Predicting stock prices for large-cap technology companies

INTERNSHIP REPORT (week 1)

Name: Kuldeep Singh (Vellore Institute of Technology)

Github Link:- https://github.com/Kuldeep-Singh-Bithu/Stock-Price-Internship

Exploratory data analysis

Importing Datasets

```
#Amazone stock price prediction
import inline as inline
import matplotlib
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# reading CSV file
s1=pd.read_csv(".\HistoricalData_AMZN.csv")
# seting date as a index
s1=pd.read_csv(".\HistoricalData_AMZN.csv",header=0,index_col="Date",parse_dates=True)
# understanding the data
#s1.head()
print(s1.head(20))
print(s1.tail())
print(s1.describe())
```

Cleaning dataset

```
# cleaning data
missing_val=s1.isnull().sum()
print(missing_val)
# there is no missing value or missing data
# checking for unique value
print(s1.nunique())
```

Output

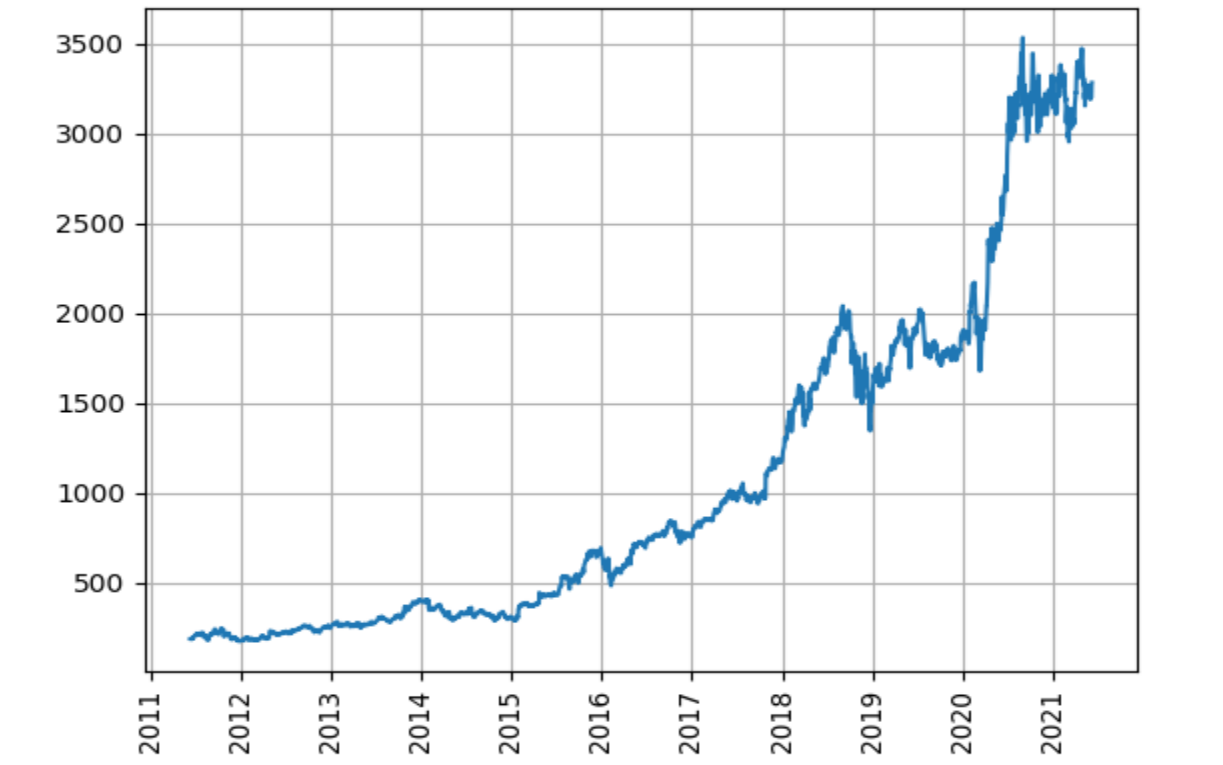
```
        Close/Last        Volume          Open          High           Low
count  2516.000000  2.516000e+03  2516.000000  2516.000000  2516.000000
mean   1063.464483  4.207630e+06  1063.903971  1074.999662  1051.288883
std     929.012066  2.295011e+06   930.056079   940.351050   917.970041
min     173.100000  8.813370e+05   169.620000   174.550000   166.970000
25%     306.772500  2.737498e+06   306.977500   310.787500   303.415000
50%     715.610000  3.615334e+06   715.610000   720.155000   711.240000
75%    1735.715000  4.936622e+06  1737.780000  1750.167500  1713.767500
max    3531.450000  2.412413e+07  3547.000000  3554.000000  3486.690000
Close/Last    0
Volume        0
Open          0
High          0
Low           0
dtype: int64
Close/Last    2483
Volume        2515
Open          2427
High          2451
Low           2464
dtype: int64
```

