TECHNO INTERNATIONAL NEWTOWN

Newtown, Megacity, Kolkata -700156.

Department of Information Technology

Submission for the partial fulfillment of B.Tech Final Year Project- PROJ CS881

Short Term Stock Market Prediction using Machine Learning

Prepared by

Supriyo Mandal(Roll No :- 18701620035)

Shovan Maji(Roll No :- 18700220041)

Maitryee Dey(Roll No :- 18700221136)

Sachin Paul(Roll No :- 18700220047)

Under The Guidance of

Prof. Dr. Nirmalya Sundar Maiti(Assistant Professor)

For

Batch:- 2020-2024 Semester : 8 th Year : Jan 2024 – June 2024

Stream:- Information Technology Year of Study: 4th

Affiliated to

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WEST BENGAL

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, WESTBENGAL

(FORMERLY KNOWN AS WEST BENGAL UNIVERSITY OF TECHNOLOGY)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Prof. (Dr.) Nirmalya Sundar Maiti(Asso. Prof.) of the department of Information Technology, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he / she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the cordial support they offered.

Place: Techno India College of Technology, Newtown Megacity

Date: _ _ _ _ _ _ _ _ _ _

Supriyo Mandal
(Roll No:- 18701620035)

Shovan Maji
(Roll No :- 18700220041

Maitryee Dey
(Roll No: - 18700221136)

Sachin Paul
(Roll No: - 18700220047)

Department of Information Technology,
Techno India College of Technology, Newtown Megacity
Kolkata – 700 156
West Bengal, India.

Approval

This is to certify that the project report entitled "……." prepared under my supervision by Supriyo Mandal (18701620035), Shovan Maji (18700220041), Maitryee Dey (18700221136), Sachin Paul(18700220047) be accepted in partial fulfillment for the degree of Bachelor of Technology in Information Technology which is affiliated to Maulana Abul Kalam Azad University of Technology, West Bengal (Formerly known as West Bengal University of Technology).

It is to be understood that by this approval, the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

Project Mentor:

……
Prof. (Dr.) Nirmalya Sundar Maiti
Assistant Professor
Dept. of Information Technology
Techno India College of Technology

External Examiner:

…….

…….
Prof. Sourav Mahapatra,
HOD, Dept. of Information Technology
Techno India College of Technology

Abstract

Stock market prediction is a critical yet challenging task due to the market's inherent volatility and the multitude of factors influencing stock prices. This project focuses on utilizing machine learning techniques to predict short-term stock prices, aiming to provide investors with actionable insights and enhance decision-making. The objective of the proposed work is to study and improve supervised learning algorithms to predict stock prices. To achieve this, the project investigates the optimal amount of data required to avoid overfitting and the best approach to train the supervised learning models, along with identifying and adding suitable features. Different combinations of input data sizes, such as 7, 14, 21, and 28 working days, were tested to determine the amount that yielded the highest accuracy. Additionally, various training methods, including Time Series Forecasting Approach, and the Sliding Window Approach, were evaluated to identify the most effective approach. Feature engineering played a crucial role in this project. Various technical indicators, like moving averages, relative strength index (RSI), and exponential moving average (EMA), were derived to enrich the feature set. One key finding was that volume indicators did not significantly improve forecasting performance on the training datasets used. The project involved a comprehensive approach, starting with extensive data collection from reliable sources such as Yahoo Finance, encompassing historical stock prices and relevant financial indicators. Several supervised machine learning models, including linear regression, support vector machines, K Nearest Neighbors, decision trees, and random forests, were trained and evaluated. This project showcases the potential of machine learning in stock market prediction, particularly for short-term forecasts. Following this, we create our personalized model by selecting the most suitable models with good accuracy and adding weightage to those supervised learning algorithms giving greater accuracy than all the traditional supervised learning algorithms available in the market. The insights gained from this study can aid investors in making informed decisions and managing risks more effectively. Future research could focus on predicting mid-term and long-term closing stock prices of a company and exploring advanced modeling techniques to further enhance prediction accuracy.

CONTENTS

LIST OF FIGURES

(For every figure in the document it should be numbered along with a caption/title in the document where it is placed. The figure along with its number & Caption should be centrally aligned)

LIST OF TABLES (OPTIONAL)

1. **Table Caption……Page No.**

   (For every table in the document it should be numbered along with a caption/title in the document where it is placed. The table along with its number & Caption should be centrally aligned)

OBJECTIVE

The aim of this project is to develop and implement machine learning techniques for short-term stock market prediction. By leveraging historical stock price data and relevant financial indicators, the project seeks to build predictive models that can forecast stock prices over short time horizons. The primary objective is to achieve accurate predictions that outperform traditional methods and offer actionable information for investors.

Another key aim is to explore and compare different supervised learning algorithms to determine the most effective approach for stock price prediction. By evaluating the performance of these algorithms on historical data, the project aims to identify the algorithm that yields the highest accuracy and reliability in predicting short-term stock prices.

By conducting thorough feature engineering, the project seeks to enhance the predictive power of the models and uncover insights that can assist investors in making informed decisions in the volatile stock market.

1. Introduction

1.1. Introduction to Stock Market

The stock market is a dynamic and complex financial system where investors buy and sell shares of publicly traded companies. When a company that wants to expand its business but needs more money to do so then instead of borrowing money from a bank, they sell shares of the company to investors. These shares represent ownership in the company, and investors buy them with the hope that the company will do well and the value of their shares will increase.

Term	Definition
Shares	Units of ownership in a company, bought and sold on the stock market.
Investors	Individuals or institutions that buy shares in companies.
Dividends	Payments made by companies to shareholders as a share of profits.
Volatility	The degree of variation in a stock's price over time.
Diversification	Spreading investments across different assets to reduce risk.
Market Trends	The general direction in which a market is moving over time.
Risk	The possibility of losing money on an investment.

Terminology

1.2. Introduction to stock market trend analysis

Trend is considered as direction of stock movement that is totally based on stock market ups and downs. Continues movement of stock in any direction upward or downward for specified duration or time period can be considered as trend. In stock market prediction trend analysis at current stage support a lot in future trend prediction. Trend growing analysis for continues interval of time can be considered as future grow or continues down in trending market share prices can be supportive for future predictions as down. Stock market prediction always based on big amount of historical data analysis.

1.3. Introduction to Machine Learning

Machine Learning, as defined by Tom Mitchell in 1998, is the study of algorithms that

- improve their performance P

- at some task T

- with experience E

A well-defined Machine Learning task is given by <P, T, E>. In simple terms, Machine Learning is a process where a system learns for a specific task through experience E, and its performance increases P as it gains more experience E. In recent times, there has been a significant increase in the amount of data produced by applications like Facebook, WhatsApp, and other platforms.

1.4. Introduction of Machine Learning in Stock Market Prediction

Machine Learning (ML) plays a crucial role in short-term stock market prediction, particularly for indices like the NIFTY 50. ML algorithms analyze historical stock data to identify patterns and trends, which are then used to predict future stock prices. These predictions are valuable for investors looking to make informed decisions about buying or selling stocks. One way ML helps in short-term prediction is by analyzing large volumes of historical data to identify patterns that may not be apparent to human analysts. ML algorithms can detect subtle trends and correlations in the data, which can help predict how stock prices might behave in the near future. This is particularly useful in the fast-paced stock market, where prices can change rapidly based on various factors. ML algorithms also excel at processing and analyzing large amounts of data quickly, which is essential for short-term stock market prediction. They can analyze complex datasets and generate predictions in real-time, allowing investors to make timely decisions. Additionally, ML algorithms can adapt to changing market conditions, making them well-suited for predicting stock prices in dynamic markets like the NIFTY 50.

Aspect	Description
ML's Role in Prediction	ML algorithms analyze historical stock data to identify patterns and trends, used to predict future stock prices.
Importance for Investors	Predictions are valuable for investors making informed decisions about buying or selling stocks.
Analyzing Historical Data	ML analyzes large volumes of historical data, identifying patterns not apparent to humans, detecting subtle trends and correlations to predict future stock behavior.
Processing Large Data Sets	ML excels at processing and analyzing large amounts of data quickly, essential for short-term stock market prediction, allowing real-time analysis and timely decision-making.
Adaptability to Market Changes	ML algorithms can adapt to changing market conditions, making them suitable for predicting stock prices in dynamic markets like the NIFTY 50.

ML Roles

1.5. Introduction to Stock Market Prediction

Everyone want to be rich in his life with low efforts and great advantages. Similarly, we want to look in our future with inner most desire as we do not want to take risks or we want to decrease risk factor. Stock market is a place where selling and purchasing can provide future aims of life. If stock market trend predicted then we can avoid wastage of money. Stock Market Prediction is a process of predicting future on the base of past data. Prediction decreases the risk level to investors and increases the confidence level for investment. If they predicted goals before reach then they can avoid loss of money. All these consideration work as Stock Market Prediction. On the basis of historical data trends, we guess future trend that is called Stock Market Prediction.

2. Problem Statement

The problem statement for the project "Short Term Stock Market Prediction for Nifty 50" is to develop a machine learning model that can accurately predict the future closing stock price of companies in the Nifty 50 index. The aim is to utilize historical stock market data to train the model, which will then be used to make predictions for the next trading day. The prediction of stock prices is a challenging task due to the complex and dynamic nature of the stock market, which is influenced by various factors such as market trends, economic indicators. By addressing

these challenges, the project aims to develop a robust machine learning model that can help investors make informed decisions in the stock market, ultimately leading to better investment outcomes.

The problem statement for this project is to predict future stock market movements with high accuracy using machine learning (ML) techniques. The accurate prediction of stock prices is crucial for investors to make informed decisions and maximize profits. By analyzing historical stock market data and implementing various ML, including Linear Regression, Support Vector Machine (SVM), K Nearest Neighbor (KNN), Decision Tree, Random Forest, and Artificial Neural Networks (ANN), we aim to identify the algorithms that provide the best accuracy for stock market prediction.

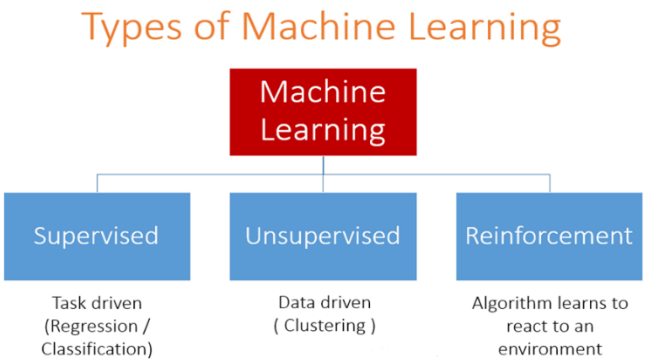
To achieve this, we will collect and analyze datasets from Yahoo Finance, focusing on companies listed in the Nifty 50 index. The datasets will cover a few weeks of historical stock price data, which will be used to train and test the ML and DL models. The objective is to train the models with the right amount of data to avoid overfitting, select ML techniques that provide the best accuracy, and use the right approach for training and predicting future stock prices.

Once we have identified the ML models with the best accuracy, we will further enhance their performance by adding weightage to the models that demonstrate higher accuracy. This weighted approach aims to create a more robust and accurate prediction model for stock market movements. The ultimate goal of this project is to provide investors with a reliable tool for predicting stock prices and making informed investment decisions in the stock market.

3. Machine Learning Techniques

Types of machine learning –

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning



Types of ML

3.1. Supervised Learning

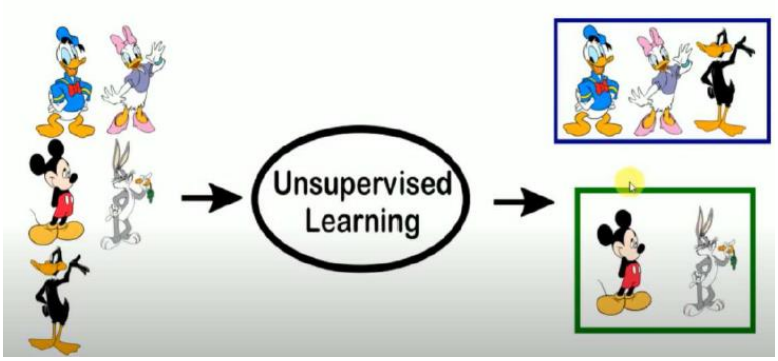
In supervised learning, an AI system is presented with labelled data, where each data point is tagged with the correct label. This labelled data is used to train the supervised model. Once trained, the model can be tested using a separate test set to evaluate its ability to predict the correct output. The labelled data serves as input to the supervised machine learning model during training, allowing the model to learn from the provided examples. As the model is trained with more labelled data, it continues to improve its ability to make accurate predictions. After training is complete, the model can be tested with new, unseen data to assess its performance. The accuracy of the model can be determined based on the results of the testing phase.

Example of Supervised LEarning

For example, users provide a set of emails, each labelled as spam or not spam. These labelled emails are used as input for the learning system, which continues to learn as more emails are provided. Once the learning process is complete, a model is created. When new, unlabelled emails are introduced, the model can classify them as either spam or not spam based on what it has learned.

3.2. Unsupervised Learning

In unsupervised learning, an AI system is presented with unlabelled and uncategorized data. For example, consider a scenario where we provide the model with images of ducks and non-ducks without specifying which is which. The unsupervised model then analyzes the data to identify patterns and structures within it. Unlike supervised learning, where data is labeled, unsupervised learning focuses on finding hidden patterns or relationships in the data without the need for predefined labels. The model learns the underlying structure or distribution of the data to gain insights into its characteristics. It does this by identifying similarities or distances between data points and grouping them into clusters or patterns. The key distinction from supervised learning is that no labels are provided to the model during training.



Unsupervised Learning

4. How Supervised Learning helps in Stock Market Prediction ?

Supervised learning is a subset of machine learning where algorithms learn from labelled data to make predictions and decisions. In the context of stock market prediction, supervised learning involves training models on historical stock market data, where each data point is labelled with the correct outcome (e.g., the future stock price movement). This labelled data is used to teach the model the relationship between input features (e.g., past stock prices, trading volumes, technical indicators) and the target variable (e.g., future stock price movement).

4.1. Steps in Supervised Learning for Stock Market Prediction:

4.1.1. Data Collection :

Data collection is a critical first step in the supervised learning process for stock market prediction. Historical stock market data is essential for training machine learning models to make predictions about future stock prices. This data typically includes information such as stock prices (open, high, low, close), trading volumes, and other relevant financial indicators.

4.1.2. Data Preprocessing :

Once the data is collected, it needs to be cleaned and processed to ensure that it is suitable for training the machine learning model. This process involves handling missing values, removing outliers, and converting the data into a format that can be easily used by the machine learning algorithms.

Data preprocessing is an essential step in preparing the data for training the machine learning model. It involves several tasks, including:

a. Handling Missing Values: Missing values in the data can be filled using techniques such as mean imputation, median imputation, or forward/backward filling.
b. Removing Outliers: Outliers in the data can be removed using statistical methods such as z-score or interquartile range (IQR).
c. Normalizing the Data: Normalizing the data to a standard scale (e.g., between 0 and 1) can help improve the performance of the machine learning model.

Overall, data collection and preprocessing are crucial steps in the supervised learning process for stock market prediction. By collecting and preparing high-quality data, machine learning models can be trained to make accurate predictions about future stock prices.

4.1.3. Feature Engineering and Feature Selection :

Feature engineering is a crucial step in the supervised learning process for stock market prediction. It involves selecting and creating relevant features from the raw data that can help the machine learning model make accurate predictions. Feature engineering is a creative process that requires domain knowledge and an understanding of the underlying patterns in the data.

One common technique in feature engineering for stock market prediction is to calculate technical indicators from the raw data. Technical indicators are mathematical calculations based on historical stock prices and volumes that can help identify trends and patterns in the data. Examples of technical indicators include moving averages, relative strength index (RSI), and exponential moving average (EMA).

Overall, feature engineering is a critical step in the supervised learning process for stock market prediction. By selecting and creating relevant features from the raw data, machine learning models can be trained to make accurate predictions about future stock prices.

4.1.4. Training and Testing :

Training and testing are fundamental steps in supervised learning for stock market prediction. These steps involve using the preprocessed data to train a machine learning model and then evaluating its performance on unseen data.

a. Training:

During the training phase, the machine learning model learns from the historical data to make predictions about future stock prices. The model is trained on a subset of the data called the training set, which includes both features (e.g., historical stock prices) and labels (e.g., future stock prices). The model learns the underlying patterns and relationships in the data by adjusting its internal parameters through a process called optimization.

b. Testing:

Once the model is trained, it is evaluated on a separate subset of the data called the testing set. The testing set is used to assess how well the model generalizes to new, unseen data. The model makes predictions about future stock prices based on the features in the testing set, and these predictions are compared to the actual stock prices to evaluate the model's performance.

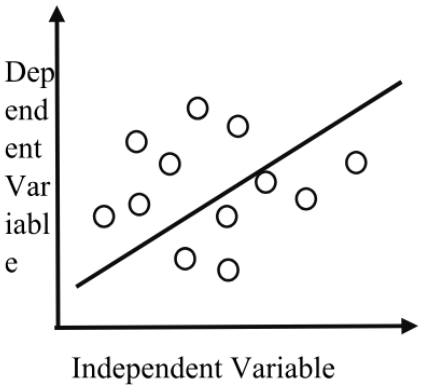
Model Evaluation

Model evaluation is a critical step in machine learning, especially in the context of stock market prediction. It involves assessing the performance of a trained model on unseen data to determine its effectiveness and reliability. In stock market prediction, accurate evaluation of models is essential for making informed investment decisions and managing risks effectively. Several key aspects of model evaluation in stock market prediction include the choice of evaluation metrics, cross-validation techniques, and the interpretation of results.

5. Algorithms used in our proposed work

5.1. Linear Regression

Regression analysis is a statistical technique used to model the relationship between a dependent variable (DV) and one or more independent variables (IVs). The goal of regression analysis is to find the best-fitting curve that describes the relationship between the variables. This curve can be a straight line, known as linear regression, or a nonlinear curve, known as nonlinear regression.



Graph of Linear Regression

The quality of the fit of the curve to the data is measured by a coefficient of correlation, denoted as 'r'.

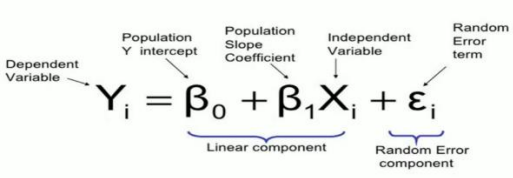
In regression analysis, the regression model is represented by a linear equation, typically written as:

$$y = \beta_0 + \beta_1 x + \varepsilon$$

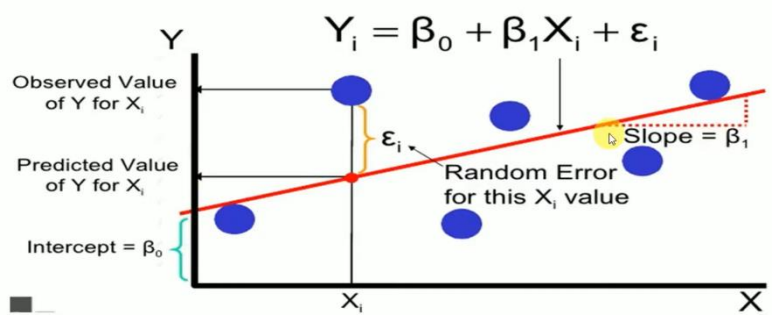
where:

a) y is the dependent variable, which is the variable we are trying to predict or explain.
b) x is the independent variable, also known as the predictor variable, which is used to predict the dependent variable.
c) β_0 is the y-intercept of the regression line, which represents the value of y when x is 0.

d) β_1 is the slope of the regression line, which represents the change in y for a one-unit change in x.
e) ε represents the error term, which accounts for the variability in y that is not explained by the regression model.



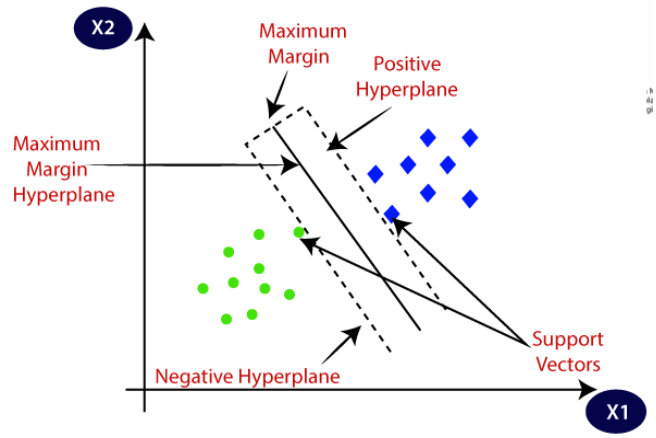
In simple linear regression, there is only one independent variable (x) and one dependent variable (y). However, in multiple linear regression, there can be multiple independent variables (x_1, x_2, ...) used to predict the dependent variable y.



Graph of Linear Regresion

5.2. Support Vector Machine

Support Vector Machine (SVM) is one of the most popular supervised learning algorithms used for classification and regression problems. Primarily, it is employed for classification tasks in machine learning.



Support Vector Graph

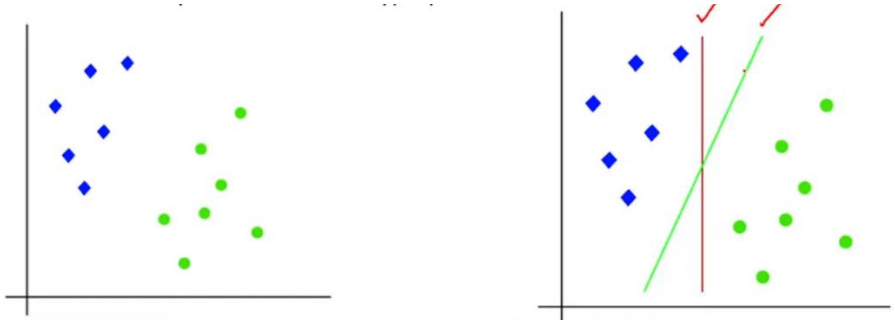
5.2.1. Goal of SVM

The primary objective of the SVM algorithm is to create the best possible decision boundary, known as a hyperplane, that can segregate n-dimensional space into classes. This segregation allows for the correct classification of new data points in the future.

5.2.2 Hyperplane and Support Vectors

Hyperplane: The best decision boundary that separates different classes. There can be multiple potential hyperplanes, but the SVM aims to identify the one with the maximum margin, i.e., the maximum distance between the data points of different classes.

Support Vectors: Data points that are closest to the hyperplane and influence its position. These points support the hyperplane, hence the name "Support Vector Machine."



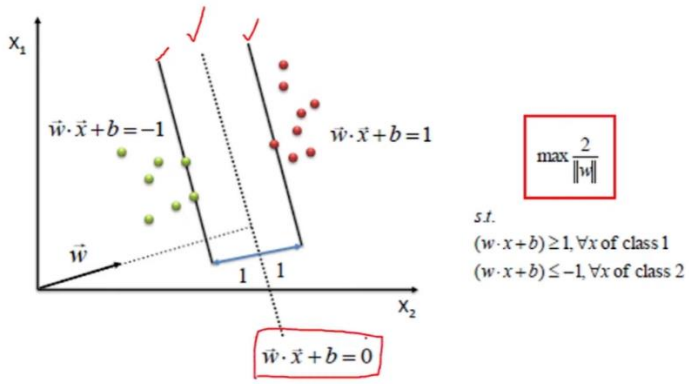
Hyperplane and Support Vectors

5.2.3 Types of SVM

a) Linear SVM: Used for linearly separable data, where a single straight line can classify the data into two distinct classes. This classifier is known as a Linear SVM classifier.
b) Non-linear SVM: Used for non-linearly separable data, where a straight line cannot classify the data. This classifier is known as a Non-linear SVM classifier.

5.2.4. Linear SVM

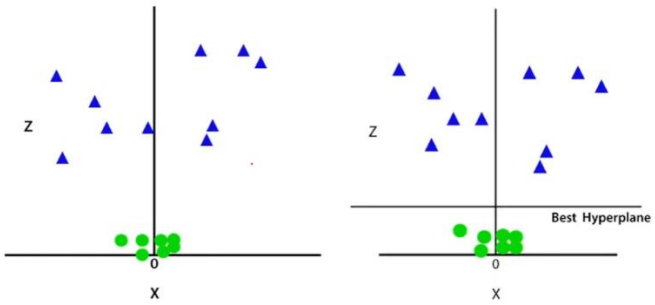
For linearly separable data, the decision boundary can be represented as a straight line. The SVM algorithm ensures that this line maximizes the margin between the data points of different classes. The conditions for a linear SVM are:



Graph of Linear SVM

5.2.5. Non-linear SVM

When dealing with non-linear data, a single straight line is insufficient to separate the classes. To address this, SVM adds additional dimensions to transform the data into a higher-dimensional space where a hyperplane can be used for classification.

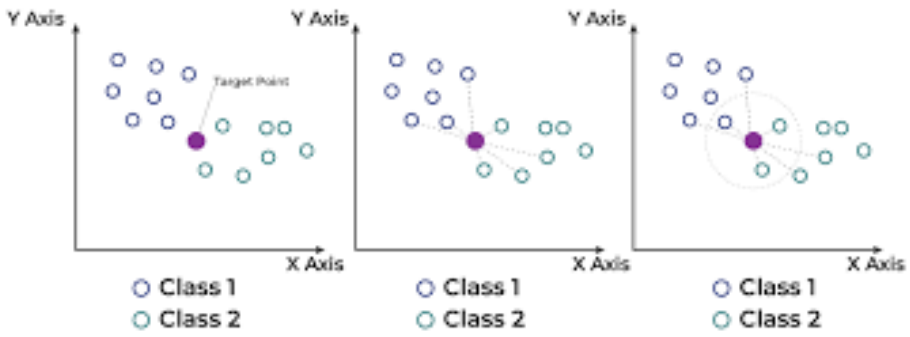


5.2.6. SVM on our proposed work

Support Vector Machine (SVM) is a robust machine learning algorithm used for short-term stock market prediction, aiming to forecast future open, close, high, and low prices. SVM works by identifying a hyperplane in the feature space that best separates historical stock market data into different classes, typically based on price movements. This hyperplane serves as the decision boundary for classifying future data points. SVM's ability to handle high-dimensional data and nonlinear relationships makes it suitable for capturing complex patterns in stock market data. By analyzing historical trends and patterns, SVM can predict future stock prices with reasonable accuracy, helping traders and investors make informed decisions. However, like any predictive model, the accuracy of SVM predictions depends on the quality of the input data, the choice of features, and the appropriateness of the model parameters. Overall, SVM stands as a valuable tool in short-term stock market prediction, offering insights that can assist in optimizing trading strategies and maximizing returns.

5.3. K-Nearest Neighbors (KNN) in Short-Term Stock Market Prediction

K-Nearest Neighbors (KNN) is a supervised machine learning algorithm that can be used for both regression and classification tasks. In the context of stock market prediction, KNN is used as a regression algorithm. It works by finding the k data points in the training set that are closest to the input data point and averaging their output values to make a prediction.



KNN Graph

In KNN, the distance between data points is calculated using a distance metric, such as Euclidean distance. The algorithm then identifies the k nearest neighbors of the input data point based on this distance metric. For regression tasks, the output value for the input data point is computed as the average of the output values of its k nearest neighbors.

5.3.1. How KNN Helps in Short-Term Market Prediction:

1. Pattern Recognition: KNN is effective in recognizing patterns in historical stock price data. By looking at the historical prices of similar stocks or the same stock at different times, KNN can identify patterns that are likely to repeat in the future.

2. Flexibility: KNN is a non-parametric algorithm, which means it does not make any assumptions about the underlying distribution of the data. This flexibility allows KNN to adapt to different types of stock price movements and patterns.

3. Adaptability: KNN is a lazy learning algorithm, which means it does not require training a model before making predictions. This makes it adaptable to changing market conditions, as it can quickly incorporate new data points into its predictions.

5.4. Decision Tree (DT) in Short-Term Stock Market Prediction:

Decision Tree (DT) is a supervised machine learning algorithm used for both regression and classification tasks. It creates a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

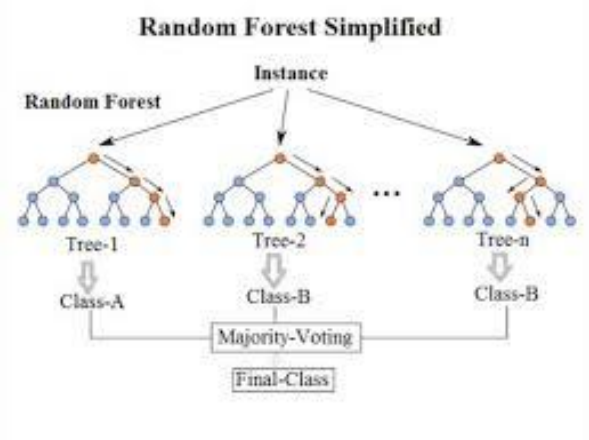
In a Decision Tree, the root node represents the entire dataset, and each internal node represents a feature or attribute. The branches represent the decision rules, and the leaf nodes represent the outcome or prediction. The tree is built by recursively splitting the dataset into subsets based on the value of a selected feature, with the goal of maximizing the homogeneity of the target variable within each subset.

5.4.1. How Decision Tree Helps in Short-Term Market Prediction:

a. Interpretability: Decision Trees are easy to interpret and explain. The model can be visualized, allowing users to understand the logic behind the predictions.
b. Handling Nonlinear Relationships: Decision Trees can capture nonlinear relationships between features and the target variable. This flexibility allows them to model complex relationships in stock price data.
c. Feature Importance: Decision Trees can provide insights into the importance of different features in predicting stock prices. This information can help identify key factors driving stock price movements.
d. Handling Missing Values: Decision Trees can handle missing values in the data. They can make predictions even when some features have missing values by using other features to make decisions.
e. Suitability for Short-Term Prediction: Decision Trees are suitable for short-term stock market prediction because they can capture short-term trends and patterns in the data without assuming a specific form for the underlying data distribution.

5.5. RANDOM FOREST:

Random Forest is an ensemble learning method that combines multiple decision trees to create a more robust and accurate model. In the context of stock market prediction, Random Forest can be used for both regression and classification tasks. It is based on the idea that multiple weak learners (individual decision trees) can come together to form a strong learner (the Random Forest) by reducing variance and overfitting.



Random Forest Architecture

a. Ensemble of Decision Trees: Random Forest builds a forest of decision trees during training. Each tree is trained on a random subset of the training data and a random subset

of the features. This randomness helps to introduce diversity among the trees, reducing the risk of overfitting.

b. Voting Mechanism: For regression tasks, the output of each tree is averaged to produce the final prediction. For classification tasks, the output of each tree is aggregated through voting to determine the majority class prediction.
c. Bootstrap Aggregating (Bagging): Random Forest uses a technique called bagging, which stands for bootstrap aggregating. It involves creating multiple bootstrap samples (random samples with replacement) from the training data and training each decision tree on a different bootstrap sample. This further increases the diversity among the trees.
d. Feature Importance: Random Forest can provide insights into the importance of different features in making predictions. It calculates the importance of each feature based on how much the tree nodes that use that feature reduce impurity (e.g., Gini impurity or entropy).

5.5.1. How Random Forest Helps in Short-Term Market Prediction:

a. Improved Accuracy: Random Forest tends to provide higher accuracy compared to individual decision trees. By combining multiple trees, it reduces overfitting and generalizes better to unseen data, which is crucial for short-term stock market prediction.
b. Handling Nonlinear Relationships: Random Forest can capture nonlinear relationships between features and the target variable. This flexibility allows it to model complex relationships in stock price data, which is often nonlinear and dynamic.
c. Robustness to Noise: Random Forest is robust to noisy data. Since it averages or aggregates the predictions of multiple trees, it tends to be less affected by outliers or noisy data points.
d. Interpretability: While Random Forest models are less interpretable compared to individual decision trees, they can still provide insights into feature importance. This information can help analysts and traders understand which factors are driving stock price movements.
e. Suitability for Short-Term Prediction: Random Forest is suitable for short-term stock market prediction because it can capture short-term trends and patterns in the data without assuming a specific form for the underlying data distribution.

5.6. Artificial Neural Networks (ANN) in Short-Term Stock Market Prediction

Artificial Neural Networks (ANN) are computational models inspired by the biological neural networks of the human brain. They consist of interconnected nodes (neurons) organized in layers. Each neuron processes inputs, applies an activation function, and passes the output to the next layer of neurons. ANNs are used for various machine learning tasks, including regression and classification.

a. Neural Network Architecture: ANN consists of an input layer, one or more hidden layers, and an output layer. Each layer contains multiple neurons, and connections between neurons have associated weights that are adjusted during training.

b. Training Process: Training an ANN involves feeding input data through the network, calculating the output, comparing it to the actual output (target), and adjusting the weights using optimization algorithms (e.g., backpropagation) to minimize the prediction error.
c. Activation Functions: Neurons in ANNs use activation functions to introduce nonlinearity into the model, allowing it to learn complex patterns in the data. Common activation functions include sigmoid, tanh, ReLU, and softmax.

5.6.1. How Artificial Neural Networks Help in Short-Term Market Prediction:

a. Nonlinear Relationships: ANNs can capture complex, nonlinear relationships in stock price data, which is crucial for short-term prediction where price movements are often influenced by multiple factors.
b. Feature Learning: ANNs can automatically learn relevant features from the input data, reducing the need for manual feature engineering. This ability is useful in stock market prediction, where identifying relevant features can be challenging.

6. Evaluation Matrix:

The evaluation matrix, also known as performance metrics, is an essential aspect of machine learning. It refers to the set of metrics used to measure the effectiveness of a machine-learning model. These metrics help to determine how well a model is able to make accurate predictions or classifications on unseen data. To compare different models Evaluation metrics can be used to compare machine learning models trained on the same dataset to solve the same problem. For example, if two models have similar accuracy scores, but if one has a higher precision score, that will be preferred.

In the regression task, we are supposed to predict the target variable which is in the form of continuous values. To evaluate the performance of such a model below mentioned evaluation metrics are used:

- Mean Absolute Error(MAE)
- Mean Squared Error(MSE)
- Mean Absolute Percentage Error (MAPE)

6.1. Mean Absolute Error(MAE) :

One of the most common metrics of model prediction accuracy, **mean absolute percentage error (MAPE)** is the percentage equivalent of mean absolute error (MAE). Mean absolute percentage error measures the average magnitude of error produced by a model, or how far off predictions are on average. While understanding this metric and how to calculate it is important, it's also critical to understand its strengths and limitations when using it in production. MAPE is defined as the average absolute percentage difference between predicted values and actual values.

MAPE = (1 / sample size) x ∑[(|actual - forecast|) / |actual|] x 100

6.2. Mean Squared Error(MSE):

In the fields of regression analysis and machine learning, the **Mean Square Error (MSE)** is a crucial metric for evaluating the performance of predictive models. It measures the average squared difference between the predicted and the actual target values within a dataset. It measures the average squared difference between the predicted and the actual target values within a dataset. The primary objective of the MSE is to assess the quality of a model's predictions by measuring how closely they align with the ground truth.

$$MSE = \frac{\sum(y_i - \hat{y}_i)^2}{n}$$

where:

y_i is the i_{th} observed value, $\hat{y}_i$ is the corresponding predicted value, n = the number of observations

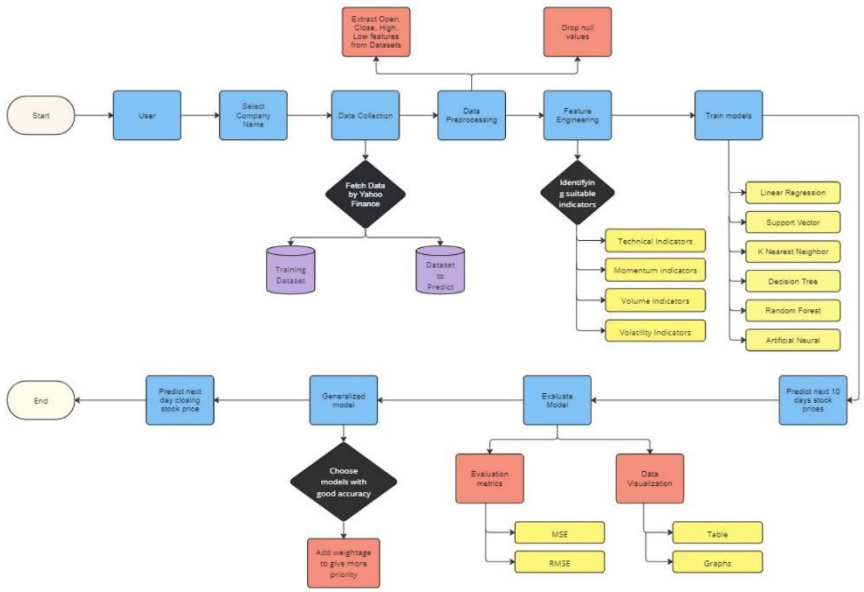
6.3. Mean Absolute Percentage Error (MAPE):

Mean absolute percentage error measures the average magnitude of error produced by a model, or how far off predictions are on average. A MAPE value of 20% means that the average absolute percentage difference between the predictions and the actuals is 20%.

Calculate MAPE:

Absolute percent error = [(| actual - forecast |) / | actual |] x 100

7. ARCHITECTURE



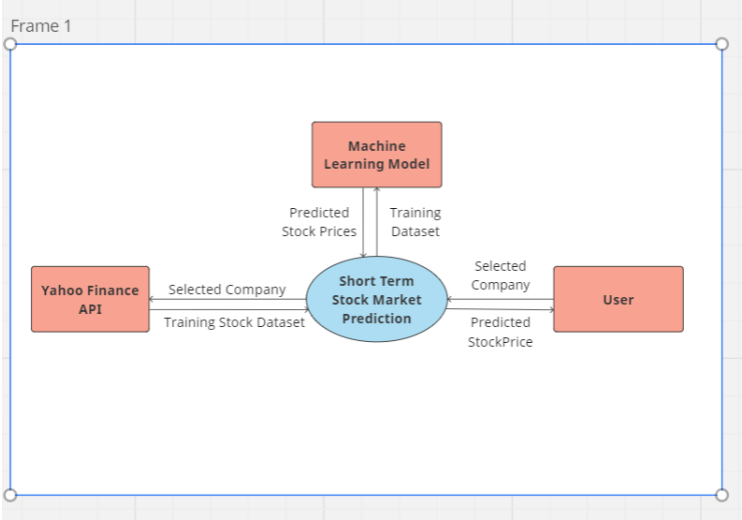
Architecture

The architecture diagram for the Short-Term Stock Market Prediction project delineates a detailed workflow analysis of the system. It begins with User Interaction, where the user selects a company for stock price prediction, initiating the data collection process. The system fetches historical stock data from Yahoo Finance and preprocesses it by extracting relevant values and handling null values. Subsequently, the data undergoes Feature Engineering, where suitable

indicators are identified and created for model training. The system then trains multiple machine learning models, including Linear Regression, Support Vector, K Nearest Neighbor, Decision Tree, Random Forest, and Artificial Neural Network. These models are used to predict stock prices for the next 10 days. Model Evaluation is performed using metrics like Mean Squared Error and Root Mean Squared Error, and the best-performing models are selected and combined to create a Generalized Model. This ensemble model provides the final predicted stock price for the next day. The architecture diagram provides a comprehensive view of the system's workflow, highlighting the key processes and interactions involved in generating stock market predictions.

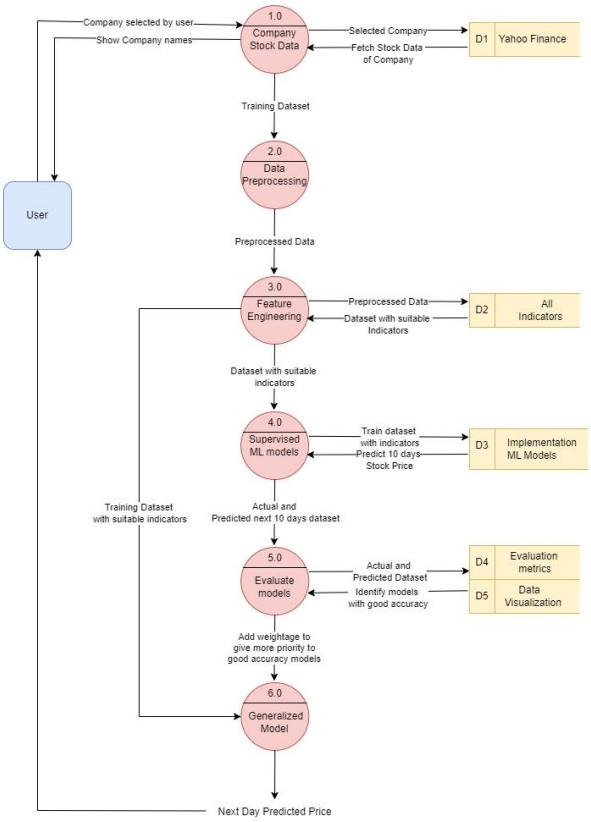
8.0 DATA FLOW DIAGRAM (DFD)

8.1 LEVEL 0 DFD /CONTEXT DIAGRAM



DFD 0/Context Diagram

8.2 LEVEL 1 DFD

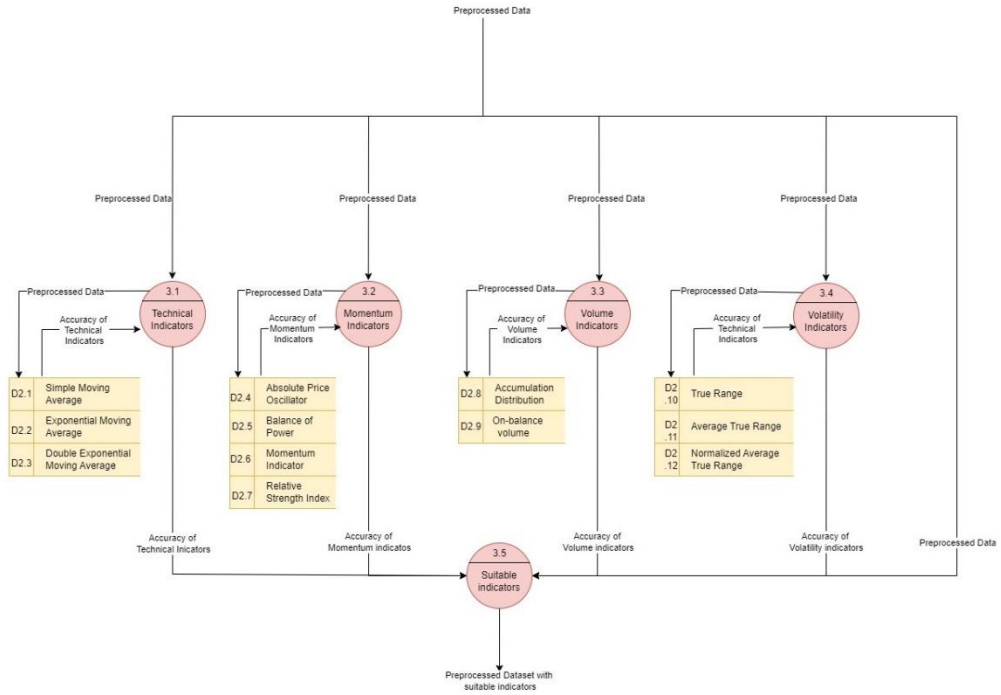


Level 1 DFD

The data flow diagram (DFD) illustrates the flow of information in the Short-Term Stock Market Prediction project. The primary external entity, the User, interacts with the system by selecting a company for stock price prediction and receiving the predicted prices. The system fetches historical stock data from Yahoo Finance (D1), preprocesses it to prepare for analysis, and performs feature engineering to generate suitable indicators (D2). Supervised machine learning models (D3) are then trained on the preprocessed data to predict stock prices for the next 10 days. The models' performance is evaluated using various metrics (D4), and the best-performing models are used to create a generalized model. This model provides the next day's predicted stock price to the user. The DFD provides a clear overview of the system's functionality, showcasing the data flow from input to output, and serves as a foundational element for the project report.

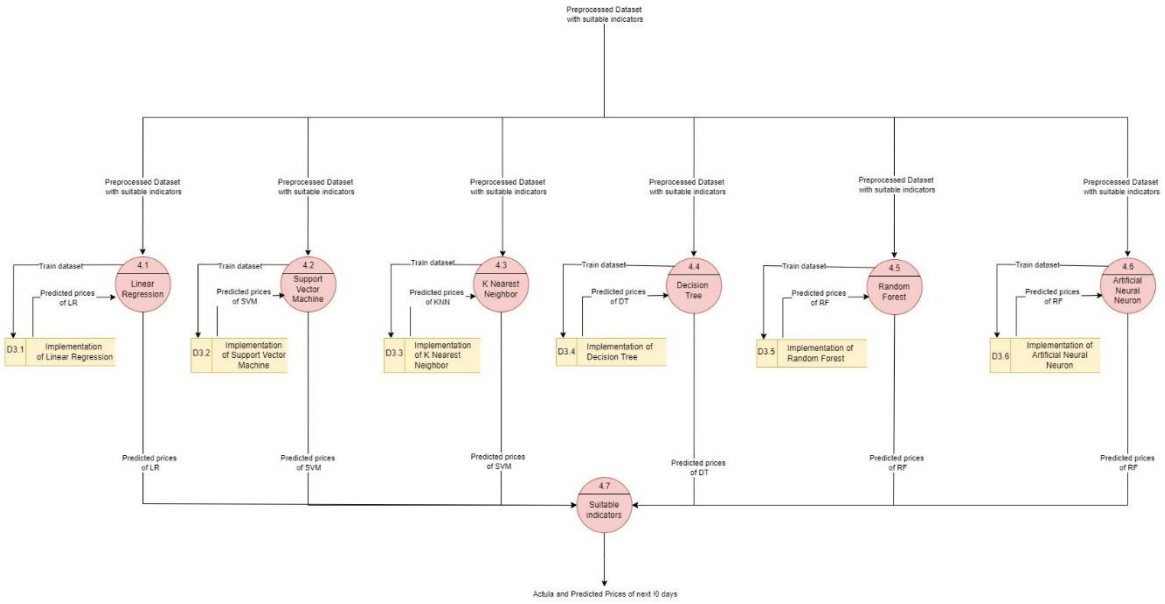
8.3 LEVEL – 2 DFD

8.3.1. FEATURE ENGINEERING



Level 2 DFD- Feature Engineering

The Level 2 Data Flow Diagram (DFD) provides a detailed view of the feature engineering process within the short-term stock market prediction project. It outlines specific processes for calculating various types of indicators, including technical, momentum, volume, and volatility indicators, from preprocessed stock data. Each process takes the preprocessed data as input and computes specific indicators related to its category, such as Simple Moving Average (SMA) and Exponential Moving Average (EMA) for technical indicators, Relative Strength Index (RSI) for momentum indicators, and On-Balance Volume (OBV) for volume indicators. The outputs of these processes include the accuracy of the calculated indicators and the preprocessed data enriched with these indicators. The final process aggregates the most suitable indicators from each category to create a refined dataset for further analysis or model training. Overall, the diagram provides a clear and detailed overview of the feature engineering process, highlighting the calculations and transformations involved in generating key indicators for stock price prediction.



Level 2 DFD – Supervised ML Model

8.3.2. SUPERVISED ML MODELS

The Level 2 Data Flow Diagram (DFD) elaborates on the model training and prediction processes within your short-term stock market prediction project. Each process corresponds to a specific machine learning model, and the diagram details the input, output, and implementation aspects of these models. Here's a breakdown of the components and their interactions:

Processes (Pink Ovals)

a. Linear Regression (LR), Support Vector Machine (SVM), K Nearest Neighbor (KNN), Decision Tree (DT), Random Forest (RF), and Artificial Neural Network (ANN): These processes represent the training of each respective machine learning model using the preprocessed dataset with suitable indicators. Each process outputs the predicted prices of its corresponding model.
b. Suitable Indicators: This process aggregates the predicted prices from all models (LR, SVM, KNN, DT, RF, ANN) to determine the actual and predicted stock prices for the next 10 days.

Data Stores (Yellow Rectangles)

• Implementation of LR, SVM, KNN, DT, RF, ANN (D3.1 to D3.6): These stores contain the trained models and their implementation details, ensuring that each model is properly documented and can be reused or updated as needed.

Data Flows (Arrows)

• Preprocessed Dataset with Suitable Indicators: This is the input flowing into each model training process (LR, SVM, KNN, DT, RF, ANN), representing the cleaned, normalized, and feature-engineered dataset ready for training.

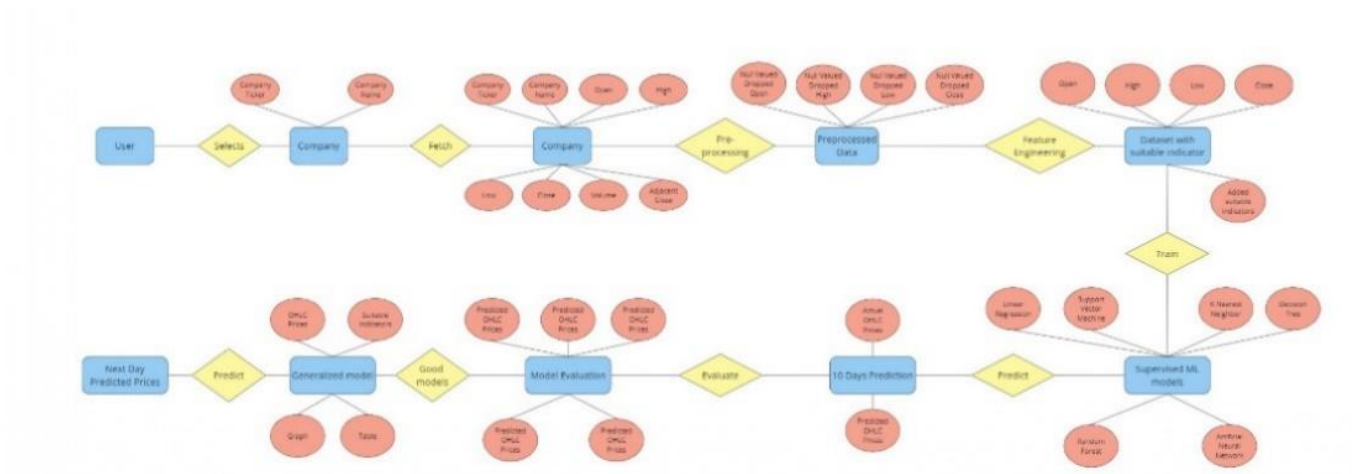
• Train Dataset: This input signifies the training phase of each model, where the models learn patterns from the dataset.

• Predicted Prices: The output from each model training process is the predicted stock prices based on the learned patterns.

• Actual and Predicted Prices of Next 10 Days: The final output provides the actual and predicted stock prices for the next 10 days, which can be used for evaluation or further analysis.

Overall, the diagram illustrates the model training process for various machine learning algorithms and how their predictions are aggregated to provide a comprehensive forecast for the stock prices.

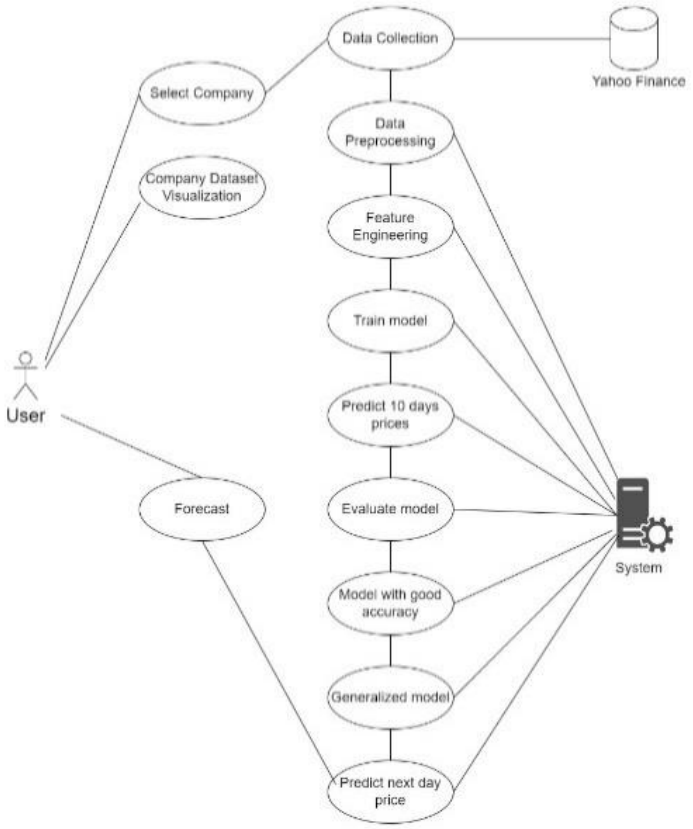
9. ER DIAGRAM



ER DIAGRAM

The Level 2 Data Flow Diagram (DFD) provides a detailed view of the feature engineering, model training, and prediction processes within the short-term stock market prediction project. It outlines specific processes for calculating technical, momentum, volume, and volatility indicators from preprocessed stock data. These indicators are then used to train multiple machine learning models, including Linear Regression, Support Vector Machine, K Nearest Neighbor, Decision Tree, Random Forest, and Artificial Neural Network. Each model is trained using the preprocessed dataset with suitable indicators and produces predicted stock prices. The diagram also includes a process to aggregate predictions from all models to determine the actual and predicted stock prices for the next 10 days. This comprehensive overview helps understand how the system transforms raw stock data into actionable insights for making informed trading decisions.

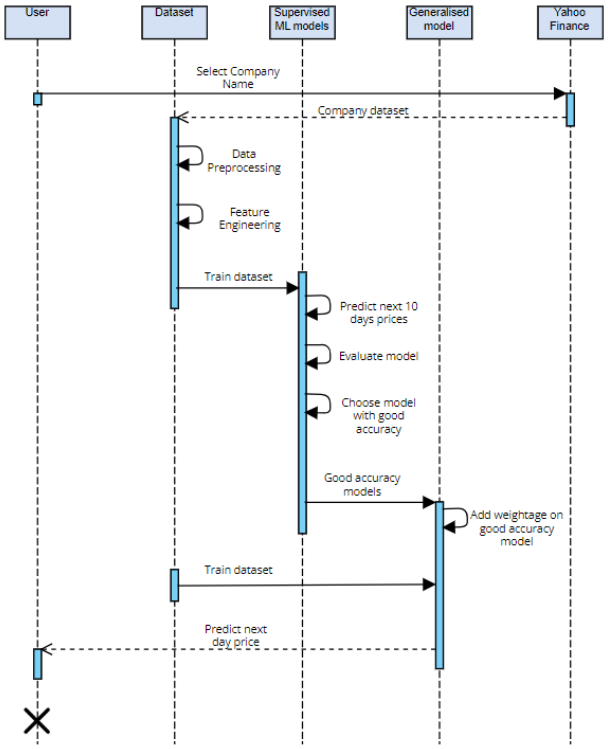
9. USER CASE DIAGRAM



USER CASE DIAGRAM

The UML diagram for the Short-Term Stock Market Prediction project illustrates the interactions and processes involved in the system. The primary actor, the User, interacts with the system by selecting a company, visualizing data, and receiving forecasted stock prices. This interaction initiates the data collection process from Yahoo Finance, followed by data preprocessing to clean and prepare the data for analysis. Feature engineering is then performed to generate indicators that are used to train machine learning models. These models predict stock prices for the next 10 days, which are evaluated to select models with good accuracy. A generalized model is created using the best-performing models, and the next day's stock price is predicted. Finally, the forecasted prices are delivered to the user. The diagram effectively captures the flow of data and the key functionalities of the system, providing a clear overview of the project's processes.

10. SEQUENCE DIAGRAM



SEQUENCE DIAGRAM

11. SELECTION OF TRAINING DATA:

To determine the optimal week for training machine learning models on stock market data, we need to evaluate the performance of various models on datasets spanning different weeks. The objective is to identify the week where the models' predictions align most closely with the actual stock prices, indicating superior training performance. This analysis involves examining datasets, tabular data, and graphical representations for each week and computing performance metrics to make an informed decision.

11.1 One Week Training Data On Reliance Industries

In the first week, we analyzed data from February 20, 2024, to February 29, 2024. The dataset included columns such as Date, Open, High, Low, Close, Adjusted Close, and Volume, representing daily stock market performance. The tabular data presented the actual close prices alongside the predicted prices from six different models: Ridge Regression (RR), Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), and Artificial Neural Network (ANN). The corresponding graph illustrated the actual close prices with black dots, while the predicted prices for each model were shown with different colored

lines. For Week 1, noticeable deviations were observed, particularly in the ANN model's predictions, indicating a optimal training period for all models except ANN model.

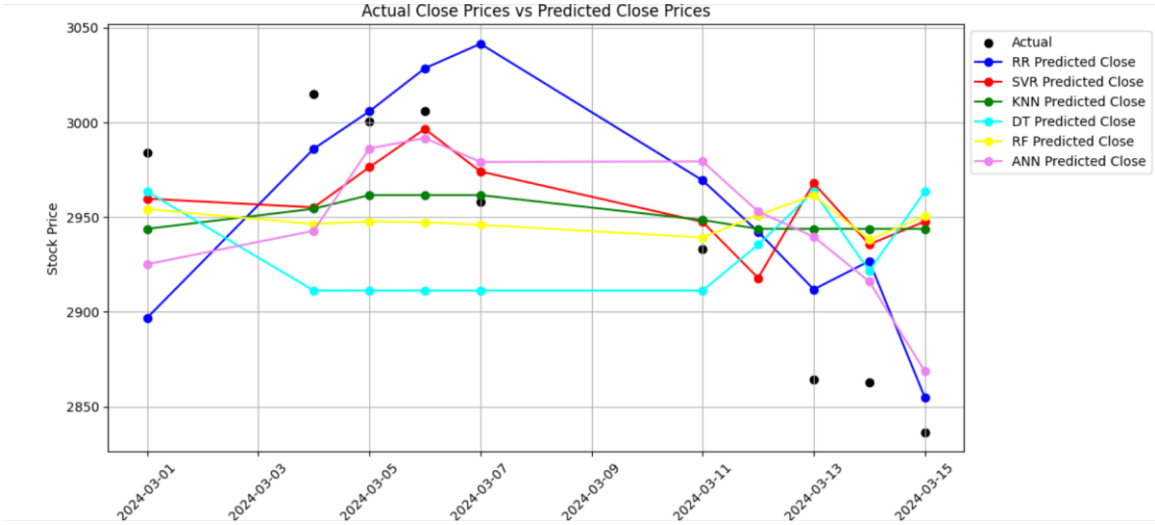
Week 1 -

Date	Open	High	Low	Close	Adj Close	Volume
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049	2942.050049	3558748
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902	2935.399902	6360146
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000	2963.500000	9246864
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000	2987.250000	7219292
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902	2974.649902	3756553
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049	2971.300049	5413022
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000	2911.250000	4323975
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098	2921.600098	11814488

Dataset of Week 1 Reliance Industries

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2896.77397658	2959.71081623	2943.80000000	2963.50000000	2954.22500000	2925.09036878
3014.80004883	2986.06214823	2955.11927495	2954.40996094	2911.25000000	2946.33749023	2942.80224691
3000.39990234	3005.78935317	2976.30473219	2961.58999023	2911.25000000	2947.75297119	2986.21808622
3006.00000000	3028.55448912	2996.54621276	2961.58999023	2911.25000000	2947.31749023	2991.71708395
2957.85009766	3041.48746010	2974.07120639	2961.58999023	2911.25000000	2945.94899658	2979.12863270
2933.19995117	2969.37585180	2947.45741153	2948.46000977	2911.25000000	2939.23500244	2979.39385984
2950.85009766	2942.27125842	2917.89311032	2943.80000000	2935.39990234	2951.12248779	2952.91430521
2864.35009766	2911.82979128	2967.88305270	2943.80000000	2963.50000000	2961.53947754	2939.63595105
2862.94995117	2926.75114067	2935.57495681	2943.80000000	2921.60009766	2938.12653320	2915.95927964
2836.44995117	2854.79489603	2947.68533459	2943.80000000	2963.50000000	2950.48500977	2868.57140620

Actual and Predicted



Graph

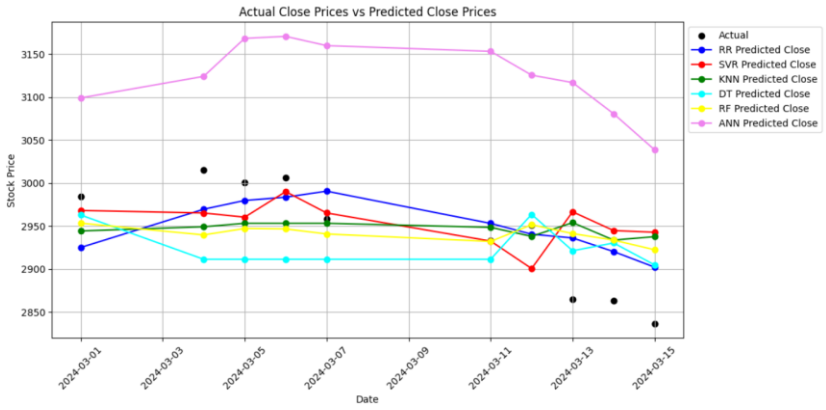
Week 2 -

Date	Open	High	Low	Close	Adj Close	Volume
2024-02-09	2908.000000	2943.949951	2901.899902	2921.500000	2921.500000	6278399
2024-02-12	2921.500000	2922.000000	2884.699951	2904.699951	2904.699951	3337215
2024-02-13	2911.000000	2958.000000	2908.000000	2930.199951	2930.199951	3857797
2024-02-14	2915.000000	2967.300049	2915.000000	2962.750000	2962.750000	3558944
2024-02-15	2966.699951	2969.449951	2933.050049	2941.199951	2941.199951	5003391
2024-02-16	2952.949951	2954.000000	2917.100098	2921.149902	2921.149902	4883749
2024-02-19	2924.100098	2959.000000	2907.050049	2948.000000	2948.000000	3364914
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049	2942.050049	3558748
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902	2935.399902	6360146
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000	2963.500000	9246864
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000	2987.250000	7219292
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902	2974.649902	3756553
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049	2971.300049	5413022
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000	2911.250000	4323975
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098	2921.600098	11814488

Dataset of Week 2 Reliance Industries

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2924.95746433	2968.06978310	2944.20000000	2962.75000000	2953.25399902	2976.48014480
3014.80004883	2969.47865251	2965.12762007	2949.05000000	2911.25000000	2939.58749268	2997.28126033
3000.39990234	2979.65613878	2960.24959158	2953.11997070	2911.25000000	2947.06796387	3043.96176269
3006.00000000	2983.47808673	2989.94569216	2953.11997070	2911.25000000	2946.53198730	3053.10974594
2957.85009766	2990.47183399	2965.15055922	2953.11997070	2911.25000000	2940.72550049	3039.41878868
2933.19995117	2952.97893980	2932.43507925	2948.36997070	2911.25000000	2931.98599121	3033.31636571
2950.85009766	2940.45991886	2900.45821084	2937.92998047	2963.50000000	2951.42299316	3006.77011059
2864.35009766	2936.06569370	2966.46254592	2953.87998047	2921.14990234	2941.18998291	2991.89608399
2862.94995117	2920.20082114	2944.66257689	2933.60000000	2930.19995117	2933.54950928	2960.42294256
2836.44995117	2902.08820658	2942.69513138	2937.53999023	2904.69995117	2922.29546631	2920.51983755

Actual and Predicted data



Graph

The second week's analysis covered data from February 9, 2024, to February 29, 2024, extending the dataset to the last three weeks of February. The dataset maintained the same structure, including essential trading columns. The tabular data again compared actual close prices with predictions from the same six models. Graphical representations for Week 2 showed actual close prices versus predicted prices for each model, with the predictions generally less closer to the actual prices compared to Week 1 indicating less optimal approach. This suggested that the models suffer from overfitting due to the additional historical data.

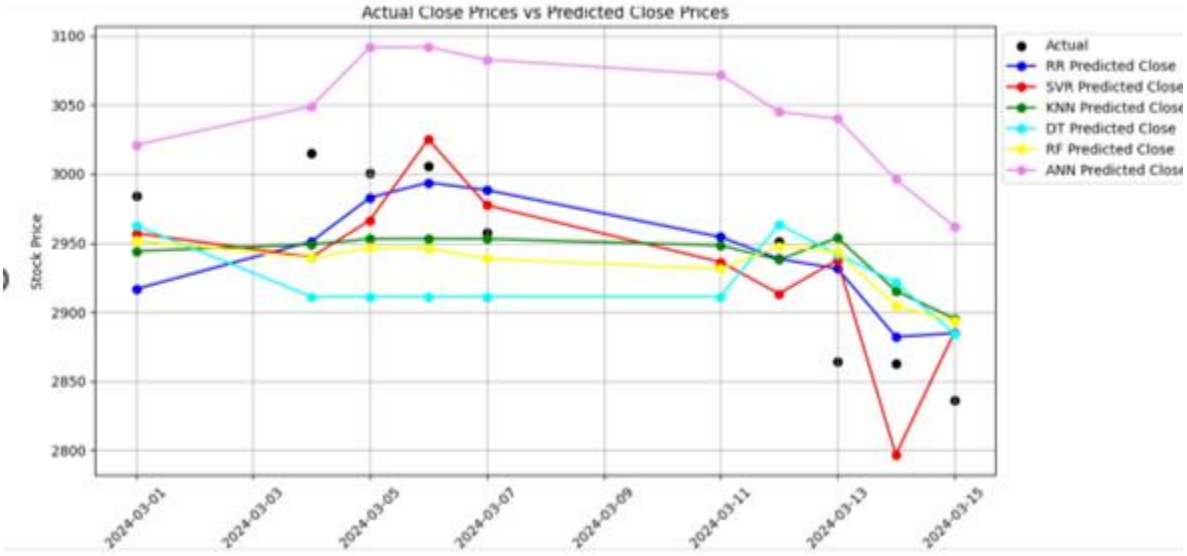
Week 3

Date	Open	High	Low	Close	Adj Close	Volume
2024-01-31	2808.000000	2868.500000	2805.000000	2853.250000	2853.250000	7565113
2024-02-01	2870.000000	2886.699951	2836.100098	2853.300049	2853.300049	6674681
2024-02-02	2866.350098	2949.800049	2866.350098	2915.399902	2915.399902	9826294
2024-02-05	2921.500000	2941.000000	2863.050049	2878.050049	2878.050049	4407216
2024-02-06	2883.699951	2883.699951	2839.649902	2855.600098	2855.600098	4523992
2024-02-07	2871.850098	2899.000000	2858.500000	2884.300049	2884.300049	4648284
2024-02-08	2900.000000	2918.949951	2855.050049	2900.250000	2900.250000	7347317
2024-02-09	2908.000000	2943.949951	2901.899902	2921.500000	2921.500000	6278399
2024-02-12	2921.500000	2922.000000	2884.699951	2904.699951	2904.699951	3337215
2024-02-13	2911.000000	2958.000000	2908.000000	2930.199951	2930.199951	3857797
2024-02-14	2915.000000	2967.300049	2915.000000	2962.750000	2962.750000	3558944
2024-02-15	2966.699951	2969.449951	2933.050049	2941.199951	2941.199951	5003391
2024-02-16	2952.949951	2954.000000	2917.100098	2921.149902	2921.149902	4883749
2024-02-19	2924.100098	2959.000000	2907.050049	2948.000000	2948.000000	3364914
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049	2942.050049	3558748
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902	2935.399902	6360146
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000	2963.500000	9246864
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000	2987.250000	7219292
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902	2974.649902	3756553
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049	2971.300049	5413022
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000	2911.250000	4323975
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098	2921.600098	11814488

Dataset of Week 3 Reliance Industries

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2916.84876904	2957.13511219	2944.20000000	2962.75000000	2951.90050049	3021.11453779
3014.80004883	2951.09123828	2939.92258051	2949.05000000	2911.25000000	2939.38398438	3049.03413028
3000.39990234	2982.89780486	2966.49692374	2953.11997070	2911.25000000	2946.67745850	3091.70005841
3006.00000000	2993.80078353	3025.15892689	2953.11997070	2911.25000000	2946.20847900	3092.00151252
2957.85009766	2988.43445021	2977.46529758	2953.11997070	2911.25000000	2938.60049072	3082.73037628
2933.19995117	2954.43995663	2936.30664700	2948.36997070	2911.25000000	2931.37948730	3071.82011373
2950.85009766	2938.56764352	2913.39885233	2937.92998047	2963.50000000	2947.64647705	3045.11380990
2864.35009766	2931.55173066	2937.80796246	2953.87998047	2942.05004883	2943.69697510	3040.10998665
2862.94995117	2882.27420198	2796.89477596	2915.38002930	2921.60009766	2904.28754639	2996.17882697
2836.44995117	2884.69219621	2885.28261470	2895.41000977	2884.30004883	2893.56450684	2962.00428854

Actual and Predicted data



Graph

For the third week, the dataset ranged from January 31, 2024, to February 20, 2024. This three-week span provided a broader historical context for model training. The dataset included the same columns, and the tabular data compared actual and predicted close prices from the six models. The graph for Week 3 demonstrated a closer alignment of predicted prices with the actual prices, showing less deviation than the previous weeks. This indicated that the models were able to capture market trends more effectively with a longer training period.

Week 4

Date	Open	High	Low	Close	Adj Close	Volume
2024-01-18	2702.800049	2742.000000	2702.500000	2735.899902	2735.899902	5139719
2024-01-19	2752.000000	2752.000000	2718.000000	2734.899902	2734.899902	5211352
2024-01-23	2743.500000	2743.500000	2645.100098	2657.149902	2657.149902	10027710
2024-01-24	2670.449951	2699.000000	2647.850098	2687.750000	2687.750000	10959564
2024-01-25	2685.899902	2728.300049	2670.399902	2706.149902	2706.149902	5904436
2024-01-29	2729.000000	2905.000000	2720.350098	2896.100098	2896.100098	11946719
2024-01-30	2919.899902	2919.949951	2808.850098	2815.250000	2815.250000	7046989
2024-01-31	2808.000000	2868.500000	2805.000000	2853.250000	2853.250000	7565113
2024-02-01	2870.000000	2886.699951	2836.100098	2853.300049	2853.300049	6674681
2024-02-02	2866.350098	2949.800049	2866.350098	2915.399902	2915.399902	9826294
2024-02-05	2921.500000	2941.000000	2863.050049	2878.050049	2878.050049	4407216
2024-02-06	2883.699951	2883.699951	2839.649902	2855.600098	2855.600098	4523992
2024-02-07	2871.850098	2899.000000	2858.500000	2884.300049	2884.300049	4648284
2024-02-08	2900.000000	2918.949951	2855.050049	2900.250000	2900.250000	7347317
2024-02-09	2908.000000	2943.949951	2901.899902	2921.500000	2921.500000	6278399
2024-02-12	2921.500000	2922.000000	2884.699951	2904.699951	2904.699951	3337215
2024-02-13	2911.000000	2958.000000	2908.000000	2930.199951	2930.199951	3857797
2024-02-14	2915.000000	2967.300049	2915.000000	2962.750000	2962.750000	3558944
2024-02-15	2966.699951	2969.449951	2933.050049	2941.199951	2941.199951	5003391
2024-02-16	2952.949951	2954.000000	2917.100098	2921.149902	2921.149902	4883749
2024-02-19	2924.100098	2959.000000	2907.050049	2948.000000	2948.000000	3364914
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049	2942.050049	3558748
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902	2935.399902	6360146
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000	2963.500000	9246864
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000	2987.250000	7219292
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902	2974.649902	3756553
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049	2971.300049	5413022
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000	2911.250000	4323975
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098	2921.600098	11814488

Dataset of Week 4 Reliance Industries

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2928.88920683	2925.66487290	2944.20000000	2948.00000000	2950.29800049	2912.38027473
3014.80004883	2959.07055840	2949.24888442	2949.05000000	2911.25000000	2942.23648438	2947.70229064
3000.39990234	2992.38100930	2970.49487971	2953.11997070	2911.25000000	2951.36745361	2983.32312130
3006.00000000	2984.99020266	2973.30708497	2953.11997070	2911.25000000	2950.96547119	2978.72606202
2957.85009766	2980.59185749	2978.21671925	2953.11997070	2911.25000000	2945.69248779	2974.76142394
2933.19995117	2958.93612690	2957.46885891	2948.36997070	2911.25000000	2934.28747803	2958.77985009
2950.85009766	2939.30769456	2948.45759802	2937.92998047	2963.50000000	2950.73097412	2932.74582657
2864.35009766	2948.93794808	2940.90626902	2953.87998047	2921.14990234	2942.42646484	2932.22342525
2862.94995117	2879.44073235	2880.84474780	2915.38002930	2884.30004883	2892.54004395	2890.73148418
2836.44995117	2880.44898680	2890.82376995	2895.41000977	2884.30004883	2892.97150391	2854.35940691

Actual and Predicted data

Graph

The fourth week's dataset, covering January 18, 2024, to February 9, 2024, provided an extensive period for model training. The dataset again included the same columns. The table displayed actual and predicted close prices for the six models, and the graph plotted these values, with the actual prices marked by black dots and the predicted prices shown with various colored lines. Week 4's analysis revealed that the predicted prices were the closest to the actual prices, indicating superior performance across all models. This comprehensive dataset allowed the models to better capture the underlying trends and patterns in the stock market.

11.1.1. Comparison and Conclusion:

To identify the optimal training period for the machine learning models, we compared their performance across different datasets spanning one, two, three, and four weeks. The comparison involved evaluating how closely each model's predictions matched the actual stock prices using performance metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R^2). These metrics provided quantitative measures of the models' accuracy and reliability.

11.1.2. Comparison of Training Periods

Week1:

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	50.67	9.25	33.91	36.25	30.19	73.46
2024-03-04	47.17	36.1	65.75	103.55	77.21	75.52
2024-03-05	27.99	39.52	47.28	89.15	59.98	20.92
2024-03-06	37.25	11.98	52.88	94.75	66.01	26.37
2024-03-07	17.97	7.77	4.73	46.6	20.97	13.64
2024-03-11	11.18	1.07	15.17	21.95	2.95	26.97
2024-03-12	13.31	47.2	12.92	12.65	4.0	16.58
2024-03-13	79.9	97.29	89.53	56.8	77.0	64.76
2024-03-14	50.67	65.76	82.26	58.65	69.16	26.72
2024-03-15	82.18	95.89	107.23	93.75	103.07	17.42
MAE	41.83	41.18	51.17	61.41	51.05	36.24

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	2567.45	85.56	1149.89	1314.06	911.44	5396.37
2024-03-04	2225.01	1303.21	4323.06	10722.6	5961.38	5703.27
2024-03-05	783.44	1561.83	2235.4	7947.72	3597.6	437.65
2024-03-06	1387.56	143.52	2796.29	8977.56	4357.32	695.38
2024-03-07	322.92	60.37	22.37	2171.56	439.74	186.05
2024-03-11	124.99	1.14	230.13	481.8	8.7	727.38
2024-03-12	177.16	2227.84	166.93	160.02	16.0	274.9
2024-03-13	6384.01	9465.34	8015.62	3226.24	5929.0	4193.86
2024-03-14	2567.45	4324.38	6766.71	3439.82	4783.11	713.96
2024-03-15	6753.55	9194.89	11498.27	8789.06	10623.42	303.46
MSE	2329.52	2836.83	3720.47	4723.05	3662.73	1863.32

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	1.7%	0.31%	1.14%	1.21%	1.01%	2.46%
2024-03-04	1.56%	1.2%	2.18%	3.43%	2.56%	2.5%
2024-03-05	0.93%	1.32%	1.58%	2.97%	2.0%	0.7%
2024-03-06	1.24%	0.4%	1.76%	3.15%	2.2%	0.88%
2024-03-07	0.61%	0.26%	0.16%	1.58%	0.71%	0.46%
2024-03-11	0.38%	0.04%	0.52%	0.75%	0.1%	0.92%
2024-03-12	0.45%	1.6%	0.44%	0.43%	0.14%	0.56%
2024-03-13	2.79%	3.4%	3.13%	1.98%	2.69%	2.26%
2024-03-14	1.77%	2.3%	2.87%	2.05%	2.42%	0.93%
2024-03-15	2.9%	3.38%	3.78%	3.31%	3.63%	0.61%
MAPE	1.43%	1.42%	1.75%	2.09%	1.75%	1.23%

Dataset of Determining MAE, MSE, MAPE

11.1.3. Week Training Period:

Dataset: This period includes data from the most recent week, offering a limited amount of historical information.

Performance: The models trained on one week of data exhibited the highest MAE and MSE values, indicating significant prediction errors. The R^2 values were low, reflecting poor explanatory power and an inability to capture broader market trends. The predictions were more volatile and less consistent, showing noticeable deviations from the actual prices in the graphs. This limited data did not provide enough context for the models to learn effectively.

Week 2:

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	3515.3	261.79	1604.0	1314.06	961.0	1124.26
2024-03-04	2053.9	2467.11	4323.06	10722.6	5656.54	1911.44
2024-03-05	430.15	1612.02	2235.4	7947.72	2844.09	9377.99
2024-03-06	507.15	257.6	2796.29	8977.56	3536.68	7223.3
2024-03-07	1064.06	53.29	22.37	2171.56	293.09	16710.73
2024-03-11	391.25	0.58	230.13	481.8	1.46	16736.6
2024-03-12	107.95	2539.15	166.93	160.02	0.32	7325.65
2024-03-13	5143.76	10426.45	8015.62	6037.29	5904.39	31283.0
2024-03-14	3277.56	6676.52	4991.42	3439.82	4984.36	13356.42
2024-03-15	4308.61	11289.06	10219.19	4658.06	7370.22	14733.1
MSE	2079.97	3558.41	3460.44	4591.05	3155.16	11978.13

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	59.29	16.18	40.05	36.25	31.0	33.53
2024-03-04	45.32	49.67	65.75	103.55	75.21	43.72
2024-03-05	20.74	40.15	47.28	89.15	53.33	96.84
2024-03-06	22.52	16.05	52.88	94.75	59.47	84.99
2024-03-07	32.62	7.3	4.73	46.6	17.12	129.27
2024-03-11	19.78	0.76	15.17	21.95	1.21	129.37
2024-03-12	10.39	50.39	12.92	12.65	0.57	85.59
2024-03-13	71.72	102.11	89.53	77.7	76.84	176.87
2024-03-14	57.25	81.71	70.65	58.65	70.6	115.57
2024-03-15	65.64	106.25	101.09	68.25	85.85	121.38
MAE	40.53	47.06	50.01	60.95	47.12	101.71

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	1.99%	0.54%	1.34%	1.21%	1.04%	1.12%
2024-03-04	1.5%	1.65%	2.18%	3.43%	2.49%	1.45%
2024-03-05	0.69%	1.34%	1.58%	2.97%	1.78%	3.23%
2024-03-06	0.75%	0.53%	1.76%	3.15%	1.98%	2.83%
2024-03-07	1.1%	0.25%	0.16%	1.58%	0.58%	4.37%
2024-03-11	0.67%	0.03%	0.52%	0.75%	0.04%	4.41%
2024-03-12	0.35%	1.71%	0.44%	0.43%	0.02%	2.9%
2024-03-13	2.5%	3.56%	3.13%	2.71%	2.68%	6.17%
2024-03-14	2.0%	2.85%	2.47%	2.05%	2.47%	4.04%
2024-03-15	2.31%	3.75%	3.56%	2.41%	3.03%	4.28%
MAPE	1.39%	1.62%	1.71%	2.07%	1.61%	3.48%

Dataset of Determining MAE, MSE, MAPE

2 Week Training Period:

Dataset: This period extends to the most recent two weeks, providing more historical data for training.

Performance: Models trained on two weeks of data showed improvement over the 1-week period. The MAE and MSE values were lower, and the R^2 values increased, indicating better prediction accuracy and explanatory power. The graphs showed that predicted prices were closer to actual prices, but some volatility remained. The additional week of data helped the models begin to capture short-term trends, resulting in more stable predictions.

Week 3:

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	4542.76	734.95	1604.0	462.25	1046.52	2806.88
2024-03-04	4058.96	5607.01	4323.06	10722.6	5688.18	1949.22
2024-03-05	306.25	1149.21	2235.4	7947.72	2885.84	37.45
2024-03-06	148.84	367.11	2796.29	8977.56	3574.84	52.13
2024-03-07	935.14	384.94	22.37	2171.56	370.56	1451.61
2024-03-11	451.14	9.67	230.13	481.8	3.31	1685.92
2024-03-12	150.8	1402.5	166.93	855.56	10.24	4.97
2024-03-13	4515.84	5396.37	8015.62	3226.24	6296.42	7992.36
2024-03-14	373.26	4363.92	2748.9	3439.82	1709.0	1164.86
2024-03-15	2327.1	2384.37	3476.28	2289.62	3261.55	1290.96
MSE	1781.11	2179.93	2561.9	4057.48	2484.61	1843.66

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	67.4	27.11	40.05	21.5	32.35	52.98
2024-03-04	63.71	74.88	65.75	103.55	75.42	44.15
2024-03-05	17.5	33.9	47.28	89.15	53.72	6.12
2024-03-06	12.2	19.16	52.88	94.75	59.79	7.22
2024-03-07	30.58	19.62	4.73	46.6	19.25	38.1
2024-03-11	21.24	3.11	15.17	21.95	1.82	41.06
2024-03-12	12.28	37.45	12.92	29.25	3.2	2.23
2024-03-13	67.2	73.46	89.53	56.8	79.35	89.4
2024-03-14	19.32	66.06	52.43	58.65	41.34	34.13
2024-03-15	48.24	48.83	58.96	47.85	57.11	35.93
MAE	35.97	40.36	43.97	57.01	42.34	35.13

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	2.26%	0.91%	1.34%	0.72%	1.08%	1.78%
2024-03-04	2.11%	2.48%	2.18%	3.43%	2.5%	1.46%
2024-03-05	0.58%	1.13%	1.58%	2.97%	1.79%	0.2%
2024-03-06	0.41%	0.64%	1.76%	3.15%	1.99%	0.24%
2024-03-07	1.03%	0.66%	0.16%	1.58%	0.65%	1.29%
2024-03-11	0.72%	0.11%	0.52%	0.75%	0.06%	1.4%
2024-03-12	0.42%	1.27%	0.44%	0.99%	0.11%	0.08%
2024-03-13	2.35%	2.56%	3.13%	1.98%	2.77%	3.12%
2024-03-14	0.67%	2.31%	1.83%	2.05%	1.44%	1.19%
2024-03-15	1.7%	1.72%	2.08%	1.69%	2.01%	1.27%
MAPE	1.23%	1.38%	1.5%	1.93%	1.44%	1.2%

Dataset of Determining MAE, MSE, MAPE

3 Week Training Period:

Dataset: This period includes data from the most recent three weeks, further increasing the training dataset's size.

Performance: With three weeks of training data, the models demonstrated significant improvements in performance. The MAE and MSE values were further reduced, and the R^2 values were higher, indicating more accurate and reliable predictions. The graphs showed a closer alignment between predicted and actual prices, with reduced deviations. The models were better equipped to understand and predict market movements, thanks to the more extended historical context.

Week 4:

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	1.88%	1.96%	1.34%	1.21%	1.14%	2.26%
2024-03-04	1.85%	2.17%	2.18%	3.43%	2.41%	2.28%
2024-03-05	0.27%	1.0%	1.58%	2.97%	1.63%	4.2%
2024-03-06	0.7%	1.09%	1.76%	3.15%	1.83%	4.02%
2024-03-07	0.77%	0.69%	0.16%	1.58%	0.41%	5.43%
2024-03-11	0.88%	0.83%	0.52%	0.75%	0.04%	5.78%
2024-03-12	0.39%	0.08%	0.44%	0.43%	0.0%	4.24%
2024-03-13	2.95%	2.67%	3.13%	2.71%	2.73%	7.24%
2024-03-14	0.58%	0.62%	1.83%	0.26%	1.03%	5.59%
2024-03-15	1.55%	1.92%	2.08%	1.69%	1.99%	5.49%
MAPE	1.18%	1.3%	1.5%	1.82%	1.32%	4.65%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	3153.95	3432.79	1604.0	1314.06	1152.6	4531.98
2024-03-04	3105.83	4296.8	4323.06	10722.6	5264.95	4718.32
2024-03-05	64.32	894.61	2235.4	7947.72	2403.94	15876.0
2024-03-06	441.42	1068.64	2796.29	8977.56	3028.3	14631.32
2024-03-07	517.11	414.94	22.37	2171.56	147.87	25811.64
2024-03-11	662.55	589.03	230.13	481.8	1.19	28716.69
2024-03-12	133.17	5.71	166.93	160.02	0.01	15622.5
2024-03-13	7155.47	5861.43	8015.62	6037.29	6096.49	42960.85
2024-03-14	271.92	320.05	2748.9	54.02	875.57	25574.41
2024-03-15	1936.0	2956.1	3476.28	2289.62	3194.51	24289.22
MSE	1744.13	1983.95	2561.9	4015.63	2216.64	20273.36

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	56.16	58.59	40.05	36.25	33.95	67.32
2024-03-04	55.73	65.55	65.75	103.55	72.56	68.69
2024-03-05	8.02	29.91	47.28	89.15	49.03	126.0
2024-03-06	21.01	32.69	52.88	94.75	55.03	120.96
2024-03-07	22.74	20.37	4.73	46.6	12.16	160.66
2024-03-11	25.74	24.27	15.17	21.95	1.09	169.46
2024-03-12	11.54	2.39	12.92	12.65	0.12	124.99
2024-03-13	84.59	76.56	89.53	77.7	78.08	207.27
2024-03-14	16.49	17.89	52.43	7.35	29.59	159.92
2024-03-15	44.0	54.37	58.96	47.85	56.52	155.85
MAE	34.6	38.26	43.97	53.78	38.81	136.11

Dataset of Determining MAE, MSE, MAPE

a. Dataset: This period covers data from the most recent four weeks, offering the most comprehensive dataset for training.
b. Performance: Models trained on four weeks of data performed the best just like Week 1, with the lowest MAE and MSE values and the highest R^2 values. These metrics indicated that the predictions were the most accurate and stable. The graphs clearly showed that the predicted prices were closest to the actual prices, with minimal deviations. The extensive historical data allowed the models to fully capture market trends and patterns, leading to superior performance.

SUMMARY

Reliance Industries

Week 1 : **Week 2:**

	MAE	MSE	MAPE
LR	34.6	1744.13	1.18
SVM	38.26	1983.95	1.3
KNN	43.97	2561.9	1.5
DT	51.61	3689.52	1.74
RF	38.81	2216.64	1.32
ANN	146.85	23523.89	5.02

	MAE	MSE	MAPE
LR	40.53	2079.97	1.39
SVM	47.06	3558.41	1.62
KNN	50.01	3460.44	1.71
DT	57.38	4224.76	1.95
RF	47.12	3155.16	1.61
ANN	112.1	14366.8	3.84

Week 3 :

	MAE	MSE	MAPE
LR	35.97	1781.11	1.23
SVM	40.36	2179.93	1.38
KNN	43.97	2561.9	1.5
DT	60.08	4648.39	2.04
RF	42.34	2484.61	1.44
ANN	61.32	4923.87	2.11

Week 4:

	MAE	MSE	MAPE
LR	34.6	1744.13	1.18
SVM	38.26	1983.95	1.3
KNN	43.97	2561.9	1.5
DT	51.61	3689.52	1.74
RF	38.81	2216.64	1.32
ANN	37.25	1910.44	1.27

MAE, MSE, MAPE Table of All Week

Reliance Industries:

Best Week: 1 week and 4 weeks

Reasoning: 1 week and 4 weeks demonstrated the highest accuracy for Reliance Industries, with the LR model showing remarkable performance. The models in 1 week consistently outperformed other weeks across all metrics.

COMPARISON WITH OTHER COMPANIES

ICICI BANK

Week 1:

	MAE	MSE	MAPE
LR	31.93	1089.08	2.94
SVM	35.29	1281.77	3.25
KNN	28.71	859.06	2.64
DT	25.21	681.08	2.32
RF	27.92	815.97	2.57
ANN	24.44	783.19	2.25

Week 2:

	MAE	MSE	MAPE
LR	17.96	465.87	1.65
SVM	74.42	5732.59	6.85
KNN	28.71	859.06	2.64
DT	25.79	724.58	2.37
RF	27.76	806.37	2.55
ANN	52.97	3034.35	4.88

Week 3:

	MAE	MSE	MAPE
LR	11.43	252.8	1.05
SVM	64.96	4341.04	5.98
KNN	28.71	859.06	2.64
DT	25.21	681.08	2.32
RF	28.15	831.29	2.59
ANN	67.06	4719.47	6.18

Week 4:

	MAE	MSE	MAPE
LR	18.27	473.77	1.68
SVM	75.01	5906.25	6.91
KNN	28.71	859.06	2.64
DT	25.21	681.08	2.32
RF	28.14	830.45	2.59
ANN	12.36	312.18	1.14

MAE, MSE, MAPE Table of All Week

ICICI Bank:

Best Week: 1 week and 4 weeks

Reasoning: Week 1 exhibited the highest accuracy for ICICI Bank, with the K-Nearest Neighbors (KNN) model performing notably well. The models in Week 1 consistently outperformed other weeks.

