DEEP LEARNING CEC PROJECT REPORT

ON

*Stock Market Prediction App using XGBoost*

*Submitted By :*

Dev Bhatt (22BT04012)

Austin Jireny(22BT04045)

Harsh Mehta(22BT04067)

Megh Modi (22BT04070)

*Submitted To:*

Mr. Deepak Parashar

*Submitted in partial fulfillment for the Contiuous Evaluation Component (CEC)*

*Of the subject*

**Deep Learning ((BTCS604))**

in

Semester-VI Computer Science & Engineering (AI ML and Data science.)

*At*

Department of Computer Science and Engineering School of Technology



GSFC University,

Vigyan Bhavan, P. O. Fertilizer Nagar,

Vadodara - 391750, Gujarat, India

# ACKNOWLEDGEMENT

We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout the duration of this project.

Firstly, we are extremely grateful to **GSFC University**, for providing us the excellent working environment to undergrow our Minor Project.

We are deeply thankful to **Mr. Deepak Parashar** , our Faculty Mentor and Assistant Professor in Computer Science & Engineering Department at School of Technology, GSFC University, whose useful suggestions, gentle soothing attitude and right decision helped us a lot to learn in this project and also for her constant encouragement and support throughout the Project.

Furthermore, we are grateful to our fellow peers for their support and collaboration throughout this experience. Their encouragement and teamwork have made this project a memorable and enriching journey

# ABSTRACT

The research aims to explore predicting stock prices via the XGBoost regression model from historic finance data. The application supports forecasting short-run price movements via picking stock ticker symbols and prediction intervals. Normalization and feature engineering are conducted during preprocessing stages for enhancing predictive model accuracy. Measures such as MAE, RMSE, and R² present good levels of predictive effectiveness, especially across short horizons. The tool presents a useful approach for investors to obtain data-based insights, and there is future potential to combine sentiment analysis with ensemble methods.

|  |  |  |
| --- | --- | --- |
| **Sr.no** | **Title** | **Page No.** |
| 1 | Introduction | 5 |
| 2 | Objectives | 5 |
| 3 | Technologies Used | 5 |
| 4 | System Architecture and Methodology | 5 |
|  | └─ Architecture Overview | 6 |
|  | └─ Workflow | 6 |
| 5 | Features | 6 |
| 6 | Evaluation Metrics | 7 |
| 7 | Sample Results | 7-8 |
| 8 | Applications | 8 |
| 9 | Limitations | 8-9 |
| 10 | Future Enhancements | 9 |
| 11 | Conclusions | 9 |
| 12 | References | 9 |
| 13 | Screenshots | 10-11 |

**INDEX**

**1. Introduction**

Stock price forecasting is a challenging problem due to market volatility, non-linearity, and the influence of various global factors. This project aims to predict the short-term future closing prices of stocks using the XGBoost regression algorithm. The interface is built with Streamlit, making it interactive and accessible to users without technical expertise.

**2. Objectives**

* To provide a simple web-based interface for users to predict stock prices.
* To demonstrate the application of machine learning models in finance.
* To offer short-term price predictions using historical data.
* To visualize stock trends interactively using Plotly charts.

**3. Technologies Used**

* Python 3.x: Core programming language.
* Streamlit: For building the user interface.
* YFinance: For retrieving historical and live stock market data.
* Pandas & NumPy: For data manipulation.
* Plotly: For creating interactive and real-time plots.
* XGBoost: Machine learning model for regression.
* Scikit-learn: Used for data preprocessing and evaluation.

