real_estate_investment_trust_analysis_instructions

August 27, 2020

1 REIT STOCK ANALYSIS

In this project, you will analyze Real Estate Investment Trusts, commonly known as REITs. REITs are companies that own or operate real estate that produces income. REITs, like the stocks of regular public companies, are traded on different stock exchanges. Investing in a REIT allows you to invest in portfolios of real estate assets the same way you can invest in a company by buying its stock.

Using financial statistics and NumPy you will analyze two REITs: Sabra Health Care REIT Inc. (NASDAQ: SBRA), which invests in health care real estate, and Equity Residential (NASDAQ:EQR), which invests in rental apartment properties.

The time period for analysis we will be using is `Jan 1 2018` to `Dec 31 2018`. The REIT data for SBRA (`SBRA.csv`) and EQR (`EQR.csv`) can be found in the same folder as this file.

1. Import the numpy module as np

[2]:
```
import numpy as np
```

2. Load the adjusted closings for SBRA

[6]:
```
adj_closings_sbra = np.loadtxt("SBRA.csv", skiprows=1, usecols=5, delimiter=',')
print(adj_closings_sbra)
```

```
[15.36233  16.012386 16.528606 16.882313 17.427212 17.522808 17.513248
 17.427212 17.264698 17.417652 17.847836 18.105947 18.344938 18.306698
 18.497892 18.77512  19.09059  19.386936 19.377378 19.55901  19.635489
 19.425177 19.664167 19.539892 19.635489 19.80756  19.654606 19.60681
 19.358257 19.32958  19.339357 19.544678 19.388243 19.241585 19.222029
 19.143812 18.635395 18.381187 18.038986 17.716337 17.31547  17.247028
 17.256807 17.149258 17.119926 17.071039 17.383911 17.608788 17.941214
 17.970543 17.892326 17.726112 17.765223 17.980322 18.58651  18.293192
 18.449629 18.762501 18.870049 19.192698 19.036261 18.899382 18.879826
 18.694059 18.762501 18.958044 18.870049 18.801609 19.04604  19.241585
 19.241585 19.222029 18.821163 18.361633 18.713614 18.410519 18.713614
 18.909159 18.909159 19.153589 18.967821 19.124256 19.094927 19.143812
 19.554455 19.456682 19.143812 19.319801 18.840717 19.310024 19.241585
 19.417574 19.574011 19.75     19.709999 19.530001 19.73     19.700001
 19.709999 19.75     19.540001 19.219999 19.370001 19.290001 19.24
 19.1      19.32     19.450001 19.49     19.32     19.290001 19.32
 19.48     19.43     19.790001 19.58     20.139999 20.049999 19.57
 19.309999 19.26     18.870001 19.450001 19.690001]
```

3. Load the adjusted closings for EQR

```
[16]: adj_closings_eqr = np.loadtxt('EQR.csv', skiprows=1,usecols=5,delimiter=',')
      print(adj_closings_eqr)
```

```
[62.800674 63.19466  63.588642 64.120514 65.322166 65.322166 66.395767
 66.533653 66.61245  67.429955 68.040634 68.858147 69.064987 69.340775
 69.28167  69.291527 69.606705 70.640915 71.281136 70.916695 71.468269
 70.936401 71.694817 71.852402 71.901649 72.433533 72.335037 72.354729
 71.448578 72.187286 72.019852 72.10849  72.423676 71.586472 72.748718
 73.113152 72.394127 72.610817 72.226692 72.581276 72.394127 72.866913
 72.817657 72.443382 72.325188 72.532021 73.319992 73.920807 73.782921
 73.930656 73.635178 72.866913 72.394127 72.512329 73.664719 73.841385
 73.910858 74.575829 74.41703  75.062149 74.754478 74.41703  75.121704
 74.823952 74.714775 75.042297 74.992676 74.823952 75.776741 76.114189
 76.868484 77.017357 75.1614   73.732208 74.506355 73.613106 74.397179
 74.903351 75.399597 75.627869 74.734627 75.846214 75.945465 75.82637
 76.014938 75.637794 74.208603 73.881088 74.258232 75.260651 75.270569
 75.240791 75.61795  76.233284 76.044716 75.181252 75.786667 76.064568
 76.531036 76.610435 75.637794 75.101845 75.250725 75.995094 75.667564
 74.913277 76.669983 77.205933 77.394508 77.245636 77.215858 77.066986
 77.344879 77.672401 78.079323 77.463982 77.841125 78.238121 77.44413
 77.682327 76.332542 75.082001 75.610001 75.919998]
```

