Stock Price Prediction Using Machine Learning: An Ensemble Approach

Nuka Nandan Vignesh - 21BRS1473
Computer Science Engineering (CSE) specialization in Artificial Intelligence (AI) and Robotics
Under
Dr. Shenbaga Velu P - 53640
VIT chennai-600127

Abstract - One of the major challenges in the financial sector is predicting the stock price. This paper attempts to look at the machine learning models-Decision Tree, K-Nearest Neighbors, and Support Vector Machine - to predict stock prices. Performance enhancement is made using ensemble methods, including simple averaging, weighted averaging, and gradient boosting. Yahoo Finance sources historical stock data from 2022 to 2024. Features used include the Close price, Simple Moving Average, Exponential Moving Average, Bollinger Bands, Relative Strength Index, Average True Range, and Moving Average Convergence Divergence. Mean Squared Error, Mean Absolute Error, R-squared, and accuracy are the evaluation metrics for model comparison. Results of the experiment suggest that ensemble techniques outperform individual models, demonstrating the effectiveness of combining weak learners for improved prediction accuracy.

Keywords - Ensemble learning, Support Vector Machines, KNN, Decision tree, yfinance, Random Forest and Back-testing

1. INTRODUCTION

The global financial landscape has witnessed significant growth in the number of retail investors over recent years. With the advent of technology and the proliferation of online trading platforms, individuals are now more empowered than ever to participate in the stock market. This surge in investment activity has increased the demand for tools and methodologies that can assist investors in making informed decisions. Consequently, the accuracy of stock price predictions has become paramount, as it directly influences investment strategies and potential returns. Investment in the stock market plays a crucial role in wealth creation and economic growth. As companies seek to expand, they rely on public investments to raise capital for development projects. Investors, on the other hand, look for opportunities that not only yield financial returns but also contribute to the overall economy. The increasing complexity of financial markets, coupled with external factors such as economic indicators, geopolitical events, and market sentiment, makes the task of predicting stock prices particularly challenging.

Traditionally, investors relied on fundamental analysis and technical indicators to make predictions about future stock prices. Fundamental analysis involves evaluating a company's financial health through its earnings, assets, and growth potential, while technical analysis focuses on historical price patterns and trading volumes. However, these methods often fall short in capturing the intricacies and dynamics of the market, especially in volatile environments.

Accurately predicting stock prices has long been challenging for both individual investors and financial institutions, as the stock market is inherently complex and influenced by a myriad of factors. In this research paper, we explore the use of an ensemble approach, combining multiple machine learning models, to enhance the accuracy of stock price prediction. Stock price prediction is a crucial task as it allows investors to make informed decisions and minimize investment risks[14]. Machine learning techniques, particularly supervised learning algorithms, have shown promising results in this domain.

This paper investigates the application of ensemble learning to stock price prediction using Decision Tree, KNN, and SVM models. These models were selected for their simplicity and effectiveness in handling regression tasks. The dataset's used in this study, sourced from Yahoo Finance, encompasses historical data from January 2022 to November 2024. A range of technical indicators, including SMA, EMA, Bollinger Bands, RSI, ATR, and MACD, is utilized to provide comprehensive feature representation. The ensemble approach, which combines multiple algorithms, further enhances predictive performance. By leveraging the strengths of various models, the ensemble method can mitigate the weaknesses of individual algorithms. This collective decision-making process results in a more robust prediction framework, capable of adapting to the inherent volatility of financial markets. The integration of sentiment analysis—analyzing news articles and social media to gauge public sentiment—adds another layer of depth to the prediction process, as market sentiment often drives stock price movements.

The objective of this research is to demonstrate how ensemble techniques can outperform standalone models in terms of accuracy and reliability. The results presented in this paper underscore the advantages of combining weak learners to generate more robust predictions. Additionally, the study offers insights into the practical implementation of ensemble methods, contributing to the growing body of knowledge on stock price forecasting through machine learning. By demonstrating the potential of machine learning techniques in stock price prediction, this research seeks to contribute to the growing body of knowledge in financial analytic. The findings will not only benefit individual investors but also provide insights for financial institutions and analysts striving to refine their predictive strategies in an increasingly complex market environment. The implications of this research underscore the importance of integrating advanced technologies in investment decision-making processes.

2. RELATED WORKS

[1] The document presents a study on stock selection strategies using the random forest (RF) algorithm in the Chinese stock market, comparing multi-factor strategies (which incorporate various fundamental and technical factors) with momentum strategies (focused on price trends). It finds that momentum strategies generally outperform multi-factor strategies in the short term and highlights the role of machine learning in identifying profitable trading patterns. The study identifies key factors influencing stock returns, such as earnings-to-price and book-to-price ratios, and demonstrates that combining fundamental and technical features can enhance stock classification and lead to consistent outperformance. The authors suggest that future research could explore more effective feature spaces and develop novel machine learning algorithms for stock selection.

[2] This article explores the use of various machine learning techniques, including deep neural networks (DNN), gradient-boosted trees (GBT), random forests (RF), and ensembles of these methods, in statistical arbitrage within finance. The models are trained using lagged returns from all S&P 500 stocks, with survivor bias removed. The goal is to predict which stocks are likely to outperform the market over a one-day horizon from 1992 to 2015. The top predictions are used for long positions, and the lowest for short positions. The study finds that a simple ensemble model (combining DNN, GBT, and RF) generates promising returns of over 0.45% per day (before transaction costs) when selecting the top and bottom 10 stocks. These results challenge the semi-strong form of market efficiency, as profits persist even with declining returns in recent years.

