

Straight_Line_Stock_Historical_Data

September 29, 2021

1 Straight Line of Stock Historical Data

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

import warnings
warnings.filterwarnings("ignore")

# yfinance is used to fetch data
import yfinance as yf
yf.pdr_override()
```

```
[2]: # input
symbol = 'AMD'
start = '2017-01-01'
end = '2019-01-01'

# Read data
dataset = yf.download(symbol,start,end)['Adj Close']

# View Columns
dataset.head()
```

[*****100%*****] 1 of 1 completed

```
[2]: Date
2017-01-03    11.43
2017-01-04    11.43
2017-01-05    11.24
2017-01-06    11.32
2017-01-09    11.49
Name: Adj Close, dtype: float64
```

```
[3]: df = dataset.reset_index()
```

```
[4]: df.head()
```

```
[4]:      Date  Adj Close
0  2017-01-03      11.43
1  2017-01-04      11.43
2  2017-01-05      11.24
3  2017-01-06      11.32
4  2017-01-09      11.49
```

```
[5]: df.tail()
```

```
[5]:      Date  Adj Close
497 2018-12-24  16.650000
498 2018-12-26  17.900000
499 2018-12-27  17.490000
500 2018-12-28  17.820000
501 2018-12-31  18.459999
```

```
[6]: max_p = df['Adj Close'].max()
min_p = df['Adj Close'].min()
avg_p = df['Adj Close'].mean()
```

```
[16]: data = df.drop(['Date'], axis=1)
data
```

```
[16]:      Adj Close
0      11.430000
1      11.430000
2      11.240000
3      11.320000
4      11.490000
5      11.440000
6      11.200000
7      10.760000
8      10.580000
9       9.820000
10     9.880000
11     9.770000
12     9.750000
13     9.910000
14    10.440000
15    10.350000
16    10.520000
17    10.670000
18    10.610000
19    10.370000
20    12.060000
```

```
21    12.280000
22    12.240000
23    13.630000
24    13.290000
25    13.560000
26    13.420000
27    13.580000
28    13.490000
29    13.260000
..    ...
472   21.490000
473   20.660000
474   19.110001
475   19.209999
476   18.730000
477   19.379999
478   20.080000
479   21.049999
480   21.340000
481   21.430000
482   21.299999
483   23.709999
484   21.120001
485   21.299999
486   19.459999
487   19.990000
488   19.980000
489   20.480000
490   19.860001
491   19.900000
492   18.830000
493   19.500000
494   18.160000
495   17.940001
496   16.930000
497   16.650000
498   17.900000
499   17.490000
500   17.820000
501   18.459999
```

```
[502 rows x 1 columns]
```

```
[17]: data = data.reset_index()
```

```
[18]: data.as_matrix()
```

```
[18]: array([[ 0.          , 11.43000031],
           [ 1.          , 11.43000031],
           [ 2.          , 11.23999977],
           ...,
           [499.         , 17.48999977],
           [500.         , 17.81999969],
           [501.         , 18.45999908]])
```

```
[8]: from numpy import ones,vstack
     from numpy.linalg import lstsq
```

```
[19]: points = data.as_matrix()
```

```
[20]: x_coords, y_coords = zip(*points)
     A = vstack([x_coords,ones(len(x_coords))]).T
     m, c = lstsq(A, y_coords)[0]
```

```
[21]: print("Line Equation is y = {m}x + {c}".format(m=m,c=c))
```

Line Equation is y = 0.021718614923358824x + 9.372574584656498

```
[25]: equation_of_line = print("y = {m}x + {c}".format(m=m,c=c))
```

y = 0.021718614923358824x + 9.372574584656498

```
[28]: plt.figure(figsize=(16,8))
     plt.plot(dataset)
     plt.title('Line of Equation', equation_of_line)
     plt.xlabel('x', color='#1C2833')
     plt.ylabel('y', color='#1C2833')
     plt.legend(loc='best')
     plt.grid()
     plt.show()
```

