



# Sarcasm Detection

*“Still the best project EVER!”*

Team Big Talk

Gennadii Styov  
Jennie Lee  
Michael Alaniz  
Mirko Draganic  
Steve Braich



# Outline

- Goal & Description
- Graphics/Analytics
- Lessons learned and issues encountered
- Conclusions
- Appendix

# Goal & Description: Recap

- Topic: Classifying text with software tool, a form of Natural Language Processing (NLP)
- Goal: Create a software tool that detects sarcasm in text
- Data used: reddit posts and responses
- Some tools explored:
  - Natural Language Toolkit (NLTK)
  - Jupyter Notebook
  - RStudio
  - MongoDB
  - Bash
  - Tableau
  - Flourish

Natural Language Toolkit  
(NLTK)



**BASH**  
THE BOURNE-AGAIN SHELL




# Goal & Description: The Data

- A repository of reddit posts and responses compiled by a team at Princeton.
- The posts are organized by category
  - 533 million total
  - 531.7 million non-sarcastic
  - 1.3 million sarcastic
- The authors organize the data into 3 different files.
  - sarc.csv: 184,340,693 KB
  - comments.json: 2,604,106 KB
  - sequences.csv: 225,507 KB

Data found at:

<https://www.kaggle.com/sherinclaudia/sarcastic-comments-on-reddit#train-balanced-sarcasm.csv>

Created by: <https://nlp.cs.princeton.edu/>




# Goal & Description: What We Hoped To Produce

- A statistical model that can identify a sarcastic comment with 70% accuracy.
  - The model will be based on the occurrence of things like key words, phrases, and the topic of conversation.
- If time permitted, we hoped to develop a code body that is easily accessible.
  - We want to leave more than just a model to the NLP community.
  - And we want our code to be easy to use and to expand on, for those who desire.



# Graphics/Analytics: Our Various Analytics

- Our group remains a collective of highly autonomous researchers.
- However, we were able to unite our efforts under 3 areas of study:
  - The part-of-speech makeup of keyword context.
  - Sarcastic content of comments by subreddit topic.
  - Keyword uniqueness to Sarcastic content.



# Little Sample of What We Did Through Jupyter

- There was quite a bit of code that was written for our project
- We wanted to demo and show a bit of it using Jupyter
- Here is the link to it:  
[https://github.com/steve3p0/cs510ds/blob/master/nlp\\_analytics.ipynb](https://github.com/steve3p0/cs510ds/blob/master/nlp_analytics.ipynb)
- Included in the Jupyter Notebook is some of the code we used to conduct some metrics and visualizations
- It does not contain all the code and metrics that we had done for the project, but it gives a little sample of some of the things we did
- The next slides give more details to some additional things we did with our project that is not included in the Jupyter Notebook



# What Makes Sarcastic Data Unique

- It occurred to us that a model could benefit from an understanding of what makes the sarcastic data distinct from the non-sarcastic data.
- Our collective research led us to believe that a keyword search would be a good place to begin some kind of analysis.
  - After all, what could say more about sarcastic data than keywords unique to it.





# A Good Start: Keywords and Keyness

- Words used to portray meaning, rather than to grammatical structure language, are called key words
  - Given their ability to conceptually distinguish one text from another, they are a good place to begin analysis
- We found the 50 most frequent keywords across each of 3 conceptual divisions of the data
  - **Reference:** 50 most frequent keywords across all the data
  - **Non-Sarcastic (nonsarc):** 50 most frequent keywords across the non-sarcastic data
  - **Sarcastic (sarc):** 50 most frequent keywords across the sarcastic data
- Several lines of research used these sets as a basis



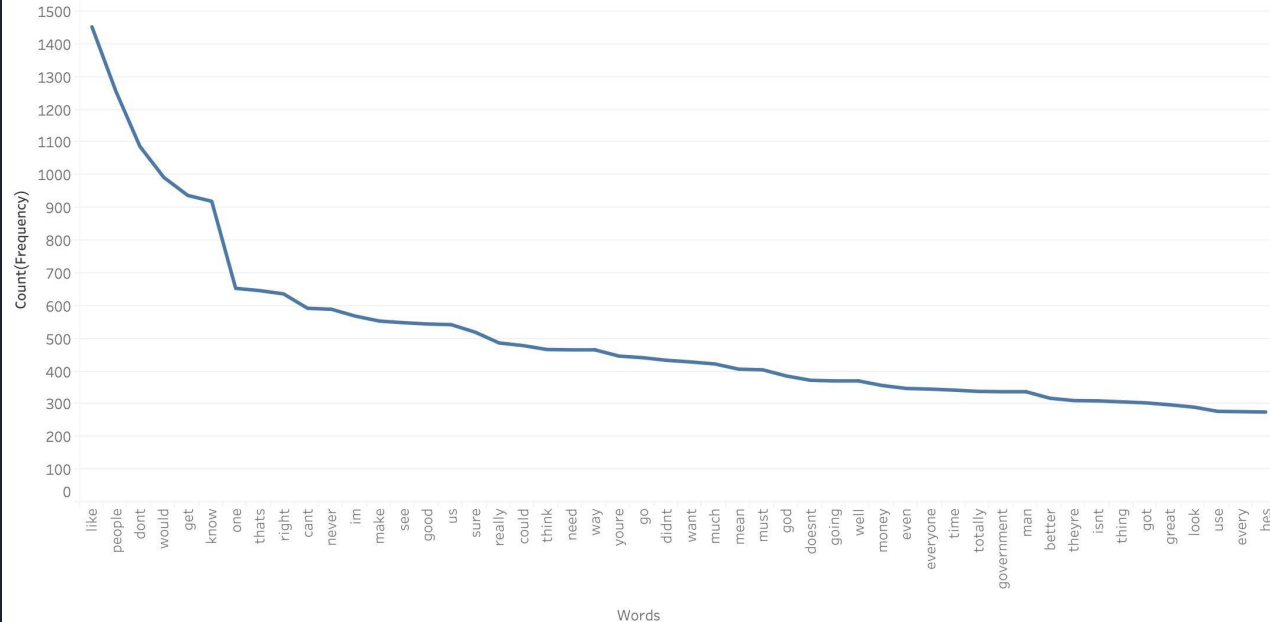
# Keywords and Keyness: Challenges

- Identifying keywords was nearly effortless due to NLTK support for such tasks
- Nevertheless creating the 3 subsets proved challenging for a number of reasons:
  - The biggest for us related to tokenization, the act of breaking text into the smallest units of meaning
  - To simplify our programmatic implementation, our analysis opted for a pseudo-tokenization in which a word like “shouldn’t” is treated as a single unit, “shouldnt”, rather than a formal tokenization in which that same word is tokenized into “should” and “n’t”
  - We removed ALL punctuation, even punctuation that may contain meaningful information
  - We suspect these subtle deviations from standard caused slight uncertainty in our result

