Internal Attachment Project
Report

Data Analytics Project Report on

Victor Oketch Sabare

SCT213-C002-0061/2021

Jomo Kenyatta University of Agriculture and Technology

SCT213: B.Sc. Data Science and Analytics

August 4, 2023

Professor James Mbao & Professor Daniel Njuguna

Author Note

sabare.victor@students.jkuat.ac.ke | victor@sabare.me

Abstract

This report will thoroughly examine the stock prices of several prominent companies, for example, Apple, Google, Amazon, Microsoft, Nvidia, and Qualcomm Inc. I strive to present you with a thorough and insightful data analysis. Moreover, I have uncovered several interrelated terms that exhibit correlations, which will be emphasized in this report.

***Keywords*:** Pandas, Seaborn, Data Analysis, Matplotlib, Numpy, Correlation, Plot, Stock Price, Analysis, Time Series Forecasting, Data Science, Financial Markets, Historical Stock Prices.

Acknowledgment

I want to express my sincere gratitude and appreciation to all who have contributed to the successful completion of my project on data analysis titled "Stock Price Analysis." This endeavor would not have been possible without the support, guidance, and encouragement I received throughout the project.

First and foremost, I extend my heartfelt thanks to my professors, **Professor James Mbao** and **Professor Daniel Njuguna**, for their invaluable guidance and expertise. Their insightful feedback, continuous encouragement, and unwavering support significantly shaped the direction and quality of this project.

I am deeply thankful to the **IT and Computing Departme**nt faculty members at **Jomo Kenyatta University of Agriculture and Technology** for providing me with a solid academic foundation and imparting the necessary knowledge and skills for this project.

I would also like to acknowledge the contributions of my classmates and peers, who offered their insights, suggestions, and constructive criticism, enriching my project's quality.

My gratitude extends to **Alpha Vantage** for providing the dataset used in this project, which served as a crucial resource for conducting the analysis.

I am indebted to the developers and contributors of open-source libraries and software, including **Matplotlib, Seaborn, Pandas, and Numpy,** which played a pivotal role in this project's data wrangling, analysis, and visualization phases.

Last, I thank my friends and family for their unwavering support, understanding, and encouragement throughout this academic journey.

In conclusion, I deeply appreciate the collective efforts contributing to completing my data analysis project. The knowledge and experiences gained from this project will undoubtedly serve as a cornerstone for my future endeavors.

Thank you all for being an integral part of my academic journey.

Table of Contents

Introduction

Data Context

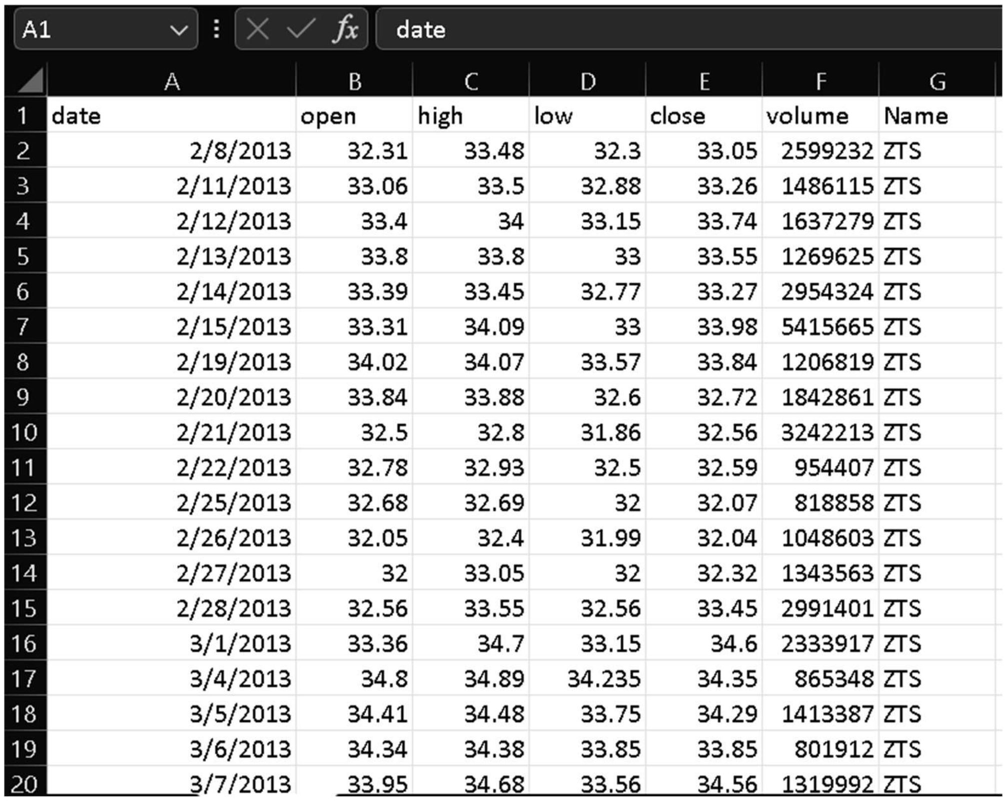
Comprehending the stock market can be daunting as it is a dynamic entity influenced by various factors, such as political decisions, economic conditions, and company-specific occurrences. Therefore, it is of utmost importance for investors, financial institutions, and policymakers alike to thoroughly analyze past trends and patterns in stock prices. The primary objective of this project is to leverage advanced data science techniques to scrutinize trends in stock prices meticulously.