1.1 Simple Rate of Return Function

4. To calculate the daily rate of return for the SBRA stock, we need the daily adjusted closing price. The formula we are using for the daily rate of return is out[n] = a[n+1] - a[n]
5. Create a function that returns the daily rate of return

   step 1. define a function named `simple_rate_of_return`

   step 2. create parameter for `adj_closings`

   step 3. Within the function use np.diff() and set it to the variable `daily_simple_ror`

   step 4. return `daily_simple_ror`

```
[13]: def simple_rate_of_return(adj_closings):
          daily_simple_ror = np.diff(adj_closings)/adj_closings[:-1]
          return daily_simple_ror
```

1.2 Calculate Daily Rate of Return for SBRA

6. Call the function `simple_rate_of_return` with the arguments `adj_closings_sbra`. Then print the results.

[18]:
```
daily_simple_returns_sbra = simple_rate_of_return(adj_closings_sbra)
```

1.3 Calculate Daily Rate of Return for EQR

7. Call the function `simple_rate_of_return` with the arguments `adj_closings_eqr`. Then print the results.

[20]:
```
daily_simple_returns_eqr = simple_rate_of_return(adj_closings_eqr)
```

1.4 Calculate Average Daily Return for SBRA

8. Use `np.mean()` with the argument `daily_simple_returns_sbra` to calculate the average daily return for SBRA. Then set it to the variable name `average_daily_simple_return_sbra`

[23]:
```
average_daily_simple_return_sbra = np.mean(daily_simple_returns_sbra)
print(average_daily_simple_return_sbra)
```

```
0.00210937212011956
```

1.5 Calculate Average Daily Return for EQR

9. Use `np.mean()` with the argument `daily_simple_returns_eqr` to calculate the average daily return for EQR. Then set it to the variable name `average_daily_simple_return_eqr`

[24]:
```
average_daily_simple_return_eqr = np.mean(daily_simple_returns_eqr)
print(average_daily_simple_return_eqr)
```

```
0.0015777637451981398
```

1.6 Compare the Average Daily Return between EQR and SBRA

10. Based on the average daily simple returns of EQR and SBRA, which stock is more likely to be profitable in the future?

SBRA 0.0021 > EQR 0.0016, SBRA is more likely to be profitable in the future

1.7 Daily Log Returns Function

11. Create a function that returns the daily rate of return

    step 1. define a function named log_returns

    step 2. create parameter for `adj_closings`

    step 3. use np.log() to get the log of each adjusted closing price and set it to the variable `log_adj_closings`

    step 4. use np.diff() to get the diff of each daily log adjusted closing price and set it to the variable `daily_log_returns`

    step 5. return `daily_log_returns`

[26]:
```
def log_returns(adj_closings):
    log_adj_closings = np.log(adj_closings)
    daily_log_returns = np.diff(log_adj_closings)
    return daily_log_returns
```

1.8 Calculate Daily Log Returns for SBRA

12. Call the function `log_returns` with the arguments `adj_closings_sbra`. Set it to the variable `daily_log_returns_sbra`. Then print the results.