Time Series Data

A time series is a sequence of numerical data points taken at successive equally spaced points in time.In investing. a time series tracks the movement of stock price, over a specified period of time.

```
# Time serise Data
plt.plot(s1.index, s1["Close/Last"])
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y"))
plt.gca().xaxis.set_major_locator(mdates.YearLocator()) # formatting data for plotting
plt.grid(True) # to read data more clearly on the graph
plt.xticks(rotation=90) # rotating dates to 90 degree so that we can read more clearly
#plt.show()
plt.savefig("ClosePriceOnGraphEDA.png")
```

Output



Observation:

Stock price of amazon has increased from 2011 to mid of 2018. After that there is a fall in the price of stocks for 6 months but from the start of 2019 stocks are continuously increasing.

Resampling

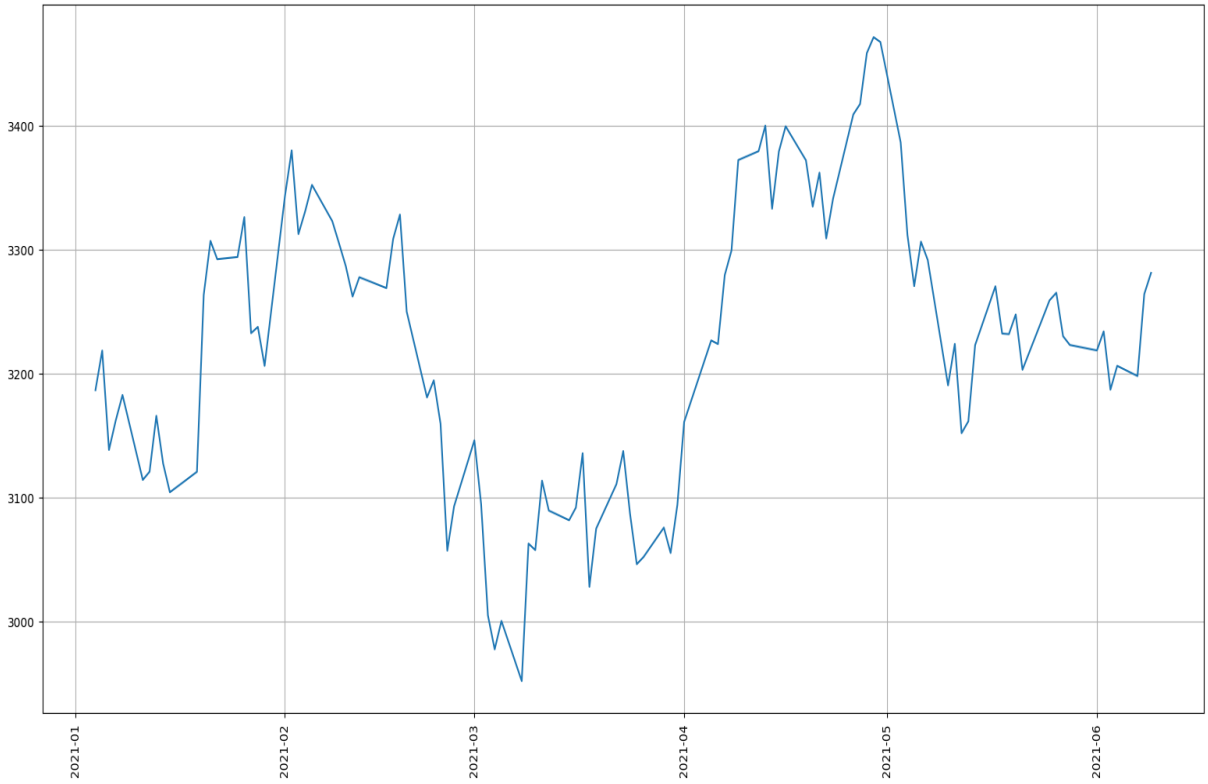
Resampling involves changing the frequency of your time series observations.

