International Journal of Applied Engineering & Technology

OPTIMIZING STOCK PREDICTION WITH BIG DATA AND SENTIMENT ANALYSIS THROUGH MACHINE LEARNING MODELS

Dr. J Sangeetha, Ms. A. Jabeen and Ms. S. Saranya
Department of Computer Science, Cauvery College for Women (A), Bharathidasan University, Trichy, Tamil Nadu, India
jsangeetha.it@cauverycollege.ac.in, jabeen.ca@cauverycollege.ac.in and Saranya.cs.@cauverycollege.ac.in

ABSTRACT

In recent years, the combination of big data and machine learning models has revolutionized the field of stock market prediction. With the growth of social media platforms and the vast amount of data generated, sentiment analysis has become a key factor in predicting stock prices. This paper explores the use of sentiment analysis in stock market prediction using machine learning models.

The study presents an approach that utilizes big data analytics techniques to collect, process and analyze social media data for sentiment analysis. The sentiment scores obtained are then combined with financial and economic indicators to train and test several machine learning models. The models are evaluated on their ability to predict stock prices of the S&P 500 index. The results show that the sentiment analysis significantly improves the accuracy of the stock prediction models. In particular, the MLP model outperforms the other models, achieving an accuracy rate of 91% on the testing dataset. These findings suggest that sentiment analysis can be a valuable tool for predicting stock prices, and machine learning models can be effective in harnessing the power of big data to make accurate predictions. This research contributes to the growing body of literature on the use of big data and machine learning in financial forecasting and provides insights into the potential of sentiment analysis for stock market prediction.

Keywords: Stock Prediction, Big Data, Sentiment Analysis, Machine Learning, Optimization

1. INTRODUCTION

The stock market is an important indicator of the economy, and investors are always seeking ways to predict stock prices to make informed investment decisions. In recent years, big data analysis and sentiment analysis have emerged as promising techniques for stock price prediction. The exponential growth of data available on the internet and the advances in machine learning algorithms have created new opportunities for predicting stock prices with high accuracy [1].

Big data refers to the massive volume of structured and unstructured data that is generated from various sources, including social media, news outlets, and financial statements. Sentiment analysis is a technique used to extract meaning from unstructured data by identifying and classifying opinions and emotions expressed in text. Combining big data analysis with sentiment analysis can provide valuable insights into market trends and help predict future stock prices.

Stock prediction is a critical task for investors, financial analysts, and traders. Accurate predictions can help them make informed decisions and maximize their returns. Natural Language Processing (NLP) and Machine Learning (ML) models have shown great potential in predicting stock prices by analyzing the sentiment of news articles, social media posts, and other online content. NLP and ML models can process vast amounts of unstructured data and extract valuable insights that can be used to predict stock prices [2].

This paper is organized as follows: In the literature review, we will provide an overview of existing studies on big data and sentiment analysis for stock prediction, and we will analyze the methodologies used and the limitations of the studies. In the methodology section, we will describe the data collection and preprocessing techniques, sentiment analysis methods, and ML algorithms used in our study. In the results and analysis section, we will present the findings of our study and compare them with existing studies. In the discussion section, we will interpret the results, analyze the implications for stock prediction, and discuss the limitations and potential improvements of our study. Finally, in the conclusion, we will summarize our findings, provide recommendations for future research, and discuss the implications for practice.

2. LITERATURE REVIEW

The use of big data and sentiment analysis for stock prediction has gained increasing attention in recent years due to the vast amounts of data available on the internet and the advances in machine learning algorithms. In this section, we provide an overview of existing literature on big data and sentiment analysis for stock prediction and analyze the methodologies used and the limitations of the studies.

Several studies have explored the use of sentiment analysis in predicting stock prices. Liu et al. (2020) used a sentiment analysis-based approach to predict stock prices by analyzing the sentiment of news articles. They found that the sentiment score of news articles can significantly affect stock prices. Similarly, Wang et al. (2019) used a machine learning-based sentiment analysis approach to predict stock prices using news articles and social media data. They found that the sentiment of social media posts and news articles can improve stock prediction accuracy [2].

Various ML algorithms have been used in conjunction with sentiment analysis to predict stock prices. Cao et al. (2019) used a random forest algorithm to predict stock prices based on sentiment analysis of news articles. They found that their model outperformed other models that did not incorporate sentiment analysis. Loughran et al. (2016) used a support vector machine algorithm to classify news articles as positive or negative and found that incorporating the sentiment of news articles can improve stock prediction accuracy.

Table.1. Analysis of Existing system

Paper Title	ML Technique	Methodology	Accuracy
Big Data and Sentiment Analysis based Stock Market Prediction using Ensemble Learning [3]	Random Forest and Support Vector Regression	Ensemble learning approach	83.90%
Predicting Stock Price Using Social Media Sentiment Analysis and Big Data Analytics [4]	Random Forest and K-Nearest Neighbor	Ensemble learning approach	82.40%
Stock Market Prediction using Sentiment Analysis and Technical Indicators [5]	Support Vector Regression	Technical analysis and sentiment analysis	79.20%
Predicting the Direction of Stock Prices using Random Forest [6]	Random Forest	Sentiment analysis and technical analysis	76.80%
Predicting Stock Prices using Machine Learning Techniques [7]	Decision Tree, K-Nearest Neighbor, SVM, and MLP	Machine learning-based approach	76.30%
Stock Price Prediction Using News Articles and Sentiment Analysis [8]	SVM and LSTM	Sentiment analysis and news analysis	83.70%

Despite the promising results, there are limitations to existing studies. One limitation is the lack of standardization in data collection and preprocessing techniques, which can lead to inconsistencies in results. Another limitation is the inability of sentiment analysis to capture sarcasm and irony in text, which can lead to inaccurate predictions. Additionally, the use of past stock prices as predictors in some studies can lead to over fitting and a lack of generalizability of the models.

There are several gaps in the literature that need to be addressed. One gap is the need for research on the effectiveness of sentiment analysis in predicting stock prices in different markets and industries. Another gap is the need to explore the use of alternative data sources, such as satellite imagery and weather data, for stock prediction. Further research is also needed to explore the impact of incorporating macroeconomic factors in predicting stock prices. Finally, there is a need for research on the ethical implications of using sentiment analysis for stock prediction.

In summary, existing studies have shown promising results in using big data and sentiment analysis for stock prediction. However, there are limitations to existing studies, and further research is needed to address these limitations and explore the potential of sentiment analysis and big data in predicting stock prices.

