# Assignment_7

March 19, 2022

[2]:
```
import pandas as pd
import numpy as np
from tiingo import TiingoClient
import datetime

_MY_SECRET_API_KEY = 'b6cb5b39baf0ee9f3f376a13d7c7707e0c0160b8'
config = {}
config['session'] = True
config['api_key'] = _MY_SECRET_API_KEY
client = TiingoClient(config)

class Analysis:
    def __init__(self, ticker: str, start_date, end_date):
        self.ticker = ticker
        self.start_date = start_date
        self.end_date = end_date
        self.df = None
        print("init called")

    def excel_sheet(self):
        chart_data = client.get_ticker_price(self.ticker, fmt='csv',␣
 ↪startDate=self.start_date, endDate=self.end_date, frequency='daily')

        filename = f'{self.ticker}.csv'
        with open(filename, 'w') as outfile:
            outfile.write(chart_data)
        print(f'{filename} was created')

    def read_from_csv(self):
        filename = f'{self.ticker}.csv'
        self.df = (pd.read_csv(filename)
                       .drop(columns=['high', 'low', 'open', 'volume',␣
 ↪'adjHigh', 'adjLow', 'adjOpen', 'adjVolume', 'divCash', 'splitFactor']))
        self.df['Close_lag'] = self.df.close.shift(periods=1)
        self.df['date'] = pd.to_datetime(self.df['date'])
        print(f'{filename} dataframe was created')
```

```
    def add_data(self):
        self.df['ret_daily'] = self.df.close / self.df.Close_lag
        self.df['ret_pct'] = (self.df['ret_daily'] - 1.0) * 100
        print('Analysis columns added to self.df')

    def ret_pct(self) -> pd.Series:
        return self.df['ret_pct']

    def mean(self) -> float:
        return self.ret_pct().mean()

    def standard_deviation(self) -> float:
        return self.ret_pct().std()

    def twenty_five(self) -> float:
        return self.ret_pct().quantile(.25)

    def fifty(self) -> float:
        return self.ret_pct().quantile(.5)

    def seventy_five(self) -> float:
        return self.ret_pct().quantile(.75)


stock = Analysis('AAPL', '2021-01-01', '2021-12-31')
stock.excel_sheet()
stock.read_from_csv()
stock.add_data()
display(stock.df.tail())

print(f'Metrics of Daily % for {stock.ticker}\n--------------')
print(f'mu \u03BC is {stock.mean():,.4f}')
print(f'sigma \u03C3 is {stock.standard_deviation():,.4f}')
print(f'25% is {stock.twenty_five():,.4f}')
print(f'50% is {stock.fifty():,.4f}')
print(f'75% is {stock.seventy_five():,.4f}')
print('\n', stock.df['ret_pct'].describe())

print('-----\nNo Difference\n-----')
```

```
init called
AAPL.csv was created
AAPL.csv dataframe was created
Analysis columns added to self.df
```

| | date | close | adjClose | Close_lag | ret_daily | ret_pct |
|---|---|---|---|---|---|---|
| 247 | 2021-12-27 | 180.33 | 180.100160 | 176.28 | 1.022975 | 2.297481 |
| 248 | 2021-12-28 | 179.29 | 179.061486 | 180.33 | 0.994233 | -0.576720 |

```
249 2021-12-29  179.38  179.151371     179.29   1.000502  0.050198
250 2021-12-30  178.20  177.972875     179.38   0.993422 -0.657821
251 2021-12-31  177.57  177.343678     178.20   0.996465 -0.353535

Metrics of Daily % for AAPL
--------------
mu   is 0.1385
sigma   is 1.5760
25% is -0.7518
50% is 0.1474
75% is 1.2455

 count    251.000000
mean       0.138493
std        1.576012
min       -4.167353
25%       -0.751789
50%        0.147394
75%        1.245491
max        5.385123
Name: ret_pct, dtype: float64
-----
No Difference
-----
```

[ ]: