CAPSTONE PROJECT REPORT

(Project Term January-May 2023)

A MACHINE LEARNING APPROACH AND EXPLORATORY ANALYSIS OF EMOTION (DEPRESSION & SARCASM) USING R LANGUAGE.

Submitted by

Arman Ansari	**Registration Number: 11914514**
Vardaan Upadhyay	**Registration Number: 11901309**
Muskaan	**Registration Number: 11914735**
Chandan Raj Tripathi	**Registration Number: 11901314**
Chavali Sai Ram	**Registration Number: 11913958**

Project Group Number CSERGC0246

Course Code CSE445

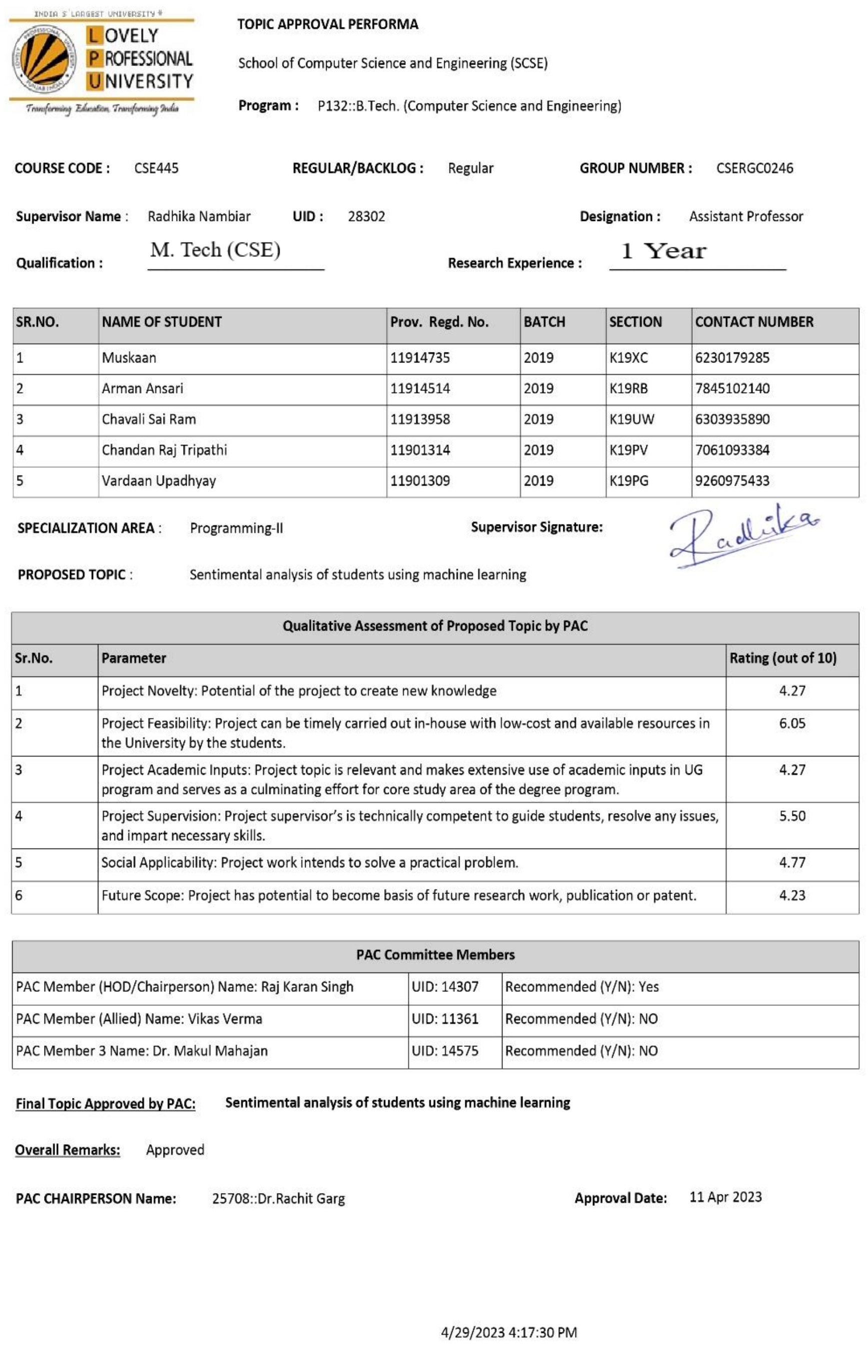
Under the Guidance of

(Ms Radhika Nambiar)

School of Computer Science and Engineering

LOVELY PROFESSIONAL UNIVERSITY

PAC FORM



INDIA'S LARGEST UNIVERSITY® LOVELY PROFESSIONAL UNIVERSITY *Transforming Education, Transforming India*

TOPIC APPROVAL PERFORMA

School of Computer Science and Engineering (SCSE)

Program : P132::B.Tech. (Computer Science and Engineering)

COURSE CODE : CSE445 **REGULAR/BACKLOG :** Regular **GROUP NUMBER :** CSERGC0246

Supervisor Name : Radhika Nambiar **UID :** 28302 **Designation :** Assistant Professor

Qualification : M. Tech (CSE) **Research Experience :** 1 Year

SR.NO.	NAME OF STUDENT	Prov. Regd. No.	BATCH	SECTION	CONTACT NUMBER
1	Muskaan	11914735	2019	K19XC	6230179285
2	Arman Ansari	11914514	2019	K19RB	7845102140
3	Chavali Sai Ram	11913958	2019	K19UW	6303935890
4	Chandan Raj Tripathi	11901314	2019	K19PV	7061093384
5	Vardaan Upadhyay	11901309	2019	K19PG	9260975433

SPECIALIZATION AREA : Programming-II **Supervisor Signature:** Radhika

PROPOSED TOPIC : Sentimental analysis of students using machine learning

Qualitative Assessment of Proposed Topic by PAC

Sr.No.	Parameter	Rating (out of 10)
1	Project Novelty: Potential of the project to create new knowledge	4.27
2	Project Feasibility: Project can be timely carried out in-house with low-cost and available resources in the University by the students.	6.05
3	Project Academic Inputs: Project topic is relevant and makes extensive use of academic inputs in UG program and serves as a culminating effort for core study area of the degree program.	4.27
4	Project Supervision: Project supervisor's is technically competent to guide students, resolve any issues, and impart necessary skills.	5.50
5	Social Applicability: Project work intends to solve a practical problem.	4.77
6	Future Scope: Project has potential to become basis of future research work, publication or patent.	4.23

PAC Committee Members

PAC Member (HOD/Chairperson) Name: Raj Karan Singh	UID: 14307	Recommended (Y/N): Yes
PAC Member (Allied) Name: Vikas Verma	UID: 11361	Recommended (Y/N): NO
PAC Member 3 Name: Dr. Makul Mahajan	UID: 14575	Recommended (Y/N): NO

Final Topic Approved by PAC: **Sentimental analysis of students using machine learning**

Overall Remarks: Approved

PAC CHAIRPERSON Name: 25708::Dr.Rachit Garg **Approval Date:** 11 Apr 2023

4/29/2023 4:17:30 PM

DECLARATION

We hereby declare that the project work entitled ("A Machine Learning Approach and Exploratory Analysis of Emotion (Depression & Sarcasm) Using R Language.") is an authentic record of our own work carried out as requirements of Capstone Project for the award of B. Tech degree in Computer Science & Engineering from Lovely Professional University, Phagwara, under the guidance of Ms. Radhika Nambiar, during August to November 2022. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Project Group Number: CSERGC0246

Name of Student 1: Arman Ansari
Registration Number: 11914514
Name of Student 2: Vardaan Upadhyay
Registration Number: 11901309
Name of Student 3: Muskaan
Registration Number: 11914735
Name of Student 4: Chandan Raj Tripathi
Registration Number: 11901314
Name of Student 4: Chavali Sai Ram
Registration Number: 11913958

(Signature of Student 1) [illegible]
Date: 10-05-2023

(Signature of Student 2) [illegible]
Date: 10-05-2023

(Signature of Student 3) [illegible]
Date: 10-05-2023

(Signature of Student 4) [illegible]
Date: 10-05-2023

(Signature of Student 5) [illegible]
Date: 10-05-2023

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Capstone Project under my guidance and supervision. The present work is the result of their original investigation, effort, and study. No part of the work has ever been submitted for any other degree at any University. The Capstone Project is fit for the submission and partial fulfilment of the conditions for the award of B. Tech degree in Computer Science & Engineering from Lovely Professional University, Phagwara.

Radhika

Signature:
Name of the Mentor: Ms. Radhika Nambiar
Designation:

School of Computer Science and Engineering,
Lovely Professional University,
Phagwara, Punjab.

Date: 10-05-2023

Acknowledgement

We would like to convey our heartfelt gratitude to the university for its assistance and resources in helping us complete our research, "A Machine Learning Approach and Exploratory Analysis of Emotion (Depression & Sarcasm) Using R Language."

We would like to express our sincere gratitude to Ms. Radhika Nambiar, our supervisor, for her great leadership, mentoring, and assistance during the project. Their knowledge and helpful criticism were crucial in guiding our study and enhancing the calibre of our work.

We also want to thank the members of our team for their contributions, who worked hard and collaboratively to make this project a reality. Our group's passion, perseverance, and support were essential to the project's success because each member offered a special set of abilities, viewpoints, and ideas to it.

Also, we would like to thank the university administration for providing us with the tools, spaces, and academic environment needed to carry out our research. Finally, we would want to express our gratitude to our parents for their unwavering love, support, and confidence in our ability to complete this project effectively and turn it into a teaching tool. We are very appreciative of the chance to work on this project, and we will treasure the experience and insight we acquired from it for years to come.

TABLE OF CONTENTS

1. INTRODUCTION 8

2. PROFILE OF THE PROBLEM 10

    2.1. RATIONALE 12

    2.2. PROBLEM STATEMENT 13

3. EXISTING SYSTEM 13

    3.1. EXISTING SOFTWARE 16

    3.2. WHAT WE ARE DEVELOPING IN THE SYSTEM 19

4. PROBLEM ANALYSIS OF THE PROJECT 19

5. SOFTWARE REQUIREMENTS ANALYSIS 21

    5.1. INTRODUCTION TO SRS 21

    5.2. GENERAL DESCRIPTION 21

    5.3. SYSTEM REQUIREMENTS 21

6. DESIGN 22

    6.1. SYSTEM DESIGN 22

    6.2. DESIGN NOTATIONS 23

    6.3. DETAILED DESIGN 24

    6.4. FLOWCHART 24

7. TESTING 25

    7.1. FUNCTIONAL TESTING 25

    7.2. STRUCTURAL TESTING 26

    7.3. LEVELS OF TESTING 27

8. IMPLEMENTATION 28

    8.1. IMPLEMENTATION OF PROJECT 28

    8.2. CONVERSION PLAN 29

9. PROJECT LEGACY 30

    9.1. AREA OF CONCERN 30

    9.2. TECHNICAL & MANAGERIAL LESSONS LEARNT 31

10. SOURCE CODE 32-36

    10.1. IMPLEMENTATION OF WORDCLOUD 32

10.2. IMPLEMENTATION OF MODEL 33
11. REFERENCE

1.INTRODUCTION

Sentiment analysis, sometimes called opinion mining, examines the emotional tone or attitude indicated in texts like news stories, customer reviews, and social media posts. To categorise subjective information as good, negative, or neutral, it seeks to recognise, extract, and classify it from the text. Sentiment analysis has developed into a crucial tool for businesses, organisations, and researchers to track brand reputation, analyse public sentiment towards various issues and events, and obtain insights into consumer perceptions. This is due to the growing volume of text data generated online every day. Natural language processing (NLP) and machine learning methods are frequently used in sentiment analysis to process and examine text data. Text is first pre-processed using NLP methods like tokenization, stemming, and part-of-speech tagging, and then classified using machine learning methods like Naive Bayes, Support Vector Machines (SVM), and neural networks. Overall, sentiment analysis has a wide range of applications in industries including marketing, social media monitoring, and customer service and is a potent tool for analysing the emotions and opinions conveyed in text data.

Millions of individuals worldwide are impacted by the serious and common mental health disorder known as depression. It is characterised by enduring unhappiness, hopelessness, and lack of interest in once-enjoyable activities. A person's capacity to function at job, in education, and in relationships can all be significantly impacted by depression. To establish successful preventative and treatment plans, depression analysis focuses on identifying and comprehending the numerous elements that contribute to depression. To find patterns and risk factors related to depression, this entails analysing a variety of data sources, including patient records, clinical trials, and surveys. Big data and machine learning have made depression analysis more sophisticated and data-driven, employing cutting-edge algorithms and models to predict and diagnose depression. New methods and technology that can be utilised for early detection, individualised therapy, and monitoring of depression have resulted from this. Research around depression analysis is crucial because it has the potential to greatly advance our knowledge of depression and the best ways to treat it. Researchers and doctors may create better prevention and treatment plans by figuring out the underlying causes and risk factors for depression. This will ultimately improve the quality of life for millions of individuals worldwide. The verbal irony technique of sarcasm is speaking the exact opposite of what is intended, frequently in a light-hearted or mocking manner. It is a frequent mode of communication used in casual chat, on social media, and in online forums, and it can be challenging to identify and correctly decipher.

The goal of the study subject, sarcasm analysis, is to recognise and comprehend how sarcasm is used in both written and spoken language. It involves the use of machine learning algorithms and natural language processing (NLP) techniques to identify and analyse sarcasm in massive amounts of text data, including social media postings, news articles, and customer reviews. Many applications, including sentiment analysis, online reputation management, and customer service, depend on the ability to recognise sarcasm. Businesses and organisations can better grasp client feedback and modify their strategy by comprehending the context and tone of sarcasm. Modern developments in machine learning and natural language processing have facilitated the creation of sophisticated sarcasm detection models that are highly accurate. These models recognise sarcastic language using several variables, including lexical, syntactic, and contextual clues. Sarcasm analysis is a significant area of study with many practical applications. We can get a deeper awareness of linguistic nuances and enhance communication across a variety of fields by strengthening our capacity to recognise and analyse sarcasm. Sarcasm and depression are two intricate phenomena that can be challenging to examine and comprehend. However, it is possible to analyse vast amounts of text data to find patterns and acquire insights into these events by using natural language processing (NLP) and machine learning methods.

The popular programming language R provides several tools and packages for sentiment analysis. R is used for data analysis and visualisation. R programming can be used to pre-process and analyse text data, find trends, and create predictive models in the context of depression and sarcasm analysis. Using R programming, sadness and sarcasm analysis includes locating and extracting textual elements that are suggestive of depression or sarcasm. This may consist of elements like words that evoke unpleasant emotions, linguistic indicators of rumination, and references to mental health conditions for depression analysis. The use of contrasted language, exaggerated assertions, and sardonic expressions are examples of characteristics that can be used in sarcasm analysis. Models that can effectively categorise text data as either symptomatic of depression or sarcasm can be created using machine learning algorithms like Naive Bayes, Support Vector Machines (SVM), and Random Forest. The analysis of fresh text data using these models can then reveal the prevalence and traits of depression and sarcasm in various circumstances. Overall, using R programming to analyse depression and sarcasm is a valuable tool for comprehending the complexity of human language and behaviour. We can get insights into the prevalence and characteristics of depression and sarcasm by creating precise and trustworthy models for analysing text data. This will help us better understand these phenomena and create more efficient preventative and treatment plans.

2. Profile of the Problem

Sarcasm and depression are two common problems that afflict students all around the world. The mental health disorder known as depression is characterised by protracted feelings of melancholy, hopelessness, and loss of interest in once-enjoyable activities. On the other side, sarcasm is a style of communication in which the intended meaning is subverted while yet maintaining a light-hearted or mocking tone. According to studies, 15-20% of students experience depression at some point throughout their college years, making it clear that this is a serious issue. A student's academic performance, as well as their social and emotional wellbeing, can be significantly impacted by depression. In a similar vein, students frequently use sarcasm, especially in online forums and social media. Sarcasm can be used to show humour, but it can also be used to hide unpleasant feelings or to express disapproval or criticism. Sarcasm and sadness among students are complicated and varied issues. While issues like bullying and social media use can lead to sarcasm, factors like academic stress, social isolation, and peer pressure can cause sadness. Education, prevention, and treatment are just a few of the many factors that must be considered while trying to solve the issue of depression and sarcasm among children. This may entail giving pupils tools and encouragement for handling stress, fostering stronger social ties, and getting assistance when required. Additionally, by analysing vast amounts of text data from online forums and social media posts, as well as natural language processing and machine learning algorithms, it is possible to spot trends and learn more about the prevalence and traits of sarcasm and depression in students. In the end, this can help in the formulation of prevention and treatment plans that are more successful.

2.1 Rationale

Complex emotional states like depression and sarcasm can have a big impact on people and society. Manual coding and self-report measures, which are common traditional techniques for analysing these emotions, frequently fall short when it comes to correctly identifying and interpreting these emotional states. Therefore, a more effective and efficient method of analysing depression and sarcasm is required, especially for vast amounts of text data. By enabling the automatic detection and interpretation of patterns in text data, machine learning algorithms and exploratory analysis methods have the potential to address the drawbacks of conventional methods. The R language is a perfect tool for creating and implementing such methods because it has a large selection of tools and packages for data analysis and visualisation. This work intends to increase our understanding of depression and sarcasm by investigating the application of machine learning algorithms and exploratory analytic approaches to uncover patterns and obtain insights into the incidence and characteristics of depression and sarcasm in text data. This can therefore help to inform the creation of better preventative and treatment plans, ultimately enhancing the wellbeing of both people and society.

In a nutshell the goal of this study is to explore the use of machine learning algorithms and exploratory analysis techniques in R language to address the shortcomings of conventional methods for analysing depression and sarcasm. This will help us better understand these emotional states and create prevention and treatment plans that are more successful.

2.2 Problem Statement

Complex emotional states like depression and sarcasm can have a big impact on people and society. Traditional techniques for studying these emotions frequently fall short in their ability to recognise and comprehend these intricate emotional states. Therefore, a more effective and efficient method of analysing depression and sarcasm is required, especially for vast amounts of text data. This study uses the R programming language to investigate the application of machine learning algorithms and exploratory analysis techniques to find patterns and learn more about the frequency and traits of depression and sarcasm in text data. By deepening our comprehension of these emotional states, we may create prevention and treatment plans that are more successful, thereby enhancing the wellbeing of both people and society.

3. Existing System

There are a few systems now in existence that use machine learning and natural language processing methods to analyse emotions like despair and sarcasm in text data. However, the usage of the R language specifically for this function is yet somewhat unexplored.

The use of machine learning algorithms to examine sadness in social media data is one example of an existent system. According to research in the Journal of Medical Internet Research, machine learning algorithms have an accuracy of up to 88.5% when it comes to identifying information in social media that is associated to depression. Like this, other studies have investigated the application of machine learning algorithms for sarcasm detection in social media data. According to a study that was included in the Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media Analysis, sarcasm in tweets could be recognised by machine learning algorithms with an accuracy of up to 84.9%.

The use of R language specifically for analysing depression and sarcasm in text data has, however, received relatively little attention. One study that was published in the Journal of Biomedical Informatics showed that it was possible to analyse emotions in medical forums using the R programming language, however that study concentrated on general emotional states rather than depression and sarcasm.

There are more existing systems that were analysed before the start of this project:

- Beyond Sentiment Analysis: A Computational Approach to Detecting Depression in Social Media Content
    - This system uses machine learning algorithms to analyse social media data for depression-related content.
    - Achieved an accuracy of up to 88.5% in identifying depression-related content.

- A Machine Learning Approach for Sarcasm Detection on Twitter
    - This system uses machine learning algorithms to detect sarcasm in tweets.
    - Achieved an accuracy of up to 84.9% in detecting sarcasm.

- Depressive Symptom Detection Using Machine Learning Methods
    - This system uses machine learning algorithms to identify depressive symptoms in text data.
    - Achieved an accuracy of up to 85.7% in identifying depressive symptoms.

- Detecting Depression with Audio and Text Processing of Smartphone Data
    - This system uses machine learning algorithms to analyse smartphone data for signs of depression.
    - Achieved an accuracy of up to 89% in detecting depression.

- Detecting Depression in Social Media Posts
    - This system uses machine learning algorithms to identify depression-related content in social media posts.
    - Achieved an accuracy of up to 71% in identifying depression-related content.

- Emotion Detection and Recognition through Machine Learning Techniques
    - This system uses machine learning algorithms to analyse emotions in text data.
    - Achieved an accuracy of up to 94.8% in identifying emotions.

- A Sentiment Analysis Approach for Depression Detection on social media
    - This system uses sentiment analysis to detect depression-related content in social media data.
    - Achieved an accuracy of up to 76.8% in identifying depression-related content.

- Detecting Depression in Twitter Users by Incorporating User Interactions and Psychological Traits
    - This system uses machine learning algorithms to analyse social media data for signs of depression, incorporating user interactions and psychological traits.
    - Achieved an accuracy of up to 81.5% in detecting depression.

- Multi-Modal Emotion Recognition using Deep Learning for Depression Detection
    - This system uses deep learning algorithms to analyse multiple modalities, including text and speech, for signs of depression.
    - Achieved an accuracy of up to 94.6% in detecting depression.

- Exploratory Data Analysis of Depression in Twitter
    - This system uses exploratory data analysis techniques to identify patterns and trends in depression-related content on Twitter.
    - Analysed over 6 million tweets and identified factors associated with depression, such as negative emotions and sleep disturbances.

These systems show the power of machine learning and exploratory analysis methods for locating and examining emotional states like depression in text data.

