Stock_Statistics

September 29, 2021

1 Stock Statistics

Statistics is a branch of applied mathematics concerned with collecting, organizing, and interpreting data. Statistics is also the mathematical study of the likelihood and probability of events occurring based on known quantitative data or a collection of data.

http://www.icoachmath.com/math_dictionary/Statistics

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

import warnings
warnings.filterwarnings("ignore")

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

[2]:
```
# input
symbol = 'AAPL'
start = '2014-01-01'
end = '2019-01-01'

# Read data
df = yf.download(symbol,start,end)

# View Columns
df.head()
```

```
[**********************100%**********************]  1 of 1 downloaded
```

[2]:

Date	Open	High	Low	Close	Adj Close	Volume
2014-01-02	79.382858	79.575714	78.860001	79.018570	71.591667	58671200
2014-01-03	78.980003	79.099998	77.204285	77.282860	70.019096	98116900
2014-01-06	76.778572	78.114288	76.228569	77.704285	70.400902	103152700

```
2014-01-07  77.760002  77.994286  76.845711  77.148575  69.897423   79302300
2014-01-08  76.972855  77.937141  76.955711  77.637146  70.340096   64632400
```

[3]:
```
returns = df['Adj Close'].pct_change()[1:].dropna()
```

1.0.1 Mean is the average number, sum of the values divided by the number of values.

1.0.2 Median is the middle value in the list of numbers.

1.0.3 Mode is the value that occurs often.

[4]:
```
import statistics as st

print('Mean of returns:', st.mean(returns))
print('Median of returns:', st.median(returns))
print('Median Low of returns:', st.median_low(returns))
print('Median High of returns:', st.median_high(returns))
print('Median Grouped of returns:', st.median_grouped(returns))
print('Mode of returns:', st.mode(returns))
```

```
Mean of returns: 0.0007357373017012066
Median of returns: 0.0006264819982997327
Median Low of returns: 0.0006264819982997327
Median High of returns: 0.0006264819982997327
Median Grouped of returns: 0.0006264819982997327
Mode of returns: 0.0
```

[5]:
```
from statistics import mode

print('Mode of returns:', mode(returns))
# Since all of the returns are distinct, we use a frequency distribution to get␣
 ↪an alternative mode.
# np.histogram returns the frequency distribution over the bins as well as the␣
 ↪endpoints of the bins
hist, bins = np.histogram(returns, 20) # Break data up into 20 bins
maxfreq = max(hist)
# Find all of the bins that are hit with frequency maxfreq, then print the␣
 ↪intervals corresponding to them
print('Mode of bins:', [(bins[i], bins[i+1]) for i, j in enumerate(hist) if j␣
 ↪== maxfreq])
```

```
Mode of returns: 0.0
Mode of bins: [(-0.0070681808335254365, 0.0010272794824504605)]
```

1.0.4 Arithmetic Average Returns is average return on the the stock or investment

[6]:
```
print('Arithmetic average of returns:\n')
print(returns.mean())
```

```
Arithmetic average of returns:

0.0007357373017012073
```

1.0.5 Geometric mean is the average of a set of products, the calculation of which is commonly used to determine the performance results of an investment or portfolio. It is technically defined as "the nth root product of n numbers." The geometric mean must be used when working with percentages, which are derived from values, while the standard arithmetic mean works with the values themselves.

https://www.investopedia.com/terms/h/harmonicaverage.asp

[7]:
```
# Geometric mean
from scipy.stats.mstats import gmean
print('Geometric mean of stock:', gmean(returns))
```

```
Geometric mean of stock: nan
```

[8]:
```
ratios = returns + np.ones(len(returns))
R_G = gmean(ratios) - 1
print('Geometric mean of returns:', R_G)
```

```
Geometric mean of returns: 0.000622187293129
```

1.0.6 Standard deviation of returns is the risk of returns

[9]:
```
print('Standard deviation of returns')
print(returns.std())
```

```
Standard deviation of returns
0.01507109969428369
```

[10]:
```
T = len(returns)
init_price = df['Adj Close'][0]
final_price = df['Adj Close'][T]
print('Initial price:', init_price)
print('Final price:', final_price)
print('Final price as computed with R_G:', init_price*(1 + R_G)**T)
```

```
Initial price: 71.591667
Final price: 156.463837
Final price as computed with R_G: 156.463837
```

1.0.7 Harmonic Mean is numerical average.

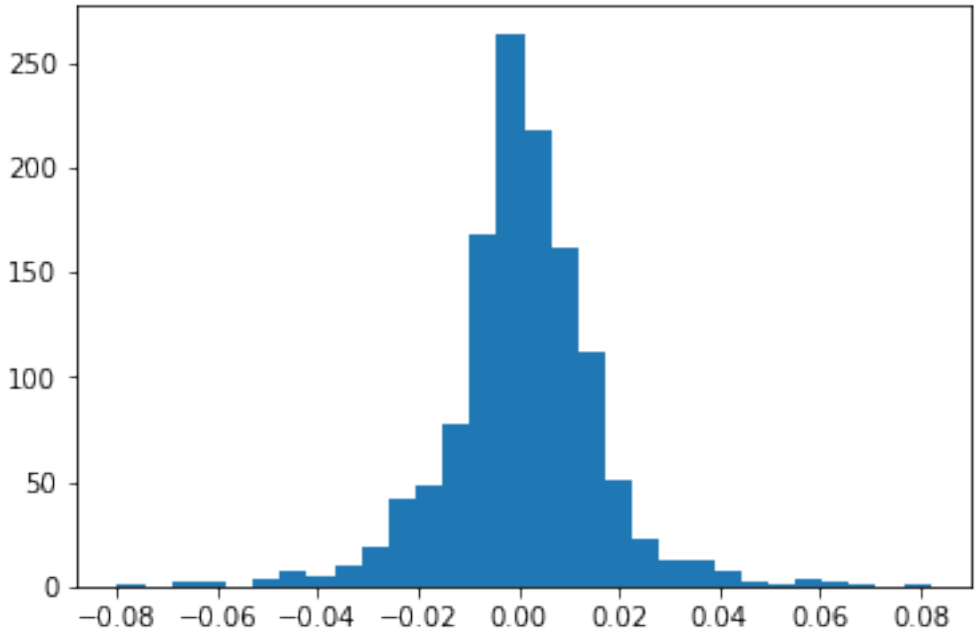
Formula: A set of n numbers, add the reciprocals of the numbers in the set, divide the sum by n, then take the reciprocal of the result.

