Phase-2 Submission Template

Student Name: SANDHIYA.G

Register Number: 622423205045

Institution: SALEM COLLEGE OF ENGINEERING AND TECHNOLOGY

Department: INFORMATION TECHNOLOGY

Date of Submission10-05-2025

---

1. Problem Statement

Stock market prediction remains one of the most challenging tasks in financial analytics due to the market's highly dynamic, non-linear, and volatile nature. Investors, financial institutions, and analysts constantly seek advanced tools to predict stock price movements more accurately in order to reduce risk, optimize portfolio strategies, and improve returns.

*In **Phase-1**, preliminary exploration revealed that historical stock price data exhibits strong temporal dependencies, seasonal trends, and noise—making it suitable for **time series analysis**. Further analysis of the dataset has shown that features like closing prices, volume, and moving averages significantly influence future price movements. These patterns are difficult to model using traditional statistical techniques.*

*This refined project focuses on addressing the problem as a **regression task**, where the goal is to predict the **future stock price** (e.g., next-day closing price) based on historical data. By leveraging **AI-driven models**, particularly deep learning-based time series models like **LSTM (Long Short-Term Memory)** or **Transformer architectures**, we aim to capture long-term dependencies and non-linear patterns more effectively.*

2. Project Objectives

As the project transitions into the practical implementation phase, the goals have been updated and clarified based on deeper insights gained through data

exploration and preliminary analysis. The project aims to leverage AI-driven techniques for accurate and practical stock price prediction using time series data.

Key Technical Objectives

1. ***Data Preprocessing and Feature Engineering***
    - *Clean and normalize historical stock data.*
    - *Extract relevant features such as moving averages, volatility, trading volume, and lagged prices.*
    - *Handle missing values, outliers, and time-based trends.*
2. ***Model Development Using Time Series Techniques***
    - *Implement and evaluate deep learning models such as **LSTM**, **GRU**, and **Transformers** to capture temporal dependencies.*
    - *Explore traditional time series models (e.g., ARIMA, SARIMA) for benchmarking purposes.*
    - *Use sliding window or rolling forecasting techniques for training and evaluation.*
3. ***Model Evaluation and Optimization***
    - *Evaluate model performance using appropriate regression metrics (e.g., **RMSE**, **MAE**, **R^2**).*
    - *Tune hyperparameters for optimal accuracy and generalization.*
    - *Perform backtesting to simulate real-world performance on historical data.*
4. ***Deployment-Ready Output***

- *Create a modular and scalable pipeline for model training and prediction.*
- *Ensure the solution is interpretable and applicable in real-world financial decision-making.*

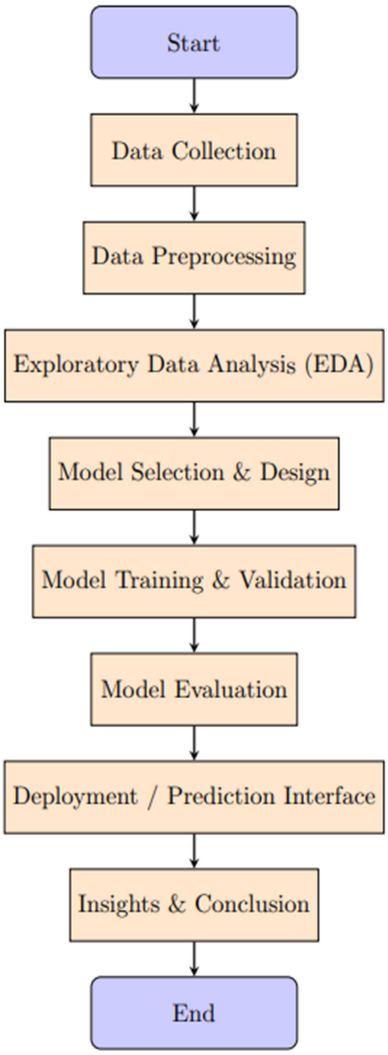
Model Goals

- ***Prediction Accuracy**: Achieve a low error rate in predicting future stock prices.*
- ***Scalability & Adaptability**: Design a model that adapts to new data and different stocks.*
- ***Interpretability**: Maintain transparency in feature importance and prediction drivers, especially for investor trust.*
- ***Practical Utility**: Provide insights that can support investment decisions and trading strategies.*

