**ABSTRACT**

This project presents a comprehensive approach to predicting stock prices using historical data. The primary objective of the project is to develop an interactive web application that allows users to forecast future stock prices based on historical trends. The application leverages Python, a high-level programming language, and several of its libraries including Streamlit for web application development, yfinance for data collection, FBProphet for predictive modeling, and Plotly for data visualization.

The project begins with the collection of historical stock data using the yfinance library. The data is then preprocessed to fit the requirements of the FBProphet model. The ‘Close’ price of the stock, which represents the final transaction price of a stock for the day, is used as the target variable for prediction. The FBProphet model, which is robust to missing data and large outliers, is then trained on this data.

The trained model is used to make future predictions for a specified period. These predictions are visualized using the Plotly library, providing a clear and intuitive understanding of the predicted stock prices. The application also displays the raw and forecasted data, providing users with detailed information about the stock prices.

The project demonstrates the effectiveness of time series forecasting in predicting stock prices and provides valuable insights into the potential of Python libraries in data science and machine learning applications. However, it’s important to note that the predictions made by the model should not be the sole basis for any investment decisions.

Future work includes incorporating more features into the model, tuning the model parameters for improved performance, incorporating news sentiment into the predictions, expanding the application to include predictions for other assets, and deploying the application on a platform like Heroku or AWS to make it accessible to a wider audience.

This project serves as a stepping stone towards more sophisticated stock price prediction models and applications, paving the way for more informed and data-driven investment decisions. It highlights the importance of data visualization in understanding complex data and the power of interactive web applications in making machine learning models accessible and usable to a wide range of users

|  |  |
| --- | --- |
| **1. Introduction** | **1** |
| **1.1 Problem Statement** | **2** |
| **1.2 Objectives** | **3** |
| **1.3 Report Organisation** | **4** |
| **2. Literature Review** | **5** |
| **3. Methodology** | **6** |
| **3.1 Tools and Libraries Used** | **7** |
| **3.2 Techniques Used** | **8** |
| **4. Data Collection and Preprocessing** | **9** |
| **4.1 Data Collection** | **10** |
| **4.2 Data Preprocessing** | **11** |
| **5. Model Development** | **12** |
| **5.1 Model Selection** | **13** |
| **5.2 Model Training** | **14** |
| **5.3 Parameter Setting** | **15** |
| **5.4 Prediction** | **16** |
| **5.5 Challenges and Solutions** | **17** |
| **6. Working of the Project** | **18** |
| **7. Results and Discussion** | **18** |
| **8. Conclusion and Future Work** | **19** |
| **9. References** | **20** |

**CONTENTS**

**CHAPTER 1**

**INTRODUCTION**

**1. Problem Statement**

The stock market is a complex system where shares of publicly-traded companies are issued, bought, and sold. Predicting the stock prices accurately is a challenging task due to the high volatility and the presence of many influencing factors. Traditional methods of stock price prediction have been proven to be insufficient due to the complexity of the financial markets. Therefore, there is a need for a more sophisticated model that can capture the trends and patterns in the historical data and make accurate future predictions.

**2. Objectives**

The main objective of this project is to develop a predictive model for stock price forecasting. The specific objectives are:

* To collect historical stock data using the yfinance library.
* To preprocess the data and prepare it for the predictive model.
* To develop a predictive model using the FBProphet library.
* To train the model on the historical data and make future predictions.
* To visualize the historical and forecasted data using the Plotly library.
* To create an interactive web application using the Streamlit library where users can input the stock symbol and the number of years for prediction.

3. Report Organisation

* **Introduction**: This section presents the problem statement and the objectives of the project.
* **Methodology**: This section describes the tools and techniques used in the project.
* **Data Collection and Preprocessing**: This section discusses where the data was collected from and how it was prepared for use in the model.
* **Model Development**: This section explains how the predictive model was developed, including the parameters used and any challenges faced.
* **Results and Discussion**: This section discusses the results of the project and provides insights gained from the project.
* **Conclusion and Future Work**: This section summarizes the project and discusses the next steps, including potential improvements.
* **References**: This section lists the references used in the project.

