ESG RISK ANALYSIS & STOCK MARKET PREDICTION

by

K.GEETHANJALI
Roll No. 214G1A3324

S.MOHAMMED GHOUSE
Roll No. 214G1A3354

G.GANESH
Roll No. 214G1A3323

S.ISMA MEHARAZ
Roll No. 214G1A3334

Under the guidance of

Dr. C. NAGESH M. Tech.
Assistant Professor

Department of Computer Science and Engineering (AI & ML)

Srinivasa Ramanujan Institute of Technology

(Affiliated to JNTUA & Approved by AICTE) (Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE)

Rotarypuram Village, B K Samudram Mandal, Ananthapuramu – 515701.

2024 - 2025

Contents

Introduction

- In recent years, the significance of Environmental, Social, and Governance (ESG) factors in investment decisions has surged, reflecting a broader societal shift toward sustainable and responsible investing.
- The stock market, as a primary vehicle for capital allocation, plays a critical role in reflecting the underlying health of the economy.
- Traditional stock market prediction models have primarily focused on quantitative financial data, often neglecting qualitative factors like ESG risks.
- This study aims to explore the relationship between ESG risk analysis and stock market prediction, investigating how the incorporation of ESG metrics can enhance the accuracy of predictive models.

Proposed System

Proposed System:

ESG Risk Analysis & Stock Market Prediction using Machine Learning involves integrating environmental, social, and governance (ESG) factors into financial forecasting models. Here's a five-point overview of a proposed system:

- **Data Collection and Preprocessing:** The system gathers data from financial markets, ESG reports, and news sentiment, cleaning and normalizing it for use in machine learning models alongside financial data.
- **Feature Engineering:** ESG factors like carbon emissions and employee diversity are transformed into features to assess their impact on stock prices, combined with economic data for financial risk prediction.
- **Model Selection and Training:** Machine learning models such as Random Forest, Gradient Boosting, and LSTM are used to predict stock trends, with ongoing training and updates for accuracy.

Proposed System

- **Risk Scoring and Forecasting:** The model predicts stock movements and assigns ESG risk scores, identifying companies with high volatility due to poor ESG performance.
- **Decision Support and Portfolio Management:** The system assists investors by displaying ESG risks alongside market predictions, helping to make informed investment decisions and create sustainable portfolios.

Advantages of Proposed System:

- Comprehensive Data Integration
- Enhanced Predictive Accuracy
- Dynamic Learning and Adaptation
- Standardization of ESG Metrics
- User-Friendly Interface and Visualization Tools

Planning

- **Objective:** Develop a machine learning model to analyze Environmental, Social, and Governance (ESG) risk factors and predict stock market trends, integrating a Gradio interface for user interaction.
- **Project Plan:**

1. Problem Understanding:

1. Analyze the impact of ESG risk factors on stock market performance.
2. Understand the correlation between stock prices and sustainability metrics.

2. Data Collection & Preprocessing:

1. Load stock market data and ESG risk ratings.
2. Handle missing values, normalize numerical data, and encode categorical features.

3. Exploratory Data Analysis (EDA):

1. Visualize stock trends, ESG risk scores, and their relationships.

Planning

4. **Feature Engineering & Selection:**

    1. Compute moving averages and other relevant stock indicators.
    2. Select essential features for stock price prediction.

5. **Model Development:**

    1. Train machine learning models (Random Forest, Gradient Boosting) on historical data.

6. **Evaluation & Performance Metrics:**

    1. Assess models using RMSE, MAE, and R^2 score.
    2. Compare different models to select the best-performing one.

7. **Deployment & User Interface:**

    1. Develop an interactive Gradio interface for real-time stock and ESG analysis.
    2. Enable users to input stock details and get ESG-adjusted price predictions.