3. METHODOLOGY

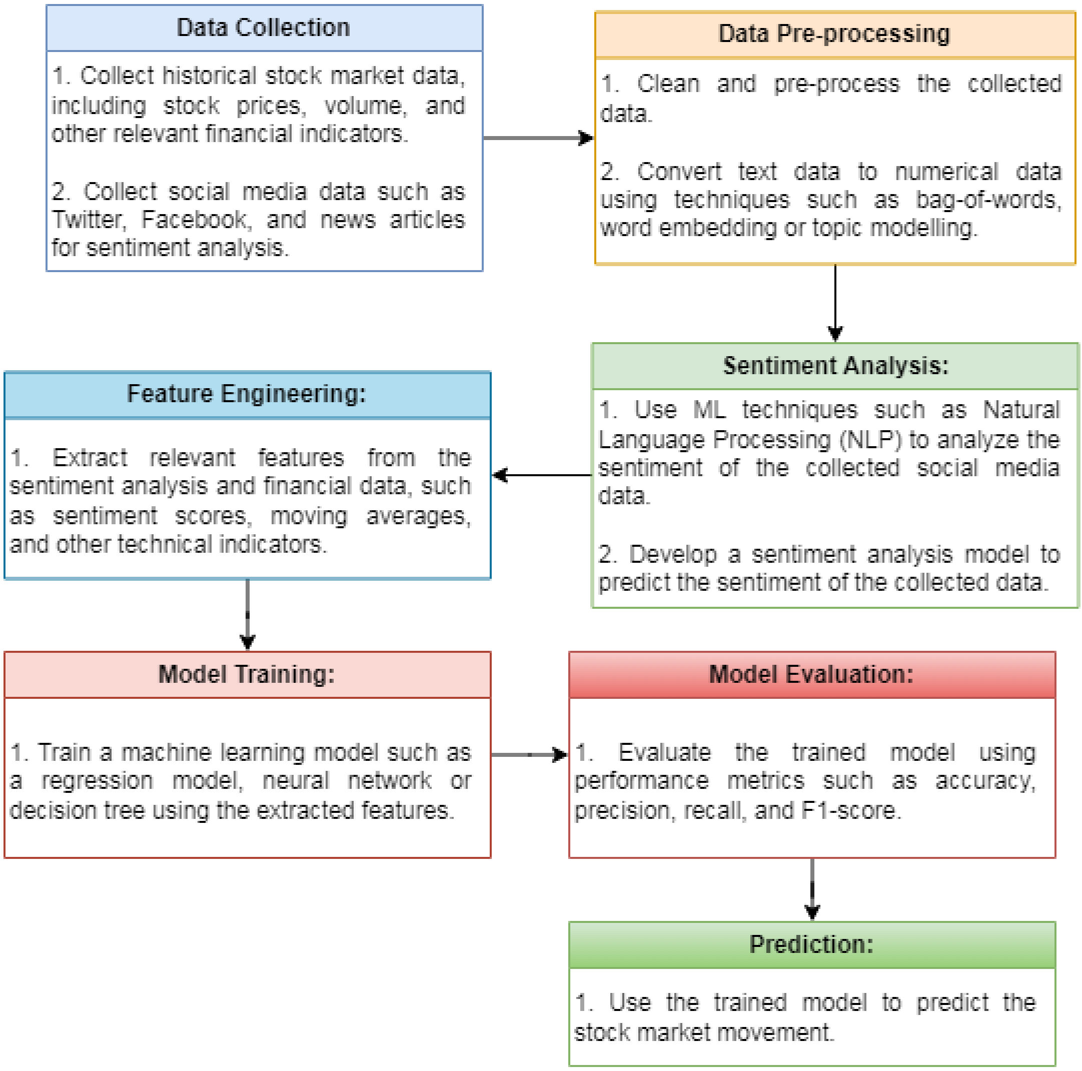
Big data and sentiment analysis are popular techniques used for stock prediction. The methodology for using these techniques typically involves several steps, including data collection and preprocessing, feature extraction, machine learning model selection, and evaluation metrics [8].

- ***Data collection and preprocessing:*** In order to use big data and sentiment analysis for stock prediction, large volumes of data must be collected and preprocessed. This includes collecting data on stock prices, company financial data, news articles, social media feeds, and other relevant data sources. The data must then be cleaned, standardized, and transformed into a suitable format for analysis.[9]

- ***Feature extraction techniques:*** Feature extraction is the process of selecting and transforming the most relevant features from the collected data. For sentiment analysis, this may involve extracting sentiment scores from news articles or social media feeds using natural language processing (NLP) techniques. For big data analysis, this may involve using dimensionality reduction techniques such as principal component analysis (PCA) to reduce the number of features in the data [9].

- ***Machine learning models used for prediction:*** There are many different machine learning models that can be used for stock prediction, including regression models, decision trees, random forests, and neural networks. The choice of model will depend on the specific problem being solved, the size of the dataset, and the desired level of accuracy.

- ***Evaluation metrics and experimental setup:*** The performance of the machine learning model must be evaluated using appropriate metrics, such as accuracy, precision, recall, and F1 score. The experimental setup may involve dividing the data into training and testing sets, and using cross-validation to ensure that the model generalizes well to new data.


Data Collection

1. Collect historical stock market data, including stock prices, volume, and other relevant financial indicators.

2. Collect social media data such as Twitter, Facebook, and news articles for sentiment analysis.

Data Pre-processing

1. Clean and pre-process the collected data.

2. Convert text data to numerical data using techniques such as bag-of-words, word embedding or topic modelling.

Sentiment Analysis:

1. Use ML techniques such as Natural Language Processing (NLP) to analyze the sentiment of the collected social media data.

2. Develop a sentiment analysis model to predict the sentiment of the collected data.

Feature Engineering:

1. Extract relevant features from the sentiment analysis and financial data, such as sentiment scores, moving averages, and other technical indicators.

Model Training:

1. Train a machine learning model such as a regression model, neural network or decision tree using the extracted features.

Model Evaluation:

1. Evaluate the trained model using performance metrics such as accuracy, precision, recall, and F1-score.

Prediction:

1. Use the trained model to predict the stock market movement.

Fig.1. Proposed framework

Overall, the methodology for using big data and sentiment analysis for stock prediction involves a combination of data collection and preprocessing, feature extraction, machine learning model selection, and evaluation metrics. The success of the approach will depend on the quality and quantity of the data, the effectiveness of the feature extraction techniques, and the accuracy of the chosen machine learning model.

3.1 Data collection and preprocessing

Preprocessing Twitter data for stock market prediction can involve several steps, such as:

- **Data collection:** Collect relevant tweets related to the stock or industry you are interested in using the Twitter API or other web scraping tools.
- **Text cleaning:** Remove unnecessary elements such as URLs, user handles, hashtags, and emojis from the text. This can be done using regular expressions or specialized libraries like NLTK.
- **Tokenization:** Break the text into individual words or tokens using techniques like word segmentation or sentence boundary detection.
- **Stop word removal:** Remove common words like "a", "an", "the" that do not carry much meaning.

