Stock Price Prediction Using Time Series Analysis and RNNs

1 Introduction

This project focuses on predicting stock prices using a combination of classical time series analysis techniques and deep learning models. The study involves statistical methods such as ARIMA and machine learning techniques like RNNs and LSTMs. The project was mentored by Ishan Gupta and Sanya.

2 Learning Journey

2.1 Week 1: Fundamentals of Probability and Statistics

In the first week, I focused on building a strong foundation in probability and statistics, which are essential for understanding machine learning models. Key topics covered included:

- Probability distributions: Normal, Poisson, Binomial, Exponential distributions
- Measures of central tendency: Mean, median, mode
- Measures of dispersion: Variance, standard deviation, covariance
- Correlation and its significance in data analysis
- Basics of inferential statistics: Confidence intervals, hypothesis testing
- Likelihood function and Maximum Likelihood Estimation (MLE)
- Ordinary Least Squares (OLS) regression and its assumptions

As a practical application, I completed an assignment on linear regression using housing price data, which involved data preprocessing, model training, and evaluation using metrics such as Mean Squared Error (MSE) and R-squared.

2.2 Week 2: Exploratory Data Analysis and Time Series Basics

The second week was dedicated to understanding the fundamentals of time series analysis and exploratory data analysis (EDA). Topics covered included:

- Visualizing time series data using line plots, histograms, and scatter plots
- Identifying trends, seasonality, and cyclic behavior in time series data
- Stationarity: Definition, importance, and ways to test (ADF test, KPSS test)
- Differencing techniques to achieve stationarity
- Components of time series: Trend, seasonal, residual, and cyclic variations
- Moving average and exponential smoothing techniques
- Feature engineering for time series: Lag features, rolling statistics

I also explored autocorrelation and partial autocorrelation functions (ACF and PACF) to understand dependencies in time series data.

2.3 Week 3: Time Series Models - AR, MA, ARIMA, SARIMA

In the third week, I learned about various time series models and their applications in forecasting. Topics included:

- Autoregressive (AR) model and its assumptions
- Moving Average (MA) model and smoothing techniques
- ARIMA (AutoRegressive Integrated Moving Average): Understanding parameters (p, d, q)
- Seasonal ARIMA (SARIMA): Handling seasonal variations
- Model selection criteria: AIC, BIC, and cross-validation
- Using ACF and PACF to determine optimal parameters for ARIMA
- Implementing and tuning ARIMA and SARIMA models

For the second assignment, I applied these models to forecast stock prices for TCS, Airtel, and NVIDIA, analyzing performance using Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE).

2.4 Week 4: Neural Networks, RNNs, and LSTMs

The fourth week introduced deep learning methods for time series forecasting, particularly focusing on Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks. Topics covered included:

- Basics of neural networks: Perceptron, activation functions, loss functions
- Introduction to RNNs: Vanishing gradient problem, BPTT (Backpropagation Through Time)
- LSTM architecture: Gates (input, forget, output), cell state, hidden state
- Bidirectional LSTMs and GRUs (Gated Recurrent Units)
- Hyperparameter tuning for LSTMs: Learning rate, batch size, number of layers
- Combining ARIMA with LSTM for hybrid modeling

For the third assignment, I implemented the following steps:

1. Perform stationarity checks using the ADF test.
2. If non-stationary, apply differencing until the series is stationary.
3. Use ACF and PACF plots to determine the ARIMA parameters (p, d, q).
4. Fit the ARIMA model to capture trends and seasonality.
5. Extract residuals from the ARIMA model for further modeling with LSTM.
6. Prepare residuals from ARIMA by creating lagged features and scaling the data.
7. Design and train an LSTM model to predict the residual component.
8. Combine ARIMA's predictions with LSTM's residual predictions to obtain final forecasts.
9. Train the final model for each stock and compare the predicted values against the test set.

