Quantitative Multifactor Investing

Hanxiao Zhu, Agam Gupta, Samuel Mankoff, Mohammad Dawood Junaid

December 10, 2022

Abstract

We build quantitative models for China's stock market by constructing return prediction factors using Gradient Boosting algorithms. The nature of our multifactor model is a multiple linear regression about current factor exposures and future stock returns. At the end of this report, models will be explored to develop diverse trading strategies and portfolios. As a benchmark, we use the CSI 300 index and outperformed it consistently from January 2015 to December 2019.

1 Introduction

Definitions and Terminology:

- **Long:** long trades are those intended to profit from rises in stock prices (buying stocks and profiting from the increase in price)
- **Short:** short trades are intended to profit from drops in stock prices (selling a borrowed stock and buying it at a lower price)
- **Stratification:** Our model predicts the % return and ranks all stocks. After the predictions from the model, we divide the stocks into 5 groups which are used for different strategies
- **Industry Neutralization:** We don't want to be influenced by the true actual values of the stocks; Therefore, neutralization is just the standardization method in financial analysis. So, we neutralize industry influence and standardize feature values
- **Sharpe ratio:** Return divided by risk (volatility)

$$Sharpe\ Ratio = \frac{R_p - R_f}{\sigma_p}$$

Where R_p is the return of the portfolio, R_f is the risk-free rate(which is small and ignored here), σ_p is the standard deviation of the portfolio's excess return.

- **Max drawdown:** Portfolio's max loss from a peak to a trough

$$Max\ drawdown = \frac{Peak\ Value - Trough\ Value}{Peak\ Value}$$

- **Rolling Training**: We initially train the model using 12-month data from 2015. This 2015 model is used to make predictions for January 2016. Now, when making predictions for February 2016, we use data from the past 12 months (Feb 2015 - Jan 2016)

As of October 2020, the total value of China's stock market has climbed to a record high of more than USD 10 trillion (RMB 67 trillion), making it the second-largest in the world, after the US— with a total value of nearly USD 39 trillion.

The gigantic market volume of the Chinese stock market makes it extremely attractive for academic research and allows us to explore financial questions that contribute to our understanding of quantitative finance.

In this project, we propose to make a multifactor quantitative model that makes stock returns monthly based on a multitude of features.

We use these predictions to stratify the stocks into 5 groups based on the percent return (the first group has the highest % returns while the last group has the worst % returns). We use a rolling training method to train the model continuously and make monthly predictions.

Chart 1 Basic information on modeling

Asset Class:	• Stocks in China A
Major Assumptions:	1. Factors keep efficient in the future 2. There are no taxes or transaction cost
Prediction Method:	• Rolling training
Training Set Length:	• 12 months
Stratification:	• Quintile (group 1 to 5, best to worse)
Stock Weight Allocation:	• Equal-weighted
Benchmark Index:	• CSI 300 (value of 300 selected stocks)
Trading Frequency:	• Monthly

Moreover, our quantitative multifactor model consists of three parts:

1. **Data**
   This part contains data preprocesses such as standardization, neutralization, extreme value removal, one-hot encoding, and feature selection.
2. **Model & prediction**
   We will construct our model here and then start rolling training by predicting the return rates monthly with different parameters.
3. **Trading Strategy**
   We design specific trading strategies and visualize relevant back-testing according to the stock prediction results.

In this report, after training the model, we devise 2 primary strategies to invest and outperform the index (value of 300 selected stocks):

1. **Long-Short Strategy (Machine Learning Model 1):** Buy the stocks stratified into group 1 and short group 5 based on gradient boosting model predictions.
2. **Long-only Strategy (Machine Learning Model 2):** Buy the stocks stratified into group 1 based on gradient boosting model predictions.

After creating the strategies and obtaining the necessary data, we compared MLM1 and MLM2 to the common stock index (CSI 300) as well as the benchmark linear regression model.

In particular, we successfully identify at least two key characteristics of the Chinese stock market. First, simple linear models are losing effectiveness in recent years. Second, our long-short strategy, if available, performs much better than our long-only strategy.

2 Review of existing literature

Our work is based on stock return predictions, which come in two basic strands.