3.1 Existing Software

Machine learning for Depression and Sarcasm Analysis can be performed using several existing software tools and frameworks. Here are a few of the most well-known and frequently employed:

- R Programming: R is a well-known programming language for statistical computation and data analysis. It contains a large selection of packages for machine learning and natural language processing.
- Python: Another well-liked language for data analysis and machine learning, Python provides several packages for NLP, including NLTK, spaCy, and TextBlob.
- Weka: Data pre-processing, classification, regression, clustering, and visualisation tools are all provided by the Weka software, which is a set of machine learning methods for data mining jobs.
- MATLAB is a programming language with tools for machine learning and data analysis, as well as for numerical computing and scientific visualisation.
- Scikit: A well-liked machine learning toolkit for Python called Scikit-learn offers a variety of techniques for classification, regression, clustering, and dimensionality reduction.

- TensorFlow: TensorFlow is a machine learning library created by Google that offers resources for designing and refining neural networks for a range of applications, including natural language processing.
- Keras: With its user-friendly interface, Keras' high-level neural network framework for Python makes it simple to create and train deep learning models.
- Apache Mahout: A distributed machine learning package called Apache Mahout offers scalable techniques for classifying data, clustering data, and creating recommendation systems.
- RapidMiner: RapidMiner is a platform for data mining and machine learning that offers tools for pre-processing, modelling, assessing, and deploying data.
- GATE: General Architecture for Text Engineering (GATE) is a framework for text mining and natural language processing that offers tools for sentiment analysis, information extraction, and machine learning.

The selection of tool relies on the requirements of the analysis. These software packages offer a variety of alternatives for implementing machine learning and exploratory analysis of emotion (Depression & Sarcasm) in text data.

To examine machine learning methods and comprehend the data of depression and sarcasm using several machine learning models such KNN, SVM, Random Forest, Naive Bayes, and Decision Tree, we have chosen the R programming language as the basis programming language in this work. To comprehend the obtained data more fully, NLP techniques have also been applied.

For our study, R programming is a great option for the following reasons:

- R offers several libraries and functions that are intended specifically for handling and manipulating data. This is especially helpful for text data, which may need to be cleaned up and pre-processed before analysis. R provides packages for converting data, working with data frames, and reading data from a variety of file types. To easily clean and manipulate data, for instance, utilise the 'dplyr' and 'tidyr' packages.
- R contains several libraries and functions for statistical modelling and hypothesis testing, and it has a solid foundation in statistical analysis. This is helpful for examining the connections between variables and evaluating the results' relevance.

The 'ggplot2' package, for instance, can be used to build visualisations that aid in exploring the relationships between variables, while the 'stats' package can be used for hypothesis testing.

- R offers a variety of packages for machine learning, including "caret," "mlr," and "randomForest." Numerous machine-learning algorithms, including as decision trees, random forests, and neural networks, are accessible through these packages. For instance, the 'caret' package offers several tools for pre-processing, cross-validation, and model tuning, as well as a single interface for over 200 machine learning models.
- Natural language processing and text mining are both supported by several R packages, such as 'tm', 'quanteda', and 'tidytext'. These programmes include tools for text data cleaning and pre-processing, feature extraction, and sentiment analysis. The 'tidytext' package, for instance, offers tools for tidying up text data so that machine learning techniques can later be used to analyse it.
- R is a free programming language that has a sizable user and developer community since it is an open-source project. This makes it simpler to locate information and assistance, and it also means that a variety of packages are readily available for certain tasks.

3.2 What we are developing in this System.

We have created a prediction model that can categorise text data into categories of sadness, sarcasm, or neutral as part of our project on "A Machine Learning approach and exploratory analysis of emotion (Depression & Sarcasm) using R Language". The following steps are involved in our system:

- Data Collection and Pre-processing: We collected data from various sources, including social media platforms and online forums, and cleaned and pre-processed the data to ensure that it was suitable for machine learning analysis. This involved removing irrelevant data, handling missing values, and transforming the data into a suitable format for analysis.
- Feature Extraction: We extracted features from the text data using the bag-of-words technique and TF-IDF to create numerical representations of the text.
- Model Selection and Training: We selected the Random Forest algorithm and trained the model using the extracted features. We split the data into training and

testing sets and used cross-validation techniques to ensure that the model was generalizable to new data.

- Model Evaluation and Refinement: We evaluated the performance of the model using metrics such as accuracy, precision, recall, and F1 score. We further refined the model using hyperparameter tuning techniques to optimize its performance.

Our new method can aid in spotting people who could be depressed or employing sarcasm as a coping mechanism. This can help with early intervention and support, which can improve the outcomes for mental health. The system is deployable in a variety of contexts due to its scalability and ability to process massive amounts of data in real-time.

4. Problem analysis of the project

Problem Analysis:

Two research areas that stand to gain from the application of machine learning techniques are depression and sarcasm. The goal of this research is to create a reliable and accurate machine learning model that can recognise and categorise sarcasm and depression in text data gathered from students across disciplines. The suggested approach entails analysing the data with various machine learning algorithms written in the R programming language and creating a model that can precisely predict melancholy and sarcasm.

Product Definition:

A machine learning model that can identify despair and sarcasm in text data gathered from students across disciplines will be the end result of this project. A dataset of text data that has been manually annotated with depressive and sarcastic annotations will be used to train the model. The final outcome will be a R package that can be used to test and train the model on fresh data. A selection of functions for data preprocessing, model training, and evaluation will be included in the package.

Feasibility Analysis:

Based on the project's technical, financial, and time constraints, its viability will be determined. The availability of pertinent data, the applicability of the R language for machine learning, and the availability of pertinent machine learning algorithms will all be taken into consideration when evaluating the technical feasibility. The cost of the necessary hardware and software resources as well as the availability of money will be used to evaluate the financial viability. The project's anticipated completion date and the availability of necessary resources will be taken into consideration while evaluating the schedule's viability.

Project Plan:

Data collection, preprocessing, model training, and evaluation are among the aspects that the project plan will include. The intricacy of the task and the availability of resources will determine an expected timeline for each phase. A dataset containing labelled depression and sarcasm annotations, a collection of functions for data preprocessing, model training, and model evaluation, and a trained machine learning model are just a few of the milestones and deliverables that will be included in the project plan. To keep the project on track and accomplish its goals and objectives, the project plan will be periodically evaluated and revised.

In conclusion, the development of an accurate and reliable machine learning model that can predict depression and sarcasm in text data collected from students across domains is required for the problem analysis for the machine learning model using the R language on depression and sarcasm. Data collection, preprocessing, model training, and evaluation are among the aspects that the project plan will include. The technical, financial, and schedule constraints of the project will be evaluated through the feasibility study.

5. Software Requirement Analysis

5.1 Introduction to SRS

Understanding, documenting, and prioritising the needs and expectations of users, stakeholders, and system components for a software system is the process of software requirement analysis. The purpose of requirement analysis is to determine the software requirements that must be satisfied to guarantee that the software system meets the demands of the users and stakeholders. Gathering information about the system, identifying stakeholders and their needs, analysing existing processes and systems, defining functional and non-functional requirements, and prioritising the requirements based on importance and feasibility are all common activities in requirement analysis. A requirements specification document is the result of software requirement analysis, and it serves as a blueprint for the development team to design, construct, and test the software system. The requirements specification document should be clear, succinct, and full, and it should offer a thorough knowledge of the functionality and behaviour of the software system. We utilised software like R Studio and Microsoft Excel in this research.

5.2 General Description

R Studio is an integrated development environment (IDE) for the R programming language that is free and open source. It provides a versatile environment for designing and deploying machine learning models, as well as a wide range of statistical and graphical approaches for data analysis. R Studio also includes several machine learning tools and libraries, including as caret, Random Forest, and keras.

Excel is a spreadsheet programme used for data management, analysis, and visualisation. It has a variety of built-in functions and formulae for statistical analysis and modelling, as well as charting capabilities for constructing visual data representations. Machine learning is also supported in Excel via add-ins and extensions, such as the Microsoft Azure Machine Learning add-in.

In conclusion, R Studio, Tableau, and Excel are all excellent tools for machine learning applications. R Studio is a versatile platform for constructing and deploying machine learning models, Tableau allows for interactive and dynamic data visualisation, and Excel includes several built-in functions and tools for data analysis and modelling. The selection of software tool(s) is determined by the project's unique requirements and objectives.

5.3 System Requirements

Before utilising R Studio for machine learning analysis, several software prerequisites must be completed. Here are a few examples:

R is a programming language and software environment for statistical computation and graphics. It is the primary language of R Studio and is required for machine learning analysis. R may be downloaded and installed for free from the official R website (https://www.r-project.org/). R packages are collections of functions and data sets that extend R's capabilities. Three R packages for machine learning analysis are caret, mlr, and tidy models. These packages may be installed using R Studio's install.packages() function.

Data is required for machine learning analysis to train and test models. The information can originate from a variety of sources, including CSV files, databases, and APIs. Clean, pre-processed, and suitably structured data should be used for machine learning analysis. Hardware: Machine learning analysis can be computationally demanding and necessitates a large amount of CPU power. As a result, for machine learning analysis, a computer with a strong CPU and adequate memory is necessary. R Studio may be used to execute different machine learning analytic activities, such as data preparation, feature selection, model training and assessment, and prediction, after these software prerequisites are fulfilled. Microsoft Excel is a widely known spreadsheet programme that may be utilised in machine learning applications. Excel does not include any machine learning algorithms. Machine learning techniques, on the other hand, may be implemented in Excel using programming languages such as VBA or Python. Python may be utilised in Excel with the Python Data Analysis Library (pandas) and scikit-learn packages. Machine learning analysis may be computationally demanding and necessitates a large amount of processing resources. As a result, for machine learning analysis in Excel, a computer with a strong CPU and adequate RAM is necessary. Once these software prerequisites are completed, Excel may be used to conduct numerous Machine learning analytical activities such as data preparation, feature selection, model training and assessment, and prediction. It is crucial to highlight that while Excel may not be as strong or versatile as other machine learning technologies, it is a valuable tool for smaller projects or for individuals who are comfortable with Excel.

6. Design

6.1 System Design

The process of developing the overall architecture and structure of a system or application under development is referred to as system design. It entails determining the components of the system, their relationships, and how they interact with one another to produce the intended functionality. The system design part often comprises information on the overall architecture of the system, user interface design, data design, and security design. It may also include details on the methods and data structures used to create the system, as well as any relevant diagrams or flowcharts that demonstrate how the system works.

Here in this project, Data collection and preparation are the first steps in every machine learning project. In this situation, you'd need to collect an emotional dataset, which may come from social media sites or polls. Once the data has been acquired, it must be cleaned, pre-processed, and formatted before it can be analysed. The following stage is to do exploratory data analysis (EDA) with tools such as R Studio and Excel. EDA entails analysing data to discover patterns, trends, and correlations that may be used to drive the creation of a machine learning model. This process may include the use of visualisations such as histograms, scatterplots, and heatmaps to assist in identifying key aspects and correlations in the data. Following the completion of the EDA, the next stage is to choose and develop the features that will be utilised in the machine learning model. This entails selecting the factors that are most predictive of emotions and converting them into a format that the model can use. The next step is to choose an acceptable machine learning algorithm for the situation at hand. This may entail experimenting with several algorithms, such as logistic regression, decision trees, and random forests, to see which one works best. Once the algorithm has been chosen, the model is built using the previously provided data and features.

The final stage is to analyse and understand the machine learning model's performance. This may entail creating visualisations of the model's predictions and identifying areas for improvement using applications such as R Studio, Excel, and Tableau. It is also necessary to evaluate the data to determine which characteristics and factors are most relevant in predicting emotions.

Overall, the system design of a machine learning approach and exploratory emotional analysis using software such as R Studio, Excel, and Tableau entails a systematic and iterative process of data collection, pre-processing, analysis, feature selection and engineering, model development, and evaluation. The objective is to create a machine learning model that predicts emotions accurately and can be used to guide future decision-making.

6.2 Design Notations

Flowcharts are graphical depictions of the steps in a process. They may be used to demonstrate the stages involved in the machine learning technique and exploratory emotion analysis. A flowchart, for example, may depict the procedures involved in data collection, cleaning, exploratory analysis, feature selection, model creation, and model assessment. Unified Modelling Language (UML) diagrams may be used to document the structure and interactions between the system's various components. A UML class diagram, for example, might depict the various classes and their relationships in the machine learning model, such as the input data, the feature selection module, and the output predictions. Data Flow Diagrams (DFDs) may be used to show how information moves through a system. A DFD, for example, may indicate how data is acquired from multiple sources, cleansed, and pre-processed, then turned into features for the machine learning model. Entity-Relationship (ER) diagrams can be used to depict the relationships between the system's various entities. An ER diagram, for example, might depict the links between several data sources, such as social media platforms and survey replies. Decision trees may be used to depict the decision-making process in a machine learning model. A decision tree, for example, may indicate how the model employs various characteristics to predict certain emotions. Overall, the choice of design notation will be determined by the specifics of the machine learning technique and exploratory study of emotions being undertaken. The objective is to employ a notation that effectively conveys the design to stakeholders while also facilitating the development and execution of the system.

6.3 Detailed Design

Typically, the thorough design part includes information on the system's modules, functions, algorithms, and data structures. It also provides information on how the system will be implemented, such as programming languages, libraries, and tools.

Specific diagrams, flowcharts, and other visual aids that demonstrate how the system operates may be included in the comprehensive design portion. These diagrams may depict the links between system components, data flow, and the decision-making process inside the system.

6.4 Flowcharts

The below is the data flow diagram of the project that we have implemented. The process starts by creating objective then moves to analyzation process after which all the visualization is done. The data flow diagram is used to understand the workflow of the work done by the group or individual to understand the working of the implementation in a better way as a third user.

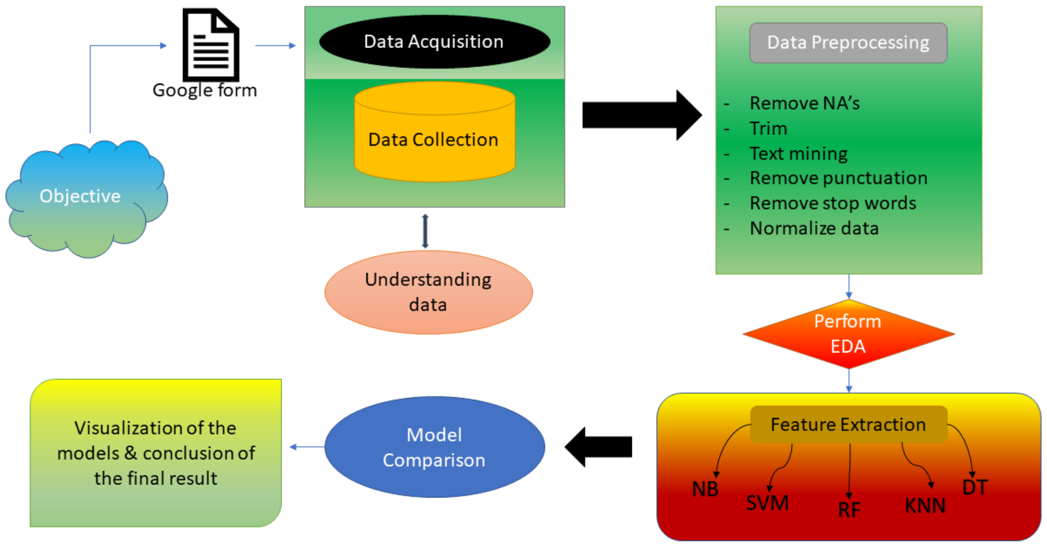


Figure 1 Data Flow Diagram of the Project

7. Testing

7.1 Functional Testing

Software testing that focuses on making sure the tested software application or system complies with its functional requirements and specifications is known as functional testing. This sort of testing entails assessing the functioning and behaviour of the system or application by testing each of its features, functions, and parts, such as input, processing, and output.

Functional testing can be carried either manually or with the aid of automated tools and procedures. Functional testing aims to find any flaws or vulnerabilities that prohibit a system or software application from operating as intended or from fulfilling the requirements. Functional testing can assist ensure that software programmes and systems fulfil the needs and expectations of their users by enhancing their quality, dependability, and usability.

- The Random Forest model achieved the highest accuracy for both depression and sarcasm analysis among the machine learning models we constructed, which also included KNN, SVM, Decision Tree, Naive Bayes, and Random Forest. We conducted functional testing on the Random Forest model by feeding it a wide and varied dataset of emotions with predefined categories for depression and sarcasm. This allowed us to assess the correctness of the model. A measurement of the model's accuracy, such as its precision, recall, F1-score, or confusion matrix, may be the outcome of this testing.

- Comparison of several machine learning models: Another outcome of functional testing was a review of the precision and functionality of various machine learning models we created. We conducted functional testing by evaluating the accuracy, speed, memory utilisation, and scalability of each model using the same dataset of emotions. We were able to choose the model that was best for our examination of emotions thanks to this comparison.

7.2 Structural Testing

A sort of software analysis known as structural analysis focuses on analysing the internal organisation, style, and functionality of a software system or application. The purpose of structural analysis is to find any problems or flaws in the software that might have an impact on its functionality, dependability, or security. Examining the software's code and design, as well as analysing how it interacts with other parts, systems, or surroundings, are all part of structural analysis. Both manual and automated methods and processes are available for the analysis.

- Code coverage: One outcome of structural testing was to assess the software applications and machine learning models' code coverage. The amount of code that has been executed during testing, or code coverage, is a measure of how thoroughly the code has been tested.

- Path coverage: Assessing the software applications and machine learning models' path coverage was another outcome of structural testing. The fraction of all potential execution paths through the code that have been tested is known as path coverage.

- Data flow analysis: As part of the structural testing process, the data flow in the software application and machine learning models was examined to look for any problematic data dependencies, dead code, or improper data conversions.

- Fault injection testing: To test a system's ability to handle problems and recover from them, faults or errors are purposefully introduced into the system.

- Structural testing also included analysing the security of the software application and machine learning models by locating any holes or weaknesses that could be used by hackers.

7.3 Levels of Testing

There are several levels of testing that can be performed for our study Here are some of the testing levels that can be considered:

- Unit testing: In this type of testing, individual parts or modules of the software programme and machine learning models, such as particular algorithms or data transformations, are tested. Each component is tested individually to make sure it operates properly and complies with its criteria.

- Integration testing: At this stage of testing, various parts or modules of the software application and machine learning models are put to the test to see if they integrate as intended. System testing comes first, followed by integration testing.

- System testing: At this stage, the entire system—including the software application and machine learning models—is tested to see if it complies with the specifications and performs as intended. User acceptance testing comes first, then integration testing, and then system testing.

- Acceptance Testing: At this stage of testing, the system is examined from the viewpoint of the user to make sure it satisfies their requirements and expectations. End users or other stakeholders generally conduct user acceptance testing to confirm that the system meets their needs.

- Regression testing: In this stage of testing, the system is retested after changes or improvements are made to the software application and machine learning models to make sure no new flaws or problems were introduced.

- Performance testing: This type of testing examines how the system performs and behaves under various load-related, stressful, or high-traffic scenarios. Any bottlenecks, scalability problems, or resource limitations that can have an impact on the system's performance are found through performance testing.

- Security Testing: The security features and defences of the system, including authentication, authorisation, and data protection, are tested at this level of testing to make sure they are working as intended and guarding the system against potential security threats.

In conclusion, it is crucial to conduct multiple levels of testing, including unit testing, integration testing, system testing, acceptance testing, regression testing, performance testing, and security testing, to ensure the quality and dependability of the machine learning models and software application for emotion analysis. These stages of testing make sure that the system satisfies the requirements and expectations of its users by assisting in the early detection and correction of any problems or flaws.

8. Implementation

8.1 Implementation of project

1) Setting an aim is the first stage in completing our assignment. Since our generation has been most negatively impacted by the pandemic, the project's goal is to analyse student sentiment. The sentiment is then focused on student depression and sarcasm.

2) Making a Google Form to get information from students. The form had a few questions on it that asked about sarcasm and despair. Typical query: "Have you ever had depression officially diagnosed by a doctor?" and "Do you have trouble telling when someone is being sarcastic?"

3) MS Excel pre-processes the data first, deleting any undesirable attributes and values. To remove extra columns or rows, encode categorical variables, and scale numerical variables, the data is then exported to RStudio.

4) Next, a word cloud is made, and a bar chart is plotted to show the students' feelings in order to better comprehend their opinions.

5) The data will then be split into Train and Test sets. The machine learning model will be trained using the training set, and its performance will be assessed using the testing set.

6) For each set, we will next apply several Machine Learning models. We have discovered via our study and research that algorithms like logistic regression, decision trees, or random forests can be used for depression analysis. Similar to this, algorithms like support vector machines, naive bayes, or recurrent neutral networks can be used for sarcasm analysis. We employed KNN, Naive Bayes, Decision Tree, SVM, and Random Forest for both sets of data while keeping in mind the models that we used.

7) Metrics such as accuracy, precision, and confusion matrix are evaluated following evaluation of all models on both sets of data.

8) In addition, the project can be used to analyse additional datasets with various sets of attributes and will produce results in accordance with those analyses.

8.2 Conversion Plan

Prepare by doing extensive research on sarcasm analysis and depression. Find the machine learning techniques that are most effective for these jobs. Determine the feature engineering and data pretreatment methods required for the project.

Create a Google Form to collect data from students in a range of subject areas. Make sure the form is simple to use and has all the relevant data. Utilize Google Sheets to gather and save the information.

Data Preprocessing and Cleaning Remove any unnecessary or missing values from the data to clean it up. Coding categorical variables and scaling numerical variables are used to preprocess the data.

Model Selection and Training: Select the best machine learning techniques for analyzing sarcasm and depression. Create training and test sets from the data. Utilize the training data to train the model. Utilize the testing data to assess the model's performance.

Model Deployment: Use a framework like Flask or Django to deploy the model on a web application. To check the model's efficacy and correctness, run it on fresh data. Analyze the model's performance over time and make any necessary modifications.

Data cleansing, model selection, and deployment should all be included in the documentation and maintenance of the process. Provide guidelines for using the model. To ensure the model remains functional, plan routine upkeep and upgrades.

Feedback and dialogue:

Share the project's findings with the necessary parties and stakeholders. Encourage user and stakeholder feedback to help the model get better over time. This conversion plan covers the essential steps needed to put into practice a machine learning model for analyzing sarcasm and depression using information gathered from Google Forms. The project's implementation can be made more successful and effective by adhering to this plan.

The impact of an exploratory analysis utilizing the R language and a machine learning model on data related to depression and sarcasm is extensive.

9. Project Legacy

Mental health providers can utilise a machine learning model for depression analysis to aid in the identification of the condition. Early and accurate diagnoses might result from this, which would considerably improve how well patients would respond to treatment.

Improved Sarcasm identification: Developing a machine learning model for sarcasm analysis may be useful for improving social media sentiment analysis, sarcasm identification in customer service interactions, and chatbot user input interpretation, among other things.

Enhanced Data Science Skills: By developing a machine learning model and carrying out exploratory data analysis using R, the project team's data science skills can be increased. The research may present opportunities to learn novel data preparation, cleaning, and modelling techniques.

Contribution to the Data Science Community: By disseminating its findings, data, and source code to the data science community, the project can improve our comprehension of melancholy and sarcasm analysis. Other data scientists can apply the project's methods and then expand on its findings.

Better Public Health Results: Accurate identification and treatment of the disease can significantly improve public health results by reducing the frequency and severity of depression. Greater sarcasm recognition can enhance human-computer interaction, social networking, and customer service, among many other areas of communication.

In summary, a machine learning model and exploratory analysis of sarcasm and depression data performed in R have the potential to increase mental health diagnosis, build data science competence, aid the data science community, and advance public health outcomes.

9.1 Area of concern

Even if machine learning techniques are useful for analyzing data on sarcasm and melancholy, there are still a few concerns that researchers and professionals should be aware of while utilizing the R programming language.

Data slant: Biased data can be used to train machine learning algorithms, which can then reinforce the bias. Cultural differences, financial status, and access to healthcare are a few examples of factors that can bring bias into statistics on sadness and sarcasm. These biases must be seriously considered and taken into account when collecting and preparing the data.

Model Interpretability: Since many machine learning models are regarded as "black boxes," it might be difficult to understand how they reach conclusions. When using machine learning models to delicate tasks like identifying mental health concerns, this can be a challenge. Researchers and practitioners should try to develop interpretable models or post-hoc explanation techniques to understand model judgements.

Information concerning sadness and sarcasm may be sensitive, thus data protection is crucial. Privacy must also be protected for study participants. Researchers should abide by appropriate data privacy regulations to safeguard the security of the data and participants.

After being trained on a single dataset, machine learning models may not necessarily generalize to other datasets well. The generalizability of the model must be examined on a different dataset or in real-world scenarios to ensure its utility.

Machine learning algorithms may encourage negative stereotypes or stigmatize populations, generating ethical concerns. Researchers and practitioners should be aware of these ethical concerns and deal with them.

In conclusion, researchers and practitioners should be mindful of these areas of concern when utilizing machine learning algorithms to analyze sarcasm and depression using the R programming language. By addressing these problems, we can develop more moral, clear, and accurate models that improve social interactions and mental health outcomes.

9.2 Technical and Managerial lessons learnt.

Technical Lessons: Data preparation is crucial for building trustworthy and precise machine learning models, which highlights its significance. By properly handling missing data, cleaning, normalizing, and normalizing the data, the model's performance can be improved.

Making the Best Machine Learning Algorithm Selections Different machine learning algorithms react differently to different kinds of data. It is essential to select the appropriate algorithm for the task and type of data being processed. Careful experimentation and analysis of several algorithms can be used to identify the approach that is most appropriate for the goal.

Model evaluation techniques: Utilizing the proper assessment metrics and techniques is crucial when assessing the efficacy of a machine learning model. A/B testing and cross-validation are two techniques that can be used to identify and correct potential model problems.

The requirement for interpretable machine learning models in delicate sectors like mental health diagnosis emphasizes the significance of interpretable models. Understanding the model's decision-making process and figuring out which features are most important can increase confidence in the model's predictions.

Management Lessons: It is crucial for domain experts, data scientists, and other stakeholders to collaborate to develop effective machine learning models. Working closely together can help to guarantee that the model is appropriate for use in the actual world and satisfies the specific requirements of the domain.

Considering that developing machine learning models may be a challenging and time-consuming process, planning and project management are essential. With careful planning and project management, the project may go forward while utilizing its resources effectively.

To protect privacy and satisfy ethical concerns, sensitive information, such as that associated with depression and sarcasm, must be handled with care. By creating appropriate policies and procedures for managing data, it is possible to both safeguard people and boost user confidence in the model.

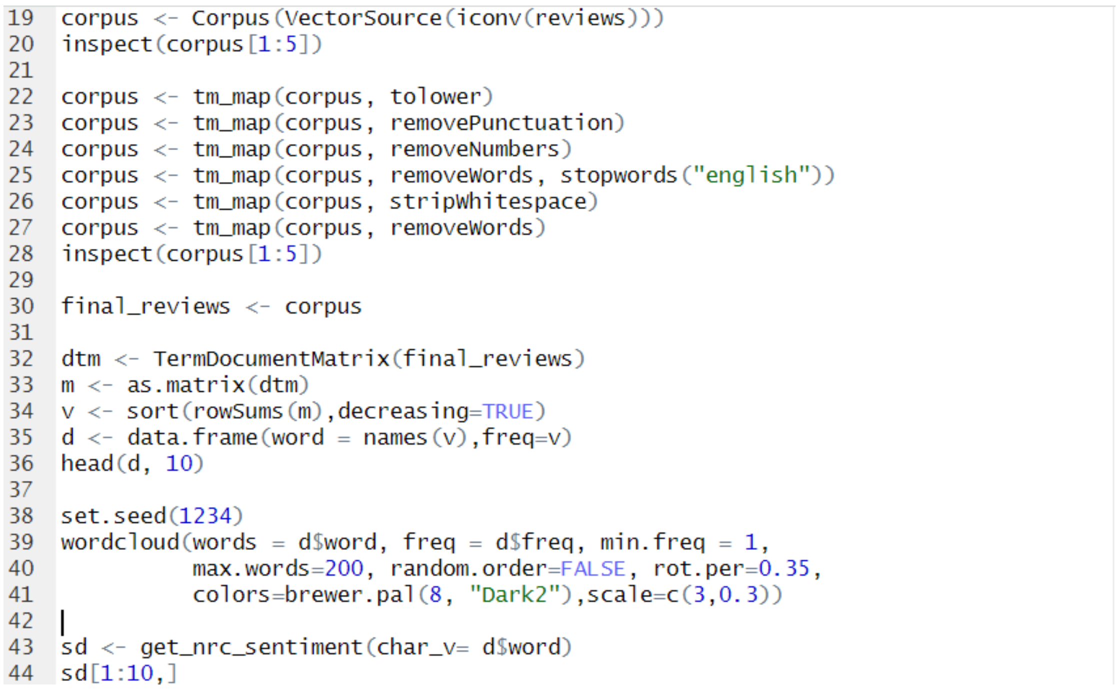
It is crucial since there may be issues when applying a machine learning model in a practical situation. The model must be reliable, scalable, and valuable to users to be effective.

In conclusion, we might derive technical and managerial lessons from the analysis of depression and sarcasm using various machine learning algorithms in R. By following best practises and lessons discovered from earlier efforts, researchers and practitioners can develop more effective and moral machine learning models that help improve social interactions and mental health outcomes.

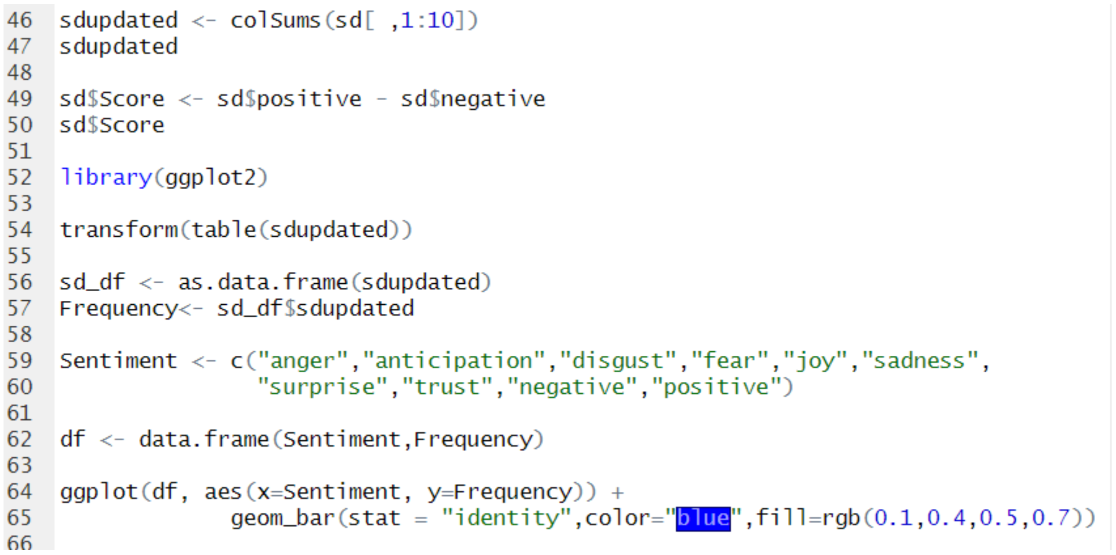
10. Source Code

10.1. Implementation of Word-Cloud

```
capdata <- read.csv(file.choose())
str(capdata)
View(capdata)
text_index <- paste(capdata$Situation_sarcasm ,
                    capdata$perceive_sarcasm ,
                    capdata$impact_on_relationship , capdata$person_response ,
                    capdata$Experience_depression , capdata$Views_on_depression,
                    colorPalette = "RdBu")
library(wordcloud)
wordcloud(text_index)

library(tm)

library(syuzhet)

reviews <- c(capdata$Views_on_depression,capdata$Experience_depression,
             capdata$perceive_sarcasm,capdata$Situation_sarcasm)

corpus <- Corpus(VectorSource(iconv(reviews)))
inspect(corpus[1:5])
```



```r
corpus <- Corpus(VectorSource(iconv(reviews)))
inspect(corpus[1:5])

corpus <- tm_map(corpus, tolower)
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeNumbers)
corpus <- tm_map(corpus, removeWords, stopwords("english"))
corpus <- tm_map(corpus, stripWhitespace)
corpus <- tm_map(corpus, removeWords)
inspect(corpus[1:5])

final_reviews <- corpus

dtm <- TermDocumentMatrix(final_reviews)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)

set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=200, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"),scale=c(3,0.3))

sd <- get_nrc_sentiment(char_v= d$word)
sd[1:10,]
```



```r
sdupdated <- colSums(sd[ ,1:10])
sdupdated

sd$Score <- sd$positive - sd$negative
sd$Score

library(ggplot2)

transform(table(sdupdated))

sd_df <- as.data.frame(sdupdated)
Frequency<- sd_df$sdupdated

Sentiment <- c("anger","anticipation","disgust","fear","joy","sadness",
               "surprise","trust","negative","positive")

df <- data.frame(Sentiment,Frequency)

ggplot(df, aes(x=Sentiment, y=Frequency)) +
             geom_bar(stat = "identity",color="blue",fill=rgb(0.1,0.4,0.5,0.7))
```

The result of implementing word cloud

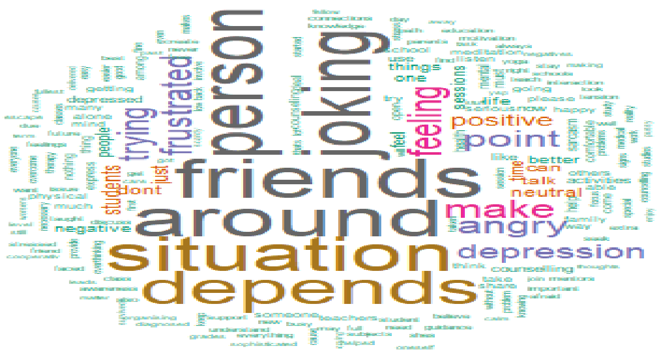


Figure 2 Word-Cloud implementation

Here is representation of frequency of bar plot of reviews terms reflecting sentiments in NLP model, using “syuzhet” library.

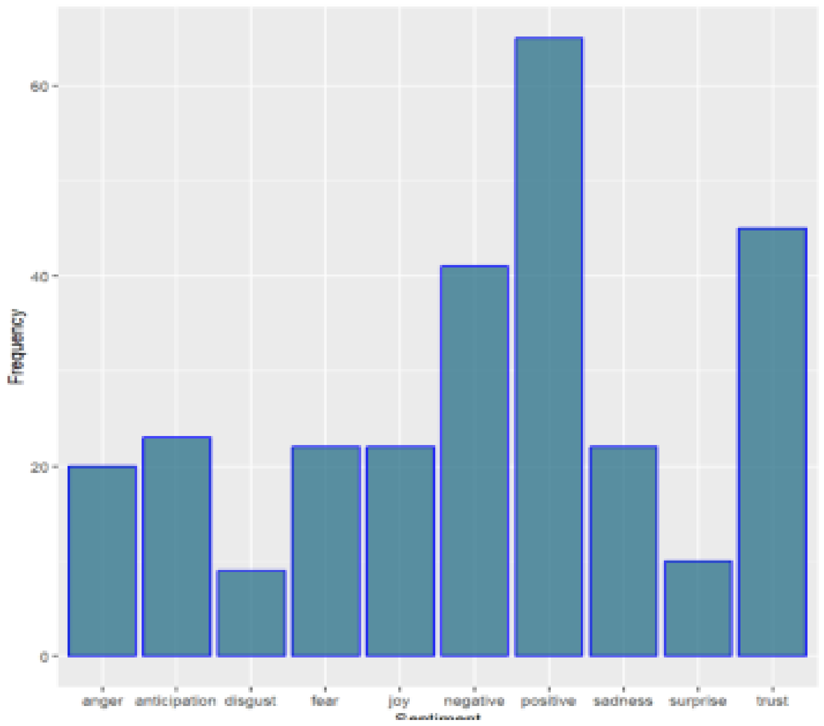
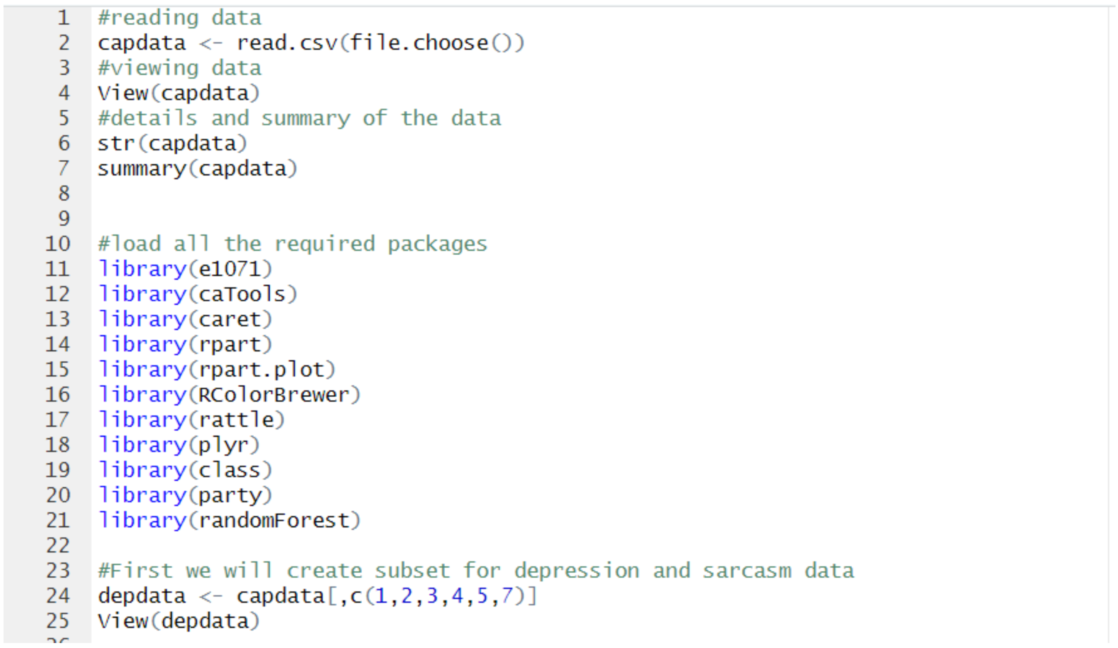
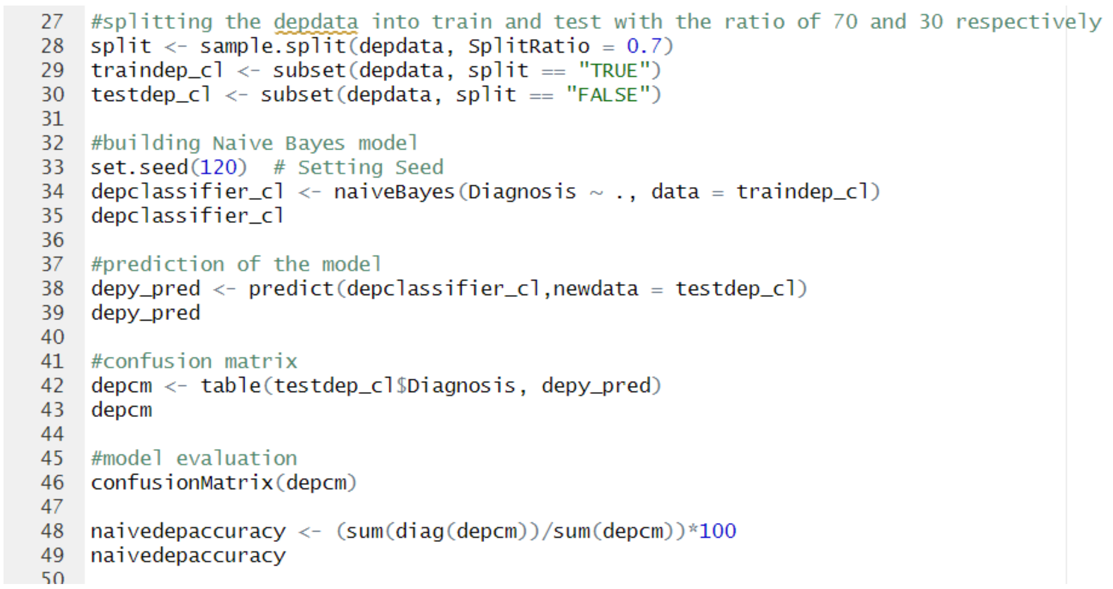