**4. System Architecture and Methodology**

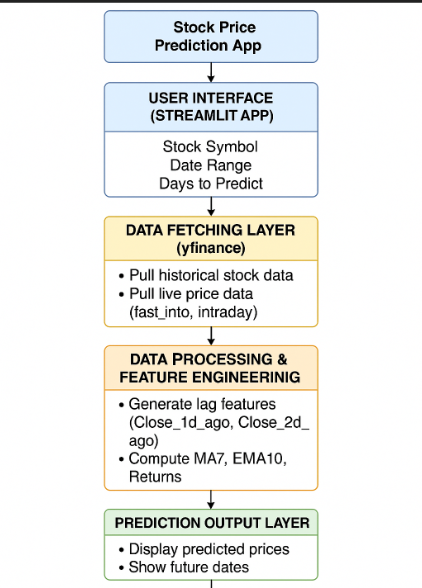
Architecture Overview

The system consists of three main components:

* Frontend: Streamlit-based user interface.
* Backend: Machine learning logic powered by XGBoost.
* Data Source: Stock data fetched from Yahoo Finance via yfinance.

Workflow

1. User provides stock symbol and date range.
2. Data is fetched from Yahoo Finance.
3. Features are engineered (moving averages, lag features, etc.).
4. The dataset is split into training and testing sets.
5. XGBoost regressor is trained on the training data.
6. Model predictions are generated and displayed.
7. Users can view historical, predicted, and live stock charts.



**5. Features**

* Dynamic stock symbol input.
* Customizable date range selection.
* Configurable prediction window (1 to 30 days).
* Tabular and graphical display of historical data.
* Real-time prediction using XGBoost.
* Live stock price and intraday chart updates.
* Data preprocessing and feature engineering included.

**6. Evaluation Metrics**

1. **Mean Absolute Error (MAE)**

MAE=n1​i=1∑n​∣yi​−y^​i​∣

* **What it measures**: The average of the absolute differences between actual and predicted values.
* **Interpretation**: Lower MAE indicates better model performance.
* **Advantage**: It treats all errors equally and is robust to outliers.
* **Example**: If the model predicts ₹2950 but the actual value is ₹3000, the absolute error is ₹50.

**2. Root Mean Squared Error (RMSE)**

**Formula:**

**RMSE=√n1​i=1∑n​(yi​−y^​i​)2**

* **What it measures**: The square root of the average of squared differences between actual and predicted values.
* **Interpretation**: Like MAE but penalizes larger errors more heavily.
* **Advantage**: Useful when large errors are particularly undesirable (e.g., in financial predictions).
* **Disadvantage**: Sensitive to outliers.
* **Example**: A large error like ₹200 will have a disproportionately higher impact due to squaring.

**3. R-squared (R² Score or Coefficient of Determination)**

**Formula:**

**R2=1−SStot​SSres​​=1−∑(yi​−yˉ​)2∑(yi​−y^​i​)2​**

**What it measures**: The proportion of the variance in the dependent variable that is predictable from the independent variables.

* **Range**: From 0 to 1 (can also be negative if the model performs worse than a horizontal line).
* **Interpretation**:
  + 1 = perfect prediction
  + 0 = model explains none of the variance
* **Advantage**: Gives a quick sense of model's overall fit.
* **Example**: An R² of 0.85 means 85% of the variability in the actual prices is explained by the model.

