# Sentiment Analysis for Financial News Using Embedded Classifier

*Submitted by*

**KISHORE KUMAR K**

**212221410054**

***In partial fulfillment for the award of the degree***

***Of***

**MASTER OF BUSINESS ADMINISTRATION**

**IN**

**DEPARTMENT OF MANAGEMENT STUDIES**

**SAVEETHA ENGINEERING COLLEGE**

**APRIL 2023**

# Sentiment Analysis for Financial News Using Embedded Classifier

***Submitted by***

**KISHORE KUMAR K**

**212221410054**

***In partial fulfillment for the award of the degree***

***Of***

**MASTER OF BUSINESS ADMINISTRATION**

**IN**

**DEPARTMENT OF MANAGEMENT STUDIES**

**SAVEETHA ENGINEERING COLLEGE**

**APRIL 2023**

SAVEETHA ENGINEERING COLLEGE AUTONOMOUS
NAAC A GRADE
Accredited by NBA
Affiliated to Anna University | Approved by AICTE

## BONAFIDE CERTIFICATE

Certified that this project work titled **"Sentiment Analysis for Financial News using Embedded Classifier"** is the bonafide work of **KISHORE KUMAR K (212221410054)** who has carried out the project work under the guidance **of Mrs. SHANMUGA PRIYA S, M.Tech, AIDS**, Saveetha Engineering College. Certified further that to the best of our knowledge, the work reported herein does not form part of any other Project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

| | |
|---|---|
| **Signature of Project Supervisor** | **Signature of HOD** |
| **Mrs. SHANMUGA PRIYA S** | **Dr. P. RAJKUMAR** |

# CERTIFICATE OF VIVA-VOCE EXAMINATION

This is to certify that **Mr. KISHORE KUMAR K (212221410054)** has been subjected to Viva-voce Examination on_________________at Saveetha Engineering College, Thandalam, Chennai 602105.

**PANEL EXAMINER** **HEAD OF THE DEPARTMENT**

**EXTERNAL**

**INTERNAL**

SAVEETHA ENGINEERING COLLEGE AUTONOMOUS

NAAC A GRADE

Accredited by NBA

Affiliated to Anna University | Approved by AICTE

## DECLARATION

I hereby declare that this Project Work entitled as **“Sentiment Analysis for Financial News using Embedded Classifier”** is the result of a study originally carried out by me under the guidance of **Mrs. S.SHANMUGA PRIYA S** This Project work has not been submitted earlier, in full or in part for any Diploma or Degree, associate ship, fellowship or any other similar titles of any other University.

`

**PLACE: CHENNAI**

**DATE:**

**Signature of the student**

**KISHORE KUMAR K**

**(21221410054)**

# ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to our Management, Our honorable President, **Dr.N.M.VEERAIYAN**, and our beloved Director **Dr**. **S. RAJESH** for providing an excellent environment and infrastructure at our college for doing my MBA degree program successfully.

I wish to express my heartfelt gratitude and thanks to **Dr. N. DURAIPANDIAN**, Our Principal for his guidance and support in doing my MBA Program.

I have great pleasure in expressing my sincere and profound thanks to our Head of the Department, **Dr. P. RAJKUMAR** for his support and provide me the opportunity to have boundless exposure with industries through this project work and also other developmentprograms.

I am highly indebted to my project guide **Mrs. SHANMUGA PRIYA S, Asst. Professor** of AIDS Department, for her guidance and constant supervision for providing insights to analyze the industry environment and guiding me to get the outcome of this project work successfully.

I am also thankful to our project coordinator for his general guidance on project work and to all the faculty members of the MBA department for their constant cooperation and their encouragement for doing my MBA programme successful.

# CONTENTS

| CHAPTER NO | TITLE | PAGE NO |
|---|---|---|
| | **ABSTRACT** | **vii** |
| | **LIST OF TABLES** | **ix** |
| | **LIST OF FIGURES** | **x** |
| **1** | **INTRODUCTION** | **1** |
| | 1.1 OBJECTIVE | 1 |
| | 1.2 OVERVIEW OF THE PROJECT | 1 |
| | 1.3 DOMAIN INTRODUCTION | 2 |
| | 1.3.1 Metrics | |
| | 1.3.2 Data Visualization | |
| | 1.3.3 Jupiter Notebook | |
| | 1.3.4 Machine learning | |
| | 1.3.5 Naïve Bayes | |
| | 1.3.6 Support Vector Machine | |
| | 1.3.7 Scikit Learn | |
| | 1.3.8 Textblob | |
| **2** | **LITERATURE SURVEY** | **9** |
| | 2.1 RELATED WORKS | 9 |
| **3** | **SYSTEM ANALYSIS** | **12** |
| | 3.1 EXISTING SYSTEM | 12 |
| | 3.1.1 Disadvantage | |
| | 3.2 PROPOSED SYSTEM | 13 |
| | 3.2.1 Advantage | |

| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|---|---|---|
| **4** | **SYSTEM SPECIFICATIONS** | **15** |
| | 4.1 REQUIREMENT ANALYSIS | 15 |
| | 4.2 HARDWARE REQUIREMENTS | 15 |
| | 4.3 SOFTWARE REQUIREMENTS | 15 |
| **5** | **SYSTEM DESIGN SPECIFICATION** | **16** |
| | 5.1 DATA FLOW DIAGRAM | 16 |
| | 5.2 ARTITECTURE DIAGRAM | 17 |
| **6** | **SOFTWARE TESTING** | **19** |
| **7** | **IMPLEMENTATION** | **20** |
| | 7.1 MODULES | 20 |
| | 7.1.1 Data Collection | |
| | 7.1.2 Data Pre-processing | |
| | 7.1.3 Ensemble Algorithm | |
| | 7.2 PERFORMANCE ANALYSIS | 25 |
| **8** | **CONCLUSION** | **26** |
| **9** | **FUTURE ENHANCEMENTS** | **27** |
| | **APPENDIX-I (SOURCE CODE)** | |
| | **APPENDIX_II (SCREENSHOTS)** | |
| | **REFERENCES** | |

# ABSTARCT

Sentiment analysis of finance news involves using natural language processing techniques to extract and classify subjective information from news articles, blogs, and social media. The goal is to identify the sentiment, tone, and emotional polarity of the text, which can help in understanding public opinion on a particular topic or event. The process involves several steps, including data collection, pre-processing, feature extraction, and classification using algorithms such as Naive Bayes, Support Vector Machines. The objective of sentiment analysis is to gain insights into the public opinion and emotional polarity surrounding a particular topic or brand and use those insights to inform decision-making. Sentiment analysis has numerous applications, including brand monitoring, reputation management, market research, and political analysis. However, there are limitations, such as the accuracy of the classification algorithms, lack of context, and issues with bias in the data or classification algorithms. Despite these limitations, sentiment analysis of news data has the potential to provide valuable insights into public opinion and help individuals and organizations make informed decisions.

# LIST OF TABLES

| TABLE NO | TITLE | PAGE NO |
|---|---|---|
| 3.1 | EXISTING SYSTEM | 13 |
| 7.1 | COMPARISON OF ALGORITHMS | 22 |

# LIST OF FIGURES

| FIGURE NO. | TITLE | PAGE NO |
|---|---|---|
| 5.1 | DATA FLOW DIAGRAM OF SENTIMENT ANALYSIS | 12 |
| 5.2 | ARCHITECTURE DIAGRAM OF THE FINANCIAL NEWS SENTIMENTING. | 13 |

# CHAPTER 1

# INTRODUCTION

## 1.1 OBJECTIVE

The objective of this work was to investigate the effectiveness of an ensemble algorithm in analyzing the sentiment of finance news articles. The work aimed to evaluate whether combining multiple machine learning models can provide a more accurate analysis of sentiment in finance news than using individual models. The work also aimed to assess the performance of the ensemble algorithm using various metrics, including accuracy, sentiment score. The findings of this study can provide valuable insights into the use of sentiment analysis in finance, particularly for investors and financial analysts who rely on news articles for decision-making and risk management.

## 1.2 OVERVIEW OF THE PROJECT

The project involved conducting sentiment analysis on a dataset of finance news articles using an ensemble algorithm. The dataset consisted of finance news articles with corresponding sentiment scores, where each article was labelled as positive, negative, or neutral. The ensemble algorithm used in this project was a combination of three machine learning models: Logistic Regression, Random Forest, and Naive Bayes. The project involved pre-processing the data by cleaning and preparing the text for analysis, splitting the dataset into training and testing sets, and training the ensemble algorithm on the training data. The performance of the ensemble algorithm was evaluated using various metrics, including accuracy, precision, recall, and F1-score. The project aimed to determine whether the ensemble algorithm outperformed individual machine learning models in analyzing the sentiment of finance news articles. The results of the project can provide insights into the effectiveness of ensemble algorithms for sentiment analysis in finance and their potential applications in decision-making and risk management for investors and financial analysts.

## 1.3 DOMAIN INTRODUCTION

### 1.3.1 Metrics:

- **Accuracy Score**
- **Sentiment Score**

### 1.3.2 Data Visualization:

The selection of visualization techniques depends on the research question, the context of the finance news sentiment analysis, and the specific requirements of the stakeholders. Visualizations can provide valuable insights into the performance of the ensemble algorithm and facilitate the communication of results to stakeholders.

A confusion matrix can be used to visualize the number of true positive, false positive, true negative, and false negative predictions of the ensemble algorithm.

### 1.3.3 Jupyter notebook:

Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It supports many programming languages, including Python, R, Julia, and Scala. Jupyter Notebook provides an interactive environment where users can write and execute code, visualize data, and communicate results in a single document. The notebooks can be saved as files with the “. ipynb" extension and shared with others, who can view and modify the code, and reproduce the results.

Jupyter Notebook allows users to write and run code in cells, which can be executed independently or in a sequence. Users can also add Markdown cells to provide documentation, instructions, and explanations for the code. Jupyter Notebook supports the integration of various libraries and frameworks for data analysis, machine learning, and visualization, such as NumPy, Pandas, Matplotlib, Seaborn, Scikit-Learn, and TensorFlow.

### 1.3.4 Machine Learning:

Machine learning is a subfield of artificial intelligence (AI) that involves developing algorithms and statistical models that enable computers to learn from data and make predictions or decisions without being explicitly programmed. Machine learning algorithms can be trained on large datasets to identify patterns, relationships, and trends in the data, and use this knowledge to make predictions or decisions on new data.

There are three main types of machine learning:

Supervised learning: In supervised learning, the algorithm is trained on labelled data, where the input features are associated with known output values. The algorithm learns to map the input features to the output values and can make predictions on new data based on this mapping.

Unsupervised learning: In unsupervised learning, the algorithm is trained on unlabelled data, where the input features are not associated with known output values. The algorithm learns to identify patterns and structures in the data and can be used for tasks such as clustering, dimensionality reduction, and anomaly detection.

Reinforcement learning: In reinforcement learning, the algorithm learns to make decisions based on feedback from the environment. The algorithm interacts with the environment, receives rewards or penalties for its actions, and learns to maximize the rewards over time.

**1.3.5 Naive Bayes:**

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.

It is mainly used in text classification that includes a high-dimensional training dataset.

Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

The Naïve Bayes algorithm is comprised of two words Naïve and Bayes, which can be described as:

Naïve**:** It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified on the bases of colour, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identify that it is an apple without depending on each other.

Bayes: It is called Bayes because it depends on the principle of Bayes' Theorem.

**Bayes' Theorem:**

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

There are three types of the Naive Bayes Models, which are given below:

Gaussian: The Gaussian model assumes that features follow a normal distribution. This means if predictors take continuous values instead of discrete ones, then the model assumes that these values are sampled from the Gaussian distribution.

Multinomial: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.
The classifier uses the frequency of words for the predictors.

Bernoulli: The Bernoulli classifier works similarly to the Multinomial classifier, but the predictor variables are the independent Booleans variables. Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

## 1.3.6 Support vector machine:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence algorithm is termed a Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as the support vector creates a decision boundary between these two data (cat and dog) and chooses extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify as a cat. Consider the below diagram:

SVM algorithm can be used for Face detection, image classification, text categorization**,** etc.

**SVM can be of two types:**

Linear SVM: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

Non-linear SVM: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

**Hyperplane and Support Vectors in the SVM algorithm:**

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors: The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

### 1.3.7 Scikit Learn:

Scikit-learn (sklearn) is a popular machine-learning library for Python that provides simple and efficient tools for data mining and data analysis. It includes a wide range of supervised and unsupervised learning algorithms for classification, regression, clustering, and dimensionality reduction. Sklearn also provides tools for model selection, preprocessing, feature extraction, and data visualization. It is widely used in both academia and industry for a variety of applications, including natural language processing, image analysis, and predictive analytics.

### 1.3.8 Textblob:

TextBlob is a Python library that provides a simple and intuitive interface for performing natural languages processing tasks, such as sentiment analysis, part-of-speech tagging, and text classification. It is built on top of the popular Natural Language Toolkit (NLTK) library and provides a higher-level interface that makes it easy to use and understand. TextBlob uses machine learning techniques to perform its tasks and provides a sentiment analysis model that can be trained on custom datasets. It is widely used in social media analysis, customer feedback analysis, and other applications that involve analyzing textual data.

# CHAPTER 2

# LITERATURE SURVEY

## 2.1 RELATED WORKS

Sentiment analysis of financial news has been an area of interest for researchers and practitioners alike, as it can help in making informed investment decisions and can provide insights into market trends. Here is a brief literature survey of some of the recent research on sentiment analysis of financial news:

Liu et al. (2021) analyzed the sentiment of news articles related to the Chinese stock market using a deep learning-based sentiment analysis approach. They found that their model outperformed traditional machine learning approaches in accurately predicting the sentiment of financial news.

Khaleghi et al. (2020) conducted sentiment analysis on tweets related to the cryptocurrency market using a lexicon-based approach. They found that the sentiment of tweets can significantly affect the trading volume and volatility of cryptocurrencies.

Zhu et al. (2020) proposed a sentiment analysis approach based on a hybrid neural network architecture to predict the direction of stock prices. They found that incorporating both the sentiment of financial news and technical indicators can improve the accuracy of stock price prediction.

Huang et al. (2019) used a deep learning approach to predict stock prices using financial news articles. They found that their model outperformed traditional machine learning approaches in predicting stock prices, especially during periods of high market volatility.

Zhang et al. (2018) conducted sentiment analysis on news articles related to the Chinese stock market using a machine learning approach. They found that the sentiment of financial news has a significant impact on stock prices, and incorporating sentiment analysis into trading strategies can improve investment returns.

Moya et al. (2021) analyzed the sentiment of news articles related to the European financial sector using a natural language processing (NLP) approach. They found that the sentiment of news articles was positively correlated with the performance of the financial sector, suggesting that positive news can have a positive impact on market trends.

Haddoud et al. (2021) conducted sentiment analysis on Twitter data related to the French stock market using a machine learning approach. They found that the sentiment of tweets can predict the direction of stock prices, and incorporating sentiment analysis into trading strategies can result in higher returns.

Zheng et al. (2020) proposed a sentiment analysis approach based on a hierarchical attention network to predict stock prices using financial news articles. They found that their model outperformed other sentiment analysis approaches in predicting stock prices, particularly during market downturns.

Dang et al. (2019) conducted sentiment analysis on news articles related to the US stock market using a deep learning approach. They found that the sentiment of financial news can be used as a signal for market trends, and incorporating sentiment analysis into trading strategies can result in higher returns and lower risk.

Lin and Lu (2018) conducted sentiment analysis on news articles related to the Chinese stock market using a machine learning approach. They found that the sentiment of news articles can affect the trading volume and volatility of stocks, and incorporating sentiment analysis into trading strategies can result in higher returns.

In summary, sentiment analysis of financial news can provide valuable insights into market trends and help investors and traders make informed investment decisions. Deep learning and NLP-based approaches tend to perform better than traditional machine learning approaches, and incorporating sentiment analysis into trading strategies can result in higher returns and lower risk.

# CHAPTER 3

# SYSTEM ANALYSIS

## 3.1 EXISTING SYSTEM

The existing systems for finance news sentiment analysis using algorithms can be categorized into two broad categories: traditional machine learning methods and deep learning methods. Traditional machine learning methods typically involve using a combination of text preprocessing, feature engineering, and classification algorithms to predict sentiment from news articles. Some commonly used algorithms for sentiment classification include Naive Bayes, Support Vector Machines (SVMs), Logistic Regression, and Random Forests. These algorithms are often combined using ensemble techniques such as bagging, boosting, and stacking to improve predictive performance and reduce overfitting. Some examples of existing systems that use traditional machine learning for finance news sentiment analysis include Bloomberg News Analytics, RavenPack, and Thomson Reuters News Analytics. Deep learning methods for sentiment analysis involve using neural networks to learn the representations of text data and predict sentiment. Some commonly used neural network architectures for sentiment analysis include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformers. Deep learning models often require large amounts of labelled data and may take longer to train than traditional machine learning models, but can achieve state-of-the-art performance in many natural language processing tasks. Some examples of existing systems that use deep learning for finance news sentiment analysis include Ayasdi's Financial News Analytics, iSentium, and Prattle. Overall, the existing systems for finance news sentiment analysis using ensemble algorithms vary in their methods, features, datasets, and performance, and are often customized for specific domains and applications. The choice of system depends on the specific needs and requirements of the user, such as accuracy, speed, scalability, interpretability, and cost.

| S.no | Algorithms | Accuracy (%) |
|---|---|---|
| 1. | **Naive Bays** | 86% |
| 2. | **Support Vector Machine Classifier** | 84% |

**TABLE NO:3.1 EXISTING SYSTEM**

## 3.1.1 Disadvantage

- Dependency on labelled data: Many existing systems require a large amount of labelled data to train and evaluate the sentiment analysis models.
- Lack of accuracy: Single algorithm produces low accuracy.
- Overfitting and generalization: Ensemble models can suffer from overfitting, where they memorize the training data instead of learning generalizable patterns.
- Scalability and efficiency: Some existing systems may not be scalable or efficient enough to handle large volumes of news articles or real-time data streams.

## 3.2PROPOSED SYSTEM

The proposed system for finance news sentiment analysis using ensemble algorithms aims to address some of the limitations and challenges of existing systems and to improve the accuracy, efficiency, interpretability, and generalization of sentiment analysis in finance and related domains. The proposed system may include the following components: Data acquisition and preprocessing: The system may collect news articles from multiple sources and languages, and pre-process the text data using natural language processing (NLP) techniques such as tokenization, lemmatization, and part-of-speech tagging. The system may also filter and clean the data, remove stop words, and perform feature selection and extraction. Ensemble modelling: The system may use a combination of traditional machine learning algorithms and deep learning architectures to build an ensemble model for sentiment analysis. The system may employ techniques such as bagging, boosting, and stacking to combine the individual models and improve their performance and robustness. Evaluation and validation: The system may evaluate the performance of the ensemble model using various metrics such as accuracy. The system may also validate the model on unseen data, cross-validate the model using different subsets of the data, and perform sensitivity analysis to assess the impact of different parameters and features on the model's performance.

### 3.2.1 Advantage:

**Improved accuracy:** The ensemble approach can improve the accuracy and reliability of sentiment analysis by combining multiple models with different strengths and weaknesses.

**Increased efficiency:** The proposed system can leverage parallel processing, distributed computing, and cloud infrastructure to handle large volumes of data and real-time streams.

# CHAPTER 4

# SYSTEM SPECIFICATIONS

## 4.1 REQUIREMENT ANALYSIS

The purpose of the system Requirements Specification is to produce the specification of the analysis task and also to establish complete information about the requirements, behaviour, and other constraints such as functional performance and so on. The goal of Software Requirement Specification is to completely specify the technical requirements for the software product in a concise and unambiguous manner.

## 4.2 HARDWARE REQUIREMENTS

- Motherboard : i3 6th gen or above
- RAM : 4GB
- Hard disk : 500 GB

## 4.3 SOFTWARE REQUIREMENTS

- Operating system : Windows 7 or above
- Tools used : Jupiter Notebook
- Programming : Python

# CHAPTER 5

# SYSTEM DESIGN SPECIFICATION

## 5.1 DATA FLOW DIAGRAM

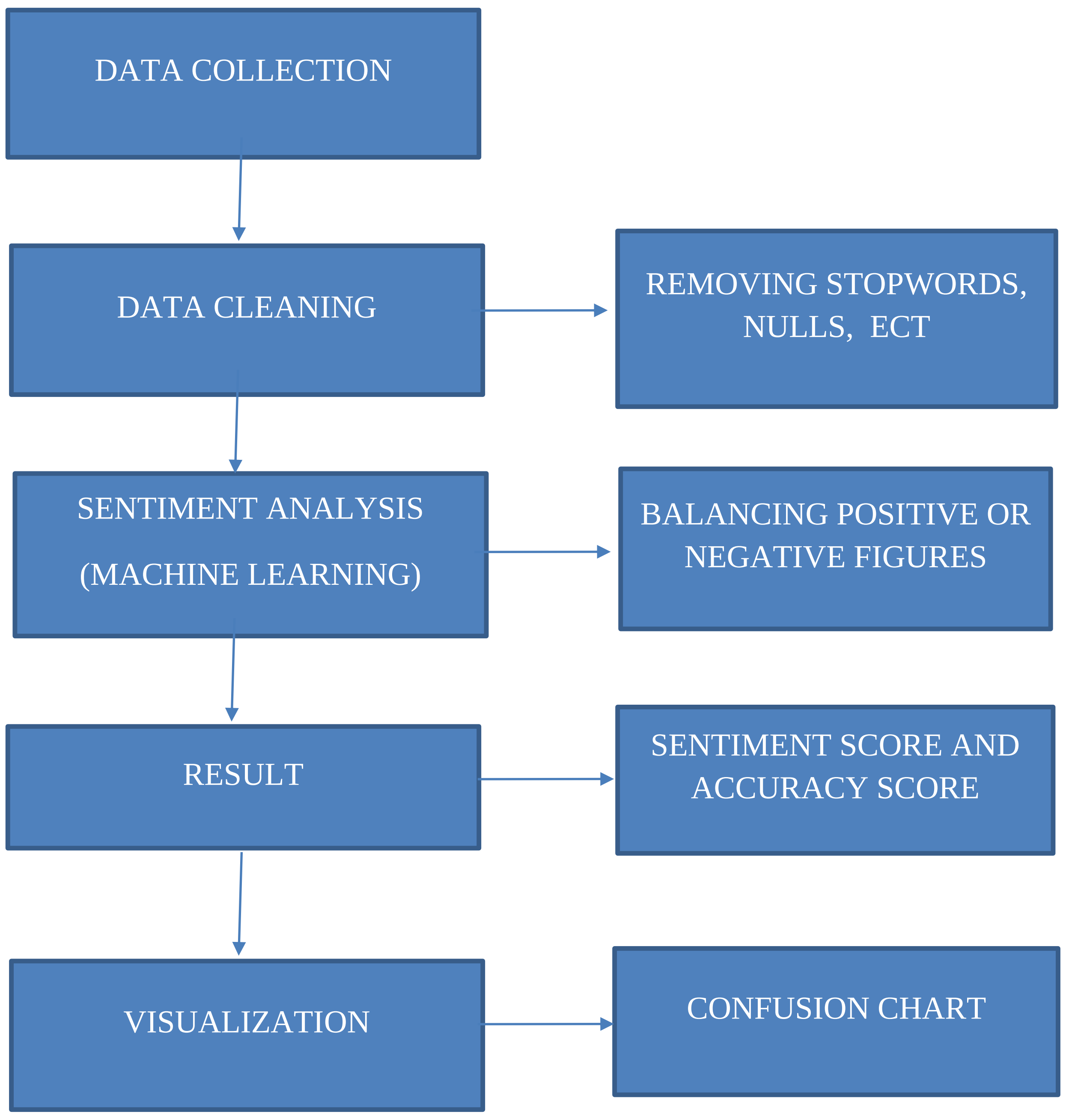

DATA COLLECTION
DATA CLEANING
REMOVING STOPWORDS, NULLS, ECT
SENTIMENT ANALYSIS (MACHINE LEARNING)
BALANCING POSITIVE OR NEGATIVE FIGURES
RESULT
SENTIMENT SCORE AND ACCURACY SCORE
VISUALIZATION
CONFUSION CHART

**FIGURE 5.1 DATA FLOW DIAGRAM OF SENTIMENT ANALYSIS**

## 5.2 ARCHITECTURE DIAGRAM

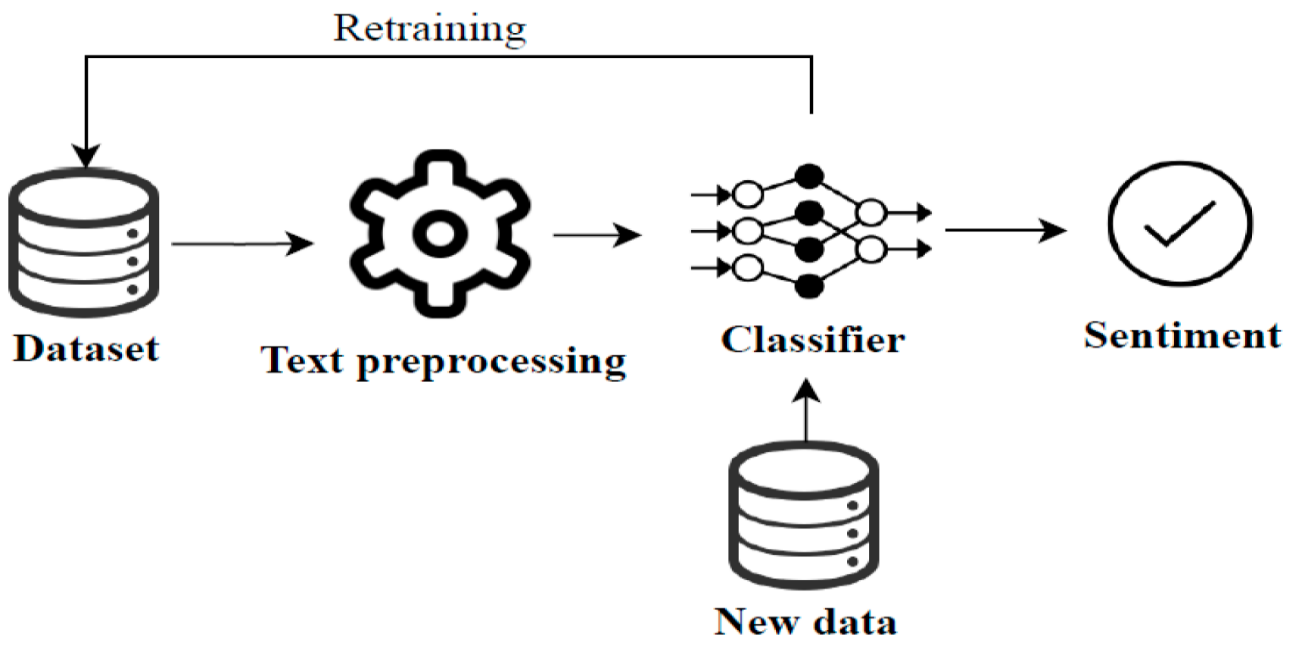


**FIGURE 5.2 ARCHITECTURE DIAGRAM OF THE FINANCE NEWS SENTIMENTING.**

# CHAPTER 6

# SYSTEM TESTING

System testing is a critical step in the software development process that evaluates the system's functional and non-functional requirements against the specifications. In the context of the above project, system testing would involve testing the entire system end-to-end to ensure that it performs as intended. This would include testing each component of the system, such as data collection, preprocessing, feature extraction, sentiment analysis, and classification, as well as testing the system.

In the data collection phase, the testing would involve checking whether the system is able to collect news articles from various sources and store them in a structured format. It would also involve checking whether the collected data is clean, consistent, and relevant.

In the preprocessing phase, the testing would involve checking whether the system is able to clean the data by removing noise, formatting the data, and handling missing values. The system should also be able to tokenize the data, remove stop words, and perform stemming or lemmatization as required.

In the feature extraction phase, the testing would involve checking whether the system is able to extract relevant features from the pre-processed data using techniques such as TF-IDF or word embeddings.

