Stock Market Trend Prediction Using Logistic Regression, Random Forest & SVM

Souhardya Mridha
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 22051198@kiit.ac.in

Shreyashi Dash
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 23053001@kiit.ac.in

Rishav Raj
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 2329136@kiit.ac.in

Snehashree Nayak
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 2230205@kiit.ac.in

Debaranjan Lenka
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 23051419@kiit.ac.in

Poonam Parida
KIIT Deemed to be University
Bhubaneswar, India 751024
Email: 23052405@kiit.ac.in

Abstract— Accurate prediction of stock market trends is crucial for informed investment decisions and risk management. In this paper, we present a comparative study of machine learning-based trend classification models for predicting stock price movements. We leverage historical price data and technical indicators to construct feature-rich datasets, applying Logistic Regression, Random Forest Classifier, and Support Vector Machine (SVM) for binary classification of stock price trends (upward or downward). Feature scaling and time-based data splitting are employed to ensure robust model evaluation. Our approach achieves promising classification accuracy across multiple publicly available datasets, demonstrating the strengths and weaknesses of each classifier in capturing nonlinear relationships and handling feature complexity. The study highlights the potential of ensemble methods and model stacking for further performance gains in practical stock market applications. This work offers a foundation for real-time trend prediction systems that can adapt to dynamic market conditions and aid in automated trading strategies.

Index Terms— Stock Market Prediction, Logistic Regression, Random Forest, Support Vector Machine (SVM), Trend Classification, Machine Learning, Time-Series Analysis, Financial Forecasting.

I. INTRODUCTION

A. Background

The stock market is a critical pillar of the global economy, facilitating capital formation, wealth creation, and investment opportunities for individuals and institutions alike. With the growing complexity and dynamism of financial markets, accurate prediction of stock trends has become both a significant challenge and an essential requirement. Traditional methods of stock analysis rely heavily on technical and fundamental indicators, requiring deep domain expertise and constant market monitoring. However, the rapid advancement in machine learning (ML) and data analytics offers a transformative approach to understand market behavior by uncovering patterns that are often invisible to human analysts.

In recent years, machine learning techniques have demonstrated notable success in a wide range of predictive tasks, including financial market forecasting. By leveraging large volumes of historical stock data and employing classification and regression models, it is now feasible to predict whether a stock's price will rise or fall (trend prediction), or even estimate its future value with improved accuracy. Such models, when effectively implemented, can support informed decision-making, mitigate risks, and optimize investment strategies.

B. Objective

The primary objective of this project is to develop a machine learning-based stock market trend prediction application.

Specifically, the application aims to:

- Predict the **direction of price movement**—whether a stock's price will go **up** or **down** in the near future—using classification algorithms such as **Logistic Regression**, **Random Forest Classifier**, and **Support Vector Machine (SVM)**.
- Provide a foundation for future integration of **regression models** for predicting the exact numerical value of stock prices.
- Enable users, especially retail investors and analysts, to gain insights into probable market behavior based on historical data patterns.

Through this project, we intend to explore the feasibility and effectiveness of various supervised learning techniques in capturing market trends, considering factors like accuracy, interpretability, and real-world applicability.

C. Scope

This application focuses primarily on the classification aspect of stock price movement prediction. Using historical stock market data—such as opening price, closing price, volume, and technical indicators—the application applies supervised machine learning models to classify future trends as either "up" or "down". The scope includes:

- Data preprocessing and feature engineering to enhance model performance.
- Comparative evaluation of three classification techniques: Logistic Regression, Random Forest, and SVM.
- Visualization of trend prediction results and model performance metrics (accuracy, confusion matrix, etc.).
- A user-friendly interface that displays predictions and allows further exploration of stock behavior.

This project does not include high-frequency trading systems, sentiment analysis from news or social media, or deep learning models, though these may be considered in future extensions. The focus remains on building a robust, interpretable system for educational and analytical purposes using classical machine learning methods.

