CSC-632 HW3_Yunting Chiu

March 24, 2021

1 Instruction

For this assignment submit a single file, either MS WORD or PDF in addition to your Python codes with descriptions. The Python codes need to have comments. The assignments must be done individually. Format Requirements for all assignments:

- No page limits
- Single spaced, 12-point or larger font size; 1-inch margins
- Use headers and/or bullets to organize and convey key elements, and page numbers
- Only Latin alphabet characters are allowed (i.e., do not include any words or phrases that contain non-English characters)
- File type: Adobe PDF (recommended) or Word document

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

1- (2 points) Definitions: Use our textbook, online sources, other books, handouts, and library materials (available virtually at American University Library Website) to answer the following questions.

a) What is "security" in the financial domain?

- A security is a tradable financial asset. There are three examples of the security part in the US:

1. Equity securities – e.g. stock
2. Debt securities – e.g. bonds and banknotes
3. Derivatives – e.g. options and futures

b) Define "drift" and explain about this concept in financial engineering.

- The definition of drift is that individual securities in the portfolio appreciate or depreciate in value over time, causing them to drift away from their original allocations, the following formula is `Drift = ( Difference of security 1 + ...... + Difference of security n) / 2`.

c) What is Brownian Motion

- Brownian motion is a simple continuous stochastic process. It is widely used in finance and physics to model random behavior.

d) Explain about the use of Lognormal in financial engineering and asset prices. How does the transformation of "return" from actual prices to logarithmic value of return helps the prediction of stock prices?

- As changes in the natural logarithm are approximately equal to percentage changes in the original series. In statistics, we mainly consider using logarithm for data transformation in priority. The result of lognormal distribution is close to the result of normal distribution.
- The main point of using lognormal distribution in financial engineering is that it has a zero lower bound. In stock marketing, we cannot predict that the stock price is below zero, meaning that the lognormal distribution is an appropriate method to support stock market forecasting.

1.0.1 References

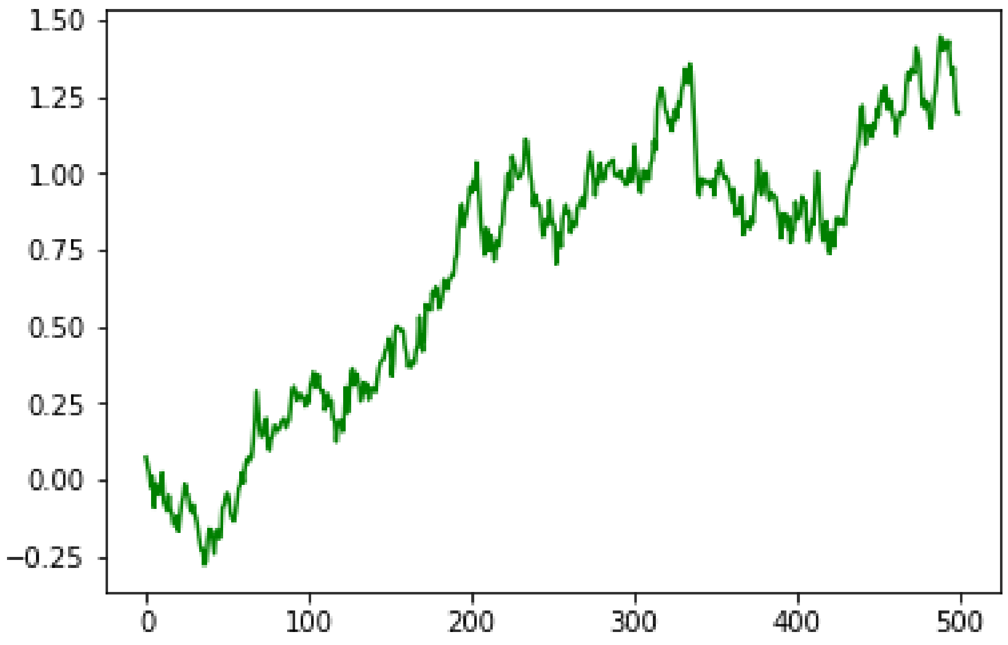
1. https://en.wikipedia.org/wiki/Security_(finance)
2. https://corporatefinanceinstitute.com/resources/knowledge/finance/security/
3. https://ycharts.com/glossary/terms/drift#:~:text=Drift%20occurs%20as%20individual%20securities,their%
4. https://financetrain.com/why-lognormal-distribution-is-used-to-describe-stock-prices/

2- (1 points) Generate a simple Brownian motion in the Python environment. To this end, use n=500 time period and generate random variable to create a Brownian Motion Graph. Explain the relationship between the graph you provided and price fluctuations in stock market. - After the model has been executed 500 times, the ROA (return on assets) is positive. The return on investment (ROI) will between 1.00 to 1.25.

