PREDICTING STOCK PRICES USING KOLMOGOROV-SMIRNOV TEST AND DISTRIBUTIONAL FITTING WITH WEIGHTED DISTRIBUTION AVERAGING

Yash Patel
Bachelor of technology in information technology
National Institute Of Technology, Jalandhar, India
Email : therealyashpatel@gmail.com
GitHub : https://github.com/royalmaddy07/MyProject_1/blob/main/README.md

Abstract

This project implements a statistical prediction model that predicts daily stock closing prices using historical data and statistical modeling. The algorithm analyzes the last 20 closing prices to identify the most appropriate probability distribution by performing a Kolmogorov-Smirnov (KS) test on four distributions: normal, lognormal, exponential, and Student's t-distribution. Based on the KS test results, the model assigns weights to the distributions with the test statistic less than the critical value for a 20 data points one sample KS test and generates a predicted value using a weighted average of the distribution's median. The algorithm continuously updates its prediction by incorporating new closing prices into the dataset, aiming to provide accurate forecasts for future stock prices. The output is saved to a CSV file for further analysis. This model focuses on utilizing statistical techniques to improve the accuracy of stock price predictions.

Statistical Assumptions Taken

1. The model assumes that the stock prices (or their transformations) follow one of the tested probability distributions: Normal, Lognormal, Exponential, or Student's t-distribution.
2. The model assumes that the data points are independent and identically distributed.
3. The weights assigned to distributions are inversely proportional to their KS statistics, assuming lower KS statistics correspond to better fits.
4. The last 20 days closing prices provide enough information for predicting the closing price of the 21st day.

1.Introduction

The model leverages four commonly used probability distributions—normal, lognormal, exponential, and Student's t-distribution—to model the stock price data. By applying the Kolmogorov-Smirnov (KS) test, the system evaluates the fit of these distributions to the historical data and selects the most appropriate distributions for future predictions. A weighted average of the predicted values from the distributions with the test statistic less than the critical value for a 20 data points one sample KS test is then used to generate the final prediction.
This project emphasizes prediction accuracy by continuously updating the model with the latest stock prices, aiming to enhance the forecasting process. The results of the predictions are saved for further analysis and evaluation, providing insights into the potential of statistical methods for improving stock price forecasting.

2. Data Collection and Preprocessing

Sourcing Historical Stock Closing Prices:
In this project, we focus on **historical stock closing prices**. For this project, the stock closing prices are collected in a **CSV file format**. These closing prices are collected from publicly available financial data source - **Yahoo Finance**. These prices are considered to be reliable indicators of the market's daily performance.
To generate an API call to Yahoo Finance, use the first code of the Jupyter notebook in the GitHub repository via the link provided at the header. Note that this program uses the yfinance library.
Replace the stock symbol with the symbol of the stock for which you want to get the data and specify the time from which date to which you want the data to be. A file with the name raw_data.csv will be saved.
To parse this file, run the second python code in the Jupyter notebook in order to process the file for use in our algorithm. Note that this code uses pandas library.

A file with the name historical_data.csv will be saved.
NOTE : Delete the headers of this file and add a single line header "Time,Closing price" for our algorithm to process it.
Each row contains the date and the corresponding closing price for a particular stock. This data is used as the input for the statistical prediction model, which aims to forecast future prices based on past trends.
The model uses a **rolling window approach**, where the last 20 days closing prices are used as the input to predict the price for the next day. As new data points become available, the window slides forward, incorporating the most recent data and updating the prediction.
The data is stored in a **vector container**, where each element represents a stock closing price. This vector is updated as new data is fetched, ensuring that the model always uses the most recent information for predictions.

3. Probability Distributions in Stock Price Modeling

In this project, four widely used probability distributions—**Normal**, **Lognormal**, **Exponential**, and **Student's t-distribution**—are employed to model the historical stock price data and make predictions about future stock prices.

Why These Distributions?