[3] In this manuscript, we present the various forecasting approaches and linear regression algorithm to successfully predict the Bombay Stock Exchange (BSE) SENSEX value with high accuracy. Depending upon the analysis performed, it can be said successfully that Linear Regression in combination with different mathematical functions prepares the best model. This model gives the best output with BSE SENSEX values and Gross Domestic Product (GDP) values as it shows the least p-value as 5.382e−10 when compared with other model's p-values.

[4] In the last few years, researchers have utilized machine learning techniques for learning the trends of stock market in order to improve the accuracy of predictions. However, authors have applied these techniques individually and compared their results. Since the aggregated opinion of a group of models is relatively less noisy as compared to the single opinion of one of the models, this paper presents an ensemble machine learning approach for predicting the stock market. The weighted ensemble model is built using weighted support vector regression (SVR), Long-short term memory (LSTM) and Multiple Regression. From the results it is observed that ensemble learning approach is able to attain maximum accuracy with reduced variance and hence better predictions.

[5] The purpose of this paper is to benchmark ensemble methods (Random Forest, AdaBoost and Kernel Factory) against single classifier models (Neural Networks, Logistic Regression, Support Vector Machines and K-Nearest Neighbor. We gathered data from 5767 publicly listed European companies and used the area under the receiver operating characteristic curve (AUC) as a performance measure. Our predictions are one year ahead. The results indicate that Random Forest is the top algorithm followed by Support Vector Machines, Kernel Factory, AdaBoost, Neural Networks, K-Nearest Neighbors and Logistic Regression. This study contributes to literature in that it is, to the best of our knowledge, the first to make such an extensive benchmark. The results clearly suggest that novel studies in the domain of stock price direction prediction should include ensembles in their sets of algorithms. Our extensive literature review evidently indicates that this is currently not the case.

[6] The stock market plays a crucial role in growing economies, and every investment aims to maximize profit while minimizing risk. Many studies have been conducted on predicting the stock market using different methods, including technical and fundamental analysis, through machine learning techniques. This study reviewed 122 research papers published between 2007 and 2018, focusing on stock market prediction using machine learning. The papers were categorized into three groups: technical analysis (66%), fundamental analysis (23%), and combined analysis (11%). The review also found that most studies (89%) used data from a single source, while a smaller percentage used data from two or three sources. The most commonly used machine learning algorithms in these studies were support vector machines and artificial neural networks.

[7] This paper compares the effectiveness of machine learning and technical analysis in predicting the stock market and generating returns. The study backtests both methods using data from January 1995 to December 2005 to predict stock movements for the next ten years, from January 2006 to December 2016. After making predictions, the research uses the results to create trading strategies aimed at outperforming the S&P 500 index. The study looks at all market conditions and also examines performance during periods of rising and falling markets. It tests the hypothesis that there is no significant difference in returns between machine learning and technical analysis. The research uses State Street's SPDR® SPY ETF as a

benchmark and gathers data from Bloomberg and Yahoo Finance, with calculations done using software like R, MATLAB, SPSS, EVIEWS, Python, and SAS.

[8] This paper states that it introduces a modified version of Yu (2011)'s weighted bagging estimation method, which greatly improves the predictability of the equity premium and other economic variables. The new machine learning approach enhances the accuracy of equity premium predictions across various models, with significant out-of-sample R2 increases of up to nearly 3% per month and annual utility gains of over 3.5%. The improvement is mainly due to better performance during economic recessions, market downturns, and periods of turbulence, as well as the method's increased diversity and built-in shrinkage. The study also finds that variables related to interest rates have the strongest ability to predict the equity premium.

[9] This paper states that it proposes a new model, StockSentiWordNet (SSWN), to predict stock market behavior by analyzing Twitter posts and Google Finance data. The SSWN model extends the standard SentiWordNet lexicon by adding terms specific to the stock market, and it uses this data to train extreme learning machine (ELM) and recurrent neural network (RNN) models for stock price prediction. The study tests the model on two datasets, Sentiment140 and Twitter, and achieves an accuracy of 86.06%. The results show that this approach outperforms other existing methods in terms of accuracy. The paper also suggests plans to enhance the model by including data from other social media platforms, such as Facebook and Google News, in the future.

[10] This paper discusses the challenges in stock market prediction using machine learning, focusing on the choice of base techniques, combination methods, and the number of classifiers or regressors used in ensemble models. The study compares different ensemble techniques such as boosting, bagging, blending, and stacking, using Decision Trees (DT), Support Vector Machine (SVM), and Neural Networks (NN) on stock data from multiple exchanges, including the Ghana Stock Exchange, Johannesburg Stock Exchange, Bombay Stock Exchange, and New York Stock Exchange, from January 2012 to December 2018. The results show that stacking and blending techniques provide the highest prediction accuracy (90–100% and 85.7–100%, respectively), outperforming bagging and boosting, which had lower accuracy rates. Additionally, stacking and blending also showed better root mean square errors (RMSE), indicating a better fit for stock market predictions. The paper concludes that ensemble techniques should be included in future studies for predicting stock market directions.

[11] This paper introduces a new technique to improve the accuracy of decision tree forests, which are popular for their strong performance in machine learning. The proposed method enhances the mixture of individual decision trees by training each tree with different sets of rotation spaces, which are then linked to a higher space at the parent node. The trees search for optimal splits within this elevated space, and the rotation technique selection depends on the best split found. The research also integrates various sources of information, such as social media, global news, financial news, and historical data, to improve predictions for the Indian stock market indices. The results show that this method provides significant accuracy in forecasting market trends.