# Keywords and Keyness: Most Frequent Keywords in Sarcastic Data Set

- 50 most frequent keywords in the sarcastic data set
- Arranged in descending order to see which words appear the most in the data set
- “like”, “people”, and “dont” are the top three frequent words that appear in the sarcastic data set

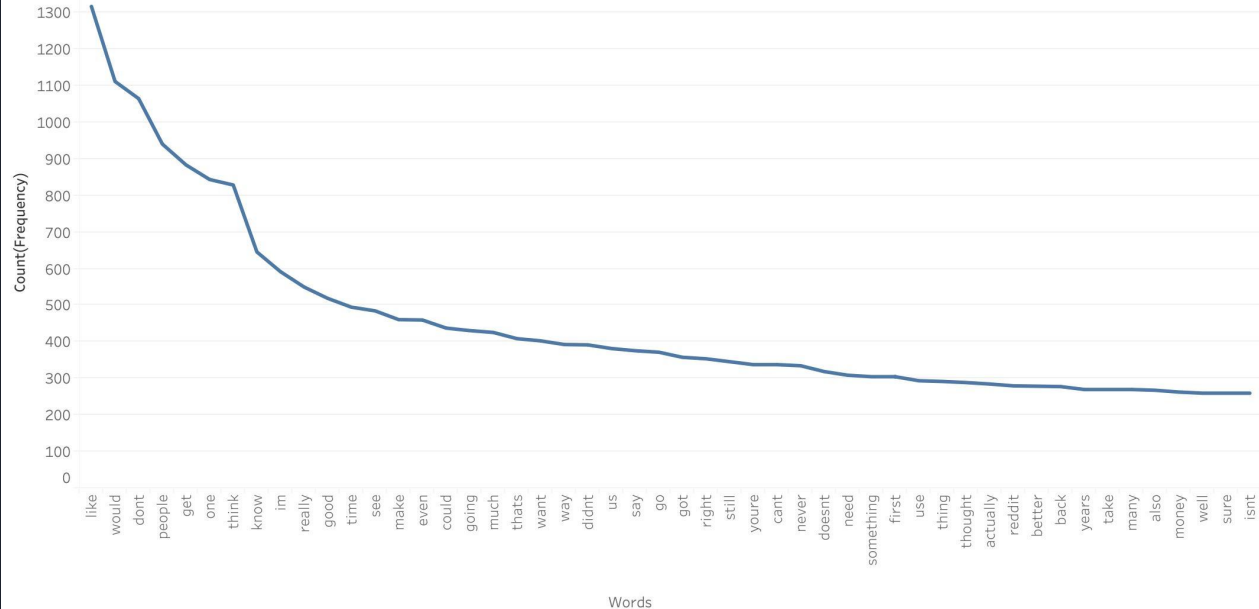
Most Common Words in Sarcastic Data Set



# Keywords and Keyness: Most Frequent Keywords in Non-Sarcastic Data Set

- 50 most frequent keywords in the non-sarcastic data set
- Arranged in descending order to see which words appear the most in the data set
- “like”, “would”, and “dont” are the top three frequent words that appear in the sarcastic data set

Most Common Words in Non-Sarcastic Data Set



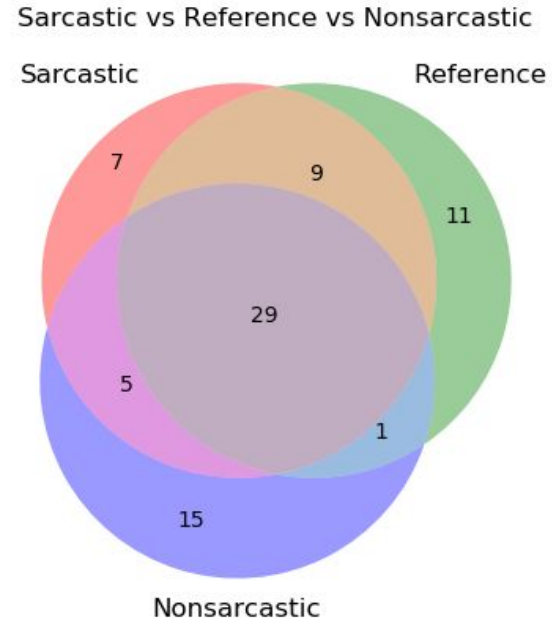


# Keywords and Keyness: Most Frequent Keywords Analysis

- As can be seen, there are many overlapping keywords that appear frequently in the datasets
- For example, “like”, “dont”, and “people” appear as frequent words in the data sets
- However, the frequency at which these words appear in the data sets differ
- Therefore, the frequency of these words that appear in the data sets are different even though they are considered the most frequent words that appear
- This then leads to the question of uniqueness among the datasets with regards to keywords

# Keywords and Keyness: What's Different

- The next logical step in our journey to distinguish the sarcastic set from the others was to see how the key words overlapped and to see if any were unique to sarcasm
- We got lucky and found 7 keywords unique to the sarcastic data set, woo!





