|  |
| --- |
| Close-up of market graph analysis |
| Stock market analysis prediction using LSTM |
| |  |  |  | | --- | --- | --- | | Shaghayegh Haghbin | 2/5/24 |  | |

**Table of Contents**

**Introduction…………………………………………………………………………………. 2**

**Abstract………………………………………………………………………………………… 3**

**Problem Definition………………………………………………………………………… 4**

**Description……………………………………………………………………………………. 4**

**Aims & Objectives…………………………………………………………………………. 6**

**Requirements Specification……………………………………………………………. 6**

**Research Methodology…………………………………………………………………… 7**

**Ethical Issues…………………………………………………………………………………. 9**

**Deliverables…………………………………………………………………………………… 10**

**Project Resources……………………………………………………………………………. 11**

**Milestones………………………………………………………………………………………. 12**

**Conclusion………………………………………………………………………………………. 13**

**Code & Graph Analysis……………………………………………………………………. 15-51**

**Introduction:**

The stock market is a pivotal force in the global economy, serving as a hub for investment, wealth generation, and economic expansion. A comprehensive understanding of stock market intricacies is imperative for investors, financial analysts, and researchers. This report embarks on an exploration of stock market complexities, centering its focus on prominent technology stocks such as Apple, Amazon, Google, and Microsoft.

In the contemporary era dominated by data, the accessibility of historical stock market data and advancements in analytical tools empower professionals to glean valuable insights, discern trends, and formulate informed predictions. This report endeavors to harness the potency of data analysis and predictive modeling to uncover the fundamental dynamics of the stock market. The particular emphasis lies in providing a roadmap for investors and analysts to navigate the intricate terrain of stock market investments.

By delving into the domain of time series data and leveraging sophisticated techniques like Long Short-Term Memory (LSTM), this report seeks to furnish readers with the requisite knowledge and tools to comprehend, scrutinize, and prognosticate stock market behavior. Beyond being a mere exercise in data analysis, this exploration delves into the core of financial markets—a realm where decisions are crafted, risks evaluated, and opportunities seized.

Throughout this intellectual journey, it is crucial to acknowledge the dynamic nature of the stock market and the perpetually evolving landscape of financial data analysis. The discernments derived from this report are poised to empower stakeholders, affording them the capacity to make judicious, data-driven decisions, effectively mitigate risks, and capitalize on the multifaceted opportunities inherent in the stock market.

**Abstract:**

This report serves as a thorough guide for analyzing and predicting stock market behavior using advanced data analysis techniques. With a primary focus on technology stocks like Apple, Amazon, Google, and Microsoft, the report explores the complexities of time series data, risk assessment, and predictive modeling.

The abstract of this report explores the world of financial data analysis. The use of the yfinance library for data retrieval, combined with Python-based data analysis and visualization tools, sets the stage. Visual insights into stock market trends, correlations, and risk assessment metrics are presented using Seaborn and Matplotlib.

At the heart of this report is the exploration of risk analysis based on historical stock performance. It sheds light on quantifying risk and evaluating potential value at stake in stock investments. Additionally, the application of Long Short-Term Memory (LSTM) for predictive modeling offers a pathway to anticipate and forecast future stock prices, providing investors and analysts with a tool for informed decision-making.

In pursuing these goals, ethical considerations are emphasized. The report underscores the ethical use of stock market data and the responsible application of predictive modeling techniques. The report's deliverables include a comprehensive analysis of stock market data, visualization of key metrics, risk assessment reports, and predictive models for future stock prices.

Ultimately, the report aims to empower stakeholders with the knowledge, tools, and insights needed to navigate the dynamic stock market landscape. It is a guidance in making informed investment decisions and harnessing the power of data-driven analysis to seize opportunities in the stock market.

**Problem Definition:**

The stock market poses numerous challenges that require careful analysis, informed decision-making, and the ability to foresee future trends. This report focuses on the essential issue of finding effective methods to analyze stock market data and make informed predictions about future stock behavior.

The stock market's volatility and unpredictability underscore the importance of understanding historical stock performance, correlations between different stocks, and quantifying the associated risks. Investors and analysts often grapple with the task of thoroughly assessing the potential value at stake in stock investments and developing strategies to mitigate risks.

Moreover, the ever-changing nature of the stock market emphasizes the need to anticipate and forecast future stock prices. This enables stakeholders to make proactive and informed investment decisions. The central challenge involves the requirement for robust analytical tools and methodologies to navigate the stock market's complexities, identify trends, assess risks, and make data-driven predictions.

This report endeavors to address these challenges by providing stakeholders with the knowledge and tools needed to comprehend, analyze, and predict stock market behavior. The application of advanced data analysis techniques and predictive modeling serves to empower investors and analysts, enabling them to make informed decisions, manage risks effectively, and seize opportunities presented by the stock market.

**Description:**

This report delves into the world of stock market analysis and prediction, specifically focusing on technology stocks like Apple, Amazon, Google, and Microsoft. The journey kicks off by obtaining historical stock market data from Yahoo Finance using the yfinance library, setting the stage for a thorough exploration of stock market dynamics.

The report takes a multifaceted approach to data analysis, utilizing Python-based tools such as pandas, NumPy, Matplotlib, and Seaborn for visualizing, analyzing, and extracting insights from stock market data. Through the lens of time series data, the report tackles essential questions such as changes in stock prices over time, daily returns, moving averages, and correlations between different stocks.

Additionally, the report delves into risk assessment, quantifying the potential value at stake in stock investments by comparing expected returns with the standard deviation of daily returns. This risk analysis provides stakeholders with a foundational understanding of the risks associated with stock investments, enabling informed decision-making and the formulation of risk mitigation strategies.

A crucial aspect of this report lies in the use of Long Short-Term Memory (LSTM) for predictive modeling, offering a pathway to anticipate and forecast future stock prices. By harnessing LSTM's power, stakeholders gain access to a predictive tool that empowers them to make proactive investment decisions based on data-driven forecasts.

Ethical considerations are paramount throughout the exploration, emphasizing the responsible use of stock market data and the ethical application of predictive modeling techniques. The report concludes by delivering comprehensive analyses, visualizing key metrics, providing risk assessment reports, and presenting predictive models for future stock prices. This equips stakeholders with the knowledge, tools, and insights needed to navigate the dynamic stock market landscape.

Ultimately, this report serves as a guide for investors, financial analysts, and researchers, providing them with the means to understand stock market behavior, make informed investment decisions, and leverage data-driven analysis to capitalize on opportunities within the stock market.

**Aims & Objectives:**

This report has a main goal: to provide stakeholders with a thorough grasp of stock market dynamics, empowering them to make well-informed investment decisions and seize opportunities within the stock market. To attain this objective, the report outlines the following goals:

1. Obtain historical stock market data from Yahoo Finance using the yfinance library.
2. Visualize and analyze stock market data using Python-based tools like pandas, NumPy, Matplotlib, and Seaborn.
3. Investigate key questions such as how stock prices change over time, daily returns of stocks, moving averages, and correlations between different stocks.
4. Quantify the potential value involved in stock investments by comparing expected returns with the standard deviation of daily returns.
5. Apply Long Short-Term Memory (LSTM) for predictive modeling, providing a means to anticipate and forecast future stock prices.
6. Maintain a focus on ethical considerations throughout the exploration, emphasizing the responsible use of stock market data and the ethical application of predictive modeling techniques.
7. Deliver comprehensive analyses, visualize key metrics, present risk assessment reports, and offer predictive models for future stock prices.

**Requirements Specification:**

To successfully carry out the stock market analysis and prediction project, it is crucial to clearly outline the requirements that will lead to achieving its objectives. The following specifications detail the essential prerequisites and technical needs for the project:

1. **Data Acquisition:** Access to historical stock market data from Yahoo Finance is vital. This involves using the yfinance library to retrieve data for specific technology stocks, namely Apple, Amazon, Google, and Microsoft.
2. **Programming Environment:** The project mandates the use of a Python programming environment with essential libraries like pandas, NumPy, Matplotlib, Seaborn, and yfinance. Additionally, installing Jupyter Notebook or a similar interactive computing environment is necessary for code execution and visualization.
3. **Data Analysis and Visualization:** Proficiency in data analysis and visualization using Python-based tools is required. This includes leveraging pandas for data manipulation, NumPy for numerical computations, and Seaborn/Matplotlib for visually representing stock market metrics.
4. **Risk Assessment and Predictive Modeling:** Proficiency in risk assessment methodologies, including quantifying potential value at stake in stock investments, is necessary. Moreover, expertise in Long Short-Term Memory (LSTM) for predictive modeling is crucial for forecasting future stock prices.
5. **Ethical Considerations:** The project emphasizes the ethical use of stock market data and responsible application of predictive modeling techniques. Adhering to ethical guidelines and best practices in data analysis and predictive modeling is essential.
6. **Documentation and Reporting:** The project requires the ability to comprehensively document and report findings. Proficiency in articulating analyses, visualizations, risk assessment reports, and predictive models is essential for conveying insights effectively.
7. **Stakeholder Engagement:** The project requires the ability to engage with stakeholders, understand their investment objectives, and tailor analyses and predictions to meet their specific requirements.

**Research Methodology:**

The research methodology employed in the stock market analysis and prediction project encompasses a multifaceted approach, integrating data acquisition, exploratory data analysis, quantitative analysis, predictive modeling, and ethical considerations. The following components delineate the research methodology:

1. **Data Acquisition**: The research commences with the acquisition of historical stock market data from Yahoo Finance using the yfinance library. This step involves retrieving comprehensive data for the specified technology stocks, namely Apple, Amazon, Google, and Microsoft, to form the foundational data set for analysis and prediction.

2. **Exploratory Data Analysis (EDA)**: The project embarks on an in-depth exploration of the acquired stock market data, leveraging Python-based tools such as pandas, NumPy, and Seaborn to conduct exploratory data analysis. This phase involves visualizing stock price trends, identifying patterns, and uncovering insights to comprehend the historical performance of the selected stocks.

3. **Quantitative Analysis**: The project delves into quantitative analysis to address fundamental questions such as the change in stock prices over time, the daily returns of stocks, moving averages, and correlations between different stocks. This phase involves the application of statistical and quantitative techniques to derive meaningful insights from the stock market data.