Zooming-in

- **resampling of data**
- **Zooming into data set**
- **Zooming in to see data for the year 2021 till current day**

```
#Amazone stock price prediction
import inline as inline
import matplotlib
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# reading CSV file
s1=pd.read_csv(".\HistoricalData_AMZN.csv")
# seting date as a index
s1=pd.read_csv(".\HistoricalData_AMZN.csv",header=0, index_col="Date", parse_dates=True)
# resampling of data
# Zooming into data set
# Zooming in to see data for the year 2021 to current day
s1_21=s1.loc[("2021-01-01"):("2021-06-09")]
plt.plot(s1_21.index, s1_21["Close/Last"])
plt.grid(True)
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.gca().xaxis.set_major_locator(mdates.MonthLocator())
plt.xticks(rotation=90)
#plt.show()
plt.savefig("zooming.png")
```

Output



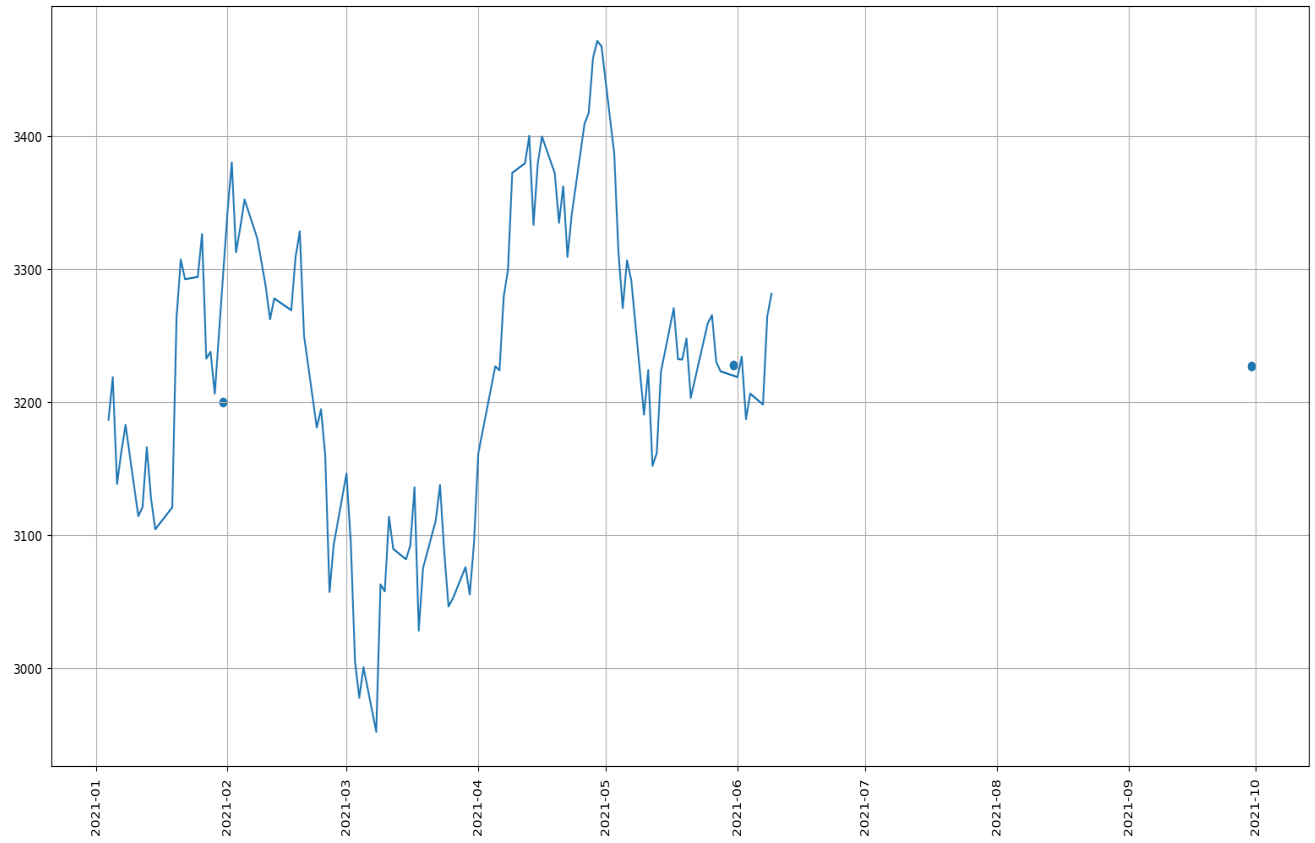
Observation:

We can observe that there is fall in the price of stock between month of march and april And after that in the month of may the price of stock fell a little bit. We can say that prices are fluctuating.