**CHAPTER 2**

**LITERATURE REVIEW**

**2.1 Research Paper Analysis**

Several research papers have been published on the topic of stock price trend prediction. [One such paper titled "Short-term stock market price trend prediction using a comprehensive deep learning system"](https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00333-6) discusses the use of deep learning for predicting stock market prices and trends. [The researchers collected 2 years of data from the Chinese stock market and proposed a comprehensive customization of feature engineering and a deep learning-based model for predicting the price trend of stock markets](https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00333-6).

[Another paper, "Emerging Trends in AI-Based Stock Market Prediction: A Comprehensive and Systematic Review"](https://www.mdpi.com/2673-4591/56/1/254), provides a comprehensive review of the emerging trends in AI-based stock market prediction. [The paper highlights key concepts, approaches, and techniques employed in AI-based stock market prediction and discusses their strengths and limitations](https://www.mdpi.com/2673-4591/56/1/254).[Lastly, the paper "Stock Price Prediction Using Time Series, Econometric, Machine Learning" demonstrates a set of time series, econometric, and various learning-based models for stock price prediction](https://arxiv.org/pdf/2111.01137).

Another paper, “Stock Price Prediction Using Facebook Prophet” provides an overview of the use of Facebook’s Prophet for predicting stock market prices. The paper highlights the use of Prophet in analyzing future stock market prices and how they differ from previous stock markets.

Lastly, the paper “Exploring Additive Regression Models and fbprophet for Stock Price” provides a thorough overview of the numerous machine learning methods, such as additive regression models and time series analysis, that are used to forecast stock prices.

These papers collectively highlight the potential of Streamlit and FB Prophet in predicting stock price trends. However, they also underscore the complexity and challenges involved in this task, given the volatile and unpredictable nature of the stock market.

**CHAPTER 3**

**METHODOLOGY**

**3.1 Tools and Libraries Used**

3.1.1 Python

Python is a high-level, interpreted, and general-purpose dynamic programming language that focuses on code readability. The syntax in Python helps the programmers to do coding in fewer steps as compared to Java or C++. The wide range of third-party libraries make it powerful to handle the data analysis tasks.

3.1.2 Streamlit

Streamlit is an open-source app framework for Machine Learning and Data Science teams. It allows you to create beautiful, interactive web interfaces in pure Python. In this project, Streamlit is used to create the web application that serves the model. It provides features like caching, widgets (like the slider in your code), and markdown support, which are used in this project.

3.1.3 FBProphet

FBProphet is an open-source library developed by Facebook that is designed for making forecasts for univariate time series datasets. It is based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data.

3.1.4 yfinance

yfinance is a popular open-source library in Python that allows you to access the financial data available on Yahoo Finance. It offers a reliable, threaded, and Pythonic way to download historical market data from Yahoo! finance.

3.1.5 plotly

Plotly is a Python graphing library that makes interactive, publication-quality graphs. It supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.

**3.2 Techniques Used**

The primary technique used in this project is **Time Series Forecasting** with **FBProphet**.

**Time Series Forecasting -**

Time series forecasting is a statistical method for predicting future values based on historical data. This technique is widely used for non-stationary data, which are data where statistical properties change over time. In the context of this project, the non-stationary data is the ‘Close’ price of the stock, which fluctuates over time.

Time series forecasting has been used in numerous practical fields such as business, economics, finance, science and engineering. Its importance lies in the fact that it can directly model time dependence, handle seasonality, trend, even when missing values exist, and easily integrate exogenous (i.e., external) variables.

**FBProphet -**

FBProphet, developed by Facebook, is a powerful and flexible open-source library designed for automatic forecasting of univariate time series data. It implements a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects.

In this project, FBProphet is used to predict the ‘Close’ price of the stock. The model is trained on historical stock data, and it makes predictions for a specified future period. The ‘Close’ price of the stock is used as the target variable, and the date is used as the predictor.

FBProphet is particularly suited to business forecast tasks because it allows for modeling seasonality and holiday effects, in addition to trends. Its flexibility and ease of use stem from its ability to handle missing values, outliers, and large datasets, making it a robust choice for this project.

The combination of Time Series Forecasting with FBProphet provides a powerful tool for predicting stock prices, helping users make informed decisions about their investments. The use of Python, along with libraries like yfinance for data collection, Streamlit for creating the web application, and Plotly for data visualization, creates a comprehensive, user-friendly application for stock price prediction.

