The Hong Kong University of Science and Technology

Data-Driven Modeling

Multi-factor Stock Selection Strategy With Different Models

-Report-

Zhuoyue WAN

zwanah@connect.ust.hk

Contents

1 Introduction **3**

1.1 Background and Purpose . . . 3

1.2 Simply explain of strategy . . . 3

2 Data Description and Preprocessing **3**

2.1 Data collection . . . 3

2.2 Feature and Label Extraction . . . 3

2.3 Feature preprocessing . . . 4

2.4 Training set and cross-validation set combined . . . 5

3 Trading Strategy Based on Backtesting **5**

3.1 Long/Short Portfolio . . . 5

3.2 Staged BackTesting Method . . . 5

3.3 Procedures . . . 5

4 Data Mining Methods and Results **6**

4.1 In-sample training . . . 6

4.1.1 Logistic regression . . . 6

4.1.2 Randomforest . . . 6

4.1.3 SVM . . . 7

4.1.4 Adaboost . . . 7

4.1.5 Gradient Boosting . . . 7

4.1.6 Stacked LSTM . . . 7

4.2 Results Analysis in Out Sample Testing . . . 8

4.2.1 Evaluation indicators . . . 8

4.2.2 Results based on Multimodels combination . . . 8

4.2.3 Further Analysis on SVM . . . 8

4.2.4 Results based on Multimodels combination . . . 9

5 Discussion **10**

6 Conclusion **11**

Abstract

Using fewer explanatory variables to predict the rise and fall of stocks is a common way to study stock models. However, the generalization ability of such stock models is usually poor, and it is also confusing to only predict the rise and fall. We hope that the risk and other more specific factor could also be included. On this basis, we covered more than 70 factors in this project, and used Logistic Regression, Random Forest, SVM, LSTM and ensemble methods such as voting, boosting methods to establish a complete set of quantitative trading models based on hierarchical backtesting strategies, using Annual Return, Sharpe Ratio, Maximum drawdown, Win Rate and other indicators comprehensively judge the performance of the trading model. In single-model trading, Random Forest performs best. In the multi-model combination, the transaction model based on Logistic Regression + Gradient Boost has the highest Annual Return and Win Rate, and the transaction model based on Logistic Regression + Gradient Boost + Random Forest has the highest Sharpe Ratio and Maximum DropDown.

1 Introduction

1.1 Background and Purpose

We hope to build a complete stock trading strategy based on different models, and be able to judge the performance results of the models from different perspectives. The models used for training are basic machine learning models such as Logistic Regression, Random Forest, SVM and deep neural networks such as LSTM, and ensemble model methods. The selection of stock indicators refers to the random forest model of [Huatai Securities] Huatai Artificial Intelligence Series 5-Artificial Intelligence Stock Selection.the problem.

1.2 Simply explain of strategy

The specific strategy is: stocks predicted to rise are labeled 1, and stocks predicted to fall are labeled 0. During a specific period of time, the top 30% of the stocks in the training set are labeled as rising, and the bottom 30% of stocks in the training set are labeled as falling. The data of 2018-2019 is training + test set, the data of 2020 is back-test data; the data of 2019-2020 is training set + test set, the data of 2021 is back-test data; the data of 2020-2021 is Training set + test set, the data in 2022 is backtest data. The models used in the machine learning part are logistic regression and random forest. The model training results of each part are used on the back-test data to obtain the predicted value (probability of prediction of 1) of each stock for each day. The detailed strategy will be explained in part 3. The main process is shown in the figure below.

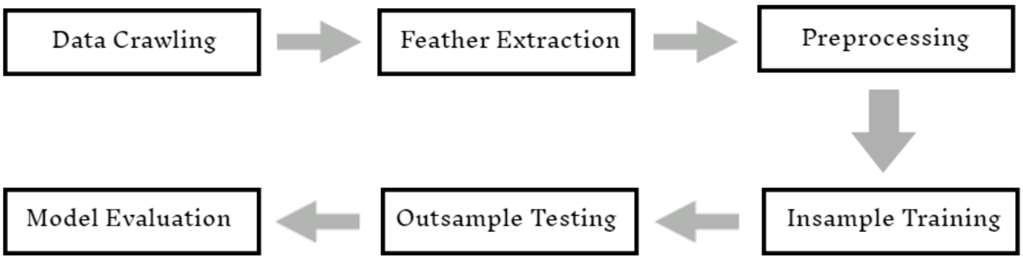

Figure 1.1: Schematic diagram of transaction model construction

2 Data Description and Preprocessing

2.1 Data collection

Stock pool: All A shares, excluding ST stocks, excluding stocks suspended from trading on the next trading day of each cross-section period, excluding stocks listed within 3 months. Each stock is considered a sample.