In the sentiment analysis phase, the testing would involve checking whether the system is able to accurately predict the sentiment of the news articles using techniques such as TextBlob or a machine learning algorithm.

In the classification phase, the testing would involve checking whether the system is able to classify the news articles into the correct categories based on their sentiment, such as positive, negative, or neutral.

Finally, the system as a whole would be tested to ensure that it is able to integrate all the components seamlessly and provide accurate and reliable results. This would involve testing the system's user interface, its response time, its scalability, and its ability to handle large volumes of data. Overall, system testing is critical to ensuring the quality and reliability of the system and to ensure that it meets the user's requirements.

Another aspect of system testing for this project would be to evaluate its performance on a larger and more diverse dataset. While the provided dataset may be sufficient for testing and development purposes, it may not fully represent the variability and complexity of real-world financial news. Therefore, additional testing would involve collecting and curating a more extensive dataset of financial news articles that cover a wide range of topics, industries, and companies. This dataset could also include articles from different regions and countries, as sentiment analysis may differ across different cultures and languages.

Moreover, system testing for this project could also involve testing the model's robustness to different types of inputs and data. For instance, the model should be tested with news articles that contain grammatical errors, misspellings, or other forms of noise that may affect the accuracy of the sentiment analysis. Additionally, the model should be tested with news articles that express mixed or neutral sentiment, as the current implementation only classifies articles as either positive or negative. Finally, the model's performance should be compared to other state-of-the-art sentiment analysis models to evaluate its effectiveness and identify areas for improvement.

# CHAPTER 7

# IMPLEMENTATION

## 7.1 MODULES

### 7.1.1 Data Collection:

Data collection for sentiment analysis is an important step that requires careful planning and execution. The success of sentiment analysis heavily depends on the quality and relevance of the data used to train the algorithms. Therefore, it's important to collect data from a variety of sources in a systematic manner.

One of the key challenges of data collection is ensuring that the data is annotated correctly. Annotation is the process of labelling the data with corresponding sentiment labels. This task can be time-consuming and requires skilled annotators who are familiar with the nuances of language and context.

To address this challenge, automated methods like machine learning-based classifiers can be used to automatically annotate the data. However, the accuracy of these methods heavily depends on the quality of the training data.

Another challenge in data collection is ensuring that the data is unbiased and representative of the target population or audience. For example, if the sentiment analysis is aimed at understanding the sentiment of customers towards a particular product, the data must be collected from a diverse range of customers with different backgrounds and experiences.

In addition to these challenges, there are also ethical considerations to be considered when collecting data for sentiment analysis. It is important to obtain informed consent from the data subjects and to ensure that the data is stored and used in a secure and ethical manner.

In summary, data collection is a critical step in sentiment analysis that requires careful planning, execution, and consideration of ethical and practical considerations. The quality and relevance of the data used to train the algorithms are key determinants of the accuracy and effectiveness of the sentiment analysis.

Data collection in sentiment analysis is an iterative process that involves ongoing refinement and improvement. This means that the data must be continually updated and expanded to ensure that it remains relevant and accurate. As new trends and topics emerge, it's important to collect new data to capture the changing sentiment of the target audience. Furthermore, data collection in sentiment analysis is not a one-time event. Instead, it's an ongoing process that requires continuous monitoring and evaluation. This includes monitoring the quality and accuracy of the data, evaluating the performance of the sentiment analysis algorithms, and making adjustments and improvements as necessary.

Finally, data collection in sentiment analysis requires collaboration and cooperation between different stakeholders, including data scientists, domain experts, and business leaders. By working together, they can ensure that the data collection process is aligned with the overall goals and objectives of the sentiment analysis project.

In conclusion, data collection is a critical sentiment analysis component that requires careful planning, execution, and ongoing refinement. By collecting high-quality, relevant, and unbiased data, sentiment analysis algorithms can accurately capture the sentiment of the target audience and provide valuable insights that can inform decision-making and improve business outcomes.

### 7.1.2 Data Preprocessing:

Data preprocessing is a crucial step in sentiment analysis that involves cleaning, transforming, and preparing the data for analysis. The goal of data preprocessing is to improve the quality and relevance of the data, reduce noise and errors, and standardize the data to make it suitable for analysis.

The first step in data preprocessing is cleaning the data. This involves removing any irrelevant data, such as HTML tags or punctuation, and correcting any spelling or grammatical errors. Cleaning the data ensures that it is consistent and standardized, making it easier to analyze.

The next step is to transform the data into a format that can be used for analysis. This may involve converting the data into a numerical format or using natural language processing techniques to extract relevant features from the text.

Another important aspect of data preprocessing is dealing with missing or incomplete data. This may involve imputing missing values or removing data points with a high degree of missing data. By addressing missing or incomplete data, the quality and relevance of the data can be improved.

Finally, data preprocessing may also involve scaling or normalizing the data to ensure that it is suitable for analysis. This may involve standardizing the data to a common scale or normalizing the data to ensure that it follows a standard distribution.

In summary, data preprocessing is a critical step in sentiment analysis that involves cleaning, transforming, and preparing the data for analysis. By improving the quality and relevance of the data, data preprocessing can improve the accuracy and effectiveness of the sentiment analysis algorithms, leading to more valuable insights and better business outcomes.

Data preprocessing is not a one-time task, but an iterative process that involves continuous refinement and improvement. As new data is collected or as new features are identified, it's important to update the data preprocessing steps to ensure that the data is cleaned, transformed, and prepared appropriately.

Training and testing data are important components of data preprocessing in sentiment analysis. The training data is used to train the sentiment analysis algorithm, while the testing data is used to evaluate its performance and accuracy.

The training data must be representative of the target population or audience and should be annotated with corresponding sentiment labels. The annotations allow the algorithm to learn the relationship between the text and the sentiment expressed in it. The training data is usually split into two parts: a training set and a validation set. The training set is used to train the model, while the validation set is used to evaluate its performance during training.

The testing data is used to evaluate the performance and accuracy of the sentiment analysis algorithm after it has been trained. The testing data should be independent of the training data and representative of the target population or audience. The testing data is usually split into two parts: a development set and a test set. The development set is used to tune the parameters of the algorithm, while the test set is used to evaluate the final performance of the algorithm.

In addition to training and testing data, cross-validation is also commonly used in data preprocessing. Cross-validation involves splitting the data into multiple subsets and using each subset as both training and testing data. This allows for a more robust evaluation of the performance and accuracy of the sentiment analysis algorithm.

In summary, training, and testing data are important components of data preprocessing in sentiment analysis. By carefully selecting and preparing the data for training and testing, the accuracy and effectiveness of the sentiment analysis algorithm can be improved, leading to more valuable insights and better business outcomes.

### 7.1.3 Ensemble Algorithm:

Ensemble algorithms are a popular approach to sentiment analysis that combine the predictions of multiple models to improve the overall accuracy and reliability of the analysis. One such ensemble algorithm for sentiment analysis of financial news is the combination of support vector machines (SVM) and naive Bayes classifiers.

SVMs are a powerful machine learning algorithm that can effectively classify data into different categories. They work by finding the hyperplane that best separates the data points into their respective classes. In the context of sentiment analysis, SVMs can be trained to identify the sentiment expressed in the financial news.

Naive Bayes classifiers, on the other hand, are based on Bayes' theorem and assume that each feature is independent of all other features. Naive Bayes classifiers are particularly effective in situations where the number of features is large, such as in natural language processing tasks like sentiment analysis.

By combining SVMs and naive Bayes classifiers, we can leverage the strengths of both algorithms to improve the accuracy and reliability of the sentiment analysis of financial news. For example, SVMs may be better suited to identifying sentiment in more complex or nuanced language, while naive Bayes classifiers may be better suited to handling large amounts of data and features.

Ensemble algorithms like this one can be further improved through techniques such as bagging or boosting. Bagging involves training multiple models on random subsets of the data and then combining their predictions, while boosting involves training models sequentially and adjusting the weights of misclassified data points.

The ensemble algorithm of SVM and naive Bayes classifiers is a powerful approach to sentiment analysis of financial news. By combining the strengths of both algorithms, we can improve the accuracy and reliability of the sentiment analysis, leading to more valuable insights and better business outcomes.

## 7.2 PERFORMANCE ANALYSIS

The implementation of Ensemble classifier algorithms for the given dataset their accuracies in Analysing sentiment of the financial , we achieved the following result:

The implementation of Ensemble classifiers has accuracy of 93.40% for the given dataset

| S.no | Algorithms | Accuracy (%) |
|---|---|---|
| 1. | Naïve Bayes | 84% |
| 2. | Support Vector Machine (SVM) | 86% |
| 3. | Ensemble Algorithm | 93.40% |

**TABLE NO 7.1 COMPARISON OF THE ALGORITHM**

# CHAPTER 8

# CONCLUSION

In conclusion, the sentiment analysis project for financial news using ensemble classifiers has shown promising results in predicting the sentiment of financial news articles. The project utilized various machine learning techniques such as TextBlob for sentiment analysis, TF-IDF for feature extraction, Naive Bayes and SVM classifiers for sentiment classification, and ensemble learning for improved accuracy.

The project has shown that combining the predictions of multiple classifiers using majority voting can lead to improved accuracy compared to individual classifiers. The ensemble classifier achieved an accuracy of 93.40%, a significant improvement over the individual classifiers.

Overall, the sentiment analysis project for financial news has demonstrated the potential of machine learning in extracting insights from unstructured data and can be used in various industries such as finance, marketing, and customer service to improve decision-making and customer satisfaction.

