EQUITY MARKET SENTIMENT ANALYSIS USING NLP TECHNIQUES

A PROJECT REPORT

Submitted by

ARUNA S (211420205021)

HARINI T (211420205057)

LAVANYA K (211420205080)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

INFORMATION TECHNOLOGY

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MARCH 2024

EQUITY MARKET SENTIMENT ANALYSIS USING NLP TECHNIQUES

A PROJECT REPORT

Submitted by

ARUNA S (211420205021)

HARINI T (211420205057)

LAVANYA K (211420205080)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

INFORMATION TECHNOLOGY

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MARCH 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonmous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"EQUITY MARKET SENTIMENT ANALYSIS USING NLP TECHNIQUES"** is the bonafide work of "**ARUNA S (211420205021), HARINI T (211420205057), LAVANYA K (211420205080)"** who carried out the project work under my supervision.

SIGNATURE

Dr. M. HELDA MERCY M.E., Ph.D.,

HEAD OF THE DEPARTMENT

Department of Information Technology

Panimalar Engineering College

Poonamallee, Chennai - 600 123

SIGNATURE

Mrs. A. REKHA M.E., HEAD

ASSISTANT PROFESSOR

SUPERVISOR

Department of Information Technology

Panimalar Engineering College

Poonamallee, Chennai - 600 123

Submitted for the project and viva-voce examination held on _____

SIGNATURE

INTERNAL EXAMINER

SIGNATURE

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project report entitled "**EQUITY MARKET SENTIMENT ANALYSIS USING NLP**" which is being submitted in partial fulfilment of the requirement of the course leading to the award of the 'Bachelor Of Technology in Information Technology ' in **Panimalar Engineering College, Autonomous institution Affiliated to Anna university- Chennai** is the result of the project carried out by us under the guidance of **Mrs. A. Rekha M.E., Assistant Professor in the Department of Information Technology**. We further declared that I or any other person has not previously submitted this project report to any other institution/university for any other degree/ diploma or any other person.

(ARUNA S)

(HARINI T)

(LAVANYA K)

Date**:**

Place**:** Chennai

It is certified that this project has been prepared and submitted under my guidance.

Date: **Mrs. A. REKHA M.E.,**

Place: Chennai (ASSISTANT PROFESSOR / IT)

ACKNOWLEDGEMENT

A project of this magnitude and nature requires kind co-operation and support from many, for successful completion . We wish to express our sincere thanks to all those who were involved in the completion of this project.

Our sincere thanks to **Our Beloved Secretary and Correspondent, Dr. P. CHINNADURAI, M.A., Ph.D.,** for his sincere endeavor in educating us in his premier institution. We would like to express our deep gratitude to **Our Dynamic Directors, Mrs. C. VIJAYA RAJESHWARI and Dr. C. SAKTHI KUMAR, M.E., M.B.A., Ph.D.,** and **Dr. SARANYA SREE SAKTHIKUMAR.,B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities for completion of this project.

We also express our appreciation and gratefulness to **Our Principal Dr. K. MANI, M.E., Ph.D.,** who helped us in the completion of the project. We wish to convey our thanks and gratitude to our head of the department, **Dr. M. HELDA MERCY, M.E., Ph.D.,** Department of Information Technology, for her support and by providing us ample time to complete our project.

We express our indebtedness and gratitude to our Project co-ordinator **Mr. M. DILLI BABU**, **M.E.,(Ph.D.,)** Associate Professor, Department of Information Technology for his guidance throughout the course of our project. We also express sincere thanks to our supervisor **Mrs. A. REKHA, M.E.**, Assistant Professor, Department of Information Technology for providing the support to carry out the project successfully. Last, we thank our parents and friends for providing their extensive moral support and encouragement during the course of the project.

ABSTRACT

Equity market sentiment analysis plays a crucial role in financial decision-making, as investor sentiment often affects market trends and stock prices. With the rapid growth of social media and online financial forums, there is an abundance of textual data available that can provide valuable insights into marketsentiment. This abstract presents a study on utilizing Natural Language Processing (NLP) techniques for equity market sentiment analysis. The overview of this research is to develop a robust framework that leverages NLP techniques to analyze and predict equity market sentiment based on textual data from various sources, including financial news articles, social media posts, and investor forums. The proposed framework consists of several stages, including data collection, pre-processing, feature extraction, sentiment analysis, and prediction.

Keywords: Equity market, sentiment analysis, Natural Language Processing (NLP), data pre-processing, feature extraction, machine learning, deep learning, sentiment prediction, market trends, stock prices.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

Equity market sentiment analysis is pivotal in guiding financial decision-making, given its influence on market trends and stock prices. With the exponential expansion of social media platforms and online financial forums, an extensive pool of textual data has become available, offering valuable insights into market sentiment. This paper explores the methodologies and techniques employed in harnessing this textual data for sentiment analysis purposes. Firstly, the paper delves into the significance of sentiment analysis in financial markets, elucidating how investor sentiment can sway market dynamics and influence trading decisions. Equity market sentiment analysis involves the examination of investors' attitudes, emotions, and opinions towards the market and specific securities. It's crucial because investor sentiment can greatly influence market trends, stock prices, and trading volumes. Understanding sentiment can help investors and financial professionals make more informed decisions about buying, selling, or holding assets.

1.2 NEED FOR THE PROJECT

With the rapid growth of social media and online financial forums, there is an abundance of textual data available. Traditional methods of market analysis often overlook the insights embedded in unstructured textual data. Analyzing sentiment from various sources can provide valuable insights into investor perceptions, which in turn affect market trends and stock prices.

Enhanced Decision-making Processes: By leveraging Natural Language Processing (NLP) techniques, this project aims to extract meaningful patterns and sentiments from textual data, thus enhancing decision-making processes in financial markets. By incorporating sentiment analysis into market analysis frameworks, investors and financial institutions can make more informed decisions, anticipate market movements, and manage risks more effectively. By leveraging NLP techniques, the project seeks to extract meaningful patterns and

sentiments from this textual data, thus enhancing decision-making processes in financial markets. the need for this project lies in harnessing the vast amount of unstructured textual data available online, uncovering valuable insights from investor sentiments, and leveraging NLP techniques to enhance decision-making processes in financial markets. By bridging the gap between traditional market analysis methods and the digital age of information, this project aims to empower market participants with actionable insights and a competitive advantage in navigating dynamic market conditions**.**

1.3 OBJECTIVE OF THE PROJECT

The primary objective is to develop a comprehensive framework that effectively integrates Natural Language Processing (NLP) techniques with traditional market analysis methodologies. This framework should be capable of handling large volumes of unstructured textual data sourced from social media, financial news articles, and online forums. The project aims to extract meaningful patterns and sentiments from the vast amount of textual data available online. This involves employing advanced NLP techniques such as sentiment analysis, topic modeling, and entity recognition to uncover valuable insights hidden within the unstructured data. One of the core objectives is to enhance decision-making processes in financial markets by leveraging insights derived from sentiment analysis. By understanding investor perceptions and sentiment trends, the framework should facilitate more informed investment decisions, thereby potentially improving portfolio performance and risk management strategies. Traditional methods of market analysis often focus solely on quantitative indicators such as price movements and financial ratios, overlooking the rich insights embedded in unstructured textual data. The project aims to address this limitation by incorporating NLP techniques to analyze sentiment from various online sources, thus providing a more holistic view of market dynamics**.**

1.4 SCOPE OF THE PROJECT

The project will involve gathering textual data from a variety of sources including financial news articles, social media platforms (like Twitter, StockTwits), and online financial forums (such as Reddit's investing communities, Seeking Alpha). These sources provide a rich pool of unstructured textual data that reflects investor sentiment and opinions about various equities and market trends. Leveraging NLP techniques, the project will focus on processing and analyzing the collected textual data. This includes techniques for data cleaning, preprocessing, tokenization, and parsing to make the data suitable for analysis. Additionally, sentiment analysis algorithms will be employed to extract sentiment polarity (positive, negative, neutral) from the text. The project will involve extracting relevant features from the preprocessed textual data to capture sentiment-related information. These features may include sentiment scores, word embeddings, sentiment lexicons, and other linguistic features that help in understanding the underlying sentiment expressed in the text. Machine learning and/or deep learning algorithms will be applied to the extracted features for sentiment analysis and prediction. Supervised learning techniques such as Support Vector Machines (SVM), Random Forests, or deep learning architectures like Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), or Transformer-based models like BERT may be explored to build robust sentiment analysis models.

1.5 PROBLEM STATEMENT

In the realm of equity market analysis, the influence of investor sentiment on market trends and stock prices cannot be overstated. However, with the advent of social media platforms, online financial forums, and the ever-expanding digital landscape, the sheer volume of textual data available poses a formidable challenge in deciphering sentiment accurately and efficiently. Traditional methods of market analysis often fall short in capturing the complexities of investor sentiment, necessitating the adoption of innovative approaches. Hence, the problem at hand is to develop a robust framework leveraging advanced Natural Language Processing (NLP) techniques to analyze and predict equity market sentiment accurately based on textual data sourced from diverse channels such as financial news articles, social media posts, and investor forums. The crux of the problem lies in effectively harnessing the wealth of textual data available and extracting meaningful insights that can guide investment decisions amidst the dynamic and often unpredictable nature of financial markets. In today's fast-paced financial markets, understanding and gauging investor sentiment is crucial for making informed decisions. However, the sheer volume and complexity of financial data make traditional analysis methods insufficient. The challenge lies in harnessing the vast amount of unstructured data, such as news articles, social media posts, and analyst reports, to accurately gauge market sentiment in real-time

CHAPTER 2

LITERATURE SURVEY

2.1 LITERATURE SURVEY

PAPER 1:

TITLE: Stock Trend Prediction Using News Sentiment Analysis

YEAR: 2016

AUTHOR: Kalyani Joshi1, Prof. Bharathi H.N.2, Prof. Jyothi Rao3

ABSTRACT:

Efficient Market Hypothesis is the popular theory about stock prediction. With its failure much research has been carried in the area of prediction of stocks. This project is about taking non quantifiable data such as financial news articles about a company and predicting its future stock trend with news sentiment classification. Assuming that news articles have impact on stock market, this is an attempt to study relationship between news and stock trend. To show this, we created three different classification models which depict polarity of news articles being positive or negative. Observations show that RF and SVM perform well in all types of testing. Naïve Bayes gives good result but not compared to the other two. Experiments are conducted to evaluate various aspects of the proposed model and encouraging results are obtained in all of the experiments. The accuracy of the prediction model is more than 80% and in comparison with news random labelling with 50% of accuracy; the model has increased the accuracy by 30%.

PAPER 2:

TITLE: Sentiment Analysis of Stocks Based on News Headlines Using NLP

YEAR: 2023

AUTHOR: Aastha Saxena, Arpit Jain, Prateek Sharma, Sparsh Singla and Amrita Ticku.

ABSTRACT:

In todays' world everyone starting from a child to an adult is studying stocks and is finding ways to earn more by studying the patterns of the market. Stock market is a compound interrelated system of various investors. It fluctuates frequently and hence is hard to predict what is yet to come. All the companies worldwide rely on these forecasts and speculations so that they can increase their profits. Everyday news on the economic front plays a vital role in defining the jump or drop in the prices of stocks. The market news helps the investor excessively in determining his bidding as it is immensely rich in information. In this study, we extract useful information from news headlines of a particular company to investigate the immediate impact of it on the company's stock growth. Having the text based dataset we use NLP and compare two approaches using two different algorithms which both collectively determine the sentiment of the news headline (whether it is positive/negative/neutral) in lieu with the company stock

PAPER 3:

TITLE: Stock Market Prediction Using Natural Language Processing -A Survey
YEAR: 2022
AUTHOR: Om mane and Saravanakumar Kandasamy
ABSTRACT:

The stock market is a network which provides a platform for almost all major economic transactions. While investing in the stock market is a good idea, investing in individual stocks may not be, especially for the casual investor. Smart stock-picking requires in-depth research and plenty of dedication. Predicting this stock value offers enormous arbitrage profit opportunities. This attractiveness of finding a solution has prompted researchers to find a way past problems like volatility, seasonality, and dependence on time. This paper surveys recent literature in the domain of natural language processing and machine learning techniques used to predict stock market movements. The main contributions of this paper include the sophisticated categorizations of many recent articles and the illustration of the recent trends of research in stock market prediction and its related areas.

PAPER 4:

TITLE: Stock Prediction using Twitter Sentiment Analysis
YEAR: 2011
AUTHOR: Anshul Mittal, Arpit Goel

ABSTRACT:

In this paper, we apply sentiment analysis and machine learning principles to find the correlation between ”public sentiment” and ”market sentiment”. We use twitter data to predict public mood and use the predicted mood and previous days' DJIA values to predict the stock market movements. In order to test our results, we propose a new cross validation method for financial data and obtain 75.56% accuracy using Self Organizing Fuzzy Neural Networks (SOFNN) on the Twitter feeds and DJIA values from the period June 2009 to December 2009. We also implement a naive protfolio management strategy based on our predicted values. Our work is based on Bollen et al's famous paper which predicted the same with 87% accuracy.

PAPER 5:

TITLE: Predicting Stock Market Behavior Using Data Mining Technique and News Sentiment Analysis

YEAR: 2020

AUTHOR: Ayman E. Khedr, S.E Salama, Nagwa Yaseen

ABSTRACT:

Stock market prediction has become an attractive investigation topic due to its important role in economy and beneficial offers. There is an imminent need to uncover the stock market future behavior in order to avoid investment risks. The large amount of data generated by the stock market is considered a treasure of knowledge for investors. This study aims at constructing an effective model to predict stock market future trends with small error ratio and improve the accuracy of prediction. This prediction model is based on sentiment analysis of financial news and historical stock market prices. This model provides better accuracy results than all previous studies by considering multiple types of news related to market and company with historical stock prices. A dataset containing stock prices from three companies is used. The first step is to analyze news sentiment to get the text polarity using naïve Bayes algorithm. This step achieved prediction accuracy results ranging from 72.73% to 86.21%. The second step combines news polarities and historical stock prices together to predict future stock prices. This improved the prediction accuracy up to 89.80%.

CHAPTER 3

PROJECT DESCRIPTION

3 PROJECT DESCRIPTION

3.1 EXISTING SYSTEM

Stock Movement Prediction (SMP) aims at predicting listed companies' stock future price trend, which is a challenging task due to the volatile nature of financial markets. Recent financial studies show that the momentum spill over effect plays a significant role in stock fluctuation. However, previous studies typically only learn the simple connection information among related companies, which inevitably fail to model complex relations of listed companies in real financial market. To address this issue, we first construct a more comprehensive Market Knowledge Graph (MKG) which contains bi-typed entities including listed companies and their associated executives, and hybrid-relations including the explicit relations and implicit relations. Afterward, we propose DANSMP, a novel Dual Attention Networks to learn the momentum spill over signals based upon the constructed MKG for stock prediction.

DISADVANTAGES

- They did not implement the deployment process.
- They did not compared more than an algorithms to getting better accuracy level.
- Accuracy was low.

3.2 PROPOSED SYSTEM

The first step in the proposed system is to collect relevant textual data from various sources such as financial news articles, social media platforms, and investor forums. Sentiment Analysis: In this stage, machine learning and deep learning models are applied to classify the sentiment of the extracted features. Several algorithms can be employed, including Support Vector Machines (SVM), Naive Bayes, Recurrent Neural Networks (RNN), or Transformers like BERT or GPT. These models are trained on labeled sentiment datasets to learn patterns and associations between the textual features and sentiment categories (positive, negative, neutral). Sentiment Prediction and Trend Analysis: The proposed system incorporates time-series analysis techniques to predict sentiment and analyse its impact on market trends and stock prices. Methods such as autoregressive integrated moving average (ARIMA) or LSTM models can be used to forecast sentiment values. By examining the relationship between sentiment and market indicators, patterns and correlations can be identified, enabling users to make informed investment decisions.

MERITS:

- We compared more than a two algorithms to getting better accuracy level.
- We figure out performance and confusion metrics value properly.
- Build an application for deployment purpose.
- Accuracy & performance level improved.

3.3 MODULES

3.3.1 DATA PRE-PROCESSING

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. It as machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of

addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model A number of different data cleaning tasks using Python's Pandas library and specifically, it focus on probably the biggest data cleaning task, missing values and it able to more quickly clean data. It wants to spend less time cleaning data, and more time exploring and modeling.

Some of these sources are just simple random mistakes. Other times, there can be a deeper reason why data is missing. It's important to understand these different types of missing data from a statistics point of view. The type of missing data will influence how to deal with filling in the missing values and to detect missing values, and do some basic imputation and detailed statistical approach for dealing with missing data. 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

MODULE DIAGRAM

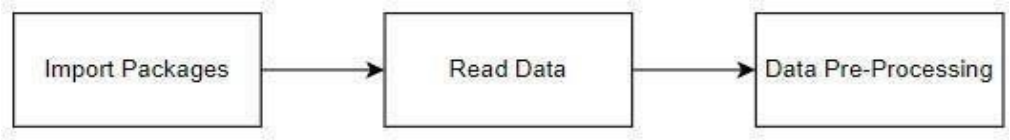

Fig 3.1 Data Pre-Processing

GIVEN INPUT EXPECTED OUTPUT

input : data

output : removing noisy data

3.3.2 DATA 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.

Sometimes data does not make sense until it can look at in a visual form, suchas with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in python and how to use them to better understand your own data

- How to chart time series data with line plots and categorical quantities with bar charts.
- How to summarize data distributions with histograms and box plots.

MODULE DIAGRAM

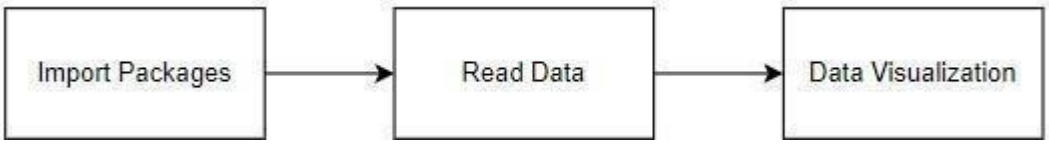

Fig 3.2 Data Visualization

GIVEN INPUT EXPECTED OUTPUT

input : data

output : visualized data

3.3.3 MULTINOMIALNB ALGORITHM

The Multinomial Naive Bayes (MultinomialNB) algorithm is a probabilistic classification algorithm commonly used in natural language processing tasks such as text classification. It is based on Bayes' theorem, assuming a multinomial distribution of features. In the context of document classification, each feature represents the frequency of a term in a document. The algorithm calculates the probability of a document belonging to a particular class by considering the frequency distribution of terms in that class. Despite its simplicity, MultinomialNB often performs well in text classification tasks, making it a popular choice for applications like spam filtering and sentiment analysis.

MODULE DIAGRAM

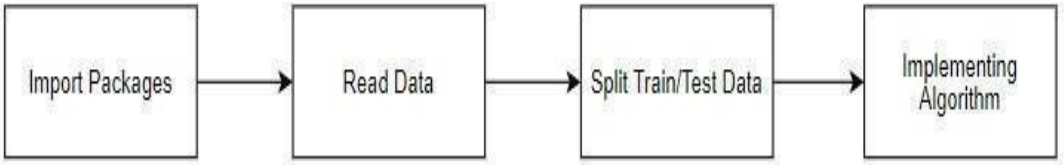

Fig 3.3 Workflow of MultinomialNB Algorithm

GIVEN INPUT EXPECTED OUTPUT

input : data

output : getting accuracy

3.3.4 LOGISTIC REGRESSION ALGORITHM

Logistic Regression is a statistical method used for binary classification tasks, predicting the probability of an event occurring based on input features. It employs the logistic function to model the relationship between the independent variables and the binary outcome. Widely used in machine learning, logistic regression is interpretable, computationally efficient, and serves as a foundational algorithm for understanding the impact of input features on the likelihood of a particular outcome.

MODULE DIAGRAM

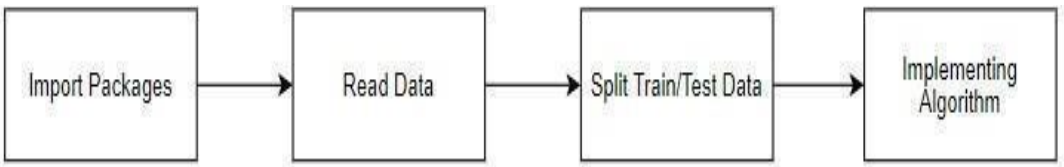

Fig 3.4 Workflow of Logistic Regression Algorithm

GIVEN INPUT EXPECTED OUTPUT

input : data

output : getting accuracy

3.3.5 RANDOM FOREST CLASSIFIER

A Random Forest Classifier is an ensemble machine learning algorithm that builds a multitude of decision trees during training and outputs the mode of the classes for classification tasks or the mean prediction for regression tasks. It excels in handling complex datasets by mitigating overfitting and improving generalization through the aggregation of diverse decision trees, making it a robust and widely-used method in various domains, including finance, healthcare, and image recognition.

MODULE DIAGRAM

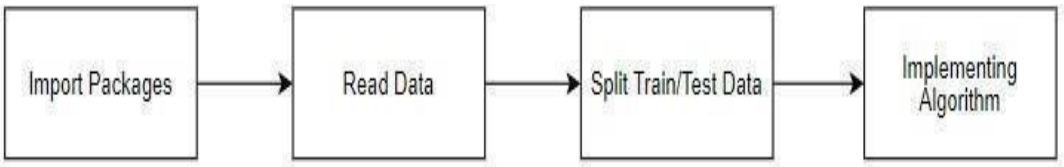

Fig 3.5 Workflow of Random Forest Classifier

GIVEN INPUT EXPECTED OUTPUT

input : data

output : getting accuracy

3.3.6 DEPLOYMENT USING DJANGO

In this module the trained deep learning model is converted into hierarchical data format file (.h5 file) which is then deployed in our django framework for providing better user interface and predicting the output whether the given image is CKD / Not CKD.

Django

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support.

MODULE DIAGRAM

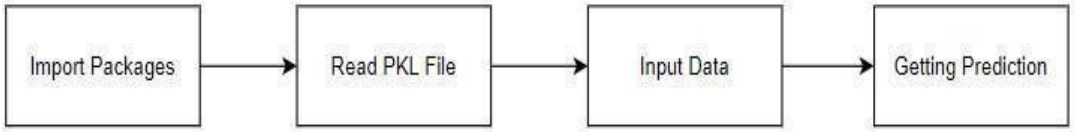

Fig 3.6 Deployment using Django and predicting the output

GIVEN INPUT EXPECTED OUTPUT

input : data values

output : predicting output

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 GENERAL

Requirements are the basic constrains 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

4.1.1 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.

4.1.2 NON FUNCTIONAL REQUIREMENTS

Process of functional steps,

1. Problem define
2. Preparing data
3. Evaluating algorithms
4. Improving results
5. Prediction the result

4.1.3 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 8 GB

4.2 SOFTWARE DESCRIPTION

Anaconda isa 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, 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.

