Text Analytics of Course Evaluation: A Case Study of AIT

by

Rounak Raj Surana

A research study submitted in partial fulfillment of the requirements for the degree of Master of Engineering in Information Management

Examination Committee:	Dr. Vatcharaporn Esichaikul (Chairperson)
	Prof. Matthew N. Dailey
	Prof. Phan Minh Dung

Nationality:	Indian
Previous Degree:	Bachelor of Technology in Computer Science Engineering
	Jawaharlal Nehru Technological University, Hyderabad
	Telangana, India
Scholarship Donor:	AIT Fellowship

Asian Institute of Technology
School of Engineering and Technology
Thailand
December 2019

Acknowledgments

First of all, I would like to express my heart full of gratitude, respect to my advisor Dr. Vatcharaporn Esichaikul and sincerely thank her for all the guidance, suggestions, support and encouragement which I received starting from selecting the courses in the first semester to selecting my research topic and until the completion of my master's degree program. I further would like to thank Professor Matthew N. Dailey in helping me with receiving the dataset from AIT on my research and for being a part of my committee member and giving me valuable suggestions. I would also like to thank Prof. Phan Minh Dung for being a part of committee member and his subject Natural Language Understanding which gave a bit of more clear understanding of my research interest.

Secondly, I would like to thank the students of ICT and CSIM department because of their evaluation I could get the data set to work on my research study.

Finally, I acknowledge the people who mean a lot to me, my parents for showing faith in me and giving me liberty to choose what I desired. I express my gratitude to all the selfless love, care, pain and sacrifice they did to shape my life. I am also grateful to my friends Prithvi, Siddharth, Manoj, Sri Ram, Abhijeeth, Balakrishna Phani and especially Kokila Panthi who has always encouraged me with a smile and made my stay in AIT memorable.

Sincerly,

Rounak Raj Surana

ST119979

Abstract

A course evaluation is a paper or online survey that includes a response to a number of questions written or chosen to determine the curriculum of a specified course. In academic institutions, course evaluation at the end of each semester is normal, which includes quantitative scores such as rating scale and qualitative such as open-ended questions/text comments which aim to study the sentiment of the students on course/instructor characteristics. The quantitative scores give a brief about review on the course whereas qualitative content gives a more accurate view about the course, helping instructors in improving the quality of the course and learning experience of students. However, going through all the qualitative content manually and finding the sentiment of the students is a tedious task to achieve.

This research study addresses the problem by finding the sentiment of course evaluation in the form of text comments given by students. The implemented method for solving the problem comprises data pre-processing, classification of the sentiment with the use of different classification algorithms such as Support Vector Machine (SVM), Multinomial Naive Bayes (MNB), Decision tree (CART), XGBoost, Stochastic Gradient Boost, testing of the classification model generated. The implemented solution also leverages the visualization of the text comments.

Evaluation of the method and visualization is done on student's feedback comments by the courses taught in ICT department, Asian Institute of Technology helping the instructors in understanding the sentiment of the students. Student feedback comments concentrate on 4 different aspects they are: course characteristics, instructor characteristics, course delivery/teaching methods/resource materials, and overall assessment. Applying the solution to classify the sentiment of qualitative data on two courses with the highest number of comments, SVM works the best in finding the sentiment on course evaluation with an accuracy of 78.2% and visualization of the text comments is done using word cloud helping the instructor in finding the opinion of students.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter consists of the background, problem statement, objectives, limitations, scope and finally the outline chapter of this report.

1.1 Background

Course evaluation is a survey, which includes a written or chosen response to a number of questions to determine the teaching of the course. Course evaluation helps in different ways to students as an important mean of giving their reviews about the course quality, teaching quality and also is an important source of information on the teaching methods and its qualities in the environment of individual classes. Student evaluations of course and instructors are collected in many institutions and they mainly provide with:

- A qualitative and/or quantitative feedback to instructors for improving teaching for future students. Teachers can review how their students interpret their teaching methods, thereby improving their teaching methodology and course material;
- A measure for finding the effectiveness of teaching, that provides information for personnel decisions, like promotions, salary raise, tenure, reappointment, and for making recommendations based on the feedback;
- Information for future upcoming students in selecting the course and instructors;
- A quality assurance exercise for international level and national level, which helps in monitoring the quality of education;
- A process detail on teaching methods such as designing education in such a way that improves the effectiveness of teaching, knowing the effects and relations related to different ways of teaching, and also finding the aspects of former students.

Course evaluation questionnaires usually consist of two parts. A quantitative part, where students can give ratings in the form of multiple-choice questions on aspects such as course characteristics, teaching characteristics, and a qualitative part, where students can write their feedback in words. Multiple choices are effective and one of the fastest ways to evaluate requiring very little effort to be applied. Multiple-choice questions also help in making sure that all the students are being asked the same questions, which makes the comparison of student's responses easier, within the courses, instructors teaching methods, about instructors and also universities. The purpose of student comments is to give individual feedback to instructors on courses, teaching methods, teaching material, or for use in one-to-one evaluation with administrators of the university for personnel decisions.

Course evaluations by students often combine computer-graded multiple-choice items and comments as seen earlier and these multiple-choice items are easy to determine because the answers are quantifiable. Nevertheless, the responses to comments are text information, and objectively understanding the student's general behavior is difficult. In fact, it is difficult to avoid risking arbitrary and incomplete definitions of the data at the time of summarizing the data by the analyst. Therefore, a study of sentiment/text mining methodology could be useful to reduce these threats as much as possible.

Analysis of sentiment is an application of natural language processing, text mining and computational linguistics, to classify text content. Education is one of the fields that has seen a beneficial application of sentiment analysis to boost attended sessions and distance education at the moment. Analyzing of course evaluations is still done manually and mainly focused on quantitative feedback.

1.2 Problem Statement

Course evaluation is where students give their feedback in the form of quantitative and qualitative content that is in the form of ratings and text comments. In the course evaluation form ratings just give a brief scaling whereas text comments or the open-ended questions or the qualitative content consist of the true emotion of the student feedback. Analyzing this qualitative content in course evaluation is useful in improving the quality of courses and other vital factors such as course teaching materials, understanding of the concepts, reach student requirements of interest, etc. Manually going through all the text comments received through course evaluation form of each course is quite a time taking process and understanding student feedback only based on ratings/quantitative data does not help in improving the course in a much better way. This is where text analytics comes in to picture to analyze the comments in the open-ended questions about the course received from the students.

1.3 Objectives

The main objective of this study is to perform text analytics and classify sentiment of the text comments given by students in course evaluation form at the end of every semester. In order to understand student requirements and also classifying the sentiment from course evaluation in the form of text comments, which will help to improve the quality of the evaluated course. Following are the four things required for accomplishing the above stated objective:

1. To visualize the text comments based on the sentiment of the comment and to perform text mining on the received answers on the open-ended questions.
2. To classify text based on the sentiment of the comments by performing text analytics which improves the course quality and also helps the instructor to understand the improvements to be made in the upcoming semesters.

3. To select the best classification model by evaluating performance of the classification models built in terms of accuracy, precision, recall, fscore.
4. To apply the best-identitfied classification model on the top two courses with the highest number of comments combining three different aspects of course evaluation.

1.4 Scope

This research study aims at extracting sentiment of the text comments from the student evaluation form, given by the students of Asian Institute of Technology (AIT) for the courses provided by the university every semester.

1.5 Limitations

The major limitation of this research study is less number of qualitative data/ text comments in course evaluation. The data set aggregates 5 years of text data with 40-60 comments per course in total which leads to difficulty in predicting sentiment with higher accuracy. Consequently making it difficult to find suggestions. Besides the accuracy of the results, clauses with negation cannot be predicted with a precise sentiment class such as a clause ”a student would be stupid taking this course” has a negative sentiment whereas negation of the clause is ”a student would be stupid not taking this course” has a positive sentiment which the models build wouldn’t be able to differentiate accurately.

1.6 Research Outline

I organize the rest of this dissertation as follows.

In Chapter 2, I describe the literature review.

In Chapter 3, I propose my methodology.

In Chapter 4, I present the experimental results.

Finally, in Chapter 5, I conclude my resarch.

Chapter 2

Literature Review

This chapter gives a description about student course evaluation, and some relevant topics which are to be understood for accomplishment of the objectives and literature which would describe the previous work done on this topic.

2.1 Student Course Evaluation

Student course evaluation is one of the most common methods to increase the qualitative education of the course and giving feedback to instructors about their courses. According to Alhija and Fresko (2009) student course evaluation is expanded into three different forms, they are so as follows: First a variety of analytical questions using multiple choice to choose their responses, the second is the open-ended questions or comments which allow students to give their opinions about the course in their own words and the third is the combination of both the methods that is analytical evaluation and open-ended questions. Ultimately evaluations are done using simple survey form where the first section consists of statistical questions with a rating system that includes various aspects as of course or instructor, and the second part deals with the open-ended questions where students are allowed to review in the form of comment/suggestions about the course. Course evaluation is majorly to facilitate institutional decisions regarding the quality, of course, assignments, faculty and also to help the instructor to identify the strength and weaknesses in a desire to improve further.

2.2 Text Mining

Data mining technology helps in extracting useful content/information or patterns from different databases. The mining process also includes collection, extraction, analyzing and statistics of various data which is also known as the knowledge discovery process which could be seen in Figure 2.1. Data mining: concepts and techniques a book by Han et al. (2011) states that data mining could be more appropriately named as ”Knowledge mining from data”, which itself is quite a long name. Data warehousing also has been doing good in managing numerical information, but could not be much success when it came to textual information. Hence, leading to understanding text mining which is a process of extracting content/information from the textual data also known as text data mining or knowledge discovery from text data.

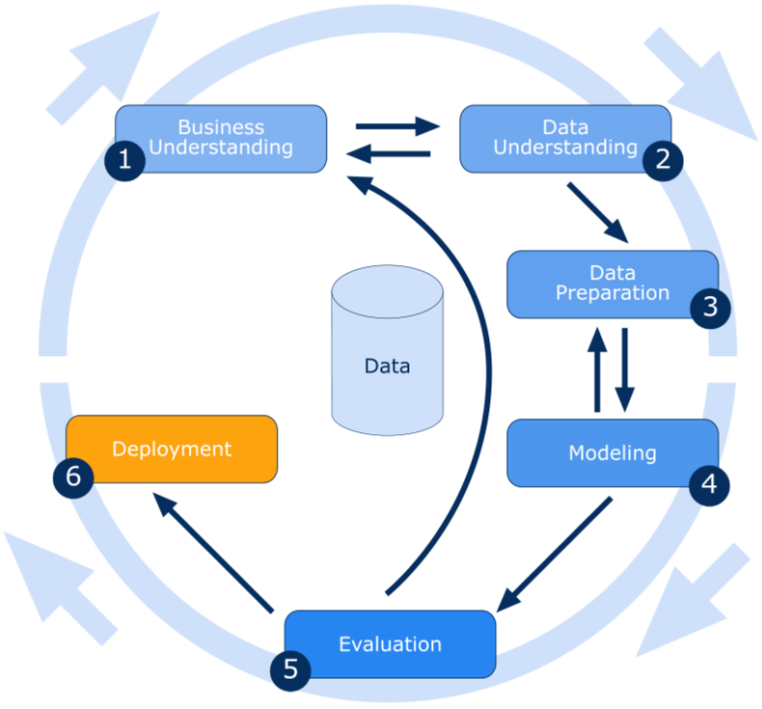


Figure 2.1: Data mining process Reprinted from the work of Petersen (2018)

As already understood text mining is used to extract information from natural text language. It can also be said as the process of text analysis which would be helpful in analyzing and extracting information for a useful purpose. Rouse (2018) described how text mining majorly deals with unstructured data or semi-structured data and as well as text data is ambiguous and difficult to be processed. Text mining it deals with an only text whose function is dealing with the opinions which are passed using text or finding the similar meaning of the text. Text mining was done manually back then which were quite labor-intensive, expensive and more of required too much time with an increase in amount of information. The study of text mining involves various techniques which allow automatic analysis of unstructured information as well as the extraction of data, and to make the text much understandable and searchable to be used. A text document not only consists of characters as they combine to form words and which further combine to generate phrases. text mining helps in not only searching for words but also helps in understanding information to identify patterns in the text data.

The text mining process explained by Gaikwad et al. (2014) notes that the process of text mining starts with report selection from various sources. Text mining software could be used to extract documents and pre-process them by testing the character sets, and the process of text analysis comes after the pre-processing of the documents. Analysis of text is something that extracts from the text a high quality of information. There are many techniques for text analysis that could be used depending on the aim and the purpose of combining different techniques. Figure 2.2 demonstrates the text mining process which is also known as the knowledge discovery of data using text mining.

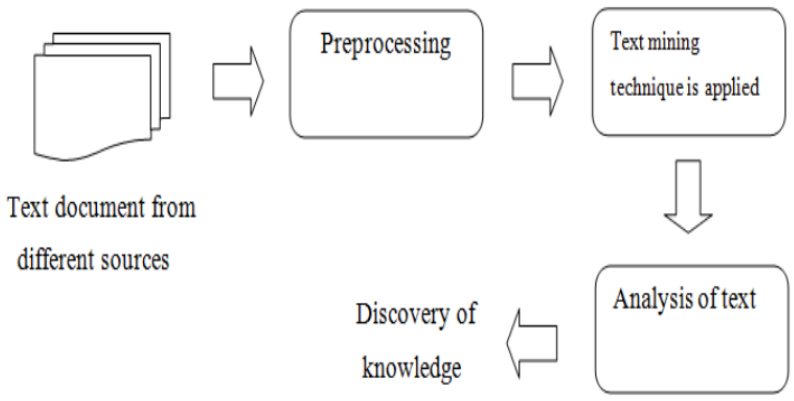

Text document from different sources
Preprocessing
Text mining technique is applied
Analysis of text
Discovery of knowledge

Figure 2.2: Text mining process. Reprinted from the work of Gaikwad et al. (2014)

2.2.1 Text Mining Challenges

Natural language is one of the major challenging issues in text mining as it is not free from the problem of ambiguity/uncertainty. Multiple words with the same meaning and one word with multiple meanings known as ambiguity or describe as an understanding of natural language in two or more possible ways. Uncertainty in the natural language cannot be completely eliminated as it varies with usability. They are many proposed ways in which the problem of uncertainty could be solved still it varies from one domain to another. It is still a challenge to understand what the user wants as they are many linguistic meanings that are uncertain (Dang & Ahmad, 2015). Some Merits and Demerits of Text Mining One of the merits would be finding the relationship between different entities could be found from the different corpus using text mining techniques as of information extraction, Another merit managing of the unstructured information for extracting patterns from the text data. If they are merits they are demerits to as of no proper programs could be made to analyze the unstructured data as it varies with the domain of the data collected.

2.3 Techniques used in Text Mining

Gaikwad et al. (2014) and Kaushik and Naithani (2016) in their work explain different techniques on how to teach a computer to analyze, to understand and to generate text. These techniques which are been talking about include information extraction, clustering, categorization, summarization, information visualization, topic tracking, questioning and answering and finally sentiment analysis which is also used in the text mining process. In the further sections, the above-mentioned techniques and roles played in the process of text mining are discussed.

2.3.1 Information Extraction

Information extraction is the first phase for the computer to evaluate unstructured text by defining important sentences and interactions in text. The process for extracting information involves tokenization, defining named individuals and segmentation of the phrases and assigning part-of-speech. The search process for predefined sequences of the text is used in this task process. Beginning with phrases and sentences, the required pieces of information entered in the database are evaluated and interpreted semantically. Many of these moves have been accompanied by significant progress in the application of data mining techniques, and the most exact systems of knowledge extraction include hand-crafted language processing. This technique of text mining can be very helpful with a large volume of text. For many uses, electronic information is more complex than organized data points, such as relational databases in the shape of free natural language documents. Hence, Information extraction addresses the question of converting a series of textual documents into a more organized database.

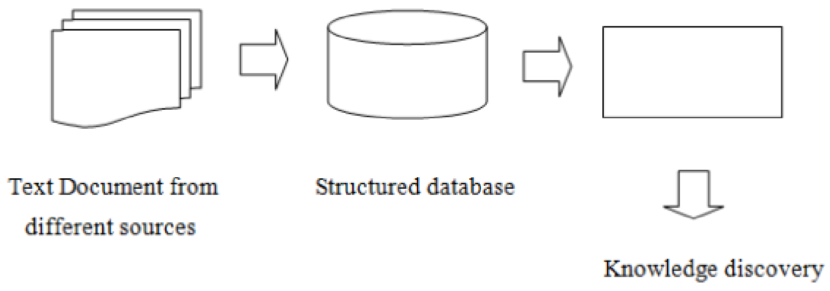


Figure 2.3: Information Extraction. Reprinted from the work of Gaikwad et al. (2014)

2.3.2 Categorization

Categorization assigns a free text document to one or more categories automatically. Categorization is a supervised learning method because the classification of new documents is dependant on the sample. The text documents are assigned predefined classes based on their content. The typical method of categorizing text comprises pre-processing, dimensional reduction, indexing, and classification. The categorization objective is to train a classifier based on known examples and then automatically categorize unknown examples. Techniques for statistical classification such as Support Vector Machines, Decision Tree, Nearest Neighbor classification and Naïve Bayesian classification are mostly used algorithms to categorize text.

2.3.3 Clustering

The clustering technique of text mining can be used to find classes of similar content in documents. The clustering result is usually a partition called clusters P and each cluster is made up of a number of documents d. The content in the documents inside a cluster is clearly comparable and the quality of clustering is deemed to be better between the clusters. Although the clustering process for the grouping of comparable data differs from categorization since the clustering of flying records does not use predefined topics. As documents can possibly occur in the collection of different subjects, it ensures that no useful document is missed by the search results. Considering the data mining approach K-means algorithm is the most frequently used clustering algorithm, it also obtains good results when it comes to text mining. For each document, a fundamental clustering algorithm generates a vector of subjects and also measuring the weights of how well each cluster fits the document. Clustering technology is used in the organization of management information systems as the organizational database contains thousands of documents.

