Twitter Sentiment Analysis

1st Tina Ruchandani
Department of Computer Engineering
San Jose State University
Student-Id: 013767312
San Jose, CA

2nd Parineeta Kishore Patil
Department of Computer Engineering
San Jose State University
Student-Id: 013829114
San Jose, CA

3rd Nivetha Jayakumar
Department of Computer Engineering
San Jose State University
Student-Id: 013758667
San Jose, CA

Abstract**—Social media networks have become the central platform for people to share their opinions on anything in this world. Analysis of these data is essential to get visualization and a clear knowledge of the type of sentiments that people majorly share on networks. In this technical paper, we propose a sentiment analysis of the rival ride-sharing unicorns - Uber and Lyft. Twitter as a rich resource for sentimental analysis, we will show the high-level abstract of our implementation based on the tweets collected by the hashtag and mentions on Uber and Lyft.**

Keywords**—** Random Forest, Naive Bayes Classifier, K-means Clustering, Support Vector Machines, Uber, Lyft

I. Introduction

Information from micro-blogging websites have emerged into being the key to different sources of knowledge. This is caused by the pull of micro blogs which attracts people to express their opinions on subjects regarding current issues, celebrity news, complains and positive thoughts as well. In fact, to gain perspective on products they use micro blogs to their advantage to get a sense on a widespread sentiment of the products. Based on user reactions companies will send replies on micro blogs based on their sentiments. One difficulty in an overall sentiment is in building a technology to find and compile an overall sentiment. In this article, we read at a worldwide used micro blog called Twitter and to develop models in classifying tweets into positive, negative and neutral sentiment. The tweets of Uber and Lyft were gathered to do sentiment analysis on the opinions of the user. The raw data obtained is then pre-processed before training them into the machine learning models. The models used for our dataset were Naive Bayes, KNN, Random Forest and Support Vector Machines. Sentiment analysis runs into a similar set of problems as emotion recognition does before deciding what the sentiment of a given sentence is, we need to figure out what "sentiment" is in the first place. Is it categorical, and sentiment can be split into clear buckets like happy, sad, angry, or bored? Or is it dimensional, and sentiment needs to be evaluated on some sort of bi-directional spectrum? In addition to the definition problem, there are multiple layers of meaning in any human-generated sentence.

People express opinions in complex ways. Rhetorical devices like sarcasm, irony, and implied meaning can mislead sentiment analysis. The only way to understand these devices is through context such as knowing how a paragraph is started can strongly impact the sentiment of later internal sentences. Most of the current thinking in sentiment analysis happens in a categorical framework.

II. Methodology

Fig 1 shows the general methods used for Sentiment Analysis. There are broadly two categories of sentiment analysis.

1) Lexical Methods: These techniques employ dictionaries of words annotated with their semantic polarity and sentiment strength. This is then used to calculate a score for the polarity and/or sentiment of the document. Usually this method gives high precision but low recall.

2) Machine Learning Methods: Such techniques require creating a model by training the classifier with labeled examples. This means that you must first gather a dataset with examples for positive, negative and neutral classes, extract the features from the examples and then train the algorithm based on the examples. These methods are used mainly for computing the polarity of the document.

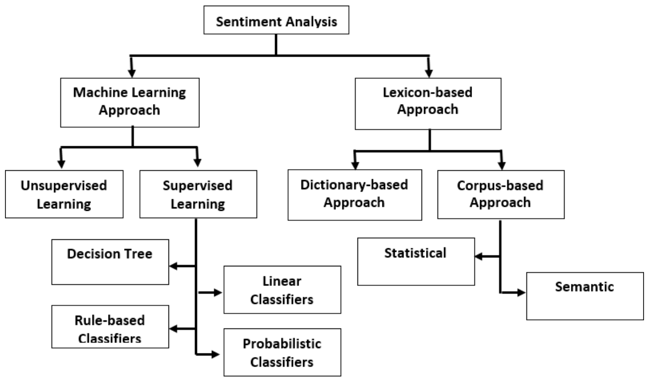


Fig. 1. Sentiment Analysis

Choice of the method heavily depends on the application, domain and language. Using lexicon-based techniques with large dictionaries enables us to achieve very good results. Nevertheless, they require using a lexicon, something which is not always available in all languages. On the other hand

Machine Learning based techniques deliver good results but they require obtaining training on labeled data.