4.2.1 ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository.

Anaconda. Now, if you are primarily doing data science work, Anaconda is also a great option. Anaconda is created by Continuum Analytics, and it is a Python distribution that comes preinstalled with lots of useful python libraries for data science.

Anaconda is a 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.

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages and use multiple environments to separate these different versions.

The command-line program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages, and update them – all inside Navigator.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook

- Spyder
- PyCharm
- VSCode
- Glueviz
- Orange 3 App
- RStudio
- Anaconda Prompt (Windows only)
- Anaconda PowerShell (Windows only)

4.2.2 JUPYTER NOTEBOOK

This website acts as “meta” documentation for the Jupyter ecosystem. It has a collection of resources to navigate the tools and communities in this ecosystem, and to help you get started.

Project Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages". It was spun off from IPython in 2014 by Fernando Perez.

Notebook documents 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.

Installation: The easiest way to install the *Jupyter Notebook App* is installing a scientific python distribution which also includes scientific python packages. The most common distribution is called Anaconda

Running the Jupyter Notebook

Launching *Jupyter Notebook App:* The Jupyter Notebook App can be launched by clicking on the *Jupyter Notebook* icon installed by Anaconda in the start menu (Windows) or by typing in a terminal (*cmd* on Windows): “jupyter notebook”

This will launch a new browser window (or a new tab) showing the Notebook Dashboard, a sort of control panel that allows (among other things) to select which notebook to open.

When started, the Jupyter Notebook App can access only files within its start-up folder (including any sub-folder). No configuration is necessary if you place your notebooks in your home folder or subfolders. Otherwise, you need to choose a Jupyter Notebook App start-up folder which will contain all the notebooks.

Save notebooks: Modifications to the notebooks are automatically saved every few minutes. To avoid modifying the original notebook, make a copy of the notebook document (menu file -> make a copy…) and save the modifications on the copy.

Executing a notebook: Download the notebook you want to execute and put it in your notebook folder (or a sub-folder of it).

- Launch the jupyter notebook app
- In the Notebook Dashboard navigate to find the notebook: clicking on its name will open it in a new browser tab.
- Click on the menu *Help -> User Interface Tour* for an overview of the Jupyter Notebook App user interface.
- You can run the notebook document step-by-step (one cell a time) by pressing *shift + enter*.
- You can run the whole notebook in a single step by clicking on the menu *Cell -> Run All*.
- To restart the kernel (i.e. the computational engine), click on the menu *Kernel -> Restart*. This can be useful to start over a computation from scratch (e.g. variables are deleted, open files are closed, etc…).

Purpose: To support interactive data science and scientific computing across all programming languages.

File Extension: An **IPYNB** file is a notebook document created by Jupyter Notebook, an interactive computational environment that helps scientists manipulate and analyze data using Python.

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

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.

CHAPTER 5

SYSTEM DESIGN

5.1 GENERAL

The system design for the equity market sentiment analysis project encompasses the architectural blueprint and the detailed specifications of each component, ensuring robustness, scalability, and maintainability. The design follows a modular approach, allowing for flexibility in integrating new features and adapting to evolving data sources and analysis techniques.

5.2 SYSTEM ARCHITECTURE

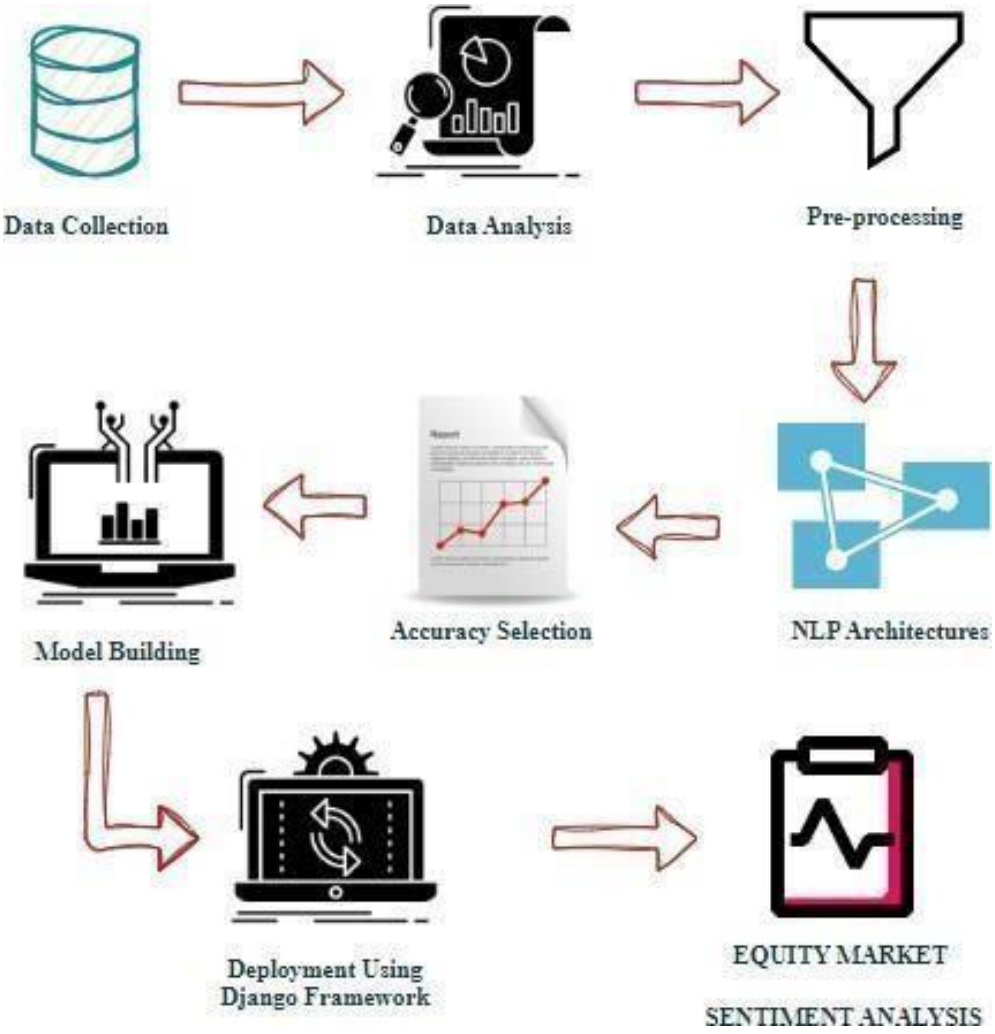

Fig 5.1 System Architecture

5.3 UML DIAGRAM

5.3.1 USE CASE DIAGRAM

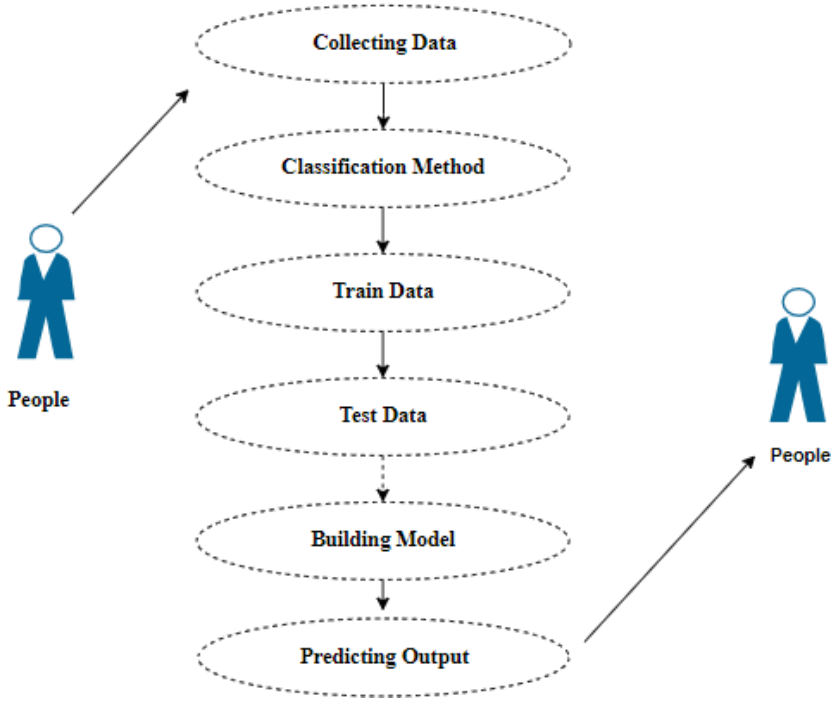

Fig 5.2 Use case diagram

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analysed the functionalities are captured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner. Overall, the use case diagram provides a high-level overview of the functionalities offered by the equity market sentiment analysis system and the interactions between users and the system to perform the actions effectively.

5.3.2 CLASS DIAGRAM

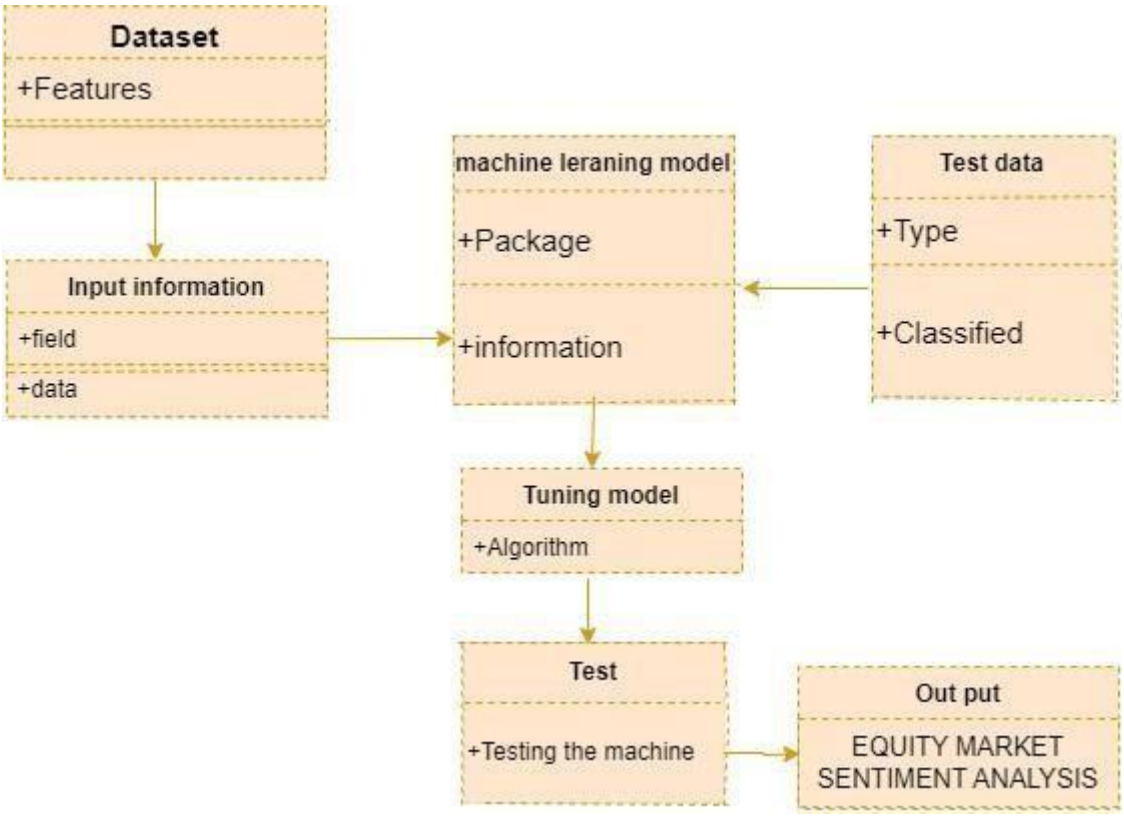

Fig 5.3 Class diagram

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.

