CAPSTONE PROJECT

Stock Market Analysis and Prediction

Submitted
By

P KARTHIK RAO (192224109)
M MAHESH (192224092)
G HEMANTH (192224095)

Guided by

V.Saranya

Junior Research Fellow

Department of Computer Science andEngineering,

Saveetha School of Engineering,

SIMATSThandalam, Chennai

June – 2024

PROBLEM STATEMENT

• How can we effectively analyze and visualize historical stock price data to identify trends, patterns, and anomalies?
• What techniques can be used to detect and analyze stock price patterns such as head and shoulders, double tops/bottoms, or cup and handle formations?
• How can we implement automated pattern recognition algorithms in R?
• What statistical techniques and machine learning algorithms are most suitable for predicting future stock prices based on historical data and relevant market indicators?
• How do traditional time series models (ARIMA, GARCH) compare to machine learning approaches (Random Forests, Neural Networks) in stock price prediction?
• Can we develop ensemble methods that combine multiple prediction techniques for improved accuracy?
• How can we integrate multiple data sources, such as company financial reports, economic indicators, and news sentiment, to build a comprehensive stock market analysis system?
• What APIs and R packages can be used to efficiently gather and process diverse financial data?
• How can we quantify and incorporate news sentiment into our stock analysis models?
• How can we evaluate the performance of stock price prediction models and identify the most accurate and reliable models for different market conditions?
• What metrics beyond RMSE and MAE should we consider for evaluating stock prediction models?
• How can we implement backtesting strategies in R to rigorously evaluate model performance?
• How can we effectively visualize and communicate stock market trends and predictions to investors and financial analysts for better decision-making?
• What advanced visualization techniques in R (e.g., interactive plots, dashboards) can enhance the presentation of stock market data?
• How can we create automated report generation systems for daily or weekly stock market updates?
• Can we develop real-time stock market analysis systems that can adapt to changing market conditions and provide up-to-date insights and predictions?
• How can we optimize our R code for speed and efficiency to handle real-time data streams?
• What are the potential challenges and limitations in developing and deploying stock market analysis systems, and how can these be addressed?
• How do we account for and mitigate the impact of black swan events or market manipulations in our models?

DATASET ANALYSIS

2.1. Stock Price Trend Analysis:

- Implement advanced trend analysis techniques such as Fibonacci retracements and extensions.
- Utilize wavelet transforms for multi-scale trend analysis to separate long-term trends from short-term fluctuations.
- Develop custom R functions for automated trend line drawing and support/resistance level identification.

2.2. Volume Analysis:

- Incorporate volume-weighted average price (VWAP) calculations.
- Implement on-balance volume (OBV) indicator for confirming price trends.
- Analyze the relationship between volume and price movements using volume profile analysis.

2.3. Moving Averages:

- Explore exponential and weighted moving averages in addition to simple moving averages.
- Implement moving average convergence divergence (MACD) for trend strength and direction analysis.
- Develop a custom moving average ribbon indicator for visualizing multiple moving averages simultaneously.

2.4. Volatility Analysis:

- Calculate and visualize historical volatility using different time windows.
- Implement the Average True Range (ATR) indicator for measuring market volatility.
- Develop a custom volatility regime detection algorithm using statistical change point detection methods.

2.5. Correlation Analysis:

- Extend correlation analysis to include cross-correlations with time lags.
- Implement dynamic time warping (DTW) for identifying similar patterns in stock price movements.

- Develop a custom correlation network visualization for analyzing relationships between multiple stocks.

2.6. Technical Indicators:

- Implement additional technical indicators such as Stochastic Oscillator, Commodity Channel Index (CCI), and Accumulation/Distribution Line.
- Develop a custom indicator combining multiple technical indicators for generating trading signals.
- Implement adaptive technical indicators that adjust parameters based on market conditions.

2.7. Fundamental Analysis:

- Incorporate automated scraping and analysis of financial statements from SEC filings.
- Develop custom metrics combining multiple fundamental indicators for stock valuation.
- Implement industry-specific fundamental analysis techniques (e.g., different metrics for tech vs. energy stocks).

