SENTIMENT ANALYSIS ON MOVIE REVIEW

A thesis

by

SAGOR CHANDROU

MD. RAKIBUL HASAN

MD. MAZHARUL ISLAM

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DHAKA UNIVERSITY OF ENGINEERING &TECHNOLOGY, GAZIPUR

NOVEMBER 2019

SENTIMENT ANALYSIS ON MOVIE REVIEW

A Thesis

by

SAGOR CHANDROU	MD.RAKIBUL HASAN	MD.MAZHARUL ISLAM
Student No.: 144046	Student No.: 144049	Student No.: 144060
Reg. No.: 8544	Reg. No.: 8547	Reg. No.: 8558
Session: 2018-2019	Session: 2018-2019	Session: 2018-2019

Supervisor: Dr. Fazlul Hasan Siddiqui

Professor, CSE, DUET

Submitted to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DHAKAUNIVERSITY OF ENGINEERING & TECHNOLOGY, GAZIPUR

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

of

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

NOVEMBER 2019

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

NOTATIONS

WWW	-World Wide Web
NLP	- Natural Language Processing
ML	-Machine Learning
BOW	- Bag-Of-Words
NLTK	-Natural Language Toolkit
ANN	-Artificial Neural networks
RNN	-Recurrent neural network
LSTM	- Long Short-Term Memory
SA	-Sentiment analysis
IMDB	-Internet Movie Database
SVM	-Support Vector Machine
NB	-Naïve Bayes
ReLU	-Rectifier linear unit
MNB	-Multinomial Naive Bayesian
KNN	-K-nearest neighbor-
PA	-Passive-Aggressive
LM	-Language Modelling
RNTN	-Recursive Neural Tensor Network
CNN	-Convolutional neural network
TF	-Term Frequency
IDF	-Inverse Data Frequency

ACKNOWLEDGEMENT

First of all, we would like to express my whole hearted gratitude to almighty Allah. We would like to express great gratitude to our supervisor Dr. Fazlul Hasan Siddiqui for introducing us to the interesting sentiment analysis and providing him valuable time to express his professional thoughts. We would like tothank him for his helps, continuous encouragement, fertile discussion and valuable suggestionsand comments throughout the thesis work. Without his helps, it couldn't possible for us to complete the thesis. We are also grateful to all of our other friends for their continues motivate and helping us in the thesis.

We are also thankful to Dr. Fazlul Hasan Siddiqui Professor & Head of the department of Computer Science and Engineering, DUET, Gazipur, for giving us support as our Head of the department.

ABSTRACT

Sentiment Analysis is the most prominent branch of natural language processing. It is the computational technique of opinions, sentiments of text on movie review. It deals with the text classification in order to determine the intention of the author of the text.Text data generated from the opinion or review or any comments is increasing rapidly. Large number of people are participating in social media to give their reviews or opinion about various topics. The intention can be of positive or negative type. This paper presents a comparison of results obtained by applying Naive Bayes (NB) , Support Vector Machine (SVM) and RNN-LSTM. These algorithms are used to classify a sentimental review having either a positive review or negative review. The dataset considered for training and testing of model in this work is labeled based on polarity movie dataset and a comparison with results available in existing literature has been made for critical examination.

CHAPTER-1

INTRODUCTION

World Wide Web has become the most popular communication platforms to the public reviews, opinions, comments and sentiments. These sentiments refers to opinions about products, places, books or research papers become daily text reviews. The number of active user bases and the size of their reviews created daily on online websites are massive. There are 2.4 billion active online users, who write and read online around the world [23]. Although the scientific domain is huge as a big world of journals and conferences, there are more than 4000 rated conferences and 5000 ranked journals [24].Notably, a large fragment of WWW researchers make their content public, allowing researchers, societies, universities, and corporations to use and analyze data.In this thesis, we try to achieve trusted movie reviews evaluation to be useful for researchers and facilitate the selection of papers that match their research direction.

1.1 What is Sentiment Analysis?

Sentiment analysis is called opinion mining which is a computational study of reviews, sentiments, opinions, evaluations, attitudes, subjective, views, emotions, etc., expressed in the text. In the following sections, we will discuss sentiment analysis area and the factors affecting them. Sentiments can be recognized as emotions, or as judgments, opinions or ideas prompted or colored by emotions or susceptibility or feelings [25]. There are two types of textual information: facts and opinions information.

Fig. 1.1: Sentiment analysis

While the facts are objective expressions about objects, features, entities, events and their characteristics, opinions are ordinarily subjective expressions that identify people’s sentiments, views or feelings toward objects, entities, events and their characteristics [26].It plans to realize the attitude or opinion of a writer with respect to a certain topic or goal. The attitude could reflect his/her opinion and evaluation, his/her effective situation (what are the feelings of the writer at the time of recording the opinion) or the purpose emotional communication (what is the effect which is situated on the reader when reading the opinion of the writer).

1.2 How Does Sentiment Analysis Work?

Sentiment analysis is the way to identify the tone and emotions expressed through written or spoken online communication. Also known as opinion mining or emotion AI, sentiment analysis performs data mining, processes the results, extracts public opinions out of the text and lets you come to the valuable conclusions. Sentiment analysis mainly follow keywords which represents positive and negative polarity. There are mainly three approaches in sentiment analysis.

i. Machine learning based approach-Needs to develop classification model which is trained using relabeled dataset of positive, negative, neutral content.

ii. Lexicon based - considers lexicon dictionary for identifying polarity of the text.

iii. Combined approach - Which uses lexicon dictionary along with pre-labelled data set for developing classification model.

1.3 Why Sentiment Analysis is Important?

There are millions online users, who write and read online and Internet usage around the world. Online daily sentiments becomes the most significant issue in making a decision. According to a new survey conducted by Dimensional Research, the survey discuss the percentage of trust online customer reviews as much as personal recommendations. According to 2011 Study 74% of customer’s confidence is based on online personal recommendation reviews, 60% in 2012 study, and 57% in 2013 Study. But this percentage increases with respect to 2014 Study: 94% of customer’s trust on online sentiment reviews [23].

1.4 Sentiment Analysis Models

Earlier, the sentiment text analysis or more exactly positive/negative classification relies on using a dataset and a classifier. The documents apply the classifier into two sets: positive and negative. The increased documents will be exhibited informatively in test phase, the more accurate the result of classification will be Since, finding the best document representatives that can describe it (document) better is of a vital importance in sentiment analysis.

1.5 Sentiment Analysis Problems

In this section, we will discuss the problems faced sentiment analysis shortly [9]. The research point of an abstraction of the problems enables us to see a rich set of inter-related sub-problems which make up the sentiment analysis problem. It is often said that if we cannot structure a problem, we probably do not understand the problem. The objective of the definitions is thus to abstract a structure from complex and intimidating unstructured natural language text. From a practical application point of view, the definitions let practitioners see what sub-problems need to be solved in a practical system, how they are related, and what output should be produced.