5.3.3 ACTIVITY DIAGRAM

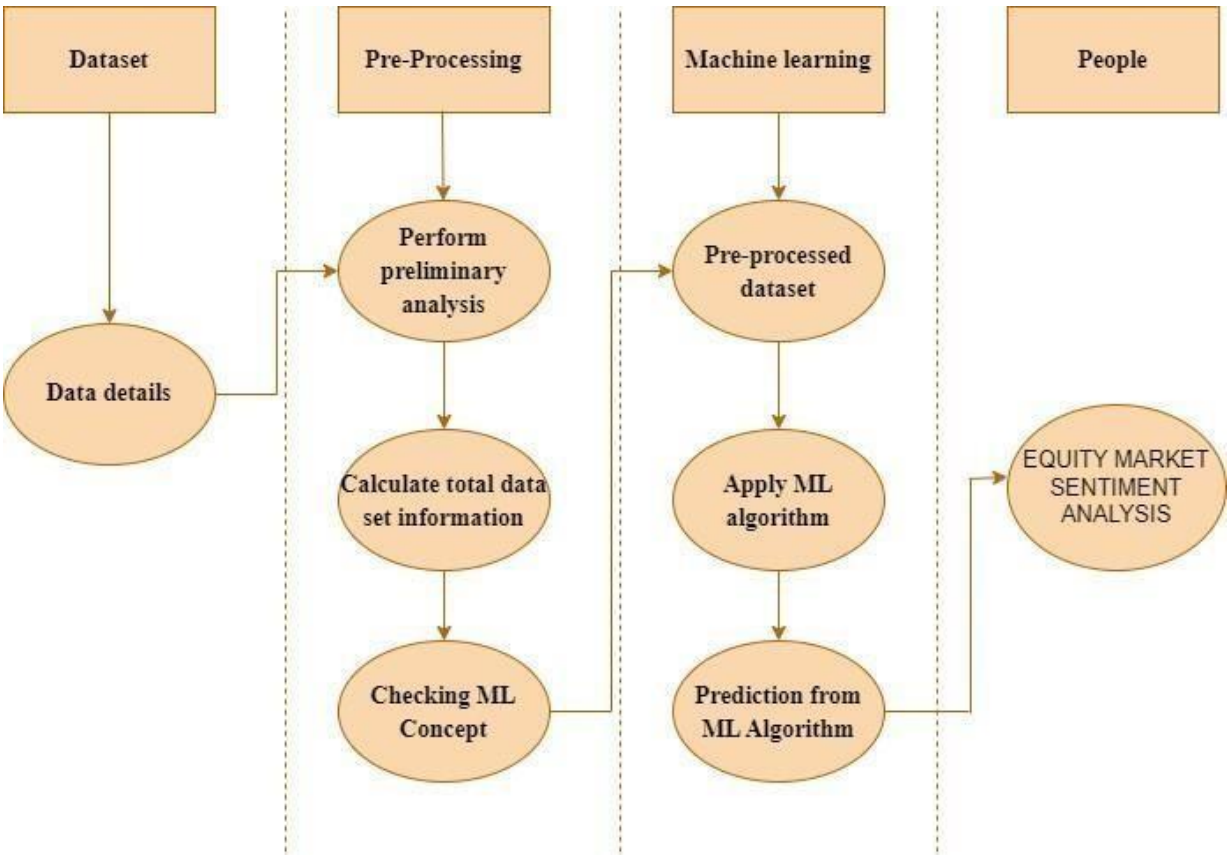

Fig 5.4 Activity diagram

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.

5.3.4 SEQUENCE DIAGRAM

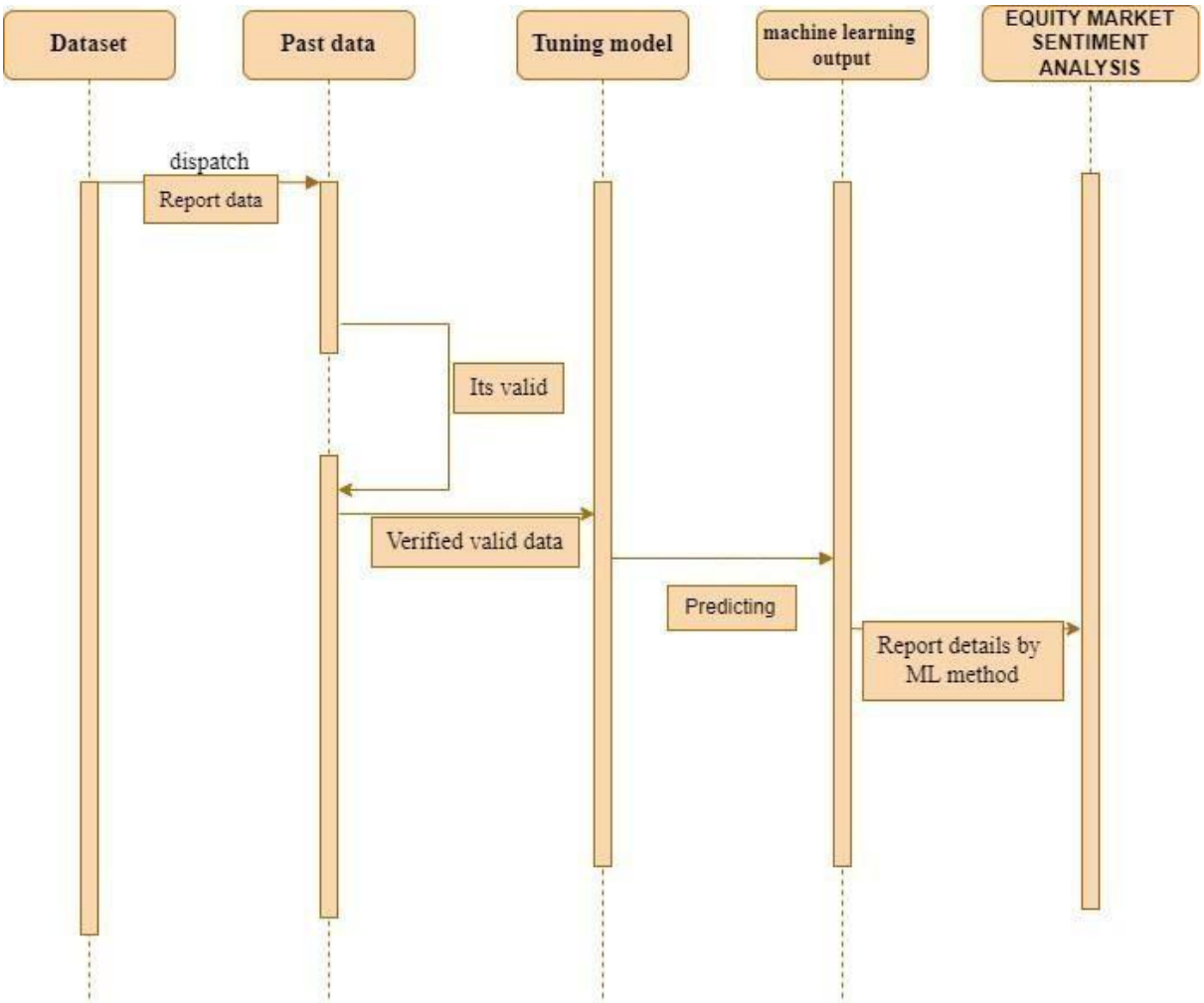

Fig 5.5 Sequence diagram

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 modelling, which focuses on identifying the behaviour within your system. Other dynamic modelling 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.

5.3.5 ENTITY RELATIONSHIP DIAGRAM

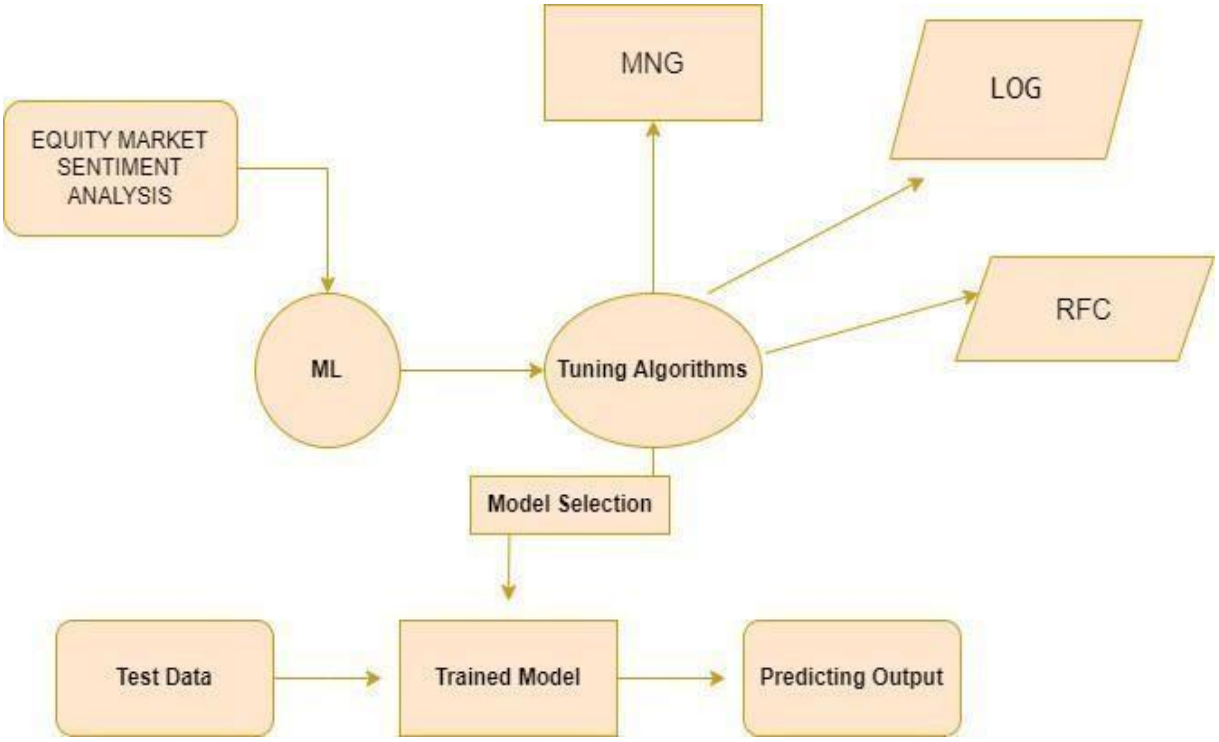

Fig 5.6 ER diagram

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationship 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.