The first strand models differences in expected returns across stocks as a function of stock-level characteristics, and is exemplified by Fama and French (2008) and Lewellen (2015), who run the cross-sectional regressions of future stock returns on a few lagged stock characteristics.

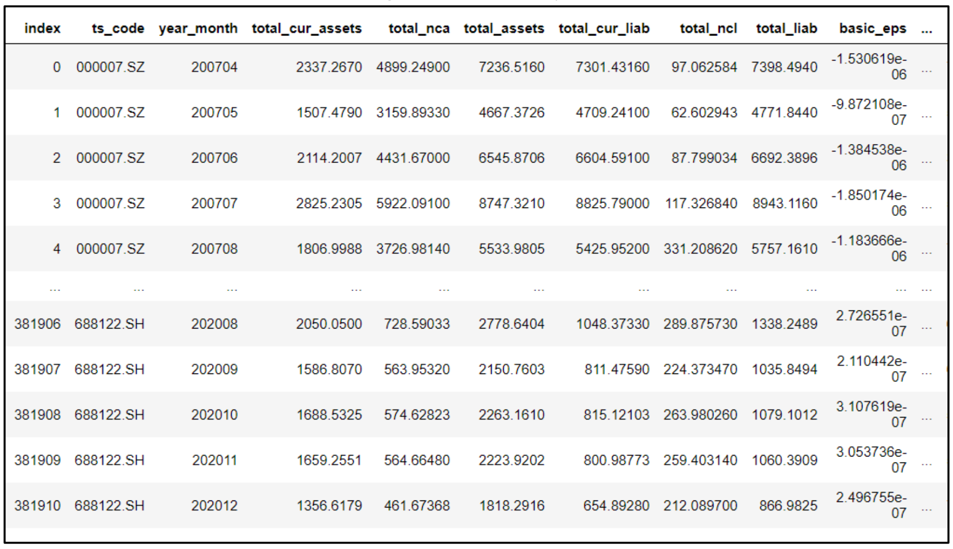
The second strand is studied by Shihao Gu (2020) and Markus Leippold (2021) who forecast the stock returns using machine learning techniques.

3 Data preprocessing and feature selection

The data source is a Chinese data provider called Tushare. The reason for choosing it owes much to our internship experience where Tushare's data was used and impressed us with its quality.

We merely choose basic stock and industry information to do our research, which is challenging and complicated enough to accomplish in the limited time span of a few weeks.

Fig. S 1 Data example

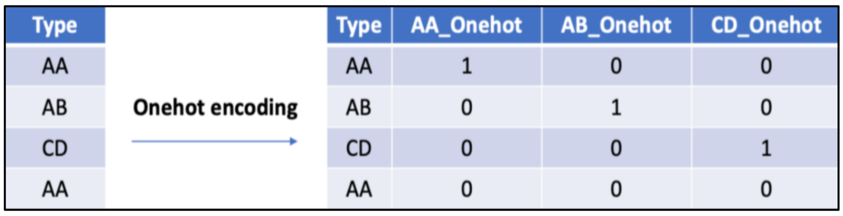


index	ts_code	year_month	total_cur_assets	total_nca	total_assets	total_cur_liab	total_ncl	total_liab	basic_eps	...
0	000007.SZ	200704	2337.2670	4899.24900	7236.5160	7301.43160	97.062584	7398.4940	-1.530619e-06	...
1	000007.SZ	200705	1507.4790	3159.89330	4667.3726	4709.24100	62.602943	4771.8440	-9.872108e-07	...
2	000007.SZ	200706	2114.2007	4431.67000	6545.8706	6604.59100	87.799034	6692.3896	-1.384538e-06	...
3	000007.SZ	200707	2825.2305	5922.09100	8747.3210	8825.79000	117.326840	8943.1160	-1.850174e-06	...
4	000007.SZ	200708	1806.9988	3726.98140	5533.9805	5425.95200	331.208620	5757.1610	-1.183666e-06	...
...	...	...	...	...	...	...	...	...	...	...
381906	688122.SH	202008	2050.0500	728.59033	2778.6404	1048.37330	289.875730	1338.2489	2.726551e-07	...
381907	688122.SH	202009	1586.8070	563.95320	2150.7603	811.47590	224.373470	1035.8494	2.110442e-07	...
381908	688122.SH	202010	1688.5325	574.62823	2263.1610	815.12103	263.980260	1079.1012	3.107619e-07	...
381909	688122.SH	202011	1659.2551	564.66480	2223.9202	800.98773	259.403140	1060.3909	3.053736e-07	...
381910	688122.SH	202012	1356.6179	461.67368	1818.2916	654.89280	212.089700	866.9825	2.496755e-07	...