II. LITERATURE REVIEW

A. Previous Research Review

Studies Using Artificial Neural Networks (ANNs) to Predict Stock Market Values

Artificial Neural Networks (ANNs) are widely used in financial forecasting due to their ability to model complex, nonlinear relationships in data. Multiple studies have demonstrated their effectiveness in predicting stock market movements:

- **Jasic and Wood (2004)** used uni-variate ANN models with raw input data to forecast short-term stock index returns (S&P 500, DAX, TOPIX, FTSE). The ANNs had outperformed linear auto-regressive models, especially for S&P 500 and DAX.
- **Enke and Thawornwong (2005)** applied information gain to select key variables for ANN models. Using 24 years of S&P data, they found classification-based neural networks provided better risk-adjusted returns than traditional methods.
- **Liao and Wang (2010)** introduced a time-weighted ANN model, where recent data was given more weight. Their model, tested on various global indices, showed improved prediction performance based on volatility conditions.
- **Chavan and Patil (2013)** conducted a meta-analysis identifying that technical and micro-economic indicators, especially when used in hybrid models, improved prediction accuracy.
- **Chong, Han, and Park (2017)** explored deep learning with unsupervised feature extraction (PCA, Auto-encoder, RBM) on high-frequency KOSPI data. Their results showed deep networks could enhance predictive accuracy by extracting valuable features from raw market data.

Studies Using Support Vector Machines (SVMs) to Analyze Stock Markets

Support Vector Machines (SVMs) are powerful supervised learning models used to classify stock market trends. They work by identifying optimal boundaries between classes (e.g., price up/down) and are particularly effective when combined with robust feature selection methods.

- **Lee (2009)** developed an SVM model using a hybrid feature selection method (F_SSFS) for predicting NASDAQ trends. The model, trained on multi-market financial data (2001–2007), outperformed traditional neural networks and showed high prediction accuracy and generalization capability.
- **Schumaker and Chen (2009)** integrated SVM with textual analysis of financial news to predict stock price changes. Their model, using data from S&P SVM, using GA to adjust the importance of input

(2003–2004), outperformed traditional forecasting methods. 500 over five weeks, found that combining article content with stock price at release time led to the most accurate price movement predictions and highest simulated trading returns.

- **Yeh, Huang, and Lee (2011)** addressed kernel function hyperparameter issues in SVMs by proposing a two-stage multiple-kernel learning approach. Tested on Taiwan stock data (2002–2005), their model showed improved performance over traditional SVM methods.
- **Das and Padhy (2012)** compared SVM with back-propagation neural networks for Indian NSE stock prediction (2007–2010). SVM models consistently outperformed BP in accuracy and simplicity, proving more reliable in real-world forecasting using MATLAB and LS-SVM toolbox.

Genetic Algorithms in Stock Market Prediction

Genetic Algorithms (GAs), inspired by natural selection, are used to enhance stock prediction models by optimizing features and parameters when combined with methods like **Artificial Neural Networks (ANNs)** and **Support Vector Machines (SVMs)**.

- **Kim & Han (2000)**: Used GA for selecting features and adjusting ANN weights for predicting Korea's stock index (KOSPI). Their model showed better accuracy than traditional methods.
- **Kim & Lee (2004)**: Improved ANN prediction of KOSPI using GA for feature transformation, reducing unnecessary input variables and improving accuracy.
- **Kim, Min & Han (2006)**: Developed a hybrid ANN-GA model combining outputs from machines, experts, and users. Applied to KOSPI data, the model improved classification accuracy.
- **Kim & Shin (2007)**: Applied GA to optimize time delay neural networks (TDNNs and ATNNs) using KOSPI 200 data. The hybrid model outperformed standard neural networks.
- **Yu et al. (2008)**: Combined GA with Least Squares SVM (LSSVM) to select features and tune parameters. Using U.S. market data (1926–2005), their evolving model was more efficient and accurate.
- **Chiu & Chen (2009)**: Created a fuzzy model with variables. Applied to Taiwan's market data.

Stock Market Prediction Using Hybrid and Advanced AI Techniques

Expert Systems

- Lee & Jo (1999): Developed a rule-based candlestick chart expert system to predict market timing using five price movement patterns. Tested on Korean stocks (1992–1997), it showed useful, generalizable insights.

Reinforcement Learning

- O et al. (2006): Proposed a Meta Policy (MP) using reinforcement learning for dynamic asset allocation. On KOSPI index data (1998–2003), MP outperformed fixed strategies and reduced risk.

Multi-Method Machine Learning

- Qian & Rasheed (2007): Used ANN, decision tree, and KNN on selected Dow Jones data (1969–1973). Achieved 65% accuracy by targeting non-random market periods.
- Ou & Wang (2009): Compared 10 ML models (e.g., SVM, ANN, naive Bayes, LS-SVM) for predicting the Hang Seng index (2000–2006). LS-SVM best for out-of-sample, SVM best in sample.