Fig 2 shows the basic approach applied here for Sentiment Analysis. The idea is to use pre trained supervised machine learning models and apply on the top of unclassified data extracted from Twitter to conclude the sentiments.

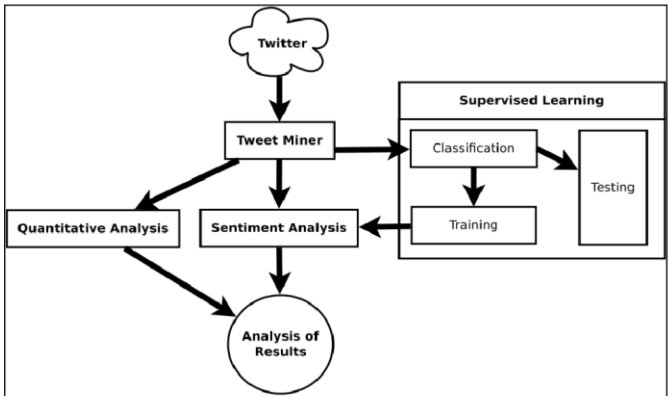


Fig. 2. Approach for Sentiment Analysis

III. Machine Learning Algorithms

1) Naive Bayes Classifier: Naive Bayes classifiers are mostly used in text classification due to better result in multi class problems and independence rule since they have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering like identifying spam e-mail and Sentiment Analysis like in social media analysis, to identify positive and negative customer sentiments.

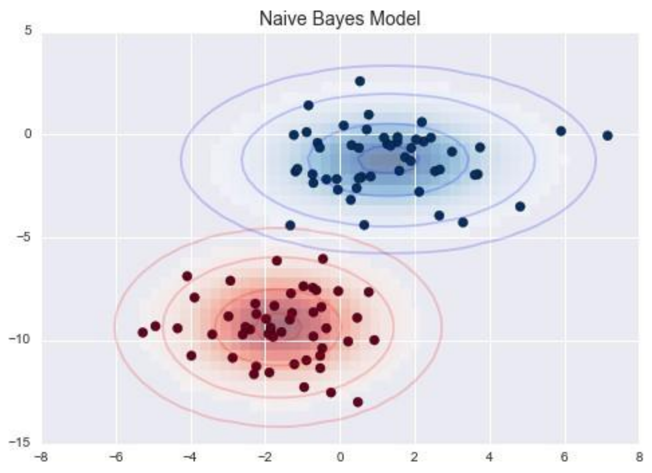


Fig. 3. Graph depicting Naive Bayes Classification

2) Random Forest Classifiers: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees habit of over fitting to their training set.

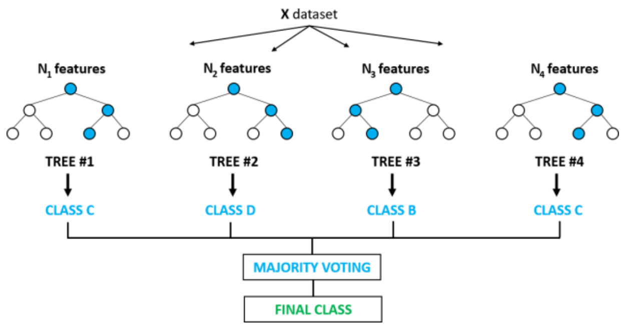


Fig. 4. Random Forest Algorithm

3) Support Vector Machines: SVM has been chosen for the classification in the experiments. The support-vector machine is a learning machine for two-group classification problems. It is used to classify the texts as positives or negatives. SVM works well for text classification due to its advantages such as its potential to handle large features. Another advantage is SVM is robust when there is a sparse set of examples and also because most of the problem are linearly separable