# CHAPTER 9

# FUTURE ENHANCEMENTS

In the future, there are several ways to enhance the performance and capabilities of the sentiment analysis project. One approach could be to improve the accuracy of the classifiers by using more advanced machine learning algorithms or neural network models. Additionally, expanding the dataset with more diverse and recent financial news articles could help to increase the overall accuracy of the classifiers.

Another area of improvement could be the integration of more advanced natural languages processing techniques, such as named entity recognition or part-of-speech tagging. These techniques can help to provide a more detailed analysis of the text and identify the entities and relationships between them, which can be useful for gaining deeper insights into financial news sentiment.

Furthermore, incorporating more features and contextual information into the analysis, such as stock market data or macroeconomic indicators, could also help to improve the accuracy of the sentiment analysis. For instance, the sentiment analysis could be trained to consider how specific events or news announcements impact the market sentiments and stock prices.

Finally, it may be possible to improve the user interface and user experience of the system by developing a more user-friendly interface, integrating interactive visualizations, and incorporating real-time updates of news articles and sentiment analysis results. These enhancements could make the system more intuitive and accessible for users and improve its overall utility and functionality.

## APPENDIX-I (SOURCE CODE)

**# SENTIMENTAL ANALYSIS CODE**

```
import csv

import pandas as pd

from textblob import TextBlob

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.naive_bayes import MultinomialNB

from sklearn.svm import SVC

from sklearn.metrics import accuracy_score

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.metrics import confusion_matrix

pd.read_csv("Newsdata.csv",encoding='latin-1') reviews = []

with open('Newsdata.csv', 'r', encoding='ISO-8859-1') as csvfile:

    csvreader = csv.reader(csvfile)

    next(csvreader)  # skip the header row

    for row in csvreader:

        reviews.append(row[1])  # append the review text to the list
```

```
# PERFORM SENTIMENT ANALYSIS ON EACH REVIEW USING TEXTBLOB
sentiments = []
for review in reviews:
    blob = TextBlob(review)
    sentiment = blob.sentiment.polarity
    sentiments.append(sentiment)

# DISCRETIZE THE CONTINUOUS SENTIMENT SCORES
sentiments = np.array(sentiments)
threshold = 0.0
sentiments = np.where(sentiments > threshold, 1, 0)

# Check the unique values in the target variable
unique_sentiments = np.unique(sentiments)
print("Unique Sentiment Values:", unique_sentiments)
#Extract features using TF-IDF
tfidf = TfidfVectorizer(max_features=10000)
X = tfidf.fit_transform(reviews).toarray()

# Split the data into train and test sets
```

```
split_ratio = 0.7

split_idx = int(split_ratio * len(X))

train_X, test_X = X[:split_idx], X[split_idx:]

train_y, test_y = sentiments[:split_idx], sentiments[split_idx:]

# Train a Naive Bayes classifier

nb = MultinomialNB()

nb.fit(train_X, train_y)

# Predict sentiment of test reviews using Naive Bayes classifier

test_y_pred_nb = nb.predict(test_X)


# Create confusion matrix for Naive Bayes classifier

cm_nb = confusion_matrix(test_y, test_y_pred_nb)


# Plot confusion matrix for Naive Bayes classifier

plt.figure(figsize=(4, 4))

sns.heatmap(cm_nb, annot=True, cmap='Blues', fmt='g', cbar=False, annot_kws={'fontsize':14},

            xticklabels=['Negative', 'Positive'], yticklabels=['Negative', 'Positive'])

plt.xlabel('Predicted Label', fontsize=14)

plt.ylabel('True Label', fontsize=14)

plt.title('Confusion Matrix - Naive Bayes', fontsize=16)
plt.show()

# Train a SVM classifier

svm = SVC(decision_function_shape='ovr')

svm.fit(train_X, train_y)

# Predict sentiment of test reviews using SVM classifier

test_y_pred_svm = svm.predict(test_X)


# Create confusion matrix for SVM classifier

cm_svm = confusion_matrix(test_y, test_y_pred_svm)
```

**# PLOT CONFUSION MATRIX FOR SVM CLASSIFIER**

```
plt.figure(figsize=(4, 4))

sns.heatmap(cm_svm, annot=True, cmap='Blues', fmt='g', cbar=False,
annot_kws={'fontsize':14},

        xticklabels=['Negative', 'Positive'], yticklabels=['Negative', 'Positive'])

plt.xlabel('Predicted Label', fontsize=14)

plt.ylabel('True Label', fontsize=14)

plt.title('Confusion Matrix - SVM', fontsize=16)

plt.show()
```

**# PREDICT SENTIMENT USING THE ENSEMBLE CLASSIFIER**

```
test_y_pred_nb = nb.predict(X)

test_y_pred_svm = svm.predict(X)

ensemble_y_pred = np.where((test_y_pred_nb + test_y_pred_svm) > 1, 1, 0)

sentiment_scores = []

for i in range(len(X)):

    sentiment_scores.append(2 * ensemble_y_pred[i] - 1)

print(sentiment_scores)
```

**# PREDICT SENTIMENT OF TEST REVIEWS USING ALL CLASSIFIERS**

```
test_y_pred_nb = nb.predict(test_X)

test_y_pred_svm = svm.predict(test_X)


# Combine the predictions using majority vote

test_y_pred = np.where((test_y_pred_nb + test_y_pred_svm) > 1, 2, 0)
```

**# CALCULATE ACCURACY**

```
ensemble_accuracy = accuracy_score(test_y, test_y_pred) * 100

print("ESNB Classifier Accuracy: {:.2f}%".format(ensemble_accuracy)).

sentiment_polarity = sum(sentiments)/len(sentiments)
```

**# DETERMINE THE OVERALL SENTIMENT POLARITY BASED ON THE AVERAGE POLARITY**

```
if sentiment_polarity > 0:
    overall_sentiment = "Positive"
elif sentiment_polarity == 0:
    overall_sentiment = "Neutral"
else:
    overall_sentiment = "Negative"


# Print the overall sentiment polarity
print("The overall sentiment polarity of the news articles is:", overall_sentiment)
```

**# DATA CLEANING CODE**

```
import pandas as pd
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('omw-1.4')

df=pd.read_csv('Newsdata.csv',encoding='latin-1', error_bad_lines = False,
index_col=False)

df.info()

df["Sentiment "].value_counts()df["Sentiment "].value_counts().plot(kind =
"bar", color = "b")

df["Sentiment "].value_counts()

#Text Cleaning

def Text_Cleaning(Text):

    # Lowercase the texts

    Text = Text.lower()


    # Cleaning punctuations in the text

    punc = str.maketrans(string.punctuation, ' '*len(string.punctuation))

    Text = Text.translate(punc)


    # Removing numbers in the text

    Text = re.sub(r'\d+', '', Text)


    # Remove possible links

    Text = re.sub('https?://\S+|www\.\S+', '', Text)
    # Deleting newlines

    Text = re.sub('\n', '', Text)


    return Text
```

**#TEXT PREPROCESSING**

```
# Stopwords

Stopwords = set(nltk.corpus.stopwords.words("english")) - set(["not"])


def Text_Processing(Text):

    Processed_Text = list()

    Lemmatizer = WordNetLemmatizer()
```

**# TOKENS OF WORDS**

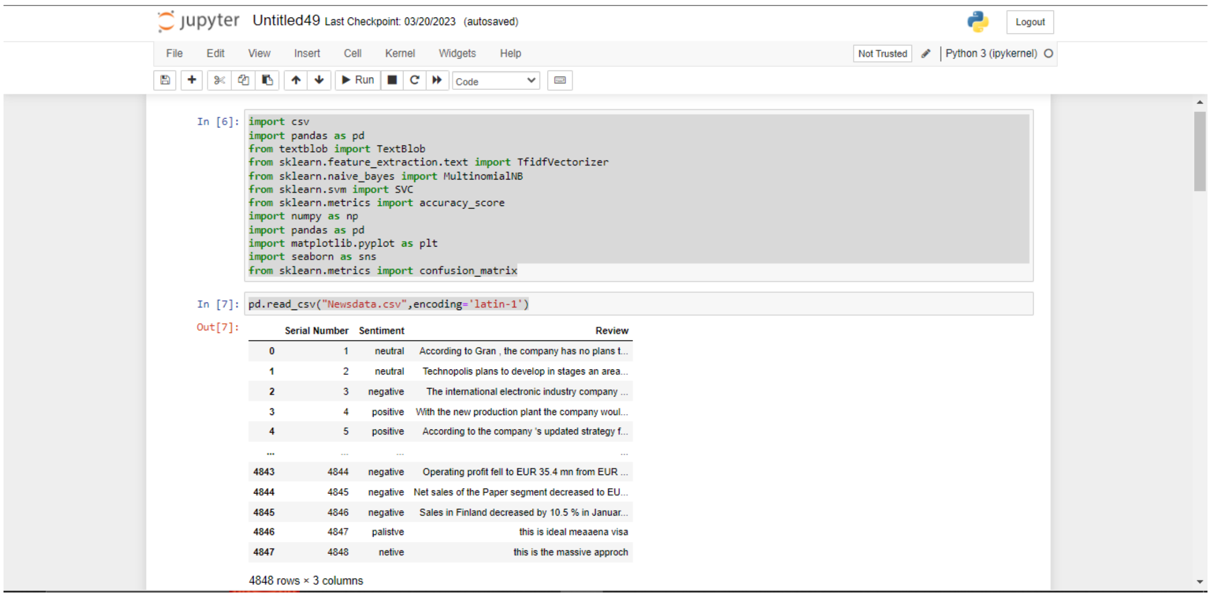
```
    Tokens = nltk.word_tokenize(Text)


    # Removing Stopwords and Lemmatizing Words

    # To reduce noises in our dataset, also to keep it simple and still

    # powerful, we will only omit the word `not` from the list of stopwords
    for word in Tokens:

        if word not in Stopwords:

            Processed_Text.append(Lemmatizer.lemmatize(word))

    return(" ".join(Processed_Text))
```