Figure 3 Frequency bar plot of sentiments

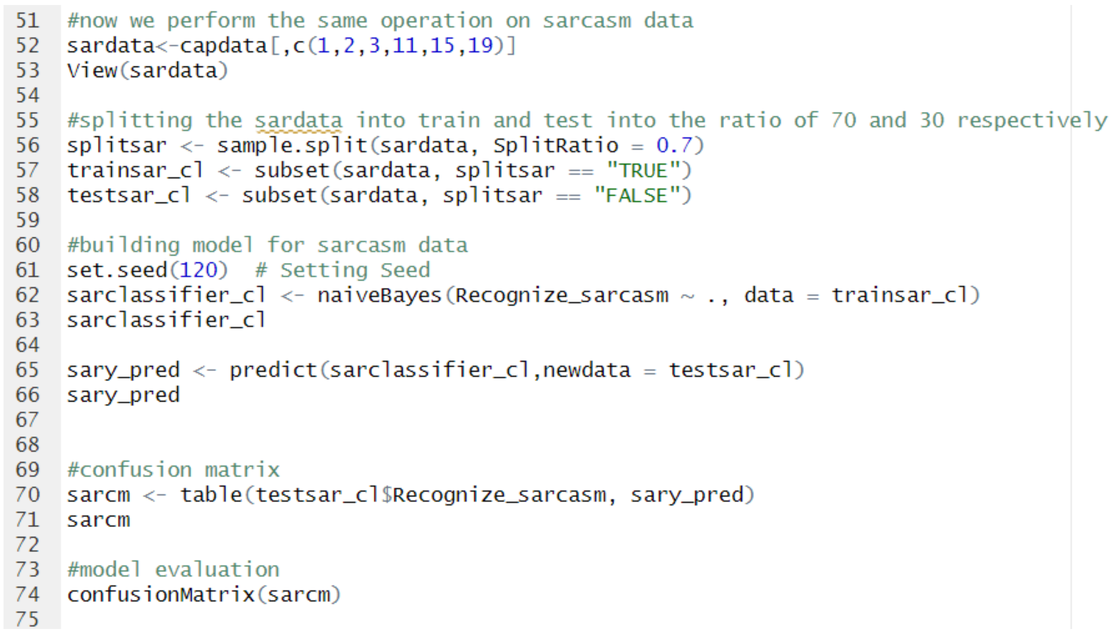
10.2. Implementation of Model



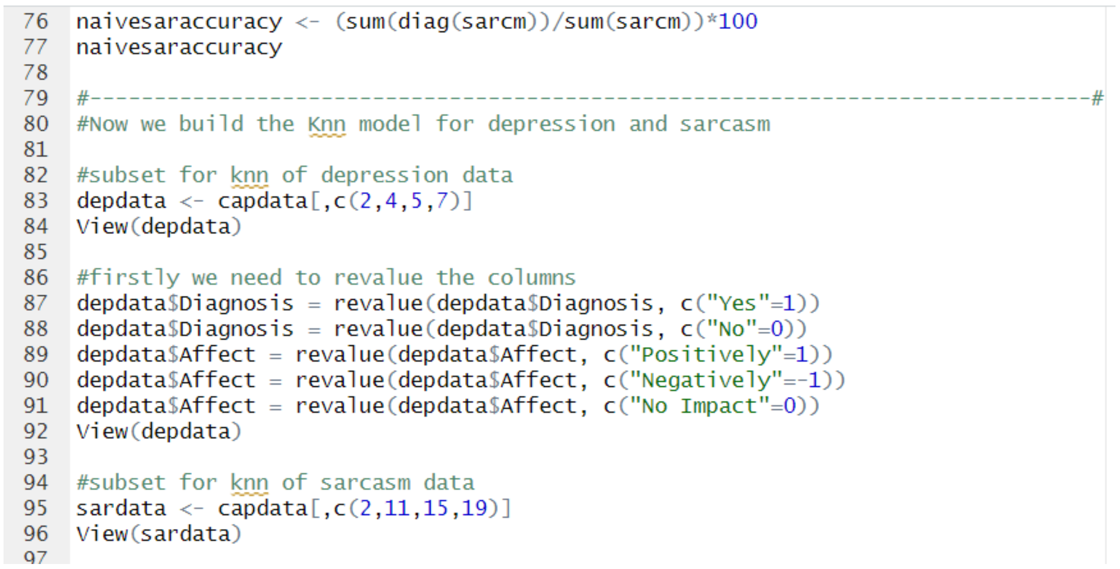
```r
#reading data
capdata <- read.csv(file.choose())
#viewing data
View(capdata)
#details and summary of the data
str(capdata)
summary(capdata)


#load all the required packages
library(e1071)
library(caTools)
library(caret)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
library(plyr)
library(class)
library(party)
library(randomForest)

#First we will create subset for depression and sarcasm data
depdata <- capdata[,c(1,2,3,4,5,7)]
View(depdata)
```



```r
#splitting the depdata into train and test with the ratio of 70 and 30 respectively
split <- sample.split(depdata, SplitRatio = 0.7)
traindep_cl <- subset(depdata, split == "TRUE")
testdep_cl <- subset(depdata, split == "FALSE")

#building Naive Bayes model
set.seed(120)  # Setting Seed
depclassifier_cl <- naiveBayes(Diagnosis ~ ., data = traindep_cl)
depclassifier_cl

#prediction of the model
depy_pred <- predict(depclassifier_cl,newdata = testdep_cl)
depy_pred

#confusion matrix
depcm <- table(testdep_cl$Diagnosis, depy_pred)
depcm

#model evaluation
confusionMatrix(depcm)

naivedepaccuracy <- (sum(diag(depcm))/sum(depcm))*100
naivedepaccuracy

```



```r
#now we perform the same operation on sarcasm data
sardata<-capdata[,c(1,2,3,11,15,19)]
View(sardata)

#splitting the sardata into train and test into the ratio of 70 and 30 respectively
splitsar <- sample.split(sardata, SplitRatio = 0.7)
trainsar_cl <- subset(sardata, splitsar == "TRUE")
testsar_cl <- subset(sardata, splitsar == "FALSE")

#building model for sarcasm data
set.seed(120)  # Setting Seed
sarclassifier_cl <- naiveBayes(Recognize_sarcasm ~ ., data = trainsar_cl)
sarclassifier_cl

sary_pred <- predict(sarclassifier_cl,newdata = testsar_cl)
sary_pred


#confusion matrix
sarcm <- table(testsar_cl$Recognize_sarcasm, sary_pred)
sarcm

#model evaluation
confusionMatrix(sarcm)

```

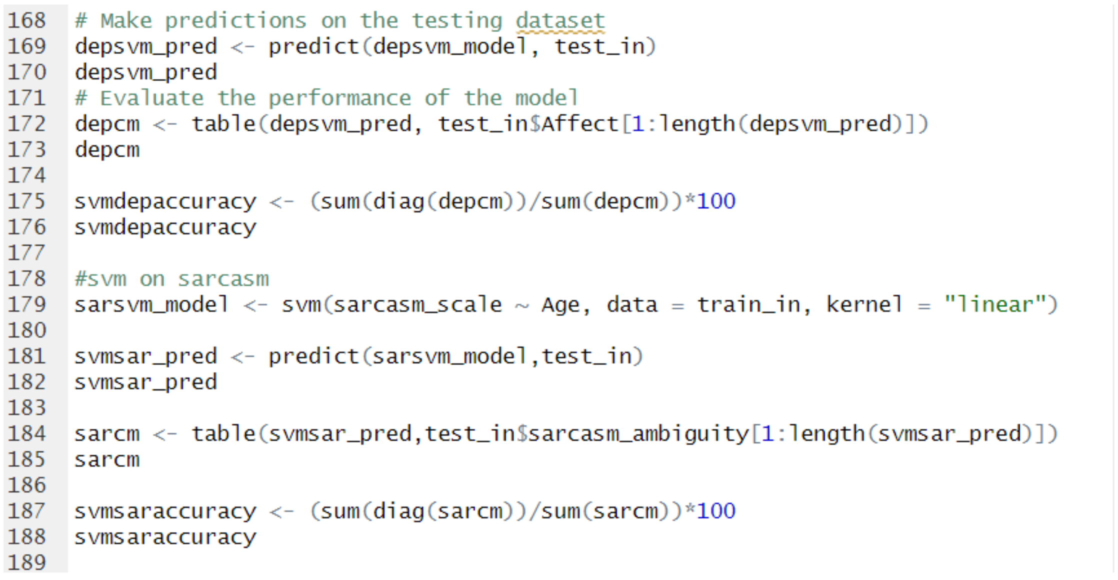


```r
naivesaraccuracy <- (sum(diag(sarcm))/sum(sarcm))*100
naivesaraccuracy

#-----------------------------------------------------------------------------#
#Now we build the Knn model for depression and sarcasm

#subset for knn of depression data
depdata <- capdata[,c(2,4,5,7)]
View(depdata)

#firstly we need to revalue the columns
depdata$Diagnosis = revalue(depdata$Diagnosis, c("Yes"=1))
depdata$Diagnosis = revalue(depdata$Diagnosis, c("No"=0))
depdata$Affect = revalue(depdata$Affect, c("Positively"=1))
depdata$Affect = revalue(depdata$Affect, c("Negatively"=-1))
depdata$Affect = revalue(depdata$Affect, c("No Impact"=0))
View(depdata)

#subset for knn of sarcasm data
sardata <- capdata[,c(2,11,15,19)]
View(sardata)

```

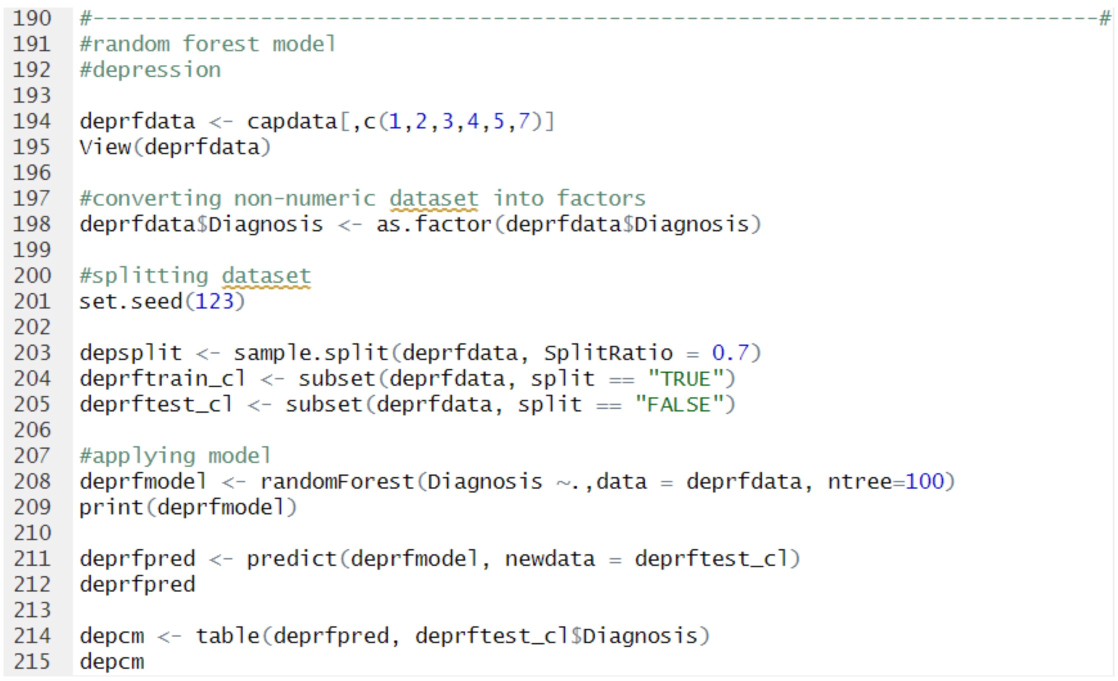
```r
sardata$Recognize_sarcasm = revalue(sardata$Recognize_sarcasm, c("Yes"=1))
sardata$Recognize_sarcasm = revalue(sardata$Recognize_sarcasm, c("No"=0))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("Yes"=1))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("Can't say"=0))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("No"=-1))
View(sardata)

# Convert the column to a factor
depdata$Affect <- as.factor(depdata$Affect)
sardata$sarcasm_ambiguity <- as.factor(sardata$sarcasm_ambiguity)


#splitting again as we have updated the subset data
#splitting the depdata into train and test with the ratio of 70 and 30 respectively
split <- sample.split(depdata, SplitRatio = 0.7)
traindep_cl <- subset(depdata, split == "TRUE")
testdep_cl <- subset(depdata, split == "FALSE")

xdeptrain <- traindep_cl[,-4]
ydeptrain <- traindep_cl[,4]
xdeptest <- testdep_cl[,-4]
ydeptest <- testdep_cl[,4]

```

```r
# Train the KNN model with k=5
depknn_model <- knn(xdeptrain,xdeptest,ydeptrain,k=5)

depcm <- table(testdep_cl$Affect,depknn_model)
depcm

# Check the accuracy of the model
knndepaccuracy <- (sum(diag(depcm))/sum(depcm))*100
knndepaccuracy

#now we apply it on sarcasm data
#splitting the sardata into train and test into the ratio of 70 and 30 respectively
splitsar <- sample.split(sardata, SplitRatio = 0.7)
trainsar_cl <- subset(sardata, splitsar == "TRUE")
testsar_cl <- subset(sardata, splitsar == "FALSE")


xsartrain <- trainsar_cl[,-4]
ysartrain <- trainsar_cl[,4]
xsartest <- testsar_cl[,-4]
ysartest <- testsar_cl[,4]

# Train the KNN model with k=5
sarknn_model <- knn(xsartrain,xsartest,ysartrain,k=5)
sarknn_model
```

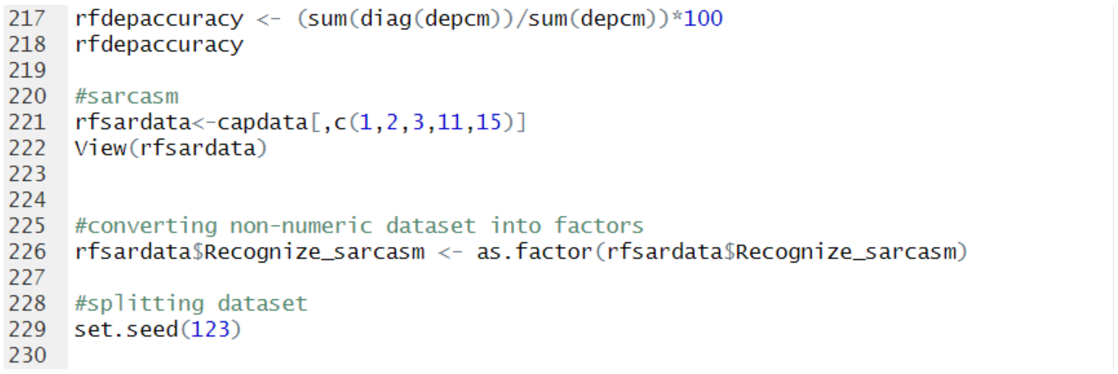
```r
sarcm <- table(testsar_cl$sarcasm_ambiguity,sarknn_model)
sarcm


# Check the accuracy of the model
knnsaraccuracy <- (sum(diag(sarcm))/sum(sarcm))*100
knnsaraccuracy

#-----------------------------------------------------------------------------#

#SVM model
#depression
# Split the data into training and testing datasets
set.seed(123)
trainIndex <- sample(1:nrow(capdata), 0.7*nrow(capdata))
train_in <- capdata[trainIndex,]
test_in <- capdata[-trainIndex,]

# Fit the SVM model
depsvm_model <- svm(Depression_scale ~ Age, data = train_in, kernel = "linear")

```



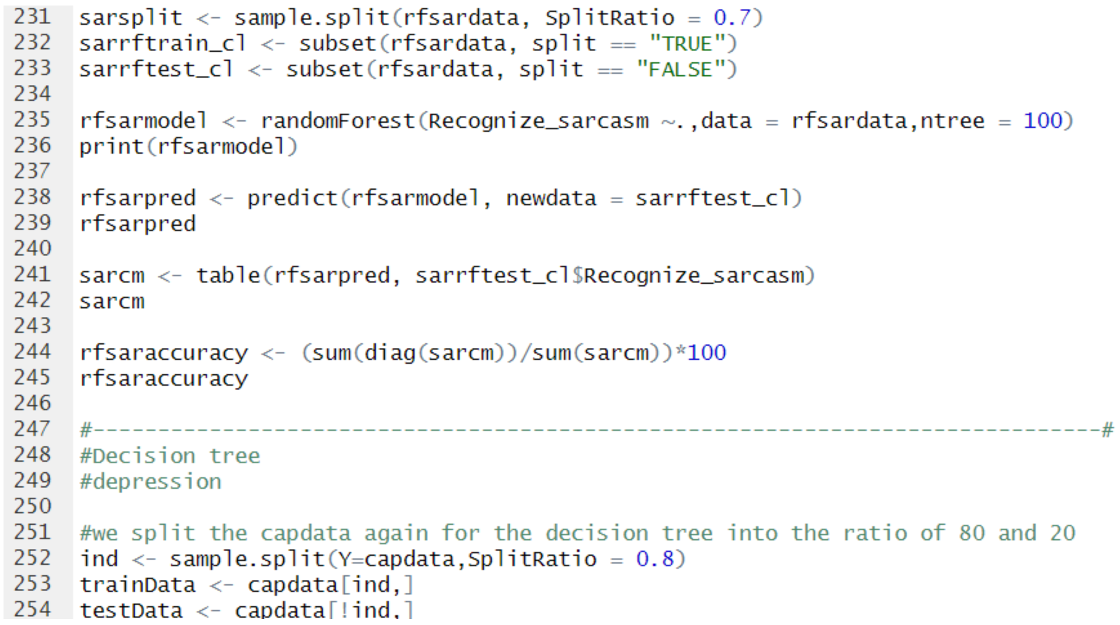
```r
# Make predictions on the testing dataset
depsvm_pred <- predict(depsvm_model, test_in)
depsvm_pred
# Evaluate the performance of the model
depcm <- table(depsvm_pred, test_in$Affect[1:length(depsvm_pred)])
depcm

svmdepaccuracy <- (sum(diag(depcm))/sum(depcm))*100
svmdepaccuracy

#svm on sarcasm
sarsvm_model <- svm(sarcasm_scale ~ Age, data = train_in, kernel = "linear")

svmsar_pred <- predict(sarsvm_model,test_in)
svmsar_pred

sarcm <- table(svmsar_pred,test_in$sarcasm_ambiguity[1:length(svmsar_pred)])
sarcm

svmsaraccuracy <- (sum(diag(sarcm))/sum(sarcm))*100
svmsaraccuracy

```



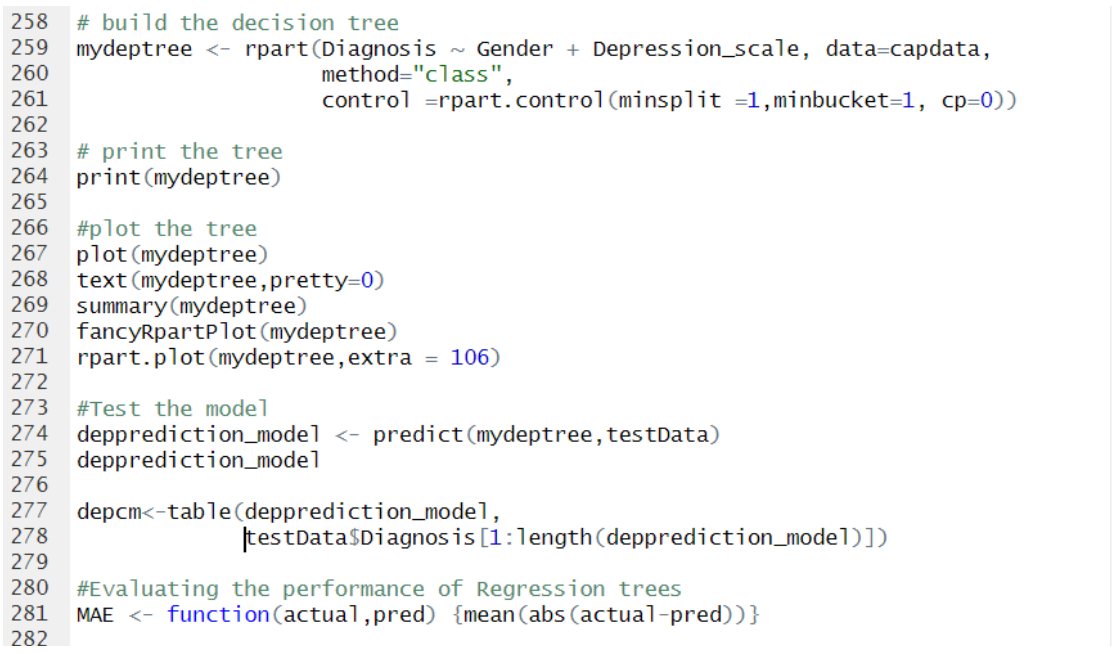
```r
#---------------------------------------------------------------------------#
#random forest model
#depression

deprfdata <- capdata[,c(1,2,3,4,5,7)]
View(deprfdata)

#converting non-numeric dataset into factors
deprfdata$Diagnosis <- as.factor(deprfdata$Diagnosis)

#splitting dataset
set.seed(123)

depsplit <- sample.split(deprfdata, SplitRatio = 0.7)
deprftrain_cl <- subset(deprfdata, split == "TRUE")
deprftest_cl <- subset(deprfdata, split == "FALSE")

#applying model
deprfmodel <- randomForest(Diagnosis ~.,data = deprfdata, ntree=100)
print(deprfmodel)

deprfpred <- predict(deprfmodel, newdata = deprftest_cl)
deprfpred

depcm <- table(deprfpred, deprftest_cl$Diagnosis)
depcm
```



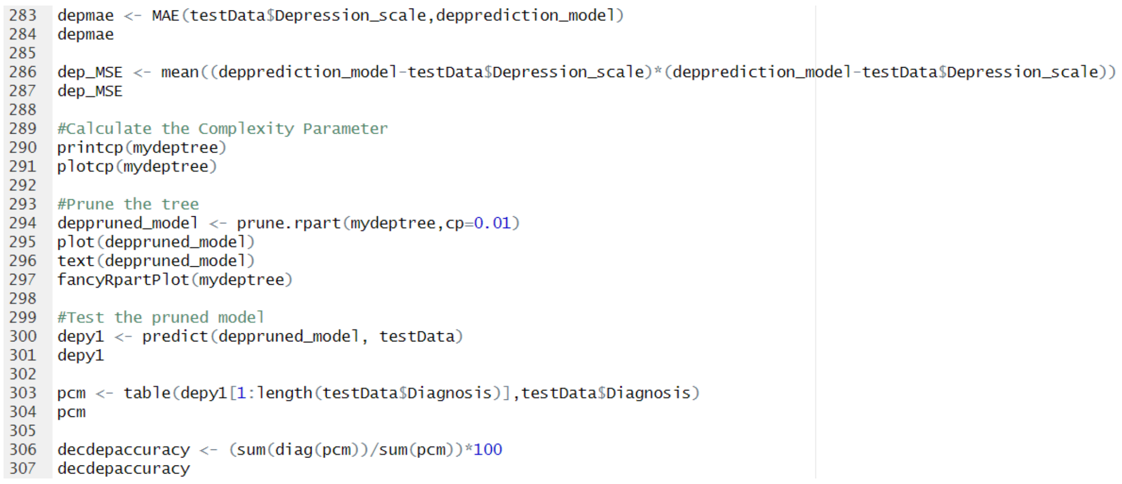
```r
rfdepaccuracy <- (sum(diag(depcm))/sum(depcm))*100
rfdepaccuracy

#sarcasm
rfsardata<-capdata[,c(1,2,3,11,15)]
View(rfsardata)


#converting non-numeric dataset into factors
rfsardata$Recognize_sarcasm <- as.factor(rfsardata$Recognize_sarcasm)

#splitting dataset
set.seed(123)