System Requirements

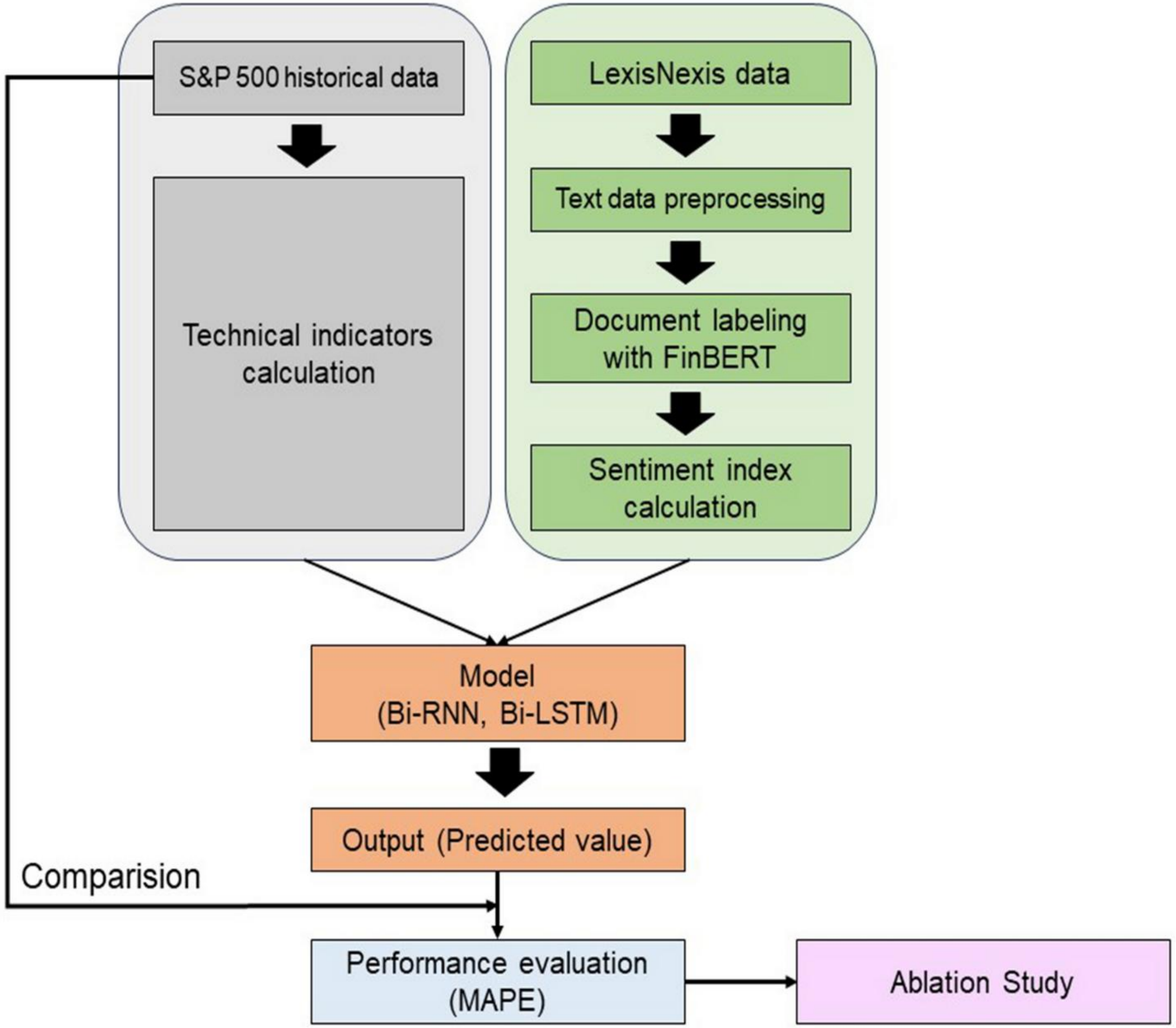
➢ HARDWARE REQUIREMENTS:

System	:	Pentium IV 2.4 GHz.
Hard Disk	:	40 GB.
Ram	:	512 Mb.

➢ SOFTWARE REQUIREMENTS:

Operating system	:	Windows.
Coding Language	:	Python.

System Architecture



Data Preprocessing Techniques

1. Handling Missing Values:

1. Used the **median imputation** technique to fill missing values in the ESG dataset.
2. Ensured no null values remained in the dataset after imputation.