The following Equation in Python

$$Yn(t) = \frac{1}{\sqrt{(n)}} * \sum_{k} Zk$$

[2]:
```python
np.random.seed(1) # random number will be reproducible
n = 500

SQN = 1/np.math.sqrt(n)
ZValues = np.random.randn(n) # return samples from normal distribution: N~(mean␣
 ↪= 0, sd = 1)

Yk = 0
SBMotion = []

for k in range(n):
    Yk = Yk + SQN*ZValues[k]
    SBMotion.append(Yk)

plt.plot(SBMotion, "g")
plt.show()
```



3- (2 points) Re-do the Amazon stock price trend for Google that can be found in the following link. Explain the details of your analysis (https://finance.yahoo.com/quote/GOOG/history/).

a) Select a 5-year time period of your choice and depict the Google shares graph similar to what the book provided in Figure 8.3.

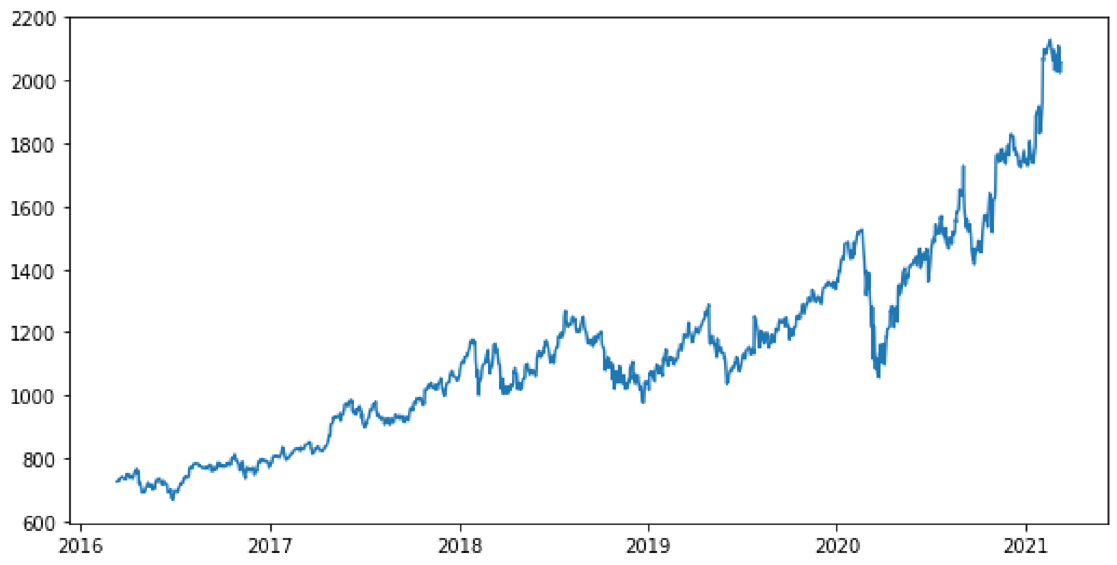
GOOG has grown significantly over the last five years. The plot below shows that the price has surpassed 2000 dollars per share. - The stock of google (GOOG) has seen a huge rise over the last five years. - The dataset is from Mar 10, 2016 to Mar 10, 2021.

