|  |  |  |
| --- | --- | --- |
| **S.No** | **Contents** | **Pg.no** |
| **1** | **Objective and Scope** | **2** |
| **2** | **Data Collection** | **3** |
| **3** | **Data preprocessing** | **4** |
| **4** | **Exploratory Data Analysis** | **5** |
| **5** | **Technical Analysis** | **8** |
| **6** | **Time Series Forecasting** | **11** |
| **7** | **Portfolio Optimization** | **12** |
| **8** | **Conclusion** | **14** |
| **9** | **FutureWork** | **14** |
| **10** | **External References and Acknowledgement** | **14** |
|  |  |  |

**Stock Market Performance Analysis**

**STOCK MARKET PERFORMANCE ANALYSIS**

Name: DHIVYA BHARATHI K BatchID:21281 RollNo.:21281003

Stock Market Performance Analysis involves calculating moving averages, measuring volatility, conducting correlation analysis and analyzing various aspects of the stock market to gain a deeper understanding of the factors that affect stock prices and the relationships between the stock prices of different companies.

**Stock Market Performance Analysis using Python**

Let’s start the task of Stock Market Performance Analysis by importing the necessary Python libraries and the dataset. For this task, I will use the Yahoo finance API (yfinance) to collect real-time stock market data for the past three months.

It’s important to collect real-time data for this task, but still, if you are a complete beginner and want a dataset only to practice the concepts covered in this article, you can download the dataset from here. But it’s recommended to use the yfinance API to collect and work on real-time data.

You can install the yfinace API in your Python environment using the pip command mentioned below (run the command below on your command prompt or terminal):

for command prompt or terminal: pip install yfinance  
for Google Colab or Jupyter notebooks: !pip install yfinance

**Background on stock markets**

Stock markets play a pivotal role in the global economy. They represent a place where companies can raise capital by issuing shares to the public, and investors can buy or sell these shares. Understanding stock markets is essential for anyone interested in investment or finance.

**1.Objective and scope**

In this project, we aim to harness the power of Python to analyze stock market performance. By the end, you'll understand how stock prices move, identify trends, and even develop potential investment strategies.

**2.Data Collection**

Fetching stock data:

To start any stock market analysis, we need data. Python offers several libraries to fetch this data seamlessly.

**Installing necessary libraries:**

To begin, we need to install yfinance and pandas\_datareader. These libraries allow us to pull stock data easily.

!pip install yfinance pandas\_datareader

**Fetching historical data:**

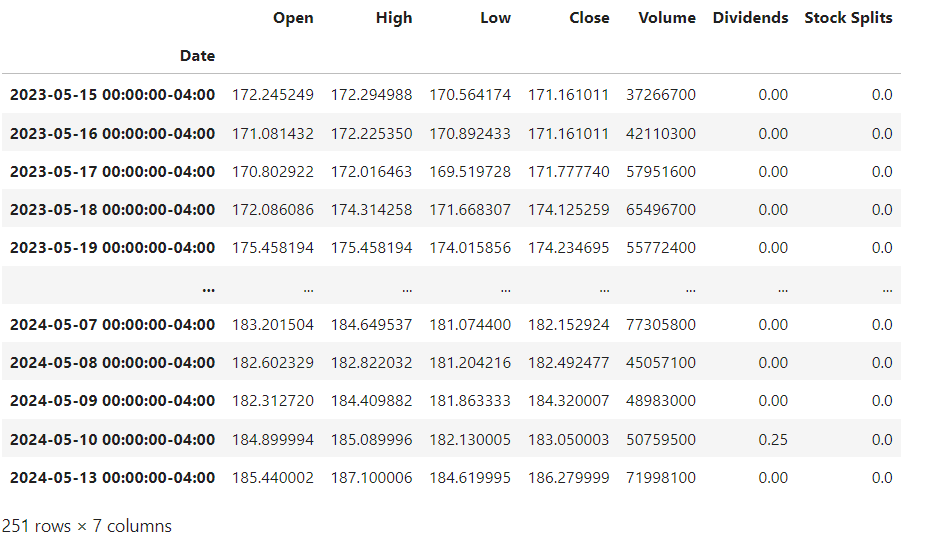
yfinance makes it easy to get stock data. In the example below, we're fetching data for Apple Inc. (AAPL) for the past year.

import yfinance as yf

stock = yf.Ticker("AAPL") # Using Apple's stock symbol

data = stock.history(period="1y") # Fetching data for 1 year

data



**Choosing stocks for analysis:**

For a comprehensive analysis, it's advisable to select stocks from various sectors. For instance, consider tech stocks (AAPL, MSFT), financials (JPM), and consumer goods (PEP). This diversification gives a broader view of the market.