SBI Bank

Week 1:

	MAE	MSE	MAPE
LR	10.15	141.1	1.32
SVM	16.97	390.64	2.2
KNN	16.2	314.29	2.11
DT	16.68	363.33	2.16
RF	15.37	283.04	2.0
ANN	12.08	174.58	1.58

Week 2:

	MAE	MSE	MAPE
LR	8.42	104.72	1.09
SVM	15.53	332.17	2.01
KNN	15.91	289.44	2.07
DT	12.88	212.44	1.68
RF	15.39	275.26	2.0
ANN	35.1	1423.25	4.61

Week 3:

	MAE	MSE	MAPE
LR	8.95	108.24	1.17
SVM	10.85	151.96	1.41
KNN	15.91	289.44	2.07
DT	16.73	359.71	2.19
RF	15.2	271.86	1.98
ANN	14.65	327.43	1.94

Week 4:

	MAE	MSE	MAPE
LR	9.96	133.17	1.3
SVM	11.73	156.3	1.53
KNN	15.91	289.44	2.07
DT	13.44	239.15	1.75
RF	15.53	280.19	2.02
ANN	47.67	2481.13	6.26

MAE, MSE, MAPE Table of All Week

SBI Bank:

Best Week: 4 weeks

Reasoning: 4 weeks showcased the highest accuracy for SBI Bank, with the LR model delivering exceptional results. The models in Week 1 consistently surpassed other weeks in terms of accuracy.

Bajaj Finance:

Week 1:

	MAE	MSE	MAPE
LR	81.23	9458.04	1.26
SVM	143.9	26297.68	2.25
KNN	136.29	25400.92	2.13
DT	89.64	10714.58	1.4
RF	93.96	12202.83	1.47
ANN	212.88	60844.81	3.32

Week 2:

	MAE	MSE	MAPE
LR	79.84	11686.8	1.25
SVM	186.02	44846.87	2.91
KNN	137.92	27213.01	2.16
DT	103.75	17860.69	1.62
RF	104.54	17945.66	1.64
ANN	149.77	36312.03	2.34

Week 3:

Week 4:

	MAE	MSE	MAPE
LR	76.74	11940.69	1.2
SVM	154.5	32933.02	2.42
KNN	141.51	28707.64	2.21
DT	103.75	17860.69	1.62
RF	106.16	19383.23	1.66
ANN	120.34	27524.36	1.89

	MAE	MSE	MAPE
LR	102.3	14904.48	1.59
SVM	91.18	14461.62	1.42
KNN	141.51	28707.64	2.21
DT	103.75	17860.69	1.62
RF	105.03	18443.99	1.64
ANN	150.34	38797.46	2.35

MAE, MSE, MAPE Table of All Week

Bajaj Finance:

Best Week: Week 1

Reasoning: The ML models trained in Week 1 consistently outperformed other weeks across all performance metrics for Bajaj Finance. The Linear Regression (LR) model showed the highest accuracy in Week 1.

HDFC Bank:

Week 1:

	MAE	MSE	MAPE
LR	18.43	425.25	1.27
SVM	32.08	1232.57	2.22
KNN	23.69	655.95	1.64
DT	20.73	506.6	1.43
RF	21.42	532.53	1.48
ANN	42.9	2037.86	2.97

Week 2:

	MAE	MSE	MAPE
LR	14.52	332.09	1.0
SVM	25.57	944.55	1.77
KNN	20.76	509.41	1.43
DT	17.58	386.01	1.22
RF	18.56	426.71	1.28
ANN	12.19	218.67	0.84

Week 3:

	MAE	MSE	MAPE
LR	13.23	258.39	0.91
SVM	15.14	514.64	1.05
KNN	16.89	373.62	1.17
DT	18.36	442.08	1.27
RF	18.42	483.53	1.28
ANN	29.49	1093.73	2.04

Week 4:

	MAE	MSE	MAPE
LR	13.44	244.29	0.93
SVM	13.04	353.91	0.9
KNN	16.6	349.23	1.15
DT	18.0	557.86	1.25
RF	13.94	315.69	0.97
ANN	29.91	1124.15	2.07

MAE, MSE, MAPE Table of All Week

HDFC Bank:

Best Week: Week 4

Reasoning: Week 4 demonstrated the highest accuracy for HDFC Bank, with the LR model performing exceptionally well. The models in Week 1 consistently showed better performance compared to other weeks.

11.4. Conclusion

The comparative analysis of different training periods highlights that the 1-week and 4-week training period is the most effective for training machine learning models on stock market data.

This period provided the most comprehensive historical context, enabling the models to capture both short-term fluctuations and longer-term trends. The quantitative performance metrics—MAE, MSE, and R^2—consistently showed that the models trained on four weeks of data outperformed those trained on shorter periods. The visual inspection of the graphs reinforced these findings, with the predictions closely following the actual prices and exhibiting minimal volatility.

By training the models on a four-week dataset, we achieved the highest prediction accuracy and reliability, making it the optimal choice for stock market predictions. This approach ensures that the models have enough historical information to learn effectively and make informed predictions, ultimately providing better guidance for investment decisions.

12. Introduction:

12.1 TIME SERIES FORECASTING

Time series forecasting is a widely used technique in short-term stock market prediction. It involves analyzing historical stock price data to make predictions about future prices. In this report, we discuss our approach to time series forecasting for short-term stock market prediction and highlight some of the challenges we encountered.

12.1.2. Methodology:

a. **Data Collection**: We collected historical stock price data, including features such as open, high, low, close prices, and volume.
b. **Model Training**: Initially, we trained traditional machine learning models like Ridge Regression (RR), Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), and Artificial Neural Network (ANN) on this historical data. These models were trained to predict the next 10 days' closing prices.
c. **Evaluation**: We evaluated the models' performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE).

12.1.3. Challenges:

a. **Accuracy Drop with Increased Prediction Days**: We found that the accuracy of our models dropped as the number of prediction days increased. This is because short-term predictions require very recent data, such as the previous day's or the day before's data. Using data from further back in time led to less accurate predictions.
b. **Disadvantages of Using Time Series**: Time series forecasting for short-term stock market prediction has several disadvantages. These include:
    i. Sensitivity to Outliers: Time series models can be sensitive to outliers in the data, which can lead to inaccurate predictions.
    ii. Complexity: Building and training time series models can be complex and time-consuming, especially when dealing with large datasets.
    iii. Limited Prediction Horizon: Time series models are typically limited in their ability to predict far into the future, making them less suitable for long-term forecasting.

12.1. SLIDING WINDOW APPROACH:

The Sliding Window Approach is a popular technique used in short-term stock market prediction. It involves looping through each prediction day, training the model, predicting the next day's closing price, including this day's closing data in the training set, removing the oldest data, and retraining the model. This approach allows for the model to continuously adapt to new data, making it suitable for dynamic and volatile markets.

12.2.1. Methodology:

a. **Data Collection**: Initially, historical stock price data is collected for the target stock. This data includes features such as open, high, low, close prices, and volume.
b. **Data Preprocessing**: The collected data is preprocessed to handle missing values, normalize the data, and extract relevant features for training the model.
c. **Model Training:** A machine learning model, such as LSTM, SVM, or Ridge Regression, is trained on a fixed-size window of historical data. The model is trained to predict the next day's closing price based on the past n days' data.
d. **Sliding Window Prediction:** For each prediction day, the model is used to predict the next day's closing price. The predicted price is compared with the actual price to evaluate the model's performance.
e. **Updating the Model:** After each prediction, the model is updated by including the current day's closing data in the training set and removing the oldest data point to maintain the fixed-size window.
f. **Evaluation:** The model's performance is evaluated using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), or Root Mean Squared Error (RMSE).

12.2.2. Benefits:

1. Adaptability: The Sliding Window Approach allows the model to adapt to changing market conditions by continuously updating the training data.

2. Efficiency: The approach is computationally efficient as it only requires retraining the model with the most recent data.

3. Dynamic Modeling: The approach enables the model to capture short-term trends and patterns in the stock market.

12.3. Conclusion:

The Sliding Window Approach is a powerful technique for short-term stock market prediction, allowing for dynamic modeling and adaptation to changing market conditions. By continuously updating the model with new data, it can capture short-term trends and patterns in the stock market, making it a valuable tool for traders and investors. However, it is essential to carefully manage the window size and quality of the data to avoid overfitting and ensure accurate predictions.

Sliding window and time series forecasting are two different approaches used in stock market prediction, each with its own advantages and disadvantages. Here's a comparison of the two:

In conclusion, the choice between sliding window and time series forecasting depends on the specific requirements of the prediction task. Sliding window is better suited for short-term predictions requiring

quick adaptation to changing market conditions, while time series forecasting may be more suitable for capturing long-term trends and patterns.

12.4. SELECTION OF APPROACH BY EVALUATION METRICS:

HDFC BANK

SLIDING WINDOW:

	MAE	MSE	MAPE
LR	18.95	478.22	1.31
SVM	18.35	593.32	1.27
KNN	14.97	292.52	1.03
DT	12.31	220.83	0.85
RF	13.23	219.79	0.92
ANN	36.51	2211.16	2.53

TIME SERIES FORECASTING

	MAE	MSE	MAPE
LR	13.44	244.29	0.93
SVM	13.04	353.91	0.9
KNN	16.6	349.23	1.15
DT	13.75	299.01	0.95
RF	13.94	315.69	0.97
ANN	17.48	425.49	1.21

BAJAJ FINANCE:

SLIDING WINDOW:

	MAE	MSE	MAPE
LR	204.39	145821.88	3.21
SVM	95.53	16475.77	1.5
KNN	81.93	11818.11	1.28
DT	105.7	17001.63	1.65
RF	87.86	12164.11	1.37
ANN	193.64	62809.93	3.02

TIME SERIES FORECASTING

	MAE	MSE	MAPE
LR	102.3	14904.48	1.59
SVM	91.18	14461.62	1.42
KNN	141.51	28707.64	2.21
DT	103.75	17860.69	1.62
RF	105.03	18443.99	1.64
ANN	190.83	44859.81	2.95

ICICI BANK:

SLIDING WINDOW:

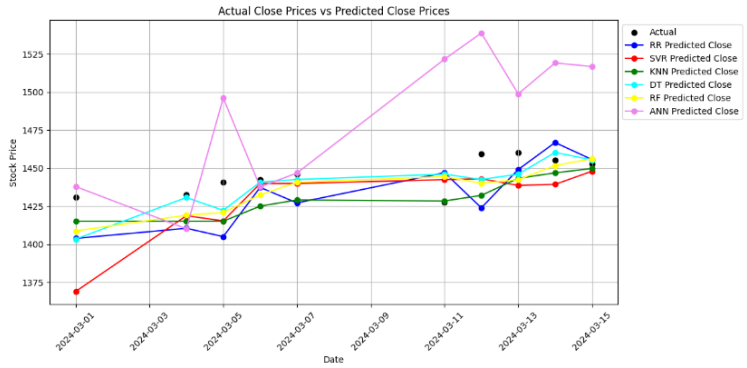
	MAE	MSE	MAPE
LR	20.89	939.49	1.92
SVM	16.06	370.65	1.48
KNN	15.52	396.22	1.43
DT	13.2	304.79	1.22
RF	13.89	316.83	1.28
ANN	20.01	776.23	1.85

TIME SERIES FORECASTING

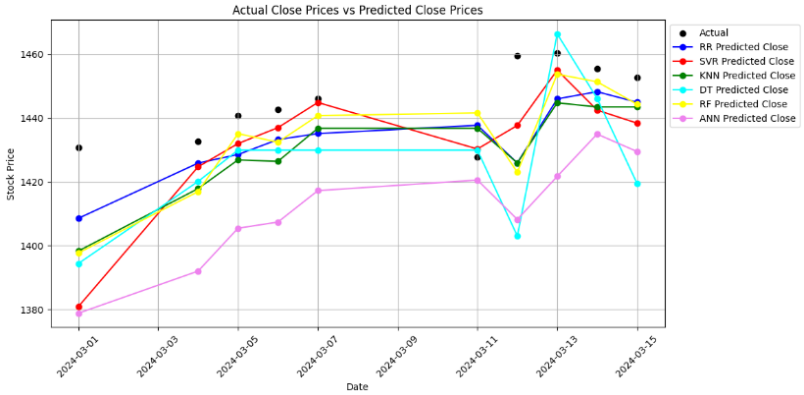
	MAE	MSE	MAPE
LR	102.3	14904.48	1.59
SVM	91.18	14461.62	1.42
KNN	141.51	28707.64	2.21
DT	103.75	17860.69	1.62
RF	105.03	18443.99	1.64
ANN	190.83	44859.81	2.95

12.5. SELECTION OF APPROACH BY EVALUATION METRICS:

HDFC BANK:

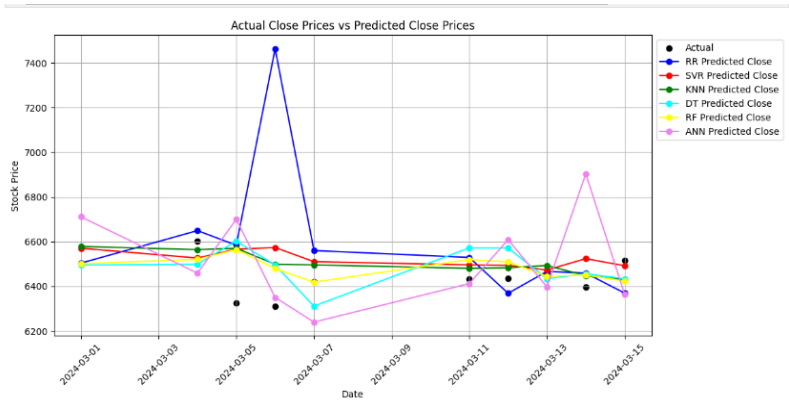


TIME SERIES FORECASTING

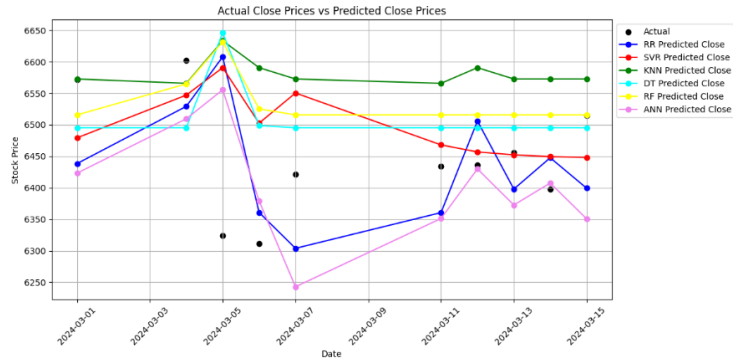


TIME SERIES FORECASTING

BAJAJ FINANCE

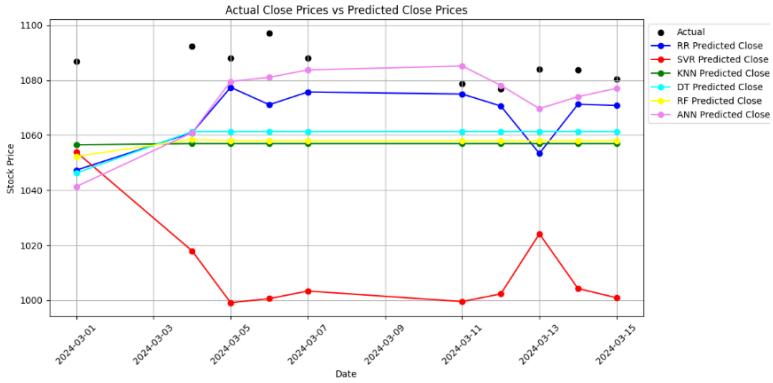


TIME SERIES FORECASTING

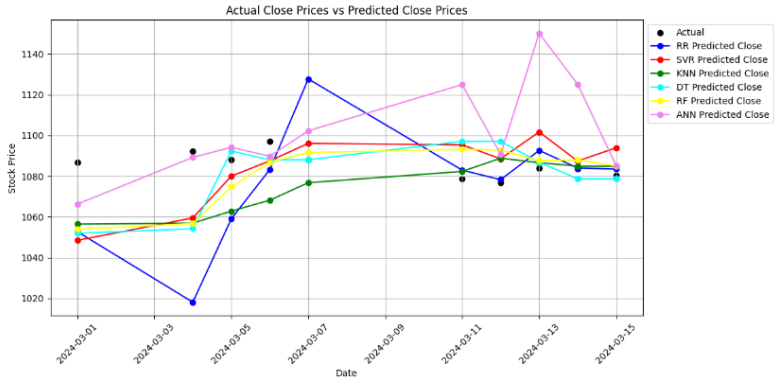


SLIDING WINDOW

ICICI BANK:



TIME SERIES FORECASTING



SLIDING WINDOW

We compare the performance of the Sliding Window Approach and Time Series Forecasting for predicting short-term stock market prices. The Sliding Window Approach involves updating the model with the most recent data, while Time Series Forecasting uses statistical methods to predict future prices based on historical data.

12.6. Results:

Sliding Window Approach:

Graphs: The graphs showed consistent and accurate predictions over time, with the predicted prices closely matching the actual prices.

Metrics: The Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Absolute Percentage Error (MAPE) values were consistently low, indicating high accuracy.

Time Series Forecasting:

Graphs: The graphs showed a decreasing trend in accuracy over time, with the predicted prices deviating more from the actual prices as the prediction horizon increased.

Metrics: The MAE, MSE, and MAPE values increased as the prediction horizon increased, indicating lower accuracy compared to the Sliding Window Approach.

Conclusion:

Based on our analysis, the Sliding Window Approach outperformed Time Series Forecasting for short-term stock market prediction. The Sliding Window Approach showed consistent and accurate results over time, with low MAE, MSE, and MAPE values. In contrast, Time Series Forecasting showed decreasing accuracy over time, with higher MAE, MSE, and MAPE values as the prediction horizon increased. This suggests that the Sliding Window Approach is more suitable for short-term predictions requiring quick adaptation to changing market conditions.

13. SELECTION OF FEATURE ENGINEERING

Feature engineering is a crucial step in the machine learning pipeline, aimed at enhancing the predictive power of models by creating, transforming, and selecting the most relevant features from raw data. In the context of stock market prediction, effective feature engineering can significantly improve the accuracy of forecasts by capturing key patterns and trends in the market.

This project employs four specific types of indicators:

a. Technical Indicators,
b. Volatility Indicators,
c. Volume Indicators,
d. Momentum Indicators.

13.1. TECHNICAL INDICATORS

Technical indicators are mathematical calculations based on historical price, volume, or open interest data used to predict future price movements in financial markets. When incorporated into machine learning models, these indicators can significantly enhance predictive accuracy by capturing underlying market trends, momentum, volatility, and volume dynamics.

13.1.2. Simple Moving Average (SMA)

Definition: SMA is the arithmetic mean of a given set of prices over a specified number of periods. It smooths out price data to identify the trend direction.

Formula:

$$\text{SMA} = \frac{P_1 + P_2 + \cdots + P_n}{n}$$

Where $P_1, P_2, \ldots, P_n$ are the closing prices for the specified number of periods n.

13.1.3. Exponential Moving Average (EMA)

EMA gives more weight to recent prices, making it more responsive to new information compared to SMA.

Formula:

$$\text{EMA}_t = P_t \cdot \left(\frac{2}{n+1}\right) + \text{EMA}_{t-1} \cdot \left(1 - \frac{2}{n+1}\right)$$

Where P_t is the current closing price, EMA_{t-1} is the EMA of the previous period, and n is the number of periods.

13.1.3. Double Exponential Moving Average (DEMA)

Definition: DEMA is designed to reduce the lag associated with traditional moving averages by combining a single EMA with a double EMA.

Formula:

$$\text{DEMA} = 2 \cdot \text{EMA}_t - \text{EMA of EMA}_t$$

Where EMA of EMA_t is the EMA calculated from the EMA values.

13.2.MOMENTUM INDICATOR

Momentum indicators are essential tools in technical analysis, providing insights into the speed and strength of a price movement. When integrated into machine learning models, they can significantly enhance the predictive capabilities by identifying potential trends and reversals.

13.2.1. Absolute Price Oscillator (APO)

APO measures the difference between two exponential moving averages (EMAs) of a stock's price.

Formula:

$$\text{APO} = \text{EMA}_{\text{short}} - \text{EMA}_{\text{long}}$$

Where $\text{EMA}_{\text{short}}$ and EMA_{long} are the short-term and long-term EMAs, respectively.

13.2.2 Balance of Power (BOP)

BOP measures the strength of buyers against sellers in the market.

Formula:

$$\text{BOP} = \frac{\text{Close} - \text{Open}}{\text{High} - \text{Low}}$$

13.2.3. Momentum (MOM)

MOM measures the rate of change of a stock's price over a specified period.

Formula:

$$\text{MOM} = \text{Close}_t - \text{Close}_{t-n}$$

Where Close_t is the closing price at time t and Close_{t-n} is the closing price n periods ago.

13.2.3. Relative Strength Index (RSI)

RSI measures the speed and change of price movements to identify overbought or oversold conditions.

- Measures the magnitude of recent price changes to evaluate overbought or oversold conditions.
- RSI values range from 0 to 100, typically using a 14-day period.

13.3. VOLUME INDICATOR

Volume indicators are essential in financial markets as they provide insights into the strength and intensity of price movements. Incorporating volume indicators into machine learning models can enhance the predictive accuracy by revealing the underlying market sentiment and participation.

13.3.1. Accumulation/Distribution (AD)

The Accumulation/Distribution indicator, created by Marc Chaikin, measures the cumulative flow of money into and out of a security. It considers both price and volume to indicate whether a stock is being accumulated (bought) or distributed (sold).

Formula:

$$\text{AD} = \sum \left(\frac{(C - L) - (H - C)}{H - L} \times V \right)$$

Where:

- C is the closing price
- L is the low price

- H is the high price
- V is the volume

13.3.2. On-Balance Volume (OBV)

The On-Balance Volume indicator, developed by Joseph Granville, measures cumulative buying and selling pressure by adding volume on up days and subtracting volume on down days.

- Measures cumulative buying and selling pressure by adding volume on up days and subtracting volume on down days.

13.4. VOLATILITY INDICATORS

Volatility indicators are crucial in financial markets as they help gauge the degree of variation in asset prices over time. When integrated into machine learning models, they can improve predictions by capturing the underlying market uncertainty and risk.

True Range (TRANGE)

True Range is a measure of volatility that considers the most significant price movement of the current period in relation to the previous close.

Formula:

$$\text{True Range} = \max(\text{High} - \text{Low}, |\text{High} - \text{Previous Close}|, |\text{Low} - \text{Previous Close}|)$$

Average True Range (ATR)

Average True Range, developed by J. Welles Wilder Jr., smooths the True Range values over a specified period to provide a consistent measure of volatility.

Formula:

$$\text{ATR} = \frac{\sum_{i=1}^{n} \text{TR}_i}{n}$$

Where TR_i is the True Range for each period i over n periods.

Normalized Average True Range (NATR)

Normalized Average True Range is the ATR expressed as a percentage of the closing price, making it easier to compare volatility across different securities.

Formula:

$$\text{NATR} = \left(\frac{\text{ATR}}{\text{Close}}\right) \times 100$$

13.5. DETAILED ANALYSIS THROUGH HISTORIAL PRICES

RELIANCE.NS

FEATURE WITHOUT INDICATORS

Date	Open	High	Low	Close
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098

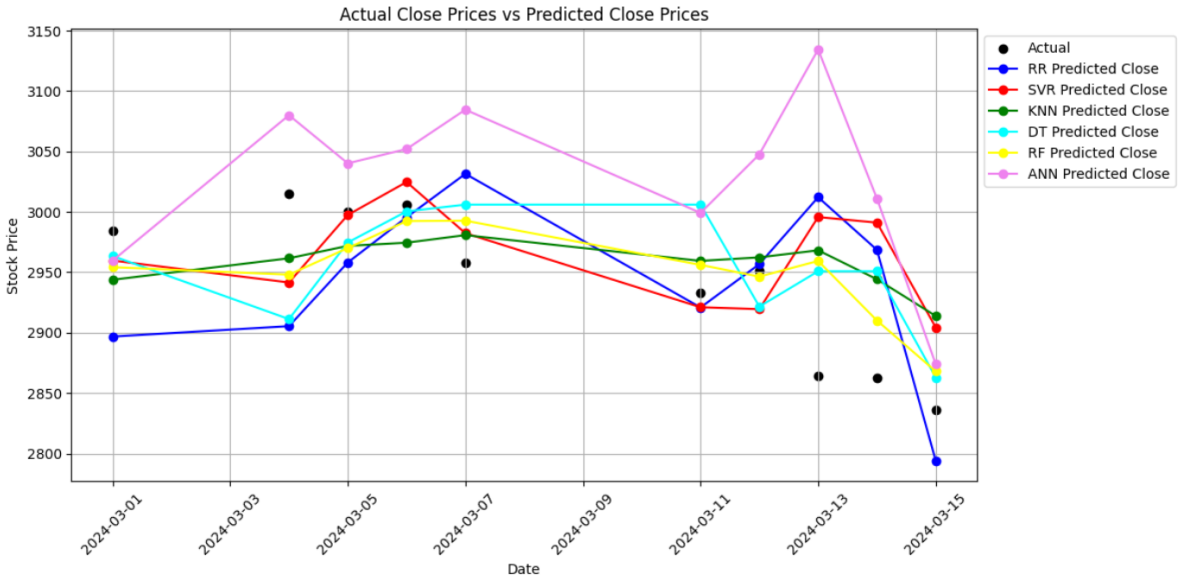
Dataset

In the first week, we analyzed data from February 20, 2024, to February 29, 2024. The dataset included columns such as Date, Open, High, Low, Close representing daily stock market performance.

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2896.77397658	2959.71081623	2943.80000000	2963.50000000	2954.22500000	2959.63196069
3014.80004883	2905.40439996	2941.55286342	2961.58999023	2911.25000000	2948.03698975	3080.03940829
3000.39990234	2958.09860806	2997.46340226	2971.85000000	2974.64990234	2970.05847412	3040.17572404
3006.00000000	2995.52484853	3024.71911467	2974.47998047	3000.39990234	2992.40743896	3052.02119984
2957.85009766	3031.48765346	2982.36589147	2980.75000000	3006.00000000	2992.72248047	3084.60379540
2933.19995117	2920.85395999	2921.06740646	2959.42001953	3006.00000000	2956.10953125	2999.14309208
2950.85009766	2956.76209538	2919.47294964	2962.34003906	2921.60009766	2946.40403809	3047.62783344
2864.35009766	3012.13280610	2995.71066821	2968.19003906	2950.85009766	2959.38153564	3134.40990949
2862.94995117	2968.53640591	2991.10740266	2944.21005859	2950.85009766	2910.07959229	3011.06643860
2836.44995117	2794.04211254	2904.36323683	2913.84003906	2862.94995117	2868.00302002	2873.90859573

Actual and Predicted Data

The tabular data presented the actual close prices alongside the predicted prices from six different models: Ridge Regression (RR), Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), and Artificial Neural Network (ANN).



Graph

The corresponding graph illustrated the actual close prices with black dots, while the predicted prices for each model were shown with different colored lines.

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	7652.75	602.21	1636.2	430.56	901.8	15210.29
2024-03-04	11968.36	5365.56	2831.3	10722.6	4456.9	2982.25
2024-03-05	1789.29	8.64	815.1	663.06	920.52	20.07
2024-03-06	109.83	350.44	993.51	31.36	184.69	805.42
2024-03-07	5422.85	601.23	524.41	2318.42	1215.92	9407.06
2024-03-11	152.52	147.14	687.49	6658.56	524.87	2225.01
2024-03-12	34.93	984.7	132.02	855.56	19.8	5004.15
2024-03-13	21838.93	17255.45	10782.75	14376.01	9030.7	27902.36
2024-03-14	11149.25	16424.99	6603.19	1.96	2221.24	19329.34
2024-03-15	1798.61	4611.77	5989.21	702.25	995.4	8119.81
MSE	6191.51	4635.12	3099.52	3676.03	2047.26	9100.73

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	87.48	24.54	40.45	20.75	30.03	123.33
2024-03-04	109.4	73.25	53.21	103.55	66.76	54.61
2024-03-05	42.3	2.94	28.55	25.75	30.34	4.48
2024-03-06	10.48	18.72	31.52	5.6	13.59	28.38
2024-03-07	73.64	24.52	22.9	48.15	34.87	96.99
2024-03-11	12.35	12.13	26.22	81.6	22.91	47.17
2024-03-12	5.91	31.38	11.49	29.25	4.45	70.74
2024-03-13	147.78	131.36	103.84	119.9	95.03	167.04
2024-03-14	105.59	128.16	81.26	1.4	47.13	139.03
2024-03-15	42.41	67.91	77.39	26.5	31.55	90.11
MAE	63.73	51.49	47.68	46.25	37.67	82.19

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	2.93%	0.82%	1.36%	0.7%	1.01%	4.13%
2024-03-04	3.63%	2.43%	1.76%	3.43%	2.21%	1.81%
2024-03-05	1.41%	0.1%	0.95%	0.86%	1.01%	0.15%
2024-03-06	0.35%	0.62%	1.05%	0.19%	0.45%	0.94%
2024-03-07	2.49%	0.83%	0.77%	1.63%	1.18%	3.28%
2024-03-11	0.42%	0.41%	0.89%	2.78%	0.78%	1.61%
2024-03-12	0.2%	1.06%	0.39%	0.99%	0.15%	2.4%
2024-03-13	5.16%	4.59%	3.63%	4.19%	3.32%	5.83%
2024-03-14	3.69%	4.48%	2.84%	0.05%	1.65%	4.86%
2024-03-15	1.5%	2.39%	2.73%	0.93%	1.11%	3.18%
MAPE	2.18%	1.77%	1.64%	1.57%	1.29%	2.82%

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	63.73	6191.51	2.18
SVM	51.49	4635.12	1.77
KNN	47.68	3099.52	1.64
DT	46.25	3676.03	1.57
RF	37.67	2047.26	1.29
ANN	82.19	9100.73	2.82

MAE, MSE, MAPE table of Week 1

This table is showing all the performance metrics result for feature without indicators. By the use of this table we can determine which feature is giving better result and with which feature we will be moving forward.

