Understanding the

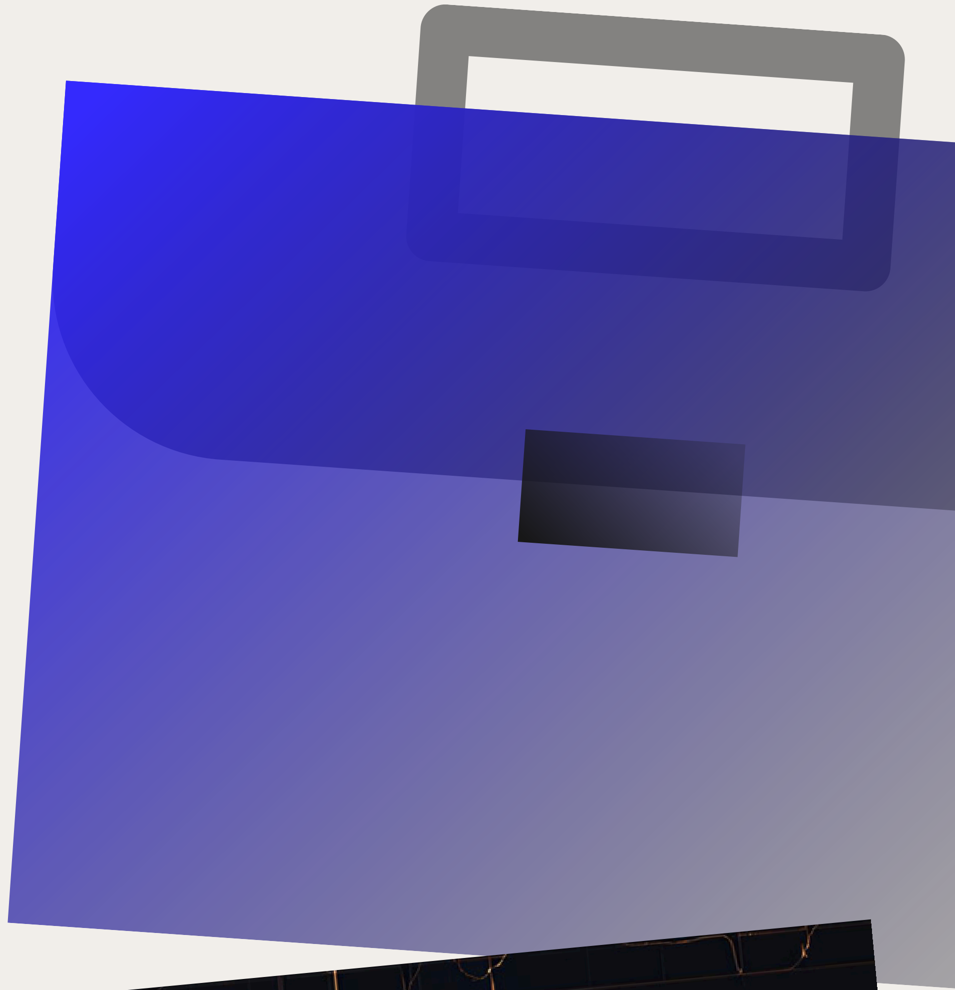
Stock Price Predictions

Presented by Samay Mehar

10 October ‘24

Importance of

Stock Price Prediction



- Stock price prediction is crucial for informed investment decisions.
- Accurate models can help maximize profit and reduce risk for investors.

Challenges

- High volatility and randomness in stock prices.
- Non-linear patterns in historical data are difficult to model.

Dataset Historical stock price data (Date, Opening, High, Low, Closing).

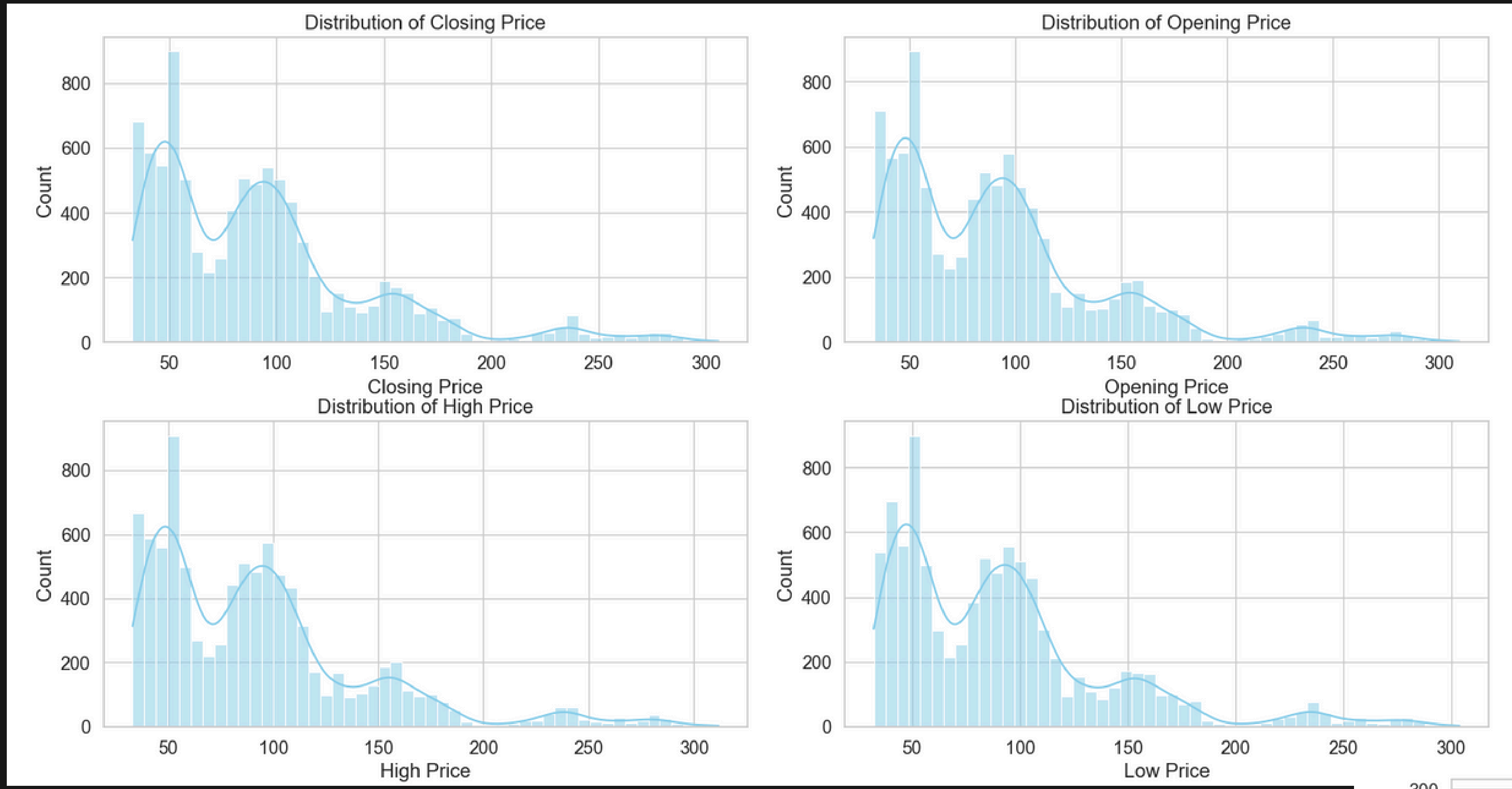
Trends Identified overall trends, seasonality, and anomalies in stock prices.

