

MULTILEVEL REVIEW CLASSIFICATION USING ML

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

Kavitha S (17SCS0421)

Rithikaa.K(17SCS5173)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC

Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI - 600 119

March - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC | 12B Status by UGC

Approved by AICTE

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Kavitha.S (17SCS0421)& Rithikaa.K(17SCS5173) who have done the Project work as a team who carried out the project entitled "Multilevel Review Classification using Machine Learning" under my supervision from November 2020 to March 2021.

Ms. Yogitha. R, M.E., Ph.d.

Internal Guide

Head of the Department
Dr.L.Lakshmanan M.E., Ph.D
Dr. S. VIGNESHWARI, M.E., Ph.D.

Submitted for Viva voce Examination held on

Internal Examiner

External Examiner

DECLARATION

I Rithikaa.K (37110644) & Kavitha.S (37110687) hereby declare that the Project Report entitled “Multilevel Review Classification using Machine Learning” done by me under the guidance of Dr. /Prof./ Mr./Ms. _____(Internal) and _____(External) at Sathyabama Institute of Science & Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in_.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D .**, Dean, School of Computing and **Dr.L.Lakshmanan M.E., Ph.D.,and Dr.S.Vigneshwari M.E., Ph.D.**, Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide Ms. Yogitha. R, M.E., Ph.d. for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Play Store is Google's official pre-installed app store on Android-certified devices. It provides access to content on the Google Play Store, including apps, books, magazines, and music, movies, and television programs. Google play store allow the user to download a mobile application and user get inspired by the rating and reviews of the mobile app. A recent study analyzes that user preferences, user opinion for improvement, user sentiment about particular feature and detail with descriptions of experiences are very useful for an application development. The aim is to classify the GOOGLE app reviews based on supervised machine learning techniques (SMLT).The analysis of dataset by supervised machine learning technique (SMLT) to capture several information's like, variable identification, uni- variate analysis, bi-variate and multi-variate analysis, missing value treatments and analyze the data validation, data cleaning/preparing and data visualization will be done on the entire given dataset. To propose a machine learning-based method to classify the google play store reviews results in the form of positive, neutral or negative best accuracy from comparing supervise classification machine learning algorithms.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	
	LIST OF FIGURES	
	LIST OF ABBREVIATIONS	
1	INTRODUCTION	1
	1.1 INTRODUCTION	1
	1.2 DOMAIN OVERVIEW	2
	1.3 USE OF THE DOMAIN IN THE SOCIETY	2
2	AIM & SCOPE OF THE PROJECT	3
	2.1 AIM	3
	2.2 SCOPE	3
	2.3 OBJECTIVE	3
3	SYSTEM DESIGN AND METHODOLOGY	4
	3.1 EXISTING SYSTEM	4
	3.2 PROPOSED SYSTEM	4
	3.3 PROJECT REQUIREMENTS	5
	3.3 SYSTEM ARCHITECTURE	6
	3.4 MODULES AND ITS DESCRIPTION	7
4	SOFTWARE DESCRIPTION AND DEVELOPMENT	11
	4.1 ANACONDA NAVIGATOR	12
	4.2 WORKING PROCESS	15
	4.3FEASIBILITY STUDY	17

	4.3.1 DATA WRANGLING	17
	4.3.2 DATA COLLECTION	17
	4.3.3 PREPROCESSING THE DATA	18
	4.3.4 BUILDING THE MODEL	18
	4.3.5 PREDICTION MODEL	18
	4.4 WORKFLOW DIAGRAM	19
	4.4.1 USE CASE DIAGRAM	20
	4.4.2 CLASS DIAGRAM	21
	4.4.3 ACTIVITY DIAGRAM	23
	4.4.4 SEQUENCE DIAGRAM	24
	4.4.5 ERD	25
5	ALGORITHMS AND TECHNIQUES USED	27
	5.1 ALGORITHM EXPLANATION	27
	5.2 LOGISTIC REGRESSION	28
	5.3 KNN	30
	5.4 RANDOM FOREST	31
	5.5 SVM	32
	5.6 DEEP LEARNING RNN	32
	5.7 OUTPUT PREDICTION OF SENTIMENT	34
6	RESULTS AND DISCUSSIONS	35
	6.1 RESULT	35
	6.2 CONCLUSION	35
	6.3 FUTURE WORK	37
	REFERENCES	39

SOURCE CODE	43
SCREENSHOTS	50

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
1.1	PROCESS OF MACHINE LEARNING	2
3.1	SYSTEM ARCHITECTURE	6
4.1	PROCESS OF DATA FLOW	19
4.2	WORKFLOW DIAGRAM	19
4.3	USE CASE DIAGRAM	20
4.4	CLASS DIAGRAM	21
4.5	ACTIVITY DIAGRAM	22
4.6	SEQUENCE DIAGRAM	23
4.7	ENTITY RELATIONSHIP DIAGRAM	24

LIST OF ABBREVIATIONS

MAE	MEAN ABSOLUT ERROR
EDA	EXPLORATORY DATA ANALYSIS
RNN	RECURRENT NEURAL NETWORK
LSTM	LONG SHORT-TERM MEMORY
GUI	GRAPHICAL USER INTERFACE

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION:

As there are innumerable products manufactured by many different brands, so providing relevant reviews to the consumers is the need of hour. Number of reviews associated with a product or a brand is increasing at an alarming rate, which is no less than handling the big data. Classifying the reviews on the basis of sentiment of customers into positive and negative sentiment provides sentiment orientation of the review, hence results in better judgement. Segregation of reviews on the basis of their sentiment can help future buyers to evaluate positive and negative feedback constructively and reach at better decisions as per their requirements. This evaluation acts as a testimony to the users who are looking to know the details and specifications of the smartphones; thereby increasing user credibility.

1.2 DOMAIN OVERVIEW:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values.

Unsupervised machine learning algorithms are used when the information used to train is neither classified nor labeled.

Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data.

There are many applications for machine learning, including:

Agriculture

Marketing

Online advertising

Medical Diagnosis

Search Engines etc.

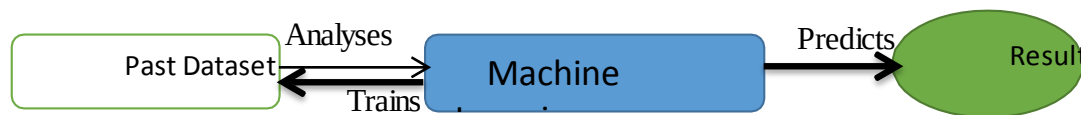


Fig:1.1 Process of Machine learning

1.2USE OF THE DOMAIN IN THE SOCIETY:

The use of text-mining tools and machine learning (ML) algorithms to aid systematic review is becoming an increasingly popular approach to reduce human burden and monetary resources required and to reduce the time taken to complete such reviews.

CHAPTER 2

AIM &SCOPE OF THE PROJECT

2.1 AIM:

To propose a machine learning-based method to classify the google play store reviews results in the form of positive, neutral or negative best accuracy from comparing supervise classification machine learning algorithms.

2.2 SCOPE:

Sentiment Analysis is not limited just to reviews or but is also applicable on stock markets, news articles or political debates. Sentiment analysis can be used to flourish consumer products related business. It uses rule-based approach for sentiment analysis to extract topic words of negative opinion sentences and thus promote the competitors of the products receiving negative feedback. Similarly, relevant ads based on a person's liking or disliking are displayed on various blogging sites targeting bloggers. Blogger-Centric Contextual Advertising Framework has been concocted to determine users' personal interests and display those ads that intersect with them.

2.2 OBJECTIVE:

The goal is to develop a machine learning model for classifying the reviews from google play store, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithm.

CHAPTER 3

SYSTEM DESIGN AND METHODOLOGY

3.1 EXISTING SYSTEM:

Feature-level ratings of the mobile products were used to influence decision-making from the customer reviews and review votes, both for new customers and manufacturers. Feature level rating is more particular than product level reviews. A system was developed to rate mobile phones in terms of 108 features based on customer reviews and review votes. They could rate around 4000+phones.They had taken the features which are mostly voted in the dataset. They accomplished this by first converting the unstructured data into structured data; then they extract the sentences comprising our feature keywords then they were able to provide the feature-level ratings through sentiment analysis of these sentences. Ranking the phones based on the number of features that they are best are done and accordingly they were able to recommend the best phones for a feature. The performance of their method was found to be decent and they obtain MAE of only 0.555, i.e., approximately just half a star. We get 52.3% accuracy if exact integer ratings have to be predicted.

