UDACITY

Return to Classroom

Backtesting

REVIEW

CODE REVIEW

HISTORY

Requires Changes

1 specification requires changes

Great job, you are almost there! 💯Clearly, you have acquired all the important concepts from this project. You only need to make some modifications and then you are ready to go. Wish you all the best for the upcoming projects!👌

Tip: If you are interested in knowing more about backtesting, I would suggest that you could read this great article and take a look this python package.

Shift Daily Returns Data

The variable `frames` contains the data from `daily_return` and `data` correctly shifted.

Well done! You correctly prepare the required data `frames` with `merge` function💯

Build Universe Based on Filters

The function `get_universe` correctly creates a stock universe by selecting only those companies that have a market capitalization of at least 1 billion dollars (1e9) OR that are in the previous day's holdings, even if on the current day, the company no longer meets the 1 billion dollar criteria.

They should use the `.copy()` attribute to create a copy of the data and they should drop the column containing the daily return from the universe dataframe.

The returned `universe` dataframe should have a shape of (2265, 93)

Good job! You successfully get the right universe.👍

- `df['IssuerMarketCap'] >= 1e+9` => market capitalization of at least 1 billion dollars
- `abs(df['h.opt.previous']) > 0` => that are in the previous day's holdings

Factor covariance matrix

The function `diagonal_factor_cov` correctly creates the factor covariance matrix. The factor matrix must be scaled by (0.01**2). They must use the given `colnames` function to get the column names from `X` and use the statement 'covariance[date]' to get the covariances for the given `date`.

The returned factor covariance matrix should have shape (77, 77)

Perfect! You correctly create the factor covariance matrix and scale it with a correct weight `0.01**2` 🎉

Alpha Combination

The function `get_B_alpha` correctly creates a matrix of alpha factors. They must use the given `get_formula` and `model_matrix` functions.

The returned `B_alpha` should be of type patsy.design_info.DesignMatrix and it should have shape (2265, 4). The 4 columns of this matrix should correspond to the 4 alpha factors chosen at the beginning, namely:

"USFASTD_1DREVRSL"
"USFASTD_EARNYILD"
"USFASTD_VALUE"
"USFASTD_SENTMT"

Fantastic! You correctly create a matrix of alpha factors with `model_matrix` and `get_formula` functions👍

The function `get_alpha_vec` correctly creates a vector of alpha factors. To do this, they must add the rows of the Matrix of Alpha Factors and multiply the result by 1e-4.

The returned `alpha_vac` should have shape (2265,)

Good job! You correctly create a vector of alpha factors with the required scaling `1e-4`.🙌

Objective function

The `obj_func(h)` function correctly implements the objective function. The equation of the objective function is given in the notebook.

Great! You successfully implement the objective function.👍

Gradient

The `grad_func(h)` function correctly implements the gradient of the objective function. The equation of the gradient of the objective function is given in the notebook.

Well done, you successfully implement the gradient of the objective function with `np.matmul`.👏

Optimize

The function `get_h_star` correctly optimizes the objective function using the following functions `obj_func`, `grad_func`, and `scipy.optimize.fmin_l_bfgs_b`.

The returned `h_star` should have shape (2265,)

Well done! You correctly optimize the objective function with `scipy.optimize.fmin_l_bfgs_b`💯

Risk Exposures

The function `get_risk_exposures` correctly calculates the portfolio's risk exposures

The returned `risk_exposures` Pandas series should have shape (77,). The index of this Pandas Series should correspond to the risk factors such as 'USFASTD_AERODEF', 'USFASTD_AIRLINES', 'USFASTD_ALUMSTEL', ……

Perfect! You correctly calculate the portfolio's risk exposures with `np.matmul` function👍

The function `get_portfolio_alpha_exposure` correctly calculates the portfolio's alpha exposures.

The returned `portfolio_alpha_exposure` Pandas series should have shape (4,). The index of this Pandas Series should should correspond to the 4 alpha factors chosen at the beginning, namely:

"USFASTD_1DREVRSL"
"USFASTD_EARNYILD"

"USFASTD_VALUE"
"USFASTD_SENTMT"

Fantastic! You correctly calculate the portfolio's alpha exposures with the given `B_alpha` matrix .🎉

Transaction Costs

The function `get_total_transaction_costs` correctly calculates the total transaction costs according to the equation given in the notebook.

Perfect! You correctly calculate the total transaction costs according to the equation given in the notebook. 🎉

Profit-and-Loss (PnL) attribution

Correctly calculate the PnL attributed to the alpha factors, the PnL attributed to the risk factors, and attribution to cost.

To calculate the alpha and risk exposures they must use the provided `partial_dot_product` function

Good job! You correctly calculate all the related PnL.💯

```
df.at[dt,"attribution.alpha.pnl"] = partial_dot_product(fr, p["alpha.exposures"])
df.at[dt,"attribution.risk.pnl"] = partial_dot_product(fr, p["risk.exposures"])
df.at[dt,"attribution.cost"] = p["total.cost"]
```

Build portfolio characteristics

Correctly calculates the sum of long positions, short positions, net positions, gross market value, and amount of dollars traded.

❌ "traded": You should first use `h.opt.previous` and `h.opt` to calculate the difference between holdings and sum up the absolute value of the difference of holdings. You should not do thing like this

```
df.at[dt,"traded"] = tradelist['h.opt'].sum() - tradelist['h.opt.previous'].sum()
```

RESUBMIT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

RETURN TO PATH

Rate this review

START