**3.Data Preprocessing**

Before diving into analysis, we need to ensure our data is clean and in the right format.

**Handling missing values:**

It's common to encounter missing values in stock data. We need to address these to avoid errors in our analysis.

data.dropna(inplace=True)

**Ensure date format is correct:**

For time series analysis, it's crucial that our date column is in the datetime format.

import pandas as pd

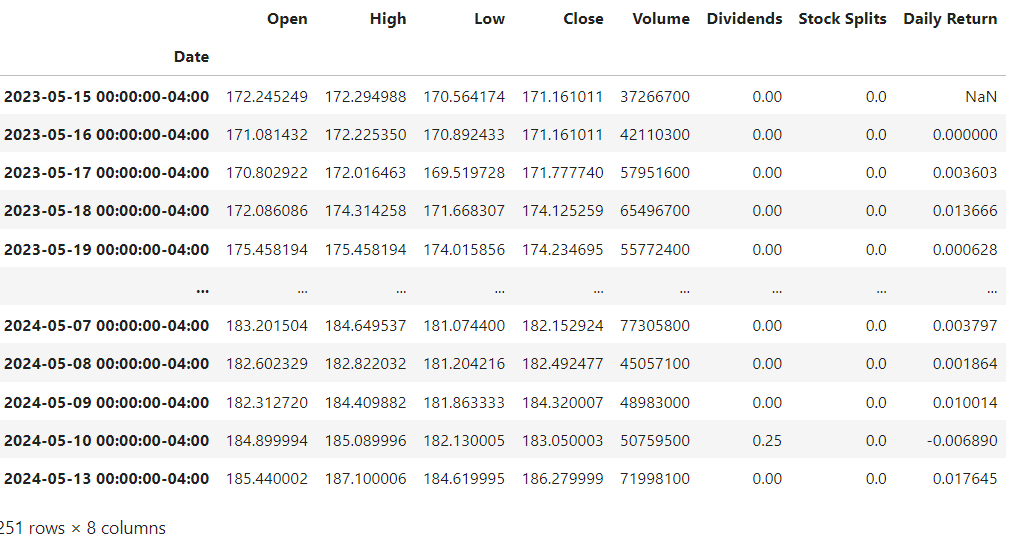
data.index = pd.to\_datetime(data.index)

**Calculate daily returns:**

Daily returns provide insights into the stock's volatility. It's calculated as the percentage change in the stock's price from the previous day.

data['Daily Return'] = data['Close'].pct\_change()

data



**4.Exploratory Data Analysis (EDA)**

EDA helps us understand the nature and structure of our data. It's the first step in identifying trends or anomalies.

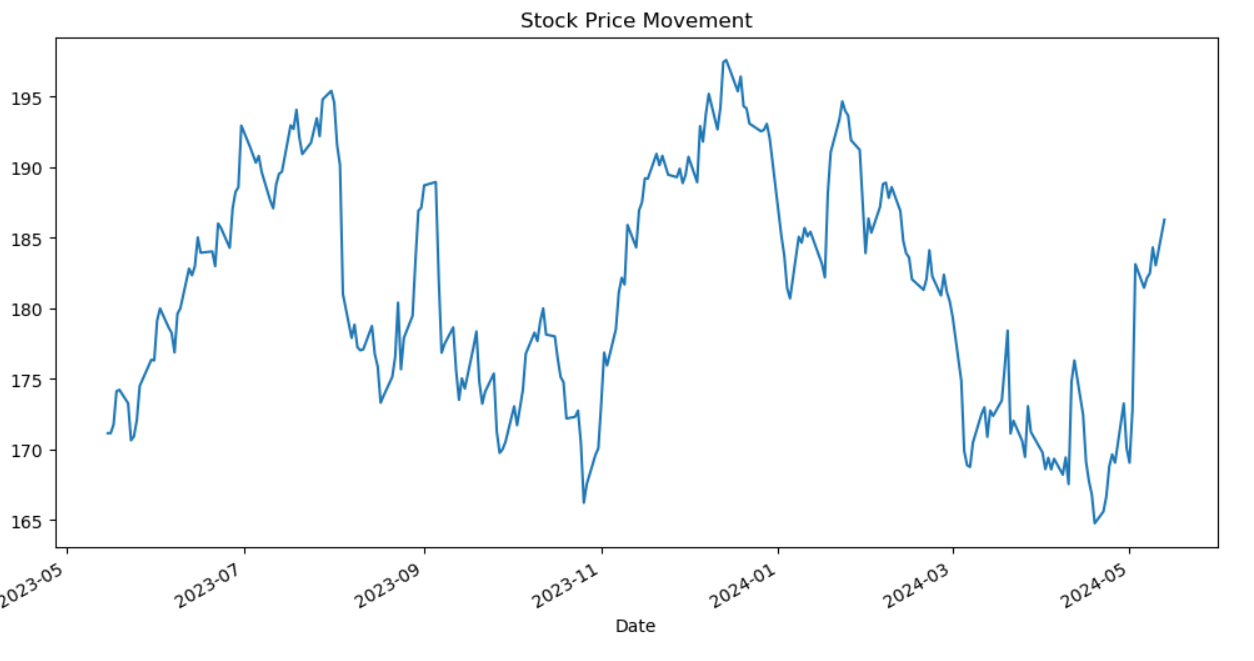
Visualize stock price movements:

A simple line plot can show us how the stock's closing price has moved over time.

import matplotlib.pyplot as plt

data['Close'].plot(figsize=(12, 6), title="Stock Price Movement")

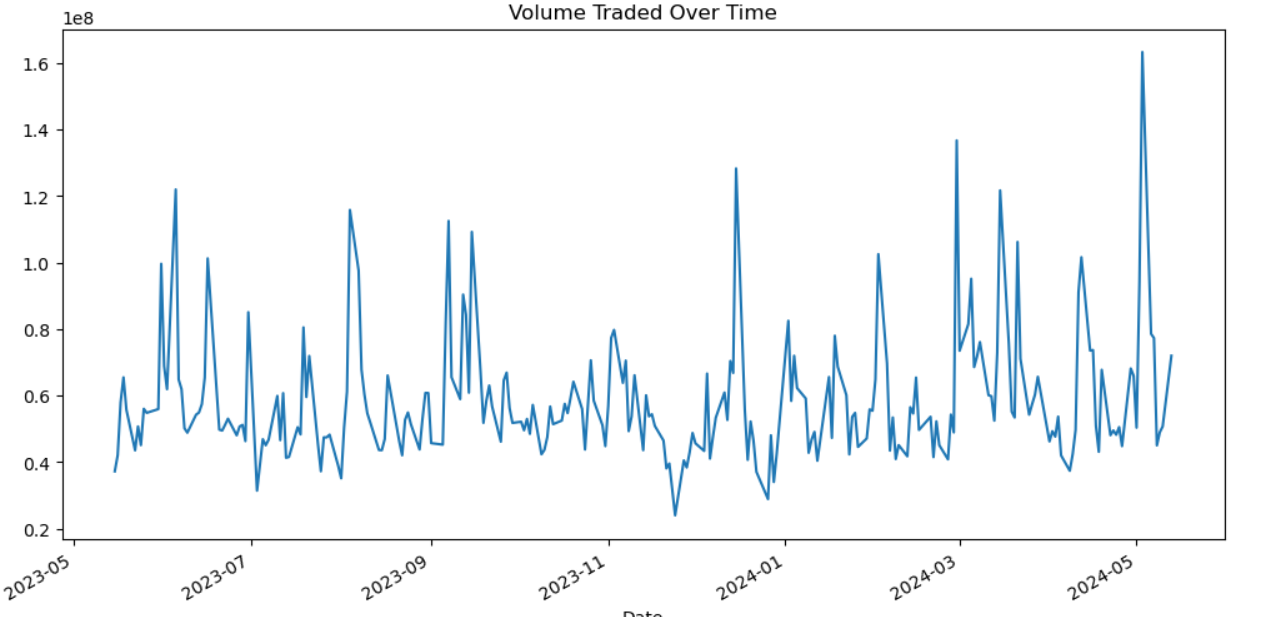
plt.show()



**Analyze volume of stocks traded:**

Volume indicates the number of shares traded in a given period. High volume can suggest significant news or events affecting the stock.

data['Volume'].plot(figsize=(12, 6), title="Volume Traded Over Time")  
plt.show()

****

**Correlation between stocks:**

When analyzing multiple stocks, it's essential to see how they move in relation to one another. A heatmap can visualize this relationship.

