Stock Price Prediction WebApp

Leveraging Deep Learning for Accurate Stock Price Predictions

Guidance by: Shri Ram Vaddadi (IBM Mentor)

Himanshu Prajapati
Mohit Maurya
Surajit Ghosh
Mahendra Kumar Bharduwaj

Problem Statement

- Develop a Stock Price Prediction Streamlit application that leverages deep learning models to recognize and classify price activities based on real data collected from yfinance API or Groww API.
- Accurately predict stock prices, enabling applications in stock price monitoring, top gainers, and top losers.

Requirements

- Data Collection: Gather a diverse dataset containing historical stock price data from yfinance API or Groww API.
- Data Preprocessing: Clean and preprocess the collected data, including steps such as normalization, feature scaling, and handling missing values.
- Feature Engineering: Extract relevant features from the stock price data that capture important patterns and characteristics indicative of stock price movements.
- Model Selection: Evaluate and select appropriate deep learning architectures for stock price prediction tasks.
- Model Training: Train the selected deep learning model using the preprocessed data, optimizing model parameters and hyperparameters to maximize predictive performance.
- Model Evaluation: Assess the performance of the trained stock price prediction model using evaluation metrics such as mean absolute error (MAE), mean squared error (MSE), and R-squared.

Model Selection

- Recurrent Neural Networks (RNNs): Suitable for modeling sequential data, such as stock prices.
- Convolutional Neural Networks (CNNs): Suitable for modeling spatial hierarchies, such as stock price charts.
- Long Short-Term Memory (LSTM) networks: A type of RNN that can learn long-term dependencies in data.
- Hybrid models: Combining RNNs and CNNs to leverage the strengths of both architectures.

Model Training

- Train the selected deep learning model using the preprocessed data.
- Optimize model parameters and hyperparameters to maximize predictive performance.
- Use techniques such as batch normalization, dropout, and regularization to prevent overfitting.

Model Evaluation

- Assess the performance of the trained stock price prediction model using evaluation metrics such as mean absolute error (MAE), mean squared error (MSE), and R-squared.
- Use techniques such as walk-forward optimization to evaluate the model's performance on unseen data.

Deployment

- Deploy the trained deep learning model into a production environment using Streamlit.
- Integrate the stock price prediction system with existing financial applications, platforms, or APIs.
- Implement mechanisms for real-time monitoring and visualization of predicted stock prices.

Real-world Use Case

- Stock Price Monitoring: Monitor and track stock prices in real-time, providing users with up-to-date information on stock market trends and patterns.
- Portfolio Optimization: Optimize investment portfolios by predicting stock prices and identifying potential investment opportunities.
- Risk Management: Identify potential risks and anomalies in stock price data, enabling investors and traders to make informed decisions to mitigate potential losses.

Conclusion

- The presentation outlines a project to develop a Stock Price Prediction Streamlit Application using deep learning models such as RNN, CNN, LSTM, and hybrid models. The application aims to accurately predict stock prices, enabling applications in stock price monitoring, top gainers, and top losers.
- Enhancing model performance.
- User interface enhancements.
- Integrating with additional data sources.
- Continuous monitoring and evaluation.

References

- Most Active Stocks: US stocks with the highest trading volume today - Yahoo Finance
- Groww - Online Demat, Trading and Direct Mutual Fund Investment in India
- Streamlit Community Cloud • Streamlit
- GitHub
- 15 Best Python Libraries for Machine and Deep Learning (springboard.com)