2.3.4 Visualization

Text visualization methods help in improving and simplifying important information. Document categorization and color density are commonly used to depict individual records or groups of records. Visual text mining brings in a visual hierarchy for big textual sources. The user can use zooming and scaling to communicate with the document. Visualization of data applies to govt in order to recognize terrorist networks or to discover crime data. Figure 2.4 will help in understanding the steps which are involved in the visualization process. Information Visualization can be branched into 3 major stages they are as follows: First, Data preparation involves determining and obtaining the original visualization data and forming the original data space. Second, Data analyzes and extractions are defined as the process of evaluating and extracting the necessary visualization data from the original data and creating a visualization data space. Third, the mapping stage of visualizations uses various data/info mapping algorithms to visualize the target.

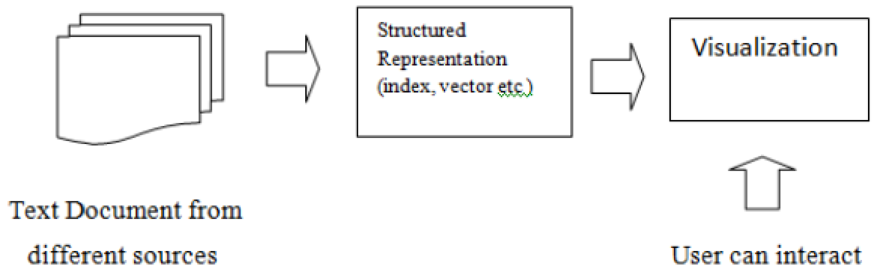


Figure 2.4: Visualization. Reprinted from the work of Gaikwad et al. (2014)

2.3.5 Summarization

The text summary is intended to decrease a document's length and detail while maintaining the most significant points and overall significance. The text summary is useful to see if a long document meets and respects the user requirements for additional data. So, summarization can substitute the collection of documents. The user takes the moment to come across and complete reading the first paragraph, in between this time gap text summarization software does help in summarizing the big text file. While machines can recognize individuals, locations, and time, it is hard to train software to describe the significance of the text documents (Dang & Ahmad, 2015). Text Summarization process includes the following steps: Starting with first, Pre-processing which as earlier seen obtaining an ordered representation of the original text/information. Secondly, To convert ordered summary from that of the text structure as seen in the earlier steps using different algorithms. and Third, obtaining the final summary from the so formed summary structure in the previous step.

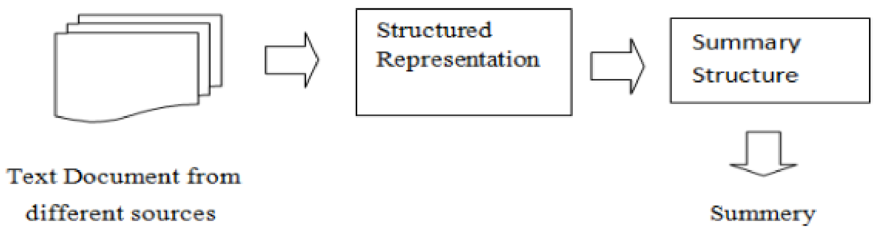

Text Document from different sources
Structured Representation
Summary Structure
Summery

Figure 2.5: Summarization. Reprinted from the work of Gaikwad et al. (2014)

2.3.6 Questioning and Answering

Natural language queries or answering questions are responsible for deciding how to find a more appropriate answer to a specific question (Kaushik & Naithani, 2016). For instance, if the user asks Google any questions or searches for something, then Google provides him the best-matched responses or connections to that question by looking in the database for keywords of the issue. This technology can use more than one mining methods at a moment to extract entities and assign the question respectively using IE and question categorization. Question and Answer can be used in multiple web applications, in medical fields as well as education.

2.4 Sentiment Analysis

Sentiment analysis, also known as opinion mining, is configured from the emotion of the user, mostly into several positive, negative, neutral and mixed classes. It is primarily used to obtain the view or attitude of people towards anything that includes services and products. For example, the website of Amazon provides space for user comments on all the products they sell. Through this people can express their attitude towards any particular item that is again helpful to other buyers as they can read reviews from previous buyers. From the perspective of the company (Amazon), it helps them improve the quality of their product by making necessary modifications in it. With the increasing popularity and reach of social networking sites, any organization can now obtain a large amount of data (reviews) related to their products. This allows them to analyze customers ’ real-time opinions very rapidly. So lastly Sentiment Analysis is a method of automatically extracting characteristics by mode of other people’s concepts about particular products, services or experiences that may be more efficient with other analytic/mining methods.

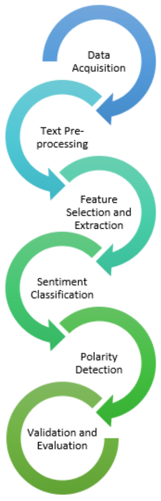


Figure 2.6: Sentiment Analysis Process. Reprinted from the work of Devopedia (2019)

2.4.1 Types of Sentiment Analysis

Sentiment Analysis does not only focus on positive, negative, neutral comments they also focus on feelings detection, emotions identification, or intentions of the ones who give comments. According to Monkeylearn (2019) and the references followed state some of the most general types of sentiment analysis:

- **Standard Sentiment Analysis** The complexity of a viewpoint is described and graded as positive, neutral or negative. For instance:
  - ’Oneplus has bought very good specification in a very low budget.’ – Positive
  - ’Examining of more Oneplus devices to say whther it satisfies my requirements of having a smart phone’ – Neutral
  - ’Oneplus is not a good model when it comes to looks’ – Negative

- **Fine-grained Sentiment Analysis** This kind of sentiment analysis, is often based on polarity, incorporates a few additional classifications in order to achieve more granular results. It classifies views like 5-star ratings as:
  Positive, Neutral, Negative, Very positive, Very negative. For instance:
  - 'The earlier product by this company had a better life ' – Negative
  - 'Worst experience. i not at all choose to own any item from this company again ' – Very Negative
  - 'I neither like this product nor dislike this product' – Neutral
- **Emotion Detection** This model of sentiment analysis helps in detecting the feelings underlying a text. It allows connections such as anger, happiness, frustration, etc. between words and feelings. For instance:
  - 'Alexa makes my learning easier :)' – Happiness
  - 'Customer service by this company is a torture! Completely unworthy!!' – Anger
- **Aspect-based Sentiment Analysis** This sort of sentiment analysis target on understanding elements or characteristics mentioned in a specified view. For instance, product reviews often consist of distinct views about distinct product features, such as Price, UX-UI, Integrations, Mobile Version, etc.
- **Intent Detection** This sort of sentiment analysis attempts to discover an activity going on behind a specified view, which the user wishes to do. Finding client ideas enables you to identify useful possibilities to assist clients, such as solution to a problem, improving a product, or related complaints:
  - 'Very frustrated. Snapchat keeps closing when I log in and more of i dont get any notifications. Can you help?' – Request for Assistance

2.5 Natural Language Processing

Natural Language Processing, usually shortened as NLP, is a branch of artificial intelligence that shows the interaction between computers and humans using natural language. The ultimate objective of NLP is to read, decipher, understand, and make sense of the human languages in a manner that is valuable. In simpler words NLP is making machines understand human language such as unstructured text, using different algorithms and techniques. Kao and Poteet (2007) has described how NLP helps in the classification of text, clustering and as well as in sentiment analysis/opinion mining. For such processing of text data pre-processing text such as tokenization, normalization (stemming and lemmatization), stopword removal is required. Further in this section, pre-processing techniques is been explained.

- **Tokenization**: Tokenization comprises of separation of text in smaller units during the time of the pre-processing data set. Text can be tokenized in single words, sentences, phrases, paragraphs.

- **Stemming**: Stemming algorithms job is cutting off the start or end of the phrase, taking into consideration a list of prevalent prefixes and suffixes found in an inflected phrase. On some occasions, this indiscriminate cutting can be effective. and that is why this strategy poses some constraints.
  Example: studies – studi, studying – study
- **Lemmatization**: Lemmatization requires the morphological analysis of the phrases into account. To do this, it is necessary to have detailed dictionaries that can be looked through by the algorithm to link the form back to its lemma.
  Example: Studies, Studying, – Study
- **Stopword removal**: Stopwords are common words used in the English language such as ”a”, ”the,”, ”their,” which certainly have no effect on the semantics of the review. To reduce noise, they can be removed. Details like ”bug” or ”add,” which could improve the reliability of the information identification, becomes more important in increasing the accuracy of the classifier. Despite, few keywords often recognized as stopwords may be essential for assessment classification.
  Example: the words ”did,” ”while,” ”because” a detail explanation, ”but” as a note and ”very,” ”too,” ”up” and ”down” as valuation.
- **POS tagging**: POS tagging aims at assigning parts of speech to the word after reading the words in the text. It is also known as grammatical tagging. According to Wikipedia, it is a method of marking a term in a text corpus on the basis of text and context as the corresponding part of speech, its relation with the close by words and related words in the sentence or paragraph. A simpler version would have nouns, verbs, pronouns, adjectives, adverbs, etc.
- **Bag of words**: After segmentation of the text into sentences, segmentation of each sentence into words, they are been tokenized and normalized, creating a straightforward bag-of-words model of the text. Considering individual words in this depiction of bag-of-words and give each word a specific score of subjectivity. This subjectivity score can be regarded in a sentimental lexicon. If the full score is negative, the text is classified as adverse and if it is positive, the text is classified as useful.
- **Tf-Idf**: Tf-idf is a abbreviation of Term frequency-inverse document frequency. Term Frequency (Tf) can be defined as the number of times a word has appeared in a document. IDF (Inverse document frequency) is defined as the measure of how important that term is in the whole collection of documents. The tf-idf is the product of these two frequencies. The higher the numerical weight value, the rarer the term and vice versa with smaller the weight, the more common the term.

$$TF = \frac{Number\ of\ time\ the\ word\ occurs\ in\ the\ text}{Total\ number\ of\ words\ in\ text} \qquad \text{(Equation 2.1)}$$

$$IDF = \frac{Total\ number\ of\ documents}{Number\ of\ documents\ with\ word\ w\ in\ it} \qquad \text{(Equation 2.2)}$$

$$TF - IDF = TF * IDF \qquad \text{(Equation 2.3)}$$

2.6 Classification techniques

In this section, different classification methods that can assist in classifying the type of comment are included. One of the major techniques of Text Mining is classifying/categorization of documents into pre-defined classes seen in Section 2.3. Training a machine is considered to be a process of supervised learning where training is carried out using an example of a collection of documents with pre-defined categories labeled. A model build is used to check/analyze a fresh document and to assess the category in which it falls. Another strategy for the classification of documents is an unsupervised teaching technique such as Clustering explained in 2.3. Text classification can be done based on different techniques such as Rule-based technique, Machine learning-based technique, and Hybrid systems (*Text Classification*, 2019).

2.6.1 Decision Tree Classifier

The decision tree can be defined as a flowchart having a tree-like structure consisting of many internal nodes also known as features/attributes and each branch of the tree representing the decision rules, further each leaf node of the tree representing the possible outcome. The topmost node is known as the root node and gives the best prediction result. In a book by Aggarwal and Zhai (2012) has stated that Decision tree classification basically is a hierarchical decomposition of the training data, in which an attribute value predicate or condition is used for hierarchical division of the data.

Classification and Regression Trees (CART) is a classification algorithm that uses past data in the construction of the decision tree. The methodology of CART was developed in the '80s by Breiman (2017) in their book ”Classification and Regression Trees”. In order to build a decision, tree CART uses learning historical data with pre-defined classes on all the tuples of the data.

2.6.2 Naive Bayes

Naive Bayes classification algorithm is a type of supervised machine learning algorithm which is completely based on the Bayes theorem having a ”naive” supposition of every feature having its own individuality (Zhang, 2004). Figure 2.7 showing that each attribute node has only one parent class node. Naive Bayes is one of the simple yet efficient models, measures the probability conditions of two events according to the probabilities of each occurrence. It means that any vector which is in the form of text has to include information on the likelihood of text phrases occurring in a given category so that the algorithm is able to determine the probability of belonging to the text class.

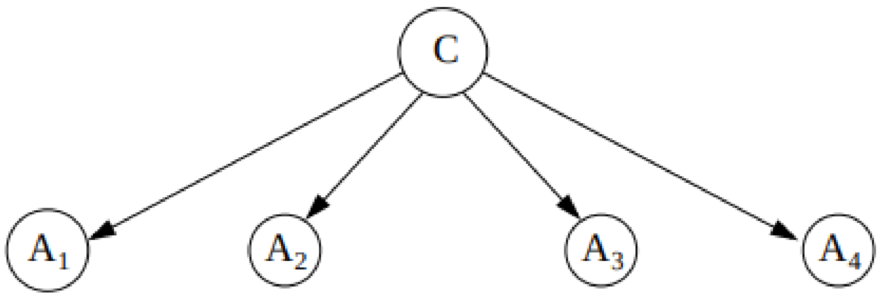


Figure 2.7: Naive Bayes Example. Reprinted from the work of (Zhang, 2004)

Naive Bayes text classification a study by McCallum et al. (1998) compared 2 different first order probabilistic models for classification depending on Naive Bayes they are:

- Multinomial Naive Bayes: This type of Naive Bayes model solves the problem of document classification such as the category of the document i.e sports, technology, etc. The features used by the model are the word frequency present in the whole document.
- Bernoulli Naive Bayes: This type of model is similar to that of Multinomial Naive Bayes but the predictions are boolean variables. The parameters which are taken for prediction are only yes or no values. The main advantage of Bernoulli Naive Bayes is that no dependencies between binary word features and words and are good for small vocabulary sizes.

2.6.3 Support vector machine

Support Vector Machines (SVM) is a supervised learning method for classification and regression problems. Similar to Naive Bayes, although requires more computing capability than Naive Bayes to reach more precise outcomes.

Advantages of support vector machine as a classifier is its effectiveness in higher dimensions, use of a subset of training called support vectors and memory efficient, versatile due to different kernel functions provided (Scikitlearn, 2019). In brief, Figure 2.8 shows SVM is responsible for creating a ”line” or ”hyperplane” dividing room into two different sub-fields: one sub-field containing vectors belonging to a class and other sub-field containing vectors not belonging to the previous class. Support Vector Machine algorithm is implemented in Python and the library used is Scikit learn. It is built on top of Numpy & SciPy library and is an open-source library available to everyone. It becomes very simple to implement the SVM algorithm using Sci-kit learn library. Their library also provides a simple predict function to predict the values. It predicts the probability of a value between zero and one and also helps in classifying models more than two classes.

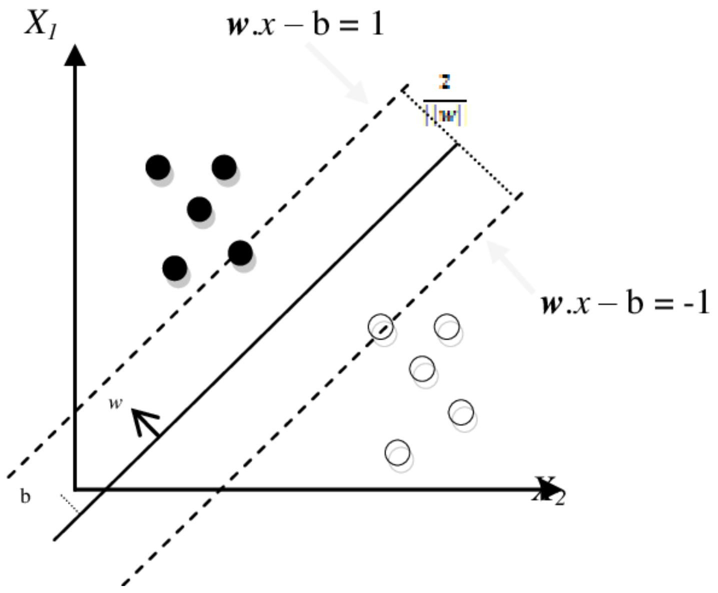


Figure 2.8: Hyperplane between two classes. Reprinted from the work of (Scikitlearn, 2019)

2.6.4 Stochastic Gradient Descent

SGD algorithm is one of the descents and efficient approach of studying linear classifiers under convex loss functions such as SVM. Gradient descent is an iterative function that starts at a random point on the error curve or the loss curve moving down until it reaches the lowest possible point of the function. Stochastic in natural terms is ’random’. Stochastic Gradient Descent is majorly helpful in solving text classification problems with much better results.

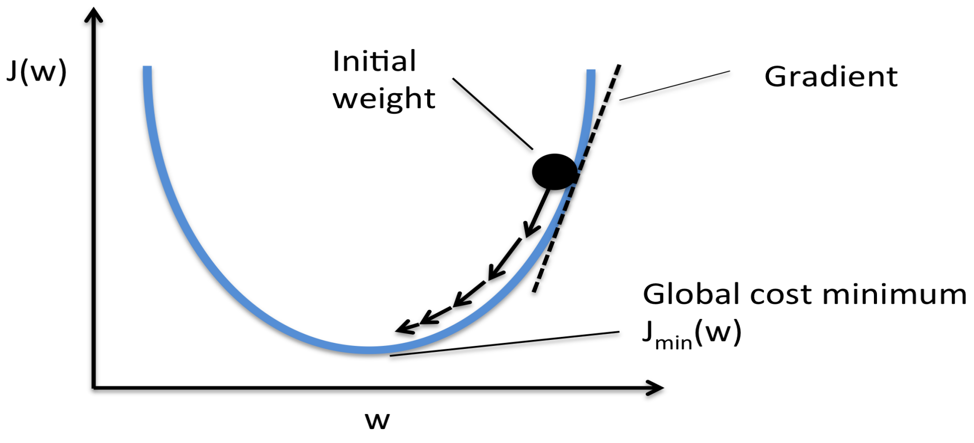


Figure 2.9: Stochastic Gradient Descent Optimization.

2.6.5 XGBoost

XGBoost is a short form for eXtreme Gradient Boosting which uses gradient boosting as a framework and is a decision tree-based machine learning algorithm. XGBoost is used for both regression and classification as of decision tree algorithms being a boosting algorithm solving all prediction problems which also involves unstructured data. XGBoost has got many different features as it can do parallel computation automatically making it faster than other tree-based algorithms. It can take different types of input but cannot specifically differentiate categorial data so preprocessing is required before giving input, customization of the objective is also supported. XGBoost has also got better performance on many different types of data compared (Chen et al., 2015).

2.7 Visualization

Visualization is any technique for creating a graphical representation, diagrams, or animations to communicate a message. Visualization through visual imagery has been an effective way to communicate both abstract and concrete ideas since the dawn of humanity (Hansen & Johnson, 2011).

The simplest and most common form of text visualization is a tag (or word) cloud. They depict tags arranged in space varied in size, color, and position based on tag frequency, categorization, or significance. (Mueller, 2019). In Figure 2.10 shows different color and position which are arbitrary but font size is varied based on word frequency. Even counting is complicated, more specifically these words vary in size based on the total word frequency as opposed to the unique word frequency (words count once per document). Example in Figure 2.10 for course evaluation shows total count and unique frequencies of words within a dataset. Text Normalization, tokenization, Part-of-Speech (POS) tagging, and the removal of stop words, punctuation, etc. give a much clear visualization. This type of cleanup is often necessary prior to analysis.

Figure 2.10: Example word cloud.

2.8 Tools for Pre-processing, Classification and Visualization

Python is the key programming language that is quite simple and robust to use in further sub-sections the libraries of python such as Scikit learn, Natural Language Toolkit (NLTK), Matplotlib and Word cloud for pre-processing, classification and as well as visualization of the comments.

2.8.1 Python Programming Language

Python is a simple, yet powerful interpreted programming language that bridges the gap between C and shell programming (Rossum, 1995). Python version 3.6 is a most stable, mature, versatile and robust programming language. It is an interpreted language which makes the testing and debugging extremely quickly as there is no compilation step. There are extensive open source libraries available for this version of python. Python is a simple yet powerful, interpreted and dynamic programming language, which is well known for its functionality of processing natural language data, i.e. spoken English using NLTK. Other high-level programming languages such as ‘R’ and ‘Matlab’ were considered because they have many benefits such as ease of use but they do not offer the same flexibility and freedom that Python can deliver.

2.8.2 Scikit Learn

Scikit learn is one of the simplest and efficient libraries of python for data mining and data analysis. An open-source library making it accessible to everyone, and also reusable for different contexts (Pedregosa et al., 2011). Scikit learn library has an inbuilt wide variety of machine learning algorithms including supervised and unsupervised learning. It also has different performance matric which compares, validates and also helps in choosing the best model for the dataset inserted (JV Bossche et al., 2019). Classifiers used in this paper such as Support Vector Machine, Naive Bayes, Decision Tree, Stochastic Gradient Descent and also the classification report used to find the validation results of the performance matric is taken from the skikit library of python.

2.8.3 Natural Language Toolkit (NLTK)

Natural Language Toolkit (NLTK) is a library in Python, which provides a base for building programs and classification of text data. NLTK is a collection of resources for Python that can be used for text processing, classification, tagging, and tokenization. This toolbox plays a key role in transforming the text data in the tweets into a format that can be used to extract sentiment from them. NLTK provides various functions that are used in pre-processing of data so that data available from twitter become fit for mining and extracting features. NLTK support various machine learning algorithms which are used for training classifier and to calculate the accuracy of the different classifier. Python being based programming language that is used for writing code snippets. NLTK is a library of Python which plays a very important role in converting natural language text to machine understandable language. NLTK also provides different sets of data that are used for training classifiers. These data sets are structured and stored in the library of NLTK, which can be accessed easily using Python.

2.8.4 Matplotlib

Matplotlib is a 2-Dimensional plotting library for Python programming language. Matplotlib is built on NumPy arrays and helps in makes plotting of various types of figures very easy. Matplotlib can be used to generate bar charts, plots, scatter plots, error charts, histograms, power spectra, etc with just a few lines of code. Matplotlib produces high-quality figures that can be used anywhere (Tosi, 2009). Figure 2.11 shows an example bar graph showing the polarity of customer reviews.

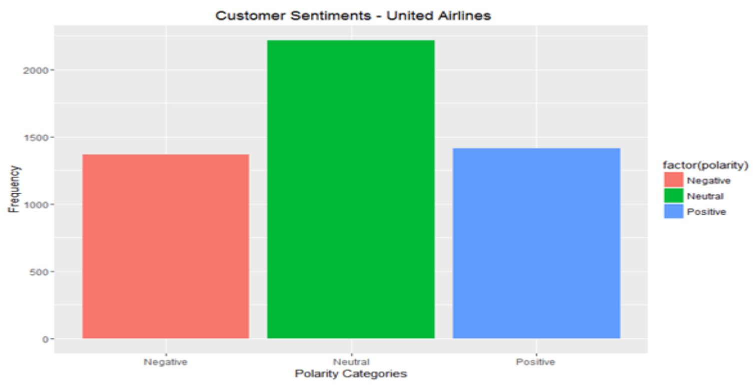


Figure 2.11: Bar Graph showing the polarity.

2.9 Evaluation Matrices of the Models

Evaluation Metric / Performance Metric helps by playing a key role in attaining the efficient classifier or the best classification model. (Hossin & Sulaiman, 2015). To find out the effectiveness of the classification model, the evaluation metric is found with the help of test data sets. Among different evaluation/performance metrics such as Accuracy, MSE, Area under Curve(AUC), etc. The most commonly used metric for evaluation of classification algorithms is confusion matric helping in finding recall, precision, F-score which help in finding the best build algorithm (Sunasra, 2017).

2.9.1 Confusion Matrix

The Confusion matrix is one among the different evaluation metrics, the most intuitive and one of the easiest metrics used for finding the accuracy of the built model. It can be used for classification problems that have output classes to be of more than two types. Example: Solving a problem of classification where a result predicting, about a comment given is a suggestion.
Labelling of the target variables: 1: A text comment is a suggestion 0: A text comment is not a suggestion.

	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True positive (*tp*)	False negative (*fn*)
Predicted Negative Class	False positive (*fp*)	True negative (*tn*)

Figure 2.12: Confusion Matrix Reprinted from Hossin and Sulaiman (2015)

Associated terms with confusion matrix to be known:

- **True Positives (TP):** True positive term defines that the predicted label and the actual classified label are 1 (True).
  Example: A comment taken is a suggestion(1) and the classified model predicts result to be a suggestion(1) comes under True positive.
- **True Negatives (TN):** True negative term defines that the actual class and the predicted class of the data point are 0(False).
  Example: A comment not being a suggestion and the the classified model predicts result as not suggestion comes under True Negatives.
- **False Positives (FP):** False positive term defines that the actual class of the data point is 0(False) and the predicted is 1(True).
  Ex: A comment not being a suggestion and the the classified model predicts result to be a suggestion comes under False Positives.
- **False Negatives (FN):** False negative term defines that the actual class of the data point is 1(True) and the predicted is 0(False).
  Ex: A comment being a suggestion and the the classified model predicts result to be a not a suggestion comes under False Negatives.

2.9.2 Accuracy

Accuracy in classification problem is described as the number of correct predictions made (True Positive (TP)+ True Negative (TN)) by the models over all predictions (True Positive (TP)+ True Negative (TN) + False Positive (FP) + False Negative (FN)). Mathematical representation on how to calculate accuracy can be seen in Equation 2.1.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \qquad \text{(Equation 2.4)}$$

2.9.3 Precision

Precision is a measure that represent the proportion of correctly predicted labels (True Positive) from the totally predicted labels (True Positive + False Positive). Equation 2.2 shows the mathematical representation for calculating precision.

$$Precision = \frac{TP}{TP + FP} \qquad \text{(Equation 2.5)}$$

2.9.4 Recall / Sensitivity

Recall is a measure that tells what proportion of labels are actually correct (True Positive) to that of the labels classified (True Positive + True Negative) by the model. Equation 2.3 shows the mathematical representation for calculating recall.

$$Recall/Sensitivity = \frac{TP}{TP + TN} \qquad \text{(Equation 2.6)}$$

2.9.5 F-score / F-measure / F1-score

F1 Score take in consideration both the above subsection precision and recall. It is the harmonic mean(average) of precision and recall. F1 Score is a better evaluation measure if there is some balance between precision and recall in the system. Equation 2.4 below shows the mean formula of calculating F1 score using precision and recall.

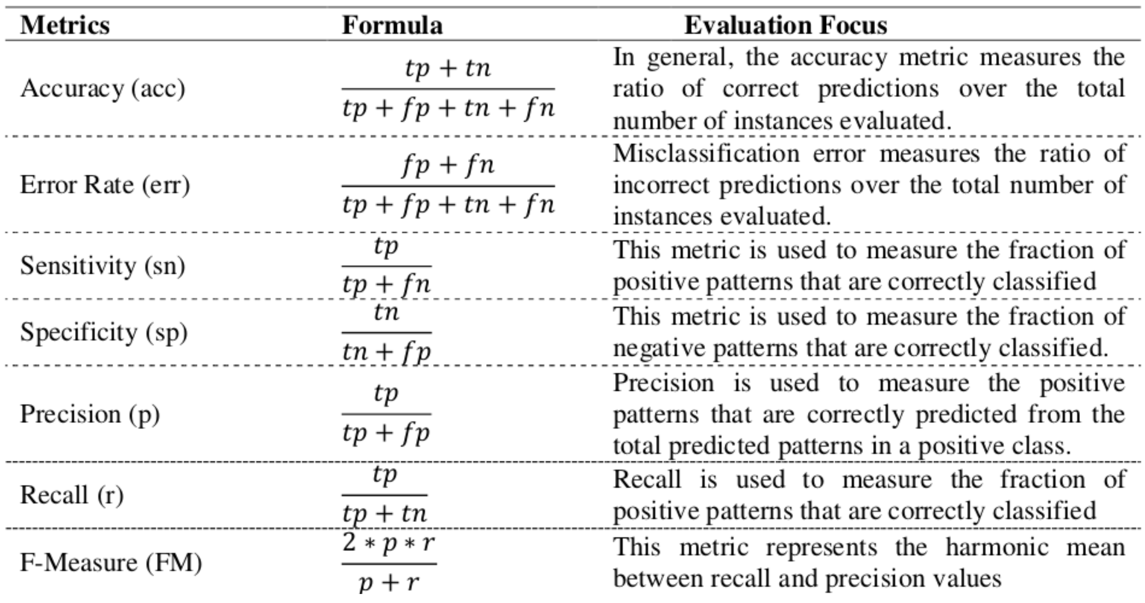
$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \qquad \text{(Equation 2.7)}$$

2.9.6 Specificity

Specificity is the correctly negative labeled by the model (True Negative) to all those who are positive in reality T. Specificity is exact (True Negative + False Positive) opposite to recall. Below Equation 2.5 shows the formula which is used to calculate specificity of the model.

$$Specificity = \frac{TN}{TN + FP} \qquad \text{(Equation 2.8)}$$

Following table below shows comparison of all the metrics with their formula and their focus of evaluation which could be used to evaluate the model which are so generated to classify the comments.



Metrics	Formula	Evaluation Focus
Accuracy (acc)	$\frac{tp + tn}{tp + fp + tn + fn}$	In general, the accuracy metric measures the ratio of correct predictions over the total number of instances evaluated.
Error Rate (err)	$\frac{fp + fn}{tp + fp + tn + fn}$	Misclassification error measures the ratio of incorrect predictions over the total number of instances evaluated.
Sensitivity (sn)	$\frac{tp}{tp + fn}$	This metric is used to measure the fraction of positive patterns that are correctly classified
Specificity (sp)	$\frac{tn}{tn + fp}$	This metric is used to measure the fraction of negative patterns that are correctly classified.
Precision (p)	$\frac{tp}{tp + fp}$	Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class.
Recall (r)	$\frac{tp}{tp + tn}$	Recall is used to measure the fraction of positive patterns that are correctly classified
F-Measure (FM)	$\frac{2 * p * r}{p + r}$	This metric represents the harmonic mean between recall and precision values

Figure 2.13: Evaluation measures for classification Reprinted from Hossin and Sulaiman (2015)

2.10 Related work

Analyzing of the open-ended comments is a very problematic work as comments do not have a proper structure. Open-ended questions also have very fewer response rates compared to that of the quantitative questions and more of comments also have outliers such as ”no comments”, ”nothing”, etc. which are not helpful. Most commonly asked open-ended questions are:

- Provide Comments on Course Characteristics
- Provide Comments on Instructor Characteristics

Open-ended questions also allow students to focus on what is important to them. The relationship between the qualitative data and the open-ended questions is one of the works done by (Sliusarenko et al., 2013). A brief description of similar research works is given below as of analyzing open-ended comments such as finding the sentiment of the comments whether being a positive or negative comment or finding suggestions or building a more accurate classification model that classifies comments.

Clemmensen et al. (2013) in their study analyzed the positive written feedback on a well-established course in two subsequent years. The study tries to apply two clustering methods: k-means and SVD, to cluster the open-ended feedback. The study illustrates that many precise points of student satisfaction are reflected in student comments. Even for a well established introductory course, changes in teaching method can change the basis vectors for clustering comments. Moreover, the study also illustrates problems, that arise when simple text-mining tools are applied to such short texts as students open-ended feedback.

The study by Sliusarenko et al. (2013) on text mining in student's course evaluations analyzed the relationship between the quantitative sores and the open-ended questions on a course from the Technical University of Denmark (DTU). A tool for extracting key-phrases which helps in finding the major topic of students comments, based on which the answers to the open-ended questions were transformed into quantitative data for statistical analysis. Application of such analysis helped in removing outlier comments such as 'nothing', 'nil', etc. and to find the key issues and order of the hidden data in the student text comments, and impact on how students were rating the course with the help of regression analysis. The data set for this study was taken from a mathematics course for 5 years from 2007-2012 enough for text analyzing.

Gottipati et al. (2018) proposed a model that extracts information from the text given in student feedback for suggestions in the course. According to their study, a suggestion could be explicit and implicit and those suggestions provide useful information on helping the instructor in improving the learning experience of students. Manually going through all the text comments/ qualitative data and extracting suggestions is an endless task. This study provided a solution for extracting the suggestions explicitly and implicitly from the comments using existing text mining techniques and also visualizing the comments. Data used for the study was collected from 7 undergraduate core courses from Singapore Management University and as these courses have 2 terms in a year, in total comments collected were 5342 comments and for a random test 399 comments were used. Their study has compared rule-based methods and statistical classification methods for extracting the suggestions and among those experiments, decision tree (C5.0) been working the best in classifying suggestions.

According to Takamatsu et al. (2018) study on text mining of student course evaluation is to provide a proper understanding of student's satisfaction with the academic courses taught in all the universities of Japan. Computer-based course evaluation technique does have a combination of multiple-choice questions and as well as open-ended questions, which consist of responses which are text data. To understand this textual data they used text mining techniques. In the study, they have applied two different methods of text analyzing they are correlational approach and dictionary-based approach, wherein correlational approach classifies words and documents of multivariate analysis and in dictionary-based approach words and documents on the coding standards. The data collected is from a course named Academic skills and Deep learning I with 19 teachers and 350 students and among which 119 comments where obtained in 2017 and 138 comments obtained in 2018 respectively.

Ullah (2016) in his study stated that student feedbacks are collected at the end of the semester with the use of survey forms which is quite a slow use of survey forms and with the use of social media, such as facebook, twitter the collections became much easier. in his paper those feedbacks collected from social media websites has been analyzed to enhance the performance of instructor and the course for which they are many machine learning techniques have been used, Support vector machine SVM, Maximum Entropy ME, Naive Bayes NB and Complement Naive Bayes are the ones used and among the algorithms so used SVM gives the best result that is higher performance with different preprocessing and features extracting techniques which outperform the state of work. Data set so used is from social media and from the University of Proutsmouth with 1036 feedbacks and 641 to be positive, 292 being negative and the rest being neutral.

Altrabsheh et al. (2014) they stated that ”Knowledge about users sentiments can be used for a variety of adaptation purposes”. In teaching, student sentiment understanding can be used to tackle issues such as confusion and boredom that influence student participation. Several techniques that could be used to learn sentiment from feedback from learners were examined. Hence, Support vector machine (SVM), Naive Bayes, Complement Naive Bayes (CNB) and Maximum Entropy were trained using the feedback of students which so consisted of 1036 instances and differentiated with 3 unique labels positive, negative and neutral. Among all the classification algorithms used two classifiers were found to be having a better understanding of the sentiment, with support vector machine (SVM) showing a higher outcome of 94 percent, following with Complement Naive Bayes to be 84 percent. Results found where higher if they were only 2 labels that is by removing the neutral label the outcome was much more higher.

In their study by (Esparza et al., 2017) model called Social mining using a collection of text comments which were in Spanish on the performance of the teacher and assessing the performance. Classification algorithms used by them are as follows Support vector machine which again has 3 different kernals: linear, radial and polynomial to predict the classification of the comments neutral, positive and negative. As measurements of assessment, they calculated sensitivity, specificity, and predictive values. The findings of this job indicate that the linear kernel has achieved above 0.80 balanced accuracy. The values acquired in finding knowledge were better than those acquired in specificity in all other kernels.

Newman and Joyner (2018) used a tool for sentiment analysis to analyze SET – ’Student Evaluations of Teaching’, the tool used is VADER – ’Valence Aware Dictionary and Sentiment Reasoner’. SET analyzes a single course adopted from three distinct sources: forum comments from other classes, formal evaluations, and an informal student review site. They compared these sites ’ positive and negative comments and then recognized keywords most frequently used in those SET comments and determined positive or negative values by questioning the SET comments on the formal course. Hence, they found out the use of frequently used keywords helped identify where the teacher of the course was powerful or weak. The interrelationship between general analytical results for feedback and general points awarded to a course appears to validate sentiment assessment as a metric.

Gutiérrez et al. (2018) their work described the improvement and process of evaluating a model called Social Mining, focussing recommending courses for improving techniques of teaching through comments given by students in the course evaluation form. The Social Mining model as described earlier analyzes students’ comments by means of different machine learning algorithms such as Random Forest and Support Vector Machines. The evaluation metric is the best way to measure the performance of the algorithms that were used to find the best machine learning model. The built model was used in the University of Aguascalientes (Mexico) and the results obtained showed that it is advantageous to analyze text comments and perform text analysis in classifying comments of Teacher Performance Assessment.

Table 2.1: Previous Work Comparison

Author	Title	Objective	Dataset	Methodology	Outcome
Takamatsu et al. (2018)	Analyzing student course evaluation using text mining.	To visualize open-ended responses/comments.	119 and 138 comments for one course in year 2017-2018 from Kobe Tokiwa University.	Extracting frequency of words and co-occurrence networks and KH coder tool used for text mining	Co-occurrence network diagram was drawn from extracted frequent words which help in finding im-rovements in the course teaching.
Esparza et al. (2017)	A Sentiment Anal-ysis Model to Analyze Students Reviews of Teacher Performance Using Support Vector Machines	To implement strategies which would benefit stu-dents and as well know in-structors performance.	Dataset comprises 1040 comments in Spanish sys-tems engineering students at Polytechnic University of Aguascalientes.	Social Mining is used to guide the assessment of teacher performance applying Support Vector Machine (SVM).	Suggestions on how teachers could improve courses for students and as well as suggest dif-ferent courses to teacher training skills.
Gottipati et al. (2018)	Text analytics approach to extract course improve-ment suggestions from students feedback	To visualize and analyze text comments given in student feedback by ex-tracting explicit sugges-tions about the course.	Dataset consists 5342 comments in total from Singapore Management University, and among which a random test 399 comments used.	Classification of explicit suggestions Decision tree classifier (C5.0), Support vector machine, Rule based classifier	Evaluation of the classi-fication models used and among which Decision tree (C5.0) gave the best result with 785 of accu-racy.
Altrabsheh et al. (2014)	Learning sentiment from students feed-back for real-time interventions in classrooms	To gain knowledge about students sentiments in the case of teaching and eval-uate the best machine learning model used	1075 labelled instances collected from the unit student feedback from the University of Portsmouth.	Classification using ma-chine learning algorithms: Naive Bayes, Comple-ment Naive Bayes, Max-imum Entropy, Support Vector Machine	Comparison of 4 machine algorithms for learning sentiment from students comments and SVM gives the best result with 94% of accuracy.
Newman and Joyner (2018)	Sentiment analysis of student evalua-tions of teaching	To analyze Student Evalu-ations of Teaching (SET) identify positive and neg-ative comments.	Data set is taken from comments of other courses, official evalu-ations and an unofficial reviews site maintained by students.	VADER (Valence Aware Dictionary and Sentiment Reasoner) tool was used for classification.	Frequently occurring key-words help to identify where the course instruc-tor was strong but partic-ular materials were weak, or vice versa.

Chapter 3

Methodology

This chapter explains about the proposed methodology and steps which are followed to accomplish my research study.

3.1 Proposed Methodology

The key focus of this study is to classify the sentiment of text comments so collected from the course evaluation form of AIT. A pictorial view on the proposed methodology for this study is shown in Figure 3.1 having five steps, namely Data-set Description, Data Pre-Processing and labeling, Feature Extraction, Classification of Comments, Evaluation of the Classification Models and further explanation on each stated step is seen in upcoming sections.

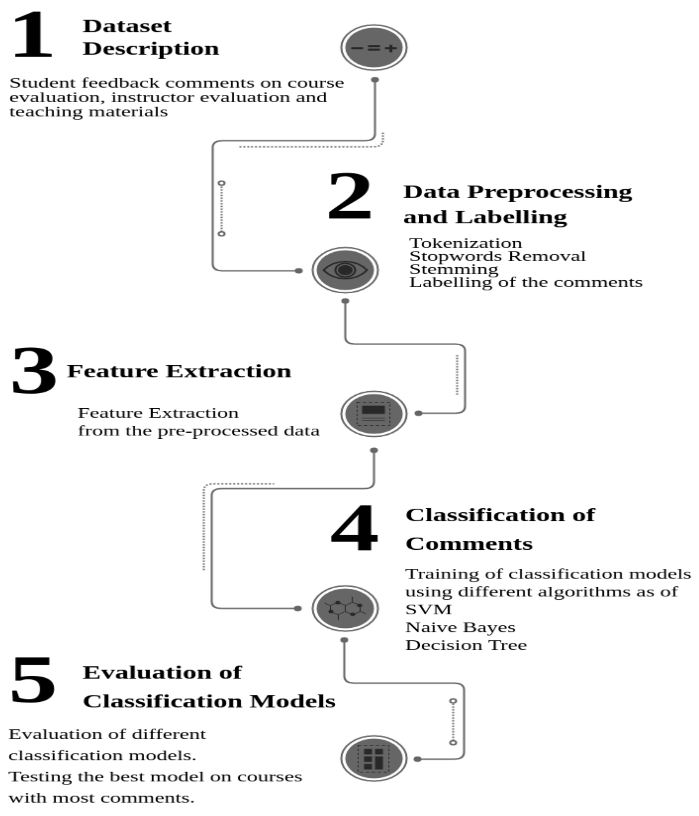
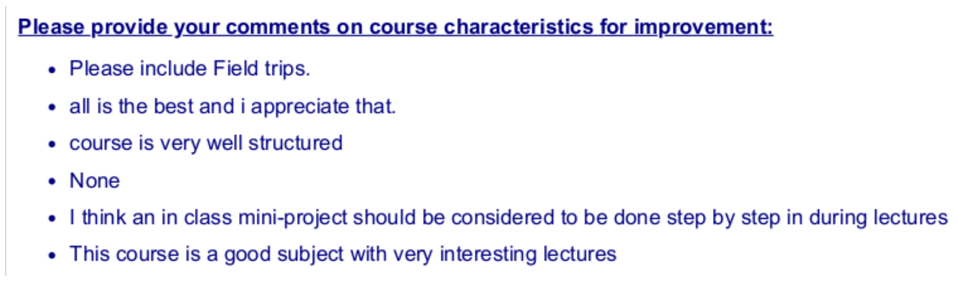

1 Dataset Description
Student feedback comments on course evaluation, instructor evaluation and teaching materials
2 Data Preprocessing and Labelling
Tokenization
Stopwords Removal
Stemming
Labelling of the comments
3 Feature Extraction
Feature Extraction from the pre-processed data
4 Classification of Comments
Training of classification models using different algorithms as of
SVM
Naive Bayes
Decision Tree
5 Evaluation of Classification Models
Evaluation of different classification models.
Testing the best model on courses with most comments.

Figure 3.1: Methodology

3.2 Dataset Description

The data set used for research study is from Asian Institute of Technology (AIT), University from Thailand, It consists of qualitative feedback as of text comments given by students from the year 2015-2019 on different aspects such as course characteristics, instructor characteristics, course delivery/teaching methods/resource materials, and overall assessment and among all the aspects this study uses only comments given to course characteristics, instructor characteristics and as well as comments on teaching methods which would help in improving the course and teaching skills by knowing the sentiment of the text comment. All the comments so received cannot be used for analyzing such as comments such as 'no comments', 'N/A' and 'Nil', as they increase the noise in the data set on removing all these from the data set so combined heading to 482 comments in total and 641 sentences.

- **Course characteristics:** Course characteristics would consists of comments given by students as a review on the course as if suggestions or any positive or negative points which students come across in the course. Example comments can be seen in Figure 3.2.

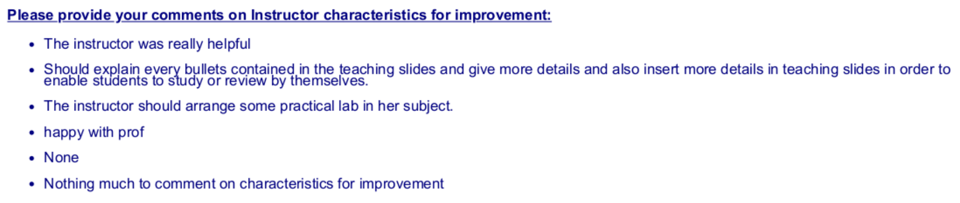


Please provide your comments on course characteristics for improvement:

- Please include Field trips.
- all is the best and i appreciate that.
- course is very well structured
- None
- I think an in class mini-project should be considered to be done step by step in during lectures
- This course is a good subject with very interesting lectures

Figure 3.2: Comments on Course Characteristics

- **Instructor Characteristics:**Instructor characterisitcs would deal with the comments given by the students as a review on the instuructor on the teaching skills or any suggestions which students would like to suggest instructor to improve about the course or in instructors teaching.

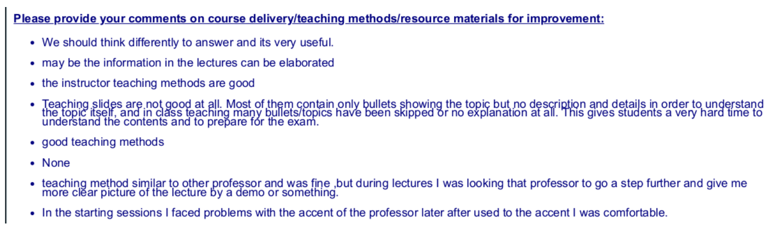


Please provide your comments on Instructor characteristics for improvement:

- The instructor was really helpful
- Should explain every bullets contained in the teaching slides and give more details and also insert more details in teaching slides in order to enable students to study or review by themselves.
- The instructor should arrange some practical lab in her subject.
- happy with prof
- None
- Nothing much to comment on characteristics for improvement

Figure 3.3: Comments on Instructor Characteristics

- **Course Delivery/Teaching Methods/Resource Materials:** This section of student feedback consists of the comments on their reviews on the Teaching methods, Resource materials provided by the instructor for the course, course delivery whether the course was delivered on time, etc,



Please provide your comments on course delivery/teaching methods/resource materials for improvement:

- We should think differently to answer and its very useful.
- may be the information in the lectures can be elaborated
- the instructor teaching methods are good
- Teaching slides are not good at all. Most of them contain only bullets showing the topic but no description and details in order to understand the topic itself, and in class teaching many bullets/topics have been skipped or no explanation at all. This gives students a very hard time to understand the contents and to prepare for the exam.
- good teaching methods
- None
- teaching method similar to other professor and was fine ,but during lectures I was looking that professor to go a step further and give me more clear picture of the lecture by a demo or something.
- In the starting sessions I faced problems with the accent of the professor later after used to the accent I was comfortable.

Figure 3.4: Comments on Course Delivery Methods/ Resource Materials/ Teaching Methods

3.3 Data Pre-Processing

In this stage of system flow pre-processing of the data is done, where sentences are extracted from the input comments using sentence tokenizers. Tokenization is splitting of text into smaller units/tokens as already described in Chapter 2. It is also the major part of text pre-processing. Other than tokenization of data pre-processing stage of the design comes across stemming and lemmatization of data, achieved using Natural Language processing described in brief in Chapter 2. In this stage to train the classification model and evaluate the model, labeling of the tokenized sentences is a much important part in helping the machine to understand natural language. Further, the removal of stopwords helps in giving a more accurate sentiment.

3.3.1 Data Analysis and labelling

During the phase of analyzing data and labeling of data. Labeling of data can be done in two different strategies they are: Labeling text classes and assigning each text sentence to a label or labels. The first approach is called multi-instance learning, while the second approach is using different methods of supervised learning. Labeling of data can also be done manually, as the dataset provided by AIT is unstructured data I am planning to manually label after sentence tokenization of the unstructured comments.

3.4 Feature Extraction

To analyze the data which is preprocessed earlier, data needs to be converted into vectors. Vectorization of the text data is much more important as a machine cannot understand text for which many different text extraction techniques could be used as of N-grams, POS tagging, word embedding.

3.4.1 Feature Construction and Weighting

In this phase of feature extraction, according to Mirończuk and Protasiewicz (2018) the data set which is labelled is represented suitably for a algorithm to learn. Textual data can be represented in two ways they are:

- Vector space representation or model where representation of document is done as a vector of weights, whereas words/phrases) are represented in the form of a document.
- Graphical representation, where representing of document is done in the form of graph, nodes of the graph representing the words, while the edges representing the relationship between those words.

Both the representations vector space and graphical are completely dependent on their weights and features. They are many approaches among which the most commonly heard feature types are as follows: Simple features include uni-grams, bi-grams and n-grams. Domain specific features, such as opinion, emotion.etc. Embedded feature as such words to vectors word2vec, or global vector for word representation GloVe

3.5 Classification of Comments

This stage involves extracting sentiment using text classification methods. In this research study, I plan to implement different classification algorithms described in chapter 2. Implementation of these algorithms will lead to generation of models that would help in classification. Hence, this stage of classification would help in classifying sentiment compared to other comments given by students which would rather improve the quality of the course and as well as the teaching methods by instructors.

3.6 Evaluation of the Models

Since this research study is aiming to build different classification models, there is a need to find the optimal model which gives the best result. Testing of the model is done on the top two courses with the highest number of comments. To check the performance of the classified sentiments confusion metrics is used and based on this the following: Accuracy, Precision, Recall, F-measure are calculated.

Chapter 4

Experimental Results

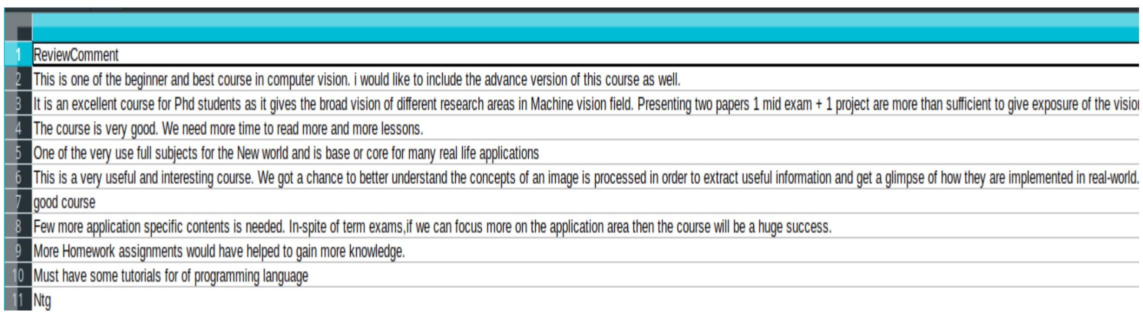
In the following chapter description about data pre-processing, labeling of the student evaluation of courses, feature extraction after pre-processing, model building, model evaluation and explaining the best results.

4.1 Data Description

The data set is being used from the Asian Institute of technology and it consists of text comments/suggestions/feedback given by students in the course evaluation form at the end of every semester. Briefing down the data set description, it consists of the feedback given by students from the year 2015-2019 with a total of 482 text comments and on different aspects such as course evaluation, instructor evaluation, and course delivery/teaching methods/resource materials as already described in Chapter 3.

4.2 Data Pre-processing and Labelling

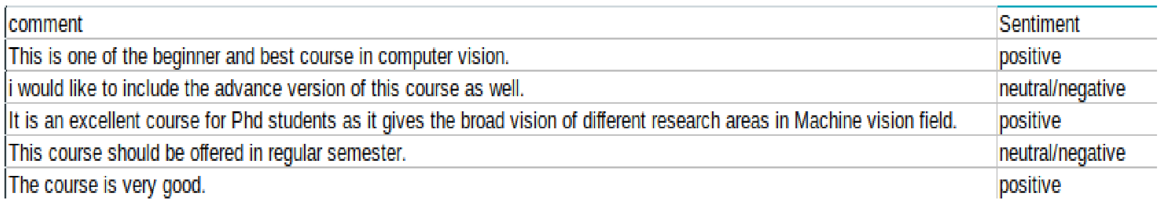
In this section of data pre-processing and labeling the data set or text comments are tokenized, normalized and then later being manually labeled. The first step of data pre-processing deals with the tokenization of the comments as already described what tokenization is splitting of phrases into smaller sentences or words.



1	ReviewComment
2	This is one of the beginner and best course in computer vision. i would like to include the advance version of this course as well.
3	It is an excellent course for Phd students as it gives the broad vision of different research areas in Machine vision field. Presenting two papers 1 mid exam + 1 project are more than sufficient to give exposure of the visio
4	The course is very good. We need more time to read more and more lessons.
5	One of the very use full subjects for the New world and is base or core for many real life applications
6	This is a very useful and interesting course. We got a chance to better understand the concepts of an image is processed in order to extract useful information and get a glimpse of how they are implemented in real-world.
7	good course
8	Few more application specific contents is needed. In-spite of term exams,if we can focus more on the application area then the course will be a huge success.
9	More Homework assignments would have helped to gain more knowledge.
10	Must have some tutorials for of programming language
11	Ntg

Figure 4.1: Sample comments from students on course/instructors/teaching material

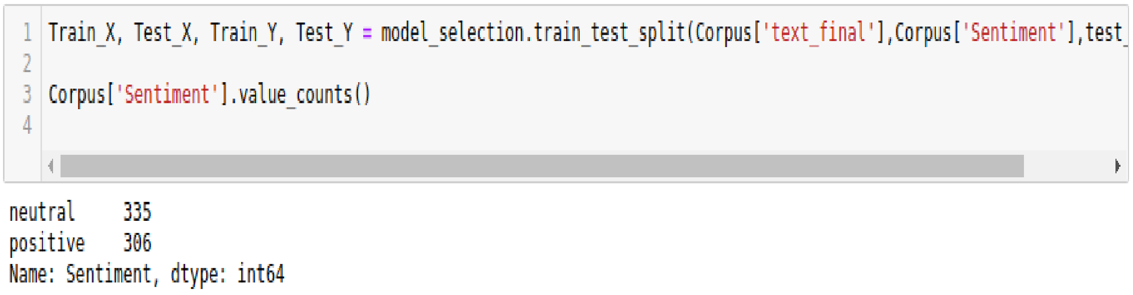
The second step after tokenization of long phrases into sentences manual labeling of the data set is done. 482 comments collected combining all the text reviews are broken down into 641 sentences and labeling of tokenized sentences is done in 2 different classes they are as follows: ’positive’, ’neutral/negative’. Here, a positive class means the comment being a positive comment to the course, instructor, and teaching material/course material. Example: ”good course”, ”instructor is good at teaching”, etc. whereas neutral/negative class refers to whether the comment is a neutral/negative comment. Example: ”no comments”, ”course needs to improve”, etc.



comment	Sentiment
This is one of the beginner and best course in computer vision.	positive
i would like to include the advance version of this course as well.	neutral/negative
It is an excellent course for Phd students as it gives the broad vision of different research areas in Machine vision field.	positive
This course should be offered in regular semester.	neutral/negative
The course is very good.	positive

Figure 4.2: Sample comments with tokenized comments and sentiment labelling

The labeled sentiment of the tokenized comments described in the second step is shown in Figure 4.2. The total number of instances which are labeled is 641 and as known the classes divided are two each class having certain number of instances such as, positive sentiment class with 306 comments, and neutral/negative sentiment class with 335, seen in Figure 4.3 giving an overview on the number of instances in each class.

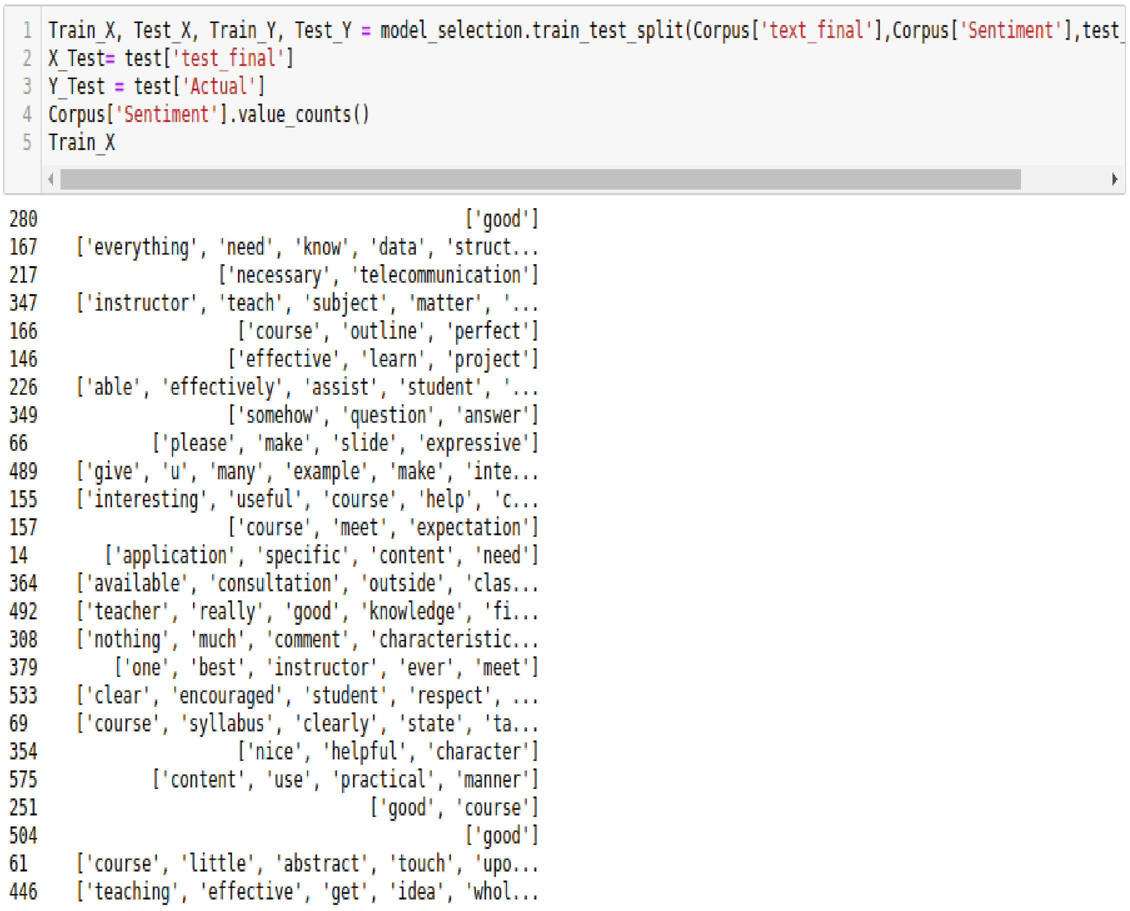


```
Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_split(Corpus['text_final'],Corpus['Sentiment'],test_

Corpus['Sentiment'].value_counts()

```

```
neutral     335
positive    306
Name: Sentiment, dtype: int64
```

Figure 4.3: Total number of positive and neutral/negative comments

The third step is tokenizing each sentence into words and normalization of the tokenized sentences on labeled sentences. In chapter 2 of this study normalization and stop word removal have been described already. Hence removing stop words help in increasing the accuracy giving a better result and normalization consists of stemming and lemmatization of words after tokenization. Figure 4.4 shows the pre-processed comments after tokenization and normalization of the text comments. Therefore these are the three steps involved in this stage of data pre-processing and labeling.



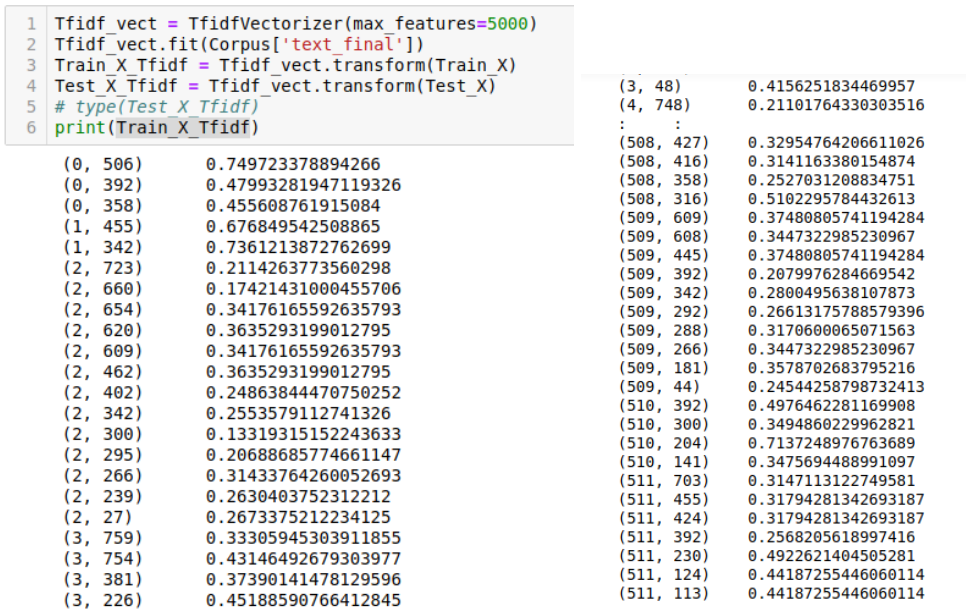
```
Train_X, Test_X, Train_Y, Test_Y = model_selection.train_test_split(Corpus['text_final'],Corpus['Sentiment'],test_
X_Test= test['test_final']
Y_Test = test['Actual']
Corpus['Sentiment'].value_counts()
Train_X
```

```
280                                         ['good']
167    ['everything', 'need', 'know', 'data', 'struct...
217               ['necessary', 'telecommunication']
347    ['instructor', 'teach', 'subject', 'matter', '...
166                  ['course', 'outline', 'perfect']
146                 ['effective', 'learn', 'project']
226    ['able', 'effectively', 'assist', 'student', '...
349                 ['somehow', 'question', 'answer']
66          ['please', 'make', 'slide', 'expressive']
489    ['give', 'u', 'many', 'example', 'make', 'inte...
155    ['interesting', 'useful', 'course', 'help', 'c...
157                 ['course', 'meet', 'expectation']
14       ['application', 'specific', 'content', 'need']
364    ['available', 'consultation', 'outside', 'clas...
492    ['teacher', 'really', 'good', 'knowledge', 'fi...
308    ['nothing', 'much', 'comment', 'characteristic...
379       ['one', 'best', 'instructor', 'ever', 'meet']
533    ['clear', 'encouraged', 'student', 'respect', ...
69     ['course', 'syllabus', 'clearly', 'state', 'ta...
354                  ['nice', 'helpful', 'character']
575         ['content', 'use', 'practical', 'manner']
251                               ['good', 'course']
504                                         ['good']
61     ['course', 'little', 'abstract', 'touch', 'upo...
446    ['teaching', 'effective', 'get', 'idea', 'whol...
```

Figure 4.4: Pre-processed text comments

4.3 Feature Construction

In Chapter 3 of this research study, feature construction has been described. In brief feature construction is vectorizing of pre-processed text data for predicting the sentiment of the comment with the help of machine learning algorithms. Vectorization can be done using many different ways such as count vectorizer, bag of words and etc. In this research study, Tf-Idf vectorizer has been used for converting text data into numerical vectors. Term Frequency is counting the number times the word has appeared in the whole text file and that of Inverse Document Frequency the measure of the importance of that term in the whole text file. Figure 4.5 shows the instance after vectorization using Tf-IDF.



```
Tfidf_vect = TfidfVectorizer(max_features=5000)
Tfidf_vect.fit(Corpus['text_final'])
Train_X_Tfidf = Tfidf_vect.transform(Train_X)
Test_X_Tfidf = Tfidf_vect.transform(Test_X)
# type(Test_X_Tfidf)
print(Train_X_Tfidf)
```

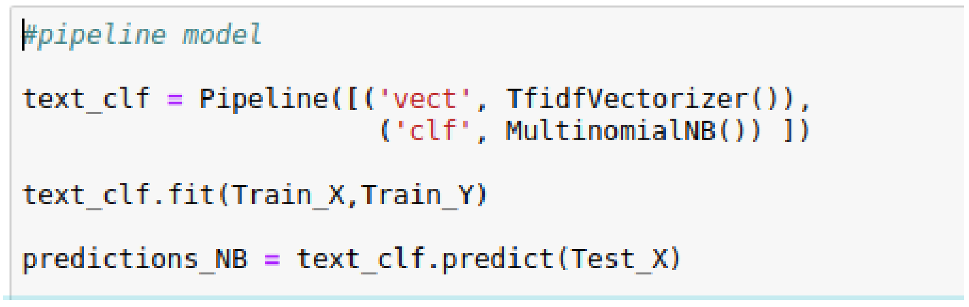
```
  (0, 506)	0.749723378894266
  (0, 392)	0.47993281947119326
  (0, 358)	0.455608761915084
  (1, 455)	0.676849542508865
  (1, 342)	0.7361213872762699
  (2, 723)	0.2114263773560298
  (2, 660)	0.17421431000455706
  (2, 654)	0.34176165592635793
  (2, 620)	0.3635293199012795
  (2, 609)	0.34176165592635793
  (2, 462)	0.3635293199012795
  (2, 402)	0.24863844470750252
  (2, 342)	0.2553579112741326
  (2, 300)	0.13319315152243633
  (2, 295)	0.20688685774661147
  (2, 266)	0.31433764260052693
  (2, 239)	0.2630403752312212
  (2, 27)	0.2673375212234125
  (3, 759)	0.33305945303911855
  (3, 754)	0.43146492679303977
  (3, 381)	0.37390141478129596
  (3, 226)	0.45188590766412845
```

```
  (3, 48)	0.4156251834469957
  (4, 748)	0.21101764330303516
  :	:
  (508, 427)	0.32954764206611026
  (508, 416)	0.3141163380154874
  (508, 358)	0.2527031208834751
  (508, 316)	0.5102295784432613
  (509, 609)	0.37480805741194284
  (509, 608)	0.3447322985230967
  (509, 445)	0.37480805741194284
  (509, 392)	0.2079976284669542
  (509, 342)	0.2800495638107873
  (509, 292)	0.26613175788579396
  (509, 288)	0.3170600065071563
  (509, 266)	0.3447322985230967
  (509, 181)	0.3578702683795216
  (509, 44)	0.24544258798732413
  (510, 392)	0.4976462281169908
  (510, 300)	0.3494860229962821
  (510, 204)	0.7137248976763689
  (510, 141)	0.3475694488991097
  (511, 703)	0.3147113122749581
  (511, 455)	0.31794281342693187
  (511, 424)	0.31794281342693187
  (511, 392)	0.2568205618997416
  (511, 230)	0.4922621404505281
  (511, 124)	0.44187255446060114
  (511, 113)	0.44187255446060114
```

Figure 4.5: Tf-Idf vectorization

4.4 Model Building

Classification is categorizing data in different classes and in this research study classification of the sentiment using different classification algorithms is been concentrated. Classification algorithms such as Support Vector Machine, Bernoulli Naive Bayes, XGBoost, Stochastic Gradient, Decision Tree are the once used in this research study and about each of these algorithms are described in Chapter 2. The classification results so obtained by each algorithm is compared based on the performance matrix so achieved on the validation set.The data set is divided into 80-20 form that is 80% for training and 20% for validation. The confusion matrix is one of the ways to show the performance matrics/classification report of each algorithm, will be further seen in the upcoming subsections.



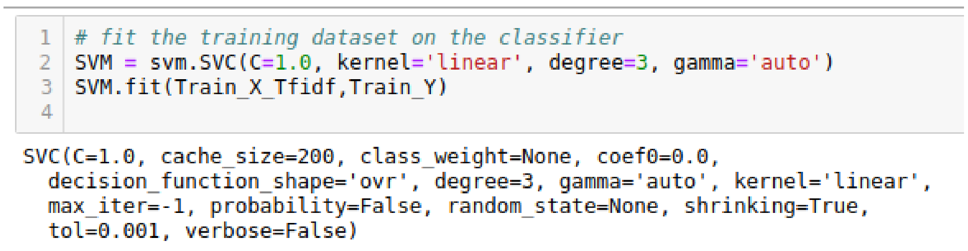
```
#pipeline model

text_clf = Pipeline([('vect', TfidfVectorizer()),
                     ('clf', MultinomialNB()) ])

text_clf.fit(Train_X,Train_Y)

predictions_NB = text_clf.predict(Test_X)
```

Figure 4.6: Example use of pipeline

In the further sections to decrease down the time complexity of building a model, pipeline is used. Pipeline helps in reducing the steps which are required for generating a model and generating a pipe-like continuous structure increasing the model to be more effective. Figure 4.6 shows the example use of pipeline. Implementation of the models or the classifiers is done using Python libraries. NumPy, SciPy, Matplotlib, Sci-kit learn are few python libraries being used. Chapter 2 describes these libraries and classification algorithms.

4.4.1 Support Vector Machine

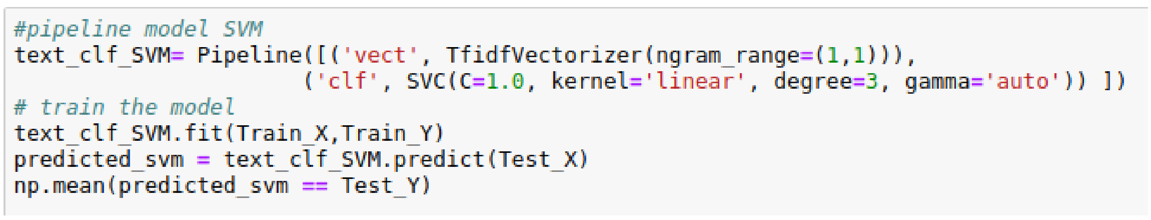
Support Vector Machine is one of the best classification algorithms known so far and implementing SVM on the text comments would require pre-processing of the data already covered in Section 4.3. A support vector classifier is built using one of the python libraries named sklearn and requires certain parameters for building a model, Figure 4.7 shows the required parameters. The kernel is one of the major parameters of SVM, function of the kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be of different types. For example linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.



```
# fit the training dataset on the classifier
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)

```

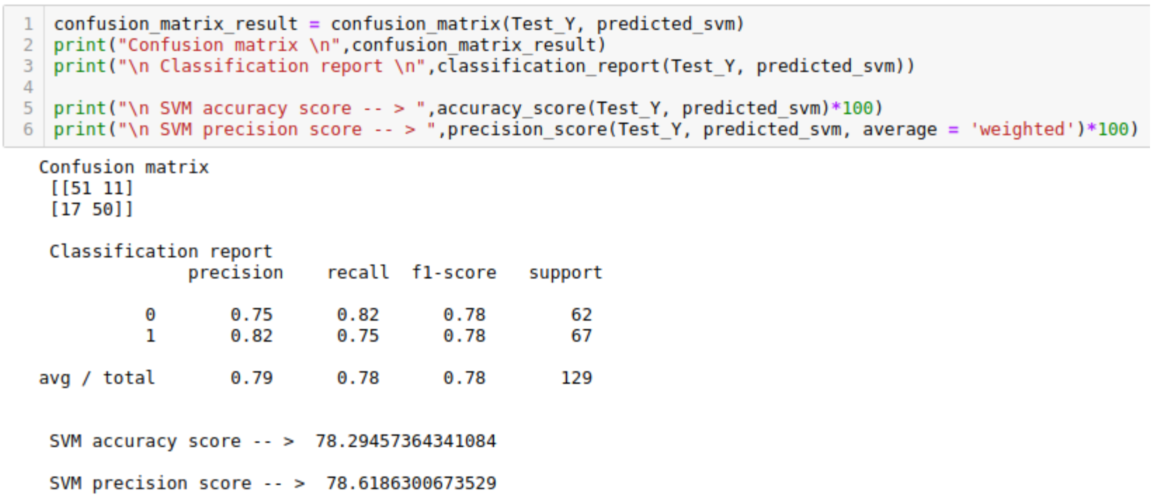
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

Figure 4.7: SVM model building



```
#pipeline model SVM
text_clf_SVM= Pipeline([('vect', TfidfVectorizer(ngram_range=(1,1))),
                      ('clf', SVC(C=1.0, kernel='linear', degree=3, gamma='auto')) ])
# train the model
text_clf_SVM.fit(Train_X,Train_Y)
predicted_svm = text_clf_SVM.predict(Test_X)
np.mean(predicted_svm == Test_Y)
```

Figure 4.8: Pipeline model for SVM classification



```
confusion_matrix_result = confusion_matrix(Test_Y, predicted_svm)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Test_Y, predicted_svm))

print("\n SVM accuracy score -- > ",accuracy_score(Test_Y, predicted_svm)*100)
print("\n SVM precision score -- > ",precision_score(Test_Y, predicted_svm, average = 'weighted')*100)
```

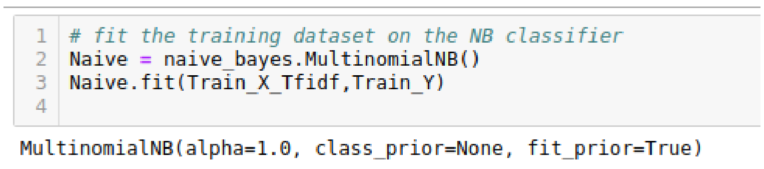
```
Confusion matrix
 [[51 11]
 [17 50]]

 Classification report
              precision    recall  f1-score   support

          0       0.75      0.82      0.78        62
          1       0.82      0.75      0.78        67

avg / total       0.79      0.78      0.78       129


 SVM accuracy score -- >  78.29457364341084

 SVM precision score -- >  78.6186300673529
```

Figure 4.9: SVM classification report

In Figure 4.7 the inputs used are Train_X defines the training data collected after the feature extraction of the text data into numerical values and where as Train_Y does consist of the two sentiment classes defined earlier which is to be predicted after the fitting of the model. In this study linear kernel has been taken as the SVM classifier and gamma to be in auto as default parameter. Results of the classifier could be seen in the Figure 4.9 with a accuracy of 78.29% showing to be the best result.