[11]:
```
# Harmonic mean

print('Harmonic mean of returns:', len(returns)/np.sum(1.0/returns))
```

```
Harmonic mean of returns: 0.0
```

[12]:
```
print('Skew:', stats.skew(returns))
print('Mean:', np.mean(returns))
print('Median:', np.median(returns))

plt.hist(returns, 30);
```

```
Skew: -0.06538797604571234
Mean: 0.0007357373017012073
Median: 0.0006264819983
```

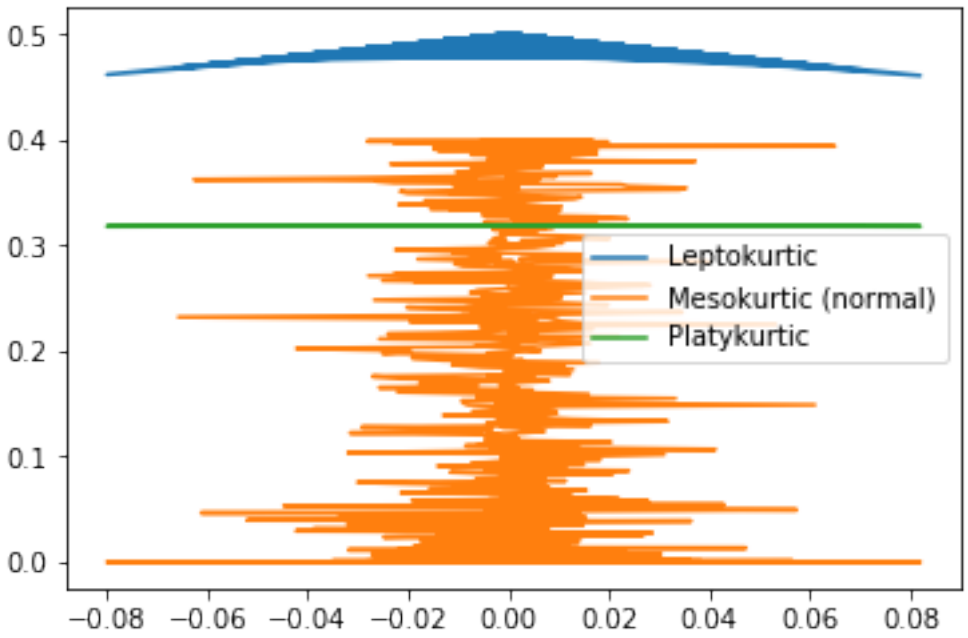


[13]:
```
# Plot some example distributions stock's returns
xs = np.linspace(-6,6, 1257)
normal = stats.norm.pdf(xs)
plt.plot(returns,stats.laplace.pdf(returns), label='Leptokurtic')
print('Excess kurtosis of leptokurtic distribution:', (stats.laplace.
 ↪stats(returns)))
```

```
plt.plot(returns, normal, label='Mesokurtic (normal)')
print('Excess kurtosis of mesokurtic distribution:', (stats.norm.
 ↪stats(returns)))
plt.plot(returns,stats.cosine.pdf(returns), label='Platykurtic')
print('Excess kurtosis of platykurtic distribution:', (stats.cosine.
 ↪stats(returns)))
plt.legend()
```

```
Excess kurtosis of leptokurtic distribution: (array([-0.02196584,  0.00545288,
-0.0071516 , …, -0.0064898 ,
        0.00051228,  0.00966536]), array([ 2.,  2.,  2., …,  2.,  2.,  2.]))
Excess kurtosis of mesokurtic distribution: (array([-0.02196584,  0.00545288,
-0.0071516 , …, -0.0064898 ,
        0.00051228,  0.00966536]), array([ 1.,  1.,  1., …,  1.,  1.,  1.]))
Excess kurtosis of platykurtic distribution: (array([-0.02196584,  0.00545288,
-0.0071516 , …, -0.0064898 ,
        0.00051228,  0.00966536]), array([ 1.28986813,  1.28986813,  1.28986813,
…,  1.28986813,
        1.28986813,  1.28986813]))
```

[13]: <matplotlib.legend.Legend at 0x22b05769fd0>



[14]:
```
print("Excess kurtosis of returns: ", stats.kurtosis(returns))
```

```
Excess kurtosis of returns:  3.73675394710252
```

[15]:
```
from statsmodels.stats.stattools import jarque_bera

_, pvalue, _, _ = jarque_bera(returns)

if pvalue > 0.05:
    print('The returns are likely normal.')
else:
    print('The returns are likely not normal.')
```

```
The returns are likely not normal.
```