Sentiment Analysis by Combining Lexicon-based and Machine Learning Methods

Satyajit Narayanan, Jagadeesh Hariharan, Riddhiman Sherlekar, Eshaan Kirpal, Venkatesh Nayak, Harsh Mehta
North Carolina State University
North Carolina, Raleigh, USA

Dr. Ranga Raju Vatsavai
North Carolina State University
North Carolina, Raleigh, USA

***Abstract*— In this report, we present a supervised lexicon-based approach to extracting sentiments from a tweet. We provide a comparative study of existing techniques for opinion mining by combining machine learning and lexicon-based approaches, together with evaluation metrics. This is done by calculating the Semantic Orientation (SO) using dictionaries of words annotated with their polarity, strength and incorporates intensification and negation. Implementation of the algorithm for Decision Tree and Naïve Bayes in order to assign sentiment to each of the tweets has also been done.**

Keywords - Sentiment classification, Lexicon, Machine Learning, Naïve Bayes, Decision Tree, SVM

I. INTRODUCTION

With the advent of social media and increased connectivity among people with each other, increasing number of people have begun expressing their feelings, opinion and attitude over Internet, which increase the amount of user generated reviews containing rich opinion and sentiment information. Armed with increased computing power and applications of sophisticated machine learning algorithms, corporations around the world have started to harness the power of textual information to better understand sentiments of their customers or potential customers to make smarter data-driven decisions and improve their services.

Sentiment Analysis is the process of determining whether a piece of writing is positive, negative or neutral. It's also known as opinion mining, deriving the opinion or attitude of a speaker. There exist two main approaches to the problem of extracting sentiment automatically; the lexicon-based approach which involves calculating orientation from the polarity of words or phrases and the text classification approach involves building classifiers from labeled instances of texts or sentences. The latter approach is essentially a supervised classification task and could also be described as a statistical or machine-learning approach. We follow the first method, in which we use dictionaries of words annotated with the word's semantic orientation, or polarity to assign polarities to words and sentences and used these as features in our machine-learning approach [1]. The flow of our proposed method is shown in Figure 1.

II. DATA PREPROCESSING

Data used for this study is the Twitter US Airline dataset which has been acquired from an open source consisting of 14,640 labelled tweets. The data is in CSV format with

Figure 1: Flow of the proposed method

emoticons removed. From the csv file which has sixteen fields, the fields important for this study is extracted. They are – tweets and labels. Data has three distinct polarity values – negative represented as -1, neutral represented as 0.5 and positive, represented as 1. A visualization of the input dataset with the frequency of the class label in figure 2.

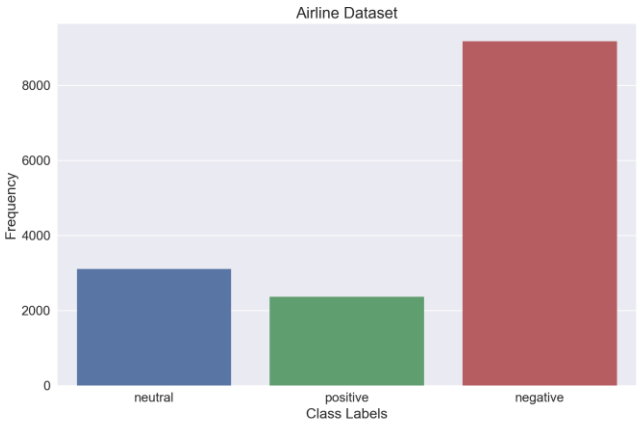


Figure 2: Frequency of the class labels of input dataset

We have used the python based Natural Language Toolkit (NLTK) to perform many of the pre-processing tasks. The following are some of the preprocessing methods that have been performed before the actual sentimental analysis.

A. Tokenizatoin

The incoming text is broken down into tokens. The delimiter used for tokenization is whitespace, however other delimiter can also be used. We should also keep in mind that the symbols including exclamation-mark, period and question-mark are also considered as separate tokens.

B. Case conversion of tokens

The tokens can be in Lowercase or Uppercase, for the convenience of the algorithm and to avoid any variability due to cases, all the tokens are first converted to lower case [2].

C. Stemming and Lemmatization

Since a word has many forms (such as drive, driving, driven) but it still it carries the same meaning. Both stemming, and lemmatization are very similar functions in which a word is reduced to 'stem' in stemming and 'lemma' in lemmatization. Through stemming, we can replace the words with their root words.