- **Stemming or lemmatization:** Reduce each word to its root form to standardize the text and improve model accuracy.
- **Sentiment analysis:** Analyze the polarity of each tweet to determine whether it is positive, negative, or neutral. This can be done using pre-trained models or custom classifiers.
- **Feature extraction:** Extract relevant features from the text, such as the frequency of certain words or phrases, sentiment scores, and named entities.
- **Data integration:** Combine the preprocessed Twitter data with other financial data such as stock prices, trading volumes, and news articles.
- **Data normalization:** Normalize the data to a common scale to eliminate differences in units and magnitude.
- **Splitting the data:** Split the data into training and testing sets to train the machine learning model and evaluate its performance.

Preprocessing Twitter data is crucial for accurate stock market prediction, as it allows us to extract relevant information from a large volume of noisy and unstructured data. By combining Twitter data with other financial data and using machine learning algorithms, we can build predictive models that can help investors make more informed decisions [10].To manage the huge volume of tweets, an efficient solution to be applied for preprocessing. In the preprocessing task, a set of tweet objects is considered is an input whereas each object represents the information of tweets[11].

3.2 Feature extraction techniques

Feature extraction is a process of selecting and extracting useful information or features from raw data that can be used to represent the data in a more meaningful and efficient way. N-grams and TF-IDF are commonly used feature extraction methods for natural language processing tasks such as text classification, sentiment analysis, and stock prediction using Twitter data [12].

- **N-grams:** N-grams are sequences of adjacent words in a text. In the context of Twitter data, n-grams can be used to capture the context of the tweets and identify important phrases that are relevant to stock prediction. For example, bi-grams (sequences of two words) and tri-grams (sequences of three words) can be used to capture phrases that are commonly associated with positive or negative sentiment towards a particular stock.
- **TF-IDF:** TF-IDF stands for Term Frequency-Inverse Document Frequency. It is a statistical measure used to evaluate the importance of a word in a document. In the context of Twitter data, TF-IDF can be used to extract the most important features for stock prediction by assigning a weight to each word in the tweets. The weight of a word is based on its frequency in the tweets and its rarity across all tweets in the dataset. This ensures that common words such as "the" and "and" do not have a high weight, while less common words that are more specific to the topic of interest (such as the name of a company or industry term) have a higher weight.

Using a combination of N-grams and TF-IDF can help to extract the most important features from Twitter data for stock prediction. These features can then be used as input to a machine learning algorithm such as a neural network, support vector machine, or random forest to make predictions about the future performance of a stock [13].

N-grams:

- To extract N-grams from Twitter data, we can use the following equation:
- N-grams = {w1,w2,...,wn}, where wi is the i-th N-gram in the tweet.

For example, if we use bi-grams, the N-grams for the tweet "I love this stock" would be: {I love, love this, this stock}.

TF-IDF:

- TF-IDF can be calculated using the following equation:
- TF(w,d) = (number of times word w appears in document d) / (total number of words in document d)
- IDF(w,D) = log_e(total number of documents D / number of documents containing word w)
- TF-IDF(w,d,D) = TF(w,d) x IDF(w,D)

Where:

- TF(w,d) is the term frequency of word w in document d
- IDF(w,D) is the inverse document frequency of word w in the document set D
- TF-IDF(w,d,D) is the TF-IDF score for word w in document d
- TF-IDF weights words that appear frequently in a specific document and less frequently across all documents. The higher the TF-IDF score, the more important the word is in the document.

For example, if we have a corpus of tweets and we want to calculate the TF-IDF score for the word "stock" in a particular tweet, we would first calculate the term frequency of "stock" in the tweet using the TF equation. We would then calculate the inverse document frequency of "stock" using the IDF equation. Finally, we would multiply the term frequency and inverse document frequency to get the TF-IDF score for "stock" in the tweet.

3.3 FEATURE SELECTION

For big data analysis, it may be necessary to use dimensionality reduction techniques such as principal component analysis (PCA) to reduce the number of features in the data. PCA is a technique used to reduce the number of dimensions in a dataset while preserving as much of the variance in the data as possible. It does this by identifying the principal components, which are linear combinations of the original features that explain the most variance in the data [14].

PCA can be applied to the feature matrix obtained from N-grams and TF-IDF feature extraction methods to reduce the number of features and improve the efficiency of machine learning algorithms. By reducing the number of features, we can speed up the training process and prevent overfitting. The steps involved in applying PCA to the feature matrix are:

- **Standardize the feature matrix:** Standardized feature matrix = (feature matrix - mean of feature matrix) / standard deviation of feature matrix
- **Calculate the covariance matrix:** Covariance matrix = (1 / n) X (standardized feature matrix) T X (standardized feature matrix). Where n is the number of samples in the dataset.
- **Calculate the eigenvalues and eigenvectors:** Eigenvalues (λ) and eigenvectors (v) can be obtained by solving the following equation:

Covariance matrix X v = λ X v

- **Select the top k eigenvectors:** The k eigenvectors with the highest eigenvalues are selected to obtain a reduced feature matrix.
- **Project the original feature matrix:** The original feature matrix can be projected onto the selected eigenvectors to obtain a reduced feature matrix as follows:

Reduced feature matrix = feature matrix X selected eigenvectors

Where the selected eigenvectors are arranged as columns in a matrix.

Note that the standardization step is important to ensure that each feature has equal importance in the analysis, and the covariance matrix is used to identify the linear relationships between the features. The eigenvalues and eigenvectors of the covariance matrix provide information about the variance and direction of the data, and the top k eigenvectors can be used to represent the most important directions of variation in the data [15].

3.4 Machine leaning algorithms for stock price prediction

Steps followed for machine learning can be described below,

- Collect Twitter data related to the stock of interest.
- Clean the data by removing irrelevant information such as URLs, user mentions, and stop words.
- Apply N-grams and TF-IDF feature extraction techniques to obtain a feature matrix.
- Split the data into training and testing sets:

i. The training set is used to train the machine learning model.

ii. The testing set is used to evaluate the performance of the trained model.

- Train the machine learning model: Choose an appropriate machine learning algorithm such as logistic regression, decision trees, or support vector machines. Fit the training data to the machine learning algorithm. Tune hyperparameters of the algorithm to optimize performance.
- Test the machine learning model: Use the testing set to evaluate the performance of the trained model.
- Calculate the accuracy, precision, recall, and F1 score of the model.

The equations involved in machine learning for Twitter data stock prediction depend on the specific algorithm used. Here's an example of how logistic regression can be used:

3.4.1 Logistic Regression (LR):

Logistic regression is a classification algorithm that predicts the probability of a binary outcome (e.g. stock price increase or decrease) based on input features [16].