Data Source

The dataset used in this project is sourced from the ALPHA VANTAGE API (https://www.alphavantage.co/) and comprises historical stock price data for companies listed on the New York Stock Exchange. The dataset spans from January 1, 2013, to December 31, 2018, providing a rich daily stock price information repository. Each record in the dataset consists of the following attributes:

- **Date**: The date of the stock price data.
- **Open**: The opening price of the stock on that day.
- **High**: The highest price the stock reached during the trading day.
- **Low**: The lowest price the stock reached during the trading day.
- **Close**: The closing price of the stock on that day.
- **Volume**: The trading volume of the stock on that day.
- **Name**: The name of the stock

Below is an image of one of the data files used in CSV format.



A1 date

	A	B	C	D	E	F	G
1	date	open	high	low	close	volume	Name
2	2/8/2013	32.31	33.48	32.3	33.05	2599232	ZTS
3	2/11/2013	33.06	33.5	32.88	33.26	1486115	ZTS
4	2/12/2013	33.4	34	33.15	33.74	1637279	ZTS
5	2/13/2013	33.8	33.8	33	33.55	1269625	ZTS
6	2/14/2013	33.39	33.45	32.77	33.27	2954324	ZTS
7	2/15/2013	33.31	34.09	33	33.98	5415665	ZTS
8	2/19/2013	34.02	34.07	33.57	33.84	1206819	ZTS
9	2/20/2013	33.84	33.88	32.6	32.72	1842861	ZTS
10	2/21/2013	32.5	32.8	31.86	32.56	3242213	ZTS
11	2/22/2013	32.78	32.93	32.5	32.59	954407	ZTS
12	2/25/2013	32.68	32.69	32	32.07	818858	ZTS
13	2/26/2013	32.05	32.4	31.99	32.04	1048603	ZTS
14	2/27/2013	32	33.05	32	32.32	1343563	ZTS
15	2/28/2013	32.56	33.55	32.56	33.45	2991401	ZTS
16	3/1/2013	33.36	34.7	33.15	34.6	2333917	ZTS
17	3/4/2013	34.8	34.89	34.235	34.35	865348	ZTS
18	3/5/2013	34.41	34.48	33.75	34.29	1413387	ZTS
19	3/6/2013	34.34	34.38	33.85	33.85	801912	ZTS
20	3/7/2013	33.95	34.68	33.56	34.56	1319992	ZTS

Objective

The primary objective is to comprehensively analyze historical stock prices and develop a highly dependable predictive model to forecast future stock prices accurately. To achieve this, I intend to employ advanced data analysis techniques that will enable us to gain in-depth insights into trends, patterns, and potential drivers that could impact the prices of selected companies.

The goal is to provide you with actionable information you can utilize to make informed investment decisions, thereby maximizing returns and minimizing risks. With a rigorous and meticulous approach, I aim to deliver a reliable and trustworthy solution to help someone navigate the complex and ever-changing world of investing in the stock market.