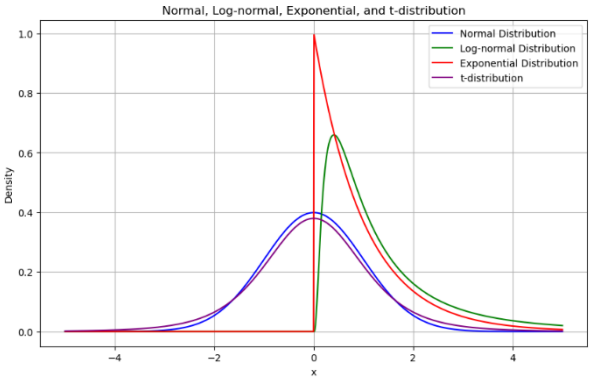
Each of these distributions is chosen based on its ability to model specific characteristics of stock price behavior:

- The **normal distribution** is useful when stock returns exhibit symmetry and relatively low volatility.
- The **lognormal distribution** captures the skewness in stock price data and is more appropriate for modeling actual stock prices, which cannot go below zero.
- The **exponential distribution** is employed to model the occurrence of significant events or changes in stock prices over time.
- The **Student's t-distribution** is included to account for extreme movements and outliers that are often observed in financial markets, particularly during periods of high volatility.

By using these different distributions, the model can capture a wide range of possible behaviours in the stock price data, improving the accuracy and reliability of the predictions.

4. Choosing the Best-Fitting Distribution

To determine which distribution best fits the historical stock price data, the model uses the **Kolmogorov-Smirnov (KS) test**. This test compares the **empirical cumulative distribution function (eCDF)** of the historical data with the **theoretical cumulative distribution function (CDF)** of each of the four distributions. The distributions with the KS test statistic less than the critical value(indicating the best fit) are chosen for generating the prediction. This process ensures that the model uses the most appropriate distributions for forecasting future stock prices.



5. Model Overview

The model uses a **rolling window** of the previous 20 days' closing prices, which is updated every day with the latest closing price. The idea is to capture the most recent trends and patterns in the stock's price movements to predict its future value. The process can be broken down into the following key steps:

- **Data Collection**: The model fetches historical daily stock closing prices from a reliable source.
- **Distribution Fitting**: The model applies four different probability distributions (Normal, Lognormal, Exponential, and Student's t-distribution) to the stock price data.
- **Kolmogorov-Smirnov (KS) Test**: The KS test is used to assess the goodness of fit of each distribution to the historical data.
- **Prediction Generation**: A weighted average of the medians of the satisfying distributions, with the respective parameters taken from the current 20 points data set, is computed to generate the final forecast for the next day's stock price.

i. Prediction Process

The prediction process begins by selecting the most appropriate probability distribution for modeling the stock price data. This is achieved through the Kolmogorov-Smirnov (KS) test, which compares the empirical distribution of the historical data with the theoretical distributions.
Once the best-fitting distribution is identified, The model then computes the prediction from the

satisfying distributions and takes a **weighted average** of the median of those distributions to produce a final forecast. The weights assigned to satisfying distributions are based on the KS test results, with the best-fitting distribution receiving the highest weight. The use of the **median** of the satisfying distributions is for **representing central tendency.** The median represents the **central value** of a dataset and ensures that 50% of the values are below and 50% are above it. Using the median ensures that the final prediction lies within a range that is **intuitively reasonable** based on the data.

ii. Rolling Window Mechanism

To ensure that the model remains relevant and accurate as new data becomes available, the model employs a **rolling window** mechanism. The window consists of the most recent 20 days of stock closing prices, and as new data is added, the oldest data point is discarded. This ensures that the model is always using the most up-to-date information to make predictions.

Each time a new day's data is added, the model updates its predictions by recalculating the distributions, running the KS test, and adjusting the weighted average. This continuous update process enables the model to adapt to changes in market conditions and make more accurate predictions over time.

6.Code Implementation Details

The main algorithm is saved in the repository by the name **main.cpp**. The algorithm is written in **C++** to leverage the computing power and faster implementation of the language for large datasets. The algorithm is written in **Functional Paradigm** for code readability and flexibility.

NOTE that the algorithm uses the Boost.Math library for statistical computations. Therefore to run it on your system, you need to download this C++ library on your system via a package manager like MSYS2 or from the web.

Key Aspects of the C++ Program:

- **File I/O Operations**: Handling of CSV files to read historical data and save predictions.
- **Rolling Window Mechanism**: A technique for updating the dataset dynamically as new data arrives.
- **Statistical Distributions**: Application of normal, lognormal, exponential, and Student's t-distributions using the Boost library.
- **Kolmogorov-Smirnov Test**: Used to compare the fit of each distribution to the historical data.
- **Prediction and Weighting**: Using a weighted average of medians based on the KS test results to generate a final stock price prediction.
- **Data Output**: Saving the predictions to a new CSV file for further evaluation.

