MAFS5440 Project2 Replication

Congxin He 21024425: OLS, OLS3, ENet
Jiping Wang 21105827: PCR, PLS, GLM
Yuwei Chen 21106455: NN1-5
Zilin Wu 21022271: Random Forest, XGBoost

Abstract

The project aims to explore stock market data to predict stock returns. The dataset covers stock characteristics and monthly total returns of all companies listed on the NYSE, AMEX, and NASDAQ from 1957 to 2016. Through data preprocessing, including handling missing values, feature interaction construction, and one-hot encoding, we created a comprehensive dataset containing 920 features. We selected the top 20 stocks with the largest market capitalization and the bottom 20 stocks with the smallest market capitalization as the datasets for model fitting.

We employed various models for fitting, including Ordinary Least Squares (OLS), Elastic Net (ENet), Principal Component Regression (PCR), Partial Least Squares (PLS), Generalized Linear Models (GLM), Random Forest (RF), XGBoost, and Neural Networks (NN1-5). Each model was trained and tested through cross-validation and recursive performance evaluation schemes. Except for the OLS model, all models exhibited positive global R^2 values, indicating a certain level of predictive accuracy. We also observed some interesting phenomena, such as a gradual decline in R^2 and specific time periods showing low points. Although our model results differ in some aspects from those of the original paper, we believe our data preprocessing methods are more robust, and in some cases, our model performance surpasses that of the original study.

1 Data Exploration

The data includes monthly total individual equity returns and some stock characteristics from CRSP for all firms listed in the NYSE, AMEX, and NASDAQ from January 1957 to December 2016. The number of stocks in the data is almost 30,000, with the average number of stocks per month exceeding 6,200. Except this, we also obtain the Treasury-bill rate to proxy for the risk-free rate. Then we do some data preprocessing. After constructing the full dataset, we selected the largest 20 stocks and the smallest 20 stocks monthly according to `mvel1` as the top dataset and bottom dataset for model fitting.

1.1 Missing Value

Sic2: Among all the features, `sic2` is a special feature, for it represents the industry in which the stock is located, as an integer, so we cannot interpolate directly. The method adopted here is to fill the null value of `sic2` into the mode of the industry in which the stock is located, and directly delete the null value if it still exists after processing.

Other Features: We fill in the missing values with the median of the time section. If there are still missing values, we fill them with 0. This will not affect the result as we will later convert the cross-sectional ranking of all data to between -1 and 1.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

1.2 Date Mapping

We cross-sectionally rank all stock characteristics period-by-period and map these ranks into the [-1,1] interval.

1.3 Interaction

In addition to the 94 stock-level characteristics contained in the raw data, we also constructed eight macro characteristics detailed in Welch and Goyal (2008). Then, according to the original paper ($z_{i,t} = x_t \otimes c_{i,t}$), we constructed the interactions between stock-level characteristics and macroeconomic state variables.

1.4 One Hot Encoding

Since `sic` represents an industry category, its value has no practical significance, so we apply one hot encoding to convert these characteristics and obtain 74 dummy variables. Finally, we get $94 * (8 + 1) + 74 = 920$ features.

1.5 Label Construction

By the macro characteristics we calculate above, we calculate excess return (`label = ret - risk_free_rate`).

2 Model Fitting

2.1 Training Scheme

For each algorithm, the original dataset needs to be divided into three parts: train, validation, and test. And we adopt the recursive performance evaluation scheme to get the final results. The example diagram is shown below 1.

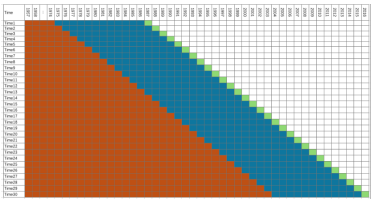


Figure 1: recursive performance evaluation scheme

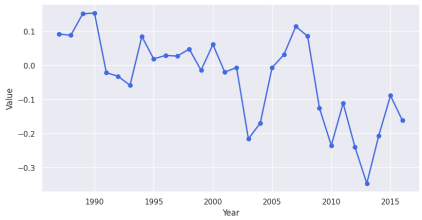


Figure 2: Yearly R^2 of OLS3

2.2 OLS and OLS3

OLS: In this algorithm, we use the Huber loss function, which can adopt the square loss for small errors and the linear loss for large errors, thus reducing the influence of outliers.