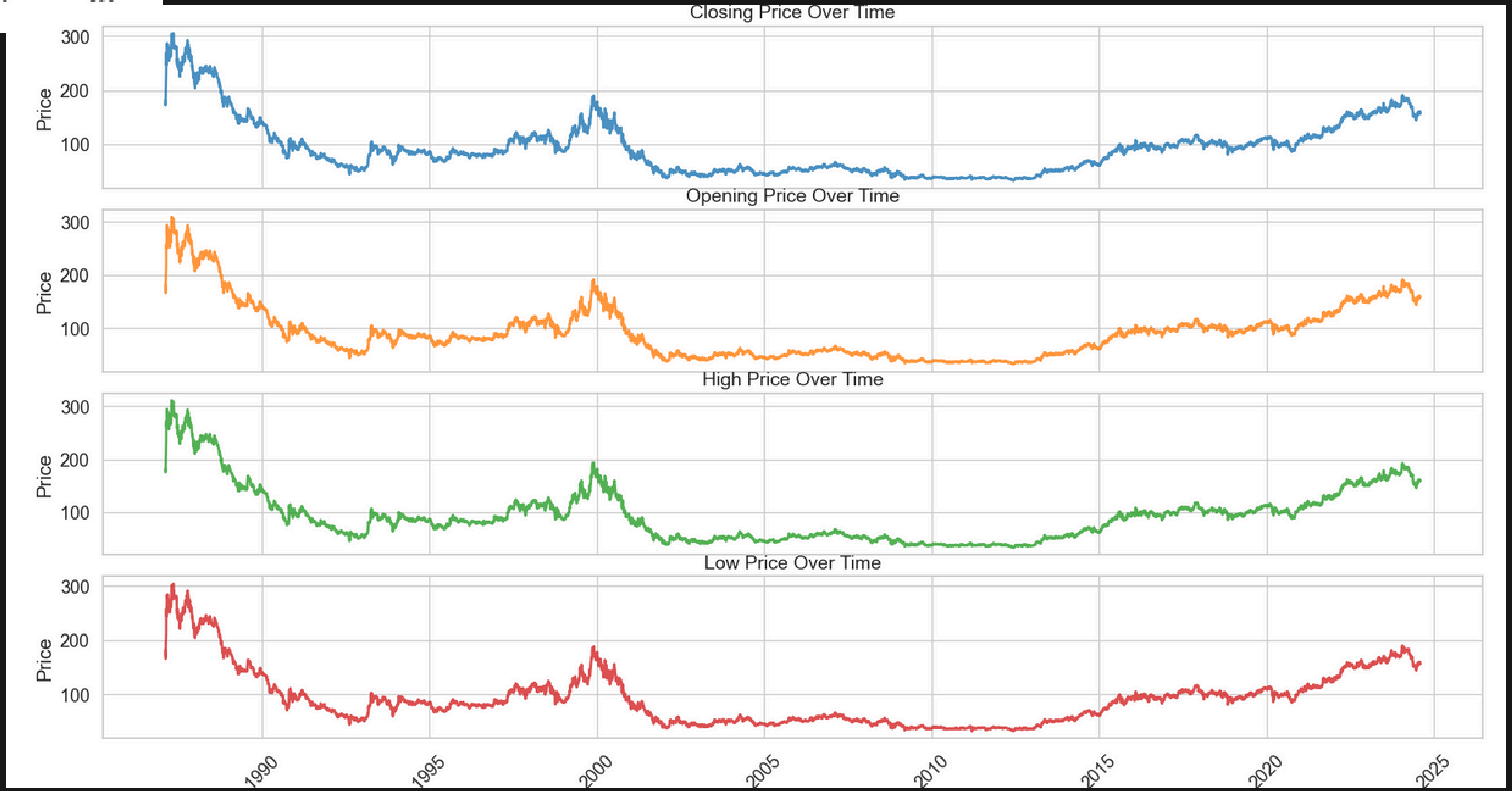
Key Observations

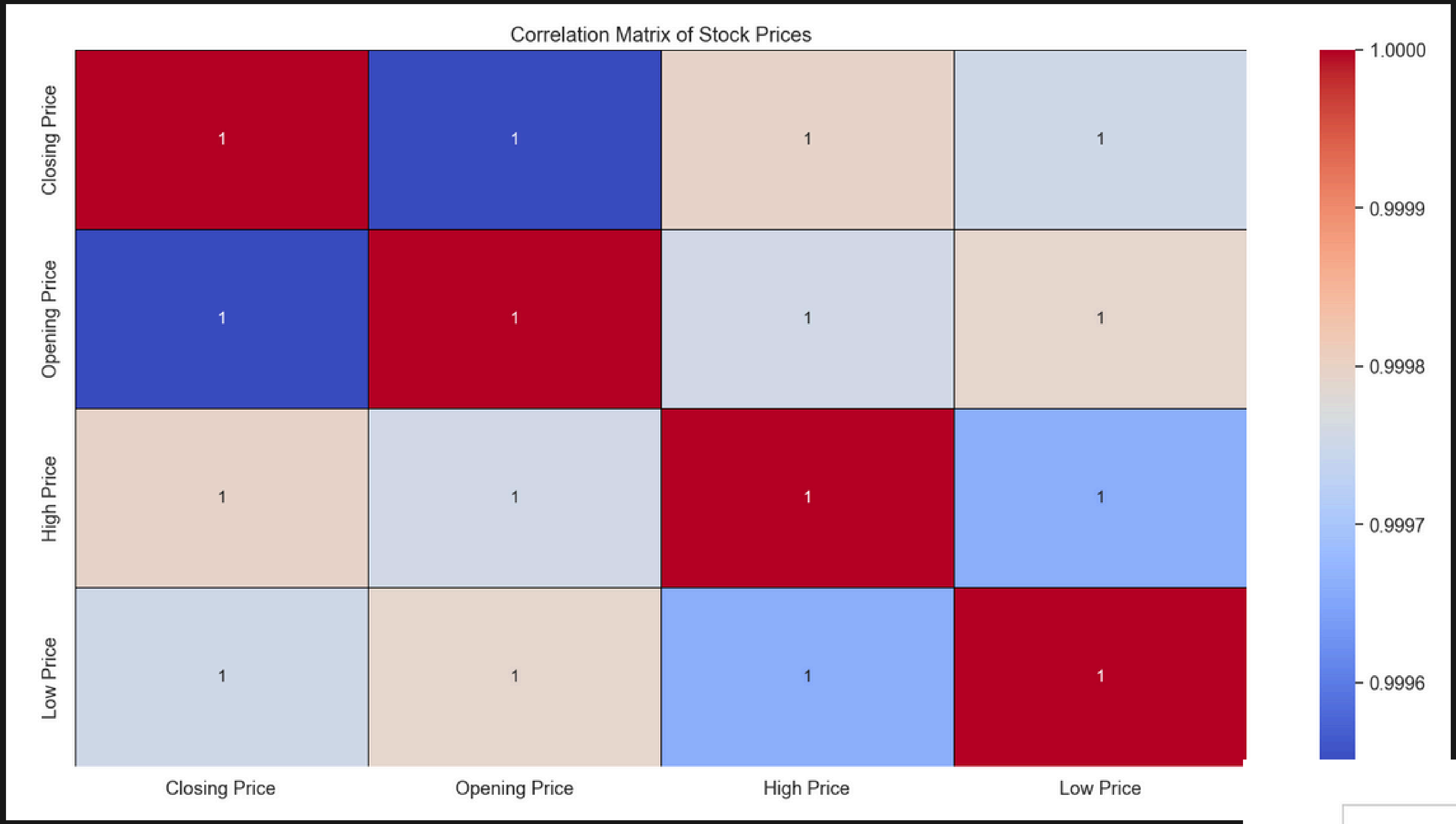
- Noticeable spikes in prices during specific periods.
- Potential for incorporating lagged data and feature engineering.