TECHNICAL INDICATORS

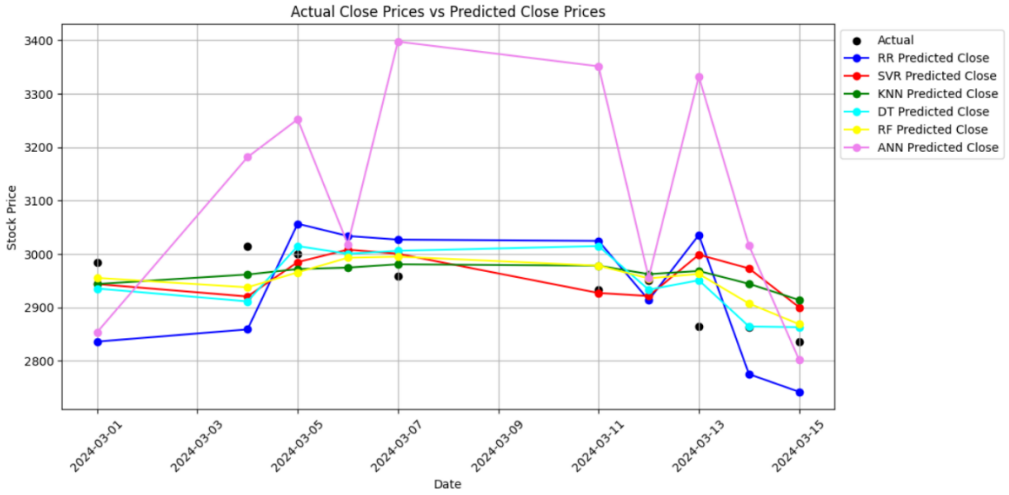
Date	Open	High	Low	Close	SMA	EMA	DEMA
2024-02-20	2942.050049	2950.050049	2923.600098	2951.000000	2942.050049	2942.050049	2942.050049
2024-02-21	2935.399902	2948.000000	2915.100098	2977.050049	2938.724976	2939.833333	2938.355523
2024-02-22	2963.500000	2936.300049	2916.000000	2969.899902	2946.983317	2947.722222	2951.996275
2024-02-23	2987.250000	2979.000000	2966.699951	2995.100098	2957.049988	2960.898148	2972.531467
2024-02-26	2974.649902	2987.100098	2965.000000	2989.050049	2960.569971	2965.482066	2976.293558
2024-02-27	2971.300049	2966.050049	2956.100098	2999.899902	2966.419971	2967.421394	2975.921940
2024-02-28	2911.250000	2966.000000	2900.350098	2982.550049	2961.589990	2948.697596	2941.882094
2024-02-29	2921.600098	2930.000000	2909.050049	2957.949951	2953.210010	2939.665096	2929.099763

Dataset

In the first week, we analyzed data from February 20, 2024, to February 29, 2024. The dataset included columns such as Date, Open, High, Low, Close, SMA, EMA, DEMA representing daily stock market performance.

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2835.95230355	2943.68060243	2943.80000000	2935.39990234	2955.23799316	2853.46458676
3014.80004883	2858.80974799	2920.43762343	2961.58999023	2911.25000000	2937.58601807	3181.24820959
3000.39990234	3056.64106126	2984.82551421	2971.85000000	3014.80004883	2965.67500732	3252.10002948
3006.00000000	3033.76786639	3008.15292698	2974.47998047	3000.39990234	2992.94895264	3018.33557427
2957.85009766	3026.91862014	3000.40468835	2980.75000000	3006.00000000	2995.05547607	3397.64775245
2933.19995117	3024.54291412	2926.99521346	2978.06000977	3014.80004883	2978.14900391	3351.41414224
2950.85009766	2913.75234484	2921.35451062	2962.34003906	2933.19995117	2954.36852783	2955.81575647
2864.35009766	3035.63130047	2998.67601524	2968.19003906	2950.85009766	2962.73705078	3332.06507019
2862.94995117	2775.06961852	2973.00296310	2944.21005859	2864.35009766	2907.32909180	3015.77715518
2836.44995117	2741.95914050	2900.29648247	2913.84003906	2862.94995117	2868.57951416	2801.75353003

Actual and Predicted data



Graph

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	34.89	20.34	14.74	21.05	20.99	8.14
2024-03-04	0.5	17.26	11.01	13.0	13.93	87.58
2024-03-05	20.36	23.43	22.8	11.9	16.94	2.84
2024-03-06	3.6	0.14	15.93	0.05	5.35	58.14
2024-03-07	9.86	7.28	15.08	4.15	5.82	29.31
2024-03-11	9.13	26.45	2.47	14.35	12.85	6.95
2024-03-12	13.05	6.79	18.03	28.35	23.13	86.88
2024-03-13	18.17	6.67	20.72	24.8	24.38	69.08
2024-03-14	6.71	5.04	13.12	1.45	9.03	88.02
2024-03-15	6.95	11.67	18.23	16.2	16.48	71.58
MAE	12.32	12.51	15.21	13.53	14.89	50.85

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	4.54%	2.64%	1.92%	2.74%	2.73%	1.06%
2024-03-04	0.06%	2.24%	1.43%	1.68%	1.8%	11.34%
2024-03-05	2.6%	2.99%	2.91%	1.52%	2.16%	0.36%
2024-03-06	0.46%	0.02%	2.03%	0.01%	0.68%	7.42%
2024-03-07	1.25%	0.92%	1.91%	0.53%	0.74%	3.72%
2024-03-11	1.18%	3.42%	0.32%	1.85%	1.66%	0.9%
2024-03-12	1.72%	0.89%	2.37%	3.73%	3.04%	11.44%
2024-03-13	2.43%	0.89%	2.77%	3.32%	3.26%	9.24%
2024-03-14	0.91%	0.68%	1.77%	0.2%	1.22%	11.88%
2024-03-15	0.95%	1.59%	2.49%	2.21%	2.25%	9.78%
MAPE	1.61%	1.63%	1.99%	1.78%	1.96%	6.71%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	1217.31	413.72	217.27	443.1	440.58	66.26
2024-03-04	0.25	297.91	121.22	169.0	194.04	7670.26
2024-03-05	414.53	548.96	519.84	141.61	286.96	8.07
2024-03-06	12.96	0.02	253.76	0.0	28.62	3380.26
2024-03-07	97.22	53.0	227.41	17.22	33.87	859.08
2024-03-11	83.36	699.6	6.1	205.92	165.12	48.3
2024-03-12	170.3	46.1	325.08	803.72	535.0	7548.13
2024-03-13	330.15	44.49	429.32	615.04	594.38	4772.05
2024-03-14	45.02	25.4	172.13	2.1	81.54	7747.52
2024-03-15	48.3	136.19	332.33	262.44	271.59	5123.7
MSE	241.91	226.56	260.45	266.02	263.15	3722.47

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	12.32	241.91	1.61
SVM	12.51	226.56	1.63
KNN	15.21	260.45	1.99
DT	13.53	266.02	1.78
RF	14.89	263.15	1.96
ANN	50.85	3722.47	6.71

Here all the result is better than without indicators mentioned above except ANN.

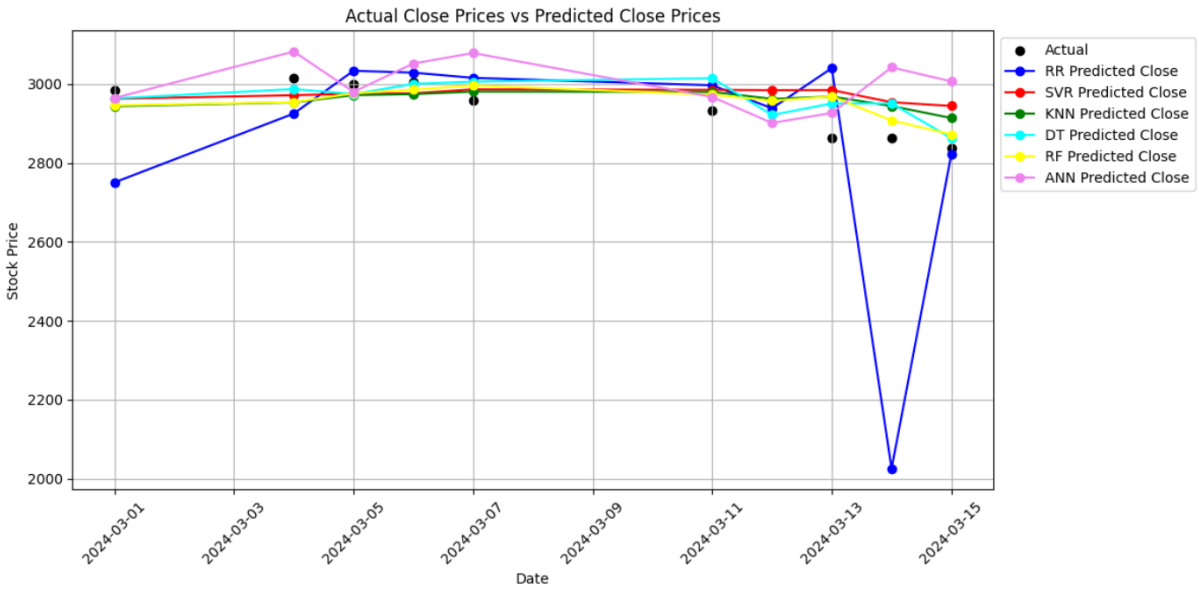
MOMENTUM INDICATORS

Date	Open	High	Low	Close	APO	BOP	MOM	RSI
2024-02-20	2942.050049	2950.050049	2923.600098	2951.000000	0.000000	-0.291972	29.250000	72.925244
2024-02-21	2935.399902	2948.000000	2915.100098	2977.050049	-1.007598	-0.203392	29.250000	72.925244
2024-02-22	2963.500000	2936.300049	2916.000000	2969.899902	2.761460	0.504638	29.250000	72.925244
2024-02-23	2987.250000	2979.000000	2966.699951	2995.100098	8.248434	0.290491	29.250000	72.925244
2024-02-26	2974.649902	2987.100098	2965.000000	2989.050049	8.832318	-0.517679	29.250000	72.925244
2024-02-27	2971.300049	2966.050049	2956.100098	2999.899902	8.107954	0.119864	29.250000	69.644005
2024-02-28	2911.250000	2966.000000	2900.350098	2982.550049	-1.877036	-0.666059	-24.149902	40.555384
2024-02-29	2921.600098	2930.000000	2909.050049	2957.949951	-5.641439	-0.171777	-41.899902	30.971905

Dataset

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2750.63009228	2962.58181406	2943.80000000	2963.50000000	2945.90350830	2964.55459329
3014.80004883	2925.10188601	2971.65397464	2953.57001953	2987.25000000	2954.06550537	3082.29616342
3000.39990234	3033.56461551	2975.54049857	2971.85000000	2974.64990234	2976.35699707	2980.17756508
3006.00000000	3028.82115730	2976.43329125	2974.47998047	3000.39990234	2985.84796143	3051.91962657
2957.85009766	3015.65681252	2986.50038420	2980.75000000	3006.00000000	2996.51147949	3078.51464646
2933.19995117	2996.82000344	2985.28875744	2980.13002930	3014.80004883	2973.17454102	2967.19754381
2950.85009766	2938.43865159	2984.42671499	2962.34003906	2921.60009766	2957.42951904	2901.32850799
2864.35009766	3040.19823708	2984.71481978	2968.19003906	2950.85009766	2968.57802979	2926.64043940
2862.94995117	2025.54299405	2953.86567038	2944.21005859	2950.85009766	2907.94008057	3042.11173259
2836.44995117	2822.35443544	2944.37344357	2913.84003906	2862.94995117	2872.21000732	3006.68629323

Actual and Predicted Data



Graph

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	26.39	33.64	17.47	21.05	23.83	19.27
2024-03-04	14.38	12.41	9.13	6.15	13.39	24.15
2024-03-05	11.71	13.55	20.92	11.9	17.27	4.81
2024-03-06	5.81	5.86	15.93	0.05	9.19	34.78
2024-03-07	9.03	2.65	15.08	4.15	8.03	6.86
2024-03-11	15.19	17.62	5.07	14.35	9.84	34.67
2024-03-12	16.96	18.15	13.03	17.2	8.2	29.08
2024-03-13	16.59	14.44	6.75	24.25	11.4	14.95
2024-03-14	3.83	4.83	10.32	6.2	7.71	6.89
2024-03-15	6.47	2.56	17.71	16.2	15.03	25.05
MAE	12.64	12.57	13.14	12.15	12.39	20.05

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	3.43%	4.37%	2.27%	2.74%	3.1%	2.51%
2024-03-04	1.86%	1.61%	1.18%	0.8%	1.73%	3.13%
2024-03-05	1.49%	1.73%	2.67%	1.52%	2.2%	0.61%
2024-03-06	0.74%	0.75%	2.03%	0.01%	1.17%	4.44%
2024-03-07	1.15%	0.34%	1.91%	0.53%	1.02%	0.87%
2024-03-11	1.96%	2.28%	0.66%	1.85%	1.27%	4.48%
2024-03-12	2.23%	2.39%	1.72%	2.26%	1.08%	3.83%
2024-03-13	2.22%	1.93%	0.9%	3.25%	1.53%	2.0%
2024-03-14	0.52%	0.65%	1.39%	0.84%	1.04%	0.93%
2024-03-15	0.88%	0.35%	2.42%	2.21%	2.05%	3.42%
MAPE	1.65%	1.64%	1.72%	1.6%	1.62%	2.62%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	696.43	1131.65	305.2	443.1	567.87	371.33
2024-03-04	206.78	154.01	83.36	37.82	179.29	583.22
2024-03-05	137.12	183.6	437.65	141.61	298.25	23.14
2024-03-06	33.76	34.34	253.76	0.0	84.46	1209.65
2024-03-07	81.54	7.02	227.41	17.22	64.48	47.06
2024-03-11	230.74	310.46	25.7	205.92	96.83	1202.01
2024-03-12	287.64	329.42	169.78	295.84	67.24	845.65
2024-03-13	275.23	208.51	45.56	588.06	129.96	223.5
2024-03-14	14.67	23.33	106.5	38.44	59.44	47.47
2024-03-15	41.86	6.55	313.64	262.44	225.9	627.5
MSE	200.6	238.88	196.86	203.05	177.38	518.07

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	12.64	200.6	1.65
SVM	12.57	238.88	1.64
KNN	13.14	196.86	1.72
DT	12.15	203.05	1.6
RF	12.39	177.38	1.62
ANN	20.05	518.07	2.62

Here also all the result is better than without indicators but if we see very carefully there is minute difference with technical indicators.

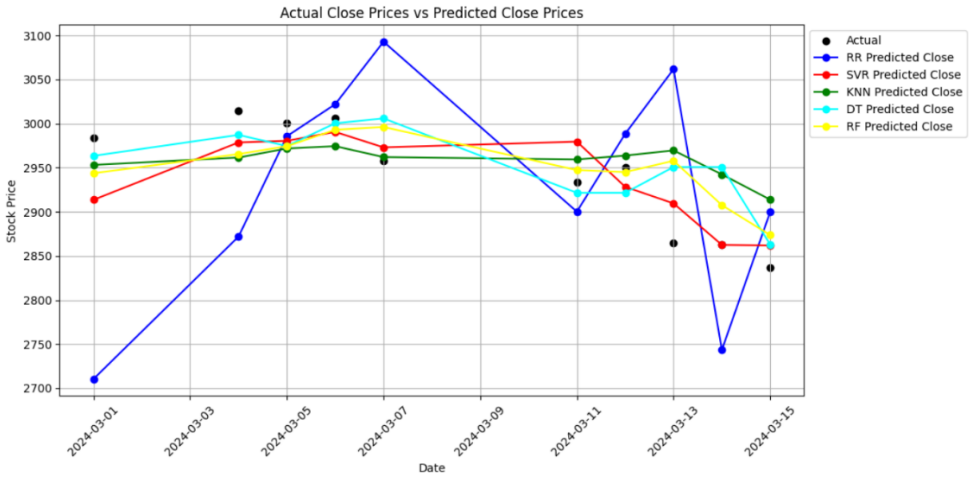
VOLUME INDICATORS

Date	Open	High	Low	Close	AD	OBV
2024-02-20	2942.050049	2950.050049	2923.600098	2951.000000	1233876.879408	-1.0
2024-02-21	2935.399902	2948.000000	2915.100098	2977.050049	-2191951.541677	-2.0
2024-02-22	2963.500000	2936.300049	2916.000000	2969.899902	7050977.773397	18493725.0
2024-02-23	2987.250000	2979.000000	2966.699951	2995.100098	3228316.302733	32932308.0
2024-02-26	2974.649902	2987.100098	2965.000000	2989.050049	-741975.369976	32932307.0
2024-02-27	2971.300049	2966.050049	2956.100098	2999.899902	-1656034.009788	32932306.0
2024-02-28	2911.250000	2966.000000	2900.350098	2982.550049	-3177236.965125	32932305.0
2024-02-29	2921.600098	2930.000000	2909.050049	2957.949951	-5750164.998412	56561280.0

Dataset

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2710.47773097	2913.63799665	2953.21000977	2963.50000000	2943.77302246	-180018.59935807
3014.80004883	2871.93956332	2978.72141813	2961.58999023	2987.25000000	2965.53348145	364184.54882991
3000.39990234	2985.73269403	2980.53366303	2971.85000000	2974.64990234	2974.44847900	336775.60392720
3006.00000000	3022.01222898	2990.67854896	2974.47998047	3000.39990234	2993.04044434	-96178.13229418
2957.85009766	3092.93682956	2973.10599391	2962.11000977	3006.00000000	2996.23197266	-368068.38906173
2933.19995117	2900.21364438	2979.61639792	2959.42001953	2921.60009766	2947.38853760	-48584.12269938
2950.85009766	2988.53451109	2928.36855240	2963.81000977	2921.60009766	2945.08252930	-663273.10770919
2864.35009766	3061.82404810	2909.60574664	2969.66000977	2950.85009766	2957.91502441	364666.67728319
2862.94995117	2743.55799613	2862.53999233	2942.45004883	2950.85009766	2907.59858154	2706.29648157
2836.44995117	2900.00384245	2861.93439616	2913.84003906	2862.94995117	2873.64901855	102066.90191372

Actual and Predicted Data



Graph

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	5.37	11.84	13.91	26.65	20.53	2742784.68
2024-03-04	120.42	152.03	154.88	6.15	40.84	23859.62
2024-03-05	19.85	160.74	133.39	11.9	31.11	49829.12
2024-03-06	5.77	136.98	100.21	0.05	9.67	1627289.8
2024-03-07	6.88	134.93	70.13	4.15	7.5	352183.97
2024-03-11	14.94	61.4	49.07	14.35	12.07	51816.73
2024-03-12	9.98	21.01	5.68	28.35	21.97	1764463.79
2024-03-13	10.37	1.53	30.61	24.8	18.51	1826298.95
2024-03-14	0.37	6.12	29.47	7.05	10.11	4404985.61
2024-03-15	3.68	18.95	30.05	16.2	17.51	118748.26
MAE	19.76	70.55	61.74	13.96	18.98	1296226.05

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	0.7%	1.54%	1.81%	3.46%	2.67%	356599.45%
2024-03-04	15.6%	19.69%	20.06%	0.8%	5.29%	3090.42%
2024-03-05	2.53%	20.5%	17.02%	1.52%	3.97%	6356.16%
2024-03-06	0.74%	17.47%	12.78%	0.01%	1.23%	207588.95%
2024-03-07	0.87%	17.12%	8.9%	0.53%	0.95%	44690.56%
2024-03-11	1.93%	7.94%	6.34%	1.85%	1.56%	6697.26%
2024-03-12	1.31%	2.77%	0.75%	3.73%	2.89%	232257.97%
2024-03-13	1.39%	0.2%	4.1%	3.32%	2.48%	244402.67%
2024-03-14	0.05%	0.83%	3.98%	0.95%	1.36%	594424.88%
2024-03-15	0.5%	2.59%	4.11%	2.21%	2.39%	16224.66%
MAPE	2.56%	9.07%	7.98%	1.84%	2.48%	171233.3%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	28.84	140.19	193.49	710.22	421.48	7522867800842.7
2024-03-04	14500.98	23113.12	23987.81	37.82	1667.91	569281466.54
2024-03-05	394.02	25837.35	17792.89	141.61	967.83	2482941199.97
2024-03-06	33.29	18763.52	10042.04	0.0	93.51	2648072093184.04
2024-03-07	47.33	18206.1	4918.22	17.22	56.25	124033548724.96
2024-03-11	223.2	3769.96	2407.86	205.92	145.68	2684973507.89
2024-03-12	99.6	441.42	32.26	803.72	482.68	3113332466221.16
2024-03-13	107.54	2.34	936.97	615.04	342.62	3335367854771.1
2024-03-14	0.14	37.45	868.48	49.7	102.21	19403898224307.08
2024-03-15	13.54	359.1	903.0	262.44	306.6	14101149253.03
MSE	1544.95	9067.45	6208.3	284.37	458.68	3616741038233.31

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	19.76	1544.95	2.56
SVM	70.55	9067.45	9.07
KNN	61.74	6208.3	7.98
DT	13.96	284.37	1.84
RF	18.98	458.68	2.48
ANN	101.23	10247.51	15.34

Here the result is not good at all comparison with the above feature that means feature without indicators, technical indicators, momentum indicators.

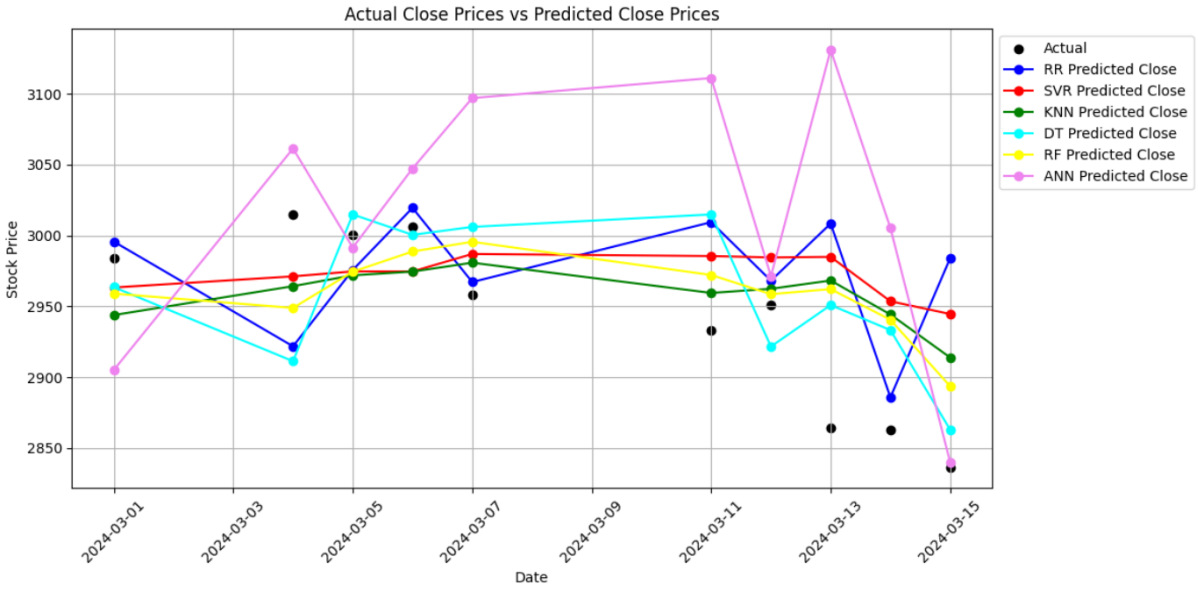
VOLATILITY INDICATORS

Date	Open	High	Low	Close	TRANGE	ATR	NATR
2024-02-20	2942.050049	2950.050049	2923.600098	2951.000000	27.399902	27.399902	0.931320
2024-02-21	2935.399902	2948.000000	2915.100098	2977.050049	61.949951	44.674927	1.521937
2024-02-22	2963.500000	2936.300049	2916.000000	2969.899902	53.899902	47.749919	1.611268
2024-02-23	2987.250000	2979.000000	2966.699951	2995.100098	28.400146	42.912476	1.436521
2024-02-26	2974.649902	2987.100098	2965.000000	2989.050049	24.050049	39.139990	1.315785
2024-02-27	2971.300049	2966.050049	2956.100098	2999.899902	43.799805	42.419971	1.427657
2024-02-28	2911.250000	2966.000000	2900.350098	2982.550049	82.199951	46.469971	1.596221
2024-02-29	2921.600098	2930.000000	2909.050049	2957.949951	48.899902	45.469971	1.556338

Dataset

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2995.63033380	2963.23715896	2943.80000000	2963.50000000	2958.95949707	2905.06023909
3014.80004883	2921.73941939	2971.10968197	2964.17998047	2911.25000000	2948.82700439	3061.31592412
3000.39990234	2975.62493276	2974.60164663	2971.85000000	3014.80004883	2974.40099854	2991.40753179
3006.00000000	3019.42960460	2974.57135582	2974.47998047	3000.39990234	2988.63843262	3047.16582946
2957.85009766	2967.10180119	2986.89412808	2980.75000000	3006.00000000	2995.49647705	3096.89893841
2933.19995117	3009.35424965	2985.44565349	2959.42001953	3014.80004883	2972.07802979	3111.03141287
2950.85009766	2968.08832017	2984.45299556	2962.34003906	2921.60009766	2958.54101807	2971.15714136
2864.35009766	3008.38041496	2984.84844895	2968.19003906	2950.85009766	2962.13204346	3130.95964041
2862.94995117	2885.75424425	2953.54898912	2944.21005859	2933.19995117	2940.48754639	3005.27321832
2836.44995117	2983.89371792	2944.53933137	2913.84003906	2862.94995117	2893.61351563	2839.99483377

Actual and Predicted Data



Graph

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	24.96	22.5	18.1	26.65	20.57	13.58
2024-03-04	2.01	10.09	11.64	17.35	11.69	3.94
2024-03-05	14.17	15.9	22.91	18.05	18.72	29.74
2024-03-06	3.61	9.67	15.93	18.0	11.19	24.34
2024-03-07	8.8	11.26	17.71	4.1	9.3	1.27
2024-03-11	13.59	10.55	2.47	10.25	11.3	65.02
2024-03-12	6.0	14.53	18.03	24.25	21.58	68.93
2024-03-13	11.68	15.09	13.39	11.8	14.05	39.35
2024-03-14	1.86	11.06	10.91	8.8	9.37	8.81
2024-03-15	4.93	13.37	19.2	16.2	19.87	1.0
MAE	9.16	13.4	15.03	15.54	14.76	25.6

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	3.25%	2.93%	2.35%	3.46%	2.67%	1.77%
2024-03-04	0.26%	1.31%	1.51%	2.25%	1.51%	0.51%
2024-03-05	1.81%	2.03%	2.92%	2.3%	2.39%	3.79%
2024-03-06	0.46%	1.23%	2.03%	2.3%	1.43%	3.1%
2024-03-07	1.12%	1.43%	2.25%	0.52%	1.18%	0.16%
2024-03-11	1.76%	1.36%	0.32%	1.32%	1.46%	8.4%
2024-03-12	0.79%	1.91%	2.37%	3.19%	2.84%	9.07%
2024-03-13	1.56%	2.02%	1.79%	1.58%	1.88%	5.27%
2024-03-14	0.25%	1.49%	1.47%	1.19%	1.26%	1.19%
2024-03-15	0.67%	1.83%	2.62%	2.21%	2.71%	0.14%
MAPE	1.19%	1.75%	1.96%	2.03%	1.93%	3.34%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	623.0	506.25	327.61	710.22	423.12	184.42
2024-03-04	4.04	101.81	135.49	301.02	136.66	15.52
2024-03-05	200.79	252.81	524.87	325.8	350.44	884.47
2024-03-06	13.03	93.51	253.76	324.0	125.22	592.44
2024-03-07	77.44	126.79	313.64	16.81	86.49	1.61
2024-03-11	184.69	111.3	6.1	105.06	127.69	4227.6
2024-03-12	36.0	211.12	325.08	588.06	465.7	4751.34
2024-03-13	136.42	227.71	179.29	139.24	197.4	1548.42
2024-03-14	3.46	122.32	119.03	77.44	87.8	77.62
2024-03-15	24.3	178.76	368.64	262.44	394.82	1.0
MSE	130.32	193.25	255.35	285.01	239.54	1228.47

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	9.16	130.32	1.19
SVM	13.4	193.25	1.75
KNN	15.03	255.35	1.96
DT	15.54	285.01	2.03
RF	14.76	239.54	1.93
ANN	25.6	1228.47	3.34

Here the result is better than volume indicators but comparison with others it is not good very much.