CHAPTER-2

RELATED WORK

2.1 Sentiment Analysis on Online Product Review

Xing Fang and Justin Zhan [1] describes that the subjective contents are extracted, it consist of sentiment sentences which contain at least one positive or negative word. These sentences are tokenized into separated English words. Depending on parts of speech in the words, corresponding tags are used. The sentiment tokens and scores are information extracted from the original dataset. These are known as features. In order to classify them these features are to be transformed to vectors called feature vector. Huge amount of content produced by amateur authors on various topics are considered in [2]. Sentiment analysis (SA) aggregates users' sentiments. Machine learning (ML) techniques for natural language. In lexicon-based techniques prevent over fitting. Corpus-based statistical techniques for stabilization. This paper highlights natural language processing (NLP) specific open challenges. Two typical approaches to sentiment analysis – lexicon look up and machine learning are used by Ji Fang and Bi Chen in [3]. Lexicon look up starts with a lexicon of positive and negative words. Current sentiment lexicons do not capture such domain and context sensitivities of sentiment expressions. The proposed system present an alternative method that incorporates sentiment lexicons as prior knowledge with machine learning approaches such as SVM to improve the accuracy of sentiment analysis. Unsupervised learning algorithm for classifying reviews as thumbs up or thumbs down by the average semantic orientation is carried out by Peter D. Turney as referenced in [4]. The semantic orientation of a phrase is calculated as the mutual information between the given phrase and the word "excellent" minus the mutual information between the given phrase and the word "poor".

2.2 Sentiment Analysis Using SVMA Systematic Literature Review

Development and refining the automated techniques of sentiment extraction and analysis is one of the hot research topics today. Many researchers have worked on sentiment analysis techniques via different approaches (Lexical, Machine Learning and Hybrid) however, in-depth analysis and review of latest literature on sentiment analysis with SVM was still required. Some of the related studies on sentiment analysis are as follows. Authors in [5] conducted a systematic literature review regarding opinion mining from the reviews of mobile app store users. The researchers focused on the importance of mobile applications in now days and further highlighted the increasing demand of user reviews about those apps. Obviously these reviews are crucial for the new users, who are going to buy these apps and also for those who develop or sell these apps. The authors highlighted the proposed solutions of mining problems, and also identified the remaining unsolved issues and new challenges. In [6], a systematic literature review is conducted to analyze the current state of Arabic text mining. The researchers in [7] conducted a literature review on sentiment analysis and opinion mining of social issues. The selected papers have taken the data from social web sites. According to authors, different types of classification techniques, if combined, can provide the better results. In [8], a literature survey is conducted about opinion and spam mining. The authors have performed sentiment classification of Arabic tweets by using Naïve Bayes, Decision Tree and Support Vector Machine. In this study, a framework for Arabic tweets classification is followed which consisted of several subtasks such as: Term Frequency Inverse Document Frequency (TFIDF) and Arabic stemming etc. Moreover three information retrieval metrics are used for performance evaluation: precision, recall, and f-measure. In [11], a literature review is conducted covering the domain of data mining applications in customer relationship management. In [12], the authors have predicted the rainfall in Malaysia by using machine-learning techniques. They have used following classification algorithms: Naïve Bayes, Decision Tree, Support Vector Machine, Neural Network and Random Forest. A comparative analysis was performed to identify the particular technique which can bring good results with little amount of training data. The comparative analysis showed that Decision Tree and Random Forest both can get well trained by using lower amount of training data and can get high F-measure score. However, Support Vector Machine and Naive Bayes both showed lower F-measure score, when trained with lower amount of training data.

Neural Network required large amount of training data to predict very little amount of test data. In [14], the authors have analyzed the performance of Support Vector Machine for polarity detection of textual data. A sentiment analysis framework is proposed and performance of SVM was evaluated on three datasets. Two datasets were taken from twitter and one from IMDB review. Performance of SVM was compared for each dataset by keeping in view three different ratios of training data and test data: 70:30, 50:50 and 30:70. Precision, recall and f-measure scores were used for performance evaluation. In [15], student's academic performance was predicted by using three data mining techniques: Decision tree (C4.5), Multilayer Perceptron and Naïve Bayes. These techniques were applied on student's data, which was collected from 2 undergraduate courses in two semesters. According to results, Naïve Bayes showed overall accuracy of 86% and outperformed MLP and Decision tree.

2.3 Sentiment Analysis for Amazon Reviews

So far, there are a lot of research papers related to product reviews, sentiment analysis or opinion mining. For example, Xu Yun[16] el al from Stanford University applied existing supervised learning algorithms such as perceptron algorithm, naive bayes and supporting vector machine to predict a review's rating on Yelp's rating dataset. They used hold out cross validation using 70% data as the training data and 30% data as the testing data. The author used different classifiers to determine the precision and recall values. In paper[17], Maria Soledad Elli and Yi-Fan extracted sentiment from the reviews and analyze the result to build up a business model. They claimed that this tool gave them pretty high accuracy. They mainly used Multinomial Naive Bayesian(MNB) and support vector machine as the main classifiers. Callen Rain[18] proposed extending the current work in the field of natural language processing. Naive Bayesian and decision list classifiers were used to classify a given review as positive or negative. Deep-learning neural networks is also popular in the area of sentiment analysis. Ronan Collobert[19] et al used a convolutional network for the semantic role labeling task with the goal avoiding excessive task-specific feature engineering. On the other hand, in paper[20], the authors proposed proposed using recursive neural networks to achieve a better understanding compositionality in tasks such as sentiment detection. In this paper, we want to apply both traditional algorithms including Naive Bayesian, K-nearest neighbor, Supporting Vector Machine and deep-learning tricks. By comparing the accuracy of these models, we would like to get a better understanding how these algorithms work in tasks such as sentiment analysis.

2.4 Sentiment Analysis for Movies Reviews Dataset Using Deep Learning Models

In a published sentiment analysis work the authors have classified online product reviews into positive and negative classes. This paper has applied different machine learning algorithms in order to experimentally evaluate various trade-offs, using approximately 100K product reviews from the web. Three classifiers have been applied (Passive-Aggressive (PA) Algorithm Based Classifier, Language Modelling (LM) Based Classifier, and Winnow Classifier). In addition, ngrams was used for linguistic features extraction[21]. The results have illustrated that when a high order n-gram as features is used with a significant classifier, a comparable result can be achieved, or higher performance than that reported in academic papers can be achieved. A second work have reported that SVM performed better classification with 82.9% accuracy compared with the Naive Bayes that achieved 81% on positive and negative movie reviews from the IMDB (imdb.com) movie reviews dataset that consists of 752 negative and 1301 positive reviews [22]. The researchers of a third paper have proposed Recursive Neural Tensor Network (RNTN) model for identifying sentences as positive or negative by using fully labelled parse trees. They used a dataset based on a corpus of 11,855 movie reviews. The RNTN have obtained 80.7% accuracy on fine-grained sentiment prediction across all phrases and captures negation of different sentiments and scope more accurately than previous models[23]. A fourth paper has focused on customer reviews. Products reviews were collected from Amazon.com (11,754 sentences). It proposed a novel deep learning framework named Weaklysupervised Deep Embedding for reviewing sentence sentiment classification with accuracy 87% [24]. Lastly, a work that was published in 2018 performed several deep learning models for a binary sentiment classification problem. They used movie reviews in Turkish from the website www. beyazperde.com to train and test the deep learning models[25]. The whole dataset consists of 44,617 samples including positive and negative reviews. 4000 samples from the dataset were used for testing the models. Two major deep learning architectures were applied, CNN and LSTM. Word embedding were created by applying the word2vec algorithm with a skip-gram model on the used dataset.