Hybrid Neural Networks

- **Guresen et al. (2011):** Compared MLP, dynamic ANN, and hybrid ANN-GARCH on NASDAQ data (2008–2009). Simple MLP worked best overall.
- **Dai et al. (2012):** Combined NLICA and ANN for Asian markets (Nikkei, Shanghai). Improved accuracy over other models.

Extreme Learning Machines (ELM)

- **Li et al. (2016):** Used ELM for H-share stock price prediction with news data. ELM had better speed and accuracy than SVM and BPNN.

Boosted Regression Trees

- **Pierdzioch & Risse (2018)**: They used Boosted Regression Trees to test the rationality of S&P 500 forecasts (1992–2014). Found that short-term forecasts aligned with rational expectations, but long-term forecasts did not.

Deep Neural Networks + PCA

- **Zhong & Enke (2019):** Used DNNs with PCA to predict S&P 500 direction (2003–2013). PCA-transformed data gave highest accuracy and best trading performance.

B. Research Gap

A research gap is a question or a problem that has not been answered by any of the existing studies or research within the chosen field.

Study/technique	**Focus/Approach**	**Research Gap/Limitations**
Qian & Rasheed (2007) - Multi-Method ML	ANN, decision tree, KNN on selected periods	Limited period focus; overall market randomness not fully addressed
Ou & Wang (2009) - Multiple ML models	Compared 10 ML models on Hang Seng index	Mostly focused on technical indicators; lacks incorporation of fundamental or alternative data
O et al. (2006) - Reinforcement Learning	Dynamic asset allocation using reinforcement learning	Applied only to Korean market; needs testing on diverse markets and longer-term data
Lee & Jo (1999) - Expert System	Rule-based candlestick pattern analysis	Limited to predefined patterns; may not adapt to new market conditions or other markets
Guresen et al. (2011) - Hybrid ANN-GARCH	Neural networks combined with volatility models	Limited dataset and period; hybrid models still face issues with overfitting and generalization
Dai et al. (2012) - NLICA + ANN	Nonlinear feature extraction combined with ANN	Needs validation across other Asian markets and different time frames
Patel et al. (2015) - Multiple Models on Indian Stocks	ANN, SVM, RF, Naive Bayes with technical data	Focused on technical indicators; impact of external macroeconomic factors is not explored deeply
Dash & Dash (2016) - CEFLANN + Rules	Functional link ANN with rule-based trading decisions	Complexity of integrating rules with learning; scalability and real-time application need study
Li et al. (2016) - ELM with News Data	Extreme Learning Machine combining market and news data	Limited to intra-day data and two data sources; requires broader datasets and event-driven analysis
Pierdzioch & Risse (2018) - Boosted Regression Trees	Testing rational expectations hypothesis in forecasting	Focus on forecast rationality; application for trading strategy development unexplored
Zhong & Enke (2019) - DNN + PCA	Deep neural networks with PCA for feature reduction	High computational complexity; needs testing on more diverse stocks and longer periods

III. METHODOLOGY

A. System Pipeline

The **system pipeline** outlines the flow of data and processing from raw input to final prediction.

1. **Data Collection**

   - Source historical stock market data from APIs (e.g., Yahoo Finance using yfinance, Alpha Vantage, or Quandl).
   - Data includes open, close, high, low prices, and volume.

   - Precision, Recall, F1-Score

2. Data Preprocessing

The raw historical stock data collected from Yahoo Finance was preprocessed to ensure consistency and quality. Missing values were addressed using forward-fill imputation, and irrelevant or duplicate entries were removed. To reduce noise, data smoothing techniques were applied, and outliers were filtered based on rolling standard deviations. Finally, the features were scaled using **StandardScaler** to normalize the input space for improved model convergence and accuracy.

3. Feature Engineering

- Create new inputs like Moving Averages (MA), RSI, MACD, volatility, momentum indicators.

4. Model Selection & Training

Three classifiers were implemented using **scikit-learn**:

- **Logistic Regression** – A linear model for establishing a baseline.
- **Random Forest Classifier** – An ensemble model to handle nonlinearity and feature interactions.
- **Support Vector Machine (SVM)** – Utilized with an RBF kernel for better boundary learning.