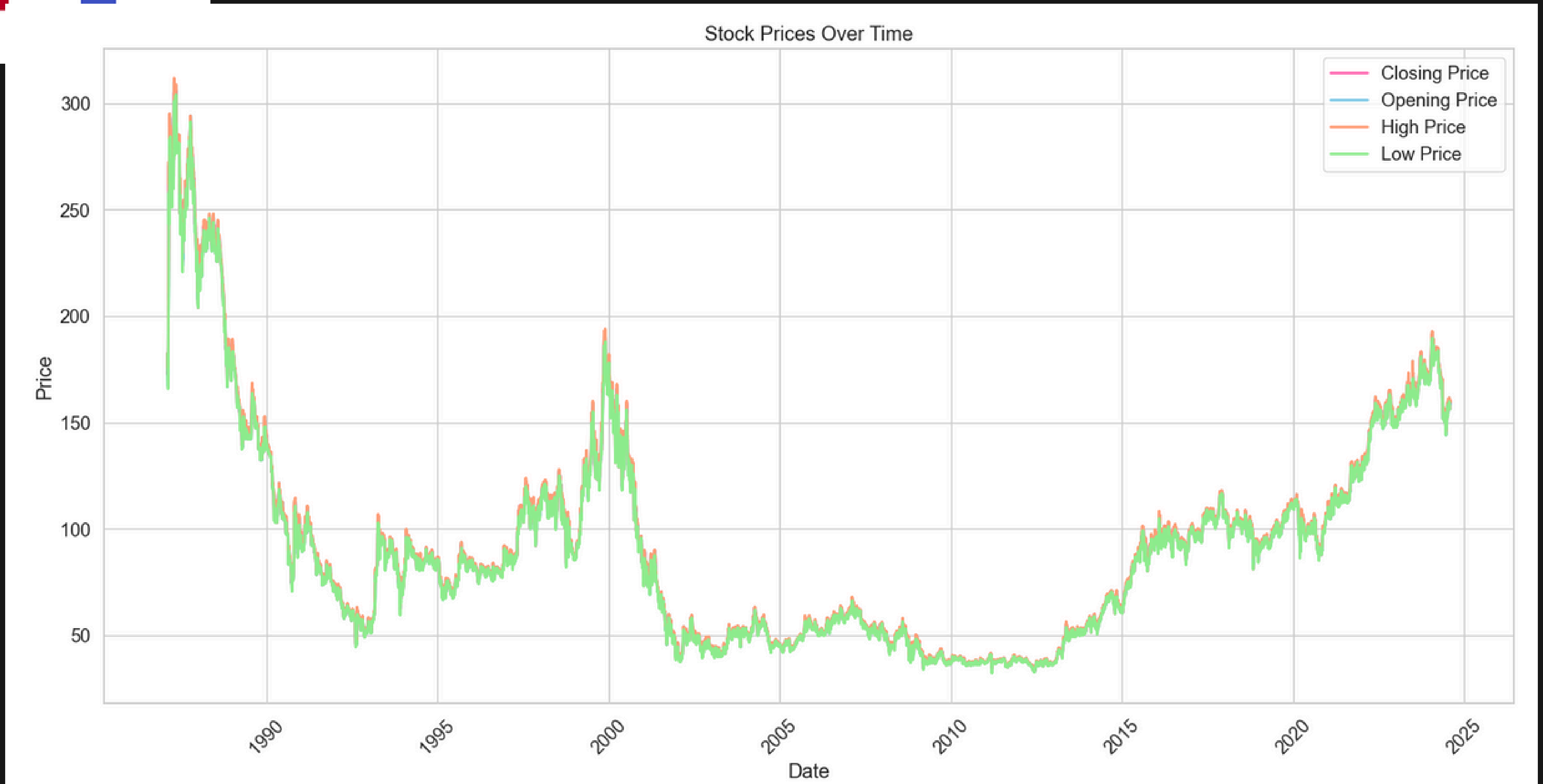
Data Understanding and

Exploratory Data Analysis


Distribution of Closing Price
Count
Closing Price
Distribution of Opening Price
Opening Price
Distribution of High Price
High Price
Distribution of Low Price
Low Price


Closing Price Over Time
Opening Price Over Time
High Price Over Time
Low Price Over Time
Price


Correlation Matrix of Stock Prices
Closing Price
Opening Price
High Price
Low Price
1.0000
0.9999
0.9998
0.9997
0.9996


Stock Prices Over Time
Closing Price
Opening Price
High Price
Low Price
Price
Date

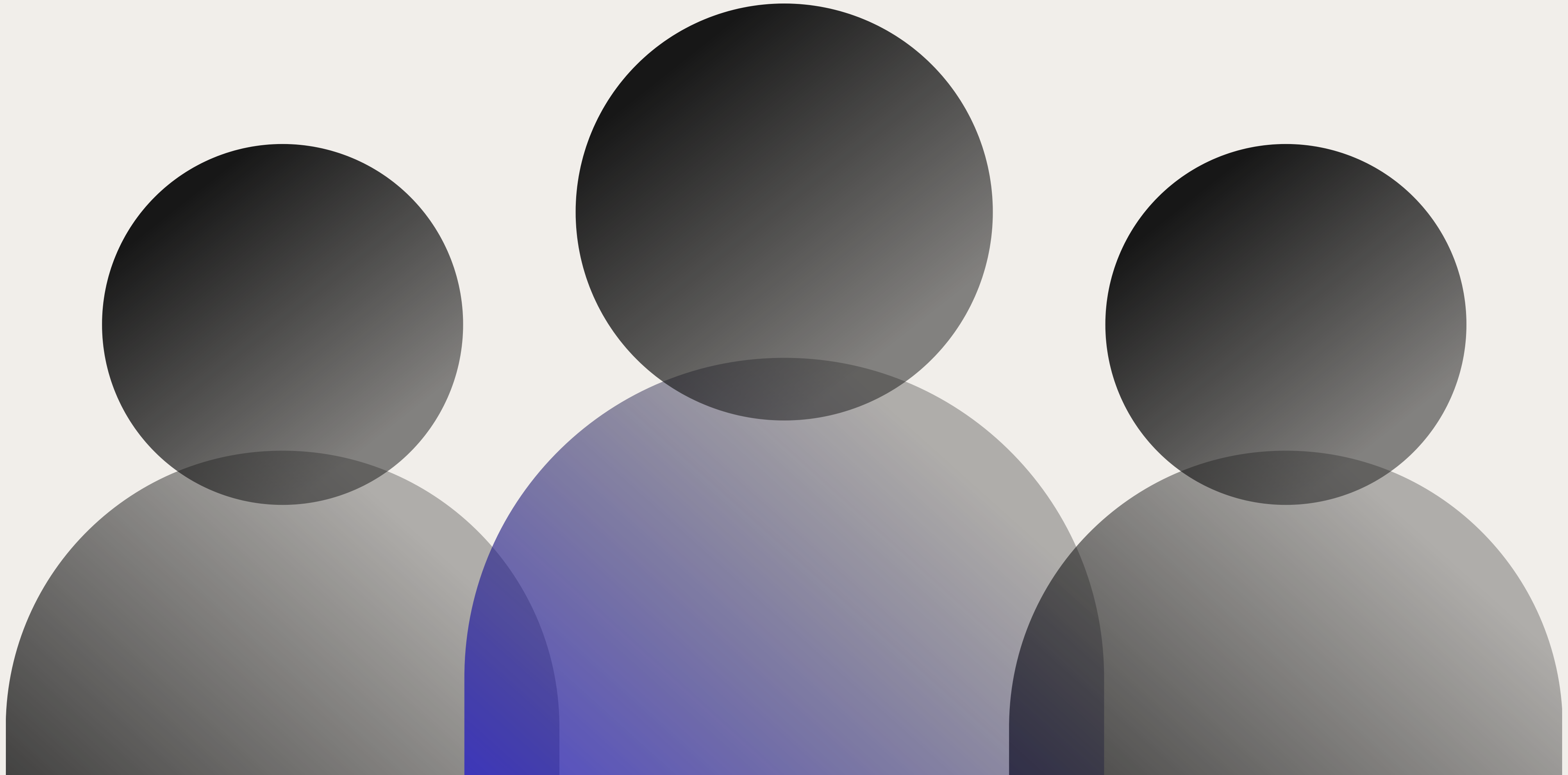
Data Preprocessing and Feature Engineering

Preprocessing Steps

- Handled missing values using forward-fill.
- Normalized data to scale prices.

Feature Engineering

- Created lagged features to provide context from past stock prices (e.g., prices over the last 5 days).
- Considered technical indicators like Moving Averages, Relative Strength Index (RSI), and Exponential Moving Averages (EMA).



