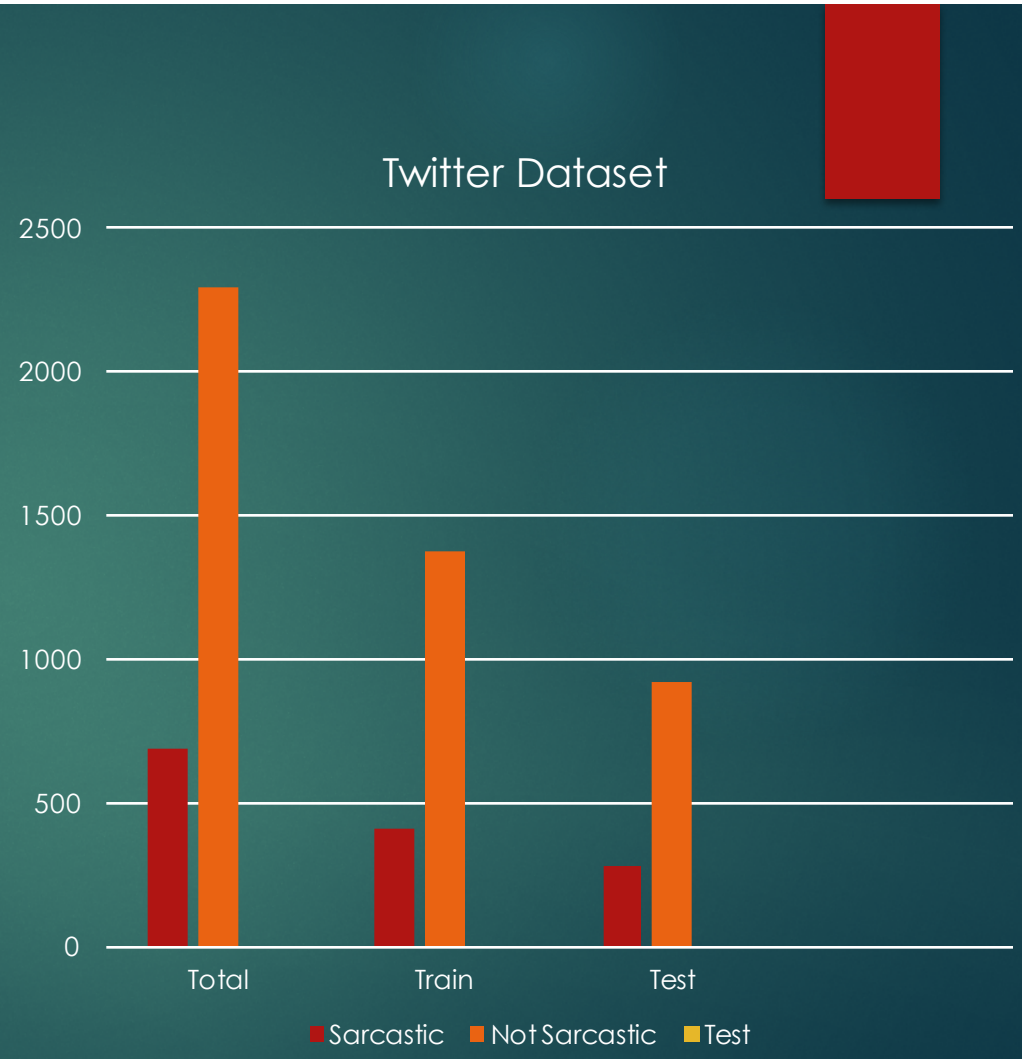
SARCASM RECOGNITION

KILARI MURALI KRISHNA TEJA AND NISHANT AGARWAL

Dataset

- 2980 tweets
- 60% Training and 40% Testing
- Stratified Sampling



Technical Approach

Data Cleaning & processing:

- Removal of #sarcasm
- Optimal splitting of tweets into multiple parts using operators such as , . ! ? : ;
- Expanding hashtags into words using a comprehensive word list of about 300000 words and Microsoft n-gram data using the code given here https://github.com/piyushbansal/hashtag-segmentation

Tf-Idf

- Technique that is used to find out the importance of a particular word in a document from a collection of documents. A weighting statistic for words often used in text mining.
- Can work well alone and often times works even better after applying SVD to the result of tf-idf matrix
- Following are some of the most important words the Logisitic regression model found out that are important for sarcasm classification.
- Great,fun,love,today,cancelled,yay,yay for,school tomorrow,thanks so,love being etc.

