Assignment 7: Estimating Neural Beta using Multilayer Perceptron

Overview (Weight: 10%)

Objective: Learn and Understand how Neural Networks can be used for calculation of Beta

The goals of this assignment are for the students to

- Become familiar with the scikit-learn and PyTorch frameworks for constructing Neural Networks.
- Practice calculating Neural Beta using multilayer perceptron (MLP) an important Neural Network (NN) tool.
- Learn how to perform hyper-parameter tuning for ML models.
- Please note that the most important part of the Assignment is not the programming. The primary learning should be to identify patterns in your analysis and draw meaningful conclusions from those patterns.

Required Files for Submission

- Jupyter Notebook file (source codes)
- Output in HTML/PDF format (source code in readable format)
- 5-page PDF with your analyses (your findings)
- Note: Please do **NOT** submit data

Starter Code

We provide starter code for you to work on. The name of code file is *AS7.ipynb*, available on Canvas.

Data

The required dataset for the assignment is available on Canvas:

- The name of data file is *MSF_1996_2023.csv*
- Description of all the variable is provided in *CRSP_MSF_Variable_Definition.pdf*
- Use monthly CRSP stock data for this assignment
- Use the value-weighted portfolio of all U.S. securities in the CRSP universe (VWRETD) from CRSP as the market portfolio (MKT)
- See the definitions of industry code provided at the end of the assignment
- You need to use data from 2005 to 2023
    - Training data: 2005-01 to 2012-12
    - Validation data: 2013-01 to 2017-12
    - Test data: 2018-01 to 2023-12
- You would need to download Fama-French factor data from the Fama-French data library.

Assignment

Neural Beta estimation

Step 0: To reduce work (computational effort), for each year, instead of analyzing the entire set of firms, select a sample of 10 firms per industry.

Multilayer perceptron (MLP) is a simple *feed-forward* networks. It consist of an *input layer* of raw predictors, one or more *hidden layers* that interact and nonlinearly transform the predictors, and an *output layer* that aggregates hidden layers into an ultimate outcome prediction.

The goal of the assignment is to construct a neural network using data from t-w to t period that can output neural Beta which minimizes the error for t+1:

$$\hat{\beta}_{t+1} = \min_{\beta} L\left(R_{i,t+1}, \beta \cdot MKT_{t+1}\right),$$

where L is a loss function (root mean square error–RMSE). Here $\hat{\beta}_{t+1}$ is assumed to be estimated from the dataset from period t-w to t. w is the look back window.

Step 1: Construct the NN model architecture to estimate $\hat{\beta}_{t+1}$ in the following steps:

- Use MSF data to construct features for input layer:
    - Using data from t-w to t in MSF data regarding price, returns and industry. Think of the features which would help estimate $\hat{\beta}_{t+1}$

- Add a dense hidden layer
- Add output layer which estimates $\hat{\beta}_{t+1}$ with 'linear' activation. Think about why 'linear' activation?
- Construct a custom loss function for the model based on equation above. You can calculate RMSE between $R_{i,t+1}$ and $\hat{\beta}_{t+1} \cdot MKT_{t+1}$, where $\hat{\beta}_{t+1}$ is output of NN.
- You need to think about decisions like what batch size to select, decide optimizer, decide momentum hyper parameters, etc.

Step 2: Perform the tuning for following hyper-parameter:

- Change number of neurons for hidden layer (consider 3 reasonable values)
- Change learning rate for the optimizer (consider 3 reasonable values)
- Change various activation functions for hidden layer ('linear', 'sigmoid', 'tanh', 'relu')
- Change look back window (w). Try 12, 24 and 36 months.

Step 3: Create a plot showing loss over the epochs for the best performing hyper-parameters.

Step 4: Compute the descriptive stats - N, mean, standard deviation, skewness, kurtosis along with the minimum value, maximum value, 1%, 5%, 25%, 50%, 75%, 95%, 99% percentiles by industry (use SIC code provide in supplementary details) for the beta values estimated above. Perform this for only the test period (2018-01 to 2023-12).

Step 5: Compute the mean and standard deviation of betas for each industry (using the SIC codes provide in the Supplementary Details) for each year and plot the betas over the test time period.

Step 6: In a few bullet points, briefly describe the findings from the beta computation and from the graphs.

Step 7: Use the Fama-French factor data to update the feature space:

- Add SMB, HML, RMW, CMA, Rf, etc to the input layer
- Use the value from period t-w to t only to avoid look-ahead bias
- Recalculate beta with the update model

Step 8: In a few bullet points, briefly describe the findings from the beta computation. How beta and RMSE differ from previous model?

Beta and Stock Returns

Step 9: Sort all stocks based on their beta into quintile portfolios.

- Form equal weighted portfolios and compute the average beta and the equal weighted $\frac{1}{N}$ portfolio excess return for each decile and the difference between portfolio 5 (high beta) and portfolio 1 (low beta)

- Repeat the previous steps for the value-weighted portfolio (weighted by the market capitalization) returns

Step 10: Compare your findings with findings from **Assignment 4: Beta Estimation** for the test time period.

Supplementary Details

Readings

- NeuralBeta from Bloomberg: Estimating Beta using Deep Learning (Link)
- NeuralFactors from Bloomberg: A Novel Factor Learning Approach to Generative Modeling of Equities (Link)
- ML Beta: Estimating Stock Market Betas via Machine Learning (Link)

Industry code:

SIC Code	Industries
1 – 999	Agriculture, Forestry and Fishing
1000 – 1499	Mining
1500 – 1799	Construction
2000 – 3999	Manufacturing
4000 – 4999	Transportation and other Utilities
5000 – 5199	Wholesale Trade
5200 – 5999	Retail Trade
6000 – 6799	Finance, Insurance and Real Estate
7000 – 8999	Services
9000 – 9999	Public Administration