Stock Price Predictor: Development Report

Introduction

The "Stock Price Predictor" is a web-based application developed using Streamlit to enable users to visualize historical stock prices, predict future stock prices, and learn fundamental stock market concepts. This project was undertaken as an academic exercise to apply machine learning, data visualization, and web development skills. The development process began with understanding the dataset and concluded with deploying the model on Streamlit Cloud. This report details the step-by-step development process as of June 1, 2025.

Understanding the Dataset

The project began with a sample dataset, `stocks.csv`, which contained historical stock price data for multiple companies. The dataset included the following columns:

- `Date`: The date of the stock price record.
- `Ticker`: The stock symbol (e.g., AAPL, MSFT, NFLX, GOOG).
- `Close`: The closing price of the stock on that date.

The initial analysis involved loading the dataset using Pandas to examine its structure, identify missing values, and ensure the data types were appropriate. The `Date` column needed to be converted to a datetime format for time-series analysis, and the dataset was checked for completeness to ensure it could support the predictive modeling and visualization tasks.

Data Preprocessing and Feature Engineering

After understanding the dataset, preprocessing was performed to prepare the data for modeling:

- **Date Conversion**: The `Date` column was converted to datetime format using `pd.to_datetime()` to enable time-series operations.
- **Feature Engineering**:
  - Calculated the 10-day moving average (MA10) for each ticker using `rolling(window=10).mean()` to capture short-term trends.
  - Computed daily returns using `pct_change()` to measure price volatility.

- **Handling Missing Values**: Rows with missing values (resulting from moving average calculations) were dropped using dropna() to ensure the dataset was clean for modeling.
- **Data Segmentation**: The dataset was segmented by ticker to train separate models for each company (AAPL, MSFT, NFLX, GOOG).

The features used for prediction included the last 5 days' closing prices, moving averages, and returns, creating a 15-dimensional input vector for each prediction.

Model Development and Training

A machine learning model was developed to predict the next day's closing price:

- **Model Selection**: A RandomForestRegressor from Scikit-learn was chosen for its robustness and ability to handle non-linear relationships in financial data.
- **Feature Extraction**: For each ticker, a function create_features was implemented to extract a sliding window of 5 days' worth of closing prices, moving averages, and returns as input features, with the target being the closing price on the 6th day.
- **Training**: Separate models were trained for each ticker using 100 trees (n_estimators=100), a maximum depth of 10 (max_depth=10), and a minimum samples split of 2 (min_samples_split=2) to balance complexity and performance. The models were trained on the preprocessed dataset using model.fit(X, y).

The trained models were stored in a dictionary for use in the Streamlit app, enabling real-time predictions based on user inputs.

Visualization and User Interface Development

The Streamlit app was developed to provide an interactive user experience:

- **Historical Price Visualization**: A line graph was created using Matplotlib to display historical closing prices for a selected ticker. The x-axis used dynamic date formatting (YearLocator and DateFormatter) to ensure clear year labels, adjusting based on the data range (e.g., showing only years for spans >5 years, or years and months for shorter spans).
- **Prediction Interface**: Users could select a ticker and input the last 5 days' closing prices via st.number_input. The app then used the trained model to predict the next day's price, displayed via st.success.
- **Educational Section**: A section titled "Stock Market Basics 📚" was added, explaining key concepts (e.g., stocks, markets, bull/bear markets, dividends).

Deployment of the Model

The final step involved deploying the app for accessibility:

- **Local Testing**: The app was tested locally using streamlit run app.py, ensuring all features (visualization, prediction, educational content) worked as expected. Dependencies were verified (streamlit, pandas, numpy, scikit-learn, matplotlib).
- **Deployment on Streamlit Cloud**:
    - The project was uploaded to a GitHub repository, including the app code (app.py), dataset (stocks.csv), and a requirements.txt file listing dependencies.
    - Streamlit Cloud was used to deploy the app by linking the GitHub repository, selecting the main branch, and specifying app.py as the entry point.
    - The deployment was successful, and the app was accessible via a public URL, allowing users to interact with the stock price predictor online.
- **Verification**: Post-deployment, the app was tested to ensure the model predictions, visualizations, and educational content rendered correctly, with no errors in the Streamlit Cloud logs.

Conclusion

The Stock Price Predictor project successfully progressed from understanding the dataset to deploying a functional web application. Starting with a dataset of historical stock prices, the project involved preprocessing, feature engineering, model training with RandomForestRegressor, and building an interactive Streamlit app with visualizations and educational content. Iterative design improvements ensured a professional, user-friendly interface, addressing all feedback and technical challenges. The deployment on Streamlit Cloud made the app accessible to users, fulfilling the project's objectives. This project provided valuable experience in machine learning, data visualization, and web development, with potential for future enhancements such as real-time data integration and advanced predictive models.

Submitted by: Aasti Priya
Internship id: UMID28042533335

DASHBOARD LINK: click here