AAPL Stock
Data Analysis

Mario Escobar

November 9, 2022

Contents

Introduction

Apple Inc., founded 46 years ago (1976) in Los Altos, California, U.S., became the first publicly traded U.S. company to be **valued at over $1 trillion** in August 2018, then $2 trillion in August 2020, and most recently **$3 trillion in January 2022.** In 1977 the company's second computer, the Apple II, became a best seller and one of the first mass-produced microcomputers. Apple went public in 1980 to instant financial success.

In this report we will be looking at the historical data for AAPL, some financial indicators and compare it to other companies.

Chapter 1

The data

1.1 Importing libraries

We import Python libraries which are a set of useful functions that eliminate the need for writing codes from scratch.

```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

import yfinance as yf
from pandas_datareader.data import DataReader
from datetime import datetime
from datetime import date
```

A function that will help us read big numbers easily.

```python
import math

millnames = ['',' Thousand',' Million',' Billion',' Trillion']

def millify(n):
    n = float(n)
    millidx = max(0,min(len(millnames)-1,
                        int(math.floor(0 if n == 0 else math.log10(abs(n))/3))))

    return '{:.1f}{}'.format(n / 10**(3 * millidx), millnames[millidx])
```

Then we will add some display settings to our work.

```
pd.options.display.precision = 3
plt.style.use('seaborn-v0_8')
plt.rcParams['lines.color'] = 'teal'
```

1.2 Downloading the Data

YFinance allows us to gather stock data from Yahoo Finance.

```
symbols = ['AAPL']

end = datetime.now()
start = datetime(end.year - 10, end.month, end.day)

for stock in symbols:
    globals()[stock] = yf.download(stock, start, end)
```

'Adj Close' and 'Close' have the same values, so we'll take one out.

```
AAPL.drop(['Adj Close'],axis=1, inplace= True)
```

We will add another column called 'Total Traded'.To calculate a day's dollar volume we will multiply the price of the stock by the volume of that same day.

```
AAPL['Total Traded'] = (AAPL.Open * AAPL.Volume).astype(int)
```

1.3 Results

We use **Pandas** to begin our analysis. The function **info()** returns information about the dataframe such as memory usage, data types and number of non-null values in the dataframe.

```
AAPL.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 2518 entries, 2012-11-07 to 2022-11-07
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Open          2518 non-null   float64
 1   High          2518 non-null   float64
 2   Low           2518 non-null   float64
 3   Close         2518 non-null   float64
 4   Volume        2518 non-null   int64
 5   Total Traded  2518 non-null   int64
dtypes: float64(4), int64(2)
memory usage: 137.7 KB
```

We can see that we have 2518 entries in 6 columns and none of them are null.

describe() gives us descriptive statistics that summarize the central tendency, dispersion and shape of the dataset's distribution.

```
AAPL.describe()
```

```
            Open       High        Low      Close       Volume  Total Traded
count   2518.000   2518.000   2518.000   2518.000    2.518e+03     2.518e+03
mean      60.583     61.265     59.924     60.619    1.794e+08     7.996e+09
std       47.957     48.585     47.349     47.991    1.393e+08     5.117e+09
min       13.856     14.271     13.754     13.948    4.100e+07     1.275e+09
25%       26.487     26.708     26.263     26.478    9.391e+07     4.288e+09
50%       39.826     40.178     39.503     39.899    1.332e+08     6.313e+09
75%       79.776     80.649     79.129     79.793    2.121e+08     1.047e+10
max      182.630    182.940    179.120    182.010    1.461e+09     4.452e+10
```