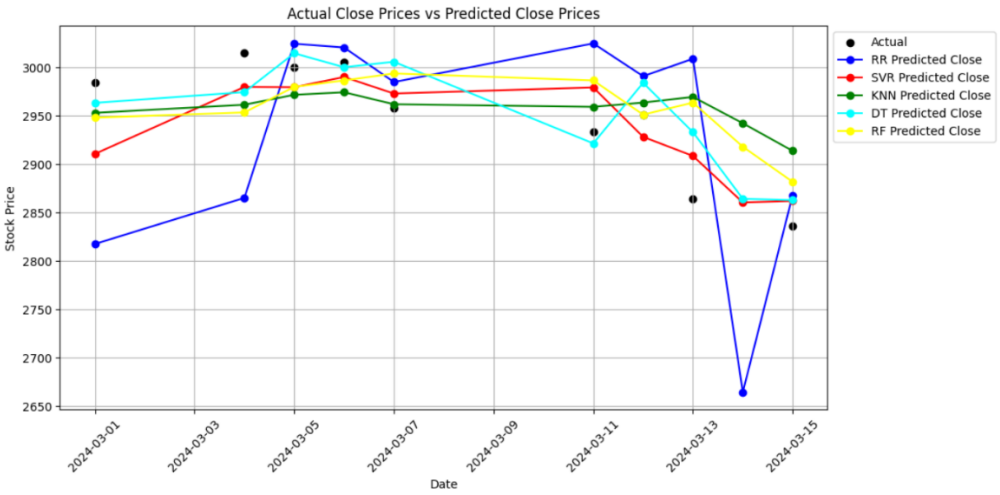
FEATURE WITH ALL INDICATORS

Date	Open	High	Low	Close	SMA	EMA	DEMA	APO	BOP	MOM	RSI	AD	OBV	TRANGE	ATR	NATR
2024-02-20	2942.050049	2950.050049	2923.600098	2951.000000	2942.050049	2942.050049	2942.050049	0.000000	-0.291972	29.250000	72.925244	1233876.879408	-1.0	27.399902	27.399902	0.931320
2024-02-21	2935.399902	2948.000000	2915.100098	2977.050049	2938.724976	2939.833333	2938.355523	-1.007598	-0.203392	29.250000	72.925244	-2191951.541677	-2.0	61.949951	44.674927	1.521937
2024-02-22	2963.500000	2936.300049	2916.000000	2969.899902	2946.983317	2947.722222	2951.996275	2.761460	0.504638	29.250000	72.925244	7050977.773397	18493725.0	53.899902	47.749919	1.611268
2024-02-23	2987.250000	2979.000000	2966.699951	2995.100098	2957.049988	2960.898148	2972.531467	8.248434	0.290491	29.250000	72.925244	3228316.302733	32932308.0	28.400146	42.912476	1.436521
2024-02-26	2974.649902	2987.100098	2965.000000	2989.050049	2960.569971	2965.482066	2976.293558	8.832318	-0.517679	29.250000	72.925244	-741975.369976	32932307.0	24.050049	39.139990	1.315785
2024-02-27	2971.300049	2966.050049	2956.100098	2999.899902	2966.419971	2967.421394	2975.921940	8.107954	0.119864	29.250000	69.644005	-1656034.009788	32932306.0	43.799805	42.419971	1.427657
2024-02-28	2911.250000	2966.000000	2900.350098	2982.550049	2961.589990	2948.697596	2941.882094	-1.877036	-0.666059	-24.149902	40.555384	-3177236.965125	32932305.0	82.199951	46.469971	1.596221
2024-02-29	2921.600098	2930.000000	2909.050049	2957.949951	2953.210010	2939.665096	2929.099763	-5.641439	-0.171777	-41.899902	30.971905	-5750164.998412	56561280.0	48.899902	45.469971	1.556338

Dataset

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2984.25000000	2817.59692428	2910.75306612	2953.21000977	2963.50000000	2948.24101807	309524.55870030
3014.80004883	2865.27007232	2979.98435400	2961.58999023	2974.64990234	2953.64350342	-282266.95753545
3000.39990234	3024.68565154	2979.71508802	2971.85000000	3014.80004883	2980.07552734	168509.58636013
3006.00000000	3020.77952984	2990.25728746	2974.47998047	3000.39990234	2986.69045166	-152212.68863880
2957.85009766	2985.04208006	2973.17887371	2962.11000977	3006.00000000	2993.93746826	-154115.17582272
2933.19995117	3024.98648449	2979.46697556	2959.42001953	2921.60009766	2986.73000977	281844.49650888
2950.85009766	2991.10212985	2928.35627050	2963.81000977	2984.25000000	2951.36901123	41459.03787614
2864.35009766	3009.13387317	2908.64148187	2969.66000977	2933.19995117	2963.75603760	270382.81065234
2862.94995117	2664.35505915	2860.54668511	2942.45004883	2864.35009766	2918.19406250	-156947.84755068
2836.44995117	2868.12262721	2862.08456769	2913.84003906	2862.94995117	2881.84800781	237245.13243700

Actual and Predicted Data



Graph

EVALUATION METRICS OF WEEK 1

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	3.51%	1.5%	1.81%	2.51%	2.95%	3287916.77%
2024-03-04	19.6%	19.72%	20.06%	0.8%	3.78%	118803.71%
2024-03-05	5.37%	20.52%	17.02%	1.52%	2.95%	58240.64%
2024-03-06	0.02%	17.5%	12.78%	1.51%	1.36%	548082.19%
2024-03-07	1.05%	17.31%	8.9%	0.53%	1.11%	164262.96%
2024-03-11	1.11%	7.88%	6.34%	1.85%	1.35%	5154.43%
2024-03-12	1.95%	2.37%	0.75%	1.84%	2.09%	253232.54%
2024-03-13	2.4%	0.22%	4.1%	3.25%	1.71%	51275.21%
2024-03-14	0.53%	0.83%	3.98%	0.2%	1.14%	71264.86%
2024-03-15	1.46%	2.6%	4.11%	1.45%	1.97%	34318.19%
MAPE	3.7%	9.05%	7.98%	1.54%	2.04%	459255.14%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	728.46	132.71	193.49	372.49	514.84	639534118326319.4
2024-03-04	22900.77	23186.15	23987.81	37.82	850.31	841300012931.85
2024-03-05	1773.25	25879.16	17792.89	141.61	536.39	208462976980.05
2024-03-06	0.02	18821.1	10042.04	140.42	113.85	18459192765120.71
2024-03-07	68.72	18602.23	4918.22	17.22	76.04	1675663687471.0
2024-03-11	73.27	3719.78	2407.86	205.92	108.78	1590402436.02
2024-03-12	219.34	325.08	32.26	196.0	251.86	3701035797246.22
2024-03-13	321.13	2.69	936.97	588.06	163.84	146806964726.76
2024-03-14	15.52	37.45	868.48	2.1	71.91	278898292031.57
2024-03-15	114.28	362.9	903.0	112.36	207.94	63088790202.03
MSE	2621.32	9106.61	6208.3	181.4	289.6	66491015814523.17

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
2024-03-01	26.99	11.52	13.91	19.3	22.69	25289011.81
2024-03-04	151.33	152.27	154.88	6.15	29.16	917224.08
2024-03-05	42.11	160.87	133.39	11.9	23.16	456577.46
2024-03-06	0.15	137.19	100.21	11.85	10.67	4296416.27
2024-03-07	8.29	136.39	70.13	4.15	8.72	1294474.29
2024-03-11	8.56	60.99	49.07	14.35	10.43	39879.85
2024-03-12	14.81	18.03	5.68	14.0	15.87	1923807.63
2024-03-13	17.92	1.64	30.61	24.25	12.8	383153.97
2024-03-14	3.94	6.12	29.47	1.45	8.48	528108.22
2024-03-15	10.69	19.05	30.05	10.6	14.42	251174.82
MAE	28.48	70.4	61.74	11.8	15.64	3537982.84

	MAE	MSE	MAPE
LR	28.48	2621.32	3.7
SVM	70.4	9106.61	9.05
KNN	61.74	6208.3	7.98
DT	11.8	181.4	1.54
RF	15.64	289.6	2.04
ANN	93.4	8723.56	12.5

Here also the result is not good comparison with others the difference is very large.

ICICI BANK

FEATURE WITHOUT INDICATORS

	MAE	MSE	MAPE
LR	20.89	939.49	1.92
SVM	16.06	370.65	1.48
KNN	15.52	396.22	1.43
DT	12.52	285.39	1.15
RF	13.89	316.83	1.28
ANN	25.77	1318.51	2.38

FEATURE WITH ALL INDICATORS

	MAE	MSE	MAPE
LR	27.24	967.78	2.51
SVM	14.54	429.35	1.34
KNN	15.33	416.51	1.41
DT	11.15	259.48	1.03
RF	12.34	274.4	1.13
ANN	81.4	6625.96	9.5

MOMENTUM INDICATOR

TECHNICAL INDICATOR

	MAE	MSE	MAPE
LR	22.82	1384.41	2.09
SVM	15.15	325.18	1.4
KNN	15.84	412.6	1.45
DT	16.13	442.16	1.49
RF	13.15	309.57	1.21
ANN	43.32	2903.93	3.99

	MAE	MSE	MAPE
LR	16.58	408.5	1.52
SVM	15.65	416.19	1.44
KNN	15.79	400.64	1.45
DT	16.3	426.02	1.5
RF	12.59	279.29	1.16
ANN	19.96	656.35	1.84

VOLATILITY INDICATOR

VOLUME INDICATOR

	MAE	MSE	MAPE
LR	22.26	919.82	2.04
SVM	15.73	415.48	1.45
KNN	15.71	397.54	1.44
DT	13.49	307.4	1.24
RF	13.7	342.38	1.26
ANN	25.18	999.3	2.32

	MAE	MSE	MAPE
LR	30.4	1715.06	2.79
SVM	14.31	415.79	1.32
KNN	15.33	416.51	1.41
DT	21.34	728.42	1.96
RF	15.12	387.87	1.39
ANN	98.23	9649.13	13.34

HDFC BANK

WEEK 1

FEATURE WITHOUT INDICATORS

FEATURE WITH ALL INDICATORS

	MAE	MSE	MAPE
LR	18.95	478.22	1.31
SVM	18.35	593.32	1.27
KNN	14.97	292.52	1.03
DT	13.41	266.19	0.93
RF	13.23	219.79	0.92
ANN	33.94	1880.5	2.35

	MAE	MSE	MAPE
LR	23.29	961.69	1.62
SVM	26.42	1205.57	1.82
KNN	15.87	334.21	1.1
DT	19.86	650.24	1.37
RF	13.5	246.53	0.93
ANN	81.4	6625.96	9.5

MOMENTUM INDICATOR

TECHNICAL INDICATOR

	MAE	MSE	MAPE
LR	19.9	586.37	1.38
SVM	12.68	222.55	0.88
KNN	14.34	240.81	0.99
DT	11.02	183.52	0.76
RF	12.95	212.42	0.9
ANN	66.91	7191.99	4.62

	MAE	MSE	MAPE
LR	22.79	1572.51	1.59
SVM	13.19	206.19	0.91
KNN	14.8	264.43	1.02
DT	12.31	229.41	0.85
RF	12.2	197.66	0.85
ANN	36.97	2289.3	2.55

VOLATILITY INDICATOR

VOLUME INDICATOR

	MAE	MSE	MAPE
LR	19.27	572.8	1.34
SVM	14.65	250.26	1.01
KNN	15.52	315.25	1.07
DT	14.2	298.46	0.98
RF	13.78	271.58	0.95
ANN	34.36	1566.32	2.38

	MAE	MSE	MAPE
LR	27.29	1081.86	1.89
SVM	26.62	1315.27	1.84
KNN	15.87	334.21	1.1
DT	19.14	631.75	1.32
RF	14.97	268.09	1.04
ANN	98.23	9649.13	13.34

SBI BANK

WEEK 1

FEATURE WITHOUT INDICATORS

	MAE	MSE	MAPE
LR	12.94	223.29	1.69
SVM	16.95	347.25	2.23
KNN	21.42	535.52	2.81
DT	14.22	284.16	1.87
RF	17.08	347.77	2.25
ANN	23.86	675.23	3.14

FEATURE WITH ALL INDICATORS

	MAE	MSE	MAPE
LR	23.91	1102.2	3.12
SVM	17.77	450.16	2.32
KNN	23.0	604.7	3.01
DT	18.61	549.37	2.46
RF	16.58	299.25	2.18
ANN	71.2	5069.44	8.2

MOMENTUM INDICATOR

	MAE	MSE	MAPE
LR	14.88	290.62	1.95
SVM	17.0	350.31	2.23
KNN	21.42	535.52	2.81
DT	15.78	372.76	2.07
RF	16.82	335.45	2.21
ANN	39.26	2155.53	5.13

TECHNICAL INDICATOR

	MAE	MSE	MAPE
LR	17.65	375.11	2.3
SVM	14.36	226.9	1.88
KNN	21.04	514.05	2.76
DT	11.52	209.14	1.51
RF	15.65	279.55	2.05
ANN	14.09	262.81	1.84

VOLATILITY INDICATOR

	MAE	MSE	MAPE
LR	13.93	246.98	1.82
SVM	20.39	457.74	2.68
KNN	21.42	535.52	2.81
DT	21.87	651.53	2.88
RF	18.49	376.72	2.43
ANN	31.96	1397.57	4.2

VOLUME INDICATOR

	MAE	MSE	MAPE
LR	19.86	756.38	2.59
SVM	17.77	449.16	2.32
KNN	23.0	604.7	3.01
DT	16.81	496.73	2.22
RF	17.61	351.96	2.31
ANN	98.23	9649.13	13.34

BAJAJ FINANCE

WEEK 1

FEATURE WITHOUT INDICATORS

	MAE	MSE	MAPE
LR	204.39	145821.88	3.21
SVM	95.53	16475.77	1.5
KNN	81.93	11818.11	1.28
DT	101.9	15462.04	1.58
RF	87.86	12164.11	1.37
ANN	134.76	41133.96	2.11

FEATURE WITH ALL INDICATORS

	MAE	MSE	MAPE
LR	409.31	334738.79	6.38
SVM	113.45	17761.65	1.76
KNN	70.67	11312.77	1.11
DT	90.5	17192.69	1.42
RF	73.42	12044.89	1.15
ANN	64.2	4121.64	7.5

MOMENTUM INDICATOR

	MAE	MSE	MAPE
LR	378.59	249954.08	5.9
SVM	84.04	13034.68	1.32
KNN	76.91	11328.49	1.2
DT	80.98	12578.3	1.26
RF	76.62	11956.13	1.2
ANN	221.08	76140.71	3.41

TECHNICAL INDICATOR

	MAE	MSE	MAPE
LR	427.44	275863.37	6.65
SVM	87.84	14715.29	1.37
KNN	81.93	11818.11	1.28
DT	111.2	19906.72	1.74
RF	78.51	13076.07	1.23
ANN	199.4	48634.29	3.11

VOLATILITY INDICATOR

	MAE	MSE	MAPE
LR	393.4	366906.43	6.15
SVM	87.48	14670.64	1.37
KNN	82.45	11870.04	1.29
DT	110.22	19541.17	1.72
RF	85.32	13208.4	1.33
ANN	190.75	48698.98	2.98

VOLUME INDICATOR

	MAE	MSE	MAPE
LR	433.5	325048.17	6.75
SVM	107.2	16087.03	1.67
KNN	70.67	11312.77	1.11
DT	98.09	15950.68	1.53
RF	77.88	10850.91	1.22
ANN	104.46	10911.89	8.34

If we analyze all the dataset of all the company mentioned above we can clearly see that in the maximum place technical indicators are giving better result among all the indicator. So for working purpose we will be moving forward with technical indicators.

14. OUR PROPOSED MODEL:

Accurately predicting stock prices is a challenging task due to the complex and volatile nature of financial markets. Traditional approaches often rely on individual models or algorithms, which may not capture the full range of factors influencing stock prices. In this report, we propose a model that combines multiple algorithms and techniques to improve accuracy in stock price prediction.

14.1. Need for Combining Models:

a. Diverse Perspectives: Each algorithm and technique has its strengths and weaknesses. By combining multiple models, we can leverage the strengths of each to improve overall accuracy.

b. Reduced Bias and Variance: Combining models can help reduce bias and variance in predictions. Bias occurs when a model consistently underestimates or overestimates the true value, while variance occurs when a model is overly sensitive to small fluctuations in the training data.

c. Robustness to Changes: Financial markets are dynamic, with trends and patterns constantly evolving. A model that performs well under one set of conditions may not perform as well under different conditions

In conclusion, our proposed model for stock price prediction aims to leverage the strengths of multiple algorithms and techniques to improve accuracy and robustness.

The need to combine models arises from the complex and dynamic nature of financial markets, where traditional approaches may fall short in capturing the full range of factors affecting stock prices. By combining models, we can reduce bias and variance, improve adaptability to changing market conditions, and enhance overall prediction accuracy.

Ensemble learning techniques, such as weighted averaging and model selection based on performance, further enhance the accuracy of our combined model. By assigning weights to each model based on its performance and selecting the most appropriate model for each type of data or market condition, we can create a more effective and accurate prediction model.

Overall, our proposed model offers a comprehensive and robust approach to stock price prediction, addressing the challenges posed by the complex and dynamic nature of financial markets.

14.2. Feature Engineering

Feature engineering is crucial for developing effective stock price prediction models. We have chosen to focus on momentum indicators (SMA, EMA, DEMA) as they provide insights into the trend and momentum of stock prices. These indicators can help identify potential buy or sell signals based on historical price data.

Combining SMA, EMA, and DEMA for Stock Price Prediction:

Trend Identification: SMA, EMA, and DEMA can help identify trends in stock prices. SMA provides a long-term trend, EMA provides a short-term trend, and DEMA provides a smoother trend line, combining the advantages of both SMA and EMA.

Signal Generation: These indicators can generate buy or sell signals based on crossovers and divergences. A crossover occurs when a short-term moving average crosses above or below a long-term moving average, indicating a potential change in trend.

Confirmation: By using multiple moving averages, traders can confirm the strength of a trend. For example, if SMA, EMA, and DEMA are all pointing upward, it indicates a strong bullish trend, providing more confidence in trading decisions.

Conclusion:

In conclusion, SMA, EMA, and DEMA are valuable technical indicators for predicting stock prices. By combining these indicators, traders and analysts can gain a more comprehensive view of market trends, identify potential trading opportunities, and make more informed decisions.

14.3. Input Data Selection

Sliding Window Approach: Our model utilizes a sliding window approach for selecting input data. This method involves moving a fixed-size window over the dataset, with each window representing a subset of the data used for training. The size of the window is chosen based on the dataset's characteristics to capture relevant information for prediction.

High Volatility Dataset: For datasets exhibiting high volatility, we employ a 1-week window. This choice allows the model to capture short-term price movements and rapid changes in volatility, which are characteristic of volatile markets.

Trending Dataset: In contrast, for datasets showing clear trends, we opt for a longer 4-week window. This extended period enables the model to identify and predict the overall trend more accurately.

Rationale: The selection of window size is crucial as it determines the amount of historical data the model considers for prediction. For volatile datasets, a shorter window is preferred to capture recent market dynamics, while for trending datasets, a longer window provides a more comprehensive view of the trend

14.4. Algorithm Selection

Random Forest (RF): RF is chosen for datasets with high volatility due to its ability to handle non-linear relationships and noisy data effectively. RF works by constructing multiple decision trees during training and outputting the average prediction of the individual trees.

Ridge Regression (RR): RR is selected for datasets showing trends, as it can handle multicollinearity and overfitting. RR adds a penalty term to the regression equation, which helps in reducing the impact of multicollinearity and prevents overfitting by imposing constraints on the coefficients.

Decision Tree (DT): DT is included in our algorithm selection to provide interpretability and capture complex relationships in the data. DTs partition the data into subsets based on the features and make predictions based on the majority class in each subset.

Rationale: The selection of RF, RR, and DT is based on their suitability for different dataset characteristics. By combining these algorithms, we aim to leverage their strengths to develop a robust stock price prediction model that can perform well across various market conditions, including high volatility and trending markets.

14.5. Weighting Strategy

High Volatility Dataset: For datasets with high volatility, we assign weights to the algorithms based on their effectiveness in capturing rapid price changes. We assign the highest weight to Random Forest (RF), followed by Decision Tree (DT) and Ridge Regression (RR). This weighting strategy acknowledges RF's ability to handle non-linear relationships and noisy data effectively, making it well-suited for volatile market conditions where price movements can be unpredictable. DT is assigned the second-highest weight due to its interpretability and ability to capture complex relationships, while RR is assigned the lowest weight as it is more suited for datasets showing trends rather than high volatility.

Trending Dataset: In contrast, for datasets showing clear trends, we assign weights based on the algorithms' ability to capture trend patterns. We assign the highest weight to Ridge Regression (RR), followed by Decision Tree (DT) and Random Forest (RF). RR is assigned the highest weight because of its effectiveness in handling multicollinearity and overfitting, providing a smoother representation of the trend. DT is assigned the second-highest weight for its interpretability and ability to capture complex relationships, while RF is assigned the lowest weight as it may not perform as well in capturing trend patterns compared to RR and DT.

Rationale: The weighting strategy is designed to leverage the strengths of each algorithm based on the dataset's characteristics. By assigning higher weights to algorithms that are more effective for specific dataset characteristics, we aim to improve the overall prediction accuracy of our model across different market conditions.

Our proposed model for stock price prediction integrates technical indicators SMA, EMA, and DEMA, employing a sliding window approach to adapt to varying market conditions. This approach utilizes a 1-week window for high volatility datasets, capturing short-term price movements, and a 4-week window for trending datasets, enhancing trend prediction accuracy. Algorithm selection includes RF for high volatility datasets, effective due to its handling of non-linear relationships and noisy data, RR for datasets showing trends, adept at managing multicollinearity and overfitting for a smoother trend representation, and DT for its interpretability and ability to capture complex relationships, complementing RF and RR. The weighting strategy assigns RF the highest weight for high volatility datasets, followed by DT and RR, recognizing RF's effectiveness in capturing rapid price changes. Conversely, RR is assigned the highest weight for trending datasets, followed by DT and RF, acknowledging RR's capability to capture trend patterns. This comprehensive approach aims to develop a robust and accurate stock price prediction model, assisting investors in making informed decisions in dynamic

financial markets by leveraging the strengths of each component to adapt to various market conditions and enhance prediction accuracy.

14.6. Weighting Strategy:

The weighting strategy assigns RF the highest weight for high volatility datasets, followed by DT and RR, recognizing RF's effectiveness in capturing rapid price changes. Conversely, RR is assigned the highest weight for trending datasets, followed by DT and RF, acknowledging RR's capability to capture trend patterns. This comprehensive approach aims to develop a robust and accurate stock price prediction model, assisting investors in making informed decisions in dynamic financial markets by leveraging the strengths of each component to adapt to various market conditions and enhance prediction accuracy.

Weighted Average Calculation: To illustrate the weighted averaging process, consider the following example. Suppose we have the following predictions for a stock price:

RF predicts 100

DT predicts 105

RR predicts 60

And we have assigned the following weights:

wRF = 0.5

wDT = 0.4

wRR = 0.1

To combine these predictions, we calculate the weighted average as follows:

Weighted Prediction = $(0.5\times100)+(0.4\times105)+(0.1\times60)$

Weighted Prediction=50+42+6

Weighted Prediction=98

So, the combined prediction using weighted averages is 98. This means that based on the assigned weights and the predictions of the three models, we estimate the stock price to be 98. This example demonstrates how the weighting strategy can be applied to combine predictions from different models, providing a more accurate and robust prediction for stock prices.

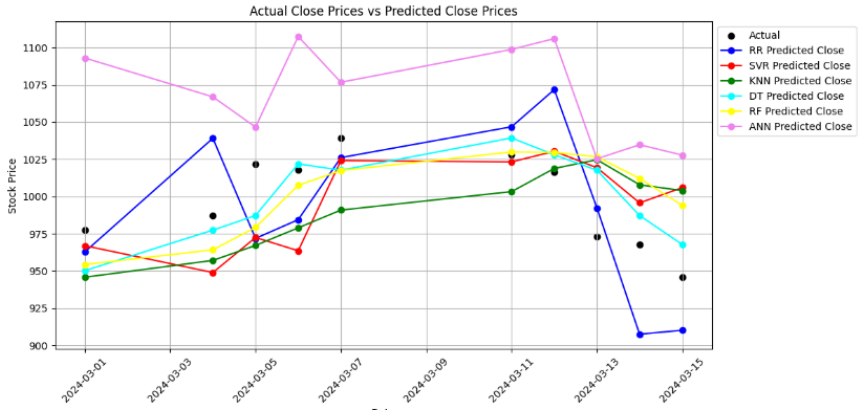
14.7. ANALYSIS OF OUR PROPOSED MODEL

14.7.1. COMPANIES WITH HIGH VOLATILITY

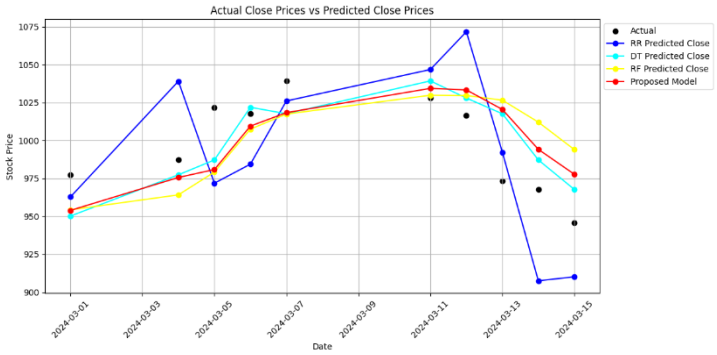
TATAMOTORS

Date	Open	High	Low	Close
2024-01-18	807.000000	822.950012	797.000000	819.049988
2024-01-19	823.849976	826.000000	819.500000	823.549988
2024-01-23	824.900024	827.599976	796.299988	800.450012
2024-01-24	802.400024	812.000000	788.500000	810.900024
2024-01-25	814.000000	814.150024	800.299988	811.849976
2024-01-29	811.849976	843.799988	811.049988	841.000000
2024-01-30	843.000000	885.950012	842.849976	858.849976
2024-01-31	865.200012	896.500000	865.200012	884.200012
2024-02-01	900.000000	900.150024	876.299988	878.500000
2024-02-02	886.000000	895.750000	876.849976	878.750000
2024-02-05	934.000000	950.000000	915.349976	926.799988
2024-02-06	936.400024	941.299988	928.400024	939.549988
2024-02-07	944.000000	944.000000	928.049988	933.799988
2024-02-08	937.000000	939.700012	918.799988	924.299988
2024-02-09	926.000000	927.400024	906.049988	915.000000
2024-02-12	916.099976	925.000000	908.000000	911.599976
2024-02-13	911.599976	919.099976	894.000000	906.900024
2024-02-14	900.000000	919.950012	894.349976	918.299988
2024-02-15	923.700012	927.000000	916.349976	920.549988
2024-02-16	925.000000	948.799988	924.099976	938.599976
2024-02-19	942.950012	942.950012	931.049988	932.599976
2024-02-20	934.450012	934.650024	920.400024	926.349976
2024-02-21	926.349976	937.200012	916.500000	921.049988
2024-02-22	924.650024	933.849976	914.599976	932.299988
2024-02-23	933.099976	939.799988	929.400024	937.400024
2024-02-26	937.099976	945.000000	930.700012	936.950012
2024-02-27	936.750000	965.000000	935.500000	962.700012
2024-02-28	966.150024	976.000000	950.299988	958.049988
2024-02-29	959.000000	959.250000	942.900024	950.200012

Dataset



Graph



Graph

RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
962.79086112	966.95689849	945.79000244	950.20001221	954.31500366	1092.95938750	953.92809197
1039.08190182	948.92334111	957.06000977	977.40002441	964.17601501	1066.89437919	975.63380651
971.85565761	972.43695100	967.11000977	987.20001221	979.20301392	1046.58253175	980.86737777
984.41419928	963.55539558	978.95001221	1021.90002441	1007.48402039	1107.43082682	1009.50183948
1026.08834733	1024.10520065	990.87001953	1017.65002441	1017.43202393	1076.69929574	1018.36305641
1046.78672979	1023.23058818	1003.25002441	1039.30004883	1029.85603821	1098.73423382	1034.38231055
1071.73884345	1030.33115641	1018.81002197	1028.00000000	1029.63602417	1105.96774783	1033.35549885
992.29255721	1019.31875605	1024.67001953	1017.65002441	1026.60952026	1025.29622488	1020.48997520
907.52381278	995.71517809	1007.85002441	987.20001221	1012.10602295	1034.72077885	994.17599871
910.17398318	1005.93813358	1003.96002197	967.75000000	994.01301208	1027.76557966	977.75020557

Actual and Predicted Data

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	213.45	108.99	999.19	739.84	532.69	421.07	550.84
2024-03-04	2691.53	1465.36	908.42	1369.0	529.92	2007.04	389.27
2024-03-05	2504.0	2446.29	3001.94	1204.09	1823.29	682.78	1683.46
2024-03-06	1104.9	2925.73	1497.69	18.06	103.43	17103.41	66.42
2024-03-07	174.5	230.74	2345.46	468.72	478.3	3771.19	438.48
2024-03-11	353.06	22.75	612.56	107.12	3.46	11352.9	0.01
2024-03-12	3051.46	191.27	5.34	519.84	172.66	1978.47	410.06
2024-03-13	364.43	2127.05	2649.16	4369.21	2852.63	3592.8	2892.29
2024-03-14	3627.65	782.32	1608.01	2932.22	1967.81	10606.94	1357.19
2024-03-15	1273.06	3610.81	3376.77	479.61	2319.39	0.4	1017.61
MSE	1535.75	1391.15	1700.46	1220.77	1078.33	5151.62	880.57

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	1.49%	1.07%	3.23%	2.78%	2.36%	2.1%	2.4%
2024-03-04	5.26%	3.88%	3.05%	3.75%	2.33%	4.54%	2.0%
2024-03-05	4.9%	4.84%	5.36%	3.4%	4.18%	2.56%	4.02%
2024-03-06	3.27%	5.32%	3.8%	0.42%	1.0%	12.85%	0.8%
2024-03-07	1.27%	1.46%	4.66%	2.08%	2.1%	5.91%	2.01%
2024-03-11	1.83%	0.46%	2.41%	1.01%	0.18%	10.36%	0.01%
2024-03-12	5.43%	1.36%	0.23%	2.24%	1.29%	4.38%	1.99%
2024-03-13	1.96%	4.74%	5.29%	6.79%	5.49%	6.16%	5.53%
2024-03-14	6.22%	2.89%	4.14%	5.6%	4.58%	10.64%	3.81%
2024-03-15	3.77%	6.35%	6.14%	2.32%	5.09%	0.07%	3.37%
MAPE	3.54%	3.24%	3.83%	3.04%	2.86%	5.96%	2.59%

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	14.61	10.44	31.61	27.2	23.08	0.63	23.47
2024-03-04	51.88	38.28	30.14	37.0	23.02	0.63	19.73
2024-03-05	50.04	49.46	54.79	34.7	42.7	0.63	41.03
2024-03-06	33.24	54.09	38.7	4.25	10.17	0.63	8.15
2024-03-07	13.21	15.19	48.43	21.65	21.87	0.63	20.94
2024-03-11	18.79	4.77	24.75	10.35	1.86	0.63	0.11
2024-03-12	55.24	13.83	2.31	22.8	13.14	0.63	20.25
2024-03-13	19.09	46.12	51.47	66.1	53.41	0.63	53.78
2024-03-14	60.23	27.97	40.1	54.15	44.36	0.63	36.84
2024-03-15	35.68	60.09	58.11	21.9	48.16	0.63	31.9
MAE	35.2	32.02	38.04	30.01	28.18	59.82	25.62

Dataset of Determining MAE, MSE, MAPE

	MAE	MSE	MAPE
LR	35.2	1535.75	3.54
SVM	32.02	1391.15	3.24
KNN	38.04	1700.46	3.83
DT	30.01	1220.77	3.04
RF	28.18	1078.33	2.86
ANN	59.82	5151.62	5.96
PM	25.62	880.57	2.59

	MAE	MSE	MAPE
LR	19.89	592.42	2.0
SVM	80.32	8609.8	7.94
KNN	53.01	3682.22	5.23
DT	48.15	3104.25	4.75
RF	46.37	2892.62	4.57
ANN	21.74	644.38	2.2

HINDUSTAN PETROLUEM CORPORATION:

	MAE	MSE	MAPE
LR	19.39	630.26	3.85
SVM	12.68	330.84	2.59
KNN	12.25	285.14	2.49
DT	13.13	265.45	2.65
RF	11.39	214.57	2.3
ANN	21.54	681.87	4.23
PM	12.1	230.12	2.44

	MAE	MSE	MAPE
LR	11.87	246.4	2.41
SVM	22.78	883.0	4.61
KNN	13.91	447.91	2.85
DT	23.01	826.48	4.61
RF	16.2	422.83	3.28
ANN	27.86	948.99	5.57

Bajaj Finance:

	MAE	MSE	MAPE
LR	48.08	3502.69	3.04
SVM	17.15	634.33	1.1
KNN	20.03	694.43	1.28
DT	26.03	969.77	1.66
RF	19.29	603.0	1.23
ANN	72.04	9706.3	4.59
PM	19.26	651.7	1.23

	MAE	MSE	MAPE
LR	19.11	635.84	1.22
SVM	24.53	1018.32	1.56
KNN	21.13	792.13	1.35
DT	15.93	523.37	1.02
RF	16.74	577.52	1.07
ANN	24.49	905.67	1.55

COMPANIES SHOWING TREND or LOW VOLATILITY :

1. HDFCBANK

	MAE	MSE	MAPE
LR	13.44	244.29	0.93
SVM	13.04	353.91	0.9
KNN	16.6	349.23	1.15
DT	18.28	632.16	1.27
RF	13.94	315.69	0.97
ANN	25.95	873.44	1.8

	MAE	MSE	MAPE
LR	12.56	217.24	0.87
SVM	18.59	505.97	1.29
KNN	19.04	411.03	1.32
DT	16.69	457.69	1.16
RF	13.66	304.95	0.95
ANN	53.52	3959.39	3.71
PM	12.44	262.51	0.86

2. BHARATI AIRTEL

	MAE	MSE	MAPE
LR	25.1	938.05	2.1
SVM	38.31	2154.81	3.2
KNN	30.53	1222.1	2.55
DT	30.62	1183.13	2.58
RF	28.79	1155.43	2.41
ANN	38.15	1953.53	3.25

	MAE	MSE	MAPE
LR	25.82	911.56	2.16
SVM	25.78	1209.23	2.17
KNN	25.51	856.68	2.14
DT	23.35	711.87	1.98
RF	21.7	704.79	1.82
ANN	53.3	4232.49	4.55
PM	23.17	694.08	1.94

After analyzing all the dataset of company we can see that wherever we are getting fluctuation in company dataset (visualize in graphs) we will be working in that situation using 1 week data and at that time we can see that RF, DT, and RR is giving better result in this sequence.