[12] This paper introduces a new technique called GASVM, which combines Support Vector Machine (SVM) with a Genetic Algorithm (GA) to improve stock price prediction. While SVM is popular in machine learning for predicting stock prices, it often suffers from overfitting when dealing with noisy, high-dimensional data. The GASVM model uses the GA to optimize feature selection and SVM kernel parameters simultaneously. The study, which analyzed over 11 years of stock data from the Ghana Stock Exchange, shows that GASVM outperforms other machine learning algorithms like Decision Tree, Random Forest, and Neural Network in predicting stock price movements 10 days ahead. GASVM achieved a prediction accuracy of 93.7%, significantly higher than the others, proving that it offers an effective solution for feature selection and parameter optimization in stock market prediction.

[13] This paper focuses on predicting the Nifty 50 Index using eight supervised machine learning models: Adaptive Boost (AdaBoost), k-Nearest Neighbors (kNN), Linear Regression (LR), Artificial Neural Network (ANN), Random Forest (RF), Stochastic Gradient Descent (SGD), Support Vector Machine (SVM), and Decision Trees (DT). The study uses historical data of the Nifty 50 Index from the Indian stock market, spanning 25 years (22nd April 1996 to 16th April 2021), with a total of 6220 trading days. The data was divided into subsets of different sizes (25%, 50%, 75%, and 100%) for training and testing. After applying tests on training data, testing data, and cross-validation, the performance of each model was compared. The results showed that Adaptive Boost, kNN, Random Forest, and Decision Trees performed worse as the dataset size increased, while Linear Regression and ANN performed similarly across all models, though ANN took more time for training and validation. Support Vector Machine performed well, but Stochastic Gradient Descent outperformed SVM as the dataset size grew.

[14] This paper reviews studies on the use of supervised machine learning models for stock price prediction, a complex challenge that involves the unpredictable behavior of stock price time series. Stock price prediction is important for investors, managers, and decision-makers who need accurate models to make informed choices. The paper highlights that Support Vector Machine (SVM) is the most commonly used technique for stock price prediction because of its strong performance and accuracy. It also discusses other techniques such as Artificial Neural Networks (ANN), K-Nearest Neighbors (KNN), Naïve Bayes, Random Forest, Linear Regression, and Support Vector Regression (SVR), all of which have shown promising results in predicting stock prices.

[15] This paper focuses on predicting stock prices using machine learning algorithms, aiming to replace traditional time-series forecasting methods that are often unreliable. Stock prices are unpredictable and risky, making it important for researchers to apply machine learning to analyze them. The study uses historical data of a firm to uncover patterns and improve prediction accuracy. The paper presents an empirical study comparing the performance of several machine learning algorithms, including Decision Tree, Linear Regression, K-Nearest Neighbors, and LSTM (Long Short-Term Memory), to determine which algorithm is the most effective at predicting stock prices.

3. PROBLEM DEFINATION

The stock prices are highly volatile and dynamic in nature, which makes it quite a complex and challenging task to predict the same accurately. Traditional methods usually fail to capture the intricate patterns and non-linear relationships present in the stock data, leading to suboptimal performance. Investors and traders are on the lookout for more reliable methods of forecasting price movements to minimize risks and make better decisions. While promising, machine learning models are often inconsistent when used alone, rendering them ineffective for making accurate and stable predictions.

Moreover, there are many factors that influence this system, such as market sentiment, economic indicators, and global events, which cannot be modeled using traditional techniques. This research study aims to combine ensemble methods with the goal of overcoming the constraints of individual models and improving stock price prediction accuracy. The current study focuses on combining Decision Tree, KNN, and SVM models to come up with powerful predictive frameworks with techniques such as simple averaging, weighted averaging, and gradient boosting. The key is to develop a practical and scalable method applicable for any given stocks and their attendant market conditions and to thereby contribute to stronger financial forecasting.

4. METHODOLOGY

4.1. Dataset Discription:

4.1.1. Source of Data

The dataset for this project was obtained using the yfinance library, which provides historical stock price data directly from Yahoo Finance. The dataset includes daily stock price information such as Open, High, Low, Close, and Volume for a specified stock over a specific time period.

4.1.2. Features used

Feature	Formula
Close	The closing price of the stock on a given day. This is usually provided directly by the dataset.
SMA	The SMA is calculated by averaging the stock's closing prices over a specified number of periods, such as 20 days.
EMA	The EMA is a weighted moving average where more recent prices are given more importance, calculated using a smoothing factor.
Upper Band	The upper band is derived by adding a specified multiple of the stock's standard deviation to the SMA.
Lower band	The lower band is derived by subtracting a specified multiple of the stock's standard deviation from the SMA.
RSI	RSI measures the speed and change of price movements, calculated by comparing the average gains and losses over a specified period.
ATR	ATR measures volatility by calculating the average of the true range over a specified period, considering price gaps and fluctuations.
MACD	MACD is calculated by subtracting the 26-period EMA from the 12-period EMA, often used with a 9-day EMA signal line.

Table 1: Description of Features and Their Calculation Methods

4.2. Data Handling:

4.2.1. Data Preprocessing and Cleaning:

Data preprocessing is an essential step in the data mining process that helps prepare data for analysis. It involves cleaning, transforming, and integrating data to ensure it is accurate, consistent, and reliable. This step is important because high-quality data leads to better insights and more trustworthy results.

The cleaning phase of data preprocessing focuses on fixing issues like missing values, errors, and inconsistencies. Missing values can be handled by filling them in with average values (mean, median, or mode) or using more advanced methods like interpolation. Outliers, which are unusual data points that don't match the rest of the data, can be identified and dealt with by removing them, adjusting their values, or using statistical techniques. Finally, standardization ensures that data is consistent by checking formats, units, and encoding, and applying techniques like normalization or scaling to make sure all data is on the same level.

Once the dataset was preprocessed, it was ready for model building. With the training and testing sets in place, machine learning algorithms could be applied to the data. The training set was used to teach the model the patterns and relationships between the features, while the testing set was reserved to evaluate the model’s performance on unseen data. This division helps prevent overfitting, ensuring the model generalizes well to new, real-world data.

During the modeling phase, different algorithms, such as decision trees, linear regression, or more advanced methods, might be tested to see which best fits the data. The model’s performance is typically evaluated using various metrics such as accuracy, precision, recall, or mean squared error, depending on the type of problem being solved (e.g., classification or regression). Finally, once a suitable model is selected, it can be fine-tuned and deployed for making predictions or generating insights from new data.

4.2.2. Feature Engineering

Feature engineering plays a vital role in improving the performance of machine learning models by enhancing the input data with more informative features. One key aspect of feature engineering is feature creation, where new features are derived from the existing ones to capture additional patterns or relationships within the data. For example, interaction terms can be created by combining two or more features to explore potential dependencies between them, while polynomial features can help model non-linear relationships. This process expands the feature space and allows the model to learn from a richer set of variables, often leading to better predictive performance.

Another important technique in feature engineering is feature selection, which focuses on identifying and keeping only the most relevant features. By selecting the most important features, we can reduce the dimensionality of the dataset and prevent the model from becoming overly complex. Too many irrelevant features can lead to overfitting, where the model learns noise instead of useful patterns. Feature selection techniques, such as backward elimination or regularization, help improve model performance by ensuring that only the most informative features are used in the learning process.

Feature scaling is another crucial aspect of feature engineering, especially when working with machine learning algorithms that are sensitive to the scale of input data, such as linear regression, support vector machines, and k-nearest neighbors. Different features may have different units or magnitudes (e.g., age in years versus income in thousands), which can cause some features to dominate the model. Scaling ensures that all features contribute equally by transforming them to a common range, such as by using normalization (scaling values between 0 and 1) or standardization (scaling to have a mean of 0 and a standard deviation of 1). This step is essential for models that rely on distance calculations or gradient-based optimization.

Incorporating these feature engineering techniques can significantly improve the effectiveness of machine learning models. By creating new features, selecting relevant ones, and scaling them appropriately, we can enhance the model's ability to generalize and make accurate predictions on unseen data. Effective feature engineering is often the difference between a model that performs well and one that struggles to capture meaningful patterns, making it a key part of the data science workflow.

4.2.3. Data Integration

Data integration is the process of combining data from multiple sources into a single, unified dataset, which is essential for creating a comprehensive view of the information. This process is often challenging because data from different sources may vary in format, structure, and quality. For example, data from one source might be stored in CSV format while another might use a database, and the fields in each source may be named differently or represent different units of measurement. To address these challenges, data integration techniques such as merging, joining, and concatenating are used to bring the data together in a meaningful way.

Merging involves combining datasets based on a common key or identifier, such as a customer ID or product code, to ensure that the data aligns correctly across sources. Joining is similar but may involve different types of joins, such as inner joins, left joins, or outer joins, depending on how much of the data from each source needs to be included. Concatenating, on the other hand, involves stacking datasets together, either vertically or horizontally, when they have the same columns or similar structures. These techniques help in creating a coherent dataset where all the relevant information is consolidated, ensuring that the integrated data can be used effectively for analysis, modeling, or reporting.

4.3. Overview of Algorithms:

4.3.1. Decision Tree

A Decision Tree is a supervised learning algorithm that excels in modeling decisions and their consequences through a tree-like structure. In stock price prediction, it splits historical data based on features such as moving averages, RSI, and trading volume, allowing it to uncover patterns in stock price movements. Each decision rule or "branch" in the tree leads to a specific outcome, helping to model the often complex and non-linear relationships in financial data. One of the key advantages of Decision Trees is their interpretability—they offer clear decision paths, making it easy for analysts to understand and explain how predictions are made. Additionally, they can automatically select the most important features, making them effective for dealing with large and complex datasets.

However, Decision Trees come with certain drawbacks. If not properly pruned, they are prone to overfitting, especially when the tree is deep, leading to poor generalization on unseen data. Furthermore, small variations in the data can result in significant changes in the tree's structure, which can impact model stability. While Decision Trees are widely used for classification tasks, they may not perform as well for regression problems like stock price prediction, where the target variable is continuous. This makes them less accurate in predicting precise numerical values for stock prices, as they are inherently better suited to discrete outcomes.

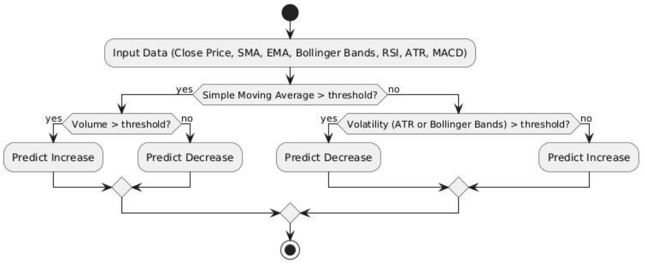
Input Data (Close Price, SMA, EMA, Bollinger Bands, RSI, ATR, MACD)
yes
Simple Moving Average > threshold?
no
yes
Volume > threshold?
no
yes
Volatility (ATR or Bollinger Bands) > threshold?
no
Predict Increase
Predict Decrease
Predict Decrease
Predict Increase

Fig 1: Decision Tree Flowchart

A Decision Tree in stock price prediction works by checking different features of the data, one at a time, and following specific paths based on certain conditions. For example, it first looks at the Simple Moving Average (SMA) of the stock. If the SMA is high, the model will follow one branch; if it is low, it will follow another. This helps the model decide which factors to focus on when making a prediction about the price trend. By comparing the stock's past behavior using these features (like Close price, SMA, EMA, Bollinger Bands, RSI, ATR, and MACD), the decision tree tries to predict whether the price will increase or decrease in the future.

The decision tree starts by checking whether the SMA is above or below a certain threshold. If the SMA is high, it next looks at the volume of trading. If the volume is high, the model predicts an increase in price; if the volume is low, it predicts a decrease. If the SMA is low, the model checks the volatility (using features like ATR or Bollinger Bands) to make a prediction. This process helps identify patterns in stock prices based on past data and can be used to predict future price movements.

4.3.2. K-Nearest Neighbour

K-Nearest Neighbors (KNN) is a simple algorithm used for classifying data points based on the closest "k" data points in a feature space. In stock price prediction, KNN works by finding historical data points that are most similar to the current stock data. It assumes that the price trend of the closest historical matches will predict the future price movement. KNN is good for recognizing patterns, as it can identify trends in stock prices by looking at how similar past behaviors have played out in the future.

One of the main advantages of KNN is its simplicity—it's easy to implement and understand. It also works well for pattern recognition, which is important in stock price prediction. Since KNN is a "lazy learner," it doesn't need a training phase, saving time on model preparation. However, KNN can be slow with large datasets because it compares every data point with each new query. It is also sensitive to irrelevant features, meaning proper feature scaling is important. Additionally, KNN may struggle if there's an imbalance in the number of data points across different classes, which can affect its performance.

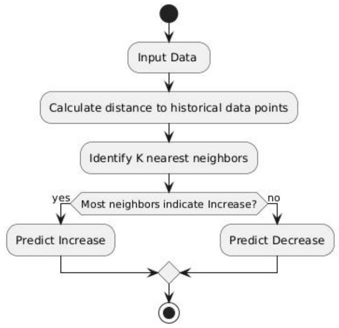
Input Data
Calculate distance to historical data points
Identify K nearest neighbors
yes
Most neighbors indicate Increase?
no
Predict Increase
Predict Decrease

Fig 2: KNN Flowchart

In the KNN approach for stock price prediction, the algorithm compares today's stock data to historical data and looks for the most similar past days, called "neighbors." It uses similarity measurements to identify which days in the past were most alike to today. Once the nearest historical days are found, KNN checks the price trend (increase or decrease) among those neighbors. The model then predicts that the future price trend will follow the majority trend of the closest matches.

This method works well for predicting stock movements because stock patterns often repeat in similar market conditions. When the market behaves in a way similar to past events, KNN can help identify those repeating patterns and make more accurate predictions about future stock price trends.

4.3.3. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm that works by finding an optimal hyperplane that separates data into different classes. In stock price prediction, SVM can be used to classify historical stock data and identify various market conditions, such as "bullish" (rising market) or "bearish" (falling market). SVM is particularly effective for analyzing complex financial data, and when kernel functions are applied, it can handle non-linear relationships, making it powerful for stock market predictions where data doesn't follow a simple linear pattern.

One of the key advantages of SVM is its ability to perform well in high-dimensional spaces, making it ideal for stock prediction where many features (like moving averages, RSI, and volume) need to be considered. The use of kernel functions allows SVM to deal with non-linear relationships effectively. Additionally, SVM uses regularization to prevent overfitting, ensuring that the model generalizes well to new data. However, SVM has some drawbacks, including high memory usage, particularly with large datasets. Choosing the right kernel and tuning its parameters can be tricky, and the algorithm can become slow when processing large amounts of stock data, which can limit its usefulness in some stock analysis scenarios.

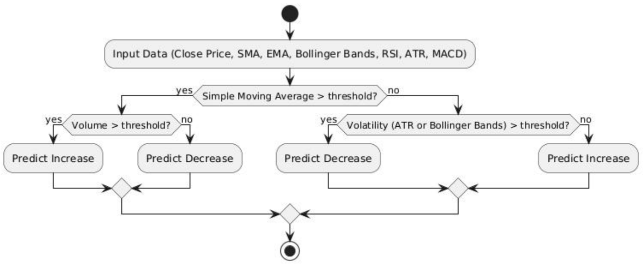
Input Data (Close Price, SMA, EMA, Bollinger Bands, RSI, ATR, MACD)
yes
Simple Moving Average > threshold?
no
yes
Volume > threshold?
no
yes
Volatility (ATR or Bollinger Bands) > threshold?
no
Predict Increase
Predict Decrease
Predict Decrease
Predict Increase

Fig 3: SVM Flowchart

In simple terms, SVM (Support Vector Machine) is an algorithm that tries to create a clear boundary, known as a hyperplane, to separate different types of data. In the case of stock market predictions, it separates two types of market conditions: Bullish (positive sentiment) and Bearish (negative sentiment). The model looks at past market data and determines the best hyperplane that divides these two classes, and then it checks where today's data falls. If today's data is on the Bullish side of the hyperplane, the model predicts a Bullish trend; if it's on the Bearish side, it predicts a Bearish trend.

A hyperplane is a flat surface that divides space into two parts. Imagine a line that divides a piece of paper into two parts; that line would be a 1-dimensional hyperplane in a 2D space. In higher dimensions, like 3D space, the hyperplane would be a flat plane that separates the space into two halves. In the context of SVM, the hyperplane helps separate different types of stock market behavior. The SVM algorithm uses this separation to predict whether the market will be Bullish or Bearish, based on how the current data compares to past market conditions.

4.4. Ensemble Approach:

Ensemble modeling is a technique that combines the predictions of several machine learning algorithms to make the final prediction more reliable and accurate. The main idea behind ensemble methods is to use the strengths of different algorithms while minimizing their individual weaknesses. When multiple models work together, their combined predictions tend to be more stable and precise. This is especially helpful in situations where relying on just one model might lead to errors or biased outcomes. For example, if one model performs well in some cases but poorly in others, combining its predictions with other models can lead to a better overall result.

In stock price prediction, ensemble methods are especially valuable because they use predictions from multiple machine learning models, such as decision trees, neural networks, or support vector machines, to make a final forecast. Instead of depending on just one model, ensemble approaches combine the strengths of different models, each of which may capture unique patterns or relationships in the data. This helps improve the overall accuracy and robustness of the predictions. Ensemble methods can also help reduce overfitting, which happens when a model becomes too focused on past data and fails to generalize well to new data. By blending different models, ensemble methods can improve how well predictions apply to real-world stock price movements and provide a more trustworthy forecast.

4.4.1. Simple Averaging

In simple averaging, the predictions from multiple machine learning models are combined by calculating their average. For example, in a stock price prediction project, we might use models like Decision Tree, K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). Each model generates its own forecast for the future stock price. The final prediction is obtained by averaging the predictions from all the models. This helps to smooth out any variations between the models, such as one predicting a higher stock price while another predicts a lower one. The average provides a balanced estimate that is less affected by individual model biases.

The process works by first training each model separately using historical stock data. After training, each model makes its prediction based on the test data. The final prediction is calculated by averaging the individual predictions from all the models. This method helps to reduce the impact of errors or biases that might be present in any single model, providing a more stable and central estimate of the stock price.

One of the main advantages of simple averaging is that it is easy to implement, making it a quick and accessible approach for combining models. It also helps to reduce the biases that could be present in individual models, leading to a more balanced forecast. Additionally, it works well when the models being used have similar performance, as their errors may tend to cancel each other out. However, a key disadvantage is that simple averaging treats each model equally, without considering whether some models are more accurate or reliable than others. This means that if one model is significantly better than the others, averaging all the predictions may reduce the overall performance of the ensemble.

In stock price prediction, simple averaging can still be effective because it combines different models that may capture various patterns in the market. This reduces the risk of making large errors due to the overfitting or biases of individual models. However, if one model consistently outperforms the others, a more advanced ensemble method, such as weighted averaging, could be more beneficial. Weighted averaging gives more importance to the better-performing models, improving the overall accuracy of the prediction.

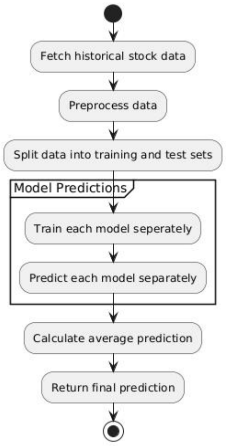
Fetch historical stock data
Preprocess data
Split data into training and test sets
Model Predictions
Train each model seperately
Predict each model separately
Calculate average prediction
Return final prediction

Fig 4: Simple Averaging Flowchart

Simple Average = Prediction_LR + Prediction_DT + Prediction_KNN

First, historical stock data is collected and prepared for analysis. The data is then split into two parts: a training set used to teach the models and a test set used to evaluate their performance. Multiple models, such as Linear Regression, Decision Tree, and K-Nearest Neighbors (KNN), are trained on the training set to learn patterns in the data. Once trained, each model makes its own prediction on the test set. Finally, the predictions from all the models are averaged to produce a single, final prediction for the stock price.

4.4.2. Weighted Averaging

Weighted averaging is an improved version of simple averaging that assigns different importance to each model based on its past performance. Instead of treating all models equally, this method gives more weight to models that have performed better in previous predictions. This approach helps improve the accuracy of the final prediction by leveraging the strengths of the most reliable models while still considering input from other models.

In practice, the process starts with fetching and preprocessing the historical stock data, just like in simple averaging. The data is then split into training and test sets. The Linear Regression, Decision Tree, and KNN models are trained on the training set. Once the models are trained, each generates its own prediction for the stock price based on the test set. However, in weighted averaging, each model's prediction is given a weight based on how well it has performed in the past. For example, if the Decision Tree has consistently made accurate predictions, it will be assigned a higher weight. The final prediction is then calculated by taking a weighted average of the models' predictions, where better-performing models have a greater influence on the result.

The main advantage of weighted averaging is that it prioritizes models that have proven to be more accurate, leading to a more reliable prediction. It also offers flexibility, as it can adjust to changes in the models' performances over time. However, one of the challenges is deciding how to choose the weights for each model. If one model is given too much weight, it might overshadow the others, and the ensemble could lose the diversity that helps improve predictions. Therefore, careful consideration is needed to balance the weights appropriately and ensure the ensemble remains effective.