4.4.2 Naive Bayes

Naive Bayes is one of the simple classification algorithms so far based on Bayes theorem assuming each attribute to be independent of others, the inputs used are the same as earlier in SVM, Train_X, and Tain_Y for the training of the model. In this algorithm, a pipeline is used to generate the model for classification taking the input to be Train_X and Train_Y which are split from the whole dataset.



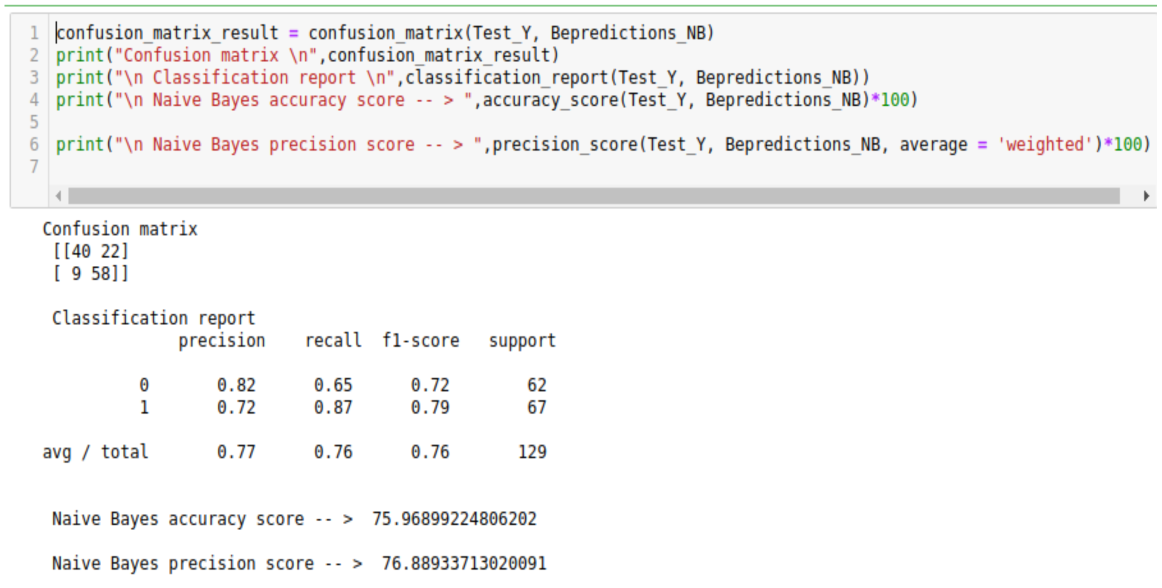
```
# fit the training dataset on the NB classifier
Naive = naive_bayes.MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)

```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Figure 4.10: Naive Bayes model building

Naive Bayes itself has got two classifiers they are already described in Chapter 2 of this research study. Bernoulli Naive Bayes is used as the dataset is too small to apply complex

Multinomial Naive Bayes algorithm giving an accuracy of 75.9% without using ngrams. Figure 4.10 shows the model build for Bernoulli classifier and the classification report showing the matrix in Figure 4.11



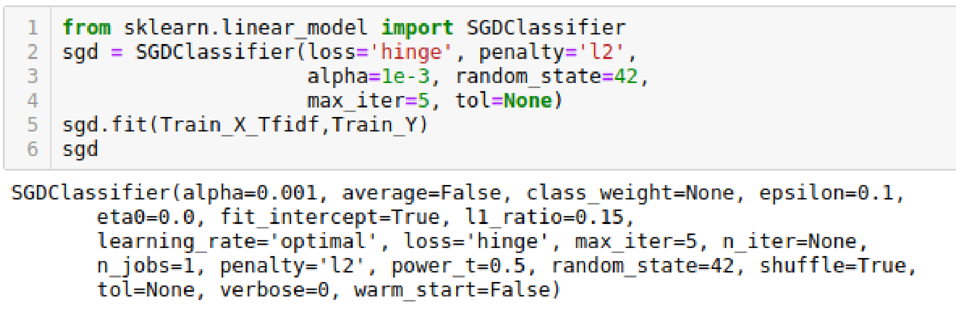
```
confusion_matrix_result = confusion_matrix(Test_Y, Bepredictions_NB)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Test_Y, Bepredictions_NB))
print("\n Naive Bayes accuracy score -- > ",accuracy_score(Test_Y, Bepredictions_NB)*100)

print("\n Naive Bayes precision score -- > ",precision_score(Test_Y, Bepredictions_NB, average = 'weighted')*100)

```

```
Confusion matrix
 [[40 22]
 [ 9 58]]

 Classification report
              precision    recall  f1-score   support

          0       0.82      0.65      0.72        62
          1       0.72      0.87      0.79        67

avg / total       0.77      0.76      0.76       129


 Naive Bayes accuracy score -- >  75.96899224806202

 Naive Bayes precision score -- >  76.88933713020091
```

Figure 4.11: Naive Bayes classification report

4.4.3 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is also one of the simplest and yet very structured approaches to discriminate against the learning of linear classifiers with smaller loss functions such as Logistic Regression and Support Vector Machines.

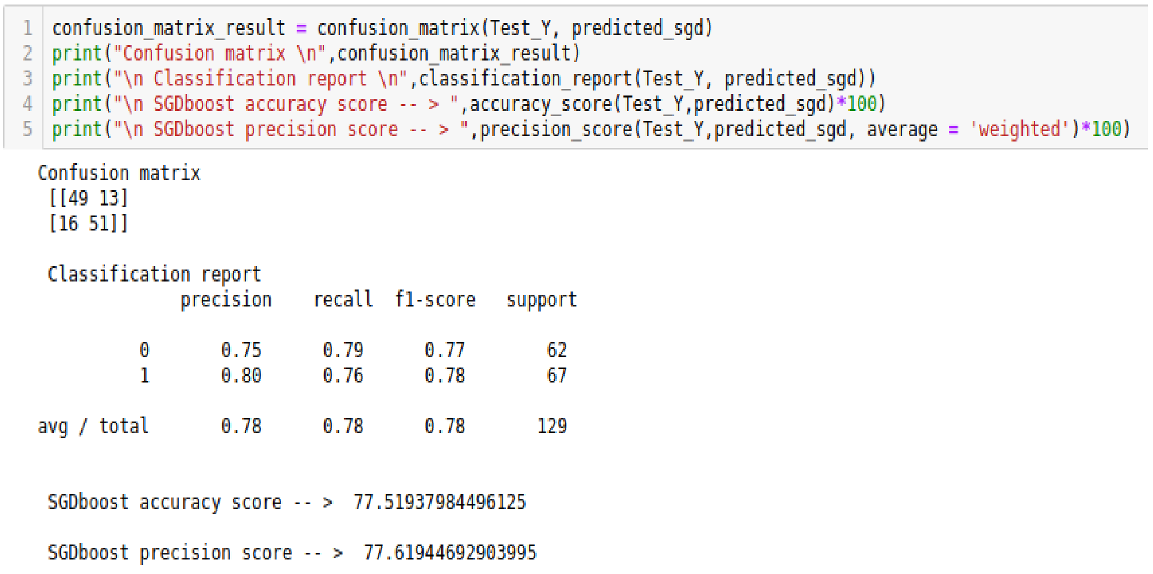


```
from sklearn.linear_model import SGDClassifier
sgd = SGDClassifier(loss='hinge', penalty='l2',
                    alpha=1e-3, random_state=42,
                    max_iter=5, tol=None)
sgd.fit(Train_X_Tfidf,Train_Y)
sgd
```

```
SGDClassifier(alpha=0.001, average=False, class_weight=None, epsilon=0.1,
       eta0=0.0, fit_intercept=True, l1_ratio=0.15,
       learning_rate='optimal', loss='hinge', max_iter=5, n_iter=None,
       n_jobs=1, penalty='l2', power_t=0.5, random_state=42, shuffle=True,
       tol=None, verbose=0, warm_start=False)
```

Figure 4.12: Stochastic Gradient Descent model building

In Figure 4.12 parameters required for the classification model could be seen in which Train_X_Tfidf is the vectorized data and Train_Y is the labeled data Sentiment of the text comment. The testing of the model is done on the validation set split shown in the earlier sections.



```
confusion_matrix_result = confusion_matrix(Test_Y, predicted_sgd)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Test_Y, predicted_sgd))
print("\n SGDboost accuracy score -- > ",accuracy_score(Test_Y,predicted_sgd)*100)
print("\n SGDboost precision score -- > ",precision_score(Test_Y,predicted_sgd, average = 'weighted')*100)
```

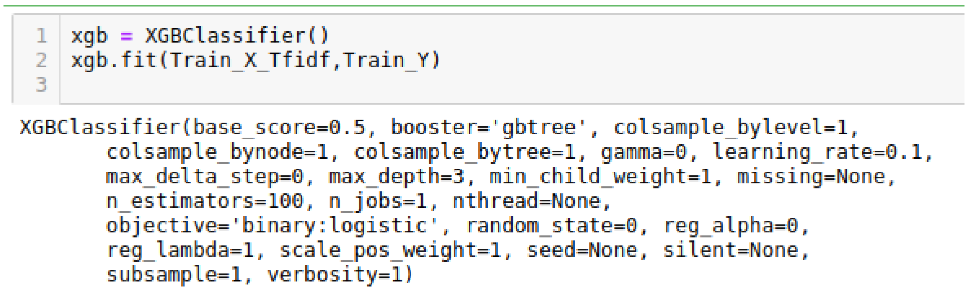
```
Confusion matrix
 [[49 13]
 [16 51]]

 Classification report
              precision    recall  f1-score   support

          0       0.75      0.79      0.77        62
          1       0.80      0.76      0.78        67

avg / total       0.78      0.78      0.78       129


 SGDboost accuracy score -- >  77.51937984496125

 SGDboost precision score -- >  77.61944692903995
```

Figure 4.13: Stochastic Gradient Descent classification report

Classification report in Figure 4.13 shows the result with a precision score of 77.6% and accuracy score of 77.51% with use of n-grams ranging from minimum being it to be one to a range of four to the maximum improving the accuracy on the validation set.

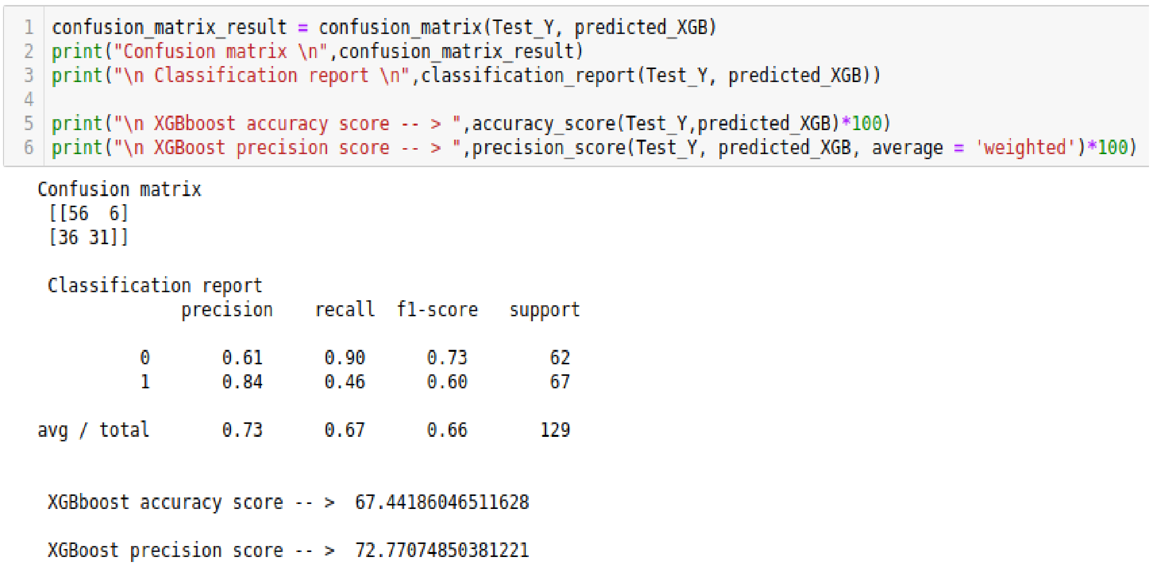
4.4.4 XGboost

XGboost is also one of the gradient boosting algorithms used for improving the performance and competitive results in machine learning. XGboost has certain different parameters to be considered when it comes to building the model for classification which could be seen in figure 4.14. The learning rate is used to improve the accuracy of the algorithm, 0.1 is the default learning rate if not defined. The ”max depth” parameter controls the depth of the tree which avoids overfitting.



```
xgb = XGBClassifier()
xgb.fit(Train_X_Tfidf,Train_Y)

XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0, learning_rate=0.1,
       max_delta_step=0, max_depth=3, min_child_weight=1, missing=None,
       n_estimators=100, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=None,
       subsample=1, verbosity=1)
```

Figure 4.14: XGboost model building



```
confusion_matrix_result = confusion_matrix(Test_Y, predicted_XGB)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Test_Y, predicted_XGB))

print("\n XGBboost accuracy score -- > ",accuracy_score(Test_Y,predicted_XGB)*100)
print("\n XGBoost precision score -- > ",precision_score(Test_Y, predicted_XGB, average = 'weighted')*100)
```

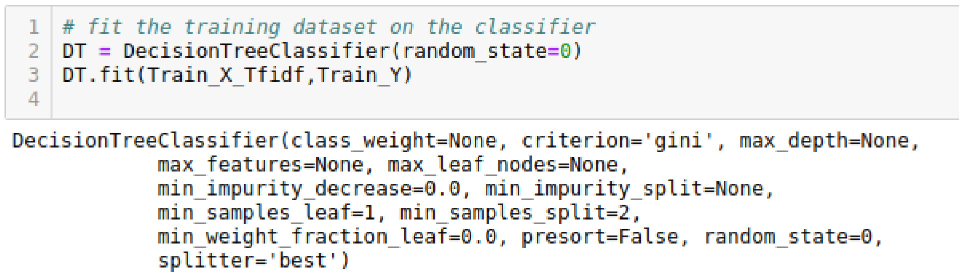
```
Confusion matrix
 [[56  6]
 [36 31]]

 Classification report
              precision    recall  f1-score   support

          0       0.61      0.90      0.73        62
          1       0.84      0.46      0.60        67

avg / total       0.73      0.67      0.66       129


 XGBboost accuracy score -- >  67.44186046511628

 XGBoost precision score -- >  72.77074850381221
```

Figure 4.15: XGboost classification report

In Figure 4.15 accuracy on the validation set to be 67.44% as the data set is quite too small making the algorithm respectable of giving a much higher accuracy compared to other algorithms and giving a precision score of 72.7% giving the least among all the algorithms used.

4.4.5 Decision Tree (Classification and Regression Tree)

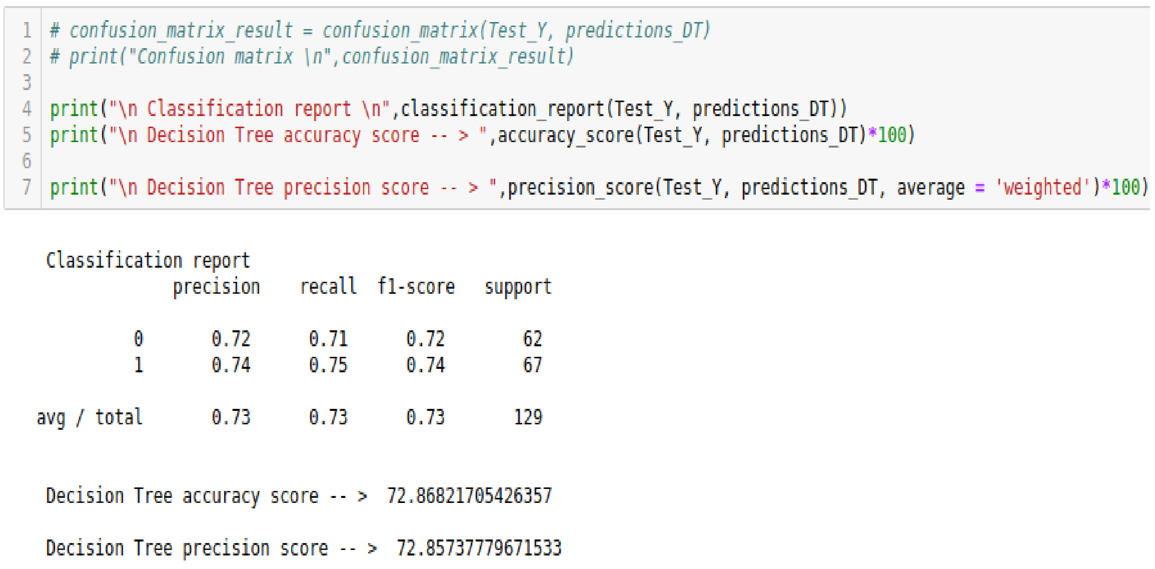
Decision Trees (DTs) are a supervised learning method used for regression and classification. The parameters required by the Decision tree model can be seen in Figure 4.16. CART is one of the decision tree algorithms which gives 72% of accuracy on validation set seen in Figure 4.17.



```python
# fit the training dataset on the classifier
DT = DecisionTreeClassifier(random_state=0)
DT.fit(Train_X_Tfidf,Train_Y)

```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=0,
            splitter='best')
```

Figure 4.16: CART model building



```python
# confusion_matrix_result = confusion_matrix(Test_Y, predictions_DT)
# print("Confusion matrix \n",confusion_matrix_result)

print("\n Classification report \n",classification_report(Test_Y, predictions_DT))
print("\n Decision Tree accuracy score -- > ",accuracy_score(Test_Y, predictions_DT)*100)

print("\n Decision Tree precision score -- > ",precision_score(Test_Y, predictions_DT, average = 'weighted')*100)
```