To run the algorithm use the historical_data.csv file generated earlier. The program main.cpp is also in the GitHub repository mentioned at the header.
The algorithm after a successful run will generate a predicted_data.csv file.

Save this file and the historical_data.csv file for performance measure, model validation and further analyses.

7. Model Evaluation

i. Performance Metrics

To evaluate the performance of the model, we focus on the following metrics:

- **Mean Absolute Error (MAE):** MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It gives an idea of how far off the predictions are from the actual values on average.
- **R-Squared (R^2)**: R^2 indicates how well the predicted values fit the actual data. A higher R^2 value suggests that the model explains a large portion of the variance in the data.

ii. Visualization

The model's performance is also evaluated visually. Predicted stock prices are plotted against actual stock prices over time to observe how well the model tracks the actual data. A good model should show a strong correlation between the predicted and actual prices.
NOTE : If the stock price data does not follow any of the 4 distributions , the model will produce an unexpected or sudden low value or a dip for the predicted stock price.

8. Results

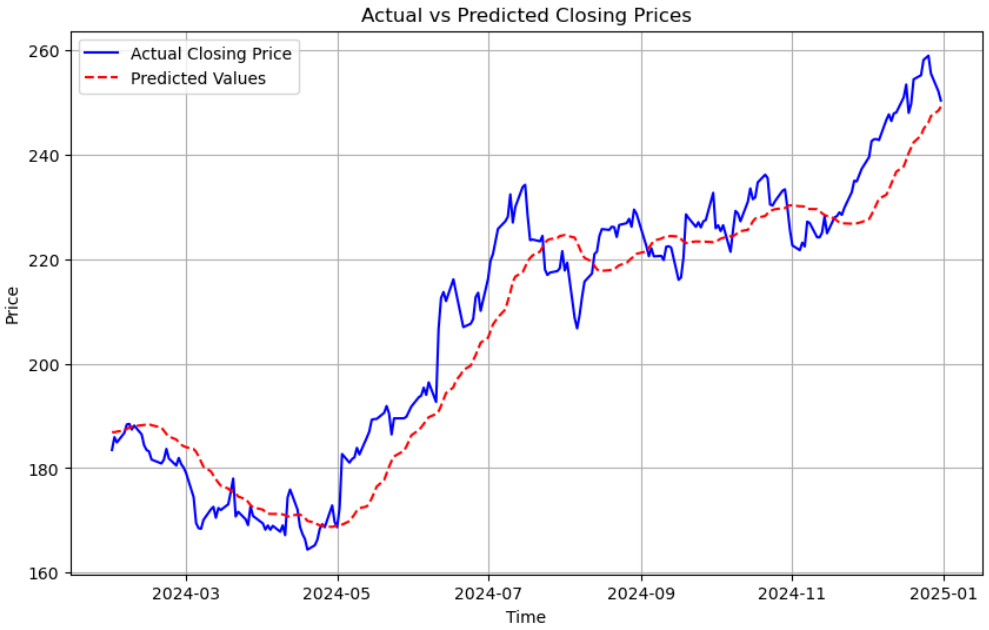
To plot the data points in the historical_data.csv file and predicted_data.csv file to compare them and generate the performance metrics run the third python code in the Jupyter notebook. This code uses the **matplotlib** and **pandas** libraries to generate plots and MAE and R^2 values.

If your selected stock contains an **extreme dip,** it suggests a data point for which the past 20 days

closing price do not follow any of the four defined probability distributions according to the algorithm. **In this case, if you want to calculate the performance metrics after removing the extreme dips, you have to run the fourth code of the Jupyter notebook by adjusting the threshold value in the code, above which the predicted price values should be, according to the plot obtained in order to remove those points for MAE and R-square calculation.**