[3]:
```
from google.colab import files
upload = files.upload()
```

```
<IPython.core.display.HTML object>
```

```
Saving GOOG.csv to GOOG.csv
```

[4]:
```
goog = pd.read_csv("GOOG.csv", header = 0, usecols = ["Date", "Close"],␣
 ↪parse_dates = True, index_col = "Date")
# print(goog.count())
# check each variable type and Data Analysis & Exploratory Data Analysis (EDA)
# print(GOOG.info()_GOOG
# print(GOOG.describe()) # EDA
plt.figure(figsize=(10,5))
plt.plot(goog)
plt.show()
```



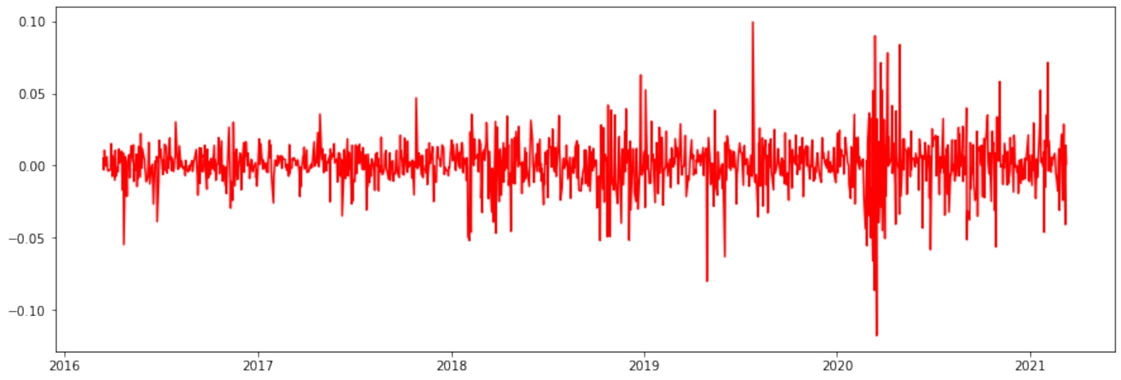
b) For the same time provide the Logarithmic Value of the returns similar to Figure 8.4. Provide the details of your analysis and include adequate comments in your Python code.

- We can see that the drift of the logarithmic scale is very similar to the original series, and the data points are follow a normal distribution ($\mu = 0$). Also, the mean is stable.

[5]:
```
# percentage change between the current and a prior element
googPctChange = goog.pct_change()
#print(googPctChange)

# return on a logarithmic scale with plot
googLogReturns = np.log(1+googPctChange) # 1: today + multiple the change of␣
 ↪percentge = total money
print(googLogReturns.tail(10))
plt.figure(figsize = (15, 5))
plt.plot(googLogReturns, "r")
plt.show()
```

```
               Close
Date
2021-02-25 -0.030929
2021-02-26  0.002704
2021-03-01  0.021684
2021-03-02 -0.002728
2021-03-03 -0.023952
2021-03-04  0.010982
2021-03-05  0.028600
2021-03-08 -0.040836
2021-03-09  0.013996
```

```
2021-03-10    0.001134
```



4- (1 points) Explain about applying Monte Carlo Simulation to estimate stock price provided in Page 216 of the book. As the procedure is already explained, you only need to provide steps using which the estimation can be performed. - In general, Monte Carlo Simulation is using random sampling to obtain numerical results. The main idea is to utilize randomness to solve problems. In textbook, Brownian motion can evaluate the daily stock price basis of the previous record. The following equation is:

$$StockPrice(t) = StockPrice(t-1) * e^r$$

whrere `e` is a exponential function and `r` is a coefficient of periodic rate. - It's hard to estimate the rate of return of an asset, so we are using a **random number** to be an alternative. Because BS equations are based on stochastic movements so Brownian motion can be used for modeling behavior over time. - Brownian motion is made up of two parts: `Drifts` and `Volatility`.

Reference: - https://medium.com/analytics-vidhya/monte-carlo-simulations-for-predicting-stock-prices-python-a64f53585662 - https://learning.oreilly.com/library/view/hands-on-simulation-modeling/9781838985097/B15737_08_Final_JM_ePub.xhtml