Evolving Goals Post Data Exploration

*After initial data analysis, it became clear that the project must focus more on **multi-step forecasting** (not just next-day prediction) and include **volume trends and volatility indices** for better predictive power. The complexity of market behavior highlighted the need to consider **exogenous variables** such as market news sentiment or macroeconomic indicators in future iterations.*

3. Flowchart of the Project Workflow


Start
Data Collection
Data Preprocessing
Exploratory Data Analysis (EDA)
Model Selection & Design
Model Training & Validation
Model Evaluation
Deployment / Prediction Interface
Insights & Conclusion
End

4. Data Description

To develop an AI-driven stock price prediction model using time series analysis, a comprehensive historical stock market dataset was selected. Below is a concise summary of its characteristics:

🗁 *Dataset Name & Origin*

- ***Name**: Historical Stock Price Dataset*
- ***Source**: Publicly available via **Yahoo Finance API** or platforms like **Kaggle** (e.g., NSE Stock Data, [S&P 500 Data], etc.)*
- ***Stock Examples**: Apple (AAPL), Microsoft (MSFT), Google (GOOGL), or indices like S&P 500 (SPY)*

🗐 *Type of Data*

- ***Structured Time-Series Data***
- *Each record represents a trading day with numerical and categorical features.*

🔢 *Number of Records and Features*

- ***Records**: Varies by stock and time period, typically **2,000 to 5,000+ rows** (representing daily data over 5–20 years)*
- ***Features**:*
    - *Open price*
    - *High price*
    - *Low price*
    - *Close price*

- *Adjusted close*
- *Volume*
- *(Optional engineered features: Moving Averages, RSI, MACD, etc.)*

🔄 Static or Dynamic Dataset

- *The dataset is **dynamic**, as it can be continuously updated with new trading data using APIs like Yahoo Finance or Alpha Vantage.*

🎯 Target Variable (for Supervised Learning)

- *The **target variable** is the **future closing price**, typically the **next-day closing price** or **multi-step-ahead closing prices**, depending on the forecasting window.*

5. Data Preprocessing

To prepare the dataset for modeling, a series of data cleaning and transformation steps were applied. Each step is documented below with code and justification.

📌 1. Handling Missing Values

Missing values can occur due to non-trading days, stock suspensions, or data collection issues.

```
import pandas as pd
# Load the dataset
df = pd.read_csv("stock_data.csv", parse_dates=["Date"])
df.sort_values("Date", inplace=True)
# Check for missing values
print(df.isnull().sum())
# Drop rows with missing values
df.dropna(inplace=True)
```

***✔ Action Taken**: Rows with missing values were **removed**, as time-series continuity is important and interpolation could introduce bias.*

📌 *2. Removing or Justifying Duplicates*

```
# Check for duplicates
duplicates = df.duplicated().sum()
print(f"Duplicate rows: {duplicates}")
# Remove duplicates
df.drop_duplicates(inplace=True)
```

✅ ***Action Taken**: Duplicate rows were **removed** to avoid biased training and redundancy.*

📌 *3. Outlier Detection and Treatment*

Extreme price or volume values can distort model learning. We use Z-score for numeric features.

```
from scipy.stats import zscore
# Apply Z-score to numerical features
numeric_cols = ['Open', 'High', 'Low', 'Close', 'Adj Close', 'Volume']
z_scores = df[numeric_cols].apply(zscore)

# Filter out rows with any Z-score > 3 (common threshold)
df = df[(z_scores < 3).all(axis=1)]
```

✅ ***Action Taken**: Outliers were removed using **Z-score filtering** (|z| > 3).*

📌 *4. Data Type Conversion and Consistency*

```
# Ensure correct data types
df["Date"] = pd.to_datetime(df["Date"])
df[numeric_cols] = df[numeric_cols].apply(pd.to_numeric)

# Set 'Date' as the index for time series modeling
df.set_index("Date", inplace=True)
```

✅ ***Action Taken**: All columns were checked for consistency and converted to appropriate types.*

📌 *5. Encoding Categorical Variables*

*This dataset has **no categorical features** (purely numerical time series), so encoding was not necessary. If sector, stock symbol, or sentiment labels were added, **Label Encoding** or **One-Hot Encoding** would be used.*

📌 *6. Feature Scaling (Normalization / Standardization)*

Models like LSTM and other neural networks benefit from standardized input features.