Methodology

The project follows a structured methodology involving several key steps:

1. **Data Acquisition:** The historical stock price data is sourced from https://www.alphavantage.co/ and is loaded into my project environment.

2. **Data Preprocessing:** The dataset undergoes thorough cleaning, handling missing values, outlier detection, and formatting adjustments.
3. **Exploratory Data Analysis (EDA):** Descriptive statistics, time series plots, and correlation analyses are performed to gain initial insights into the dataset's characteristics.
4. **Time Series Analysis:** Time series decomposition and autocorrelation analysis are conducted to understand seasonality and autocorrelation patterns in stock prices.
5. **Results and Interpretation:** The findings from the analysis are presented, including insights into potential market trends and future price movements.

Data Acquisition

The historical stock price data is often collected and provided by financial data providers and APIs, such as Alpha Vantage, Quandl, Yahoo Finance, and others. These platforms aggregate real-time and historical financial data from various stock exchanges, making it easily accessible to researchers, analysts, and data enthusiasts for analysis and modeling.

Alpha Vantage, for instance, is a prominent financial data provider that offers a comprehensive API to access historical and real-time stock price data, technical indicators, and other financial metrics. Below is an illustrative explanation of how data is typically acquired using an API like Alpha Vantage:

API Access

Users register on the Alpha Vantage website to obtain an API key, a unique identifier allowing them to access the data and services offered by Alpha Vantage.

Requesting Data

Users can request HTTP to Alpha Vantage's endpoints with the API key to retrieve specific data. Users might use the "TIME_SERIES_DAILY" endpoint for historical stock price data, specifying the stock symbol and the desired date range.

Data Response

Alpha Vantage responds to the API request with the requested data in JSON or CSV format. The response includes a timestamp, opening price, closing price, high price, low price, trading volume, and possibly other information.

Here's a sample Python code snippet demonstrating how you might use an API (not specifically Alpha Vantage) to acquire stock price data:

```
import requests

# Replace 'YOUR_API_KEY' with your actual API key
api_key = 'YOUR_API_KEY'
symbol = 'AAPL'
start_date = '2023-01-01'
end_date = '2023-07-31'

# Construct the API request URL
url =
f"https://api.example.com/stock/{symbol}/prices?start_date={start_date}&end_d
ate={end_date}&apikey={api_key}"

# Send the API request
response = requests.get(url)

# Parse the response
data = response.json()
```

Data Loading

The historical stock data I acquired was downloaded and kept in a directory since the data is for different stocks in the financial market. Using pre-existing datasets from reputable sources provides a solid foundation for my analysis. It allows me to focus on the core data science tasks without the complexities of API integration. The data had to be loaded into my **project environment (Jupyter Notebook – anaconda 3)**. I first had to count the number of CSV files in the directory using the glob python package and then load each into a unified data frame using the **pandas append () method.** We then proceeded to note the shape of the new data frame we would use and the unique values in the data frame to identify the individual stocks we would work with. Here is the code to collect and load data into the jupyter notebook.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import glob

# Counting up the total number of CSV files that we have in the directory.
len(glob.glob(r'individual_stocks_5yr/*csv'))
505
# Lets store files of those stock that we have to consider for analysis ..
company_list = [
    r'individual_stocks_5yr\\AAPL_data.csv' ,
    r'individual_stocks_5yr\\AMZN_data.csv' ,
    r'individual_stocks_5yr\\GOOG_data.csv' ,
    r'individual_stocks_5yr\\MSFT_data.csv' ,
    r'individual_stocks_5yr\\ABBV_data.csv' ,
    r'individual_stocks_5yr\\ABC_data.csv' ,
    r'individual_stocks_5yr\\NVDA_data.csv' ,
    r'individual_stocks_5yr\\QCOM_data.csv' ,
    r'individual_stocks_5yr\\TEL_data.csv'

]
# Import warnings package to get read of any future warnings in the project
import warnings
from warnings import filterwarnings
filterwarnings('ignore')

# We use pandas append() to do collect all the .csv files
all_data = pd.DataFrame()

for file in company_list:

    current_df = pd.read_csv(file)

    all_data = current_df.append(all_data , ignore_index=True)