[28]:
```
daily_log_returns_sbra = log_returns(adj_closings_sbra)
print(daily_log_returns_sbra)
```

```
[ 0.04144414  0.03173003  0.02117393  0.03176639  0.00547045 -0.00054572
 -0.00492473 -0.00936905  0.00882034  0.02439809  0.01435818  0.01311323
 -0.00208667  0.01038978  0.01487581  0.01666295  0.01540389 -0.00049313
  0.00932975  0.00390254 -0.01076858  0.01222804 -0.00633993  0.00488047
  0.00872509 -0.00775197 -0.00243476 -0.01275791 -0.00148248  0.00050568
  0.01056078 -0.00803617 -0.00759303 -0.00101686 -0.00407743 -0.0269168
 -0.01373503 -0.01879239 -0.0180481  -0.02288689 -0.00396048  0.00056684
 -0.00625177 -0.00171186 -0.00285965  0.01816172  0.01285297  0.01870243
  0.00163339 -0.00436201 -0.0093331   0.00220398  0.01203516  0.03315811
 -0.01590708  0.00851529  0.01681599  0.00571571  0.01695394 -0.00818426
 -0.00721641 -0.00103528 -0.00988817  0.00365448  0.01036808 -0.00465237
 -0.0036335   0.01291675  0.01021462  0.         -0.00101686 -0.02107504
 -0.0247186   0.01898796 -0.01632909  0.01632909  0.01039513  0.
  0.01284371 -0.0097462   0.00821357 -0.00153478  0.00255683  0.02122361
 -0.00501258 -0.01621103  0.009151   -0.0251102   0.02460401 -0.00355052
  0.00910471  0.00802418  0.00895077 -0.00202742 -0.00917427  0.01018852
 -0.00152163  0.00050738  0.00202742 -0.01068979 -0.01651235  0.00777418
 -0.00413865 -0.00259543 -0.00730311  0.01145249  0.00670629  0.00205439
 -0.00876068 -0.00155395  0.00155395  0.00824747 -0.00257003  0.01835855
 -0.01066817  0.0281992  -0.00447873 -0.02423132 -0.01337474 -0.00259264
 -0.02045699  0.03027371  0.01226382]
```

1.9 Calculate Daily Log Returns for EQR

13. Call the function `log_returns` with the arguments `adj_closings_eqr`. Set it to the variable `daily_log_returns_eqr`. Then print the results.

[29]:

```
daily_log_returns_eqr = log_returns(adj_closings_eqr)
print(daily_log_returns_eqr)
```

```
[ 0.006254    0.00621507  0.00832947  0.01856708  0.          0.01630188
  0.00207458  0.00118362  0.01219786  0.00901573  0.01194346  0.00299935
  0.00398522 -0.00085275  0.00014226  0.00453827  0.01474861  0.00902221
 -0.00512584  0.00774768 -0.00746985  0.01063474  0.00219559  0.00068516
  0.00737016 -0.00136074  0.0002722  -0.01260281  0.01028593 -0.00232213
  0.00122999  0.00436147 -0.01162714  0.01610517  0.00499698 -0.00988309
  0.00298873 -0.00530423  0.00489731 -0.0025818   0.00650949 -0.0006762
 -0.00515315 -0.00163287  0.00285568  0.01080518  0.00816103 -0.00186706
  0.00200029 -0.0040047  -0.01048821 -0.00650949  0.00163143  0.01576737
  0.00239537  0.0009404   0.0089567  -0.00213163  0.00863161 -0.00410731
 -0.0045243   0.0094247  -0.00397147 -0.00146018  0.00437405 -0.00066146
 -0.00225241  0.01265335  0.0044433   0.00986126  0.00193485 -0.02439301
 -0.01919808  0.01044471 -0.01206134  0.01059494  0.0067806   0.0066033
  0.00302292 -0.01188132  0.01476426  0.00130773 -0.0015694   0.00248375
 -0.00497379 -0.01907599 -0.00442321  0.00509176  0.01340879  0.00013177
 -0.00039569  0.00500017  0.00810448 -0.00247663 -0.01141964  0.00802049
  0.00366018  0.0061138   0.00103694 -0.01277722 -0.00711095  0.00198041
  0.00984325 -0.0043192  -0.01001845  0.02317914  0.00696603  0.00243952
 -0.0019254  -0.00038557 -0.00192986  0.00359938  0.00422563  0.00522528
 -0.00791219  0.00485681  0.00508712 -0.01020024  0.00307101 -0.01752843
 -0.01651849  0.0070077   0.00409157]
```

1.10 Annualize Daily Log Return Function

14. Create a function that returns the daily rate of return

    step 1. define a function named `annualize_log_return`

    step 2. create parameter for `daily_log_returns`

    step 3. use `np.mean()` with the argument `daily_log_returns` to calculate the average daily return. Then set it to the variable name `average_daily_log_returns`

    step 4. then multiply `average_daily_log_returns` by 250 and set it to the variable `annualized_log_return`

    step 5. return `annualized_log_return`

[30]:

```
def annualize_log_return(daily_log_returns):
    average_daily_log_returns = np.mean(daily_log_returns)
    annualized_log_return = average_daily_log_returns * 250
    return annualized_log_return
```

1.11 Calculate Annualize Daily Log Return for SBRA

10. Call the function `annualize_log_return` with the arguments `daily_log_returns_sbra`. Set it to the variable `annualized_log_return_sbra`. Then print the results.