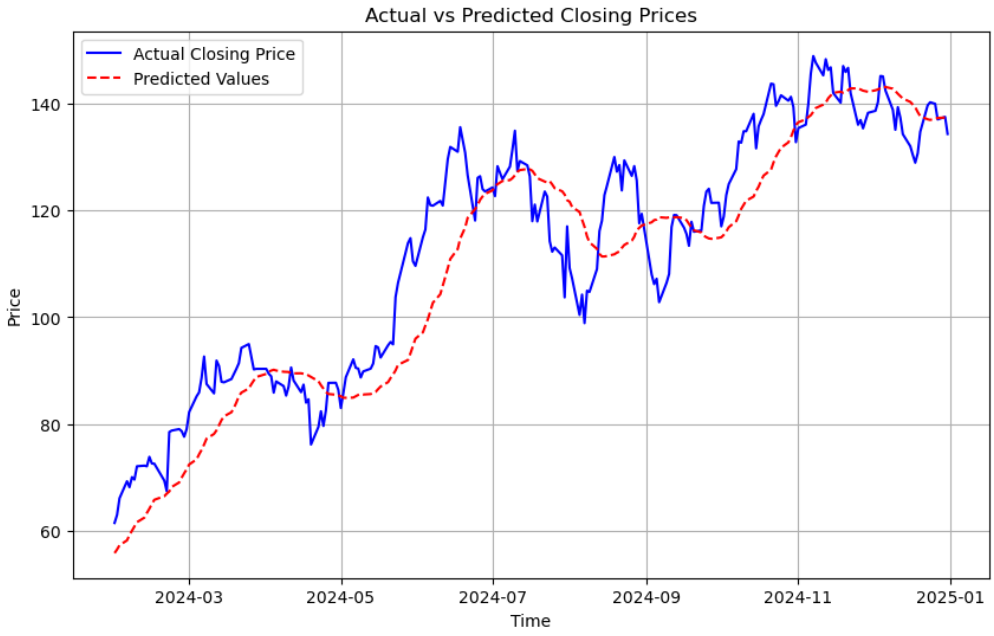
SOME PLOTS AND PERFORMANCE METRICS AFTER GENERATING PREDICTED VALUES USING THE MODEL ON STOCKS OF DIFFERENT COMPANIES. (Previous 1 year timeframe)

APPLE STOCK



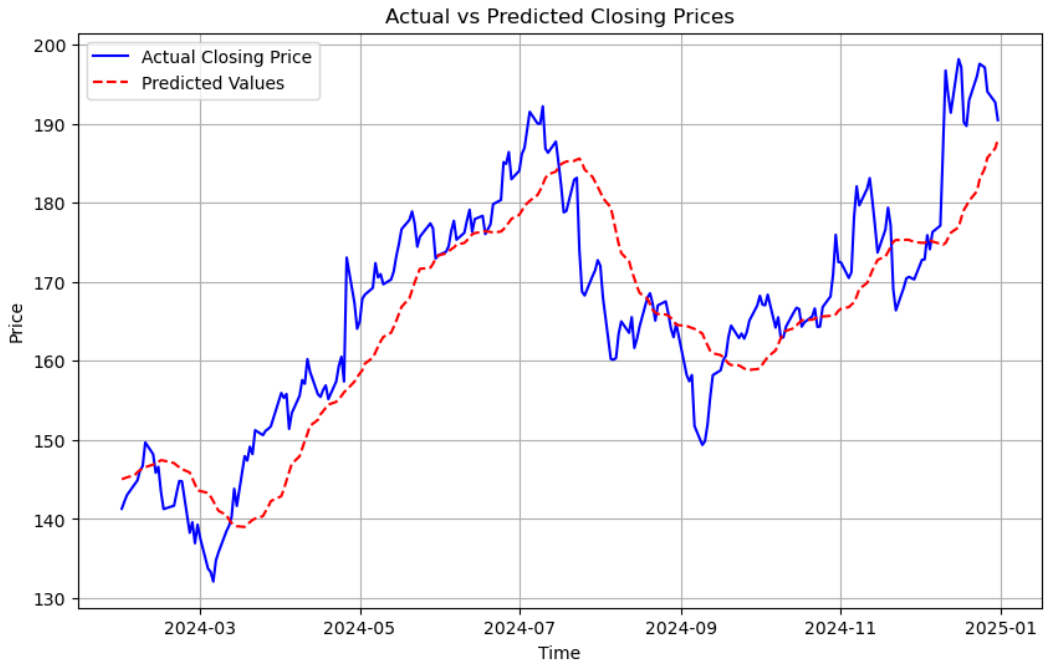
(%ERROR: 3.16%)MAE: 6.626533635632745 R^2: 0.9014714997176454