Experimental results have shown that the use of word embedding with deep neural networks effectively yields performance improvements in terms of run time and accuracy.

CHAPTER-3

THEORETICAL FOUNDATION

3.1 What is a Bag-of-Words?

The bag-of-words model is a way of representing text data when modeling text with machine learning algorithms. It is a simplifying representation used in natural language processing and information retrieval (IR). In this model, a text is represented as the bag (multi set) of its words, disregarding grammar and even word order but keeping multiplicity. A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things. Those area vocabulary of known words and a measure of the presence of known words.

A very common feature extraction procedures for sentences and documents is thebag-of-words approach (BOW). In this approach, we look at the histogram of thewords within the text, i.e. considering each word count as a feature.The intuition is that documents are similar if they have similar content. Further, that from the content alone we can learn something about the meaning of the document. The bag-of-words can be as simple or complex as you like. The complexity comes both in deciding how to design the vocabulary of known words (or tokens) and how to score the presence of known words. We will take a closer look at both of these concerns.

3.1.1 Example of the Bag-of-Words Model

To understand the bag of words approach, let’s first start with the help of an example.

Step 1: Collect Data

The following models a text document using bag-of-words. Here are two simple text documents.Suppose we have a corpus with three sentences. Now if we have to perform text classification, or any other task, on the above data using statistical techniques, we can not do so since statistical techniques work only with numbers. Therefore we need to convert these sentences into numbers.

Sentence_1	Movie is good
Sentence_2	Movie is not good
Sentence_1	Movie is not bad

Table 3.1: Sample of text

Step 2: Design the Vocabulary

Now we can make a list of all of the words in our model vocabulary. The unique words here(Ignoring case and punctuation).

Movie
is
good
not
bad

Table 3.2: List of unique words

Step 2: Create a Dictionary of Word Frequency

The next step is to create a dictionary that contains all the words in our corpus as keys and the frequency of the occurrence of the words as values. In other words, we need to create a histogram of the words in our corpus. Look at the following table:

Word	**Frequency**
Movie	3
Is	3
Good	2
Not	2
Bad	1

Table 3.3: Frequency words

In the table above, you can see each word in our corpus along with its frequency of occurrence. For instance, you can see that since the word movie occurs three times in the corpus (once in each sentence) its frequency is 3.

In our corpus, we only had three sentences, therefore it is easy for us to create a dictionary that contains all the words. In the real world scenarios, there will be millions of words in the dictionary. Some of the words will have a very small frequency. The words with very small frequency are not very useful, hence such words are removed. One way to remove the words with less frequency is to sort the word frequency dictionary in the decreasing order of the frequency and then filter the words having a frequency higher than a certain threshold.

Step 3: Creating the Bag of Words Model

To create the bag of words model, we need to create a matrix where the columns correspond to the most frequent words in our dictionary where rows correspond to the document or sentences. Suppose we filter the 4 most occurring words from our dictionary. Then the document frequency matrix will look likes Table 4.4.

	Movie	Is	Good	Not
Sentence_1	1	1	1	0
Sentence_2	1	1	1	1
Sentence_3	1	1	0	1

Table 3.4: Bag of Words Model

In the above matrix, the first row corresponds to the first sentence. In the first, the word "movie" occurs once, therefore we added 1 in the first column. The word in the forth column is "not", it doesn't occur in the first sentence, therefore we added a 0 in the second column for sentence 1. Similarly, in the second sentence, both the words "movie" and "not" occur once, therefore we added 1 in the first two columns. In this way, all the cells in the above matrix are filled with either 0 or 1, depending upon the occurrence of the word. Final matrix corresponds to the bag of words model. In each row, you can see the numeric representation of the corresponding sentence. For instance, the first row shows the numeric representation of Sentence 1. This numeric representation can now be used as input to the statistical models.

3.2 What is TF-IDF?

TF-IDF stands for “Term Frequency - Inverse Data Frequency”. First, we will learn what this term means mathematically.

Term Frequency (tf) gives us the frequency of the word in each document in the corpus. It is the ratio of number of times the word appears in a document compared to the total number of words in that document. It increases as the number of occurrences of that word within the document increases. Each document has its own tf.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

Inverse Data Frequency (idf) used to calculate the weight of rare words across all documents in the corpus. The words that occur rarely in the corpus have a high IDF score. It is given by the equation below.

$$idf(w) = log(\frac{N}{df_t})$$

Combining these two we come up with the TF-IDF score (w) for a word in a document in the corpus. It is the product of tf and idf:

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

Here,

tfi,j = number of occurrences of i in j
dfi = number of documents containing i
N = total number of documents

3.2.1 Example of the TF-IDF

Let's take an example to get a clearer understanding.

Sentence_1	Movie is good
Sentence_2	Movie is not bad

Table 3.5 : Sample of text

In this example, each sentence is a separate document. We will now calculate the TF-IDF for the above two documents, which represent our corpus.

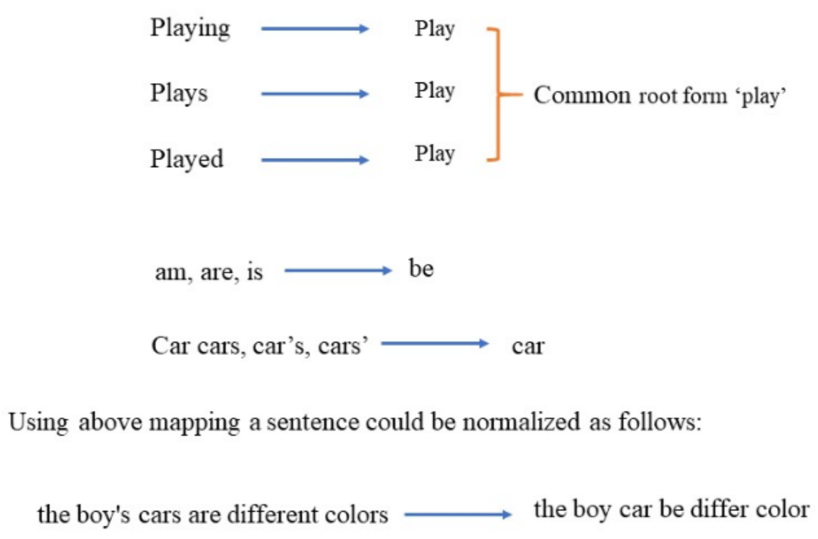
Word	TF		IDF	TF*IDF	
	Sentence_1	Sentence_2		Sentence_1	Sentence_2
Movie	1/3	1/4	Log(2/2) = 0	0	0
is	1/3	1/4	Log(2/2) = 0	0	0
Good	1/3	0	Log(2/1) = 0.3	0.09	0
Not	0	1/4	Log(2/1) = 0.3	0	0.9
Bad	0	1/4	Log(2/1) = 0.3	0	0.9

Table 3.6: TF-IDF Calculation

From the above table, we can see that TF-IDF of common words was zero, which shows they are not significant. On the other hand, the TF-IDF of "Good", "Not, and "Bad", are non-zero. These words have more significance.