## 7 Unique Words...Is Great, and Also Not

- Finding the unique words was great, but it reduced our working data set from 50 to 7 words
- So we turned our attention to the context of the keywords: the words that occur directly around the keywords
- Inclusion of the contextual words greatly increased our working dataset
- The increased data led to new findings and new hardships



# Keyword in Context Search: Some Finer Points

- Inclusion of the contextual words led to hyperparameter considerations:
  - Window size: how many words to include in the context
  - Window orientation: is the window centered on the keyword, is it strictly to the right, or strictly to the left
  - Do we include non-keywords (stop words) in the context or not
- To focus our research we:
  - Set the window size to 5 words
  - Looked for words strictly to the right of keyword
  - Filtered the context for keywords, removing all stop words
- Identifying keywords was of negligible complexity due to NLTK support
- NLTK contextual search is more complicated as it is made for continuous data, data disjoint reddit posts





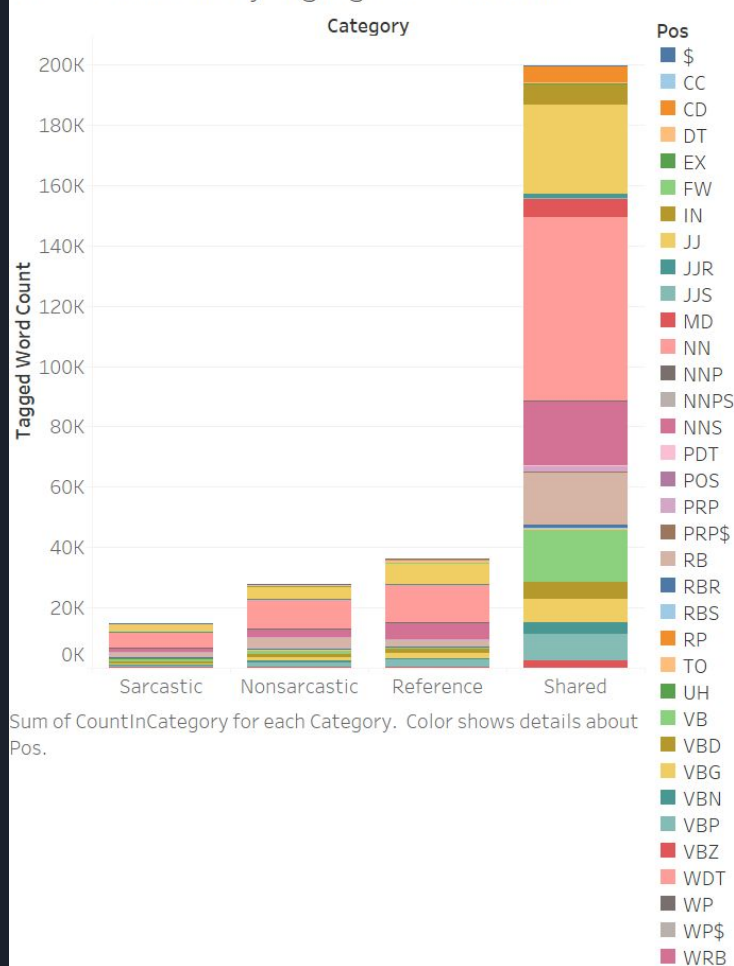
# Beyond Keyness: Part-of-Speech (POS) Tagging

- POS is determined by a words syntactic function and is another point of analysis
- POS tagging is of negligent complexity due to NLTK support
- We decided to see what new findings could be generated by POS tagging the contextual words
- There were two immediate results:
  - We gained a greater diversity of information as a single word can have several POS tags depending on usage
  - The additional passes over the data set greatly increased the run time of our burgeoning data pipeline

# Discouraging Results

- The raw data after the inclusion of the POS data was hard to examine, so we moved the results out of pycharm and into Tableau
- An initial pass in Tableau gave discouraging results.
  - The data sets were even more similar than previously thought
  - They all seemed to be composed of the same POS, in the same proportions

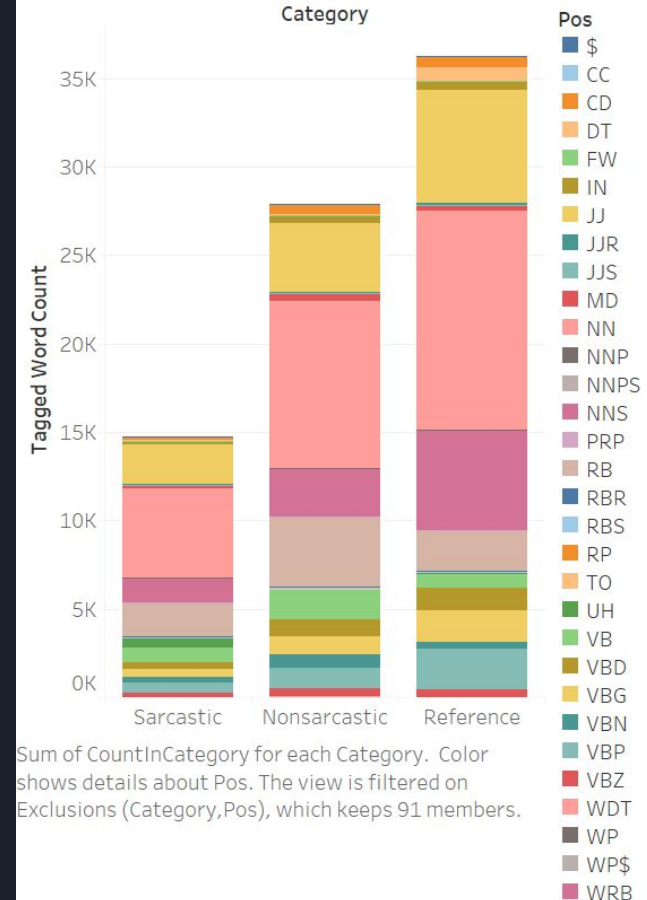
POS Variation by Significant Subsets



# SUCCESS!

- Never ones to quit, we took a closer look
- Removing the examination of words shared between all the data sets removed a lot of noise
- As a result we were able to see a subtle but definite distinction between the sarcastic set and the other data sets
- The 'UH' POS occurs in the sarcastic set in a noticeably greater proportion (the dark green bar seen only in the sarcastic set)

