Sentiment Analysis of IMDB Movie Review Using Naïve Bayes

Samsheeth Kosgi and Vamsi Krishna Madala, Computer Science department, SDSU

Abstract— **Sentiment analysis (also known as opinion mining) refers to the use of natural language processing, text analysis and computational linguistics to identify and extract subjective information in source materials. Sentiment analysis is widely applied to reviews and social media for a variety of applications, ranging from marketing to customer service. Generally speaking, sentiment analysis aims to determine the attitude of a speaker or a writer with respect to some topic or the overall contextual polarity of a document. The attitude may be his or her judgment or evaluation (see appraisal theory), affective state (that is to say, the emotional state of the author when writing), or the intended emotional communication (that is to say, the emotional effect the author wishes to have on the reader).We extract the data set from the amazon, yelp and IMDB reviews and then process those data for sentiment analysis using naïve bayes classifier.**

***Index Terms*— API (Application program interface),sentiment analysis**

I. Introduction

A basic task in sentiment analysis is classifying the *polarity* of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion in a document, a sentence or an entity feature/aspect is positive, negative, or neutral. Advanced, "beyond polarity" sentiment classification looks, for instance, at emotional states such as "angry", "sad", and "happy".

Types of analysis might discern sentiment for documents or messages or for particular entities, topics, or concepts with documents or messages. Analyses might look for individual cases ("mentions") or for aggregates over populations or sources or trends over time. They may rate sentiment on an absolute scale or they may look for relative/comparative sentiment and/or measure such as variation, intensity, and change. They may seek to link sentiment to psychological profile, behaviors, demographic characteristics, transactions, events, and/or other data for analyzing and maintaining data.

Functionally, there are analysis by trained humans, crowd-sourced analysis by untrained humans, automated analysis of information extracted from "unstructured" sources such as text, audio, images, and video, for instance for text via natural-language processing, analysis of categorical poll or survey questions, e.g., "Rate your hotel stay on a scale of 1 to 5," and the equivalent, star ratings, and inference of sentiment from numerical statistics, for instance, commercial inventories, consumer spending, investment levels, etc. Automated NLP may apply linguistic, statistical, and/or machine-learning techniques.

II. APPLICATIONS OF SENTIMENT ANALYSIS

When consumers have to make a decision or a choice regarding a product, an important information is the reputation of that product, which is derived from the opinion of others. Sentiment analysis can reveal what other people think about a product. The first application of sentiment analysis is thus giving indication and recommendation in the choice of products according to the wisdom of the crowd. When you choose a product, you are generally attracted to certain specific aspects of the product. A single global rating could be deceiving. Sentiment analysis can regroup the opinions of the reviewers and estimate ratings on certain aspects of the product. Another utility of sentiment analysis is for companies that want to know the opinion of customers on their products. They can then improve the aspects that the customers found unsatisfying. Sentiment analysis can also determine which aspects are more important for the customers. Finally, sentiment analysis has been proposed as a component of other technologies. One idea is to improve information mining in text analysis by excluding the most subjective section of a document or to automatically propose internet ads for products that fit the viewer’s opinion (and removing the others). Knowing what people think gives numerous possibilities in the Human/Machine interface domain. Sentiment analysis for determining the opinion of a customer on a product (and consequently the reputation of the product) is the main focus of this paper. In the following section, we will discuss solutions that allow to determine the expressed opinion on products. Similarly, we use the movie reviews data for sentiment analysis and we represent ‘1’ for positive review and ‘0’ for negative review.

III. CLASSIFICATION OF EXISTING SOLUTION

The existing work on sentiment analysis can be classified from different points of views: technique used, view of the text, level of detail of text analysis, rating level, etc. From a technical point of view, we identified this as machine

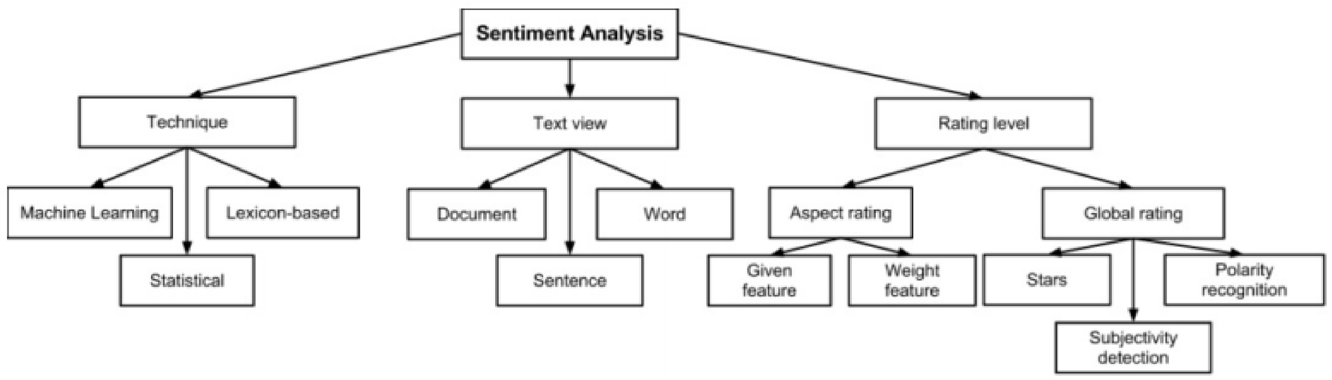

Fig. 1: Classification

learning, lexicon-based, statistical and rule-based approaches.