import seaborn as sns

stock\_list = ["AAPL", "MSFT", "JPM", "PEP", "GOOGL"]

close\_prices = pd.DataFrame()

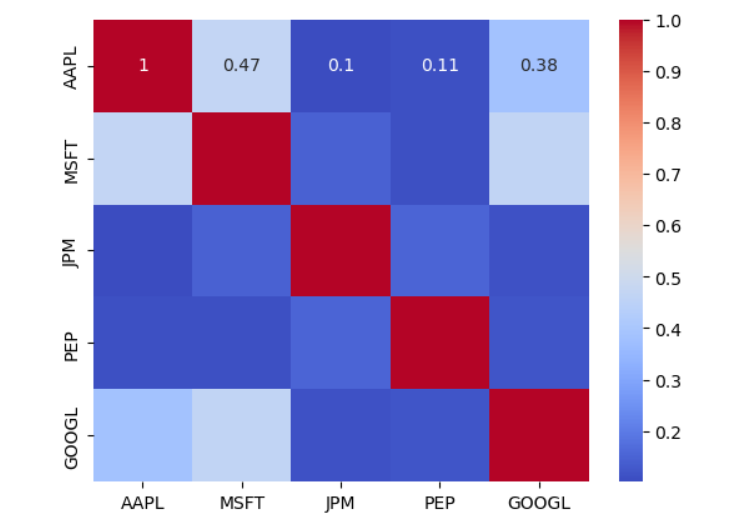
for stock in stock\_list:

close\_prices[stock] = yf.Ticker(stock).history(period="1y")['Close']

correlation = close\_prices.pct\_change().corr()

sns.heatmap(correlation, annot=True, cmap='coolwarm')

plt.show()



***Note: Correlation values range between -1 and 1. A value closer to 1 implies that two stocks move in tandem, while a value closer to -1 indicates they move in opposite directions.***

1. **Technical Analysis**

Technical analysis involves studying past market data, primarily price and volume, to forecast future price movements. This analysis can be performed on any security with historical trading data.

Moving Averages:

A moving average smoothens price data to create a single flowing line, which makes it easier to identify the direction of the trend. The two most common types of moving averages are the Simple Moving Average (SMA) and the Exponential Moving Average (EMA).

**Simple Moving Average (SMA):**

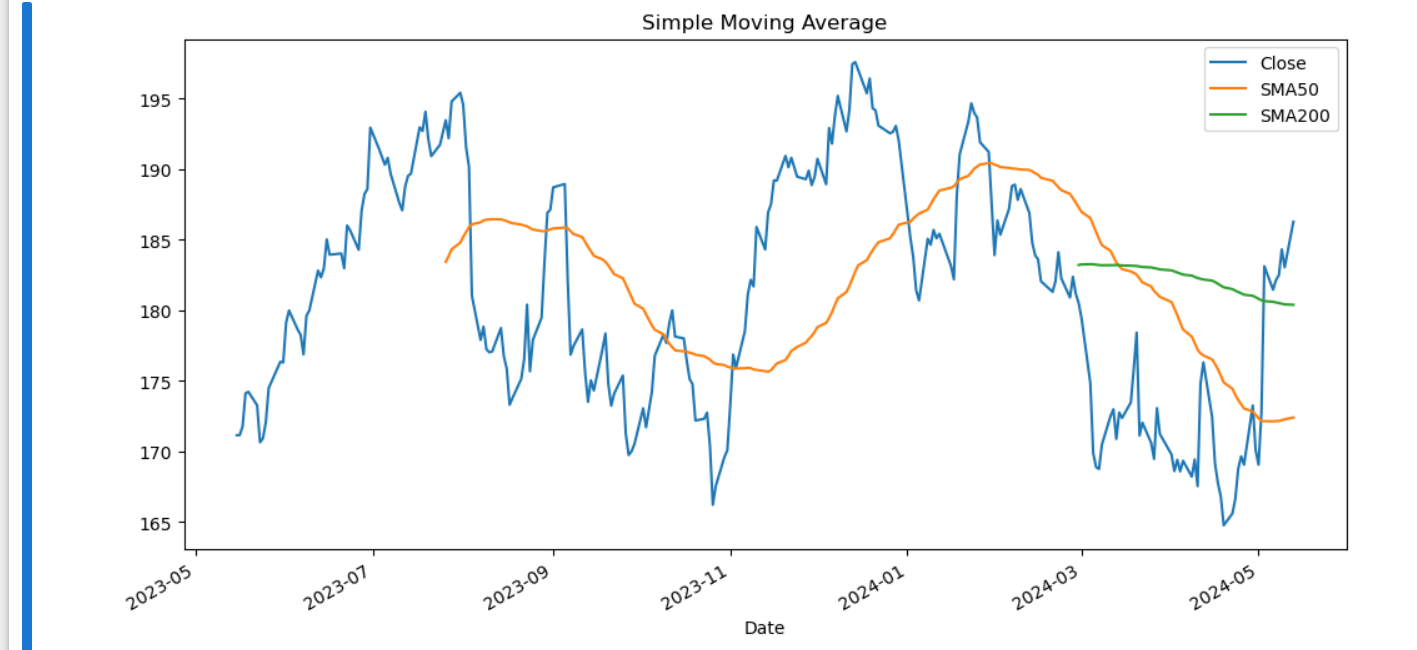
data['SMA50'] = data['Close'].rolling(window=50).mean()