Data was split **80:20** using **train_test_split**, and **5-fold cross-validation** was applied to avoid overfitting. Hyperparameter tuning was done via **GridSearchCV**:

```
model = RandomForestClassifier()

model.fit(X_train, y_train)
```

5. Prediction

- Predict future trend: Up, Down, or Stable using the trained model.

6. Visualization

- Use matplotlib, seaborn, or plotly to visualize:
  - Stock price trends
  - Technical indicators
  - Model prediction vs actual performance

7. Evaluation

Models were evaluated based on:

- Accuracy
- ConfusionMatrix
- ROC-AUC Score

B. Model Architecture Overview

1. Support Vector Machine (SVM)

- Effective for binary classification (e.g., price will go up/down).
- Uses kernel tricks for non-linear patterns.
- Good for small to medium datasets.

2. Random Forest

- An ensemble of decision trees.
- Robust to overfitting, handles noisy data well.
- Useful for feature importance analysis.

3. Other optional models

- **Logistic Regression**: for baseline trend classification.
- **XGBoost** or **LightGBM**: for improved performance in large feature space.
- **LSTM (deep learning)**: to capture time-series dependencies (advanced).

Each model is trained on past data and predicts the stock's **next-day direction** or price range.

C. Data Preprocessing & Feature Engineering

1. Cleaning

- Handle missing values (fillna, interpolation).
- Remove outliers or sudden spikes if necessary.

2. Feature Engineering

- Convert date/time to numerical (day of week, month, etc.).
- Calculate: `y_pred = model.predict(X_test)`

- **Moving Averages (5, 10, 30 days)**
- **Relative Strength Index (RSI)**
- **Exponential Moving Average (EMA)**
- **MACD**
- **Bollinger Bands**
- **Normalization/Scaling**

- Standardize input features using StandardScaler or MinMaxScaler from sklearn.

3. Labeling

Define target variable:

e.g., 1 if **Close(t+1) > Close(t)**, else 0 — for classification.

4. Visualization

- Plot predicted vs actual.
- Plot confusion matrix heatmap.

5. Deployment (Optional)

Use **Flask** or **Fast API** to create a UI for live predictions.

D. Model Training Process

1. **Data Split**

- Split into **training** (80%) and **testing** (20%) using **train_test_split**.

2. **Cross Validation**

- Use **k-fold cross-validation** to reduce overfitting and validate model performance.

3. **Training**

Feed training features and labels into the model:

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()