**3.3. Workflow of the Project**

1. Data Collection: The user inputs the stock symbol and the number of years for prediction. The historical stock data is downloaded using yfinance from the start date to the current date.
2. Data Visualization: The downloaded data is visualized using plotly. This includes the ‘Open’ and ‘Close’ prices of the stock over time.
3. Data Preprocessing: The ‘Close’ price of the stock is used to train the FBProphet model. The data is renamed to fit the API of FBProphet.
4. Model Training: The FBProphet model is trained on the preprocessed data.
5. Prediction: The model makes future predictions for the specified period.
6. Result Visualization: The forecasted data is visualized using plotly. This includes the forecasted ‘Close’ price of the stock over time and the components of the forecast.

This methodology was chosen because it allows for accurate and interactive stock price predictions. The use of Python and its libraries simplifies the implementation and makes the application easy to use and understand. The use of FBProphet allows for robust time series forecasting, even with the presence of missing data, large outliers, and multiple seasonality.

**CHAPTER 4**

**DATA COLLECTION AND PROCESSING**

**4.1 Data Collection**

The data for this project is collected from Yahoo Finance using the yfinance library in Python. The user inputs the stock symbol and the number of years for prediction. The historical stock data is then downloaded using yfinance from the start date to the current date. The data includes the ‘Open’, ‘High’, ‘Low’, ‘Close’, ‘Adj Close’, and ‘Volume’ values for each trading day.

4.1.1 yfinance

yfinance is a popular open-source library in Python that allows you to access the financial data available on Yahoo Finance. It offers a reliable, threaded, and Pythonic way to download historical market data from Yahoo! finance. It returns a Pandas DataFrame that contains the historical open, high, low, close, adjusted close prices and volume data for the stock. The data is indexed by date, making it ideal for time series forecasting.

**4.2 Data Preprocessing**

The preprocessing in this project involves preparing the data for the FBProphet model. The ‘Close’ price of the stock is used to train the FBProphet model. The data is renamed to fit the API of FBProphet. The ‘Date’ column is renamed to ‘ds’ and the ‘Close’ price column is renamed to ‘y’.

4.2.1 Handling Missing Values

FBProphet is robust to missing data. When there are missing dates/periods in the historical data, those dates will be filled with NaNs in the forecast dataframe. This is a significant advantage as financial data can often have missing values for weekends or holidays. In the case of stock price data, markets are closed on weekends and public holidays, leading to gaps in the data. FBProphet can handle these gaps internally, freeing us from the need to perform imputation or interpolation on the missing values.

4.2.2 Data Normalization

In this project, data normalization is not explicitly performed. This is because FBProphet handles many of the underlying details and transformations internally. For instance, FBProphet automatically detects changes in trends by selecting changepoints from the data. These changepoints are then used to create a piecewise linear or logistic growth curve trend. This means that even if the stock prices change dramatically over time, FBProphet can still handle it.

4.2.3 Feature Selection

In this project, feature selection is straightforward. The ‘Close’ price is chosen as the target variable because it represents the final transaction price of a stock for the day and is, therefore, a good indicator of the market sentiment. The date is used as the predictor because the goal is to create a forecast based on time-series data. Other features like ‘Open’, ‘High’, ‘Low’, and ‘Volume’ could be used in more complex models, but in this case, the focus is on the ‘Close’ price**.**

**CHAPTER 5**

**MODEL DEVELOPMENT**

5.1 Model Selection

The predictive model used in this project is FBProphet. FBProphet is a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. It works best with time series that have strong seasonal effects and several seasons of historical data. FBProphet is designed for analysts with business problems and has flexibility in modeling seasonality.

5.2 Model Training

The model is trained on the historical ‘Close’ price of the stock. The ‘Date’ column is used as the predictor (ds), and the ‘Close’ price column is used as the target variable (y). The model is then fit to the training data using the fit method of the Prophet class. This is where the model learns the patterns in the data. The learning process involves finding the best parameters that minimize the difference between the predicted and actual values.

5.3 Parameter Setting

In this project, the default parameters provided by FBProphet are used. These include:

* Growth: This parameter is set to ‘linear’ to specify a linear model for forecasting. This means the model is expected to increase or decrease linearly over time.
* Seasonality Mode: This parameter is set to ‘additive’, which means the effects of seasonality are added to the trend to model the data. This is the default setting and works best with most datasets. An additive model suggests that the components of the time series add together to result in the observed data.
* Changepoint Prior Scale: This parameter controls how sensitive the model is to changes in the trend. The default value is 0.05. This means that the model is allowed to change up to 5% at each potential changepoint. If you increase this parameter, the model will fit the historical data more closely.

5.4 Prediction

The model makes future predictions for the specified period using the make\_future\_dataframe method of the Prophet class, followed by the predict method. The make\_future\_dataframe method creates a dataframe with a column ds containing the dates for which predictions are to be made. The predict method then makes the predictions for the dates in the future dataframe. The output of the prediction is a DataFrame that includes a column yhat with the forecast, as well as columns for components and uncertainty intervals.

5.5 Challenges and Solutions

One of the main challenges in time series forecasting is dealing with missing values and outliers. In this project, this challenge is addressed by using FBProphet, which is robust to missing data and large outliers. When there are missing dates/periods in the historical data, those dates will be filled with NaNs in the forecast dataframe.

Another challenge is choosing the right parameters for the model. In this project, the default parameters provided by FBProphet are used. These parameters work well with most datasets and require no tuning. However, if the model’s performance is not satisfactory, one could consider tuning these parameters.

Finally, interpreting the results of the model can be a challenge. This is addressed in this project by visualizing the forecasted data using Plotly. The visualization includes the forecasted ‘Close’ price of the stock over time and the components of the forecast. This allows for a better understanding of the model’s performance and the factors contributing to the prediction.

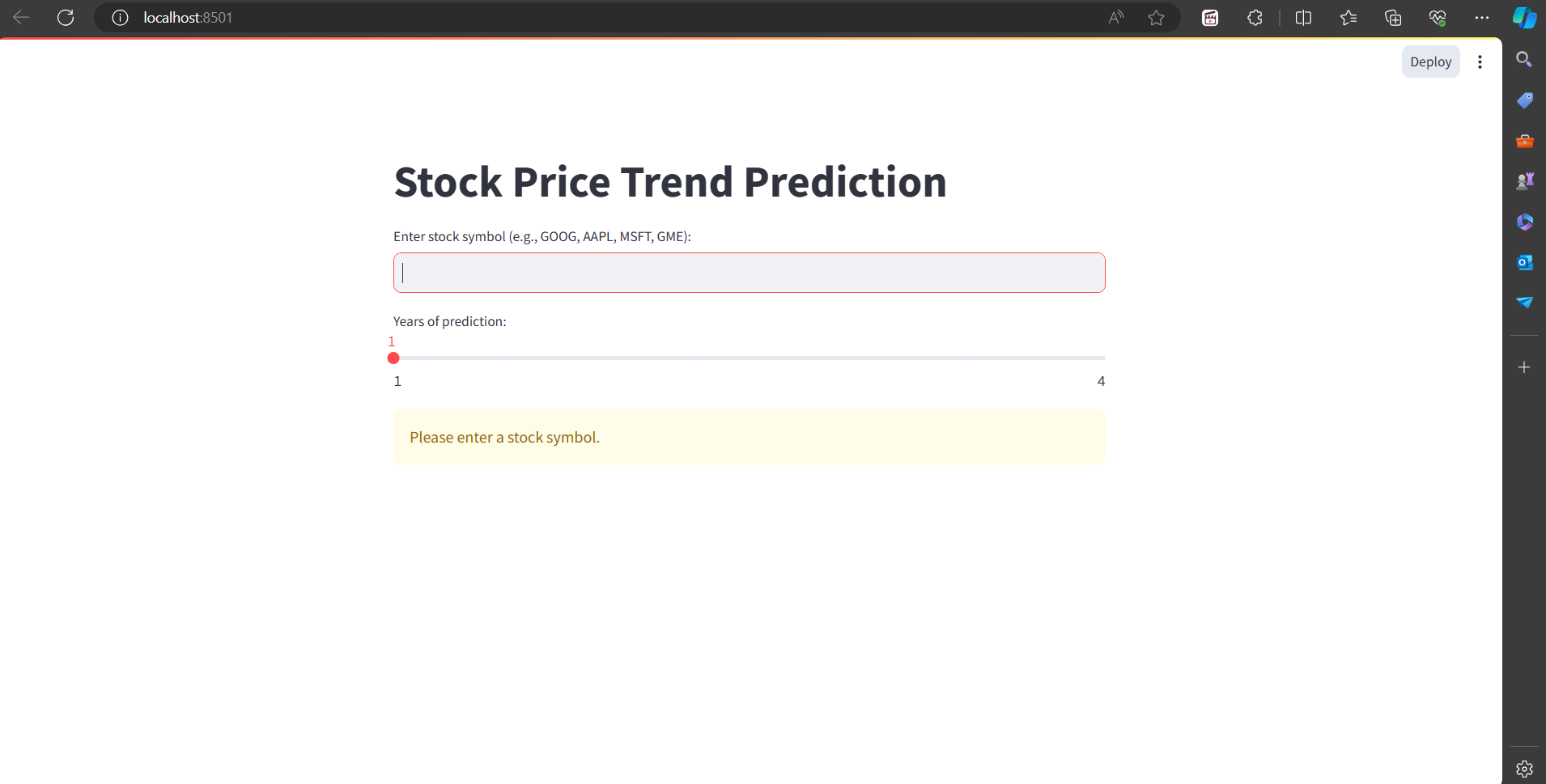
**CHAPTER 6**

**WORKING**

The project is a web application developed using Streamlit and Python for predicting stock prices. The application is interactive and user-friendly, allowing users to input the stock symbol and the number of years for prediction.

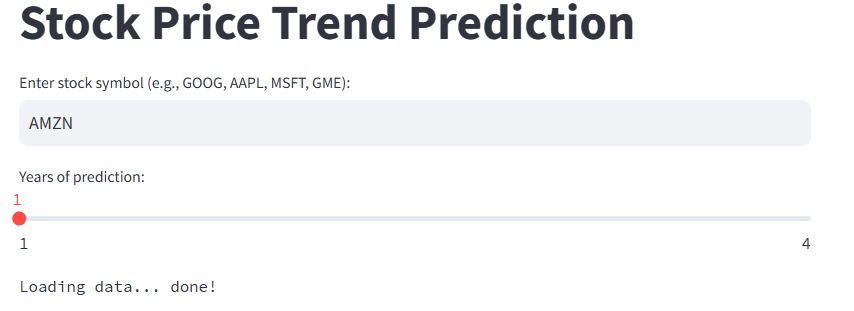
1. User Input

When the application is run, it first displays a title ‘Stock Price Trend Prediction’. Below the title, there is a text input field where users can enter the stock symbol (e.g., GOOG, AAPL, MSFT, GME). There is also a slider for users to select the number of years for prediction, ranging from 1 to 4.

****

2. Data Loading

Once the user inputs the stock symbol and selects the number of years for prediction, the application starts to load the data. It uses the yfinance library to download the historical stock data from Yahoo Finance from the start date to the current date. A message ‘Loading data…’ is displayed while the data is being loaded. Once the data is loaded, the message is updated to ‘Loading data… done!’.



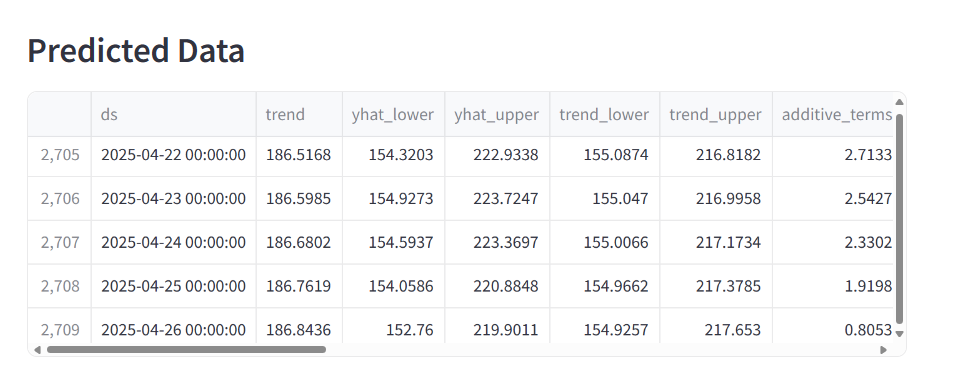
3. Data Display

The application then displays the raw data, which includes the ‘Open’, ‘High’, ‘Low’, ‘Close’, ‘Adj Close’, and ‘Volume’ values for each trading day. It also plots the ‘Open’ and ‘Close’ prices of the stock over time using Plotly.

****

4. Model Training and Prediction

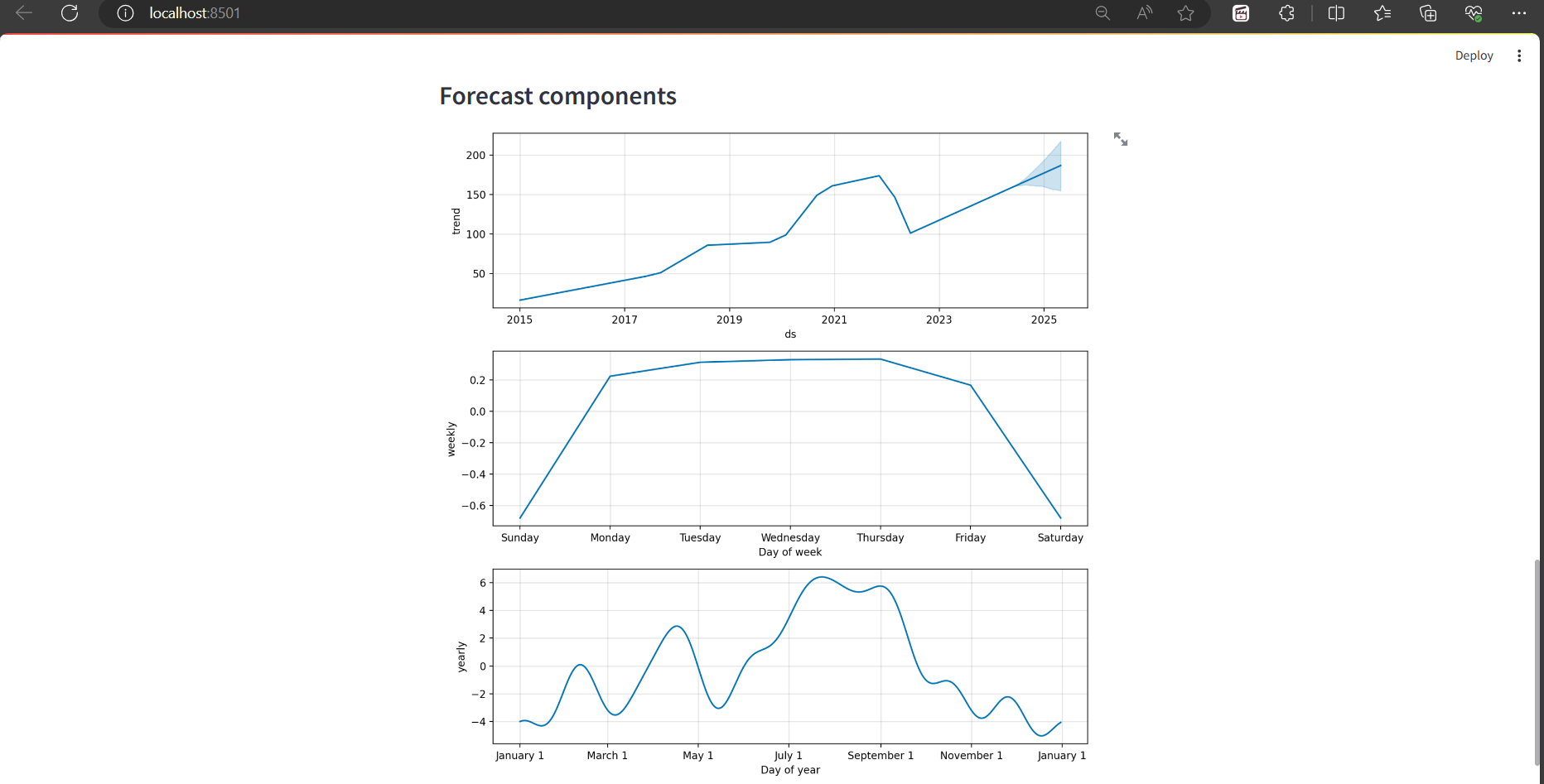
The ‘Close’ price of the stock is used to train the FBProphet model. The model is then used to make future predictions for the specified period. The forecasted data is displayed, which includes the forecasted ‘Close’ price of the stock over time and the components of the forecast.