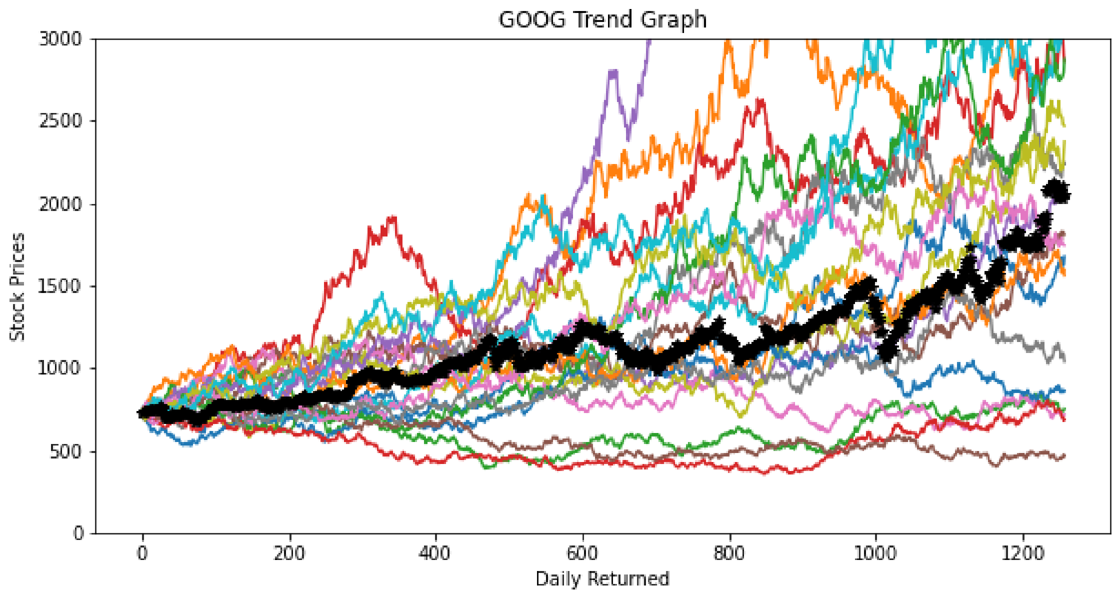
5- (2 points) Use the data in Problem 3 above to evaluate the first (drift) and second (daily change) components of the Black-Scholes formula. Prepare the Google trend graph similar to Figure 8.5.

- We estimated 20 trends of GOOG historical drift and volatility in recent five years.

[6]:
```
# Black-Scholes model (BS) - stocks, currencies, and interest rates
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import norm
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()

MeanLogReturns = np.array(googLogReturns.mean()) # mean
# print(MeanLogReturns)

VarLogReturns = np.array(googLogReturns.var()) # variance
```

```
StdevLogReturns = np.array(googLogReturns.std()) # standard deviation

# we need the mean and variance to evaluate the drift
Drift = MeanLogReturns - (0.5 * VarLogReturns) # why -0.5? Because historical␣
 ↪values are eroded in the future
print("Drift = ",Drift)

# set the number of intervals and iterations
NumIntervals = int(goog.count()) # equal to the number of the observations: 1258
Iterations = 20 # the number of simulation models that we intend to develop,␣
 ↪that is, 20 simulated examples

# generate the random distribution, will return 1258*20 matrix
np.random.seed(1)
SBMotion = norm.ppf(np.random.rand(NumIntervals, Iterations))
"""
np.random.rand: Create an array of the given shape and populate it with random␣
 ↪samples from a uniform distribution over [0, 1): return 1258 rows*20 columns
norm.ppf: The .ppf() function calculates the probability for a given normal␣
 ↪distribution value. That is, inverse of a normal distribution.
"""

DailyReturns = np.exp(Drift + StdevLogReturns * SBMotion) # A positive return␣
 ↪means the stock has grown in value
# print(DailyReturns)

# Start to predict the prices
StartStockPrices = goog.iloc[0] # extract the specific observation, recover the␣
 ↪first value
# print(StartStockPrices)
StockPrice = np.zeros_like(DailyReturns) # return an array of zeros with the␣
 ↪same shape and type as a given array
StockPrice[0] = StartStockPrices

for time in range(1, NumIntervals):

    StockPrice[time] = StockPrice[time - 1] * DailyReturns[time] # BS model

# Plot
plt.figure(figsize=(10,5))
plt.plot(StockPrice)
# add actual trend as Google stock prices in the last 5 years as black line
googTrend = np.array(goog.iloc[:, 0:1])
plt.plot(googTrend,'k*') # "k*" = * boad line, k = balck color
plt.xlabel("Daily Returned"); plt.ylabel("Stock Prices"); plt.title("GOOG Trend␣
↪Graph")
plt.ylim(0, 3000) # we only want to focus on the most part of simulating result
plt.show()
```

```
Drift =  [0.0006866]
```



[7]:
```
# testing zone
from scipy.stats import norm
print(norm.ppf(0.95))
print(norm.cdf(1.6448536269514722))
```

```
1.6448536269514722
0.95
```

6- (2 points) Estimating the VaR for some S&P 500 assets by doing the following steps:

a) S&P 500 is an important stock market index. Using online sources (with reference) provide a two-paragraph description of the index including the history and major assets.