D. Stop-words

Stop words are commonly used words, which generally are structure words of a sentence such as articles, prepositions, etc. We used the stop-words corpus of NLTK and removed them from the sentence, so that we can focus more on the important words. It is a common assumption that the removal of stop-words from a manually prepared commonly used words, we can focus on the important words [4].

E. Part of Speech (POS) Tagging

Part of Speech (POS) tagging is the process of assigning a part of speech to each word in a sentence as noun, verb, adjective, adverb, etc. Part-of-speech (POS) information is commonly exploited in sentiment analysis and opinion mining. One simple reason holds for general textual analysis, not just opinion mining: part-of-speech tagging can be considered to be a crude form of word sense disambiguation [5]. POS are useful because of the large amount of information they give us about a word and its neighbors. There is no common opinion about whether POS tagging improves the results of classification or not. [6] got positive results using POS tagging, while [7] reported the decrease in performance. In the current study we have used the NLTK's part-of-speech (POS) tagging. Given a word (the tokenized input) in a tweet, NLTK maps it to its part-of-speech.

III. LEXICON BASED APPROACH

The lexicon-based approach involves calculating orientation for a sentence from the semantic orientation of words or phrases in the document. We use dictionaries of words annotated with the word's semantic orientation, or polarity. [1].

A. Dictionary Creation

There are two primary assumptions with respect to the calculation of semantic orientation: that individual words have a prior polarity and that the semantic orientation can be expressed numerically. Dictionaries for lexicon-based approaches can be created manually, as authors describe in this article [8] or automatically. How a dictionary is created affects the overall accuracy of subsequent results. Automatically or semi-automatically created dictionaries have some advantages like the occurrence of novel words. But it lacks stability, therefore, we decided to create manual one and considered valence shifters (intensifiers and down-toners) too.

We have used the polarity values of the Semantic Orientation Calculator (SO-CAL) [1] and positive and negative words from the General Inquirer dictionary [10]. The complete dictionary contains adjectives, nouns, verbs, and adverbs. In addition to these, we also incorporated words from SentiWordNet [11] and Opinion Lexicon [12]. After extracting the sentiment-bearing words, we use them to calculate semantic orientation.

We have a collection of adjectives, verbs, adverbs and nouns with pre-defined semantic orientations ranging from +5 to -5 that denotes the degree of orientation for each these words. Examples are shown in Table 1.

Table 1: Examples from the dictionary.

Word	SO Value
abominable	−5
absurd	−3
acrimonious	−2
satisfactorily	1
sweeten	2
panic	-4

In order to account for the intensification, we created a list of intensifier words from Taboada et al. [1]. These words each have an intensification percentage value associated with it; amplifiers are positive, whereas down-toners are negative as shown in Table 2.

Table 2: Percentages for some intensifiers

Intensifier	Modifier (%)
minor	-30
somewhat	−30
only	-50
really	15
tremendously	40
considerable	30
utmost	100

B. Polarity based on Parts of Speech

While creating features for the model, we assumed that the occurrences of certain parts of speech associated with the words in the tweet would be an indicator of the sentiment of the tweet, mainly adjectives and nouns [13]. The POS tagging, which was obtained from NLTK and then bucketed into four parts: adjectives, verbs, adverbs and nouns.

For each of these four buckets a distinct polarity was calculated and used as a feature in that model. For example, the individual semantic orientation scores for each adjective in a tweet are added together to form the polarity of adjectives for that tweet.

Upon performing univariate and bivariate analysis on each of the variables, we saw clearly that adverbs and verbs weren't good indicators of sentiment. Adjectives and Nouns, although

better predictors, alone weren't enough. Thus, we found the summation of the polarities of all the words in a sentence to get an overall polarity of a tweet based on the semantic orientation. Using the polarities generated, the class labels are predicted, and the lexicon-based approach is validated. This unsupervised method of classification yielded an accuracy of 46%. The predicted class labels are depicted in Figure 3, which can be compared with Figure 1 for how well the lexicon-based approach has performed. From Figure 3, we can infer that the dictionary-based approach may fail to find opinion words with domain and context specific orientations [14]. Supervised learning methods on the other hand are in general more accurate, but much slower than lexicon-based methods [15]. Hence, in the next section, some more features which will be useful for supervised learning are created.