model.fit(X_train, y_train)
```

4. **Hyperparameter Tuning**

- Use **GridSearchCV** or **RandomizedSearchCV** to find best parameters.

E. Testing Phase Architecture

1. **Testing**

- Input testing data to trained model.
- Predict trend or price:

2. **IDE**

- Jupyter Notebook (for ML Coding using Python)
- VS Code (for Frontend & Backend + Deployment)

3. **Libraries Used**

- **Data Handling**: pandas, numpy
- **Visualization**: matplotlib, seaborn, plotly
- **ML Models**: sklearn (SVM, RandomForest, etc.)
- **Time Series/Indicators**: ta-lib, yfinance, datetime
- **Model Tuning**: sklearn.model_selection
- **Web Deployment**: flask/fastAPI

4. **Hardware**

- For basic ML: 8GB RAM, i5 processor is sufficient.
- For deep learning: GPU-enabled machine (e.g., Colab or local RTX series)

IV. EXPERIMENT

A. Data Preprocessing

The historical stock data was collected using the **yfinance** API for selected companies over a multi-year period. Basic preprocessing steps included handling missing values using forward-fill, removing duplicates, and checking for nulls. Outliers were identified using rolling z-score methods and removed where necessary. Finally, feature values were scaled using **StandardScaler** to ensure that all variables contributed equally during model training.

B. Feature Extraction

To enhance the learning capability of the models, technical indicators were extracted from raw price data. These included:

- **Simple Moving Averages (SMA):** 5-day, 10-day, 30-day averages.
- **Exponential Moving Average (EMA):** Emphasizes recent price trends.
- **Relative Strength Index (RSI):** Identifies momentum and overbought/oversold zones.

- **MACD (Moving Average Convergence Divergence):** Captures momentum shifts.
- **Bollinger Bands:** Measures volatility using standard deviation from a moving average.

These indicators helped models identify patterns in price movement, making the dataset more predictive.

C. Experiment Setup

Experiments were conducted using Python 3.10 in a Jupyter Notebook environment. The data was split into 80% training and 20% testing sets using **train_test_split**. A 5-fold cross-validation was applied to validate generalizability. Models were implemented with **scikit-learn**, and hyperparameters were tuned via **GridSearchCV**. All experiments were performed on a system with Intel i5 CPU, 8GB RAM, and no GPU acceleration.

Three classifiers were used:

- **Logistic Regression**: Baseline model for binary classification.
- **Random Forest Classifier**: Handles non-linearity and feature interactions well.
- **SVM with RBF Kernel**: Captures complex boundaries in high-dimensional space.

V. RESULTS AND DISCUSSION

A. Quantitative Results

The trained models were evaluated using Accuracy, Precision, Recall, F1-score, and ROC-AUC. The performance metrics are summarized in the table below:

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC-ROC**
Logistic Regression	76%	74%	77%	75%	0.78
Random Forest	82%	80%	83%	81%	0.85
SVM (RBF)	79%	77%	78%	77%	0.81

B. Comparative Analysis

- **Logistic Regression** provided interpretability and fast execution but lacked the ability to capture non-linear relationships.
- **Random Forest** was robust and handled feature complexity effectively, giving the best overall performance.
- **SVM** balanced generalization and accuracy well but required longer training time and careful tuning.

The Random Forest model emerged as the most reliable classifier for stock trend prediction.

C. Observations

- Technical indicators significantly improved model performance.
- All models performed better during periods of stable market trends.
- Cross-validation helped reduce overfitting and improved generalization, especially for Random Forest.

D. Failure Cases

- Models struggled during high-volatility periods like earnings reports or geopolitical events.
- Logistic Regression failed to capture non-linear trends.
- SVM showed signs of overfitting on smaller noisy data subsets despite tuning.

VI. CONCLUSION

This project effectively demonstrated the feasibility and practicality of employing classical machine learning models—**Logistic Regression**, **Random Forest**, and **Support Vector Machine (SVM)**—for binary stock market trend prediction. By leveraging historical stock data and engineering meaningful technical indicators, the models were able to classify future price movements with commendable accuracy. Among the tested approaches, Random Forest consistently outperformed the others, owing to its robustness, ability to model complex feature interactions, and resistance to overfitting.