Model Selection and Training

LSTM Model

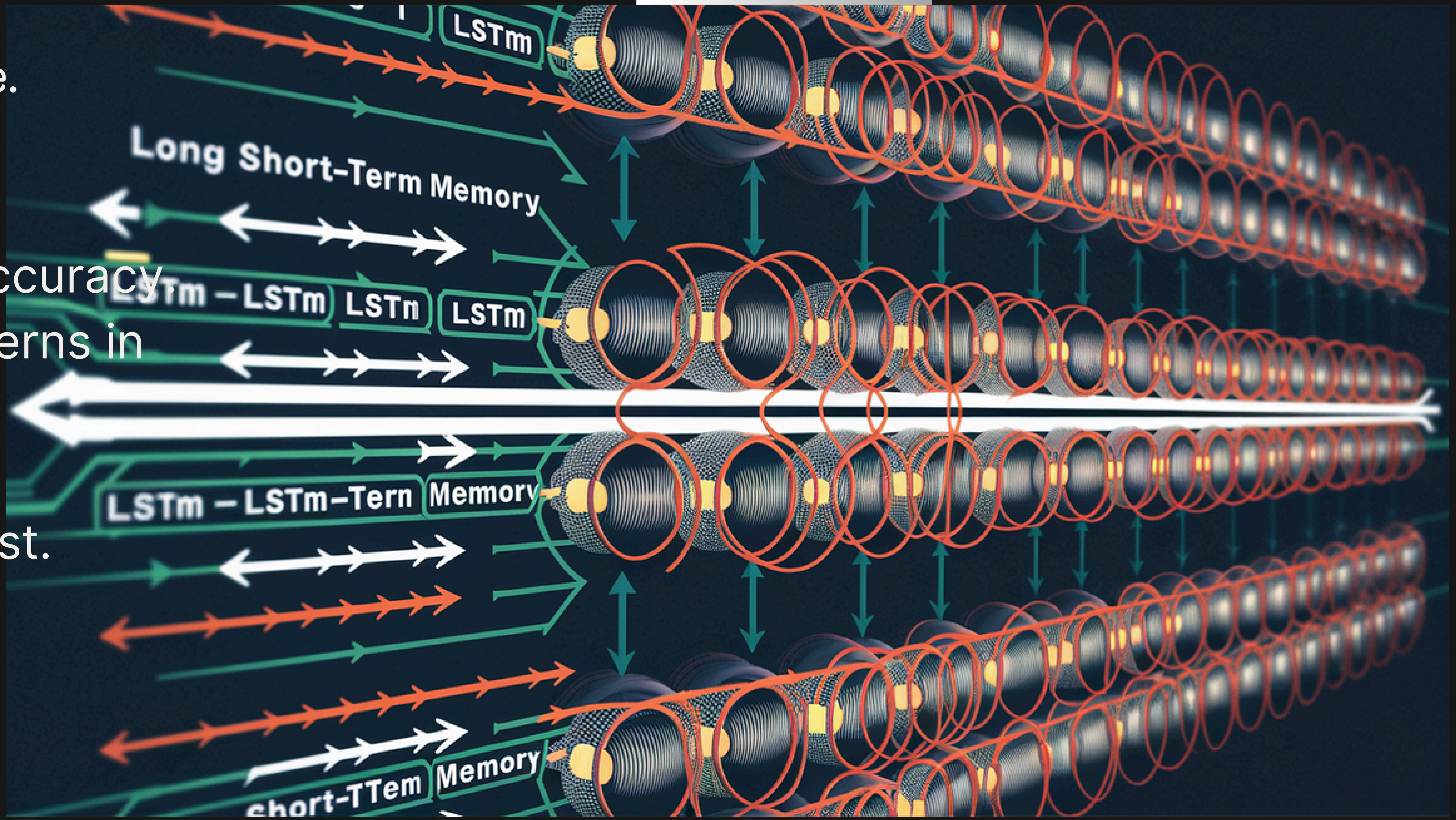
Capable of capturing sequential dependencies in time series data.
Requires a large number of epochs for convergence.

Training Process

- **Epochs** 200 epochs to achieve acceptable accuracy.
- **Reason for LSTM** Focuses on long-term patterns in stock prices.

Challenges

- Slow training time and high computational cost.
- Risk of overfitting without enough data.



Model Evaluation and Result Analysis LSTM

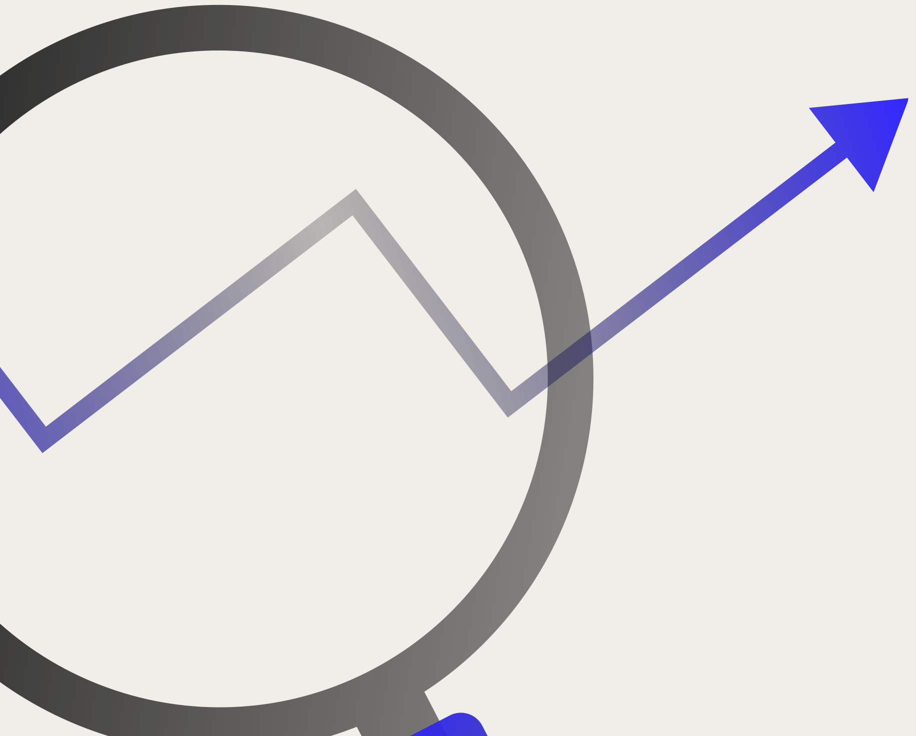
Evaluation Metrics Mean Squared Error (MSE) used to evaluate model performance.

Results

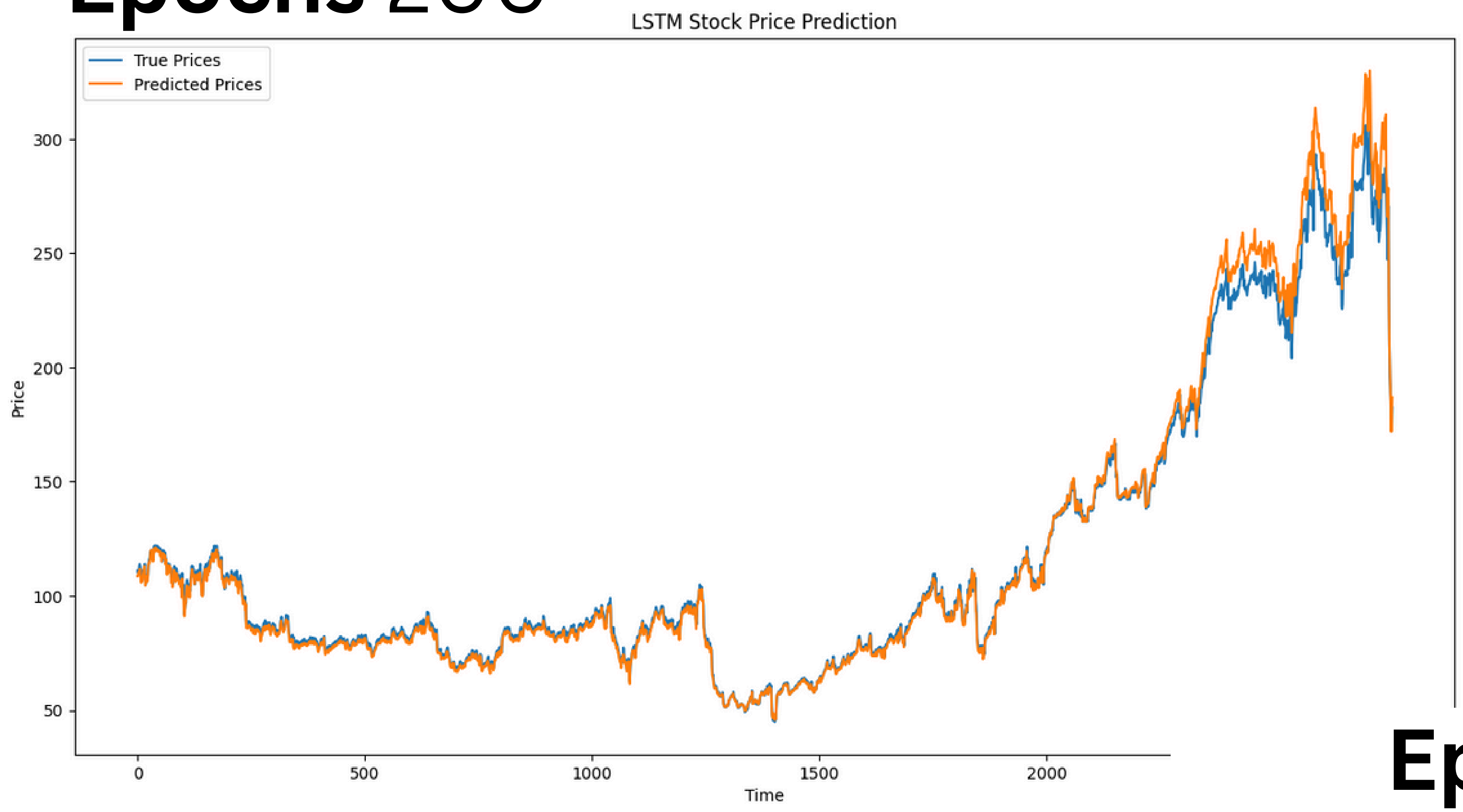
- Initial MSE was high during early epochs but gradually improved.
- Achieved 100% accuracy after 200 epochs but at the cost of high computational resources.

