INTELLIGENT SENTIMENT CLASSIFICATION ON TWEETS

A report on Project Phase – II submitted

in partial fulfillment of the requirements for the award of the degree of

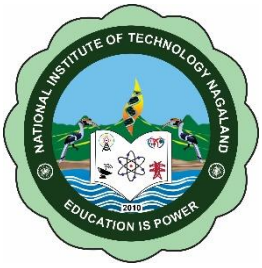
Bachelor of Technology
in
Computer Science and Engineering

by

LIPOKMERENLA
Registration No.: 2013105075

DURGA SHANKAR MISHRA
Registration No.: 2013105070

INDRASEN SINGH
Registration No.: 2013105071



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY NAGALAND

MAY 2017

राष्ट्रीय प्रौद्योगिकी संस्थान नागालैंड
NATIONAL INSTITUTE OF TECHNOLOGY NAGALAND
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
Chumukedima, Dimapur – 797 103, Nagaland

BONAFIDE CERTIFICATE

Certified that this Project titled **"INTELLIGENT SENTIMENT CLASSIFICATION ON TWEETS"** is the bonafide work of **Lipokmerenla(2013105075), Durga Shankar Mishra(2013105070) & Indrasen Singh(2013105071)** who carried out the work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other student.

Mr. Dilwar Hussian Mazumder
(Project Coordinator)
Assistant Professor
Department of CSE
NIT Nagaland
Dimapur – 797103

Mr. J. Arul Valan
(Project Supervisor)
Assistant Professor
Department of CSE
NIT Nagaland
Dimapur – 797103

Dr. THEMRICHON TUITHUNG
Professor & Head
Department of CSE
NIT Nagaland
Dimapur – 797103

Dr. SHIRSHENDU DAS
(External Examiner)
Assistant Professor
Department of CSE
IIIT Guwahati
GNB Road, Guwahati-781001

ABSTRACT

Sentiment analysis is a broad research area in academic as well as business field. The term sentiment refers to the feelings or opinion of the person towards some particular domain. Hence it is also known as opinion mining. It leads to the subjective impressions towards the domain, not facts. It can be expressed in terms of polarity, reviews or previously by thumbs up and down to denote positive and negative sentiments respectively. Sentiments can be analyzed using NLP, statistics or machine learning techniques. Sentiment analysis may ask questions regarding "customer satisfaction and dissatisfaction, "public opinion towards new iPhone series launched" etc. In real world, public or consumer opinions about some product or brand are very important for its sell. Hence sentiment analysis is a very important research area for real life applications i.e. decision making.

One of the most visited social networking sites by millions of users is twitter where they share their opinion about various domains like politics, brands, products, celebrities etc. Many research works are carried out in the field of sentiment analysis. But they are only useful in modeling and tracking public sentiments. They had not found exact reasons behind the sentiment variations and hence not useful in decision making.

Sentiment analysis has many applications in various domains like political domain, sociology and real time event detection like earthquakes. Previously research was carried out to model and track public sentiments. But with the advancement in research, today we can use it for interpreting the reasons of the sentiment change in public opinion, mining and summarizing products reviews, to solve the polarity shift problem by performing dual sentiment analysis. Here we use different algorithms/models to perform the above tasks like Naïve Bayes (NB) classifier, Bernoulli Naïve Bayes classifier, Decision Tree classifier algorithm and so on.

ACKNOWLEDGEMENT

We wish to place on record our deep sense of gratitude to our honorific Guide **Mr. J. ARUL VALAN** , Assistant Professor, National Institute of Technology Nagaland, for his supervision, valuable guidance and moral support leading to the successful completion of the work. Without his continuous encouragement and involvement, this project would not have been a reality.

We would like to extent my sincere thanks to **Mr. NAGARAJU BAYDETI** & **Mr. DILWAR HUSSAIN MAZUMDAR** for their valuable suggestions. We wish to thank **Dr. THEMRICHON TUITHUNG**, Department of Computer Science & Engineering, NIT Nagaland for continuous support. We would also like to thank all our friends who have developed us to gain a sense of dutifulness, perfection and sincerity in the effort .

We would like to thank our parents who have motivated us to work harder and do our best. Last but not least, we would like to owe our sincere and incessant gratitude to the almighty God for the immense blessing on us.

LIPOKMERENLA,

DURGA SHANKAR MISHRA

&

INDRASEN SINGH

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	ii
	LIST OF TABLES	vi
	LIST OF FIGURES	vii
	LIST OF ABBREVIATIONS	vii
1	**INTRODUCTION**	1
	1.1 GENERAL	1
	1.2 STATE OF THE ART	3
	1.3 OBJECTIVES	4
	1.4 ORGANIZATION OF THE REPORT	4
2	**PREPROCESSING**	6
	2.1 INTRODUCTION	6
	2.2 BASICS CONCEPT OF TEXT CLASSIFICATION	7
	2.2.1 TEXT PREPROCESSING	7
	2.2.2 TOKENIZATION	7
	2.2.3 STOP WORD REMOVAL	7
	2.2.4 STREAMING	8
	2.2.5 LOWER CASE CONVERSION	8
	2.3 FEATURE EXTRACTOR	8
	2.4 CONCLUSION	10
3	**NAIVE BAYES CLASSIFIER**	11
	3.1 INTRODUCTION	11
	3.2 ALGORITHM	12
	3.2.1 NAIVE BAYES TEXT CLASSIFICATION	14

CHAPTER NO.	TITLE	PAGE NO.
	3.3 TYPES OF NAIVE BAYES MODEL	17
	3.4 APPLICATION OF NAIVE BAYES MODEL	18
	3.5 TIPS TO IMPROVE THE NAIVE BASE MODEL	18
	3.6 CONCLUSION	19
4	**DECISION TREE CLASSIFIER**	20
	4.1 INTRODUCTION	20
	4.2 ALGORITHM	22
	4.2.1 ENTROPY	22
	4.2.2 INFORMATION GAIN(IG)	22
	4.3 DECISION TREE TO DECISION RULES	23
	4.4 ADVANTAGES AND DISADVANTAGES	23
	4.5 DECISION TREE CLASSIFIER ISSUES	25
	4.6 DECISION TREE CLASSIFIER APPLICATION	25
	4.7 CONCLUSION	27
5	**TOOLS AND IMPLEMENTATION**	28
	5.1 INTRODUCTION	28
	5.2 NLTK	29
	5.3 PYTHON PROGRAMMING LANGUAGE	29
	5.4 IMPLEMENTATION DETAILS	30
6	**CONCLUSION**	33
	6.1 INTRODUCTION	33
	6.2 HIGHLIGHTS OF THE WORK DONE	33
	6.3 FUTURE WORK	34
	REFERENCES	35