```
 Classification report
              precision    recall  f1-score   support

          0       0.72      0.71      0.72        62
          1       0.74      0.75      0.74        67

avg / total       0.73      0.73      0.73       129

 Decision Tree accuracy score -- >  72.86821705426357

 Decision Tree precision score -- >  72.85737779671533
```

Figure 4.17: CART classification report

4.5 Results

In this section, the results of all the classification models built with or without use of ngrams are compared making it easy to decide the best model and using the best model for testing on an individual course, this section also shows visualization of the text comments represented with the help of word cloud separating based on the positive or negative/neutral comments.

4.5.1 Model Evaluation

Model evaluation is choosing the best classification algorithm built based on the performance metric, as this research study focused on classifying the sentiment of the text comment choosing the best model is important. Comparison of the performance metrics obtained by each algorithm using ngrams shown in Table 4.1 and performance metric of each algorithm without the use of ngrams is shown in Table 4.2.

Table 4.1: Comparison of performance metrics on the classification algorithms using ngrams

Algorithm	Accuracy	Precision	Recall	F1 score
Support Vector Machine	0.78	0.79	0.78	0.78
Naive Bayes	0.62	0.71	0.62	0.57
Stochastic Gradient	0.77	0.78	0.78	0.77
XGboost	0.67	0.72	0.67	0.66
Decision Tree (CART)	0.71	0.72	0.71	0.71

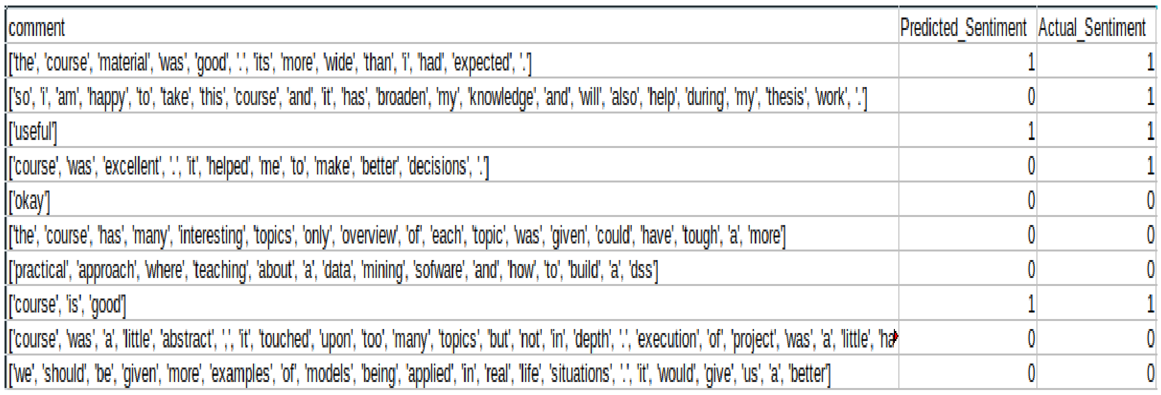
Classifying sentiment of comments help instructors to come up with understanding reviews of students. Support vector machine does give a better result with an accuracy of 78% with the use of n-grams seen in comparison Table 4.1 and Bernoulli Naive Bayes gave better result with 75% without n-grams seen in comparison Table 4.2 due to lack of dataset the results couldn't be much more accurate which is one of the limitations of this study.

Table 4.2: Comparison of performance metrics on the classification algorithms without using ngrams

Algorithm	Accuracy	Precision	Recall	F1 score
Support Vector Machine	0.74	0.74	0.74	0.74
Naive Bayes	0.75	0.76	0.76	0.75
Stochastic Gradient	0.72	0.73	0.72	0.72
XGboost	0.66	0.76	0.67	0.64
Decision Tree (CART)	0.72	0.73	0.73	0.73

4.5.2 Model Testing

Support Vector Machine giving the best results with the use of n-grams manually verification on the classification is done on the testing set with only comments as input. The testing set is taken from the courses with the most number of comments in the received data set which is in the evaluation form of different courses. Course 1 and course 5 with the highest number of comments and is stored manually in a .csv file for testing the classification results.
Figure 4.18 shows some predicted results on the example comments using the SVM classifier. First attribute comment being the input and the second attribute sentiment being the predicted class of the comment where 1 represents positive and 0 represents as a negative or neutral comment, and the third attribute actual sentiment is the manual labeling of the comment for verification of the predicted value or can be described as the ground truth sentiments of the comments.



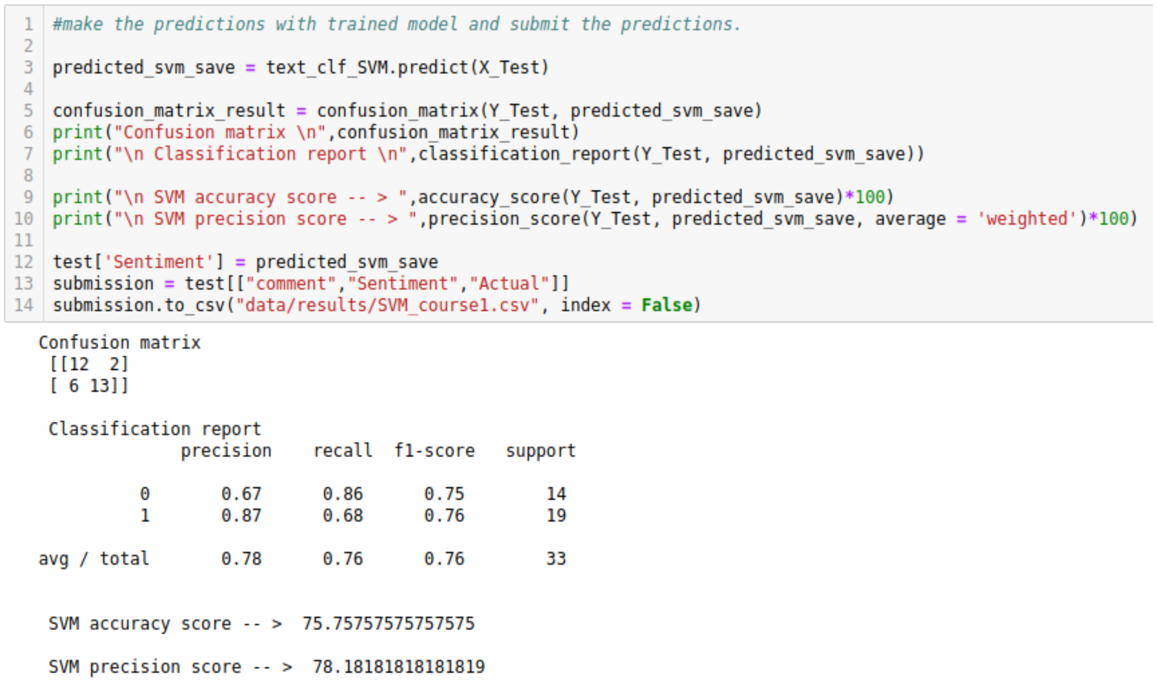
comment	Predicted_Sentiment	Actual_Sentiment
['the', 'course', 'material', 'was', 'good', '.', 'its', 'more', 'wide', 'than', 'i', 'had', 'expected', '.']	1	1
['so', 'i', 'am', 'happy', 'to', 'take', 'this', 'course', 'and', 'it', 'has', 'broaden', 'my', 'knowledge', 'and', 'will', 'also', 'help', 'during', 'my', 'thesis', 'work', '.']	0	1
['useful']	1	1
['course', 'was', 'excellent', '.', 'it', 'helped', 'me', 'to', 'make', 'better', 'decisions', '.']	0	1
['okay']	0	0
['the', 'course', 'has', 'many', 'interesting', 'topics', 'only', 'overview', 'of', 'each', 'topic', 'was', 'given', 'could', 'have', 'tough', 'a', 'more']	0	0
['practical', 'approach', 'where', 'teaching', 'about', 'a', 'data', 'mining', 'sofware', 'and', 'how', 'to', 'build', 'a', 'dss']	0	0
['course', 'is', 'good']	1	1
['course', 'was', 'a', 'little', 'abstract', ',', 'it', 'touched', 'upon', 'too', 'many', 'topics', 'but', 'not', 'in', 'depth', '.', 'execution', 'of', 'project', 'was', 'a', 'little', 'ha	0	0
['we', 'should', 'be', 'given', 'more', 'examples', 'of', 'models', 'being', 'applied', 'in', 'real', 'life', 'situations', '.', 'it', 'would', 'give', 'us', 'a', 'better']	0	0

Figure 4.18: Sample predicted sentiment results on the courses

Classification report and the confusion metrics obtained of sentiment predicted comments of Course 1 and Course 5 is shown in Figure 4.19 and Figure 4.20 giving a accuracy of 75.75% and 76.3% using SVM classification algorithm as it is the best-built model with use of n-grams as seen in Table 4.1. Comparison of the performance metrics is shown in Table 4.3 of Course 1 and Course 5 as they have the highest number of comments received among all the courses. .

Table 4.3: Performance matrix on two courses with highest number of comments

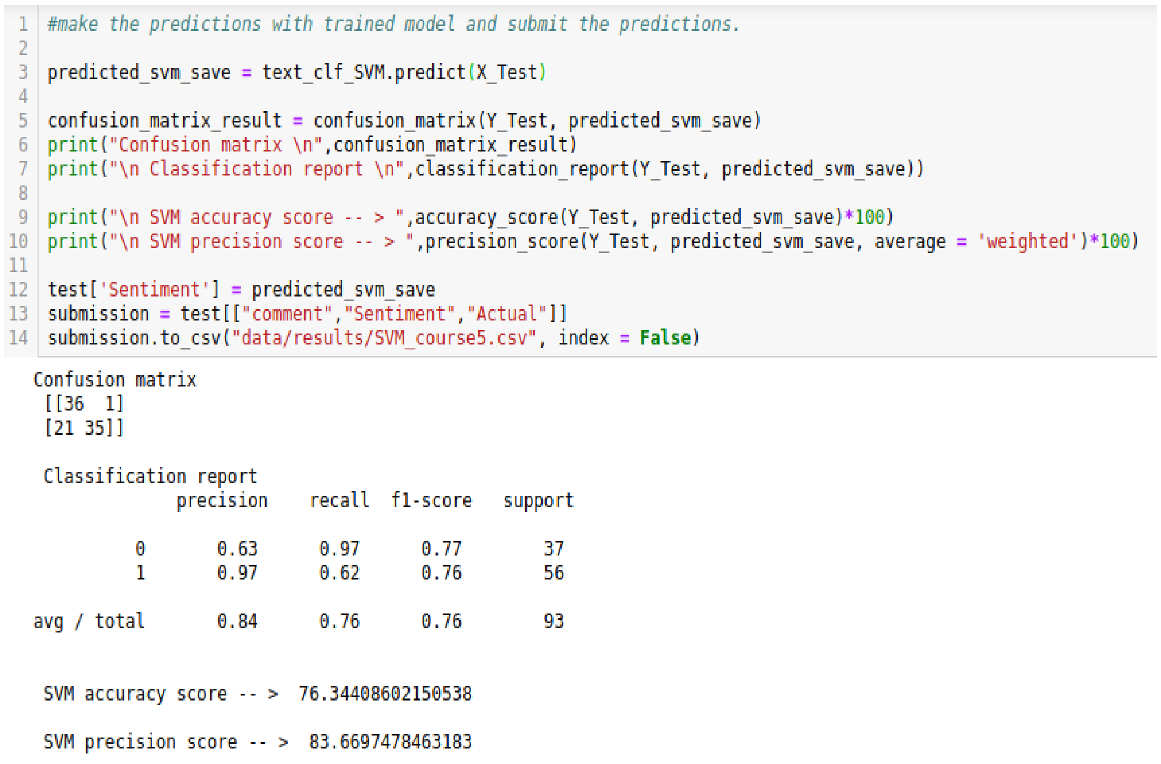
Courses	Accuracy	Precision	Recall	F1 score
Course 1	0.75	0.78	0.76	0.76
Course 5	0.76	0.83	0.76	0.76



```
#make the predictions with trained model and submit the predictions.

predicted_svm_save = text_clf_SVM.predict(X_Test)

confusion_matrix_result = confusion_matrix(Y_Test, predicted_svm_save)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Y_Test, predicted_svm_save))

print("\n SVM accuracy score -- > ",accuracy_score(Y_Test, predicted_svm_save)*100)
print("\n SVM precision score -- > ",precision_score(Y_Test, predicted_svm_save, average = 'weighted')*100)

test['Sentiment'] = predicted_svm_save
submission = test[["comment","Sentiment","Actual"]]
submission.to_csv("data/results/SVM_course1.csv", index = False)
```

```
Confusion matrix
 [[12  2]
 [ 6 13]]

 Classification report
              precision    recall  f1-score   support

          0       0.67      0.86      0.75        14
          1       0.87      0.68      0.76        19

avg / total       0.78      0.76      0.76        33


 SVM accuracy score -- >  75.75757575757575

 SVM precision score -- >  78.18181818181819
```

Figure 4.19: Classification report of text comments given on course 1



```
#make the predictions with trained model and submit the predictions.

predicted_svm_save = text_clf_SVM.predict(X_Test)

confusion_matrix_result = confusion_matrix(Y_Test, predicted_svm_save)
print("Confusion matrix \n",confusion_matrix_result)
print("\n Classification report \n",classification_report(Y_Test, predicted_svm_save))

print("\n SVM accuracy score -- > ",accuracy_score(Y_Test, predicted_svm_save)*100)
print("\n SVM precision score -- > ",precision_score(Y_Test, predicted_svm_save, average = 'weighted')*100)

test['Sentiment'] = predicted_svm_save
submission = test[["comment","Sentiment","Actual"]]
submission.to_csv("data/results/SVM_course5.csv", index = False)
```

```
Confusion matrix
 [[36  1]
 [21 35]]

 Classification report
              precision    recall  f1-score   support

          0       0.63      0.97      0.77        37
          1       0.97      0.62      0.76        56

avg / total       0.84      0.76      0.76        93


 SVM accuracy score -- >  76.34408602150538

 SVM precision score -- >  83.6697478463183
```

Figure 4.20: Classification report of text comments given on course 5

4.5.3 Visualization

Visualization of the comments seen in Figure 4.21 shows the bar graph of total positive and neutral/negative comments received among all the courses combined. Whereas Figure 4.24 shows the visualization of positive comments and Figure 4.25 shows the visualization of the neutral/negative comments in the form of wordcloud.

```
Sentiment: ['positive' 'neutral']
neutral     335
positive    306
Name: Sentiment, dtype: int64
```

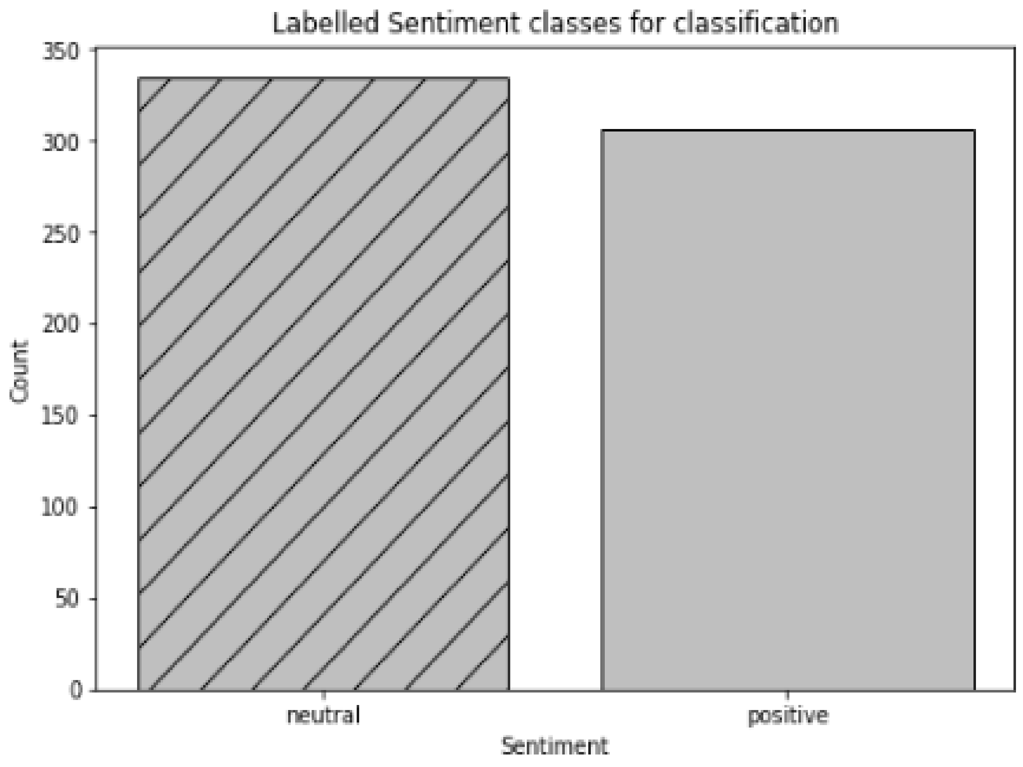


Figure 4.21: Bar graph shows the number of comments of each sentiment class

```
Sentiment: [1 0]
Sentiment: [positive, neutral/negative]
0    18
1    15
Name: Predicted_Sentiment, dtype: int64
```

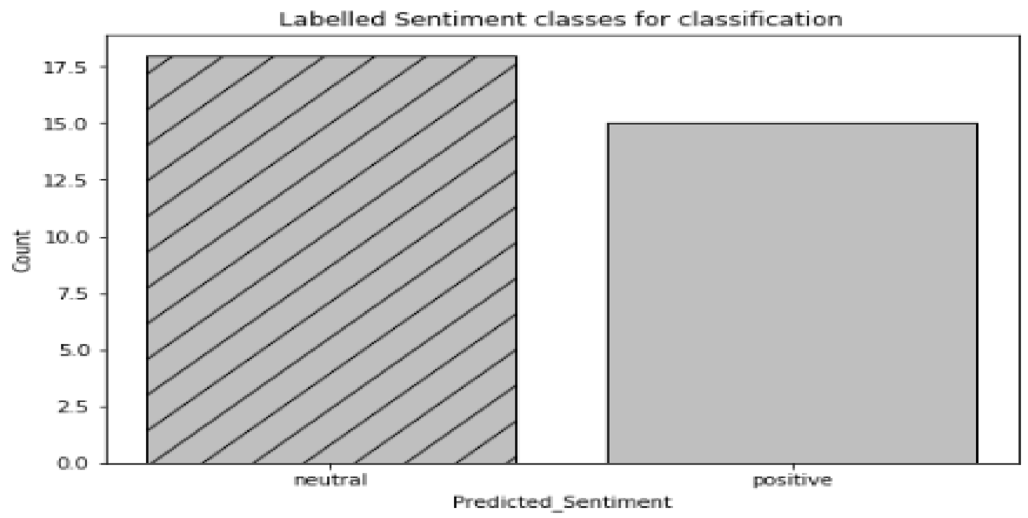


Figure 4.22: Bar graph shows the number of comments of each sentiment class in Course 1