Some values are too big to read easily, so we will use our function from earlier.With **pd.option_context()** we can change the data type from number to a string (text) for this instance only

```
with pd.option_context('float_format', '{:.f}'.format): print(AAPL['Volume'].
 ↪describe().apply(millify),AAPL['Total Traded'].describe().apply(millify))
```

```
count      2.5 Thousand
mean     179.4 Million
std      139.3 Million
min       41.0 Million
25%       93.9 Million
50%      133.2 Million
75%      212.1 Million
max        1.5 Billion
Name: Volume, dtype: object
count     2.5 Thousand
mean      8.0 Billion
std       5.1 Billion
min       1.3 Billion
25%       4.3 Billion
50%       6.3 Billion
75%      10.5 Billion
max      44.5 Billion
Name: Total Traded, dtype: object
```

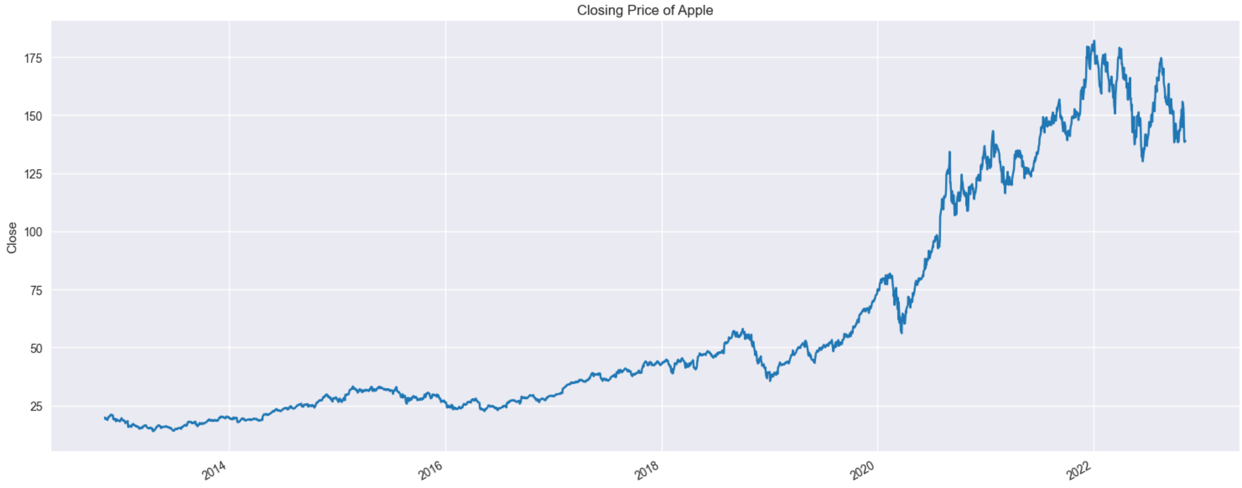
Chapter 2

Findings

2.1 Closing price & Volume

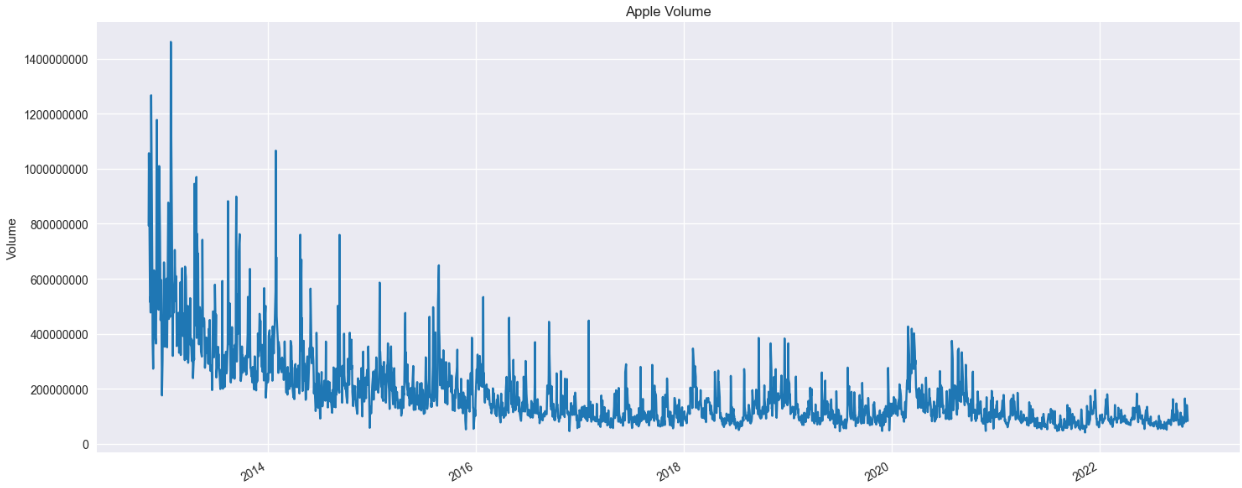
Let’s view of the closing price for 10 years:

```
plt.figure(figsize=(15, 6))

AAPL['Close'].plot(colormap= "tab10")
plt.ylabel('Close')
plt.xlabel(None)
plt.title("Closing Price of Apple")

plt.tight_layout()
```