• The machine learning method uses several learning algorithms to determine the sentiment by training on a known dataset.

• The lexicon-based approach involves calculating sentiment polarity for a review using the semantic orientation of words or sentences in the review. The "semantic orientation" is a measure of subjectivity and opinion in text.

• The rule-based approach looks for opinion words in a text and then classifies it based on the number of positive and negative words. It considers different rules for classification such as dictionary polarity, negation words, booster words, idioms, emoticons, mixed opinions etc. A Study and Comparison of Sentiment Analysis Methods for Reputation Evaluation

• Statistical models represent each review as a mixture of latent aspects and ratings. It is assumed that aspects and their ratings can be represented by multinomial distributions and try to cluster head terms into aspects and sentiments into ratings.

Another classification is oriented more on the structure of the text: document level, sentence level or word/feature level classification. Document-level classification aims to find a sentiment polarity for the whole review, whereas sentence level or word-level classification can express a sentiment polarity for each sentence of a review and even for each word. Our study shows that most of the methods tend to focus on a document-level classification. We can also distinguish methods which measure sentiment strength for different aspects of a product and methods which attempt to rate a review on a global level. Most of the solutions focusing on global review classification consider only the polarity of the review (positive/negative) and rely on machine learning techniques. Solutions that aim a more detailed classification of reviews (e.g., three or five star ratings) use more linguistic features including intensification, negation, modality and discourse structure [dAPGD11]. Figure 1 presents a detailed classification of existing methods. This classification is not exclusive. One solution can fit into more than one category.

IV. METHODS OF SENTIMENT ANALYSIS

Software used for this study is scikit-learn [23], an open source machine learning software package in Python. The classification models selected for categorization are: Naïve Bayesian, Random Forest, and Support Vector Machine [22].

Naïve Bayesian classifier

The Naïve Bayesian classifier works as follows: Suppose that there exist a set of training data, *D*, in which each tuple is represented by an *n*-dimensional feature vector, $X=x_1,x_2,..,x_n$, indicating *n* measurements made on the tuple from *n* attributes or features. Assume that there are *m* classes, $C_1,C_2,...,C_m$.

Given a tuple *X*, the classifier will predict that *X* belongs to C_i if and only if: $P(C_i|X)>P(C_j|X)$, where $i,j\in[1,m] and\ i\neq j$. $P(C_i|X)$ is computed as:
P(Ci|X)=∏k=1nP(xk|Ci)P(Ci|X)=∏k=1nP(xk|Ci)

Random forest

The random forest classifier was chosen due to its superior performance over a single decision tree with respect to accuracy. It is essentially an ensemble method based on bagging. The classifier works as follows: Given *D*, the classifier firstly creates *k* bootstrap samples of *D*, with each of the samples denoting as D_i. A D_i has the same number of tuples as *D* that are sampled with replacement from *D*. By sampling with replacement, it means that some of the original tuples of *D* may not be included in D_i, whereas others may occur more than once. The classifier then constructs a decision tree based on each D_i. As a result, a "forest" that consists of *k* decision trees is formed. To classify an unknown tuple, *X*, each tree returns its class prediction counting as one vote. The final decision of *X*'s class is assigned to the one that has the most votes. The decision tree algorithm implemented in scikit-learn is CART (Classification and Regression Trees). CART uses Gini index for its tree induction. For *D*, the Gini index is computed as:
Gini(D)=1−∑i=1mp2iGini(D)=1−∑i=1mpi2

where p_i is the probability that a tuple in D belongs to class C_i. The Gini index measures the impurity of D. The lower the index value is, the better $D$was partitioned. For the detailed descriptions of CART, please see.