```



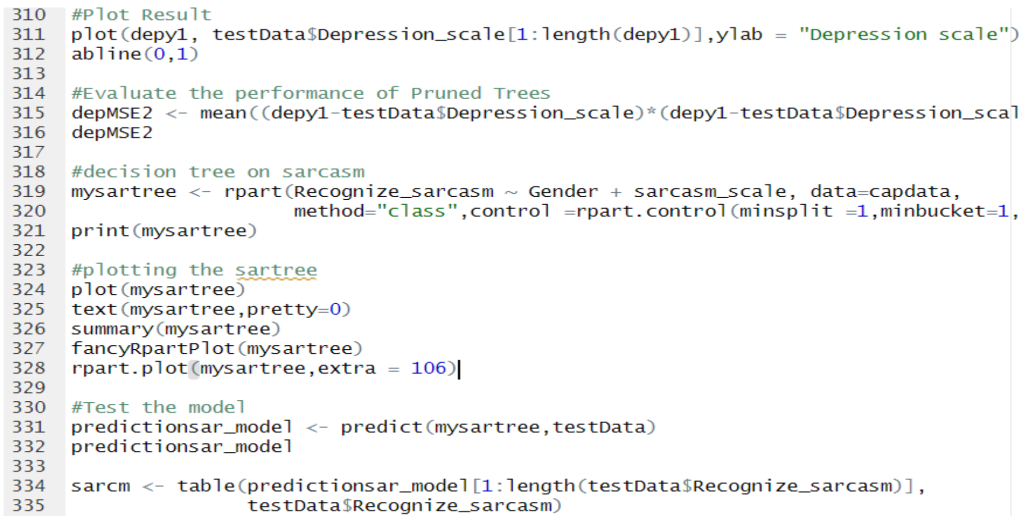
```r
sarsplit <- sample.split(rfsardata, SplitRatio = 0.7)
sarrftrain_cl <- subset(rfsardata, split == "TRUE")
sarrftest_cl <- subset(rfsardata, split == "FALSE")

rfsarmodel <- randomForest(Recognize_sarcasm ~.,data = rfsardata,ntree = 100)
print(rfsarmodel)

rfsarpred <- predict(rfsarmodel, newdata = sarrftest_cl)
rfsarpred

sarcm <- table(rfsarpred, sarrftest_cl$Recognize_sarcasm)
sarcm

rfsaraccuracy <- (sum(diag(sarcm))/sum(sarcm))*100
rfsaraccuracy

#--------------------------------------------------------------------------------#
#Decision tree
#depression

#we split the capdata again for the decision tree into the ratio of 80 and 20
ind <- sample.split(Y=capdata,SplitRatio = 0.8)
trainData <- capdata[ind,]
testData <- capdata[!ind,]
```



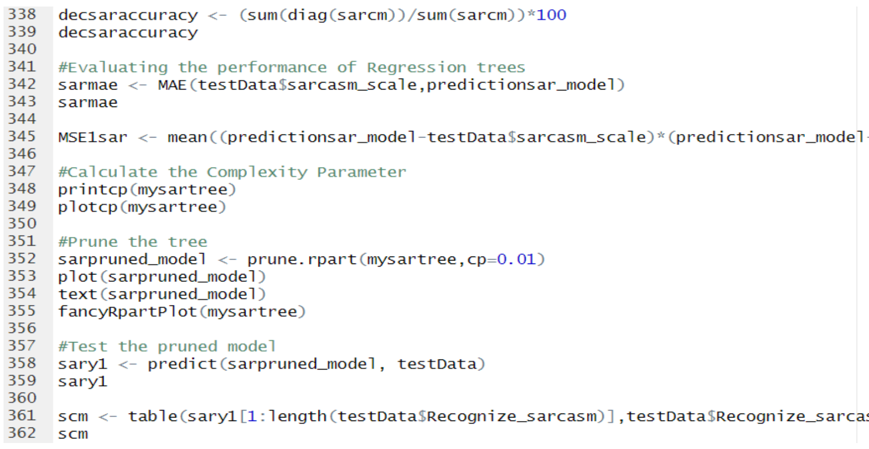
```r
# build the decision tree
mydeptree <- rpart(Diagnosis ~ Gender + Depression_scale, data=capdata,
                   method="class",
                   control =rpart.control(minsplit =1,minbucket=1, cp=0))

# print the tree
print(mydeptree)

#plot the tree
plot(mydeptree)
text(mydeptree,pretty=0)
summary(mydeptree)
fancyRpartPlot(mydeptree)
rpart.plot(mydeptree,extra = 106)

#Test the model
depprediction_model <- predict(mydeptree,testData)
depprediction_model

depcm<-table(depprediction_model,
             testData$Diagnosis[1:length(depprediction_model)])

#Evaluating the performance of Regression trees
MAE <- function(actual,pred) {mean(abs(actual-pred))}
```



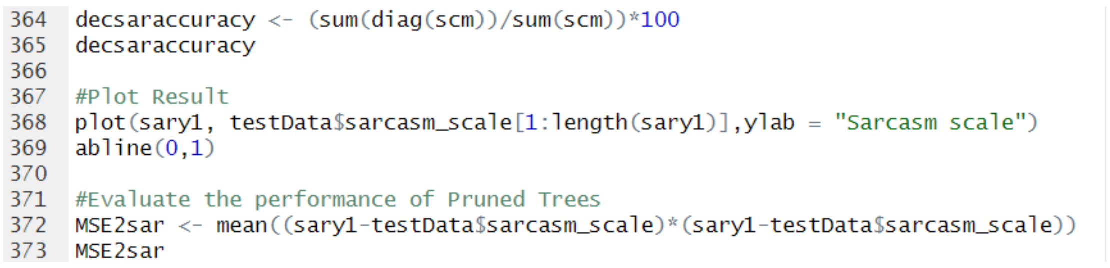
```r
depmae <- MAE(testData$Depression_scale,depprediction_model)
depmae

dep_MSE <- mean((depprediction_model-testData$Depression_scale)*(depprediction_model-testData$Depression_scale))
dep_MSE

#Calculate the Complexity Parameter
printcp(mydeptree)
plotcp(mydeptree)

#Prune the tree
deppruned_model <- prune.rpart(mydeptree,cp=0.01)
plot(deppruned_model)
text(deppruned_model)
fancyRpartPlot(mydeptree)

#Test the pruned model
depy1 <- predict(deppruned_model, testData)
depy1

pcm <- table(depy1[1:length(testData$Diagnosis)],testData$Diagnosis)
pcm

decdepaccuracy <- (sum(diag(pcm))/sum(pcm))*100
decdepaccuracy
```



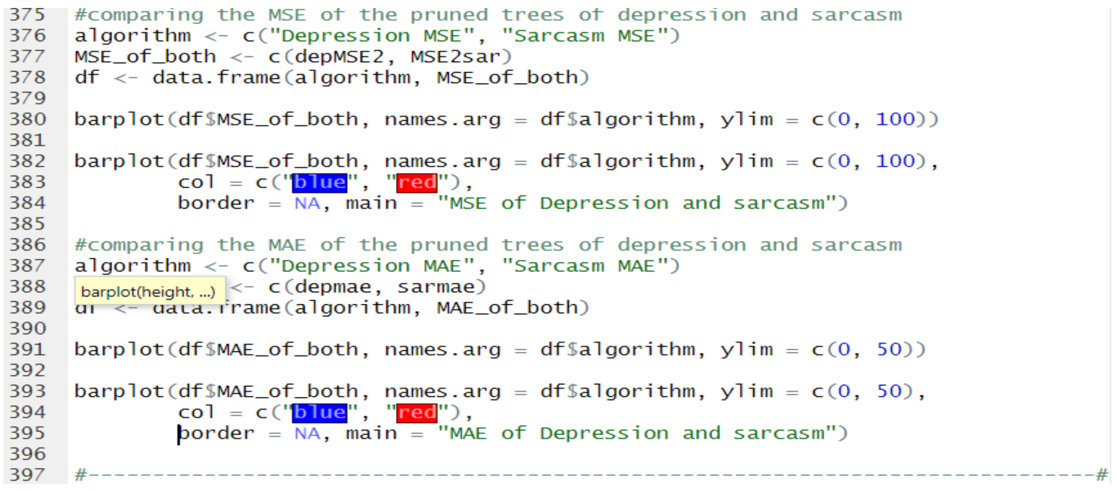
```r
#Plot Result
plot(depy1, testData$Depression_scale[1:length(depy1)],ylab = "Depression scale")
abline(0,1)

#Evaluate the performance of Pruned Trees
depMSE2 <- mean((depy1-testData$Depression_scale)*(depy1-testData$Depression_scale))
depMSE2

#decision tree on sarcasm
mysartree <- rpart(Recognize_sarcasm ~ Gender + sarcasm_scale, data=capdata,
                   method="class",control =rpart.control(minsplit =1,minbucket=1, cp=0))
print(mysartree)

#plotting the sartree
plot(mysartree)
text(mysartree,pretty=0)
summary(mysartree)
fancyRpartPlot(mysartree)
rpart.plot(mysartree,extra = 106)

#Test the model
predictionsar_model <- predict(mysartree,testData)
predictionsar_model

sarcm <- table(predictionsar_model[1:length(testData$Recognize_sarcasm)],
               testData$Recognize_sarcasm)
```



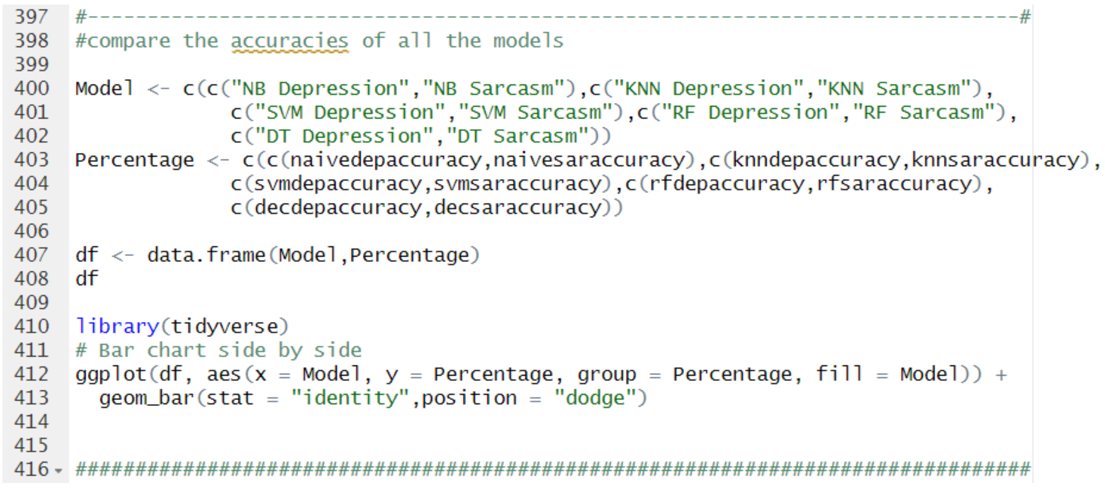
```r
decsaraccuracy <- (sum(diag(sarcm))/sum(sarcm))*100
decsaraccuracy

#Evaluating the performance of Regression trees
sarmae <- MAE(testData$sarcasm_scale,predictionsar_model)
sarmae

MSE1sar <- mean((predictionsar_model-testData$sarcasm_scale)*(predictionsar_model-testData$sarcasm_scale))

#Calculate the Complexity Parameter
printcp(mysartree)
plotcp(mysartree)

#Prune the tree
sarpruned_model <- prune.rpart(mysartree,cp=0.01)
plot(sarpruned_model)
text(sarpruned_model)
fancyRpartPlot(mysartree)

#Test the pruned model
sary1 <- predict(sarpruned_model, testData)
sary1

scm <- table(sary1[1:length(testData$Recognize_sarcasm)],testData$Recognize_sarcasm)
scm
```



```r
decsaraccuracy <- (sum(diag(scm))/sum(scm))*100
decsaraccuracy

#Plot Result
plot(sary1, testData$sarcasm_scale[1:length(sary1)],ylab = "Sarcasm scale")
abline(0,1)

#Evaluate the performance of Pruned Trees
MSE2sar <- mean((sary1-testData$sarcasm_scale)*(sary1-testData$sarcasm_scale))
MSE2sar
```



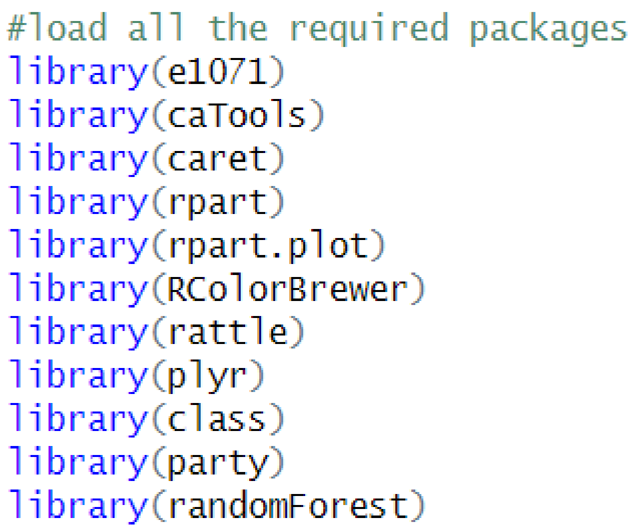
```r
#comparing the MSE of the pruned trees of depression and sarcasm
algorithm <- c("Depression MSE", "Sarcasm MSE")
MSE_of_both <- c(depMSE2, MSE2sar)
df <- data.frame(algorithm, MSE_of_both)

barplot(df$MSE_of_both, names.arg = df$algorithm, ylim = c(0, 100))

barplot(df$MSE_of_both, names.arg = df$algorithm, ylim = c(0, 100),
        col = c("blue", "red"),
        border = NA, main = "MSE of Depression and sarcasm")

#comparing the MAE of the pruned trees of depression and sarcasm
algorithm <- c("Depression MAE", "Sarcasm MAE")
             <- c(depmae, sarmae)
df <- data.frame(algorithm, MAE_of_both)

barplot(df$MAE_of_both, names.arg = df$algorithm, ylim = c(0, 50))

barplot(df$MAE_of_both, names.arg = df$algorithm, ylim = c(0, 50),
        col = c("blue", "red"),
        border = NA, main = "MAE of Depression and sarcasm")

#-----------------------------------------------------------------------------#
```



```
#---------------------------------------------------------------------------------#
#compare the accuracies of all the models

Model <- c(c("NB Depression","NB Sarcasm"),c("KNN Depression","KNN Sarcasm"),
             c("SVM Depression","SVM Sarcasm"),c("RF Depression","RF Sarcasm"),
             c("DT Depression","DT Sarcasm"))
Percentage <- c(c(naivedepaccuracy,naivesaraccuracy),c(knndepaccuracy,knnsaraccuracy),
             c(svmdepaccuracy,svmsaraccuracy),c(rfdepaccuracy,rfsaraccuracy),
             c(decdepaccuracy,decsaraccuracy))

df <- data.frame(Model,Percentage)
df

library(tidyverse)
# Bar chart side by side
ggplot(df, aes(x = Model, y = Percentage, group = Percentage, fill = Model)) +
  geom_bar(stat = "identity",position = "dodge")


#################################################################################
```

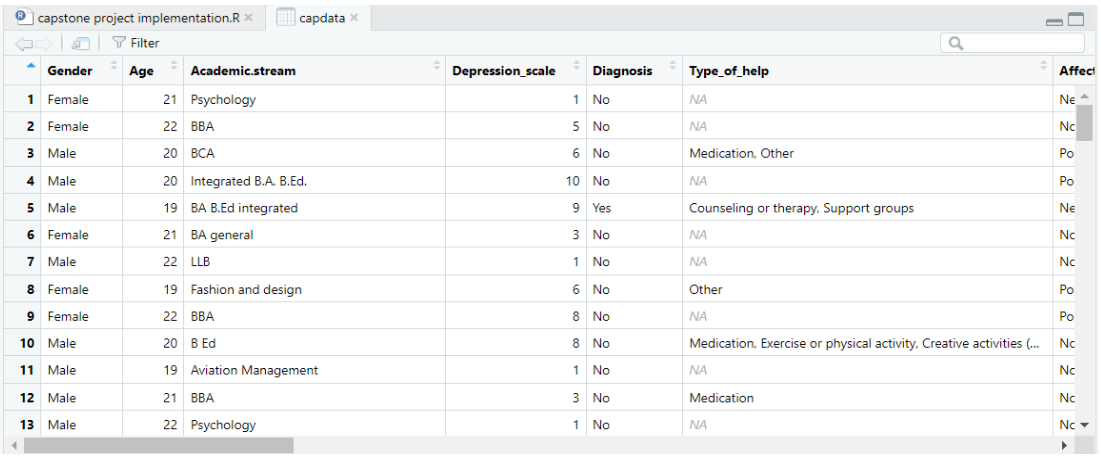
END OF CODE

Loading all the required packages that are used for model building and various functions.



```
#load all the required packages
library(e1071)
library(caTools)
library(caret)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
library(plyr)
library(class)
library(party)
library(randomForest)
```

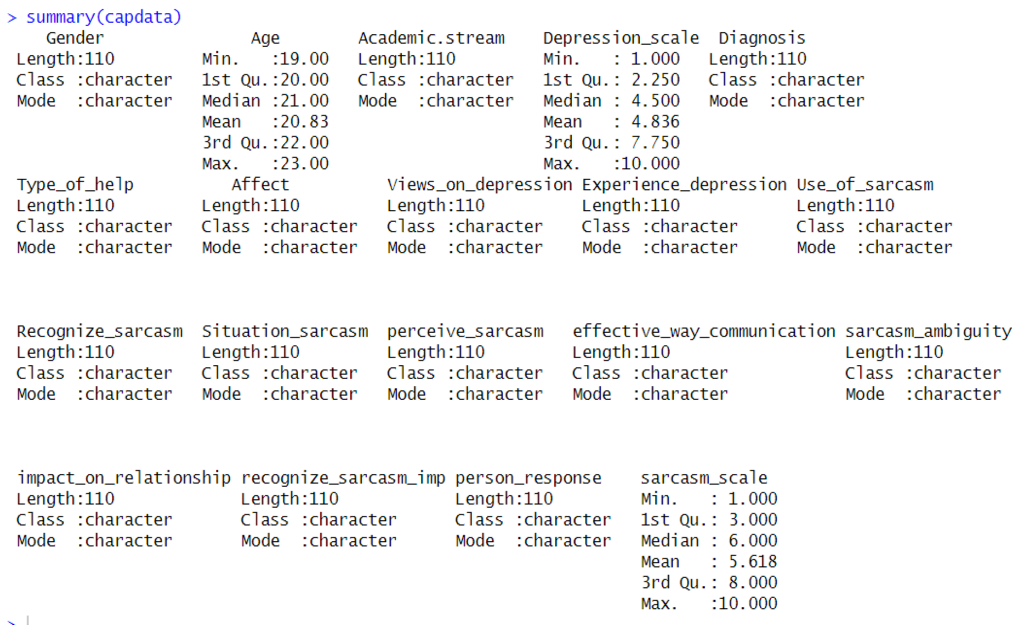
The representation of main dataset that includes all the 19 attributes and 110 records.



	Gender	Age	Academic.stream	Depression_scale	Diagnosis	Type_of_help	Affect
1	Female	21	Psychology	1	No	NA	Ne
2	Female	22	BBA	5	No	NA	Nc
3	Male	20	BCA	6	No	Medication, Other	Po
4	Male	20	Integrated B.A. B.Ed.	10	No	NA	Po
5	Male	19	BA B.Ed integrated	9	Yes	Counseling or therapy, Support groups	Ne
6	Female	21	BA general	3	No	NA	Nc
7	Male	22	LLB	1	No	NA	Nc
8	Female	19	Fashion and design	6	No	Other	Po
9	Female	22	BBA	8	No	NA	Po
10	Male	20	B Ed	8	No	Medication, Exercise or physical activity, Creative activities (...	Nc
11	Male	19	Aviation Management	1	No	NA	Nc
12	Male	21	BBA	3	No	Medication	Nc
13	Male	22	Psychology	1	No	NA	Nc

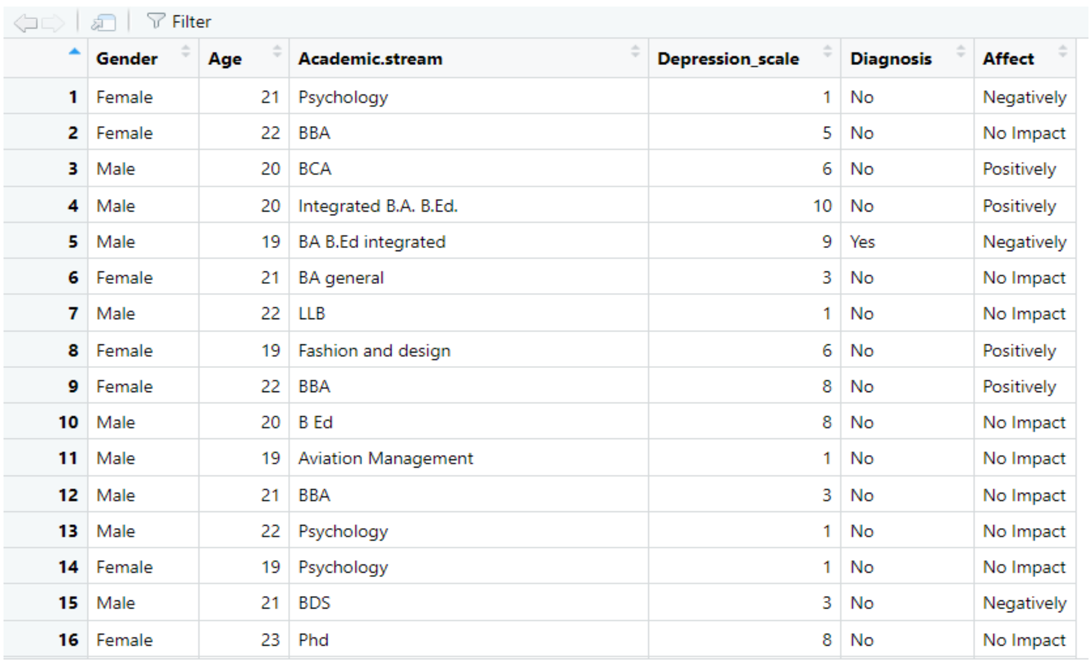
To create a summary of a dataset or statistical object like a linear model, generalized linear model, or mixed-effects model in the R programming language, use the summary() function. The data, including measures of central tendency, dispersion, and other descriptive statistics, can be quickly viewed using the summary() function.

