IDS 561: Analytics for Big Data
Pfizer Vaccine Tweets - Sentiment Analysis
FINAL REPORT

PRITHA GHOSH , SNEHAL MHATRE, VIBHANSHU

Problem Statement

Sentiment Analysis is a sub-field of Natural Language Processing (NLP) that tries to identify and extract opinions within a given text. The aim of sentiment analysis is to gauge the attitude, sentiments and emotions of the writer based on the computational treatment of subjectivity in a text.

Businesses today are heavily dependent on data. Majority of this data is unstructured text coming from sources such as emails and social media sites. The content from social media sites like Twitter poses serious challenges, not only because of the huge volume of data but also because the language used is quite different from dictionary English, since it uses short-forms, emoticons and memes to express sentiment.

Sentiment analysis allows businesses to make sense of this kind of unstructured data and derive vital insights from it without having to manually analyse it.

There are several challenges to Sentiment Analysis:

- Understanding sentiments via text is not straightforward since a text may contain multiple sentiments at once.
- Computers are not very good at comprehending figurative speech like metaphors, similes.
- Use of emoticons and slangs in social media texts makes sentiment analysis difficult.

Business Aspect

The Pfizer vaccine by BioNTech has saved millions of lives already, there are several vaccines against Covid but Pfizer stood out as the most reliable name. Pfizer not only became the highest selling Covid vaccine but it's parent company BioNTech has left several names behind in the list of Fortune 500. Pfizers has its fair share of supporters as well as critics. Several critics oppose Pfizer because of it's enormous profit margins, others just oppose vaccines in general. We have performed sentiment analysis on Pfizer vaccine through the 110K live tweets we extracted over a period of last two months to gain insight about how the Pfizer vaccine is perceived in general.

Dataset Extraction

We have extracted the data from Twitter API using the Tweepy library. In order to do this, we applied for the Twitter Developer Account, where we created an application and generated the keys and tokens necessary for data extraction.

```
tweet_list=[]
class MyStreamListener(tweepy.StreamListener):
    def __init__(self,api=None):
        super(MyStreamListener,self).__init__()
        self.num_tweets=0
        self.file=open("/content/drive/Shareddrives/IDS561/twitter_data_consolidated/tweet (12).txt","w")
    def on_status(self,status):
        tweet=status._json
        self.file.write(json.dumps(tweet)+ '\n')
        tweet_list.append(status)
        self.num_tweets+=1
        if self.num_tweets<100000:
            return True
        else:
            return False
        self.file.close()
```

We then streamed live tweets with the mention of 'Pfizer' and saved it as a text file. We have extracted 109,288 tweets (680MB).

```
#create streaming object and authenticate
l = MyStreamListener()
stream =tweepy.Stream(auth,l)
#this line filters twitter streams to capture data by keywords
stream.filter(track=['Pfizer'],languages=['en'])
```

The output data is in an unrefined format, which needs to be properly cleaned before it can be used for further analysis.

Initial Analysis

The data extracted from the Twitter API has 37 columns:

```
Index(['created_at', 'id', 'id_str', 'text', 'source', 'truncated',
       'in_reply_to_status_id', 'in_reply_to_status_id_str',
       'in_reply_to_user_id', 'in_reply_to_user_id_str',
       'in_reply_to_screen_name', 'user', 'geo', 'coordinates', 'place',
       'contributors', 'retweeted_status', 'is_quote_status', 'quote_count',
       'reply_count', 'retweet_count', 'favorite_count', 'entities',
       'favorited', 'retweeted', 'filter_level', 'lang', 'timestamp_ms',
       'possibly_sensitive', 'quoted_status_id', 'quoted_status_id_str',
       'quoted_status', 'quoted_status_permalink', 'display_text_range',
       'extended_tweet', 'extended_entities', 'withheld_in_countries'],
      dtype='object')
```

Missing Data

We have analyzed all the columns to determine which columns have a large amount of missing data and removed them since they would not provide any important information.

created_at	0
id_str	0
text	0
truncated	0
in_reply_to_status_id	94137
in_reply_to_status_id_str	94137
in_reply_to_user_id	93627
in_reply_to_user_id_str	93627
in_reply_to_screen_name	93628
user	0
geo	109278
coordinates	109278
place	108929
contributors	109288
retweeted_status	27299
is_quote_status	0
quote_count	0
reply_count	0
retweet_count	0
favorite_count	0
entities	0
favorited	0
retweeted	0
filter_level	0
lang	0
timestamp_ms	0

possibly_sensitive	88620
quoted_status_id	86664
quoted_status_id_str	86664
quoted_status	86669
quoted_status_permalink	86669
display_text_range	93000
extended_tweet	94451
extended_entities	103570
withheld_in_countries	109286