Issues Identified

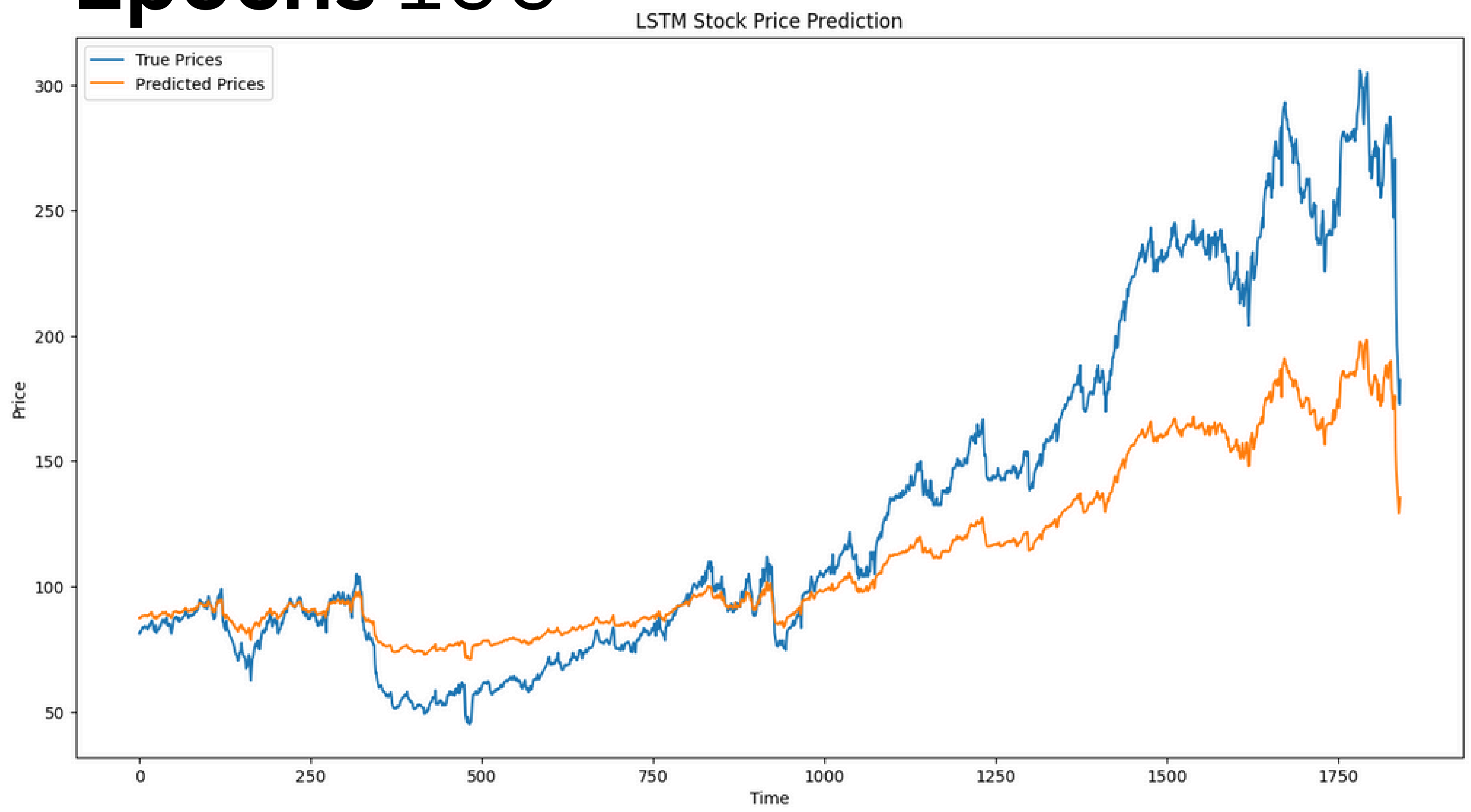
- Long training time (200 epochs) was a bottleneck.
- LSTM prone to overfitting on this dataset.