Drawbacks:

- Their approach was unsupervised learning method.
- The accuracy was low.

3.2 PROPOSED SYSTEM:

● **Exploratory Data Analysis**

Machine learning supervised classification algorithms will be used to give the given dataset and extract patterns, which would help in classifying the reviews, thereby helping the apps for making better decisions of their features in the future.

● **Data Wrangling:**

In this section of the report, you will load in the data, check for cleanliness, and then trim and clean your dataset for analysis.

● **Data collection:**

The data set collected for classifying the given data is split into Training set and Test set. Generally, 70:30 percentage are applied to split the Training set and Test set. The Data Model which was created using the SMLT is applied on the Training set and based on the test result accuracy, Test set prediction is done.

Advantages:

- These reports are to the investigation of applicability of machine learning techniques to classify the reviews.
- Finally, it highlights some observations on future research issues, challenges, and needs.

3.3 PROJECT REQUIREMENTS:

Requirements are the basic constraints that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environment requirements
 - A. Hardware requirements
 - B. software requirements

Functional requirements:

The software requirements specification is a technical specification of requirements for the software product. It is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like sk-learn, pandas, numpy, matplotlib and seaborn.

Non-Functional Requirements:

Process of functional steps,

1. Problem defines
2. Preparing data

3. Evaluating algorithms
4. Improving results
5. Prediction of the result

Environmental Requirements:

1. Software Requirements:

- ☐ Operating System : Windows
- ☐ Tool : Anaconda with Jupyter Notebook

2. Hardware requirements:

- ☐ Processor : Pentium IV/III
- ☐ Hard disk : minimum 80 GB
- ☐ RAM : minimum 2 GB

3.4 SYSTEM ARCHITECTURE:

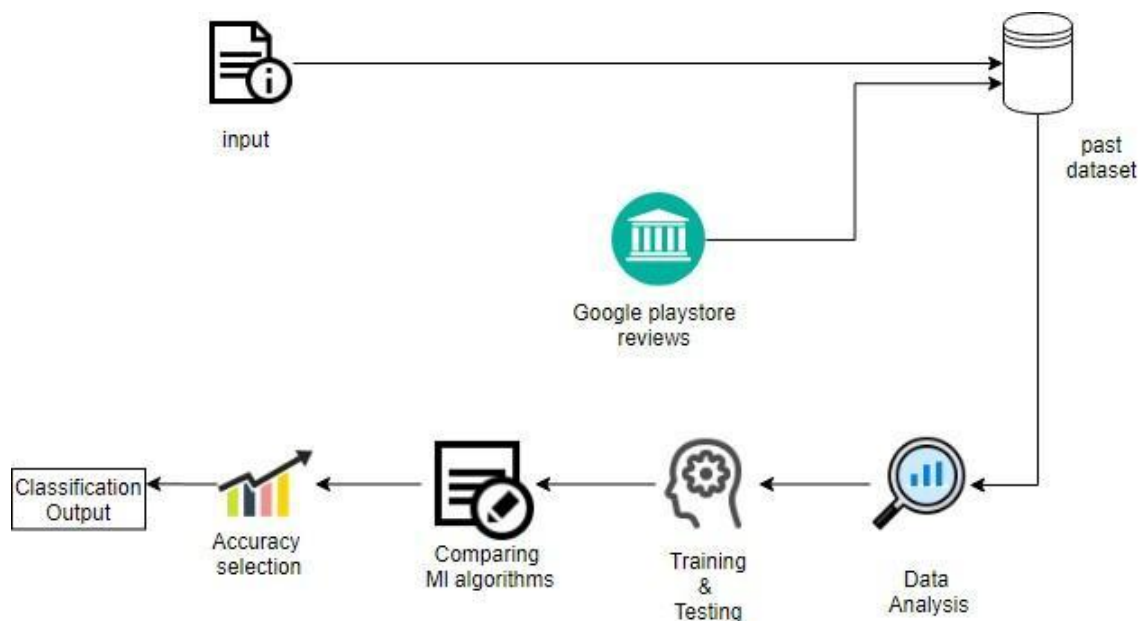


Fig3.1 SYSTEM ARCHITECTURE

3.5 MODULES and its DESCRIPTIONS:

List of Modules:

- a)Data validation Process EDA b)Exploration data analysis of visualization
- c)Comparing Algorithm with prediction in the form of best accuracy result
- d)Deep Learning RNN with LSTM get best accuracy result
- e) Output for prediction of sentiment by giving input sentences.

a)MODULE 1: DATA VALIDATION PROCESS :

Importing the library packages with loading given dataset. To analyzing the variable identification by data shape, data type and evaluating the missing values, duplicate values. A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill while tuning model's and procedures that you can use to make the best use of validation and test datasets when evaluating your models. Data cleaning / preparing by rename the given dataset and drop the column etc. to analyze the uni-variate, bi-variate and multi-variate process. The steps and techniques for data cleaning will vary from dataset to dataset. The primary goal of data cleaning is to detect and remove errors and anomalies to increase the value of data in analytics and decision making. Before, joint into code, it's important to understand the sources of missing data. Here are some typical reasons why data is missing:

- User forgot to fill in a field.
- Data was lost while transferring manually from a legacy database.
- There was a programming error.
- Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.

Variable identification with Uni-variate, Bi-variate and Multi-variate analysis:

- ☐ import libraries for access and functional purpose and read the given dataset
- ☐ General Properties of Analyzing the given dataset
- ☐ Display the given dataset in the form of data frame
- ☐ show columns
- ☐ shape of the data frame
- ☐ To describe the data frame
- ☐ Checking data type and information about dataset
- ☐ Checking for duplicate data
- ☐ Checking Missing values of data frame
- ☐ Checking unique values of data frame
- ☐ Checking count values of data frame
- ☐ Rename and drop the given data frame
- ☐ To specify the type of values
- ☐ To create extra columns

b) EXPLORATION DATA ANALYSIS OF VISUALIZATION:

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data

analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

c) MODULE 3: COMPARING ALGORITHM WITH PREDICTION IN THE FORM OF BEST ACCURACY OF RESULT:

It is important to compare the performance of multiple different machine learning algorithms consistently and it will discover to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. When have a new dataset, it is a good idea to visualize the data using different techniques in order to look at the data from different perspectives. The same idea applies to model selection. You should use a number of different ways of looking at the estimated accuracy of your machine learning algorithms in order to choose the one or two to finalize. A way to do this is to use different visualization methods to show the average accuracy, variance and other properties of the distribution of model accuracies.

d) MODULE 4: DEEP LEARNING WITH RNN:

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

input. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. In other neural networks, all the inputs are independent of each other. But in RNN, all the inputs are related to each other. First, it takes the $X(0)$ from the sequence of input and then it outputs $h(0)$ which together with $X(1)$ is the input for the next step. So, the $h(0)$ and $X(1)$ is the input for the next step. Similarly, $h(1)$ from the next is the input with $X(2)$ for the next step and so on. This way, it keeps remembering the context while training.

e)MODULE 5: OUTPUT FOR PREDICTION OF SENTIMENTAL ANALYSIS :

code2vec is a neural model that learns analogies relevant to source code. The model was trained on the Java code database but you can apply it to any codebase. Then there's GloVe. the GloVe vocabularies from the website. We took the largest one because then there's a higher chance of it finding all of our words. You can choose where you want to download it but, for convenience, it's better to store it in the working directory, Now we can find out-of-vocabulary words and count the percentage of these words for code2vec vocabulary. The following code will also work for GloVe. We've tried three different word embeddings algorithms for OpenAPI specification. Despite the fact that all three perform quite well on this dataset, an extra comparison of the most similar words shows.

CHAPTER 4

SOFTWARE DESCRIPTION AND DEVELOPMENT

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system “Conda”. The Anaconda distribution is used by over 12 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and MacOS. So, Anaconda distribution comes with more than 1,400 packages as well as the Conda package and virtual environment manager called Anaconda Navigator and it eliminates the need to learn to install each library independently. The open source packages can be individually installed from the

Anaconda repository with the `conda install` command or using the `pip install` command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in most cases they can work together. Custom packages can be made using the `conda build` command, and can be shared with others by uploading them to Anaconda Cloud,^[10] PyPI or other repositories. The default installation of Anaconda2 includes Python

2.7 and Anaconda3 includes Python 3.7. However, you can create new environments that include any version of Python packaged with conda.