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO
1.1	Classes of Sentiment Analysis	3

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
1.1	Example of Sentiment Analysis	2
2.1	Steps and Techniques used in SC	6
3.2	Example of Naive Bayes	13
3.2.1	Learning to Classify Text	17
4.1.1	Example of Decision Tree	21
5.4	Accuracies of the various classifiers.	31
5.5	Sentiment Trend	32

LIST OF ABBREVIATIONS

NB	-	Naïve Bayes
BNB	-	Bernoulli Naïve Bayes
DTC	-	Decision Tree Classifier
NLTK	-	Natural Language Tool Kit
IDE	-	Integrated Development Environment
MNB	-	Multinomial Naïve Bayes
IG	-	Information Gain
SC	-	Sentiment Classification
LRC	-	Logistic Regression Classifier
DTC	-	Decision Tree Classifier
SDGC	-	Stochastic Gradient Descent Classifier

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Sentiment Analysis is to detect the polarity of text in consideration in textual form. It is also known as opinion mining as it derives the opinion of the speaker or the user about some topic. In other words, it determines whether a piece of writing is positive, negative or neutral. For example, do people on Twitter think that prime minister Shri Narendra Modi is doing his job properly or not? To find out the answer we can refer the social networking site twitter. There are millions of opinions of people about Shri Narendra Modi, some of them positive and some will be negative or neutral. We can get the exact ideas of why people think Modi Ji is fulfilling his responsibilities or not, by extracting the exact word indicating the positive or negative opinion.

It can be carried out at various levels like document level, phrase level or sentence level. When the sentence consists of positive as well as negative sentiments at word level, the whole sentence becomes neutral at sentence level. As the sentiment analysis on twitter or any social media site tracks particular topic, many politicians as well as companies use twitter to track their position in politics and monitor their products and services respectively. The major benefit of sentiment analysis in previous work was to find out whether the expressed opinion in the document or sentence is positive, negative or neutral. But it was not useful in decision making as no reasons were known about why the sentiments has changed. Hence there was need to build a system for interpreting the public sentiment variations.

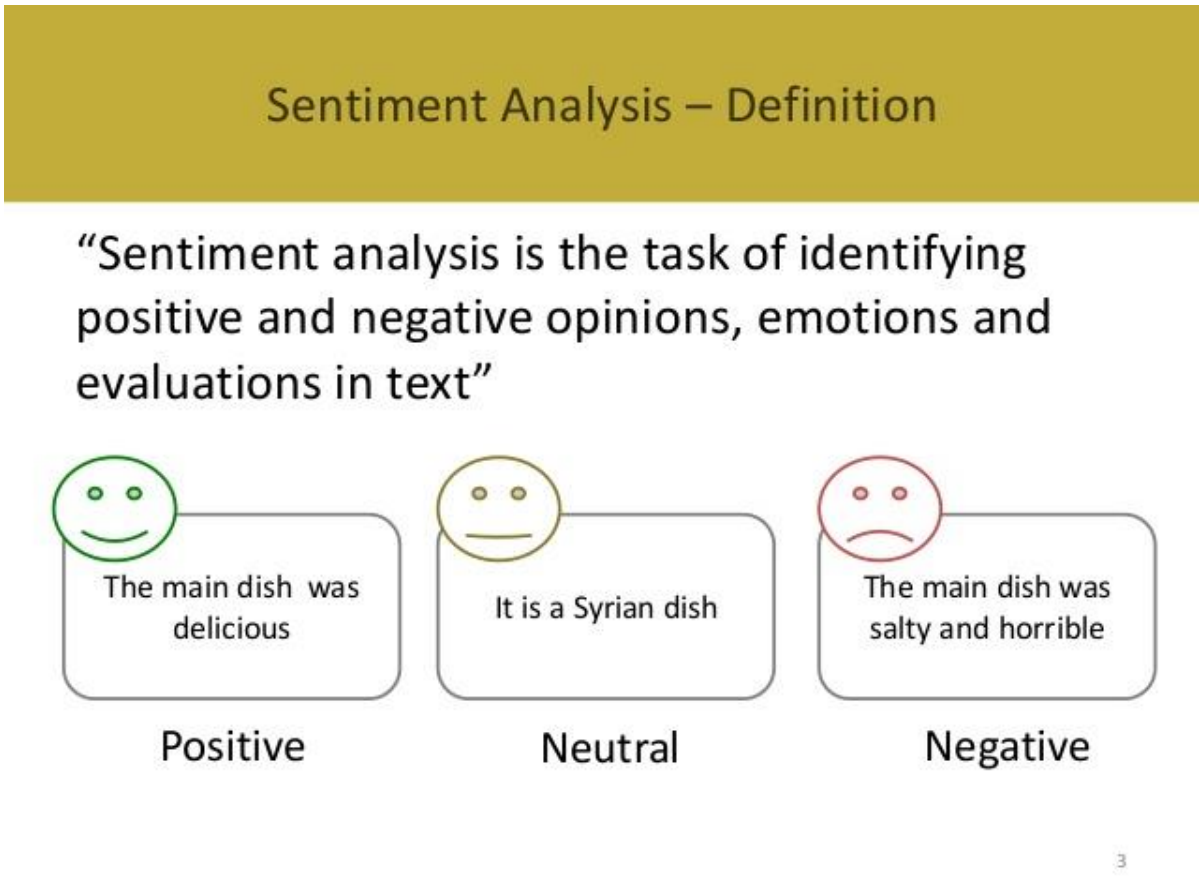


Figure 1.1 Example of Sentiment Analysis

Different Classes of Sentiment Analysis

Sentiments can be classified into three classes .i.e. positive, negative and neutral sentiments.

a. **Positive Sentiments**: These are the good words about the target in consideration. If the positive sentiments are increased, it is referred to be good. In case of product reviews, if the positive reviews about the product are more, it is bought by many customers.

b. **Negative Sentiments**: These are the bad words about the target in consideration. If the negative sentiments are increased, it is discarded from the preference list. In case of product reviews, if the negative reviews about the product are more, no one intend to buy it.

c. **Neutral Sentiments**: These are neither good nor bad words about the target. Hence it is neither preferred nor neglected.

POSITIVE	• Positive indicator on topic
NEUTRAL	• Neither positive nor negative indicators • Mixed positive and negative indicators • On topic, but indicator undeterminable • Simple factual statements • Questions with no strong emotions
NEGATIVE	• Negative indicator on topic
IRRELEVANT	• Not English language • Not on-topic (e.g. spam)

Table 1.1 Classes of Sentiment Analysis

1.2 STATE OF THE ART

In this section, literature survey on previous works on twitter sentiment classification presented.