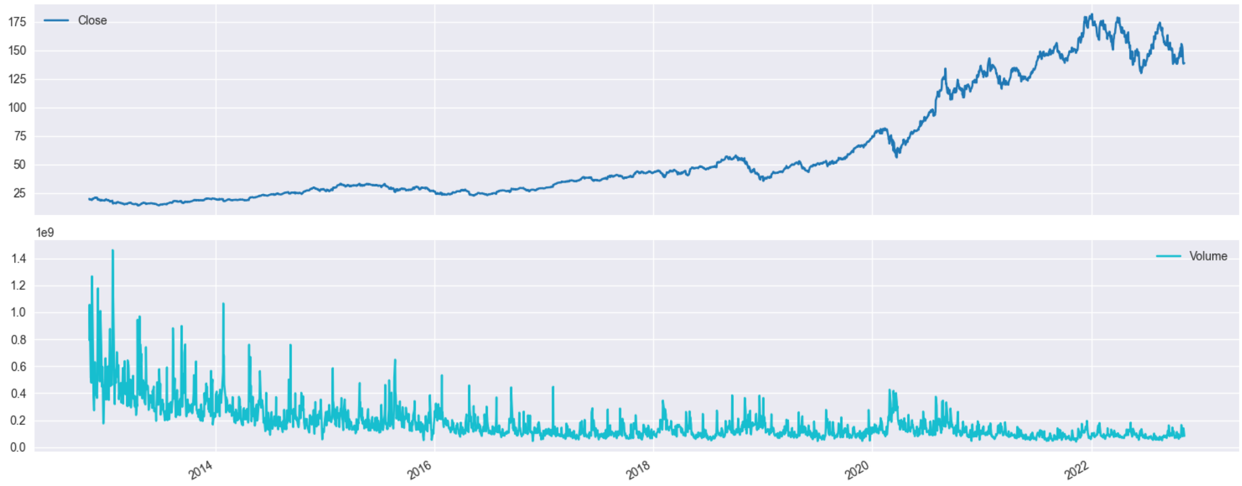
Then, the **volume**. for that same period. Ranging from 41 million to 1.5 billion.

```
plt.figure(figsize=(15, 6))

AAPL['Volume'].plot(colormap= "tab10")
plt.ylabel('Volume')
plt.xlabel(None)
plt.title("Apple Volume")
plt.ticklabel_format(style='plain',axis='y',useLocale='English')

plt.tight_layout()
```



Now let’s compare them side by side.