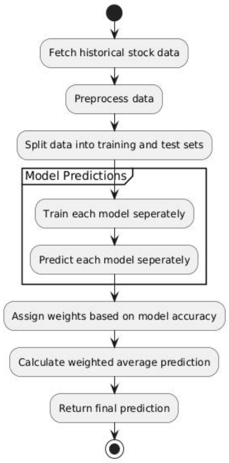
Fetch historical stock data
Preprocess data
Split data into training and test sets
Model Predictions
Train each model seperately
Predict each model seperately
Assign weights based on model accuracy
Calculate weighted average prediction
Return final prediction

Fig 5: Weighted Averaging Flowchart

Weighted Average = w1*Prediction_LR + w2*Prediction_DT + w3*Prediction_KNN
w1, w2, w3 - weights

First, historical stock data is collected and prepared for analysis. The data is then divided into two sets: a training set to teach the models and a test set to evaluate their performance. The Linear Regression, Decision Tree, and KNN models are trained using the training data. After training, each model makes its own prediction for the stock price based on the test set. The predictions are then assigned weights according to how well each model has performed in the past. Finally, the weighted predictions from all the models are combined to produce a final prediction, with more reliable models having a greater influence on the result.

4.4.3. Gradient Boosting

Gradient Boosting is an advanced ensemble method that builds a group of models one after another, where each new model works to fix the mistakes made by the previous one. In the context of stock price prediction, it starts with a basic model that may not be very accurate. Then, the next model is trained to correct the errors in the predictions made by the first model. This process is repeated with more models, each one improving the prediction by learning from the mistakes of the previous models. Eventually, the final prediction is a combination of all the models, leading to a much more accurate result.

The way it works is that the first model makes predictions, but these predictions have some errors. A second model is trained to predict these errors, and it corrects them when making its own predictions. This process continues, with each new model focusing on the mistakes of the previous ones. Over time, this approach helps the ensemble to become more accurate. However, the downside is that it can be slow and computationally expensive because the models are trained one after the other. Additionally, Gradient Boosting can be prone to overfitting, meaning it may work well on the training data but fail to generalize to new data if not tuned carefully.

The advantage of Gradient Boosting is that it has a high potential for accuracy, particularly when dealing with complex data, like stock market prices, which can have many variables and non-linear relationships. It's well-suited for capturing the volatility and patterns in the market. However, due to its complexity and the fact that it builds models sequentially, it can take more time and computing resources compared to other methods. Additionally, because it keeps adjusting models based on previous errors, if it's not properly tuned, it may end up being too sensitive to the training data, causing overfitting.

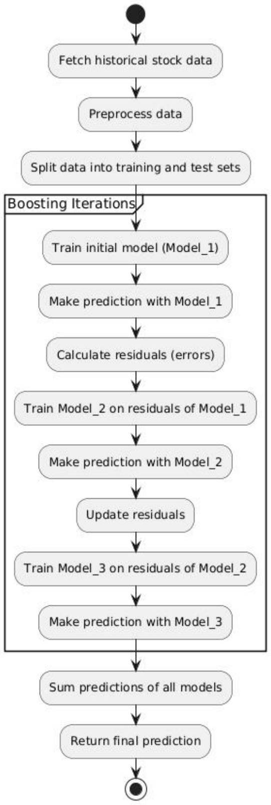
Fetch historical stock data
Preprocess data
Split data into training and test sets
Boosting Iterations
Train initial model (Model_1)
Make prediction with Model_1
Calculate residuals (errors)
Train Model_2 on residuals of Model_1
Make prediction with Model_2
Update residuals
Train Model_3 on residuals of Model_2
Make prediction with Model_3
Sum predictions of all models
Return final prediction

Fig 6: Gradient Boosting Flowchart

In Gradient Boosting, the process begins by fetching, preprocessing, and splitting the stock data into training and test sets. First, a base model (Model_1) is trained using the training data to make an initial prediction. However, this model usually makes some errors in its predictions. To improve the accuracy, the residual errors (the difference between the model's predictions and the actual values) are calculated. These residual errors indicate where the model went wrong.

Next, a second model (Model_2) is trained specifically to correct the errors made by Model_1. Instead of training from scratch, Model_2 focuses on predicting the residuals and adjusts the model's predictions accordingly. This process continues with additional models, each one correcting the mistakes of the previous models. Each new model works to improve the overall prediction, refining it step by step. The final prediction is then made by combining the results from all the models, resulting in a highly accurate and fine-tuned prediction.

In simple terms, Gradient Boosting works by building models one after another, with each new model helping to fix the errors of the one before it. This iterative approach leads to improved predictions over time, making it especially effective for complex problems like stock price prediction, where patterns can be hard to capture.

5. EXPERIMENTAL SETUP

The project workflow, as shown in Fig. 7, starts with the preprocessing of stock price data. This includes steps like feature engineering, where technical indicators such as RSI, MACD, Bollinger Bands, ATR, and moving averages are calculated to capture essential market trends. Furthermore, data cleaning is performed to remove any inconsistencies or missing values. The cleaned dataset is then split into an 80/20 ratio, where 80% of the data is used for training the models, and 20% is reserved for testing purposes. This split ensures a robust evaluation framework, allowing the models to generalize well on unseen data.

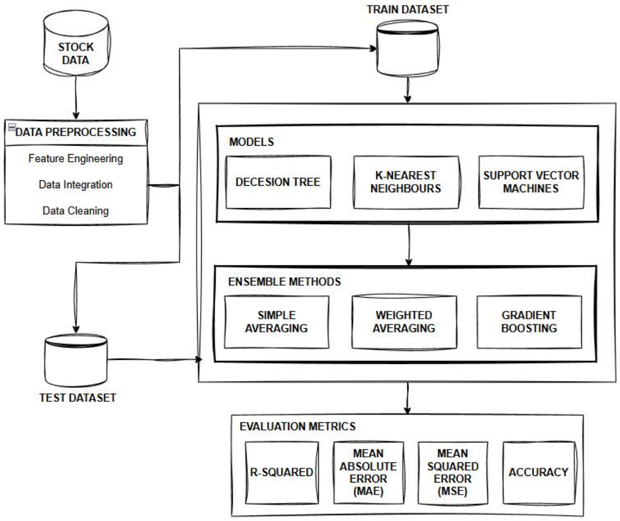

Fig 7: Workflow

During the training phase, the 80% training dataset is used to train three core machine learning models: Decision Tree (DT), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). Each of these models brings unique capabilities to the table. The Decision Tree identifies patterns based on feature importance and hierarchical splits, KNN classifies and predicts based on nearest data points, and SVM draws a hyperplane to separate stock price classes efficiently. These diverse approaches ensure that the models can capture various aspects of stock price fluctuations.