Sentiment analysis is the most important research area in business fields. Previously research was carried out for sentiment analysis in various domains like company product, movie reviews, politics etc. Previous research like **Pang et al**. has provided with the baseline for carrying out research in various domains. It uses star ratings as polarity signals in their training data. Even many authors have used

T.Sakaki et al. developed an event notification system which monitors the tweets and delivers notifications considering the time constraint. They detect real-time events in Twitter such as earthquakes. They have proposed an algorithm to

monitor tweets detecting target event. Each Twitter user is considered as a sensor. Kalman filtering and particle filtering are used for estimation of location.

Esuli and Sebastiani 2006)). Given the character limitations on tweets, classifying the sentiment of Twitter messages is most similar to sentence-level sentiment analysis .

1.3 OBJECTIVES

First objective of this problem mainly is to extract the tweets from twitter micro blogging website, followed by preprocessing of tweets which includes removal of stop words, acronym, emoticons and url(https://twitter.com/) & username(example. @akshaykumar). Then on the processed tweets we generate the word feature set.

Primary objective of our problem mainly is to implement an algorithm for automatic classification of text into positive ,negative or neutral. Techniques viz. Naïve Bayes(NB), Bernoulli Naïve Bayes(BNB) & Decision Tree [Information Gain (IG), Gain Ratio (GR)] Classifier on sample tweets using unigram (one word) and bigram (merge two word eg. Not bad). After execution, implement it with sample data, use these algorithms for predicting the polarity of test data. Suggestions to enhance these techniques in order to implement it in an efficient way like working out in a less time.

1.4 ORGANIZATION OF THE REPORT

The rest of the thesis is organized as follows.

- **Chapter 1:** Gives the introduction of the theory, brief introduction about the project and the review of related work.

- **Chapter 2:** Give the details about the importance of preprocessing of the tweets and the various approaches, how to achieve it, and about feature selection.
- **Chapter 3:** Introduction to Naive Bayes Classifier , its Algorithm and various types.
- **Chapter 4:** Introduction to the Decision Tree Classifier, its Algorithm and its features.
- **Chapter 5:** Explanation about the tools and the implementation methods.
- **Chapter 6:** Conclusion, highlights of the work and Future work explanation.

CHAPTER 2

PREPROCESSING

2.1 INTRODUCTION

An overview of sequential steps and techniques commonly used in sentiment classification approaches, as shown in Figure 2.1.Parts of speech is a model which aims to classify roles that means according to parts of speech has also been explored. In this model , information is used as part of a feature set which leads to sentiment classification on a dataset

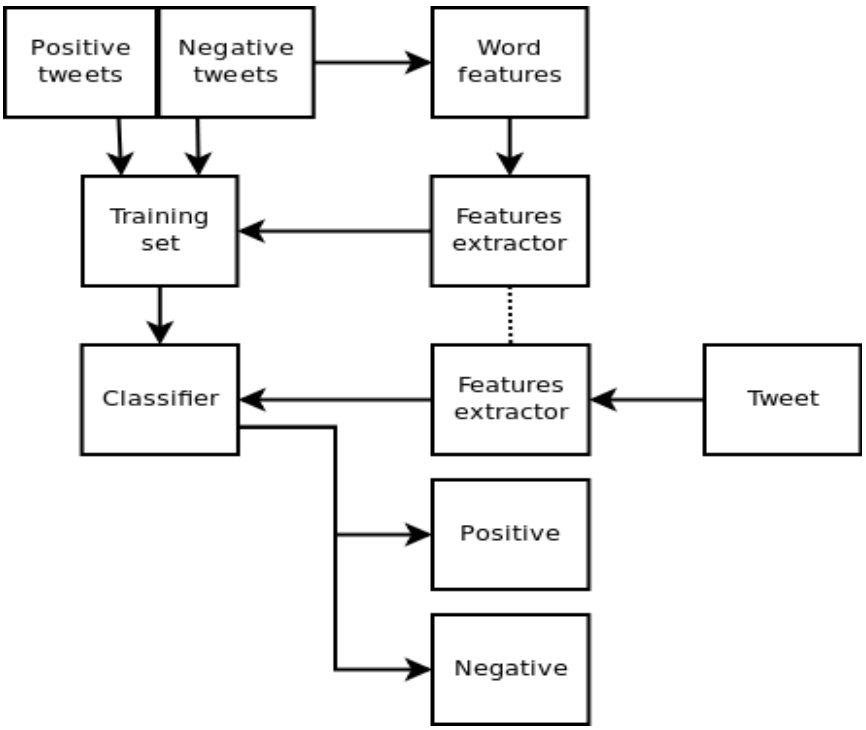


Figure.2.1 Steps and techniques used in sentiment classification.

2.2 BASICS CONCEPT OF TEXT CLASSIFICATION

- Text Preprocessing
- Tokenization
- Stop Word Removal
- Streaming
- Lower Case Conversion

2.2.1. Text Preprocessing

Preprocessing of data is the process of preparing and cleaning the data of dataset for classification. Here is the hypothesis of having the data properly pre-processed: to reduce the noise in the text should help improve the performance of the classifier and speed up the classification process, thus aiding in real time sentiment analysis.

2.2.2. Tokenization:

Given input as character sequence, tokenization is a task of chopping it up into pieces called tokens and at the same time removing certain characters such as punctuation marks. A token is an instance of sequence of characters that are grouped together as a useful semantic unit for processing.

2.2.3. Stop word removal

A stop-list is the name commonly given to a set or list of stop words. It is typically language specific, although it may contain words. A search engine or other natural language processing system may contain a variety of stop-lists, one per language, or it may contain a single stop-list that is multilingual. Some of the more frequently used stop words for English include "a", "of", "the", "I", "it", "you", and ”and” these are generally regarded as 'functional words' which do not carry meaning. When assessing the contents of natural language, the meaning can be conveyed more clearly by ignoring the functional words .Hence it is practical to remove those words which appear too often that support no information for the task.

2.2.4. Stemming*:*

It is the process for reducing derived words to their stem, or root form. Stemming programs are commonly referred to as stemmers or stemming algorithms. A simple stemmer looks up the inflected form in a lookup table, this kind of approach is simple and fast. The disadvantage is that all inflected forms must be explicitly listed in table.eg. “developed”, “development” , ”developing” are reduced to the stem “develop”.

2.2.5. Lowercase Conversion:

Tweet may be normalized by converting it to lowercase which makes it’s comparison with an English dictionary easier

2.5 FEATURE EXTRACTOR:

Feature Selection is used to make classifiers more efficient by reducing the amount of data to be analyzed as well as identifying relevant features to be considered in classification process. Ideally, feature selection stage will refine features, which are input into a classification / learning process.

- Identify the parts of corpus to contribute to positive and negative sentiment.
- Join these parts of corpus in such a way that the document falls into one of these polar categories.