NVIDIA STOCK



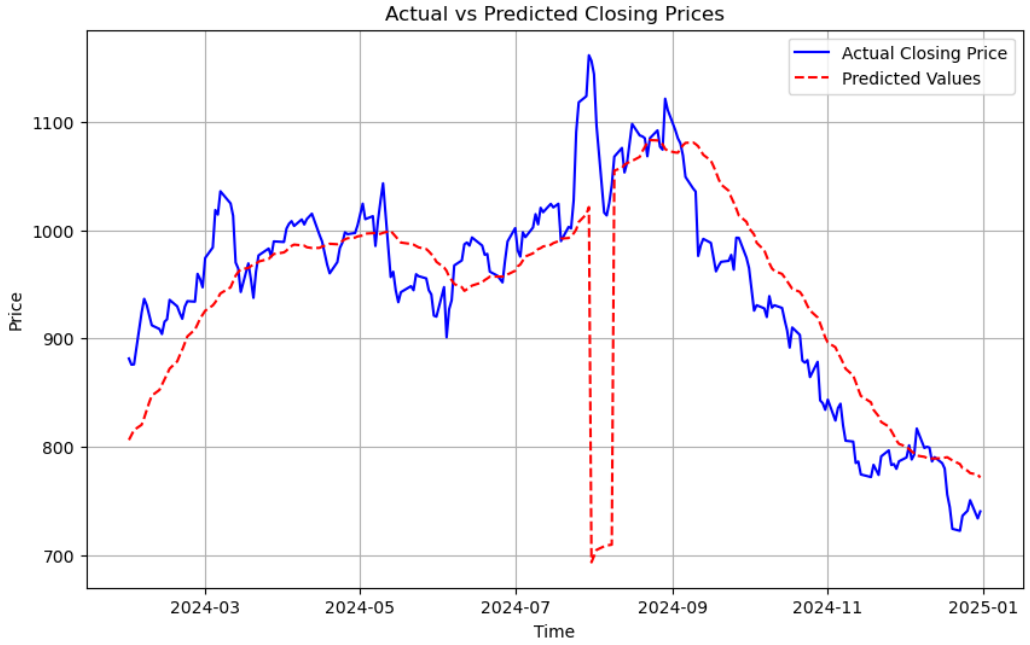
(%ERROR: 7.06%)MAE: 7.607704354779475 R^2: 0.8370734349216443