3.3 Lemmatization Approaches with Examples

Lemmatization is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item. Lemmatization is similar to stemming but it brings context to the words. So it links words with similar meaning to one word. Text preprocessing includes both Stemming as well as Lemmatization. Many times people find these two terms confusing. Some treat these two as same. Actually, lemmatization is preferred over Stemming because lemmatization does morphological analysis of the words.Let's look at a few examples


Playing
Play
Plays
Play
Played
Play
Common root form 'play'
am, are, is
be
Car cars, car's, cars'
car
Using above mapping a sentence could be normalized as follows:
the boy's cars are different colors
the boy car be differ color

Above examples must have helped us understand the concept of normalization of text, although normalization of text is not restricted to only written document but to speech as well.

3.4 Confusion Matrix

A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. We wanted to create a "quick reference guide" for confusion matrix terminology because couldn't find an existing resource that suited our requirements compact in presentation, using numbers instead of arbitrary variables and explained both in terms of formulas and sentences.

Let's start with an example confusion matrix for a binary classifier (though it can easily be extended to the case of more than two classes)

N=165	Predict: No	Predict: Yes
Actual No	50	10
Actual Yes	5	100

Table 3.7: confusion matrix for a binary classifier

What can we learn from this matrix?

- There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a disease, for example, "yes" would mean they have the disease, and "no" would mean they don't have the disease.
- The classifier made a total of 165 predictions (e.g., 165 patients were being tested for the presence of that disease).
- Out of those 165 cases, the classifier predicted "yes" 110 times, and "no" 55 times.
- In reality, 105 patients in the sample have the disease, and 60 patients do not.

Let's now define the most basic terms, which are whole numbers (not rates):

- true positives (TP): These are cases in which we predicted yes (they have the disease), and they do have the disease.
- true negatives (TN): We predicted no, and they don't have the disease.
- false positives (FP): We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- false negatives (FN): We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

W've added these terms to the confusion matrix, and also added the row and column totals:

N=165	Predict: No	Predict: Yes	
Actual No	TN = 50	FP = 10	60
Actual Yes	FN = 5	TP = 100	105
	55	110	

Table 3.8: Confusion Matrix For A Binary Classifier Total

This is a list of rates that are often computed from a confusion matrix for a binary classifier:

- Accuracy: Overall, how often is the classifier correct?
    - (TP+TN)/total = (100+50)/165 = 0.91
- Misclassification Rate: Overall, how often is it wrong?
    - (FP+FN)/total = (10+5)/165 = 0.09
    - equivalent to 1 minus Accuracy
    - also known as "Error Rate"
- True Positive Rate: When it's actually yes, how often does it predict yes?
    - TP/actual yes = 100/105 = 0.95
    - also known as "Sensitivity" or "Recall"
- False Positive Rate: When it's actually no, how often does it predict yes?
    - FP/actual no = 10/60 = 0.17
- True Negative Rate: When it's actually no, how often does it predict no?
    - TN/actual no = 50/60 = 0.83
    - equivalent to 1 minus False Positive Rate
    - also known as "Specificity"
- Precision: When it predicts yes, how often is it correct?
    - TP/predicted yes = 100/110 = 0.91

CHAPTER-4

CLASSIFICATION

This chapter will give a detailed description of the classification methods used in this project. Those are Support Vector Machines, Naïve Bayes and RNN(using LSTM). Some other approaches and concepts are also presented, to make the presentation of those algorithms easier.

4.1 Support Vector Machines

Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyper plane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples. In two dimensional space this hyper plane is a line dividing a plane in two parts where in each class lay in either side. Suppose we are given plot of two label classes on graph as shown in image. We are separating line for the classes.

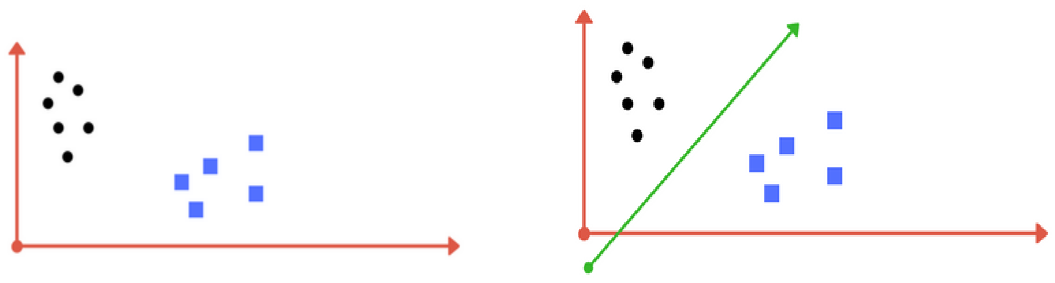


Fig. 4.1: Sample cut to divide into two classes.

It fairly separates the two classes. Any point that is left of line falls into black circle class and on right falls into blue square class. Separation of classes. That’s what SVM does. It finds out a line/ hyper-plane (in multidimensional space that separate outs classes).

Now consider what if we had data as shown in image below?.

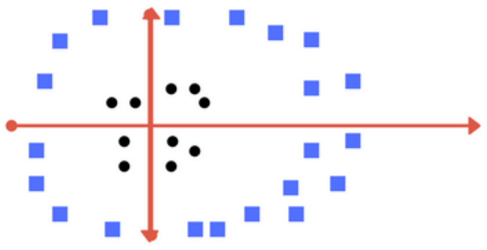


Fig. 4.2: Can you draw a separating line in this plane?

Clearly, there is no line that can separate the two classes in this x-y plane. So what do we do? We apply transformation and add one more dimension as we call it z-axis. Let's assume value of points on z plane, w = $x^2 + y^2$. In this case we can manipulate it as distance of point from z-origin. Now if we plot in z-axis, a clear separation is visible and a line can be drawn.

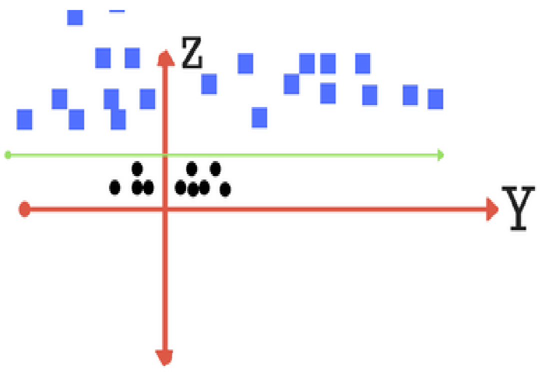


Fig. 4.3: Plot of z y axis. A separation can be made here.

When we transform back this line to original plane, it maps to circular boundary as shown in image E. These transformations are called kernels.

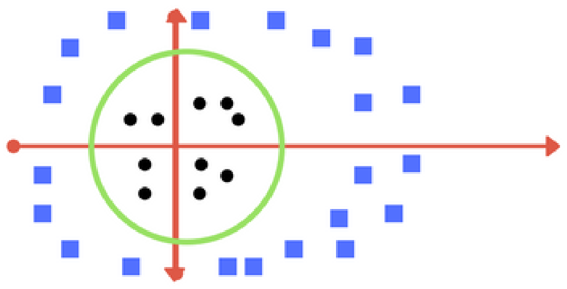
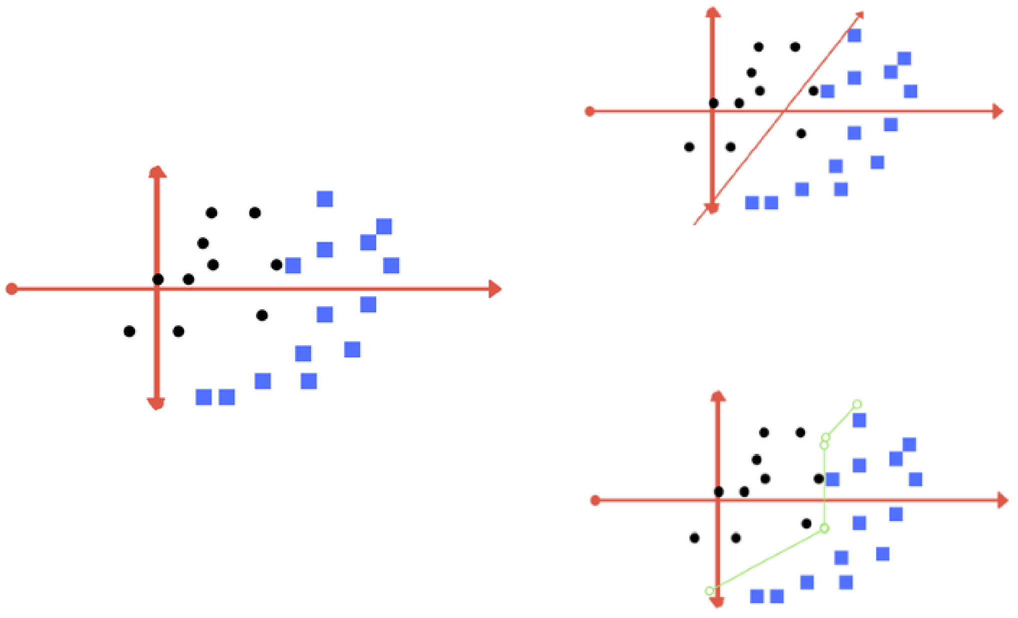


Fig. 4.4: Transforming back to x-y plane, a line transforms to circle.

What if data plot overlaps? Or, what in case some of the black points are inside the blue ones?



(b)

Fig. 4.5: Which line among fig (a)? or fig(b) ? should we draw?

Which one do we think? Well, both the answers are correct. The first one tolerates some outlier points. The second one is trying to achieve 0 tolerance with perfect partitioning real world application, finding perfect class for millions of training data set takes lot of time. This is called regularization parameter.We define two terms regularization parameter and gamma. These are tuning parameters in SVM classifier. Varying those we can achieve considerable nonlinear classification line with more accuracy in reasonable amount of time. We shall see how we can increase the accuracy of SVM by tuning these parameters. One more parameter is kernel. It defines whether we want a linear of linear separation.

4.1.1 Tuning Parameters

4.1.1.1 Kernel

The learning of the hyper plane in linear SVM is done by transforming the problem using some linear algebra. This is where the kernel plays role. For linear kernel the equation for prediction for a new input using the dot product between the input (x) and each support vector (xi) is calculated as follows

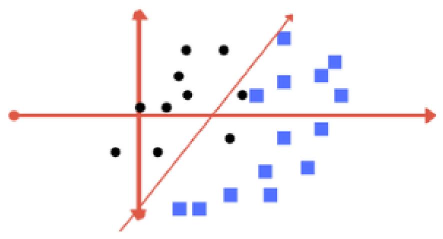
$$f(x) = B(0) + sum(ai * (x,xi))$$

This is an equation hat involves calculating the inner products of a new input vector (x) with all support vectors in training data. The coefficients B(0)and ai (for each input) must be estimated from the training data by the learning algorithm.

4.1.1.2 Regularization

The Regularization parameter (often termed as C parameter in python's sklearn library) tells the SVM optimization how much we want to avoid misclassifying each training example. For large values of C, the optimization will choose a smaller-margin hyper plane if that hyper plane does a better job of getting all the training points classified correctly. Conversely, a very small value of C will cause the optimizer to look for a larger-margin separating hyper plane, even if that hyper plane misclassifies more points.

The images below (same as image 1 and image 2 in section 2) are example of two different regularization parameter. Left one has some misclassification due to lower regularization value. Higher value leads to results like right one.



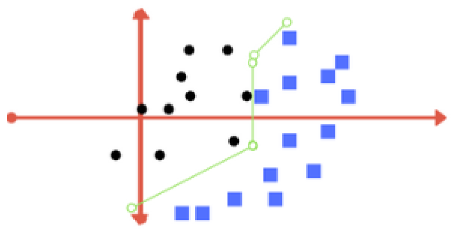
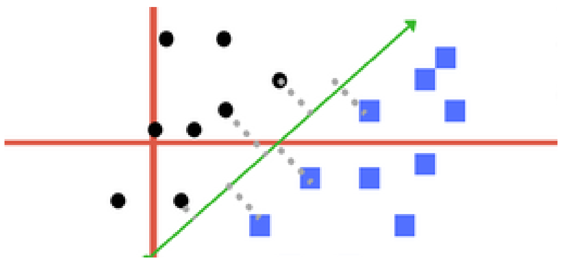


Fig. 4.6: Left: low regularization value *Fig. 4.7: high regularization value*

4.1.1.3 Gamma

The gamma parameter defines how far the influence of a single training example reaches, with low values meaning ‘far’ and high values meaning ‘close’. In other words, with low gamma, points far away from plausible separation line are considered in calculation for the separation line. Whereas high gamma means the points close to plausible line are considered in calculation.