We continue our analysis with the 18 columns mentioned below:

```
Index(['created_at', 'id_str', 'text', 'truncated', 'user', 'is_quote_status',
       'quote_count', 'reply_count', 'retweet_count', 'favorite_count',
       'entities', 'favorited', 'retweeted', 'filter_level', 'lang',
       'timestamp_ms', 'possibly_sensitive', 'withheld_in_countries'],
      dtype='object')
```

Data Preprocessing

Next, we have performed some data cleaning steps using pyspark:

1. Dropping the columns which have no significance to the dataset - example serial number, created date (since all the tweets are live streamed and hence created on the same day)
2. Removal of URLs, hashtags and user mentions from the tweets text
3. Removal of special characters, numbers, multiple spaces and single characters from the tweets text
4. Removing the retweets prefix - 'RT : ' from the body of the tweet

Data Exploration:

- Counting the number of words in each tweet

```
+--------------------+-------------+--------------+--------------------+----------+
|                text|retweet_count|favorite_count|       text_original|word_count|
+--------------------+-------------+--------------+--------------------+----------+
| BREAKING: Pfizer...|            0|             0|RT @EricSpracklen...|        16|
| Toronto lockdown...|            0|             0|RT @CrazeeKanuck:...|        11|
| &amp; year olds,...|            0|             0|RT @LeonardRoxon:...|        18|
| You will find he...|            0|             0|RT @Alfath2021: Y...|        14|
| When Twitter giv...|            0|             0|RT @laralogan: Wh...|        20|
| BREAKING: Pfizer...|            0|             0|RT @EricSpracklen...|        16|
| Uh oh what is th...|            0|             0|RT @JackPosobiec:...|         6|
|Insane! How many ...|            0|             0|Insane! How many ...|         4|
| Man maid virus m...|            0|             0|@JackPosobiec @pf...|        12|
| BREAKING: Pfizer...|            0|             0|RT @EricSpracklen...|        16|
| Just wait for it...|            0|             0|RT @johncardillo:...|        17|
| BREAKING: Pfizer...|            0|             0|RT @EricSpracklen...|        16|
| Before you settl...|            0|             0|RT @OrangeCoFL: B...|        20|
| If true, most of...|            0|             0|RT @JDHughes4: If...|        25|
| Two small childr...|            0|             0|RT @ke11ybender: ...|        23|
| Dude what is you...|            0|             0|@JackPosobiec @pf...|        21|
| PFIZER SCIENTIST...|            0|             0|RT @RealMattCouch...|        18|
| The Pfizer phase...|            0|             0|RT @jengleruk: Th...|        17|
| Keep sharing thi...|            0|             0|RT @Centaur_UK: K...|         5|
| More vaccine harms?|            0|             0|RT @TonyHinton201...|         4|
+--------------------+-------------+--------------+--------------------+----------+
```

- **Sentiment Analysis:** We use VADER (Valence Aware Dictionary and Sentiment Reasoner), which is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.

The reason we decided to use VADER for analyzing tweet sentiments is because it is sensitive to both polarity (positive/negative) and intensity (strength) of emotion.

VADER generates a 'compound' score, which is calculated by summing the valence scores of each word in the lexicon, adjusted according to the rules, and then normalized between -1 and +1. This is the most useful metric if we want a single unidimensional measure of sentiment for a given sentence. This 'compound' score can be referred to as a normalized, weighted composite score.

	text	retweet_count	favorite_count	word_count	sentiments
0	BREAKING: Pfizer 'Fetal Cell' Whistleblower M...	0	0	16	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...
1	Toronto lockdown czar's husband has 'financia...	0	0	11	{'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound...
2	& year olds, "ACCIDENTALLY" given Pfizer ...	0	0	18	{'neg': 0.242, 'neu': 0.758, 'pos': 0.0, 'comp...
3	You will find here what changes your life for...	0	0	14	{'neg': 0.0, 'neu': 0.775, 'pos': 0.225, 'comp...
4	When Twitter gives mewarning about anything, ...	0	0	20	{'neg': 0.0, 'neu': 0.87, 'pos': 0.13, 'compou...