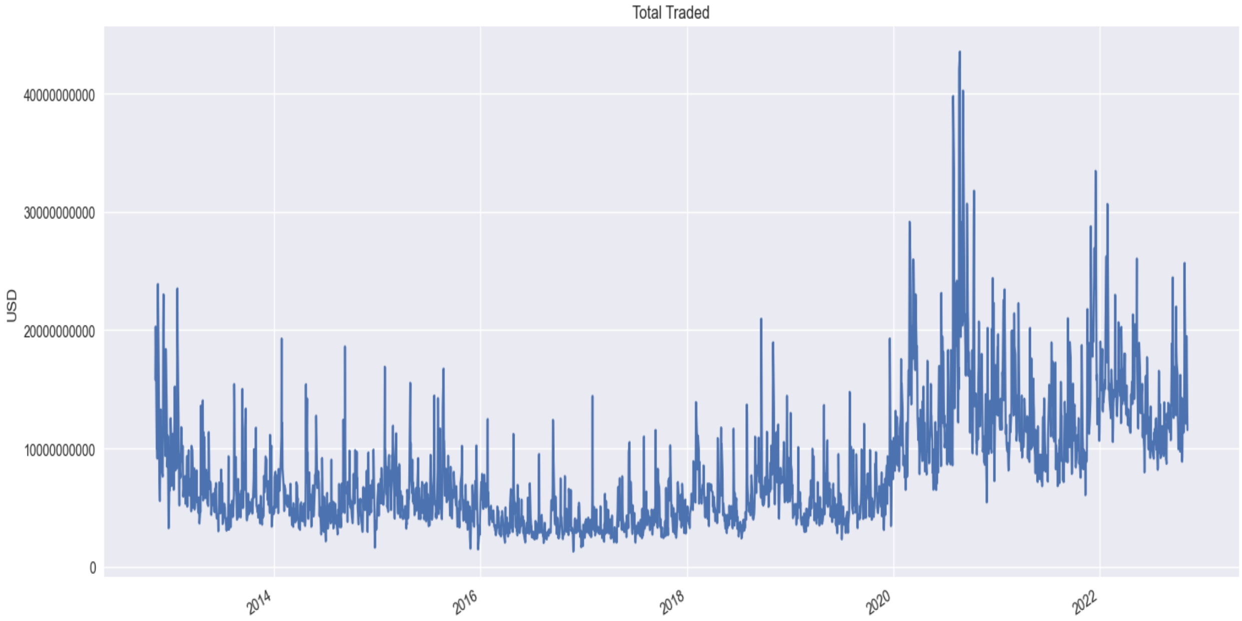
```
AAPL[['Close','Volume']].plot(subplots=True,colormap= "tab10",figsize=(15, 6))
plt.xlabel(None)
plt.tight_layout()
```



2.2 Total Traded

Ranges from 1.3 Billion to 44.5 Billion USD

```
plt.figure(figsize=(15, 6))

AAPL['Total Traded'].plot()
plt.ylabel('USD')
plt.xlabel(None)
plt.title("Total Traded")
plt.ticklabel_format(style='plain',axis='y',useLocale='English')


plt.tight_layout()
```



When was the max traded total?

```
AAPL['Total Traded'].idxmax()
```

```
Timestamp('2020-08-24 00:00:00')
```

We can sum the total traded amounts by each year.

```
df = AAPL.groupby(pd.Grouper(freq='Y'))
```

```
df['Total Traded'].sum().apply(millify)
```

```
Date
2012-12-31     2.7 Trillion
2013-12-31     1.7 Trillion
2014-12-31     1.4 Trillion
2015-12-31     1.6 Trillion
2016-12-31     1.0 Trillion
2017-12-31     1.0 Trillion
2018-12-31     1.6 Trillion
2019-12-31     1.5 Trillion
2020-12-31     3.7 Trillion
2021-12-31     3.2 Trillion
2022-12-31     3.0 Trillion
Freq: A-DEC, Name: Total Traded, dtype: object
```