Support vector machine

Support vector machine (SVM) is a method for the classification of both linear and nonlinear data. If the data is linearly separable, the SVM searches for the linear optimal separating hyperplane (the linear kernel), which is a decision boundary that separates data of one class from another. Mathematically, a separating hyperplane can be written as: $W \cdot X + b = 0$, where W is a weight vector and $W = w_1, w2, ..., w\ n$. X is a training tuple. b is a scalar. In order to optimize the hyperplane, the problem essentially transforms to the minimization of $\|W\|$, which is eventually computed as: ∑i=1nαiyixi∑i=1nαiyixi, where $\alpha_i$are numeric parameters, and y_i are labels based on support vectors, X_i. That is: if $y_i = 1$ then ∑i=1nwixi≥1∑i=1nwixi≥1; if $y_i = -1$ then ∑i=1nwixi≥−1∑i=1nwixi≥−1.
If the data is linearly inseparable, the SVM uses nonlinear mapping to transform the data into a higher dimension. It then solve the problem by finding a linear hyperplane. Functions to perform such transformations are called kernel functions. The kernel function selected for our experiment is the Gaussian

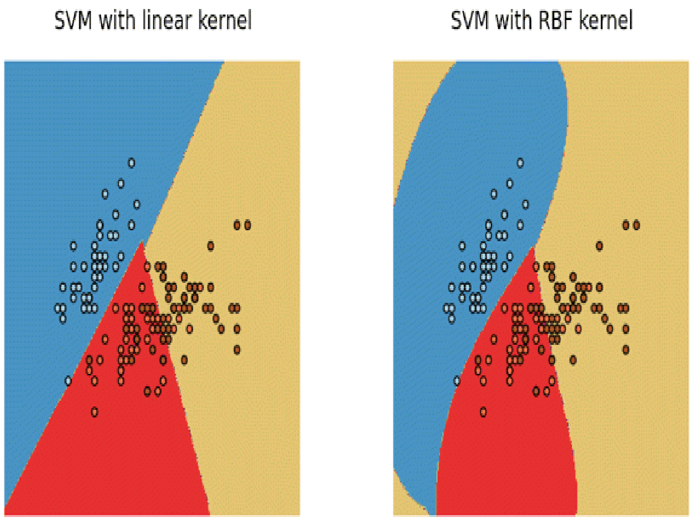

Fig.2.SVM with linear & kernel

Radial Basis Function (RBF):
K(Xi,Xj)=e−γ‖Xi−Xj‖2/2K(Xi,Xj)=e−γ‖Xi−Xj‖2/2

where X_i are support vectors, X_j are testing tuples, and γ is a free parameter that uses the default value from scikit-learn in our experiment. Figure 2shows a classification example of SVM based on the linear kernel and the RBF kernel.

V. EXTRACTION OF DATA USING IMDB EXTRACTOR

IMDB extractor transforms Internet Movie Database data files into a topic map browsable with Wandora. Extractor has been created for demonstration purposes only. Wandora does not contain any IMDB data files. Also, be aware that Wandora or Wandora authors have no rights to give you any permission to use IMDB data. If you plan to use IMDB topic maps beyond personal usage, you should contact IMDB Licensing department. As datafiles are extremely large you can't extract data to memory topic maps but have to use database topic maps. Wandora does not transfer all IMDB files. Current extractor transfers only actors, actresses, keywords, countries, language, locations, genres, movies, biographies, producers, directors, plot summaries, running times and release dates.

To prepare the extraction download all required data files and unpack them to your local file system. Then create a database topic map and start extractor with File > Extract > Media > IMDB Extractor. Wandora requests a folder containing IMDB data files or a single data file and starts the extraction after successful data file or folder identification. IMDB data files are very large and you should be patient as the extraction may take a while.

Extractions were made in a Ubuntu Linux 8.1 running on top of Sun's VirtualBox(running on top of Windows XP). system properties of the Ubuntu Linux used for IMDB extractions.Notice the memory amount given for the Linux. We gave the Ubuntu 1500 MB of memory. Our experiences suggest you should give Linux memory as much as possible. With small memory footprints, the IMDB extraction fails after heavy swapping. Now start Ubuntu Linux and log in.

A. Downloading IMDB datafiles

After Ubuntu launch, start WWW browser in Ubuntu and Unzip all data files in shell with gunzip or right click each data file icon and select option Extract Here.
Now you should have all required IMDB data files ready for extraction

B. Setting up Wandora

We prepare Wandora application next. In Ubuntu

- Download Wandora application.
- Install Wandora
- Start Linux shell with menu option Applications > Accessories > Terminal
  - Open Wandora's bin directory.
  - Change execution rights of Wandora-huge.sh to allow execution.
  - Finally add Java's bin directory to the PATH environment variable.

C. Setting up databases for IMDB topic maps

As stated in the beginning of IMDB extractor documentation above, you need a database topic map to store extracted topic map as it is very large. To prepare database topic map start another terminal window in Ubuntu with option Applications > Accessories > Terminal. In terminal

- Install MySQL server with command sudo apt-get install mysql-server.
- Log into the MySQL server with command mysql --user=<your-username> --password=<your-password>
- Create empty databases with MySQL command create database <database-name>**;**

Prepare each created database with Wandora specific database table structures in wandora/build/resources/conf/database/db_mysql**.**sql. In detail:

- Select database with MySQL command use <database-name>**;**
    - Read database table creation clauses from external file with MySQL command source wandora/build/resources/conf/database/db_mysql**.**sql**;** (notice ending semicolon). Notice that you may have to change the path of db_mysql**.**sql depending on you Wandora installation directory and your current directory.