****

5. Result Visualization

Finally, the application visualizes the forecasted data using Plotly. This includes the forecasted ‘Close’ price of the stock over time and the components of the forecast. The visualization provides a clear and intuitive understanding of the predicted stock prices.

****

****

**CHAPTER 7**

**RESULTS AND DISCUSSION**

7.1 Results

The primary objective of the project was to predict the future stock prices based on historical data. This objective was achieved using the FBProphet model. The model was trained on the ‘Close’ price of the stock and was able to make future predictions for the specified period.

The results of the project were visualized using Plotly. The visualization included the forecasted ‘Close’ price of the stock over time and the components of the forecast. The visualization provided a clear and intuitive understanding of the predicted stock prices.

The application also displayed the raw and forecasted data, providing users with detailed information about the stock prices. Users could input the stock symbol and the number of years for prediction, making the application interactive and user-friendly.

2. Discussion

The project provided valuable insights into the potential of time series forecasting in predicting stock prices. It demonstrated the effectiveness of the FBProphet model in handling time series data and making accurate predictions.

One of the key insights from the project was the importance of data visualization in understanding the results. The visualizations provided a clear picture of the stock price trends and the forecasted values, making it easier for users to interpret the results.

Another insight was the flexibility and ease of use of the Streamlit library in developing web applications. Streamlit allowed for quick and easy development of the application, with features like caching and widgets enhancing the user experience.

However, it’s important to note that while the model can predict the trend of stock prices with reasonable accuracy, it’s not guaranteed to be 100% correct. Stock prices are influenced by a multitude of factors, many of which cannot be accounted for in a historical data-based model. Therefore, the predictions made by the model should not be the sole basis for any investment decisions.

In conclusion, the project was successful in achieving its objective of predicting stock prices. It provided valuable insights into time series forecasting and demonstrated the potential of Python libraries in data science and machine learning applications.

**CHAPTER 8**

**CONCLUSION AND FUTURE WORK**

8.1 Conclusion

The project successfully developed a web application for predicting stock prices using Python, Streamlit, and FBProphet. The application is interactive and user-friendly, allowing users to input the stock symbol and the number of years for prediction. The FBProphet model was trained on the ‘Close’ price of the stock and was able to make future predictions for the specified period. The results were visualized using Plotly, providing a clear and intuitive understanding of the predicted stock prices.

The project demonstrated the effectiveness of time series forecasting in predicting stock prices and provided valuable insights into the potential of Python libraries in data science and machine learning applications. However, it’s important to note that the predictions made by the model should not be the sole basis for any investment decisions.

8.2 Future Work

While the project was successful, there are several improvements and extensions that could be made:

* Incorporating More Features: The current model uses only the ‘Close’ price of the stock for prediction. Incorporating more features, such as the ‘Open’, ‘High’, ‘Low’, and ‘Volume’ values, could potentially improve the accuracy of the predictions.
* Parameter Tuning: The default parameters provided by FBProphet were used in this project. Tuning these parameters could improve the model’s performance.
* Incorporating News Sentiment: Stock prices are often influenced by news sentiment. Incorporating news sentiment analysis could make the predictions more accurate.
* Expanding to Other Assets: The application could be expanded to include predictions for other assets, such as commodities, currencies, or cryptocurrencies.
* Deploying the Application: The application could be deployed on a platform like Heroku or AWS to make it accessible to a wider audience.

The next steps would be to implement these improvements and continue to evaluate and refine the model based on feedback and results.

**CHAPTER 9**

**REFERENCES**

1. **Python**: Python Software Foundation. Python Language Reference, version 3.7. Available at http://www.python.org
2. **Streamlit**: Streamlit Inc. Streamlit — The fastest way to build data apps. Available at https://www.streamlit.io/
3. **FBProphet**: Facebook Open Source. Prophet. Available at https://facebook.github.io/prophet/
4. **yfinance**: Ran Aroussi. yfinance. Available at https://pypi.org/project/yfinance/
5. **Plotly**: Plotly Technologies Inc. Collaborative data science. Available at https://plotly.com/python/
6. **Pandas**: The pandas development team. pandas-dev/pandas: Pandas. Available at https://github.com/pandas-dev/pandas
7. **Yahoo Finance**: Yahoo! Inc. Yahoo Finance - Stock Market Live, Quotes, Business & Finance News. Available at https://finance.yahoo.com/