4.1 Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glueviz
- Orange
- Rstudio
- Visual Studio Code

Conda:

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-languages. The Conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

The Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Notebook document:

Notebook documents (or “notebooks”, all lower case) are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc...). Notebook documents

are both human-readable documents containing the analysis description and the results (figures, tables, etc.) as well as executable documents which can be run to perform data analysis.

Jupyter Notebook App:

The *Jupyter Notebook App* is a server-client application that allows editing and running notebook documents via a web browser. The *Jupyter Notebook App* can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. In addition to displaying/editing/running notebook documents, the *Jupyter Notebook App* has a “Dashboard” (Notebook Dashboard), a “control panel” showing local files and allowing to open notebook documents or shutting down their kernels.

kernel:

A notebook *kernel* is a “computational engine” that executes the code contained in a Notebook document. The *ipython kernel*, referenced in this guide, executes python code. Kernels for many other languages exist (official kernels). When you open a Notebook document, the associated *kernel* is automatically launched. When the notebook is *executed* (either cell-by-cell or with menu *Cell -> Run All*), the *kernel* performs the computation and produces the results. Depending on the type of computations, the *kernel* may

consume significant CPU and RAM. Note that the RAM is not released until the *kernel* is shut-down.

Notebook Dashboard:

The *Notebook Dashboard* is the component which is shown first when you launch Jupyter Notebook App. The *Notebook Dashboard* is mainly used to open notebook documents, and to manage the running kernels (visualize and shutdown). The *Notebook Dashboard* has other features similar to a file manager, namely navigating folders and renaming/deleting files.

4.2 Working Process:

- ☐ Download and install anaconda and get the most useful package for machine learning in Python.
- ☐ Load a dataset and understand its structure using statistical summaries and data visualization.
- ☐ machine learning models, pick the best and build confidence that the accuracy is reliable.

Python is a popular and powerful interpreted language. Unlike R, Python is a complete language and platform that you can use for both research and development and developing production systems. There are also a lot of modules and libraries to choose

from, providing multiple ways to do each task. It can feel overwhelming.

The best way to get started using Python for machine learning is to complete a project.

- It will force you to install and start the Python interpreter (at the very least).
- It will give you a bird's eye view of how to step through a small project.
- It will give you confidence, maybe to go on to your own small projects.

When you are applying machine learning to your own datasets, you are working on a project. A machine learning project may not be linear, but it has a number of well-known steps:

- Define Problem.
- Prepare Data.
- Evaluate Algorithms.
- Improve Results.
- Present Results.

The best way to really come to terms with a new platform or tool is to work through a machine learning project end-to-end and cover the key steps. Namely, from loading data, summarizing data, evaluating algorithms and making some predictions.

Here is an overview of what we are going to cover:

1. Installing the Python anaconda platform.
2. Loading the dataset.
3. Summarizing the dataset.
4. Visualizing the dataset.
5. Evaluating some algorithms.
6. Making some predictions.

4.3 Feasibility study

4.3.1 Data Wrangling

In this section of the report will load in the data, check for cleanliness, and then trim and clean given dataset for analysis. Make sure that the document steps carefully and justify for cleaning decisions.

4.3.2 Data collection

The data set collected for predicting given data is split into Training set and Test set. Generally, 7:3 ratios are applied to split the Training set and Test set. The Data Model which was created using Random Forest, logistic, Decision tree algorithms, K-Nearest Neighbor (KNN) and Support vector classifier (SVC) are applied on

the Training set and based on the test result accuracy, Test set prediction is done.

4.3.3 Preprocessing

The data which was collected might contain missing values that may lead to inconsistency. To gain better results data need to be preprocessed so as to improve the efficiency of the algorithm. The outliers have to be removed and also variable conversion need to be done.

4.3.4 Building the classification model

The predicting the air quality problem, decision tree algorithm prediction model is effective because of the following reasons: It provides better results in classification problem.

- It is strong in preprocessing outliers, irrelevant variables, and a mix of continuous, categorical and discrete variables.
- It produces out of bag estimate error which has proven to be unbiased in many tests and it is relatively easy to tune with.

4.3.5 Construction of a Predictive Model

Machine learning needs data gathering have lot of past data's. Data gathering have sufficient historical data and raw data. Before data pre-processing, raw data can't be used directly. It's used to preprocess then, what kind of algorithm with model. Training and testing this model working and predicting correctly with minimum errors. Tuned model involved by tuned time to time with improving the accuracy.



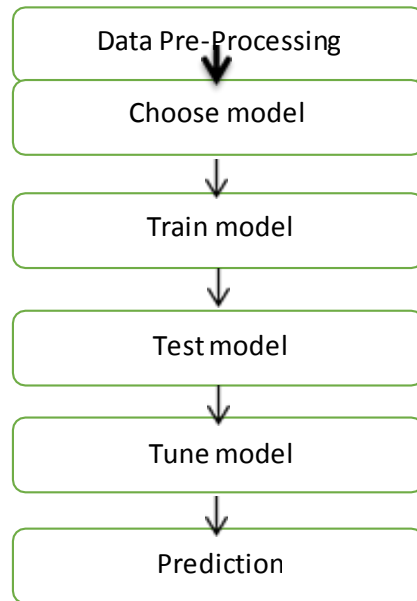
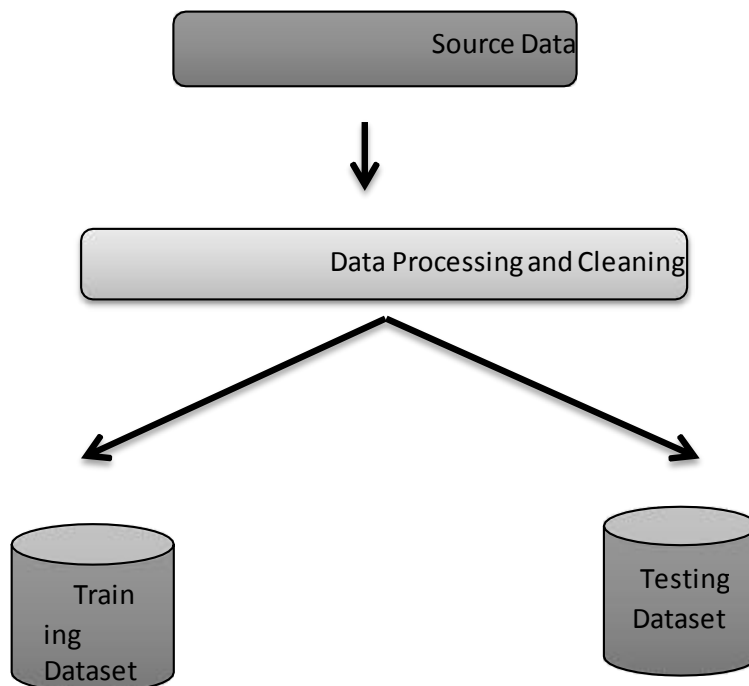