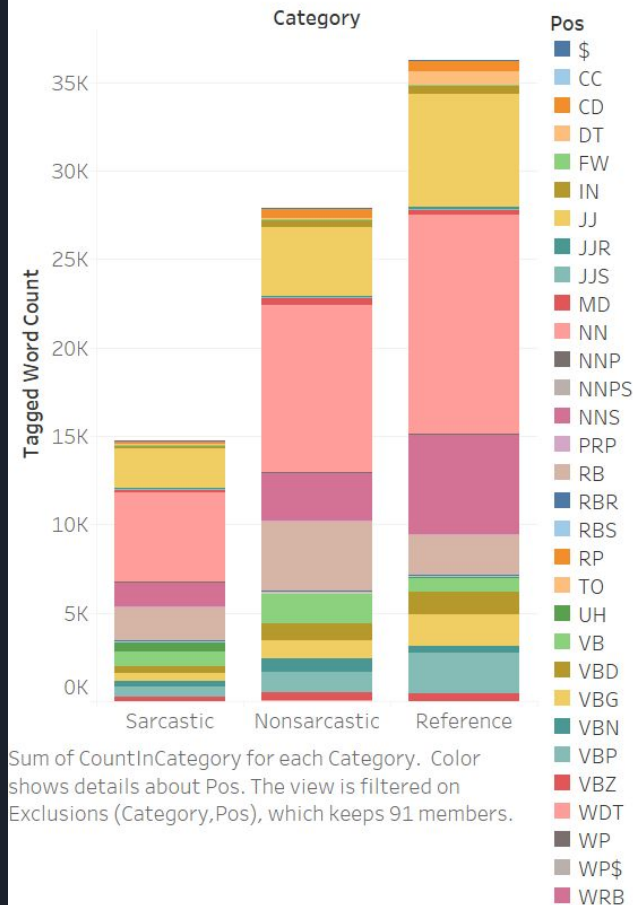
POS Variation by Distinct Subsets



# SUCCESS!

- The 'UH' stands for interjection
- The definition per Wiki: "An interjection is a word or expression that occurs as an utterance on its own and expresses a spontaneous feeling or reaction"
- This makes intuitive sense when one considers the widely held image of a problematic internet user making flippant or rude sarcastic comments safely and comfortably from behind their computer screen

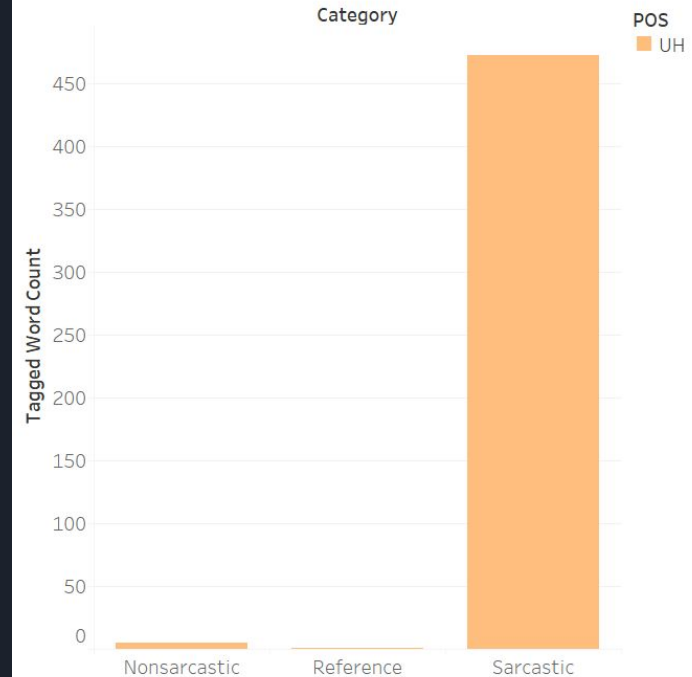
POS Variation by Distinct Subsets



# A Significant Data Point in a Predictive Model

- Upon further inspection, it is even more clear just of significant interjections are to sarcastic data
- Given more time, the presence of this POS data could be integrated into a predictive model with a great deal of confidence

UH (interjection tag) Counts by Distinct Subsets



Sum of Count In Subset for each Category. Color shows details about POS. The view is filtered on POS, which keeps UH.