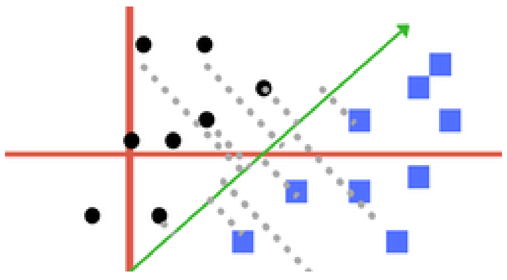


Fig. 4.8: High gamma (only nearby points are considered) and low gamma (far way points are considered

4.1.1.4 Margin

Finally last but very important characteristic of SVM classifier. SVM to core tries to achieve a good margin. A margin is a separation of line to the closest class points. A good margin is one where this separation is larger for both the classes. Images below gives to visual example of good and bad margin. A good margin allows the points to be in their respective classes without crossing to other class.

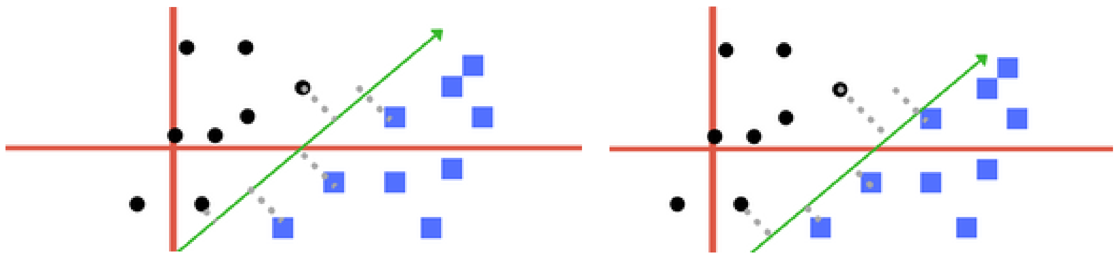


Fig. 4.9: Good margin and Bad margin

4.2 Naive Bayes Classifier

4.2.1 What is Naive Bayes algorithm?

In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods. Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below

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

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

Where,

P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).

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.

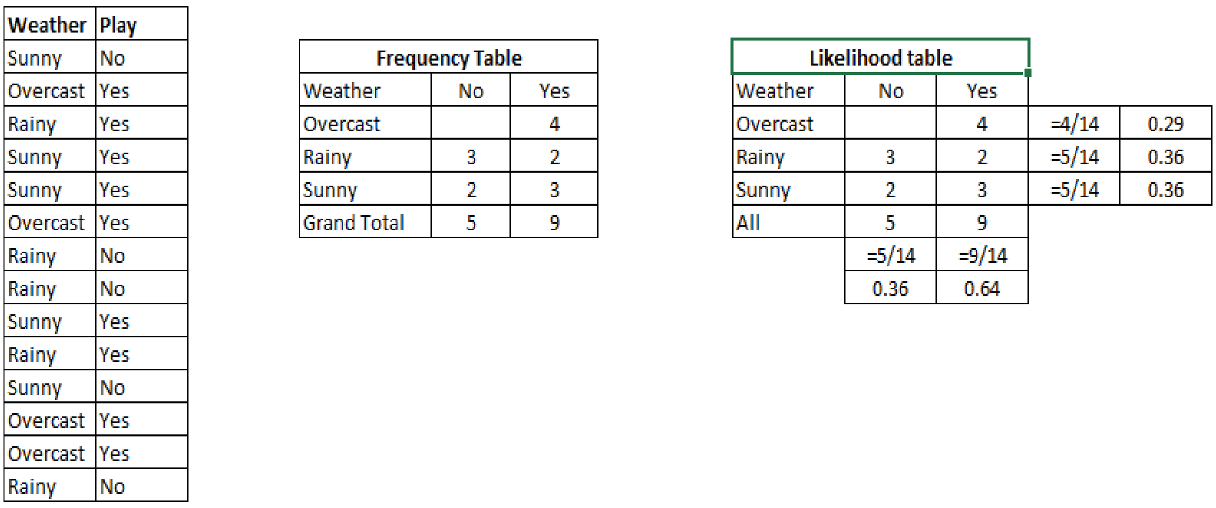
4.2.2 How Naive Bayes algorithm works?

Let’s understand it using an example. Below I have a training data set of weather and corresponding target variable ‘Play’ (suggesting possibilities of playing). Now, we need to classify whether players will play or not based on weather condition. Let’s follow the below steps to perform it.

Step 1: Convert the data set into a frequency table.

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.



Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

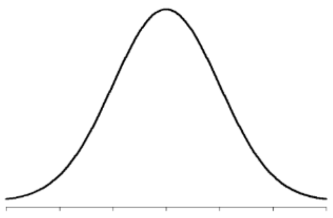
Fig. 4.10: Naive Bayes algorithm works

4.2.3 Types of Naive Bayes Classifier

Multinomial Naive Bayes This is mostly used for document classification problem, i.e whether a document belongs to the category of sports, politics, technology etc. The features/predictors used by the classifier are the frequency of the words present in the document.

Bernoulli Naive Bayes This is similar to the multinomial naive bayes but the predictors are boolean variables. The parameters that we use to predict the class variable take up only values yes or no, for example if a word occurs in the text or not.

Gaussian Naive Bayes When the predictors take up a continuous value and are not discrete, we assume that these values are sampled from a Gaussian distribution.



$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Fig. 4.11: Gaussian distribution.

4.3 Understanding RNN and LSTM

4.3.1 What is Neural Network?

Neural Networks are set of algorithms which closely resemble the human brain and are designed to recognize patterns. They interpret sensory data through a machine perception, labelling or clustering raw input. They can recognize numerical patterns, contained in vectors, into which all real-world data (images, sound, text or time series), must be translated. Artificial neural networks are composed of a large number of highly interconnected processing elements (neuron) working together to solve a problem. An ANN usually involves a large number of processors operating in parallel and arranged in tiers. The first tier receives the raw input information - analogous to optic nerves in human visual processing. Each successive tier receives the output from the tier preceding it, rather than from the raw input - in the same way neurons further from the optic nerve receive signals from those closer to it. The last tier produces the output of the system.

4.3.2 What is Recurrent Neural Network (RNN)?

Recurrent Neural Network is a generalization of feed forward neural network that has an internal memory. RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input.

Unlike feed forward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unregimented, connected handwriting recognition or speech recognition. In other neural networks, all the inputs are independent of each other. But in RNN, all the inputs are related to each other.

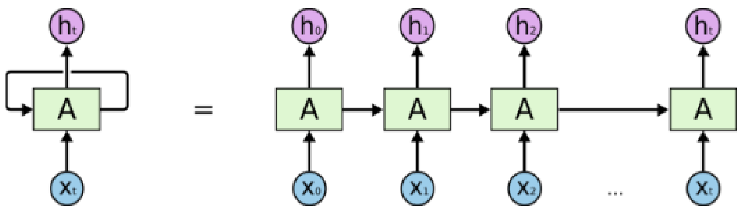


Fig. 4.12: An unrolled recurrent neural network

First, it takes the X(0) from the sequence of input and then it outputs h(0) which together with X(1) is the input for the next step. So, the h(0) and X(1) is the input for the next step. Similarly, h(1) from the next is the input with X(2) for the next step and so on. This way, it keeps remembering the context while training.