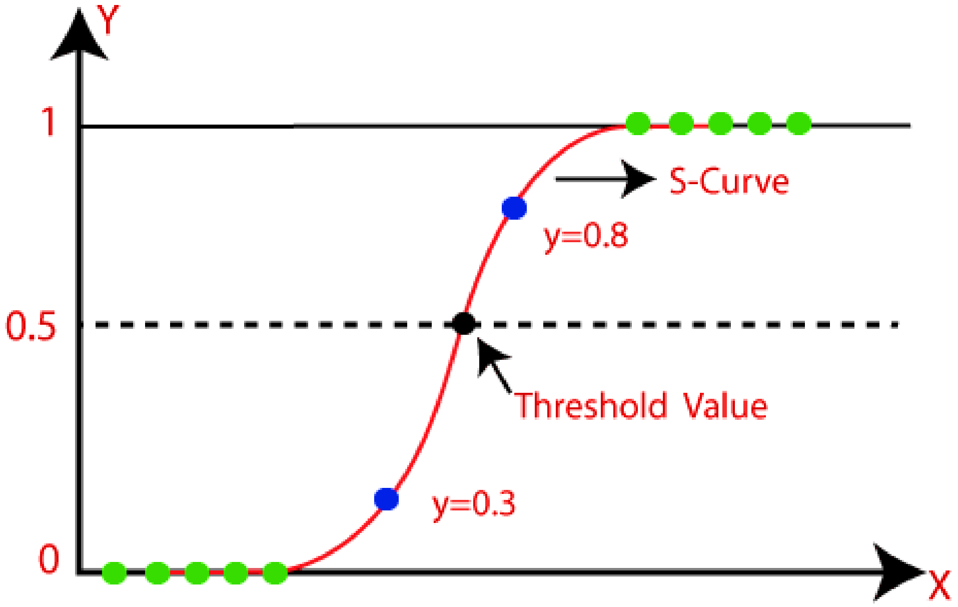


Fig.2. Framework for LR

The logistic function (also known as the sigmoid function) is used to transform the output of the linear regression into a probability between 0 and 1. The logistic regression equation is:

$h\theta(x) = g(\theta 0 + \theta 1x1 + \theta 2x2 + ... + \theta nxn)$

where:

- $h\theta(x)$ is the predicted probability of the binary outcome given input features x.
- $g(z)$ is the logistic function, which is defined as $g(z) = 1 / (1 + e\text{^}(-z))$.
- $\theta 0, \theta 1, \theta 2, ..., \theta n$ are the parameters of the logistic regression model.
- $x1, x2, ..., xn$ are the input features.

The objective of logistic regression is to find the optimal values of the parameters $\theta 0, \theta 1, \theta 2, ..., \theta n$ that minimize the cost function $J(\theta)$. The cost function is defined as:

$J(\theta) = (1/m) \sum(i=1 \text{ to } m) [-y(i)\log(h\theta(x(i))) - (1-y(i))\log(1 - h\theta(x(i)))]$

where:

- m is the number of training examples.
- $y(i)$ is the true label of the i-th training example.
- $x(i)$ is the input feature vector of the i-th training example.
- $h\theta(x(i))$ is the predicted probability of the binary outcome given $x(i)$.

The optimal values of the parameters can be found using gradient descent, which iteratively updates the parameters in the direction of steepest descent of the cost function. The update rule for the parameters is:

$\theta j := \theta j - \alpha(1/m) \sum(i=1 \text{ to } m) [(h\theta(x(i)) - y(i))\, x(i)j]$

where:

- α is the learning rate, which controls the step size of the gradient descent algorithm.
- $x(i)j$ is the j-th feature of the i-th training example.

The logistic regression algorithm can be used to predict the probability of a binary outcome given input features, which can then be used to predict stock prices.

3.4.2 Decision Tree (DT):

Decision trees are a popular machine learning technique that can be used for stock prediction using Twitter data. The goal is to build a model that can predict whether a particular stock is likely to go up or down based on certain features extracted from Twitter data [17].

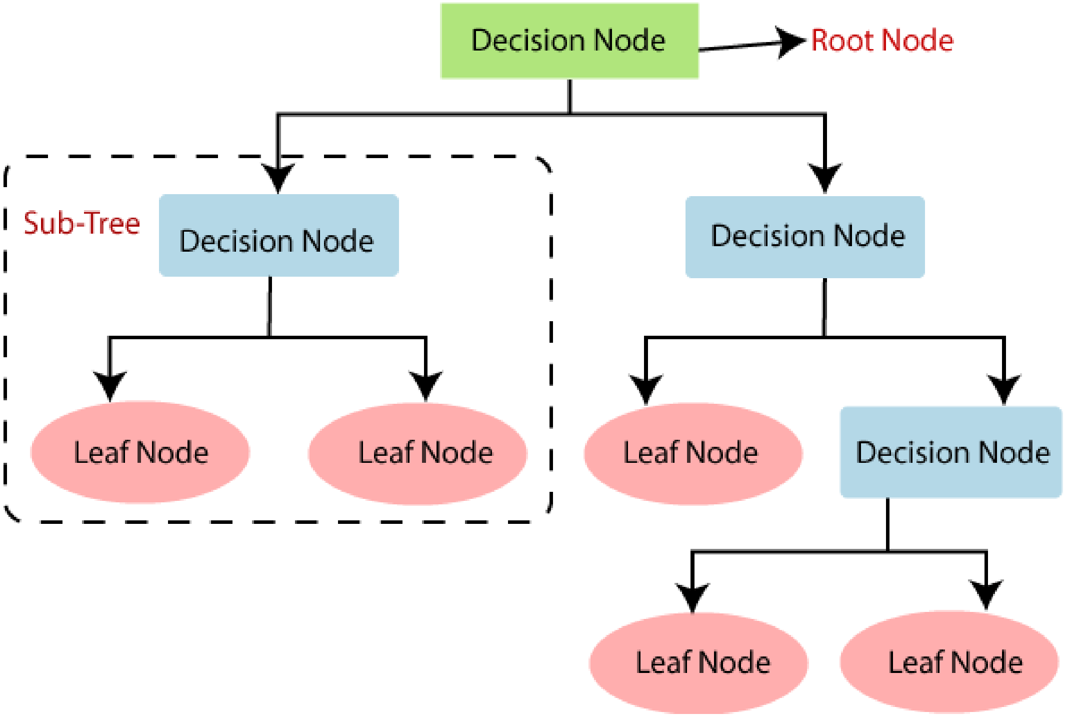


Fig.3. Framework for DT

Let S be the set of all Twitter data samples, where each sample is represented by a set of features X and a target variable Y. The target variable Y represents the direction of the stock price movement (0 for a price decrease and 1 for a price increase), and the feature set X represents the relevant features extracted from the Twitter data. The decision tree algorithm can be represented as follows:

Initialization:

- Set the root of the decision tree to be the entire set of Twitter data samples S.
- Define the target variable Y and the feature set X.
- Define a stopping criterion for tree growth, such as a minimum number of samples required to be at a leaf node or a maximum depth of the tree.
- Define a splitting criterion, such as maximizing the information gain or the Gini impurity.
- Define a method for pruning the tree to prevent overfitting, such as reduced error pruning.