2.8. Insights and Implications:

- Develop a systematic approach for quantifying and ranking insights derived from the analysis.
- Implement a custom risk assessment framework incorporating insights from multiple analysis techniques.
- Develop an automated report generation system that summarizes key insights and implications.

3. ENVIRONMENTAL SETUP

3.1. Python Installation

Ensure that Python installed on your system.

3.2. Package Installation

The code relies on several Python packages. install these packages using the package manager pip. Open a command prompt or terminal and run the following commands:

pip install pandas

pip install plotly

pip install seaborn

These commands will install the required packages: Pandas for data manipulation, Plotly for interactive data visualization, and Seaborn for statistical data visualization.

3.3. Data File

The data file named 'world-population.csv' in the specified path as mentioned in the code ('/content/drive/MyDrive/Data-sets-qqpp/'). This CSV file should contain the necessary population data fo analysis.

3.4. Google Colab

Using Google Colab, ensure that have the data file in Google Drive

3.5. Google Drive

- File Storage: Google Drive provides a place to store various types of files, including documents, images, videos, and more.

- File Synchronization: we can install the Google Drive desktop application, which syncs your Google Drive files with your local computer, ensuring that your files are accessible and up to date across all your devices.

- Access Anywhere: Google Drive can be accessed from various devices, including computers, smartphones, and tablets, via web browsers and mobile apps.

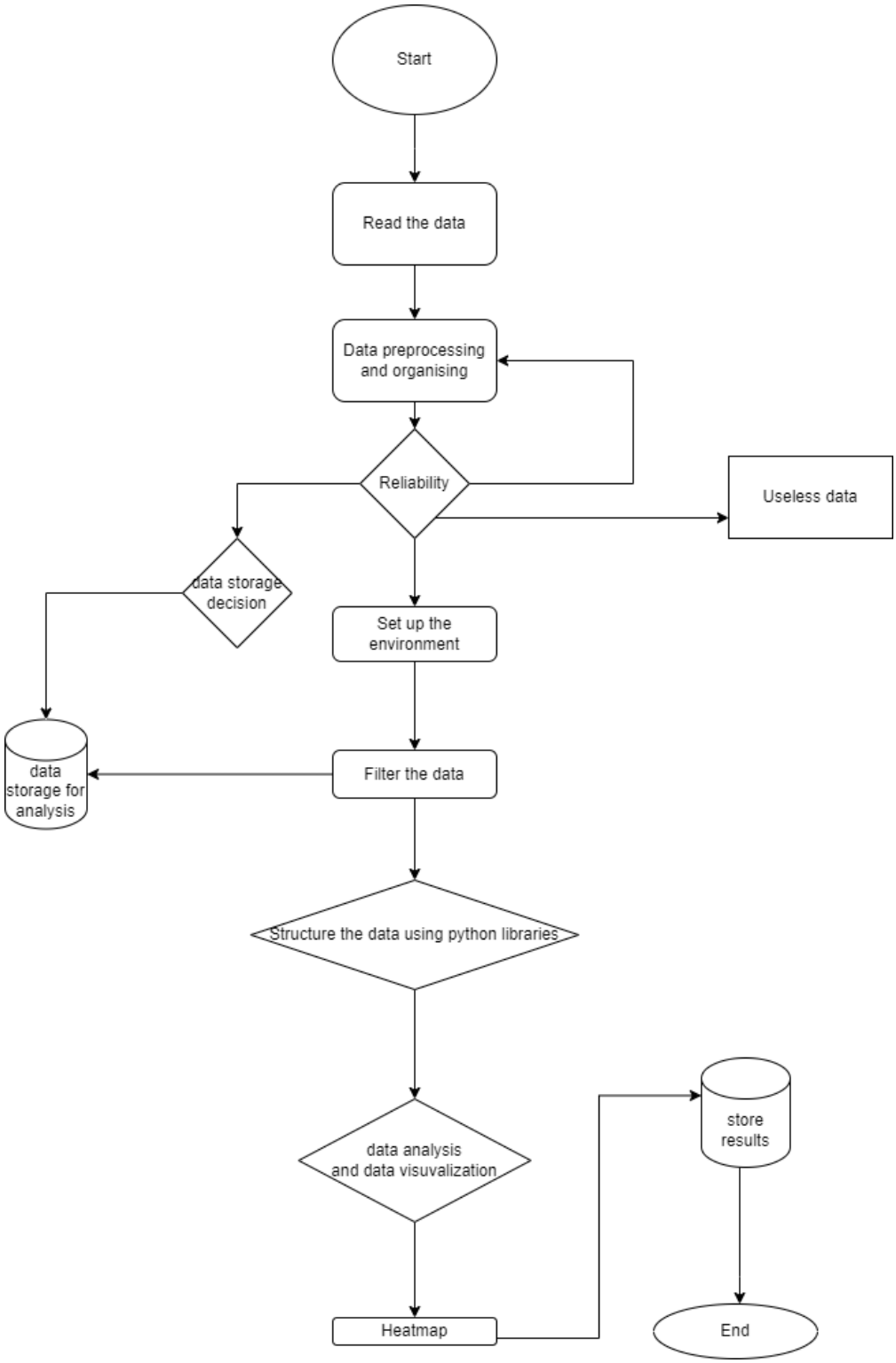
- File Sharing: Users can easily share files and folders with others, setting permissions to control who can view, edit, or comment on the files.