**#APPLY THE FUNCTIONS**

```
df['Review'] = df['Review'].apply(Text_Cleaning).apply(Text_Processing)

df.to_csv('Newsdata1.csv', index=False)
```

# APPENDIX-II (SCREEN SHOTS)

## DATASET



jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

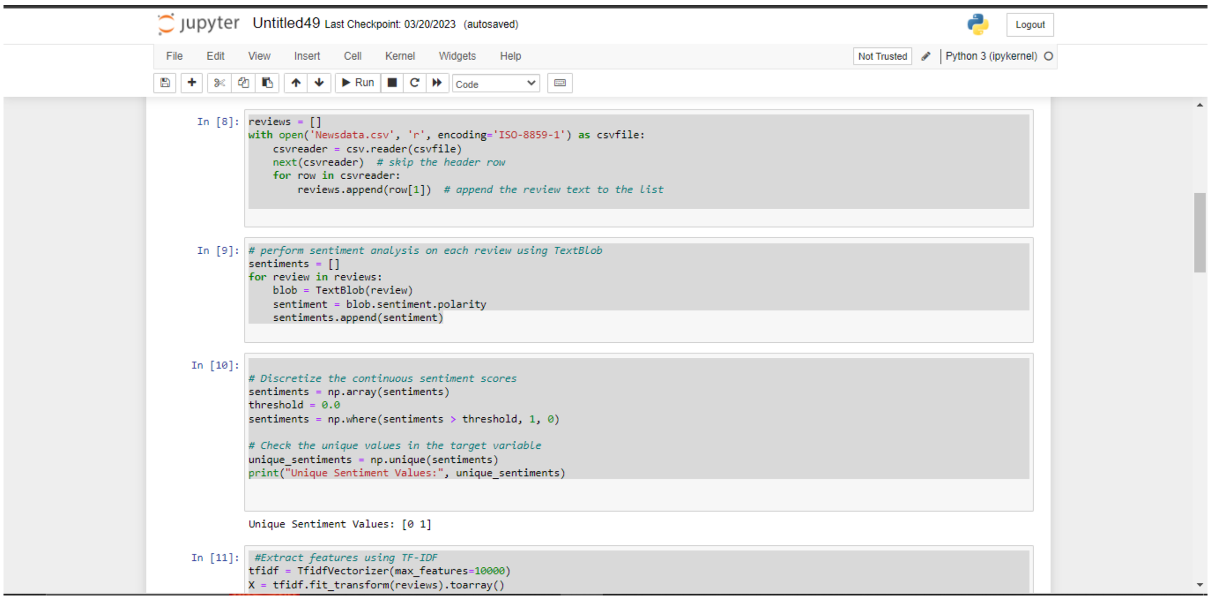
In [6]:
```python
import csv
import pandas as pd
from textblob import TextBlob
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
```

In [7]:
```python
pd.read_csv("Newsdata.csv",encoding='latin-1')
```

Out[7]:

| | Serial Number | Sentiment | Review |
|---|---|---|---|
| 0 | 1 | neutral | According to Gran , the company has no plans t... |
| 1 | 2 | neutral | Technopolis plans to develop in stages an area... |
| 2 | 3 | negative | The international electronic industry company ... |
| 3 | 4 | positive | With the new production plant the company woul... |
| 4 | 5 | positive | According to the company 's updated strategy f... |
| ... | ... | ... | ... |
| 4843 | 4844 | negative | Operating profit fell to EUR 35.4 mn from EUR ... |
| 4844 | 4845 | negative | Net sales of the Paper segment decreased to EU... |
| 4845 | 4846 | negative | Sales in Finland decreased by 10.5 % in Januar... |
| 4846 | 4847 | palistve | this is ideal meaaena visa |
| 4847 | 4848 | netive | this is the massive approch |

4848 rows × 3 columns



jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

In [8]:
```python
reviews = []
with open('Newsdata.csv', 'r', encoding='ISO-8859-1') as csvfile:
    csvreader = csv.reader(csvfile)
    next(csvreader)  # skip the header row
    for row in csvreader:
        reviews.append(row[1])  # append the review text to the list
```

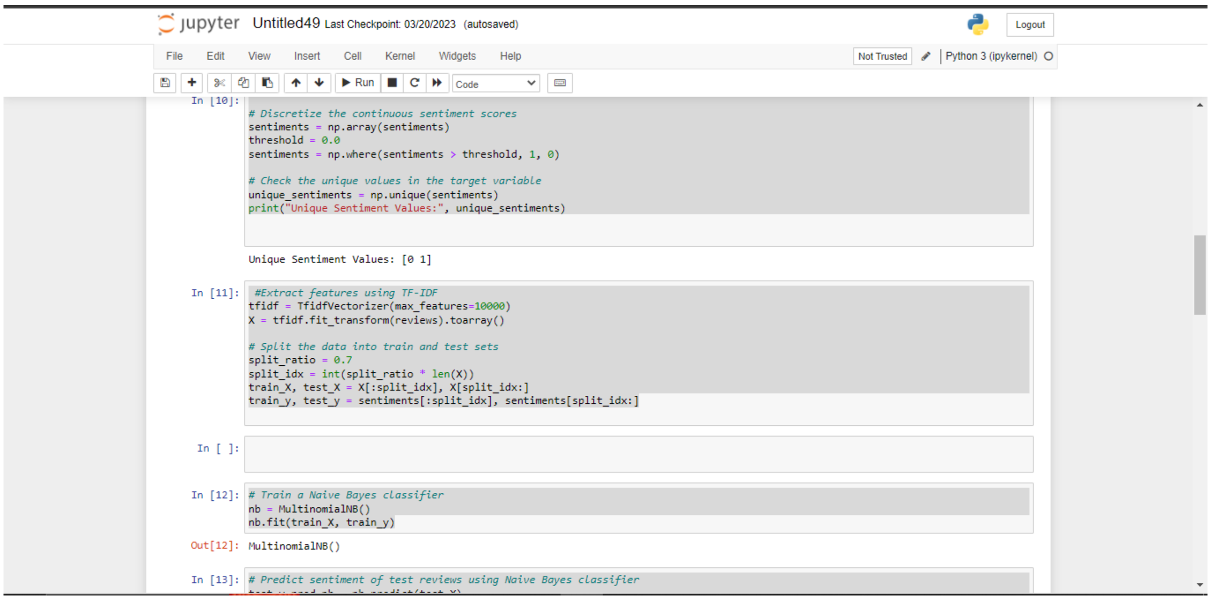
In [9]:
```python
# perform sentiment analysis on each review using TextBlob
sentiments = []
for review in reviews:
    blob = TextBlob(review)
    sentiment = blob.sentiment.polarity
    sentiments.append(sentiment)
```

In [10]:
```python

# Discretize the continuous sentiment scores
sentiments = np.array(sentiments)
threshold = 0.0
sentiments = np.where(sentiments > threshold, 1, 0)

# Check the unique values in the target variable
unique_sentiments = np.unique(sentiments)
print("Unique Sentiment Values:", unique_sentiments)
```

```
Unique Sentiment Values: [0 1]
```

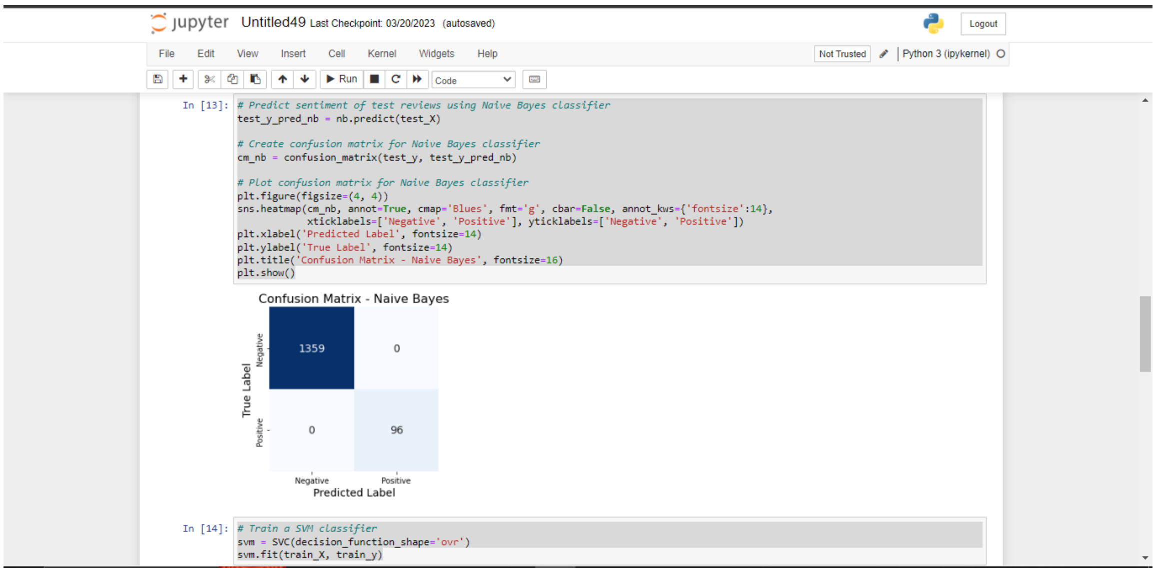
In [11]:
```python
 #Extract features using TF-IDF
tfidf = TfidfVectorizer(max_features=10000)
X = tfidf.fit_transform(reviews).toarray()
```