    ##full_df = pd.concat([full_df , current_df] , ignore_index=True)
# Dimensions of all_data dataframe
all_data.shape
(11047, 7)
all_data.head(6)
date    open    high    low     close   volume  Name
0       2013-02-08      40.09   40.61   40.07   40.52   1146224 TEL
1       2013-02-11      40.49   40.56   40.27   40.44   1056745 TEL
2       2013-02-12      40.44   40.69   40.37   40.49   876110  TEL
3       2013-02-13      40.58   40.99   40.54   40.92   1426745 TEL
4       2013-02-14      40.62   41.09   40.46   41.08   2091112 TEL
5       2013-02-15      41.08   41.29   40.75   41.03   1533139 TEL
all_data['Name'].unique()
all_data['Name'].unique()
array(['TEL', 'QCOM', 'NVDA', 'ABC', 'ABBV', 'MSFT', 'GOOG', 'AMZN',
       'AAPL'], dtype=object)
```

Data Preprocessing

Handling Missing Data

Firstly, I checked whether there were any missing data records in the dataset I used since there were all the values.

```
## checking missing values

all_data.isnull().sum()
```

Data Transformation

I proceeded to check the data types in the dataset. I realized the “date” column was not in the Python ‘date-time’ format, so I had to convert it to make the feature useful in the data analysis. Time series analysis is important when dealing with historical stock data; hence this feature had to be transformed.

```
## Converting data-type of "date" feature into date-time

all_data['date'] = pd.to_datetime(all_data['date'])

tech_list = all_data['Name'].unique()

tech_list
```

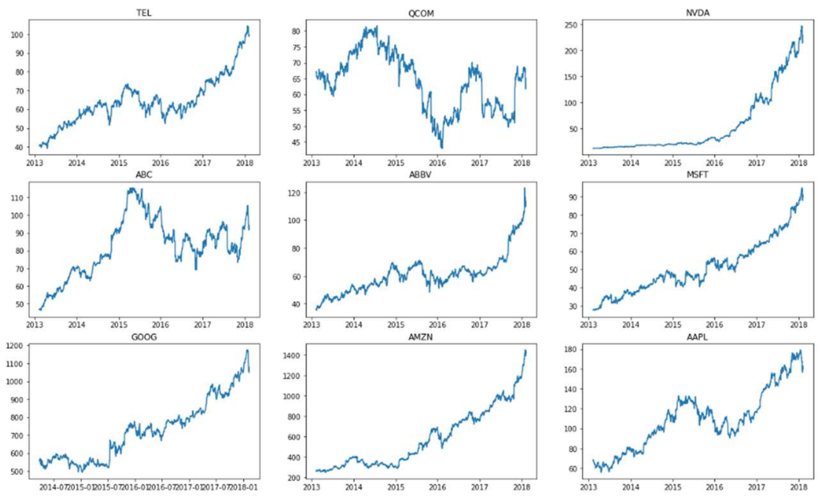
Explanatory Data Analysis

Exploratory Data Analysis (EDA) is a crucial phase in any data analysis project. It involves examining and summarizing the dataset's main characteristics and patterns, which helps understand the data, identify potential outliers, and formulate hypotheses for further analysis.

Time Series Plots

Time series plots are essential for understanding how stock prices evolve. These plots help visualize trends, seasonality, and potential anomalies. Line plots are common choices.
In this project, I created the time series line plots of all the companies' stocks over time in the data set. We can see the stock prices rising steadily.
Here's a sample Python code snippet demonstrating what was used for a time series plot using the matplotlib library:

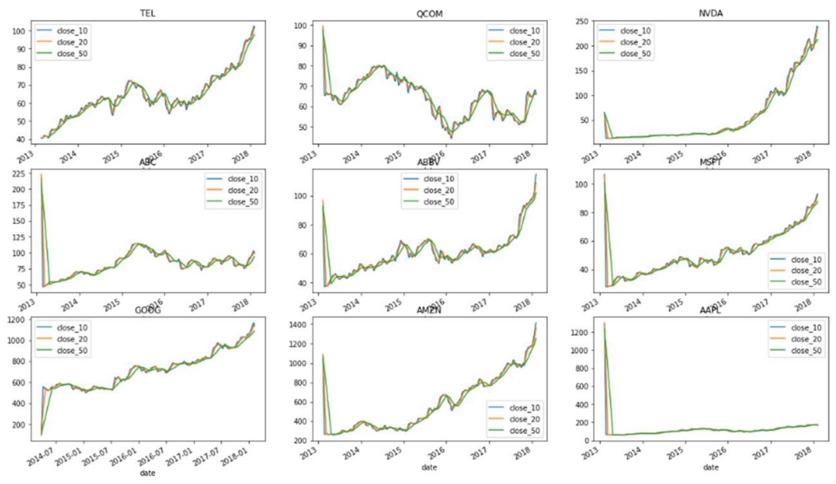
```
plt.figure(figsize=(20,12))