Feature selection methods are grouped into four main categories NLP or heuristic based, Statistical, Clustering based and Hybrid. Natural language processing based techniques mainly operate on three basic principles:

(a) Noun, noun phrases, adjectives, adverbs usually express product features
(b) Terms occurring near subjective expressions can act as features.
(c) P is product and F is feature in phrases like ‘F of P’ or ‘P has F’.

They have got high accuracy, but low recall with dependency on accuracy of part of speech of tagging. Clustering or Machine Learning based feature extraction techniques are implemented by, requiring few parameters to tune. Key weakness of clustering is that only major features can be extracted and it is difficult to extract minor features.

Statistical techniques are further divided into three sub types, univariate, multivariate and hybrid. Univariate methods, also called feature filtering methods, take attributes separately, examples of this type include information gain (IG), chi–square, occurrence frequency, log likely-hood and minimum frequency thresholds.

Univariate techniques have computational efficiency, but they ignore attribute interactions. Decision tree models, recursive feature elimination and genetic algorithms are the examples of multivariate methods, which consider group of attributes and use wrapper model for attribute selection. As compared to univariate, multivariate methods are expensive in terms of computational efficiency, as they evaluate attribute interactions. Hybrid techniques combine univariate multivariate and other methods for achieving accuracy and efficiency

After the text has been segmented into sentences, each sentence has been segmented into words, the words have been tokenized and normalized, we can make a simple bag-of-words model of the text. In this bag-of-words representation you only take individual words into account and give each word a specific subjectivity score. This subjectivity score can be looked up in a sentiment lexicon. If the total score is negative the text will be classified as negative and if it’s positive the text will be classified as positive.

2.6 CONCLUSION:

In order to fit our model to our dataset we need to clean and process our data. it helps to elevate the performance of sentiment analysis and helps to reduce the dimensionality it helps to compute the results faster we have mentioned a few ways how to preprocess the data Keeping those words makes the dimensionality of the problem high and hence the classification more difficult since each word in the text is treated as one dimension. Here is the hypothesis of having the data properly pre-processed: to reduce the noise in the text should help improve the performance of the classifier and speed up the classification process, thus aiding in real time sentiment analysis.

CHAPTER 3

NAIVE BAYES CLASSIFIER

3.1 INTRODUCTION

In machine learning, **Naive Bayes classifiers** are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong(naive) independence assumptions between the features. Naive Bayes has been studied extensively since the 1950s. It was introduced under a different name into the text retrieval community in the early 1960s, and remains a popular (baseline) method for text categorization, the problem of judging documents as belonging to one category or the other (such as spam or legitimate, sports or politics, etc.) with word frequencies as the features. With appropriate pre-processing, it is competitive in this domain with more advanced methods including support vector machines. It also finds application in automatic medical diagnosis.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics and computer science literature, Naive Bayes models are known under a variety of names, including **simple Bayes** and **independence Bayes**. All these names reference the use of Bayes' theorem in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method.

. Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for

training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For example, a fruit may be considered to be an apple if it is red, round, and about 10 cm in diameter. A naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the colour, roundness, and diameter features.

For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without accepting Bayesian probability or using any Bayesian methods.

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. In 2004, an analysis of the Bayesian classification problem showed that there are sound theoretical reasons for the apparently implausible efficacy of naive Bayes classifiers. Still, a comprehensive comparison with other classification algorithms in 2006 showed that Bayes classification is outperformed by other approaches, such as boosted trees or random forests.

An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification

3.2 ALGORITHM

Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c)$, $P(x)$, and $P(x|c)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors.

This assumption is called class conditional independence.

$$P(c|x) = \frac{p(x|c)p(c)}{p(x)}$$

$$P(c|x) = P(x_1|c) \times P(x_2|c) \times \ldots\ldots\ldots \times P(x_n|c) \times P(c)$$

- P(c|x) is the posterior probability of class (target) given predictor (attribute).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- *P*(*x*) is the prior probability of predictor.

Example:

The posterior probability can be calculated by first, constructing a frequency table for each attribute against the target. Then, transforming the frequency tables to likelihood tables and finally use the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

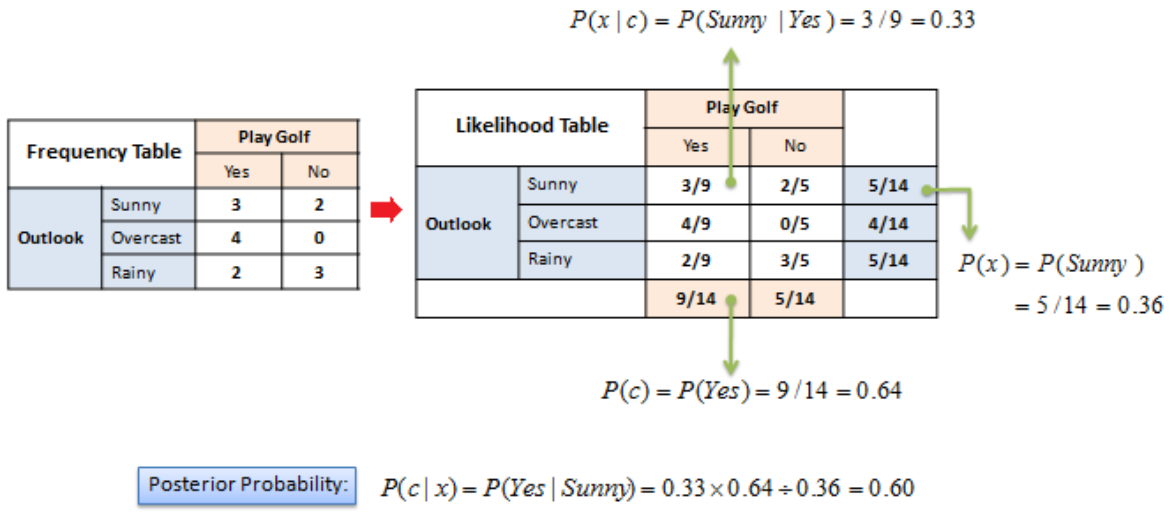


Figure 3.2 Example of Naïve Bayes

The zero-frequency problem

Add 1 to the count for every attribute value-class combination (Laplace estimator) when an attribute value (Outlook=Overcast) doesn't occur with every class value (Play Golf=no).

3.2.1. Naive Bayes Text Classification

The first supervised learning method we introduce is the multinomial Naive Bayes or multinomial NB model, a probabilistic learning method. The probability of a document **d** being in class **c** is computed as

$$p(c|d) \propto P(c) \prod_{1 \leq k \leq nd} p(t_k|c)$$

where $P(t_k|c)$ is the conditional probability of term t_k occurring in a document of class c .We interpret $P(t_k|c)$ as a measure of how much evidence t_k contributes that c is the correct class. $P(c)$ is the prior probability of a document occurring in class c.