4. **Risk Assessment**: The research methodology encompasses the quantification of risk associated with stock investments, comparing expected returns with the standard deviation of daily returns. This risk assessment phase provides stakeholders with a comprehensive understanding of the risks inherent in stock investments, facilitating informed decision-making and risk mitigation strategies.

5. **Predictive Modeling**: Leveraging Long Short-Term Memory (LSTM) for predictive modeling, the project endeavors to forecast future stock prices. This phase involves the application of advanced machine learning techniques to anticipate and visualize future stock behavior, empowering stakeholders with data-driven insights for proactive investment decisions.

6. **Ethical Considerations**: Throughout the research process, ethical considerations remain paramount. The responsible use of stock market data and the ethical application of predictive modeling techniques are emphasized, ensuring that the research adheres to ethical guidelines and best practices in data analysis and predictive modeling.

7. **Documentation and Reporting**: The research methodology culminates in the documentation and reporting of comprehensive analyses, visualization of key metrics, risk assessment reports, and predictive models for future stock prices. This phase involves the articulation of findings in a clear, concise manner to convey insights effectively to stakeholders.

**Ethical Issues:**

The stock market analysis and prediction project raise several ethical issues that must be addressed to ensure the responsible use of stock market data and the ethical application of predictive modeling techniques. The following ethical considerations are paramount:

1. **Data Privacy**: The project must adhere to data privacy regulations and guidelines, ensuring that the acquisition and use of stock market data does not infringe on the privacy rights of individuals or organizations.

2. **Data Security**: The project must prioritize data security, ensuring that stock market data is stored, processed, and transmitted securely to prevent unauthorized access, theft, or misuse.

3. **Transparency**: The project must be transparent in its data acquisition, analysis, and predictive modeling techniques, ensuring that stakeholders understand the methodologies employed and the limitations of the analyses and predictions.

4. **Fairness**: The project must ensure fairness in its analyses and predictions, avoiding bias or discrimination against individuals or organizations based on factors such as race, gender, ethnicity, or socioeconomic status.

5. **Accountability**: The project must be accountable for its analyses and predictions, ensuring that stakeholders can trace the decision-making process and hold the project team responsible for any errors or inaccuracies.

6. **Responsible Use of Predictive Modeling**: The project must emphasize the responsible use of predictive modeling techniques, ensuring that the models are not used to discriminate against individuals or organizations or to perpetuate existing biases.

7. **Informed Consent**: The project must obtain informed consent from stakeholders before using their data for analysis or predictive modeling, ensuring that stakeholders understand the purpose and scope of the project and the potential risks and benefits of participation.

**Deliverables:**

The stock market analysis and prediction project aim to provide stakeholders with comprehensive insights, actionable recommendations, and predictive models through various deliverables. The key deliverables include:

1. **Data Acquisition and Preprocessing:** The project provides a carefully curated dataset containing historical stock market data for specified technology stocks like Apple, Amazon, Google, and Microsoft. The dataset undergoes preprocessing to ensure data integrity and consistency.
2. **Exploratory Data Analysis (EDA) Report:** A detailed report presents the findings of exploratory data analysis, featuring visualizations of stock price trends, patterns, and key metrics. This report offers stakeholders a thorough understanding of the historical performance of the chosen stocks.
3. **Quantitative Analysis and Risk Assessment Report:** A quantitative analysis report explains changes in stock prices over time, daily returns of stocks, moving averages, and correlations between different stocks. The risk assessment report quantifies potential value at stake in each stock investment, providing insights into associated risks.
4. **Predictive Modeling and Future Price Forecast:** The project delivers predictive models based on Long Short-Term Memory (LSTM) for forecasting future stock prices. Stakeholders receive visualizations and reports showcasing the predicted future behavior of selected stocks, enabling data-driven insights for proactive investment decisions.
5. **Ethical Considerations and Compliance Report:** A comprehensive report outlines ethical considerations addressed throughout the project, ensuring responsible use of stock market data and ethical application of predictive modeling techniques. The compliance report demonstrates adherence to ethical guidelines and best practices in data analysis and predictive modeling.
6. **Stakeholder Engagement and Presentation:** The project includes stakeholder engagement sessions and a final presentation to convey findings, insights, and recommendations. Stakeholders receive personalized insights tailored to their investment objectives, fostering informed decision-making and strategic investment planning.
7. **Documentation and Code Repository:** A well-documented repository containing project code, methodologies, and technical documentation is provided. This repository serves as a valuable resource for stakeholders and future researchers seeking to understand project methodologies and replicate analyses and predictive models.

**Project Resources:**

The successful execution of the stock market analysis and prediction project hinges on various essential resources. Here are the key resources integral to the project:

1. **Hardware Resources:** A robust computing infrastructure is necessary, capable of handling large datasets, intricate analyses, and predictive modeling.
2. **Software Resources:** The project relies on various software tools, including Python-based libraries such as pandas, NumPy, Seaborn, and Matplotlib for data analysis and visualization. Machine learning libraries like TensorFlow and Keras are also used for predictive modeling.
3. **Data Resources:** Comprehensive and reliable stock market data from reputable sources like Yahoo Finance is crucial. The project also incorporates external data sources, such as economic indicators, news articles, and social media sentiment analysis, to enhance its analyses and predictions.
4. **Time Resources:** The project demands a significant time commitment for successful execution. The project allocated ample time for tasks such as data acquisition, exploratory data analysis, quantitative analysis, predictive modeling, and stakeholder engagement.

**Milestones:**

The stock market analysis and prediction project involve critical milestones, which are key stages marking its progress and ensuring value delivery to stakeholders. These milestones serve as checkpoints for effective project management. Here are the integral milestones:

1. **Data Acquisition and Preprocessing:** The project kicks off by acquiring historical stock market data for chosen technology stocks like Apple, Amazon, Google, and Microsoft from trustworthy sources such as Yahoo Finance. This data undergoes thorough preprocessing to ensure its reliability, consistency, and suitability for analysis.
2. **Exploratory Data Analysis (EDA):** The EDA milestone involves a comprehensive exploration of data to reveal stock price trends, patterns, and essential metrics. Visualizations and insights obtained during this phase offer stakeholders a foundational understanding of the historical performance of the selected stocks.
3. **Quantitative Analysis and Risk Assessment:** This milestone focuses on conducting quantitative analyses to evaluate changes in stock prices over time, daily returns, moving averages, and correlations between different stocks. The risk assessment phase quantifies the potential value at stake in each stock investment, providing stakeholders with crucial insights into investment risks.
4. **Predictive Modeling Development:** A significant milestone, this phase involves developing predictive models based on Long Short-Term Memory (LSTM). The models are trained and validated to forecast future stock prices, utilizing machine learning techniques to derive actionable insights for stakeholders.
5. **Ethical Considerations and Compliance:** This milestone addresses ethical considerations and compliance with regulatory guidelines. It ensures the project adheres to ethical standards, data privacy regulations, and best practices in predictive modeling, promoting the responsible and transparent use of stock market data.
6. **Stakeholder Engagement and Presentation:** Engaging stakeholders and presenting the project's findings, insights, and recommendations is a pivotal milestone. This phase involves tailoring insights to stakeholders' investment objectives, fostering informed decision-making, and strategic investment planning based on the project's analyses and predictions.
7. **Documentation and Knowledge Transfer:** The final milestone involves documenting the project's methodologies, code, and technical documentation in a comprehensive repository. This knowledge transfer phase ensures stakeholders and future researchers can access and understand the project's methodologies, analyses, and predictive models effectively.

**Conclusion:**

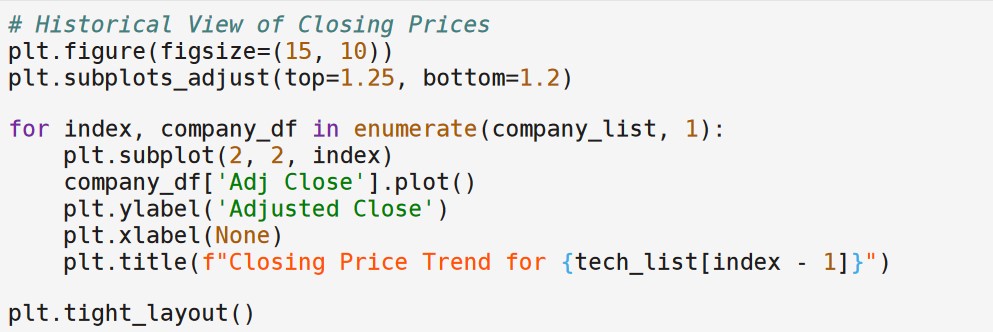
The stock market analysis and prediction project are a comprehensive initiative designed to furnish stakeholders with valuable insights, practical recommendations, and predictive models for informed investment decisions. To accomplish its goals, the project draws upon a variety of resources, including hardware and software infrastructure, data sources, human resources, budgetary allocations, and time.

The project involves several crucial milestones: data acquisition and preprocessing, exploratory data analysis, quantitative analysis and risk assessment, predictive modeling development, ethical considerations and compliance, stakeholder engagement and presentation, and documentation and knowledge transfer. These milestones serve as progress checkpoints, ensuring the project stays on course and delivers tangible benefits to stakeholders.

The project's outputs include a carefully curated dataset, an exploratory data analysis (EDA) report, a quantitative analysis and risk assessment report, predictive models, an ethical considerations and compliance report, stakeholder engagement and presentation materials, and a comprehensive documentation and code repository. By providing these outputs, the project aims to equip stakeholders with the knowledge, tools, and insights needed to make informed investment decisions, seize opportunities in the stock market, and navigate the intricate financial landscape with confidence.

In conclusion, the stock market analysis and prediction project are a significant endeavor that demands various resources, expertise, and time commitment for successful execution. Through effective resource utilization and milestone achievements, the project aims to deliver meaningful value to stakeholders, empowering them with data-driven insights for proactive investment decisions.