Based on the compound score, we have classified tweets as positive where the score>=0.05, negative where the score is <=-0.05 and neutral where the score is between -0.05 and 0.05.

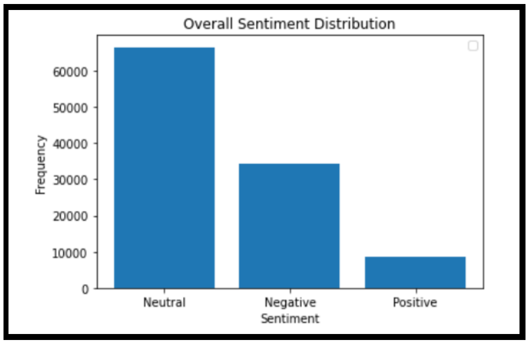
	text	retweet_count	favorite_count	word_count	Positive Sentiment	Neutral Sentiment	Negative Sentiment	Overall Sentiment	Overall Sentiment Description
0	BREAKING: Pfizer 'Fetal Cell' Whistleblower Melissa Strickler has been TERMINATED\n\n"You are not under any circumstances…	0	0	16	0.000001	1.000001	0.000001	0.000001	Neutral
1	Toronto lockdown czar's husband has 'financial ties' to Pfizer, AstraZeneca\n	0	0	11	0.000001	1.000001	0.000001	0.000001	Neutral
2	& year olds, "ACCIDENTALLY" given Pfizer VACCINE & they BOTH HAVE DEVELOPED HEART ISSUES NOW! 💉🚫\n\nAccidentally??? Hmm…	0	0	18	0.000001	0.758001	0.242001	-0.585899	Negative
3	You will find here what changes your life for the better:\n \n …	0	0	14	0.225001	0.775001	0.000001	0.440401	Neutral
4	When Twitter gives mewarning about anything, it makes me more determined to read it. Censorship does not belong infree…	0	0	20	0.130001	0.870001	0.000001	0.400501	Neutral

Exploratory Analysis

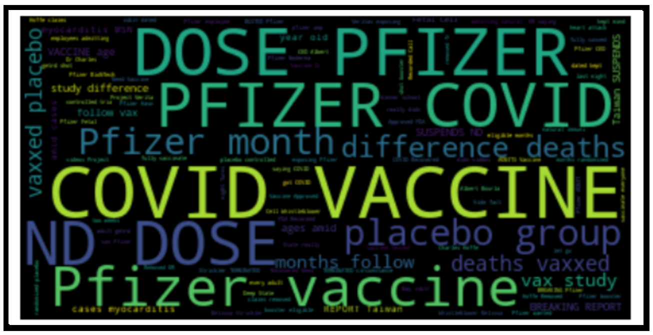
- We first look at the distribution of positive, negative and neutral sentiments in our dataset

	Overall Sentiment Description	frequency
0	Neutral	66544
2	Negative	34182
96	Positive	8562

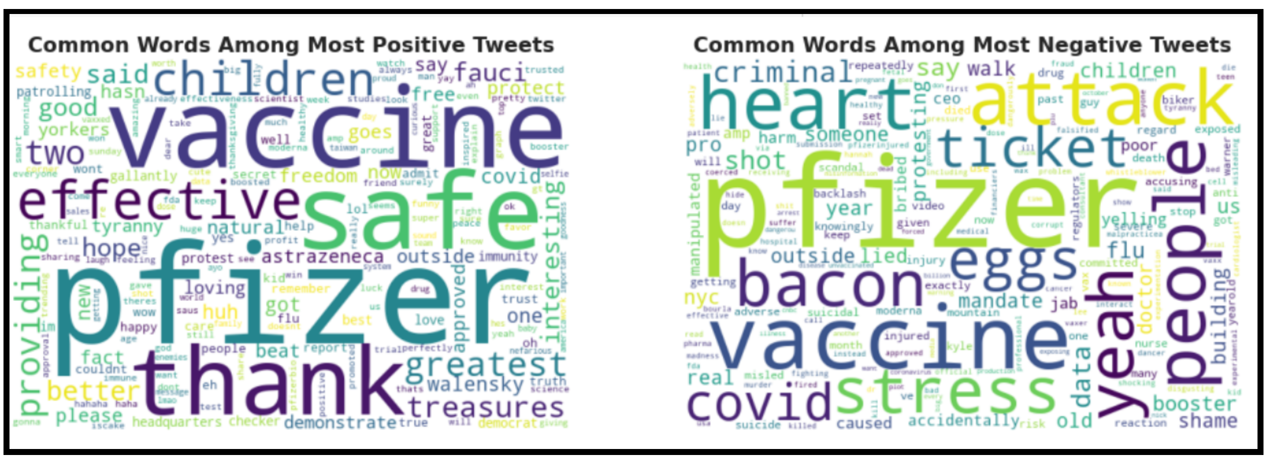
We can see from the distribution that ~60% of the tweets have neutral sentiments, ~30% of the tweets have negative sentiments and ~10% of the tweets have positive sentiments.