Backtest interval: 2020-01-01 to 2022-04-30.

2.2 Feature and Label Extraction

On the last trading day of each natural month, calculate the exposure of 45 factors (**exposure is just the value of variable**) in the previous report as the original characteristics of the sample; calculate the excess return of individual stocks (based on the CSI 300 Index) for the next whole natural month as the label of the sample . The factor pool is shown in Figure 1. We choose different types of factors, hoping to make a coverage on all the directions affecting the stock prices.

Factor Type	**Variable**
Valuation Factor	CON_EPS_FY12
Valuation Factor	CON_INCOME_FY12_CHGPCT1Y
Valuation Factor	CON_PEG2_FY12
Valuation Factor	CON_PE_FY12
Valuation Factor	CON_PROFIT_CGR2Y
Valuation Factor	CON_PROFIT_CHGPCT_1H
Valuation Factor	CON_PROFIT_CHGPCT_1Q
Valuation Factor	CON_PROFIT_FY12_CHGPCT1Y
Valuation Factor	afterAnnoRet
Tech Factor	LRreversal
Tech Factor	MAWVAD
Tech Factor	MaxRet
Tech Factor	SELL_VOL
Tech Factor	TOBT
Tech Factor	TVSTD6
Tech Factor	VOL20
Tech Factor	VSTD10
Tech Factor	candel_up_std
Sentimental Factor	sentimentScoreR45Core
Sentimental Factor	sentimentScoreR45_x
Sentimental Factor	snetimentScoreR45_y
Risk Factor	CpmpleUpward
Risk Factor	Ffactor
Risk Factor	HSIGMA
Risk Factor	MOM_6M
Risk Factor	MOM_9M
Risk Factor	S_phi_wts
Risk Factor	Volatility
Risk Factor	skew_12
Leverage Factor	APR
Leverage Factor	ARBP
Leverage Factor	CFOA
Leverage Factor	TotalAssetsRate
Growth Factor	CON_PROFIT_YOY
Growth Factor	CON_ROE
Growth Factor	GB25Mean
Growth Factor	GROSS_PROFIT_QOQ
Growth Factor	NI_QOQ
Growth Factor	N_CF_OPA_TR
Growth Factor	TL_YTD
Growth Factor	T_REVENUE_QOQ
Financial Factor	ALL_ORG_COVER_1M
Financial Factor	GROSS_PROFIT

2.3 Feature preprocessing

(a) Determination of the median: Let the exposure sequence of a factor on all stocks in the T period be D_i,D_M is the median of the sequence, and D_{M1} is the median of the sequence $|D_i - D_M|$ set all numbers in the sequence D_i greater than $D_M + 5D_{M1}$ to the sequence's maximum value and all numbers in the sequence D_i less than $D_M - 5D_{M1}$ to the sequence's maximum value;

(b) Missing value treatment: After obtaining a new factor exposure sequence, set the missing factor exposure as the average value of the same stocks in the CITIC tier-one industry.

(c) Neutralization of industry market value: Perform a linear regression on the factor exposure after filling in the missing values (as explained variable) and the industry dummy variables, the log(market value) (as Explanatory variables), and take the residual as the new factor exposure.

(d) Normalization: Order the neutralized factor exposure sequence on the cross section (In our project, the length of cross section is one month.) and divide it by the number of stocks in the current coupon pool to obtain a sequence uniformly distributed on $(0, 1]$.