**Graph & Code Analysis**

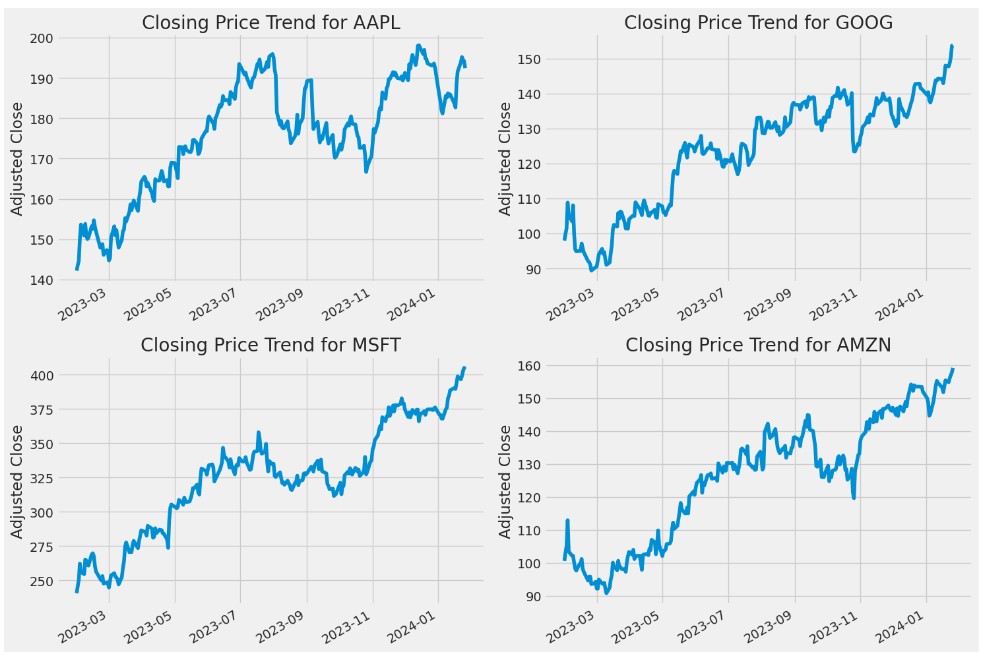
****

**Code snippet (1)**

**Code snippet (1)** represents a snippet of Python code that utilizes the matplotlib library for data visualization. The intention of this code is to create a figure displaying the historical view of closing prices for AAPL, GOOG, MSFT, and AMZN stocks. The closing price is a significant indicator as it represents the final price at which a stock is traded during a regular trading day and serves as a standard benchmark for investors to track performance over time.

From the code snippet, we can delve into the specifics of the implementation:

* The `plt.figure(figsize=(15, 10))` function call establishes the size of the plot to be generated.
* The code iterates over a list of stocks (AAPL, GOOG, MSFT, and AMZN) using a `for` loop, with `enumerate` to keep track of the current index.
* For each stock, it plots the adjusted closing prices using the `.plot()` method on the subset DataFrame `company\_df['Adj Close']`, labeling the y-axis as "Adjusted Close" and the x-axis is left with no label.
* Each plot's title is dynamically set to reflect the stock or test item at the current index.
* Finally, `plt.tight\_layout()` is called to adjust the layout so that everything fits neatly within the figure boundaries without overlapping content.

 **Graph (1)**

* Graph (1) presents a collection of four-line graphs, each showing the closing price trend for different tech companies' stocks over time.
* These companies are Apple Inc. (AAPL), Alphabet Inc. (GOOG, previously known as Google), Microsoft Corporation (MSFT), and Amazon.com Inc. (AMZN).
* These graphs typically serve to provide investors and analysts with a visual representation of stock performance, indicating the ups and downs in the closing price of a stock within a given timeframe.
* Each graph in the image has a labeled x-axis (horizontal) that represents time and a y-axis (vertical) that represents the closing stock price in US dollars.
* The line within each graph tracks the day-to-day changes in the stock's closing price, allowing for a quick assessment of trends, volatility, and performance over time.
* Based on the trends, all four companies show a general upward trajectory in their stock prices during the depicted periods.

****

**Code snippet 2**

**Code snippet 2** provides a Python script using `matplotlib`, a popular plotting library, to generate a series of plots. These plots represent the daily total stock trading volume of different companies. Furthermore, each part of the code is intended to do:

1. `plt.figure(figsize=(15, 10))` sets up the figure with a width of 15 inches and a height of 10 inches, providing a large enough canvas for the plots.

2. `plt.subplots\_adjust(top=0.85, bottom=0.2)` adjusts the spacing at the top and bottom of the plots within the figure to ensure that titles, axis labels, and other annotations fit without overlapping.

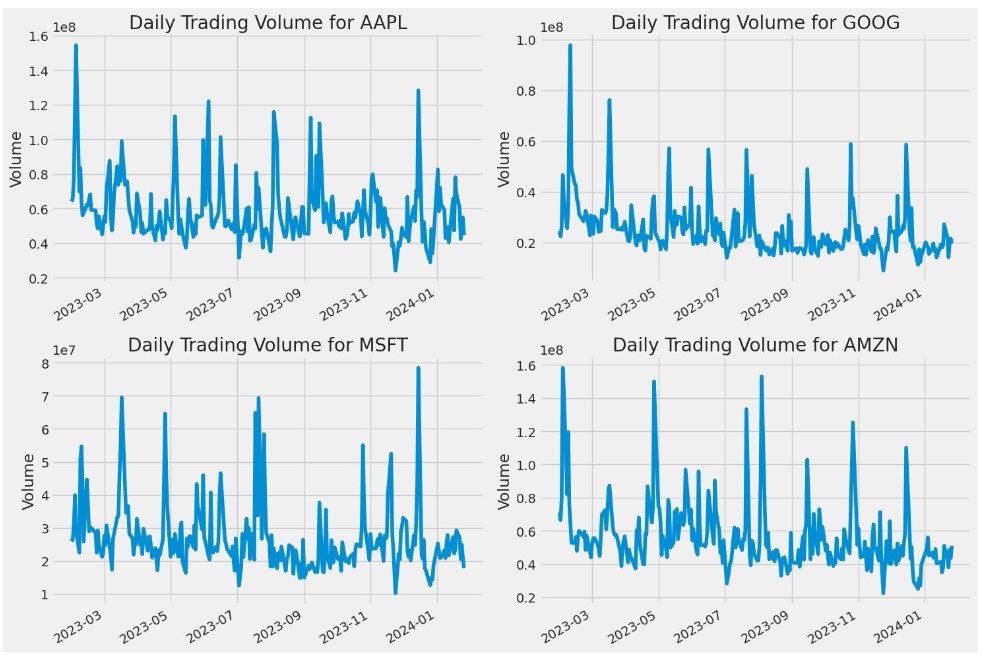
3. The for loop `for index, company\_df in enumerate(company\_list):` iterates over a list of dataframes `company\_list`. For each dataframe (representing a company), it also keeps track of the index. This index seems to be used later in creating a title that includes the company's name or identifier.

4. `company\_df['Volume'].plot()` creates a line plot using the 'Volume' column of the dataframe, showing the trading volume data.

5. `plt.ylabel('Volume')` sets the label of the y-axis to 'Volume', which is appropriate because the plot reflects trading volumes.

6. `plt.title(f"Daily Trading Volume for {tech\_list[index - 1]}")` sets the title of each subplot using string formatting. It looks like there's an intention to pull company names from a `tech\_list` that corresponds to each dataframe in `company\_list`. There is a minor error in the code – the `- 1` in `tech\_list[index - 1]` may lead to an off-by-one error unless `tech\_list` is deliberately offset.

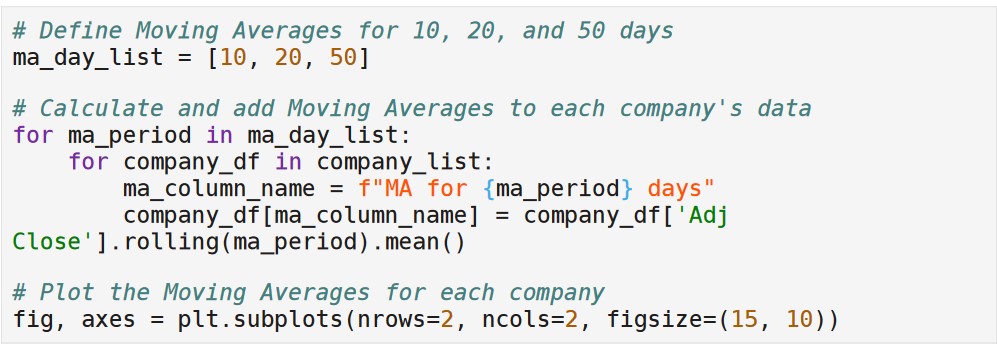
7. `plt.tight\_layout()` adjusts the layout of the figures to ensure everything is neatly presented without any overlapping elements.

****

**Graph (2)**

**Graph (2)** appears to be graphs of daily trading volumes for four different stocks, illustrated over a certain period. Each graph corresponds to a major publicly traded company, as denoted by their stock ticker symbols: AAPL (Apple Inc.), GOOG (Alphabet Inc., the parent company of Google), MSFT (Microsoft Corporation), and AMZN (Amazon.com, Inc.).

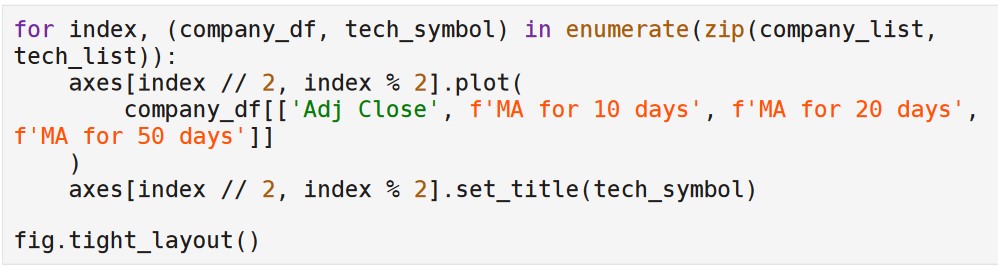
* These line graphs, or time series plots, show how the number of shares traded for each stock fluctuated from day to day, providing insight into the level of activity or liquidity of these stocks.
* Trading volume is an important metric for investors as it can reflect the market's sentiment towards a stock's potential price movement, with higher volumes often associated with price changes.
* Peaks in the graphs might suggest significant company events, earnings reports, or market-wide movements, while lower volumes might indicate less trading activity, which could be associated with a range of economic or company-specific factors.