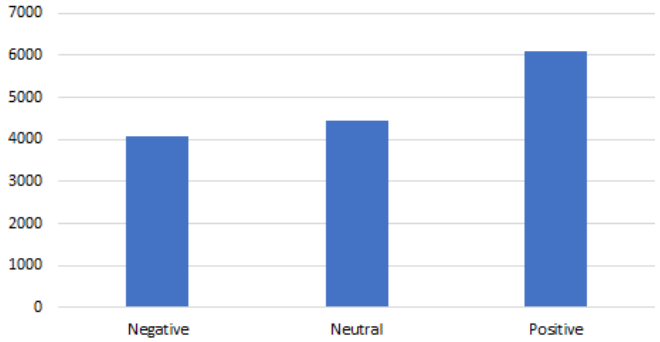


Figure 3: Classification Based on Lexicon Method.

IV. FEATURES FOR THE MODEL

A. *Number of occurrences of frequently occuring negative words and abusive words*

We decided to incorporate variables that include information about the proportion of positive and negative words, the quantity of interrogation and exclamation marks, obscene language [16]. We used to the negative word list from Opinion Lexicon [4] as the base to create our new feature, number of occurrences of frequently occurring negative words. Upon analysis of the variable, we observed a very poor correlation with our predictor variable. To improve upon that, we added many domain specific negative words into the list.

Similarly, another feature using a list of abusive words was created. On performing bivariate analysis with the frequency of negative words, we observed a high interaction between them. Thus, we decided to merge the two variables by merging the lists. This variable, in essence is the count of frequently occurring negative or abusive words in a tweet.

B. *Number of occurrences of frequently occuring positive words*

Similar to the list of negative words from Opinion Lexicon [12], there exists a list of positive words too which we used to create the feature. Domain specific positive words were added into this list. This feature, is the count of frequently occurring positive words in a tweet.

V. FEATURE SELECTION

The performance of any machine learning algorithm depends on the set of features fed to the model. Therefore, feature selection is one of the major factor which structures how good a model can perform. As stated in Section IV, features considered in our study are adjective, noun, adverb, verb, polarity, number of positive words, number of negative words. The correlation matrix can be formed to find out the best relationship between these complete set of features and the class label [17].

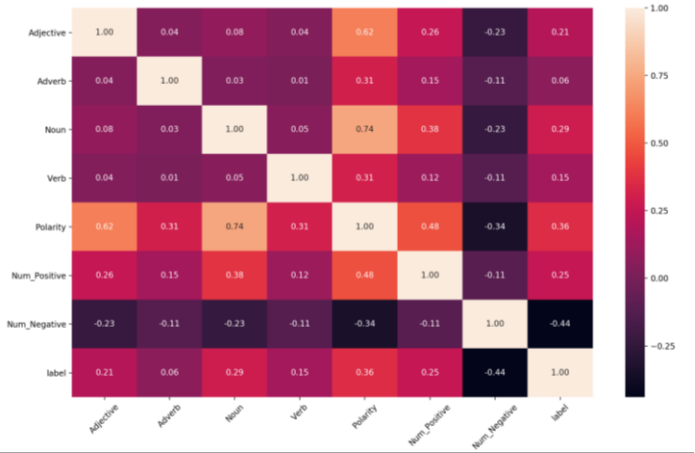


Figure 4. Visualization of the correlation between variables

As shown in Figure 4, the list of the top variables with respect to the class label are number of negative words, polarity, noun, number of positive words and adjective.

From the study of survey of different features selections, we have used the wrapper features selection process [19]. The algorithms of this method are wrapped around the adaptive systems providing them subsets of features and receiving their feedback (usually accuracy). These wrapper approaches are aimed at improving results [20].