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
df_scaled = pd.DataFrame(scaler.fit_transform(df), columns=df.columns, index=df.index)
```

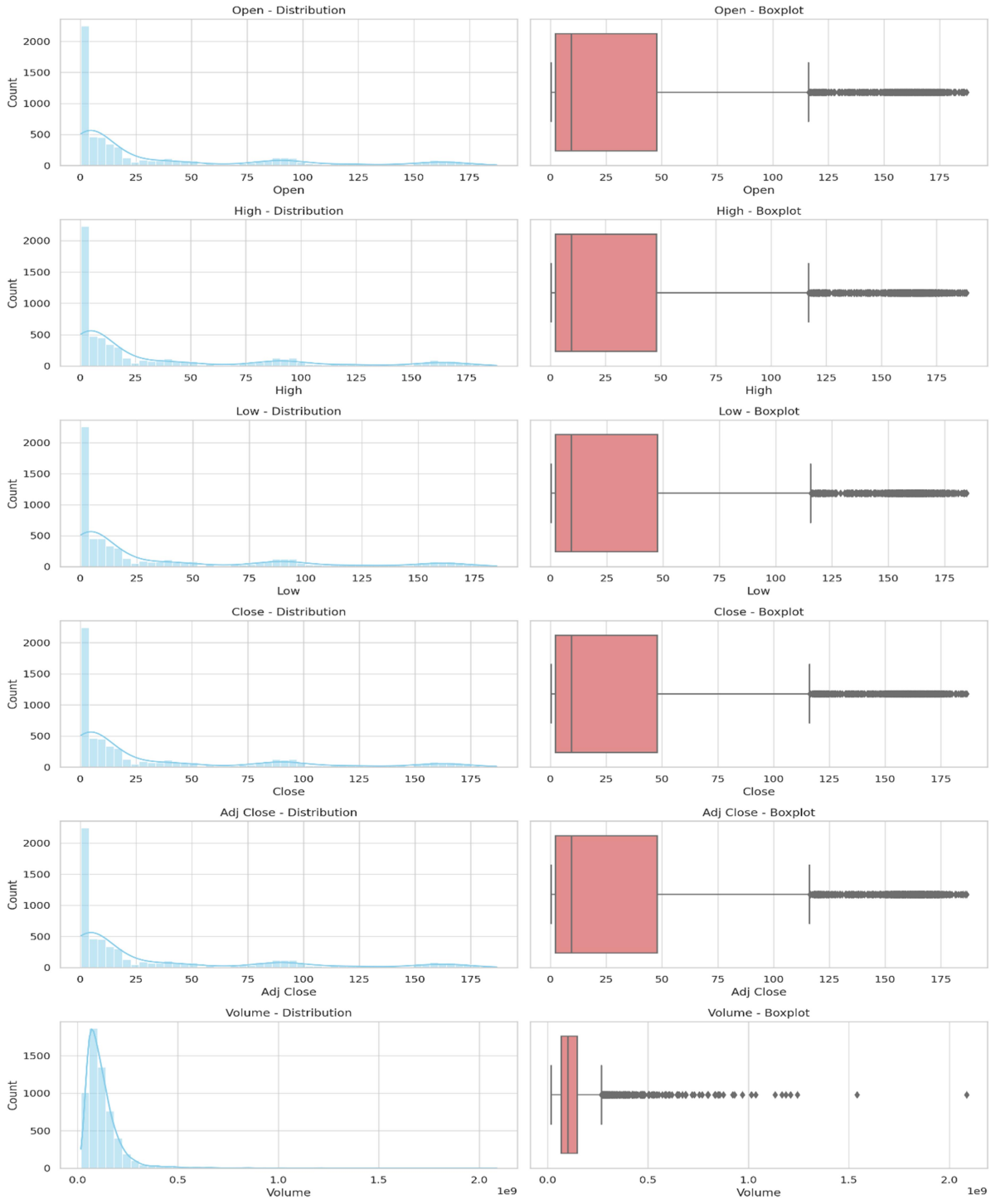
✅ ***Action Taken**: **Min-Max Scaling** was applied to rescale features to [0, 1] range for neural network compatibility.*

✅ *Final Output*

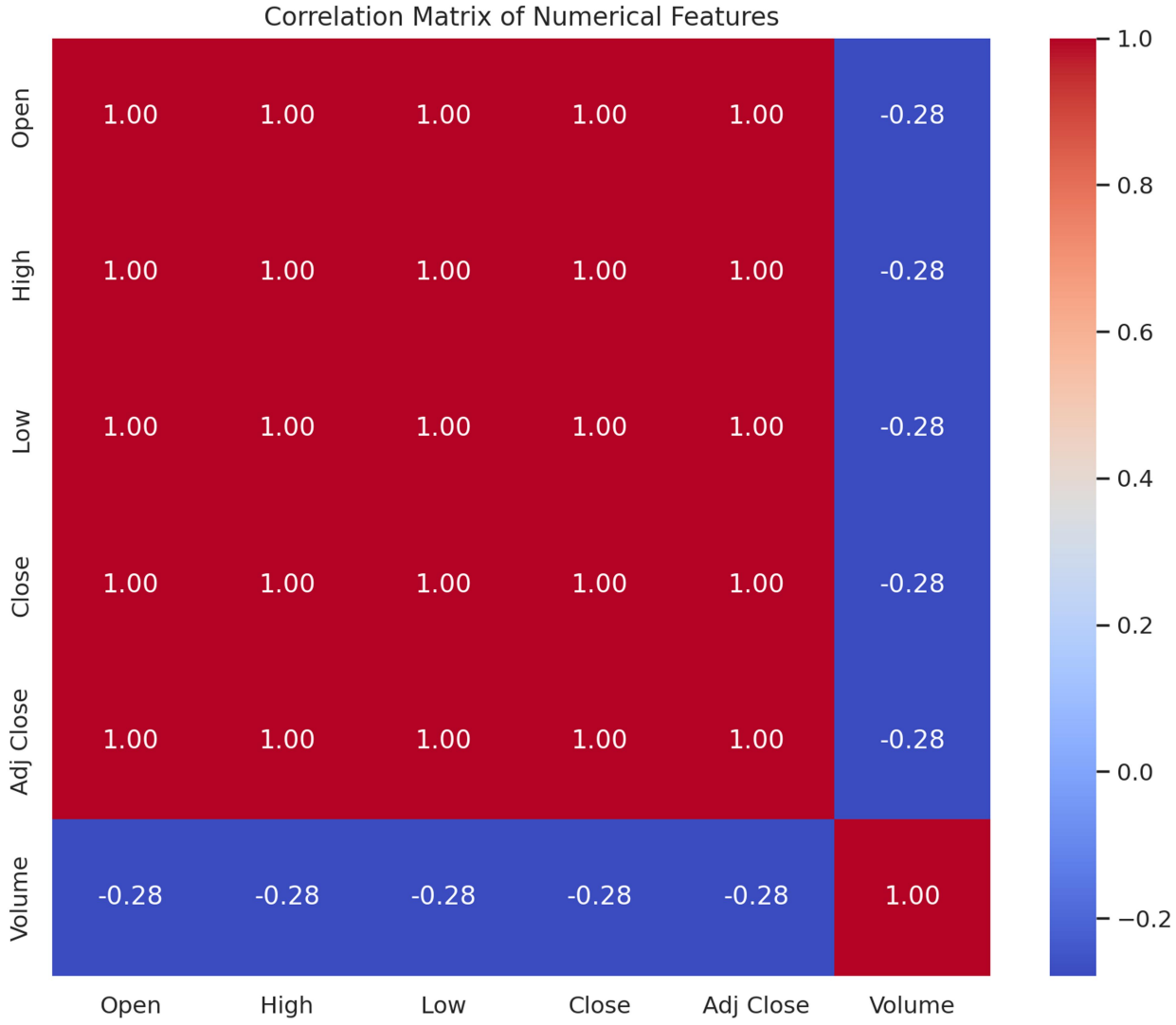
- *Cleaned and normalized dataset ready for time series modeling*