****

**Code snippet (3)**

**Code snippet (3)** is a Python script that calculates moving averages for stock prices and subsequently plots these averages on a graph. Moreover, it is common practice in financial analysis to examine moving averages as they can help to identify trends in stock prices by smoothing out the noise from random short-term fluctuations.

* In the first part of the code, a list named ‘ma\_day\_list’ is defined, containing the integers 10, 20, and 50.
* These numbers represent the time frames, in days, for which moving averages will be calculated.
* The code then loops over each period in ‘ma\_day\_list’ and creates a new column in the ‘company\_df’ DataFrame for each moving average.
* The new columns are named according to the time frame and are calculated as the rolling mean of the ‘Adj Close’ prices over the respective time frame.
* The `rolling` method along with the `mean` function from the pandas library is used to calculate the moving average.
* In the second part, there's a comment indicating that the calculated moving averages will be plotted on a graph, although the actual plotting code is not present in the snippet.
* The ‘plt.subplots’ function from the matplotlib library is mentioned in a comment, implying that this function would be used to initialize a figure and a set of subplots for plotting.
* The figure size is specified as (15, 10), which defines the width and height of the graph.
* This visualization step is crucial for technical analysts or investors who may use such graphs to make informed decisions about buying or selling stocks based on the identified trends.

****

**Code snippet 3 (continued)**

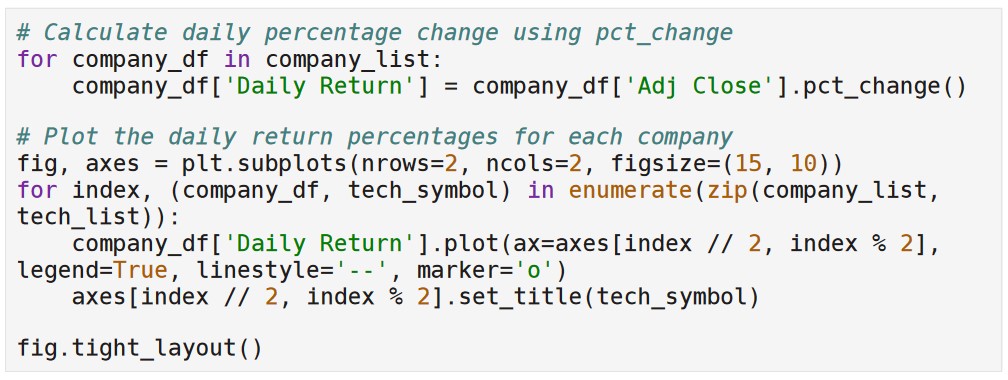
The continuation of **Code snippet 3** creates a series of plots, one for each item in a list of companies/stocks (with corresponding data frames and technical symbols) provided by the ‘company\_list’ and ‘tech\_symbol’ lists.

* For each iteration, the code plots the adjusted closing prices (‘Adj Close’) along with three different simple moving averages (SMAs) over 10, 20, and 50 days, presumably calculated earlier and stored in the respective data frame ‘company\_df’.
* These are labeled as ‘FMA for 10 days’, ‘FMA for 20 days’, and ‘FMA for 50 days’.
* The ‘index // 2, index % 2’ part is used as grid coordinates for the subplot positioning within a matplotlib Figure.
* The plot is given a title corresponding to the stock symbol (‘tech\_symbol’), suggesting that the plots may be organized in a grid with 2 columns (as suggested by ‘% 2’).
* Finally, ‘fig.tight\_layout()’ is called to adjust the spacing between the plots to prevent overlap of plot elements, where ‘fig’ would be a Figure object created outside of the shown code snippet.

**Graph (4)**

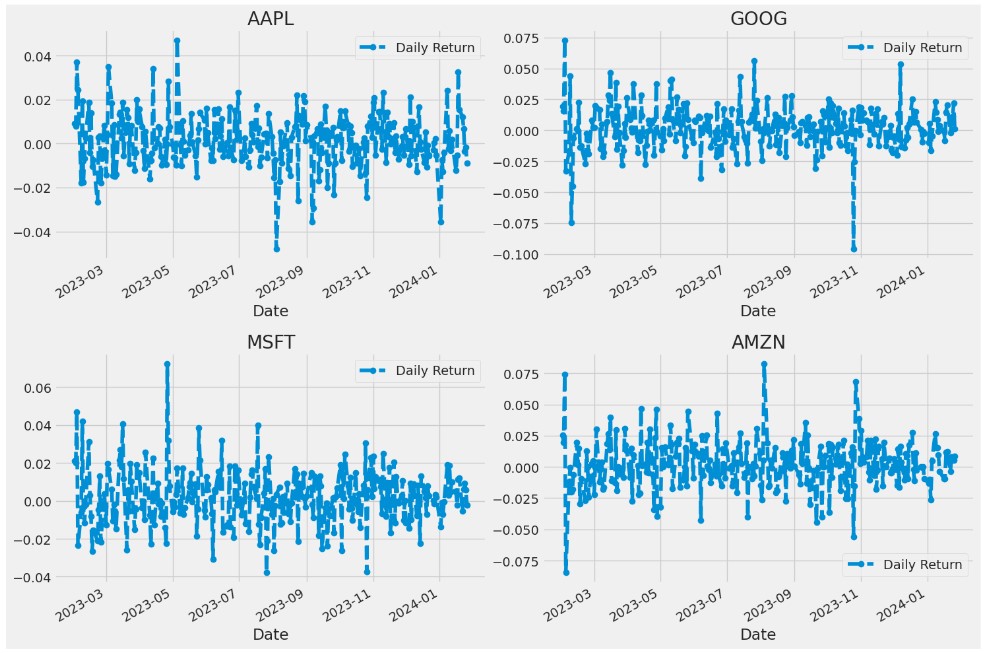
**Graph (4)** appears to be a set of line graphs representing the stock prices of four major companies over time: AAPL (Apple Inc.), GOOG (Google LLC, which is owned by Alphabet Inc.), MSFT (Microsoft Corporation), and AMZN (Amazon.com, Inc.). Each graph charts the daily closing prices of the company's stock and overlays moving averages to smooth out short-term fluctuations and highlight longer-term trends.

* Moving averages are commonly used in stock analysis to smooth out short-term fluctuations and highlight longer-term trends.
* The lines likely represent different spans of time (such as a 50-day or 200-day moving average), where the average stock price over that period is plotted on each day.
* All the graphs are trending upwards suggests that over the period shown, these stocks have generally been increasing in value, which might be indicative of a bullish market for tech stocks, or positive investor sentiment toward these companies.
* The time appears to be within a single year, as inferred by the notation on the x-axis (2023-03 signifies March 2023, for example), but the specific start and end dates are not clear.

**Code snippet (4)**

**Code snippet (4)** presents a Python code and is utilizing Pandas, a data manipulation library, and Matplotlib, a plotting library.

* In the first section, the code calculates the percentage change in the Adjusted Close price of a company's stock.
* This is done by creating a new column in the ‘company\_df’ DataFrame called ‘Daily Return’, which is the result of applying the ‘pct\_change()’ method to the ‘Adj Close’ column.
* In the second section, the code creates a plot of these daily returns.
* It sets up a figure and a set of subplots with a specific size, by providing the arguments ‘figsize=(15, 10)’.
* For each subplot, it takes the ‘Daily Return’ column from ‘company\_df’ DataFrame belonging to each company in the ‘company\_list’ and plots it on the created axes using ‘plot()’.
* It cycles through the subplots by the indices and aligns each plot with the corresponding company symbol using ‘enumerate(zip(company\_list, tech\_symbol))’.
* It customizes each plot by setting a title, enabling a legend, and specifying a line style and marker for the plot. Finally, ‘fig.tight\_layout()’ is called to adjust the spacing between the plots to prevent overlap.
* Overall, the code is designed to visualize the daily percentage returns of several technology companies' stocks.

**Graph (4)**

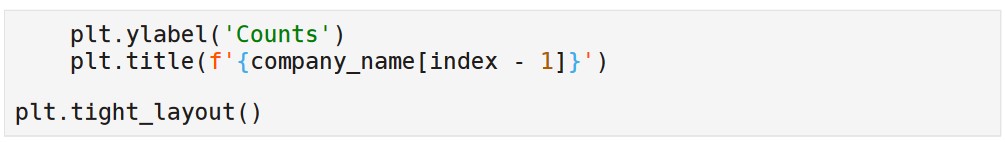
**Graph (4)** presents a set of four scatter plots, each representing the daily returns of a different stock over time. The stocks featured in these plots are for AAPL, GOOG, MSFT, and AMZN. Daily return data points are scattered across the time axis to show the fluctuation in returns from day to day.

* Each plot's x-axis indicates time, although specific dates are not completely legible, it suggests the data spans several years.
* The y-axis represents the daily return values calculated for the stocks.
* Daily returns of a stock are often computed as the percentage change in price from one day to the next which provides investors and analysts with an immediate sense of how the stock's value is changing over time.
* Variability in the scatter plot indicates the volatility of the stock's returns over the period shown.