- The S&P 500 is a stock market index that monitors the performance of 500 major companies that are listed on US stock exchanges (NYSE and NASDAQ). It is one of the most widely tracked stock market indices. The index can be used to predict the direction of the economy.
- The S&P 500 was introduced by Standard & Poor's in 1957. The SP 500 index reached nearly 700 points within its first decade, representing the post-World War II economic boom. The index gradually fell from 1969 to early 1981, finally dipping below 300, because the US economy struggled with stagnant growth and high inflation. Then, in 2007 - 2009, the S&P 500 dropped 57.7% during the financial crisis that became known as the Great Recession. By

March 2013, the S&P had recoved all of its losses from the financial crisis, and it had climbed more than 400% over the previous decade to new all-time highs. Thus, we can conclude that S&P 500 is a good indicator of the economy based on the history.

1.0.2 References:

- https://en.wikipedia.org/wiki/S%26P_500
- *Investopedia.* (2020, September 16). What is the history of the S&P 500. https://www.investopedia.com/ask/answers/041015/what-history-sp-500.asp.

b) Using the following stock list, retrieve the data from Yahoo! Finance and provide the statistics of the Portfolio similar to Figure 8.7.

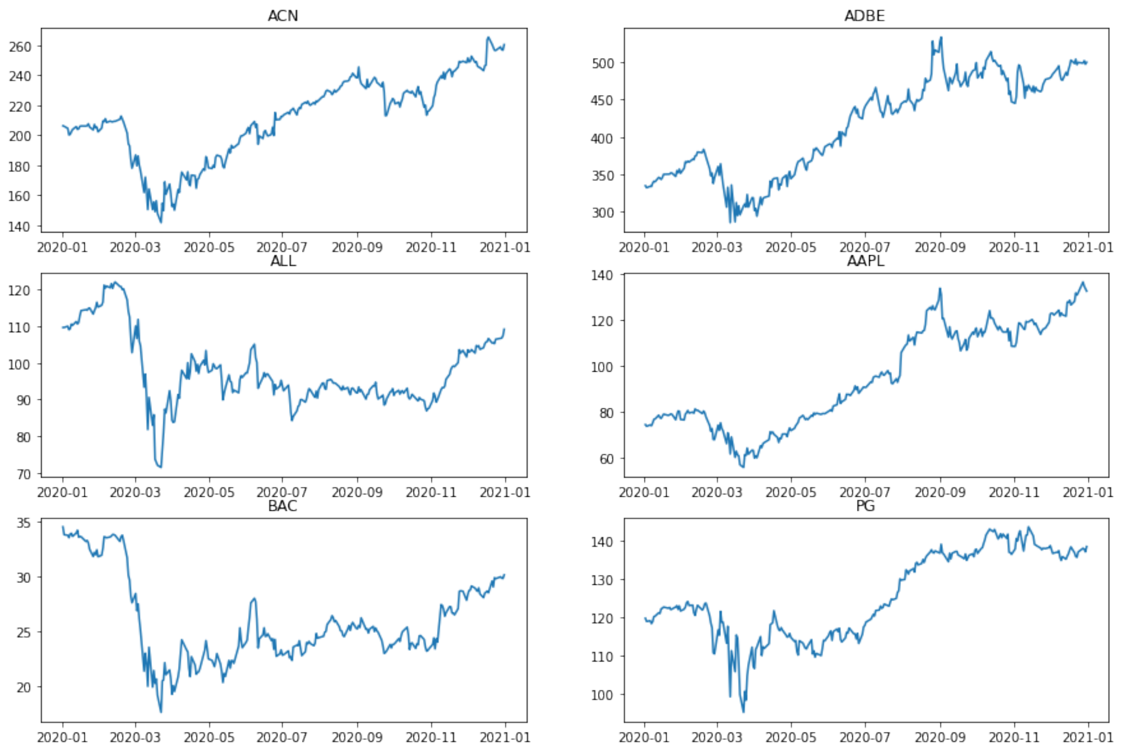
StockList = {ACN, ADBE, ALL, AAPL, BAC, PG}