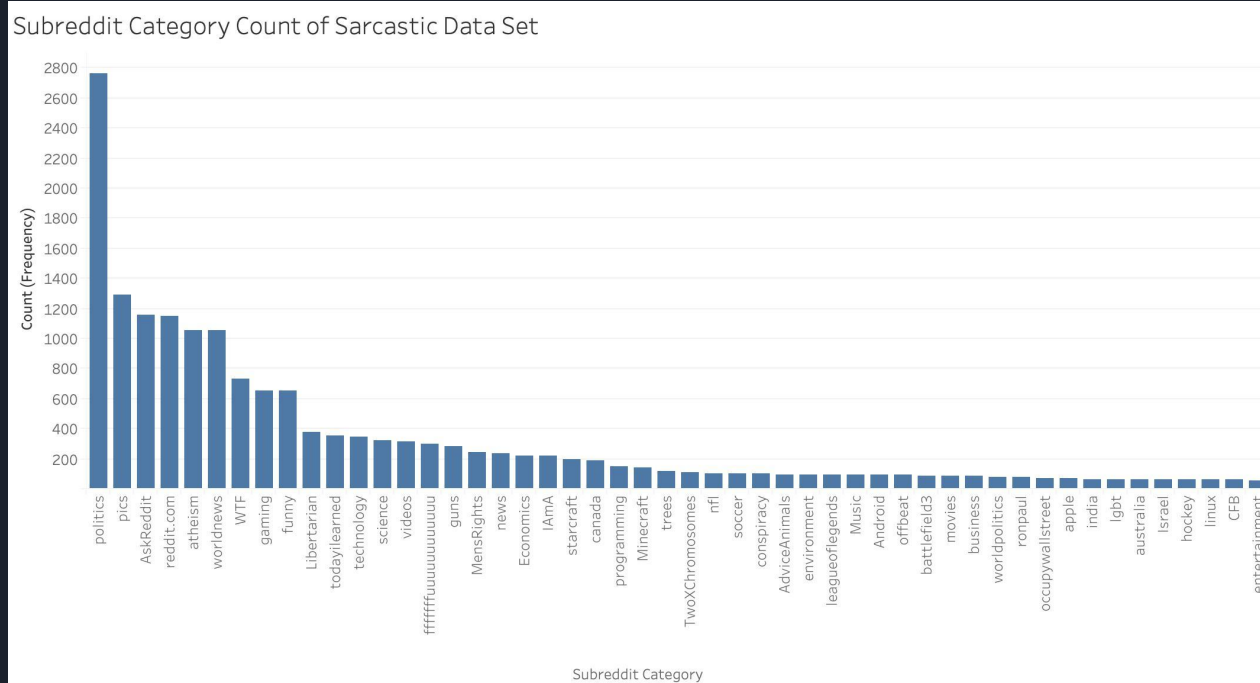


# Popular or not...? Subreddit Categories

- Wanted to see which categories yielded more comments being made
- Interested in what topics/categories were popular among the public
- Our ultimate goal was to create a predictive model to predict sarcastic comments, so we wanted to see how subreddit categories can aid in doing that
- The question that came to mind was “which subreddit categories have more sarcastic comments being made and what/why would those categories possibly have more comment”

# Popular or not...? Subreddit Categories - Most Popular in Sarcastic Set

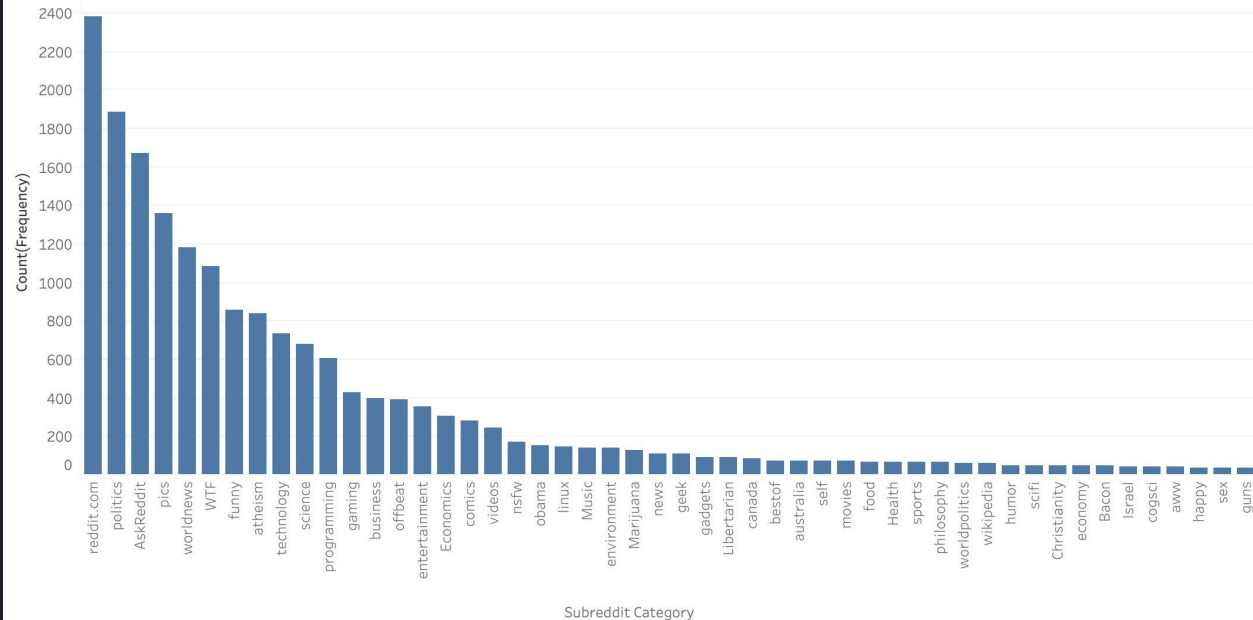
- Top 50 subreddit categories in the sarcastic dataset
- Politics, pics, and AskReddit are the top three subreddit categories



# Popular or not...? Subreddit Categories - Most Popular in Non-Sarcastic Set

- Top 50 subreddit categories in the non-sarcastic dataset
- Reddit.com, politics, and AskReddit are the top three subreddit categories