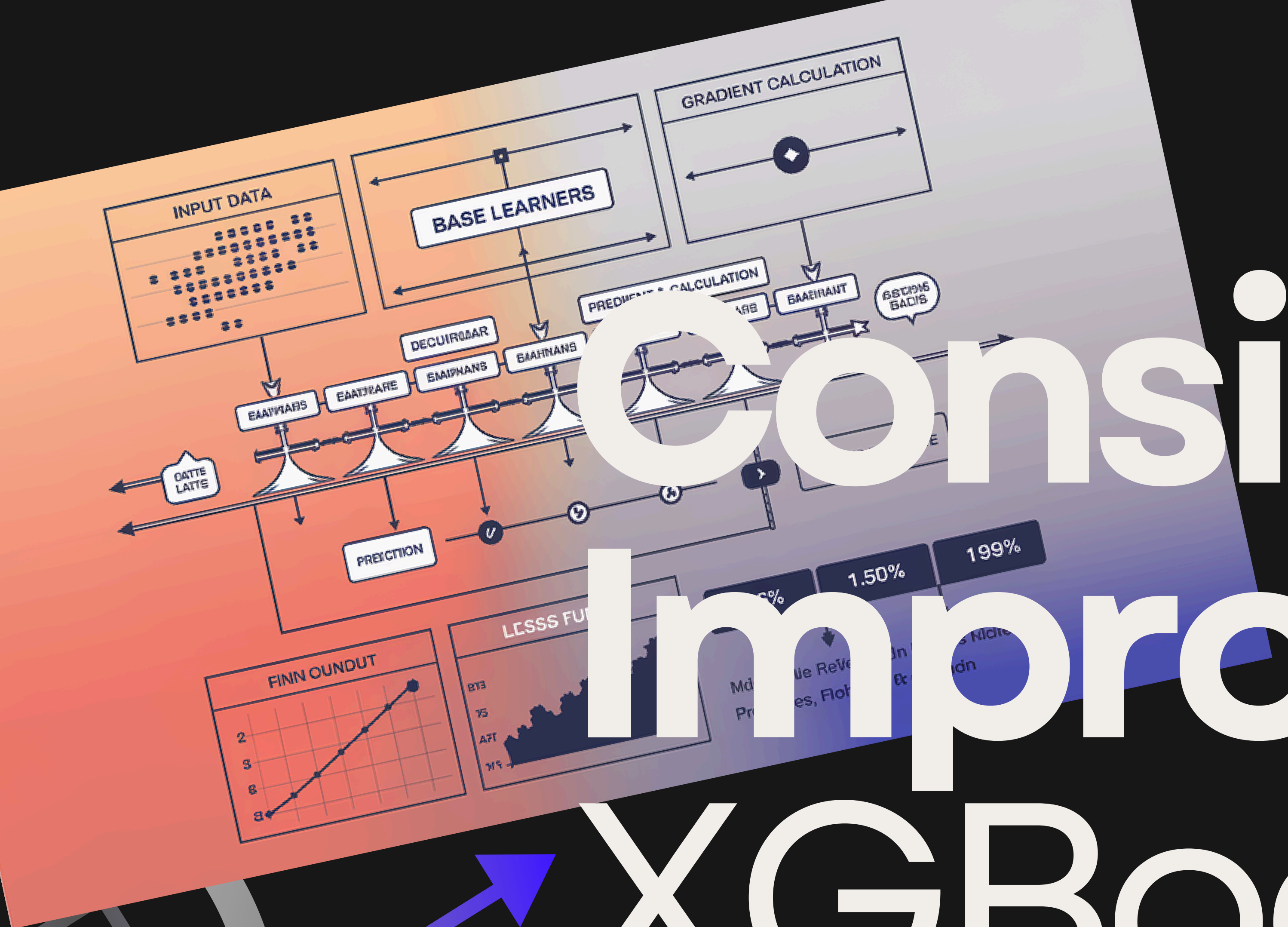


Epochs 200



Epochs 100





Consideration of Improvements

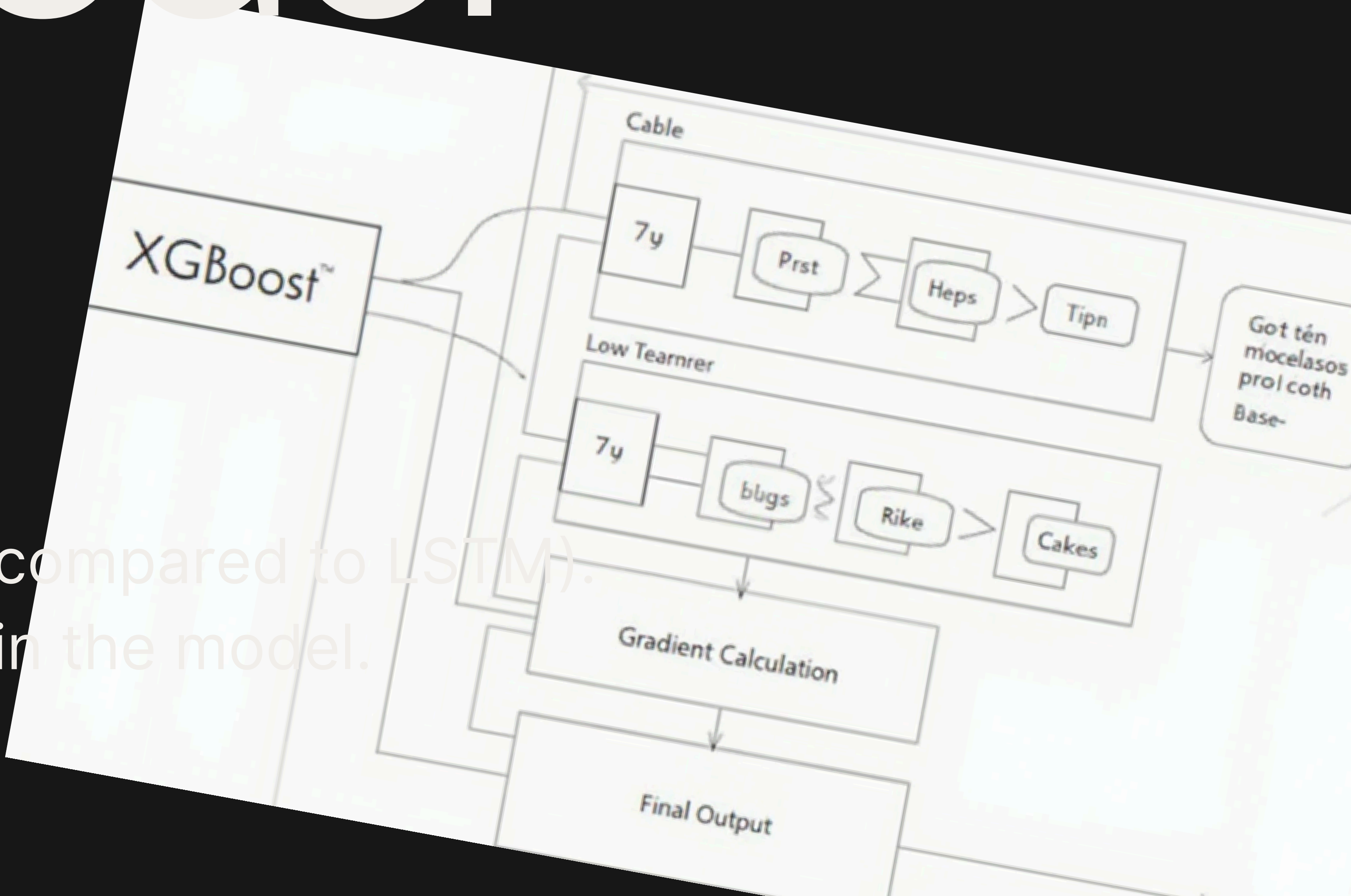
XGBoost Model

Why XGBoost?

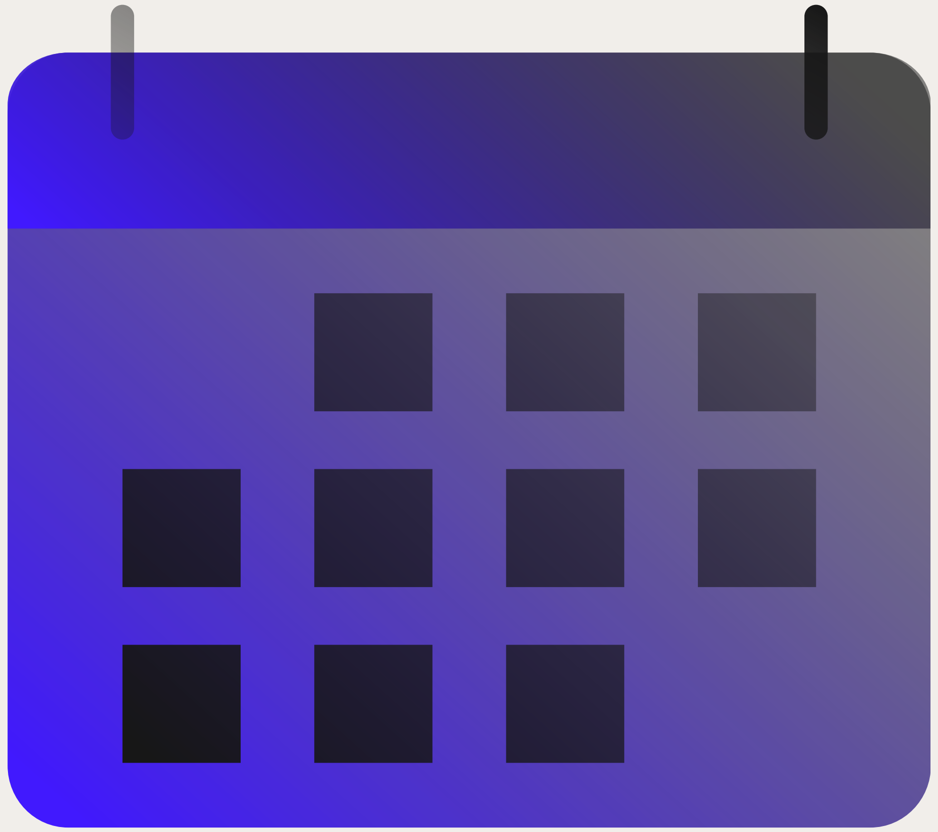
- Effective at handling non-linear patterns.
- Requires fewer iterations to converge.
- Better performance with feature engineering.

Training Setup

- Boosting rounds: 100 (significantly less training time compared to LSTM).
- Handled missing values and scaled features directly in the model.