5.3.6 WORKFLOW DIAGRAM

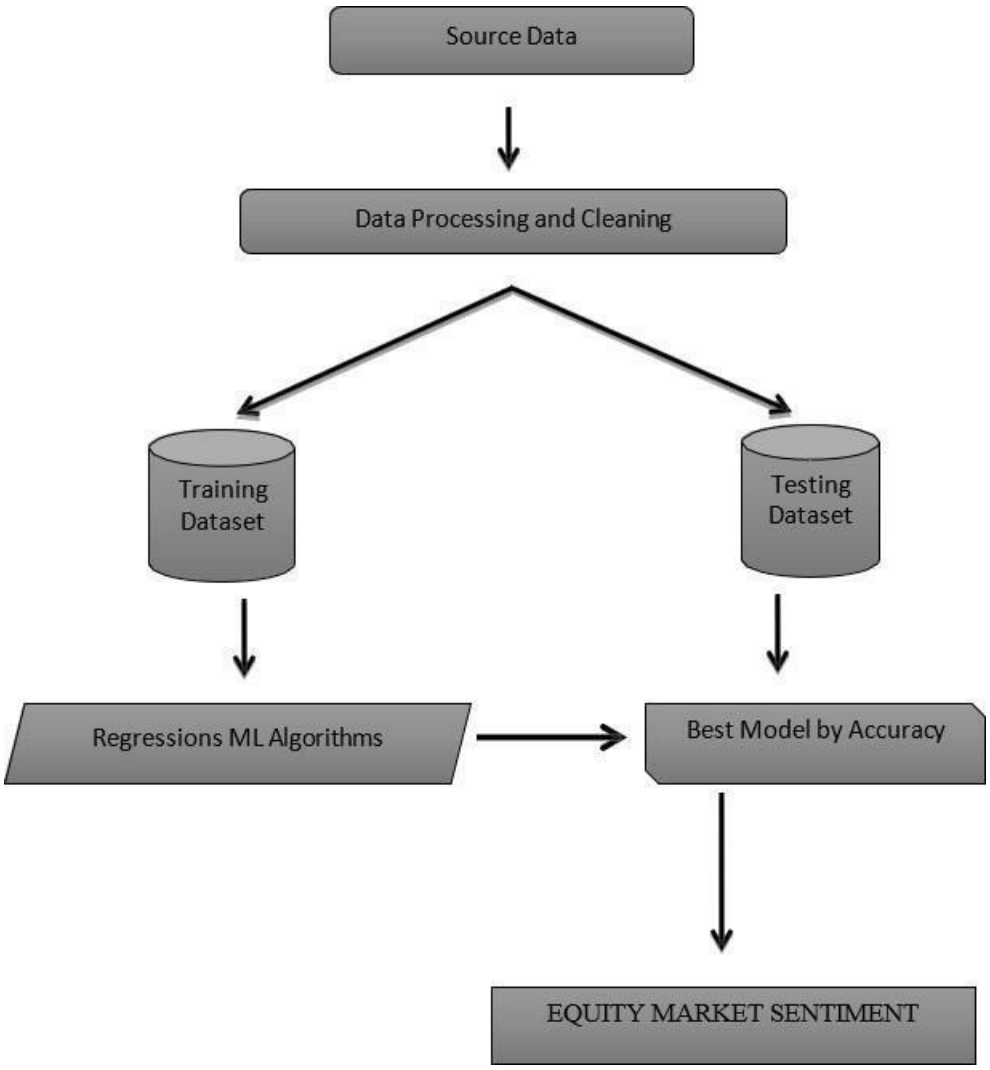

Fig 5.7 Workflow diagram

A workflow diagram in the context of machine learning (ML) represents the sequence of steps or processes involved in a machine learning project. It outlines the high-level structure of how data is collected, prepared, processed, and evaluated to build and deploy machine learning models. Below is a description of the key components and steps typically found in a machine learning workflow diagram:
Data Collection:

The workflow begins with data collection. This step involves gathering relevant data from various sources, which may include databases, external datasets, APIs, or sensors.

Data collection may also encompass the annotation of data, such as labelling images or categorizing text.

Data Pre-processing:

After data collection, the raw data often needs to be cleaned and pre-processed. This step includes handling missing values, removing duplicates, and addressing outliers.

Data pre-processing also involves data normalization, scaling, and feature engineering to make the data suitable for modelling.

Data Splitting:

The dataset is typically divided into training, validation, and testing sets. This separation is crucial for training and evaluating the machine learning model effectively.

Feature Selection and Engineering:

Feature selection involves identifying and selecting the most relevant features from the dataset, which can improve the model's performance and reduce overfitting.

Feature engineering includes creating new features or transforming existing ones to capture meaningful information for the model.

Model Building:

In this step, machine learning models are designed, implemented, and trained using the pre-processed data. This may involve selecting a suitable algorithm, architecture, or framework.

Hyper parameter tuning, cross-validation, and other techniques are applied to optimize the model's performance.

Model Evaluation:

The trained models are evaluated using the validation dataset to assess their performance and make necessary improvements. Common evaluation metrics include accuracy, precision, recall, F1 score, and more, depending on the problem type (classification, regression, etc.).

Model Testing:

After successful validation, the models are tested on the separate testing dataset to assess their generalization ability and ensure they perform well on unseen data.

Model Deployment:

Once a satisfactory model is obtained, it can be deployed into a production environment where it can make predictions on new, real-world data.

Monitoring and Maintenance:

Continuous monitoring of the deployed model is crucial to ensure that it performs as expected over time. This includes tracking data drift and model drift.

Maintenance involves updating the model and retraining it as new data becomes available or as the model's performance degrades.

Feedback Loop: A feedback loop is an integral part of a machine learning workflow. It allows for the continuous improvement of models based on feedback from real- world usage.

Documentation and Reporting:

Throughout the workflow, documentation is essential for keeping track of the entire process, including data sources, pre-processing steps, model configurations, and results.

Regular reporting of model performance and updates may be necessary to inform stakeholders and decision-makers.

A machine learning workflow diagram provides a visual representation of these steps, helping teams and stakeholders understand the process and dependencies. It serves as a guide for the entire ML project, from data collection to model deployment and maintenance, ensuring a systematic and organized approach to building and managing machine learning systems.

CHAPTER 6

IMPLEMENTATION

CHAPTER 6

IMPLEMENTATION

6.1 CODING

```
from django.shortcuts import render, redirect
from . models import UserPersonalModel
from . forms import UserPersonalForm, UserRegisterForm
from django.contrib.auth import authenticate, login,logout
from django.contrib import messages
import numpy as np
import re
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
from sklearn.feature_extraction.text import TfidfVectorizer
import joblib
import pandas as pd
def Landing_1(request):
    return render(request, '1_Landing.html')
def Register_2(request):
    form = UserRegisterForm()
    if request.method =='POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            form.save()
            user = form.cleaned_data.get('username')
```

```
            messages.success(request, 'Account was successfully created. ' + user)
            return redirect('Login_3')
    context = {'form':form}
    return render(request, '2_Register.html', context)
def Login_3(request):
    if request.method =='POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('Home_4')
        else:
            messages.info(request, 'Username OR Password incorrect')
    context = {}
    return render(request,'3_Login.html', context)
def Home_4(request):
    return render(request, '4_Home.html')
def Teamates_5(request):
    return render(request,'5_Teamates.html')
def Domain_Result_6(request):
    return render(request,'6_Domain_Result.html')
def Problem_Statement_7(request):
    return render(request,'7_Problem_Statement.html')
```

```
def Per_Info_8(request):
    if request.method == 'POST':
        fieldss = ['firstname','lastname','age','address','phone','city','state','country']
        form = UserPersonalForm(request.POST)
        if form.is_valid():
            print('Saving data in Form')
            form.save()
        return render(request, '4_Home.html', {'form':form})
    else:
        print('Else working')
        form = UserPersonalForm(request.POST)
        return render(request, '8_Per_Info.html', {'form':form})
vectorizer=joblib.load('C:/Users/harin/DOC/MAIN_PROJECT/CODE/DEPLOYMENT/PROJECT/ APP/VECTOR.pkl')
model = joblib.load('C:/Users/harin/Music/MAIN_PROJECT/CODE/DEPLOYMENT/PROJECT/APP/MODEL.pkl')
def Deploy_9(request):
    if request.method == "POST":
        int_features = [x for x in request.POST.values()]
        input_text2 = int_features[1:]
        print(input_text2)
        if isinstance(input_text2[0], str):
            result = input_text2[0]
        else:
            result = None
```

```
print(result)
def preprocess_text(text):
    if pd.isnull(text):
        return ""
    text = text.lower()
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    stop_words = set(stopwords.words('english'))
    words = [word for word in word_tokenize(text) if word not in stop_words]
    ps = PorterStemmer()
    words = [ps.stem(word) for word in words]
    preprocessed_text = ' '.join(words)
    return preprocessed_text
preprocessed_input = preprocess_text(result)
input_features = vectorizer.transform([preprocessed_input])
predicted_label = model.predict(input_features)[0]
print(f"Predicted Label: {predicted_label}")
if predicted_label == -1:
    return render(request, '9_Deploy.html', {"prediction_text":f"THE NEGATIVE CONTENTS DETECTED IN THIS REVIEWS IN STOCK MARKET."})
elif predicted_label == 1:
    return render(request, '9_Deploy.html', {"prediction_text":f"THE POSITIVE CONTENTS DETECTED IN THIS REVIEWS IN STOCK MARKET ."})
```

```
    else:
        return render (request, '9_Deploy.html')
def Per_Database_10(request):
    models = UserPersonalModel.objects.all()
    return render(request, '10_Per_Database.html', {'models':models})
def Logout(request):
    logout(request)
    return redirect('Landing_1')
```

Form.py

```
from django import forms
from . models import UserPersonalModel
from django.contrib.auth.forms import UserCreationForm
from django.contrib.auth.models import User
class UserRegisterForm(UserCreationForm):
    class Meta:
        model = User
        fields = ['username', 'email', 'password1', 'password2']
class UserPersonalForm(forms.ModelForm):
    class Meta:
        model = UserPersonalModel
        fields = '__all__'
```

Model.py

```
from django.db import models
from django.contrib.auth.models import User
```

```
class UserPersonalModel(models.Model):
    firstname = models.CharField(max_length=100)
    lastname = models.CharField(max_length=100)
    age = models.IntegerField()
    address = models.TextField(null=True, blank=True)
    phone = models.IntegerField()
    city = models.CharField(max_length=100)
    state = models.CharField(max_length=100)
    country = models.CharField(max_length=100)
def__str__(self):
     return
self.firstname,self.lastname,self.age,self.address,self.phone,self.city,self.state,self
.country
```

Urls.py

```
from django.urls import path
from . import views
urlpatterns =[
    path('', views.Landing_1, name='Landing_1'),
    path('Register_2/', views.Register_2, name='Register_2'),
    path('Login_3/', views.Login_3, name='Login_3'),
    path('Home_4', views.Home_4, name='Home_4'),
    path('Teamates_5/', views.Teamates_5, name='Teamates_5'),
    path('Domain_Result_6/',views.Domain_Result_6, name='Domain_Result_6'),
    path('Problem_Statement_7/',views.Problem_Statement_7,
```

```
name='Problem_Statement_7'),
    path('Per_Info_8/', views.Per_Info_8, name='Per_Info_8'),
    path('Deploy_9/', views.Deploy_9, name='Deploy_9'),
    path('Per_Database_10/', views.Per_Database_10, name='Per_Database_10'),
    path('Logout/', views.Logout, name='Logout'),
]
```

Deploy.html

```
{% load static %}
<!DOCTYPE html>
<html>
<head>
 <title>Home</title>
   <link rel="stylesheet" type="text/css" href="{% static 'css/bootstrap.min.css' %}">
 <style>
 .back{
                        background-image: url("/static/images/S9.jpg");
                        background-repeat: no-repeat;
                        background-attachment: fixed;
                        background-size: 100% 100%;
                        }
                        .white{
                        color:white;
                        }
                        .space{
```

```
                    margin:10px 30px;
                    padding:10px 10px;
                    background: lightblue;
                    width:500px
                    }
                    .gap{
                    padding:10px 20px;
                    }
        </style>
</head>
<body class="back">
 <header class="jumbotron" style="height:100px;">
        <div class="container">
        <center>
        <h2>STOCK MARKET SENTIMENT ANALYSIS USING NLP</h2>
        </center>
    </div>
 </header>
 <center><br><br><br>
 <div class="card ml-container" style="width:40%" >
        <form class="form-group" action="{% url 'Deploy_9' %}" method="POST">
                {% csrf_token %}
                <a href="{% url 'Home_4' %}" style="color: red;"><h3 style="color: black;">HOME</h3></a>
```

```
                    <label class="black"  for="">ENTER THE DESCRIPTION HERE</label>
                    <textarea name="message" class="space form-control" rows="9" cols="60"></textarea> <br/>

                    <input type="submit" class="btn btn-success btn-block" style="width:350px;padding:20px" value="PREDICT">
     </form>
  </div><br><br><br><br><br><br><br><br><br><br>
   <div style="background:black;padding:3% 40%">
         <h3 style="color: coral;"> {{ prediction_text }}</h3>
   </div>
</center>
</body>
</html>
```