for index , company in enumerate(tech_list , 1):
    plt.subplot(3 , 3 , index) ## creating subplot for each stock
    filter1 = all_data['Name']==company
    df = all_data[filter1]
    plt.plot(df['date'] , df['close']) ## plotting "date" vs "close"
    plt.title(company)
```



I then proceeded to investigate the moving average of the stocks after different rolling day windows (i.e., 10 Days, 20 Days. 50 Days) using time series plots.

Here is the sample of Python code that was used in the project:

```
## Moving average of the various stocks.

all_data.head(15)

all_data['close'].rolling(window=10).mean().head(14)

new_data = all_data.copy()

## Now lets consider different windows of rolling,ie 10 days ,20 days ,30
days

ma_day = [10 ,20, 50]

for ma in ma_day:
    new_data['close_'+str(ma)] = new_data['close'].rolling(ma).mean()

new_data.tail(7)

new_data.set_index('date' , inplace=True)

new_data

new_data.columns

plt.figure(figsize=(20,12))

for index , company in enumerate(tech_list , 1):
    plt.subplot(3 , 3, index)
    filter1 = new_data['Name']==company
    df = new_data[filter1]
    df[['close_10','close_20', 'close_50']].plot(ax=plt.gca())
    plt.title(company)
```



Analyze the Closing price change in Apple stock.

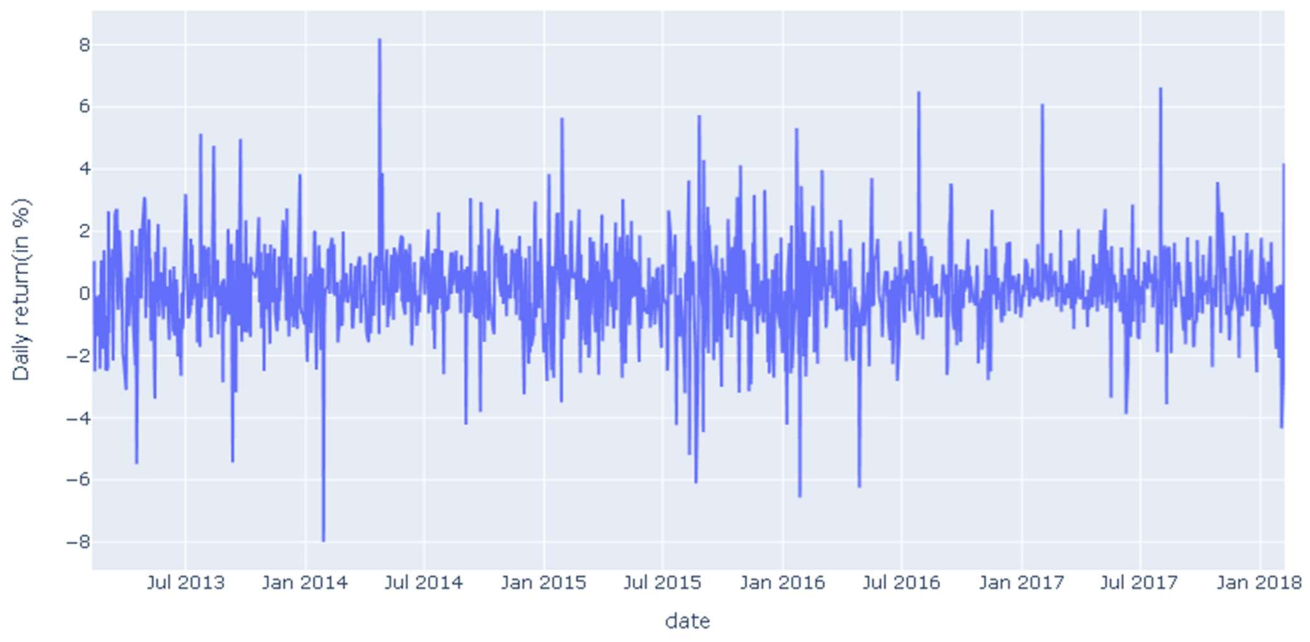
Next, I analyzed the closing prices of one of the company stocks. I used Apple Stock for this analysis.

Daily Stock Return Formula

I subtracted the opening price from the closing price to calculate how much they gained or lost per day for a stock. Then, multiply the result by the number of shares they own in the company.

Here is the sample code that was used in the project:

```
#company_list

apple = pd.read_csv(r'individual_stocks_5yr\\AAPL_data.csv')

apple.head(15)

apple['close']

apple.head(4)


apple['Daily return(in %)'] = apple['close'].pct_change() * 100

## pct_change() returns : Percentage change between the current and a prior
elements

apple.head(4)

import plotly.express as px

px.line(apple , x="date" , y="Daily return(in %)")
## Plotting Line-plot of "date" vs "Daily return(in %)"..
```

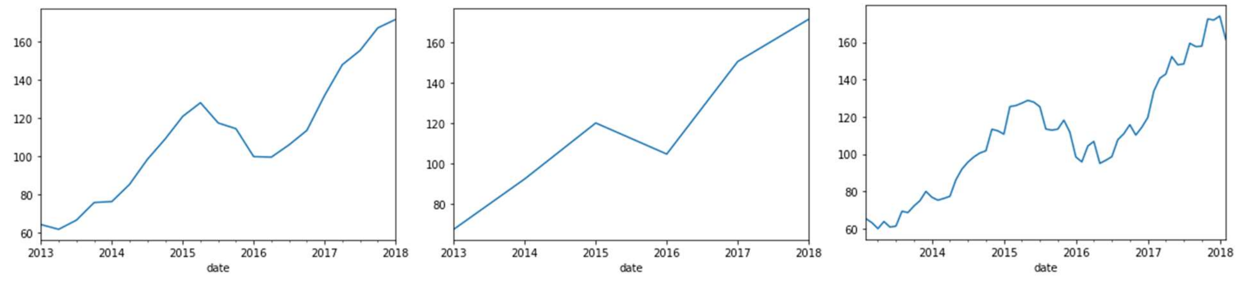


Performing a resampling analysis of the closing price.

Before doing resampling, I first had to make my date feature 'row-index' so that I could resample data on various basis:

a. yearly('Y'),
b. quarterly('Q'),
c. monthly('M'),
d. weekly basis ('W'),
e. Daily basis('D')
f. minutes ('3T'),
g. 30-second bins('30S'),
h. resample('17min')

```
apple.dtypes

apple['date'] =pd.to_datetime(apple['date'])

apple.dtypes

apple.head(4)

apple.set_index('date' , inplace=True)

apple.head(4)

apple['close'].resample('M').mean() ## resample data on monthly basis

apple['close'].resample('M').mean().plot()

apple['close'].resample('Y').mean() ## resample data on Yearly basis

apple['close'].resample('Y').mean().plot()

apple['close'].resample('Q').mean() ## resample data on Quarterly basis ..

