

Portfolio_Functions

September 29, 2021

1 Portfolio Functions

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

```
import warnings
warnings.filterwarnings("ignore")

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

```
[2]: def get_historical_price(ticker, start_date, end_date):
    df = yf.download(ticker, start_date, end_date)['Adj Close']
    return df
```

```
[3]: symbols = ['FB', 'JNJ', 'LMT']
start = '2012-01-01'
end = '2019-01-01'
```

```
[4]: closes = get_historical_price(symbols, start, end)
```

```
[*****100%*****] 3 of 3 downloaded
```

```
[5]: closes[:5]
```

```
[5]:
```

	FB	JNJ	LMT
Date			
2012-01-03	NaN	52.659668	63.418526
2012-01-04	NaN	52.339931	62.560261
2012-01-05	NaN	52.275982	61.910774
2012-01-06	NaN	51.820370	61.841187
2012-01-09	NaN	51.900299	61.887573

```
[6]: def calc_daily_returns(closes):
      return np.log(closes/closes.shift(1))
```

```
[7]: daily_returns = calc_daily_returns(closes)
      daily_returns = daily_returns.dropna()
      daily_returns[:5]
```

```
[7]:
```

	FB	JNJ	LMT
Date			
2012-05-21	-0.116378	0.001893	0.010216
2012-05-22	-0.093255	0.000787	0.000717
2012-05-23	0.031749	-0.003943	-0.004071
2012-05-24	0.031680	0.006997	0.000240
2012-05-25	-0.034497	-0.009394	-0.007948

```
[8]: def calc_month_returns(daily_returns):
      monthly = np.exp(daily_returns.groupby(lambda date: date.month).sum())-1
      return monthly
```

```
[9]: month_returns = calc_month_returns(daily_returns)
      month_returns
```

```
[9]:
```

	FB	JNJ	LMT
1	0.667468	-0.035793	0.009406
2	-0.053291	0.169760	0.309037
3	-0.138119	0.159261	0.105750
4	0.210959	0.077376	-0.048124
5	-0.179184	0.024447	0.184214
6	0.200839	0.247562	0.018851
7	0.319302	0.241532	0.554246
8	-0.037284	-0.110904	0.106437
9	0.465323	0.043674	0.250755
10	0.041294	0.281684	0.005145
11	0.083944	0.070452	0.280294
12	0.005143	-0.130625	-0.145793

```
[10]: def calc_annual_returns(daily_returns):
       grouped = np.exp(daily_returns.groupby(lambda date: date.year).sum())-1
       return grouped
```

```
[11]: annual_returns = calc_annual_returns(daily_returns)
      annual_returns
```

```
[11]:
```

	FB	JNJ	LMT
2012	-0.303688	0.137447	0.154956
2013	1.052968	0.346244	0.681139
2014	0.427630	0.173373	0.347909

```
2015  0.341451  0.011422  0.161846
2016  0.099274  0.153292  0.183716
2017  0.533768  0.244253  0.317660
2018 -0.257112 -0.051315 -0.163459
```

```
[12]: def calc_portfolio_var(returns, weights=None):
      if (weights is None):
          weights = np.ones(returns.columns.size) / returns.columns.size
      sigma = np.cov(returns.T, ddof=0)
      var = (weights * sigma * weights.T).sum()
      return var
```

```
[13]: calc_portfolio_var(annual_returns)
```

```
[13]: 0.06497657266656308
```

```
[14]: def Sharpe_ratio(returns, weights = None, risk_free_rate = 0.001):
      n = returns.columns.size
      if (weights is None):
          weights = np.ones(n)/n
      var = calc_portfolio_var(returns, weights)
      means = returns.mean()
      sr = (means.dot(weights) - risk_free_rate)/np.sqrt(var)
      return sr
```

```
[15]: Sharpe_ratio(daily_returns)
```

```
[15]: -0.027644734305394318
```