data['SMA200'] = data['Close'].rolling(window=200).mean()

data[['Close', 'SMA50', 'SMA200']].plot(figsize=(12,6))

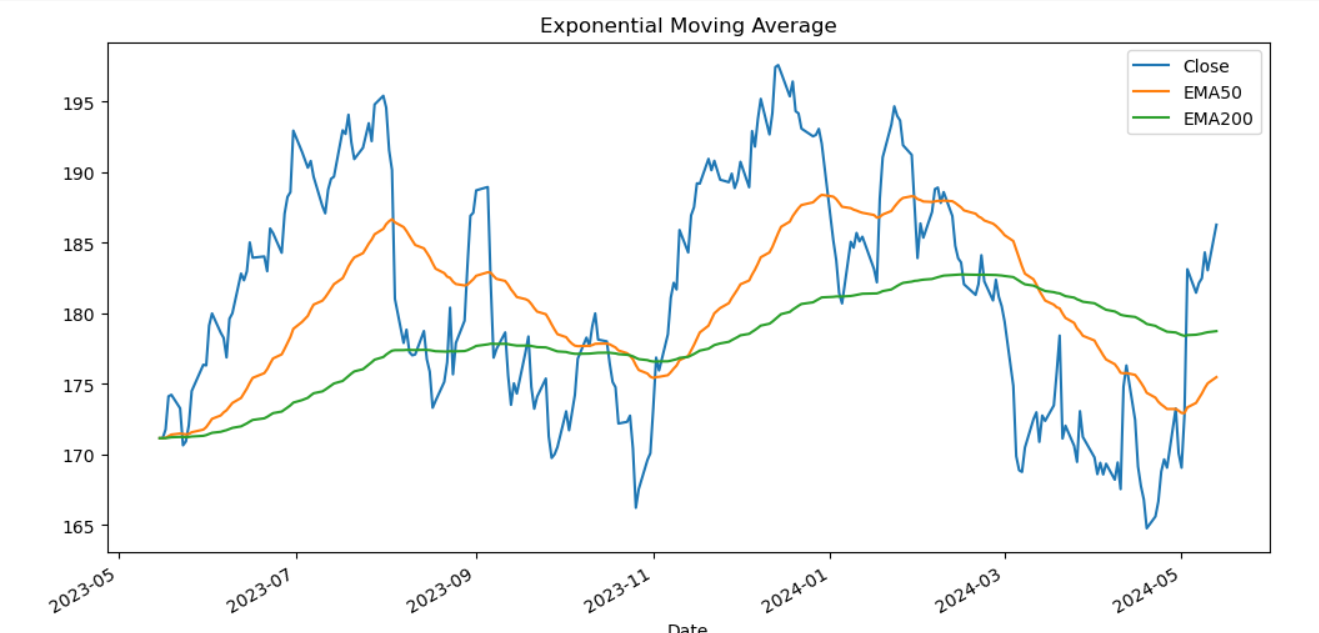
plt.title('Simple Moving Average')

plt.show()



**Exponential Moving Average (EMA):**

|  |
| --- |
| data['EMA50'] = data['Close'].ewm(span=50, adjust=False).mean() data['EMA200'] = data['Close'].ewm(span=200, adjust=False).mean()  data[['Close', 'EMA50', 'EMA200']].plot(figsize=(12,6)) plt.title('Exponential Moving Average') plt.show() |