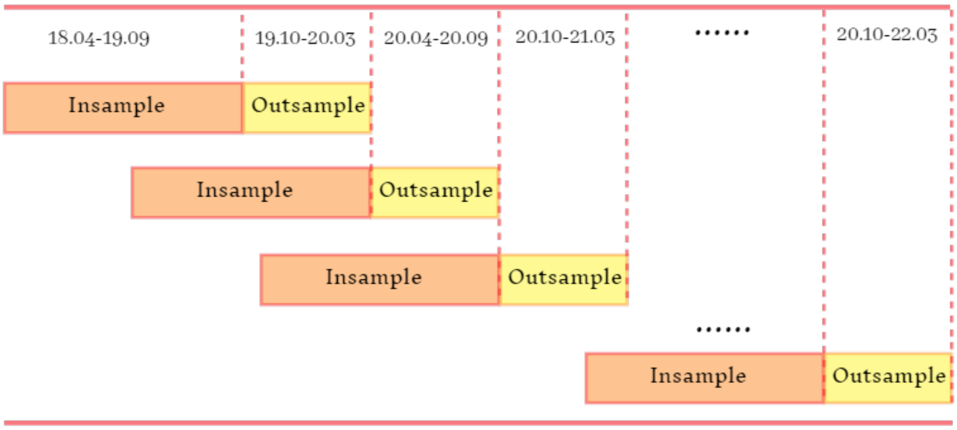

Figure 3.1: Insample training Schematic

2.4 Training set and cross-validation set combined

In the cross-section period at the end of each month, the top 30% stocks in the next month's return are selected as positive examples ($y = 1$), and the bottom 30% stocks are selected as negative examples ($y = 0$). All-A stock selection model: Combine the samples from the current year forward 24 months, randomly select 90% of the samples as the training set, and the remaining 10% of the samples as the cross-validation set.

3 Trading Strategy Based on Backtesting

3.1 Long/Short Portfolio

Each data in the out-sample is predicted based on our model. The input is a vector containing 45 factors, and the output is the probability of the stock boom and bust which is denoted as 'ret'. At the same time, we also have the weight ratio of different industries (i.e:Transportation, Entertainment Services, IT and so on). At the end of each month, each stock can be retrieved to the corresponding industry to which it belongs, denoted as weights-ins. Using the product of the above two parameters as the final score, we take the top 20% of the stocks for the long investment and the last 20% of the stocks for short investment which is our best portfolio.

3.2 Staged BackTesting Method

Stock data is time-series, and data that is outdated should not have much impact on the present. We hope to use a relatively concentrated cycle to train and perform performance testing of our model, and at the same time hope to be able to make full use of the data set . Therefore, we can neither simply shuffle the data set for training, nor can test split just use the data of the previous years to train the model and then use the data of the next few years as the test set. Instead, backtesting is a test method that can comprehensively evaluate the performance of a model over different time periods, while also preserving the effect of time on stock data.

3.3 Procedures

For each sample we divided into in-sample and out-sample according to the ratio of 3:1 (18 months and 6 months, respectively). The former is used for training and the latter is used for evaluation.

The construction of training sets

Filtering: At the end of each month, the top 30% stocks in the next month's return are selected as positive examples ($y = 1$), and the bottom 30% stocks are selected as negative examples ($y = 0$).

Grouping: After filtering, we get 49 months of economic data. Starting from the earliest date, every six months is a 'step', and continuously merge 24 months (data) as a group, and finally the data of the extra month is discarded. At this time, we get five sets of samples.(Shown in Figure1)

In-sample training

The training set is trained using different regression models. As we divide the back-test interval by half a year, it is divided into 5 sub-intervals. Hence, it is necessary to repeat the training with different models for different training