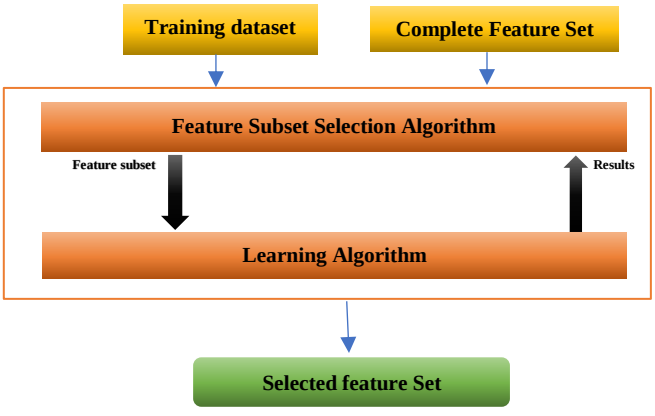


Figure 5. Features Selection Process

After performing the wrapper feature selection and using multiple classifiers such as decision tree, adaboost, SVM, etc., the selected feature set were polarity, number of positive words and number of negative words.

VI. COMPARITIVE STUDY

A study of the various supervised classifiers is initially completed to find out the best possible classifier, which can be implemented from scratch. This was done by making use of the scikit learn library which is the standard machine learning library of python. The data was first split into training and test data with a split percent of 70-30. Then various classifiers were run on the selected feature set {polarity, number of positive words, number of negative words}. The performance of the classifiers was compared based on the accuracy measure.

We performed a comparative study and obtained the results as in Figure 6 and 7.

```
/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7
Accuracy of LogisticRegressionis 0.614526411658
Accuracy of KNeighborsClassifieris 0.659608378871
Accuracy of SVCis 0.690346083789
Accuracy of DecisionTreeClassifieris 0.706511839709
Accuracy of RandomForestClassifieris 0.704918032787
Accuracy of AdaBoostClassifieris 0.706056466302
Accuracy of GaussianNBis 0.677823315118

Process finished with exit code 0
```

Figure 6: Program snippet of different model performances

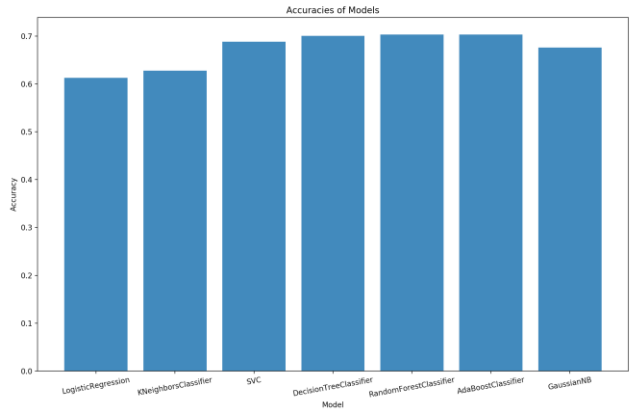
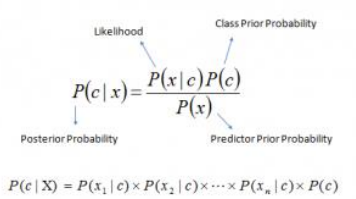


Figure 7: Visualization of different model performances

VII. METHDOLOGY & IMPLEMENTATION

A. Naïve Bayes Classifier

Naïve Bayes is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'. The equation for this is given below:



$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

The implementation was carried out in Python and the packages used were numpy, pandas, sklearn, io and math.

First the dataset was split based on predefined ratio. The larger portion was assigned as the training data on which the model is trained, and the rest is the test data. We used the Gaussian function to estimate the probability of a given feature value, given the mean and standard deviation for the feature estimated from the training data.

Classification involved calculating the likelihood that the given data instance belonged to each predictor class and then selecting the class with the largest likelihood as the prediction to the test entry. Running the algorithm on the Twitter US Airline dataset with the initial four variables, observed very low accuracy values, 48%. After incorporating the new variables, and feature selection, we observed accuracies of 54% – 55% as shown in Figure 8.

```
14640 rows are split into 13176 rows for training and 1464 rows for testing
Accuracy obtained = 54.37158469945356%
```

Figure 8: Naïve Bayes output

We did not get the level of accuracy we desired, hence we chose to implement another classification algorithm.

B. Decision Tree Classifier

We have implemented the ID3 algorithm of the Decision Tree. The basic idea of this algorithm is to construct the decision tree based on the information gain criteria. The algorithm determines the class labels of the object by testing different properties. Starting from the parent node, at each node a property is tested and based on the minimum entropy, the decisions are made. The implementation was carried out in Python and the packages used were numpy, pandas, sklearn.