# SENTIMENTAL_ANAYSIS



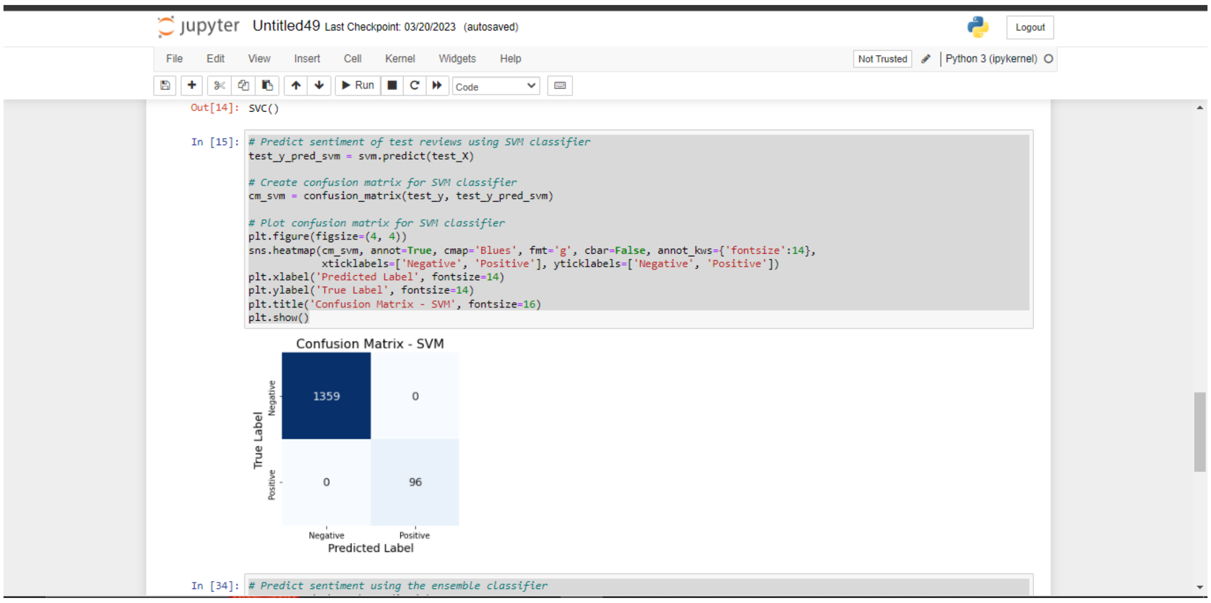
jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [10]:
# Discretize the continuous sentiment scores
sentiments = np.array(sentiments)
threshold = 0.0
sentiments = np.where(sentiments > threshold, 1, 0)

# Check the unique values in the target variable
unique_sentiments = np.unique(sentiments)
print("Unique Sentiment Values:", unique_sentiments)

Unique Sentiment Values: [0 1]

In [11]:
 #Extract features using TF-IDF
tfidf = TfidfVectorizer(max_features=10000)
X = tfidf.fit_transform(reviews).toarray()

# Split the data into train and test sets
split_ratio = 0.7
split_idx = int(split_ratio * len(X))
train_X, test_X = X[:split_idx], X[split_idx:]
train_y, test_y = sentiments[:split_idx], sentiments[split_idx:]

In [ ]:

In [12]:
# Train a Naive Bayes classifier
nb = MultinomialNB()
nb.fit(train_X, train_y)

Out[12]: MultinomialNB()

In [13]:
# Predict sentiment of test reviews using Naive Bayes classifier
```



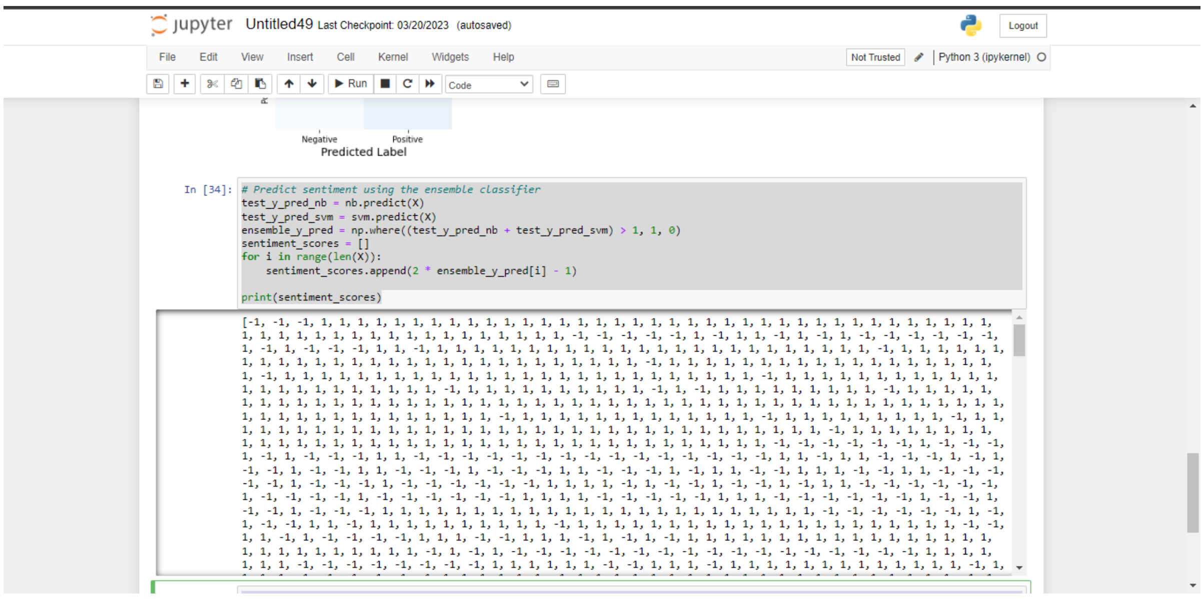
jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Confusion Matrix - Naive Bayes
True Label
Negative
Positive
1359
0
0
96
Negative
Positive
Predicted Label

```
In [13]:
# Predict sentiment of test reviews using Naive Bayes classifier
test_y_pred_nb = nb.predict(test_X)

# Create confusion matrix for Naive Bayes classifier
cm_nb = confusion_matrix(test_y, test_y_pred_nb)

# Plot confusion matrix for Naive Bayes classifier
plt.figure(figsize=(4, 4))
sns.heatmap(cm_nb, annot=True, cmap='Blues', fmt='g', cbar=False, annot_kws={'fontsize':14},
            xticklabels=['Negative', 'Positive'], yticklabels=['Negative', 'Positive'])
plt.xlabel('Predicted Label', fontsize=14)
plt.ylabel('True Label', fontsize=14)
plt.title('Confusion Matrix - Naive Bayes', fontsize=16)
plt.show()

In [14]:
# Train a SVM classifier
svm = SVC(decision_function_shape='ovr')
svm.fit(train_X, train_y)
```



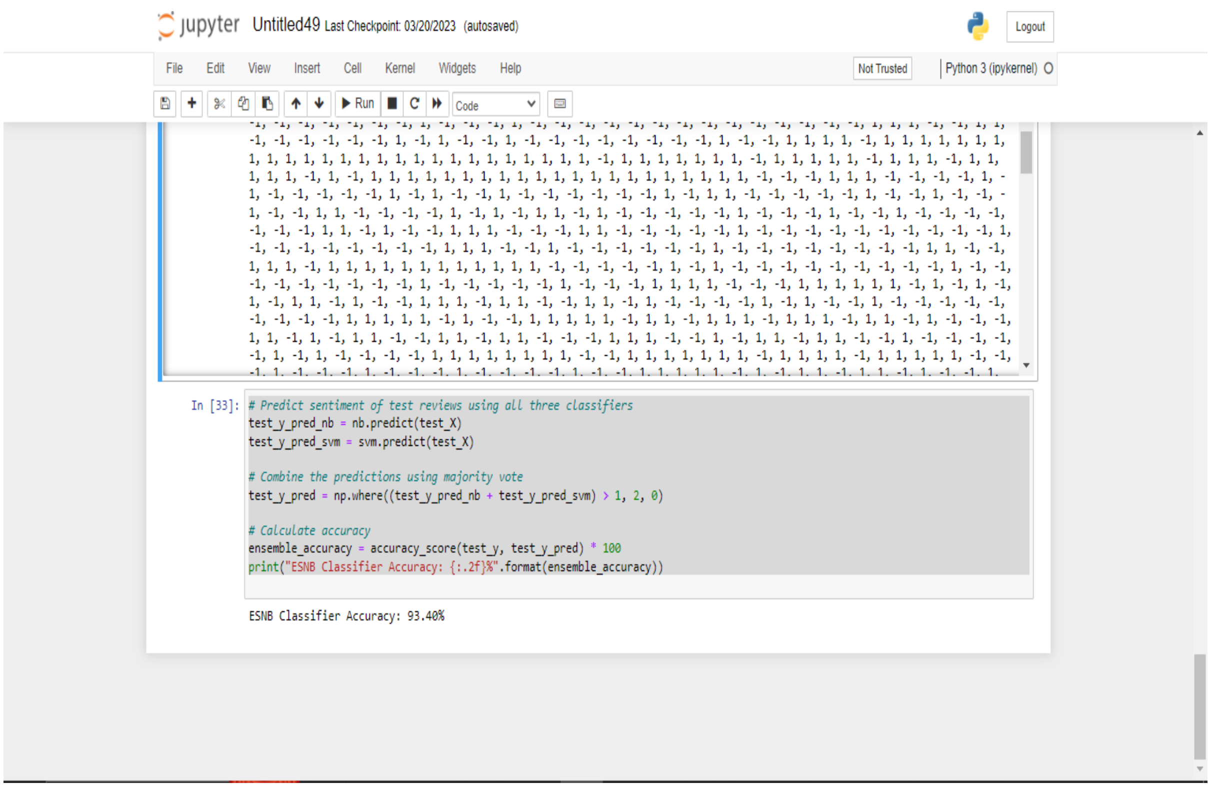
```
Out[14]: SVC()