Datebase.html

```
{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
   <style>
         [role="table"] {
```

```
display: table;
margin: 10px auto 40px;
width: min(600px, 95%);
border-radius: 8px;
border: 1px solid rgb(51, 211, 59);
border-top: 0;
box-shadow: 0 6px 20px rgba(0,0,0,0.2);
box-sizing: border-box;
font: 400 1.0625em/1 "Segoe UI", "Roboto", "Liberation Sans", Arial, sans-serif;
overflow: hidden;
}
[role="table"] > div[id] {
display: table-caption;
font-style: italic;
}
[role="table"] [role="row"] {
display: table-row;
transition: background-color 300ms linear;
}
.gradient [role="row"]:hover {
background:#5e28db;
color:#FFFFFF;
font-weight: 350;
letter-spacing: .015em;
```

```
}
@media screen and (max-width:500px) {
[role="table"] [role="row"] span+span {width: 28%;}
}

[role="table"] [role="cell"],
[role="table"] [role="columnheader"] {
display: table-cell;
padding: clamp(0.25em,4vw,1em);
width: 20em;
}
[role="table"] [role="columnheader"] {
background: linear-gradient(165deg,rgb(44, 192, 192),#062006);
color: #fafafa;
border-bottom: 3px solid black;
font-weight: bold;
border-bottom: thin solid #888;
border-radius: 6px 6px 0 0;
}
[role="table"] h3 {margin-top: 0;}
/*** Color de fondo para las filas alternas (pares o impares)
[role="table"] [role="rowgroup"]:last-child [role="row"]:nth-child(odd)
{
background-color: rgba(0, 0, 0, 0.2);
}
***/
```

```
            .gradient {
            background: #54c029;
            background: -moz-linear-gradient(top, rgb(0, 102, 255) 0%, #8ED433
100%);

            background: -webkit-linear-gradient(top, #ffff00 0%,#40e6cf 100%);
            background: linear-gradient(to bottom, #3cc4d6 0%,#2c88bd 100%);
            filter:progid:DXImageTransform.Microsoft.gradient(
startColorstr='#ffff00', endColorstr='#8ed433',GradientType=0 );
            border-radius: 0 0 6px 6px;
            }
   </style>
</head>
<body>
   <divrole="table"aria-label="Students"ariadescribedby="students_table_desc">
      <div id="students_table_desc">
        <center><h1>PERSONAINFORMATIONS DATABASE</h1></center>
      </div>
      <div role="rowgroup">
        <div role="row">
          <span role="columnheader">FIRST_NAME</span>
          <span role="columnheader">LAST_NAME</span>
          <span role="columnheader">AGE</span>
          <span role="columnheader">ADDRESS</span>
          <span role="columnheader">PHONE</span>
          <span role="columnheader">CITY</span>
```

```
        <span role="columnheader">STATE</span>
        <span role="columnheader">NATIONALITY</span>
      </div>
    </div>
    {% for model in models %}
    <div role="rowgroup" class="gradient">
      <div role="row">
        <span role="cell">{{ model.firstname }}</span>
        <span role="cell">{{ model.lastname }}</span>
        <span role="cell">{{ model.age }}</span>
        <span role="cell">{{ model.address }}</span>
        <span role="cell">{{ model.phone }}</span>
        <span role="cell">{{ model.city }}</span>
        <span role="cell">{{ model.state }}</span>
        <span role="cell">{{ model.country }}</span>
      </div>
    {% endfor %}
    </div>
</body>
</html>
```