D. Extracting IMDB with Wandora

Go back to the Wandora application started earlier and select menu option Layers > New layer. A dialog window opens. Select Database on drop down selector labeled Type

Select MySQL test in database settings list and click Edit button. Another dialog opens for database settings (see image below). In this dialog, you can enter database's name, user name, and password. Change database name to imdb_actors. Change user field to your database user name. Change password field to the user's password.
Now click OK button and database configuration window closes reveling previous dialog window. Enter name for the layer, say imdb_actors, keep the MySQL test database configuration selected, and click OK button. Wandora creates a new topic map layer and shows it left bottom corner of Wandora application window (see below). Now select the created layer by clicking it. Selected layer is little darker than unselected. Now all "write" operations go to the selected database topic map layer.
If created layer is dark red, your new layer is broken. Layer is broken when database connection fails for some reason. Check Wandora's terminal window for specific error message. I managed to break a layer couple of times by entering wrong user name and password for the database.
Next, we are going to start the IMDB extraction. Select menu option File > Extract > Media > IMDB extract**...**. Wandora opens a Files/Urls/Raw selector. Keep the Files tab open and click Browsebutton. A file selector opens. Go to the directory you uncompressed IMDB data files and select actors**.**list (see below). To start extraction press Extract button. As IMDB data files are extremely large, it is not very surprising that it is extraction takes several hours.

When extraction finishes, you can request statistics from the database topic map layer with menu option Layers > Statistics > Layer info**...**. It took my system several minutes to open layer statistics dialog window.

Extracted topic map contained little over 2 million topics and near 3 million associations. It is very important you to understand that trying to access such topic map in Wandora is extremely slow and causes OutOfMemory exceptions easily. As a thumb rule, do not try to search anything that could generate a result set with millions of hits. Also, do not open association type topics, role topics, or class topics as they probably generate extremely large topic table structures Wandora can't handle.

Now, to continue extracting other IMDB files, drop extracted layer imdb_actors with menu option Layers > Delete layer**...** Database topic map layer deletion doesn't touch the database content and you can open it again later on. It's just more convenient to do the extraction when there are no other topic map layers disturbing.

E. Merging IMDB database topic map layers

Now you should have all IMDB data files extracted. Final step is to open all generated topic maps to Wandora as separate layers. In Wandora, for each database topic map

- Select menu option Layers > New layer**...**
- Change topic map type to Database
- Edit default settings of MySQL test as you did while preparing the extraction.
- Give unique name for the layer and hit OK.

As a result, your Wandora should look something like below and you can continue accessing the merged IMDB topic. Be careful, the layer stack is huge and you get easily OutOfMemory exceptions as said above

VI. BACKGROUNG AND LITERATURE REVIEW

One fundamental problem in sentiment analysis is categorization of sentiment polarity [11-15]. Given a piece of written text, the problem is to categorize the text into one specific sentiment polarity, positive or negative (or neutral). Based on the scope of the text, there are three levels of sentiment polarity categorization, namely the document level, the sentence level, and the entity and aspect level [16]. The document level concerns whether a document, as a whole, expresses negative or positive sentiment, while the sentence level deals with each sentence’s sentiment categorization; The entity and aspect level then targets on what exactly people like or dislike from their opinions. Since reviews of much work on sentiment analysis have already been included in [16], in this section, we will only review some previous work, upon which our research is essentially based. Hu and Liu [17] summarized a list of positive words and a list of negative words, respectively, based on customer reviews. The positive list contains 2006 words and the negative list has 4783 words. Both lists also include some misspelled words that are frequently present in social media content. Sentiment categorization is essentially a classification problem, where features that contain opinions or sentiment information should be identified before the classification. For feature selection, Pang and Lee [5] suggested to remove objective sentences by extracting subjective ones. They proposed a text-categorization technique that is able to identify subjective content using minimum cut. Gann et al. [18] selected 6,799 tokens based on Twitter data, where each token is assigned a sentiment score, namely TSI (Total Sentiment Index), featuring itself as a positive token or a negative token. Specifically, a TSI for a certain token is computed as:

$$TSI=(p-tp/tn\times n)/(p+tp/tn*n)$$

where p is the number of times a token appears in positive tweets and n is the number of times a token appears in negative tweets. Tp/Tn is the ratio of total number of positive tweets over total number of negative tweets.