apple['close'].resample('Q').mean().plot()
```

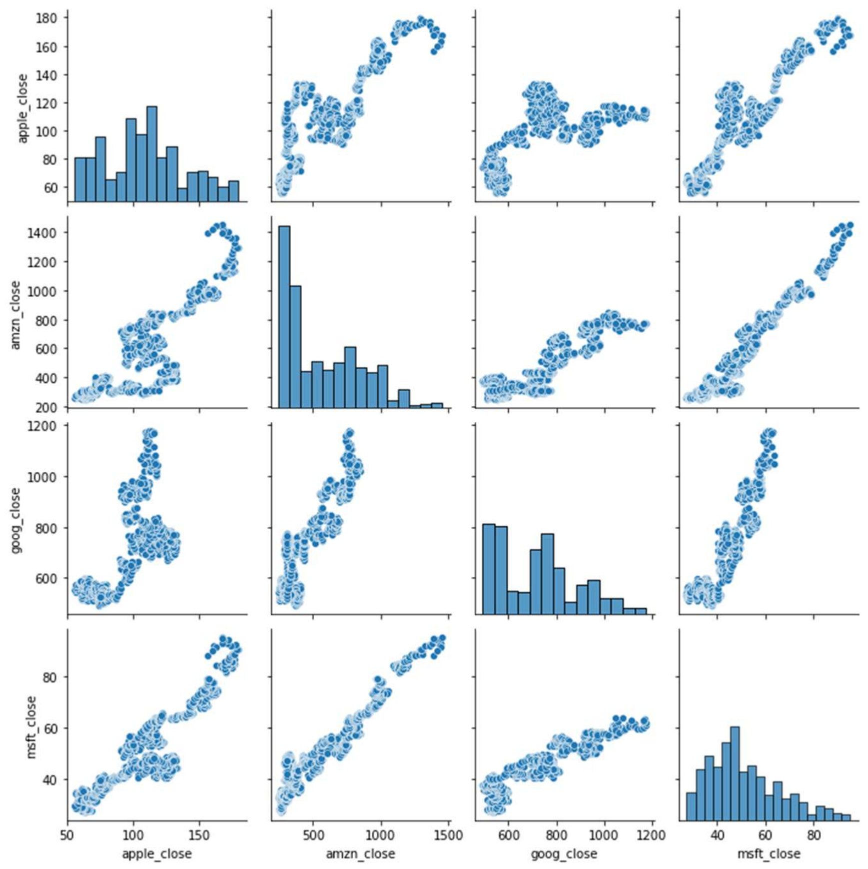


Correlation Analysis Plots

Analyzing whether the closing prices of the tech companies (Apple, Amazon, Google, and Microsoft) are correlated.

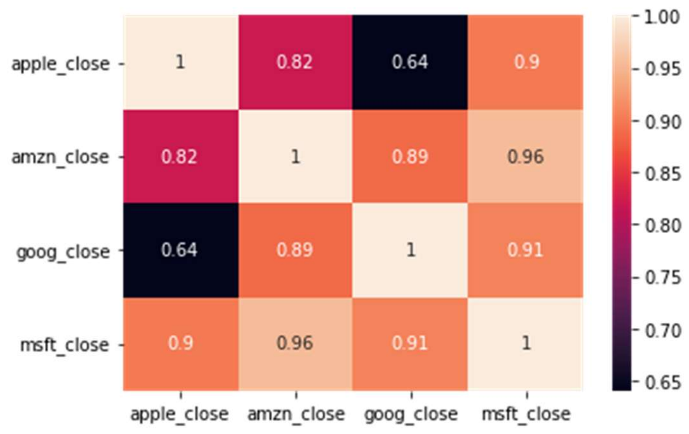
In the next section, I looked at the closing prices of the tech companies to see whether they were correlated.

```
company_list

company_list[0]

app = pd.read_csv(company_list[0])
amzn = pd.read_csv(company_list[1])
google = pd.read_csv(company_list[2])
msft = pd.read_csv(company_list[3])

closing_price = pd.DataFrame()

closing_price['apple_close'] = app['close']
closing_price['amzn_close'] = amzn['close']
closing_price['goog_close'] = google['close']
closing_price['msft_close'] = msft['close']

closing_price


# Pair-plot is all about. We are considering some pairs in the dataset and
trying to plot their scatterplots.
#
#     Unique plots : 4c2 = 6 unique plots
#
#     Total plots : 15 ( 6 unique + 6 mirror images of these
#     unique one + 3 diagonal plots(histogram))

# ## Disadvantages:
# a) Can't be used when the number of features is high.
# b) Cannot visualize higher dimensional patterns in 3-D and 4-D.
# c) Only possible to view 2D patterns.