One-hot encoding

With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns.

Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1. Here is a chart for understanding:



Type
AA
AB
CD
AA

Type	AA_Onehot	AB_Onehot	CD_Onehot
AA	1	0	0
AB	0	1	0
CD	0	0	1
AA	0	0	0

Shrinkage: Impurity-based feature importance

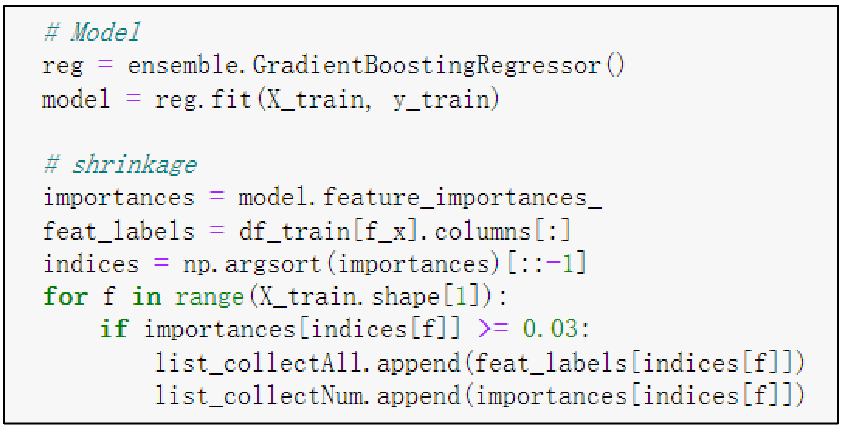
The relative rank (i.e. depth) of a feature used as a decision node in a tree can be used to assess the relative importance of that feature with respect to the predictability of the target variable.

Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The expected fraction of the samples they contribute can thus be used as an estimate of the relative importance of the features.

By averaging the estimates of predictive ability over several randomized trees one can reduce the variance of such an estimate and use it for feature selection.

Along the process of conducting the shrinkage, we also remove the extreme feature values and null values using the classes “MedianExtremeValueTransformer” and “GroupValueFiller” as defined in the code.

Fig. S 2 Code of shrinkage



```python
# Model
reg = ensemble.GradientBoostingRegressor()
model = reg.fit(X_train, y_train)

# shrinkage
importances = model.feature_importances_
feat_labels = df_train[f_x].columns[:]
indices = np.argsort(importances)[::-1]
for f in range(X_train.shape[1]):
    if importances[indices[f]] >= 0.03:
        list_collectAll.append(feat_labels[indices[f]])
        list_collectNum.append(importances[indices[f]])
```

To select features, we set the threshold of feature importance as 0.03, in which case the number of all 88 features would shrink to only 40.

4 Methods &Modeling

There are some methods we used in our coding:

1. **Benchmark Model: Linear Regression**

In order to measure the incremental predictive power of our next model in which machine learning techniques will be used, we need a benchmark to compare.

We train a multifactor linear regression model and use stratified group 3.

2. **Machine Learning Model(MLM): Gradient Boosting Regression**

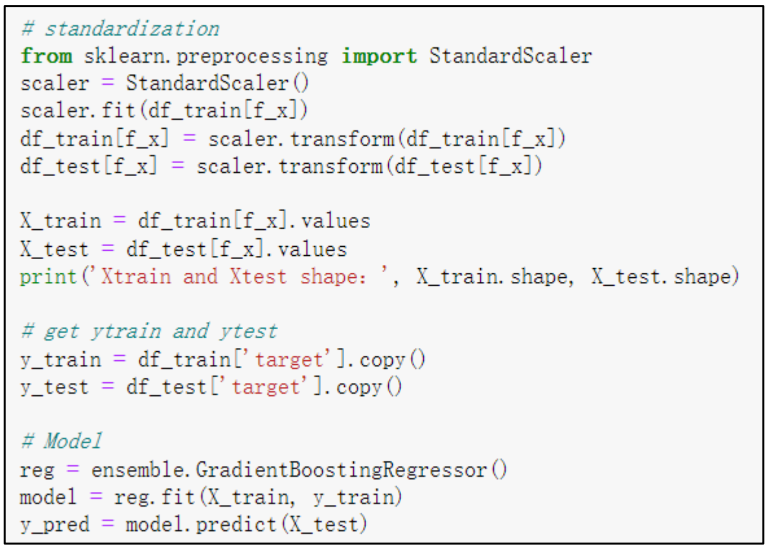
Gradient boosting is one of the most powerful techniques for building predictive models for both classification and Regression problems.

Fig. S 3 Gradient Boosting Algorithm

```
1. $F_0(\mathbf{x}) = \arg\min_\rho \sum_{i=1}^{N} L(y_i, \rho)$
2. For $m = 1$ to $M$ do:
3. $\tilde{y}_i = -\left[\frac{\partial L(y_i, F(\mathbf{x}_i))}{\partial F(\mathbf{x}_i)}\right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})}, \; i = 1, N$
4. $\mathbf{a}_m = \arg\min_{\mathbf{a},\beta} \sum_{i=1}^{N} [\tilde{y}_i - \beta h(\mathbf{x}_i; \mathbf{a})]^2$
5. $\rho_m = \arg\min_\rho \sum_{i=1}^{N} L(y_i, F_{m-1}(\mathbf{x}_i) + \rho h(\mathbf{x}_i; \mathbf{a}_m))$
6. $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \rho_m h(\mathbf{x}; \mathbf{a}_m)$
7. endFor
   end Algorithm
```

Intuitively, gradient boosting is a stage-wise additive model that generates learners during the learning process (i.e., trees are added one at a time, and existing trees are not changed).