GOOGLE STOCK



(%ERROR: 3.62%)MAE: 6.0727341234930625 R^2: 0.7391846163449787

TATA MOTORS STOCK

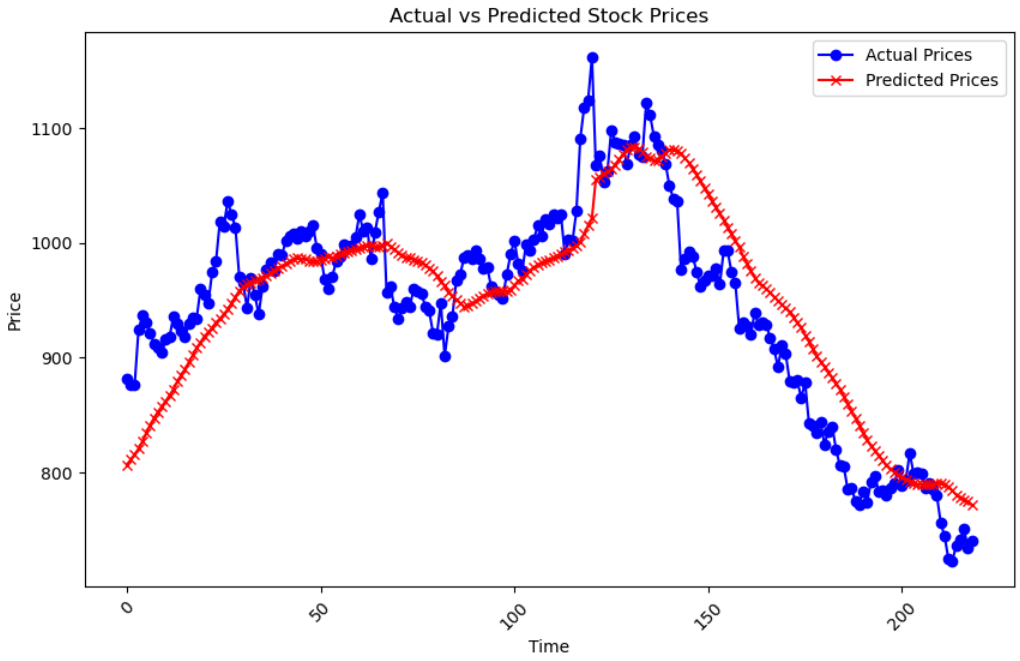


(%ERROR: 4.91%)MAE: 47.09835790799572 R^2: 0.3127106656280183

AS WE CAN SEE AN EXTREME DIP HAS OCCURRED, WE CAN REMOVE THE EXTREME DIP POINTS BY DEFINING A THRESHOLD POINT IN THE CODE BELOW WHICH, IF THERE IS A PREDICTED VALUE THE CODE WILL REMOVE IT AND NOT CONSIDER IT FOR METRIC EVALUATION. HERE, WE CAN TAKE THE THRESHOLD VALUE TO BE 720 OR 725.

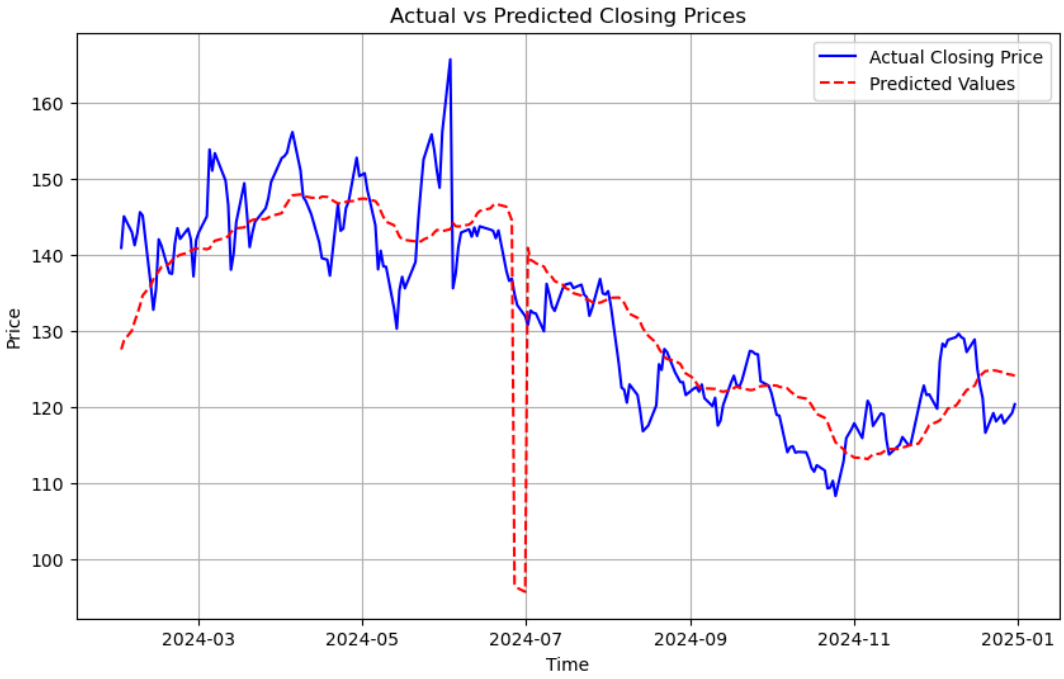
THE PLOT AND METRICS AFTER REMOVING THE EXTREME DIP POINTS ->

TATA MOTORS STOCK



(%ERROR: 3.98%)Mean Absolute Error (MAE): 36.898336771368434 R^2: 0.7662456460736339