Resampling (Quarterly)

```
 # Resampling(Quarterly)
monthly_s1_21=s1_21.resample("4M").mean()
plt.scatter(monthly_s1_21.index,monthly_s1_21["Close/Last"])
plt.grid(True)
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m"))
plt.gca().xaxis.set_major_locator(mdates.MonthLocator())
plt.xticks(rotation=90)
plt.show()
plt.savefig("resamplingQuarterly.png")
```

Output



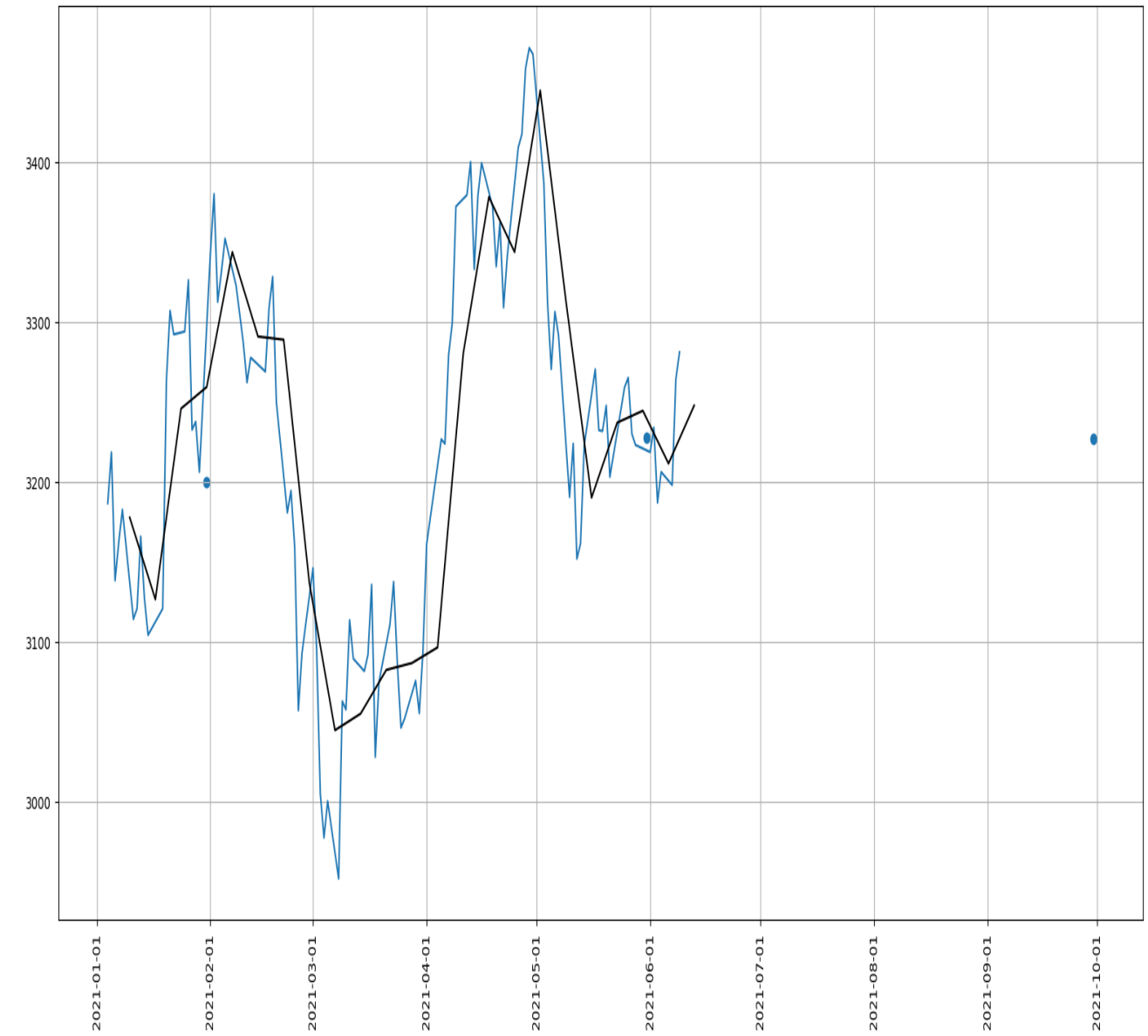
Observation:

By taking the mean of price for every 4 months of the year we can say that stock price is increasing little by little. Dots represent the price of stocks for 4 months on the graph.