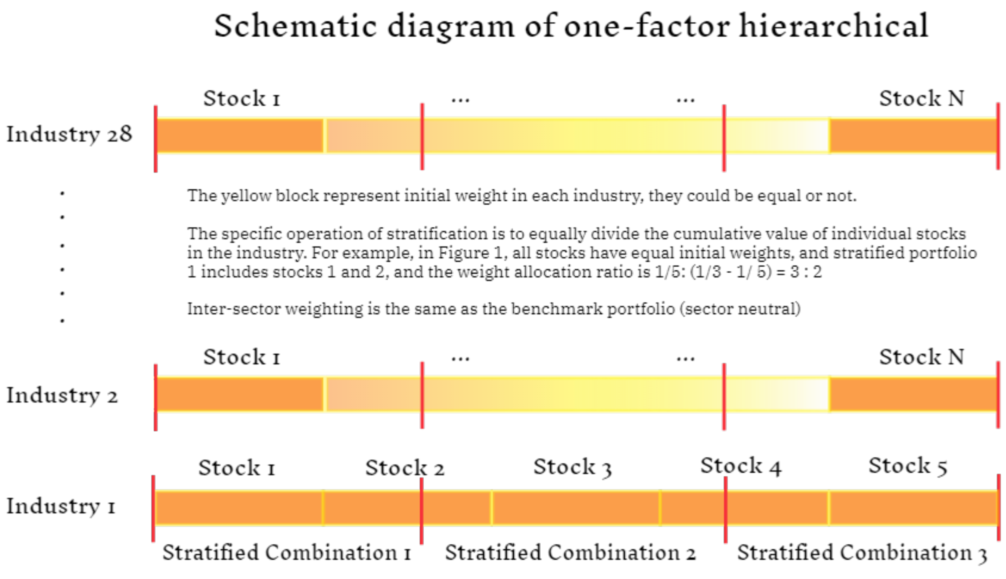
Schematic diagram of one-factor hierarchical
Industry 28
Stock 1
...
...
Stock N
The yellow block represent initial weight in each industry, they could be equal or not.
The specific operation of stratification is to equally divide the cumulative value of individual stocks in the industry. For example, in Figure 1, all stocks have equal initial weights, and stratified portfolio 1 includes stocks 1 and 2, and the weight allocation ratio is 1/5: (1/3 - 1/ 5) = 3 : 2
Inter-sector weighting is the same as the benchmark portfolio (sector neutral)
Industry 2
Stock 1
...
...
Stock N
Industry 1
Stock 1
Stock 2
Stock 3
Stock 4
Stock 5
Stratified Combination 1
Stratified Combination 2
Stratified Combination 3

Figure 3.2: One-factor Hierarchical Model Diagram

sets of each subinterval (see below for details). The predicted results belong to $[0, 1]$, and those with probability close to zero tend to fall whereas the others tend to surge.

Grid search parameter adjustment

Instead of applying K fold cross-validation, we utilize grid search to select the optimal hyper-parameters and then train for our dataset.

Out-sample evaluation

After determining the optimal parameters, use the pre-processed 45 factors of out-sample (stocks) as the input of the model, the predicted value of each out-sample shows every stock's tendency.(synthetic factor, for example, it is the voting average of the classification results of each decision tree in the random forest classifier). In the end, the portfolio can be constructed according to the predicted value.

4 Data Mining Methods and Results

4.1 In-sample training

4.1.1 Logistic regression

As the dependent variable we need to predict is a "buying signal" range from 0 to 1, the first model that comes into mind is logistic regression. LR is a simple model calculated by strictly defined mathematical formulas instead of complex algorithms, therefore earning a good reputation for computation and time efficiency. Easily acquisition of odds ratio is also an advantage for LR, making the model more interpretable. Nevertheless, LR is too simple to underfit the data we are trying to predict, resulting in low accuracy and low return in the stock market. We need to use more sophisticated models such as random forest, boosting and SVM.

4.1.2 Randomforest

One of the weaknesses of the Decision Tree Approach is that its structure might be altered significantly if there are a few samples changed, and its predictions become unstable. Random forest is a bagging method aiming to improve the robustness of decision trees. In Random Forest, each tree selects samples drawn with replacement from the training set, and uses a random subset of the features to construct the splits. We use scikit-learn to combine Tree classifiers' predictions by averaging their probabilities. Random forest might be very useful in our project, since we aim to predict stock price utilizing a large amount of features, which is a high-rank data. Besides, when dueling with stock price, we don't expect the model to be interpretable, since stock price itself is a complex joint effect of many factors. Accuracy is the only thing we care about. We use the grid search method to select the best parameters. The combination of parameters are shown as below:

Max depth	n estimators	max features
$[4, 8, 12]$	$[50, 100, 150]$	$[6, 8, 10]$

4.1.3 SVM

SVM is a special classifier with an elegant mathematical expression based on operational research theories. Kernel trick enables it to map the feature space into a higher one, and be versatile in different shapes of classifications. The scikit-learn package we use provides prediction probabilities that we can still draw ROC curves to compare SVM with other models. The biggest drawback of SVM we found in our real-case analysis is that it is computational and time consuming, when dueling with our big data. Compared with Random Forest, for instance, we need to spend 20 times more time training a SVM model, which is certainly not suitable for such a "Black-Box" model as we need too much time to tune the parameters.