Fig:4.1 Process of dataflow diagram

4.4 Work flow diagram



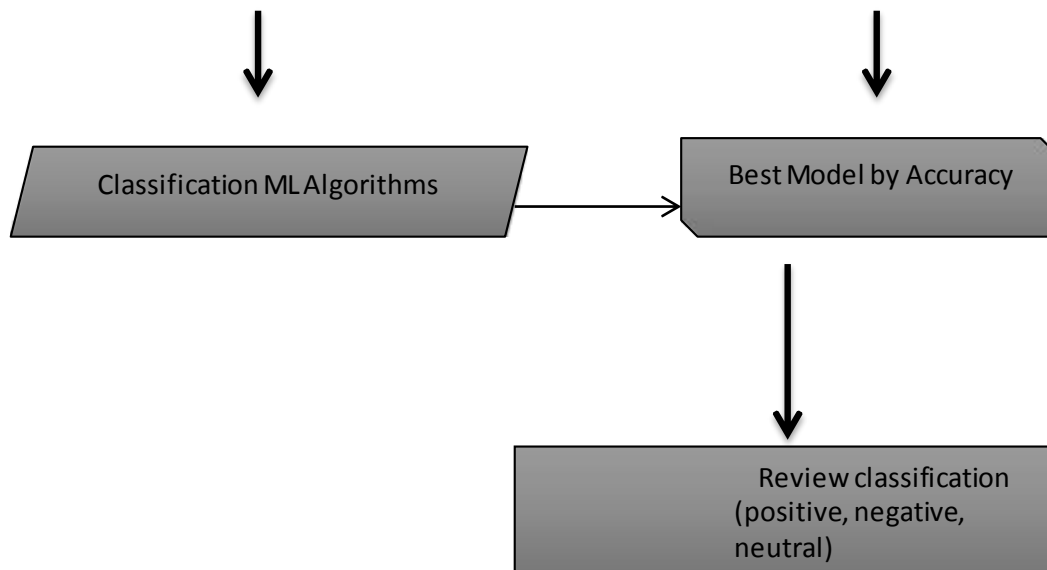


Fig:4.2 Work flow diagram

4.5 Use Case Diagram

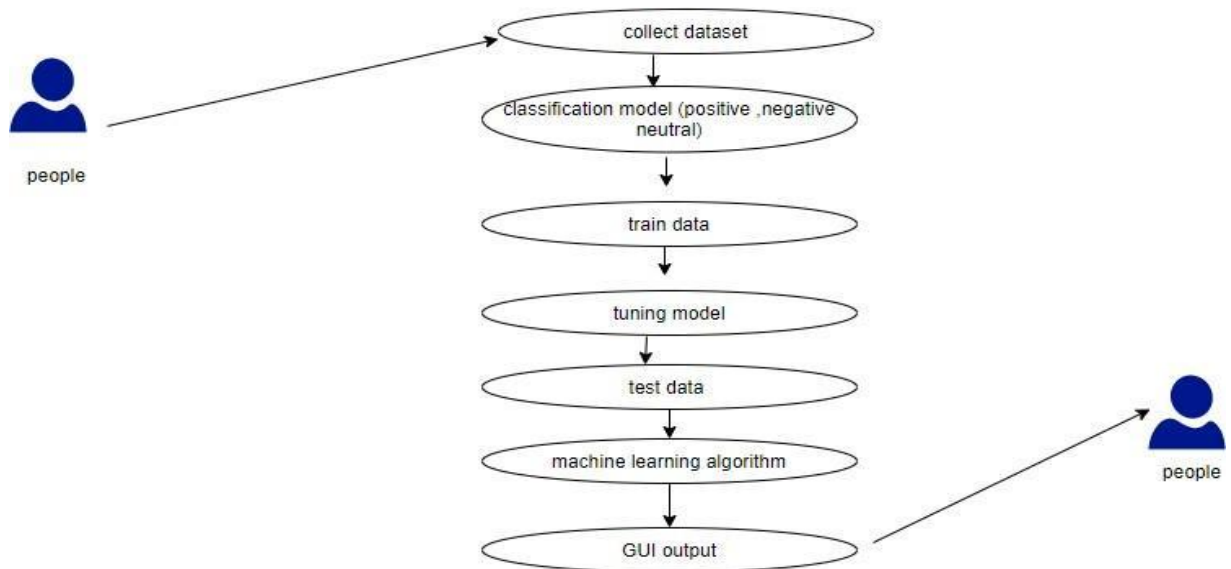


Fig:4.3 Use case diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So, it can say that use cases are nothing but the system functionalities written in an organized manner.

4.6 Class Diagram:

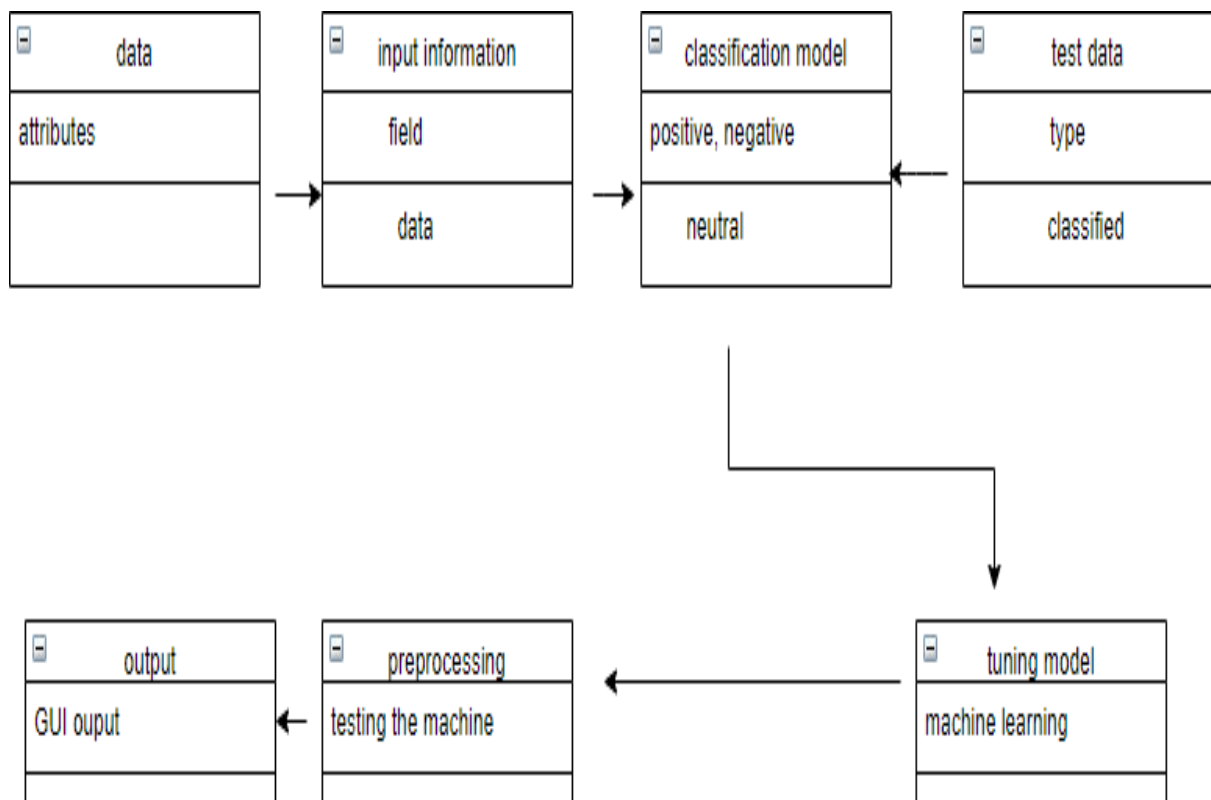


Fig:4.4 Class diagram

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified and because, unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable to the developer/coder. Finally, before making the final version, the diagram should be drawn on plain paper and