The summary of capstone data is represented below with all the detailed attributes with maximum and minimum value, mode, class, median, mean etc.



```
> summary(capdata)
    Gender               Age          Academic.stream     Depression_scale  Diagnosis
 Length:110         Min.   :19.00   Length:110         Min.   : 1.000   Length:110
 Class :character   1st Qu.:20.00   Class :character   1st Qu.: 2.250   Class :character
 Mode  :character   Median :21.00   Mode  :character   Median : 4.500   Mode  :character
                    Mean   :20.83                      Mean   : 4.836
                    3rd Qu.:22.00                      3rd Qu.: 7.750
                    Max.   :23.00                      Max.   :10.000
 Type_of_help          Affect          Views_on_depression Experience_depression Use_of_sarcasm
 Length:110         Length:110         Length:110          Length:110            Length:110
 Class :character   Class :character   Class :character    Class :character      Class :character
 Mode  :character   Mode  :character   Mode  :character    Mode  :character      Mode  :character



 Recognize_sarcasm  Situation_sarcasm  perceive_sarcasm   effective_way_communication sarcasm_ambiguity
 Length:110         Length:110         Length:110         Length:110                  Length:110
 Class :character   Class :character   Class :character   Class :character            Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character            Mode  :character



 impact_on_relationship recognize_sarcasm_imp person_response    sarcasm_scale
 Length:110             Length:110            Length:110         Min.   : 1.000
 Class :character       Class :character      Class :character   1st Qu.: 3.000
 Mode  :character       Mode  :character      Mode  :character   Median : 6.000
                                                                 Mean   : 5.618
                                                                 3rd Qu.: 8.000
                                                                 Max.   :10.000
```

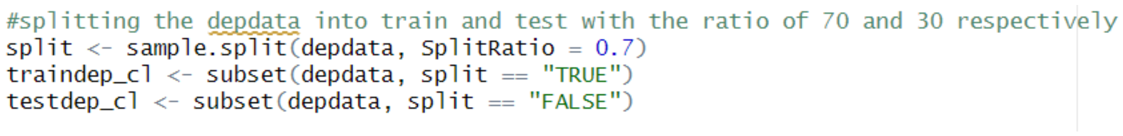
Below is the representation of Depression dataset which is the subset of the main dataset. The subset is created to analyze the depression dataset parallel with sarcasm dataset using the same algorithms.



Filter

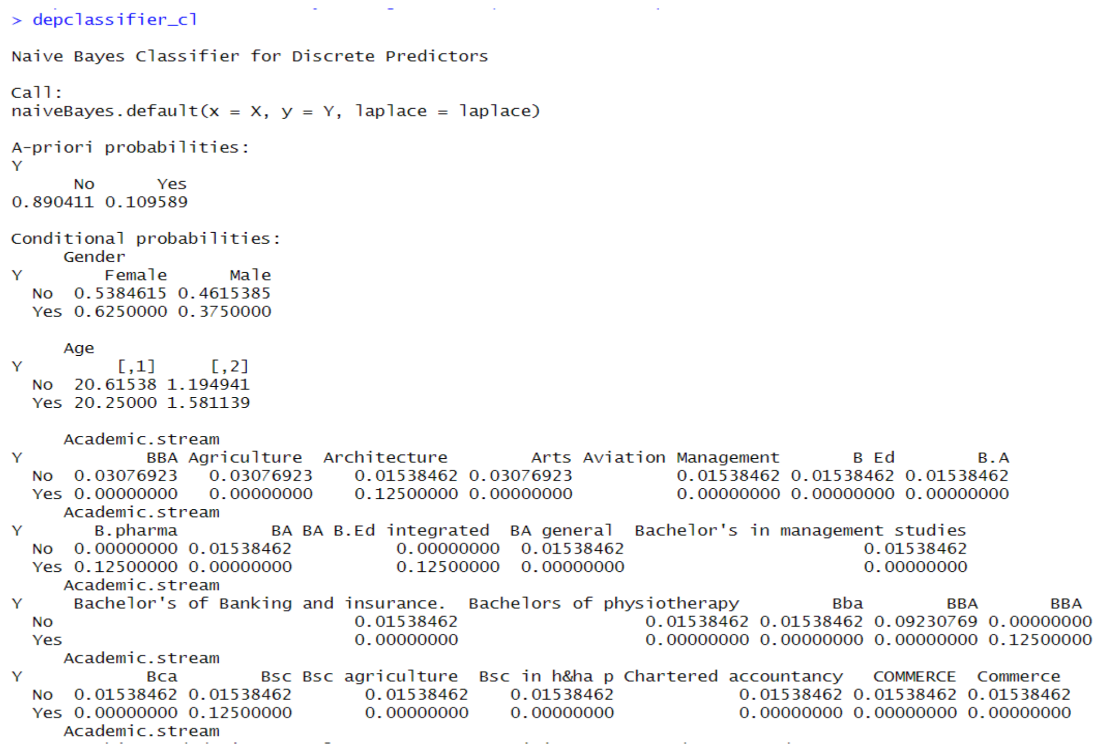
	Gender	Age	Academic.stream	Depression_scale	Diagnosis	Affect
1	Female	21	Psychology	1	No	Negatively
2	Female	22	BBA	5	No	No Impact
3	Male	20	BCA	6	No	Positively
4	Male	20	Integrated B.A. B.Ed.	10	No	Positively
5	Male	19	BA B.Ed integrated	9	Yes	Negatively
6	Female	21	BA general	3	No	No Impact
7	Male	22	LLB	1	No	No Impact
8	Female	19	Fashion and design	6	No	Positively
9	Female	22	BBA	8	No	Positively
10	Male	20	B Ed	8	No	No Impact
11	Male	19	Aviation Management	1	No	No Impact
12	Male	21	BBA	3	No	No Impact
13	Male	22	Psychology	1	No	No Impact
14	Female	19	Psychology	1	No	No Impact
15	Male	21	BDS	3	No	Negatively
16	Female	23	Phd	8	No	No Impact

Since we are using Supervised learning algorithms, we must create a train and test of the data as it works on previously generated data and gives the output accordingly.



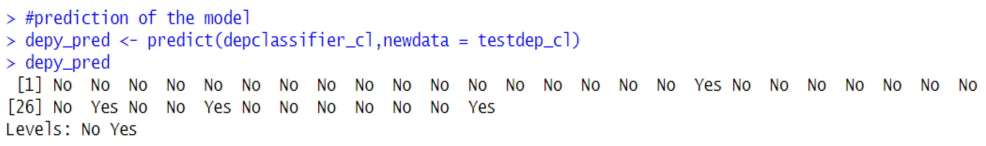
```
#splitting the depdata into train and test with the ratio of 70 and 30 respectively
split <- sample.split(depdata, SplitRatio = 0.7)
traindep_cl <- subset(depdata, split == "TRUE")
testdep_cl <- subset(depdata, split == "FALSE")
```

After applying naïve bayes model on depression dataset, here is a small snapshot of the output.



```
> depclassifier_cl

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
      No      Yes 
0.890411 0.109589 

Conditional probabilities:
     Gender
Y        Female      Male
  No  0.5384615 0.4615385
  Yes 0.6250000 0.3750000

     Age
Y         [,1]     [,2]
  No  20.61538 1.194941
  Yes 20.25000 1.581139

     Academic.stream
Y            BBA Agriculture  Architecture        Arts Aviation Management       B Ed         B.A
  No  0.03076923    0.03076923    0.01538462 0.03076923          0.01538462 0.01538462 0.01538462
  Yes 0.00000000    0.00000000    0.12500000 0.00000000          0.00000000 0.00000000 0.00000000
     Academic.stream
Y        B.pharma         BA BA B.Ed integrated  BA general  Bachelor's in management studies
  No  0.00000000 0.01538462          0.00000000  0.01538462                        0.01538462
  Yes 0.12500000 0.00000000          0.12500000  0.00000000                        0.00000000
     Academic.stream
Y     Bachelor's of Banking and insurance.  Bachelors of physiotherapy        Bba        BBA        BBA
  No                            0.01538462                          0.01538462 0.01538462 0.09230769 0.00000000
  Yes                           0.00000000                          0.00000000 0.00000000 0.00000000 0.12500000
     Academic.stream
Y            Bca        Bsc Bsc agriculture  Bsc in h&ha p Chartered accountancy   COMMERCE   Commerce
  No  0.01538462 0.01538462      0.01538462     0.01538462            0.01538462 0.01538462 0.01538462
  Yes 0.00000000 0.12500000      0.00000000     0.00000000            0.00000000 0.00000000 0.00000000
     Academic.stream
```

Prediction of naïve bayes model on depression dataset



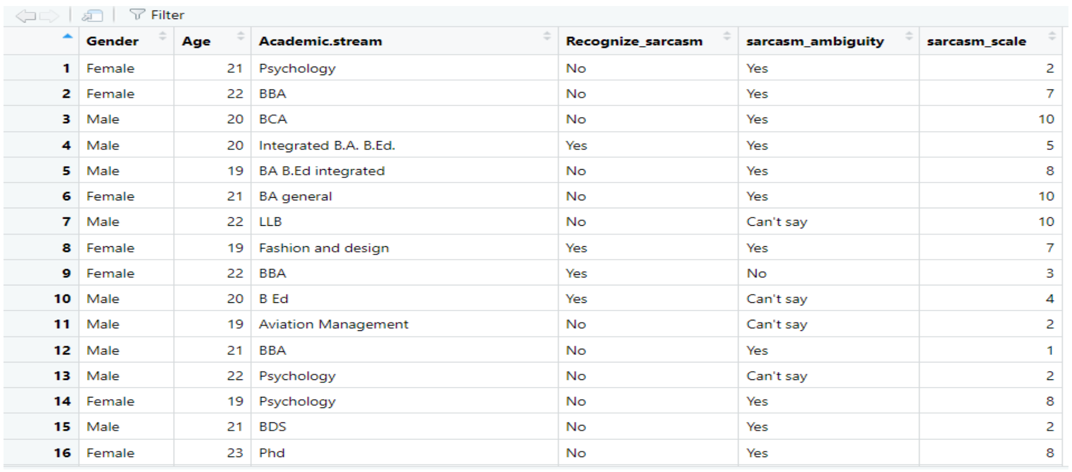
```
> #prediction of the model
> depy_pred <- predict(depclassifier_cl,newdata = testdep_cl)
> depy_pred
 [1] No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  Yes No  No  No  No  No  No  No 
[26] No  Yes No  No  Yes No  No  No  No  No  No  Yes
Levels: No Yes
```

Confusion matrix of depression data. It gives a matrix of TF(True False) data with accuracies with all the other required information.

```
> #model evaluation
> confusionMatrix(depcm)
Confusion Matrix and Statistics

     depy_pred
      No Yes
  No  31   4
  Yes  2   0

               Accuracy : 0.8378
                 95% CI : (0.6799, 0.9381)
    No Information Rate : 0.8919
    P-Value [Acc > NIR] : 0.9015

                  Kappa : -0.0777

 Mcnemar's Test P-Value : 0.6831

            Sensitivity : 0.9394
            Specificity : 0.0000
         Pos Pred Value : 0.8857
         Neg Pred Value : 0.0000
             Prevalence : 0.8919
         Detection Rate : 0.8378
   Detection Prevalence : 0.9459
      Balanced Accuracy : 0.4697

       'Positive' Class : No
```

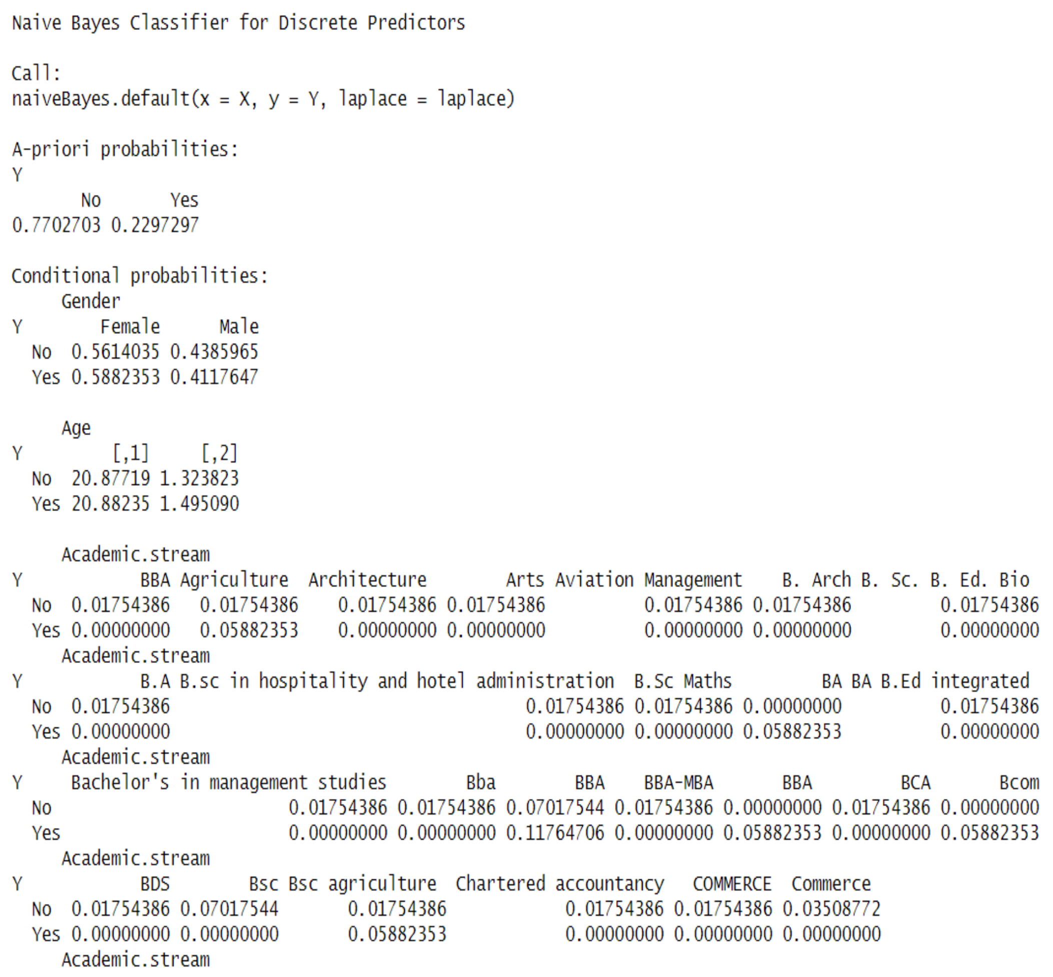
After the evaluation of matrix, the accuracy is found to be 88.88% and it changes every time we execute the machine learning program as the dataset is selected randomly every time it is executed.

The next subset of the data is of sarcasm whose sample is shown below.



	Gender	Age	Academic.stream	Recognize_sarcasm	sarcasm_ambiguity	sarcasm_scale
1	Female	21	Psychology	No	Yes	2
2	Female	22	BBA	No	Yes	7
3	Male	20	BCA	No	Yes	10
4	Male	20	Integrated B.A. B.Ed.	Yes	Yes	5
5	Male	19	BA B.Ed integrated	No	Yes	8
6	Female	21	BA general	No	Yes	10
7	Male	22	LLB	No	Can't say	10
8	Female	19	Fashion and design	Yes	Yes	7
9	Female	22	BBA	Yes	No	3
10	Male	20	B Ed	Yes	Can't say	4
11	Male	19	Aviation Management	No	Can't say	2
12	Male	21	BBA	No	Yes	1
13	Male	22	Psychology	No	Can't say	2
14	Female	19	Psychology	No	Yes	8
15	Male	21	BDS	No	Yes	2
16	Female	23	Phd	No	Yes	8

Implementation of naïve bayes model on sarcasm dataset. After creating the model and executing the same the output is as follows, where it depicts all the information of every attribute.



```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
       No       Yes 
0.7702703 0.2297297 

Conditional probabilities:
     Gender
Y        Female      Male
  No  0.5614035 0.4385965
  Yes 0.5882353 0.4117647

     Age
Y         [,1]     [,2]
  No  20.87719 1.323823
  Yes 20.88235 1.495090

     Academic.stream
Y            BBA Agriculture  Architecture        Arts Aviation Management    B. Arch B. Sc. B. Ed. Bio
  No  0.01754386   0.01754386    0.01754386 0.01754386          0.01754386 0.01754386         0.01754386
  Yes 0.00000000   0.05882353    0.00000000 0.00000000          0.00000000 0.00000000         0.00000000
     Academic.stream
Y            B.A B.sc in hospitality and hotel administration  B.Sc Maths          BA BA B.Ed integrated
  No  0.01754386                                    0.01754386 0.01754386 0.00000000          0.01754386
  Yes 0.00000000                                    0.00000000 0.00000000 0.05882353          0.00000000
     Academic.stream
Y     Bachelor's in management studies        Bba        BBA    BBA-MBA       BBA         BCA       Bcom
  No                        0.01754386 0.01754386 0.07017544 0.01754386 0.00000000 0.01754386 0.00000000
  Yes                       0.00000000 0.00000000 0.11764706 0.00000000 0.05882353 0.00000000 0.05882353
     Academic.stream
Y            BDS        Bsc Bsc agriculture  Chartered accountancy   COMMERCE  Commerce
  No  0.01754386 0.07017544       0.01754386            0.01754386 0.01754386 0.03508772
  Yes 0.00000000 0.00000000       0.05882353            0.00000000 0.00000000 0.00000000
     Academic.stream
```

Confusion matrix evaluation of the sarcasm data, which gives accuracies and all the related information regarding it.

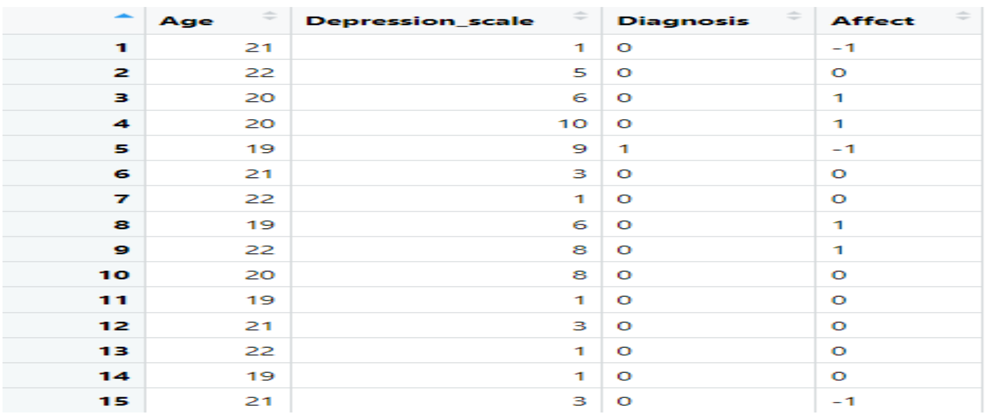
```
> #model evaluation
> confusionMatrix(sarcm)
Confusion Matrix and Statistics

        sary_pred
         No Yes
  No     20   6
  Yes    10   0

               Accuracy : 0.5556
                 95% CI : (0.381, 0.7206)
    No Information Rate : 0.8333
    P-Value [Acc > NIR] : 1.0000

                  Kappa : -0.2632

 Mcnemar's Test P-Value : 0.4533

            Sensitivity : 0.6667
            Specificity : 0.0000
         Pos Pred Value : 0.7692
         Neg Pred Value : 0.0000
             Prevalence : 0.8333
         Detection Rate : 0.5556
   Detection Prevalence : 0.7222
      Balanced Accuracy : 0.3333

       'Positive' Class : No
```

The accuracy of the sarcasm data is found to be 55.55% which also can be improved by adding extra attributes and extra resources.

After implementation of naïve bayes model we now build a KNN model on depression dataset and the data now is modified as the attributes are transformed numerical values.