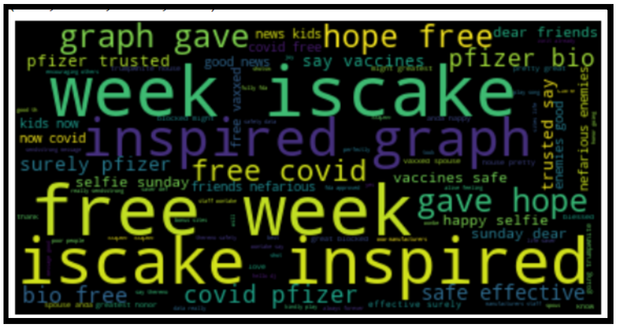
- Analyzing the most frequently occurring words: We have used WordCloud, which is a technique used to visualize frequently occurring words in text, where the size of words represents the frequency.

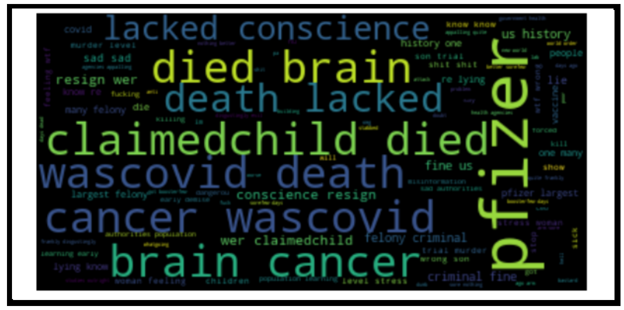


We can see from the visual above that the most frequent words in the dataset are: COVID, VACCINE, PFIZER, DOSE.

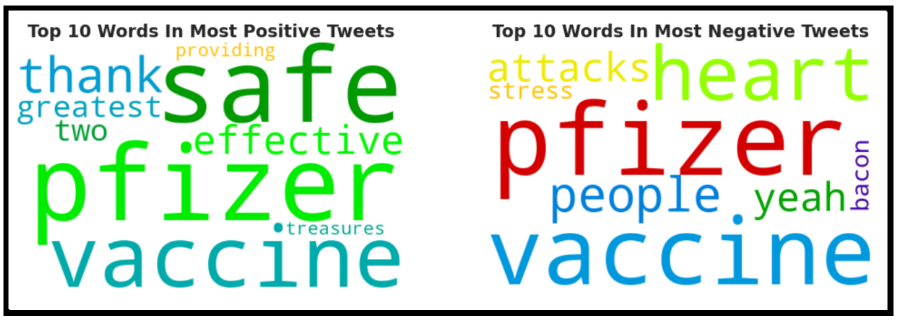


There are some common words that occur in most tweets. We can see that the most frequent words in the positive tweets contain words like : effective, safe, thank, better, treasures, good, greatest - which are usually related with positive sentiments. In contrast, the most frequent words in the negative tweets have words like: stress, attack, shame, injury, lied, protesting, manipulated, harm - which we relate with negative sentiments.





The above visual depicts some positive and negative tweets. We can see words with a positive sentiment attached to them in the first visual : hope, free, inspired. In contrast, we can see words with only negative sentiments in the second visual: "died", "cancer", "brain cancer", "lacked" and "death".



The above visual depicts the top 10 frequently occuring words in both the positive and negative tweets.

Top 10 words in positive tweets: safe, pfizer, vaccine, thank, effective, greatest, treasures, two, providing.

Top 10 words in negative tweets: pfizer, vaccine, heart, attacks, stress, people, yeah, bacon.

Building Predictive Models:

Our aim is now to build a predictive model that can classify tweets as positive or negative. We use pyspark to accomplish this. We have compared these results with the output of Vader Library to compute accuracies.