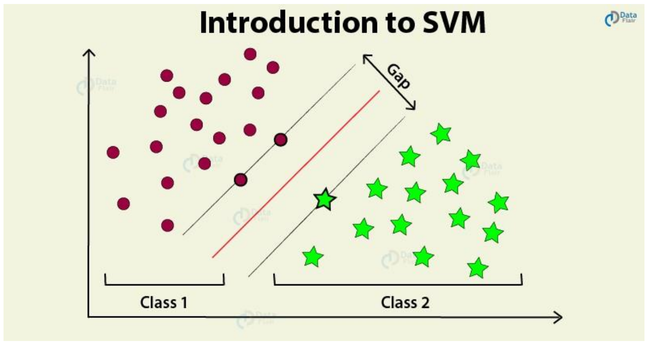


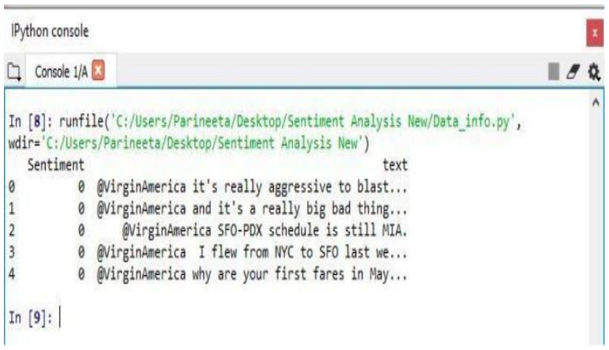
Fig. 5. Graph depicting Classification using Support Vector Machine

IV. DataSet Overview and Visualization

For this project, we have used three datasets. Our supervised dataset consists of two columns: sentiment and text. The dataset is amalgamation of various airline tweets and Kaggle datasets.

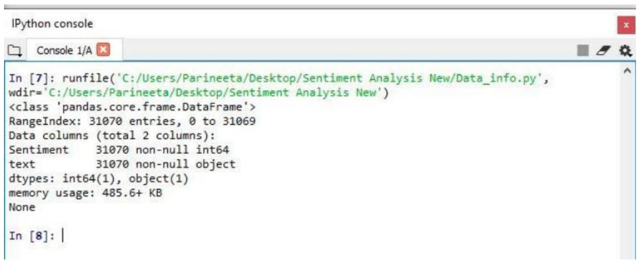
The overall number of data instances are about 31k. The features in all datasets belong to one of the types: numerical values and object data types. The metadata and data samples are shown in fig 6 and 7.

For Uber and Lyft datasets, we have used Tweepy API to fetch various tweets from Twitter that has Uber and Lyft hashtags. We have collected about 5k data instances for both. The dataset is unlabeled and has four columns: timestamp (tweet creation time), tweet text, username and all the hashtags that



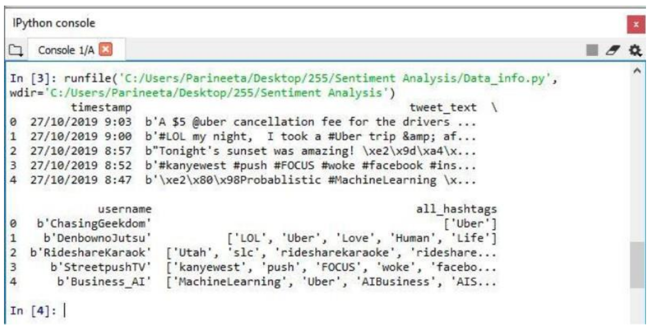
```
In [8]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/Data_info.py',
wdir='C:/Users/Parineeta/Desktop/Sentiment Analysis New')
   Sentiment                                           text
0          0  @VirginAmerica it's really aggressive to blast...
1          0  @VirginAmerica and it's a really big bad thing...
2          0      @VirginAmerica SFO-PDX schedule is still MIA.
3          0  @VirginAmerica  I flew from NYC to SFO last we...
4          0  @VirginAmerica why are your first fares in May...

In [9]: |
```

Fig. 6. Sample Data



```
In [7]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/Data_info.py',
wdir='C:/Users/Parineeta/Desktop/Sentiment Analysis New')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31070 entries, 0 to 31069
Data columns (total 2 columns):
Sentiment    31070 non-null int64
text         31070 non-null object
dtypes: int64(1), object(1)
memory usage: 485.6+ KB
None

In [8]: |
```

Fig. 7. Metadata for Supervised dataset



```
In [3]: runfile('C:/Users/Parineeta/Desktop/255/Sentiment Analysis/Data_info.py',
wdir='C:/Users/Parineeta/Desktop/255/Sentiment Analysis')
         timestamp                                       tweet_text  \
0  27/10/2019 9:03  b'A $5 @uber cancellation fee for the drivers ...
1  27/10/2019 9:00  b'#LOL my night,  I took a #Uber trip &amp; af...
2  27/10/2019 8:57  b"Tonight's sunset was amazing! \xe2\x9d\xa4\x...
3  27/10/2019 8:52  b'#kanyewest #push #FOCUS #woke #facebook #ins...
4  27/10/2019 8:47  b'\xe2\x80\x98Probablistic #MachineLearning \x...

             username                                       all_hashtags
0   b'ChasingGeekdom'                                          ['Uber']
1    b'DenbownoJutsu'            ['LOL', 'Uber', 'Love', 'Human', 'Life']
2  b'RideshareKaraok'  ['Utah', 'slc', 'ridesharekaraoke', 'rideshare...
3     b'StreetpushTV'  ['kanyewest', 'push', 'FOCUS', 'woke', 'facebo...
4       b'Business_AI'  ['MachineLearning', 'Uber', 'AIBusiness', 'AIS...

In [4]: |
```