Wherever we are getting steady upward or downward in company dataset we will be working in that situation using 4 week data and at that time we can see RR, DT and RF is giving better result in this sequence.

And after all of this analyzing we can see that our proposed model is giving better result and time forecasting is giving bad result.

15. METHOLOGY OF EACH ML TECHNIQUES:

Implementation of Linear Regression for our Proposed Work

Importing The Libraries

- *numpy* : For efficient numerical operations and array handling.
- *pandas* : To manipulate and analyze data in structured formats.
- *yfinance* : To download historical stock data directly from Yahoo Finance.
- *matplotlib.pyplot* : To create visualizations and plot graphs of stock prices.

```
import numpy as np
import pandas as pd

# Import YFinance Library
import yfinance as yf
```

Download Stock Prices for Training and Testing from YFinance

1) Download Stock Data for Training (train_df)
Download historical stock prices for 1 week to create a training dataset *(`train_df`).* This dataset is used to train the stock prediction model by capturing past trends and patterns.

```
# Download Stock Prices

# Ticker and start date, end date for Training
stock_symbol = 'RELIANCE.NS'
train_start_date = '2024-02-20'
train_end_date = '2024-03-01'

# Download Stock Price Data from Yahoo Finance
train_df = yf.download(stock_symbol, start=train_start_date, end=train_end_date, progress=False)
```

Date	Open	High	Low	Close	Adj Close	Volume
2024-02-20	2950.050049	2951.000000	2923.600098	2942.050049	2942.050049	3558748
2024-02-21	2948.000000	2977.050049	2915.100098	2935.399902	2935.399902	6360146
2024-02-22	2936.300049	2969.899902	2916.000000	2963.500000	2963.500000	9246864
2024-02-23	2979.000000	2995.100098	2966.699951	2987.250000	2987.250000	7219292
2024-02-26	2987.100098	2989.050049	2965.000000	2974.649902	2974.649902	3756553
2024-02-27	2966.050049	2999.899902	2956.100098	2971.300049	2971.300049	5413022
2024-02-28	2966.000000	2982.550049	2900.350098	2911.250000	2911.250000	4323975
2024-02-29	2930.000000	2957.949951	2909.050049	2921.600098	2921.600098	11814488

Dataset

2) Download Stock Data for Testing (test_df)
Download next 10 days stock prices to create a testing dataset *(`test_df`).* This dataset is used to evaluate the performance of the trained model by comparing its predictions with actual stock prices.

```
# Download Stock Data from Predicting

# Ticker and start date, end date for Predicting
test_start_date = '2024-03-01'
test_end_date = '2024-03-16'

# Download Stock Price Data from Yahoo Finance
test_df = yf.download(stock_symbol, start=test_start_date, end=test_end_date, progress=False)
print(test_df)
```

```
                  Open         High          Low        Close
Date
2024-03-01  2927.000000  3000.000000  2925.000000  2984.250000
2024-03-04  2980.949951  3024.899902  2974.449951  3014.800049
2024-03-05  3011.550049  3014.800049  2972.100098  3000.399902
2024-03-06  2986.899902  3018.000000  2957.000000  3006.000000
2024-03-07  3005.949951  3006.199951  2951.100098  2957.850098
2024-03-11  2978.000000  2978.000000  2927.000000  2933.199951
2024-03-12  2933.199951  2976.000000  2930.050049  2950.850098
2024-03-13  2959.550049  2966.199951  2855.550049  2864.350098
2024-03-14  2879.399902  2897.050049  2851.000000  2862.949951
2024-03-15  2851.899902  2866.449951  2825.800049  2836.449951
```

Data Preprocessing

1) Remove any rows with null values.
2) Extract 'Close', 'Open', 'Low', and 'High' prices from training data.

```
# Drop null values
train_df.dropna()

# Extract Close Price from Training data
prices_to_train = train_df[['Close', 'Open', 'Low', 'High']].to_numpy()
print(prices_to_train)
```

```
[[2942.05004883 2950.05004883 2923.60009766 2951.        ]
 [2935.39990234 2948.         2915.10009766 2977.05004883]
 [2963.5        2936.30004883 2916.         2969.89990234]
 [2987.25       2979.         2966.69995117 2995.10009766]
 [2974.64990234 2987.10009766 2965.         2989.05004883]
 [2971.30004883 2966.05004883 2956.10009766 2999.89990234]
 [2911.25       2966.         2900.35009766 2982.55004883]
 [2921.60009766 2930.         2909.05004883 2957.94995117]]
```

Extracting Working Days for which we want to predict the stock prices

```
# Extract dates to predict
prediction_dates = test_df.index
print(prediction_dates)
```

```
DatetimeIndex(['2024-03-01', '2024-03-04', '2024-03-05', '2024-03-06',
               '2024-03-07', '2024-03-11', '2024-03-12', '2024-03-13',
               '2024-03-14', '2024-03-15'],
              dtype='datetime64[ns]', name='Date', freq=None)
```

Importing Models from SkLearn

```
# Ridge Regression
from sklearn.linear_model import Ridge
```

Prepare Features and Targets:

- *train_features*: List to store the features (every day {Close, Open, High, Low}).
- *train_targets:* List to store the targets (next day's {Close, Open, High, Low}).
- *train_features = train_data[:-1]:* Add every day's {Close, Open, High, Low} to features (from first to second last).
- *train_targets = train_data[1:]:* Add the next day's {Close, Open, High, Low} to targets (from second to last).

```
train_data = train_df[['Close', 'Open', 'Low', 'High']]
train_data = train_data.values

train_features = train_data[:-1]
train_targets = train_data[1:]
```

```
---- Close -------- Open -------- Low -------- High ------
2942.05004883 | 2950.05004883 | 2923.60009766 | 2951.00000000
2935.39990234 | 2948.00000000 | 2915.10009766 | 2977.05004883
2963.50000000 | 2936.30004883 | 2916.00000000 | 2969.89990234
2987.25000000 | 2979.00000000 | 2966.69995117 | 2995.10009766
2974.64990234 | 2987.10009766 | 2965.00000000 | 2989.05004883
2971.30004883 | 2966.05004883 | 2956.10009766 | 2999.89990234
2911.25000000 | 2966.00000000 | 2900.35009766 | 2982.55004883
2921.60009766 | 2930.00000000 | 2909.05004883 | 2957.94995117
```

Train Linear Regression model

- model = *Ridge(alpha=10)* - Create a Ridge regression model with alpha (regularization strength) set to higher values
- *model.fit(train_features, train_targets)* - Train the model on the features and targets

```
model_rr = Ridge(alpha=10)
model_rr.fit(train_features, train_targets)
```

```
========================================== Features & Targets ============================================================

----------------------Features------------------------------ || ----------------------Targets------------------------------
  Close       |     Open      |      Low      |     High      ||  Close       |     Open      |      Low      |     High
-----------------------------------------------------------------------------------------------------------------------------
2942.05004883 | 2950.05004883 | 2923.60009766 | 2951.00000000 || 2935.39990234 | 2948.00000000 | 2915.10009766 | 2977.05004883
2935.39990234 | 2948.00000000 | 2915.10009766 | 2977.05004883 || 2963.50000000 | 2936.30004883 | 2916.00000000 | 2969.89990234
2963.50000000 | 2936.30004883 | 2916.00000000 | 2969.89990234 || 2987.25000000 | 2979.00000000 | 2966.69995117 | 2995.10009766
2987.25000000 | 2979.00000000 | 2966.69995117 | 2995.10009766 || 2974.64990234 | 2987.10009766 | 2965.00000000 | 2989.05004883
2974.64990234 | 2987.10009766 | 2965.00000000 | 2989.05004883 || 2971.30004883 | 2966.05004883 | 2956.10009766 | 2999.89990234
2971.30004883 | 2966.05004883 | 2956.10009766 | 2999.89990234 || 2911.25000000 | 2966.00000000 | 2900.35009766 | 2982.55004883
2911.25000000 | 2966.00000000 | 2900.35009766 | 2982.55004883 || 2921.60009766 | 2930.00000000 | 2909.05004883 | 2957.94995117
```

Make predictions for the next day's {Close, Open, High, Low}:

- *target_to_predict = train_targets[-1]* - Use the last day's data to predict the next day's {Close, Open, High, Low} prices.
- *rr_predict = model_rr.predict(target_to_predict.reshape(1, -1))* - Predict the next day's prices using a regression model.
- *rr_predicted_price[date] = rr.predict[-1]* - Store the predicted prices.

```
target_to_predict = train_targets[-1]

for date in prediction_dates:

    rr_predict = model_rr.predict(target_to_predict.reshape(1, -1))
    rr_predicted_price[date] = rr_predict[-1]

    specific_date_row = test_df.loc[test_df.index == date, ['Close', 'Open', 'Low', 'High']].to_numpy()
    target_to_predict = specific_date_row.flatten()
```

OUTPROPOSED MODEL:

STOCK MARKET PREDICTION

IMPORTING THE LIBRARIES

- **numpy** for numerical operations
- **pandas** data manipulation and analysis
- **yfinance** for downloading stock data
- **matplotlib.pyplot** for plotting graphs

```
import numpy as np
import pandas as pd

# Import YFinance Library
import yfinance as yf
```

Download Stock Prices for Training and Testing from YFinance

1) Download Stock Data for Training (train_df)

```
# Download Stock Prices

# Ticker and start date, end date for Training
stock_symbol = 'BHARTIARTL.NS'
train_start_date = '2024-01-18'
train_end_date = '2024-03-01'

# Download Stock Price Data from Yahoo Finance
train_df = yf.download(stock_symbol, start=train_start_date, end=train_end_date, progress=False)
train_df
```

Date	Open	High	Low	Close
2024-01-18	1075.199951	1096.000000	1075.199951	1087.050049
2024-01-19	1091.000000	1136.349976	1089.800049	1125.000000
2024-01-23	1145.000000	1172.550049	1135.000000	1158.000000
2024-01-24	1140.099976	1194.000000	1140.099976	1189.949951
2024-01-25	1186.000000	1200.650024	1157.050049	1160.550049
2024-01-29	1158.000000	1169.750000	1158.000000	1162.150024
2024-01-30	1162.150024	1180.849976	1154.550049	1158.650024
2024-01-31	1159.000000	1175.050049	1146.199951	1170.699951
2024-02-01	1165.050049	1170.199951	1148.949951	1151.199951
2024-02-02	1155.000000	1175.199951	1145.300049	1150.800049
2024-02-05	1154.349976	1159.650024	1110.000000	1113.550049
2024-02-06	1125.000000	1156.099976	1123.000000	1134.050049
2024-02-07	1138.800049	1148.250000	1131.800049	1134.300049
2024-02-08	1144.400024	1146.500000	1116.199951	1142.150024
2024-02-09	1145.949951	1145.949951	1116.250000	1120.250000
2024-02-12	1122.900024	1129.000000	1111.300049	1118.699951
2024-02-13	1111.000000	1125.000000	1103.800049	1117.849976
2024-02-14	1110.000000	1121.150024	1104.699951	1115.849976
2024-02-15	1122.300049	1131.449951	1109.400024	1120.699951
2024-02-16	1124.000000	1127.050049	1116.849976	1120.000000
2024-02-19	1121.050049	1145.000000	1118.349976	1142.199951
2024-02-20	1145.000000	1151.599976	1135.000000	1143.949951
2024-02-21	1155.000000	1159.150024	1132.599976	1139.900024
2024-02-22	1137.599976	1138.750000	1097.650024	1135.550049
2024-02-23	1127.000000	1131.849976	1115.300049	1125.750000
2024-02-26	1118.099976	1125.900024	1104.349976	1110.050049
2024-02-27	1113.000000	1131.000000	1101.300049	1127.500000
2024-02-28	1133.400024	1152.650024	1123.500000	1128.750000
2024-02-29	1118.849976	1137.949951	1099.000000	1123.349976

TREND AND FLUCTUATIONS:

```
train_input = yf.download(stock_symbol, start='2024-02-20', end=train_end_date, progress=False)

def calculate_trend_and_fluctuations(df, window_size=3):
    trend = 0
    fluctuations = 0
    for i in range(len(df) - window_size + 1):
        window = df['Close'].iloc[i:i+window_size]
        if all(window.iloc[i] < window.iloc[i+1] for i in range(len(window) - 1)):
            trend += 1
        elif all(window.iloc[i] > window.iloc[i+1] for i in range(len(window) - 1)):
            trend += 1
        else:
            fluctuations += 1
    print("T",trend)
    print("F",fluctuations)
    if trend > fluctuations:
        return "Trend"
    else:
        return "Fluctuations"

# Calculate trend and fluctuations for train_df
result = calculate_trend_and_fluctuations(train_input)
if result == 'Fluctuations':
    train_df = train_input
train_df
```

2) Downloading Stock Data for Testing (test_df)

```
# Download Stock Data from Predicting

# Ticker and start date, end date for Predicting
test_start_date = '2024-03-01'
test_end_date = '2024-03-16'

# Download Stock Price Data from Yahoo Finance
test_df = yf.download(stock_symbol, start=test_start_date, end=test_end_date, progress=False)
print(test_df)
```

```
# Extract Close Price from Training data
prices_to_train = train_df[['Close', 'Open', 'Low', 'High']].to_numpy()
print(prices_to_train)
```

```
[[1087.05004883 1075.19995117 1075.19995117 1096.        ]
 [1125.         1091.         1089.80004883 1136.34997559]
 [1158.         1145.         1135.         1172.55004883]
 [1189.94995117 1140.09997559 1140.09997559 1194.        ]
 [1160.55004883 1186.         1157.05004883 1200.65002441]
 [1162.15002441 1158.         1158.         1169.75      ]
 [1158.65002441 1162.15002441 1154.55004883 1180.84997559]
 [1170.69995117 1159.         1146.19995117 1175.05004883]
 [1151.19995117 1165.05004883 1148.94995117 1170.19995117]
 [1150.80004883 1155.         1145.30004883 1175.19995117]
 [1113.55004883 1154.34997559 1110.         1159.65002441]
 [1134.05004883 1125.         1123.         1156.09997559]
 [1134.30004883 1138.80004883 1131.80004883 1148.25      ]
 [1142.15002441 1144.40002441 1116.19995117 1146.5       ]
 [1120.25       1145.94995117 1116.25       1145.94995117]
 [1118.69995117 1122.90002441 1111.30004883 1129.        ]
 [1117.84997559 1111.         1103.80004883 1125.        ]
 [1115.84997559 1110.         1104.69995117 1121.15002441]
 [1120.69995117 1122.30004883 1109.40002441 1131.44995117]
 [1120.         1124.         1116.84997559 1127.05004883]
 [1142.19995117 1121.05004883 1118.34997559 1145.        ]
 [1143.94995117 1145.         1135.         1151.59997559]
 [1139.90002441 1155.         1132.59997559 1159.15002441]
 [1135.55004883 1137.59997559 1097.65002441 1138.75      ]
 [1125.75       1127.         1115.30004883 1131.84997559]
 [1110.05004883 1118.09997559 1104.34997559 1125.90002441]
 [1127.5        1113.         1101.30004883 1131.        ]
 [1128.75       1133.40002441 1123.5        1152.65002441]
 [1123.34997559 1118.84997559 1099.         1137.94995117]]
```

Extracting Working Days for which we want to predict the stock prices

```
# Extract dates to predict
prediction_dates = test_df.index
print(prediction_dates)
```

```
DatetimeIndex(['2024-03-01', '2024-03-04', '2024-03-05', '2024-03-06',
               '2024-03-07', '2024-03-11', '2024-03-12', '2024-03-13',
               '2024-03-14', '2024-03-15'],
              dtype='datetime64[ns]', name='Date', freq=None)
```

```
from sklearn.feature_selection import f_regression

# Calculate SMA
sma = train_df['Close'].rolling(window=5, min_periods=1).mean()
# Convert Series to numpy array and reshape
sma_array = sma.to_numpy().reshape(-1, 1)

# Calculate EMA
ema = train_df['Close'].ewm(span=5, adjust=False).mean()
ema_array = ema.to_numpy().reshape(-1, 1)

# Calculate DEMA
dema = 2 * ema - ema.ewm(span=5, adjust=False).mean()
dema_array = dema.to_numpy().reshape(-1, 1)

# Add values as a column at the end of the numpy array
prices_to_train = np.hstack((prices_to_train, sma_array, ema_array, dema_array))

print(prices_to_train)
```

```
# Convert the numpy array to a pandas DataFrame
df = pd.DataFrame(prices_to_train, columns=['Open','High','Low','Close','SMA','EMA','DEMA']
df.index = train_df.index
# Display the DataFrame
print(df)
```

Date	Open	High	Low	Close	SMA	EMA	DEMA
2024-01-18	1087.050049	1075.199951	1075.199951	1096.000000	1087.050049	2024-01-18	1087.050049	1087.050049
2024-01-19	1125.000000	1091.000000	1089.800049	1136.349976	1106.025024	2024-01-19	1099.700033	1108.133355
2024-01-23	1158.000000	1145.000000	1135.000000	1172.550049	1123.350016	2024-01-23	1119.133355	1137.711118
2024-01-24	1189.949951	1140.099976	1140.099976	1194.000000	1140.000000	2024-01-24	1142.738887	1170.861084
2024-01-25	1160.550049	1186.000000	1157.050049	1200.650024	1144.110010	2024-01-25	1148.675941	1171.382108
2024-01-29	1162.150024	1158.000000	1158.000000	1169.750000	1159.130005	2024-01-29	1153.167302	1171.298988
2024-01-30	1158.650024	1162.150024	1154.550049	1180.849976	1165.860010	2024-01-30	1154.994876	1168.301049
2024-01-31	1170.699951	1159.000000	1146.199951	1175.050049	1168.400000	2024-01-31	1160.229901	1172.590700
2024-02-01	1151.199951	1165.050049	1148.949951	1170.199951	1160.650000	2024-02-01	1157.219918	1163.453795
2024-02-02	1150.800049	1155.000000	1145.300049	1175.199951	1158.700000	2024-02-02	1155.079962	1157.809242
2024-02-05	1113.550049	1154.349976	1110.000000	1159.650024	1148.980005	2024-02-05	1141.236657	1133.827308
2024-02-06	1134.050049	1125.000000	1123.000000	1156.099976	1144.060010	2024-02-06	1138.841121	1132.304531
2024-02-07	1134.300049	1138.800049	1131.800049	1148.250000	1136.780029	2024-02-07	1137.327430	1131.960576
2024-02-08	1142.150024	1144.400024	1116.199951	1146.500000	1134.970044	2024-02-08	1138.934962	1136.428747
2024-02-09	1120.250000	1145.949951	1116.250000	1145.949951	1128.860034	2024-02-09	1132.706641	1126.883617
2024-02-12	1118.699951	1122.900024	1111.300049	1129.000000	1129.890015	2024-02-12	1128.037744	1121.043131
2024-02-13	1117.849976	1111.000000	1103.800049	1125.000000	1126.650000	2024-02-13	1124.641822	1117.714797
2024-02-14	1115.849976	1110.000000	1104.699951	1121.150024	1122.959985	2024-02-14	1121.711206	1115.139446
2024-02-15	1120.699951	1122.300049	1109.400024	1131.449951	1118.669971	2024-02-15	1121.374121	1116.768225
2024-02-16	1120.000000	1124.000000	1116.849976	1127.050049	1118.619971	2024-02-16	1120.916081	1117.540123
2024-02-19	1142.199951	1121.050049	1118.349976	1145.000000	1123.319971	2024-02-19	1128.010704	1130.489815
2024-02-20	1143.949951	1145.000000	1135.000000	1151.599976	1128.539966	2024-02-20	1133.323787	1138.518582
2024-02-21	1139.900024	1155.000000	1132.599976	1159.150024	1133.349976	2024-02-21	1135.515866	1140.440449
2024-02-22	1135.550049	1137.599976	1097.650024	1138.750000	1136.319995	2024-02-22	1135.527260	1138.817912
2024-02-23	1125.750000	1127.000000	1115.300049	1131.849976	1137.469995	2024-02-23	1132.268173	1132.289217
2024-02-26	1110.050049	1118.099976	1104.349976	1125.900024	1131.040015	2024-02-26	1124.862132	1119.938800
2024-02-27	1127.500000	1113.000000	1101.300049	1131.000000	1127.750024	2024-02-27	1125.741421	1123.045393
2024-02-28	1128.750000	1133.400024	1123.500000	1152.650024	1125.520020	2024-02-28	1126.744281	1125.615502
2024-02-29	1123.349976	1118.849976	1099.000000	1137.949951	1123.080005	2024-02-29	1125.612846	1124.106036

```
# Calculate SMA
test_df['SMA'] = test_df['Close'].rolling(window=5, min_periods=1).mean()

# Calculate EMA
test_df['EMA'] = test_df['Close'].ewm(span=5, adjust=False).mean()

# Calculate DEMA
test_df['DEMA'] = 2 * test_df['EMA'] - test_df['EMA'].ewm(span=5, adjust=False).mean()

test_df
```

Import Library for Ridge regression model from scikit-learn

```
from sklearn.linear_model import Ridge
from sklearn.multioutput import MultiOutputRegressor
from sklearn.svm import SVR
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neural_network import MLPRegressor
```

Initialize an empty dictionary to store predicted prices " predicted_price = {} " ¶

```
rr_predicted_price = {}
svr_predicted_price = {}
knn_predicted_price = {}
dt_predicted_price = {}
rf_predicted_price = {}
ann_predicted_price = {}
```

Using Sliding Window Approach to predict {Close, Open, High, Low} ¶

-> Loop through each date in the "prediction_dates" then train the model and make prediction of next day's {Close, Open, High, Low} and then include this prediction in feature data and remove the oldest one and again trained the model -

1) Prepare 'features' (every days' {Close, Open, High, Low}) and 'targets' (next day's {Close, Open, High, Low}) for training the models:

- **train_features = []** - List to store the features (everyday {Close, Open, High, Low})
- **train_targets = []** - List to store the targets (next day's {Close, Open, High, Low})
- **for date in prediction_dates:** - Loop through each day to predict
- **train_features = train_data[:-1]** - Add every days' {Close, Open, High, Low} to features (from first to 2nd last)
- **train_targets = train_data[1:]** - Add the next day's {Close, Open, High, Low} to targets (from second to last)

2) Train all the models

For Example:

- **model = Ridge(alpha=10)** - Create a Ridge regression model with alpha (regularization strength) set to higher values
- **model.fit(train_features, train_targets)** - Train the model on the features and targets

3) Make predictions for the next day's {Close, Open, High, Low}:

- **target_to_predict = train_targets[-1]** - Get the {Close, Open, High, Low}
- **rr_predict = model_rr.predict(target_to_predict.reshape(1, -1))** - Predict the next day's prces
- **rr_predicted_price[date] = rr.predict[-1]** -Store the predicted price for the current day

4) Update the prices_to_train for the next iteration:

- **train_data = train_data[1:]:]** - Remove the first element from the training data
- **specific_date_row = test_df.loc[test_df.index == date, ['Close', 'Open', 'Low', 'High']].to_numpy()1]** - Add the predicted price to the training data for the next iterationte

```
for date in prediction_dates:

    train_features = prices_to_train[:-1]
    train_targets = prices_to_train[1:, :4]

    print(f"================ Date :  {date} ========================")
    print(f"\n================ Training Dataset : ==================")
    print("\n---- Close -------- Open -------- Low -------- High ------")
    print(prices_to_train)
    print(f"\n============================================== Features & Targets =============
    print()
    print("----------------------Features--------------------------- | --------------------
    for feature, target in zip(train_features, train_targets):
        print(f"{feature} | {target}")

    model_rr = Ridge(alpha=10)
    model_rr.fit(train_features, train_targets)

    model_svr = MultiOutputRegressor(SVR(kernel='rbf', C=1e3))
    model_svr.fit(train_features, train_targets)

    model_knn = KNeighborsRegressor(n_neighbors=5)
    model_knn.fit(train_features, train_targets)

    model_dt = DecisionTreeRegressor()
    model_dt.fit(train_features, train_targets)

    model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
    model_rf.fit(train_features, train_targets)

target_to_predict = prices_to_train[-1]

rr_predict = model_rr.predict(target_to_predict.reshape(1, -1))
rr_predicted_price[date] = rr_predict[-1]

svr_predict = model_svr.predict(target_to_predict.reshape(1, -1))
svr_predicted_price[date] = svr_predict[-1]

knn_predict = model_knn.predict(target_to_predict.reshape(1, -1))
knn_predicted_price[date] = knn_predict[-1]

dt_predict = model_dt.predict(target_to_predict.reshape(1, -1))
dt_predicted_price[date] = dt_predict[-1]

rf_predict = model_rf.predict(target_to_predict.reshape(1, -1))
rf_predicted_price[date] = rf_predict[-1]

ann_predict = model_ann.predict(target_to_predict.reshape(1, -1))
ann_predicted_price[date] = ann_predict[-1]

print()
print("================ Features for Prediction : ==================")
print("\n---- Close -------- Open -------- Low -------- High ------")
print(target_to_predict)
print()

print("================ Predicted Prices : ========================")
print("\n---- Close -------- Open -------- Low -------- High ------")
print(rr_predict)
print()

prices_to_train = prices_to_train[1:]
specific_date_row = test_df.loc[test_df.index == date, ['Close', 'Open', 'Low', 'High',
# Add the specific_date_row to prices_to_train
prices_to_train = np.append(prices_to_train, specific_date_row, axis=0)
```

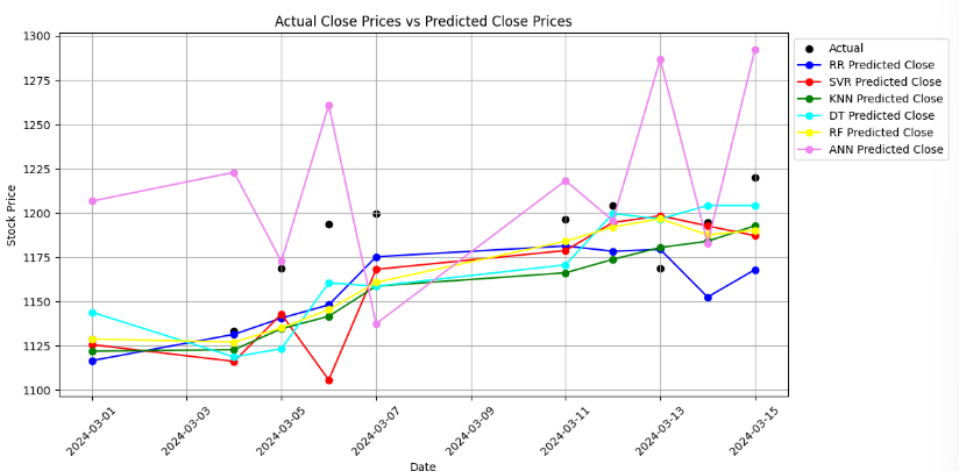
Combine the actual and predicted prices DataFrames

- **result_df = pd.concat([actual_prices, predicted_prices], axis=1)** - Combine actual and predicted prices into a single DataFrame
- **result_df.columns = ['Actual', 'Predicted']** - Rename the columns for clarityrity

Actual Close	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close
1128.69995117	1116.61681019	1125.65977264	1121.98999023	1143.94995117	1128.84098755	1206.72077584
1133.50000000	1131.46259053	1116.17623909	1122.77998047	1118.69995117	1127.08048706	1222.97742480
1168.90002441	1140.63773180	1142.89081147	1134.64001465	1123.34997559	1135.09501343	1172.68573896
1193.69995117	1148.08678096	1105.72876783	1141.64003906	1160.55004883	1145.42252930	1260.78212771
1199.69995117	1175.27695654	1168.16713184	1158.70000000	1158.65002441	1160.82349487	1137.40852857
1196.59997559	1181.39704717	1178.78603884	1166.20998535	1170.69995117	1184.03045532	1218.28160939
1204.25000000	1178.38431426	1194.59159394	1173.79997559	1199.69995117	1192.08396851	1195.67061859
1168.75000000	1179.41000456	1198.45091189	1180.50998535	1196.59997559	1196.73548218	1286.75620753
1194.59997559	1152.50846366	1192.69950865	1184.01999512	1204.25000000	1187.59399292	1182.86914618
1220.00000000	1167.99841757	1187.18502182	1192.77998047	1204.25000000	1190.27648438	1292.31955759

Plot the actual and predicted prices

- **plt.figure(figsize=(12, 6))** - Set the size of the plot
- **plt.plot(result_df.index, result_df['Actual'], marker='o', label='Actual', color='blue')** - Plot actual prices
- **plt.plot(result_df.index, result_df['Predicted'], marker='o', label='Predicted', color='red')** - Plot predicted prices
- **plt.xlabel('Date')** - Set the label for the x-axis
- **plt.ylabel('Stock Price')** - Set the label for the y-axis
- **plt.title('Actual vs Predicted Stock Prices')** - Set the title of the plot
- **plt.legend()** - Show the legend
- **plt.xticks(rotation=45)** - Rotate x-axis labels for better visibility
- **plt.grid(True)** - Add gridlines to the plot
- **plt.show()** - Display the plot



Weightage:

```
if result == "Fluctuations":
    result_df['Proposed Model'] = result_df['RF Predicted Close']*0.6+result_df['DT Predicted Close']*0.3+result_df['RR Predicted Close']*0.1
else:
    result_df['Proposed Model'] = result_df['RR Predicted Close']*0.6+result_df['DT Predicted Close']*0.3+result_df['RF Predicted Close']*0.1
result_df['Proposed Model']
```

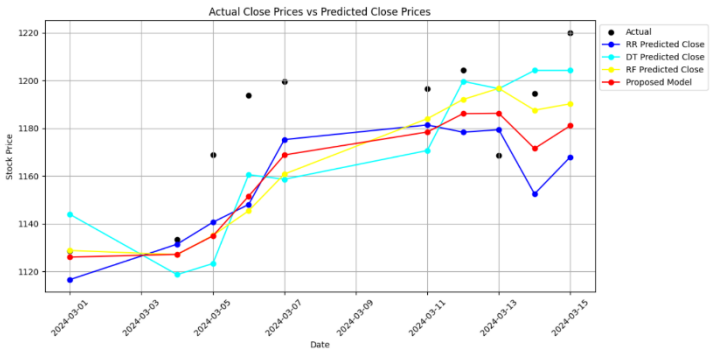
```
2024-03-01    1126.039170
2024-03-04    1127.195588
2024-03-05    1134.897133
2024-03-06    1151.559336
2024-03-07    1168.843531
2024-03-11    1178.451259
2024-03-12    1186.148971
2024-03-13    1186.299544
2024-03-14    1171.539477
2024-03-15    1181.101699
Name: Proposed Model, dtype: float64
```

Plot:

```
import matplotlib.pyplot as plt

# Plot actual and predicted close prices
plt.figure(figsize=(12, 6))
plt.scatter(result_df.index, result_df['Actual Close'], marker='o', label='Actual', color='black')
plt.plot(result_df.index, result_df['RR Predicted Close'], marker='o', label='RR Predicted Close', color='blue')
plt.plot(result_df.index, result_df['DT Predicted Close'], marker='o', label='DT Predicted Close', color='cyan')
plt.plot(result_df.index, result_df['RF Predicted Close'], marker='o', label='RF Predicted Close', color='yellow')
plt.plot(result_df.index, result_df['Proposed Model'], marker='o', label='Proposed Model', color='red')

plt.xlabel('Date')
plt.ylabel('Stock Price')
plt.title('Actual Close Prices vs Predicted Close Prices')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```



RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
1116.61681019	1125.65977264	1121.98999023	1143.94995117	1128.84098755	1206.72077584	1126.03917022
1131.46259053	1116.17623909	1122.77998047	1118.69995117	1127.08048706	1222.97742480	1127.19558838
1140.63773180	1142.89081147	1134.64001465	1123.34997559	1135.09501343	1172.68573896	1134.89713310
1148.08678096	1105.72876783	1141.64003906	1160.55004883	1145.42252930	1260.78212771	1151.55933615
1175.27695654	1168.16713184	1158.70000000	1158.65002441	1160.82349487	1137.40852857	1168.84353073
1181.39704717	1178.78603884	1166.20998535	1170.69995117	1184.03045532	1218.28160939	1178.45125918
1178.38431426	1194.59159394	1173.79997559	1199.69995117	1192.08396851	1195.67061859	1186.14897076
1179.41000456	1198.45091189	1180.50998535	1196.59997559	1196.73548218	1286.75620753	1186.29954363
1152.50846366	1192.69950865	1184.01999512	1204.25000000	1187.59399292	1182.86914618	1171.53947749
1167.99841757	1187.18502182	1192.77998047	1204.25000000	1190.27648438	1292.31955759	1181.10169898

```
# Initialize the data with headers
data = [
    ['RR Predicted Close', 'SVR Predicted Close', 'KNN Predicted Close', 'DT Predicted Close', 'RF Predicted Close', 'ANN Predicted Close'
     , 'Proposed Model']
]

# Calculate the error metrics and update the data list
for index in result_df.index:
    actual_close = round(result_df.loc[index, 'Actual Close'], 2)
    rr_predicted_close = round(result_df.loc[index, 'RR Predicted Close'], 2)
    svr_predicted_close = round(result_df.loc[index, 'SVR Predicted Close'], 2)
    knn_predicted_close = round(result_df.loc[index, 'KNN Predicted Close'], 2)
    dt_predicted_close = round(result_df.loc[index, 'DT Predicted Close'], 2)
    rf_predicted_close = round(result_df.loc[index, 'RF Predicted Close'], 2)
    ann_predicted_close = round(result_df.loc[index, 'ANN Predicted Close'], 2)
    pm_predicted_close = round(result_df.loc[index, 'Proposed Model'], 2)

    rr_absolute_error = round(abs(actual_close - rr_predicted_close), 2)
    svr_absolute_error = round(abs(actual_close - svr_predicted_close), 2)
    knn_absolute_error = round(abs(actual_close - knn_predicted_close), 2)
    dt_absolute_error = round(abs(actual_close - dt_predicted_close), 2)
    rf_absolute_error = round(abs(actual_close - rf_predicted_close), 2)
    ann_absolute_error = round(abs(actual_close - ann_predicted_close), 2)
    pm_absolute_error = round(abs(actual_close - pm_predicted_close), 2)

    rr_error_percentage = round((rr_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    svr_error_percentage = round((svr_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    knn_error_percentage = round((knn_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    dt_error_percentage = round((dt_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    rf_error_percentage = round((rf_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    ann_error_percentage = round((ann_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
    pm_error_percentage = round((pm_absolute_error / actual_close) * 100, 2) if actual_close != 0 else 0
```

```
    data.append([
        index.strftime('%Y-%m-%d'),
        rr_error_percentage,
        svr_error_percentage,
        knn_error_percentage,
        dt_error_percentage,
        rf_error_percentage,
        ann_error_percentage,
        pm_error_percentage
    ])


# Calculate MAE, MSE, and MAPE
n = len(result_df)
rr_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['RR Predicted Close']) / result_df['Actual Close']) / n, 2)
svr_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['SVR Predicted Close']) / result_df['Actual Close']) / n, 2)
knn_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['KNN Predicted Close']) / result_df['Actual Close']) / n, 2)
dt_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['DT Predicted Close']) / result_df['Actual Close']) / n, 2)
rf_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['RF Predicted Close']) / result_df['Actual Close']) / n, 2)
ann_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['ANN Predicted Close']) / result_df['Actual Close']) / n, 2)
pm_mape = round(100 * sum(abs(result_df['Actual Close'] - result_df['Proposed Model']) / result_df['Actual Close']) / n, 2)


# Update the data list with MAE, MSE, and MAPE
data.append(['MAPE', rr_mape, svr_mape, knn_mape, dt_mape, rf_mape, ann_mape, pm_mape])


# Pop the headers from the data array
```

```
# Update the data list with MAE, MSE, and MAPE
data.append(['MAPE', rr_mape, svr_mape, knn_mape, dt_mape, rf_mape, ann_mape, pm_mape])


# Pop the headers from the data array
column_headers = data.pop(0)
row_headers = [x.pop(0) for x in data]
# Table data needs to be non-numeric text. Format the data
# while I'm at it.
cell_text = []
for row in data:
    formatted_row = []
    for i, x in enumerate(row):
        formatted_row.append(f'{x}%')
    cell_text.append(formatted_row)

# Reverse colors and text labels to display the last value at the top.
# colors = colors[::-1]
# cell_text.reverse()
rcolors = plt.cm.BuPu(np.full(len(row_headers), 0.1))
ccolors = plt.cm.BuPu(np.full(len(column_headers), 0.1))
# Add a table at the bottom of the axes
the_table = plt.table(cellText=cell_text,
                      rowLabels=row_headers,
                      rowColours=rcolors,
                      rowLoc='right',
                      colColours=ccolors,
                      colLabels=column_headers,
                      loc='center',
                      fontsize=20
                      )
```

```
# Adjust layout to make room for the table:
plt.box(on=None)
plt.subplots_adjust(left=0.2, bottom=0.2)
# plt.ylabel("Loss in ${0}'s".format(value_increment))
# plt.yticks(values * value_increment, ['%d' % val for val in values])
plt.xticks([])


# plt.title('Linear Regression Prediction based on 1 week(7 Days) training datasets of Reliance Industries ')
ax = plt.gca()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
# plt.show()
# Create image. plt.savefig ignores figure edge and face colors, so map them.
fig = plt.gcf()
the_table.scale(3, 2)
plt.savefig('5. Errors_mape.png',
            bbox_inches='tight',
            dpi=150
            )
```

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	1.07%	0.27%	0.59%	1.35%	0.01%	6.91%	0.24%
2024-03-04	0.18%	1.53%	0.95%	1.31%	0.57%	7.89%	0.56%
2024-03-05	2.42%	2.23%	2.93%	3.9%	2.89%	0.32%	2.91%
2024-03-06	3.82%	7.37%	4.36%	2.78%	4.04%	5.62%	3.53%
2024-03-07	2.04%	2.63%	3.42%	3.42%	3.24%	5.19%	2.57%
2024-03-11	1.27%	1.49%	2.54%	2.16%	1.05%	1.81%	1.52%
2024-03-12	2.15%	0.8%	2.53%	0.38%	1.01%	0.71%	1.5%
2024-03-13	0.91%	2.54%	1.01%	2.38%	2.39%	10.1%	1.5%
2024-03-14	3.52%	0.16%	0.89%	0.81%	0.59%	0.98%	1.93%
2024-03-15	4.26%	2.69%	2.23%	1.29%	2.44%	5.93%	3.19%
MAPE	2.16%	2.17%	2.14%	1.98%	1.82%	4.55%	1.94%

```
# Initialize the data with headers
data = [
    ['RR Predicted Close', 'SVR Predicted Close', 'KNN Predicted Close', 'DT Predicted Close', 'RF Predicted Close', 'ANN Predicted Close'
    , 'Proposed Model']
]

# Calculate the error metrics and update the data list
for index in result_df.index:
    actual_close = round(result_df.loc[index, 'Actual Close'], 2)
    rr_predicted_close = round(result_df.loc[index, 'RR Predicted Close'], 2)
    svr_predicted_close = round(result_df.loc[index, 'SVR Predicted Close'], 2)
    knn_predicted_close = round(result_df.loc[index, 'KNN Predicted Close'], 2)
    dt_predicted_close = round(result_df.loc[index, 'DT Predicted Close'], 2)
    rf_predicted_close = round(result_df.loc[index, 'RF Predicted Close'], 2)
    ann_predicted_close = round(result_df.loc[index, 'ANN Predicted Close'], 2)
    pm_predicted_close = round(result_df.loc[index, 'Proposed Model'], 2)

    rr_absolute_error = round(abs(actual_close - rr_predicted_close), 2)
    svr_absolute_error = round(abs(actual_close - svr_predicted_close), 2)
    knn_absolute_error = round(abs(actual_close - knn_predicted_close), 2)
    dt_absolute_error = round(abs(actual_close - dt_predicted_close), 2)
    rf_absolute_error = round(abs(actual_close - rf_predicted_close), 2)
    ann_absolute_error = round(abs(actual_close - ann_predicted_close), 2)
    pm_absolute_error = round(abs(actual_close - pm_predicted_close), 2)

    rr_error_square = round(rr_absolute_error ** 2, 2)
    svr_error_square = round(svr_absolute_error ** 2, 2)
    knn_error_square = round(knn_absolute_error ** 2, 2)
    dt_error_square = round(dt_absolute_error ** 2, 2)
    rf_error_square = round(rf_absolute_error ** 2, 2)
    ann_error_square = round(ann_absolute_error ** 2, 2)
    pm_error_square = round(pm_absolute_error ** 2, 2)
```

```
    data.append([
        index.strftime('%Y-%m-%d'),
        rr_error_square,
        svr_error_square,
        knn_error_square,
        dt_error_square,
        rf_error_square,
        ann_error_square,
        pm_error_square,
    ])


# Calculate MAE, MSE, and MAPE
n = len(result_df)
rr_mse = round(sum((result_df['Actual Close'] - result_df['RR Predicted Close']) ** 2) / n, 2)
svr_mse = round(sum((result_df['Actual Close'] - result_df['SVR Predicted Close']) ** 2) / n, 2)
knn_mse = round(sum((result_df['Actual Close'] - result_df['KNN Predicted Close']) ** 2) / n, 2)
dt_mse = round(sum((result_df['Actual Close'] - result_df['DT Predicted Close']) ** 2) / n, 2)
rf_mse = round(sum((result_df['Actual Close'] - result_df['RF Predicted Close']) ** 2) / n, 2)
ann_mse = round(sum((result_df['Actual Close'] - result_df['ANN Predicted Close']) ** 2) / n, 2)
pm_mse = round(sum((result_df['Actual Close'] - result_df['Proposed Model']) ** 2) / n, 2)


# Update the data list with MAE, MSE, and MAPE
data.append(['MSE', rr_mse, svr_mse, knn_mse, dt_mse, rf_mse, ann_mse, pm_mse])
```

```
# Update the data list with MAE, MSE, and MAPE
data.append(['MSE', rr_mse, svr_mse, knn_mse, dt_mse, rf_mse, ann_mse, pm_mse])


# Pop the headers from the data array
column_headers = data.pop(0)
row_headers = [x.pop(0) for x in data]
# Table data needs to be non-numeric text. Format the data
# while I'm at it.
cell_text = []
for row in data:
    formatted_row = []
    for i, x in enumerate(row):
        formatted_row.append(f'{x}' if isinstance(x, (int, float)) else x)
    cell_text.append(formatted_row)

# Reverse colors and text labels to display the last value at the top.
# colors = colors[::-1]
# cell_text.reverse()
rcolors = plt.cm.BuPu(np.full(len(row_headers), 0.1))
ccolors = plt.cm.BuPu(np.full(len(column_headers), 0.1))
# Add a table at the bottom of the axes
the_table = plt.table(cellText=cell_text,
                      rowLabels=row_headers,
                      rowColours=rcolors,
                      rowLoc='right',
                      colColours=ccolors,
                      colLabels=column_headers,
                      loc='center',
                      fontsize=20
                      )
```

```
# Adjust layout to make room for the table:
plt.box(on=None)
plt.subplots_adjust(left=0.2, bottom=0.2)
# plt.ylabel("Loss in ${0}'s".format(value_increment))
# plt.yticks(values * value_increment, ['%d' % val for val in values])
plt.xticks([])


# plt.title('Linear Regression Prediction based on 1 week(7 Days) training datasets of Reliance Industries ')
ax = plt.gca()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
# plt.show()
# Create image. plt.savefig ignores figure edge and face colors, so map them.
fig = plt.gcf()
the_table.scale(3, 2)
plt.savefig('5. Errors_mse.png',
            bbox_inches='tight',
            dpi=150
            )
```

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	145.93	9.24	45.02	232.56	0.02	6087.12	7.08
2024-03-04	4.16	299.98	114.92	219.04	41.22	8006.67	39.69
2024-03-05	798.63	676.52	1173.75	2074.8	1142.44	14.36	1156.0
2024-03-06	2080.27	7738.72	2710.24	1098.92	2330.96	4499.73	1775.78
2024-03-07	596.34	994.14	1681.0	1685.1	1511.65	3880.04	952.34
2024-03-11	231.04	317.2	923.55	670.81	158.0	470.02	329.42
2024-03-12	669.26	93.32	927.2	20.7	148.11	73.62	327.61
2024-03-13	113.64	882.09	138.3	775.62	783.44	13926.36	308.0
2024-03-14	1771.57	3.61	111.94	93.12	49.14	137.59	531.76
2024-03-15	2704.0	1076.5	740.93	248.06	883.28	5230.18	1513.21
MSE	911.56	1209.23	856.68	711.87	704.79	4232.49	694.08

```
# Initialize the data with headers
data = [
    ['RR Predicted Close', 'SVR Predicted Close', 'KNN Predicted Close', 'DT Predicted Close', 'RF Predicted Close', 'ANN Predicted Close'
    , 'Proposed Model']
]

# Calculate the error metrics and update the data list
for index in result_df.index:
    actual_close = round(result_df.loc[index, 'Actual Close'], 2)
    rr_predicted_close = round(result_df.loc[index, 'RR Predicted Close'], 2)
    svr_predicted_close = round(result_df.loc[index, 'SVR Predicted Close'], 2)
    knn_predicted_close = round(result_df.loc[index, 'KNN Predicted Close'], 2)
    dt_predicted_close = round(result_df.loc[index, 'DT Predicted Close'], 2)
    rf_predicted_close = round(result_df.loc[index, 'RF Predicted Close'], 2)
    ann_predicted_close = round(result_df.loc[index, 'ANN Predicted Close'], 2)
    pm_predicted_close = round(result_df.loc[index, 'Proposed Model'], 2)

    rr_absolute_error = round(abs(actual_close - rr_predicted_close), 2)
    svr_absolute_error = round(abs(actual_close - svr_predicted_close), 2)
    knn_absolute_error = round(abs(actual_close - knn_predicted_close), 2)
    dt_absolute_error = round(abs(actual_close - dt_predicted_close), 2)
    rf_absolute_error = round(abs(actual_close - rf_predicted_close), 2)
    pm_absolute_error = round(abs(actual_close - pm_predicted_close), 2)

    data.append([
        index.strftime('%Y-%m-%d'),
        rr_absolute_error,
        svr_absolute_error,
        knn_absolute_error,
        dt_absolute_error,
        rf_absolute_error,
        ann_absolute_error,
        pm_absolute_error
```

```
    data.append([
        index.strftime('%Y-%m-%d'),
        rr_absolute_error,
        svr_absolute_error,
        knn_absolute_error,
        dt_absolute_error,
        rf_absolute_error,
        ann_absolute_error,
        pm_absolute_error

    ])


# Calculate MAE, MSE, and MAPE
n = len(result_df)
rr_mae = round(sum(abs(result_df['Actual Close'] - result_df['RR Predicted Close'])) / n, 2)
svr_mae = round(sum(abs(result_df['Actual Close'] - result_df['SVR Predicted Close'])) / n, 2)
knn_mae = round(sum(abs(result_df['Actual Close'] - result_df['KNN Predicted Close'])) / n, 2)
dt_mae = round(sum(abs(result_df['Actual Close'] - result_df['DT Predicted Close'])) / n, 2)
rf_mae = round(sum(abs(result_df['Actual Close'] - result_df['RF Predicted Close'])) / n, 2)
ann_mae = round(sum(abs(result_df['Actual Close'] - result_df['ANN Predicted Close'])) / n, 2)
pm_mae = round(sum(abs(result_df['Actual Close'] - result_df['Proposed Model'])) / n, 2)
```

```
# Update the data list with MAE, MSE, and MAPE
data.append(['MAE', rr_mae, svr_mae, knn_mae, dt_mae, rf_mae, ann_mae, pm_mae])


# Pop the headers from the data array
column_headers = data.pop(0)
row_headers = [x.pop(0) for x in data]
# Table data needs to be non-numeric text. Format the data
# while I'm at it.
cell_text = []
for row in data:
    formatted_row = []
    for i, x in enumerate(row):
        formatted_row.append(f'{x}' if isinstance(x, (int, float)) else x)
    cell_text.append(formatted_row)

# Reverse colors and text labels to display the last value at the top.
# colors = colors[::-1]
# cell_text.reverse()
rcolors = plt.cm.BuPu(np.full(len(row_headers), 0.1))
ccolors = plt.cm.BuPu(np.full(len(column_headers), 0.1))
# Add a table at the bottom of the axes
the_table = plt.table(cellText=cell_text,
                      rowLabels=row_headers,
                      rowColours=rcolors,
                      rowLoc='right',
                      colColours=ccolors,
                      colLabels=column_headers,
                      loc='center',
                      fontsize=20
                      )
```

```
                    )
# Adjust layout to make room for the table:
plt.box(on=None)
plt.subplots_adjust(left=0.2, bottom=0.2)
# plt.ylabel("Loss in ${0}'s".format(value_increment))
# plt.yticks(values * value_increment, ['%d' % val for val in values])
plt.xticks([])


# plt.title('Linear Regression Prediction based on 1 week(7 Days) training datasets of Reliance Industries ')
ax = plt.gca()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
# plt.show()
# Create image. plt.savefig ignores figure edge and face colors, so map them.
fig = plt.gcf()
the_table.scale(3, 2)
plt.savefig('5. Errors_mae.png',
            bbox_inches='tight',
            dpi=150
            )
```

	RR Predicted Close	SVR Predicted Close	KNN Predicted Close	DT Predicted Close	RF Predicted Close	ANN Predicted Close	Proposed Model
2024-03-01	12.08	3.04	6.71	15.25	0.14	72.32	2.66
2024-03-04	2.04	17.32	10.72	14.8	6.42	72.32	6.3
2024-03-05	28.26	26.01	34.26	45.55	33.8	72.32	34.0
2024-03-06	45.61	87.97	52.06	33.15	48.28	72.32	42.14
2024-03-07	24.42	31.53	41.0	41.05	38.88	72.32	30.86
2024-03-11	15.2	17.81	30.39	25.9	12.57	72.32	18.15
2024-03-12	25.87	9.66	30.45	4.55	12.17	72.32	18.1
2024-03-13	10.66	29.7	11.76	27.85	27.99	72.32	17.55
2024-03-14	42.09	1.9	10.58	9.65	7.01	72.32	23.06
2024-03-15	52.0	32.81	27.22	15.75	29.72	72.32	38.9
MAE	25.82	25.78	25.51	23.35	21.7	53.3	23.17

```
# Update the columns of the `data` list
errors = [
    ['MAE', 'MSE', 'MAPE']
]
errors.append(['LR', rr_mae, rr_mse, rr_mape])
errors.append(['SVM', svr_mae, svr_mse, svr_mape])
errors.append(['KNN', knn_mae, knn_mse, knn_mape])
errors.append(['DT', dt_mae, dt_mse, dt_mape])
errors.append(['RF', rf_mae, rf_mse, rf_mape])
errors.append(['ANN', ann_mae, ann_mse, ann_mape])
errors.append(['PM', pm_mae, pm_mse, pm_mape])

# Pop the headers from the data array
column_headers = errors.pop(0)
row_headers = [x.pop(0) for x in errors]
# Table data needs to be non-numeric text. Format the data
# while I'm at it.
cell_text = []

for row in errors:
    cell_text.append([f'{x}' for x in row])
# Reverse colors and text labels to display the last value at the top.
# colors = colors[::-1]
# cell_text.reverse()
rcolors = plt.cm.BuPu(np.full(len(row_headers), 0.1))
ccolors = plt.cm.BuPu(np.full(len(column_headers), 0.1))
# Add a table at the bottom of the axes
the_table = plt.table(cellText=cell_text,
                      rowLabels=row_headers,
                      rowColours=rcolors,
                      rowLoc='right',
                      colColours=ccolors,
                      colLabels=column_headers,
```

```
# Adjust layout to make room for the table:
plt.box(on=None)
plt.subplots_adjust(left=0.2, bottom=0.2)
# plt.ylabel("Loss in ${0}'s".format(value_increment))
# plt.yticks(values * value_increment, ['%d' % val for val in values])
plt.xticks([])
# plt.title('Linear Regression Prediction based on 1 week(7 Days) training datasets of Reliance Industries ')
ax = plt.gca()
ax.get_xaxis().set_visible(False)
ax.get_yaxis().set_visible(False)
# plt.show()
# Create image. plt.savefig ignores figure edge and face colors, so map them.
fig = plt.gcf()
the_table.scale(0.5, 2)
plt.savefig('10. Error Comparison.png',
            bbox_inches='tight',
            dpi=150
            )
```

	MAE	MSE	MAPE
LR	25.82	911.56	2.16
SVM	25.78	1209.23	2.17
KNN	25.51	856.68	2.14
DT	23.35	711.87	1.98
RF	21.7	704.79	1.82
ANN	53.3	4232.49	4.55
PM	23.17	694.08	1.94

Future Work on Short-Term Stock Market Prediction

Future work on short-term stock market prediction can explore several promising avenues to enhance the accuracy and applicability of predictive models. One key area is the incorporation of additional features, such as alternative data sources including social media sentiment, news articles, and macroeconomic indicators, which can provide more comprehensive insights into market behavior. Additionally, exploring and including more technical indicators and patterns may further improve predictive capabilities.

Another area of future work is the development of real-time prediction systems capable of processing streaming data to provide up-to-the-minute stock price predictions. Integrating these predictive models into algorithmic trading systems can test and validate their effectiveness in live trading environments. Additionally, implementing explainable AI techniques will make the predictions of complex models more transparent and understandable to traders and investors, creating user-friendly decision support systems that present model predictions along with explanations and confidence intervals.

Finally, developing systems that allow for user feedback and interaction can enable continuous improvement of the predictive models based on real-world performance and user input. Creating customizable prediction models tailored to individual user preferences and trading strategies can also provide more personalized and effective tools for traders and investors.

By pursuing these future directions, the field of short-term stock market prediction can advance significantly, providing more accurate, reliable, and actionable insights for traders and investors.

19. BIBLOGRAPHY

[1] Abu-Mostafa, Y. S., & Atiya, A. F. (1996).*''Introduction to financial forecasting''*

[2] Atsalakis, G. S., & Valavanis, K. P. (2009) *''Surveying stock market forecasting techniques – Part I: Conventional methods.''*

[3]Bachelier, L. (1900) *"Theory of Speculation.Paris: Gauthier-Villars. (Reprinted in The Random Character of Stock Market Prices",* MIT Press, 1964).

[4] Bengio, Y., Simard, P., & Frasconi, P. (1994*)." Learning long-term dependencies with gradient descent is difficult."* -IEEE Transactions on Neural Networks

[5] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). *"Time Series Analysis: Forecasting and Control."* Hoboken, NJ: John Wiley & Sons.

[6] Chen, K. Y., & Wang, C. H. (2007).*"Support vector regression with genetic algorithms in forecasting tourism demand"*

[7] Chollet, F. (2017).*"Deep Learning with Python" Shelter Island"*

[8] Fausett, L. V. (1994).*"Fundamentals of Neural Networks: Architectures, Algorithms, and Applications"* Englewood Cliffs, NJ: Prentice-Hall.

[9] Gately, E. (1996)."Neural Networks for Financial Forecasting. "John Wiley & Sons.

[10] Goodfellow, I., Bengio, Y., & Courville, A. (2016).*"Deep Learning."* Cambridge, MA: MIT Press.

[11] Jain, A. K., Duin, R. P., & Mao, J. (2000*)."Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence"*

[12] Kingma, D. P., & Ba, J. (2015).Adam: "*A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations (ICLR)"*.

[13] Murphy, J. J. (1999)."Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications",York Institute of Finance.

[14] Nelson, D. B. (1991) "Conditional heteroskedasticity in asset returns: A new approach."

[15] Patterson, D. W. (1996)."Artificial Neural Networks: Theory and Applications."- Englewood Cliffs, NJ: Prentice-Hall.

[16] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986).*"Learning representations by back-propagating errors."*

[17] Schmidhuber, J. (2015*).-"Deep learning in neural networks: An overview. Neural Networks*,

[18] Shadbolt, N., O'Hara, K., & Hall, W. (2006). *"The semantic web revisited. IEEE Intelligent Systems"*