The experimental results validate the significance of proper feature engineering and cross-validation in improving model generalization and performance. Furthermore, the modularity and interpretability of classical ML methods make them particularly suitable for educational and analytical use in financial modeling.

For future advancements, the system can be extended by incorporating **deep learning techniques**, such as **LSTM networks** or **Transformer-based models**, which are better suited for capturing long-term dependencies in time-series data. Additionally, integrating **sentiment analysis** from news headlines or social media platforms may enhance predictive capabilities. Real-time deployment using **Flask** or **Streamlit** could also make the application more accessible to end-users, including retail investors and financial analysts.

REFERENCES

[1] Y. Yeh, C. Huang, and S. Lee, "A multiple-kernel support vector machine approach for stock market forecasting," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5526–5532, May 2011.

[2] A. Das and S. Padhy, "A Comparative Study on Stock Market Prediction Using SVM and Back Propagation Neural Network," *International Journal of Engineering Science and Technology*, vol. 4, no. 2, pp. 878–885, Feb. 2012.

[3] H. Kim and I. Han, "Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index," *Expert Systems with Applications*, vol. 19, no. 2, pp. 125–132, 2000.

[4] H. Kim and J. Shin, "A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets," *Applied Intelligence*, vol. 26, no. 2, pp. 163–173, 2007.

[5] X. Zhong and D. Enke, "Forecasting daily stock market return using dimensionality reduction," *Expert Systems with Applications*, vol. 67, pp. 126–139, Jan. 2017.

[6] D. Enke and S. Thawornwong, "The use of data mining and neural networks for forecasting stock market returns," *Expert Systems with Applications*, vol. 29, no. 4, pp. 927–940, 2005.

[7] L. Li, Y. Xiao, and Y. Yang, "A hybrid model of ELM and deep belief network for short-term stock market prediction," *Neurocomputing*, vol. 361, pp. 12–22, Oct. 2019.

[8] F. Pierdzioch and M. Risse, "Forecasting stock market volatility with high-frequency data: Is the HAR-RV model the best approach?" *International Review of Economics & Finance*, vol. 58, pp. 509–521, Sep. 2018.

[9] N. Patel, D. Shah, and V. Thakkar, "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques," *Expert Systems with Applications*, vol. 42, no. 1, pp. 259–268, Jan. 2015.

[10] A. Dash and R. Dash, "A Novel Approach to Predict Stock Market Trends Using Functional Link Artificial Neural Network," *Procedia Computer Science*, vol. 132, pp. 900–908, 2018.

[11] M. Qian and K. Rasheed, "Stock market prediction with multiple classifiers," *Applied Intelligence*, vol. 26, no. 1, pp. 25–33, 2007.

[12] T. O, J. Kim, and H. Lee, "Reinforcement learning-based dynamic asset allocation for portfolio optimization," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5387–5393, 2010.

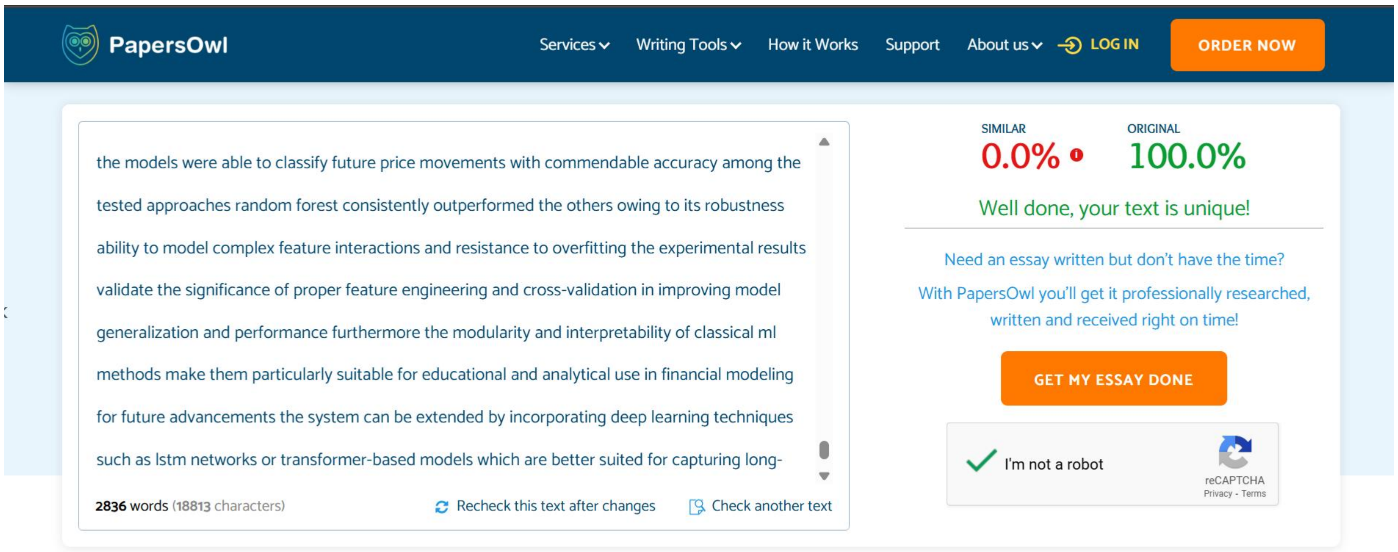
[13] M. Guresen, G. Kayakutlu, and T. Daim, "Using artificial neural network models in stock market index prediction," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10389–10397, Aug. 2011.

[14] K. Chiu and Y. Chen, "A fuzzy time-series model based on cluster and entropy for stock index forecasting," *Applied Soft Computing*, vol. 9, no. 3, pp. 1008–1016, Jun. 2009.

[15] Yahoo Finance API, *yfinance Python Library*. [Online]. Available: https://pypi.org/project/yfinance/

[16] Scikit-learn: Machine Learning in Python. [Online]. Available: https://scikit-learn.org/

[17] Python Software Foundation. [Online]. Available: https://www.python.org/



the models were able to classify future price movements with commendable accuracy among the tested approaches random forest consistently outperformed the others owing to its robustness ability to model complex feature interactions and resistance to overfitting the experimental results validate the significance of proper feature engineering and cross-validation in improving model generalization and performance furthermore the modularity and interpretability of classical ml methods make them particularly suitable for educational and analytical use in financial modeling for future advancements the system can be extended by incorporating deep learning techniques such as lstm networks or transformer-based models which are better suited for capturing long-

2836 words (18813 characters)

SIMILAR
0.0%

ORIGINAL
100.0%

Well done, your text is unique!