The formula for the current state is-

$$h_t = f(h_{t-1}, x_t)$$

Applying Activation Function

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

W is weight, h is the single hidden vector, Whh is the weight at previous hidden state, Whx is the weight at current input state, tanh is the Activation funtion, that implements a Non-linearity that squashes the activations to the range[-1.1].

Output:

$$y_t = W_{hy}h_t$$

Where, Yt is the output state

4.3.3 Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using

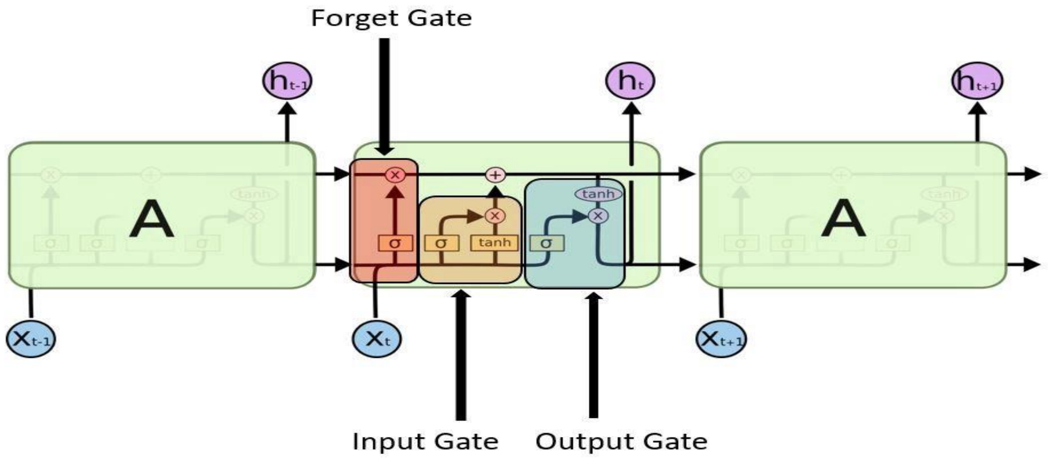


Fig. 4.13: LSTM gates

Input gate - discover which value from input should be used to modify the memory. Sigmoid function decides which values to let through 0,1. and tanh function gives weightage to the values which are passed deciding their level of importance ranging from-1 to 1.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Forget gate- discover what details to be discarded from the block. It is decided by the sigmoid function. it looks at the previous state(ht-1) and the content input(Xt) and outputs a number between 0(omit this)and 1(keep this)for each number in the cell state Ct−1.

$$o_t = \sigma\left(W_o\ \left[h_{t-1}, x_t\right]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

CHAPTER-5

SYSTEM MODEL

5.1. Data Understanding

The following section provides information about the data that is used in my research. The large movie review dataset from IMDB. IMDB dataset having 50K movie reviews for natural language processing or Text analytics. This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. This is a dataset for binary sentiment classification containing substantially more data than previous benchmark datasets. The dataset contains an even number of positive and negative reviews. The dataset is divided into training and test sets. The training set is the same 25,000 labeled reviews.

	review	sentiment
1	One of the other reviewers has mentioned that …	positive
2	A wonderful little production. <br /><br />The …	positive
3	I thought this was a wonderful way to spend …	positive
4	Basically there's a family where a little boy …	negative
--	------------------------------	---
49999	West being tamed and kicked aside by the stead …	negative
50000	No one expects the Star Trek movies to be high ar…	negative

Table 5.1: This table provides details about datasets that were used for thesis. Collected from link: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

5.2. Data Preparation

Preparation of texts is crucial for the success of their analysis. The thesis theoretical part discussed various text preprocessing techniques. Text data preparation is different for each problem. Preparation starts with simple steps, like loading data, but quickly gets difficult with cleaning tasks that are very specific to the data we are working with. We needed help as to where to begin and what order to work through the steps from raw data to data ready for modeling.

i. Load Text Data
ii. Clean Text Data
iii. Create Feature
iv. Create Train Dataset and Test Dataset

5.2.1 Word Preprocessing

The step in implementation is the transformation of texts into tokens. The theory of tokenization was explained in theoretical part. The Tokenizer splits a whole text based on the occurrence of space symbols. The second phase is lemmatization or stemming. Both approaches are tested to check which one works better. It makes sense to apply lemmatization before stop word removal, due to the fact that lemmas are contained in stop word lists. In contrast, stemming should be applied afterwards, unless a stemmed stop word list is available. Lemmatization is done by Lemmatizer and NLTK provides the Snowball-Stemmer. The next step is stop word removal, which is applied to reduce the overall vocabulary in the corpus. I will test whether my approach performs better with or without stop word removal, because stop words could be essential for receiving correct information.

5.2.2 Document and Word Representations

Transforming words and documents to machine readable representations is the last part in preprocessing pipeline, as they work as input for the modeling phase. Two different approaches are tested separately: Vector Space Model with tf-idf and Bag-of-word. They represent words and documents in different forms. This subsection describes how the word representations are constructed and in what kind of format we will use them.

5.2.3 Train and Test Split

This subsection will shortly illustrate the train- and testsplit for our implementation. indicates that 50% of the data are used to test our models, while 50% are used to train them. We pick the test and train instances at random. With this split approximately 50,000 instances can be used for training a model for the newer cars dataset, which is enough to train complex ANN-models.

5.3 Modeling

The last section in my approach focuses on details of the predictive models. Overall detailed described into classification chapter.We usedsome Artificial Neural Networks to predict sentiment.

i. Support Vector Machine(SVM)

ii. Naïve Bayes(NB)

iii. Recurrent Neural Network(RNN) with Long-Short-Term-Memory(LSTM)

CHAPTER-6

PERFORMANCE EVALUATION

6.1 Evaluation and Discussion

The fifth chapter of this thesis is devoted to the evaluation of my approach. Chapter 3 and 4 provided details about the applied part of this thesis and includes preliminary work to answer the second question: How do such approaches perform on real description text data for sentiment analysis on movie review. The following chapter will outline the performance of our proposed approaches on the way to comparison with previous thesis/research work. This is done to investigate whether the prediction of performance works better when using the various model with a new approach of working technique. We will provide the results of Support vector machine, the Naïve bayes and the RNN-LSTM. wehave compared the models in terms of their test accuracy.

SL	Classifier and ANN	Accuracy(%)	Accuracy(%) by the reference
01	SVM	88.81	87.45 (reference [28])
02	NB	86.09	81.83 (reference [28])
03	RNN-LSTM	82.30	

Table 6.1: Result

CHAPTER-7

CONCLUTION AND FUTURE WORK

Sentiment Analysis is considered as one of the hot topics in the domain of knowledge discovery. Large amount of online data is being added on daily basis ranging from social media posts and comments to movie and product reviews. By using sentiment analysis techniques, these data sources can be used to fetch the useful information such as: prediction of election results, getting user's feedback about any product, analyzing the market reputation of particular brand and obtaining public opinion before launching a new product etc. Multiple approaches are available for sentiment analysis such**.** as lexicon based, machine learning based and the hybrid of both. This paper presents a comparison of results obtained by applying Naive Bayes (NB) and Support Vector Machine (SVM) classification algorithm and RNN-LSTM. Now days, along with conventional machine learning classification techniques, many customized and integrated models have been proposed for sentiment analysis and polarity detection. This study has provided a compact and comprehensive review of latest research by focusing on NB, SVM and RNN-LSTM technique of sentiment analysis. This study has followed a systematic framework for review and provided the answers of identified thesis questions after critical review of selected papers. For future work it is suggested to perform a comparative analysis of the customized techniques with same dataset.