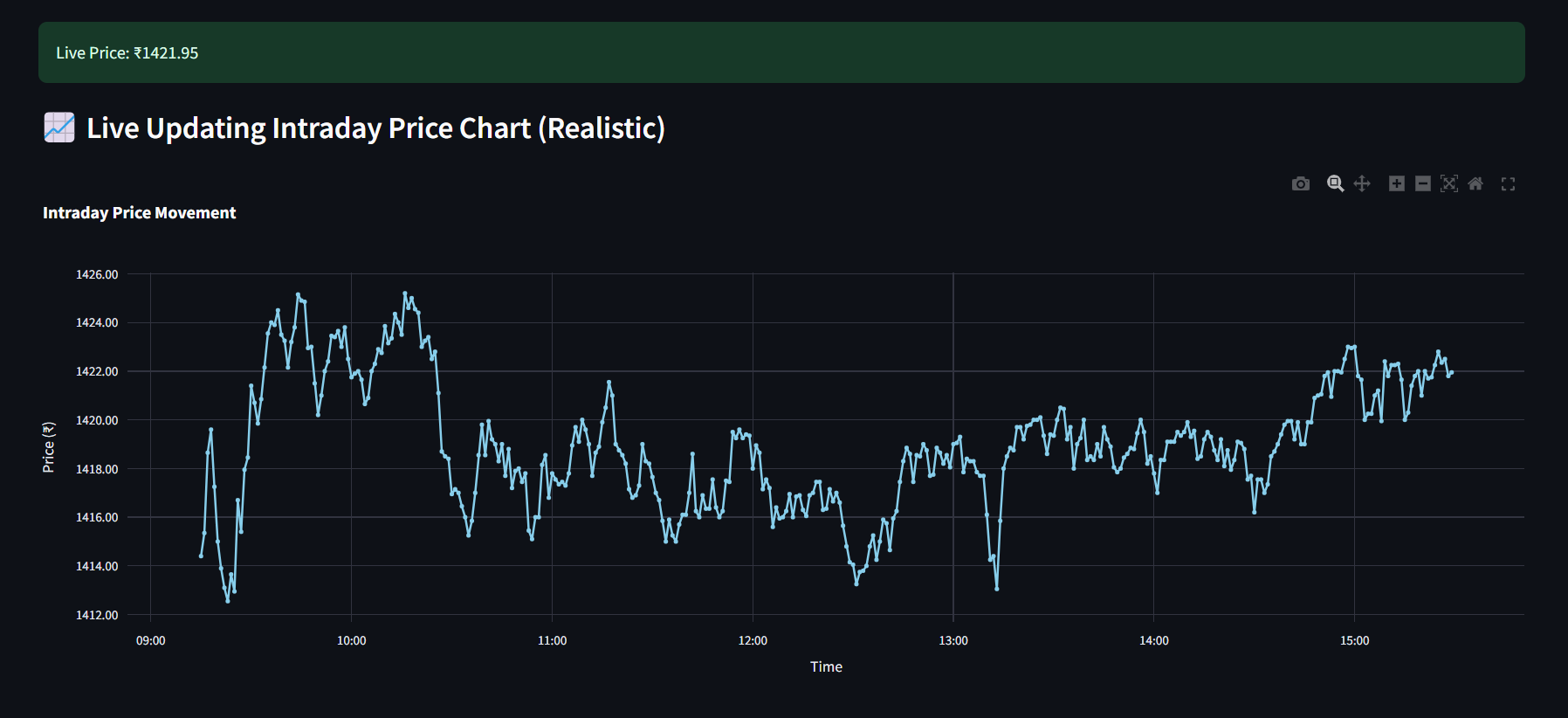
**7. Sample Results**

Example Prediction for "RELIANCE.BO" with a 5-day prediction window:

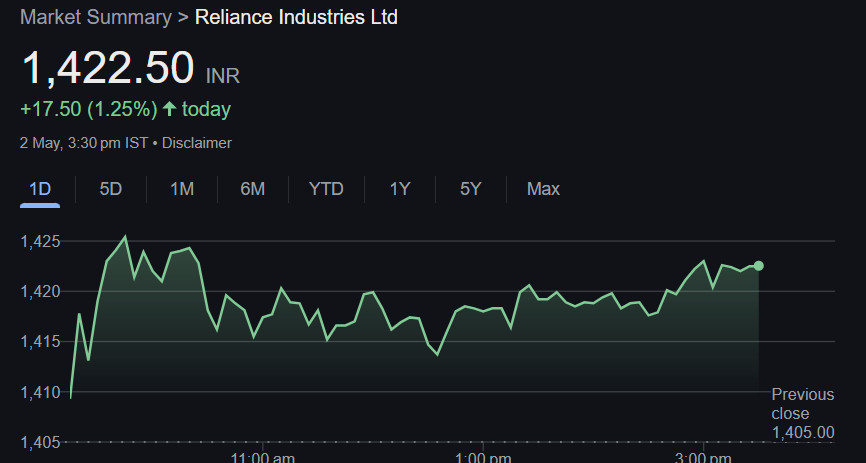
|  |  |
| --- | --- |
| Date | Predicted Close Price (INR) |
| 2025-05-03 | ₹2870.45 |
| 2025-05-04 | ₹2885.23 |
| 2025-05-05 | ₹2900.12 |
| 2025-05-06 | ₹2915.34 |
| 2025-05-07 | ₹2921.88 |