A. Sentiment sentences extraction and POS tagging

It is suggested by Pang and Lee [24] that all objective content should be removed for sentiment analysis. Instead of removing objective content, in our study, all subjective content was extracted for future analysis. The subjective content consists of all sentiment sentences. A sentiment sentence is the one that contains, at least, one positive or negative word. All of the sentences were firstly tokenized into separated English words. Every word of a sentence has its syntactic role that defines how the word is used. The syntactic roles are also known as the parts of speech. There are 8 parts of speech in English: the verb, the noun, the pronoun, the adjective, the adverb, the preposition, the conjunction, and the interjection. In natural language processing, part-of-speech (POS) taggers [19-21] have been developed to classify words based on their parts of speech. For sentiment analysis, a POS tagger is very useful because of the following two reasons: 1) Words like nouns and pronouns usually do not contain any sentiment. It is able to filter out such words with the help of a POS tagger; 2) A POS tagger can also be used to distinguish words that can be used in different parts of speech. For instance, as a verb, “enhanced" may conduct different amount of sentiment as being of an adjective. The POS tagger used for this research is a max-entropy POS tagger developed for the Penn Treebank Project [21]. The tagger is able to provide 46 different tags indicating that it can identify more detailed syntactic roles than only 8.

B. Negation phrases identification

Words such as adjectives and verbs are able to convey opposite sentiment with the help of negative prefixes. For instance, consider the following sentence that was found in an electronic device’s review: “The built-in speaker also has its uses but so far nothing revolutionary." The word, “revolutionary" is a positive word according to the list in [17]. However, the phrase “nothing revolutionary" gives more or less negative feelings. Therefore, it is crucial to identify such phrases. In this work, there are two types of phrases have been identified, namely negation-of-adjective (NOA) and negation-of-verb (NOV). Most common negative prefixes such as not, no, or nothing are treated as adverbs by the POS tagger. Hence, we propose Algorithm 1 for the phrases identification. The algorithm could identify 21,586 different phrases with total occurrence of over 0.68 million, each of which has a negativeprefix.

Algorithm 1 Negation phrases identification

```
Input: Tagged Sentences, Negative Prefixes
Output: NOA Phrases, NOV Phrases
1:  for every Tagged Sentences do
2:    for $i/i+1$ as every word/tag pair do
3:      if $i+1$ is a Negative Prefix then
4:        if there is an adjective tag or a verb tag in next pair then
5:          NOA Phrases ← $(i, i+2)$
6:          NOV Phrases ← $(i, i+2)$
7:        else
8:          if there is an adjective tag or a verb tag in the pair after next then
9:            NOA Phrases ← $(i, i+2, i+4)$
10:           NOV Phrases ← $(i, i+2, i+4)$
11:         end if
12:       end if
13:     end if
14:   end for
15: end for
16: return NOA Phrases, NOV Phrases
```

C. Sentiment score computation for sentiment tokens

A sentiment token is a word or a phrase that conveys sentiment. Given those sentiment words proposed in [17], a word token consists of a positive (negative) word and its part-of-speech tag. In total, we selected 11,478 word tokens with each of them that occurs at least 30 times throughout the dataset. For phrase tokens, 3,023 phrases were selected of the 21,586 identified sentiment phrases, which each of the 3,023 phrases also has an occurrence that is no less than 30. Given a token t, the formula for t’s sentiment score (SS) computation is given as:

SS(t)=∑i=15i×γ5,i×Occurrencei(t)∑i=15γ5,i×Occurrencei(t)S S(t)=∑i=15i×γ5,i×Occurrencei(t)∑i=15γ5,i×Occurrencei(t) (2)

$Occurrence_i(t)$ is t’s number of occurrence in i-star reviews, where i=1...,5 our dataset is not balanced indicating that different number of reviews were collected for each star level. Since 5-star reviews take a majority amount through the entire dataset, we hereby introduce a ratio, $\gamma_{5,i}$, which is defined as:

γ5,i=|5−star||i-star|γ5,i=|5−star||i-star| (3)

In equation 3, the numerator is the number of 5-star reviews and the denominator is the number of i-star reviews, where i=1,...,5. Therefore, if the dataset were balanced, $\gamma_{5,i}$ would be set to 1 for every i. Consequently, every sentiment score should fall into the interval of [1,5]. For positive word tokens, we expect that the median of their sentiment scores should exceed 3, which is the point of being neutral. For negative word tokens, it is to expect that the median should be less than 3.

As a result, the sentiment score information for positive word tokens is showing in Figure 3(a). The histogram chart describes the distribution of scores while the box-plot chart shows that the median is above 3. Similarly, the box-plot chart in Figure 3(b) shows that the median of sentiment scores for negative word tokens is lower than 3. In fact, both the mean and the median of positive word tokens do exceed 3, and both values are lower than 3, for negative word tokens (Table 1).