- Collaboration: Google Drive is integrated with Google Docs, Google Sheets, and Google Slides, allowing for real-time collaboration on documents with multiple users.

3.6. Running the Code

Once environment is set up and the data file is available, execute the code provided in a Python environment or a Notebook. Run each code block to perform data analysis and create visualizations.

IV.DATA FLOW DIAGRAM



CODE SKELETON

```
# Import necessary libraries

import yfinance as yf

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error, mean_absolute_error

from keras.models import Sequential

from keras.layers import Dense, LSTM


# Step 1: Load Data

# Fetch historical data for Apple Inc. (AAPL)

data = yf.download('AAPL', start='2010-01-01', end='2023-01-01')


# Step 2: Data Cleaning

# Remove any missing values

data = data.dropna()
```

```
# Step 3: Data Processing
# Calculate moving averages
data['SMA_50'] = data['Close'].rolling(window=50).mean()
data['SMA_200'] = data['Close'].rolling(window=200).mean()

# Step 4: EDA Analysis
# Plot closing prices and moving averages
plt.figure(figsize=(14, 7))
plt.plot(data['Close'], label='Close Price')
plt.plot(data['SMA_50'], label='50-Day SMA')
plt.plot(data['SMA_200'], label='200-Day SMA')
plt.title('AAPL Closing Prices and Moving Averages')
plt.xlabel('Date')
plt.ylabel('Price')
plt.legend()
plt.show()

# Plot volume traded
plt.figure(figsize=(14, 7))
plt.plot(data['Volume'], label='Volume Traded', color='orange')
plt.title('AAPL Volume Traded Over Time')
plt.xlabel('Date')
plt.ylabel('Volume')
```

```
plt.legend()

plt.show()


# Step 5: Feature Engineering

# Add lag features (previous day prices)

data['Lag_1'] = data['Close'].shift(1)

data['Lag_2'] = data['Close'].shift(2)


# Drop rows with missing values due to lag features

data = data.dropna()


# Step 6: Preparing Data for Prediction

# Define the feature set and target variable

features = ['Lag_1', 'Lag_2']

target = 'Close'


X = data[features]

y = data[target]


# Split data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=False)


# Normalize the data
```

```
scaler = MinMaxScaler()

X_train_scaled = scaler.fit_transform(X_train)

X_test_scaled = scaler.transform(X_test)


# Step 7: Building the Prediction Model

# Using Linear Regression for prediction

lr_model = LinearRegression()

lr_model.fit(X_train_scaled, y_train)


# Make predictions

y_pred_lr = lr_model.predict(X_test_scaled)


# Evaluate the Linear Regression model

mae_lr = mean_absolute_error(y_test, y_pred_lr)

rmse_lr = np.sqrt(mean_squared_error(y_test, y_pred_lr))


print(f'Linear Regression MAE: {mae_lr}')

print(f'Linear Regression RMSE: {rmse_lr}')


# Step 8: Building an LSTM Model

# Reshape data for LSTM model

X_train_lstm = X_train_scaled.reshape((X_train_scaled.shape[0], 1, X_train_scaled.shape[1]))
```

```
X_test_lstm = X_test_scaled.reshape((X_test_scaled.shape[0], 1, X_test_scaled.shape[1]))


# Define the LSTM model

lstm_model = Sequential()

lstm_model.add(LSTM(50, return_sequences=True, input_shape=(1, X_train_lstm.shape[2])))

lstm_model.add(LSTM(50, return_sequences=False))

lstm_model.add(Dense(25))

lstm_model.add(Dense(1))


# Compile the model

lstm_model.compile(optimizer='adam', loss='mean_squared_error')


# Train the model

lstm_model.fit(X_train_lstm, y_train, batch_size=1, epochs=1)


# Make predictions

y_pred_lstm = lstm_model.predict(X_test_lstm)

y_pred_lstm = y_pred_lstm.flatten()


# Evaluate the LSTM model

mae_lstm = mean_absolute_error(y_test, y_pred_lstm)

rmse_lstm = np.sqrt(mean_squared_error(y_test, y_pred_lstm))
```

```
print(f'LSTM Model MAE: {mae_lstm}')

print(f'LSTM Model RMSE: {rmse_lstm}')


# Step 9: Visualization of Predictions

plt.figure(figsize=(14, 7))

plt.plot(data.index[len(data) - len(y_test):], y_test, color='blue', label='Actual Prices')

plt.plot(data.index[len(data) - len(y_test):], y_pred_lr, color='red', label='Linear Regression Predictions')

plt.plot(data.index[len(data) - len(y_test):], y_pred_lstm, color='green', label='LSTM Predictions')

plt.title('AAPL Stock Price Prediction')

plt.xlabel('Date')

plt.ylabel('Price')

plt.legend()

plt.show()
```