3 Process Documentation

3.1 Preprocessing Steps

- Data cleaning and handling missing values - Transformations for stationarity - Feature engineering for time series

3.2 ARIMA Modeling

- Selection of p, d, q using ACF and PACF plots - Model diagnostics and parameter tuning

3.3 LSTM Implementation

- Data normalization and sequence generation - Model architecture and hyperparameter choices

3.4 Hybridization Process

- Combining ARIMA and LSTM outputs - Justification for hybrid approach

4 Evaluation Analysis

Stock	Model	MAPE	RMSE
TCS	ARIMA	0.90%	7.46
TCS	Hybrid	0.89%	7.86
Nifty 50	ARIMA	0.55%	34.37
Nifty 50	Hybrid	0.54%	37.00

Table 1: Comparison of ARIMA and Hybrid Models

5 Graphs

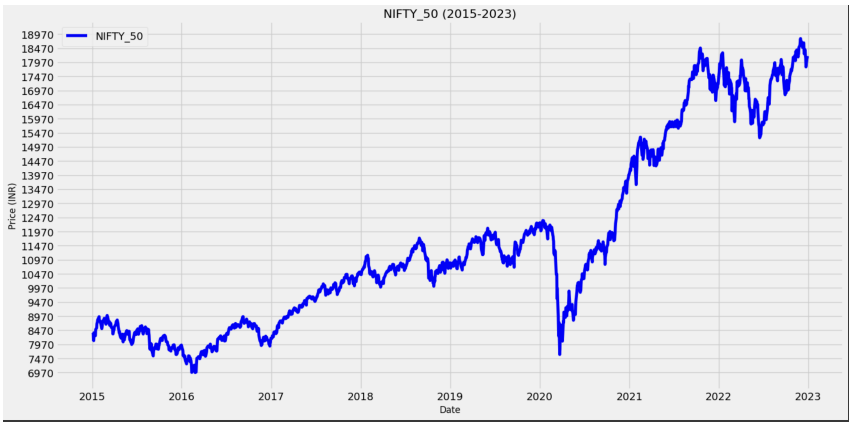

Figure 1: Time Series Plot of NIFITY 50

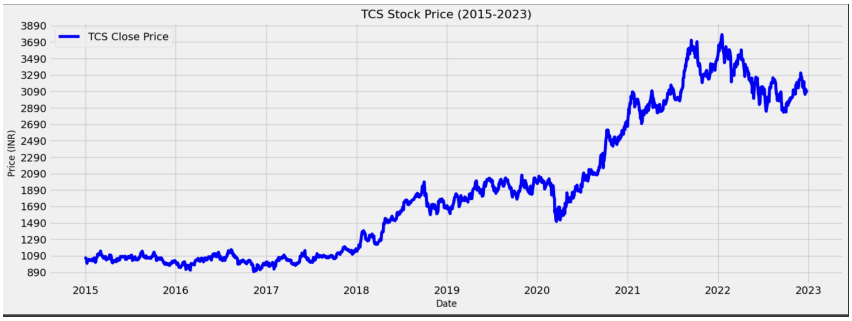

Figure 2: Time Series Plot of TCS

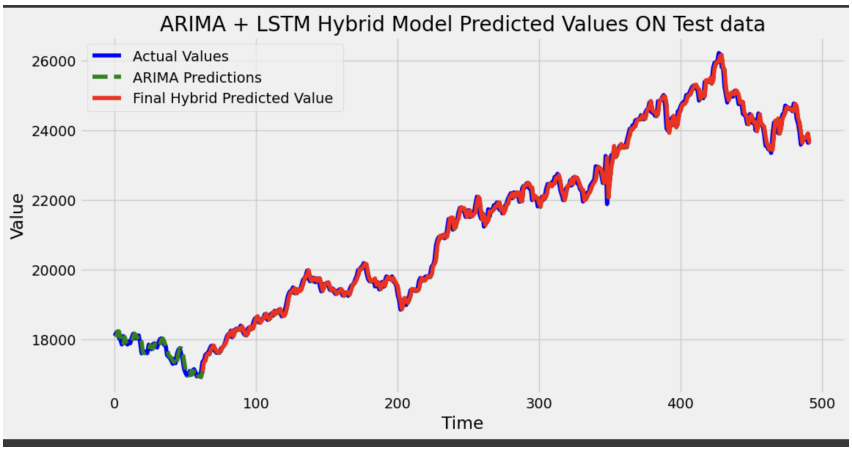

Figure 3: Predicted values from ARIMA and ARIMA + LSTM Hybrid Model for NIFTY 50

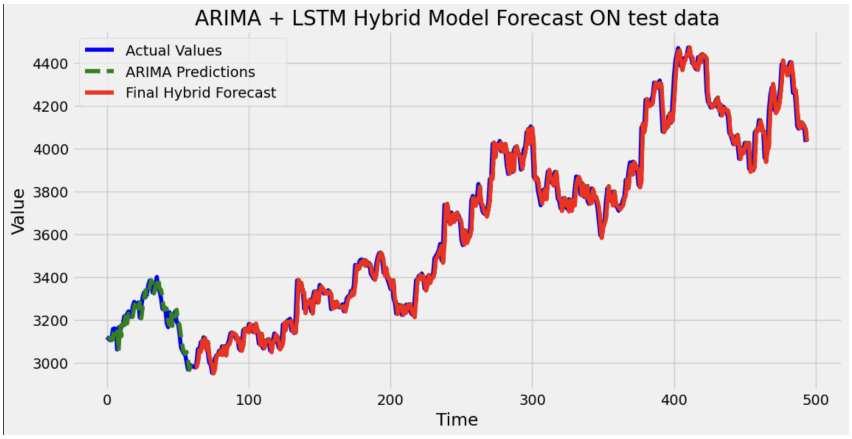

Figure 4: Predicted values from ARIMA and ARIMA + LSTM Hybrid Model for TCS

6 Why Use a Hybrid Model Instead of ARIMA?

Traditional time series forecasting models like ARIMA work well for linear patterns in data but struggle with capturing complex nonlinear dependencies. On the other hand, deep learning models like LSTMs are effective at modeling nonlinear relationships but may fail to capture trends and seasonality effectively when used alone. To overcome these limitations, we use a hybrid ARIMA-LSTM model that leverages the strengths of both approaches.

