UDACITY

‹ Return to Classroom

Smart Beta Portfolio and Portfolio Optimization

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Dear student,

Great job!! 👏 👍 You have successfully implemented the `Smart Beta Portfolio and Portfolio Optimization Project` by implementing the functions provided in the Jupyter Notebook.
You have put a lot of effort in the project to successfully implement the `Smart Beta Portfolio and Portfolio Optimization Project`. Congratulations! 👌 💯
You passed all the tests when I ran the notebook! 💯
Keep up the good work and stay Udacious 😃

Thanks!

Part 1: Smart Beta Portfolio

The function `generate_dollar_volume_weights` computes dollar volume weights.

You have correctly calculated the dollar volume to generate dollar volume weights in the `generate_dollar_volume_weights` function. Normalization of values is done correctly for the sum of the total dollar-volume of the stocks traded every day 👍.

```
Tests Passed
```

Congratulations!

An alternative way to implement `generate_dollar_volume_weights` is as following:

```
return (close*volume).apply(lambda x: x/x.sum(),axis=1)
```

The function `calculate_dividend_weights` computes dividend weights.

You have correctly implemented the `calculate_dividend_weights` function by calculating the total dividend yield over time. 👌

```
Tests Passed
```

Congratulations!

An alternative way to implement `calculate_dividend_weights` is as following:

```
return dividends.cumsum().apply(lambda x:x/x.sum(),axis=1)
```

The function `generate_returns` computes returns.

You have correctly implemented the `generate_returns` function to generate returns data for all the stocks and dates from price data. You would have noticed that we're implementing returns and not log returns because we're not dealing with volatility, hence we don't have to use log returns. 👍

```
Tests Passed
```

Congratulations!

An alternative way to compute returns is as follows `return prices.pct_change()`

The function `generate_weighted_returns` computes weighted returns.

You have implemented the `generate_weighted_returns` function correctly to generate weighted returns using the simple multiplication of returns and weights. 👌

```
Tests Passed
```

Congratulations!

The function `calculate_cumulative_returns` computes cumulative returns.

You have implemented the `calculate_cumulative_returns` correctly to calculate the cumulative returns over time given the returns. We can use the cumulative returns to compare the performance between the ETF and index. 👏

The function `tracking_error` computes tracking error.

You have correctly implemented the `tracking_error` function to return the tracking error between the ETF and benchmark. Your `Smart Beta Portfolio` is complete now, we are now ready for `Portfolio optimization` from the next cell. 👍

```
Tests Passed
```

Congratulations!

Part 2: Portfolio Optimization

The function `get_covariance_returns` computes covariance of the returns.

You have correctly implemented the `get_covariance_returns` to calculate the covariance of the `returns` which can be used to calculate the portfolio variance. 👌

```
Tests Passed
```

Congratulations!

The function `get_optimal_weights` computes optimal weights.

You have correctly optimized the weights using `cvxpy` to implement the `get_optimal_weights` function. Variable, Objective and Constraints are set properly to solve the optimization problem with the help of

`cvx.Problem(objective, constraints).solve()`. You have also returned `x.value` correctly in the end. 👏

```
Tests Passed
```

Congratulations!

The function `rebalance_portfolio` computes weights for each rebalancing of the portfolio.

You have correctly rebalanced the portfolio over the same period instead of using the same weights for the entire history. You have rebalanced the portfolio every n number of days, using `shift_size`. When rebalancing, you also looked back at a certain number of days of data in the past, denoted as `chunk_size`. You have also computed the optimal weights using `get_optimal_weights` and `get_covariance_returns` functions. 👌.

```
Tests Passed
```

Congratulations!

The function `get_portfolio_turnover` computes cost of all the rebalancing.

You have now successfully implemented `Smart Beta Portfolio` which is evident by the impressive portfolio turnover value of `16.594080020340048` you have achieved from your portfolio. However, we need to be cognizant that this is a simulated market cap weighted index with large dollar volume stocks only which is not comparable one on one with the real world portfolios.

```
Tests Passed
```

Congratulations!

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START