**Code snippet (5)**

**Code snippet (5)** presents a Python code and is intended for use with data visualization libraries such as Matplotlib. The purpose of this code is to plot histograms for daily returns of various companies. Furthermore, here’s a breakdown of the code’s functionality:

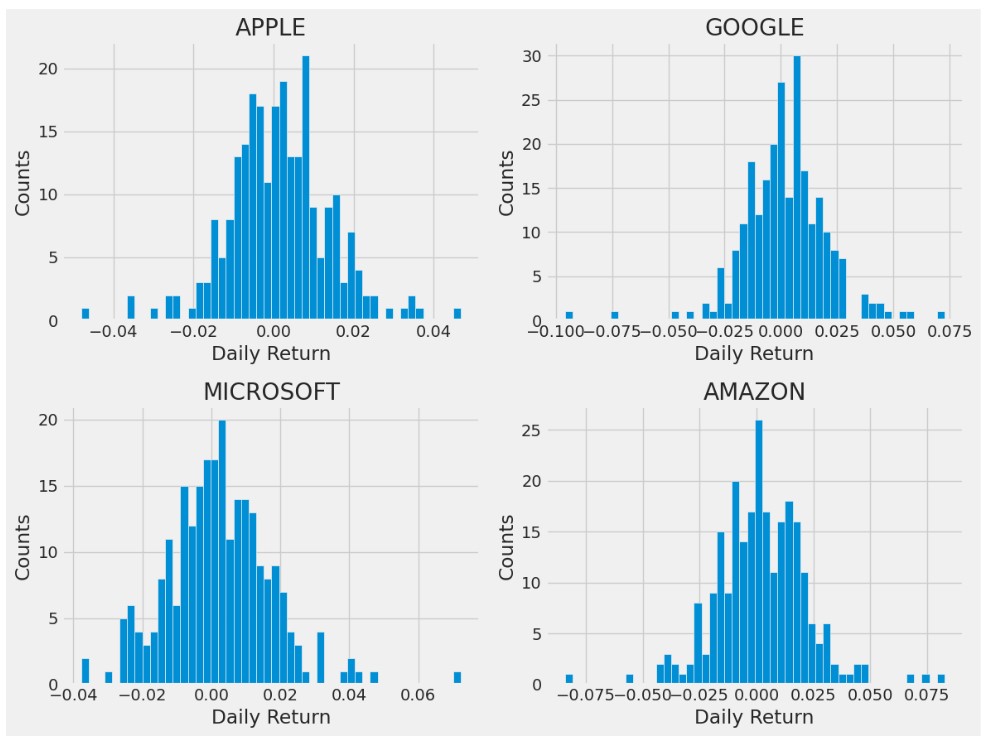
* First, ‘plt.figure(figsize=(12, 9))’ sets up a new figure for plotting, with a specified size of 12 inches by 9 inches.
* This provides a canvas on which all the subsequent plots will be drawn.
* Next is a ‘for’ loop that iterates through a list called ‘company\_list’, which is presumably a list of dataframes where each dataframe contains data for a different company.
* The loop uses ‘enumerate()’ to retrieve both the index of each item in the list (starting from 1 as specified by the second argument to ‘enumerate()’) and the dataframe itself (‘company\_df’).
* Inside the loop, ‘plt.subplot(2, 2, index)’ creates a subplot on a grid that is 2 rows by 2 columns, placing each plot in the position indicated by ‘index’.
* The ‘company\_df[‘Daily Return’].hist(bins=50)’ is calling the ‘hist()’ method on the ‘Daily Return’ column of the dataframe to create a histogram with 50 bins.
* Finally, ‘plt.xlabel(‘Daily Return’)’ labels the x-axis of each subplot with the text ‘Daily Return’.



**Code snippet 5 (continued)**

**Code snippet 5** configures the appearance of a plot. The function `plt.xlabel('Counts')` sets the label of the x-axis of the plot to "Counts". This is the label that will appear below the x-axis, informing the viewer what the axis represents. Meanwhile, the function `plt.title(f'{company\_name} ({index} - JJJ)')` sets the title of the plot.

Finally, `plt.tight\_layout()` is a function that automatically adjusts subplot params so that the subplot(s) fits into the figure area. This is helpful for making sure that the labels, titles, and other elements are displayed properly, and nothing is cut off.



**Graph (5)**

**Graph (5)** demonstrates four histograms, each representing the frequency distribution of daily returns for four different stocks: Apple, Google, Microsoft, and Amazon.

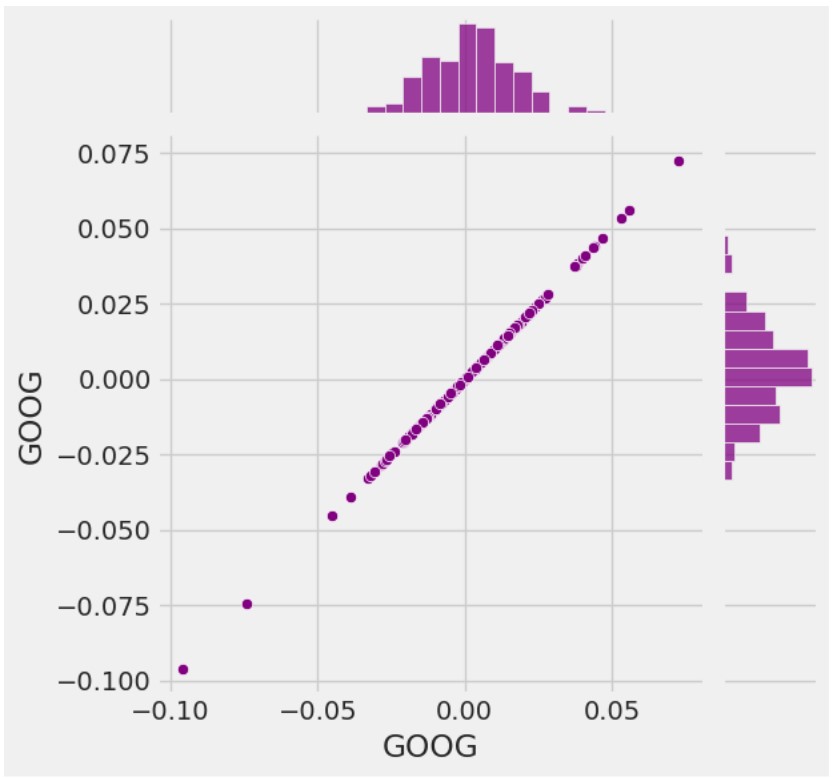
The x-axis of each histogram represents the daily return, which is typically calculated as the percentage change in price from one day to the next. The y-axis represents the counts, or the number of times a certain range of daily returns has occurred within the dataset being analyzed. The shape of each histogram gives us an insight into the volatility and the nature of the returns for each stock. The histograms for these four stocks appear to be roughly bell-shaped, suggesting that the daily returns on these stocks are normally distributed, which is a common assumption in financial analysis.



**Code snippet (6)**

**Code snippet 6** demonstrates a snippet of Python code that uses the seaborn library to plot data. This specific snippet uses the `jointplot` function to create a scatter plot comparing two variables.

* The variables `x='GOOG'` and `y='GOOG'` refer to columns in the `data\_tech\_returns\_df` DataFrame that presumably contain return data for Google's stock.
* The plot is intended to display how these two variables relate to each other, which in this case would be a perfectly linear relationship since the same dataset is being compared against itself.
* The `kind='scatter'` argument specifies that a scatter plot is being created, while `color='purple'` dictates that the points in the plot will be colored purple.
* After the jointplot function is called, there is a representation of the object (<seaborn.axisgrid.JointGrid at 0x7eee079ae80>) indicating the plot has been created and stored in that JointGrid object.

 **Graph (6)**

**Graph (6)** demonstrates a scatter plot with marginal histograms, which is a form of data visualization that combines a scatter plot with histograms to show the distribution of the individual variables involved along with their joint distribution. The horizontal and vertical axes of the scatter plot are labeled "GOOG", suggesting that the data points represent some form of comparison of Google-related metrics.

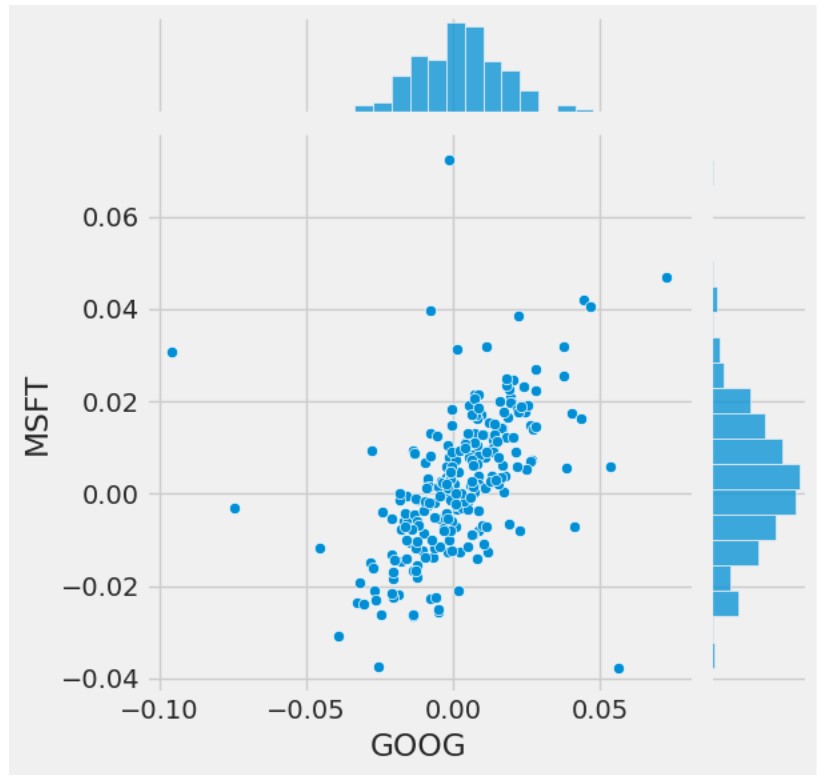
* The marginal histogram at the top of the scatter plot shows the frequency distribution of the values on the x-axis, while the histogram on the right side illustrates the frequency distribution of values on the y-axis.
* In both histograms, the shape of the distribution appears somewhat normal but possibly slightly skewed, indicating that most data points are clustered around a central value with fewer occurrences toward the tails.
* If this is stock data, such a visualization could be useful for understanding the relationship between two different measurement periods or variables and gives a quick insight into the individual variable distributions.
* The scatter plot itself shows a positive correlation between the two sets of GOOG data, meaning that as one value increases, the other tends to increase as well.



**Code snippet (7)**

**Code snippet (7)** shows a snippet of code written in Python, specifically for use with the Seaborn data visualization library.

* The code uses the **jointplot** function to create a scatter plot comparing the daily returns of Google and Microsoft.
* The data for this plot is being taken from a DataFrame named **tech\_returns\_df**. After the **jointplot** function is called, the output displayed is a reference to a Seaborn **JointGrid** object in memory, rather than a visual representation of the plot.