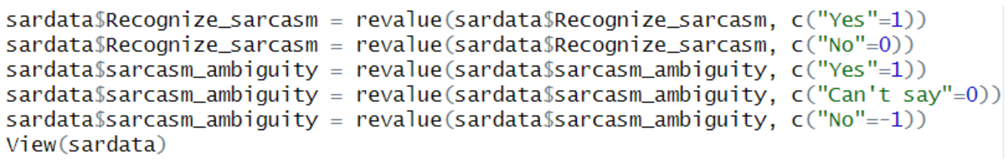
```
#firstly we need to revalue the columns
depdata$Diagnosis = revalue(depdata$Diagnosis, c("Yes"=1))
depdata$Diagnosis = revalue(depdata$Diagnosis, c("No"=0))
depdata$Affect = revalue(depdata$Affect, c("Positively"=1))
depdata$Affect = revalue(depdata$Affect, c("Negatively"=-1))
depdata$Affect = revalue(depdata$Affect, c("No Impact"=0))
View(depdata)
```

The view of the data after modification is shown below.

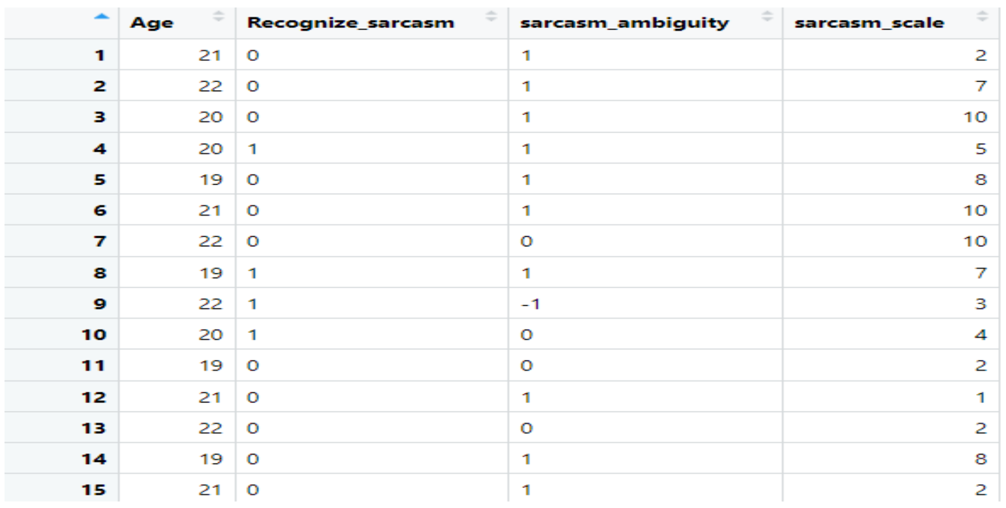


	Age	Depression_scale	Diagnosis	Affect
1	21	1	0	-1
2	22	5	0	0
3	20	6	0	1
4	20	10	0	1
5	19	9	1	-1
6	21	3	0	0
7	22	1	0	0
8	19	6	0	1
9	22	8	0	1
10	20	8	0	0
11	19	1	0	0
12	21	3	0	0
13	22	1	0	0
14	19	1	0	0
15	21	3	0	-1

The sarcasm data is also modified for further analysis.

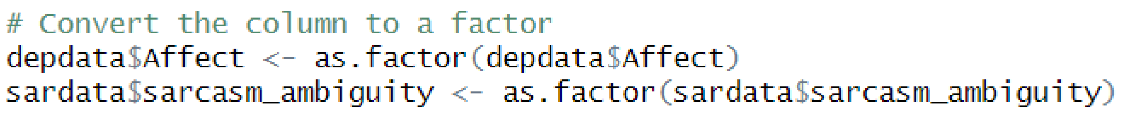


```
sardata$Recognize_sarcasm = revalue(sardata$Recognize_sarcasm, c("Yes"=1))
sardata$Recognize_sarcasm = revalue(sardata$Recognize_sarcasm, c("No"=0))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("Yes"=1))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("Can't say"=0))
sardata$sarcasm_ambiguity = revalue(sardata$sarcasm_ambiguity, c("No"=-1))
View(sardata)
```



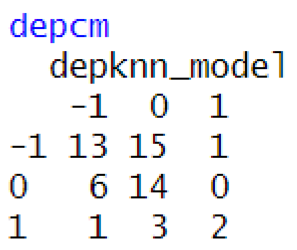
	Age	Recognize_sarcasm	sarcasm_ambiguity	sarcasm_scale
1	21	0	1	2
2	22	0	1	7
3	20	0	1	10
4	20	1	1	5
5	19	0	1	8
6	21	0	1	10
7	22	0	0	10
8	19	1	1	7
9	22	1	-1	3
10	20	1	0	4
11	19	0	0	2
12	21	0	1	1
13	22	0	0	2
14	19	0	1	8
15	21	0	1	2

The next step for the analysis is to convert the attributes into factors.



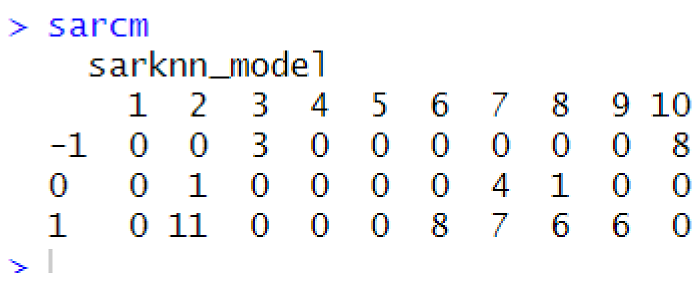
```
# Convert the column to a factor
depdata$Affect <- as.factor(depdata$Affect)
sardata$sarcasm_ambiguity <- as.factor(sardata$sarcasm_ambiguity)
```

Confusion matrix of depression dataset



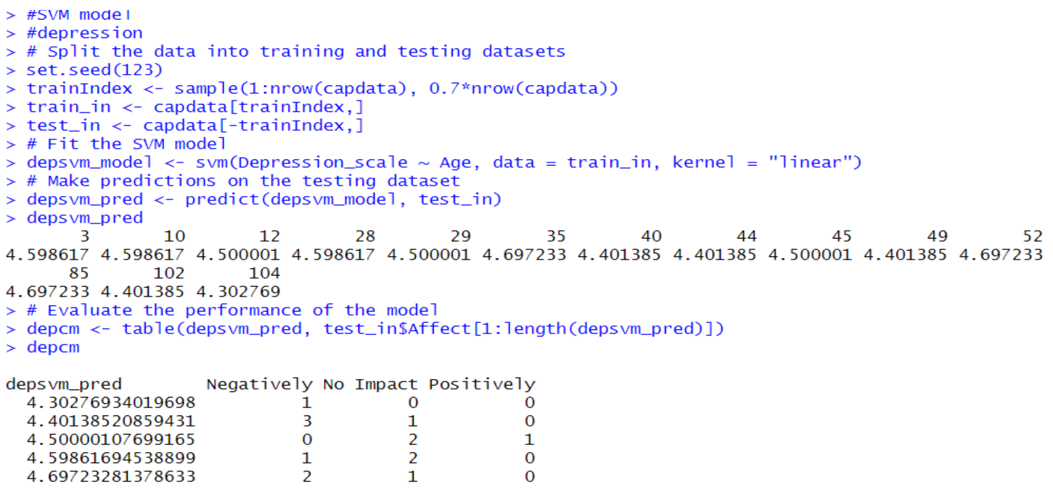
```
depcm
   depknn_model
    -1  0  1
-1  13 15  1
0    6 14  0
1    1  3  2
```

Confusion matrix of sarcasm dataset



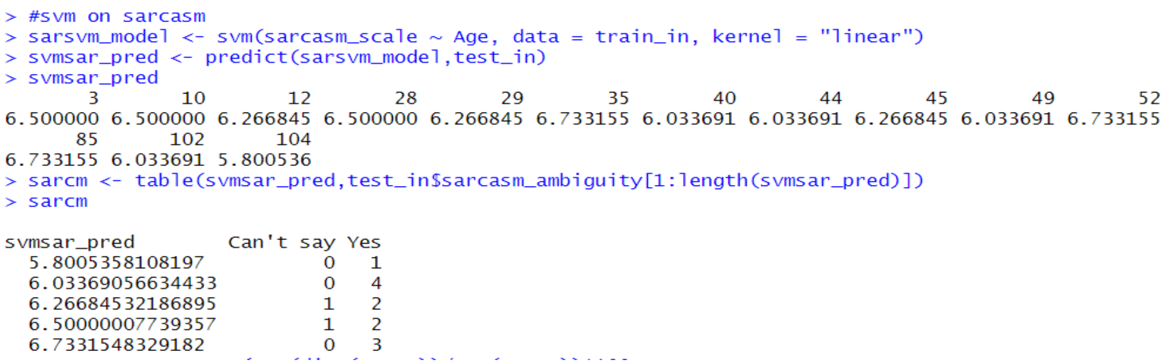
```
> sarcm
    sarknn_model
      1  2  3  4  5  6  7  8  9 10
  -1  0  0  3  0  0  0  0  0  0  8
  0   0  1  0  0  0  0  4  1  0  0
  1   0 11  0  0  0  8  7  6  6  0
> 
```

SVM model applied on depression model and the all the required output are depicted below.



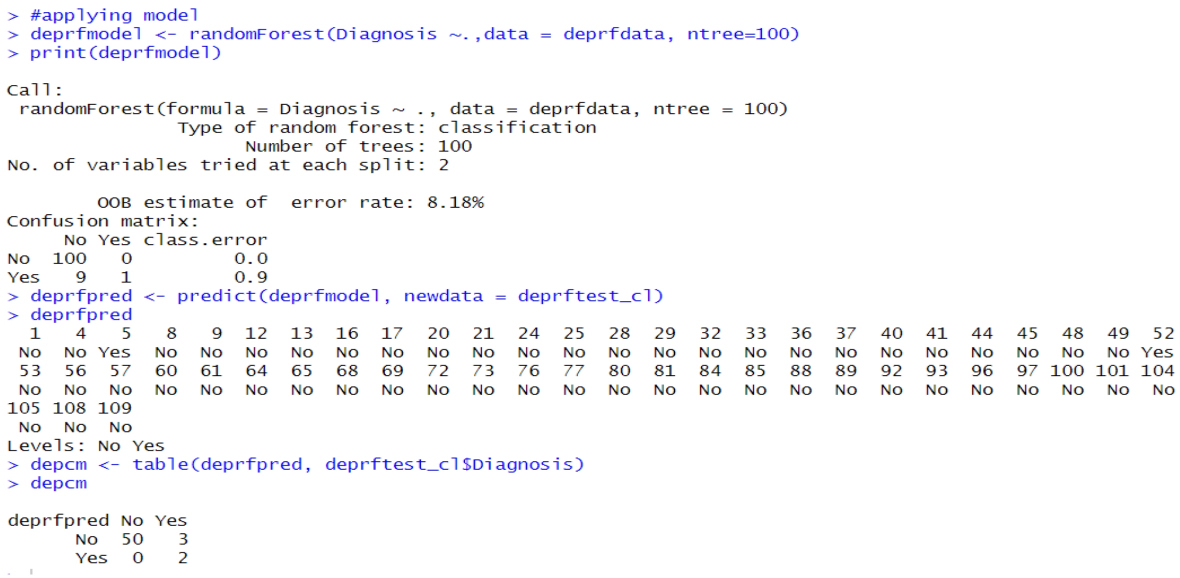
```
> #SVM model
> #depression
> # Split the data into training and testing datasets
> set.seed(123)
> trainIndex <- sample(1:nrow(capdata), 0.7*nrow(capdata))
> train_in <- capdata[trainIndex,]
> test_in <- capdata[-trainIndex,]
> # Fit the SVM model
> depsvm_model <- svm(Depression_scale ~ Age, data = train_in, kernel = "linear")
> # Make predictions on the testing dataset
> depsvm_pred <- predict(depsvm_model, test_in)
> depsvm_pred
       3       10       12       28       29       35       40       44       45       49       52 
4.598617 4.598617 4.500001 4.598617 4.500001 4.697233 4.401385 4.401385 4.500001 4.401385 4.697233 
      85      102      104 
4.697233 4.401385 4.302769 
> # Evaluate the performance of the model
> depcm <- table(depsvm_pred, test_in$Affect[1:length(depsvm_pred)])
> depcm
                  
depsvm_pred        Negatively No Impact Positively
  4.30276934019698          1         0          0
  4.40138520859431          3         1          0
  4.50000107699165          0         2          1
  4.59861694538899          1         2          0
  4.69723281378633          2         1          0
```

Implementation of SVM model on Sarcasm dataset



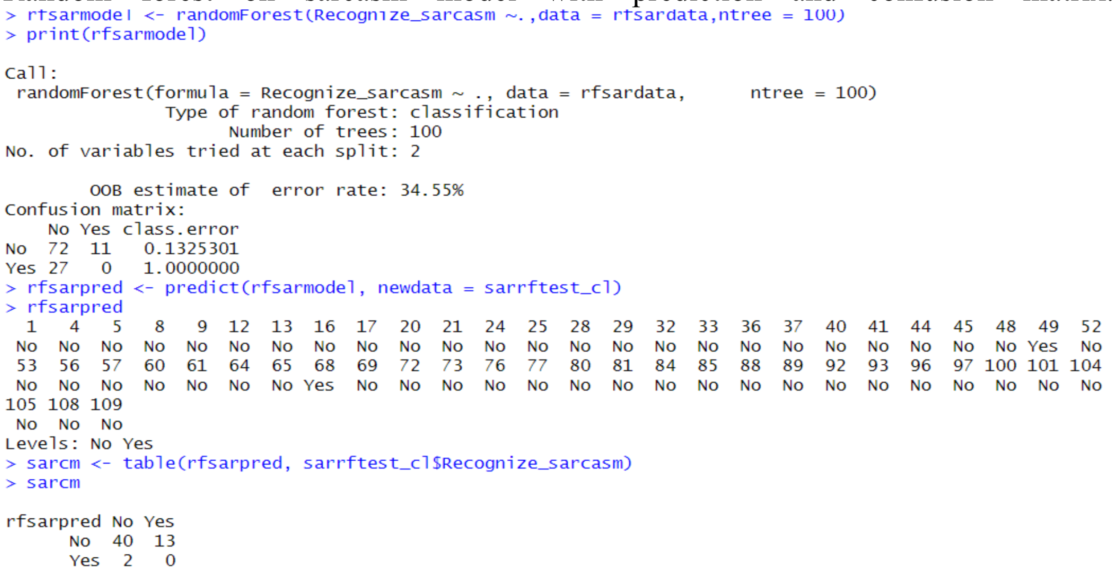
```
> #svm on sarcasm
> sarsvm_model <- svm(sarcasm_scale ~ Age, data = train_in, kernel = "linear")
> svmsar_pred <- predict(sarsvm_model,test_in)
> svmsar_pred
       3       10       12       28       29       35       40       44       45       49       52 
6.500000 6.500000 6.266845 6.500000 6.266845 6.733155 6.033691 6.033691 6.266845 6.033691 6.733155 
      85      102      104 
6.733155 6.033691 5.800536 
> sarcm <- table(svmsar_pred,test_in$sarcasm_ambiguity[1:length(svmsar_pred)])
> sarcm
                  
svmsar_pred        Can't say Yes
  5.8005358108197          0   1
  6.03369056634433         0   4
  6.26684532186895         1   2
  6.50000007739357         1   2
  6.7331548329182          0   3
```

Model Implementation of random forest on depression dataset and required prediction and confusion matrix are depicted below.



```
> #applying model
> deprfmodel <- randomForest(Diagnosis ~.,data = deprfdata, ntree=100)
> print(deprfmodel)

Call:
 randomForest(formula = Diagnosis ~ ., data = deprfdata, ntree = 100) 
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 8.18%
Confusion matrix:
     No Yes class.error
No  100   0         0.0
Yes   9   1         0.9
> deprfpred <- predict(deprfmodel, newdata = deprftest_cl)
> deprfpred
  1   4   5   8   9  12  13  16  17  20  21  24  25  28  29  32  33  36  37  40  41  44  45  48  49  52 
 No  No Yes  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No Yes 
 53  56  57  60  61  64  65  68  69  72  73  76  77  80  81  84  85  88  89  92  93  96  97 100 101 104 
 No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No 
105 108 109 
 No  No  No 
Levels: No Yes
> depcm <- table(deprfpred, deprftest_cl$Diagnosis)
> depcm
         
deprfpred No Yes
      No  50   3
      Yes  0   2
```

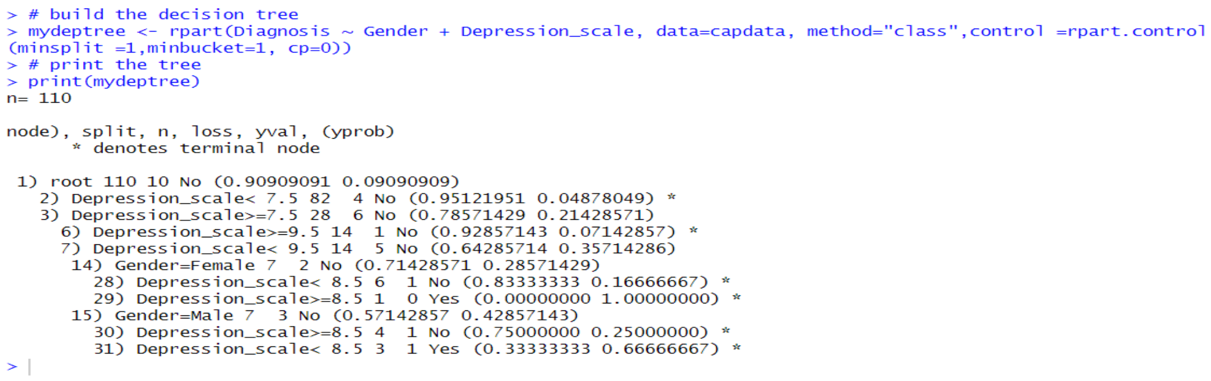
Random forest on sarcasm model with prediction and confusion matrix.



```
> rfsarmodel <- randomForest(Recognize_sarcasm ~.,data = rfsardata,ntree = 100)
> print(rfsarmodel)

Call:
 randomForest(formula = Recognize_sarcasm ~ ., data = rfsardata,      ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 34.55%
Confusion matrix:
    No Yes class.error
No  72  11   0.1325301
Yes 27   0   1.0000000
> rfsarpred <- predict(rfsarmodel, newdata = sarrftest_cl)
> rfsarpred
  1   4   5   8   9  12  13  16  17  20  21  24  25  28  29  32  33  36  37  40  41  44  45  48  49  52 
 No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No Yes  No 
 53  56  57  60  61  64  65  68  69  72  73  76  77  80  81  84  85  88  89  92  93  96  97 100 101 104 
 No  No  No  No  No  No  No Yes  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No  No 
105 108 109 
 No  No  No 
Levels: No Yes
> sarcm <- table(rfsarpred, sarrftest_cl$Recognize_sarcasm)
> sarcm
         
rfsarpred No Yes
      No  40  13
      Yes  2   0
```

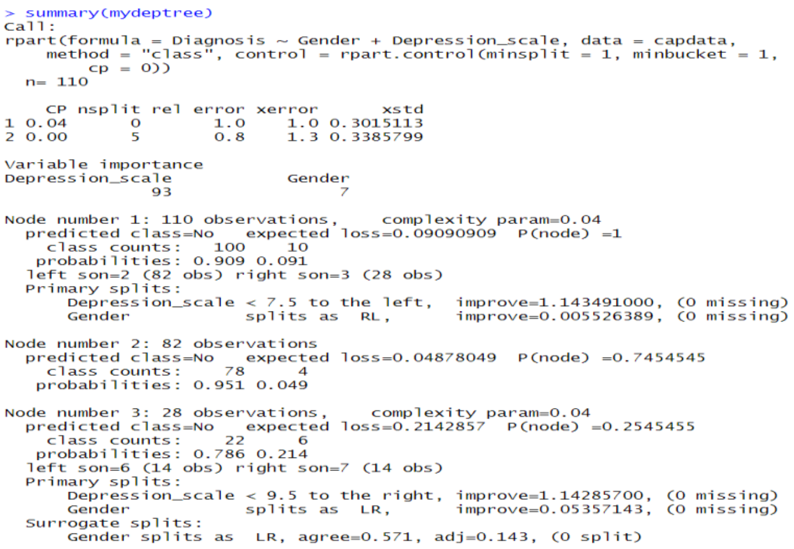
The accuracy of depression and sarcasm on random forest are 94.54% and 72.72% respectively, It is the most accurate and well predicted model amongst all the applied machine learning model.

