UDACITY

Return to Classroom

DISCUSS ON STUDENT HUB

Smart Beta Portfolio and Portfolio Optimization

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Great job, you are ready to go! 💯 Clearly, you have acquired all the important concepts from this project. Wish you all the best for the upcoming projects! 🤞
Tip: If you are interested in finding the optimal weights by other techniques, such as **backtracking algorithm**, I strongly suggest that you can read this article.
Also, if you still find Pandas a little confusing, you might find this cheatsheet very useful.

Part 1: Smart Beta Portfolio

The function `generate_dollar_volume_weights` computes dollar volume weights.

Well done! You successfully compute the dollar volume weights.👏
Tip: Here is another way to do the job.

```
def generate_dollar_volume_weights(close, volume):
     dollar_volume = close * volume
     return (dollar_volume.T / dollar_volume.T.sum()).T
```

The function `calculate_dividend_weights` computes dividend weights.

Well done! You successfully compute the dollar volume weights.👏
Tip: Here is another way to do the job.

```
def calculate_dividend_weights(ex_dividend):
    dividend_cumsum_per_ticker = ex_dividend.cumsum().T
    return (dividend_cumsum_per_ticker/dividend_cumsum_per_ticker.sum()).T
```

The function `generate_returns` computes returns.

Fantastic, you correctly compute returns with `shift` function.💪
Tip: Here is another way to do the job.

```
def generate_returns(prices):
    return prices / prices.shift(1) - 1
```

The function `generate_weighted_returns` computes weighted returns.

Excellent, you correctly get the weighted returns by multiplying `returns` and `weights` 👍

The function `calculate_cumulative_returns` computes cumulative returns.

Good job!🎉 You successfully generate the cumulative returns with the `cumprod` function.

The function `tracking_error` computes tracking error.

Good, you successfully generate the tracking error with a correct annualized term `np.sqrt(252)`.🎉

Part 2: Portfolio Optimization

The function `get_covariance_returns` computes covariance of the returns.

Well done, you correctly calculate the covariance of the returns with `np.cov` function.🎉

The function `get_optimal_weights` computes optimal weights.

Fantastic, you correctly compute the optimal weights with `cvx.Minimize`.💪

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

Excellent, you correctly get the weights for each rebalancing of the portfolio👍

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

Good job!🎉 You successfully generated the cost of all the rebalancing.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START