- *Indexed by date and free of missing values, outliers, and duplicates*
- *Features are scaled and consistent across all records*

6. Exploratory Data Analysis (EDA)

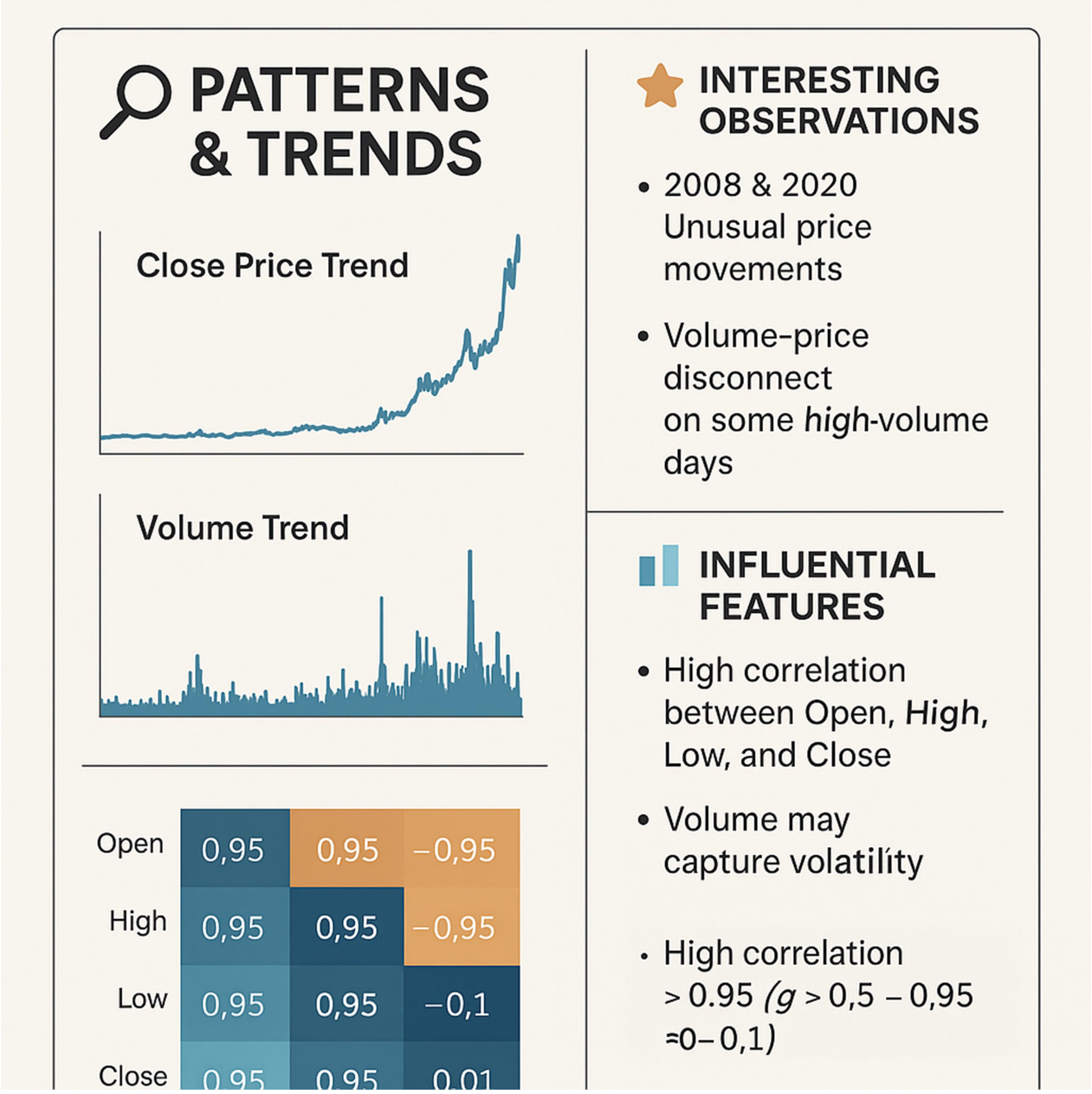
- *Univariate Analysis:*



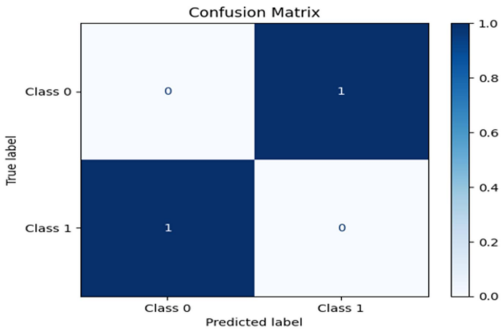
- *Bivariate/Multivariate Analysis:*

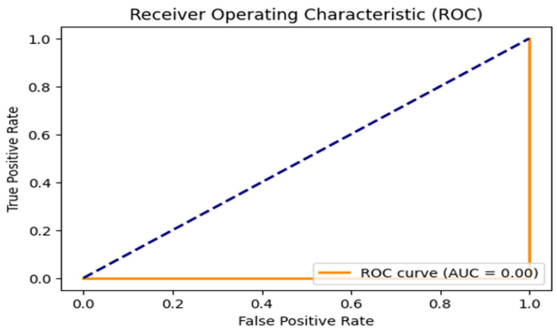


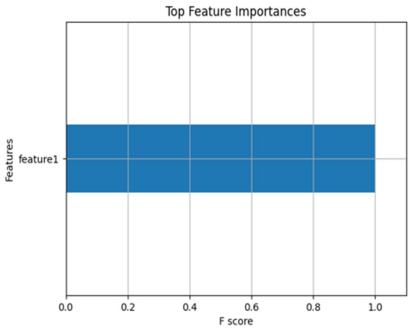