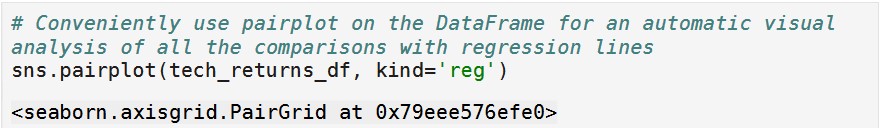
**Graph (7)**

**Graph (7)** illustrates the output of the Seaborn **jointplot** function from the previous code snippet. It is a scatter plot comparing the daily returns of Google (GOOG) on the x-axis and Microsoft (MSFT) on the y-axis. Each point on the scatter plot represents the daily return values for both stocks on a given day.

In addition to the central scatter plot, there are two histograms on the margins of the graph. The histogram at the top shows the distribution of daily returns for Google, while the histogram on the right displays the distribution of daily returns for Microsoft. These histograms provide insights into the variability and the typical range of daily returns for each stock.

From this joint scatter plot, it analyzes the relationship between the daily returns of the two technology companies. If the points tend to form a pattern or a line with a positive slope, it indicates that there is a positive correlation between the returns of the two stocks, meaning that they often move in the same direction on the same day. Conversely, a negative slope would indicate an inverse relationship.

The concentration of data points around the center suggests that on most days, the returns for both stocks are relatively small and close to zero. Outliers or points further from the center signify days with higher volatility or larger movements in stock prices.



**Code snippet (8)**

**Code snippet 8** shows a snippet of code that appears to be part of a Python script written for data analysis. The code contains a comment and a command using the Seaborn library, which is a popular visualization library in Python.

The comment in the code is:

**# Conveniently use pairplot on the DataFrame for an automatic visual analysis of all the comparisons with regression lines**

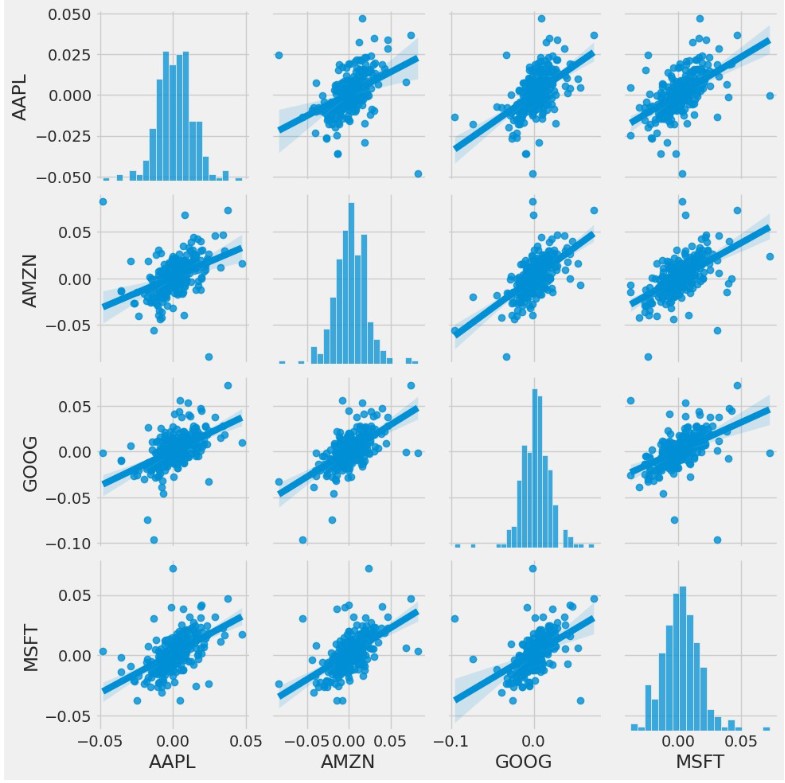
The Python command is:

**sns.pairplot(tech\_returns\_df, kind="reg")**

Here, **sns** is commonly used as the abbreviation for the Seaborn library. The **pairplot** function is being called with two arguments: **tech\_returns\_df** and **kind="reg"**. **tech\_returns\_df** is presumably a pandas DataFrame that contains technology stock returns data. The **kind="reg"** argument indicates that the pairplot should include regression lines in the plots it generates.

The pairplot function in Seaborn creates a grid of scatter plots for each pair of variables in a DataFrame, allowing for a quick and convenient way to visualize potential relationships and correlations between variables. The inclusion of regression lines helps in identifying trends and linear relationships between the variables.

The **<seaborn.axisgrid.PairGrid at 0x7fee957efe90>** looks like the output of the **pairplot** function, which is a PairGrid object at a specific memory location. This object can be further customized or simply used to display the plots.



**Graph (8)**

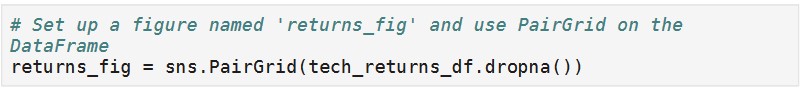
**Graph 8** illustrates a Seaborn pairplot generated by the Python script. This pairplot visualizes the relationships between the daily returns of four technology stocks: Apple (AAPL), Amazon (AMZN), Google (GOOG), and Microsoft (MSFT).

Each row and column in the grid correspond to one of these stocks. The diagonal plots are histograms that show the distribution of daily returns for each stock. These histograms demonstrate how the returns are spread out over time, providing insight into the volatility and normality of the returns.

The off-diagonal plots are scatter plots that compare the daily returns of two stocks. For instance, the plot at the intersection of the AAPL row and the AMZN column shows the relationship between Apple's and Amazon's daily returns. Each point in these scattered plots represents the daily returns of the two stocks on a given day.

The blue lines in the scatter plots are regression lines, which indicate the overall trend and direction of the relationship between the pairs of stock returns. A positive slope suggests a positive correlation, meaning that as one stock's returns increase, the other stock's returns also tend to increase.

The daily returns of these technology stocks have positive correlations with each other, as indicated by the positive slopes of the regression lines. The closer the points are to the regression line, the stronger the linear relationship between the stock returns.



**Code snippet (9)**

**Code snippet 9** shows a snippet of Python code that is commented to describe its function. The code snippet is as follows:

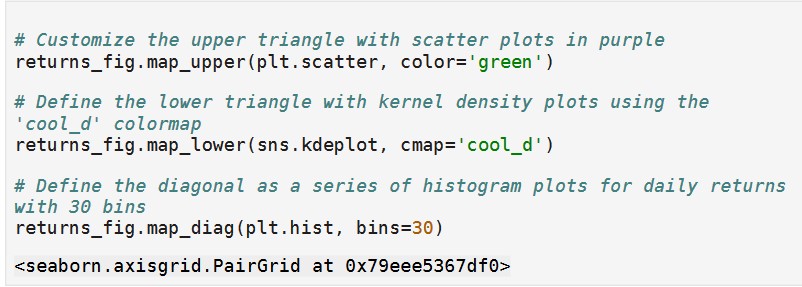
**# Set up a figure named ‘returns\_fig’ and use PairGrid on the DataFrame**

**returns\_fig = sns.PairGrid(tech\_returns\_df.dropna())**

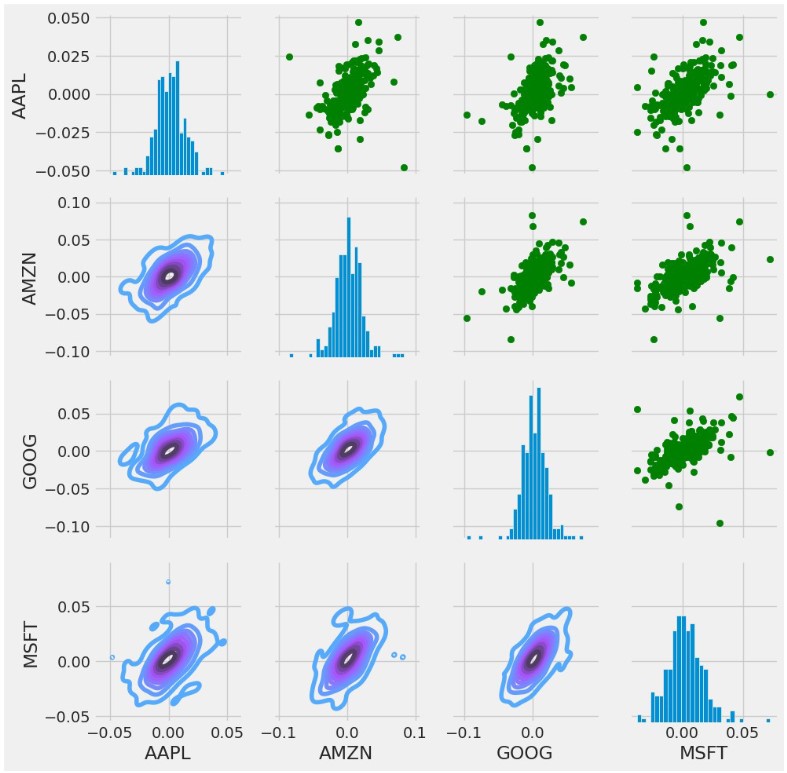
This code is utilizing the seaborn library, which is commonly used for statistical data visualization in Python. The **sns** is the standard shorthand for the seaborn library. The **PairGrid** function is being used to create a pair grid plot, which is a subplot grid for plotting pairwise relationships in a dataset.

The variable **tech\_returns\_df** seems to be a DataFrame containing the returns of technology stocks. The **dropna()** method is called **tech\_returns\_df**, which removes any rows that have missing data (NA, NaN).

The comment indicates that this grid plot is intended to be stored in a figure named **returns\_fig**.

**Code snippet 9 (continued)**

* The first comment in the code suggests customization of the upper triangle of the PairGrid with scatter plots in purple. However, the code specifies the color as 'green'. This might be an inconsistency in the comment or the code. The method **map\_upper** is used to apply a plotting function to the upper triangle of the grid.
* The second line of code uses the **map\_lower** method to apply a kernel density estimate (KDE) plot to the lower triangle of the grid. The KDE plot is a way to visualize the distribution of two variables. The **cmap** parameter specifies the colormap 'cool\_d', which is a predefined seaborn palette for the plots.
* The third line of code uses the **map\_diag** method to apply a histogram to the diagonal of the grid. This means that for each variable in the PairGrid, a histogram with 30 bins is plotted along the diagonal, showing the distribution of that variable.