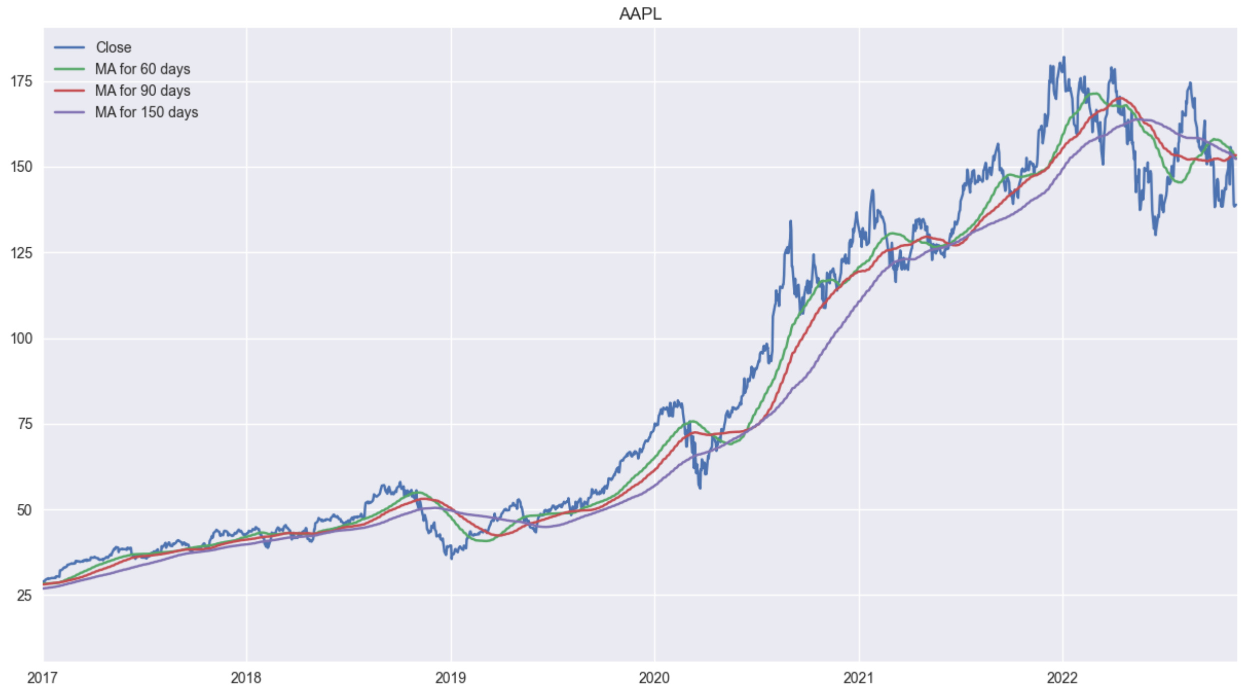
2.3 Moving average

We look at the moving average for the last 5 years in 3 different intervals. 60, 90 and 150 days.

```
ma_day = [60, 90, 150]

for ma in ma_day:
        column_name = f"MA for {ma} days"
        AAPL[column_name] = AAPL['Close'].rolling(ma).mean()
```

The moving average is a stock indicator commonly used in technical analysis, used to help smooth out price data by creating a constantly updated average price.

