

# **StockOracle**

*Harnessing Machine Learning for Smarter Investments and Market Insights*

**A PROJECT REPORT ON**  
**AI-Powered Stock Prediction & Forecasting System**

**Submitted to**

**KIIT Deemed to be University**

**In Partial Fulfilment of the Requirement for the Award of**

**BACHELOR'S DEGREE IN**  
**COMPUTER SCIENCE ENGINEERING**

**Neeladri Bandopadhyay 22052317**

**Lopamudra Mukherjee 22051524**

**Aritro Dutta 2205715**

**Harsh Kumar Singh 22052026**

# StockOracle

## Comprehensive Report on Stock Trend Prediction and Forecasting Application

---

## 1. Introduction

### 1.1 Overview

The stock market is a highly dynamic financial environment where accurate predictions can lead to profitable investment decisions. This report details a Python-based application that facilitates stock trend analysis and forecasting using machine learning techniques.

### 1.2 Purpose of the Application

The primary objective of this application is to provide users with insights into stock price movements by leveraging historical data. It aims to assist investors in making informed buy, hold, or sell decisions.

### 1.3 Technologies Used

The application is built using:

- **Dash** – For developing an interactive web-based dashboard.
  - **Plotly** – For visualizing stock price trends and forecasts.
  - **Pandas** – For data manipulation and preprocessing.
  - **Scikit-learn** – For implementing linear regression-based predictions.
  - **Prophet** – For time-series forecasting.
  - **Regular Expressions (Regex)** – For extracting relevant numerical data from text-based financial sources.
- 

## 2. Data Processing and Extraction

### 2.1 Importing Data

- The script loads stock data from `refined_textual_data.csv` (<https://in.docworkspace.com/d/sIOLCg7vDAeeg8b4G>).
- Data is processed to extract key details such as stock ticker symbols, closing prices, and corresponding dates.

### 2.2 Extracting Closing Prices

- A function `extract_price(text)` is implemented using regex to identify and retrieve stock closing prices embedded in textual financial data.
- Any records missing price details are removed from the dataset to maintain accuracy.

## 2.3 Handling Dates

- The application converts raw date strings into a `datetime` format, enabling time-series analysis.
  - The sorted dataset ensures that chronological trends are accurately captured.
- 

# 3. Stock Trend Prediction

## 3.1 Methodology




The application uses **Linear Regression**, a fundamental machine learning technique, to predict future stock trends.

## 3.2 Implementation Steps

- The function `predict_stock_trend(ticker)` selects stock data based on the user's chosen company ticker.
- The date column is transformed into numerical values representing the number of days since the earliest recorded entry.
- A **linear regression model** is trained on these transformed data points to estimate the future price trend.

## 3.3 Decision Criteria

The model predicts a stock's price for the next 30 days. Based on the forecast:

- If the price is projected to increase by more than **5%**, the stock is marked as **BUY** .
  - If the price is projected to decrease by more than **5%**, the stock is marked as **SELL** .
  - Otherwise, it is categorized as **HOLD** .
- 

# 4. Stock Price Forecasting

## 4.1 Forecasting with Prophet Model

- The function `forecast_stock(ticker, period, unit)` utilizes the **Prophet** model, developed by Facebook, for advanced time-series forecasting.
- The user specifies a forecasting period in **days, months, or years**.

- The model predicts future stock prices and confidence intervals.

## 4.2 Visualizing Forecasted Trends

The forecasted stock prices are plotted with:

- **Predicted Price Line (Blue):** Shows the estimated stock price over the selected period.
- **Upper Confidence Bound (Green, Dashed Line):** Represents the maximum expected stock price based on confidence intervals.
- **Lower Confidence Bound (Red, Dashed Line):** Represents the minimum expected stock price.

## 4.3 Forecast-Based Recommendations

- The application evaluates the predicted price against the latest stock price.
  - A **BUY**, **SELL**, or **HOLD** recommendation is made based on price changes beyond a 5% threshold.
- 

# 5. Interactive Dashboard and User Interface

## 5.1 Key Dashboard Components

The dashboard, built using **Dash**, allows users to:

- **Select a Company** – A dropdown menu enables users to filter stock data based on a specific ticker.
- **View Stock Trends** – A real-time graph displays historical stock price movements.
- **Forecast Future Prices** – Users input a forecast period and unit (days, months, years) to generate predictions.