This process is recursively performed to develop a decision tree. The node where no further splits are possible are then referred to as a leaf node. The ID3 algorithm uses a greedy search. When tested on the complete dataset, the maximum depth of the decision tree was 9 and the maximum width of the decision tree was 45.

The main data is first divided into train and test datasets in the proportion 70% and 30%. The ID3 algorithm is executed on the train data. Entropy is used as the measure to decide the best possible attribute and split value to split the dataset. As the attribute values in the data is of the type float, the entropy of an attribute is calculated for all the points ranging between minimum and maximum points. The best split value and entropy for each attribute is calculated and then the attribute having highest gain is selected for splitting the dataset. A zero or negative value of gain will result in termination of the present dataset and creating a leaf node to which a label will be assigned. If the gain is positive, then the dataset is split according to the attribute and split value; the algorithm is re-executed for the split datasets until termination occurs. At the end of execution, the rules are stored through a class in an object. The rules are extracted from the object which is then used to assign class to the records of test data. The test data is converted into a

dictionary where each record is a key and the value of key is the class that will be assigned using the rules from the object.

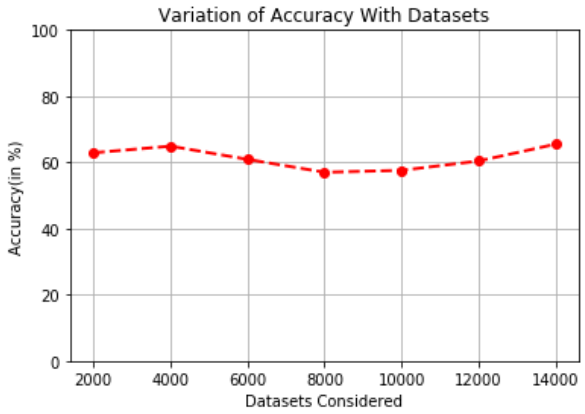


Figure 9: Variation of accuracy vs Dataset size

Figure 9 explains that the accuracy of our model is dependent upon the datasets under consideration and shows an increase with the increase in the dataset volume considered.

VIII. Conclusion

This study shows us that sentiment analysis performed using a combination of lexicon and machine learning approaches, performs better that using only lexicon-based approach. Results show that AdaBoost, Random Forest and Decision Trees give an average accuracy of ~70%

The Decision Tree algorithm implemented in this study gives us an accuracy of 65.7% based on the features obtained after feature selection. Incorporating other features, such as n-grams, count of exclamation and interrogation marks, could help yield better accuracy results. Moreover, the accuracy could improve with procurement of larger training values for the model to learn from.

References

[1] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M.. Lexiconbased methods for sentiment analysis. Computational linguistics, 2011: 37(2), 267-307.

[2] End-to-End Sentiment Analysis of Twitter Data by Apoor v Agarwal and Jasneet Singh Sabharwal. [c]

[3] Porter M.F. “An algorithm for suffix stripping”. Program. 1980; 14, 130-137. [a]

[4] Schofield A, Magnusson M, Mimno D (2017) Pulling out the stops: rethinking stopword removal for topic models. In: Proc, the 15th Conference of the European Chapter of the Association for Computational Linguistics: vol 2, Short Papers, pp 432–436 [b]

[5] Yorick Wilks and Mark Stevenson. The grammar of sense: Using part-of-speech tags as a first step in semantic disambiguation. Journal of Natural Language Engineering, 4(2):135–144, 1998 [d]

[6] L. Barbosa and J. Feng, \Robust sentiment detection on twitter from biased and noisy data," in Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING '10, (Stroudsburg, PA, USA), pp. 36-44, Association for Computational Linguistics, 2010. [e]

[7] E. Kouloumpis, T. Wilson, and J. Moore, Twitter sentiment analysis: The good, the bad and the omg!," in ICWSM (L. A. Adamic, R. A. Baeza-Yates, and S. Counts, eds.), The AAAI Press, 2011. [f]

[8] Goyal A. and Daume III H, “Generating Semantic Orientation Lexicon using Large Data and Thesaurus” [j]

[9] https://machinelearningmastery.com/

[10] Stone, Philip J., Dexter C. Dunphy, Marshall S. Smith, and Daniel M. Ogilvie. 1966. The General Inquirer: A Computer Approach to Content Analysis. MIT Press, Cambridge, MA.