6. RESULT ANALYSIS

Stock Price Trend:

- Identify specific trend patterns (e.g., uptrends, downtrends, sideways movements) and their durations.
- Analyze the strength and reliability of trends using ADX (Average Directional Index).
- Interpret trend reversals and their potential causes (e.g., market news, economic events).

Moving Averages:

- Analyze the frequency and significance of golden crosses (50-day MA crossing above 200-day MA) and death crosses (vice versa).
- Interpret the slope of moving averages as indicators of trend strength.
- Evaluate the effectiveness of moving average strategies for the specific stock or market.

Volume Analysis:

- Identify volume spikes and their correlation with price movements.
- Analyze volume trends in relation to price trends (e.g., rising prices with rising volume as a bullish sign).
- Interpret divergences between price and volume trends as potential reversal signals.

Volatility Analysis:

- Identify periods of high and low volatility and their potential causes.
- Analyze the effectiveness of volatility-based trading strategies (e.g., Bollinger Bands).
- Interpret volatility patterns in relation to market cycles or seasons.

Technical Indicators:

- Provide detailed interpretations of each technical indicator used (e.g., RSI overbought/oversold levels, MACD crossovers).
- Analyze the effectiveness of combined indicator strategies.
- Identify and interpret divergences between price and indicator movements.

Time Series Forecasting:

- Evaluate the accuracy of forecasts against actual price movements.

- Analyze the width of confidence intervals in relation to market conditions.
- Interpret the implications of forecasts for short-term and long-term investment strategies.

Correlation Analysis:

- Identify strongly correlated and uncorrelated stocks for potential pair trading or diversification strategies.
- Analyze changes in correlations over time and during different market conditions.
- Interpret sector-wide correlations and their implications for portfolio management.