Subreddit Category Count of Non-Sarcastic Data Set





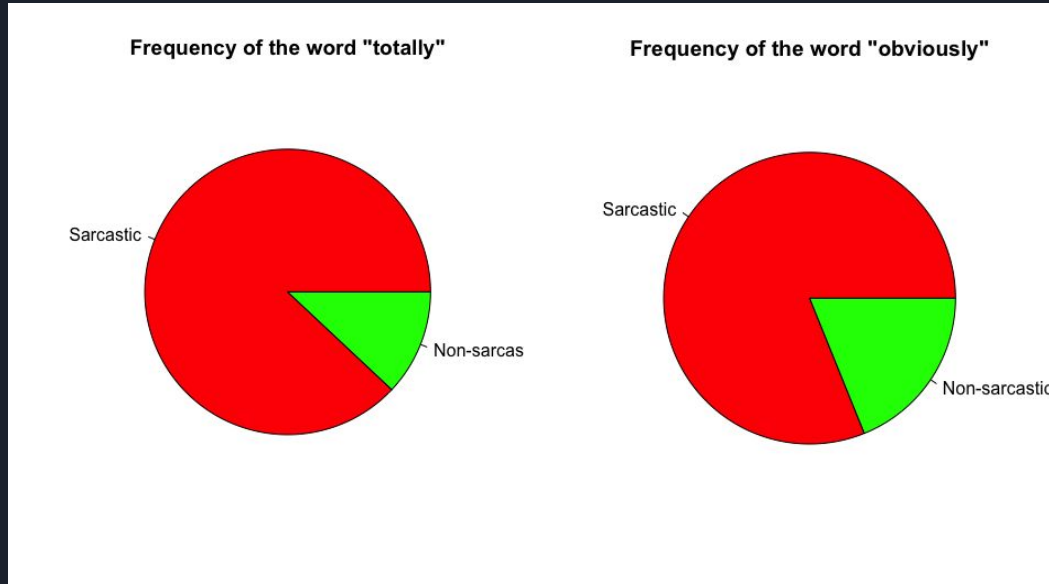


# Popular or not...? Subreddit Categories Analysis

- Overall, most of these categories listed are often open to opinions/comments being made
- The graphs show that certain categories resulted in more comments being versus other categories regardless of whether or not it was a sarcastic comment
- It would be interesting to see which categories are unique to which set
- Also, in relation to subreddit categories, another evaluation that would be interesting to look at is if certain subreddit categories of parent comments yielded in more comments being made (i.e., if there is a correlation between subreddit categories and number of comments being made)
  - Note: Attempted at performing this analysis, but ran into a few problems with grabbing and saving the correct data in the correct format for analysis, so this was not completed.

# Most Significant Words to Identify Sarcasm

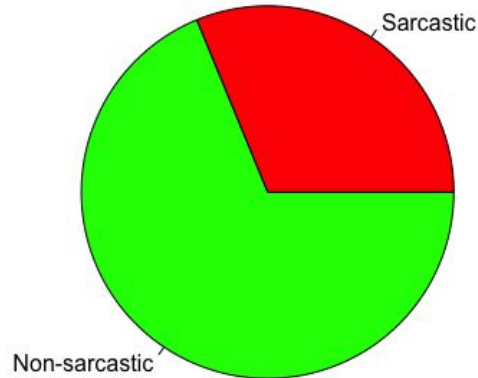
- Used results of 50 most frequent words in sarcastic and non-sarcastic comments
- Analyzed percentage of sarcastic comments containing these words in comparison to non-sarcastic comments
- Used R to process data, RStudio and Tableau for visualization
- The most significant words to identify sarcasm are:
  - Totally - 7.33 : 1 sarcastic comments to non-sarcastic comments ratio
  - Obviously - 4.30:1 sarcastic comments to non-sarcastic comments ratio



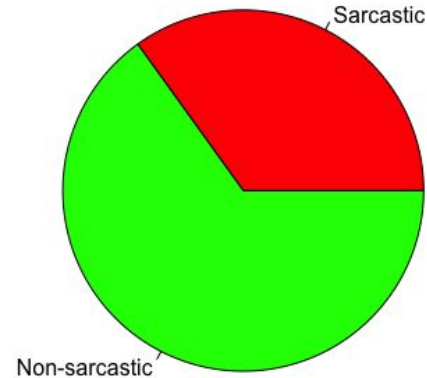
# Most Significant Words to Identify Non-Sarcasm

- The most significant words to identify non-sarcasm are:
  - Still - 0.46 : 1 sarcastic comments to non-sarcastic comments ratio
  - Think - 0.54 : 1 sarcastic comments to non-sarcastic comments ratio

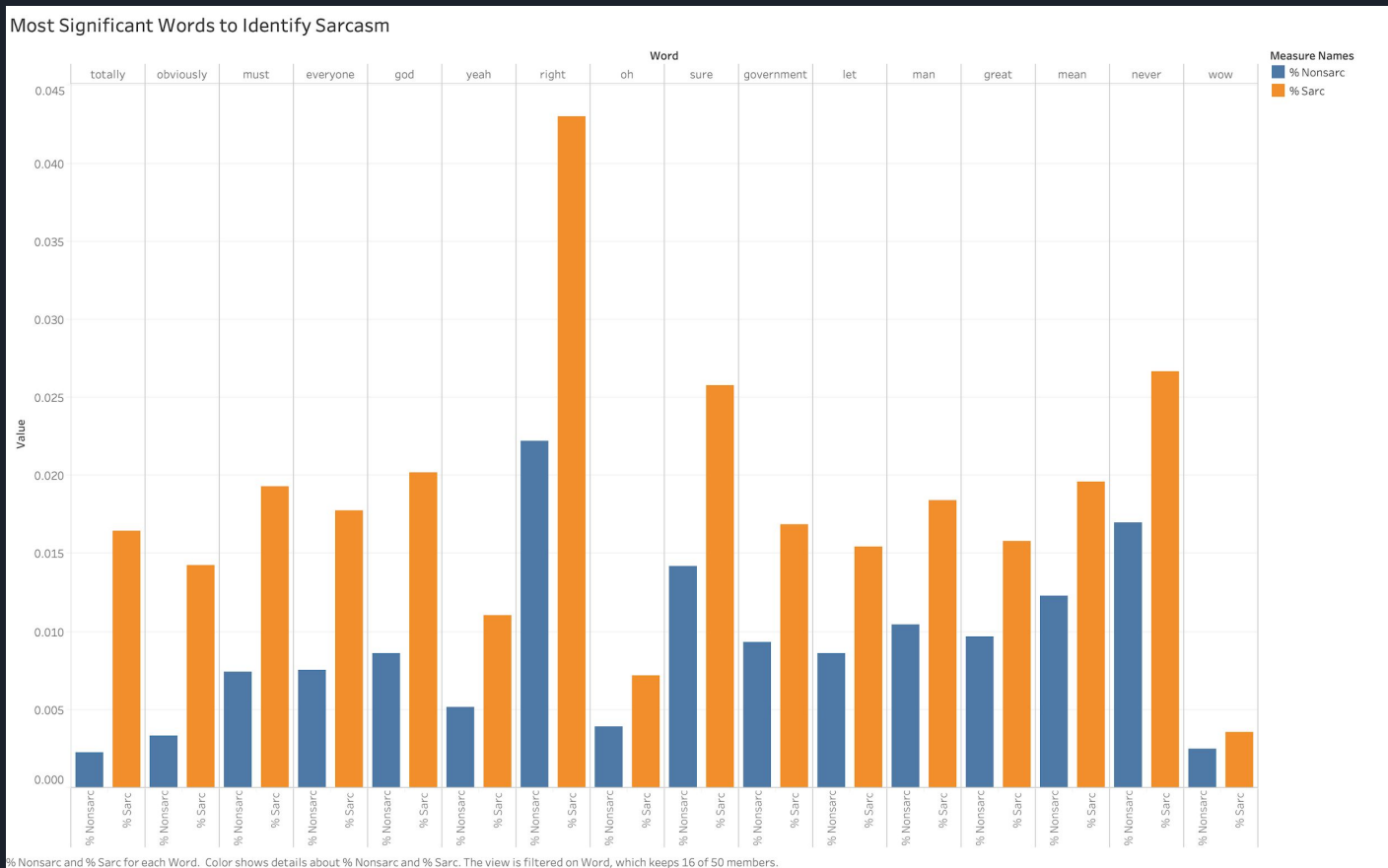
Frequency of the word "still"