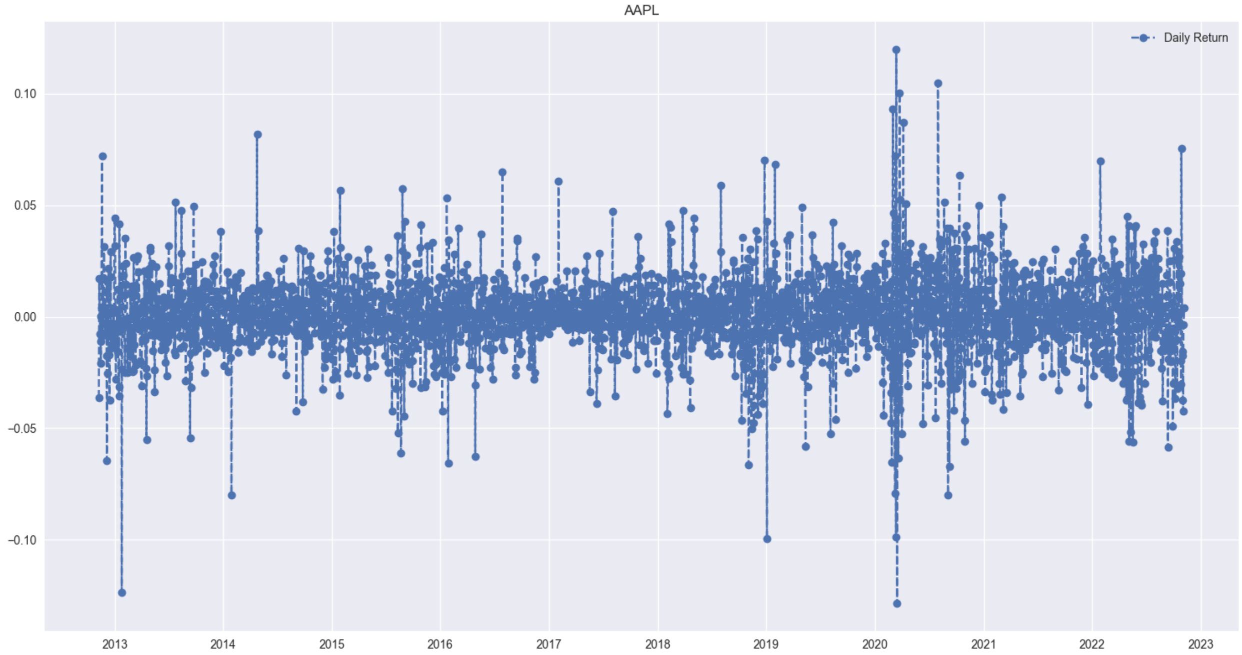
```
columns= ['Close', 'MA for 60 days', 'MA for 90 days', 'MA for 150 days']
fig, ax = plt.subplots()

fig.set_figheight(8)
fig.set_figwidth(15)
ax.plot(AAPL[columns])
ax.set_title('AAPL')
plt.legend(columns)
plt.xlim([date(2017,1,1),date(2022,11,10)])
plt.show()
```



2.4 Daily return