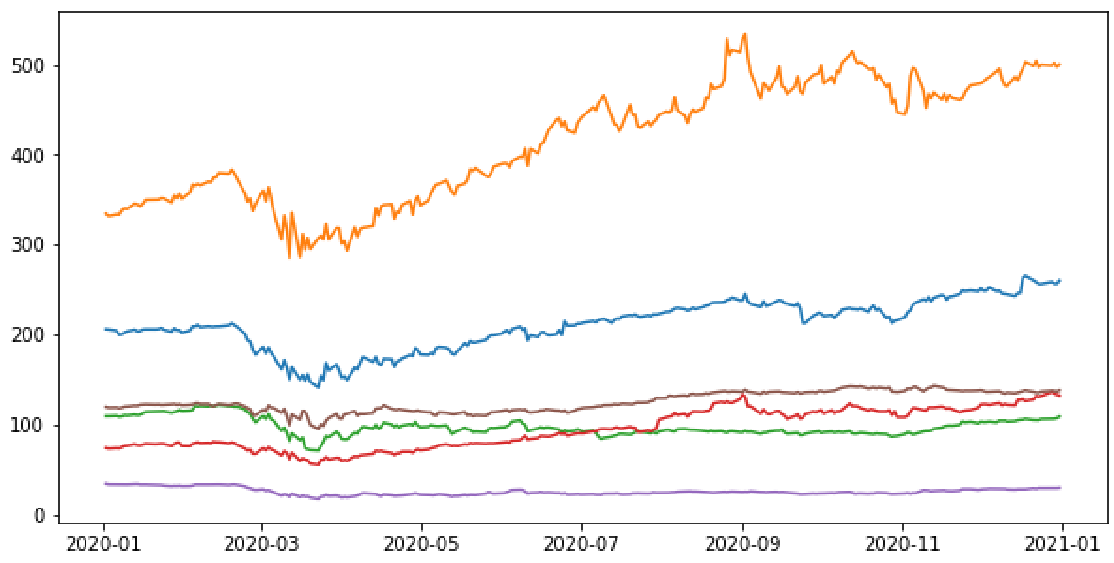
[8]:
```
import datetime as dt # manipulating dates and times
import numpy as np
import pandas_datareader.data as wb # extract data from various Internet␣
↪sources into a pandas DataFrame
import matplotlib.pyplot as plt
from scipy.stats import norm

StockList = ["ACN", "ADBE", "ALL", "AAPL", "BAC", "PG"]
StartDay = dt.datetime(2020, 1, 1)
EndDay = dt.datetime(2020, 12, 31)

StockData =  wb.DataReader(StockList, "yahoo", StartDay, EndDay) # from Yahoo␣
↪data and select StartDay to EndDay observations
StockClose = StockData["Adj Close"] # Adjusted Closing Price (variable) from␣
↪yahoo
print(StockClose.describe())
# print(type(StockClose))
# print(StockClose.count()) # 253 workdays

# plot
plt.tight_layout()
fig, axs = plt.subplots(3, 2, figsize=(15, 10))
axs[0, 0].plot(StockClose['ACN'])
axs[0, 0].set_title('ACN')
axs[0, 1].plot(StockClose['ADBE'])
axs[0, 1].set_title('ADBE')
axs[1, 0].plot(StockClose['ALL'])
axs[1, 0].set_title('ALL')
axs[1, 1].plot(StockClose['AAPL'])
axs[1, 1].set_title('AAPL')
axs[2, 0].plot(StockClose['BAC'])
axs[2, 0].set_title('BAC')
axs[2, 1].plot(StockClose['PG'])
axs[2, 1].set_title('PG')
```

```
plt.figure(figsize=(10,5))
plt.plot(StockClose)
plt.show()
```

```
Symbols         ACN        ADBE         ALL        AAPL         BAC          PG
count    253.000000  253.000000  253.000000  253.000000  253.000000  253.000000
mean     210.914809  415.971266   97.689952   94.901985   25.797423  125.390345
std       27.779251   66.101381    9.844359   21.921424    3.861900   11.294401
min      141.483017  285.000000   71.498421   55.661041   17.632553   95.336548
25%      194.072678  351.660004   91.463394   76.630219   23.298763  115.972321
50%      213.064743  433.010010   94.898758   91.173805   24.672913  122.858360
75%      231.946030  476.260010  103.570297  115.378235   28.228962  136.622757
max      265.343262  533.799988  121.998520  136.486053   34.537598  143.414948
```

```
<Figure size 432x288 with 0 Axes>
```





