MAYNOOTH UNIVERSITY

MSC DATA SCIENCE AND DATA ANALYTICS THESIS

Sentiment Analysis of Movie Reviews

Author
LAKSHYA GAZARESEN
STUDENT NUMBER:19251460

Supervisor
DR. JOSEPH TIMONEY

A thesis submitted in fulfillment of the requirements
for the degree for the MSc in Data Science and Data Analytics 2019-2020
in the
Department of Computer Science
Maynooth University
August 10, 2020

Abstract

The overall effect and influence movies have on our culture is enormous. The failure or success of a movie is hugely dependent on the reviews of the viewers. Additionally, with the advancements with social media platforms, the ease of giving reviews has become quite smooth. There are two types of reviews for a movie; one is a professional critic based and the other is user-based reviews. Although critics review is helpful in understanding the movie, user reviews plays major role in adding to the success of a movie. Huge amount of data is generated online in the form of reviews by movie fans. Popular websites are IMDB, CinemaBlend, Rotten Tomatoes, Amazon etc and sentimental analysis of such type of data has gained huge prominence. Sentiment Analysis is used to capture the sentiments or the context behind the large number of opinions on a movie. The purpose of the analysis is to evaluate the perspective of the movie reviews of the largest online retailer, Amazon. The traditional approach of sentiment analysis is used, which identifies the dynamics of an opinion and classifies it as Positive, Neutral and Negative. Machine Learning algorithms such as Random Forrest, Naive Bayes, K-Nearest Neighbour Classifier and Support Vector Machine and a Deep Learning algorithm named Bert is applied to build a sentiment classification model. Finally, the accuracy of respective algorithm is compared and in conclusion it is observed that although it takes more processing time it is preferable to use a Deep Learning method over all other Machine Learning methods examined.

Declaration

I declare that the thesis titled 'Sentiment Analysis of Movie Reviews' under the supervision of Dr. Joseph Timoney, is my own work and is submitted for assessment by the Department of Computer Science as part of the course 'MSc in Data Science and Data Analytics'. Additionally, I have consulted various resources and references to finish the thesis and all the references have been acknowledged in the bibliography provided.

LAKSHYA GAZARESEN — AUGUST 10, 2020

STUDENT NUMBER: 19251460 — DATE

Acknowledgement

I would like to acknowledge and extend gratitude to my supervisor Dr. Joseph Timoney from Department of Computer Science, Maynooth University for all his time, guidance, coordination on this project. I greatly admire the time he spent in discussing the project consistently throughout the course of the thesis. The completion of the project could not have been possible without his sincere help.

I would also like to extend my sincere gratitude to all the professors of the Maynooth University for their valuable knowledge and rigorous training they provided me to gain the concepts and skills to finish dissertation.

I would also like to thank my peers and friends, having continuous discussions with them eased my understanding of certain topics.

Contents

List of Figures

Chapter 1

Introduction

Opinions are the key parameter of human behaviour and with the ubiquitous presence of digital platforms through Internet, the amount of opinionated information available, is humongous. People generally make a purchase, or watch a movie based on the opinions or perceptions of others which is in the form of reviews. As these reviews are generally the reason behind the success or failure of a product, hence, it is plausible to analyse the sentiment of the reviews. The future growth of businesses, organizations and market are hugely depended on the analysis of the reviews and this analysis has been taking place from quite some time. One person may generate various sentences, each sentence having its own complexity, but when thousands or millions of individuals or declarations has to be processed then the scenario becomes unmanageable and therefore machines has to be used. The machine's ability to understand the textual data is called as NLP (Natural Language Processing), "NLP is an Artificial Intelligence field that enables the machines to interpret, understand and extract meaning from human languages" [Narendra et al., 2016]. NLP is thriving due to tremendous advancements in data and improved computational capacity, which enables practitioners to produce practical outcomes in fields such as healthcare, finance, advertising and human resources, among others. Sentiment Analysis is an automated technique for extracting views from text and opinion mining has become a hot spot in many fields of inquiry, including NLP, information retrieval and data mining with a variety of applications such as advertising and recommendation systems, analysis of feedback from customers and customers decision making.

Motivation

Since the success rate of movies are tremendously dependent on the reviews the movies receive and there is an immense availability of reviews, the ability of a machine to understand and process the text is quite fascinating. Research on NLP is ever-growing and there have been many developments on making the machine understand the texts and its context. Research on analysing the sentiments of movie reviews had started before the 2000, and since then the research has been very successful. There has been research activities to analyse the sentiments of the movie reviews such as lexicon based and machine learning based approaches and with the advent of neural networks and Deep Learning, it has given a big boost to the quality of the results. The main aim of this project is to understand how some of the Machine Learning algorithms such as Naive Bayes, Random Forest, K-Nearest Neighbour Classifier and Support Vector Machine and a Deep Learning algorithm called BERT works in understanding and processing textual data. It is quite intriguing to discover how these algorithms are able to predict the sentiment when a raw data is given to a trained model.

Objectives

The project's purpose is to design a mechanism for carrying out sentiment analysis of movie reviews data using some of NLP's effective Machine Learning approaches and a Deep Learning tool. The goal is to create a framework which understands the context of the sentences provided and is smart enough to understand the satire and humor generally found in movie reviews. Additionally, the project scope includes the use of various techniques from NLP to pre-process the textual data to allow it to be used on all algorithms for Machine Learning.. In this report Naive Bayes, Random Forest, K-Nearest Neighbour Classifier and Support Vector Machines have been applied. The accuracy obtained from different model is compared and then the accuracy obtained from the machine learning algorithms are compared with the accuracy obtained from Deep Learning algorithm called Bert. Figure 1.1 shows the various approaches to obtain sentiments from text using computational techniques and it highlights the Machine Learning algorithms and a Deep Learning algorithm used in this project.

The dataset is taken from one of the online API (Application Programming Interface) [Ni et al., 2019] to retrieve reviews from Amazon. The raw data was pre-processed using the Python Programming Language to obtain the pertinent format. Once the text data is converted in word tokens and unnecessary stop words are removed, the tokenized data and its associated labelled sentiment was

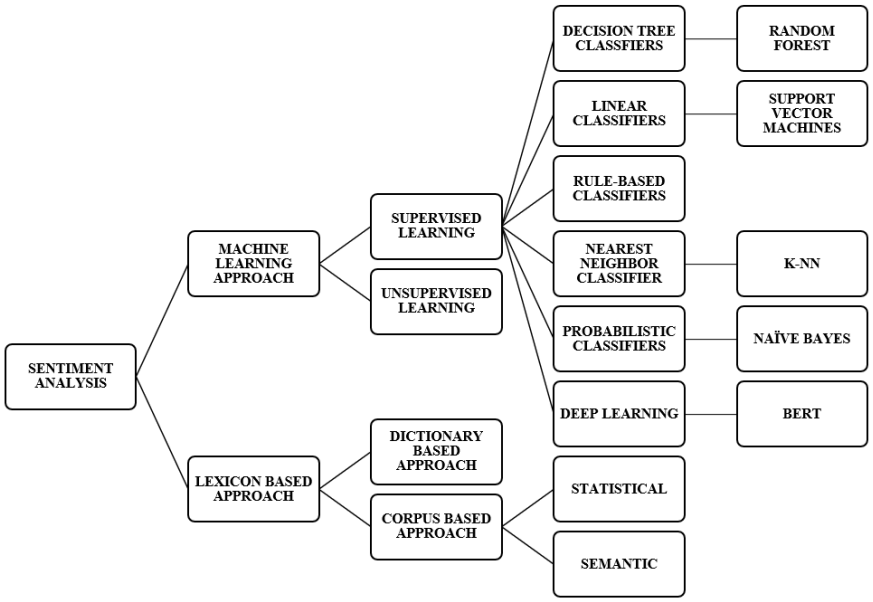


Figure 1.1: Sentiment Analysis Approaches

divided into two parts, 80% of the data has been taken as training dataset to train the various algorithms and remaining 20% as test dataset, to test the working of the method on which it got trained. The percentage of dataset for training was then used to train the mentioned Machine Learning algorithms and the accuracy of predicting the correct sentiment of the movie reviews was checked on the test dataset. Further, the same text data was used to train the data for the Deep Learning algorithm, BERT. The accuracy was observed for each epoch as the trained model achieved optimal accuracy with each epoch. The objective of the report is to provide a comparative analysis of existing techniques of sentiment analysis using Machine Learning and Deep Learning methods. Further, the aim is to achieve an optimal accuracy for the mentioned models to predict the sentiments of the movie reviews efficiently.