Fig. 8. Uber data collected from Twitter

the corresponding tweet has. Fig 8 shows a sample of data collected from Uber.

Following figure shows distribution of the number of words in a tweet per target class (negative, neutral and positive respectively).

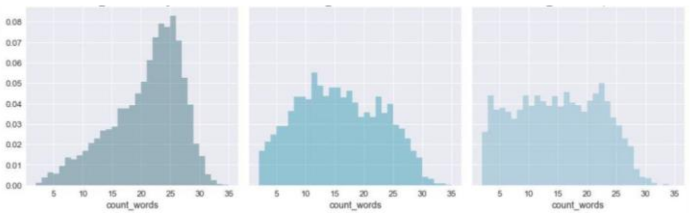


Fig. 9. Distribution of words

Most common words in the dataset as visible from Fig.10

V. Data Preprocessing

Data preprocessing is a process of cleansing data for acquiring more knowledge about it. We are surrounded by

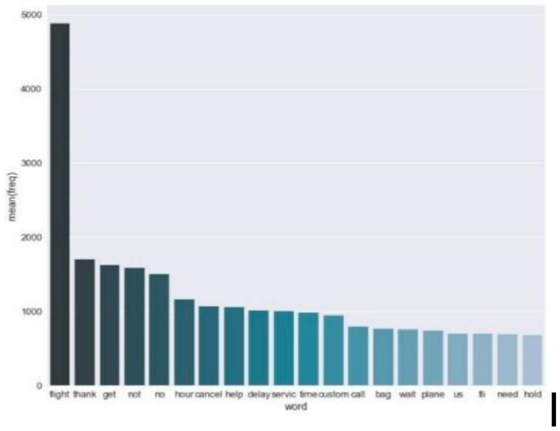


Fig. 10. Frequency of common words

immense data for which we lack knowledge as there exists a lot of inconsistencies in data. The data is available in various formats such as files, databases or comma-separated values. The data obtained can have missing values, incorrect data or misspellings during data entry. These inconsistencies are reviewed, and the outliers are removed before the data is exposed to further processing. Data preprocessing certainly increases the quality and readability of the data. Every model requires its own kind of data preprocessing, although a basic step should be performed before applying any model. Following are the steps followed to preprocess the data in this project,

- Removing missing values
- Removing mentions
- Replacing emojis with the text they represent (using emoji module)
- Removing any URLs
- Removing punctuation
- Converting all the words to lowercase
- Removing HTML tags
- Removing stop words from the tweet
- Performing stemming

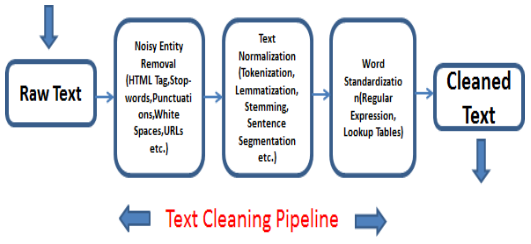
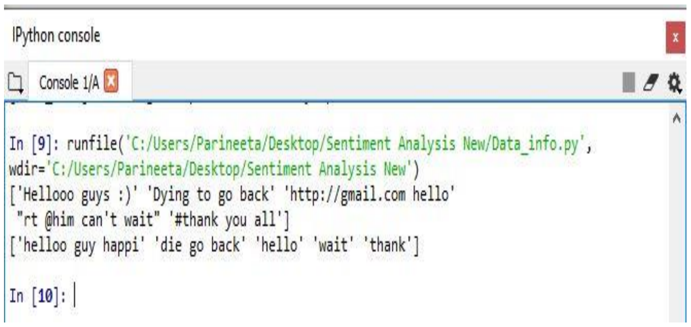

Raw Text
Noisy Entity Removal (HTML Tag,Stop-words,Punctuations,White Spaces,URLs etc.)
Text Normalization (Tokenization, Lemmatization, Stemming, Sentence Segmentation etc.)
Word Standardization (Regular Expression, Lookup Tables)
Cleaned Text
Text Cleaning Pipeline

Fig. 11. Dataset after preprocessing

After completing data preprocessing the dataset will look as shown in fig 10.