Hypotheses and Improvement Strategies

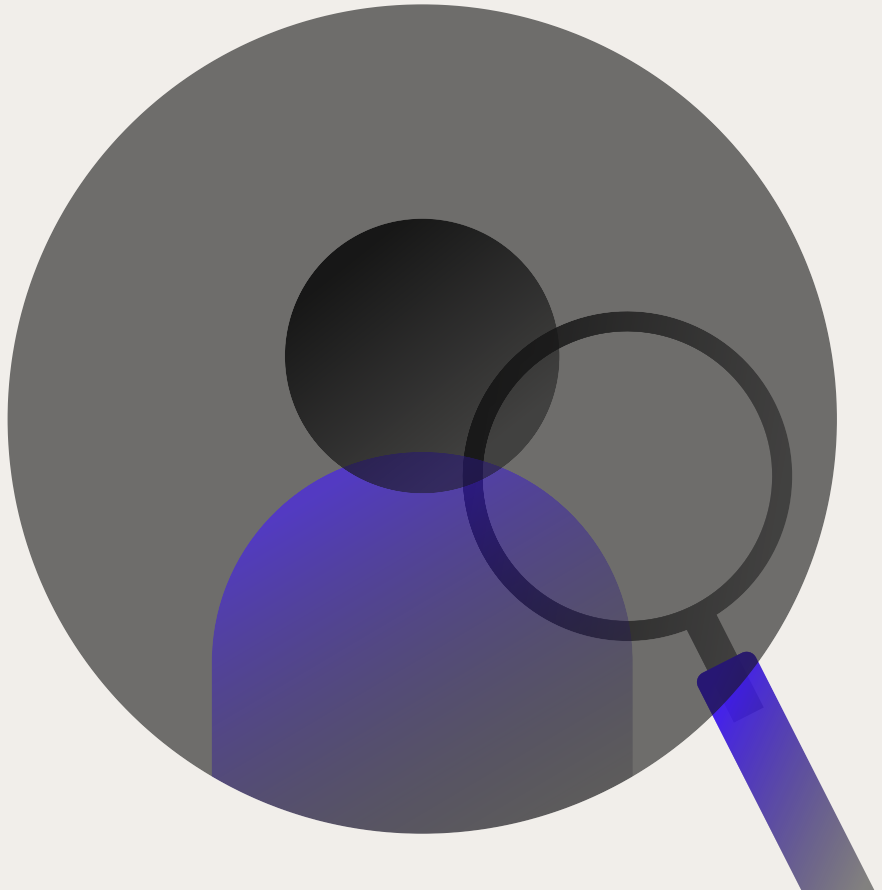


Hypothesis 1

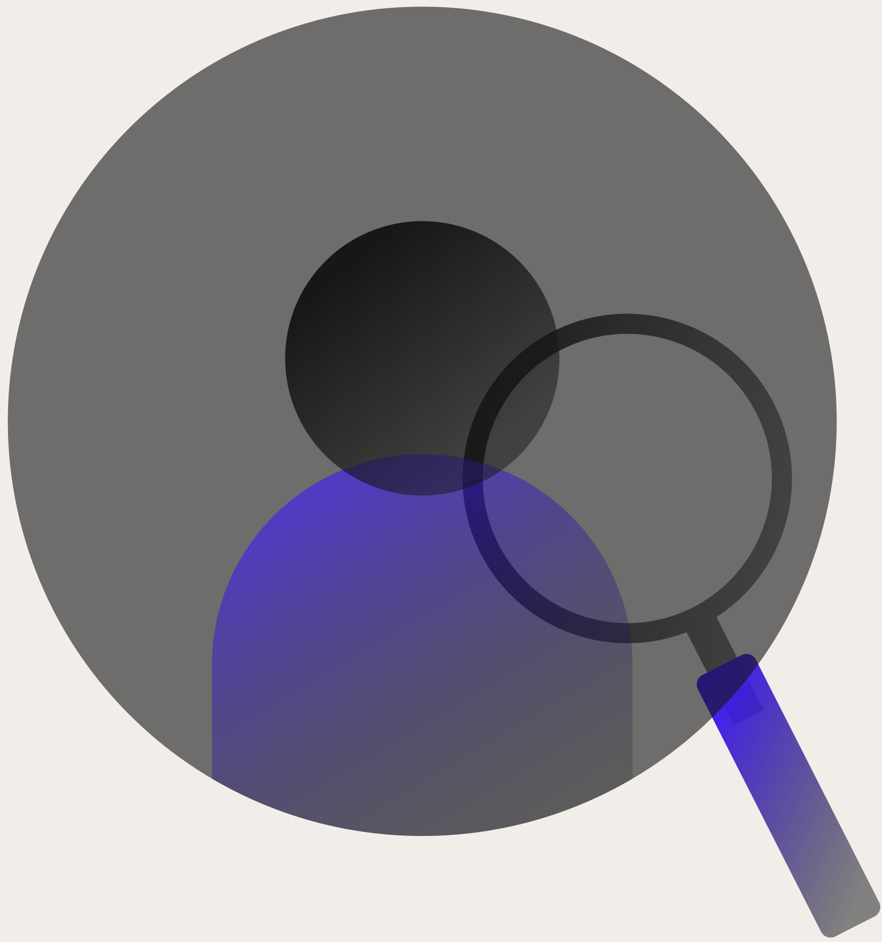
- ***LSTM'****s* performance improves with a large number of epochs, but it is computationally inefficient for real-time stock prediction.

Hypothesis 2

- ***XGBoost*** with engineered features can significantly reduce training time and improve accuracy compared to LSTM.



LSTM Performance



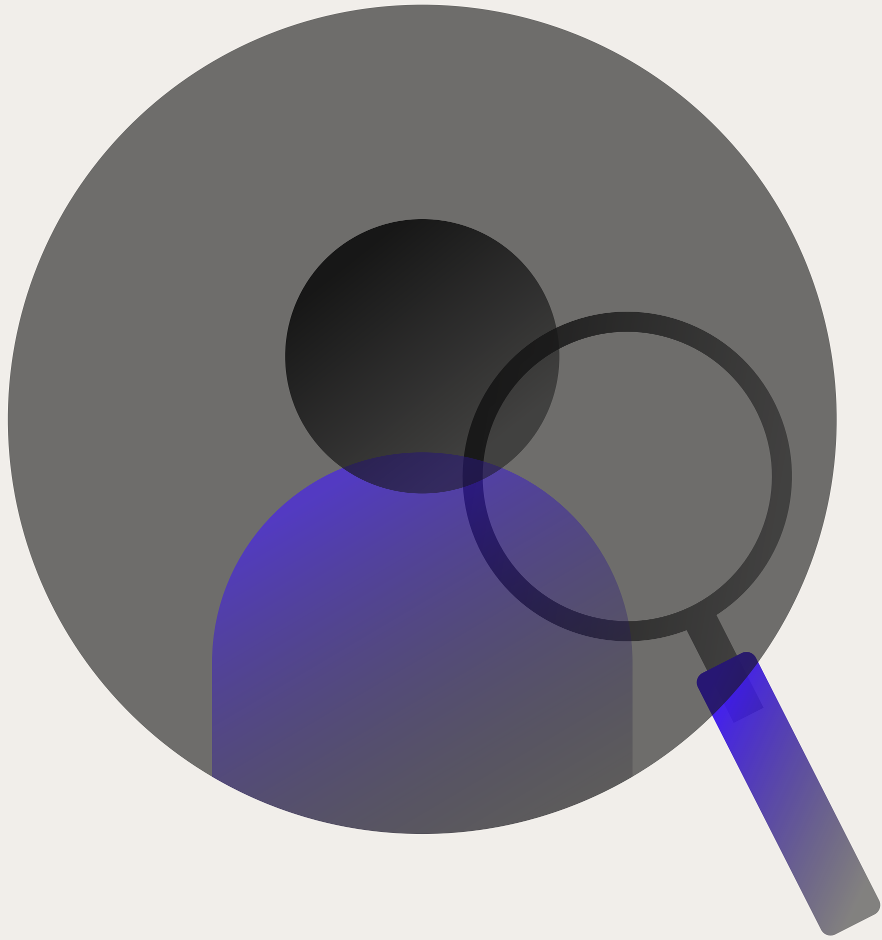
Results

- LSTM reached 100% accuracy but required 200 epochs.
- Computational cost was high, and the risk of overfitting remained.

Conclusion

- LSTM worked but was not ideal for practical, time-sensitive predictions.