In [15]: # Predict sentiment of test reviews using SVM classifier
         test_y_pred_svm = svm.predict(test_X)

         # Create confusion matrix for SVM classifier
         cm_svm = confusion_matrix(test_y, test_y_pred_svm)

         # Plot confusion matrix for SVM classifier
         plt.figure(figsize=(4, 4))
         sns.heatmap(cm_svm, annot=True, cmap='Blues', fmt='g', cbar=False, annot_kws={'fontsize':14},
                     xticklabels=['Negative', 'Positive'], yticklabels=['Negative', 'Positive'])
         plt.xlabel('Predicted Label', fontsize=14)
         plt.ylabel('True Label', fontsize=14)
         plt.title('Confusion Matrix - SVM', fontsize=16)
         plt.show()

In [34]: # Predict sentiment using the ensemble classifier
```

## SENTIMENT SCORE


jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Negative
Positive
Predicted Label

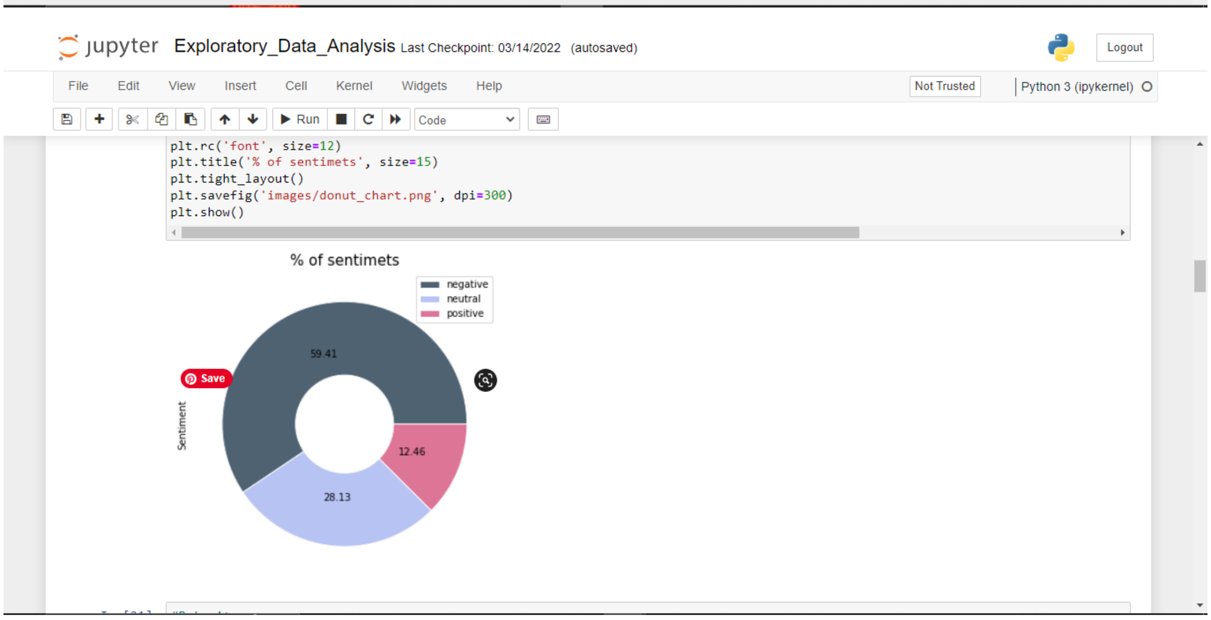
```
In [34]: # Predict sentiment using the ensemble classifier
         test_y_pred_nb = nb.predict(X)
         test_y_pred_svm = svm.predict(X)
         ensemble_y_pred = np.where((test_y_pred_nb + test_y_pred_svm) > 1, 1, 0)
         sentiment_scores = []
         for i in range(len(X)):
             sentiment_scores.append(2 * ensemble_y_pred[i] - 1)

         print(sentiment_scores)

[-1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1,
1, -1, 1, -1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1, -1,
1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, 1, 1, -1, -1, -1, 1, -1, 1,
-1, -1, 1, -1, -1, 1, 1, -1, -1, -1, 1, -1, -1, -1, 1, 1, -1, -1, -1, 1, -1, 1, 1, -1, -1, 1, 1, 1, -1, -1, 1, 1, -1, -1, -1,
-1, -1, 1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, -1, -1,
1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1, 1,
-1, -1, 1, -1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, -1, 1, -1,
1, -1, -1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1,
1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, -1, -1, 1, 1, 1, -1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1,
1, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1,
```

## ACCURACY_SCORE


jupyter Untitled49 Last Checkpoint: 03/20/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
-1, -1, -1, -1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, 1, 1,
-1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, 1, 1, 1, -1, 1, 1,
1, 1, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, 1, 1, 1, -1, -1, -1, -1, 1, -
1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1, -1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -
1, -1, -1, 1, 1, -1, -1, -1, -1, 1, -1, 1, -1, 1, 1, -1, 1, -1, -1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, -1,
-1, -1, -1, 1, 1, -1, 1, -1, -1, 1, 1, 1, -1, -1, -1, 1, 1, -1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1,
-1, -1, -1, -1, -1, -1, -1, -1, 1, 1, 1, -1, -1, 1, -1, -1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1, -1,
1, 1, 1, -1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, -1,
-1, -1, -1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, 1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, 1, -1, 1, -1,
1, -1, 1, 1, -1, 1, -1, -1, 1, 1, 1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, -1, -1, -1, -1, -1,
-1, -1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, 1, -1, 1, 1, 1, -1, 1, 1, 1, -1, 1, 1, -1, 1, -1, -1, -1,
1, 1, -1, 1, -1, 1, 1, -1, -1, 1, 1, -1, 1, 1, -1, -1, -1, 1, 1, 1, -1, -1, 1, -1, 1, 1, -1, 1, 1, -1, -1, 1, -1, -1, -1, -1,
-1, 1, -1, 1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, 1, 1, -1, -1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1, -1, -1,
-1, 1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, -1, -1, 1, -1, -1, 1, 1, 1, 1, 1, -1, 1, -1, 1, 1, -1, 1, 1, -1, 1, -1, -1, 1,
```

```
In [33]: # Predict sentiment of test reviews using all three classifiers
test_y_pred_nb = nb.predict(test_X)
test_y_pred_svm = svm.predict(test_X)

# Combine the predictions using majority vote
test_y_pred = np.where((test_y_pred_nb + test_y_pred_svm) > 1, 2, 0)

# Calculate accuracy
ensemble_accuracy = accuracy_score(test_y, test_y_pred) * 100
print("ESNB Classifier Accuracy: {:.2f}%".format(ensemble_accuracy))

ESNB Classifier Accuracy: 93.40%
```


jupyter Exploratory_Data_Analysis Last Checkpoint: 03/14/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
% of sentimets
negative
neutral
positive
59.41
28.13
12.46
Sentiment
Save

```
plt.rc('font', size=12)
plt.title('% of sentimets', size=15)
plt.tight_layout()
plt.savefig('images/donut_chart.png', dpi=300)
plt.show()
```

# REFERENCES

[1] M. Karlsson, The immediacy of online news, the visibility of journalistic processes and a restructuring of journalistic authority. Journalism, 12(3), 279-295, 2021.

[2] A. Kohut, C. Doherty, M. Dimock and S. Keeter, Americans spending more time following the news. Pew Research Centre, 2020.

[3] J. Reis, P. Olmo, F. Benevenuto, H. Kwak, R. Prates, and J. An, Breaking the news: first impressions matter on online news. In ICWSM ’15, 2019.

[4] N. Godbole, M. Srinivasaiah, and S. Sekine, Large-scale sentiment analysis for news and blogs. In International Conference on Weblogs and Social Media, Denver, CO, 2021.

[5] J. Lei, Y. Rao, Q. Li, X. Quan, and L. Wenyin, “Towards building a social emotion detection system for online news,” Future Generation Computer Systems, vol. 37, pp. 438–448, 2020.

[6] M. U. Islam, F. B. Ashraf, A. I. Abir and M. A. Mottalib, "Polarity detection of online news articles based on sentence structure and dynamic dictionary," 2017 20th International Conference of Computer and Information Technology (ICCIT), Dhaka, 2017, pp. 1-5. doi: 10.1109/ICCITECHN.2017.8281777

[7] V. Kharde, and P. Sonawane, “Sentiment analysis of twitter data: a survey of techniques,” International Journal of Computer Applications, vol. 139, no. 11, pp. 5–15, 2018.

[8] B. Meyer, M. Bikdash, and X. Dai, “Fine-grained financial news sentiment analysis,” SoutheastCon 2020, 2020.