**Graph (9)**

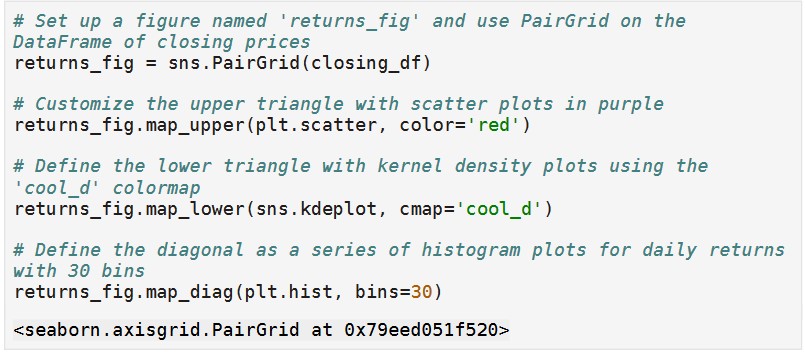
**Graph 9** shows a matrix of scatter plots and histograms, commonly referred to as a pair plot or a scatterplot matrix. The data visualized here represents the daily returns of four technology stocks: Apple (AAPL), Amazon (AMZN), Google (GOOG), and Microsoft (MSFT).

In the diagonal, the histograms indicate the distribution of daily returns for each stock. For AAPL, AMZN, GOOG, and MSFT, these histograms show how often specific ranges of daily returns occurred over the period analyzed.

The off-diagonal plots are scatter plots showing the relationship between the daily returns of two different stocks. Each point on these scatter plots represents a day, with the daily return of one stock on the x-axis and the daily return of another stock on the y-axis. For instance, the scatter plot in the first row and second column compares the daily returns of AAPL (y-axis) with AMZN (x-axis).

Additionally, for AMZN, GOOG, and MSFT, there are contour plots overlaid on the scatter plots, which indicate the density of the points, with areas of higher density being shown with more tightly packed contour lines.

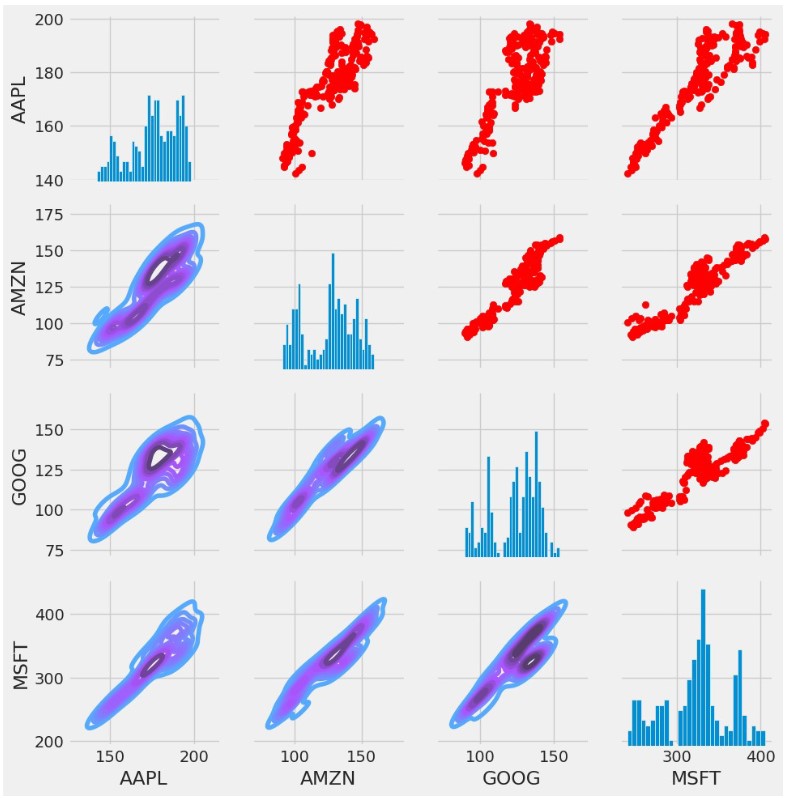
The scatter plots can be used to infer the correlation between the daily returns of these stocks. If the points form a pattern that trend from bottom left to top right, it suggests a positive correlation, meaning that as one stock's returns go up, the others tend to go up as well. Conversely, a pattern from top left to bottom right would suggest a negative correlation.



**Code snippet (10)**

**Code snippet 10** displays:

* **returns\_fig = sns.PairGrid(closing\_df)**: This line initializes a PairGrid object with a DataFrame **closing\_df** that presumably contains the closing prices of stocks. The PairGrid object is stored in the variable **returns\_fig**.
* **returns\_fig.map\_upper(plt.scatter, color='red')**: This line applies a scatter plot to the upper triangle of the PairGrid. The scatter plots will have red-colored points.
* **returns\_fig.map\_lower(sns.kdeplot, cmap='cool\_d')**: This line sets the lower triangle of the PairGrid to display kernel density estimation (kde) plots with the colormap 'cool\_d', which usually represents cool colors like blues and purples.
* **returns\_fig.map\_diag(plt.hist, bins=30)**: This line sets the diagonal of the PairGrid to show histograms with 30 bins. These histograms represent the distribution of daily returns for each stock.
* The last line is an output reference to the PairGrid object in memory, which is a typical output when creating objects in an interactive Python environment such as Jupyter Notebook or Google Colab.



**Graph (10)**

**Graph 10** shows a matrix of scatter plots and histograms, referred to as a pair plot or a scatterplot matrix. This type of visualization is used to understand the relationship between multiple variables in a dataset. Each row and column represent a different variable, and the plot where the row and column meet shows the relationship between the two variables. For the non-diagonal plots, the scatter plots show how the data points are distributed with respect to one another; this gives an insight into the correlation between the variables. Red points in scatter plots typically suggest individual data points.

On the diagonal, where the variable meets itself, histograms are provided instead of scatter plots. The histograms give a quick view of the distribution of a single variable: its spread and its skewness. The histograms show the frequency of data points within certain ranges of values for each individual stock's price here.

The pair plot represents stock prices for four different tech companies: AAPL, AMZN, GOOG, and MSFT on both the rows and columns. The pair plots reveal the correlations between the stock prices of these companies over a certain period. Additionally, the contour lines on top of the scatter plots indicate the density of the points, giving a clearer view of the concentration of the data.

A computer screen shot of a code

Description automatically generated

**Code snippet (11)**

**Code snippet 11** displays:

In the first line, plt.figure(figsize=(12, 10)), the script is setting up a figure with a width of 12 inches and a height of 10 inches.

The next lines create a subplot and generate a heatmap.

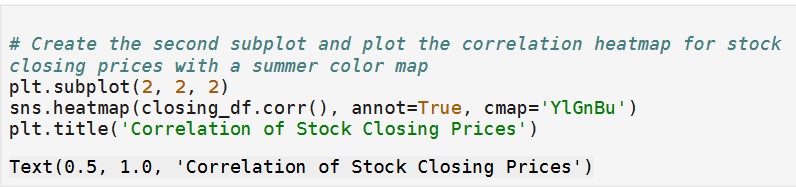
The command plt.subplot(2, 2, 1) sets up a subplot grid with 2 rows and 2 columns and makes the first subplot active (the upper left quadrant if you visualize the grid).

Then, the sns.heatmap() function from the seaborn library is used to create a heatmap of the correlation matrix obtained by calling .corr() on stock\_returns.

The annot=True argument specifies that the correlation values should be written on the cells of the heatmap, and the cmap='YlGnBu' argument sets the colormap to "Yellow-Green-Blue" shades.

Lastly, plt.title('Correlation of Stock Returns') adds a title to the subplot.

This piece of code is intended to help visualize the pairwise correlations between stock returns, often an important step in financial data analysis to understand how different stocks move with respect to each other.



**Code snippet 11 (continued)**

The continuation of this code displays:

The function plt.subplot(2, 2, 2) is used to set up a subplot grid that has 2 rows and 2 columns. The third argument 2 specifies that the current plot will be placed in the second position of this grid.

The next line of code, sns.heatmap(closing\_df.corr(), annot=True, cmap='YlGnBu'), is creating the heatmap using the popular Seaborn library, which is an extension of Matplotlib tailored for more complex statistical plots.

Here, sns is a common abbreviation for the Seaborn package. heatmap is a function that plots rectangular data as a color-encoded matrix.

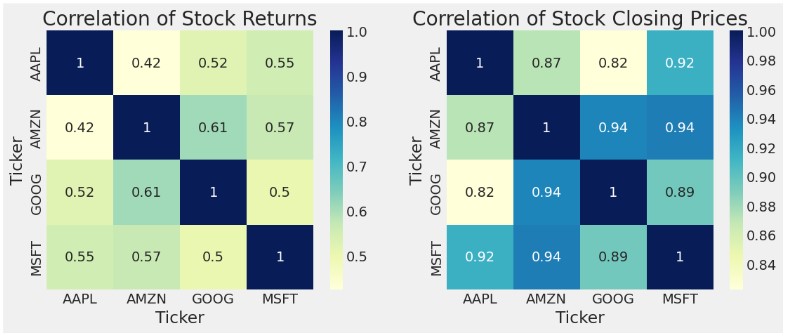
The method closing\_df.corr() computes the pairwise correlation of columns, assuming closing\_df is a Pandas DataFrame containing the stock closing prices.

The parameter annot=True enables annotations inside the squares of the heatmap, displaying the correlation coefficients.

The cmap='YlGnBu' argument sets the color map of the heatmap to "Yellow-Green-Blue," which is a sequential colormap often used for displaying gradients in data values.

The line plt.title('Correlation of Stock Closing Prices') adds a title to the heatmap subplot.

The last line Text(0.5, 1.0, 'Correlation of Stock Closing Prices') is an output from a Matplotlib function. It’s not part of the code to create the plot but rather an indication of what has been rendered by the plt.title() function in an interactive or script environment.