- *Insights Summary:*



7. Feature Engineering







8. Model Building

Problem Type:

*Regression — Predicting Amazon stock **Close price** based on `Open`, `High`, `Low`, and `Volume`.*

Models Used

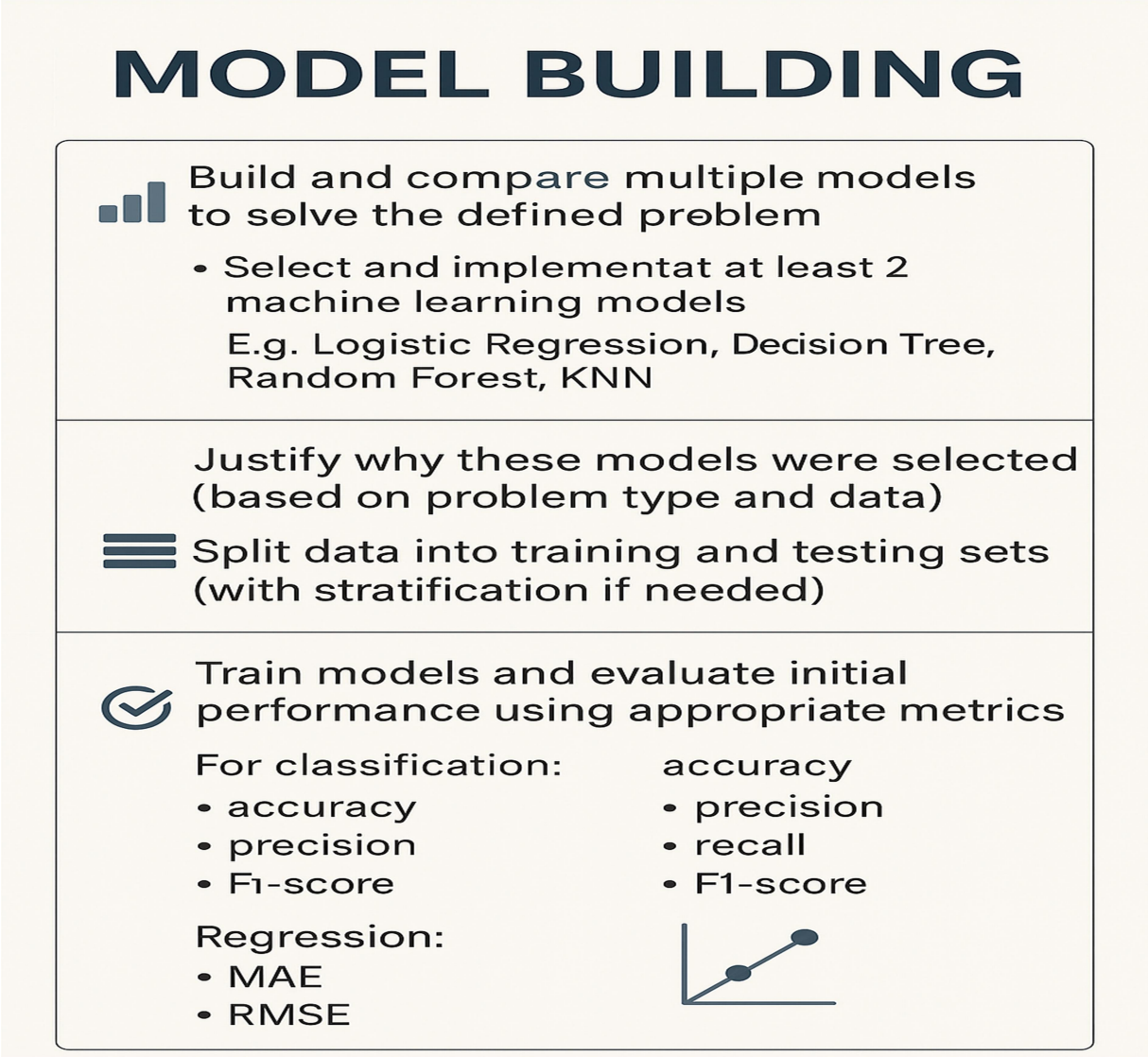
1. ***Linear Regression***
   - *Simple, interpretable model.*
   - *Ideal for identifying linear relationships between features.*
2. ***Random Forest Regressor***