In text classification, our goal is to find the best class for the document. The best class in NB classification is the most likely or maximum a posteriori (MAP) class $\mathbf{c_{map}}$:

$$C_{MAP} = argmax_{c \in C}\, p(c|d) = argmax_{c \in C}\, p(c) \prod_{1 \leq k \leq n_d} p(t_k|c).$$

many conditional probabilities are multiplied, one for each position $1 \leq k \leq n_d$. This can result in a floating point underflow. It is therefore better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. The class with the highest log probability score is still the most probable; $log(xy)=log(x)+log(y)$ and the logarithm function is monotonic. Hence, the maximization that is actually done in most implementations of NB is:

$$C_{map} = argmax \left[\log p(c) + \sum_{1 \le k \le n_d} \log p(t_k\, c) \right] \quad (1)$$

Equation 1 has a simple interpretation. Each conditional parameter $logP(t_k|c)$ is a weight that indicates how good an indicator t_k is for c . Similarly, the prior $logP(c)$ is a weight that indicates the relative frequency of c. More frequent classes are more likely to be the correct class than infrequent classes. The sum of log prior and term weights is then a measure of how much evidence there is for the document being in the class, and Equation 1 selects the class for which we have the most evidence.

How do we estimate the parameters $P(c)$ and $P(t_k|c)$? We first try the maximum likelihood estimate (MLE; prob theory), which is simply the relative frequency and corresponds to the most likely value of each parameter given the training data. For the priors this estimate is:

$$p(c) = \frac{N_c}{N}$$

where N_c is the number of documents in class c and N is the total number of documents.

We estimate the conditional probability $P(t_k|c)$ as the relative frequency of term t in documents belonging to class c:

$$P(t|c) = \frac{T_{ct}}{\sum t' \epsilon v \;\; T_{ct'}}$$

where T_{ct} is the number of occurrences of t in training documents from class c, including multiple occurrences of a term in a document. We have made the positional independence assumption here, which we will discuss in more detail in the next section: T_{ct} is a count of occurrences in all positions k in the documents in the training set. Thus, we do not compute different estimates for different positions and,

for example, if a word occurs twice in a document, in positions $\mathbf{k_1}$ and $\mathbf{k_2}$, then $\mathbf{P(t_{k1}|c) = P(t_{k2}|c)}$.

Naive Bayes algorithm (multinomial model): Training and testing.

TRAINMULTINOMIALNB(C,D)

1. V ← ExtractVocabulary(D)
2. N ← CountDocs(D)
3. **for each** c ϵ **C**
4. **do** N_C ← CountDocsInClass(D,C)
5. *prior*[c] ← N_C/N
6. $\boldsymbol{text_c}$ ← ConcatenateTextOfAllDocsInClass(D,C)
7. **for each** t Є **V**
8. **do** T_{CT} ← CountTokensOfTerm($text_c, t$)
9. **for each** t Є **V**
10. **do** *condpro*b[t][c] ← $\frac{T_{CT}+1}{\sum_{t'}(T_{CT'}+1)}$
11. **return V**, *prior*, *condprob*

APPLYMULTINOMILANB(C,V, *prior*,*condprob*,*d*)

1. W ← ExtractNomialNBFormDoc(V,d)
2. **for each** c Є C
3. **do** *score*[c] ← log *prior*[c]
4. **for each** t Є W
5. **do** *score*[c] += log *condprob*[t][c]
6. **return argmax**$_{c\ \epsilon\ C}$ *score*[c]

To eliminate zeros, we use add-one or Laplace smoothing, which simply adds one to each count :

$$P(t|c) = T_{ct} + \frac{1}{\sum_{t' \epsilon v}(T_{ct'} + 1)} + \frac{T_{ct} + 1}{\sum_{t' \epsilon v} T_{ct'} + B'}$$

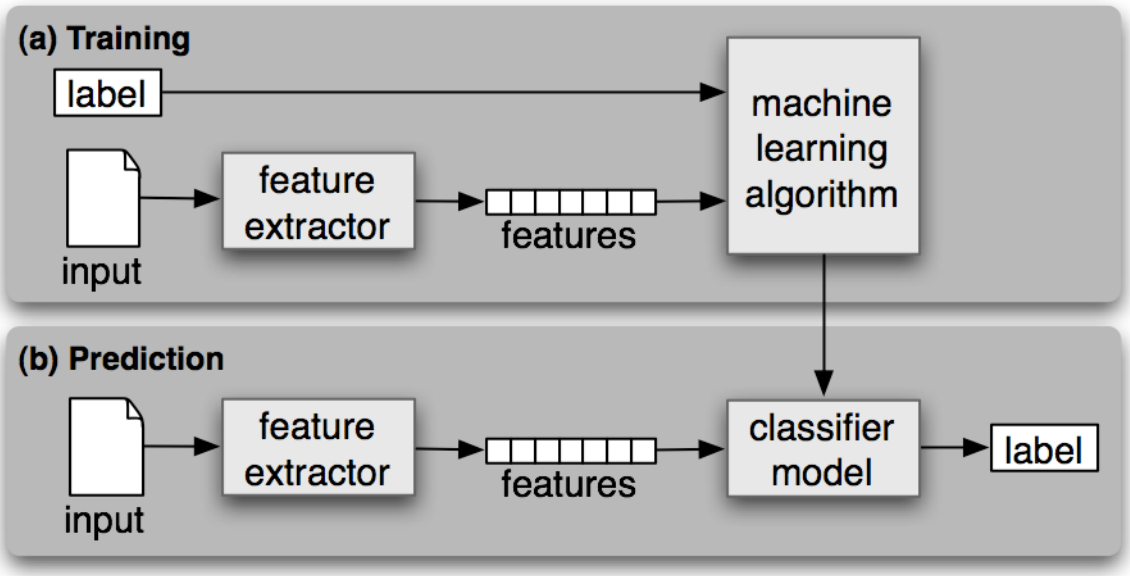


Figure 3.2.1 Learning to classify text

3.3 TYPES OF NAÏVE BAYES MODEL

- **Gaussian:** It is used in classification and it assumes that features follow a normal distribution.

- **Multinomial:** It is used for discrete counts. For example, let's say, we have a text classification problem. Here we can consider bernoulli trials which is one step further and instead of "word occurring in the document", we have "count how often word occurs in the document", you can think of it as "number of times outcome number x_i is observed over the n trials".

- **Bernoulli:** The binomial model is useful if your feature vectors are binary (i.e. zeros and ones). One application would be text classification with 'bag of words' model where the 1s & 0s are "word occurs in the document" and "word does not occur in the document" respectively.