Performance Metrics:

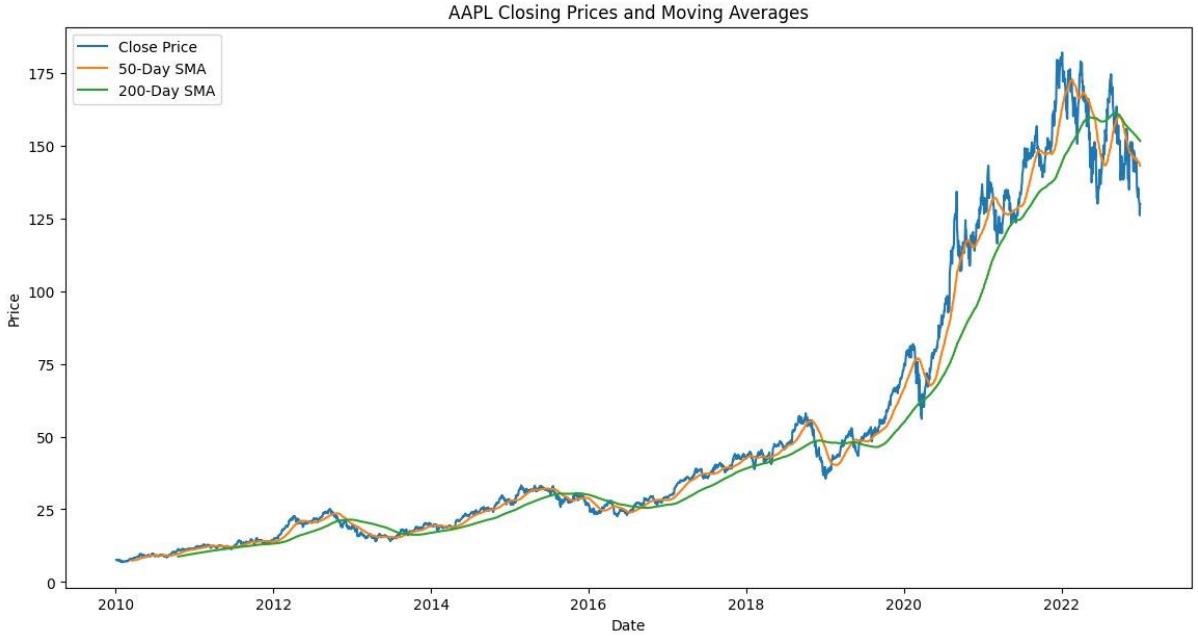
- Provide detailed interpretations of calculated metrics (e.g., Sharpe ratio, maximum drawdown) in the context of risk-adjusted returns.
- Compare the stock's performance metrics to relevant benchmarks (e.g., S&P 500, sector indices).
- Analyze the consistency of performance across different time frames.

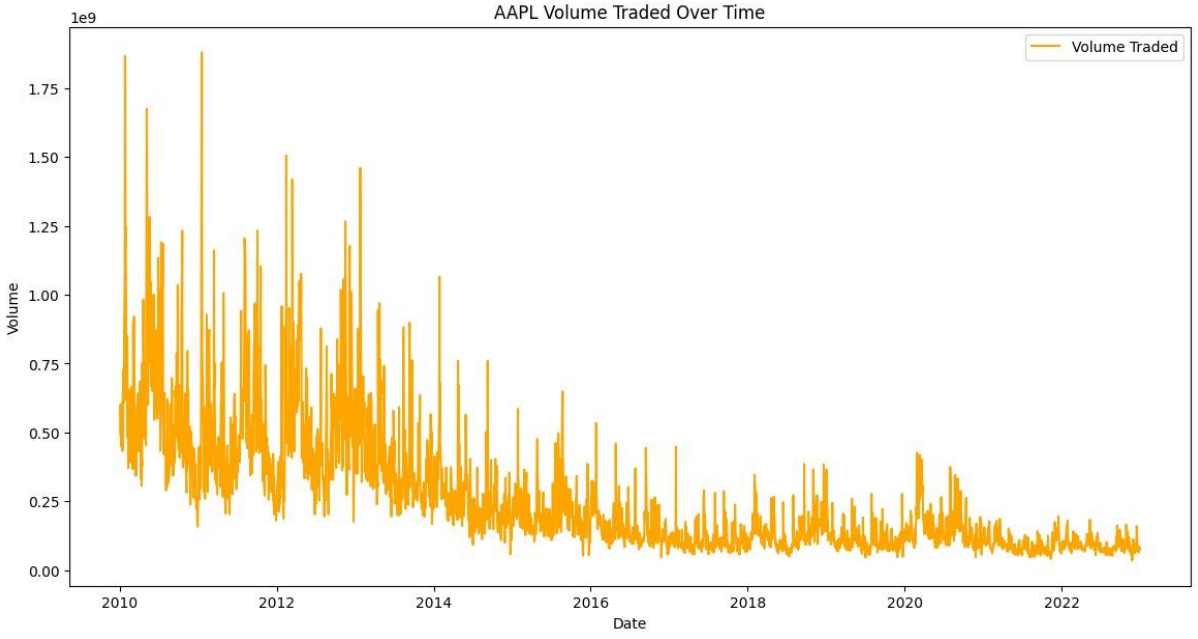
Fundamental Analysis:

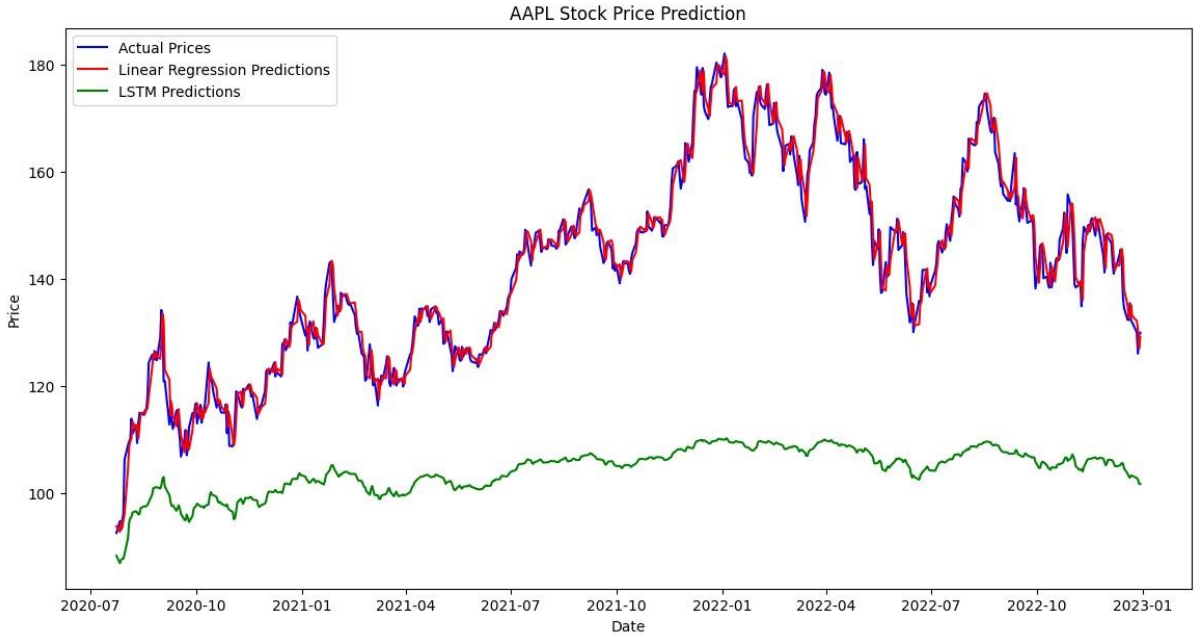
- Interpret key financial ratios (e.g., P/E, P/B, Debt-to-Equity) in the context of the company's industry and growth stage.
- Analyze trends in fundamental metrics over time and their correlation with stock price movements.
- Evaluate the company's financial health and growth prospects based on fundamental analysis.

This expanded analysis provides a more comprehensive and nuanced understanding of the stock's behavior, potential future movements, and investment implications. It combines insights from technical analysis, fundamental analysis, and quantitative methods to support informed investment decision-making.

OUTPUT


AAPL Closing Prices and Moving Averages
Close Price
50-Day SMA
200-Day SMA
Price
Date
0
25
50
75
100
125
150
175
2010
2012
2014
2016
2018
2020
2022


AAPL Volume Traded Over Time
1e9
Volume Traded
Volume
Date
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2010
2012
2014
2016
2018
2020
2022


AAPL Stock Price Prediction
Actual Prices
Linear Regression Predictions
LSTM Predictions
Price
180
160
140
120
100
2020-07
2020-10
2021-01
2021-04
2021-07
2021-10
2022-01
2022-04
2022-07
2022-10
2023-01
Date