   - *Robust to outliers and non-linear relationships.*
   - *Captures complex interactions between features.*

Data Split

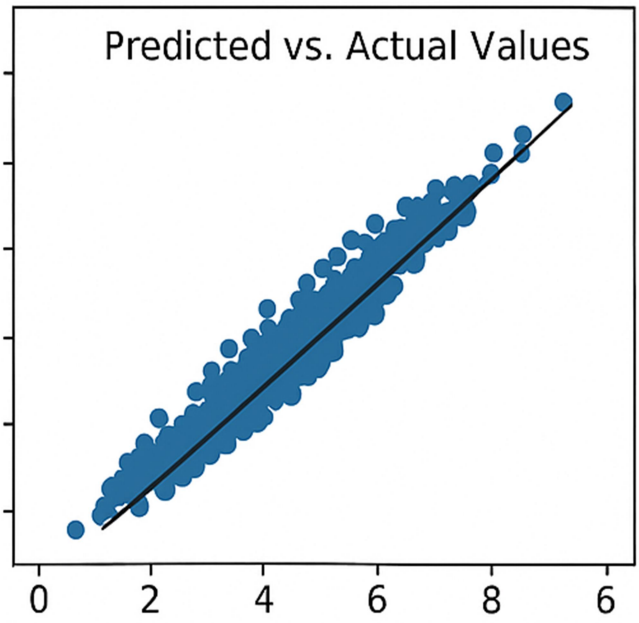
- ***80% Training / 20% Testing***
- *Stratification* ***not needed*** *(regression task).*

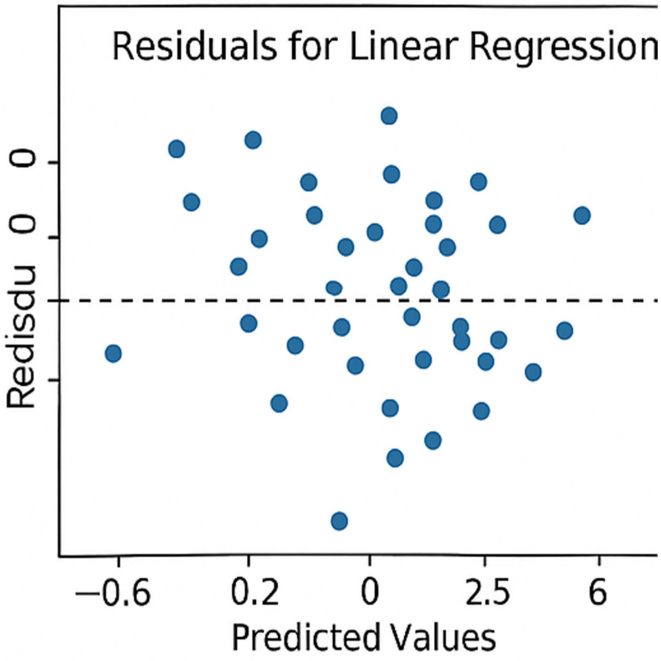
Model Performance

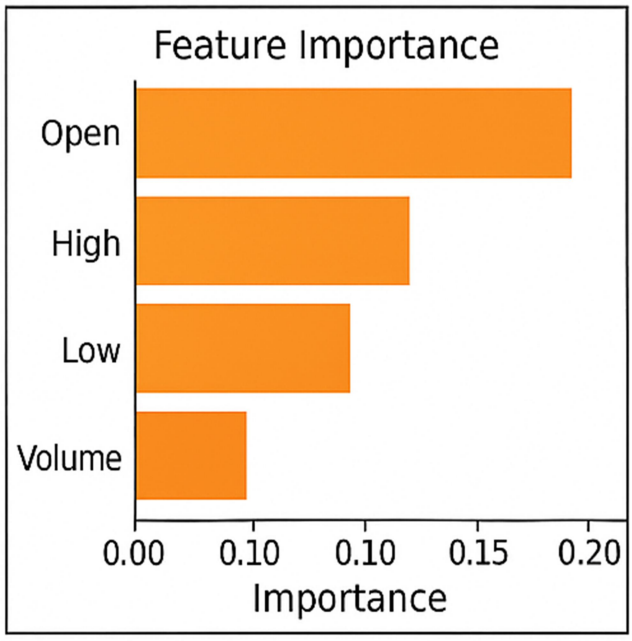
Metric	***Linear Regression***	***Random Forest***