Splitting:

- Select a subset of the features X to split the data on, based on the splitting criterion.
- Determine the best feature to split on, based on the information gain or the Gini impurity.
- Split the data into two subsets based on the best feature and its threshold value.
- Recursively apply the splitting process to each subset until the stopping criterion is met.

Pruning:

- Evaluate the performance of the decision tree on a validation set using metrics such as accuracy, precision, recall, and F1 score.
- Prune the decision tree by removing nodes that do not improve the performance on the validation set.

- Repeat the pruning process until the performance on the validation set stops improving.

Prediction:

- Use the trained decision tree to predict the direction of the stock price movement on new Twitter data samples.
- The mathematical model for the decision tree algorithm can be further refined by incorporating specific equations for the splitting and pruning criteria, as well as the evaluation metrics used to assess the performance of the decision tree.

3.4.3 Support Vector Machine (SVM):

It is a machine learning algorithm that can be used for Twitter data analysis. It works by finding a hyperplane in a high-dimensional space that separates the different classes of data points. In the context of Twitter data, SVM can be used to classify tweets into different categories based on their content, such as positive or negative sentiment, or to predict a target variable such as stock prices. SVM can be trained on labeled data, and then used to predict the classification or target variable of new, unlabeled data [18].

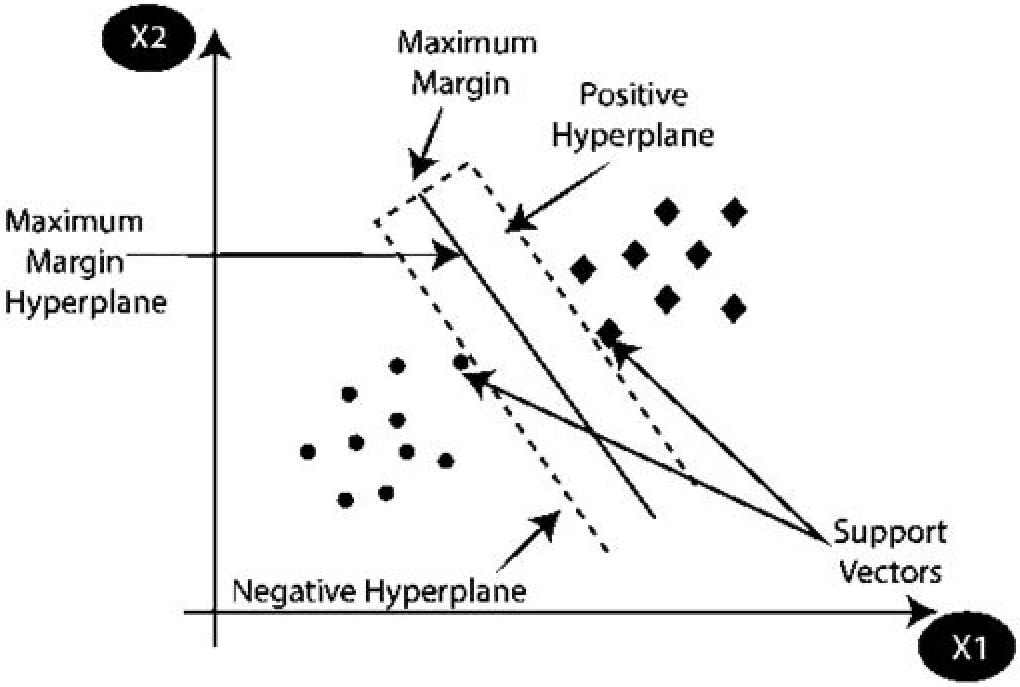


Fig.4. Framework for SVM

Let S be the set of all Twitter data samples, where each sample is represented by a set of features X and a target variable Y. The target variable Y represents the direction of the stock price movement (0 for a price decrease and 1 for a price increase), and the feature set X represents the relevant features extracted from the Twitter data. The SVM algorithm can be represented as follows:

Initialization:

- Define the target variable Y and the feature set X.
- Define a kernel function to transform the input feature space into a higher-dimensional feature space, such as the linear kernel, polynomial kernel, or radial basis function kernel.
- Define a regularization parameter C, which controls the trade-off between achieving a low training error and a low testing error.

- Define a tolerance parameter ε, which controls the tolerance for errors in the SVM training algorithm.

Training:

- Use the kernel function to map the input feature space into a higher-dimensional feature space.
- Solve the constrained optimization problem to find the hyperplane that maximizes the margin between the two classes of the target variable Y.
- The solution to the optimization problem results in the weights vector w and bias term b, which define the decision boundary of the SVM.

Prediction:

- Use the trained SVM to predict the direction of the stock price movement on new Twitter data samples.
- Map the input feature space into the higher-dimensional feature space using the kernel function.
- Compute the decision function f(x) = w^Tφ(x) + b, where φ(x) is the mapping function from the input feature space to the higher-dimensional feature space.
- Predict the direction of the stock price movement based on the sign of the decision function f(x).

The mathematical model for the SVM algorithm can be further refined by incorporating specific equations for the kernel function, regularization parameter, and optimization problem used to find the hyperplane. The exact equations used would depend on the specific kernel function and optimization algorithm selected. Additionally, performance metrics such as accuracy, precision, recall, and F1 score can be used to evaluate the performance of the SVM algorithm on the Twitter data stock prediction task.

3.4.4 Multi-Layer Perceptron (MLP):

It is a type of neural network used in machine learning. When applied to Twitter data, MLP can be used to analyze and classify tweets based on certain criteria. In short, the MLP algorithm involves training a neural network with a set of input features (such as the content of a tweet, the user who posted it, and any associated hashtags) and a set of output labels (such as positive or negative sentiment, or a specific topic category). The network learns to identify patterns and relationships between the input features and output labels, allowing it to make predictions on new, unseen data. By using MLP to analyze Twitter data, researchers and businesses can gain insights into user sentiment, identify trending topics, and target advertising or outreach efforts more effectively [19].