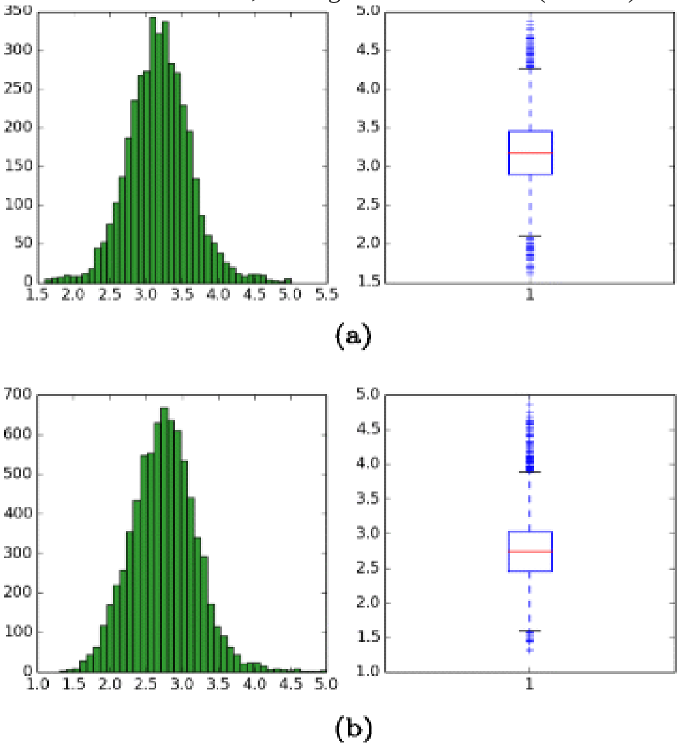

Figure 4 Sentiment score information for word tokens **(a)** Positive word tokens **(b)**Negative word tokens

Table1: Statistical information for word tokens

Token Type	Mean	Median
Positive Word Token	3.18	3.16
Negative Word Token	2.75	2.71

D. Feature vector formation

Sentiment tokens and sentiment scores are information extracted from the original dataset. They are also known as features, which will be used for sentiment categorization. In order to train the classifiers, each entry of training data needs to be transformed to a vector that contains those features, namely a feature vector. For the sentence-level (review-level) categorization, a feature vector is formed based on a sentence (review). One challenge is to control each vector's dimensionality. The challenge is actually twofold: Firstly, a vector should not contain an abundant amount (thousands or hundreds) of features or values of a feature, because of the curse of dimensionality [22]; secondly, every vector should have the same number of dimensions, in order to fit the classifiers. This challenge particularly applies to sentiment tokens: On one hand, there are 11,478 word tokens as well as 3,023 phrase tokens; On the other hand, vectors cannot be formed by simply including the tokens appeared in a sentence (or a review), because different sentences (or reviews) tend to have different number of tokens, leading to the consequence that the generated vectors are in different dimensions. Since we only concern each sentiment token's appearance inside a sentence or a review, to overcome the challenge, two binary strings are used to represent each token's appearance. One string with 11,478 bits is used for word tokens, while the other one with a bit-length of 3,023 is applied for phrase tokens. For instance, if the *i*th word (phrase) token appears, the word (phrase) string's *i*th bit will be flipped from "0" to "1". Finally, instead of directly saving the flipped strings into a feature vector, a hash value of each string is computed using Python's built-in hash function and is saved. Hence, a sentence-level feature vector totally has four elements: two hash values computed based on the flipped binary strings, an averaged sentiment score, and a ground truth label. Comparatively, one more element is exclusively included in review-level vectors. Given a review, if there are *m* positive sentences and *n* negative sentences, the value of the element is computed as: $-1 \times m + 1 \times n$.

VII. Evaluation

The accuracy of a sentiment analysis system is, in principle, h ow well it agrees with human judgments. This is usually meas ured by precision and recall. However, according to research h uman raters typically agree 79% of the time (see Inter-rater reli ability). Thus, a 70% accurate program is doing nearly as wel l as humans, even though such accuracy may not sound impres sive. If a program were "right" 100% of the time, humans wou ld still disagree with it about 20% of the time, since they disag ree that much about *any* answer. More sophisticated measure can be applied, but evaluation of sentiment analysis systems re mains a complex matter. For sentiment analysis tasks returnin g a scale rather than a binary judgement, correlation is a better measure than precision because it takes into account how close the predicted value is to the target value. In our program, we achieved the accuracy of 0.8791.can be applied, but evaluation of sentiment analysis systems remains a complex matter. For s entiment analysis tasks returning a scale rather than a binary ju dgement, correlation is a better measure than precision becaus e it takes into account how close the predicted value is to the t arget value. In our program, we achieved the accuracy of 0.87 91.

VIII. Results And Discussion

Performance of each classification model is estimated base on its averaged F1-score: $F1avg = \sum (2 \times Pi \times Ri)/(Pi+Ri)/n$. where P_i is the precision of the *i*th class, R_i is the recall of the *i*th class, and *n* is the number of classes. P_i and R_i are evaluated using 10-fold cross validation.