2. Categorical Data Encoding:

1. Applied **Label Encoding** to convert categorical columns (except 'Symbol') into numerical values for machine learning compatibility.

3. Feature Scaling (Normalization):

1. Used **MinMaxScaler** to normalize the stock price and ESG risk score features.
2. This ensures that all numerical features are on a comparable scale, improving model performance.

Data Preprocessing Techniques

4. Feature Engineering:

1. Calculated **10-day and 50-day moving averages** for stock closing prices to capture short-term and long-term trends.

5. Date Feature Extraction:

1. Extracted **year, month, and day** from the date column to analyze trends over time.

6. Data Merging & Transformation:

1. Merged **ESG risk scores with stock market data** based on the 'ticker' symbol for comprehensive analysis.
2. Filtered and transformed stock data for company-wise performance evaluation.

Implementation

Algorithms:

1. **Random Forest:** Random Forest is used as a **regression model** to predict future stock prices based on historical data and ESG risk scores. Stock prices fluctuate due to multiple complex factors; Random Forest captures non-linear relationships well.

- Stock market data is unpredictable, and Random Forest helps smooth out random fluctuations.
- Helps in understanding which features (e.g., closing price, moving averages, ESG scores) impact stock prediction the most.

2. **Gradient Boosting:** Gradient Boosting is an **ensemble learning technique** Unlike Random Forest (which averages multiple trees), Gradient Boosting learns from mistakes and refines predictions.

- Stock price trends involve hidden patterns; Gradient Boosting is effective at capturing them.
- Since it focuses on **minimizing errors**, it helps achieve better stock price forecasts.

Implementation

Metrics:

1. R^2 Score: The **R^2 Score** measures how well the model explains the variance in stock prices. It indicates the proportion of the variation in the dependent variable (stock price) that is predictable from the independent variables (features).

Formula: $$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

2. Root Mean Squared Error (RMSE): The **Root Mean Squared Error (RMSE)** measures the standard deviation of prediction errors. It calculates how much the predicted stock prices deviate from the actual stock prices, with a higher penalty for larger errors.

Formula: $$RMSE = \sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$$

Implementation

3. Mean Absolute Error (MAE) : The **Mean Absolute Error (MAE)** represents the average absolute difference between actual and predicted stock prices. Unlike RMSE, MAE treats all errors equally without squaring them, making it less sensitive to outliers.

Formula: $$MAE = \frac{1}{n} \sum |y_i - \hat{y}_i|$$

Sample Code

```
# Fetch stock data using yfinance
stock_data = yf.download(stock_ticker, start="2023-01-01", end="2024-01-01")
stock_data.reset_index(inplace=True)

# Feature Engineering
stock_data['MA_10'] = stock_data['Close'].rolling(window=10).mean()
stock_data['MA_50'] = stock_data['Close'].rolling(window=50).mean()
stock_data.dropna(inplace=True)

# Selecting features and target
X = stock_data[['Close', 'MA_10', 'MA_50']].values
Y = stock_data['Close'].values.reshape(-1, 1)

# Normalizing Features
X = features_scaler.fit_transform(X)
Y = target_scaler.fit_transform(Y)

# Splitting dataset
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=42)

# Training Models
rf_model = RandomForestRegressor(n_estimators=100, max_depth=10)
rf_model.fit(X_train, y_train.ravel())

gb_model = GradientBoostingRegressor(n_estimators=100, max_depth=10)
gb_model.fit(X_train, y_train.ravel())

input_data = np.array([[close, ma_10, ma_50]])
input_data = features_scaler.transform(input_data)
```