VI. Experimental Results

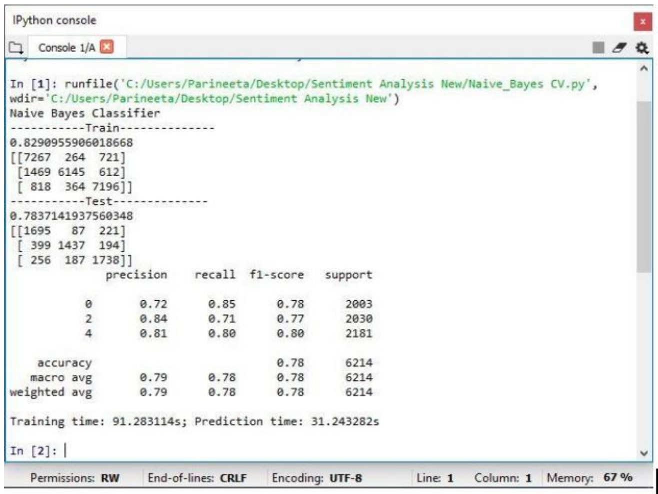
1) Accuracy and Precision Results for Models: Precision and Recall are recognized as evaluating indicator about the recommendation effect of the recommended system. The biggest difference between the models we are building from a "feature"



```
In [9]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/Data_info.py',
wdir='C:/Users/Parineeta/Desktop/Sentiment Analysis New')
['Hellooo guys :)' 'Dying to go back' 'http://gmail.com hello'
 "rt @him can't wait" '#thank you all']
['helloo guy happi' 'die go back' 'hello' 'wait' 'thank']

In [10]: |
```

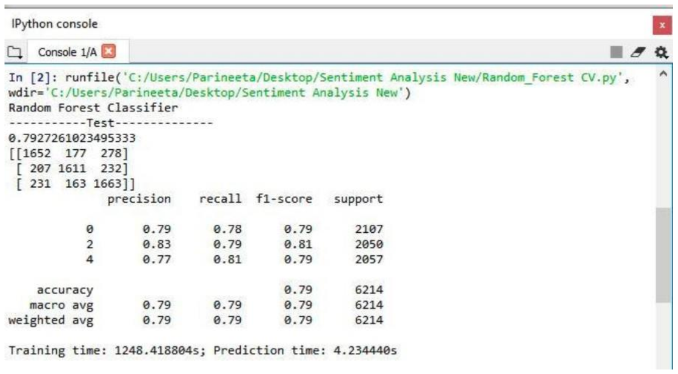
Fig. 12. Dataset after Pre-Processing

point of view is that Naive Bayes treats them as independent, whereas SVM looks at the interactions between them to a certain degree.



```
In [1]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/Naive_Bayes CV.py',
wdir='C:/Users/Parineeta/Desktop/Sentiment Analysis New')
Naive Bayes Classifier
-----------Train--------------
0.8290955906018668
[[7267  264  721]
 [1469 6145  612]
 [ 818  364 7196]]
-----------Test--------------
0.7837141937560348
[[1695   87  221]
 [ 399 1437  194]
 [ 256  187 1738]]
              precision    recall  f1-score   support

           0       0.72      0.85      0.78      2003
           2       0.84      0.71      0.77      2030
           4       0.81      0.80      0.80      2181

    accuracy                           0.78      6214
   macro avg       0.79      0.78      0.78      6214
weighted avg       0.79      0.78      0.78      6214

Training time: 91.283114s; Prediction time: 31.243282s

In [2]: |
```