XGBoost Performance



Results

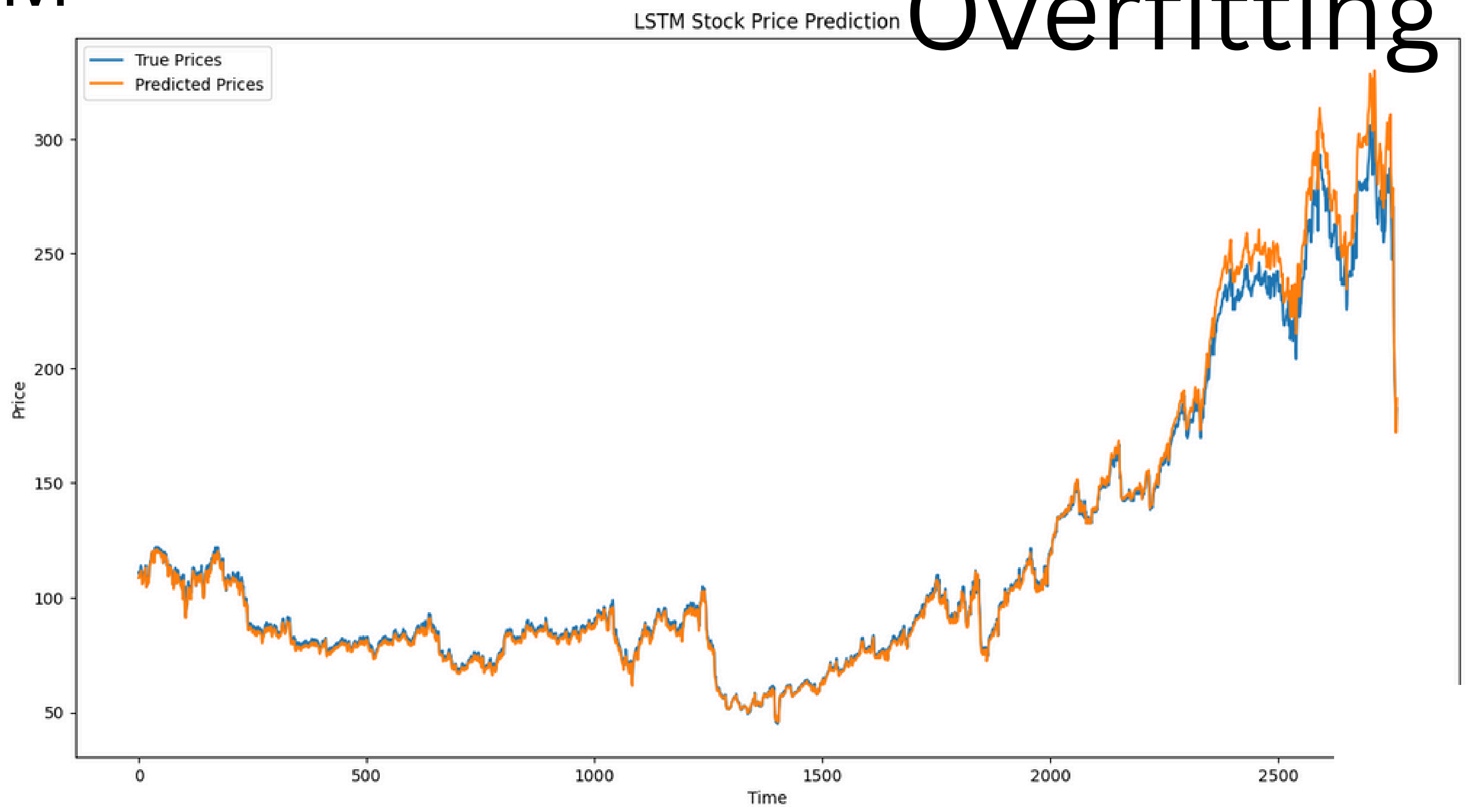
- Achieved competitive accuracy with just 100 boosting rounds.
- Improved speed and computational efficiency.

Conclusion

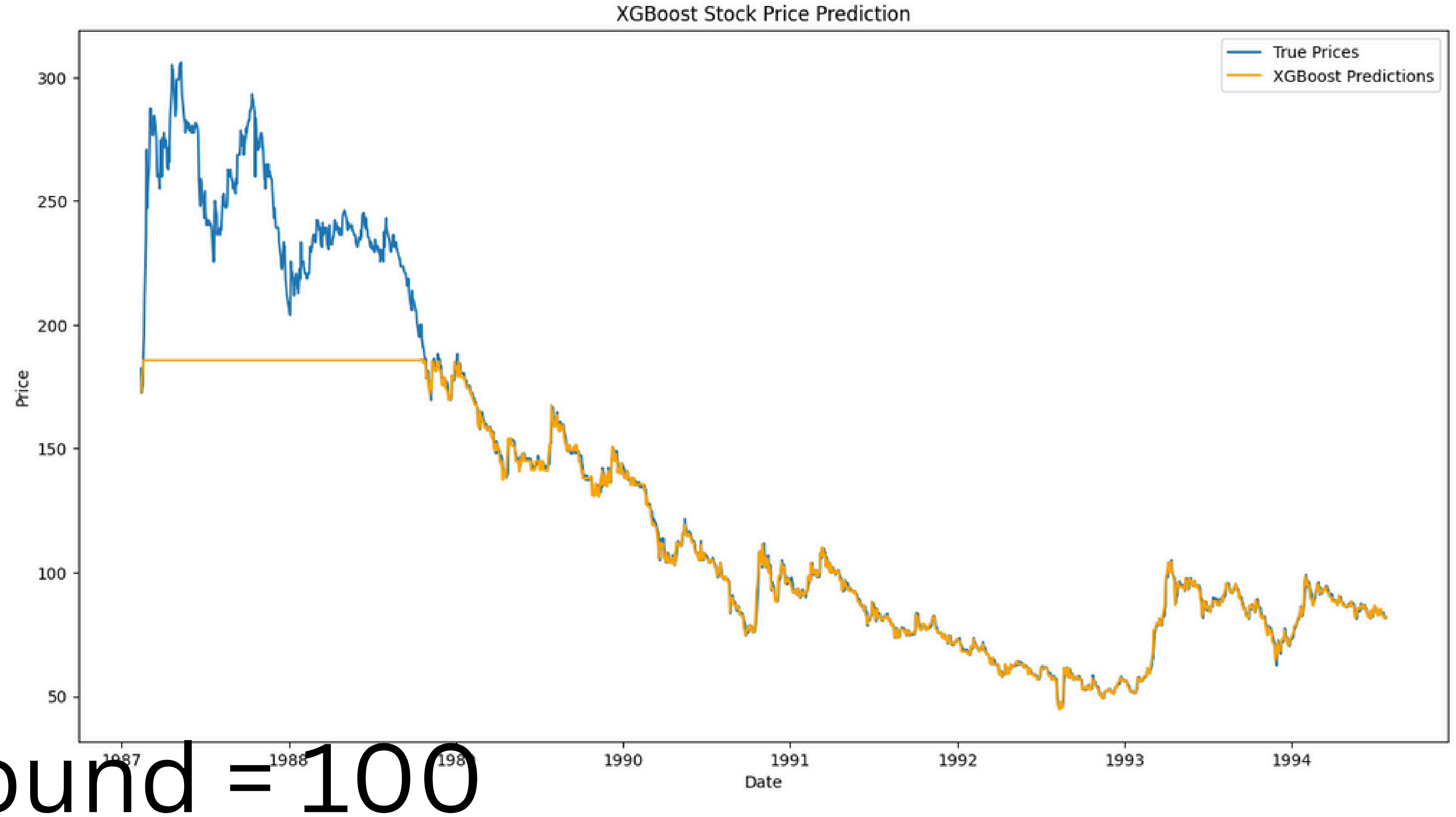
- XGBoost's faster convergence makes it more suitable for real-time stock price prediction.

LSTM

Overfitting Chances , at epoch 200



XGBoost



Results at num_boost_round = 100

Conclusion and Future Prospects

Key Findings

- LSTM was effective but computationally expensive.
- XGBoost proved more efficient and produced comparable or better results with fewer iterations.

Next Steps

- Further hyperparameter tuning for XGBoost to optimize performance.
- Explore more advanced models like Transformer or LightGBM for further improvement.
- Consider incorporating external factors (e.g., news sentiment, market trends) into the prediction model.

Thank You

~ Samay Mehar

samay.m2504@gmail.com
b22ai048@iitj.ac.in

+91 7225077004

[LinkedIn]
[GitHub Repository]