**Accuracy Of APP**

**Our APP**

****

**Google Stocks**

****

**8. Applications**

* Educational use in demonstrating time-series ML.
* Financial analysts and retail investors for trend prediction.
* Integration in financial dashboards and apps.
* Prototyping algorithmic trading tools.

**9. Limitations**

* Limited to short-term predictions.
* No consideration of market news, sentiment, or events.
* Data source limitations may cause occasional inaccuracies.
* Overfitting possible without regular tuning.

**10. Future Enhancements**

* Include additional models (LSTM, ARIMA, etc.) for comparison.
* Add support for sentiment analysis from news or tweets.
* Incorporate risk assessment and confidence intervals.
* Enable cloud deployment with auto-refresh features.

**11. Conclusions**

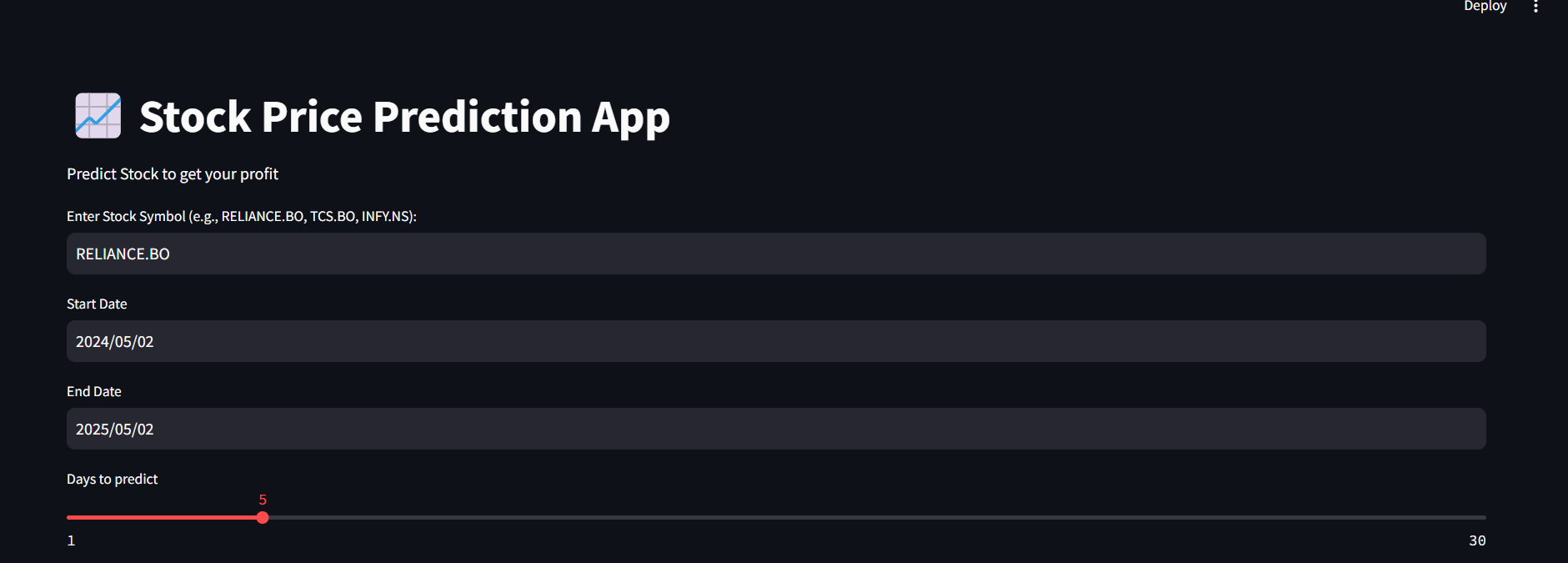
This project successfully combines financial data, machine learning, and interactive visualizations. It offers a practical entry point for exploring stock market forecasting using real-time data and stream-based interfaces. Although limited in scope, it lays a solid foundation for more complex predictive systems.