**Graph (11)**

**Graph 11** displays two heatmaps representing the correlation matrices for a set of stocks.

On the left, is the Correlation of Stock Returns, and on the right, the Correlation of Stock Closing Prices.

Both maps use a color scale to illustrate the strength of the correlation between the stock pairs defined by their tickers: AAPL, AMZN, GOOG, and MSFT.

In the heatmap on the left, the correlation values range from 0.42 to 0.61, indicating a positive correlation between the returns of these stocks, albeit not extremely strong.

Stronger correlations are typically desired when constructing a diversified investment portfolio as they suggest that the stocks are more likely to move in the same direction under similar economic conditions.

On the right, the heatmap shows very high correlation values ranging from 0.82 to 0.94 between the closing prices of the stocks.

This suggests that the price movements of these stocks are very closely aligned on a day-to-day basis, potentially due to being large-cap tech stocks that can be affected by similar industry and market factors.

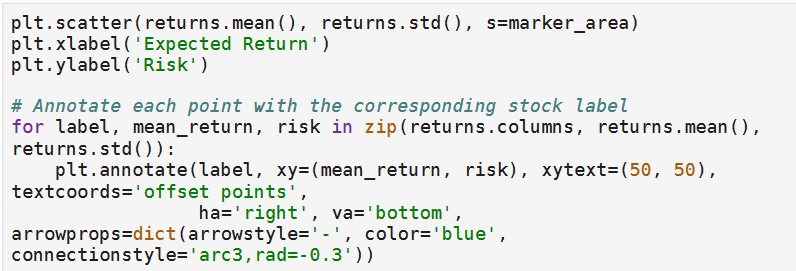
A computer code with text

Description automatically generated

**Code snippet (12)**

**Code snippet 12** shows:

* **Removing NaN Values**: returns = tech\_returns\_df.dropna() - This line removes all rows with NaN (Not a Number) values from the dataframe tech\_returns\_df, which presumably contains stock returns data for technology companies. The dropna() method is a built-in pandas function that deals with missing data. The resultant dataframe without NaN values is assigned back to the variable returns.
* **Defining an Area for Scatter Points**: marker\_area = np.pi \* 20 - This line sets the variable marker\_area to be π multiplied by 20. It's likely this value will be used to define the size of the markers in a scatter plot, with the area of each marker being proportional to the product of π and 20.
* **Creating a Figure**: plt.figure(figsize=(10, 8)) - This line creates a new figure object using matplotlib with a specified size of 10 inches by 8 inches. The figsize parameter takes a tuple that represents the width and height of the figure in inches.
* **Comment about Scatter Plot**: The last line is a comment and not actual code. It hints at what is to come next in the script, which seems to be the generation of a scatter plot displaying the mean vs. standard deviation for expected return and risk. The scatter plot is a common method of visualizing the relationship between two variables, in this case it could be used to visually assess the risk-return profile of different tech stocks.



**Code snippet 12 (continued)**

The continuation of **Code snippet 12** demonstrates:

A Python code that uses the matplotlib library to create a scatter plot visualizing the relationship between the expected returns and the risks (standard deviations) of a set of financial instruments, commonly stocks.

The points on the scatter plot represent individual stocks, with their mean returns plotted on the x-axis and their standard deviations on the y-axis.

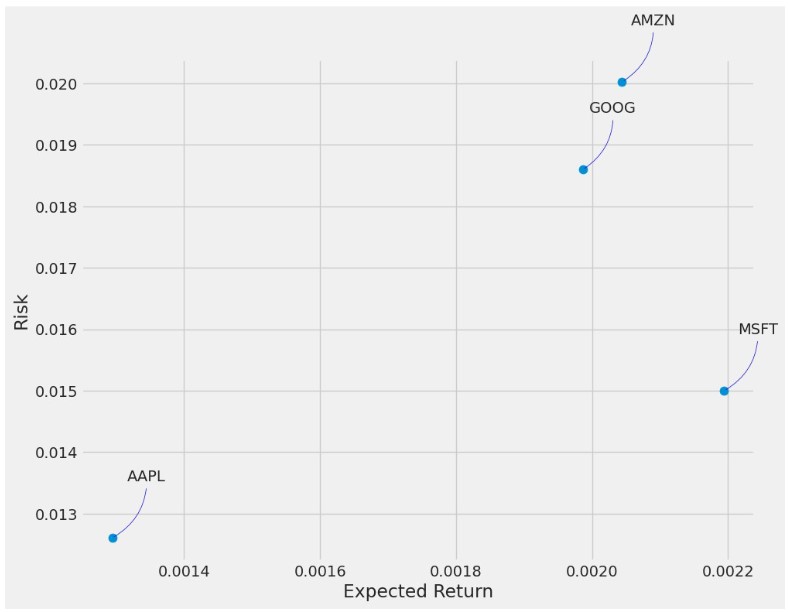
The size of each marker (s=marker\_area) can be adjusted to reflect additional information, such as the volume of trades or market capitalization, although the actual size (marker\_area) is not defined.

The annotation is placed at an offset of (50, 50) points from the (x, y) coordinates of the point on the plot, which is defined by the mean return and standard deviation of each asset (stock).

The text is aligned to the right of the point (ha='right') and vertically aligned to the bottom (va='bottom').

The annotation includes a stylistic arrow with specified properties, such as the style, color, and radius of curvature of the connecting line.

The connectionstyle argument specifies the type of connection path and its arc3,rad=-0.3 argument defines an arched connection with a specific radius.



**Graph (12)**

**Graph 12** shows a risk-return scatter plot typically used in finance to assess the performance of different investments. In this plot, each point represents a different investment, which in this case is labelled with the ticker symbols of various technology companies. AAPL, MSFT, GOOG, and AMZN —are used to denote each company's stock on a stock exchange.

The x-axis (horizontal) shows the expected return of the investments, which indicates the profit that an investor can anticipate over a period.

The y-axis (vertical) shows the risk, often measured as the standard deviation of the returns, which represents the uncertainty or the degree of variation in the returns of an investment.

Looking at the positions of the points, it analyzes the risk and expected return relationship for these companies, where a higher expected return is usually associated with higher risk.

In this plot, for instance, AMZN appears to have both a higher expected return and higher risk, while AAPL shows a lower expected return, but also lower risk compared to the others.

A computer code with text

Description automatically generated with medium confidence

**Code snippet (13)**

**Code snippet 13** demonstrates:

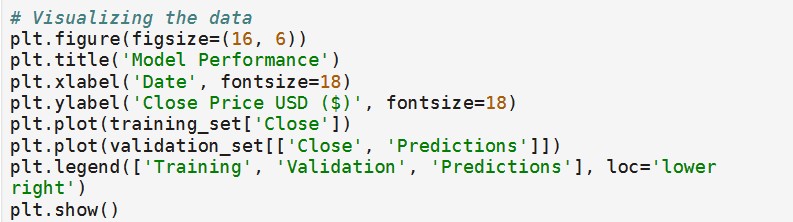
* plt.figure(figsize=(16, 6)): This line is setting up a figure with a defined size of 16 by 6 inches, which determines how large the plot will be when displayed.
* plt.title('Close Price History'): This command gives the plot a title, in this case, "Close Price History".
* plt.plot(stock\_data['Close']): Here, the plot function is being used to create a line plot of the closing prices from the stock\_data DataFrame or similar data structure. The 'Close' column from this data structure is selected, which likely contains the historical closing prices of a stock.
* plt.xlabel('Date', fontsize=18): This sets the label for the x-axis as "Date" and specifies the font size to be 18 for better readability.
* plt.ylabel('Close Price USD ($)', fontsize=18): Similarly, this line sets the label for the y-axis to "Close Price USD ($)" and uses a font size of 18.
* plt.show(): Finally, this line of code is used to display the plot. Without this command, the figure would not be shown to the user by default in some environments.



**Graph (13)**

**Graph 13** illustrates a line graph that represents the "Close Price History" of stocks over time. The x-axis (horizontal axis) of the graph denotes the time interval from the year 2012 to just beyond 2023, while the y-axis (vertical axis) reflects the closing price of the stocks in U.S. dollars (USD).

Following a brief graph analysis, the stock experienced a relatively slow increase in value from 2012 until around early 2016, whereupon the price seems to have begun an upward trajectory, indicating a period of growth. This growth became more pronounced starting from 2020, with the chart showing a steep incline in the stock's price, suggesting significant appreciation in value.



**Code snippet (14)**

**Code snippet 14** demonstrates:

* The first line, preceded by a comment, indicates that the following lines of code are for data visualization purposes.
* The plt.figure(figsize=(16, 6)) function sets up a new figure with a specified size: 16 units wide and 6 units tall.
* The plt.title('Model Performance') function adds the title "Model Performance" to the chart.
* The next two functions, plt.xlabel('Date', fontsize=18) and plt.ylabel('Closing Price [USD]'), add labels to the x-axis and y-axis with specified font sizes, indicating that the data being plotted is related to dates and closing prices in US dollars.
* The plt.plot() function is then called three times, each time being passed a different set of data—these lines will represent the training data, validation data, and predictions on the chart.
* The plt.legend() function adds a legend to the chart to help distinguish between these three lines, and the location is set to 'lower right' on the chart.
* Finally, plt.show() displays the resulting plot.

A graph showing a line

Description automatically generated

**Graph (14)**

**Graph 14** illustrates a chart depicting the performance of a predictive model over time, specifically applied to forecasting the closing price of stocks in US dollars (USD). It's a time series plot with the x-axis representing the date from the year 2012 to a point slightly beyond 2023, while the y-axis represents the close price of the stocks in USD.

* There are three different datasets illustrated in the chart: 'Training', which is likely the historical data used to train the predictive model and is shown in blue; 'Validation', in orange, which is a subset of the data not used for training but rather for evaluating the model's performance and preventing overfitting; and 'Predictions', in yellow, which represents the model's forecasts for future prices.
* The plot shows that the model's predictions closely follow the actual closing prices during the validation period and extend into the future, suggesting what the model expects the price trajectory to be based on the learned patterns from the historical data.
* The clear trend shown is an upward trajectory in close price over time, with the caveat that actual future prices could diverge from these predictions as they are subjected to an array of unpredictable market forces.