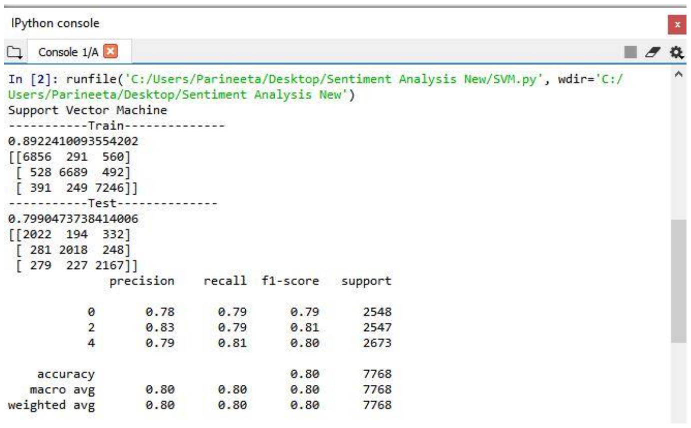
Fig. 13. Naive Bayes Results



```
In [2]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/Random_Forest CV.py',
wdir='C:/Users/Parineeta/Desktop/Sentiment Analysis New')
Random Forest Classifier
-----------Test--------------
0.7927261023495333
[[1652  177  278]
 [ 207 1611  232]
 [ 231  163 1663]]
              precision    recall  f1-score   support

           0       0.79      0.78      0.79      2107
           2       0.83      0.79      0.81      2050
           4       0.77      0.81      0.79      2057

    accuracy                           0.79      6214
   macro avg       0.79      0.79      0.79      6214
weighted avg       0.79      0.79      0.79      6214

Training time: 1248.418804s; Prediction time: 4.234440s
```

Fig. 14. Random Forest Classifier Results

2) K-fold Results for Models: Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. Fig 18,19 and 20 shows the k-fold graphs obtained for each model respectively.



```
In [2]: runfile('C:/Users/Parineeta/Desktop/Sentiment Analysis New/SVM.py', wdir='C:/
Users/Parineeta/Desktop/Sentiment Analysis New')
Support Vector Machine
-----------Train--------------
0.8922410093554202
[[6856  291  560]
 [ 528 6689  492]
 [ 391  249 7246]]
-----------Test--------------
0.7990473738414006
[[2022  194  332]
 [ 281 2018  248]
 [ 279  227 2167]]
              precision    recall  f1-score   support

           0       0.78      0.79      0.79      2548
           2       0.83      0.79      0.81      2547
           4       0.79      0.81      0.80      2673

    accuracy                           0.80      7768
   macro avg       0.80      0.80      0.80      7768
weighted avg       0.80      0.80      0.80      7768
```

Fig. 15. Support Vector Machine Results

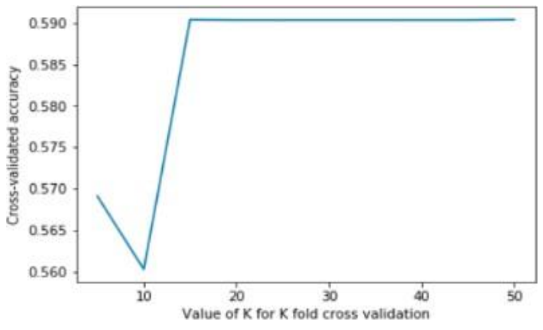


Fig. 16. Naive Bayes K-fold graph

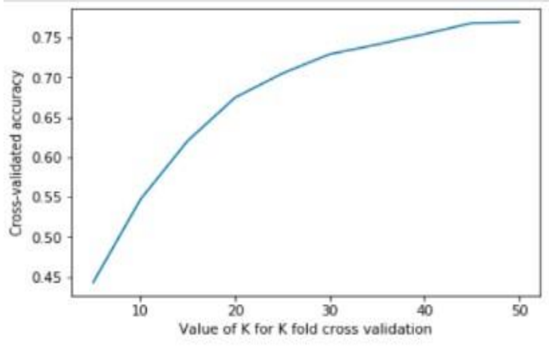


Fig. 17. Random Forest K-fold graph

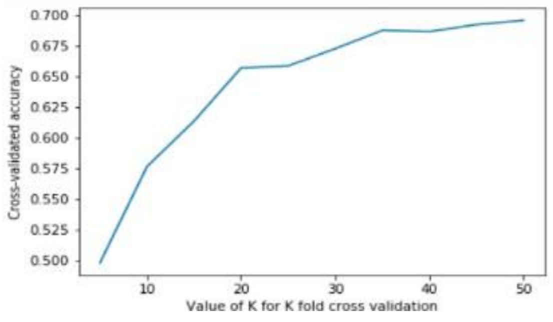
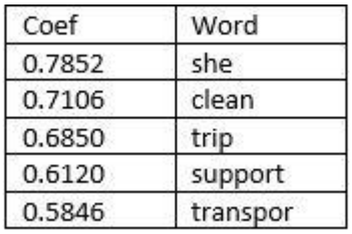


Fig. 18. SVM K-fold graph

All experimentation is done on Windows 8.1 with Python3 and Ubuntu 18.03 with Python3.