6.1 Limitations of ARIMA

- ARIMA assumes a linear relationship in the data, making it unsuitable for capturing complex patterns in stock prices.
- The model struggles with sudden market shocks and non-stationary data without extensive preprocessing.
- It requires careful tuning of parameters (p, d, q), which can be computationally expensive.
- ARIMA performs poorly when long-term dependencies exist in the data.

6.2 Advantages of the Hybrid Approach

- ARIMA effectively models the trend and seasonality in stock prices.
- LSTM captures complex nonlinear patterns and short-term fluctuations.

- By combining the residuals of ARIMA with LSTM predictions, we improve accuracy over using either model alone.
- The hybrid model ensures that the final prediction retains both interpretability (from ARIMA) and flexibility (from LSTM).

6.3 Performance Comparison

From the evaluation metrics in Table 1, we observe that the hybrid model achieves a lower Mean Absolute Percentage Error (MAPE) compared to ARIMA alone, demonstrating improved forecasting accuracy. However, the Root Mean Squared Error (RMSE) is slightly higher, indicating that LSTM adds some noise to predictions. Despite this, the hybrid approach is preferred because of its ability to generalize better in real-world scenarios where stock prices are highly volatile.

Thus, the hybrid model provides a more robust forecasting framework by integrating the best features of statistical and deep learning methods.

Acknowledgments

I would like to thank my mentors, Ishan Gupta and Sanya, for their valuable guidance throughout this project.