Result on manually-labeled sentences

200 feature vectors are formed based on the 200 manually-labeled sentences. As a result, the classification models show the same level of performance based on their F1-scores, where the three scores all take a same value of 0.85. With the help of the ROC curves (Figure 4), it is clear to see that all three models performed quite well for testing data that have high posterior probability. (A posterior probability of a testing data point, *A*, is estimated by the classification model as the probability that *A* will be classified as positive, denoted as $P(+|A)$.) As the probability getting lower, the Naïve Bayesain classifier outperforms the SVM classifier, with a larger area under curve. In general, the Random Forest model performs the best.

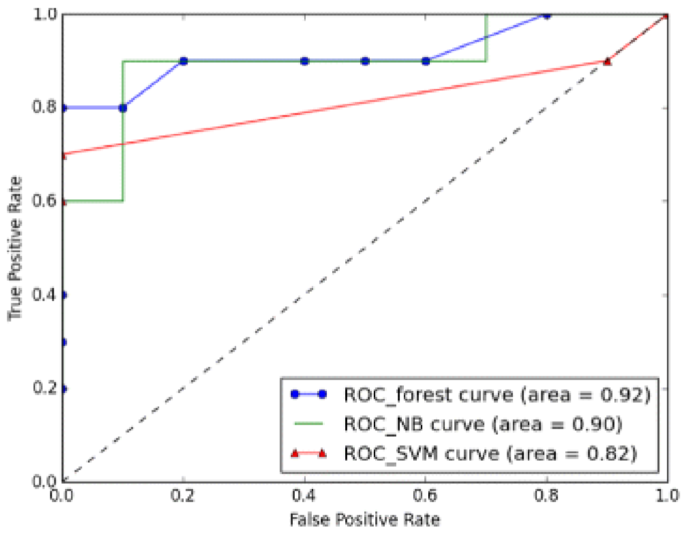

Fig 4. ROC curves based on the manually labeled set.

The results of this project is evolved using naïve bayes classifier, we vectorizer the text by using TFID and then convert the df.txt file from txt to features. We then used the test train set as usual and then trained it using naïve bayes classifier, we have also calculated the accuracy of the Sentiment analysis using the probability function and obtained an accuracy of 0. 8791.We test our algorithm as: the given sentence is then classified as follows if the given sentence is positive then the output is 1 and if the given sentence is negative the output is 0. In this way, we classify the data as a positive review and negative review.

IX. Conclusion

This study shows that the field of sentiment analysis has been well studied by researchers in the past few years. Many different methods have been developed and tested. However, a lot of work is yet to be done. The most common approach is machine learning, a method that needs a significant data set for training and learning the aspects and sentiments associated with naïve bayes. Also, models tend to target a simple global classification of reviews, rather than rating individual aspects of the reviewed product. Only a few of the methods are able to reach a somewhat high level of accuracy. Thus, the solutions for sentiment analysis still have a long way to go before reaching the confidence level demanded by practical applications. We obtain the accuracy of 0.8791.

References

[1] Amazon API Gateway: https://aws.amazon.com/api-gateway/
[2] https://www.yelp.com/developers/documentation/v2/search_api Yelp developers document.
[3] The OMDb API is a free web service to obtain movie information, all content and images on the site are contributed and maintained by our users. https://www.omdbapi.com/.
[4] Stanford (2014) Sentiment 140. http://www.sentiment140.com/.
[5] http://www.wandora.org/wandora/wiki/index.php?title=IMDB_extractor.
[6] http://opendata.stackexchange.com/questions/1073/where-to-get-imdb-datasets.
[7] https://www.themoviedb.org/documentation/api.
[8] http://imdbpy.sourceforge.net/
[9] http://liris.cnrs.fr/Documents/Liris-6508.pdf.
[10] http://www.wandora.org/wandora/wiki/index.php?title=IMDB_extractor
[11] Pang B, Lee L (2008) "Opinion mining and sentiment analysis", *Found Trends Inf Retr2(1-2): 1–135.*.
[12] Chesley P, Vincent B, Xu L, Srihari RK (2006) "Using verbs and adjectives to automatically classify blog sentiment", *Training580(263): 233.*.
[13] Choi Y, Cardie C (2009) "Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification In": *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2 - Volume 2, EMNLP '09, 590–598.. Association for Computational Linguistics, Stroudsburg, PA, USA.*
[14] Jiang L, Yu M, Zhou M, Liu X, Zhao T (2011) "Target-dependent twitter sentiment classification In": *Proceedings of the 49th, Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, 151–160.. Association for Computational Linguistics, Stroudsburg, PA, USA.*
[15] Tan LK-W, Na J-C, Theng Y-L, Chang K (2011) "Sentence-level sentiment polarity classification using a linguistic approach In": Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation, 77–87.. Springer, Heidelberg, Germany.
[16] Liu B (2012)" Sentiment Analysis and Opinion Mining. Synthesis Lectures on Human Language Technologies". *Morgan & Claypool Publishers*.
[17] Hu M, Liu B (2004) "Mining and summarizing customer reviews In": *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 168–177.. ACM, New York, NY, USA.*
[18] Gann W-JK, Day J, Zhou S (2014) "Twitter analytics for insider trading fraud detection system In": *Proceedings of the sencond ASE international conference on Big Data.. ASE.*
[19] Roth D, Zelenko D (1998) "Part of speech tagging using a network of linear separators In": *Coling-Acl, The 17th International Conference on Computational Linguistics, 1136–1142.*
[20] Kristina T (2003) Stanford log-linear part-of-speech tagger. http://nlp.stanford.edu/software/tagger.shtml.
[21] Marcus M (1996) Upenn part of speech tagger. http://www.cis.upenn.edu/~treebank/home.html.
[22] Han J, Kamber M, Pei J (2006) "Data Mining: Concepts and Techniques", *Second Edition (The Morgan Kaufmann Series in Data Management Systems), 2nd ed.. Morgan Kaufmann, San Francisco, CA, USA.*
[23] (2014) Scikit-learn. http://scikit-learn.org/stable/.
[24] Pang B, Lee L (2004) "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts In": *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics, ACL '04.. Association for Computational Linguistics, Stroudsburg, PA, USA.*
[25] Kim S-M, Hovy E (2004) "Determining the sentiment of opinions In": *Proceedings of the 20th international conference on Computational*