rework as many times as possible to make it correct

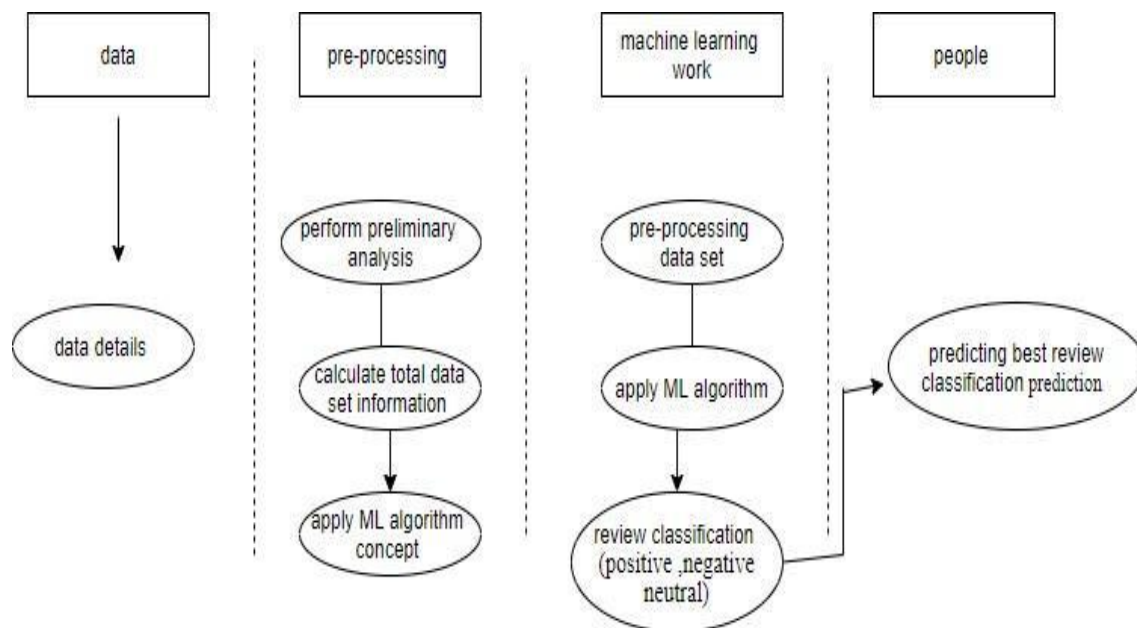


Fig:4.5 Activity diagram

4.7 Activity Diagram

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity

diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

4.8 Sequence Diagram:

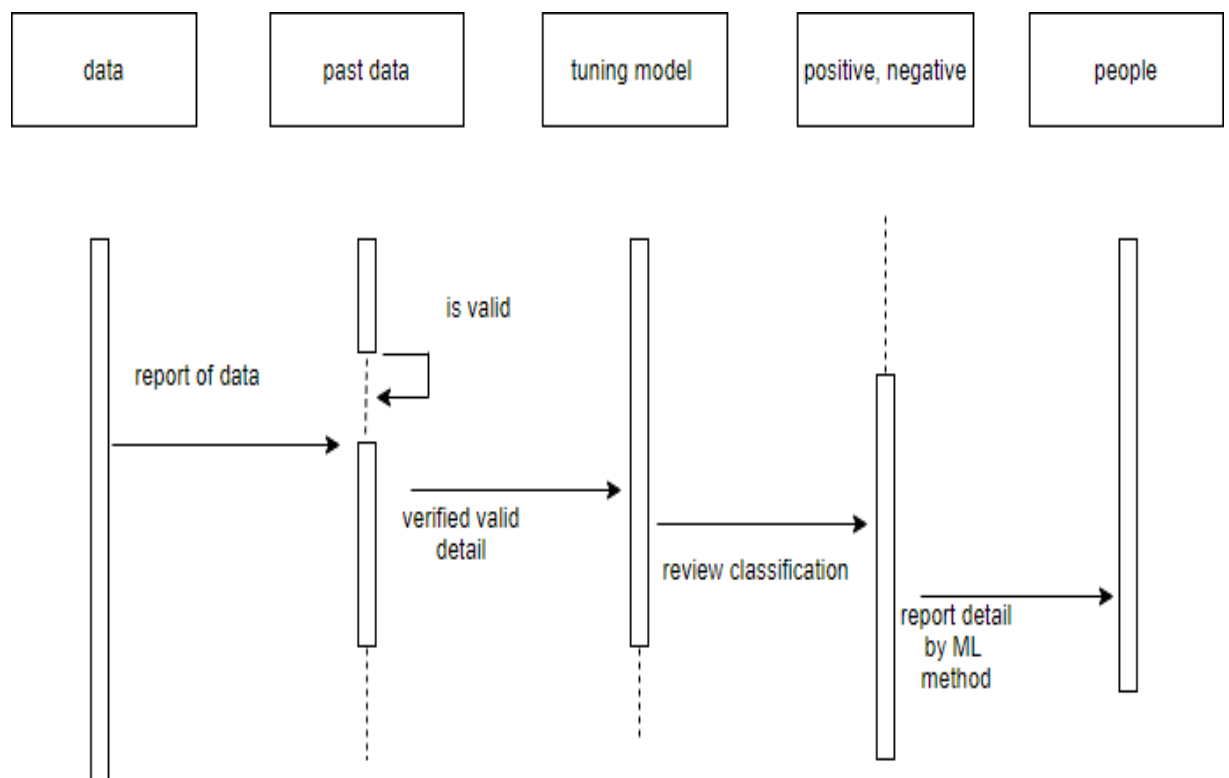


Fig:4.6 Sequence diagram

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system. Other dynamic modeling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview diagramming. Sequence diagrams, along with class diagrams and physical data models are in my opinion the most important design-level models for modern business application development.

4.9 Entity Relationship Diagram (ERD)

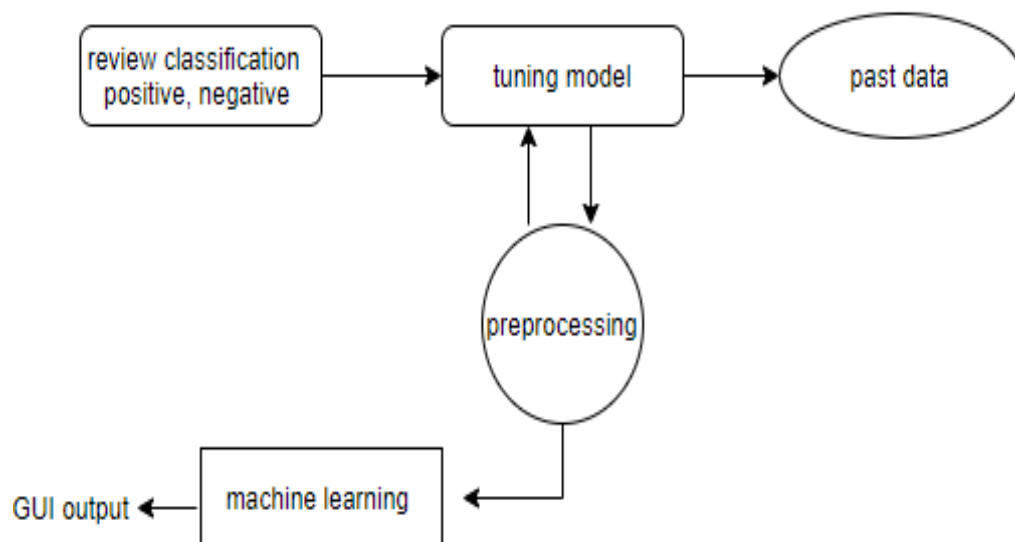


Fig:4.7 Entity Relationship diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

CHAPTER 5

ALGORITHMS AND TECHNIQUES USED

5.1 Algorithm Explanation

In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc. In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines which label should be given to new data based on pattern and associating the patterns to the unlabeled new data.

Used Python Packages:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, we are using some of its modules like `train_test_split`, `DecisionTreeClassifier` or `Logistic Regression` and `accuracy_score`.

NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.

- Data manipulation can be done easily with data frames.

Matplotlib:

- Data visualization is a useful way to help with identify the patterns from given dataset.
- Data manipulation can be done easily with data frames.

5.2 Logistic Regression

It is a statistical method for analysing a data set in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes). The goal of logistic regression is to find the best fitting model to describe the relationship between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a set of independent (predictor or explanatory) variables. Logistic regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).

In other words, the logistic regression model predicts $P(Y=1)$ as a function of X . Logistic regression Assumptions:

- ☐ Binary logistic regression requires the dependent variable to be binary.
- ☐ For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
- ☐ Only the meaningful variables should be included.

- The independent variables should be independent of each other. That is, the model should have little.
- The independent variables are linearly related to the log odds.
- Logistic regression requires quite large sample sizes.

Decision Tree

It is one of the most powerful and popular algorithm. Decision- tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables. Assumptions of Decision tree:

- At the beginning, we consider the whole training set as the root.
- Attributes are assumed to be categorical for information gain, attributes are assumed to be continuous.
- On the basis of attribute values records are distributed recursively.
- We use statistical methods for ordering attributes as root or internal node.

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a data set into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. A decision node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data. Decision tree builds classification or regression models in the form of a tree structure. It utilizes an if- then rule set which is mutually exclusive and exhaustive for

classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed.

This process is continued on the training set until meeting a termination condition. It is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact towards in the classification and they are identified using the information gain concept. A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers.

5.3 K-Nearest Neighbor (KNN)

K-Nearest Neighbor is a supervised machine learning algorithm which stores all instances correspond to training data points in n-dimensional space. When an unknown discrete data is received, it analyzes the closest k number of instances saved (nearest neighbors) and returns the most common class as the prediction and for real-valued data it returns the mean of k nearest neighbors. In the distance-weighted nearest neighbor algorithm, it weights the contribution of each of the k neighbors according to their distance using the following query giving greater weight to the closest neighbors.

Usually KNN is robust to noisy data since it is averaging the k-nearest neighbors. The k-nearest-neighbors algorithm is a classification algorithm, and it is supervised: it takes a bunch of labeled points and uses them to learn how to label other points. To label a new point, it looks at the labeled points closest to that new point (those are its nearest neighbors), and has those neighbors vote, so whichever label the most of the neighbors have is the label

for the new point (the “k” is the number of neighbors it checks). Makes predictions about the validation set using the entire training set. KNN makes a prediction about a new instance by searching through the entire set to find the k “closest” instances. “Closeness” is determined using a proximity measurement (Euclidean) across all features.