Chapter 2

Background

2.1 Literature Review

The target of the sentiment analysis is to make a machine understand a viewer's emotion while giving the reviews online. Opinion mining and sentiment analysis has been an area of interest for many study and research with the inception and growth in usage blogs, social-media websites, online product reviews etc. There are various approach to obtain the sentiment of the texts. An extensive research on this field with all the approaches was published by [Lee et al., 2008]. The authors identify the current techniques and strategies for an opinion-oriented retrieval of information in their survey. One of the most common used method to discover the sentiments is the Lexicon approach. In one of the publisehd work of MIT by [Taboada et al., 2011] the discussion and impact of Lexicon based approach was discussed. The approach based on lexicon involves determining a text's orientation from the semantic orientation of words and sentences [Turney, 2002]. The lexicon approach is a word-based method and the sentiments of the words are correlated with a pre-defined or manually built dictionary and the precision of the model is based on how efficiently the used dictionary is functioning. It is also known as rule-based approach and the sentiment score is calculated based on the total number of negative and positive words out of total number of words. In their research [Godbole et al., 2007] presented a framework which quantifies a positive or negative opinion in the text corpus for each distinct entity. Their method consisted of two phases, a process of sentiment identification in which opinionated entities are calculated then a process of scoring in which a score is calculated for each entity. In [Jurek et al., 2015] analysis of data from Twitter was applied in an attempt to estimate box-office sales of movies using Lexicon based approach. After their analysis it was

found that there was a relation between the box-office performance of the movie and the rate of the movie tweets. A Lexicon based approach has been used in many studies of sentiment analysis and improvement on the precision of this approach is an ongoing research.

The evolution of Machine Learning to process textual data created a benchmark in the field of analysis of sentiments obtained from opinionated statements. The text classifiers based on Machine Learning belong to the paradigm of supervised learning, where the classifier shall be trained on some labelled data before it can be applied to the actual task of classification. The training data is the proportion of the actual labelled dataset. In the survey performed by [Hailong et al., 2014] the accuracy of a Lexicon based approach and the Machine Learning approach were compared. It was observed that Machine Learning techniques such as Naive Bayes and SVM have performed better and can be viewed as standard learning strategies, whereas as there have been advancement in lexicon-based approaches still the capability of Machine Learning algorithms to predict the sentiment is more precise. This study also highlighted the observation that although Machine Learning algorithm is able to achieve better accuracy but this form of methodology requires a great deal of work in human text annotation and whose performance is highly reliant on the quantity and content of the training dataset, and may struggle when training data is skewed or insufficient. In their study [Annett and Kondrak, 2008], it has been observed that Machine Learning algorithms of classification of sentiments of reviews of movies is very effective and it has also been highlighted that the selective type of features has an effective impact on the classifier's accuracy. One of the standard reference in sentiment analysis of movie reviews is considered to be the publication of [Pang et al., 2002]. This study considered the issue of classifying documents by overall sentiment and not by topic, for example, determining if the movie review is negative or positive and concluded that the classical Machine Learning methods achieve greater outcomes than the human-produced baseline. Even in the survey by [Singh et al., 2013] it was observed that Machine Learning methods comparatively performed better than one of the improved lexicon-based methods named SentiWord, which uses the SentiwordNet Library [Gatti et al., 2015], this provides the values of polarity of each words in a review. The ability of the Machine Learning model to understand sarcasm or humor in the reviews has also to be considered for better analysis of sentiments. This issue was rectified by using Deep Learning methods.

Deep Learning which is also referred as "Deep Neural Networks" is a recent technology within Machine Learning. Deep Learning has proven quite effective in both supervised and unsupervised learning and many researchers apply Deep

Learning methods to achieve sentiment analysis. It consists of multiple useful and effective models such as Convolutional Neural Networks (CNN), Recursive Neural Networks (RNN) etc. The study by [Ain et al., 2017] outlined the advantage of Deep Learning for sentiment analysis. Because the sentiment analysis is used to forecast the opinions of the consumers and Deep Learning models are about anticipating and mimicking the human mind, the Deep Learning models provides more accuracy than other types of approaches for sentiment analysis. One of the recent and effective Deep Learning model for understanding the language is provided by Google in 2018 and is called BERT (Bidirectional Encoder Representations from Transformers) [Devlin et al., 2018]. The comparison of different Deep Learning algorithms with BERT was studied by [Munikar et al., 2019]. In this report various different algorithm were tried on different datasets and in both the cases BERT performed better than other Deep Learning algorithm. It was also explained BERT posses a simple downstream architecture compared to recurrent, convolutional and recurrent neural networks. In our report, we have explored BERT algorithm on our dataset and examine the accuracy it provides.

As the data to be processed is vast and these data require many predetermined libraries for NLP tasks, the Python programming language is important. Additionally, data processing libraries such as numpy, pandas, scikit learning, libraries specific to NLP such as nltk and Deep Learning libraries for BERT. These libraries proved quite helpful during the development phase.

2.2 Machine Learning Techniques

"Machine Learning is the science of getting computers to function without being "explicitly" programmed" [Jordan and Mitchell, 2015]. Machine learning is targeted at classifications and predictions and data is the soul of Machine Learning. In the last two decades, Machine Learning has advanced dramatically from research concerns to industrial use of functional technology. The artificial intelligence approach for the development of practical applications for computer vision , speech recognition and robot control is an application of Machine Learning [Jordan and Mitchell, 2015]. The process of Machine Learning depends on the way the algorithm gets trained from the input data and there are two ways in which a machine learns, unsupervised learning and supervised learning. The key distinction between the two types of Machine Learning approaches is that in supervised learning previous knowledge of the output values for corresponding input values is known. The goal of supervised is therefore, to learn a function which, given a sample of input data and desired outputs, best approximates

the relationship between output and input that can be observed in the data. In comparison, unsupervised learning does not have labelled outputs, and its goal is to obtain a structure present within a collection of data points. The Amazon DVD reviews data is a labelled data, hence it falls under supervised learning. The supervised and unsupervised learning can be of a classification (discrete output variable) or a regression (continuous output variable) function. The text data is continuous and falls under classification. Prediction is achieved with a mapping function that maps independent variables to dependent variables [Mohammed et al., 2016]. The various Machine Learning algorithms used to train the dataset to predict sentiments are Naive Bayes, Random Forest, Support Vector Machine and K- Nearest Neighbour Classifier algorithms. The working of the mentioned algorithms are explained in the subsequent sections.

2.2.1 Naive Bayes Algorithm

Naive Bayes classification describes both supervised method of learning and predictive approach of classification. This is an approach that helps to consider the model's ambiguity with the assistance of probabilities in a rational manner. It helps in diagnostic and predictive problem solving. The term "naive" is used because it implies that the input features of the model are independent of each other which means changing the value of one feature, does not explicitly alter or influence the value of any other features used in the algorithm and this also takes less processing time to run the model. As the input features are independent it generally fairs better for text classification and sentiment analysis [Baid et al., 2017]. Bayesian classification delivers useful learning algorithms and can be with observed data and past knowledge. This helps to assess specific hypothesis probabilities, and is also resilient to noise and input data. Bayes theorem is shown in Figure 2.1.

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

where,

- $P(c \mid x)$ - Posterior probability of class (c(target)) given predictor (x, attributes).
- $P(c)$ - Prior probability of class (c).
- $P(x \mid c)$ - Probability of predictor (x) given class (c).
- $P(x)$ - Prior probability of predictor (x).

Figure 2.1: Bayes Equation (source: [Baid et al., 2017])

2.2.2 K-Nearest Neighbor

The K-Nearest Neighbors (KNN) is a basic supervised Machine Learning approach that is easily implemented and used for both regression and classification problems. KNN uses a distance calculation to identify the nearest cases from the training data set when making a prediction. The distance calculation chosen must take into account the essence of the problem in such a way that related data instances belong to the same class according to the distance measure. The widely used measure to calculate the distance for the input values that have same scale or measure is the 'Euclidean Distance' and it is determined as the square root of the sum of squared differences between data point 'a' and 'b' for all the input data 'i'[Brownlee, 2016].

$$\text{Euclidean Distance(a, b)} = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

The entities for which the Euclidean Distance is small are grouped together based on the "k" value. The value of 'k' has to be chosen precisely in order to get minimum error and better accuracy to predict the desired results in KNN algorithm [Trstenjak et al., 2014]. The "k" value is chosen as 3 in this project.

2.2.3 Random Forest

Random Forest is a supervised Machine Learning ensemble algorithm, it is called as an ensemble algorithm because it is an ensemble of decision trees. For making predictions this algorithm leverages the power of multiple decision trees.The algorithm includes choosing of the decision trees at random and to calculate output, each node in the decision tree operates on a random subset of features [Da Silva et al., 2014]. The result of each individual decision tree is combined in order to obtain the final result. The figure 2.2 visualizes the working of the Random Forest algorithm.[Liaw et al., 2002] and it showcases how Random Forest algorithm efficiently uses the combination of Decision Trees to achieve the desired result.

The biggest benefit of the Random Forest algorithm is that it can be used effectively for both regression and classification type problems. It also uses hyper-parameters to produce relatively better prediction results or to make the model faster, few of the hyper-parameters are "`n_estimators`" to select the number of decision trees, "`max_features`" to select the maximum number of features to split a node, "`n_jobs`" to define the number of processors to be used, "`random_state`" to have the same output for a specific value of random_state and specific input values [Brownlee, 2016] . The main drawback of Random Forest is that the

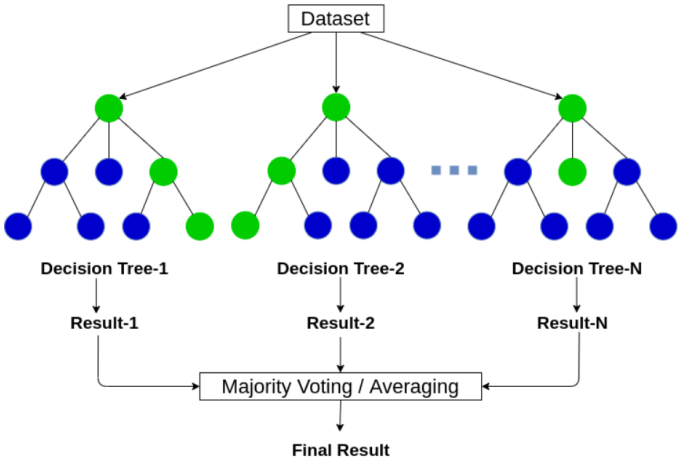


Figure 2.2: Random Forest Algorithm (source: [Sharma, 2020])

algorithm can be rendered too slow for real-time predictions if huge number of trees is chosen. The algorithm is usually quick to learn, but very slow to make predictions once they are trained, a more precise estimation requires more trees. resulting in a slower model. The Random Forest algorithm is considered to be one of the best algorithms for classification - capable of accurately classifying large quantities of data [Al Amrani et al., 2018].

2.2.4 Support Vector Machine

The Support Vector Machine (SVM) method is a statistical classification method focused on maximizing the difference between the instances and hyper-plane separation. It has proven effective in the classification [Xia et al., 2011]. It is a binary linear classifier that would be able to divide classes linearly by a wide margin. This classifier is one of the most powerful ones, able to handle infinite-dimensional vectors. The SVM is a supervised Machine Learning algorithm which can be used both for regression and classification problems. In this algorithm, each data variable is drawn as point in n-dimensional space, with the value of the unique coordinate being n (where n is the number of characteristics). The classification is then carried out by obtaining the hyperplane that properly distinguishes between the two groups [Al Amrani et al., 2018].

An important assumption is that the linear SVM can be re-framed with any two observations using the internal point variables. This element is known as 'Kernel' and the equation is determined between the input (x) and the support vector (x_i) of the new input prediction with the dot product which is calculated

as follows:

$$K(x, x_i) = \sum(x \times x_i)$$

It is an equation which involves the calculation by all support vectors of the products from the internal dot of a new input vector (x). The kernel defines the similarity or distance comparison between the data input and the support vectors. The dot product is the measure of the similarity of the linear kernel if the distance is a linear input combination. The input space is converted into higher dimensions like a radial kernel and a polynomial kernel based on the dot product. Complex kernels are recommended because the lines distinguish groups in curved form, or more complex form, resulting in more detailed classifications.

2.3 Deep Learning

A thorough research on sentimental aspects or entities must be undertaken to identify entities and relevant aspects and to describe sentiments associated with them. In general, this kind of fine-grained analysis is achieved successfully through Machine Learning which although require domain specific, large data sets and training of data manually [Hu and Liu, 2004]. Experimental Machine Learning methods have shown higher accuracy with Deep Learning approaches [Poria et al., 2016]. Deep Learning comprises of multi-layer processing procedure that uses consecutive layers of modules to obtain the desired result. The input is transformed into numerical representations for each layer, which are then subsequently classified. Therefore, a higher level of precision is achieved. A collection of algorithms such as Deep Neural Networks (DNN), Convolutional Neural networks (CNN), Recurrent Neural Networks (RNN), Recursive neural networks (RecNN) etc, facilitate research in various fields with deep neural networks especially suited to fine-grained work because of deeply connected layers of connected processors [Schmidhuber, 2015]. The NLP by deep neural networks consists multiple hidden layers between the output and input units and dense word embedding. The Figure 2.2 showcases the difference between classical NLP approach and Deep Learning based NLP.

The dense embedding in Figure 2.3 are n-dimensional representations [Thanaki, 2017] of word tokens encoded as numerical vectors, these vectors exhibits the probability of the word present within a specific word matrix [Rojas-Barahona, 2016]. There are various pre-trained word embedding methods so that words can obtain general semantic information such as word2vec, Glove etc [Mikolov et al., 2013]. The second function for NLP are the hidden layers which can be designed for different techniques, feed networks and recurrent NLP networks.

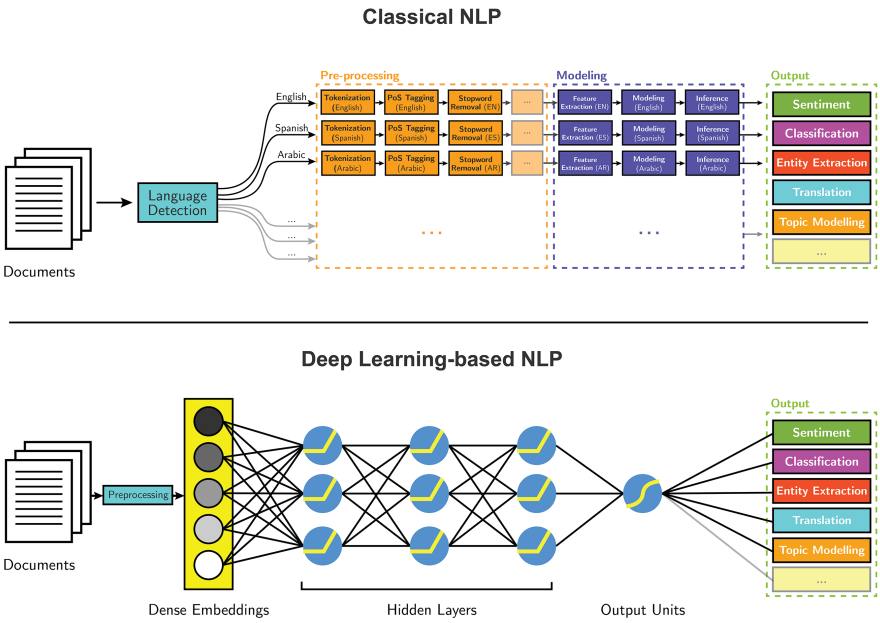


Figure 2.3: Comparison of Classical NLP and Deep Learning based NLP (source : [Schmidhuber, 2015])

Each hidden layer consists of several neurons stacked to calculate non-linear outputs. In general, the higher layers developed by training to manipulate the complex nonlinear compositional functions of lower layers and thus, obtains better abstract representation than the lower layers. The calculation of hidden feature begins with neurons which takes 'n' inputs to give single output. If the inputs $x_1, x_2, ...x_n \epsilon R$ with associated 'n' weights/ parameters $w_1, w_2 ... w_n \epsilon R$ and a bias scalar $b \epsilon R$, the activation of neurons is obtained as $a = \sum_{i}^{n} w_i x_i + b$ [Do et al., 2019]. Hence, the activation function is used to calculate the output:

$$o = s(a) = s(\sum_{i}^{n} w_i x_i + b)$$

The non-linear activation is either the hyperbolic tangent function, or the sigmoid function, or the rectified linear function[Do et al., 2019]:

$$sigmoid(a) = \frac{1}{1 + e^{-a}}$$

$$tanh(a) = \frac{e^{2a} - 1}{e^{2a} + 1}$$

$$ReLU(a) = max(0, a)$$

The qualitative element (output units) represents the probability distributed over all the labels or groups. If the last layer is 'z' and there are 'K' labels/groups, the likelihood of using the softmax function can be obtained for the label 'i' as shown below [Do et al., 2019]:

$$y_i = softmax(z)_i = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

2.3.1 BERT Algorithm

Sentimental analysis from text data has undergone a monumental transformation with the introduction of pre-trained transformer models such as Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018]. Before the introduction of BERT, the standard language models were unidirectional which means the transformer models used to follow left-to-right architecture [Vaswani et al., 2017]. In such unidirectional models, in the self-attention layers of the transformer. The tokens could only attend to the previous tokens. These constraints are not optimal for sentence-level tasks, and might be inaccurate while applying fine-tuned based solutions to token-level tasks such as answering questions and sentiment analysis, where context from both directions is essential to implement. BERT rectifies the constraint of unidirectionality by using a MLM (Masked Language Model) [Devlin et al., 2018]. The MLM masks randomly few of the data tokens. The goal is to predict, based on their meaning, the masked word 's original ID. Contrary to the left-to-right method of pre-training, the MLM aims to combine the left and right contexts, allowing a bi-directional transformer to be prepared. [Radford et al., 2018]. The masked tokens are predicted by BERT instead of the entire input. The pre-trained representation of BERT eliminates the need for other heavily designed, specific to task algorithms. BERT is the first model focused on fine tuning representations that achieves efficiency on a wide resource of statement-level and token-level tasks, surpassing many task-specific algorithms [Devlin et al., 2018].

Chapter 3

Methodology

The strategy pursued in project execution is mentioned in this chapter. It provides a high degree of understanding of the overall process and the technical specifics of the approaches employed in the project.

3.1 Data Collection

For this project the Amazon movies and TV shows reviews is taken and it is extracted from an API which contains reviews of all type of products sold by Amazon [Ni et al., 2019]. From the website "Movies and TV" reviews are extracted as we are interested in analysing the sentiments reviews of movies and television shows. The parameters in the dataset is shown in Figure 3.1.

Variables	Description
reviewerID	ID of the reviewer
asin	ID of the product
reviewerName	Name of the reviewer
vote	helpful votes of the review
style	A dictionary of the product metadata
vote	Helpful votes of the review
reviewText	Text of the review
overall	Rating of the product
summary	Summary of the review
unixReviewTime	Time of the review (unix time)
reviewTime	Time of the review (raw)
image	Images that users post after they have received the product

Figure 3.1: Terminologies of the Dataset

3.2 Data Pre-Processing

3.2.1 Pre-Processing for Machine Learning Algorithms

The study of unstructured text data and the collection of useful information from the data which can be used for further research are the focus of text mining / text processing. Text mining is a diverse research area that typically involves Machine Learning , statistics and data mining applications [Kao and Poteet, 2007].The text review process can be accomplished with the use of a various methodologies provided by NLP. NLP and its components are used to perform automated operations on raw text data that manage tasks such as language translation, speech recognition, sentiment analysis analysis, etc. The library used for processing NLP in the Python programming language is 'nltk' (The Natural Language Toolkit [Thanaki, 2017]). NLTK allows convenient access to a wide variety of word processing NLP function libraries, such as trailing, tokenization, lemmatization, tagging, parsing and grouping etc.

The dataset contains several irrelevant details. Only the useful variables were retained when converting the data from 'JSON' format to pandas dataframe, such as, 'text' referring to the actual text of the review, 'reviewerName' to identify the reviewer, 'overallscore' which contained the scores given by the reviewer with 1 being lowest and 5 being highest. Based on the 'overallscore' the sentiment have been labelled under three ratings with 0 being negative, 1 as neutral and 2 as the positive rating. It was observed that the positive reviews are dominant in the dataset, hence, a sample of the dataset is taken with an equal amount of positive, neutral and negative reviews to have a balanced dataset. All missing and duplicate values from the dataset were removed. Figure 3.2 shows the class balance in the dataset and highlights that equal amount of all three types of reviews were taken so that the Machine Learning and Deep Learning algorithms are trained efficiently.

The precision of an NLP method varies with different approaches of cleaning of the text data. We should, therefore, have an accurate technique that provides a framework for the cleaning and parsing of textual data. In our case, first the whole review is divided into sentences, then each sentence is divided into tokens of words [Kao and Poteet, 2007]. After converting the word to tokens, the data is applied on raw review-tokens to transform them into functional data for analysis of sentiments, the steps taken to clean the text data are as follows [Kao and Poteet, 2007]:

1. **Convert all the tokens to lower case tokens:** In the data cleaning process, the first step taken is to convert all the review tokens into lower case letters.

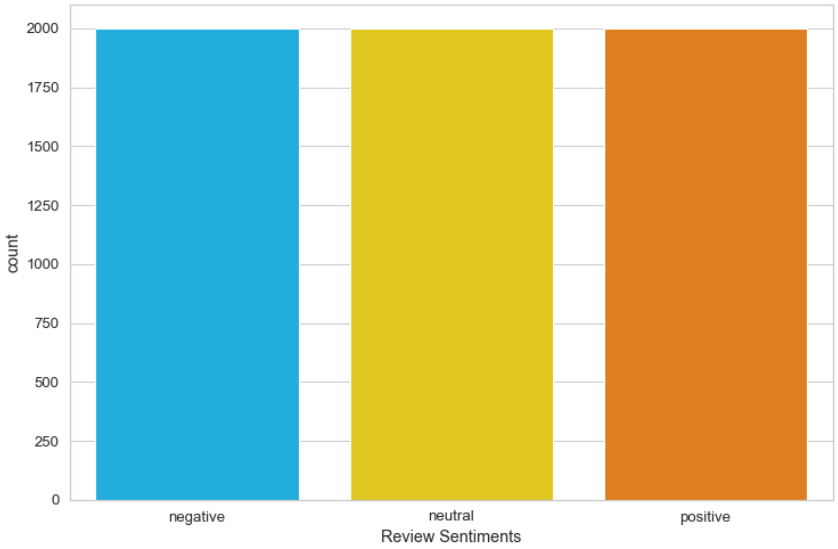


Figure 3.2: Balanced Dataset

This helps to avoid having having different versions of the same word in the dataset. For instance, 'Awesome', 'AWESOME' and 'awesome' are considered different words by the word embedding libraries.

2. **Converting numbers to words:** Users can also give reviews of movies based on scores, for example " I rate this movie 5 stars". For movie reviews these type of data plays an important role, hence, the all the numbers have been converted to its alphabetical form. Therefore '5' becomes 'five'.

3. **Expanding Contractions:** Generally reviewers use shortened words while giving reviews online, these types of words are called contractions. For instance "do not" is written as "don't", " could not" as "couldn't" etc.These contraction form of words cannot be interpreted by the Machine Learning algorithms and has to expanded. Hence, it has been expanded with the help of pre-defined dictionary. For instance 'hasn't' transforms to 'has not',

4. **Remove non-alpha characters:** The next step is to remove the punctuation from the reviews because the punctuation are not relevant in obtaining the sentiments of these reviews. It also helps to reduce the size of the training set by removing all the unused tokens.

5. **Text Standardization:** The reviews are generally not in formal format. For

instance a reviewer can type "happy" as "haaaappyy" to put emphasis on the word and denote their emotions. Using simple regular expressions these cases can be dealt with. Without this step, lemmas of this kind of words cannot be found.

6. **Lemmatization:** There are two methods for reducing inflexible and derivative word forms, stemming and lemmatization, to simple unique basic forms. Stemming refers to a primitive heuristic method that chops off the end terms in the expectation that the goal will be accomplished correctly most of the time and also involves eliminating derivative affixes. On the other hand, Lemmatization uses a vocabulary and morphological examination of words, generally aimed solely at eliminating inflection endings and restoring the basic or dictionary form of word known as "lemma". The essential distinction to highlight is that lemma is the fundamental component of all it's inflective forms, while a stem is not. Few examples of difference between Lemmatization and Stemming are shown in Figure 3.3 and it can be observed that Lemmatization is better to apply in Machine Learning algorithms [Jabeen, 2018], in our report we have used the Lemmatization technique [Schütze et al., 2008].

Form	Suffix	Morphological Information	Stem	Lemma
sudies	"-es"	Third person, present tense of verb study	studi	study
studying	"-ing"	Continous Tense of the verb study	study	study
cries	"-es"	Third-person singular simple present indicative form of cry	cri	cry

Table 3.2: Stemming vs Lemmatization

Figure 3.3: Example of Stemming and Lemmatization

7. **Removing Stop Words:** Words which have are commonly used and have very little meaning are called as stop words such as "is", "an", "the etc. Stop words are often omitted from the text before applying Deep Learning or Machine Learning methods as they occur in abundance, thereby offering little or no specific knowledge that can be used for clustering or classification. In Python stop words are removed using pre-defined nltk library. Although removing stop words are important for sentiment analysis but some of the stop words like "no", "never" and "not" can impact the overall sentiment of the reviews,hence, in this report these stop words are not omitted from the main reviews.

In the case of analysis of textual data , a 'Word Cloud' has been used to to visualize the information from the text magnificently. Word Cloud is constructed based on the frequency of the words in the text. The more frequently a certain

word appears in the text in word cloud visualization, the bolder and larger it appears in the word cloud. This helps us to measure text based data to a observable measure. The word cloud for the Amazon DVD reviews dataset is shown in Figure 3.4. Evidently, words such as 'movie', 'film' are bolder and larger because the dataset we have used is of movie reviews so hence, it is quite rational to frequently see such words. The word cloud also states that words used more often in reviewing the movies are 'good', 'great', 'love', 'story' etc, are more frequent words which viewers use while giving movie reviews online.

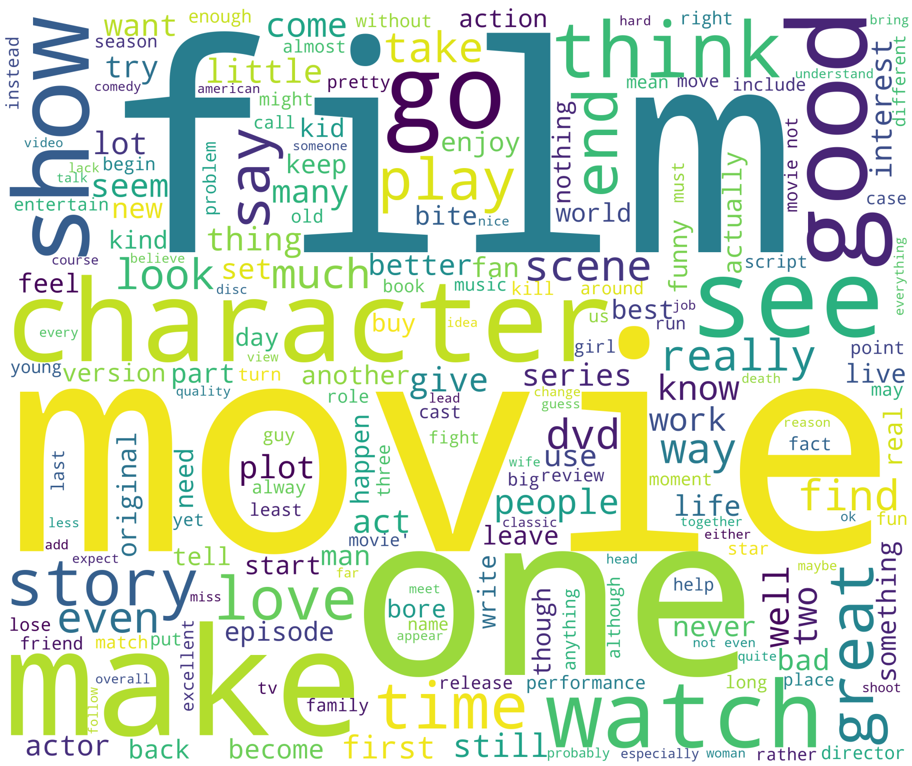


Figure 3.4: Word Cloud of Movie Reviews

An example of the text pre-processing for ML algorithms is shown in Figure 3.5 below: A sample review is taken and then in order to make the text compliant with the Machine Learning algorithms, the sentence is cleaned and pre-processed with the techniques mentioned earlier, i.e, converting to tokens, removing special character, unnecessary stop words and finally lemmatizinng the tokens.

Once the reviews are cleaned and tokenized, it has to be converted to vectors in order to make it compliant for applying the Machine Learning Algorithm. Using bag-of-words model vectorisation is achieved in three steps: the first step is to

```
print(df_reduced['text'][11])
```

```
Picture was awful Story was awful
Worst movie I've seen in a long time
Very disappointed
Picture very fuzzy
```

```
print(Preprocessing(df_reduced['text'][11]))
```

```
['picture', 'awful', 'story', 'awful', 'worst', 'movie', 'see', 'long', 'time', 'very', 'disappoint', 'picture', 'very', 'fuzz
y']
```

Figure 3.5: Text Pre-processing using NLTK Library

measure the count of the number of times a word appears in each review, which is known as the frequency. The second step is to evaluate the counts to such an degree that the most frequent word is assigned a lower weight. The term frequency is obtained in python programming language using a library named "CountVectorizer", while the second and third step is obtained by tf-idf, which stands for "term frequency-inverse document frequency". The tf-idf weight is a transform method typically used in retrieval of information and in text mining. The tf-idf weight is a statistical measure use to determine importance of a word is in the review. TTf-idf contains two main words. One is the normalized Term Frequency (TF) metric, it is the number of times a word appears in each sentence, and the second is the Inverse Document Frequency (IDF) measured as the number of documents in the corpus and the number of documents in the corpus as logarithm. The second expression is calculated by taking the logarithm of the ratio of the total numbers of documents in the corpus to the total number of documents having the same words [Wu et al., 2008]. The Figure 3.6 shows the expressions in the tf-idf transform.

$$\text{tf(t)} = \frac{\text{The number of times term t appears in a document}}{\text{Total number of terms in the document}}$$

$$\text{tf(t)} = \log_e \frac{\text{Total number of documents}}{\text{Number of documents with term t in it}}$$

Figure 3.6: TF-IDF expressions

3.2.2 Pre-Processing for BERT Algorithms

The pre-processing of textual data for BERT is slightly different from other Machine Learning algorithms but the basic concept remains same as the machine cannot understand raw text, the sentences have to be tokenized and the text has

to be converted to numbers. To tokenize the sentences, a pre-defined library is used for BERT which is a pre-trained model [Devlin et al., 2018]. The pre-defined model has two types of tokenizers: cased and uncased, since online movie reviews are of unstructured form and same words with different case may have different impact on the reviews, for example, "BAD" and "bad". By using capital words, reviewers try to put more emphasis or stress on the word to express their opinions about the movies, hence we have chosen 'cased' BERT tokenizer[Devlin et al., 2018]. Deep Learning methods have two main frameworks namely, "TensorFlow" and "Pytorch" developed by Google and Facebook respectively [Simmons and Holliday, 2019]. Both the frameworks have compelling tensor computation with strong GPU acceleration. In this report we have used Pytorch as our framework to carry out Deep Learning algorithm.

After tokenizing the sentence, the tokens are given an ID based on the BERT tokenizer so that the algorithm can understand what each token represents. There are some special tokens in BERT tokenizer such as [SEP] (Token ID - 102) which indicates the ending of a sentence, [CLS] (Token ID - 101) represents the start of the each sentence and is required to be used for classification, [PAD] (Token ID - 0) is used to pad the a set of reviews to a chosen fixed length and [UNK] (Token ID - 100) is for tokens which are not identified by the BERT tokenizer.

Since the tokens are of varying lengths and BERT works on a Masked Language Model approach, a variable named attention mask is used, which represents "1" where the tokens have been correctly tokenized with the help of BERT tokenizer and "0" for the tokens where padding has been used. An example of how BERT tokenizer pre-processes on a sample text is shown in Figure 3.7. A sample sentence is taken as "When I was last outside? I am stuck at home for 3 months due to current situation". BERT cased tokenizer is used in this sentence to convert the sentence to tokens. The "`Max_Length`" is taken as 32 for this example, then pre-processing of these tokens has been highlighted with assigning token ids to the words based on BERT tokenizer. It can be observed in Figure 3.7 that the respective token IDs are assigned for the respective words till 20 word tokens (as the sample text comprises of 20 words) and the remaining 12 tokens are padded with the token '0', highlighting that remaining tokens are the padding tokens ([PAD]) till the "`Max_Length`" of 32 in this case.

The variable "`Max_Length`" is chosen based on the average length of all the reviews as BERT works with the fixed-length sequences. For the sequences whose token-length is less than the "`Max_Length`", the sequences is padded to the "`Max_Length`" using [PAD] token, and for the sequence whose token-length is more than the "`Max_Length`", only tokens till "`Max_Length`" is taken and remaining

```
tokens = tokenizer.tokenize(sample_txt)
token_ids = tokenizer.convert_tokens_to_ids(tokens)

print(f' Sentence: {sample_txt}')
print(f' Sentence: {tokens}')
print(f' Sentence: {token_ids}')
print(encoding['attention_mask'])
```

```
 Sentence: When was I last outside? I am stuck at home for 3 months due to the current situation.
 Sentence: ['When', 'was', 'I', 'last', 'outside', '?', 'I', 'am', 'stuck', 'at', 'home', 'for', '3', 'months', 'due', 'to', 't
he', 'current', 'situation', '.']
 Sentence: [1332, 1108, 146, 1314, 1796, 136, 146, 1821, 5342, 1120, 1313, 1111, 124, 1808, 1496, 1106, 1103, 1954, 2820, 119]
tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0]])
```

Figure 3.7: An Example of Pre-Processing for BERT

words are skipped, Hence, "`Max_Length`" should be chosen in such a way that it should contain maximum of words of the review. The length of each review from the dataset is plotted in Figure 3.8. The bar plots in the Figure 3.8 shows the histogram of the length variable and the superimposed line shows the probability density of the length variable. It can be observed that most of length of the reviews are around 200, hence, the "`Max_Length`" is chosen as 200 for the BERT model.