[11] Baccianella, Stefano, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC’10), pages 2200–2204, Valletta.

[12] Bing Liu, Minqing Hu and Junsheng Cheng. "Opinion Observer: Analyzing; and Comparing Opinions on the Web." Proceedings of the 14th; International World Wide Web conference (WWW-2005), May 10-14, 2005, Chiba, Japan.

[13] Asher, Nicholas, Farah Benamara, and Yvette Yannick Mathieu. 2008. Distilling opinion in discourse: A preliminary study. In Proceedings of COLING, pages 7–10, Manchester.

[14] Medhat, W.; Hassan, A.; Korashy, H. Sentiment analysis algorithms and applications: A survey. *Ain Shams Eng. J.* 2014, *5*, 1093–1113. [k]

[15] Łukasz Augustyniak, Piotr Szyman´ski, Tomasz Kajdanowicz and Włodzimierz Tuligłowicz, Comprehensive Study on Lexicon-based Ensemble Classification Sentiment Analysis [l]

[16] Pavel Blinov, Maria Klekovkina, Eugeny Kotelnikov and Oleg Pestov. 2013. Research of lexical approach and machine learning methods for sentiment analysis. Computational Linguistics and Intellectual Technologies, 2(12):48–58.

[17] Ting Wang, Sheng, Fei Liu, Correlation-based feature ordering for classification based on neural incremental attribute learning, International Journal of Machine Learning and Computing, Vol. 2, No. 6, December 2012. [i]

[18] http://kldavenport.com/pure-python-decision-trees/

[19] Miao, Jianyu & Niu, Lingfeng. (2016). A Survey on Feature Selection. Procedia Computer Science. 91. 919-926. 10.1016/j.procs.2016.07.111. [g]

[20] Wlodzislaw Duch; “Filter methods”, Springer- Feature Extraction Studies in Fuzziness and Soft Computing [h]

Link to project code: Google Drive

https://drive.google.com/drive/folders/11Qs0LA7kCm_Hua6ItUMEQkVmtdnMUAD3?usp=sharing

Project Self-Assessment

Satyajit Narayanan
Unity ID: snaraya6

- The main objective of our project is to understand the implementation and effectiveness of lexicon-based and machine learning approach to classify sentiments of tweets.
- The major tasks in the project are procuring twitter sentiment data with class labels, dictionary creation, lexicon-based approach, data preprocessing, feature creating and selection, implementation of classification algorithm, comparative study of other algorithms and report and poster creation.
- I was responsible for the dictionary creation, implementation of lexicon-based approach and Naïve Bayes algorithm and was involved in parts of data preprocessing, feature creating (created 5 features and enhanced the list of negative, abusive and positive words), report and poster creation.
- The parts of the code I implemented were creation of features (Adjective, Adverb, Verb, Noun and Polarity), implementation of Naïve Bayes algorithm. I roughly contributed to close to 130 lines of code.
- In the poster I have written parts of Introduction, Feature creation and selection, lexicon-based approach and conclusion.
- In the report I contributed to abstract, introduction, feature creation and selection, lexicon-based approach and conclusion.
- Our group met 7 times out of which I participated 6 times. We also met with Dr. Raju for feedback and inputs on our progress 4 times out of which I participated 3 times. I rate my contribution to the project as above equal.

Jagadeesh Hariharan
Unity ID: jhariha

- What is the main objective of your project
    - The objective of this course project was to understand how to approach a data mining problem and what all are the steps to be taken on the selecting parameters on each tasks of the project. The problem domain that helped me achieve that was sentimental analysis by combining multiple approaches such as lexicon-based approach and machine learning based approach.

- What are the major tasks in the project
    - Data Extraction
    - Data Pre-processing
    - Implementing Lexicon based approach
    - Feature Extraction
    - Feature Selection
    - Implementation of supervised machine learning algorithm
    - Comparative Study

- Which task did you worked on, and how much did you finish so far?
    - I worked on data extraction, pre-processing, feature extraction, feature selection, comparative study, visualization of outputs, and created reports /posters.

- What part of the code did you implement (list functions or sections of the code in GitHub or whatever code development platform your group used)?
    - The following are certain functions that I implemented in Python which mainly deals with the preprocessing, feature extraction and post processing.
    - Nltk_process, pos_tagging, pos_words, neg_words, main, data input and calling functions, choosing the best classifier (comparative study) and compilation of the entire code.
    - Roughly list how many lines of code. ~100

- What are you contributing to the poster?