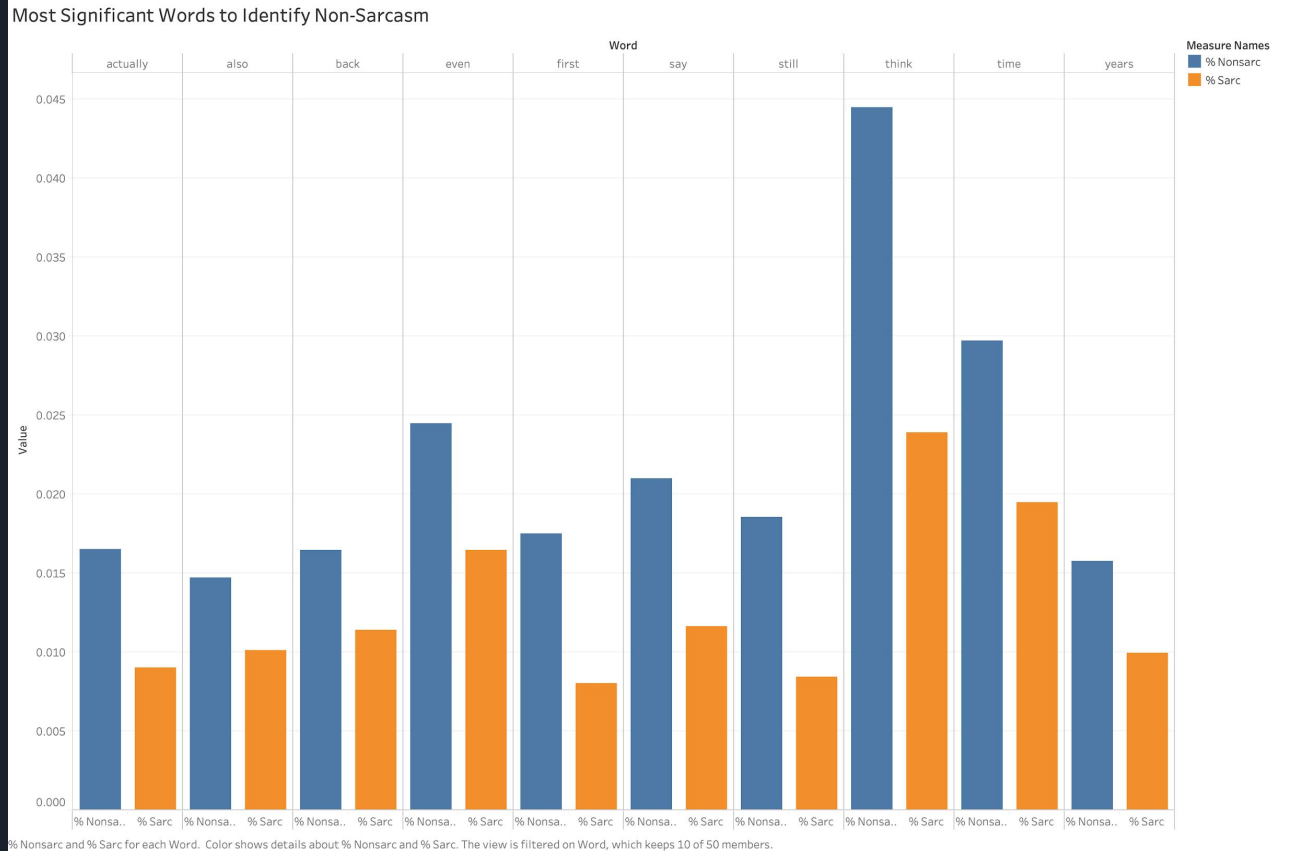
Frequency of the word "think"