5.4 Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees’ habit of over fitting to their training set. Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision *trees*, resulting in *a forest of trees*, hence the name "Random Forest". The random forest algorithm can be used for both regression and classification tasks.

The following are the basic steps involved in performing the random forest algorithm:

- ☐ Pick N random records from the dataset.
- ☐ Build a decision tree based on these N records.
- ☐ Choose the number of trees you want in your algorithm and repeat steps 1 and 2.

- In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

5.5 Support Vector Machines

A classifier that categorizes the data set by setting an optimal hyper plane between data. I chose this classifier as it is incredibly versatile in the number of different kernelling functions that can be applied and this model can yield a high predictability rate. Support Vector Machines are perhaps one of the most popular and talked about machine learning algorithms. They were extremely popular around the time they were developed in the 1990s and continue to be the go-to method for a high-performing algorithm with little tuning.

- How to disentangle the many names used to refer to support vector machines.
- The representation used by SVM when the model is actually stored on disk.
- How a learned SVM model representation can be used to make predictions for new data.
- How to learn an SVM model from training data.
- How to best prepare your data for the SVM algorithm.

Where you might look to get more information on SVM.

5.6 Deep Learning RNN with LSTM get best accuracy result.

Recurrent Neural Network is a generalization of feedforward neural network that has an internal memory. RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input. Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. In other neural networks, all the inputs are independent of each other. But in RNN, all the inputs are related to each other. First, it takes the $X(0)$ from the sequence of input and then it outputs $h(0)$ which together with $X(1)$ is the input for the next step. So, the $h(0)$ and $X(1)$ is the input for the next step. Similarly, $h(1)$ from the next is the input with $X(2)$ for the next step and so on. This way, it keeps remembering the context while training.

Advantages of Recurrent Neural Network:

RNN can model sequence of data so that each sample can be assumed to be dependent on previous ones Recurrent neural

network are even used with convolutional layers to extend the effective pixel neighborhood.

5.3 Output for prediction of sentiment by giving input sentences.

code2vec is a neural model that learns analogies relevant to source code. The model was trained on the Java code database but you can apply it to any codebase. Then there's GloVe, the GloVe vocabularies from the website. We took the largest one because then there's a higher chance of it finding all of our words. You can choose where you want to download it but, for convenience, it's better to store it in the working directory, Now we can find out-of-vocabulary words and count the percentage of these words for code2vec vocabulary. The following code will also work for GloVe. We've tried three different word embeddings algorithms for OpenAPI specification. Despite the fact that all three perform quite well on this dataset, an extra comparison of the most similar words shows.

CHAPTER 6

RESULTS AND DISCUSSIONS

6.1 RESULT:

we have developed a system to rate applications in terms of feature based on the downloader's reviews and review votes. we could rate 10,000 apps. This makes downloading decisions and improve the application. We accomplish this by first converting the unstructured data into structured data; then, we extract the sentences comprising our feature keywords; then, we were able to provide the feature-level ratings through sentiment analysis of these sentences. We rank the phones based on the number of features that they are best at, and accordingly, we were able to recommend the best phones for a feature. We tested our methodology on the “phone” named feature by considering the overall customer ratings as ground-truth ratings. The performance of our method is found to be decent. We obtain MAE of only 0.555, i.e., approximately just half a star. We get 52.3% accuracy if exact integer ratings have to be predicted. However, if we can tolerate the one-star integer rating error, the accuracy jumps to 93.8%. The proposed approach is unsupervised. As an extension, we will work on improving the performance by taking a weakly supervised or supervised approach to this problem, for which we will have to annotate the available data in terms of all our features.

6.2 CONCLUSION

In this digital world, every aspect of our software life has moved from computers to mobile phones. In other words, App stores (i.e., Apple AppStore and Google Play Store) provide various mobile applications for every field of our life. SVM models are easier to understand. There are different kernels that provide a different level of flexibilities beyond the classical linear kernel, such as the Radial Basis Function kernel (RBF). Unlike the linear kernel, the RBF can handle the case when the relation between class labels and attributes is nonlinear. It is the extended version of linear classifier and it usually gives.

1. It is more accurate than its competitors.
2. It is more robust i.e. due to optimal margin gap between separating hyper planes, it could do predictions better with test data.
3. It is computationally more efficient. This is because of using Kernel trick in dual problem. Thus it is faster in training, better in accuracy with stability/robustness.

SVM is a method with better performance for many applications but not for all. SVM is also a best classifier if there is a two class problem with balanced data sets and free of noise or with little bit of noise.

Today's world Google play store has become the major hub for downloading and uploading Android application. The Android application users download these applications for their personal use. Each user of the application has their own experience with the application. Users download and use these applications and express the experience of the application in the shape of comments or reviews, also give a rating to this application on the scale of 0-5. In this research work, we have scrapped 251661 user reviews through different data scraping techniques in an

evaluated result of this research, we have applied text classification on our dataset after preprocessing. In our research to find the accuracy, recall, precision, F1 score we are using different machine learning algorithms. After a comparison of the different attributes, we have also found the best algorithm which has the best accuracy. In the future, increase the category of applications and increase the number of reviews. Compare the logistic regression algorithm accuracy results with other different algorithms. Generate clusters and check the relationship between reviews and ratings of the application that can analyze each application more precisely

6.3 FUTURE WORK:

Hundreds and thousands of apps uploaded by developers and downloaded by users are on the Google Play Store. Users use these applications for their specific purpose, and they have their personal experiences. Users download and use these applications and express the application's experience in the form of comments or reviews and give the applications a 0-5 scale rating. We have scraped 506259 reviews for 14 different categories of Google Play Store applications in this research work. We have analyzed the class of the reviews that may be positive, negative, and neutral. We have checked the application semantics with different algorithms of machine learning. We have used three different machine learning algorithms, such as Logistic Regression Algorithm, Random Forest, and Multinomial Naïve Bayes. Evaluate Bigram, Trigram, and N-gram with various parameters such as precision, accuracy, recall, and F1 score, and compared the statistical results of these algorithms. After contrast, we have evaluated that the Logistic regression algorithm is the most active algorithm with a high precision score,

and we can use this machine-learning algorithm to test the user reviews. In the future we will increase the number of categories of applications and number of reviews. We will compare the accuracy of the logistic regression algorithm with other different algorithms. We will generate the clusters and check the relationship between application reviews and ratings that can help to more accurately analyze each application. To automate this process by show the prediction result in web application or desktop application. To optimize the work so that we can implement it in artificial interlligence.

REFERENCES

- [1] Y. Goldberg, "Neural network methods for natural language processing," *Synthesis Lectures on Human Language Technologies*, vol. 10, no. 1, pp. 1-309, 2017.
- [2] N. Genc-Nayebi and A. Abran, "A systematic literature review: Opinion mining studies from mobile app store user reviews," *Journal of Systems and Software*, vol. 125, pp. 207-219, 2017.
- [3] E. Cambria, B. Schuller, Y. Xia, and B. White, "New avenues in knowledge bases for natural language processing," *Knowledge-Based Systems*, vol. 108, no. C, pp. 1-4, 2016.
- [4] R. Agerri, X. Artola, Z. Beloki, G. Rigau, and A. Soroa, "Big data for Natural Language Processing: A streaming approach," *Knowledge-Based Systems*, vol. 79, pp. 36-42, 2015.
- [5] Y. Man, C. Gao, M. R. Lyu, and J. Jiang, "Experience report: Understanding cross-platform app issues from user reviews," in *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE)*, 2016, pp. 138-149: IEEE.
- [6] T. Denoeux, "Logistic regression, neural networks and dempster-shafer theory: a new perspective," *Knowledge-Based Systems*, 2019.
- [7] C. Gao, Y. Zhao, R. Wu, Q. Yang, and J. Shao, "Semantic trajectory compression via multi-resolution synchronization-based

clustering," Knowledge-Based Systems, 2019.

[8] K. Santo, S. S. Richtering, J. Chalmers, A. Thiagalingam, C. K. Chow, and J. Redfern, "Mobile phone apps to improve medication adherence: a systematic stepwise process to identify high-quality apps," JMIR mHealth and uHealth, vol. 4, no. 4, p. e132, 2016.