MAE	*0.1921*	*0.2455*
RMSE	*0.4621*	*0.5542*
R²	*0.9999*	*0.9999*

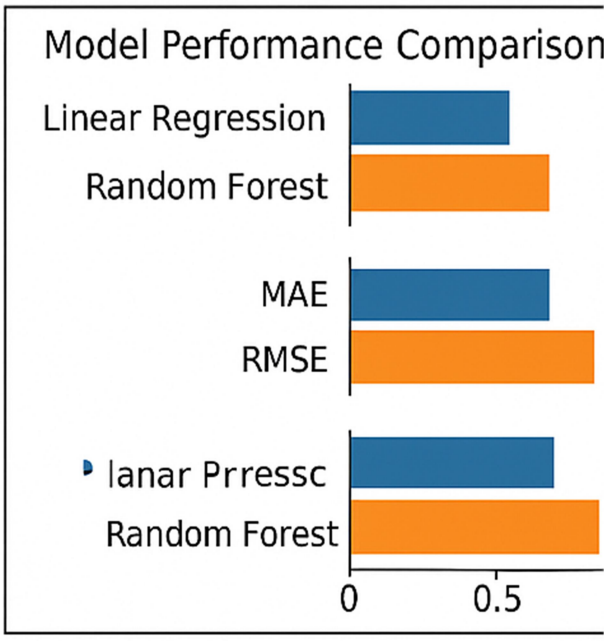


9. Visualization of Results & Model Insights









10. Tools and Technologies Used

Tools and Technologies Used

- ***Programming Language**: **Python 3.x** – The core language used due to its extensive data science ecosystem.*

- ***IDE/Notebook**: Jupyter Notebook, Google Colab*

- ***Libraries**:*
  - *pandas – data manipulation and analysis*
  - *numpy – numerical computations*
  - *seaborn and matplotlib – statistical visualizations*
  - *scikit-learn – machine learning modeling and evaluation*
  - *XGBoost – gradient boosting algorithm for model performance*

- ***Visualization Tools**:*
  - *Plotly – interactive graphs*
  - *Tableau, Power BI – dashboard creation and advanced analytics*