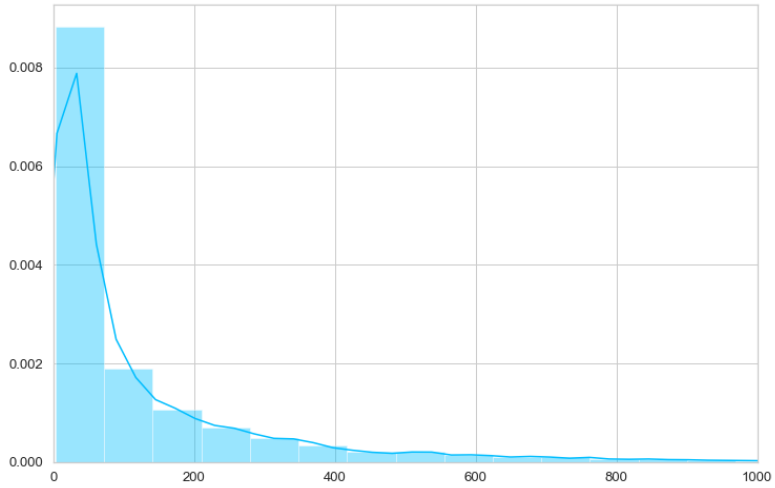


Figure 3.8: Token Lengths

3.3 Implementation of Machine Learning

3.3.1 Pipelining

The overall flow of a Machine Learning algorithm consists of text pre-processing, then the application of Machine Learning algorithm leads to predicted sentiments of the reviews shown in the Figure 3.9. The figure 3.9 elaborates the steps of

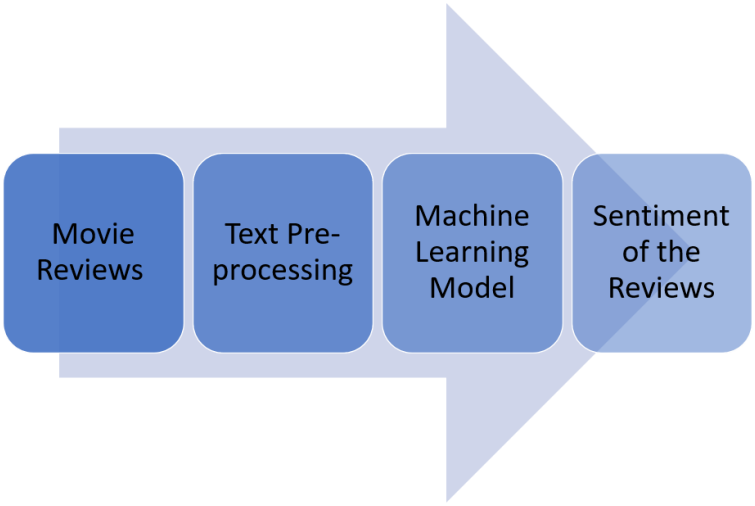


Figure 3.9: Flow Model of ML algorithm

Machine Learning algorithm for sentiment analysis and it can be observed that it is a sequential flow of processes, in another terms, the whole process is in a pipelined manner. The sklearn library in the python programming language provides the feature of pipelining the steps of Machine Learning. Pipelining helps to calibrate models and make predictions directly using the original data. All the Machine Learning algorithms are applied using the pipeline function. The count-vectorizer, TF-IDF transformer and each of the various mentioned Machine Learning classifiers has been pipelined to obtain the prediction of sentiments of the movie reviews.

3.3.2 Classification Report

The classification report displays a per-class representation of the key classification metrics. This gives the behaviour of classifier a deeper intuition over global accuracy. The visualizer for the classification report displays the model's precision, recall, F1 scores and support scores. There are four parameters on which the correctness of a prediction is judged [Narkhede, 2018].

1. True Negative (TN): The observation is said to be True Negative (TN) when the predicted value is negative and the actual observation is negative.
2. True Positive (TP): The observation is said to be True Positive (TP) when the actual observation is positive and the predicted value is positive as well.
3. False Positive (FP): The observation is said to be False Positive (FP) when the predicted value is positive but the actual observation is negative.
4. False Negative (FN): The observation is said to be False Negative (FN) when the predicted value is negative but the actual observation is positive.

A confusion matrix is a table that is often used to describe a classification model's performance on a set of test data for which the true values are known [Santra and Christy, 2012]. This allows visualization of an algorithm's performance. The advantage of using a confusion matrix is that it gives better picture of what the classification model is doing correct and how many times it is not predicting the result as per the expectation.

The classification report and the confusion matrix is judged based on few parameters such as:

1. **Precision**: Precision is calculated by taking the ratio of total number of correctly predicted positive observation to the total predicted positive observations. It is the ability of a classifier not to predict an observation negative that is actually positive [Narkhede, 2018].

$$\text{Precision} = \frac{\text{TP}}{\text{(TP+FP)}}$$

2. **Recall**: Recall is the ratio of predicted correct observations to all the observations in the model. It highlights the ability of the classifier to predict all the positive instances [Narkhede, 2018].

$$\text{Recall} = \frac{\text{TP}}{\text{(TP+FN)}}$$

3. **F1 Score**: F1 score is the weighted average of Recall and Precision. It takes both false negatives and false positives into account. Usually F1 score is more useful than accuracy, especially if there is an uneven class distribution [Narkhede, 2018].

$$\text{F1 Score} = \frac{2 * \text{(Recall * Precision)}}{\text{(Recall + Precision)}}$$

4. **Support**: Support is the measure of the actual occurrences of a class in a particular dataset. If the support is imbalanced in the training data then it may indicate systematic deficiencies in the classifier's scores and may indicate the need for stratified sampling or re-calibration. Support will not change between models but instead treat the process of evaluation [Narkhede, 2018].

5. **Accuracy**: Accuracy is the intuitive indicator of performance of a classifier, it is calculated by taking the ratio of correctly predicted observations to overall number of observations. It's a way of determining how often the selected algorithm is classified correctly. It is obtained by calculating the following equation [Narkhede, 2018].

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{(TP+TN+FP+FN)}}$$

3.4 Implementation of BERT Algorithm

The BERT algorithm works in two phases; pre-training phase and fine tuning phase based on domain specific dataset. This step can be leveraged by the research by Google [Devlin et al., 2018] and is used to fine tune the model based on Amazon DVD movie reviews. The pre-training phase is done on sentences retrieved from BookCorpus (800M words) and Wikipedia (2500M words) [Kiros et al., 2015]. The pre-trained model for BERT is built undertaking certain hyper-parameters such as Masked LM of 512 tokens per sequence, batch size of 256 sequences with epoch for training the model is set to 40. To obtain the learning rate for optimal gradient descent in neural network, Adam coefficient with learning rate of ($1e^{-4}$) is chosen. As mentioned above this step is taken care by Google and with the help of this pre-trained mode, BERT can be applied with different hyper-parameters to the data set for fine-tuning. The learning rate for Adam is chosen as ($2e^{-5}$) based on [Devlin et al., 2018]. The batch-size of 16 sequences is chosen with 20 epochs for training and validation model [Devlin et al., 2018]. As taken in Machine Learning algorithms, 20% of the dataset is chosen as test data and remaining as training data, whereas 50 % of the dataset is chosen as validation data as cross-validation technique is applied to obtain better accuracy [Brownlee, 2018].

Chapter 4

Results and Discussion

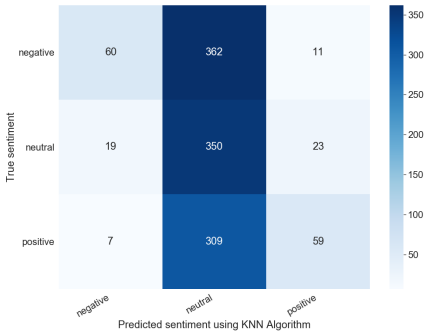
4.1 Results

4.1.1 Machine Learning Algorithms - Results

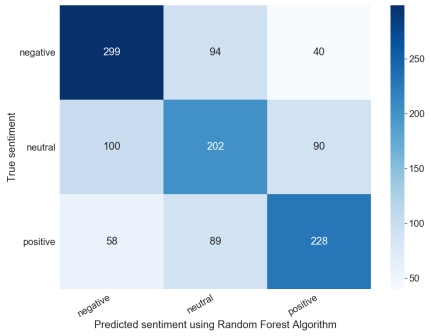
The table 4.1 showcases the ability of each machine learning algorithms to predict positive, negative and neutral labelled sentiments. It can be observed that algorithms such as SVM and Naive Bayes have given similar result of accuracy of 59% and 60% respectively while Random Forest also fared similar with the accuracy of 57%. The KNN classifier could not provide much better accuracy and it was able to just reach an accuracy of 39%.