3.4 APPLICATION OF NAÏVE BAYES

- **Real time Prediction:** Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- **Multi class Prediction:** This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- **Text classification/ Spam Filtering/ Sentiment Analysis:** Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- **Recommendation System:** Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not

3.5 METHODS TO IMPROVE THE NAÏVE BAYES MODEL

- If continuous features do not have normal distribution, we should use transformation or different methods to convert it in normal distribution.
- If test data set has zero frequency issue, apply smoothing techniques "Laplace Correction" to predict the class of test data set.
- Remove correlated features, as the highly correlated features are voted twice in the model and it can lead to over inflating importance.

- Naive Bayes classifiers has limited options for parameter tuning like alpha=1 for smoothing, fit_prior=[True|False] to learn class prior probabilities or not and some other options (look at detail here). I would recommend to focus on your pre-processing of data and the feature selection.
- You might think to apply some classifier combination technique like ensembling, bagging and boosting but these methods would not help. Actually, “ensembling, boosting, bagging” won’t help since their purpose is to reduce variance. Naive Bayes has no variance to minimize.

3. 6 CONCLUSION

We have delved into building and understanding a Naive Bayesian Classifier. As we have learned it, this algorithm is well suited for data that can be asserted to be independent. Being a probabilistic model, it works well for classifying data into multiple directions given the underlying score. This supervised learning method is useful for text classification and sentiment analysis.

CHAPTER 4

DECISION TREE CLASSIFIER

4.1 INTRODUCTION

A Decision Tree is a graphical representation of possible solutions to a decision based on certain conditions. It's called a decision tree because it starts with a single box (or root), which then branches off into a number of solutions, just like a tree.

Decision Trees are helpful, not only because they are graphics that help you 'see' what you are thinking, but also because making a decision tree requires a systematic, documented thought process. Often, the biggest limitation of our decision making is that we can only select from the known alternatives. Decision trees help formalize the brainstorming process so we can identify more potential solutions.

Suppose you have a population. You want to divide this population into relevant subgroups based on specific features characterizing each subgroup, so that you can accurately predict outcomes associated with each subgroup. For instance, you could:

- Use the list of the people on the Titanic, and by dividing them into subgroups depending on specific criteria (e.g. female vs. male, passengers in 1st class vs. 2nd and 3rd class, age class....) determines if they were (probably) going to survive or not**.**
- Look at the people who bought product on your e-commerce website, divide this population into segments depending on specific features (e.g. returning visitors vs. new visitors, localization, ...) and determines for future visitors if they are (probably) going to buy your product or not.

In sum, you want to create a model that **predicts the value of a target variable** (e.g. survive/die; buy/not buy) based on simple decision rules inferred from the data **features** (e.g. female vs. male, age, etc.).

The result is a **decision tree** that offers the great advantage to be easily visualized and simple to understand. For instance, the picture below, from Wikipedia, shows the probability of passengers of the Titanic to survive depending on their sex, age and number of spouses or siblings aboard. Note how each branching is based on **answering a question (the decision rule)** and how the graph looks like an inverted **tree**.

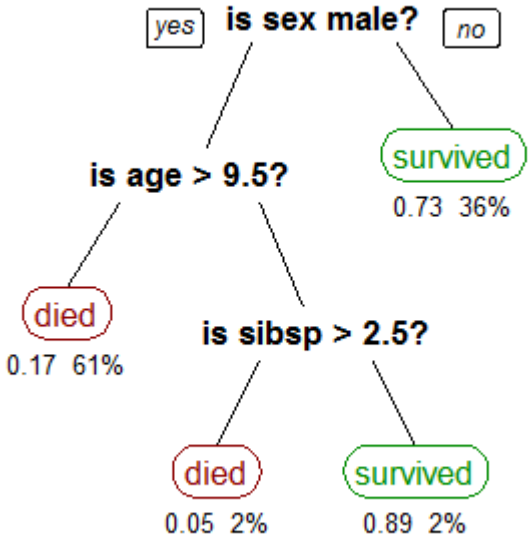


Figure 4.1.1 Example of Decision Tree

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision nodes** and **leaf nodes**. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision.

The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.

4.2 ALGORITHM

The core algorithm for building decision trees called **ID3** by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct a decision tree.

4.2.1 Entropy

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 algorithm uses entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.

4.2.2 Information Gain

The information gain is based on the decrease in entropy after a dataset is split on an attribute. Constructing a decision tree is all about finding attribute that returns the highest information gain (i.e., the most homogeneous branches).

Step 1: Calculate entropy of the target.

Step 2: The dataset is then split on the different attributes. The entropy for each branch is calculated. Then it is added proportionally, to get total entropy for the split.

The resulting entropy is subtracted from the entropy before the split. The result is the Information Gain, or decrease in entropy.

Step 3: Choose attribute with the largest information gain as the decision node.

4.3 DECISION TREE TO DECISION RULES

A decision tree can easily be transformed to a set of rules by mapping from the root node to the leaf nodes one by one.

4.4 ADVANTAGES & DIS ADVANTAGES

Some advantages of decision trees are:

- Simple to understand and to interpret. Trees can be visualized.
- Requires little data preparation. Other techniques often require data normalization, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data. Other techniques are usually specialized in analyzing datasets that have only one type of variable. See algorithms for more information.
- Able to handle multi-output problems.
- Uses a white box model. If a given situation is observable in a model, the explanation for the condition is easily explained by Boolean logic. By contrast, in a black box model (e.g., in an artificial neural network), results may be more difficult to interpret.

- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.

The disadvantages of decision trees include:

- Decision-tree learners can create over-complex trees that do not generalize the data well. This is called over fitting. Mechanisms such as pruning (not currently supported), setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
- Decision trees can be unstable because small variations in the data might result in a completely different tree being generated. This problem is mitigated by using decision trees within an ensemble.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.
- There are concepts that are hard to learn because decision trees do not express them easily, such as XOR, parity or multiplexer problems.
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

4.5 DECISION TREE CLASSIFIER -ISSUES

- Working with continuous attributes (binning)
- Avoiding over fitting
- Super Attributes (attributes with many values)
- Working with missing values

4.6 DECISION TREE CLASSIFIER APPLICATION

- **Agriculture:** Application of a range of machine learning methods to problems in agriculture and horticulture is described in .
- **Astronomy:** Astronomy has been an active domain for using automated classification techniques. Use of decision trees for filtering noise from Hubble Space Telescope images. Decision trees have helped in star-galaxy classification , determining galaxy counts and discovering quasars in the Second Palomar Sky Survey. Use of neural trees for ultraviolet stellar spectral classification is described in .
- **Biomedical Engineering:** Use of decision trees for identifying features to be used in implantable devices can be found in .
- **Control Systems:** Automatic induction of decision trees was recently used for control of nonlinear dynamical systems .
- **Financial analysis:** Use of CART for asserting the attractiveness of buy-writes is reported in .
- **Manufacturing and Production:** Decision trees have been recently used to non-destructively test welding quality , for semiconductor manufacturing , for increasing productivity , for material procurement method selection , to accelerate rotogravure printing , for process optimization in electrochemical machining , to schedule printed circuit board assembly lines , to uncover flaws in a Boeing manufacturing process and for quality control . For a recent review of the use of machine learning (decision trees and other techniques) in scheduling, see .