[32]:

```
annualized_log_return_sbra = annualize_log_return(daily_log_returns_sbra)
print(annualized_log_return_sbra)
```

```
0.5044563709645333
```

1.12 Calculate Annualize Daily Log Return for EQR

11. Call the function `annualize_log_return` with the arguments `daily_log_returns_eqr`. Set it to the variable `annualized_log_return_eqr`. Then print the results.

[33]:

```
annualized_log_return_eqr = annualize_log_return(daily_log_returns_eqr)
print(annualized_log_return_eqr)
```

```
0.3855982155640554
```

1.13 Compare the Annualize Daily Log Return between EQR and SBRA

12. Based on the differences between the Annualize Daily Log Return for EQR and SBRA, Which could be more profitable in the future and why?

SBRA 0.504 > EQR 0.385, SBRA would be more profitable in the future

1.14 Calculate Variance of Daily Log Return for SBRA

13. Calculate the variance of the daily logarithmetic return for SBRA. Use the function `.var()` with the argument `log_daily_ror`. Set it to the variable `daily_varaince_sbra`. Then print the results.

[35]:

```
daily_varaince_sbra = np.var(daily_log_returns_sbra)
print(daily_varaince_sbra)
```

```
0.00017844226355047074
```

1.15 Calculate Variance of Daily Log Return for EQR

14. Calculate the variance of the daily logarithmetic return for EQR. Use the function `.var()` with the argument `daily_log_returns_eqr`. Set it to the variable `daily_varaince_eqr`. Then print the results.

[37]:

```
daily_varaince_eqr = np.var(daily_log_returns_eqr)
print(daily_varaince_eqr)
```

```
6.833881310511606e-05
```

1.16 Compare the Variance of Daily Log Return between EQR and SBRA

15. Explain which investment is more riskier based on the Variance of daily log return between EQR and SBRA ?

SBRA 0.000178 > EQR 6.833e-05, SBRA is more riskier to invest

1.17 Calculate the Daily Standard Deviation for SBRA

16. Calculate the Standard Deviation of the daily logarithmetic return for SBRA. Use the function `.std()` with the argument `daily_log_returns_sbra`. Set it to the variable `daily_sd_sbra`. Then print the results.

[38]:

```
daily_sd_sbra = np.std(daily_log_returns_sbra)
print(daily_sd_sbra)
```

```
0.013358228308816658
```

1.18 Calculate the Daily Standard Deviation for EQR

17. Calculate the Standard Deviation of the daily logarithmetic return for EQR. Use the function `.std()` with the argument `daily_log_returns_eqr`. Set it to the variable `daily_sd_eqr`. Then print the results.

[39]:

```
daily_sd_eqr = np.std(daily_log_returns_eqr)
print(daily_sd_eqr)
```

```
0.008266729287034635
```

1.19 Compare the Daily Standard Deviation between EQR and SBRA

18. Has your previous variance risk assessment changed based on the Daily Standard Deviation and why?

SBRA std 0.0133 > EQR std 0.0083, previous variance risk assement didn't change, because standard deviation is the square root of the variance.

1.20 Calculate the Correlation between SBRA and EQR

19. Calculate the Correlation of the daily logarithmetic return between SBRA and ERQ assets. Use the function `.corrcoef()` with the arguments `daily_log_returns_sbra` and `daily_log_returns_eqr`. Set it to the variable `corr_sbra_eqr`. Then print the results.

[40]:

```
corr_sbra_eqr = np.corrcoef(daily_log_returns_sbra, daily_log_returns_eqr)
print(corr_sbra_eqr)
```

```
[[1.         0.62096591]
 [0.62096591 1.        ]]
```

1.21 Interpret the Correlation between SBRA and EQR

20. Interpret and explain the correlation between the stocks SBRA and EQR?

The assets have a positive correlation so I don't recommend investing in both at the same time. This is because if the price of one of them changes, it is most likely that the price of the other asset will also change in the same direction.

1.22 Final Analysis

21. Which stock would you invest in based on risk and profitability?

I would invest EQR. Although EQR might has less profit than SBRA in the future, the risk is lower too.

[ ]:

```

```