Topic Modelling

- Is a way to discover abstract topics that occur in a collection of documents
- represents each document as a combination of topics where each word is probability distribution of words that occur in those topics
- How it learns- We go through each document and randomly assign each word in the document to one of the k topics.(random assignment)
- For each document d and for each word w in d compute prob(topic t| doc d) and prob(w| topic t). Reassign w to a topic t where t is chosen with probability prob(topic t| doc d) * prob(w| topic t).
- Repeat until convergence
- Examples: Day,great,start,today,exact,degree,people,yes,way,school
- Examples: Thought,late,would,never,wait,realli,homework,fun

POS Tagging

- We used the general POS tagger from NLTK and then we tried grouping POS tags.
- We grouped all verb tags VB,VBZ,VBD,VBN etc into one tag "V" and we did the same with "adverb" and "adjective" tags.
- The F1 score increased to 0.39 from 0.33 with this approach.
- Below are some of the most important features our Logistic regression model reported,
- To,noun adv v,noun in cd,in dt

Word2Vec

- Used for capturing semantic similarity between the words.
- Scales very well with large amount of data
- Makes use of context window to capture similarity
- Advantageous compared to one hot encoding of vectors
- We ran word2vec with skip-gram model on an additional larger sarcasm dataset and we performed k-means clustering to obtain 500 clusters. Below are some of the words which are in the same cluster as the word "love"
- Wrong,saying,interviewer,offline,counselling,reap,realize,frankly,wake,awake,judgements

Sentiment Features

- Polarity of the tweet (using TextBlob and Afinn)
- Subjectivity of the tweet
- Sentiment shifts between different parts of the sentence (optimal splitting of tweets becomes important here)
- Polarity of the hashtag
- Subjectivity of the hashtag
- Sentiment contrast between the tweet and the hashtags (using TextBlob and Afinn)

Stacking

- Stacking involves training a learning algorithm to combine the predictions of several other learning algorithms.
- We reported stacking with Logistic regression in our final results table and it gave us the best F1 score
- However, when we did the same with Gradient Boosting trees and with the precision score as our optimization measure, we got a precision score of 0.75 but the recall score went to 0.05

Phrase extraction

- We used Gensim phrase extraction tool for this purpose and we assumed that all sarcastic tweets are more of positive polarity than negative.
- We reported the phrases the tweet contains along with the most positive sentiment word in the tweet.
- Conditional Probability of bigrams, trigrams, 4-grams, 5-grams of POS tags using CMU TweetNLP. Most unique sequence of tags for sarcasm came from 4-grams and 5-grams with probability of 1.
- E.g. Tweet - I just love missing the bus! ?
- Phrases - I just love missing and I just love missing the were extracted.
- Although the exact extraction love missing would have been perfect but it was quite close to it.

External Resources

- TextBlob – POS tagging, Spell Check, Sentiment Analysis
- TweetNLP – Unique POS tagger for tweets
- GenSim – Phrase Extraction, Word2Vec, Topic Modelling
- Afinn – Sentiment strength score
- Scikit learn – TF-IDF, ML algorithms such as clustering, SVD, GridSearch and CV
- WikiWords – List of most frequent English words (approx 300,000 words)

Results

Model	Train F1 score	Train Precision	Train Recall	Test F1 score	Test Precision	Test Recall
Baseline Model Tf-Idf (with SVD) + 5 syntactic features	0.6312	0.5240	0.796	0.539	0.470	**0.6327**
POS bigrams (No SVD)	0.424	0.323	0.614	0.3969	0.305	**0.567**
Topic Modelling (200 Topics)	0.4353	0.3630	0.5436	0.416	0.3764	0.465
Sentiment features + Word2Vec	0.5569	0.4546	0.7188	0.4809	0.4029	**0.5963**
Stacking	0.66535	0.5596	0.820	0.569	0.4933	**0.6727**

Conclusion

- Sarcasm detection is hard
- Sarcasm extraction is even harder
- As we can see from the results , syntactic features can prove to be useful even though there is a general notion that sarcasm has more to do with semantics
- We plan to further explore Word2vec to detect sarcasm.