[9] P. Barlas, I. Lanning, and C. Heavey, "A survey of open source data science tools," International Journal of Intelligent Computing and Cybernetics, vol. 8, no. 3, pp. 232-261, 2015.

[10] R. Zeng and P. M. Greenfield, "Cultural evolution over the last 40 years in China: Using the Google Ngram Viewer to study implications of social and political change for cultural values," International Journal of Psychology, vol. 50, no. 1, pp. 47-55, 2015.

[11] C. Li, Y. Liu, and L. Zhao, "Using external resources and joint learning for bigram weighting in ilp-based multi-document summarization," in Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 778-787.

[12] A. Pagán, H. I. Blythe, and S. P. Liversedge, "Parafoveal preprocessing of word initial trigrams during reading in adults and

children," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 42, no. 3, p. 411, 2016.

[13] C. Mellish and X. Sun, "The semantic web as a linguistic resource: Opportunities for natural language generation," in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, 2005, pp. 77-87: Springer.

[14] J. Schütte, R. Fedler, and D. Titze, "Condroid: Targeted dynamic analysis of android applications," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, 2015, pp. 571-578: IEEE.

[15] B. Yu, Z.-b. Xu, and C.-h. Li, "Latent semantic analysis for text categorization using neural network," *Knowledge-Based Systems*, vol. 21, no. 8, pp. 900-904, 2008.

[16] R. Liu and X. Zhang, "Generating machine-executable plans from end-user's natural-language instructions," *Knowledge-Based Systems*, vol. 140, pp. 15-26, 2018.

[17] M. Gomez, "Towards Improving the Quality of Mobile Apps by Leveraging Crowdsourced Feedback," *Universite Lille 1; Inria Lille-Nord Europe*, 2016.

[18] M. Gómez, R. Rouvoy, M. Monperrus, and L. Seinturier, "A recommender for moderators," in *Proceedings of the Second ACM International Conference on Mobile Software Engineering and Systems*, 2015,

pp. 1-11: IEEE Press.

[19] G. V. Georgiev and D. D. Georgiev, "Enhancing user creativity: Semantic measures for idea generation," *Knowledge-Based Systems*, vol. 151, pp. 1-15, 2018.

[20] I. J. M. Ruiz, M. Nagappan, B. Adams, T. Berger, S. Dienst, and A. E. Hassan, "Examining the rating system used in mobile-app stores," *IEEE Software*, vol. 33, no. 6, pp. 86-92, 2015.

[21] T. Xu, Q. Peng, and Y. Cheng, "Identifying the semantic orientation of terms using S-HAL for sentiment analysis," *Knowledge-Based Systems*, vol. 35, pp. 279-289, 2012.

[22] N. Milosevic, A. Dehghantanha, and K.-K. R. Choo, "Machine learning aided Android malware classification," *Computers & Electrical Engineering*, vol. 61, pp. 266-274, 2017.

[23] J. Atkinson, A. Ferreira, and E. Aravena, "Discovering implicit intention- level knowledge from natural-language texts," in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, 2008, pp. 249-262: Springer.

SAMPLE CODE:

```
import warnings

warnings.filterwarnings("ignore"
) import tqdm

import numpy as np

import keras_metrics # for recall and precision metrics

from keras.preprocessing.sequence import
pad_sequences from keras.preprocessing.text import
Tokenizer

from keras.layers import Embedding, LSTM, Dropout, Dense

from keras.models import Sequential

from keras.utils import to_categorical

from keras.callbacks import ModelCheckpoint, TensorBoard

from sklearn.model_selection import train_test_split

import time

import numpy as
np import pickle

SEQUENCE_LENGTH = 100 # the length of all sequences (number of words per
sample)

EMBEDDING_SIZE = 100 # Using 100-Dimensional GloVe embedding vectors

TEST_SIZE = 0.25 # ratio of testing set
```

```
BATCH_SIZE = 64
```

```
EPOCHS = 30 # number of epochs
```

```
# to convert labels to integers and vice-versa
```

```
label2int =
```

```
{ "Positive":0, 'Neutral':1, "Negative":2 }
```

```
int2label =
```

```
{0:"Positive",1:"Neutral",2:'Negative'} import
```

```
pandas as pd
```

```
data=pd.read_csv("googleplaystore_user_reviews.csv")
```

```
data=data.dropna()
```

```
data['Sentiment'].unique()
```

```
x=data['Translated_Review']
```

```
y=data['Sentiment']
```

```
#To describe the
```

```
dataframe df.describe()
```

```
#Checking datatype and information about
```

```
dataset df.info()
```

```
#Checking for duplicate
```

```
data df.duplicated()
```

```
#plotting graph for
```

```
distribution import
```

```
matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```

sns.countplot(x = "Sentiment", data =
df) df.loc[:,
'Sentiment'].value_counts()

plt.title('Distribution of Sentiments')

from sklearn.preprocessing import LabelEncoder

var_mod = ['App', 'Translated_Review', 'Sentiment', 'Sentiment_Polarity',
'Sentiment_Subjectivity']

le =
LabelEncoder()

for i in var_mod:

df[i] =

le.fit_transform(df[i]).astype(str) #

Text tokenization

# vectorizing text, turning each text into sequence of integers

tokenizer = Tokenizer()

tokenizer.fit_on_texts(x)

# convert to sequence of integers

x =

tokenizer.texts_to_sequences(x) #

convert to numpy arrays

x =

np.array(x)

y =

```

np.array(y)

pad sequences at the beginning of each sequence with

0's # for example if SEQUENCE_LENGTH=4:

[[5, 3, 2], [5, 1, 2, 3], [3, 4]]

```

# will be transformed to:

# [[0, 5, 3, 2], [5, 1, 2, 3], [0, 0, 3, 4]]

x = pad_sequences(x,

maxlen=SEQUENCE_LENGTH) # One Hot

encoding labels

# [spam, ham, spam, ham, ham] will be converted to:

# [1, 0, 1, 0, 1] and then to:

# [[0, 1], [1, 0], [0, 1], [1, 0], [0, 1]]


y = [ label2int[label] for label in

y ] y = to_categorical(y)

# split and shuffle

X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=TEST_SIZE,

random_state=7)

def get_embedding_vectors(tokenizer, dim=100):

    embedding_index = {}

    with open(f"glove.6B.{dim}d.txt", encoding='utf8') as

        f: for line in tqdm.tqdm(f, "Reading GloVe"):

            values =

            line.split() word

            = values[0]

            vectors = np.asarray(values[1:], dtype='float32')

            embedding_index[word] = vectors

```

```

word_index = tokenizer.word_index

embedding_matrix = np.zeros((len(word_index)+1,
dim)) for word, i in word_index.items():

    embedding_vector =

    embedding_index.get(word) if

    embedding_vector is not None:

        # words not found will be 0s

        embedding_matrix[i] = embedding_vector


return embedding_matrix

def get_model(tokenizer,

    lstm_units): """

    Constructs the model,

    Embedding vectors => LSTM => 2 output Fully-Connected neurons with softmax
activation

    """

    # get the GloVe embedding vectors

    embedding_matrix =

    get_embedding_vectors(tokenizer) model =

    Sequential()

    model.add(Embedding(len(tokenizer.word_index)+1,

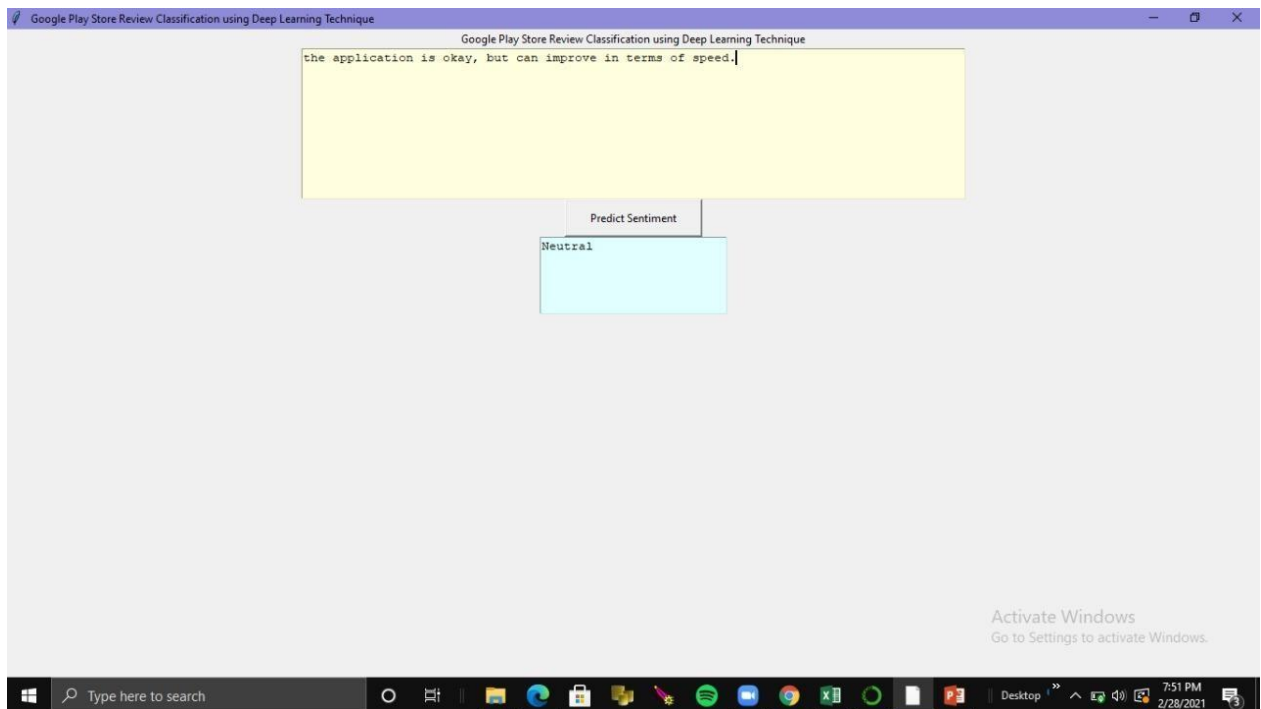
        EMBEDDING_SIZE,