4.1.4 Adaboost

Adaptive Boost is one of the boosting algorithms that iteratively learns from data that is sampled with different weight. At each iteration, a weak learner will assign each misclassified sample with a higher weight, so that they will be sampled with higher probabilities in the next iteration. There are mainly two parameters needed to be tuned in our real-case study: **the number of iteration** and **the learning rate**. For the choice of the tree used inside the Adaboost, we don't think it needs to be well-orchestrated, because Schapire proved in 1990 that if multiple weak classifiers are integrated together, it will have the generalization ability of PAC (Probably Approximately Correct) strong classifiers. We use grid search to find out the best parameters for Adaboosting, the selections are as follows.

The number of estimator	Learning rate
$[40, 80, 160]$	$[0.001, 0.05]$

4.1.5 Gradient Boosting

Like adaptive boosting, gradient boosting uses decision trees of a fixed size as base learners, then combines weak "learners" into a single strong learner in an iterative fashion. Hastie et al. comment that typically the depth of trees work well in between 4 and 8 for boosting and results are fairly insensitive to the choice of depth in this range. In our case study, we mainly focus on tuning the number of iterations and the learning rate.

4.1.6 Stacked LSTM

When utilizing LSTM to train our model, we will set **shuffle=False** as we want to preserve the past time information. Hence, the batch-size we set should be small enough to persist the time sequence information. Multiple layers LSTM [2] is an extension to LSTM that has at least two hidden layers where each layer contains multiple memory cells. When the neural network goes deeper, the model has the ability to continuously recombine the well-learned features from prior layers and construct advanced representations for final output. By the way, we tried to utilize Bi-directional LSTM but it does not perform well as the model absorbs both the past and future information and loses the ability to predict future value.

Hyperparameters	Optimal
Layers	2
Shuffle	False
Time Steps	5
Learning Rate	0.001
Hidden Size	15

* Multiple layers LSTM could enhance the representation extraction, but the training time would increase dramatically while the number of layers increases.

* If we shuffle the data, the training part is dissatisfied with chronological order.

* Similar to 'Shuffle', we need to set the number of cells small enough to ensure each training persists in time order. In the monthly frequency multi-factor stock selection, the number of samples that meet the requirements will be inadequate if we increase the number of cells (the stock is suspended and the factors will be missing), and if the parameter is too small, it will be difficult to exert the characteristics of the neural network.

* If the learning rate is too small, the convergence rate is very slow. However, in this task, the change of learning rate has no obvious effect on the final result.

* Hidden size is the number of features in the hidden state. There is no restrict limit for this parameter but we choose a smaller hidden size as we want to enhance the representation extraction.

4.2 Results Analysis in Out Sample Testing

4.2.1 Evaluation indicators

We chose four evaluation indicators and their functions are:

$$Annual\_Rate = \prod_{i=0}^{n} ret^{\frac{12}{n}} - 1 \tag{1}$$

$$Sharp\_Ratio = \frac{annual\_rate}{std(ret) * n^{0.5}} \tag{2}$$

$$Maximum\_drawdown = \frac{\prod_{i=0}^{n} ret^{\frac{12}{n}} - 1}{max_t \prod_{i=0}^{n} ret^{\frac{12}{n}} - 1} - 1 \tag{3}$$

$$WinRate\_of\_LongShort\_Portfolio = \frac{\sum_0^n I\{ret > 1\}}{n} \tag{4}$$

4.2.2 Results based on Multimodels combination

Cross validation We select the first segment to train the model, and after the training is complete, use the model to make predictions on the cross-validation set. Select the set of parameters with the highest cross-validation set AUC (or average AUC) as the optimal parameters of the model.
After determining the optimal parameters, use the preprocessed features of all samples (ie individual stocks) in the cross-section period at the end of month T as the input of the model, and obtain the predicted value $f(x)$ each sample for month $T + 1$ (synthetic factor, that is, the average of the classification vote results of each decision tree in the forest). Combinations of strategies can be constructed based on this predicted value.