In addition to data cleaning in previous steps, we removed punctuations, converted text to lowercase, removed stop words and tokenized the text.

We start with a set of sentences. We split each sentence into words using Tokenizer. For each sentence (bag of words), we use HashingTF to hash the sentence into a feature vector. We use IDF to rescale the feature vectors; this generally improves performance when using text as features. Our feature vectors could then be passed to a learning algorithm.

Tokenizing: A tokenizer converts the input string to lowercase and then splits it by white spaces.

HashingTF: HashingTF is a transformer which takes a set of terms and converts those sets into fixed-length feature vectors. In text processing, a "set of terms" might be a bag of words. HashingTF utilizes the hashing trick. A raw feature is mapped into an index (term) by applying a hash function. Then term frequencies are calculated based on the mapped indices.

IDF: IDF is an Estimator which fits on a dataset and produces an IDFModel. The IDFModel takes feature vectors (generally created from HashingTF or CountVectorizer) and scales each feature. Intuitively, it down-weights features which appear frequently in a corpus.

We have split our dataset into a train set and test set with a 80:20 split.

- **Logistic Regression:** Logistic regression is a supervised machine learning technique for classification problems. Supervised machine learning algorithms train on a labeled dataset along with an answer key which it uses to train and evaluate its accuracy. The goal of the model is to learn and approximate a mapping function f(Xi) = Y from input variables {x1, x2, xn} to an output variable(Y). It is called supervised because the model predictions are iteratively evaluated and corrected against the output values, until an acceptable performance is achieved.

  We perform Multi-class logistic regression, where we assign a target label of 0 - Neutral Sentiment, 1 - Negative Sentiment and 2 - Positive Sentiment.

  Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.95      0.95      0.95     13356
         1.0       0.92      0.94      0.93      6877
         2.0       0.83      0.84      0.84      1662

    accuracy                           0.94     21895
   macro avg       0.90      0.91      0.91     21895
weighted avg       0.94      0.94      0.94     21895

[[12623   485   248]
 [  392  6449    36]
 [  212    49  1401]]
```

```
Accuracy Score: 0.9351
ROC-AUC: 0.9352
```

- **Naive Bayes:** Naive Bayes classifiers are a family of simple probabilistic, multiclass classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between every pair of features. Naive Bayes can be trained very efficiently. With a single pass over the training data, it computes the conditional probability distribution of each feature given each label. For prediction, it applies Bayes' theorem to compute the conditional probability distribution of each label given an observationThe most likely class is defined as the one having the highest probability.

  Classification Report:

```
              precision    recall  f1-score   support

         0.0       0.95      0.87      0.91     13356
         1.0       0.85      0.91      0.88      6877
         2.0       0.62      0.85      0.72      1662

    accuracy                           0.88     21895
   macro avg       0.81      0.88      0.83     21895
weighted avg       0.89      0.88      0.88     21895

[[11600  1018   738]
 [  481  6270   126]
 [  185    65  1412]]
```

```
Accuracy Score: 0.8807
ROC-AUC: 0.8836
```

- **Decision Tree:** Decision trees are supervised methods, so they need to be trained on some annotated data. Thus the general idea is the same as for any text classification: given a set of tweets (for instance represented as TFIDF vectors) together with their labels, the algorithm will calculate how much each word correlates with a particular label.

  For instance it might find that the word "excellent" often appears in tweets labeled as positive, whereas the word "terrible" mostly appears in negative tweets. By combining all such observations it builds a model able to assign a label to any document.

Classification Report:

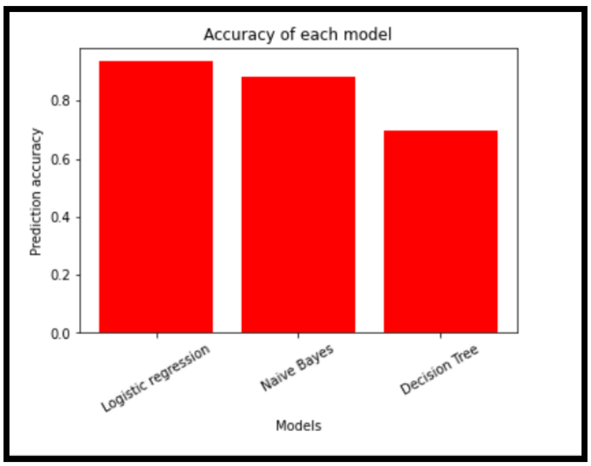
```
              precision    recall  f1-score   support