- **Medicine:** Medical research and practice have long been important areas of application for decision tree techniques. Recent uses of automatic induction of decision trees can be found in diagnosis , cardiology , psychiatr, gastroenterology , for detecting microcalcifications in mammography , to analyze Sudden Infant Death (SID) syndrome and for diagnosing thyroid disorders .
- **Molecular biology:** Initiatives such as the Human Genome Project and the GenBank database offer fascinating opportunities for machine learning and other data exploration methods in molecular biology. Recent use of decision trees for analyzing amino acid sequences can be found in and .
- **Object recognition:** Tree based classification has been used recently for recognizing three dimensional objects and for high level vision .
- **Pharmacology:** Use of tree based classification for drug analysis can be found in .
- **Physics:** Decision trees have been used for the detection of physical particles .
- **Plant diseases:** CART was recently used to assess the hazard of mortality to pine trees .

4.7 CONCLUSION

We have delved into building and understanding a Decision Tree Classifier. As we have learned it, this algorithm is well suited for data that can be asserted to be independent. Being a probablistic model, it works well for classifying data into multiple directions given the underlying score. This supervised learning method is useful for text classification and sentiment analysis.

CHAPTER 5

TOOLS AND IMPLEMENTATION

5.1 INTRODUCTION

The Feature selection algorithms such as Naive Bayes, and Decision Tree(DT) are implementable in Python programming language .The version of python 3 is required .In order to run the program in any computer we need to download the NTLK(Natural Language tool kit) packages, and many other required packages.

The feature set generated is very large in size , so it is recommended to use a computer with high processing power and has more RAM, the desired configuration is i7 processor and at least 8GB of RAM.

We are using the tools details are as follows:

- Software
    1. Natural Language Toolkit(NLTK)
    2. Any operating system
    3. Python programming language
- Hardware
    1. At least 8 GB RAM
    2. At least i7 processor

5.2 NATURAL LANGUAGE TOOLKIT :

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

5.3 PYTHON (PROGRAMMING LANGUAGE) :

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

5.4 IMPLEMENTATION DETAILS

Proposed implementation has been implemented in the following way-

1. Gather the tweets.

2. We need the manually separated tweets for training and testing.

3. Randomly select 2/3 of sample data for training purpose and select the remaining 1/3 of the sample data for testing purpose and to get the accuracy of the classifiers.

4. In order to process the tweets efficiently we need to preprocess the tweets in order to achieve better results.
   Preprocessing includes the following steps
   - Convert to lower case
   - Convert www.* or https?://* to URL
   - Convert @username to AT_USER
   - Remove additional white spaces
   - Replace #word with word
   - trim
   - Remove Stop-Words

5. Once the tweets are being pre-processed , word-features are formed, which leads to the formation of feature vector , it in training and predicting the results

6. The feature vector is given to the various classifiers in order to train the classifiers on the training set.

7. Once the training is completed , the accuracies of the various classifiers are found out, it helps us to understand which classifier is giving us better results.

8. Plot the accuracy of the various classifiers.

9. We can check the sentiment of the tweets on these classifiers.

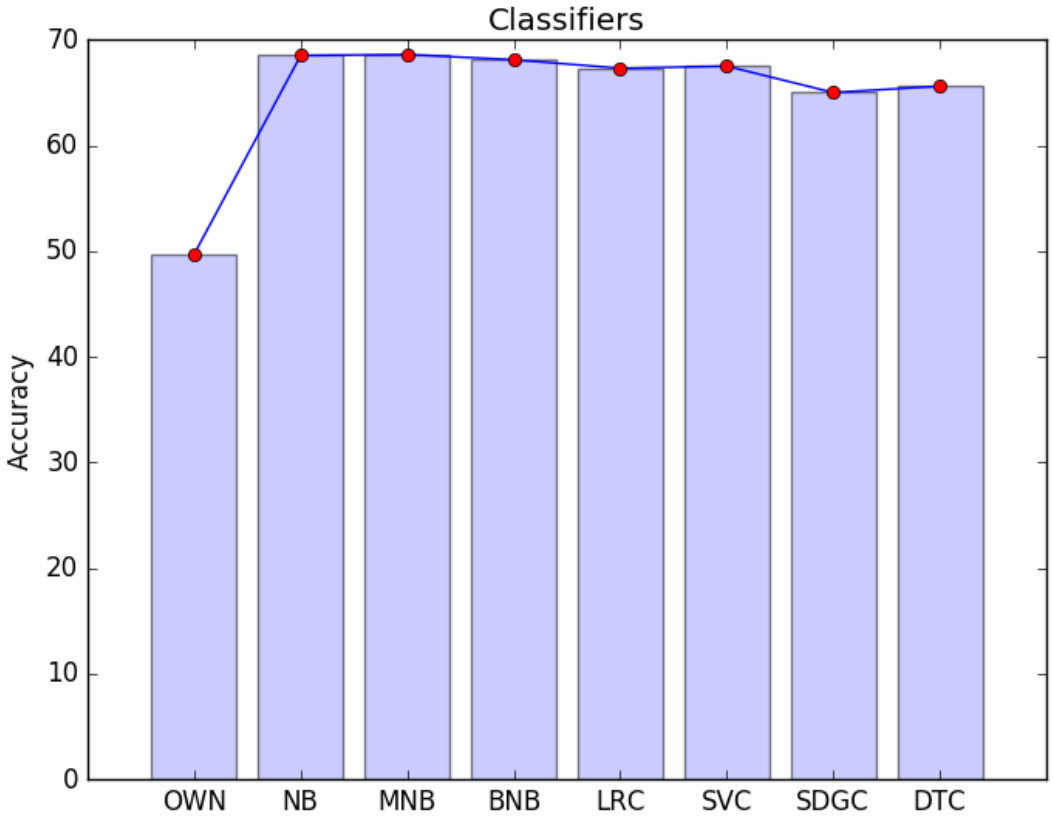


Figure 5.4: Accuracies of the various classifiers.

10. Upon training the classifiers we check the performance of the overall system , we do so by taking the results from all the classifiers and taking the majority voting of the results to decide the overall sentiment of the tweets .