We'll use **pct_change()** to find the percent change for each day. Then we'll plot the daily return percentage

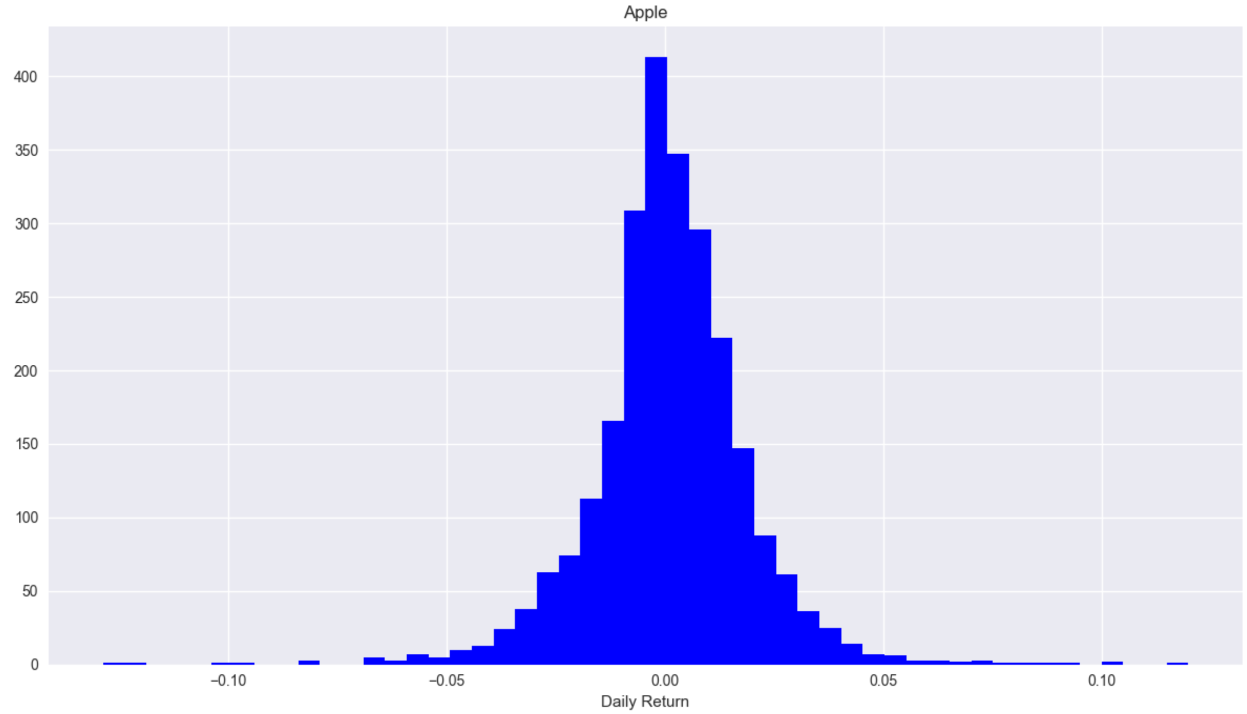
```
AAPL['Daily Return'] = AAPL['Close'].pct_change()

fig, axes = plt.subplots()

fig.set_figheight(8)
fig.set_figwidth(15)

plt.plot(AAPL['Daily Return'], linestyle='--', marker='o')
axes.set_title('AAPL')
plt.legend(['Daily Return'])

fig.tight_layout()
```