c) Assess the investment risk of a substantial portfolio of stocks of these companies. Provide detailed explanations.

According to the ouput below, with a 95 % confidence interval, if the value of `Value-at-Risk` was low, it indicates that there was more potential risk in investing in the stock. The stock with the highest risk return was Bank of America (BAC) . If we invest one million dollars in BAC in 2020, we could lose up to 3816.07 dollars at most. On the other hands, the least risky option was the Procter & Gamble Company (PG) with a 95 percent confidence interval.

The following investment targets are ranked from high to low risk is: `BAC`, `ADBE`, `AAPL`, `ALL`, `ACN`, `PG`.

1.0.3 References:

- https://stackoverflow.com/questions/60699836/how-to-use-norm-ppf

[9]:
```
StockReturns = StockClose.pct_change() # the percentage difference between the␣
 ↪current close price and the previous value is returned.
# print(StockReturns.tail(15)) # last 15 days returned
print(StockReturns)

# start to analyze the investment risk
PortvolioValue = 1000000 # one million dollars
ConfidenceValue = 0.95 # 95 % confidence inteval
MeanStockRet = np.mean(StockReturns) # mean
StdStockRet = np.std(StockReturns) # sd

WorkingDays2020 = 253. # Yahoo website contained 253 records that we extracted
AnnualizedMeanStockRet = MeanStockRet/WorkingDays2020
AnnualizedStdStockRet = StdStockRet/np.sqrt(WorkingDays2020)
```

```
# calculate the VaR (Value at Risk)
# INPD = dangerous time, below (1- ConfidenceValue)
INPD = norm.ppf(1-ConfidenceValue, AnnualizedMeanStockRet,␣
↪AnnualizedStdStockRet) # percentage point function = ppf = inverse normal␣
↪distribution (left-tail)
VaR = PortvolioValue * INPD

RoundVaR= np.round_(VaR, 2)
orderValue = []
for i in range(len(StockList)):
    orderValue.append(RoundVaR[i])
    print("Value-at-Risk for", StockList[i], "is equal to ", RoundVaR[i])
print(sorted(orderValue))
```

```
Symbols          ACN      ADBE       ALL      AAPL       BAC        PG
Date
2020-01-02       NaN       NaN       NaN       NaN       NaN       NaN
2020-01-03 -0.001666 -0.007834  0.000089 -0.009722 -0.020763 -0.006726
2020-01-06 -0.006530  0.005726  0.002925  0.007968 -0.001433  0.001387
2020-01-07 -0.021590 -0.000959 -0.008572 -0.004703 -0.006600 -0.006192
2020-01-08  0.001962  0.013438  0.002763  0.016086  0.010110  0.004263
…                …         …         …         …         …         …
2020-12-24 -0.000544  0.005937  0.011322  0.007712 -0.002995  0.010122
2020-12-28  0.009250 -0.001820  0.002612  0.035766  0.005674  0.006971
2020-12-29 -0.007394  0.006333  0.000465 -0.013315 -0.003983 -0.001875
2020-12-30 -0.000854 -0.009281  0.005859 -0.008527 -0.001000 -0.004696
2020-12-31  0.014250  0.005367  0.016459 -0.007703  0.011007  0.009944

[253 rows x 6 columns]
Value-at-Risk for ACN is equal to  -2712.14
Value-at-Risk for ADBE is equal to  -3112.03
Value-at-Risk for ALL is equal to  -2793.75
Value-at-Risk for AAPL is equal to  -3025.31
Value-at-Risk for BAC is equal to  -3816.07
Value-at-Risk for PG is equal to  -2109.83
[-3816.07, -3112.03, -3025.31, -2793.75, -2712.14, -2109.83]
```