VII. Data Analysis

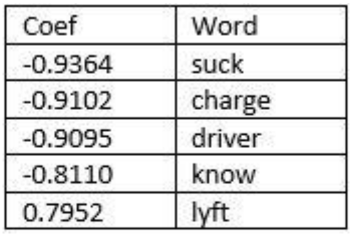
A look at the word cloud generated from Uber and Lyft's dataset gives us following points:



Coef	Word
0.7852	she
0.7106	clean
0.6850	trip
0.6120	support
0.5846	transpor

Fig. 19. Most common words for positive data-samples of Uber

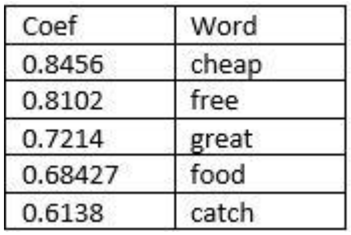
- "Clean", "Support" are common words that are expected in the positive sentiment.
- "She" is a very unexpected word to appear in this list. May be customers enjoy their ride more with female Uber drivers.
- "Trip", "Transport" might have been fetched from the tweets were customers express their positive opinion about trip/transportation with Uber.



Coef	Word
-0.9364	suck
-0.9102	charge
-0.9095	driver
-0.8110	know
0.7952	lyft

Fig. 20. Most common words for negative data-samples of Uber

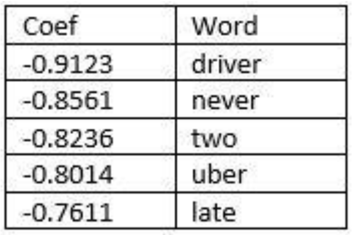
- "Suck" is the most negatively correlated word which represents customers that are not fond of Uber.
- "Charge" represents customers that feel Uber is slightly overpriced.
- "Driver" indicates that customers might have issue with behavior or punctuality of Uber drivers.
- "Lyft" is an interesting word to appear in this list. Customers might be feeling that Lyft which is a competitor of Uber is better.



Coef	Word
0.8456	cheap
0.8102	free
0.7214	great
0.68427	food
0.6138	catch

Fig. 21. Most common words for positive data-samples of Lyft

- "Cheap", "Great" are common words that are expected in the positive sentiment.
- The dataset contains various datapoints where customers advertised their promo code. Such tweets have "free ride" words in them.



Coef	Word
-0.9123	driver
-0.8561	never
-0.8236	two
-0.8014	uber
-0.7611	late

Fig. 22. Most common words for positive data-samples of Lyft

- "Driver" indicates that customers might have issue with behavior or punctuality of Lyft drivers
- "Uber" is an interesting word to appear in this list. Customers might be feeling that Uber which is a competitor of Lyft is better.
- "Late" indicates that Lyft drivers might be reaching late at pick-up/ drop locations.

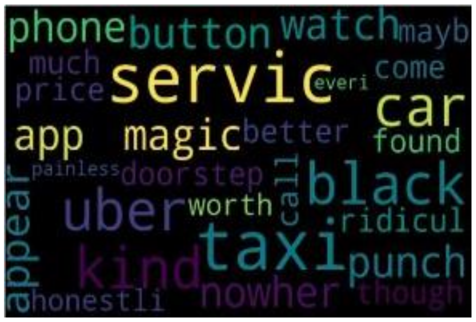


Fig. 23. Word Cloud for Uber

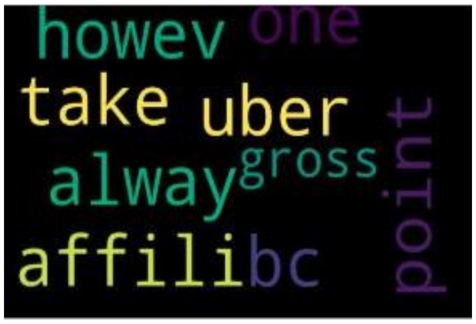


Fig. 24. Word Cloud for Lyft

- Fig 23 and 24 shows most common words in the Uber and Lyft dataset respectively. These words are not specific to a label but are generated considering overall dataset.
- Word clouds add simplicity and clarity. The most used keywords stand out better in a word cloud.
- Word clouds are a potent communication tool. They are easy to understand, to be shared and are impactful.
- Word clouds are visually engaging than a table data.