UNION BANK OF INDIA STOCK

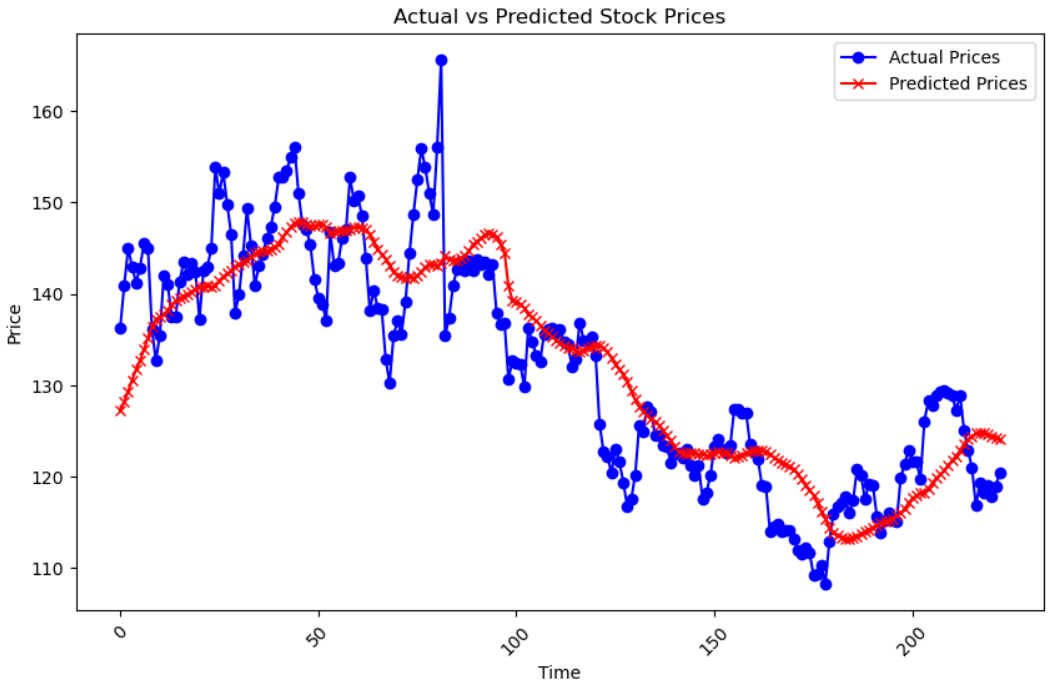


(%ERROR: 4.04%)MAE: 5.371127386309649 R^2: 0.6300413098256475

AGAIN WE CAN REMOVE THE EXTREME DIP POINTS BY DEFINING A THRESHOLD POINT IN THE CODE BELOW WHICH IF THERE IS A PREDICTED VALUE BELOW THE THRESHOLD THE CODE WILL REMOVE IT AND NOT CONSIDER IT FOR METRIC EVALUATION. HERE. WE CAN TAKE THE THRESHOLD = 100.

THE PLOT AND METRICS AFTER REMOVING EXTREME DIP POINTS ->

UNION BANK OF INDIA STOCK



(%ERROR: 3.72%)Mean Absolute Error (MAE): 4.894725698702003 R-squared: 0.7547709343889906

INFERENCE : The %error in the price prediction on average is equal to 5% and the R^2 values are between 0.7 and 1 suggesting that the statistical prediction model is performing well in explaining the variance in the stock prices over the previous 20 days.

10. Future Work and Improvements

To enhance the robustness and accuracy of the statistical prediction model, several improvements and future directions can be explored:

i. Expanding the Range of Distributions

Incorporating additional probability distributions, such as Weibull, Pareto, or Gamma, could improve the model's flexibility in fitting diverse stock price patterns. This would reduce the likelihood of generating unrealistic predictions when data does not conform to the current set of distributions.

ii. Handling Non-Stationarity

Developing methods to address the non-stationary nature of stock prices, such as using rolling windows with adaptive parameters or applying time-series decomposition techniques, can make the model more resilient to changing market conditions.

iii. Integration of External Factors

Including macroeconomic indicators, news sentiment analysis, and sector-specific data could provide a more comprehensive view of the market. Machine learning models could be employed to incorporate these variables alongside statistical approaches.

iv. Hybrid Approaches

Combining the statistical prediction model with machine learning algorithms, such as regression trees, support vector machines, or neural networks, could enhance its predictive power and adaptability to complex data patterns.

11. Conclusion

This project lays a solid foundation for statistical stock price prediction and provides a stepping stone for more advanced research in financial modeling. By addressing its limitations and embracing future improvements, the model can evolve into a robust tool for understanding and navigating the complexities of financial markets.