**12. References**

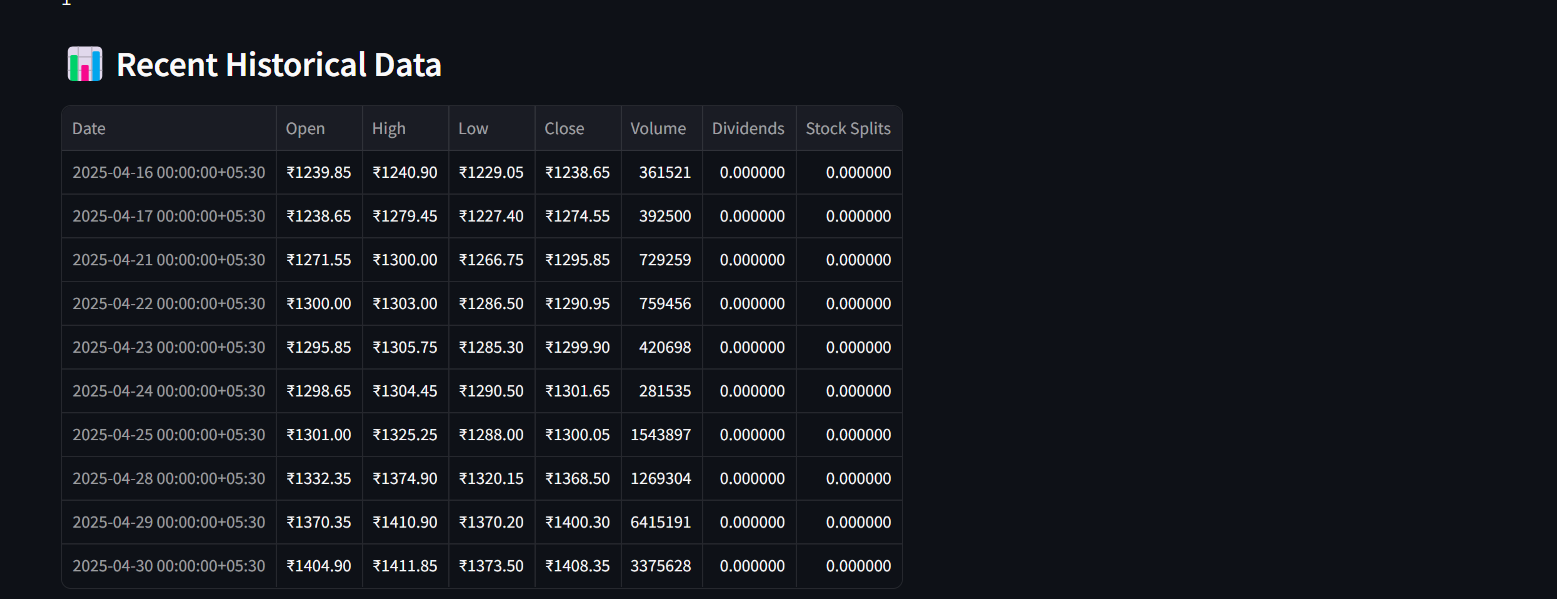
* https://pypi.org/project/yfinance/
* https://xgboost.readthedocs.io/
* https://docs.streamlit.io/
* https://scikit-learn.org/
* <https://plotly.com/python/>

13. Screenshots of the project

Dashboard



Historical Data



Historical Data Graph



Prediction Stock Price



Live Stock Monitoring