6.2 SCREENSHOTS

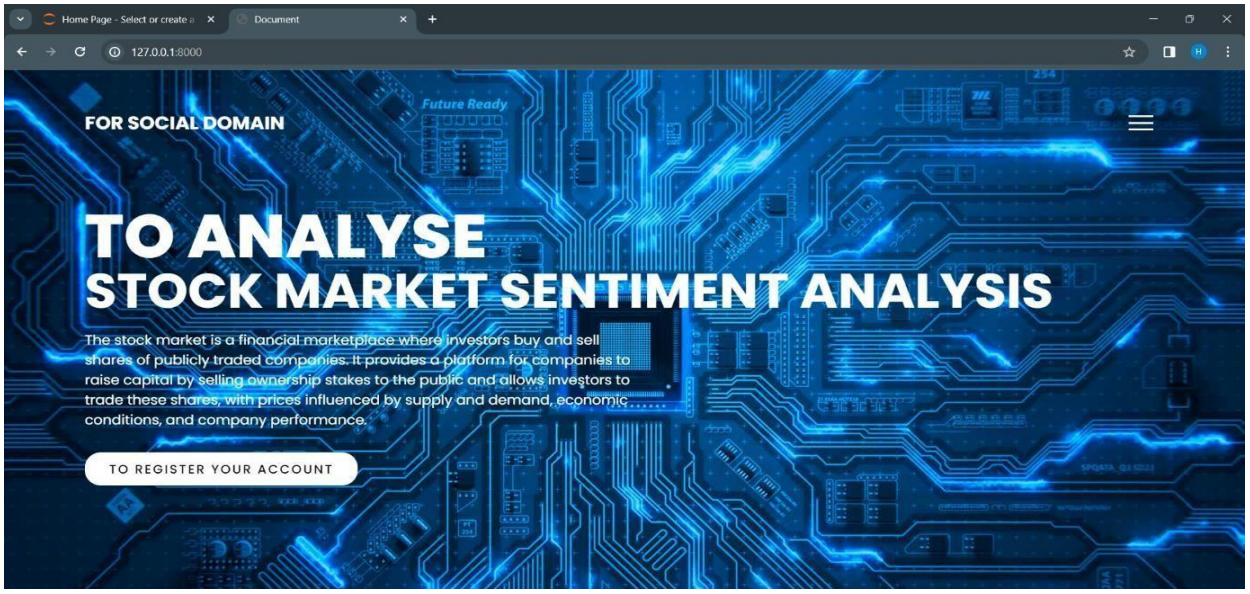

Fig 6.1 Home Page

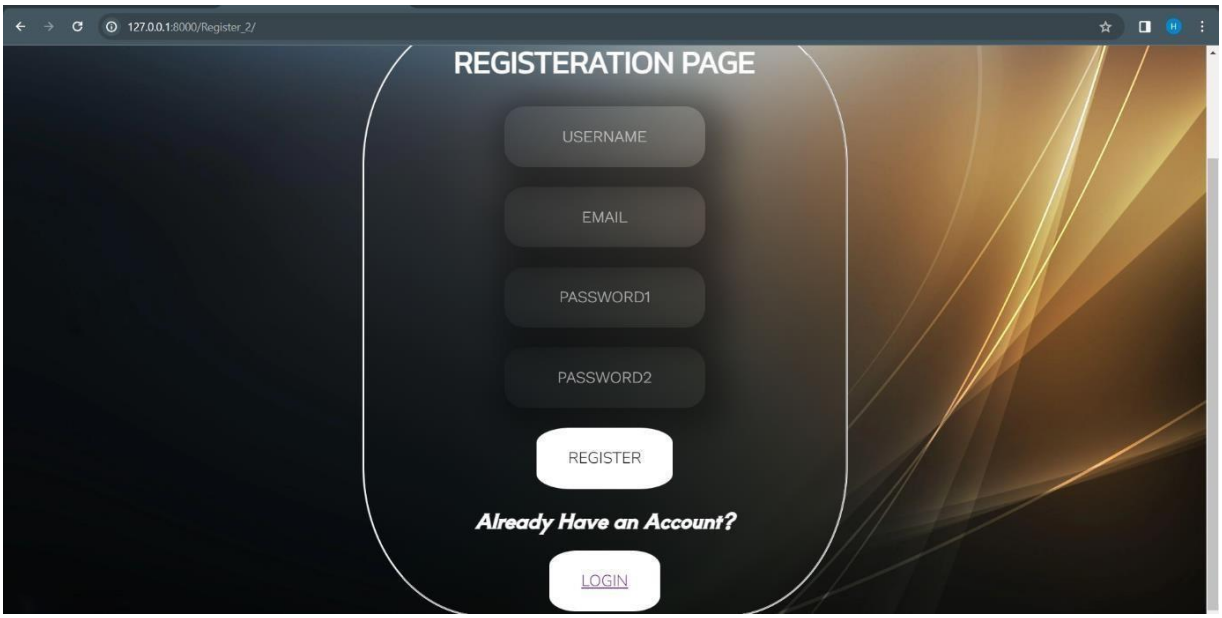

Fig 6.2 Registration Page

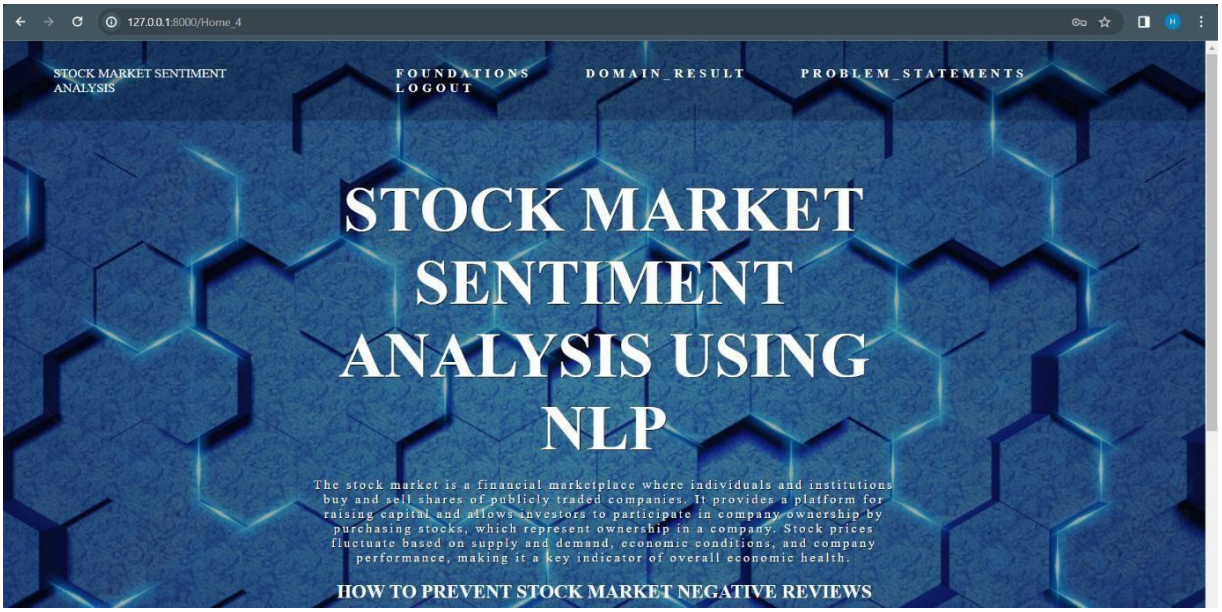

Fig 6.3 Application Page

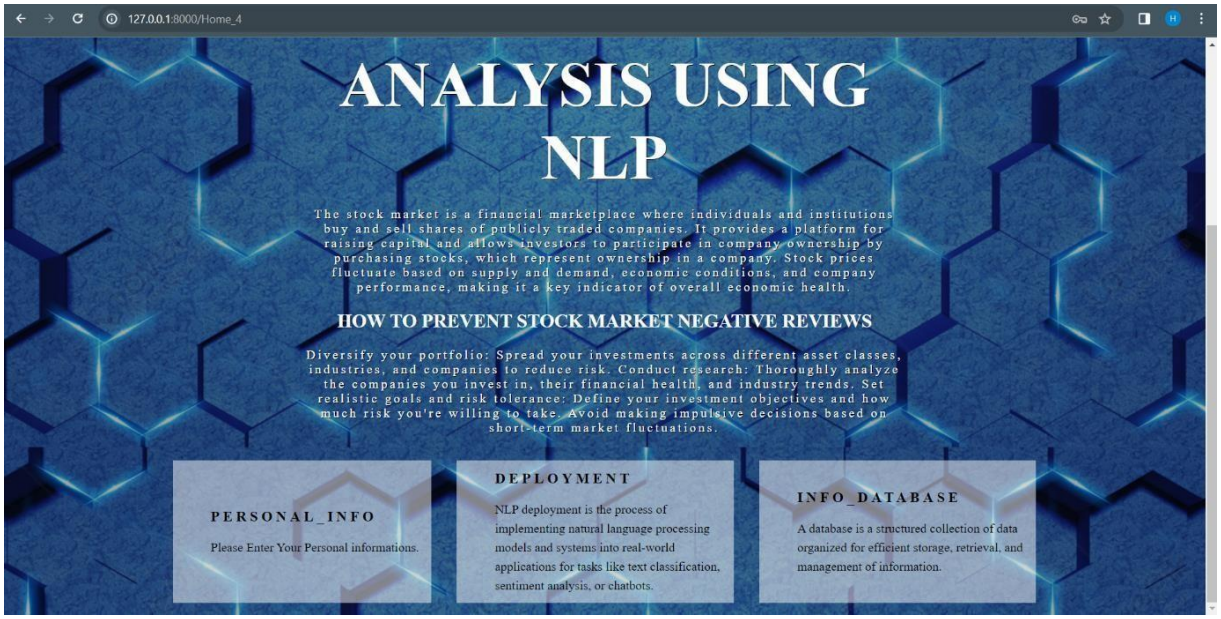

Fig 6.4 Application Detail Page

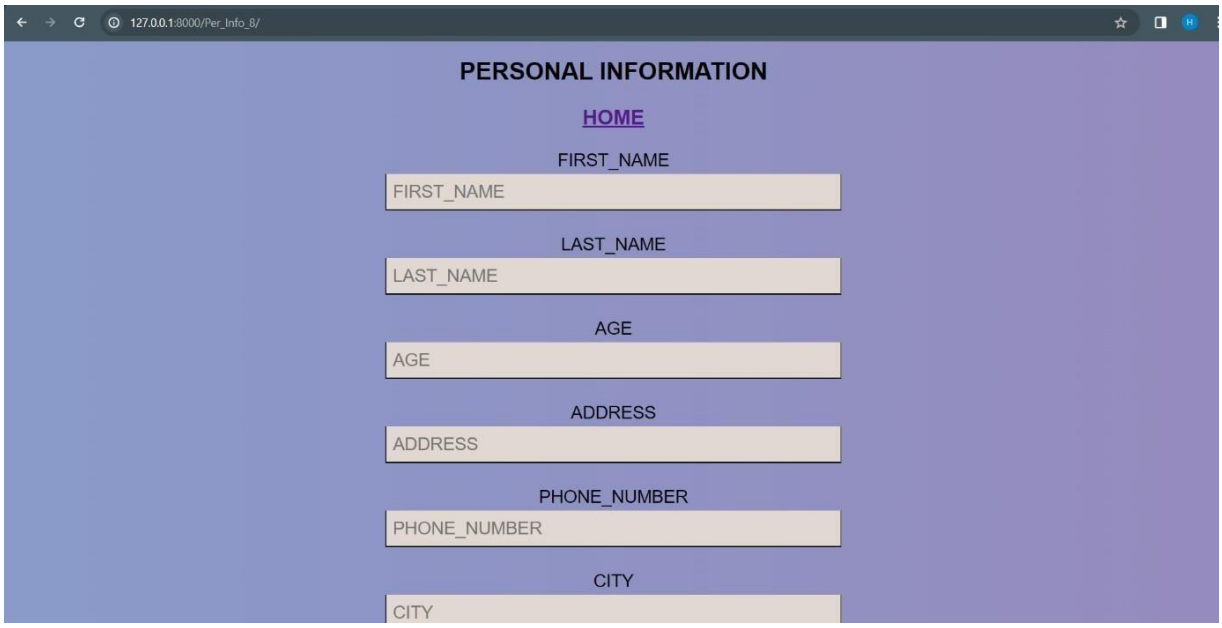

Fig 6.5 Page to enter user information

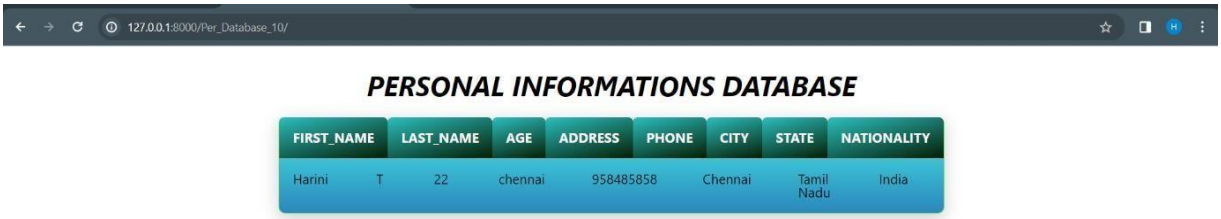

Fig 6.6 Database Page: User login details are stored

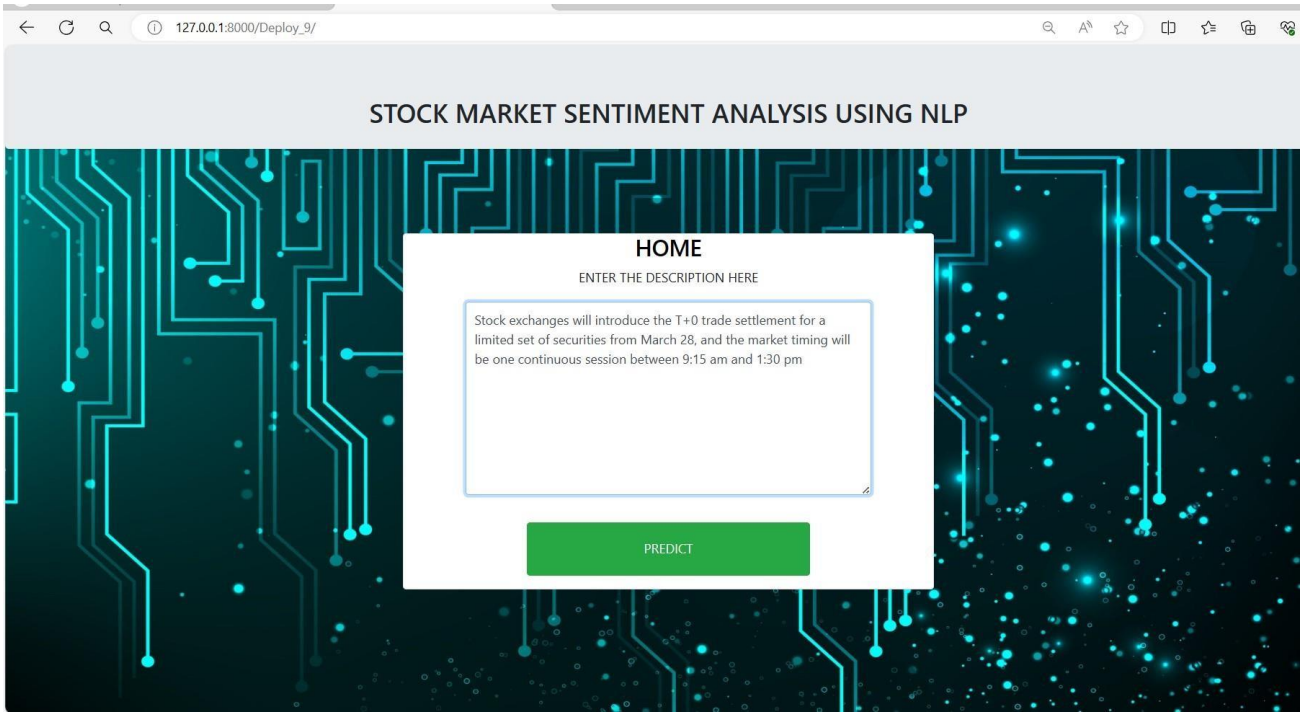

Fig 6.7 Positive content query

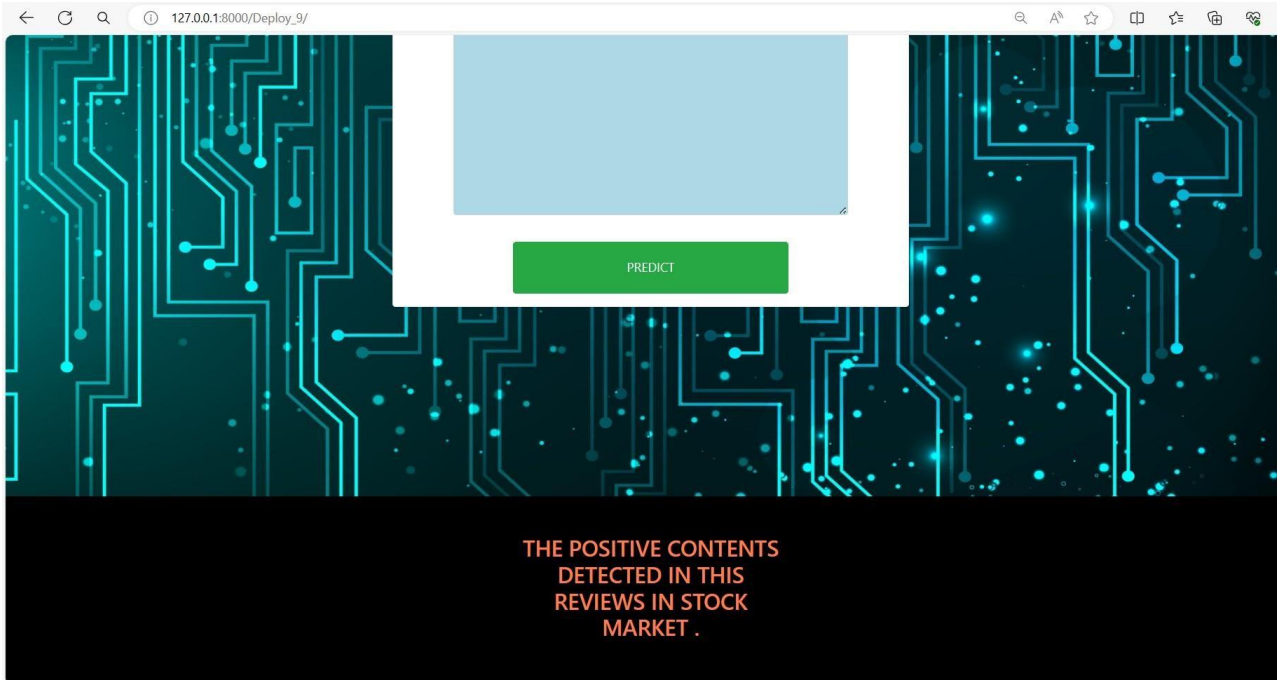

Fig 6.8 Expected Positive Output

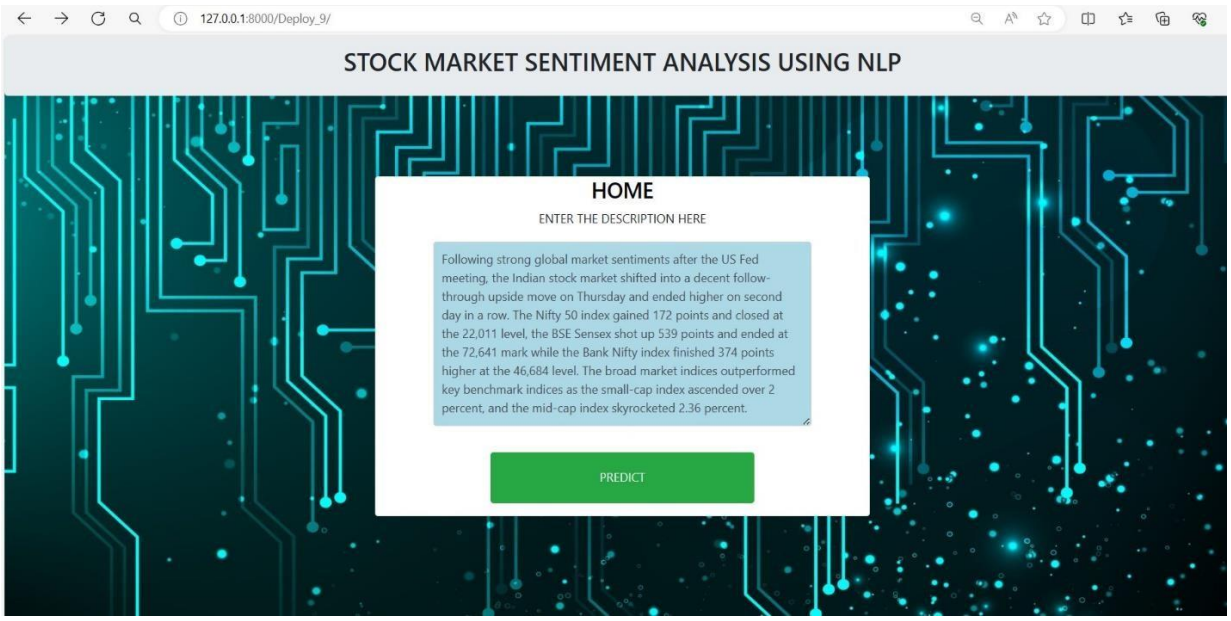

Fig 6.9 Negative Content Query

Fig 6.10 Expected Output

CHAPTER 7

TESTING AND MAINTENANCE

CHAPTER 7

TESTING AND MAINTENANCE

7.1 WHITE BOX TESTING

White-box testing, sometimes called glass-box, is a test case design method that uses the control structure of the procedural design to derive test cases. Using White Box testing methods, we can derive test cases that

- Guarantee that all independent paths within a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.

7.2 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the Software.

7.3 UNIT TESTING

Unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object- oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. Unit testing is software verification and validation method in which the individual units of source code are tested fit for use. A unit is the smallest testable part of an application. In this testing, each class is tested to be working satisfactorily. Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

7.4 FUNCTIONAL TESTING

Functional testing is a quality assurance (QA) process and a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional Testing usually describes what the system does. Functional testing differs from system testing in that functional testing Functional testing typically involves five steps. The identification of functions that the software is expected to perform.

7.4.1 The creation of input data based on the function's specifications.

7.4.2 The determination of output based on the function's specifications

7.4.3 The execution of the test case.

7.4.4 The comparison of actual and expected outputs**.**

7.5 PERFORMANCE TESTING

In general testing performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage. Performance testing is a subset of performance engineering, an emerging computer science practice which strives to build performance into the implementation, design and architecture of a system.

7.6 INTEGRATION TESTING

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when put together. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready. Simulated usage of shared data areas and inter-process communication is tested and individual subsystems are exercised through their input interface.

7.7 VALIDATION TESTING

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it full fills its intended purpose. These are critical components of a quality management system such as ISO 9000. The words "verification" and "validation" are sometimes preceded with "Independent" (or IV&V), indicating that the verification and validation is to be performed by a disinterested third party. It is sometimes said that validation can be expressed by the query "Are you building the right thing?" and verification by "Are you building it right?" In practice, the usage of these terms varies. Sometimes they are even used interchangeably.

7.8 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system. The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together or between any of the assemblages and the hardware. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification.

7.9 OUTPUT TESTING

After performing the validation testing, next step is output testing of the proposed system since no system could be useful if it does not produce the required output generated or considered in to two ways. One is on screen and another is printed format. The output comes as the specified requirements by the user. Hence output testing does not result in any correction in the system. The software may undergo other testing phases and be completely functional but might still not meet its requirements if it is not well received by its intended users. This can happen if software requirements were not clearly defined to the developers, if certain modifications made during development changed the scope of the project or if the software just was not ready to be tested in a dynamic, real-world environment. Overall, UAT safeguards against faulty, ineffective or unfinished software products being released.

7.10 USER ACCEPTANCE TESTING

User acceptance of a system is the factor for the success of any system. The system under consideration is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

7.10.1 Input screen design.

7.10.2 Output screen design.

7.11 TEST CASES

MODULES	INPUT	EXPECTED O/P	ACTUAL O/P	RESULT
REGISTER	User name, mobile number, email id and password	Redirect to login screen	Display login screen	Pass
LOGIN	Registered email id and password	Redirect to home page	Display menu list in the home page	Pass
DEPLOYMENT	Upload the any Financial or stock Related news.	Hit submit	Submitted successfully	Pass
POSITIVE CONTENT	Enter the positive Content for check	Positive feed	Positive feed	Pass
NEGATIVE CONTENT	Enter the negative Related stock News	Negative feed	Negative feed	Pass