The ensemble methods—simple averaging, weighted averaging, and gradient boosting—are then applied to combine predictions from these individual models. The simple averaging method takes an unweighted mean of the model outputs, while weighted averaging assigns importance to each model based on its performance. Gradient boosting further refines predictions by iteratively minimizing the error in ensemble results. These methods significantly improve the predictive accuracy, making the model more robust. Finally, the trained ensemble model is tested on the remaining 20% of the data. Metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), R-squared (R²), and accuracy are evaluated to confirm the effectiveness of the approach.

ACRONYM	FULL NAME	FORMULA
MAE	Mean Absolute Error	$\frac{1}{n}\sum_{i=0}^{n} t_i - p_i$
MSE	Mean Squared Error	$\frac{1}{n}\sum (t_i - p_i)^2$
R^2	R-squared	$1 - \frac{\sum_{i=1}^{n}(t_i - p_i)^2}{\sum_{i=1}^{n}(t_i - m_i)^2}$
ACC	Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$

Table 2: Description of Evaluation Metrics and Their Calculation Methods

The results demonstrate that the ensemble approach enhances the overall predictive performance compared to individual models, leveraging the combined strengths of each algorithm. This methodology, as visualized in Fig. 7, provides investors and traders with a reliable tool to navigate the volatile stock market with confidence.

6. REFERENCES

[1] Tan Z, Yan Z, Zhu G. Stock selection with random forest: an exploitation of excess return in the Chinese stock market. Heliyon. 2019;5:e02310. https://doi.org/10.1016/j.heliyon.2019.e02310.

[2] Krauss C, Do XA, Huck N. Deep neural networks, gradient-boosted trees, random forests: statistical arbitrage on the S&P 500. Eur J Oper Res. 2017;259:689–702. https://doi.org/10.1016/j.ejor.2016.10.031.

[3] Yadav S, Sharma N. Homogenous ensemble of time-series models for indian stock market. Springer. 2018. https://doi.org/10.1007/978-3-030-04780-1_7.

[4] Mehta S, Rana P, Singh S, Sharma A, Agarwal P. Ensemble learning approach for enhanced stock prediction. In: 2019 12th international conference on contemporary computing IC3 2019. 2019, pp. 15. https://doi.org/10.1109/ic3.2019.8844891.

[5] Ballings M, Van den Poel D, Hespeels N, Gryp R. Evaluating multiple classifiers for stock price direction prediction. Expert Syst Appl. 2015;42:7046–56. https://doi.org/10.1016/j.eswa.2015.05.013.

[6] Nti IK, Adekoya AF, Weyori BA. A systematic review of fundamental and technical analysis of stock market predictions. Artif Intell Rev. 2019. https://doi.org/10.1007/s10462-019-09754-z.

[7] Macchiarulo A. Predicting and beating the stock market with machine learning and technical analysis. J Intern Bank Commer. 2018;23:1–22.

[8] Jacobsen B, Jiang F, Zhang H. Ensemble machine learning and stock return predictability. SSRN Electron J. https://doi.org/10.2139/ssrn.3310289.

[9] Pimprikar R, Ramachadran S, Senthilkumar K. Use of machine learning algorithms and twitter sentiment analysis for stock market prediction. Int J Pure Appl Math. 2017;115:521–6. https://www.mdpi.com/2079-9292/11/20/3414

[10] Nti, I.K., Adekoya, A.F. & Weyori, B.A. A comprehensive evaluation of ensemble learning for stock-market prediction. *J Big Data* 7, 20 (2020). https://doi.org/10.1186/s40537-020-00299-5

[11] Agrawal, L., & Adane, D. (2021). Improved Decision Tree Model for Prediction in Equity Market Using Heterogeneous Data. *IETE Journal of Research*, *69*(9), 6065–6074. https://doi.org/10.1080/03772063.2021.1982415

[12] Nti, I., Adekoya, A. & Weyori, B. (2020). Efficient Stock-Market Prediction Using Ensemble Support Vector Machine. *Open Computer Science*, *10*(1), 153-163. https://doi.org/10.1515/comp-2020-0199

[13] Singh, G. (2022). Machine Learning Models in Stock Market Prediction. arXiv. https://doi.org/10.48550/ARXIV.2202.09359

[14] Z. K. Lawal, H. Yassin and R. Y. Zakari, "Stock Market Prediction using Supervised Machine Learning Techniques: An Overview," *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, Gold Coast, Australia, 2020, pp. 1-6, doi: 10.1109/CSDE50874.2020.9411609.

[15] V. Gaur, S. Sood, L. Uppal and M. Kaur, "Revitalizing Stock Predictions with Machine Learning Algorithms – An Empirical Study," *2020 IEEE 17th India Council International Conference (INDICON)*, New Delhi, India, 2020, pp. 1-6, doi: 10.1109/INDICON49873.2020.9342571.