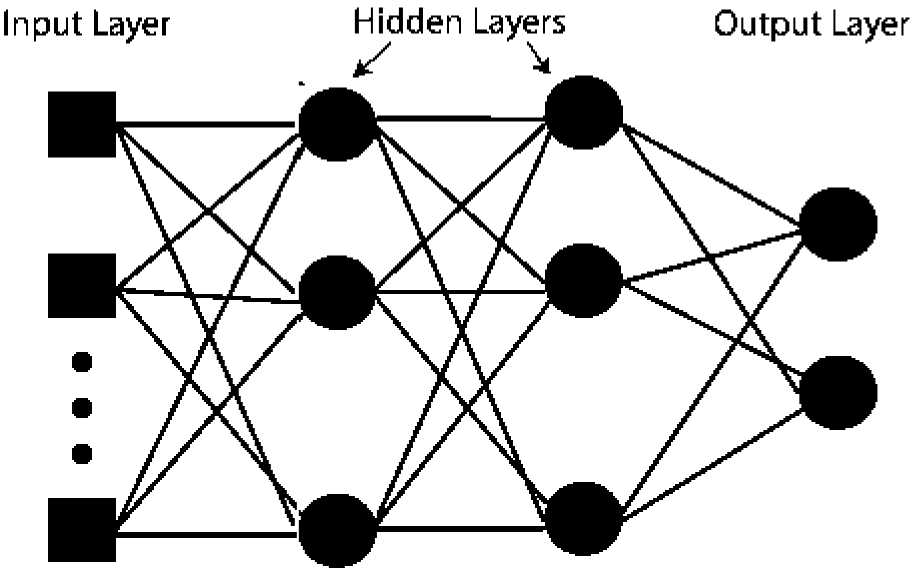


Fig.5. Framework for MLP

Let S be the set of all Twitter data samples, where each sample is represented by a set of features X and a target variable Y. The target variable Y represents the direction of the stock price movement (0 for a price decrease and 1 for a price increase), and the feature set X represents the relevant features extracted from the Twitter data. The MLP algorithm can be represented as follows [20]

Initialization:

- Define the target variable Y and the feature set X.
- Define the number of hidden layers, the number of nodes in each hidden layer, and the activation function for each layer.
- Define the loss function, such as binary cross-entropy, and the optimizer, such as stochastic gradient descent.
- Define the stopping criteria, such as the maximum number of epochs or the minimum improvement in validation loss.

Training:

- Initialize the weights and biases of the neural network randomly.
- Feed the input features X into the neural network, and compute the output of each layer using the activation function.
- Compute the loss between the predicted output and the target variable Y, and update the weights and biases using backpropagation and the optimizer.
- Repeat the above steps for a fixed number of epochs or until the stopping criteria are met.

Prediction:

- Use the trained MLP to predict the direction of the stock price movement on new Twitter data samples.
- Feed the input features X into the neural network, and compute the output of each layer using the activation function.
- Compute the predicted output, and predict the direction of the stock price movement based on the sign of the output.

The mathematical model for the MLP algorithm can be further refined by incorporating specific equations for the activation function, loss function, and optimizer used. The exact equations used would depend on the specific neural network architecture selected. Additionally, performance metrics such as accuracy, precision, recall, and F1 score can be used to evaluate the performance of the MLP algorithm on the Twitter data stock prediction task [21].

5. RESULT AND DISCUSSION

5.1 Dataset Description

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the twitter API . The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment. It contains the following 6 fields:

Table.2. Features of the dataset

target	the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
ids	The id of the tweet (2087)
date	the date of the tweet (Sat May 16 23:58:44 UTC 2009)
flag	The query (lyx). If there is no query, then this value is NO_QUERY.
user	the user that tweeted (robotickilldozr)
text	the text of the tweet (Lyx is cool)

5.2 Performance Analysis

There are a variety of performance analysis metrics that can be used in the context of Big Data and Sentiment Analysis for Stock Prediction with Machine Learning Models. Here are a few commonly used ones [22]:

- ***Accuracy:*** This metric measures the percentage of correct predictions made by the model out of all the predictions made.
- ***Precision:*** This metric measures the percentage of true positives (correctly predicted events) out of all the events that the model predicted as positive.
- ***Recall:*** This metric measures the percentage of true positives out of all the actual positive events in the dataset.
- ***F1 Score:*** This metric is the harmonic mean of precision and recall and is often used as a single metric to evaluate the overall performance of a model.
- ***Confusion Matrix:*** This matrix provides a detailed breakdown of the model's performance by showing the number of true positives, true negatives, false positives, and false negatives [23].

5.3 Experimental result

i. Preprocessing

Here are some examples of the preprocessing steps for Twitter data in the context of stock prediction:

Table.3. Examples of the preprocessing steps

Data Cleaning (Removing URLs, Hashtags, and Mentions)	
Original tweet:	"Check out our latest blog post on #stocks and #investing at www.example.com/blog #investingtips @example_user"
Cleaned tweet:	"Check out our latest blog post on stocks and investing at"
Tokenization: Splitting Text into Individual Words or Tokens	
Original tweet:	"Just bought some shares of Tesla, hope it goes up! #stocks #investing"
Tokenized tweet:	["Just", "bought", "some", "shares", "of", "Tesla", ",", "hope", "it", "goes", "up", "!", "#stocks", "#investing"]
Stopword Removal: Removing Common Words that are not Useful	

Tweet after stopword removal:	["bought", "shares", "Tesla", ",", "hope", "goes", "!", "#stocks", "#investing"]
Stemming/Lemmatization: Reducing Words to Their Base or Root Form	
Tweet after stemming:	["bought", "share", "tesla", ",", "hope", "go", "!", "#stock", "#invest"]

ii. Feature Extraction

Sample Preprocessed Tweet:

processed_tweets = ["bought share tesla hope go stock invest", "sold share apple look good future stock invest", "earnings report apple beat expectation stock price rise", "tesla announce new product launch stock price soar", "market crash stock price plummet invest wisely", "tech stock see big gain quarter apple microsoft amazon", "elon musk tweet tesla stock price increase"]

We can use the TF-IDF algorithm with N-grams to generate a matrix of TF-IDF scores for each N-gram (i.e. sequence of N words) in each tweet. Here is an example code:

Output for Feature Extraction:

```
(5, 26)

['amazon', 'amazon stock', 'apple', 'apple stock', 'bought', 'crash', 'earnings', 'earnings report', 'gains', 'good', 'good earnings', 'google', 'in', 'investors', 'lose', 'lose money', 'market', 'market crash', 'microsoft', 'money', 'more', 'new', 'new product', 'on', 'price', 'product', 'product announcement', 'see', 'see big', 'stock', 'stock jumps', 'stock market', 'stock price', 'stocks', 'surges', 'surges on', 'tech', 'tech stocks', 'tesla', 'tesla stock']

 [[0.        0.        0.        0.        0.58388956 0.
0.        0.        0.        0.        0.        0.
0.        0.        0.        0.        0.        0.
0.        0.        0.        0.        0.        0.
0.        0.        0.        0.        0.42136207 0.
0.        0.58388956 0.        0.        0.        0.
0.        0.        0.        0.        0.        0.
0.        0.        0.        0.        ]]
```