Sample Code

```
if model_choice == "Random Forest":
    predicted_price = rf_model.predict(input_data)
else:
    predicted_price = gb_model.predict(input_data)

predicted_price = target_scaler.inverse_transform(predicted_price.reshape(-1, 1)).ravel()[0]

# Generate Prediction vs Actual Graph
y_pred_test = rf_model.predict(X_test) if model_choice == "Random Forest" else gb_model.predict(X_test)
y_pred_test = target_scaler.inverse_transform(y_pred_test.reshape(-1, 1))
y_test_actual = target_scaler.inverse_transform(y_test.reshape(-1, 1))

plt.figure(figsize=(10, 5))
plt.plot(y_test_actual, label="Actual Price", color="blue", alpha=0.6)
plt.plot(y_pred_test, label="Predicted Price", color="red", alpha=0.8)
plt.legend()
plt.xlabel("Days")
plt.ylabel("Stock Price")
plt.title(f"{model_choice} - Predicted vs Actual Prices")
plt.grid(True)
plt.savefig("prediction_graph.png")
plt.close()

# Feature Importance Graph for Random Forest
if model_choice == "Random Forest":
    feature_importances = rf_model.feature_importances_
    feature_names = ['Close', 'MA_10', 'MA_50']

    plt.figure(figsize=(8, 5))
```

Output

ESG Risk Analysis And Stock Market Prediction

Enter stock details to predict future prices using machine learning models. The graphs display actual vs predicted prices and feature importance for Random Forest.

Stock Ticker

GOOG

Closing Price

244.60

10-Day Moving Average

233.74

50-Day Moving Average

240.51

Select Model

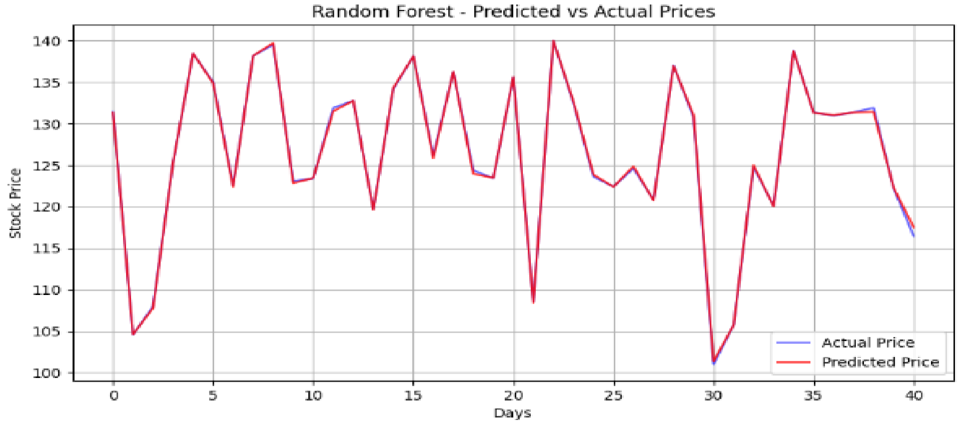
Random Forest | Gradient Boosting

Clear | Submit

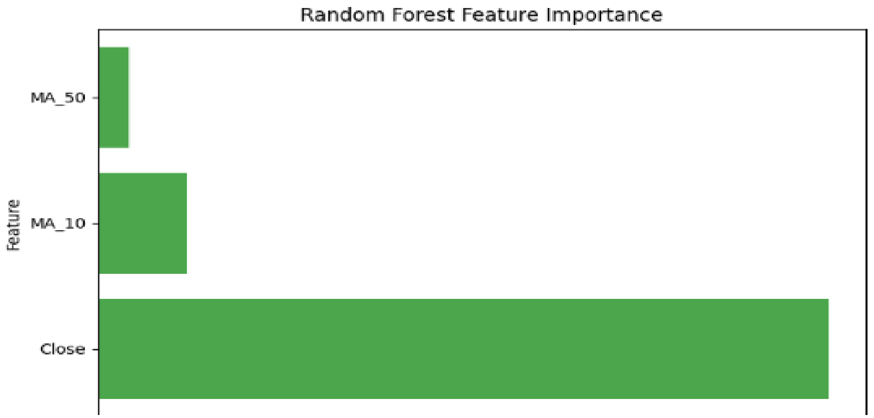
output 0

Predicted Stock Price: $141.98

output 1



output 2



Any Queries?

Thank You!!!