REFERENCES

1. Xing Fang* and Justin Zhan “Sentiment analysis using product review data” Fang and Zhan journal of Big Data (2015) 2:5
2. Muhammad Taimoor Khan, Mehr Durrani2, Armughan Ali, Irum Inayat, Shehzad Khalid and Kamran Habib Khan “Sentiment analysis and the complex natural language” Khan et al. Complex Adapt Syst Model (2016) 4:2
3. [3] Ji fang, Bi Chen, “Incorporating Lexicon Knowledge into SVM Learning to Improve Sentiment Classification”, Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology (SAAIP), IJCNLP 2011, pp. 94–100.
4. Turney, Peter D., “Thumbs up or thumbs down? Semantic Orientation Applied to Unsupervised Classification of Reviews”, Proceedings of Association for Computational Linguistics, Philadelphia, PA. July 2002, pp. 417-424.
5. N. Genc-Nayebi and A. Abran, “A systematic literature review: Opinion mining studies from mobile app store user reviews,” J. Syst. Softw., vol. 125, no. November, pp. 207–219, 2017.
6. H. Al-Mahmoud and M. Al-Razgan, “Arabic Text Mining a Systematic Review of the Published Literature 2002-2014,” 2015 Int. Conf. Cloud Comput., no. November, pp. 1–7, 2015.
7. V. Singh and S. K. Dubey, “Opinion Mining and Analysis: A Literature Review,” 2014 5Th Int. Conf. Conflu. Next Gener. Inf. Technol. Summit, pp. 232–239, 2014.
8. A. A. Sheibani, “Opinion mining and opinion spam: A literature review focusing on product reviews,” 2012 6th Int. Symp. Telecommun. IST 2012, pp. 1109–1113, 2012. Arabic Twitter Sentiment Analysis,” vol. 6, no. 6, pp. 1067–1073, 2016.
9. E. Ngai, L. Xiu, and D. Chau, “Application of data mining techniques in customer relationship management: A literature review and classification,” Expert Syst. Appl., vol. 36, no. 2, pp. 2592–2602, 2009.
10. S. Zainudin, D. S. Jasim, and A. A. Bakar, “Comparative Analysis of Data Mining Techniques for Malaysian Rainfall Prediction,” Int. J. Adv. Sci. Eng. Inf. Technol., vol. 6, no. 6, pp. 1148–1153, 2016.
11. M. Ahmad and S. Aftab, “Analyzing the Performance of SVM for Polarity Detection with Different Datasets,” Int. J. Mod. Educ. Comput. Sci., vol. 9, no. 10, pp. 29–36, 2017.
12. A. Mueen, “Modeling and Predicting Students ’ Academic Performance Using Data Mining Techniques,” no. November, pp. 36–42, 2016.

13. Y. Xu, X. Wu, and Q. Wang. Sentiment analysis of yelps ratings based on text reviews, 2015.
14. [14] M. S. Elli and Y.-F. Wang. Amazon reviews, business analytics with sentiment analysis.
15. C. Rain. Sentiment analysis in amazon reviews using probabilistic machine learning. Swarthmore College, 2013.
16. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug):2493–2537, 2011.
17. R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In Proceedings of the 2013 conference on empirical methods in natural language processing, pages 1631–1642, 2013.
18. H. Cui, V. Mittal, And M. Datar, "Comparative Experiments On Sentiment Classification For Online Product Reviews," In Aaai'06 Proceedings Of The 21st National Conference On Artificial Intelligence, 2006.
19. B. Pang, L. Lee, And S. Vaithyanathan, "Thumbs Up? Sentiment Classification Using Machine Learning Techniques."
20. A. Y. N. And C. P. Richard Socher, Alex Perelygin, Jean Y.Wu, Jason Chuang, Christopher D. Manning, "Recursive Deep Models For Semantic Compositionality Over A Sentiment Treebank," Plos One, 2013.
21. Guan, L. Chen, W. Zhao, Y. Zheng, S. Tan, And D. Cai, "Weakly-Supervised Deep Learning For Customer Review Sentiment Classification," In Ijcai International Joint Conference On Artificial Intelligence, 2016.
22. B. Ay Karakuş, M. Talo, İ. R. Hallaç, And G. Aydin, "Evaluating Deep Learning Models For Sentiment Classification," Concurr. Comput. Pract. Exp., Vol. 30, No. 21, P. E4783, Nov. 2018.
23. Louis, F.& Wojciech, C., Book "Advances in The Human Side of Service Engineering", 5th International Conference on Applied Human Factors and Ergonomics, Volume Set, Proceedings of the 5th AHEE Conference 19-23 July 2014.
24. Larsen, Peder Olesen, and Markus von Ins. "The Rate of Growth in Scientific Publication and the Decline in Coverage Provided by Science Citation Index.",Scientometrics 84.3 (2010): 575–603. PMC. Web. 25 Sept. 2015.
25. Hassena, R.P., "Challenges and Applications", International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 3, Issue 5, 2014

26. Banea, C., Mihalcea, R., and Wiebe, J., "Multilingual Sentiment and Subjectivity Analysis", In Multilingual Natural Language Processing", editors Imed Zitouni and Dan Bikel, Prentice Hall, 2011.
27. Dushyant, B.R., & Samrat, K., "A Review on Emerging Trends of Web Mining and IT's Application", International Journal of Engineering Development and Research | IJEDR, 2013.
28. Yasen, Mais & Tedmori, Sara. (2019). Movies Reviews Sentiment Analysis and Classification. 10.1109/JEEIT.2019.8717422.

Essential links

1. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72.
2. https://www.analyticsvidhya.com/blog/2019/07/dont-miss-out-24-amazing-python-libraries-data-science/
3. https://www.edureka.co/blog/python-libraries/#z10
4. https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e
5. https://pdfs.semanticscholar.org/1082/9de79184fb0068f772e4c4e7632d4215c7b1.pdf
6. https://www.researchgate.net/publication/323536661_Sentiment_Analysis_using_SVM_A_Systematic_Literature_Review
7. http://cs229.stanford.edu/proj2018/report/122.pdf
8. https://stackabuse.com/python-for-nlp-creating-bag-of-words-model-from-scratch/
9. https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/
10. https://www.machinelearningplus.com/nlp/lemmatization-examples-python/

APPENDIX

1. All code is here https://github.com/sagorchandrou/sentimentAnalysisMovieReview