## NOTE: Each feature's diagonal elements are (histogram).

sns.pairplot(closing_price)
```



The correlation heatmap plot for the stock prices revealed that the closing price of Google and Microsoft are well correlated & Closing prices of Amazon and Microsoft have a co-relation of 0.96.

```
closing_price.corr()

## Co-relation plot for stock prices

sns.heatmap(closing_price.corr() , annot=True)


## Conclusions
# Closing prices of Google and Microsoft are well correlated & Closing prices
of Amazon and Microsoft have a co-relation of 0.96
```



Analyze whether Daily changes in Closing price or Daily Returns in Stock are co-related.

Analyzing the correlation between daily changes in closing price and daily returns of stock can provide insights into how price fluctuations are related to the overall performance of the stock. Daily changes in closing prices can be calculated by taking the difference between consecutive closing prices. Daily returns can be calculated as the percentage change in closing price from the previous day. After calculating daily changes in closing price and daily returns, I performed a correlation analysis to determine the relationship between these two variables.

Since I had used Pairplot already, I used an extension of Pairplot called Pairgrid.

On Pairplot, we have a histogram on diagonals & scatterplot, KDE, and any other plot which tells dist on rest of the plot.

On Pairgrid, once we create a grid, we can set the plot as per our need, i.e., if we have four features, it creates a total of 16 graphs/plots or matrices of 4*4.
There would be various possibilities for the type of plots in our Pairgrid, which we can set as per our needs:

a. All plots can be scatterplot.
b. On the diagonal, we have a histogram & the rest will be a scatterplot.
c. On the diagonal, we have a histogram & the rest will be kdeplot.
d. On the diagonal, we have a histogram & below the diagonal will be a kdeplot & upper diagonal will be a scatterplot.

Note: kdeplot for two features, also known as **contour plots** which returns density.

But kdeplot returns distribution if we are performing univariate analysis; else, it will show density.

```
closing_price

closing_price['apple_close']

closing_price['apple_close'].shift(1)

(closing_price['apple_close'] -
closing_price['apple_close'].shift(1))/closing_price['apple_close'].shift(1)
* 100

for col in closing_price.columns:
    closing_price[col + '_pct_change'] = (closing_price[col] -
closing_price[col].shift(1))/closing_price[col].shift(1) * 100

closing_price

closing_price.columns

clsing_p = closing_price[['apple_close_pct_change', 'amzn_close_pct_change',
       'goog_close_pct_change', 'msft_close_pct_change']]

clsing_p

g = sns.PairGrid(data = clsing_p)
g.map_diag(sns.histplot)
g.map_lower(sns.scatterplot)
g.map_upper(sns.kdeplot)

## Conclusion:
# While Comparing 'AAPL_close_pct_change' to 'AMZN_close_pct_change'  , it
shows a linear relationship upto some extent.

clsing_p.corr()
```

Conclusion and Results

This project is a comprehensive study of stock price analysis and forecasting, showcasing the application of data science techniques to financial data. By analyzing historical stock prices and developing forecasting models, we aim to contribute valuable insights that can aid investors and financial professionals in making informed decisions within the complex landscape of stock markets. The results and methodologies presented in this report underscore the significance of data-driven approaches in finance and investment.

References

Sabare, V. (2023). Stock Price Data Analysis. GitHub. https://github.com/Sabareh/Stock_Price_Data_Analysis/

Alpha Vantage. (n.d.). Alpha Vantage: A Leading Provider of Free APIs for Historical and Real-Time Data on Stocks, Forex, Cryptocurrencies, and Indices. Retrieved from https://www.alphavantage.co/

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference (pp. 51-56).

Developers. (2021). NumPy: The fundamental package for scientific computing with Python. Retrieved from https://numpy.org/

Pandas Development Team. (2021). pandas-dev/pandas: Pandas. Retrieved from https://github.com/pandas-dev/pandas

Seaborn. (n.d.). Seaborn: Statistical Data Visualization. Retrieved from https://seaborn.pydata.org/

Matplotlib Development Team. (2021). Matplotlib: Visualization with Python. Retrieved from https://matplotlib.org/

Plotly Technologies Inc. (n.d.). Plotly: Interactive Data Visualization. Retrieved from https://plotly.com/