Table 7.1: Test Cases

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

CHAPTER 8

CONCLUSION AND FUTURE ENHANCEMENTS

8.1 CONCLUSION

In conclusion, the integration of Natural Language Processing (NLP) techniques into equity market sentiment analysis represents a pivotal advancement in the realm of investment decision-making. Through the systematic extraction and analysis of textual data sourced from various channels such as news articles, social media platforms, and financial reports, NLP facilitates a comprehensive understanding of investor sentiments and market dynamics. One of the key advantages of leveragingNLP in market sentiment analysis is its ability to provide nuanced insights into the collective mood and opinions of investors. By parsing through vast amounts of unstructured textual data, NLP algorithms can identify subtle shifts in sentiment, uncover emerging trends, and detect sentiment anomalies that may signal potential market movements. Moreover, the application of NLP enables investors to stay abreast of market developments in real-time. By processing and analyzing news articles and social media posts as they are published, NLP-powered systems can swiftly identify relevant information and assess its impact on market sentiment. This real-time analysis equips investors with timely insights, allowing them to adapt their investment strategies promptly to changing market conditions. Furthermore, NLP- driven sentiment analysis contributes to the development of predictive models that can anticipate market movements with greater accuracy. By training machinelearning algorithms on historical data enriched with sentiment analysis, these models can learn to recognize patterns and correlations between textual data and market behavior

8.2 FUTURE ENHANCEMENTS

The Future Enhancements include Advanced Sentiment Analysis Models: Incorporating state-of-the-art models like BERT or GPT-3 can significantly enhance theaccuracy and nuance of equity market sentiment predictions. These models excel at understanding contextual nuances and capturing subtle shifts in sentiment, enabling more precise predictions of market movements based on textual data from diverse sources. **Real-Time Data Feeds and Social Media Sentiment Analysis:** By integrating real-time data feeds and advanced social media sentiment analysis techniques, the system can capture dynamic market emotions and provide up-to-the-minute assessmentsof equity market sentiment. This enhancement enables investors to react swiftly to changing market sentiments, improving their ability to make timely and informed investment decisions.

REFERENCES

REFERENCES

[1] Kalyani Joshi, Prof. Bharathi H. N. , Prof. Jyothi Rao (2016) "Stock Trend Prediction using News Sentiment Analysis" International Journal of Computer Science and Information Technology 8(3):67-76 DOI:10.5121/ijcsit.2016.8306

[2] Aastha Saxena, Arpit Jain, Prateek Sharma, Sparsh Singla, and Amrita Ticku (2023) "Sentiment Analysis of Stocks Based on News Headlines Using NLP" V. E. Balas et al. (Eds.): AITEES 2022, AHIS 3, pp. 124–135, 2023. https://doi.org/10.2991/978-94-6463-074- 9_12

[3] Om mane and Sarvanakumar kandasamy (2022) "Stock market prediction using Natural Language Processing -A survey" David C. Wyld et al. (Eds): NLPA, AIS, BDAP, SOENG, IPPR - 2022 pp. 33-48, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121404

[4] Anshul Mittal and Arpit Goel (2011) "Stock Prediction Using TwitterSentiment Analysis" Standford University, CS229(2011 http://cs229. stanford.edu/proj2011/Goel Mittal Stock market Prediction Using Twitter Sentiment Analysis.pdf)

[5] Ayman E. Khedr. et.al., S.E.Salama and Nagwa Yaseen.et.al.,(2017) "Predicting Stock Market Behavior using Data Mining Technique and News Sentiment Analysis" I.J. Intelligent Systems and Applications, 2017, 7, 22-30 Published Online July 2017 in MECS (http://www.mecs-press.org/) DOI: 10.5815/ijisa.2017.07.03

[6] Sheikh Mohammad Idrees, M. Afshar Alam, and Parul Agarwal (2019) "A Prediction Approach for Stock Market Volatility Based on Time Series Data" Digital Object Identifier 10.1109/ACCESS.2019.2895252

[7] S. AL Wadi , Mohammad Almasarweh & Ahmed Atallah Alsaraireh (2018)

"Predicting Closed Price Time Series Data Using ARIMA Model" Modern Applied Science; Vol. 12, No. 11; 2018 ISSN 1913-1844E-ISSN 1913-1852 doi:10.5539/mas.v12n11p181 URL: https://doi.org/10.5539/mas.v12n11p181

[8] Fazlija, B.; Harder, P. (2022) "Using Financial News Sentiment for Stock Price Direction Prediction" Mathematics 2022, 10, 2156. https:// doi.org/10.3390/math10132156

[9] Karlo Puh and Marina Bagic Babac (2023) "Predicting stock market using natural language processing" American Journal of Business Vol. 38 No. 2, 2023 pp. 41-61 Emerald Publishing Limited 1935- 5181 DOI 10.1108/AJB-08-2022-0124.

[10] L, S. and B R, S. (2023), "Combined deep learning classifiers for stock market prediction: integrating stock price and news sentiments", Kybernetes, Vol. 52 No. 3, pp. 748- 773. https://doi.org/10.1108/K-06-2021- 0457

[11] Ji, X., Wang, J. and Yan, Z. (2021), "A stock price prediction method based on deep learning technology", International Journal of Crowd Science, Vol. 5 No. 1, pp. 55-72.

[12] Kameshwari, S., Kaniskaa, S., Kaushika, S . and Anuradha, R. (2021), "Stock trend prediction using news headlines", 2021 IEEE India Council International Subsections Conference (INDISCON),pp1-5,doi: 10.1109/INDISCON53343.2021.9582 219