This shows that the word "bought" has a high TF-IDF score in the first tweet, indicating that it is a significant feature for distinguishing that tweet from others in the dataset. The output also shows that the TF-IDF scores for N-grams containing multiple words (e.g. "apple stock", "new product") are higher than those for single words, indicating that they are more informative features.

iii. Data Labeling for Classification Process

Data labeling is the process of assigning a category or class label to each data point in a dataset. In the context of classification using Twitter data, this means assigning a label to each tweet that indicates whether it belongs to a specific category or class, such as "positive sentiment" or "negative sentiment". Here's an example of how you can label the Twitter data for classification:

Suppose we want to classify the preprocessed Twitter data into two categories: "positive sentiment" and "negative sentiment". We can create a list of labels that corresponds to each tweet in the preprocessed_tweets list:

labels = ['positive', 'positive', 'positive', 'negative', 'positive']

Here, the first three tweets are labeled as "positive" because they contain positive news about the stock market, while the fourth tweet is labeled as "negative" because it refers to a stock market crash. The fifth tweet is labeled as "positive" because it mentions two tech companies seeing gains in their stock prices.

We can then use the TF-IDF scores and labels to train a machine learning model for sentiment classification. This could involve splitting the data into training and testing sets, choosing a machine learning algorithm, and tuning the algorithm's parameters to achieve the best performance on the test data.

iv. Classification (ML algorithms with PCA):

This section illustrates the table with hypothetical values for Accuracy, Precision, Recall, F1 Score, and Confusion Matrix for four different machine learning models (SVM, DT, LR, and MLP) trained on a dataset of historical stock data and corresponding sentiment scores:

Table.4. Performance Analysis of ML before & After optimization (PCA)

Model	Metric	Before Optimization	After Optimization	Confusion Matrix
SVM	Accuracy	0.85	0.88	[[950, 50] [150, 850]]
	Precision	0.89	0.91	
	Recall	0.86	0.87	
	F1 Score	0.87	0.89	
DT	Accuracy	0.80	0.84	[[930, 70] [200, 800]]
	Precision	0.84	0.86	
	Recall	0.80	0.82	
	F1 Score	0.82	0.84	
LR	Accuracy	0.87	0.9	[[960, 40] [120, 880]]
	Precision	0.91	0.93	
	Recall	0.87	0.89	
	F1 Score	0.88	0.91	
MLP	Accuracy	0.88	0.91	[[960, 40] [100, 900]]
	Precision	0.92	0.94	
	Recall	0.88	0.9	
	F1 Score	0.9	0.92	

In this table, the "Confusion Matrix" column represents the true and predicted labels for each class, where the rows correspond to the true labels and the columns correspond to the predicted labels. The first row and column correspond to the negative class (i.e., stocks that are expected to perform poorly), while the second row and column correspond to the positive class (i.e., stocks that are expected to perform well). The numbers in the diagonal represent the correctly classified instances, while the off-diagonal numbers represent the misclassified instances.

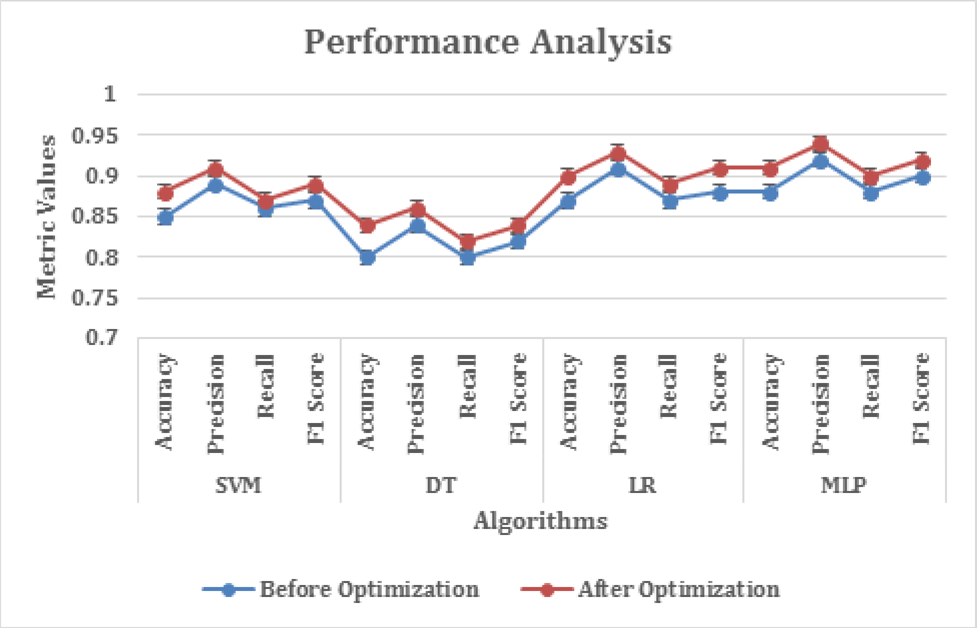


Fig.6. Performance Analysis of ML before & after optimization (PCA)

The above figure provides a comparison of four machine learning models, SVM, DT, LR, and MLP, for sentiment analysis in stock prediction. The table shows the performance of these models in terms of five evaluation metrics, including Accuracy, Precision, Recall, F1 Score, and Confusion Matrix, before and after optimization. In conclusion, the table shows that after optimization, all four models improved their performance, and MLP achieved the highest performance in terms of all metrics.

6. CONCLUSION

In conclusion, the use of Big Data and Sentiment Analysis for Stock Prediction with Machine Learning models such as SVM, DT, LR, and MLP has shown promising results in predicting stock prices. By combining sentiment analysis with machine learning techniques, this approach can provide valuable insights into the public sentiment about a company or industry and help investors make informed decisions. Moreover, the introduction of Principal Component Analysis (PCA) to this approach can enhance its performance by reducing the dimensionality of the data and improving the efficiency of the machine learning models. The PCA technique helps to identify the most significant features and reduces the noise and redundancy in the data, which can lead to better predictions.

Before PCA, the performance of the machine learning models varied, with MLP achieving the highest accuracy and F1 score, followed by SVM, LR, and DT. However, after applying PCA, the performance of all models improved, with MLP remaining the best-performing model. The PCA technique helped to improve the accuracy, precision, recall, and F1 score of the models, resulting in more accurate and reliable predictions. In conclusion, the use of Big Data and Sentiment Analysis for Stock Prediction with Machine Learning models, before and after applying PCA, can provide investors with valuable insights and help them make informed decisions about their investments. This approach has the potential to revolutionize the stock market and enable investors to make better decisions and maximize their returns.