11. We perform the test on the live tweets from tweeter , gained by twitter api .

12. We display the sentiment of the tweet and the confidence value returned by the various classifiers .

13. Once we get the results we plot the sentiment of the tweets on the live graph.

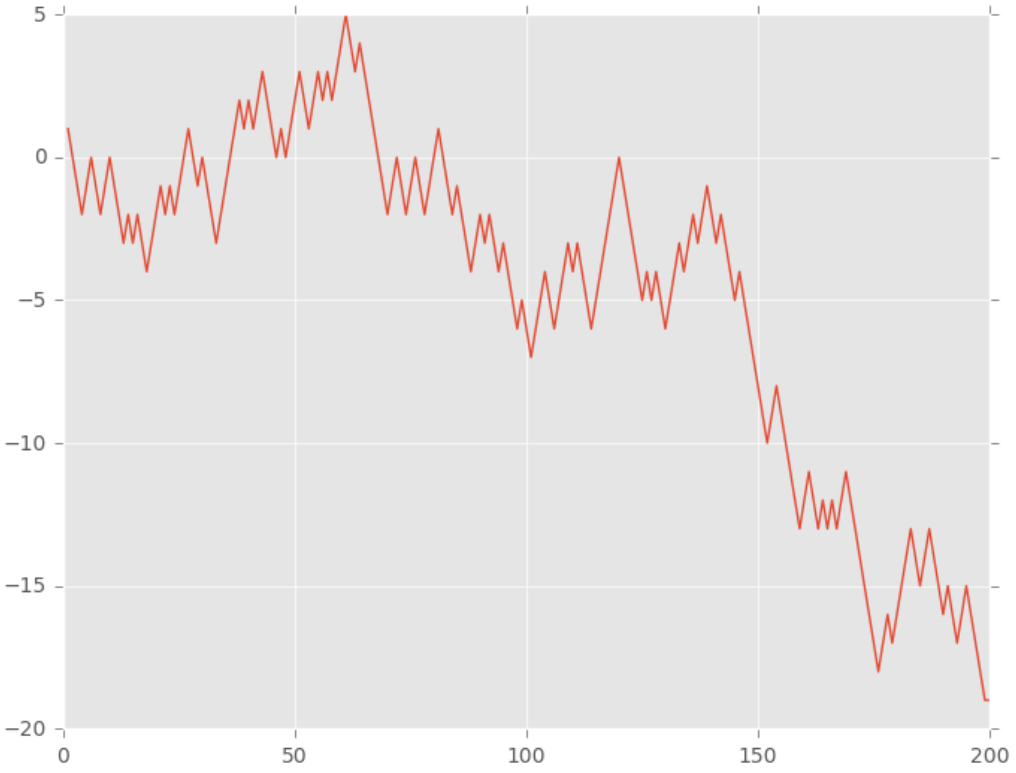


Figure 5.5 : Sentiment Trend of topic "music"

14. We also display the tweets with their corresponding sentiment and confidence values on the screen .

15. We store the meaningful tweets on .csv file for future references .

CHAPTER 6

CONCLUSION

6.1 INTRODUCTION

The main aim of Intelligent sentiment classification on tweets is to visualize the sentiment trends and intensities on tweets. Sentiment analysis deals with identifying and classifying opinions or sentiments expressed in source text. Sentiment analysis, as an interdisciplinary field that crosses natural language processing, artificial intelligence, and text mining. Sentiment analysis has Strong commercial interest because Companies want to know how their products are being perceived and also Prospective consumers want to know what existing users think. Sentiment analysis of user generated data is very useful in knowing the opinion of the crowd. Twitter sentiment analysis is difficult compared to general sentiment analysis due to the presence of slang words and misspellings. The maximum limit of characters that are allowed in Twitter is 140. Knowledge base approach and Machine learning approach are the two strategies used for analyzing sentiments from the text.

6.2 HIGHLIGHTS OF THE WORK DONE

We conclude that using different NLTK classifier it is easier to classify the tweets and more we improve the training data set more we can get more accurate results. Different classifiers are used in order to check the performance. From that observation we found out the following the illustrations, performance of various

classifiers are varying , we have build our own classifier , which is very similar to Naive Bayes, and then we are comparing its accuracies with various other classifiers that we have explained above, like Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes ,Decision Tree Classifier. The performance of our classifier is giving almost same performance as other classifiers.

Upon training our various classifiers , we are testing it on various live tweets from twitter . And we are plotting the result of the sentiment achieved on the live graph , and storing the valuable tweets on a csv file . The performance of the classifier that we have build is descent .

6.3 FUTURE WORK

- The accuracy of the overall system can be improved.
- Need to take care of the bigram .
- Need to improve the performance of the word feature , so it can predict more accurately.

REFERENCES

[1] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury, “Twitter power: Tweets as electronic word of mouth,” Journal of the American society for information science and technology, vol. 60, no. 11, pp. 2169–2188, 2009.

[2]J.Bollean, H. Mao and X.Zeng "Twitter mood predict the stock market," Journal of Computer Science , vol.2, no. 1 , pp. 1-8, 2011

[3] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe, “Predicting elections with twitter: What 140 characters reveal about political sentiment.” ICWSM, vol. 10, pp. 178–185, 2010

[4] M. Thelwall, K. Buckley, and G. Paltoglou, “Sentiment in twitter events,” Journal of the American Society for Information Science and Technology, vol. 62, no. 2, pp. 406–418, 2011.

[5] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, “Sentiment analysis of twitter data,” in Proceedings of the Workshop on Languages in Social Media. Association for Computational Linguistics, 2011, pp. 30–38.

[6] B. Liu, “Sentiment analysis and opinion mining,” Synthesis Lectures on Human Language Technologies, vol. 5, no. 1, pp. 1–167, 2012.

[7] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff, “Overview of the trec-2011 microblog track,” in Proceeddings of the 20th Text REtrieval Conference (TREC 2011), 2011.

[8] I. Ounis, C. Macdonald, J. Lin, and I. Soboroff, “Micro-blogging as online word of mouth branding,” in CHI’09 Extended Abstracts on Human Factors in Computing Systems. ACM, 2009, pp. 3859–3864.

[9] E. Kouloumpis, T. Wilson, and J. Moore, “Twitter sentiment analysis: The good the bad and the omg!” in ICWSM, 2011.

[10] www.wikepedia.com

[11]https://www.ravikiranj.net/posts/2012/code/how-build-twitter-sentiment-analyzer/

[12] https://www.slideshare.net/sumit786raj/sentiment-analysis-of-twitter-data

[13] http://www.laurentluce.com/posts/twitter-sentiment-analysis-using-python-and-nltk/

[14] http://www.cs.cornell.edu/home/llee/omsa/omsa.pdf

[15]http://ataspinar.com/2016/02/15/sentiment-analysis-with-the-naive-bayes-classifier/