2.5 Daily return histogram

The next chart is a histogram that shows us the distribution of the daily returns.

```
plt.figure(figsize=(12, 7))

plt.subplot()

AAPL['Daily Return'].hist(bins=50,color='blue')
plt.xlabel('Daily Return')
plt.title('Apple')

plt.tight_layout()
```



2.6 Comparing the stock

We'll compare the last 5 years of data for Apple (AAPL), Microsoft (MSFT), Google(GOOGL), Amazon(AMZN) and Tesla(TSLA)

```
symbols = ['AAPL','MSFT','GOOGL','AMZN','TSLA']

df = DataReader(symbols, 'yahoo', datetime(end.year - 5, end.month, end.day), end)['Close']

df.head()
```

```
Symbols       AAPL   MSFT   GOOGL    AMZN    TSLA
Date
2017-11-07  43.702  84.27  52.619  56.159  20.403
2017-11-08  44.060  84.56  52.915  56.644  20.293
2017-11-09  43.970  84.09  52.386  56.457  20.199
2017-11-10  43.667  83.87  52.208  56.268  20.199
2017-11-13  43.493  83.93  52.060  56.458  21.027
```

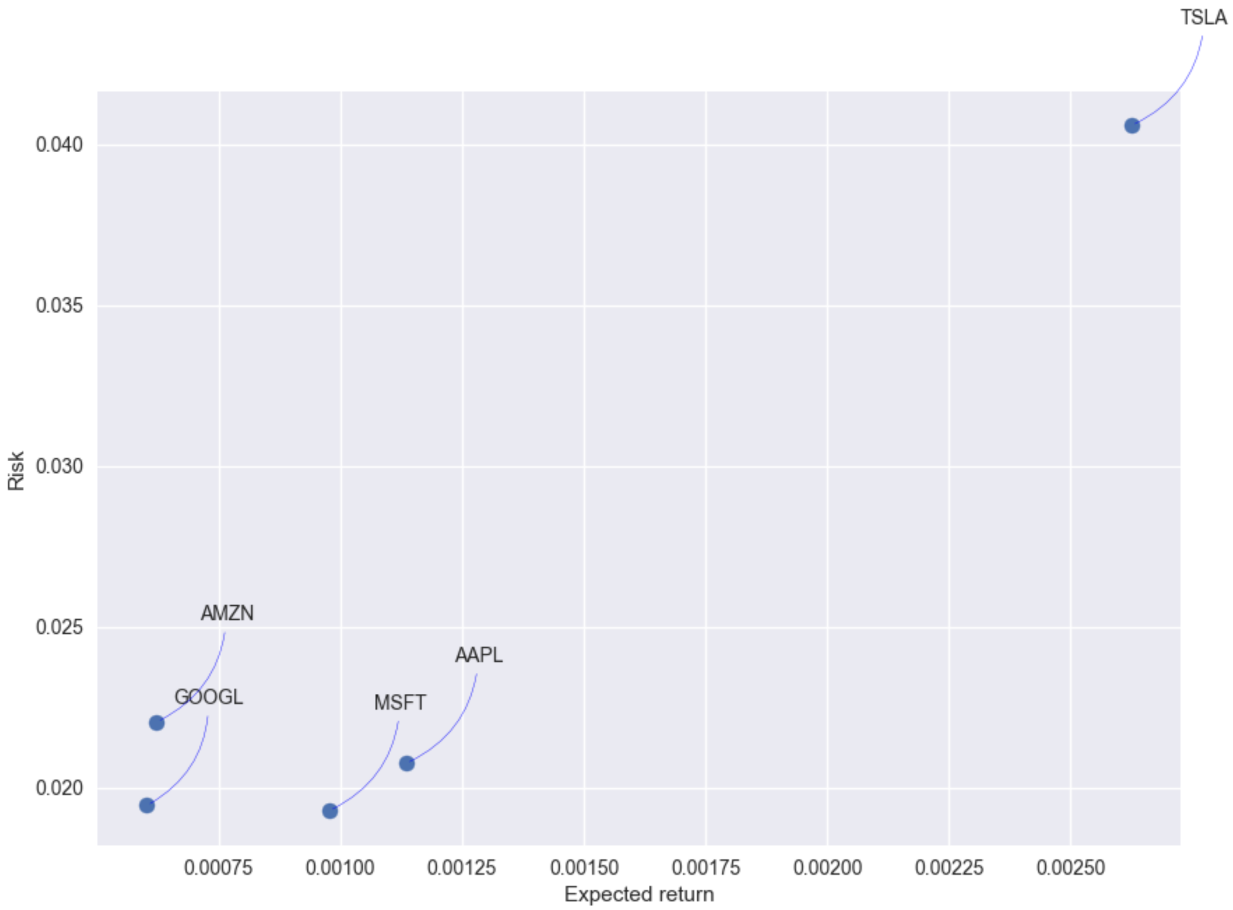
We will calculate the percentage change from the closing values at each day.

```
tech_rets = df.pct_change()
tech_rets.head()
```

```
Symbols      AAPL       MSFT  GOOGL   AMZN   TSLA
Date
2017-11-07    NaN        NaN    NaN    NaN    NaN
2017-11-08  0.008  0.003      0.006  0.009 -0.005
2017-11-09 -0.002 -0.005     -0.010 -0.003 -0.005
2017-11-10 -0.007 -0.002     -0.003 -0.003  0.000
2017-11-13 -0.004  0.000     -0.003  0.003  0.041
```

We can quantify risk by comparing the expected return against the standard deviation of the daily returns.

```
rets = tech_rets.dropna()

area = np.pi * 20

plt.figure(figsize=(10, 7))
plt.scatter(rets.mean(), rets.std(), s=area)
plt.xlabel('Expected return')
plt.ylabel('Risk')
for label, x, y in zip(rets.columns, rets.mean(), rets.std()):
    plt.annotate(label, xy=(x, y), xytext=(50, 50), textcoords='offset points',␣
 ↪ha='right', va='bottom',
                 arrowprops=dict(arrowstyle='-', color='blue',␣
 ↪connectionstyle='arc3,rad=-0.3'))
```



Conclusion

1. Over the last 10 years the Volume has decreased, but the value of the stock increased **up to 815%**
2. 2020 had the highest traded total at **3.7 trillion USD**

Apple has performed great over the last 10 years, even after COVID. But, will it also become the first US company to reach 4 Trillion USD market capitalization?

Made with: LaTeX