Resampling (weekly)

```
# Resampling (Weekly)
weekly_s1_21=s1_21.resample("W").mean()
print(weekly_s1_21.head())
plt.plot(weekly_s1_21.index,weekly_s1_21["Close/Last"],"-0")
plt.grid(True)
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.gca().xaxis.set_major_locator(mdates.MonthLocator())
plt.xticks(rotation=90)
#plt.show()
plt.savefig("resamplingWeekly.png")
```

Output



Observation:

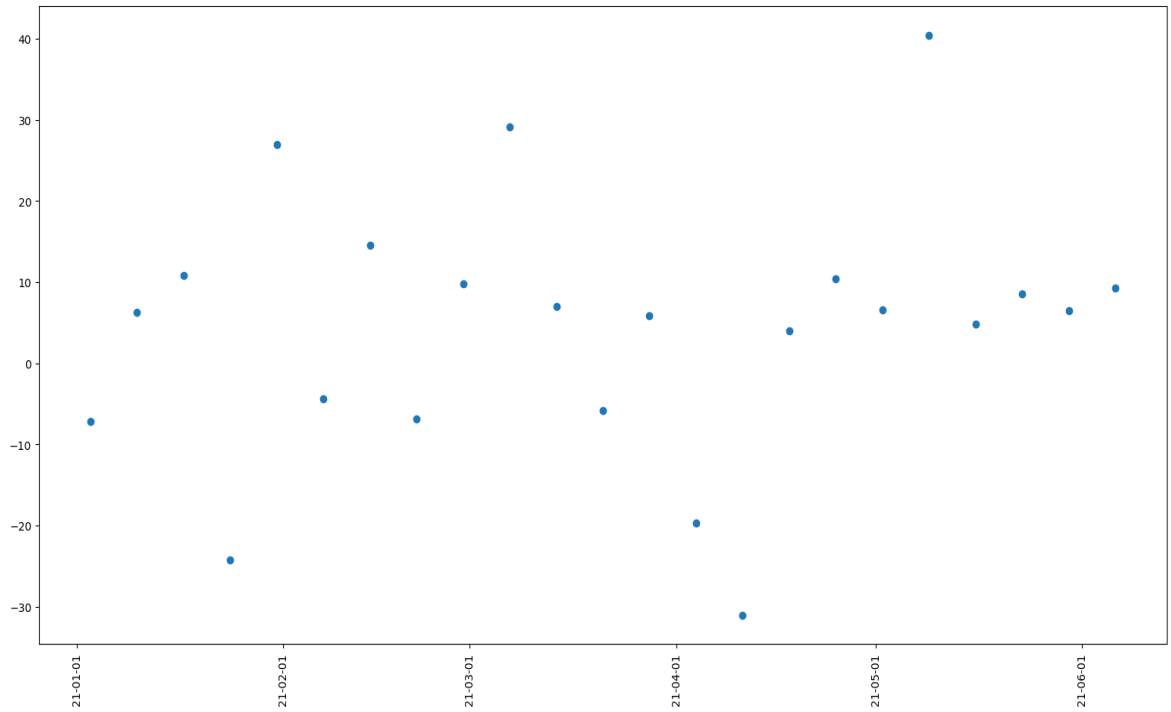
By taking the mean of the price for every week. We can say that the price of stock is fluctuating, there is an increase in the price and then there is a fall in the price after that again price increase. That's why price is fluctuating.Blck line represent weekly value of stocks.

Analysing Difference between Levels (Resampling Weekly)

- **Analysing Difference between Levels(Resampling Weekly)**
- **Difference Between Open value and close value**
- **this will give better pitcher for examine the company whether profit or lose**

```
#Amazone stock price prediction
import inline as inline
import matplotlib
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# reading CSV file
s1=pd.read_csv(".\HistoricalData_AMZN.csv")
# seting date as a index
s1_2=pd.read_csv(".\HistoricalData_AMZN.csv",header=0, index_col="Date",parse_dates=True)
# resampling of data
# Analysing Difference between Levels(Resampling Weekly)
# Difference Between Open value and close value
# this will give better pitcher for examine the company whether profit or lose
s1_2["diff"]=s1_2["Open"]-s1_2["Close/Last"]
s1_diff=s1_2.resample("W").mean()
plt.scatter(s1_diff.loc["2021-01-01":"2021-06-09"].index, s1_diff.loc["2021-01-01":"2021-06-09"]["diff"])
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%y-%m-%d"))
plt.gca().xaxis.set_major_locator(mdates.MonthLocator())
plt.xticks(rotation=90)
#plt.show()
plt.savefig("DifferenceBetweenOpenCloseResamplingWeekly.png")
```

Output



Observation:
We can say that profit is not predictable for a short amount of time (week,month). There is a chance of gain and loss at the same time if we buy stock for a short period of time.But for long term there is very less chance of getting loss and profit is not that much good. So we can say that it is neutral.