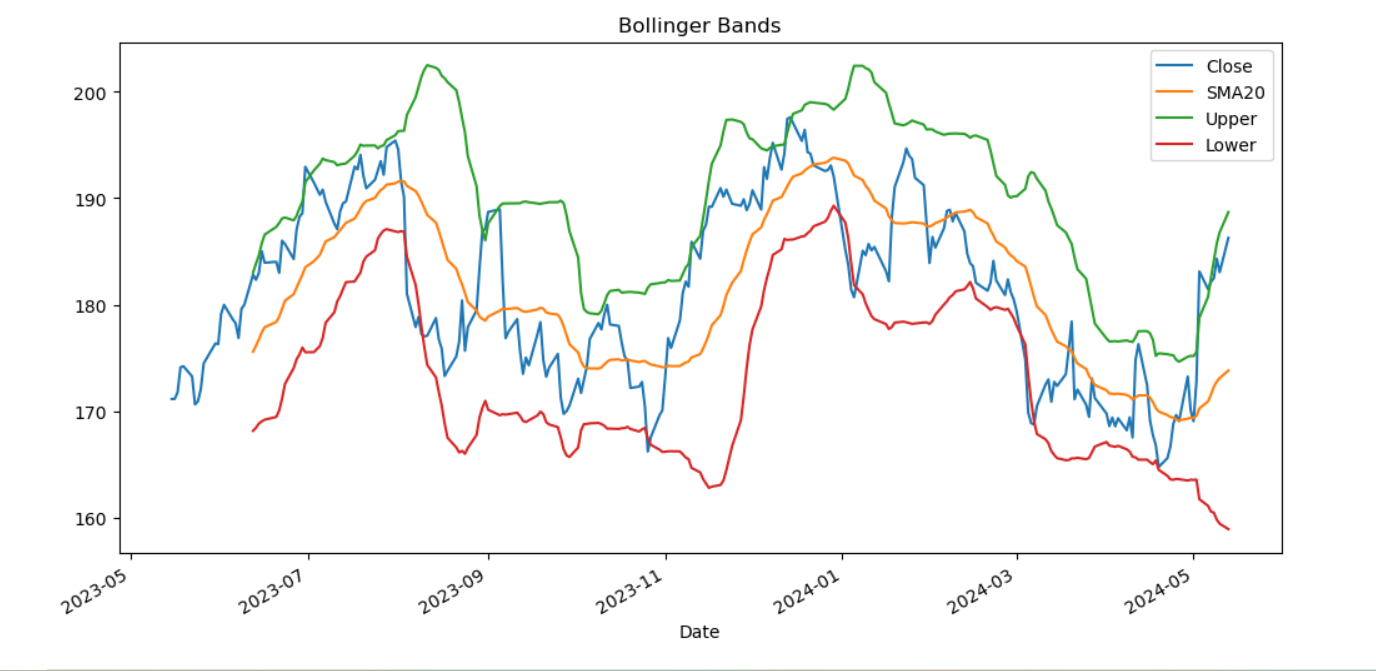
****

**Bollinger Bands:**

Bollinger Bands consist of a middle band being an N-period simple moving average (SMA), an upper band at K times an N-period standard deviation above the middle band, and a lower band at K times an N-period standard deviation below the middle band.

|  |
| --- |
| data['SMA20'] = data['Close'].rolling(window=20).mean() data['Upper'] = data['SMA20'] + 2\*data['Close'].rolling(window=20).std() data['Lower'] = data['SMA20'] - 2\*data['Close'].rolling(window=20).std()  data[['Close', 'SMA20', 'Upper', 'Lower']].plot(figsize=(12,6)) plt.title('Bollinger Bands') plt.show() |

*Note: Bollinger Bands are used to determine overbought and oversold levels; when the price reaches the upper band, it might be overbought, and when it reaches the lower band, it might be oversold.*



1. **Time Series Forecasting**

Time series forecasting involves predicting future values based on previously observed values.