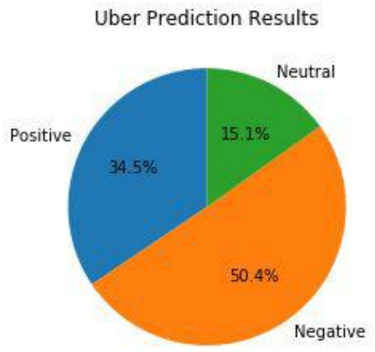


Fig. 25. Pie chart for Uber

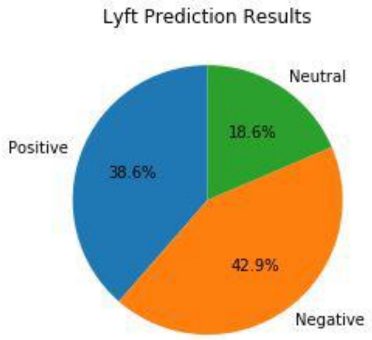


Fig. 25. Pie chart for Lyft

As visible from the pie-charts, Lyft gets significantly better reviews and sentiment ratings than Uber does. Although, Uber is more profitable and popular across most markets where it directly competes with Lyft, the latter's ability to keep its customers more satisfied could pay off in the long term. It's certainly something the Lyft management tries to promote the idea that when customers join Lyft, they're not simply joining another ridesharing company, they're joining a community.

VIII. Problems Faced

Following are some problems faced during the execution of this project:

- Comparing the sparsity directly to the other datasets, such as iTunes store and Google store, our dataset was extremely sparse and noisy.
- Many of the tweets were sarcastic in nature and irrelevant to the ride sharing companies.
- Applying decision tree algorithms like Random forest was quite a challenge since it consumed memory, CPU, power and would take days to run making it difficult to fine tune.
- Precision value is average, which is most likely due to the sparsity of our dataset.

IX. Future Work

As a future work for this project, we plan to include sarcasm detection and other algorithms like clustering to improve the accuracy. Also, we wish to include libraries like Textblob and Vader to improve the performance. Another future task is to identify and automatically remove the irrelevant tweets and hashtags.

X. Conclusion

Our work gives demonstration of the potential of machine learning models for getting insights from the data. We used three machine learning models: Random Forest, Support Vector Machine and Naive Bayes to compare widely famous public transportation companies, Uber and Lyft. We applied this models on supervised dataset to serialize object which we later applied on unsupervised Uber and Lyft datasets. We performed different experiments over k-fold cross validation techniques to evaluate our models and induce analytical results. Further, we used voting classifier to boost the overall performance. We were able to achieve 75-85 percentage accuracy. This work can be further extended by collecting more real-world data and applying advanced algorithms to enhance data pre-processing.

XI. Acknowledgment

We would like to thank Professor Christian Zuniga for guidance and support in successfully executing this project. We would also like to thank our team members who worked long hours and on weekends to make this happen.

References

[1] M Hu and B Liu. 2004. Mining and summarizing customer reviews. KDD.
[2] S M Kim and E Hovy. 2004. Determining the sentiment of opinions. Coling.
[3] B. Pang and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity analysis using subjectivity summarization based on minimum cuts. ACL.
[4] T. Wilson, J. Wiebe, and P. Hoffman. 2005. Recognizing contextual polarity in phrase level sentiment analysis. ACL.
[5] Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In WWW '03: Proceedings of the 12th nternational conference on World Wide Web, pages 519–528, New York, NY, USA. ACM.
[6] Alec Go, Lei Huang, and Richa Bhayani. 2009. Twitter sentiment analysis. Final Projects from CS224N for Spring 2008/2009 at The Stanford Natural Language Processing Group.
[7] Jotheeswaran, J., Kumaraswamy, S." Opinion Mining Using Decision Tree Based Feature Selection Through Manhattan,"Journal of Theoretical and Applied Information Technology, 2013, 58(1).
[8] Liu, B. Synthesis Lectures on Human Language Technologies: Sentiment Analysis and Opinion Mining. California: Morgan Claypool Publishers, 2012.
[9] Narayanan, V., Arora, I., Bhatia, A. (n.d.)." Fast and accurate sentiment classification using an enhanced Naive Bayes model," Intelligent Data Engineering and Automated Learning IDEAL 2013 Lecture Notes in Computer Science, 2013, 8206, 194-201