```
Sentiment: [1 0]
Sentiment: [positive, neutral/negative]
0    57
1    36
Name: Predicted_Sentiment, dtype: int64
```

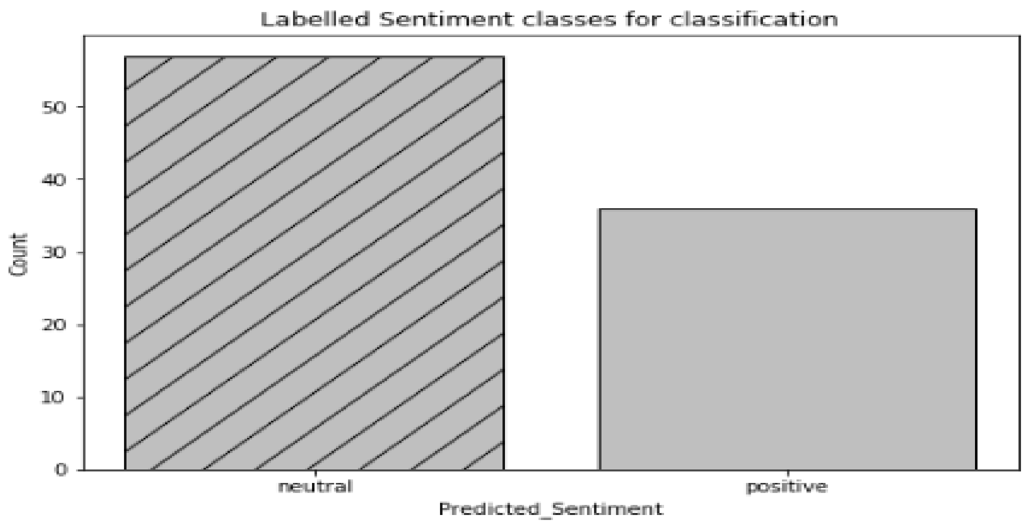


Figure 4.23: Bar graph shows the number of comments of each sentiment class in Course 5

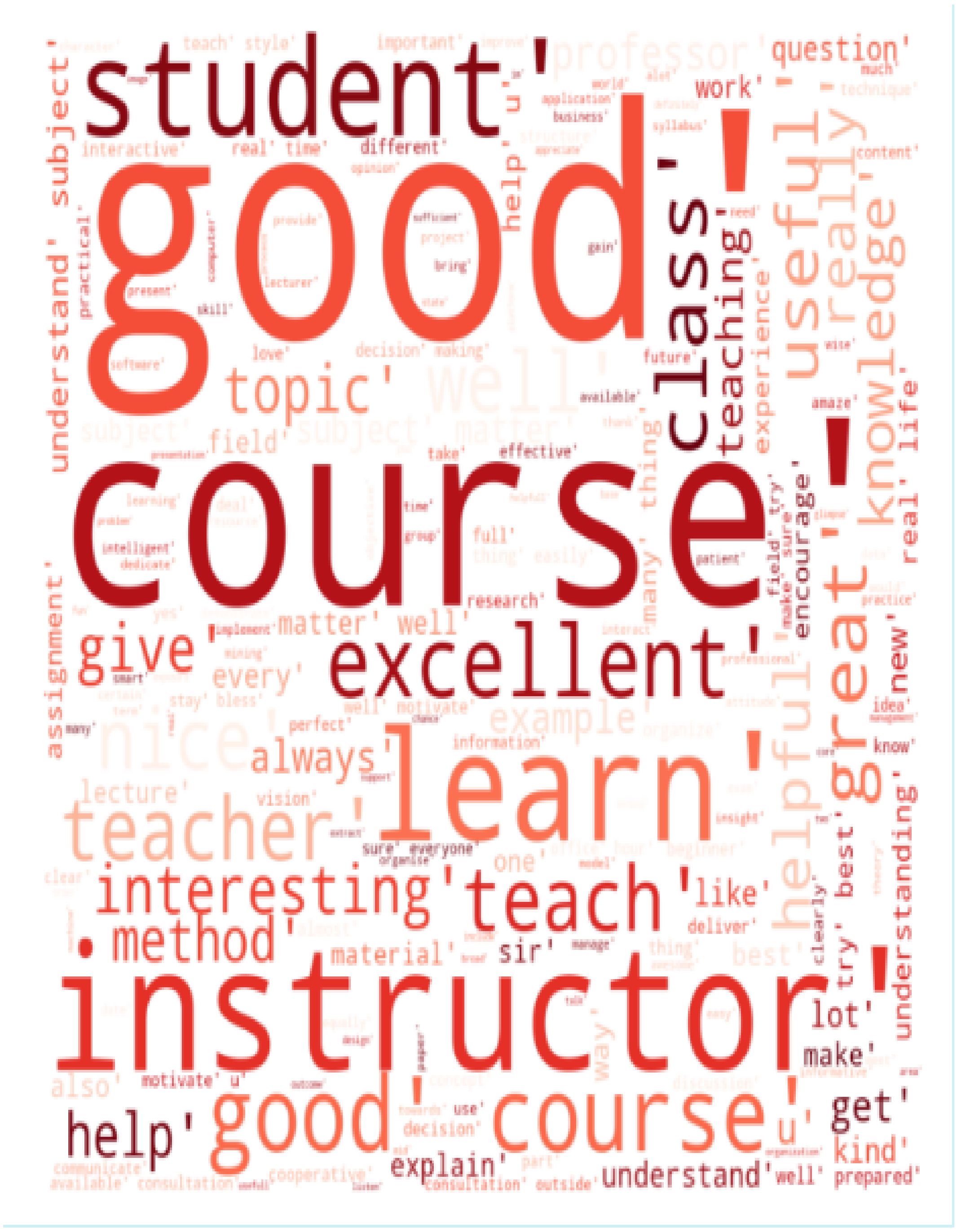


Figure 4.24: Wordcloud shows all the positive comments

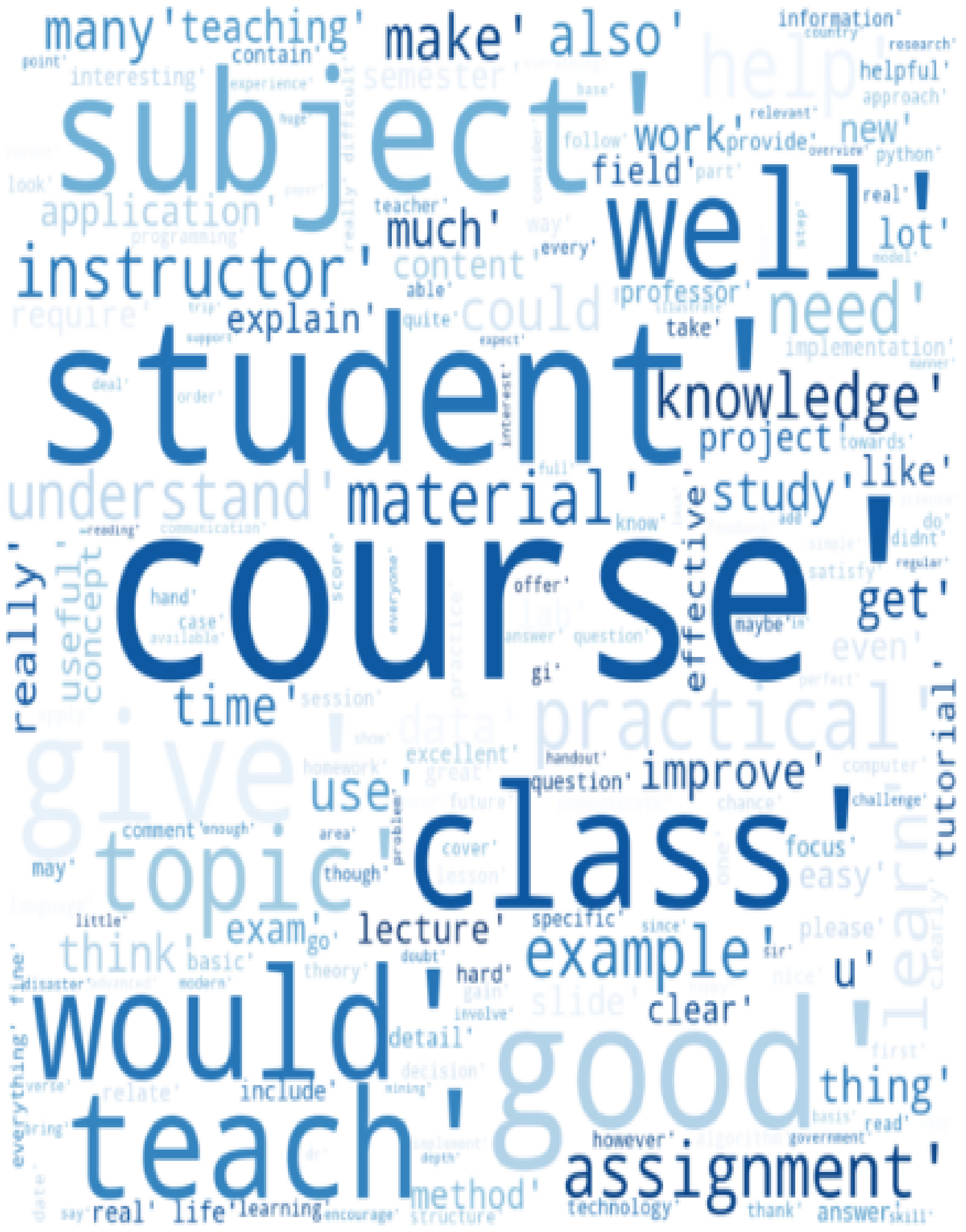


Figure 4.25: Wordcloud shows neutral/negative comments

Chapter 5

Conclusion

This chapter gives conclusion to the research study and describing the further work that can be done.

5.1 Conclusion

Text analytics helps in extracting important information within the text. In this research study, a solution model is built for classifying the sentiment of the text comments received from the student course evaluation form. The student course evaluation form consists of reviews given by students on course, instructor and as well as course material/teaching methods, etc. Reviews being manually read is a time taking process which leads to this research study of finding the sentiment of the comment and visualizing of the comment making it easy for the instructors in going through reviews received.

The data set used for text classification in this research study is received from the Asian Institute of Technology from the year 2015-2019 consisting of a total of 482 comments combining the text feedback given for all the courses on course characteristics, instructor characteristics, resource material/ course delivery methods/ teaching methods. Sentence tokenization is used to split comments as each comment has more than one sentence lead to the total number of text sentences to 641 sentences, an attribute is added named Sentiment which is used in labeling comment with 'positive' or 'neutral/negative' sentiment classes. Pre-processing on the tokenized sentences is done by changing the text to all lower case letters, tokenizing words, normalization of non-numeric sentences by performing stemming and lemmatization and finally removal of stop words. Further in this study, Tf-Idf is used for vectorizing text comments and on these vectorized data different classification algorithms are used in the process of finding the best model for classifying the sentiment.

The classification algorithms used in this study are Naive Bayes, Support Vector Machine, Stochastic Gradient Descent, XGboost, Decision tree (CART). Among which SVM gives an optimal solution with an accuracy of 78.9% with the use of n-grams, compared to the classification report of other algorithms. SVM using n-grams is also applied for testing on classifying the sentiment of comments on individual courses. Course 1 and Course 5 having the most comments were chosen giving an accuracy of 75% and 76% which satisfies one of the objectives of this study and other important focus is to visualize the comments, word cloud of the comments with positive sentiment in orange and the negative/neutral sentiment in blue has helped in accomplishing the other objective. Hence, text analytics approach helps in providing proper insights of student comments which timely helps instructors in improving their teaching process.

5.2 Future Work

In this research study classification of sentiment is done only in two different classes, this can be further extended by increasing the classes of sentiment and refining of results using advanced machine learning algorithms and visualization techniques. Future work includes classifying suggestions in a comment and extracting the key topics of improvement. Word embedding can be used for feature extraction and as well as using parts of speech for each comment can help get much understanding of the feature of the text comment, this will help in giving the accurate emotion of the comment rather than just giving the polarity of the comments.

Text comments can also be analyzed using rule-based classifiers which follows any of the following different rules pattern matching, POS tagged and POS tagged extended in finding the better sentiments. The dataset used in this study is only from ICT department of Asian Institute of Technology decreasing possibilities of extracting suggestions or proper sentiment of different courses individually. In future work, collecting datasets from all the departments of AIT and also studying the impact of this study in other universities and other facilities is an interesting future work. Designing a system can also be included in future work for a user-friendly and interactive model for instructors easy to find the sentiment and visualize easily.

References

Aggarwal, C. C., & Zhai, C. (2012). *Mining text data*. Springer Science & Business Media.

Alhija, F. N.-A., & Fresko, B. (2009). Student evaluation of instruction: what can be learned from students' written comments? *Studies in Educational evaluation*, *35*(1), 37–44.

Altrabsheh, N., Cocea, M., & Fallahkhair, S. (2014). Learning sentiment from students' feedback for real-time interventions in classrooms. In *International conference on adaptive and intelligent systems* (pp. 40–49).

Breiman, L. (2017). *Classification and regression trees*. Routledge.

Chen, T., He, T., Benesty, M., Khotilovich, V., & Tang, Y. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1–4.

Clemmensen, L. K. H., Sliusarenko, T., Ersbøll, B. K., & Christiansen, B. L. (2013). Effects of mid-term student evaluations of teaching as measured by end-of-term evaluations: An emperical study of course evaluations. In *5th international conference on computer supported education (csedu 2013).*

Dang, S., & Ahmad, P. H. (2015). A review of text mining techniques associated with various application areas. *International Journal of Science and Research (IJSR)*, *4*(2), 2461–2466.

Devopedia. (2019). *Sentiment analysis process.* https://devopedia.org/sentiment-analysis.

Esparza, G. G., Luna, A. de, Zezzatti, A. O., Hernandez, A., Ponce, J., Álvarez, M., et al. (2017). A sentiment analysis model to analyze students reviews of teacher performance using support vector machines. In *International symposium on distributed computing and artificial intelligence* (pp. 157–164).

Gaikwad, S. V., Chaugule, A., & Patil, P. (2014). Text mining methods and techniques. *International Journal of Computer Applications*, *85*(17).

Gottipati, S., Shankararaman, V., & Lin, J. R. (2018, Jun 04). Text analytics approach to extract course improvement suggestions from students' feedback. *Research and Practice in Technology Enhanced Learning*, *13*(1), 6.

Gutiérrez, G., Canul-Reich, J., Zezzatti, A. O., Margain, L., & Ponce, J. (2018). Mining: Students comments about teacher performance assessment using machine learning algorithms. *International Journal of Combinatorial Optimization Problems and Informatics*, *9*(3), 26–40.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hansen, C. D., & Johnson, C. R. (2011). *Visualization handbook*. Elsevier.

Hossin, M., & Sulaiman, M. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), 1.

JV Bossche, A. G. O. G., Thomas J Fan, et al. (2019). *Skikit learn developers.* https://scikit-learn.org/stable/documentation.html.

Kao, A., & Poteet, S. R. (2007). *Natural language processing and text mining*. Springer Science & Business Media.

Kaushik, A., & Naithani, S. (2016). A comprehensive study of text mining approach. *International*

Journal of Computer Science and Network Security (IJCSNS), *16*(2), 69.

McCallum, A., Nigam, K., et al. (1998). A comparison of event models for naive bayes text classification. In *Aaai-98 workshop on learning for text categorization* (Vol. 752, pp. 41–48).

Mirończuk, M. M., & Protasiewicz, J. (2018). A recent overview of the state-of-the-art elements of text classification. *Expert Systems with Applications*, *106*, 36–54.

Monkeylearn, I. (2019). *Sentiment analysis example.* https://monkeylearn.com/sentiment-analysis-examples/.

Mueller, A. (2019). *Word cloud.* https://amueller.github.io/word$_{c}loud/$.

Newman, H., & Joyner, D. (2018). Sentiment analysis of student evaluations of teaching. In *International conference on artificial intelligence in education* (pp. 246–250).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, *12*(Oct), 2825–2830.

Petersen, R. (2018). *6 essential steps to the data mining process.* https://barnraisersllc.com/2018/10/data-mining-process-essential-steps/.

Rossum, G. (1995). Python reference manual.

Rouse, M. (2018). *Text mining/text analytics.* https://searchbusinessanalytics.techtarget.com/definition/text-mining/.

Scikitlearn. (2019). *Support vector machine.* https://scikit-learn.org/stable/modules/svm.html.

Sliusarenko, T., Clemmensen, L. K. H., & Ersbøll, B. K. (2013). Text mining in students' course evaluations: Relationships between open-ended comments and quantitative scores. In *5th international conference on computer supported education (csedu 2013).*

Sunasra, M. (2017). *Performance measure for classification problems.* https://medium.com/thalus-ai/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b/.

Takamatsu, K., Kozaki, Y., Kishida, A., Bannaka, K., Mitsunari, K., & Nakata, Y. (2018). Analyzing students'course evaluations using text mining: Visualization of open-ended responses in a co-occurrence network. *PEOPLE: International Journal of Social Sciences*, *4*(3).

Text classification. (2019). https://monkeylearn.com/text-classification/.

Tosi, S. (2009). *Matplotlib for python developers*. Packt Publishing Ltd.

Ullah, M. A. (2016). Sentiment analysis of students feedback: A study towards optimal tools. In *2016 international workshop on computational intelligence (iwci)* (pp. 175–180).

Zhang, H. (2004). The optimality of naive bayes. *AA*, *1*(2), 3.