         0.0       0.67      0.99      0.80     13356
         1.0       0.98      0.27      0.42      6877
         2.0       0.73      0.13      0.23      1662

    accuracy                           0.70     21895
   macro avg       0.79      0.46      0.48     21895
weighted avg       0.77      0.70      0.64     21895

[[13249    42    65]
 [ 5025  1835    17]
 [ 1437     3   222]]
```

```
Accuracy Score: 0.6991
ROC-AUC: 0.6376
```

<u>Comparison:</u>

The overall accuracy is highest for Logistic Regression (93.5%), followed by Naive Bayes (88%) and then Decision Tree (70%).

Diving some more into the individual class predictions, we can see that the Logistic Regression model has done a better job at predicting the neutral and negative sentiment tweets (>90% accurate) than the positive sentiment tweets (83% accurate).



The Naive Bayes model has done a better job at predicting the neutral sentiment tweets (>90%), average at predicting the negative sentiment tweets(85%) and not very well for the positive sentiment tweets (62%).

Finally, the Decision Tree model has predicted the negative sentiment tweets most accurately (~98%), while the accuracies for the positive sentiment and neutral sentiment tweets are at 73% and 67% respectively.

Therefore, we can conclude that the Logistic Regression model is our best choice because it has higher individual class accuracies as well as higher overall accuracy in predicting the sentiments of the tweets.

There's another way to get term frequency for IDF (Inverse Document Frequency) calculation. It is a CountVectorizer in SparkML. Its aim is to help convert a collection of texts to vectors of token counts. During the fitting process, CountVectorizer will select the top vocabSize words ordered by term frequency across the corpus.

In comparison to HashingTF, there are a few differences. Apart from the reversibility of the features (vocabularies), there is an important difference in how each of them filters top features. In the case of HashingTF it is dimensionality reduction with possible collisions. CountVectorizer discards infrequent tokens. We now run the three prediction algorithms again - Logistic Regression, Naive Bayes and Decision Tree.

- **<u>Logistic Regression:</u> Classification Report:**

	precision	recall	f1-score	support
0.0	0.96	0.95	0.95	13356
1.0	0.93	0.94	0.93	6877
2.0	0.84	0.86	0.85	1662
accuracy			0.94	21895
macro avg	0.91	0.92	0.91	21895
weighted avg	0.94	0.94	0.94	21895

```
[[12657   466   233]
 [  387  6449    41]
 [  182    43  1437]]
```

Accuracy Score: 0.9382
ROC-AUC: 0.9383

- **<u>Naive Bayes:</u> Classification Report:**

	precision	recall	f1-score	support
0.0	0.96	0.86	0.91	13356
1.0	0.86	0.92	0.89	6877
2.0	0.58	0.88	0.70	1662
accuracy			0.88	21895
macro avg	0.80	0.89	0.83	21895
weighted avg	0.90	0.88	0.89	21895

```
[[11509   950   897]
 [  389  6329   159]
 [  151    49  1462]]
```

Accuracy Score: 0.8815
ROC-AUC: 0.8857

- **<u>Decision Tree:</u> Classification Report:**

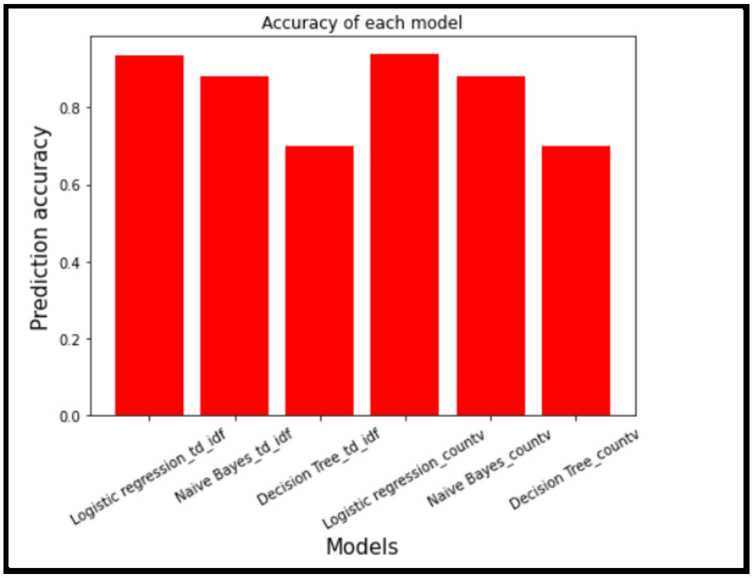
	precision	recall	f1-score	support
0.0	0.67	0.99	0.80	13356
1.0	0.97	0.27	0.42	6877
2.0	0.73	0.13	0.23	1662
accuracy			0.70	21895
macro avg	0.79	0.46	0.48	21895
weighted avg	0.77	0.70	0.64	21895

```
[[13246    45    65]
 [ 5006  1853    18]
 [ 1437     3   222]]
```

Accuracy Score: 0.6997
ROC-AUC: 0.6388

<u>Comparison:</u>

We can see from our findings that the Logistic Regression Model has the highest accuracy (93.8%), followed by Naive Bayes (88.1%) and finally Decision Trees (~70%). Similar to the earlier models, Logistic Regression and Naive Bayes perform better at classifying the neutral and negative sentiments more than the positive sentiments, while the Decision Tree model performs best at classifying the negative sentiments..



Model	Positive	Negative	Neutral	Overall
Logistic Regression - HashingTF+IDF	83%	92%	95%	93.50%
Logistic Regression - CountVectorizer+IDF	84%	93%	96%	93.80%
Naïve Bayes - HashingTF+IDF	62%	85%	95%	88%
Naïve Bayes - CountVectorizer+IDF	58%	86%	96%	88.10%
Decision Tree - HashingTF+IDF	73%	98%	67%	69.90%
Decision Tree - CountVectorizer+IDF	73%	97%	67%	69.90%

Overall, we can see that there is not a significant advantage of using HashingTF or CountVectorizer, both perform similarly. In terms of accuracy, Logistic Regression performs better than Naive Bayes and Decision Trees in classifying all the sentiments correctly.

Conclusion and future work

We found in our exploratory analysis that 60% of the tweets about the Pfizer vaccine are neutral, 30% negative and only 10% positive. Looking at these results one might assume that there are more negative words towards the vaccine as compared to positive But we would like to point out that even the neutral tweets that contain general information about the vaccine signifies acceptance towards the vaccine.

Here for our analysis we compared our output with vader library considering output of Vader library as absolute truth. Given more time we would like to include other libraries such as TextBlob and compare our results with both. There is also a possibility that for a certain number of tweets our models performed better than the Vader library and we would want to analyse that. Using a combination of sentiment analysis libraries and our own model we can come up with a more exhaustive model for sentiment analysis.

During modelling we found that logistic regression performs significantly better than Naive Bayes and Decision Tree. We would like to extend the modelling part to include other regression and ensemble methods and try to improve accuracy.

During exploratory analysis we analyzed the most common words and sentiment but a lot more in depth analysis is possible. The most important one being demographic analysis. We could not include that in our current scope because the 'Region' column in our tweets extract had more than 90% null values.

Overall it was a great learning experience for us, we learnt how to extract live data from twitter and we also got to work on such a relevant problem statement that is nowadays “talk of the town”. At the beginning of the project we were highly skeptical about Spark ML library but it turned out to be quite user friendly like scikit learn. We also learnt the concept of ML pipelines for the first time which is a very powerful tool for a flawless implementation of ML model.

Role of Team Members:

- Data Extraction - Vibhanshu
- Data Cleaning and Preparation - Pritha, Snehal
- Sentiment Analysis - Pritha
- Exploratory Analysis - Snehal, Vibhanshu
- Data Modelling- Pritha, Snehal & Vibhanshu

References

https://www.tweepy.org/

https://towardsdatascience.com/extracting-data-from-twitter-using-python-5ab67bff553a

https://towardsdatascience.com/sentiment-analysis-in-10-minutes-with-rule-based-vader-and-nltk-72067970fb71

Simplifying Sentiment Analysis using VADER in Python (on Social Media Text) | by Parul Pandey | Analytics Vidhya | Medium

GitHub - cjhutto/vaderSentiment: VADER Sentiment Analysis. VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on texts from other domains.

https://spark.apache.org/docs/latest/ml-features

Twitter sentiment analysis using Logistic Regression | by Kolamanvitha | Nerd For Tech | Medium

https://spark.apache.org/docs/latest/ml-classification-regression.html