Fig. S 4 Code of Gradient Boosting



```
# standardization
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(df_train[f_x])
df_train[f_x] = scaler.transform(df_train[f_x])
df_test[f_x] = scaler.transform(df_test[f_x])

X_train = df_train[f_x].values
X_test = df_test[f_x].values
print('Xtrain and Xtest shape: ', X_train.shape, X_test.shape)

# get ytrain and ytest
y_train = df_train['target'].copy()
y_test = df_test['target'].copy()

# Model
reg = ensemble.GradientBoostingRegressor()
model = reg.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

It also allows for the optimization of arbitrary differentiable loss functions. In each stage, a regression tree is fit on the negative gradient of the given loss function.

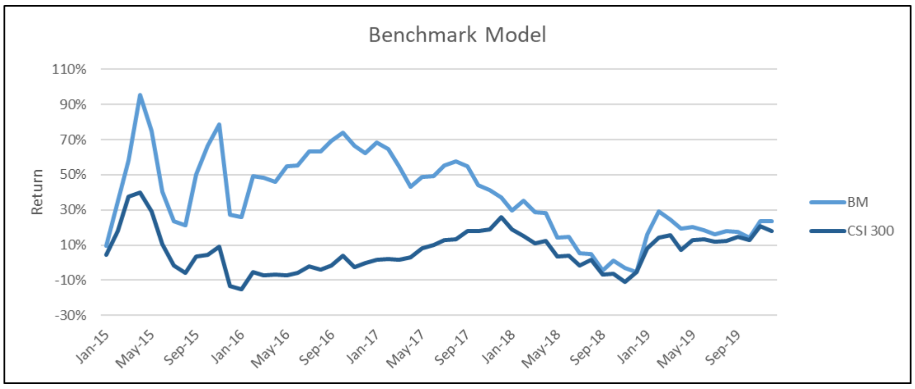
We use the predictions from this model to generate trends and develop long-short and long-only strategies.

5 Results

Here are our models with different trading strategies and summaries (Re-plot in Excel):

A. Benchmark Model (BM): Long-only

To measure the incremental predictive power of our next model where machine learning skills will be used, we build a benchmark to compare with.

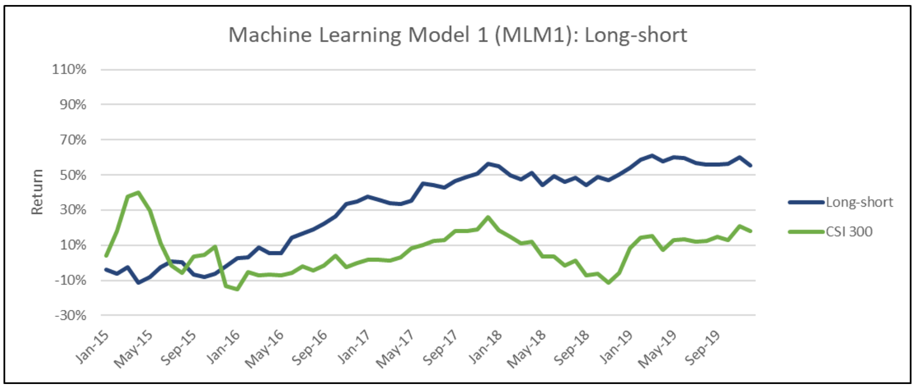


- **Model Algorithm:** Multiple Linear Regression
- **Trading Strategy:** Long stock group 3

B. Machine Learning Model 1 (MLM1): Long-short

Boosting philosophy has gained tremendous popularity in many fields of science, technology, and increasingly in our daily lives, including finance and investment.

In MLM1, we hope it can perform better than our benchmark model after machine learning techniques are introduced.

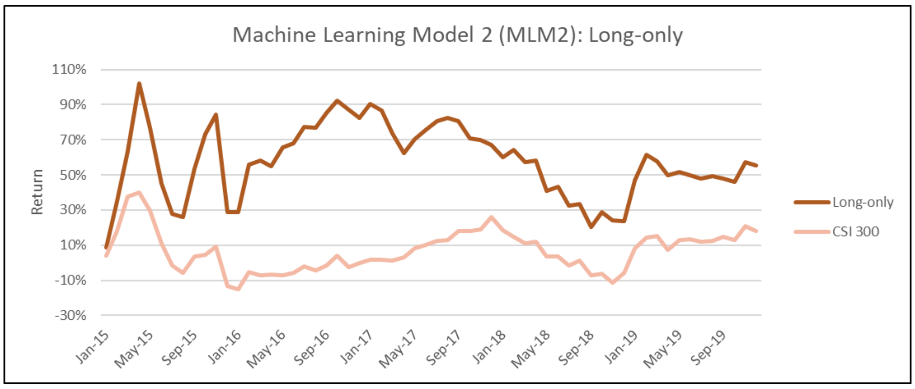


- **Model Algorithm:** Gradient Boosting Regression
- **Trading Strategy:** Long stock group 1 and short group 5

C. Machine Learning Model 2 (MLM2): Long-only

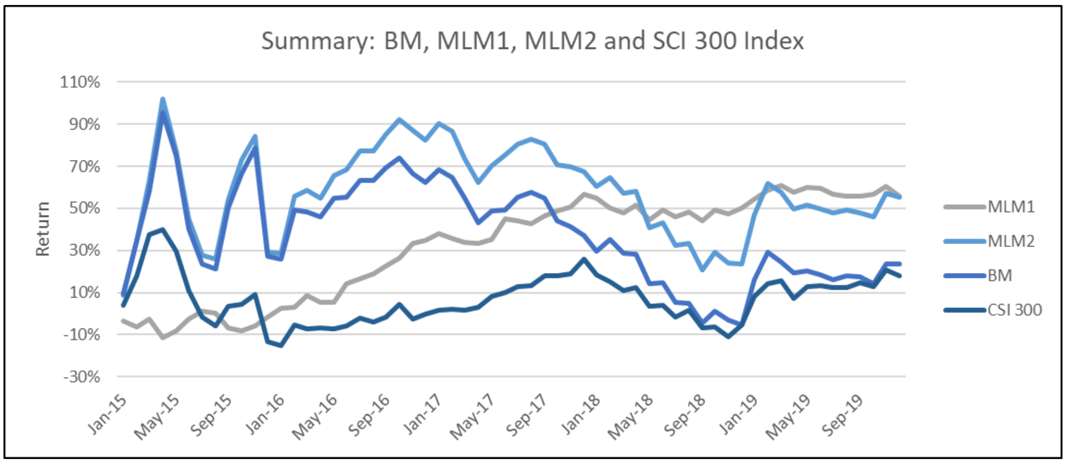
Considering that the Chinese stock market has many restrictions on shorting, we modify our machine learning model to make it as practical as possible.

In MLM2, we simulate a real trading environment by using the long-only strategy.

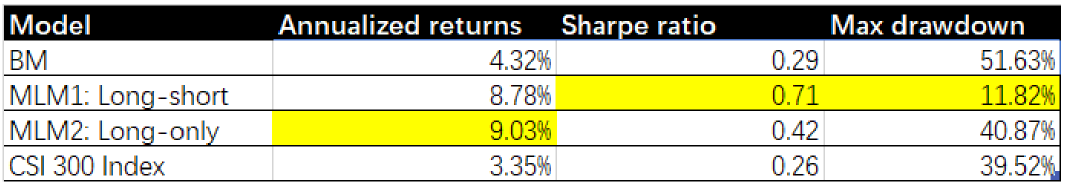


- **Model Algorithm:** Gradient Boosting Regression
- **Trading Strategy:** Long stock group 1

D. Summary 1: Back-testing Plotting



E. Summary 2: Result Table



Model	Annualized returns	Sharpe ratio	Max drawdown
BM	4.32%	0.29	51.63%
MLM1: Long-short	8.78%	0.71	11.82%
MLM2: Long-only	9.03%	0.42	40.87%
CSI 300 Index	3.35%	0.26	39.52%

6 Conclusion

Our portfolio analysis shows that the high predictability of our model translates into the highest Sharpe ratios and lowest Max drawdown for long-short portfolios(MLM1).