Algorithm	Sentiment	Precision	Recall	F1-Score	Support	Accuracy
Naïve-Bayes	Negative	0.65	0.65	0.65	433	0.59
	Neutral	0.48	0.61	0.54	392	
	Positive	0.68	0.5	0.57	375	
Random Forest	Negative	0.62	0.67	0.65	433	0.57
	Neutral	0.47	0.44	0.45	392	
	Positive	0.62	0.61	0.61	375	
SVM	Negative	0.66	0.64	0.65	433	0.6
	Neutral	0.5	0.5	0.5	392	
	Positive	0.63	0.65	0.64	375	
KNN	Negative	0.7	0.14	0.23	433	0.39
	Neutral	0.34	0.89	0.5	392	
	Positive	0.63	0.16	0.25	375	

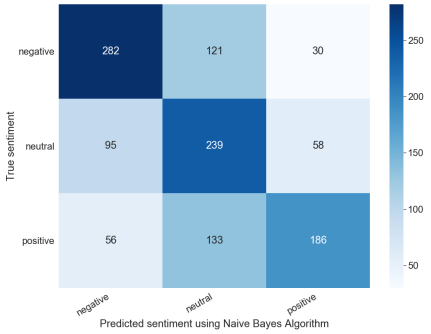
Table 4.1: Classification Report for Machine Learning Algorithms



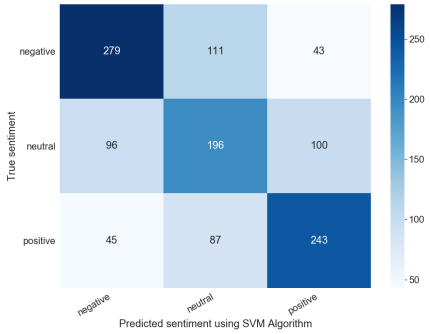
(a) Confusion Matrix - KNN



(b) Confusion Matrix - Random Forest



(c) Confusion Matrix - Naive Bayes



(d) Confusion Matrix - SVM

Figure 4.1: Confusion Matrix of all Machine Learning Algorithms

The table 4.1 is based on the confusion matrix of each Machine Learning algorithm which provides the number of True Negatives, False Negatives, False Positives and True Positives and of each algorithm. It can be observed that the True Negatives and True Positives of the KNN is very less compared to other algorithms, hence it's accuracy is also less. Although the precision is higher for positive and negative reviews, the recall is quite low and the precision for predicting neutral reviews is also less. The Naive Bayes algorithm has predicted the negative sentiments comparatively better than other algorithms with comparatively better recall but the best Machine Learning algorithm to predict the sentiments for our dataset is SVM. SVM has predicted all the three sentiments almost equally well with predicting neutral reviews are slightly on the lower side.

4.1.2 BERT - Results

The results obtained from BERT algorithm is much better than all the Machine Learning algorithms used in the report. The Figure 4.2 shows the history of the accuracy of training and validation datasets over 20 epochs. Although there was a bit difference between train and test accuracy in the early stages of epochs but both the accuracy were almost similar for the later stages of epochs. The test accuracy of the overall model is 82.22% which has been showcased in the Table 4.2.

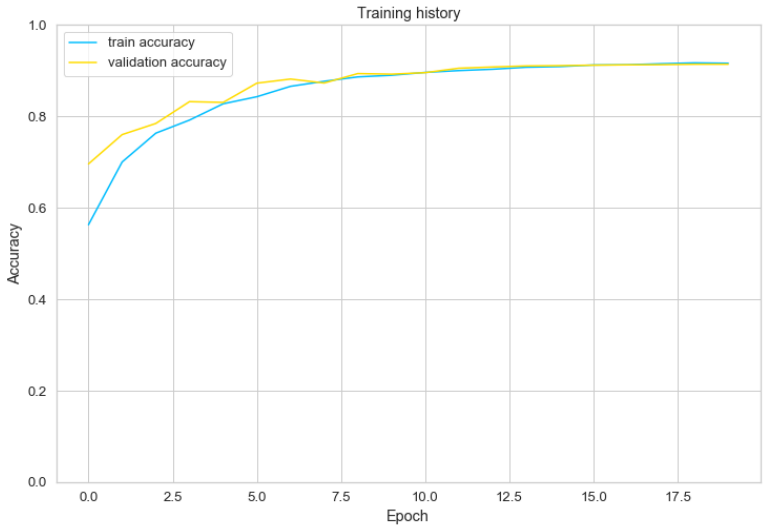


Figure 4.2: Training History for BERT Algorithm

Algorithm	Test Accuracy
BERT	.8222

Table 4.2: Test Accuracy for BERT Algorithm

The confusion matrix of the BERT algorithm for our dataset is shown in Figure 4.3. It can be observed that all the three sentiments are classified correctly with almost equal number of true positives and true negatives with their respective predicted counterparts. Although the accuracy is quite high compared to the Machine Learning models, the time taken to process the same amount of data was quite high.

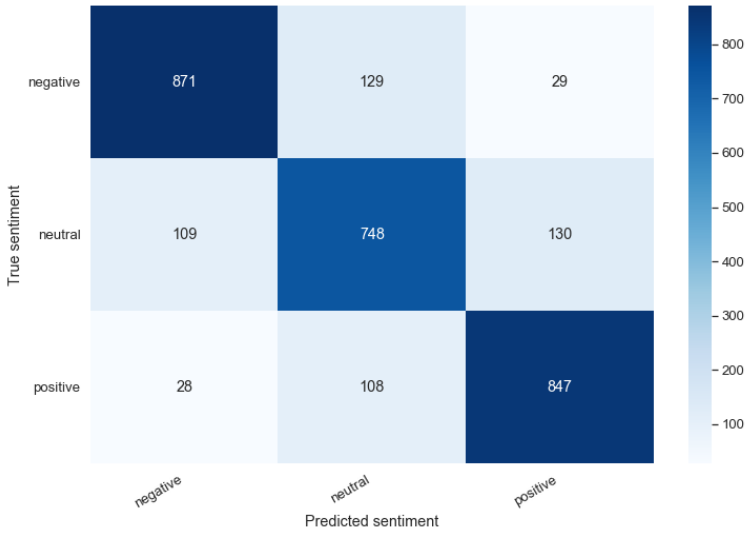


Figure 4.3: Confusion Matrix for BERT Algorithm

Using the BERT algorithm a review text option has been created wherein any statement can be put as the review text and the BERT algorithm provides the sentiment of the statement either positive, neutral or negative. One such example is shown in the Figure 4.4 where a random review of a show has been given as input "Dark that kind of series that keeps you guessing whats coming, with excellent music and outstanding character development" and the sentiment obtained is "Positive" as expected.

4.2 Discussion

While comparing the machine learning and deep learning algorithm (BERT) it was observed that Machine Learning models are dependent on the type of the data, specially data generated from online reviews contain lot of irregularities and is quite unstructured. This requires lot of human effort in pre-processing the data and all unnecessary stop words, contraction words and other special characters of the data to get a better and more precise result through machine learning.

Since BERT performed better than all the algorithms, few examples were checked where random movies and TV show reviews has been provided and the associated sentiments is obtained and using BERT algorithm. The Figure 4.4 shows few examples of the movies reviews and taken online and fed to our model as review

```
input_ids = encoded_review['input_ids'].to(device)
attention_mask = encoded_review['attention_mask'].to(device)
output = model(input_ids, attention_mask)
_, prediction = torch.max(output, dim=1)
print(f'Review text: {review_text}')
print(f'Sentiment  : {class_names[prediction]}')
```

```
Review text: I've had mosquito bites that were more passionate than this undead, unrequited, and altogether unfun pseudo-romant
ic riff on Romeo and Juliet.
Sentiment  : negative
```

(a) Example of Negative Review

```
input_ids = encoded_review['input_ids'].to(device)
attention_mask = encoded_review['attention_mask'].to(device)
output = model(input_ids, attention_mask)
_, prediction = torch.max(output, dim=1)
print(f'Review text: {review_text}')
print(f'Sentiment  : {class_names[prediction]}')
```

```
Review text: This monster of a show did not end how I expected it to, but I can't imagine wanting it to end any other way.
Sentiment  : neutral
```

(b) Example of Neutral Review

```
input_ids = encoded_review['input_ids'].to(device)
attention_mask = encoded_review['attention_mask'].to(device)
output = model(input_ids, attention_mask)
_, prediction = torch.max(output, dim=1)
print(f'Review text: {review_text}')
print(f'Sentiment  : {class_names[prediction]}')
```

```
Review text: Very good movie, great cinematography. People complain about no character development, which is because the film h
as a broader scale. It is a war film with a complex plot, and is a great watch.
Sentiment  : positive
```

(c) Example of Positive Review

Figure 4.4: Review of working of BERT Algorithm

text. Interestingly the model was also able to understand few of the sarcastic comments such as in Figure 4.4 (a) the review is a bit sarcastic and humorous and the model has correctly predicted the sentiment as negative. Similarly in (b) and (c) of Figure 4.4 a bit complex reviews have been fed to test the strength of the model and it accurately predicts the sentiments as neutral and positive respectively.