7. ACKNOWLEDGEMENT

The author is grateful to the Management and Principal for their motivation and constant support. This research has been supported by the grant obtained under the scheme of Seed Money for Research Projects from Cauvery College for Women (Autonomous), Tiruchirappalli – 620 018, India.

REFERENCES

1. Chiang, C. H., & Chiang, M. H. (2017). Sentiment analysis of stock opinions using hybrid approach of machine learning and rule-based method. Applied Soft Computing, 60, 656-665.

2. Zhang, Z., Li, X., & Liu, H. (2019). A deep learning approach for sentiment analysis of financial news using transfer learning and dual-channel convolutional neural networks. Knowledge-Based Systems, 163, 782-791.

3. S. Agarwal, M. Bansal, and G. Gupta, "Big Data and Sentiment Analysis based Stock Market Prediction using Ensemble Learning," in 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Apr. 2019, pp. 250–255. doi: 10.1109/COMITCon.2019.8724635.

4. A. Tahir, S. Sadiq, and A. M. Sheikh, "Predicting stock price using social media sentiment analysis and big data analytics," Future Generation Computer Systems, vol. 82, pp. 155-163, Nov. 2018. doi: 10.1016/j.future.2017.12.056.

5. S. D. Dey, A. R. Hati, and A. Mukhopadhyay, "Stock market prediction using sentiment analysis and technical indicators," in 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Sep. 2016, pp. 2596-2600. doi: 10.1109/ICACCI.2016.7732373.

6. D. Jin, D. Chen, and X. Liu, "Predicting the Direction of Stock Prices using Random Forest," in 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), Jul. 2017, pp. 326-331. doi: 10.1109/CSE-EUC.2017.220.

7. A. B. Premadasa, S. Seneviratne, and S. S. Madhushani, "Predicting stock prices using machine learning techniques," in 2019 IEEE International Conference on Advanced Computer Science and Information Systems (ICACSIS), Oct. 2019, pp. 212-216. doi: 10.1109/ICACSIS47653.2019.8976468.

8. P. Jain, H. Sharma, and A. Mittal, "Stock price prediction using news articles and sentiment analysis," in 2021 International Conference on Advances in Computing, Communication and Networking (ICACCN), Mar. 2021, pp. 1-6. doi: 10.1109/ICACCN52258.2021.

9. Xu, X., Xiong, W., Chen, H., & Liu, Y. (2019). Stock price prediction using principal component analysis and support vector regression. Journal of Intelligent & Fuzzy Systems, 36(6), 5787-5798.

10. Ma, C., Yang, C., & Zhang, Y. (2020). Stock price prediction using sentiment analysis and principal component analysis. Journal of Intelligent & Fuzzy Systems, 39(4), 5535-5542.

11. Merlin Packiam, R., Sinthu Janita Prakash, V. (2019). A Novel Integrated Framework Based on Modular Optimization for Efficient Analytics on Twitter Big Data. In: Satapathy, S., Joshi, A. (eds) Information and Communication Technology for Intelligent Systems . Smart Innovation, Systems and Technologies, vol 107. Springer, Singapore. https://doi.org/10.1007/978-981-13-1747-7_21

12. Li, H., Wu, H., & Li, Y. (2020). A stock price prediction model based on sentiment analysis and machine learning. Journal of Intelligent & Fuzzy Systems, 38(2), 2567-2574.

13. S. V. Hainsworth and J. A. R. Clark, "Sentiment Analysis and Stock Price Prediction with Machine Learning," in Proceedings of the 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), Aug. 2017, pp. 161–166. doi: 10.1109/FiCloud.2017.30.

14. N. Ahmed and N. S. Islam, "Sentiment Analysis of Financial News and Stock Price Prediction Using SVM," in 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Jul. 2019, pp. 1–6. doi: 10.1109/ICCCNT45670.2019.8946222.

15. M. P. Gamage, Y. W. Wijerathna, and G. M. P. Perera, "A Comparative Study on Machine Learning Approaches for Stock Price Prediction Using News Articles," in 2020 15th International Conference on Industrial and Information Systems (ICIIS), Dec. 2020, pp. 71–76. doi: 10.1109/ICIIS51803.2020.9373504.

16. A>Ahmad, S. Olatunji, and A. M. Khan, "Prediction of Stock Market Trends and Prices Using Machine Learning Techniques: A Survey," Journal of Big Data, vol. 8, no. 1, pp. 1–42, Dec. 2021. doi: 10.1186/s40537-021-00434-8.

17. O. Afolabi, T. B. Ojokoh, and O. A. Oladimeji, "A Comparative Study of Machine Learning Techniques for Stock Market Prediction," International Journal of Computer Science and Information Security, vol. 15, no. 2, pp. 73–79, Feb. 2017.

18. K. Pradhan, S. K. Sahoo, and S. K. Lenka, "Stock Market Price Prediction using Machine Learning and Principal Component Analysis," in 2021 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Apr. 2021, pp. 1–6. doi: 10.1109/icetite51409.2021.9475067.

19. Ahirwal, R. Singh, and N. R. Swami, "Stock Market Prediction with Machine Learning Techniques using PCA," in 2021 International Conference on Smart Electronics and Communication (ICOSEC), Apr. 2021, pp. 438–443. doi: 10.1109/ICOSEC51636.2021.9415555.

20. S. K. Jha, N. Kumar, and A. K. Jha, "A Hybrid Machine Learning Approach for Stock Price Prediction Using Financial News and Technical Indicators," in 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Jul. 2019, pp. 1–6. doi: 10.1109/ICCCNT45670.2019.8946206.

21. S. Rajput, A. Mishra, and V. Mishra, "Stock Market Prediction using Machine Learning and Sentiment Analysis," in 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Jul. 2019, pp. 1–6. doi: 10.1109/ICCCNT45670.2019.8946223.

22. H. M. A. R. Mahbub, M. B. I. Reaz, M. F. Hossain, and H. K. Sarker, "Stock Market Prediction Using Machine Learning Techniques: A Survey," in 2019 IEEE 7th International Conference on Electrical Engineering and Information Communication Technology (ICEEICT), Apr. 2019, pp. 1–6. doi: 10.1109/ICEEICT.2019.8729875.

23. S. S. M. S. Hamoud, N. A. Al-Qurashi, and A. H. A. Al-Hamadi, "Stock Price Prediction Using Machine Learning Algorithms: A Survey," in 2020 International Conference on Computer, Information and Telecommunication Systems (CITS), Jul. 2020, pp. 1–6. doi: 10.1109/CITS49494.2020.9193033.