**ARIMA Model:**

ARIMA stands for AutoRegressive Integrated Moving Average. It is a forecasting method for univariate time series data.

from statsmodels.tsa.arima.model import ARIMA

model = ARIMA(data['Close'], order=(5,1,0))

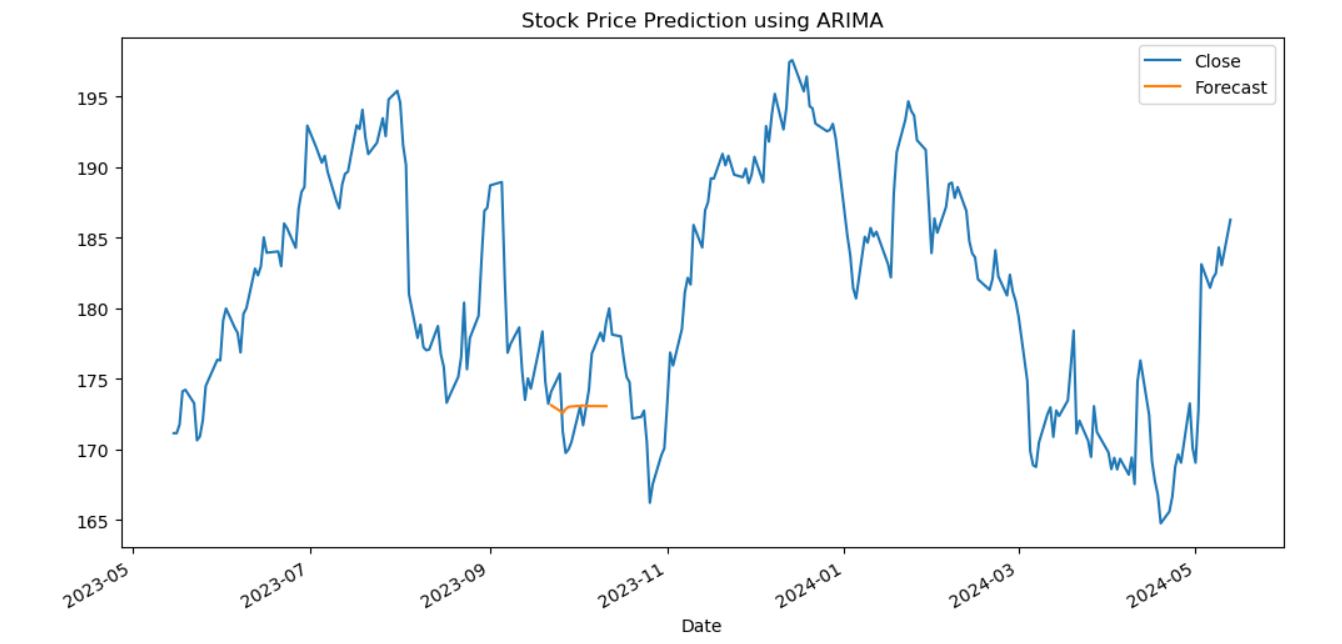
results = model.fit()

data['Forecast'] = results.predict(start=90, end=103, dynamic=True)

data[['Close', 'Forecast']].plot(figsize=(12,6))

plt.title('Stock Price Prediction using ARIMA')

plt.show()



1. **Portfolio Optimization**

When dealing with multiple stocks, it's crucial to determine how much of each stock you should hold in your portfolio.

**Efficient Frontier:**

This is a concept from Modern Portfolio Theory. The efficient frontier represents the set of optimal portfolios that offer the highest expected return for a specific level of risk.

from scipy.optimize import minimize

import numpy as np

**# Assuming stock\_list contains the symbols of the stocks in the portfolio**

returns = close\_prices.pct\_change()

mean\_returns = returns.mean()

cov\_matrix = returns.cov()

num\_portfolios = 10000

risk\_free\_rate = 0.0178 # example value

**# Portfolio optimization function**

def portfolio\_annualised\_performance(weights, mean\_returns, cov\_matrix):

returns = np.sum(mean\_returns\*weights ) \*252

std = np.sqrt(np.dot(weights.T, np.dot(cov\_matrix, weights))) \* np.sqrt(252)

return std, returns

**# Minimize negative Sharpe Ratio to get optimal portfolio**

def neg\_sharpe\_ratio(weights, mean\_returns, cov\_matrix, risk\_free\_rate):

p\_var, p\_ret = portfolio\_annualised\_performance(weights, mean\_returns, cov\_matrix)

return -(p\_ret - risk\_free\_rate) / p\_var

**# Constraints for optimization**

constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})

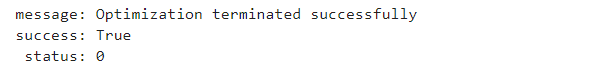
**# Running the optimization**

initial = [1./len(stock\_list) for stock in stock\_list]

bounds = tuple((0, 1) for asset in range(len(stock\_list)))

result = minimize(neg\_sharpe\_ratio, initial, args=(mean\_returns, cov\_matrix, risk\_free\_rate), bounds=bounds, constraints=constraints)

result



**8.Conclusion**

Therefore,the analysis of the real-time stock market has been done with the following insights:

* Stock price trends over the past year.
* Technical indicators and their implications.
* Potential investment strategies based on historical data.
* Performance of the developed trading strategy.

