*** have on model predictions?

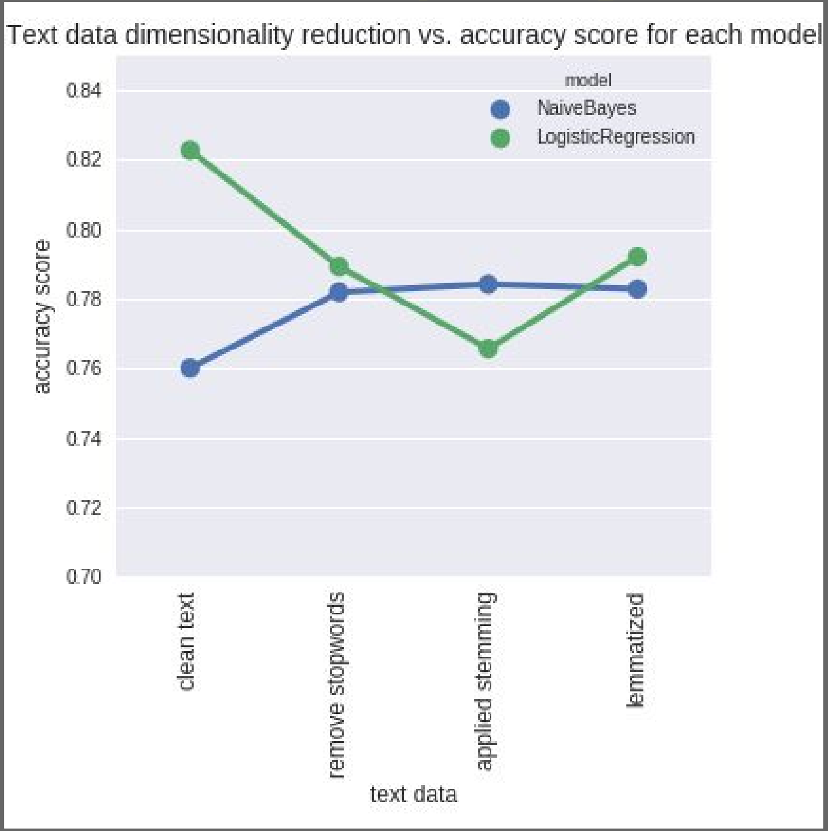
- What effects ***tuning*** the model's hyper parameters have on model predictions?

