Machine Learning for Financial Engineering

Final Project Instructions

Due Date: May 17, 2019.

Final project will be done individually and will require the application of machine learning methods to a large, real-world dataset. In this project, you will build and back-test a momentum trading strategy in the US equities market. The goal of this project is to motivate financial modeling and implement models you acquired throughout this course.

- The dataset provided comprises of stock market data for approximately 1500 stock instruments listed on the US stock exchange. It contains daily data from Feb 2008 to July 2017. There are 3,547,259 instances/samples in the dataset.

- First, implement a baseline strategy using logistic regression model and record the back-test results in terms of yearly return and yearly Sharpe ratio. To help you get started, we generated 33 input features (X), as well as the output/target label (Y). We provide a back-test framework programmed using python. Everyone will obtain the same back-test results for logistic regression baseline strategy. *If you have difficulty with this, the instructor will demonstrate how to obtain the baseline result in class on Wednesday, April 24.*

- Next, try to improve on the baseline result using a "classical ML" method (from first half of the course) and a "deep learning" method (from second half of the course). To obtain apples-to-apples comparison with the baseline strategy, use the same 33 input features, output target and back-test framework as the baseline.

Dataset feature attributes are:

- date
- id: stock ticker ID
- industry: sort each id by GICS industry group
- flag 2: is the label true every 20 trading days, use this as holding period in the back-test.
- ret_raw: daily return calculated from daily adjusted closing price
- ret_raw_norm: normalized ret_raw with regards to trading universe on that day
- ret_20_raw: cumulative sum of past 20 trading days return, i.e. monthly return
- ret_20_raw_norm: normalized ret_20_raw with regards to trading universe on that day

- ret_raw_norm_lag_21: ret_raw_norm of previous 1 day
- ret_raw_norm_lag_22: ret_raw_norm of previous 2 days
- ret_raw_norm_lag_23: ret_raw_norm of previous 3 days
- ret_raw_norm_lag_24: ret_raw_norm of previous 4 days
- ret_raw_norm_lag_25: ret_raw_norm of previous 5 days
- ret_raw_norm_lag_26: ret_raw_norm of previous 6 days
- ret_raw_norm_lag_27: ret_raw_norm of previous 7 days
- ret_raw_norm_lag_28: ret_raw_norm of previous 8 days
- ret_raw_norm_lag_29: ret_raw_norm of previous 9 days

- ret_raw_norm_lag_30: ret_raw_norm of previous 10 days
- ret_raw_norm_lag_31: ret_raw_norm of previous 11 days
- ret_raw_norm_lag_32: ret_raw_norm of previous 12 days
- ret_raw_norm_lag_33: ret_raw_norm of previous 13 days
- ret_raw_norm_lag_34: ret_raw_norm of previous 14 days
- ret_raw_norm_lag_35: ret_raw_norm of previous 15 days
- ret_raw_norm_lag_36: ret_raw_norm of previous 16 days
- ret_raw_norm_lag_37: ret_raw_norm of previous 17 days
- ret_raw_norm_lag_38: ret_raw_norm of previous 18 days
- ret_raw_norm_lag_39: ret_raw_norm of previous 19 days
- ret_raw_norm_lag_40: ret_raw_norm of previous 20 days

- ret_20_raw_norm_lag_41_60: ret_20_raw_norm of previous 2 months
- ret_20_raw_norm_lag_61_80: ret_20_raw_norm of previous 3 months
- ret_20_raw_norm_lag_81_100: ret_20_raw_norm of previous 4 months
- ret_20_raw_norm_lag_101_120: ret_20_raw_norm of previous 5 months
- ret_20_raw_norm_lag_121_140: ret_20_raw_norm of previous 6 months
- ret_20_raw_norm_lag_141_160: ret_20_raw_norm of previous 7 months
- ret_20_raw_norm_lag_161_180: ret_20_raw_norm of previous 8 months
- ret_20_raw_norm_lag_181_200: ret_20_raw_norm of previous 9 months
- ret_20_raw_norm_lag_201_220: ret_20_raw_norm of previous 10 months
- ret_20_raw_norm_lag_221_240: ret_20_raw_norm of previous 11 months
- ret_20_raw_norm_lag_241_260: ret_20_raw_norm of previous 12 months
- ret_20_raw_norm_lag_261_280: ret_20_raw_norm of previous 13 months
- isJan: indicator variable for the month of January to capture turn of the year pattern.

- target: label 1 if ret_20_raw_norm > ret_20_raw_norm median value of trading universe for that day, otherwise label as 0.

Model Input Features:

33 momentum driven features in data.iloc[: , 'ret_raw_norm_lag_21': 'isJan']. We replicated these features from *Takeuchi and lee (2003)*, and *Jagadeesh and titman (1993).* For every month *t*, there are 12 monthly returns for month *t-2* through *t-13*, and 20 daily returns approximately corresponding to month *t*. There is also an indicator variable if the holding period falls in January.

Model Output/Target Label:

For each stock ticker id, we label its output/target as 1 if the normalized monthly return over the subsequent month is above the median value, otherwise label as 0, see data.iloc[:, 'target']. Thus, the project is a binary classification task.

Back-test framework:

Alpha is the predicted probability of a stock belonging to target class 1 or 0. We then rank these predicted probabilities for all stocks in the trading universe. To construct portfolio, we long the top decile and short the bottom decile of the ranked probabilities. Once the portfolio is constructed, we hold our positions for 20 trading days (1-month).

Project Deliverables

Write a report to include all the relevant details of your work on the project. Suggested report length is 8 pages, but shorter or longer reports are also acceptable as long as they address the above problem. Also, please make sure your report includes all of the following:

a. **Methodology:** Describe your “classical ML” and “deep learning” methods in detail and defend your choice of methods.
b. **Training Procedure:** Describe techniques you used to optimize your “deep learning” model? What are your hyper-parameters?
c. **Results:** Present quantitative performance results (in-sample and out-of-sample). Why did your model perform better or worse than the baseline? Evaluate pros and cons associated with using “classical ML” vs “deep learning” models.
d. **Conclusion and Further Work**.
e. **Provide code** written for the project in Jupiter Notebook.

Relevant Papers:

- Takeuchi, L., & Lee, Y. Y. A. (2013). Applying deep learning to enhance momentum trading strategies in stocks. In *Technical Report*. Stanford University.

- Jegadeesh, N., & Titman, S. (1993). Returns to buying winners and selling losers: Implications for stock market efficiency. The Journal of finance, 48(1), 65-91.

****IMPORTANT**** Please note that the goal is not to produce the highest out-of-sample yearly return or Sharpe ratio, and if you did outperform the baseline by a lot you may have used forward-bias information, or something was done incorrectly.

We will rate each project on a 0-10 scale for each of the following criteria:

- Correctness: Are appropriate methods used to address the task? Are the results interpreted correctly and appropriate conclusions drawn?
- Novelty: Are ML techniques applied to this problem in a novel way?
- Clarity/Presentation: Is the work well-written, clear, and understandable?