 The stock market is influenced by a myriad of factors, including economic data, corporate earnings reports, geopolitical events, and others. While historical data provides valuable insights, it's essential to stay updated with current events and adjust strategies accordingly.

**9.Future Work**

Incorporating more sophisticated models like neural networks for stock price prediction.

Exploring algorithmic trading strategies using tools like backtrader or QuantConnect.

Integrating real-time data feeds for live trading.

**10.External References**

**Websites:**

1. [Investopedia](https://www.google.com/url?q=https://www.investopedia.com/&sa=D&source=editors&ust=1715656107549269&usg=AOvVaw08FVyyPy9w43c33pSc5t88): A comprehensive resource for all things related to finance and investing. It provides detailed explanations of financial terms and concepts.
2. [Yahoo Finance](https://www.google.com/url?q=https://finance.yahoo.com/&sa=D&source=editors&ust=1715656107549508&usg=AOvVaw2-YoeSrmtfEJH1_gG-GZeW): A widely-used platform for checking stock prices, financial news, and historical data. It also offers educational articles and videos.
3. [StockCharts.com](https://www.google.com/url?q=https://stockcharts.com/&sa=D&source=editors&ust=1715656107549631&usg=AOvVaw3hXig2brp4OguFKCAL_rTx): A resource for technical analysis enthusiasts. It offers tutorials, articles, and tools for charting and technical analysis.
4. [Modern Portfolio Theory (Investopedia)](https://www.google.com/url?q=https://www.investopedia.com/terms/m/modernportfoliotheory.asp&sa=D&source=editors&ust=1715656107549765&usg=AOvVaw0rcHOUVWbXQs5SdJRjqMdJ): Learn more about the concept of the Efficient Frontier and Modern Portfolio Theory.

**YouTube Channels:**

1. [The Plain Bagel](https://www.google.com/url?q=https://www.youtube.com/c/ThePlainBagel&sa=D&source=editors&ust=1715656107550079&usg=AOvVaw3ccBIEm4YhYW3_Ti-3m2lU): Provides clear explanations of finance and investing concepts, including topics related to stock market analysis.
2. [Investopedia](https://www.google.com/url?q=https://www.youtube.com/user/investopediacom&sa=D&source=editors&ust=1715656107550222&usg=AOvVaw1kae53E3xXzVKnYZIIuYic): The official YouTube channel of Investopedia features video explanations of financial terms, strategies, and concepts.
3. [UKspreadbetting](https://www.google.com/url?q=https://www.youtube.com/user/ukspreadbetting&sa=D&source=editors&ust=1715656107550350&usg=AOvVaw0TEV0xfFJJCUlzAIzL1pMM): Offers educational content on trading strategies, technical analysis, and market psychology.

**Videos:**

1. [How the Stock Market Works (Investopedia Video)](https://www.google.com/url?q=https://www.youtube.com/watch?v%3D7aJ2y7wWVdU&sa=D&source=editors&ust=1715656107550567&usg=AOvVaw2jWfauoJ1jvBmotH9MFBeo): An introductory video explaining the basics of the stock market.
2. [What is Technical Analysis? (Investopedia Video)](https://www.google.com/url?q=https://www.youtube.com/watch?v%3DjT-sEzHTTto&sa=D&source=editors&ust=1715656107550736&usg=AOvVaw0IsRz51wnj46mLZbWMrzk2): Provides an overview of technical analysis and its key components.
3. [Introduction to ARIMA Time Series Forecasting (Data School)](https://www.google.com/url?q=https://www.youtube.com/watch?v%3De8Yw4alG16Q&sa=D&source=editors&ust=1715656107550873&usg=AOvVaw33eZm7NQM2DWsuA_01AXcK): A detailed video tutorial on ARIMA modeling for time series forecasting.
4. [LSTM Time Series Prediction Example with Keras in Python (Machine Learning Mastery)](https://www.google.com/url?q=https://www.youtube.com/watch?v%3DftMq5ps503w&sa=D&source=editors&ust=1715656107551014&usg=AOvVaw1M3vZTp2oGVRvLuLWTekgU): A practical guide to implementing LSTM for time series prediction.

**Acknowledgement : Mrs.SriDevi, Trainer,Able Jobs.**