In particular, machine learning algorithms such as gradient boosting provide us with robust performance with a long-short strategy.

Shorting stocks in the Chinese market, however, is not practicable. Therefore, we analyze the long-only portfolio(MLM2) and find that its performance remains economically significant.

Overall, we show that machine learning methods can be successfully applied to the Chinese stock market by using multifactor investment strategies, even if the financial market is becoming more and more mature and efficient.

7 Discussion about future directions

Potential future research could expand in some of these directions:

First, other data can be added to our models such as market analyst sentiment, news, events, or user information, which can contribute to new features which could hugely improve our models' performance.

Second, instead of only trading monthly, allowing the possibility of adjusting the trading frequency could also be a prospective direction. This could cause factors about the volatility of prices to become more impactful when trading more frequently.

Third, portfolio management techniques like giving different weights to stocks in the same group could also be useful, in which the optimization theory would become relevant.

Fourth, other machine learning or deep learning algorithms could be considered and validated as well, allowing us to understand how the model could be improved.

Reference

[1] Friedman J H. Greedy Function Approximation: A Gradient Boosting Machine[J]. ***Annals of Statistics***, 2001, 29(5):1189-1232.

[2] Qi M. LightGBM: A Highly Efficient Gradient Boosting Decision Tree[C]. ***Neural Information Processing Systems***. Curran Associates Inc. 2017.

[3] Friedman J, Hastie T, Tibshirani R. Additive Logistic Regression: A Statistical View of Boosting[J]. ***The Annals of Statistics***, 2000, 28(2):337-407.

[4] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System[J]. ***ACM***, 2016.

[5] Fama, E. F., and K. R. French. Dissecting anomalies. ***The Journal of Finance***, 2008, 63:1653–1678.

[6] Fama, E. F., and K. R. French. A five-factor asset pricing model. ***Journal of Financial Economics***, 2015, 116:1–22.

[7] Lewellen, J. The Cross-section of Expected Stock Returns. ***Critical Finance Review***, 2015, 4:1–44.

[8] Gu S, Kelly B, Xiu D. Empirical Asset Pricing via Machine Learning[J]. ***Review of Financial Studies***, 2020, 33.

[9] Leippold M, Wang Q, Zhou W. Machine learning in the Chinese stock market[J]. ***Journal of Financial Economics***, 2021(1).