Moving Windows

- **Moving windows are there when you compute the statistic on a window of data represented by a particular period of time and then slide the window across the data by a specified interval. That way, the statistic is continually calculated as long as the window falls first within the dates of the time series.**

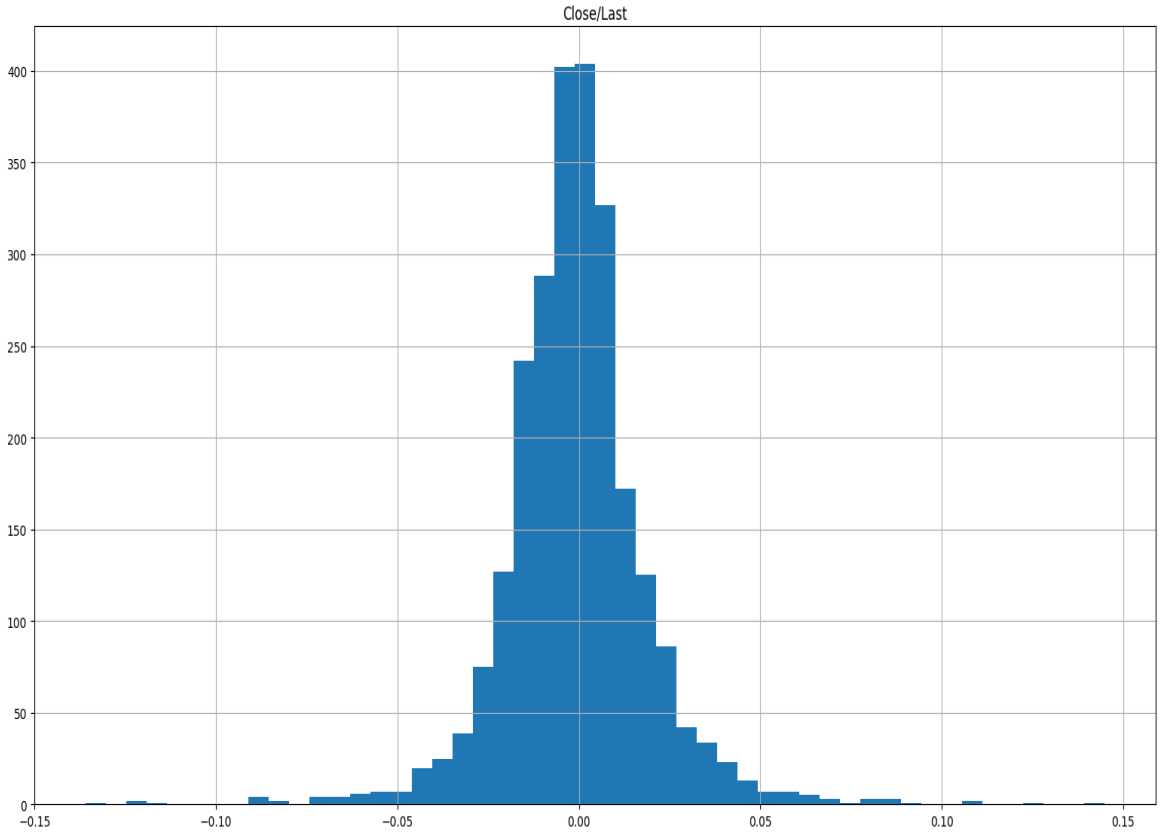
- **A rolling mean smoothens out short-term fluctuations and highlight longer-term trends in data.**

Moving windows

to analyse long term trends with respect to Daily percentages

```python
#Amazone stock price prediction
import inline as inline
import matplotlib
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# reading CSV file
s1=pd.read_csv(".\HistoricalData_AMZN.csv")
# seting date as a index
s1_2=pd.read_csv(".\HistoricalData_AMZN.csv",header=0, index_col="Date",parse_dates=True)
# Moving windows
# to analyse long term trends
# with respect to Daily percentages
daily_close_s1=s1_2[["Close/Last"]]
# daily returns
# pct means percentage
daily_pct_change_s1=daily_close_s1.pct_change()
# replacing NA values with 0
daily_pct_change_s1.fillna(0,inplace=True)
print(daily_pct_change_s1.head())
# lets plot the histogram
daily_pct_change_s1.hist(bins=50)
plt.show()
plt.savefig("percentageChangeHistogram")
```

Output



Observation:

By analyzing daily percentage we can say that there is equal chance of getting loss and profit at the same time for year 2021. But there is a very little chance of getting profit.It can neither lose you money nor make you money. it's neutral.