Results of long-short portfolio

Model	K-fold cross-validation	Annualized Return	Sharp Ratio	Maximum drawdown	Win Rate
Logistic Regression	k=5	22%	2.07	-7%	57%
Random Forest	k=5	**24%**	**2.245**	**-5%**	**73%**
SVM(linear)	k=1	2%	0.281	-7%	67%
Ada Boost	k=5	8%	1.162	-10%	57%
Gradient Boost	k=1	21%	2.4	-5%	73%
LSTM	k=1	8%	1.04	-9%	70%

As can be seen from the table above, the random forest model with a cross-validation achieved the best results in all the directions in this multi-factors strategy. We also find logistic regression performed well with the quickest speed, whereas the model based on SVM and Adaboost took so much longer time in training but performed badly. So we can arrive at the conclusion that when doing multi-factor-cross-profile stocks selection, binary choice is enough and random forest can better our choice by voting and multi-trial. We also tried whether to use k-fold cross-validation or not, the idea is using the auc score in the different cross validation folds to choose the best parameters. The results turned out to be very time-consuming and less contributory. So we think when we choose stocks from the all market pool, the dataset is very large and don't need further validation, while when choosing from indexes' component stocks like stocks in HS300, the dataset is much smaller and we may need cross validation to make our model stable.

4.2.3 Further Analysis on SVM

In all model experiments, the time spent by the SVM model is almost inversely proportional to its performance, we spent almost 5 times more than the average training time of other models to train the SVM, but got the worst results, we decided to further explore is What factors affect the performance of SVM on multi factor strategy. Under the premise of fixing other parameters, we experiment with different kernel functions and get the following results.

Results of long-short portfolio

Model	K-fold cross-validation	Annualized Return	Sharp Ratio	Maximum drawdown	Win Rate
SVM(linear)	k=1	2%	0.281	-7%	67%
SVM(Polynomial)	k=1	15%	1.961	-3%	73%
SVM(Sigmoid)	k=1	-1%	-0.162	-18%	37%
SVM(Gaussian)	k=1	**21%**	**3.249**	**-2%**	**73%**

We found that using the same SVM model, the performance of different kernel functions is very different. The Annual Return using the Gaussian kernel function reaches 21%, which is close to the optimal level. Sharpe Ratio even exceeds the performance of Random Forest. The Annual Return of the SVM of the Sigmoid kernel function turned out to be a negative number. Different kernel functions are different mappings to the data, so we believe that in a relatively complex model, feature extraction often has a greater impact on the final performance of the model. Tree models may be less sensitive to this.

4.2.4 Results based on Multimodels combination

Based on the conclusion above, we think we can further improve our model by training multiple models at the same time and make decisions by voting from the model level. So we trained the models separately and added up the probability in their predictions to stimulate a vote with equal rights. The results are as follows:

Results of long-short portfolio (combinational model)

Model	K-fold cross-validation	Annualized Return	Sharp Ratio	Maximum drawdown	Win Rate
LR + GB	k=1	**27%**	2.281	-6%	**75%**
LR + AB	k=1	13%	1.776	-5%	65%
AB+LR+GB+RF	k=1	16%	1.86	-5%	62%
LR+GB+RF	k=1	23%	**2.565**	**-4%**	73%

It is clear that the voting system doesn't always help. LR+GB and LR+GB+RF are the two best combinations among all combinations. LR+GB has the largest long-short portfolio and win rate and LR+GB+RF has the smallest MDD and largest sharpe ratio, which mean they are suitable for risk-preferred and risk-avoid investors, respectively.
The long-short portfolio comparison plot and 5-tier portfolio of LR+GB are as follows:

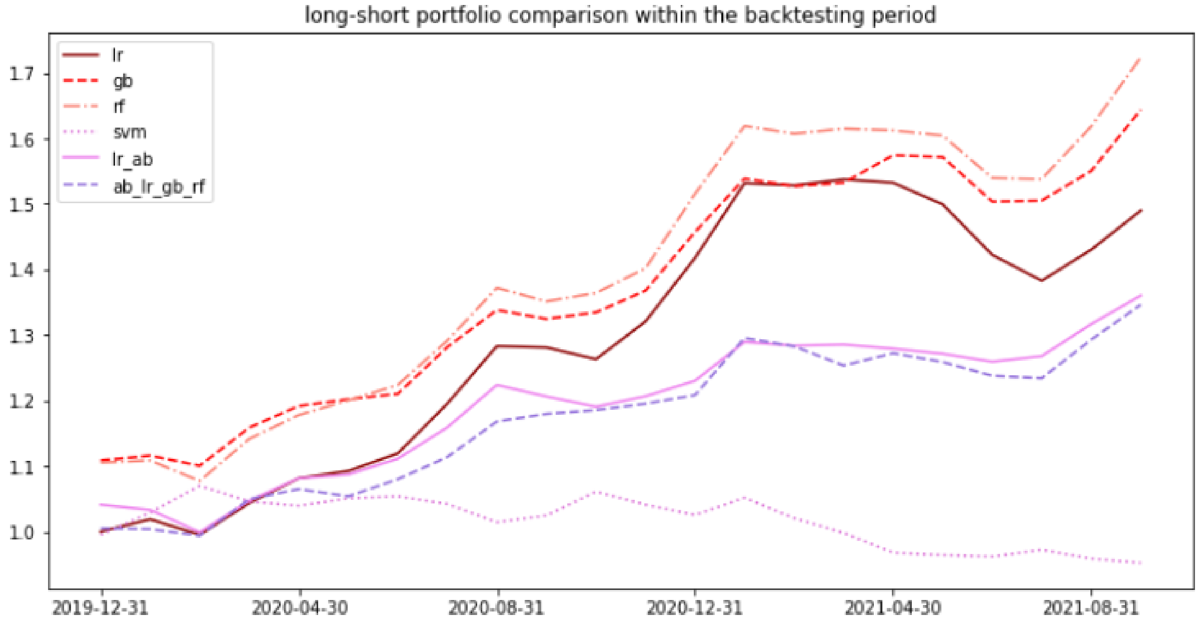

Figure 4.1: Long Short Portfolio comparison with different Models

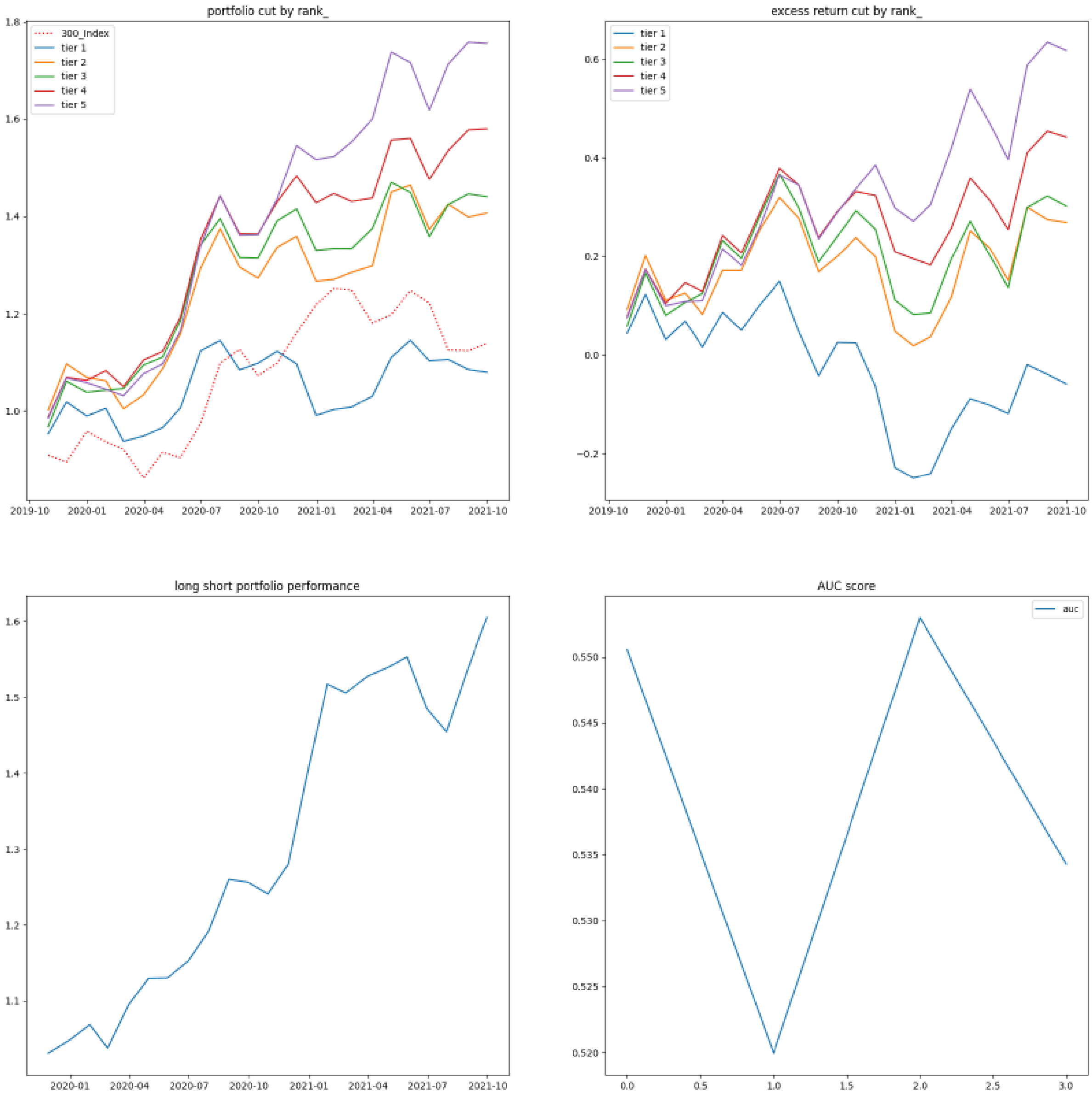

Figure 4.2: Evaluation of Best performance Model

From the 5-tier portfolio we can find out the long tier's portfolio performed much better than the HS300 index during this period, which means the long indicator we find can actually gain alpha; and in subplot 2, we can find out the excess portfolios achieved by these 5 tiers didn't have much overlap period and are very distinguishable in longer terms, which means our strategy works well in separating the long indicators and short indicators successfully. We can also compare the HS300 index with the long-short portfolio performance in subplot3, when HS300 went very well (2020.07 2020.08 & 2020.11 2021.01), the portfolio of our strategy went even better, which means our strategy can capture the bull market and enhance the portfolio; when HS300 fell into a adjusting period (2021.07 2021.10), the portfolio of our strategy was still making distinguishable excess portfolio, which means we can still use it in bear market, the AR and WINRATE also backed this conclusion. Also we have to notice that when the HS300 had a big drop (2021.06 2021.07), the portfolio of our strategy may drop accordingly, which means its ability to reduce risk is not very good and the MDD also showed this conclusion, that is also the reason why our strategy is more suitable for risk-preferred investors and with its sharpe ratio at 2.281, this strategy can be a good alpha maker.

5 Discussion

In this project, the first difficulty we encountered was data cleaning. Since we used real stock data and its corresponding factors, there were a large number of missing values or outliers in the data, although we had preprocessing and filling at the beginning but there are lots of table connection required and when merging between tables, it is still prone to data mismatch and error, so we need to strictly pay attention to the missing values in the table when modifying the model or using the algorithm, which is the reason we use many dropna() functions in the code.

We also encountered some difficulties when building the model. For example, sklearn's model function is relatively complete, but LSTM needs to be rebuilt with pytorch and implement similar functions. How to make the two use the same interface in the function has also puzzled us for a long time.

When using the LSTM model, we did not design the DataLoader and Dataset functions well, which resulted in not making good use of the time series information of the data. We suspect this is the reason for the poor performance of the LSTM model. Further, perhaps we can compare the results predicted by factors with those predicted by time series models such as AR and MA, and analyze who has the advantage in quantitative trading.

6 Conclusion

References

[1] Huatai Artificial Intelligence Series 5: Random Forest for Artificial Intelligence Stock Picking (2017)

[2] Stacked Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction

[3] Application of logistic regression models to assess household financial

[4] Euro area GDP forecasting using large survey