# Most Significant Words to Identify Sarcasm

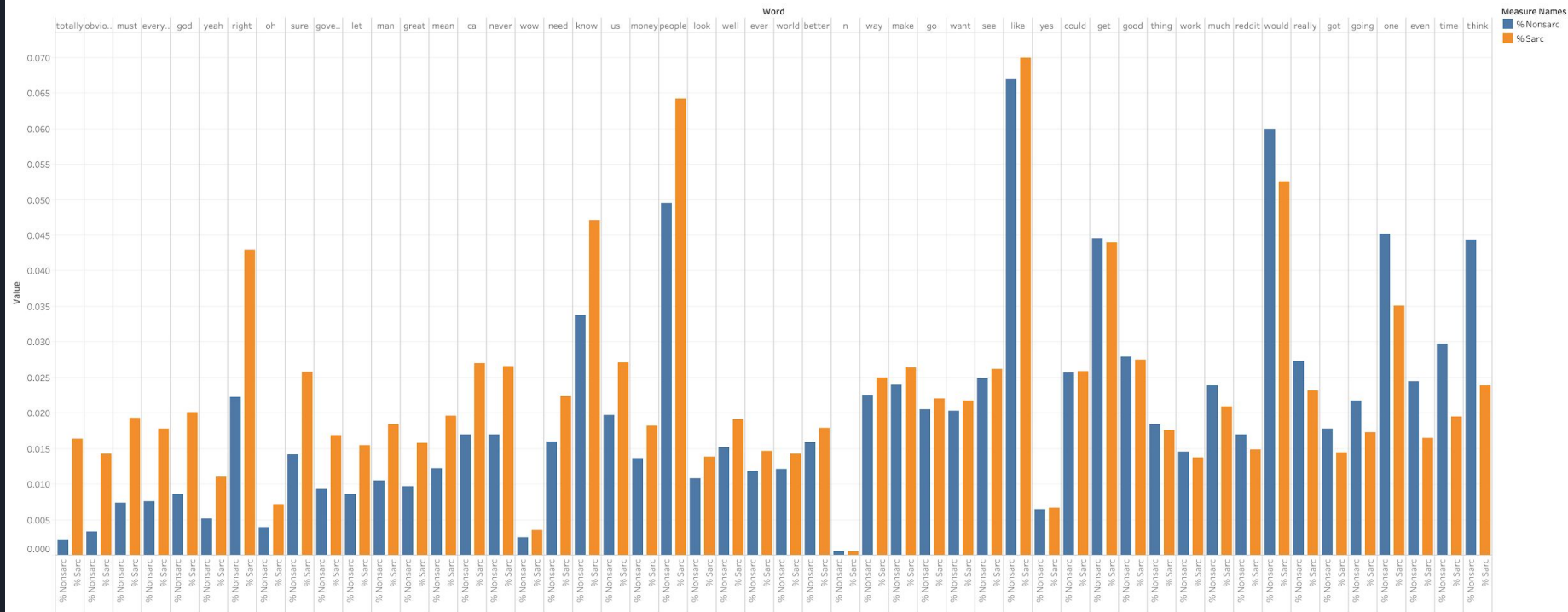


# Most Significant Words to Identify Non-Sarcasm



# Percentage of Comments that Contain Most Frequent Sarcastic Words

Percentage of Comments that Contain Most Frequent Sarcastic Words




% Nonsarc and % Sarc for each Word. Color shows details about % Nonsarc and % Sarc.



# Analysis and Difficulties/Lessons Learned

- Analyzed 50 most frequent sarcastic and 50 non-sarcastic words in 25934 sarcastic and 25000 non-sarcastic comments
- Used R and RStudio for analysis, and Tableau for visualisation
- Very easy to work with “.csv” files in R
- Visualization in R is much more difficult than in Tableau
- R is much slower than Python, execution of certain functions took several seconds on my machine
- There is sufficient evidence to conclude that certain words are more likely to be used in sarcasm
- The results have been saved into “most\_freq\_words\_sarc\_analysis.csv” and “most\_freq\_words\_nonsarc\_analysis.csv”



# Lessons Learned and Issues Encountered: What Went Well

- Distributing tasks among members - each member was willing to take on any task
- Group meetings and flexibility with meeting - we were able to meet up often and if we needed to change group meeting times and locations, each member was willing to adjust to the changes
- Visualization with Tableau - once we started playing around with Tableau once we ran some metrics, we were able to create some great visualizations
- Our project was not as well defined, so there were many routes we could take in defining our project - we were open to suggestions and changes that needed to be made to our project to ensure that it was going well





# Lessons Learned and Issues Encountered: What Issues We Encountered

- Visualizations in R - visualizations in R proved to be hard to do
- Visualizations in Python was not the best - visualizations Python were not as good as they could be
- Some limitations of Tableau made it a bit difficult to create graphs easily - graphs could be made, but there were minor details that needed to be added, but Tableau did not support well
- Our project was not as well defined - while we had control, we had a ton of decisions to make to define our project, such as what tools to use, to use a database, etc
- Our predictive model was not created - we were just short of making this due to lack of information and time constraints




# Lessons Learned and Issues Encountered: How We Resolved Those Issues

- Since visualizations for R and Python did not go well, our solution to this issue was to just use Tableau
- Despite the minor issues with Tableau, we were able to work around these limitations to produce some great visualizations to represent our metrics/analysis
- It took some time, but after some research and experimentation, we were able to decide on the tools and direction we wanted to go with for our project
- While we were not able to create the predictive model to predict sarcasm, we were getting close to it. We are satisfied with the progress we have made, so it is okay that we did not get around to making the predictive model



# Lessons Learned and Issues Encountered: What Took More Time and/or Was More Difficult or Easy Than We Expected?

- What took more time:
  - Making decisions on what direction we wanted to go with our project since there were different routes that we could take
  - Deciding what tools we wanted to use since there were many different tools that we could use, so it took time to explore and research what worked for our project
- What was more difficult:
  - Making the predictive model was difficult to accomplish since there were many metrics to run and time constraint
- What was easy:
  - Deciding what metrics we wanted to do - after cleaning up the data, we easily decided what metrics we wanted to run
  - Working as a group was easy because we were in communication with each other and we agreed upon what we wanted to accomplish



# Lessons Learned and Issues Encountered: Our Opinions About the Tools Selected

- NLTK and Python was good (to a point) - good at some aspects, but complicated at other aspects
- Mixed feelings about working with R - for language analysis, it's not great, but it was good to explore and see how it worked with NLP
- Tableau was great for visualizations
- Jupyter was good as an interactive demo for what we were doing for our project
- MongoDB was terrible with our project because it did not work well with Tableau - there were lots of extra things to download to make it work with Tableau



# Conclusion

- Overall, project went well!
- We didn't get to making our predictive model, but we got a significant amount of work done in our project and we are satisfied with how far we got
- We explored many tools and doing research on tools and visualizations was a great experience for us - we were able to learn about more about tools we were already using and learn about new tools
- The project was a great learning experience for us and something that we can apply to future projects



# Appendix

- For those who are more interested in seeing all that we had accomplished for our project, this is the link to our github page: <https://github.com/steve3p0/cs510ds>
- The github page contains additional results, graphics, and analytics