Implementation of Decision Tree on depression dataset.



```
> # build the decision tree
> mydeptree <- rpart(Diagnosis ~ Gender + Depression_scale, data=capdata, method="class",control =rpart.control
(minsplit =1,minbucket=1, cp=0))
> # print the tree
> print(mydeptree)
n= 110 

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 110 10 No (0.90909091 0.09090909)  
   2) Depression_scale< 7.5 82  4 No (0.95121951 0.04878049) *
   3) Depression_scale>=7.5 28  6 No (0.78571429 0.21428571)  
     6) Depression_scale>=9.5 14  1 No (0.92857143 0.07142857) *
     7) Depression_scale< 9.5 14  5 No (0.64285714 0.35714286)  
      14) Gender=Female 7  2 No (0.71428571 0.28571429)  
        28) Depression_scale< 8.5 6  1 No (0.83333333 0.16666667) *
        29) Depression_scale>=8.5 1  0 Yes (0.00000000 1.00000000) *
      15) Gender=Male 7  3 No (0.57142857 0.42857143)  
        30) Depression_scale>=8.5 4  1 No (0.75000000 0.25000000) *
        31) Depression_scale< 8.5 3  1 Yes (0.33333333 0.66666667) *
> |
```

Summary of depression tree

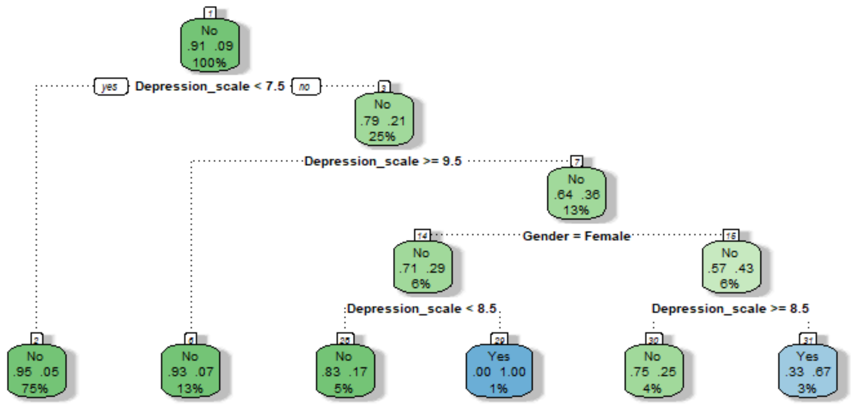


```
> summary(mydeptree)
Call:
rpart(formula = Diagnosis ~ Gender + Depression_scale, data = capdata, 
    method = "class", control = rpart.control(minsplit = 1, minbucket = 1, 
        cp = 0))
  n= 110 

    CP nsplit rel error xerror      xstd
1 0.04      0       1.0    1.0 0.3015113
2 0.00      5       0.8    1.3 0.3385799

Variable importance
Depression_scale           Gender 
              93                7 

Node number 1: 110 observations,    complexity param=0.04
  predicted class=No   expected loss=0.09090909  P(node) =1
    class counts:   100    10
   probabilities: 0.909 0.091 
  left son=2 (82 obs) right son=3 (28 obs)
  Primary splits:
      Depression_scale < 7.5 to the left,  improve=1.143491000, (0 missing)
      Gender           splits as  RL,      improve=0.005526389, (0 missing)

Node number 2: 82 observations
  predicted class=No   expected loss=0.04878049  P(node) =0.7454545
    class counts:    78     4
   probabilities: 0.951 0.049 

Node number 3: 28 observations,    complexity param=0.04
  predicted class=No   expected loss=0.2142857  P(node) =0.2545455
    class counts:    22     6
   probabilities: 0.786 0.214 
  left son=6 (14 obs) right son=7 (14 obs)
  Primary splits:
      Depression_scale < 9.5 to the right, improve=1.14285700, (0 missing)
      Gender           splits as  LR,      improve=0.05357143, (0 missing)
  Surrogate splits:
      Gender splits as  LR, agree=0.571, adj=0.143, (0 split)
```

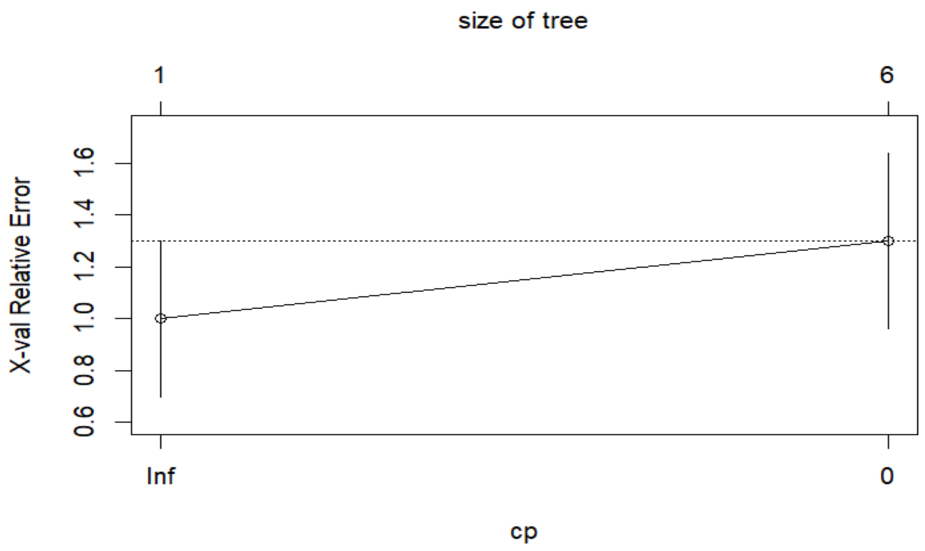
Implementation of decision tree on sarcasm dataset.

```
> print(mysartree)
n= 110

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 110 27 No (0.7545455 0.2454545)
   2) sarcasm_scale>=5.5 60 11 No (0.8166667 0.1833333) *
   3) sarcasm_scale< 5.5 50 16 No (0.6800000 0.3200000)
     6) Gender=Female 31  8 No (0.7419355 0.2580645) *
     7) Gender=Male 19  8 No (0.5789474 0.4210526)
      14) sarcasm_scale< 3.5 12  3 No (0.7500000 0.2500000) *
      15) sarcasm_scale>=3.5 7  2 Yes (0.2857143 0.7142857) *
```

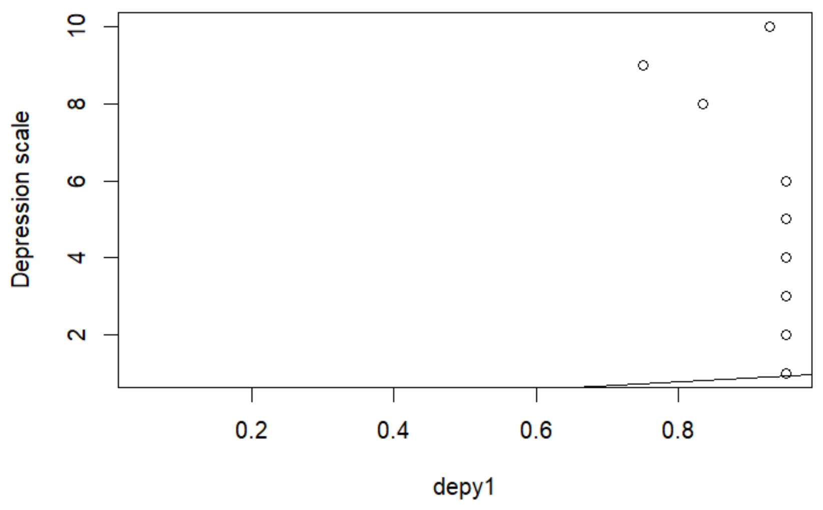
Depression tree



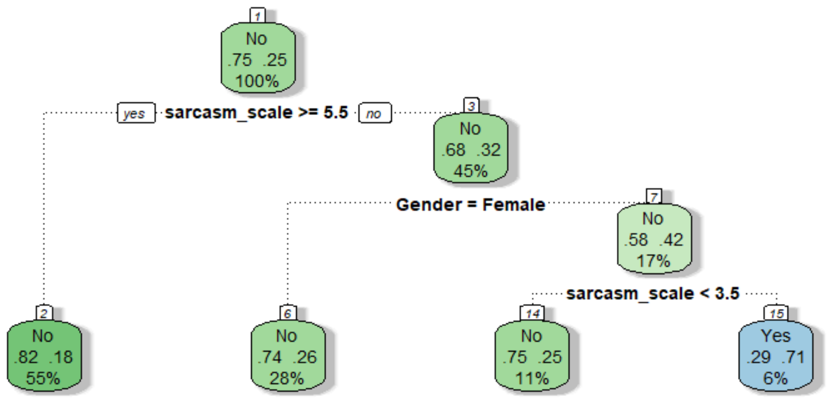
Complexity parameter of depression tree



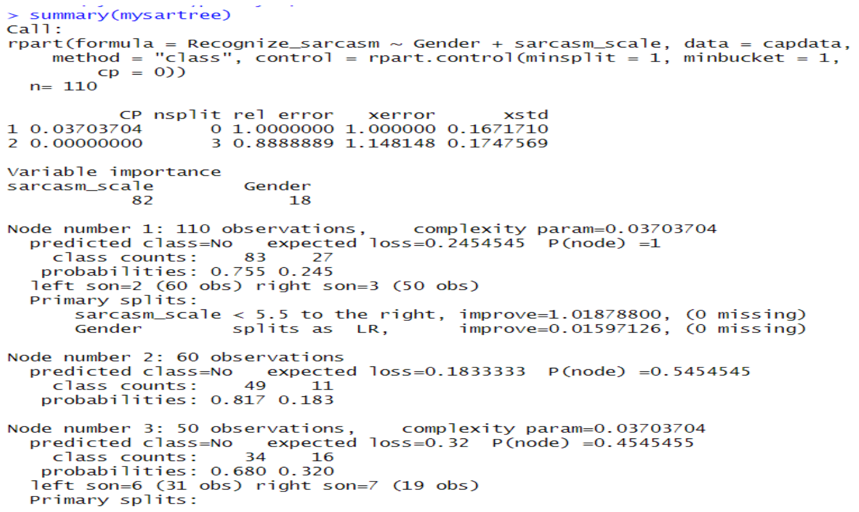
Plotting of Depression tree



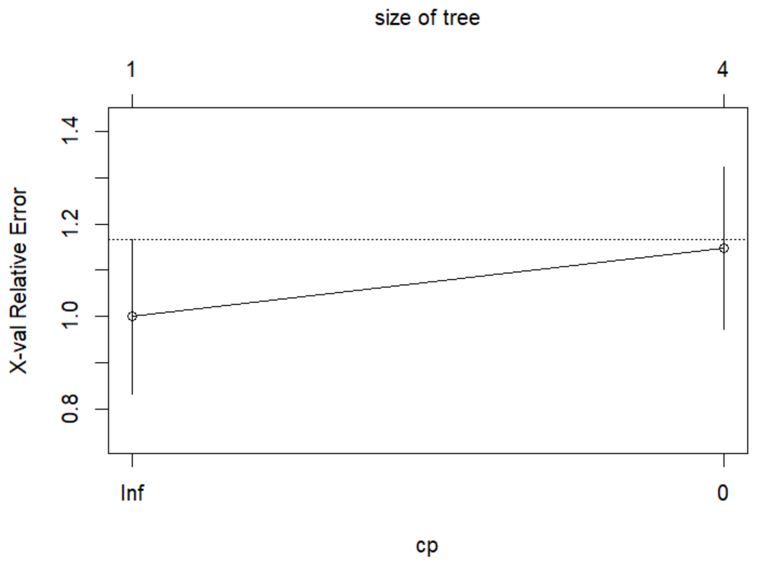
Sarcasm tree



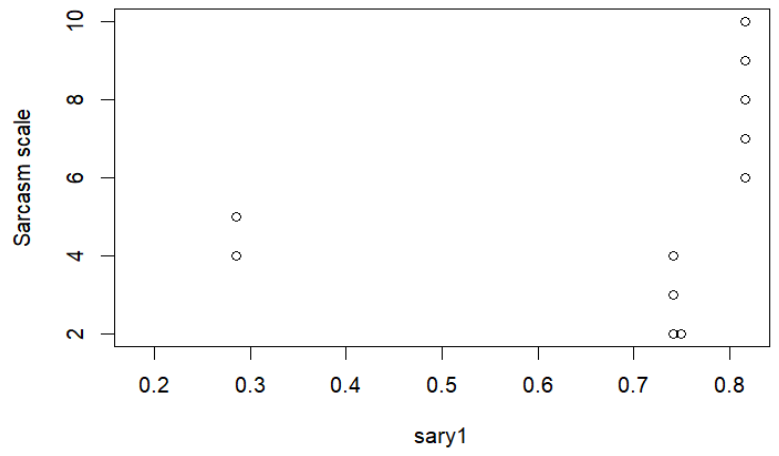
Summary of sarcasm tree.


```
> summary(mysartree)
Call:
rpart(formula = Recognize_sarcasm ~ Gender + sarcasm_scale, data = capdata,
    method = "class", control = rpart.control(minsplit = 1, minbucket = 1,
        cp = 0))
  n= 110

          CP nsplit rel error   xerror      xstd
1 0.03703704      0 1.0000000 1.000000 0.1671710
2 0.00000000      3 0.8888889 1.148148 0.1747569

Variable importance
sarcasm_scale        Gender
           82            18

Node number 1: 110 observations,    complexity param=0.03703704
  predicted class=No   expected loss=0.2454545  P(node) =1
    class counts:    83    27
   probabilities: 0.755 0.245
  left son=2 (60 obs) right son=3 (50 obs)
  Primary splits:
      sarcasm_scale < 5.5 to the right, improve=1.01878800, (0 missing)
      Gender        splits as  LR,      improve=0.01597126, (0 missing)

Node number 2: 60 observations
  predicted class=No   expected loss=0.1833333  P(node) =0.5454545
    class counts:    49    11
   probabilities: 0.817 0.183

Node number 3: 50 observations,    complexity param=0.03703704
  predicted class=No   expected loss=0.32  P(node) =0.4545455
    class counts:    34    16
   probabilities: 0.680 0.320
  left son=6 (31 obs) right son=7 (19 obs)
  Primary splits:
```

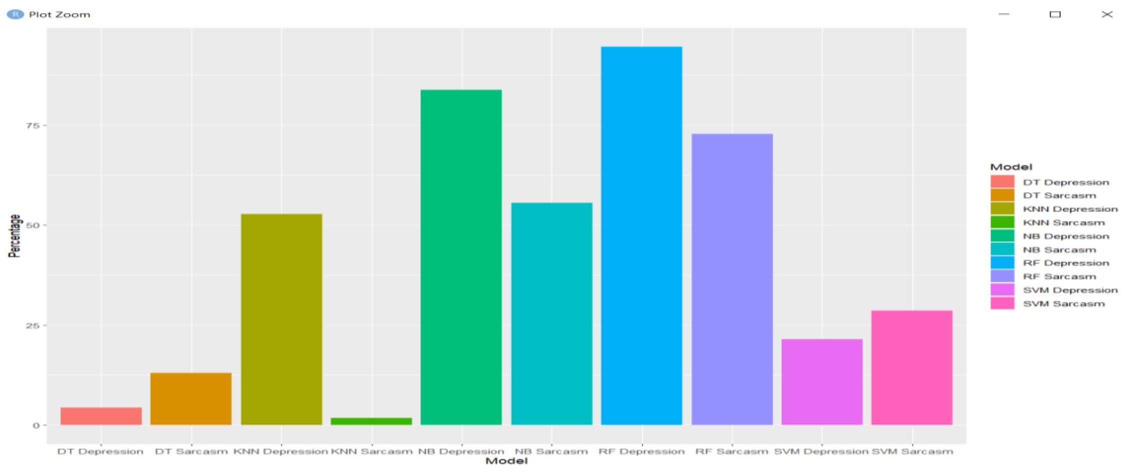
Complexity parameter of depression tree



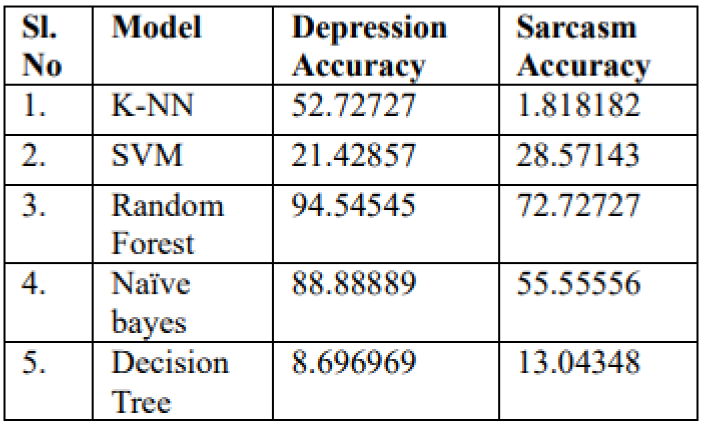
Plotting of Sarcasm Tree



Comparison of All the Algorithms used on both the datasets (Depression & Sarcasm).



Here is the tabular comparison of all the algorithms that we have used on our respective data.



Sl. No	Model	Depression Accuracy	Sarcasm Accuracy
1.	K-NN	52.72727	1.818182
2.	SVM	21.42857	28.57143
3.	Random Forest	94.54545	72.72727
4.	Naïve bayes	88.88889	55.55556
5.	Decision Tree	8.696969	13.04348

11. BIBLIOGRAPHY

K. Dashtipour, M. Gogate, A. Adeel, H. Larijani, and A. Hussain, “Sentiment analysis of persian movie reviews using deep learning,” *Entropy*, vol. 23, no. 5, 2021, doi: 10.3390/e23050596.

[2] M. R. Islam, M. A. Kabir, A. Ahmed, A. R. M. Kamal, H. Wang, and A. Ulhaq, “Depression detection from social network data using machine learning techniques,” *Health Inf Sci Syst*, vol. 6, no. 1, Dec. 2018, doi: 10.1007/s13755-018-0046-0.

[3] “WHO (World Health Organization)”, Accessed: Apr. 06, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/depression

[1] K. Dashtipour, M. Gogate, A. Adeel, H. Larijani, and A. Hussain, “Sentiment analysis of persian movie reviews using deep learning,” *Entropy*, vol. 23, no. 5, 2021, doi: 10.3390/e23050596.

[2] M. R. Islam, M. A. Kabir, A. Ahmed, A. R. M. Kamal, H. Wang, and A. Ulhaq, “Depression detection from social network data using machine learning techniques,” *Health Inf Sci Syst*, vol. 6, no. 1, Dec. 2018, doi: 10.1007/s13755-018-0046-0.

[3] “WHO (World Health Organization)”, Accessed: Apr. 06, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/depression

[4] “The Daily star”, Accessed: Apr. 05, 2023. [Online]. Available: The Daily Star. (2019). High prevalence of depression and anxiety dis-orders. [online] Available at: https://www.thedailystar.net/editorial/high-prevalence-depression-and-anxiety-disorders-1367674

[5] S. Kumar, M. Gahalawat, P. P. Roy, D. P. Dogra, and B. G. Kim, “Exploring impact of age and gender on sentiment analysis using machine learning,” *Electronics (Switzerland)*, vol. 9, no. 2, Feb. 2020, doi: 10.3390/electronics9020374.

[6] A. Naresh and P. Venkata Krishna, “An efficient approach for sentiment analysis using machine learning algorithm,” *Evol Intell*, vol. 14, no. 2, pp. 725–731, Jun. 2021, doi: 10.1007/s12065-020-00429-1.

[7] M. S. Razali, A. A. Halin, L. Ye, S. Doraisamy, and N. M. Norowi, “Sarcasm Detection Using Deep Learning with Contextual Features,” *IEEE Access*, vol. 9, pp. 68609–68618, 2021, doi: 10.1109/ACCESS.2021.3076789.

[8] I. Ali Kandhro, M. Ameen Chhajro, K. Kumar, H. N. Lashari, and U. Khan, “Student Feedback Sentiment Analysis Model Using Various Machine Learning Schemes A Review,” *Indian J Sci Technol*, vol. 14, no. 12, pp. 1–9, Apr. 2019, doi: 10.17485/ijst/2019/v12i14/143243.

[9] Y. Lee *et al.*, “Applications of machine learning algorithms to predict therapeutic outcomes in depression: A meta-analysis and systematic review,” *Journal of Affective Disorders*, vol. 241. Elsevier B.V., pp. 519–532, Dec. 01, 2018. doi: 10.1016/j.jad.2018.08.073.

[10] J. Singh, G. Singh, and R. Singh, “Optimization of sentiment analysis using machine learning classifiers,” *Human-centric Computing and Information Sciences*, vol. 7, no. 1, Dec. 2017, doi: 10.1186/s13673-017-0116-3.

[11] M. Sykora, S. Elayan, and T. W. Jackson, “A qualitative analysis of sarcasm, irony and related #hashtags on Twitter,” *Big Data Soc*, vol. 7, no. 2, 2020, doi: 10.1177/2053951720972735.

[12] H. Yu, E. Lee, and S. B. Lee, “SymBiosis: Anti-Censorship and Anonymous Web-Browsing Ecosystem,” *IEEE Access*, vol. 4, pp. 3547–3556, 2016, doi: 10.1109/ACCESS.2016.2585163.

[13] B. Le and H. Nguyen, “Twitter sentiment analysis using machine learning techniques,” in *Advances in Intelligent Systems and Computing*, Springer Verlag, 2015, pp. 279–289. doi: 10.1007/978-3-319-17996-4_25.

[14] P. Baid, A. Gupta, and N. Chaplot, “Sentiment Analysis of Movie Reviews using Machine Learning Techniques,” *Int J Comput Appl*, vol. 179, no. 7, pp. 45–49, Dec. 2017, doi: 10.5120/ijca2017916005.

[15] I. A. Kandhro, S. Wasi, K. Kumar, M. Rind, and M. Ameen, “Sentiment Analysis of Student�s Comment by using Long-Short Term Model,” *Indian J Sci Technol*, vol. 12, no. 8, pp. 1–16, Feb. 2019, doi: 10.17485/ijst/2019/v12i8/141741.

[16] A. Mitra, “Sentiment Analysis Using Machine Learning Approaches (Lexicon based on movie review dataset),” *Journal of Ubiquitous Computing and Communication Technologies*, vol. 2, no. 3, pp. 145–152, Sep. 2020, doi: 10.36548/jucct.2020.3.004.

[17] A. Rahman and M. S. Hossen, “Sentiment Analysis on Movie Review Data Using Machine Learning Approach,” in *2019 International Conference on Bangla Speech and Language Processing, ICBSLP 2019*, Institute of Electrical and Electronics Engineers Inc., Sep. 2019. doi: 10.1109/ICBSLP47725.2019.201470.