    - I designed and created the entire poster including introduction, data preprocessing, feature creation, feature selection, comparative study.

- What (sections) are contributing to the report?
    - I created the most part of the report including introduction, data preprocessing, feature creation, feature selection, comparative study, formatting of the entire report and visual block diagrams and charts to aid the explanation.

- How many times your group met; and how many times did you participate
    - The entire team met around 7 times and I was there 5 times. The group also met the professor

- How do you rate your contribution to the project (marginal, below equal, equal, above equal, almost everything)?
    - Above equal

Harsh Mehta
hkmehta

- Objective of our project: The main purpose of our project is to see the effect of Semantic Orientation score on sentiments of user's tweets on airlines and predict the same using classification methods
- The major tasks in our project are as follows:
    - Twitter data extraction of airline's tweets
    - Pre-processing of data
    - Implementing classification methods to predict the sentiment of tweets
    - Report making
    - Poster making
- I have worked on the implementation part with Riddhiman where we have implemented the algorithm of decision tree which is used to predict the class of sentiments. I have defined some functions in Python using Pandas library and Numpy Library. A function was defined which is used to calculate the entropy of the input train data for each value of split for each attribute and then the best split from each attribute and the corresponding entropy values were stored in a dictionary. Another function defined was used to calculate the gain of the split for each attribute and then the attribute, split value and entropy corresponding to the highest gain was selected. And then these functions were used in the main decision tree function which is used to store the decision rules and labels corresponding to these rules in an object using a class. After storing rules, the functions defined by Riddhiman were used to assign labels to the test data and the accuracy was calculated and the plots were created. I have also contributed in extracting twitter data by using twitter API.
- I have implemented some functions of ID3 algorithm which are as follows:
    - Entropy_data
    - Parent_node
    - Split_data
    - Unique_values
    - Complete_tree
- Contribution to poster : technical part of the poster which has the implementation of the decision tree algorithm
- My contribution in the report is in explaining the functionality of ID3 algorithm .
- I met about 5 times with the group where in we discussed the further direction of the project.
- I rate my contribution to the project to be equal.

Venkatesh Nayak
vanayak

1) The main objective of our project is to determine the sentiment of a tweet by utilizing a lexicon-based approach. This sentiment (be it positive, negative or neutral) helps us understand the opinion of the person who made it. As a student of Financial Mathematics, my personal objectives were to understand investor sentiments by analyzing the tweets.

2) The major tasks in the project are
   a) Project Definition: - Whereby, we understand the domain to which we want to work with and develop a roadmap to proceed by consulting the team and the mentors.
   b) Data Gathering: - To accumulate the tweets, (from the most recent ones to a certain amount of time in the past)
   c) Pre-Processing: - This involved cleaning of the hashtags, emoticons, links, hyperlinks, punctuations and stop-words
   d) Obtaining POS: - This consisted of getting the Part of Speeches of the most relevant words in the tweet.
   e) Dictionary Generation: - Creating a dictionary of the most frequent/ relevant words, along with the polarity of the words and the scale of their polarity
   f) Assigning Score to Tweets: - By separating a tweet according to part of speech, and assigning polarity to the part of speeches of the tweet, we score the tweet as positive, negative or neutral. This is then used for our Classification Models
   g) Implementing Classification Models: - By suitable dividing the dataset into training and testing, we implemented Decision Trees and Naïve Bayes

3) My first task was to implement the twitter API connection and gather the tweets. My code downloads the tweets in a text file, and helps in taking out the hashtags, links and hyperlinks.

   Secondly, I implemented the Naïve Bayes Algorithm. The features I used were Part of Speeches, Polarity, Number of Positive words and Number of Negative words. The accuracy obtained was approximately 52%

4) I implemented the code for gathering tweets. Approximately 60 lines.
   Furthermore, I implemented the code for Naïve Bayes Implementation. Approximately 180 lines of code. My code included functions to calculate means and standard deviations of multiple lists at once, function to calculate gaussian probabilities, calculating of conditional probabilities, separating out the part of speeches by class labels, functions to create a dictionary of summaries(mean and variance) of each class label according to the part of speeches, function to learn the model, make predictions about the test data and calculate the accuracy.I also worked on the Stanford nlp initially but later on the team decided to switch to nltk.