```

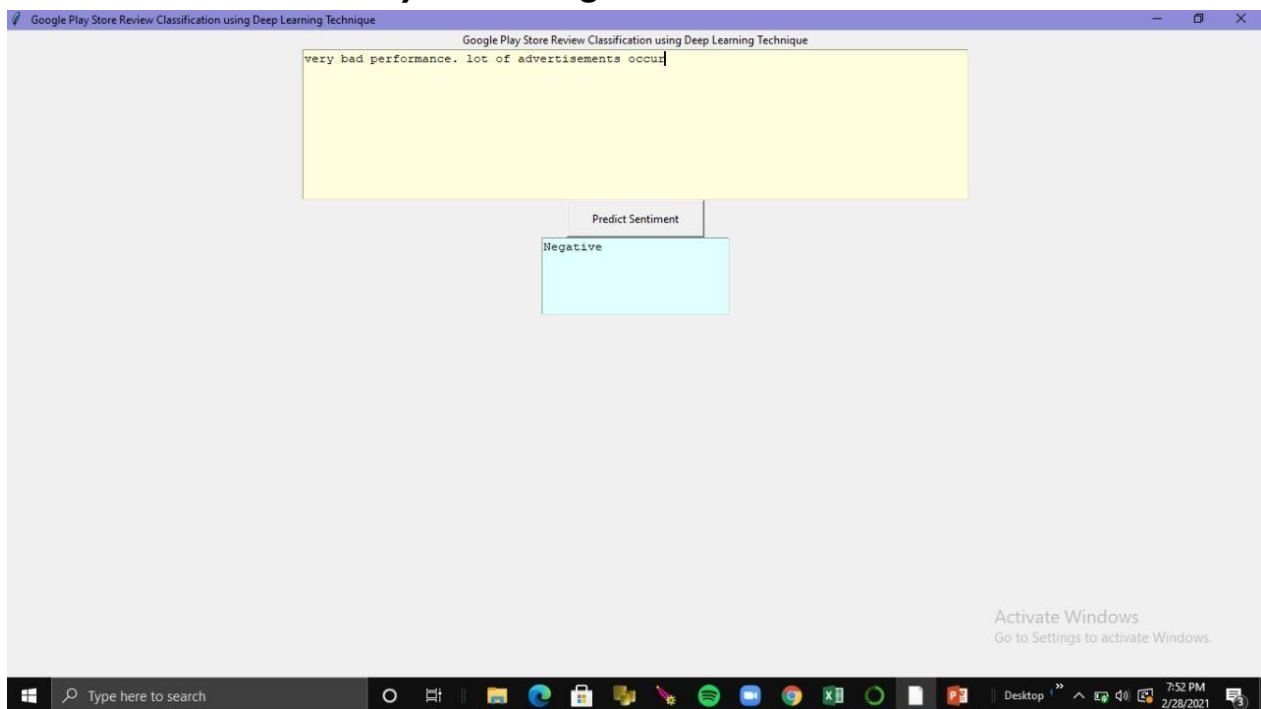
[illegible]

```
tensorboard =  
  
TensorBoard(f"opinion_spam_classifier_{time.time()}") # print our  
  
data shapes  
  
print("X_train.shape:",  
  
X_train.shape)  
  
print("X_test.shape:",  
  
X_test.shape)  
  
print("y_train.shape:",  
  
y_train.shape)  
  
print("y_test.shape:", y_test.shape)  
  
# train the model  
  
model.fit(X_train, y_train, validation_data=(X_test, y_test),  
  
          batch_size=BATCH_SIZE, epochs=EPOCHS,verbose=1)
```

SCREENSHOTS :



a) detecting neutral review



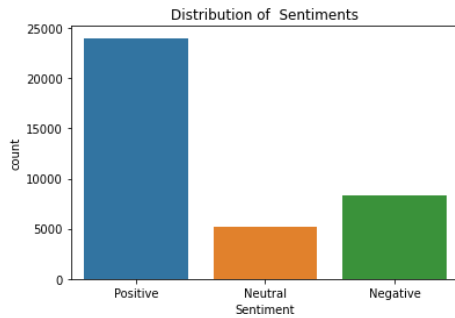
c) detecting negative review

e) Visualize Sentiment types on Bar plot

dtype= object)

```
In [20]: #plotting graph for distribution
import matplotlib.pyplot as plt
import seaborn as sns
sns.countplot(x = "Sentiment", data = df)
df.loc[:, 'Sentiment'].value_counts()
plt.title('Distribution of Sentiments')
```

Out[20]: Text(0.5, 1.0, 'Distribution of Sentiments')



f) Knn Accuracy

Accuracy result of K-Nearest Neighbor is: 98.95805503606732

Classification report of K-Nearest Neighbor Results:

	precision	recall	f1-score	support
0	0.99	0.97	0.98	2481
1	0.99	1.00	0.99	8748
accuracy			0.99	11229
macro avg	0.99	0.98	0.98	11229
weighted avg	0.99	0.99	0.99	11229

Confusion Matrix result of K-Nearest Neighbor is:

```
[[2396  85]
 [ 32 8716]]
```

Sensitivity : 0.9657396211205159

Specificity : 0.9963420210333791

g) Nave bayes Accuracy

Accuracy result of Naive bayes is: 97.80924392198771

Classification report of Naive bayes: Results:

	precision	recall	f1-score	support
0	1.00	0.90	0.95	2481
1	0.97	1.00	0.99	8748
accuracy			0.98	11229
macro avg	0.99	0.95	0.97	11229
weighted avg	0.98	0.98	0.98	11229

Confusion Matrix result of Naive bayes: is:

```
[[2235 246]
 [ 0 8748]]
```

Sensitivity : 0.9008464328899637

Specificity : 1.0

h) SVM Accuracy

Accuracy result of Support Vector Machines is: 98.76213376079794

Classification report of Support Vector Machines: Results:

	precision	recall	f1-score	support
0	1.00	0.94	0.97	2481
1	0.98	1.00	0.99	8748
accuracy			0.99	11229
macro avg	0.99	0.97	0.98	11229
weighted avg	0.99	0.99	0.99	11229

Confusion Matrix result of Support Vector Machines: is:

```
[[2342 139]
 [ 0 8748]]
```

Sensitivity : 0.9439742039500202

Specificity : 1.0

i) Comparing algorithms

[20]: graph()



j) Giving input sentence getting sentiment output:

```
In [66]: text=str(input("enter the statement: "))
```

enter the statement: This help eating healthy exercise regular basis

```
In [67]: #text = "We stayed for a one night getaway with family on a thursday. Triple AAA "  
print(get_predictions(text))
```

Positive

```
In [68]: text = "Language barrier. I understand Korea. I want English."  
print(get_predictions(text))
```

Neutral