Chapter 5

Conclusion

The project was aimed to determine the sentiment analysis of Movie reviews. The availability of providing online reviews for movies by viewers on various digital platforms has resulted in huge generation of data and with the impact of analysis of sentiments of these movie reviews, the overall success of a movie can be easily determined. This project focus to realize various tools and methods to make a machine understand sentiment of text reviews. Using various machine learning algorithms, accuracy was obtained and the algorithms were compared with each other based on accuracy, precision scores. The Machine Learning algorithm Naive Bayes and Support Vector Machines provided accuracy of around 60% with thorough text cleaning and pre-processing. The accuracy might have been improved by more stringent cleaning of text to comply with the algorithm and it would require more human effort to achieve the desired result. To overcome this issues of Machine Learning algorithms, a sophisticated and popular Deep Learning Algorithm for natural language processing, named BERT was used. The BERT algorithm provided much better results from all the machine learning algorithms used and it was able to catch the sarcastic comments made on the movie reviews and neutral movie reviews efficiently along with the positive and negative movie reviews. The test accuracy of BERT algorithm is 82.22% and could have reached even higher percentage of accuracy if more powerful GPU is used to train the data with BERT.

Undertaking this project helped in understanding how a machine understands text as input and is able to predict sentiment of the given text based on the training it undergoes with various algorithms. This project also helped to understand the advancement machine learning has reached in natural language process with technologies such as Deep Learning and Neural Networks. Although deep learning algorithm, BERT took more time to process the data as it is bi-directional,

it was able to reach much higher precision in predicting the sentiments of text reviews. Hence, it can be concluded that it is preferable to use Deep Learning algorithm to obtain precise sentiments of movie reviews.

Future Work

A better accuracy and more advanced analysis of sentiments of the text data can be achieved on an efficient computer with powerful GPU capability [Mizell, 2017]. With the help of powerful machine, the analysis can be done on overall reviews of the Amazon data as the data chosen in this report is the subset of the total Amazon DVD reviews. Sentiment Analysis of unsupervised dataset using BERT is another prospect where studies have already begun and an in-depth research should take place [Valipour et al., 2019]. Analysis of sentiments in big data framework is also another prospect of growth for sentiment analysis, although there are quite a few studies about the challenges of sentiment analysis on big data such as speed, volume and variety problems, the volatility and veracity have not been investigated much and more research could result to a brighter prospect for Sentiment Analysis on Big Data [Sharef et al., 2016].

Appendix A

Appendix

The complete programming code for the algorithms described and analysed in this report is available at the GitHub repository: Sentiment Analysis of Movie Reviews

"https://github.com/lakshyagazaresen/Sentiment-Analysis-of-Movie-Reviews"

Bibliography

Qurat Tul Ain, Mubashir Ali, Amna Riaz, Amna Noureen, Muhammad Kamran, Babar Hayat, and A Rehman. Sentiment analysis using deep learning techniques: a review. *Int J Adv Comput Sci Appl*, 8(6):424, 2017.

Yassine Al Amrani, Mohamed Lazaar, and Kamal Eddine El Kadiri. Random forest and support vector machine based hybrid approach to sentiment analysis. *Procedia Computer Science*, 127:511–520, 2018.

Michelle Annett and Grzegorz Kondrak. A comparison of sentiment analysis techniques: Polarizing movie blogs. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 25–35. Springer, 2008.

Palak Baid, Apoorva Gupta, and Neelam Chaplot. Sentiment analysis of movie reviews using machine learning techniques. *International Journal of Computer Applications*, 179(7):45–49, 2017.

Jason Brownlee. *Master Machine Learning Algorithms: discover how they work and implement them from scratch*. Machine Learning Mastery, 2016.

Jason Brownlee. *A Gentle Introduction to k-fold Cross-Validation*, 2018. URL `https://machinelearningmastery.com/k-fold-cross-validation/`.

Nadia FF Da Silva, Eduardo R Hruschka, and Estevam R Hruschka Jr. Tweet sentiment analysis with classifier ensembles. *Decision Support Systems*, 66: 170–179, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Hai Ha Do, PWC Prasad, Angelika Maag, and Abeer Alsadoon. Deep learning for aspect-based sentiment analysis: a comparative review. *Expert Systems with Applications*, 118:272–299, 2019.

Lorenzo Gatti, Marco Guerini, and Marco Turchi. Sentiwords: Deriving a high

precision and high coverage lexicon for sentiment analysis. *IEEE Transactions on Affective Computing*, 7(4):409–421, 2015.

Namrata Godbole, Manja Srinivasaiah, and Steven Skiena. Large-scale sentiment analysis for news and blogs. *Icwsm*, 7(21):219–222, 2007.

Zhang Hailong, Gan Wenyan, and Jiang Bo. Machine learning and lexicon based methods for sentiment classification: A survey. In *2014 11th web information system and application conference*, pages 262–265. IEEE, 2014.

Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, 2004.

Hafsa Jabeen. *Stemming and Lemmatization in Python*, 2018. URL `https://www.datacamp.com/community/tutorials/stemming-lemmatization-python`.

Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

Anna Jurek, Maurice D Mulvenna, and Yaxin Bi. Improved lexicon-based sentiment analysis for social media analytics. *Security Informatics*, 4(1):1–13, 2015.

Anne Kao and Steve R Poteet. *Natural language processing and text mining*. Springer Science & Business Media, 2007.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

Dongjoo Lee, Ok-Ran Jeong, and Sang-goo Lee. Opinion mining of customer feedback data on the web. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 230–235, 2008.

Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Eric Mizell. *Introduction to GPUs for Data Analytics: Advances and Applications for Accelerated Computing*. O'Reilly Media, 2017.

Mohssen Mohammed, Muhammad Badruddin Khan, and Eihab Bashier Mohammed Bashier. *Machine learning: algorithms and applications*. Crc Press, 2016.

Manish Munikar, Sushil Shakya, and Aakash Shrestha. Fine-grained sentiment classification using bert. In *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, volume 1, pages 1–5. IEEE, 2019.

B Narendra, K Uday Sai, G Rajesh, K Hemanth, MV Chaitanya Teja, and K Deva Kumar. Sentiment analysis on movie reviews: a comparative study of machine learning algorithms and open source technologies. *International Journal of Intelligent Systems and Applications*, 8(8):66, 2016.

Sarang Narkhede. *Understanding Confusion Matrix*, 2018. URL `https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62`.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, 2019.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*, 2002.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49, 2016.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

Lina Maria Rojas-Barahona. Deep learning for sentiment analysis. *Language and Linguistics Compass*, 10(12):701–719, 2016.

AK Santra and C Josephine Christy. Genetic algorithm and confusion matrix for document clustering. *International Journal of Computer Science Issues (IJCSI)*, 9 (1):322, 2012.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge, 2008.

Nurfadhlina Mohd Sharef, Harnani Mat Zin, and Samaneh Nadali. Overview

and future opportunities of sentiment analysis approaches for big data. *J. Comput. Sci.*, 12(3):153–168, 2016.

Abhishek Sharma. *Decision Tree vs. Random Forest – Which Algorithm Should you Use?*, 2020. URL `https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/`.

Chance Simmons and Mark A Holliday. A comparison of two popular machine learning frameworks. *Journal of Computing Sciences in Colleges*, 35(4):20–25, 2019.

VK Singh, R Piryani, Ahsan Uddin, and P Waila. Sentiment analysis of movie reviews and blog posts. In *2013 3rd IEEE International Advance Computing Conference (IACC)*, pages 893–898. IEEE, 2013.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2): 267–307, 2011.

Jalaj Thanaki. *Python natural language processing*. Packt Publishing Ltd, 2017.

Bruno Trstenjak, Sasa Mikac, and Dzenana Donko. Knn with tf-idf based framework for text categorization. *Procedia Engineering*, 69:1356–1364, 2014.

Peter D Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *arXiv preprint cs/0212032*, 2002.

Mehrdad Valipour, En-Shiun Annie Lee, Jaime R Jamacaro, and Carolina Bessega. Unsupervised transfer learning via bert neuron selection. *arXiv preprint arXiv:1912.05308*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Ho Chung Wu, Robert Wing Pong Luk, Kam Fai Wong, and Kui Lam Kwok. Interpreting tf-idf term weights as making relevance decisions. *ACM Transactions on Information Systems (TOIS)*, 26(3):1–37, 2008.

Rui Xia, Chengqing Zong, and Shoushan Li. Ensemble of feature sets and classification algorithms for sentiment classification. *Information sciences*, 181 (6):1138–1152, 2011.