## 5.2 Clickable Stock Points

- Clicking on a stock price point reveals sentiment information related to that date.
  - **Sentiment labels** such as "Optimistic", "Worried", or "Excited" are mapped to corresponding emojis.
- 

# 6. Sentiment Analysis and Market Insights

## 6.0 Considered Sentiment Emojis

The application uses the following emojis to represent market sentiment:

- Optimistic 🤗
- Worried 😬
- Anxious 😰
- Excited 🚀
- Disappointed 😞
- Confident 😎
- Indifferent 😐
- Uncertain 😕
- Calm 😌

## 6.1 Extracting Market Sentiment

- The dataset includes an `emo_label` column that categorizes stock market sentiments.
- Sentiments are mapped to visual emoji representations to provide a quick, intuitive understanding of market mood.

## 6.2 Importance of Sentiment in Trading

- **Positive Sentiment:** Often associated with bullish trends.
- **Negative Sentiment:** Can indicate potential stock price declines.
- **Neutral Sentiment:** Suggests market uncertainty or stability.

---

# 7. Advantages of the Model

## 7.1 Key Benefits

This stock trend prediction and forecasting model offers several advantages that enhance its effectiveness in market analysis and decision-making:

- **Data-Driven Decision Making** – Utilizes historical data and machine learning to make informed predictions.
  - **Combines Short-Term and Long-Term Analysis** – Uses **Linear Regression** for short-term trend prediction and **Prophet** for long-term forecasting.
  - **User-Friendly Interface** – A well-structured **Dash-based dashboard** makes stock analysis accessible for users with varying expertise.
  - **Sentiment Integration** – Incorporates **market sentiment analysis** using labeled sentiments and emojis to enhance prediction accuracy.
  - **Customizable Forecasting** – Users can select **different time periods** (days, months, years) to tailor predictions to their investment strategy.
  - **Interactive and Visual** – Provides intuitive **graphical representations** of stock trends, forecasts, and confidence intervals.
  - **Scalability** – Can be expanded with additional **financial indicators** like moving averages, volatility indices, and more advanced AI model
-

## 8. Performance Evaluation and Future Improvements

### 8.1 Current Strengths

- **User-Friendly Interface:** The dashboard simplifies stock market analysis for users of all levels.
- **Machine Learning Integration:** Uses statistical and AI-based models to provide data-driven insights.
- **Interactive Visualizations:** Provides clear and concise graphical representations of stock trends.

### 8.2 Potential Enhancements

- **Improved Prediction Models:** Incorporate deep learning models for better trend detection.
  - **Enhanced Sentiment Analysis:** Use **Natural Language Processing (NLP)** to extract sentiments from financial news and social media.
  - **Incorporate Additional Financial Indicators:** Add moving averages, volatility indices, and other key metrics for better predictions.
  - **Automated Data Ingestion:** Develop real-time data streaming functionality for up-to-the-minute stock tracking.
- 

## 9. Contribution

### 9.1 Task Distribution

To ensure efficient project execution, responsibilities were divided among four team members:

#### 1. Neeladri – Data Processing and Extraction

- Importing and cleaning stock data from `refined_textual_data.csv`.
- Extracting closing prices using regex.
- Converting and handling date formats for time-series analysis.
- Ensuring data consistency and accuracy.

#### 2. Arito – Stock Trend Prediction

- Implementing **Linear Regression** for stock trend prediction.
- Developing the `predict_stock_trend()` function.
- Setting up criteria for Buy, Sell, and Hold decisions.
- Evaluating the accuracy of trend predictions.

#### 3. Lopamudra – Stock Price Forecasting & Documentation

- Implementing the **Prophet** model for forecasting.
- Developing the `forecast_stock()` function.
- Creating visualization for forecasted stock prices.
- Designing logic for forecast-based recommendations.
- Writing detailed **documentation** for the project, including code comments and a structured README file.
- Report writing, formatting, and maintaining project records.

#### 4. Harsh – Dashboard, Sentiment Analysis, and UI Enhancements

- Developing the interactive user interface.
  - Implementing sentiment analysis with emoji mapping.
  - Handling user interactions and sentiment-based insights.
  - Optimizing UI for better user experience and interactivity.
- 

## 10. Conclusion

This application provides a **powerful and user-friendly solution** for stock market analysis, leveraging machine learning to offer valuable insights. By integrating predictive models, sentiment analysis, and interactive visualizations, it enables investors to make informed decisions efficiently.

The use of **linear regression and the Prophet model** ensures that both short-term trends and long-term forecasts are considered, enhancing decision-making accuracy. Additionally, the sentiment analysis feature allows users to gauge market emotions, which plays a crucial role in stock movements.

Overall, this project represents a significant step toward data-driven investment decision-making, offering practical tools that align with modern financial market needs.

---