Model Accuracy

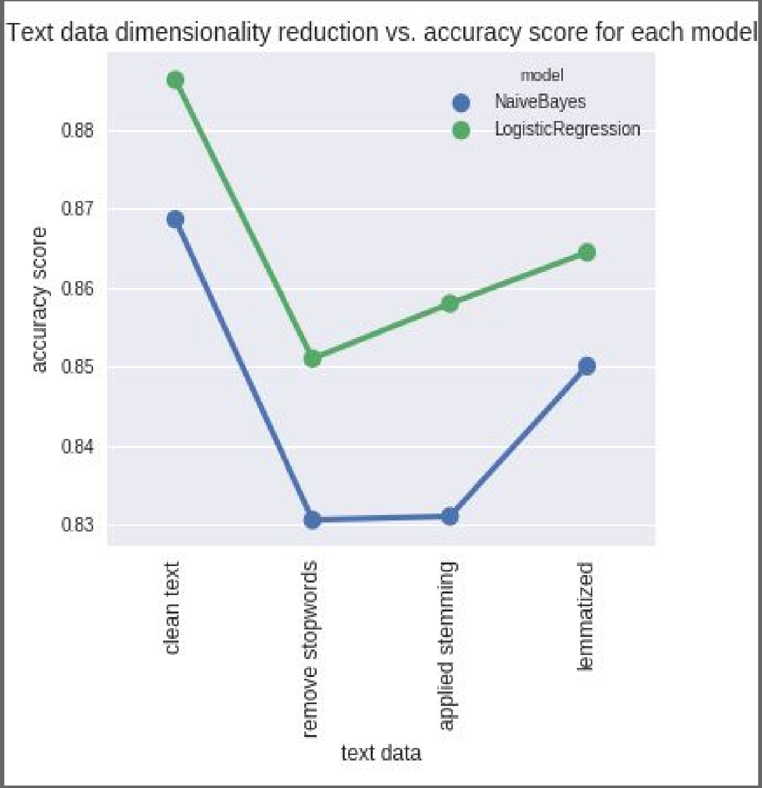
Without emoji encoding

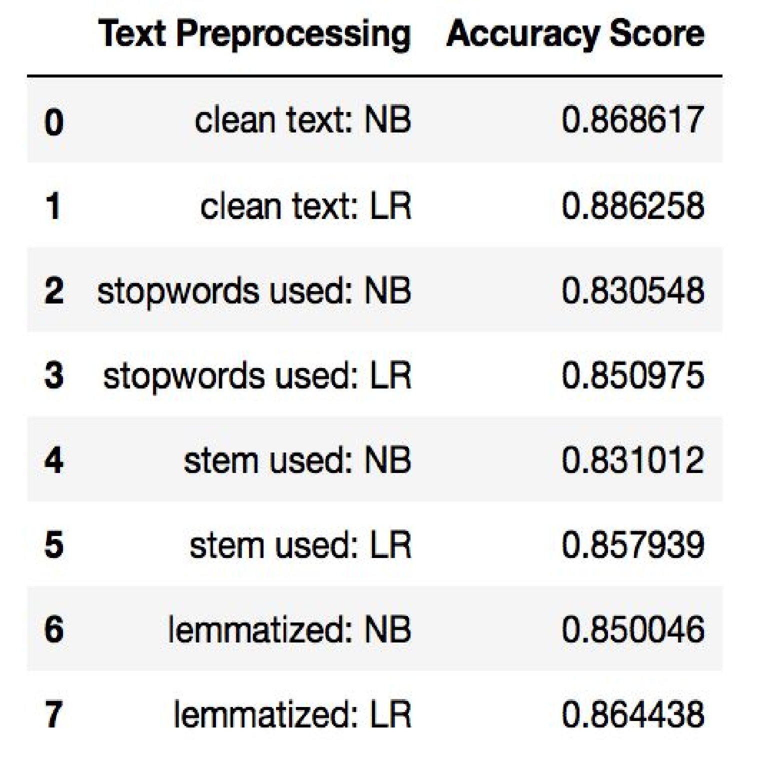


With emoji encoding



Best Tuned Model Results





	Text Preprocessing	Accuracy Score
0	clean text: NB	0.868617
1	clean text: LR	0.886258
2	stopwords used: NB	0.830548
3	stopwords used: LR	0.850975
4	stem used: NB	0.831012
5	stem used: LR	0.857939
6	lemmatized: NB	0.850046
7	lemmatized: LR	0.864438

Conclusions about Models

- Encoding Emojis do benefit model predictions.
- For the naive models, Logistic Regression does worse with feature dimensionality reduction, while Naive Bayes actually improves.
- With the tuned models, both Naive Bayes and Logistic Regression do worse by removing stop words, however they seem to recover ground when stemming and lemmatization are applied.
- Accuracy scores with data processed by cleaning+stopwords+lemmatization.

  Naive Bayes: 0.85

  Logistic Regression: 0.86

  Both are **higher than the baseline accuracy of 0.7!**

Test Tuned Models on Pulled Airline Twitter Data

Tweet:
Being nagged over and over and over about bag stowage during boarding is horrible and I'm sure largely pointless. Shut up #UnitedAirlines
Sentiment Prediction NB: negative
Sentiment Prediction LR: negative

Tweet:
@united YESSSSSSSSSSSSSSSSSSSSSSSSSSSSSS! 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:
2017: #Trump, Woman's March, #Protests, #TheWall, #TravelBan, Ariana Grande Concert, #UnitedAirlines, #FidgetSpinners, #Despacito, Paris Climate Agreement, #IT, Solar Eclipse, Hurricanes, Earthquakes, #NFL, Elections, #NorthKorea, #LasVegas, Assaults, #iPhoneX......#CrazyYear!
Sentiment Prediction NB: neutral
Sentiment Prediction LR: neutral

Next Step: Dealing with class imbalance

- Resample the most frequent class to have a similar size as the other classes.
- Tune the penalty hyperparameter of Logistic Regression.
- Evaluate other algorithms which deal well with imbalanced datasets.
  - Decision trees may perform well on imbalanced datasets. The splitting rules that look at the class variable used in the creation of the trees, can force both classes to be addressed.

Questions?

To test out my airline sentiment classifiers go to:

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