5) Another implementation of the Naïve Bayes algorithm by my team mates, showed a better accuracy and a more dynamic code then my own. Hence, the poster has no content of my work.

6) My contribution to the report includes the part of downloading the tweets and implementation of Naïve Bayes algorithm. In financial domain, Naïve Bayes algorithm works well, because we assume independence while working in portfolio analytics. However, I wish to highlight the fact that as market conditions become more stressed, the correlations tend to increase, and Naïve Bayes may not be the best model to proceed with.

7) Our group met for dividing the project tasks, for research on existing research papers in this domain, understanding the research papers, and choosing the classification algorithms. Approximately 5 Times.

8) Given the fact that Tweet gathering is not a Data Mining task and my code for Naïve Bayes Implementation was sub-par in comparison with my team mates, I rate my contribution, humbly as Below Equal.

Riddhiman Sherlekar
Unity ID: rsherle

- What is the main objective of your project

    - The main objective of the project was to build a classification model using the attributes derived using the lexicon-based approach on an airline related tweets data set.
- What are the major tasks in the project?
    - The major tasks in the project were data preprocessing, feature creation , comparative study of different classifiers , implementation of naive bayes and decision tree algorithms , report and poster creation.
- Which task did you worked on, and how much did you finish so far?
    - During the intial phase of the project , I extracted the data and then the preprocessing was done by others. I and harsh Mehta divided the implementation of decision tree . we implemented the code right from scratch which is uploaded on the repository. I contributed in the intial feature selection which included the buckets which were made according to the penn tree classification. I also worked initially on Stanford nlp .
- What part of the code did you implement (list functions or sections of the code in GitHub or whatever code development platform your group used)? Roughly list how many lines of code.
    - The intial part of the implementation code which includes the function key_labels , entropy_cal was done by me and the complete model building part which includes the traverse() function and the loop to fetch the labels from the developed decision tree was my part. Roughly around 100 lines of code.I contributed in the intial feature selection which included the buckets which were made according to the penn tree classification. I also worked initially on Stanford nlp .
- What are you contributing to the poster?
    - The technical part of the implementation of the ml model which includes the decision tree was done by me .I developed few visualisations.
- What (sections) are contributing to the report?
    - The implementation and methodology part was done by me and harsh Mehta since we have done the implementation together.
- How many times your group met; and how many times did you participate
    - The group met for 7 times in total , out of which I was present 5 times. The group met Dr. Raju 3 times out of which I was present 2 times and once I met Dr. Raju personally for the project.
- How do you rate your contribution to the project (marginal, below equal, equal, above equal, almost everything)?
    - I rate my contribution above equal. I actually learnt a lot of python due to this project.

Eshaan Kirpal
evkirpal

1. What is the main objective of your project ?

To use lexicon-based methods along with different machine learning algorithms to perform sentiment analysis of twitter data. Also, we are required to do a comparative study of the existing opinion mining techniques with our implemented algorithms.

2. What are the major tasks in the project?

Main tasks in the project are as below:

1. Harvest data from the twitter.
2. Dictionary Creation
3. Extracting relevant features from the twitter dataset
4. Data preprocessing
5. Implementing machine-learning algorithms (Decision Tree and Naive Bayes) for our application.
6. Developing visualization to display our algorithm performance.
7. Performing a comparative study of our algorithm performance with the existing

3.Which task did you worked on, and how much did you finish so far?
Implemented decision tree algorithm using entropy as disorder measure. Developed visualizations for the report and the poster.

4. What part of the code did you implement (list functions or sections of the code in GitHub or whatever code development platform your group used)? Roughly list how many lines of code.

We developed two decision tree algorithms of which I developed one of them.[Decision Tree Implementation 2] I developed the code in which we do pruning using minimum threshold entropy gain. I also wrote the code for the various visualizations developed for the project.

5. What are you contributing to the poster?

Developed a few visualizations for the poster.

6. What (sections) are contributing to the report?

Assisted in writing the methodology and implementation part of the report.

7. How many times your group met; and how many times did you participate?

Once in two weeks.

8. How do you rate your contribution to the project (marginal, below equal, equal, above equal, almost everything)?

I believe it was between equal to above equal.