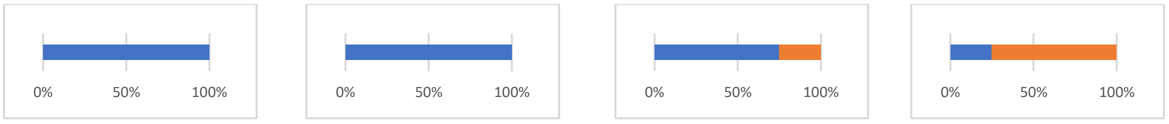
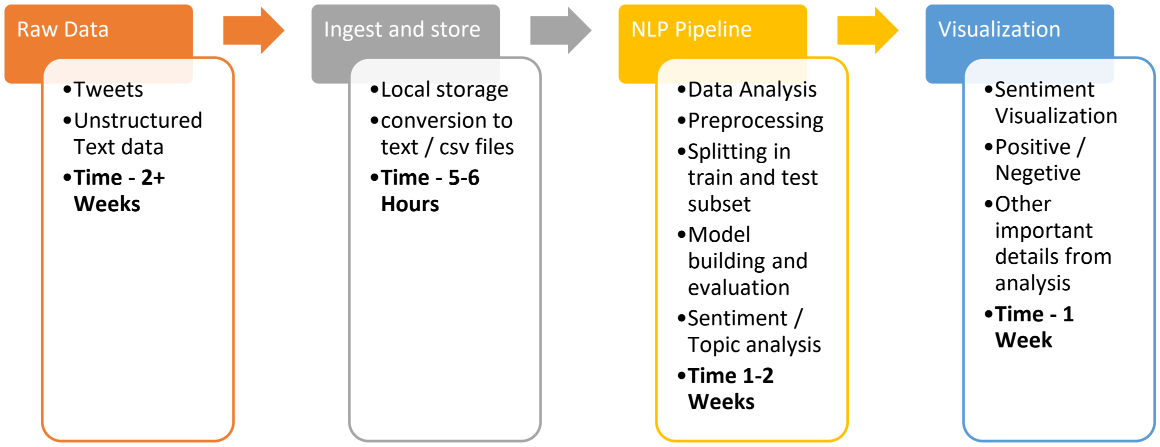
Project Progress Report

Course:	**CS410 Text Information System**	MCS DS Fall 2022
Project Name	**US Midterm Elections Twitter Sentiment Analysis**	
Team Name	Googolplex	GitHub Repo: https://github.com/pd-illinois/CS410Googolplex.git
Team Members	Prateek Dhiman (Captain)	pdhiman2@illinois.edu
	Unnati Hasija	uhasija2@illinois.edu

Implementation Progress by phase





From an Implementation perspective we divided the project into four phases as shown above

1) Raw Data Scrapping (Completed)
2) Ingestion and Storage (Completed)
3) NLP Pipeline (In progress)
4) Visualization (In Progress)

1) **Raw Data Scrapping**: We settled on using *SNSCRAPE* instead of tweepy mainly due to the ease of scrapping tweets, no need for a developer account with twitter or API limitation. Snscrape was easy to install with pip / conda. And we would easily run it from the command line or embed it our python script and scrape the tweets as needed. It provided us with all needed data for the tweets We used generic twitter search with common and unbiased words like “vote” , “midterm2022” , “democrats” and “republicans” to accumulate a wide range of tweets. We

searched tweets from July 2022 till the election date of 8th Nov 2022 and accumulated over 100000 tweets for our analysis.

2) **Ingestion and Storage**: Using *sntwitter.TwitterSearchScraper* the tweets were captured in a dataframe and saved as a CSV File locally. The overall size of the CSV file was around 30MB and manageable as a local storage / Git Repository. As a result, we do not require any cloud storage.

3) **NLP Pipeline:** This is the most important part and crucial part of our project. Based on the initial project proposal we sub-divided this into two core areas

   a. **Sentiment Analysis and Prediction using Naïve-Bayes**

   As a part of our project, we wish to categorize and predict the sentiment of future tweets by building and training a model. As mentioned earlier, we collected/scraped tweets on US midterm election data. After cleaning the tweets and removing stop words, we decided on the polarity of the tweets as positive, negative, and neutral. Since deciding the polarity manually on such a large dataset would have been a daunting task, we used "TextBlob(tweet).sentiment.polarity" to decide the polarity on this large data (> 20,000 rows). We assume a tweet to be positive if the output of the polarity is > 0, negative if < 0, and neutral if = 0.

   We know that Naive Bayes (NB) is the simplest and fastest classification algorithm for large datasets and relies on the Bayes probability theorem for class prediction. In our project, we have tried to classify the scraped tweets using Naïve Bayes (NB) classification algorithm and tried to predict the sentiment of various tweets after training on 1/4th of the data.

   We are currently trying to implement different variations of NB algorithms like Gaussian, Multinomial etc., and trying different n-grams (1,2,3 etc.). We are using "CountVectorizer" to count the word frequencies. It tokenizes the string and gives an integer ID to each token. It counts the occurrence of each of those tokens. We are currently also trying to evaluate the accuracy. To do so, we are using "metrics" from "sklearn". We plan to plot the accuracy of different variations of algorithms with different n-grams of words.

   b. **Topic Modeling using LDA (Latent Dirichlet Allocation):** Latent Dirichlet Allocation (LDA) is a generative statistical model that explains a set of observations through unobserved groups, and each group explains why some parts of the data are similar. In this, observations (e.g., words) are collected into documents, and each word's presence is attributable to one of the document's topics. Each document will contain a small number of topics.

   In order to prepare data for LDA , we cleaned the tweets using regex , removed stopwords , emoji, URLs and @ and # . We used GENSIM Library to create dictionary , corpus and its term document frequency. We restricted the maximum number of topics to 10 and created a LDA Model.

   The next steps are to analyze the LDA Model and derive meaningful topics from it .

4) **Visualization** : This area is still in progress. We plan to use matplotlib / pyLDAvis / wordcloud for visualization and analysis of data and the observations from Naïve Bayes and LDA Models.

Challenges

So far we have encountered the following challenges

- Data Cleaning : The tweets come in all forms. Some have full words , small sentences or just images , icons or emoji. So Data cleaning and tokenization are come of the key aspects before we attempt data clearning
- Choosing right N-Gram representation for Naïve Bayes. – We are working to evaluate, compare and fine tune Naïve Bayes for optimal results
- Further evaluation of LDA – We are working to fine tune the LDA Models. At the moment the challenge is to make the Language Model more relevant to election results rather then any other type of voting (example voting for best music album)