Linguistics, page 1367.. Association for Computational Linguistics, Stroudsburg, PA, USA.

[26] Liu B, Hu M, Cheng J (2005) "Opinion observer: Analyzing and comparing opinions on the web In": Proceedings of the 14th International Conference on World Wide Web, WWW '05, 342–351.. ACM, New York, NY, USA.

[27] Pak A, Paroubek P (2010) "Twitter as a corpus for sentiment analysis and opinion mining In": *Proceedings of the Seventh conference on International Language Resources and Evaluation.. European Languages Resources Association, Valletta, Malta.*

[28] Pang B, Lee L (2008) "Opinion mining and sentiment analysis". *Found Trends Inf Retr2(1-2): 1–135.*

[29] Turney PD (2002) Thumbs up or thumbs down?: "Semantic orientation applied to unsupervised classification of reviews In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*", ACL '02, 417–424.. Association for Computational Linguistics, Stroudsburg, PA, USA.*

[30] Whitelaw C, Garg N, Argamon S (2005) "Using appraisal groups for sentiment analysis In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management", *CIKM '05, 625–631.. ACM, New York, NY, USA.*

[31] Liu B (2014) The science of detecting fake reviews. http://content26.com/blog/bing-liu-the-science-of-detecting-fake-reviews/.

[32] Jindal N, Liu B (2008) "Opinion spam and analysis In: Proceedings of the 2008 International Conference on, Web Search and Data Mining, WSDM "*'08, 219–230.. ACM, New York, NY, USA.*

[33] Mukherjee A, Liu B, Glance N (2012) "Spotting fake reviewer groups in consumer reviews In: Proceedings of the 21st, International Conference on World Wide Web*", WWW '12, 191–200.. ACM, New York, NY, USA.*

[34] Go A, Bhayani R, Huang L (2009) "Twitter sentiment classification using distant supervision", 1–12.. CS224N Project Report, Stanford.

[35] Lin Y, Zhang J, Wang X, Zhou A (2012) "An information theoretic approach to sentiment polarity classification In: Proceedings of the 2Nd Joint WICOW/AIRWeb Workshop on Web Quality", *WebQuality '12, 35–40.. ACM, New York, NY, USA.*

[36] Sarvabhotla K, Pingali P, Varma V (2011) "Sentiment classification: a lexical similarity based approach for extracting subjectivity in documents". *Inf Retrieval14(3): 337–353.*

[37] Wilson T, Wiebe J, Hoffmann P (2005) "Recognizing contextual polarity in phrase-level sentiment analysis In: Proceedings of the conference on human language technology and empirical methods in natural language processing", *347–354.. Association for Computational Linguistics, Stroudsburg, PA, USA.*

[38] Yu H, Hatzivassiloglou V (2003) "Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences In: Proceedings of the 2003 conference on, Empirical methods in natural language processing*", 129–136.. Association for Computational Linguistics, Stroudsburg, PA, USA.*

[39] Zhang Y, Xiang X, Yin C, Shang L (2013) "Parallel sentiment polarity classification method with substring feature reduction In: Trends and Applications in Knowledge Discovery and Data Mining, volume 7867 of Lecture Notes in Computer Science", *121–132.. Springer Berlin Heidelberg, Heidelberg, Germany.*

[40] Zhou S, Chen Q, Wang X (2013) "Active deep learning method for semi-supervised sentiment classification". *Neurocomputing120(0): 536–546. Image Feature Detection and Description.*