Volatility

- **The volatility of a stock is a measurement of the change in variance in the returns of a stock over a specific period of time.**

- **To compare the volatility of a stock with another stock to get a feel for which may have less risk or to use a market index to examine the stock's volatility in the overall market.**

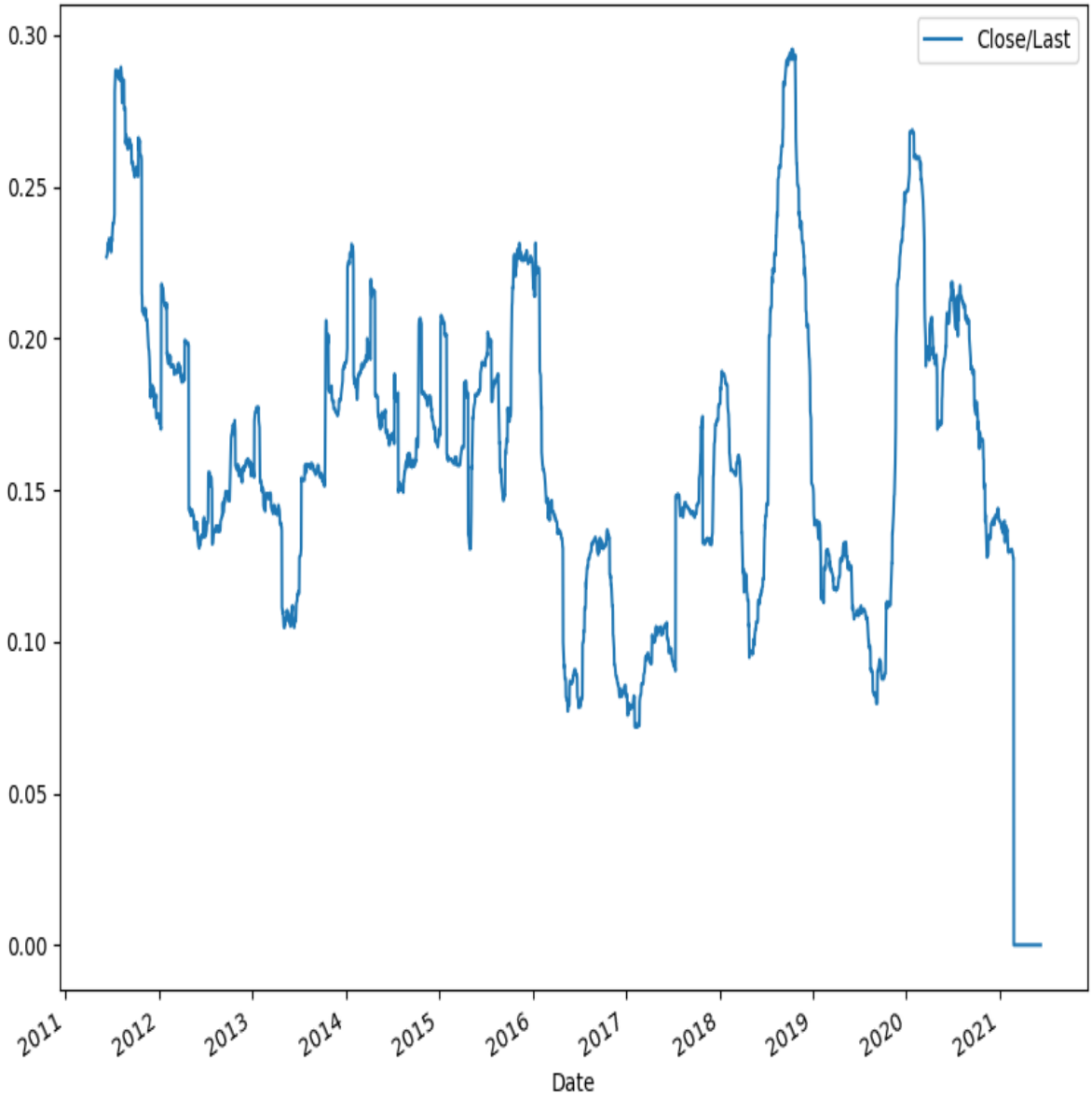
- **Generally, the higher the volatility, the riskier the investment in that stock.**

- **measurement of the change in variance in the returns of stock over specific period of time**
- **Higher the volatility Higher the risk**

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
# reading CSV file
s1=pd.read_csv(".\HistoricalData_AMZN.csv")
# seting date as a index
s1_2=pd.read_csv(".\HistoricalData_AMZN.csv",header=0, index_col="Date",parse_dates=True)
# Moving windows
# to analyse long term trends
# with respect to Daily percentages
daily_close_s1=s1_2[["Close/Last"]]
# daily returns
# pct means percentage
daily_pct_change_s1=daily_close_s1.pct_change()
# replacing NA values with 0
daily_pct_change_s1.fillna(0,inplace=True)
print(daily_pct_change_s1.head())

min_periods=75
# Calculating the Volatility
# Calculating variance of these dail percentage values
vol=daily_pct_change_s1.rolling(min_periods).std()*np.sqrt(min_periods)
vol.fillna(0,inplace=True)
print(vol.tail())
# Plotting the Volatility
vol.plot(figsize=(10,8))
plt.show()
plt.savefig("VolatilityGraph.png")
```

Output

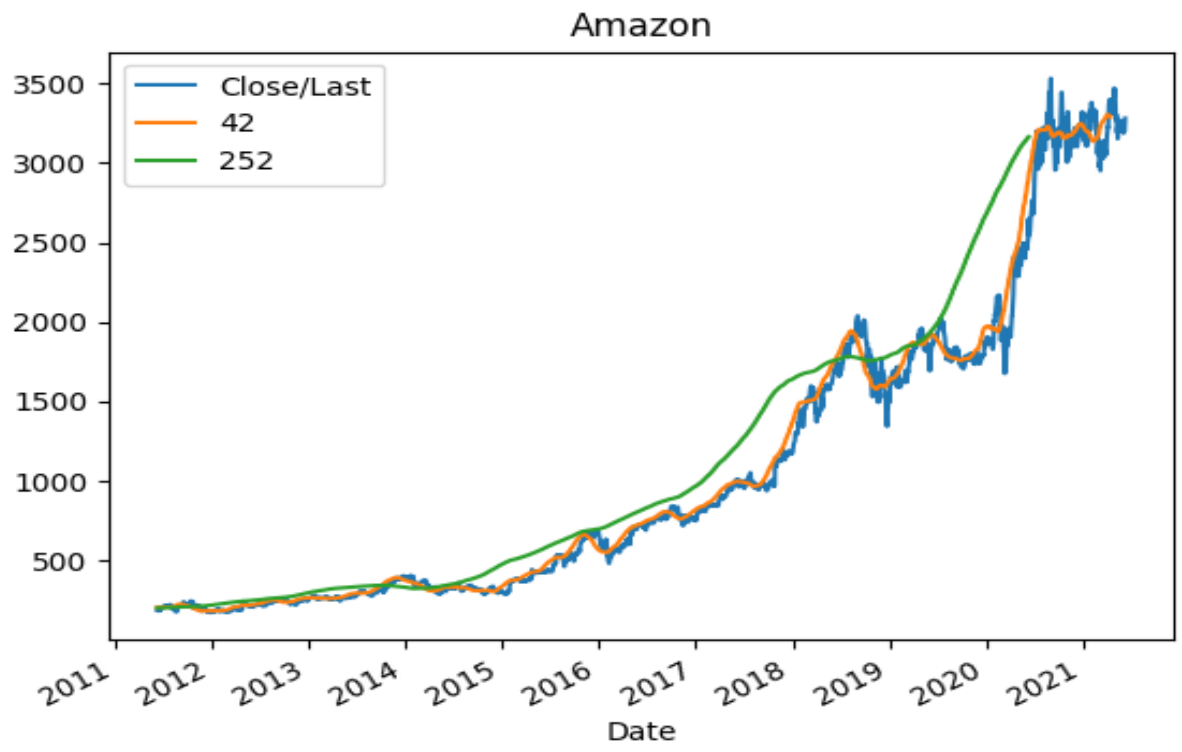


This is the volatility of Amazon stock price from 2011 to 2021.Generally, the higher the volatility, the riskier the investment in that stock.we can say that from 2020 to 2021 volatility is very less and there is a chance of getting profit.

Rolling means(Trend and Seasonality)

```
# Rolling means(Trend and Seasonality)
s1_close_px=s1_2["Close/Last"]
# Short term moving window rolling mean
s1_2["42"]=s1_close_px.rolling(window=40).mean()
# Long term moving window rolling mean
s1_2["252"]=s1_close_px.rolling(window=252).mean()
# plot the  closing price,the short and long terms windows of rolling means
print(s1_2.head())
s1_2[["Close/Last","42","252"]].plot(title="Amazon")
#plt.show()
plt.savefig("TrendAndSeasonality.png")
```

Output



Observation:

Orange line represents trend and green line represents seasonality and blue line represents normal stock price.i observe that if we buy stock for short time like few weeks and 2 to 3 months there is a chance of getting lose but when you buys stock for long time there is more chance of getting profit.And seasonality graph is most of the time rise more than trends graph.