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{if } |y - f(x)| \le \delta \\ \delta \cdot (|y - f(x)| - \frac{1}{2}\delta) & \text{otherwise} \end{cases}$$

We used all the features, but due to equipment problems, OLS with 920 features was very slow, so we did not use the recursive performance evaluation scheme to train the model, but directly divided and fitted the model according to Time1. The resulting out-of-sample R^2 is -0.062. This may be unsurprising as the lack of regularization leaves OLS highly susceptible to in-sample overfit. Here, we show the feature importance of OLS 9.

OLS3: In this algorithm, we use the three features of `mvel1`, `bm` and `mom1m` for training, whose meanings are size, book-to-market and momentum respectively. We observe that the annual R^2 shows a downward trend 2, and the negative part accounts for the majority, which indicates that, due to the large amount of data, our model can not fully capture the trend of data change, so as to cause overfitting.

2.3 Elastic Net

Since OLS and OLS3 appeared to a certain degree of overfitting, we tried to use an elastic network model that could limit the model complexity and the number of parameters. We used two sets of models to fit. First, The parameters of our fixed model are $alpha = 1, l1\_ratio = 0$, which means that we use the Lasso model to fit the data recursively. Not surprisingly, we get R^2 greater than 0 with a value of 0.0047, which confirms our conjecture about why OLS is not working well. After that, we no longer used fixed parameters, but fixed $l1\_ratio$ at 0.5 and $alpha$ varying between 0.0001 and 0.1. Parameters were dynamically changed at each training time, and the best parameters were selected by the validation dataset for training and prediction. Here we show some hyperparameter tuning examples 20.

Except this, we observed the feature importance and the annual complexity of the model. We found that the complexity of the model (the number of features selected to have nonzero coefficients) stabilized at about 60 at the beginning of training. Then we increased year by year 10, which is also consistent with our hypothesis that the noise of the training data gradually increased. The noise increases gradually, and the model requires higher complexity (more nonzero coefficients) to fit large amounts of data.

However, we found an interesting phenomenon. We found that there was no big difference between Lasso and ENet of parameter adjustment, and Lasso was only a little worse than ENet. However, we found that during parameter adjustment, the maximum R^2 of the validation dataset showed a very rapid decline trend and stabilized at around 0.33 in the early period of training time. However, after 2007, there was even -0.08. We guessed the reason was that our training data increased year by year, resulting in increased noise, resulting in worse and worse model effects. Even though we adjusted the parameters, it could not completely solve the problem of noise, so R^2 showed a large downward trend. The R^2 on the test set is mostly smaller than the R^2 on the validation set, which is why our test set is positive overall, but monthly R^2 and yearly R^2 are both negative 8a.

2.4 PCR

Principal Component Regression (PCR) and Partial Least Squares (PLS) are two types of dimension reduction techniques.

For the original regression problem:

$$Y = X\theta + \epsilon$$

where Y and ϵ are both n-dimensional vectors, X is an $n \times m$ matrix, and θ is the weight vector.

They choose an $m \times k$ matrix W (where m is usually much less than k), so that the formula can be written as:

$$Y = (XW)\theta_W + \epsilon_W$$

where Y and ϵ_W are still n-dimensional vectors. However, XW is now an $n \times k$ matrix while θ_W becomes a k-dimensional vector.
In detail,PCR chooses W by maximizing the variance of XW. Specifically, the j-th principal component should satisfy:

$$w_j = \arg\max_w \mathrm{Var}(Xw), \quad \text{s.t. } w'w = 1, \quad \mathrm{Cov}\,(Xw, Xw_l) = 0, \quad l = 1, 2, \ldots, j-1$$

In our algorithm, we set the number of components (`n_components`) from 1 to 100.The final optimal parameter is `n_components` = 4.

For the predictive performance of the PCR model, we obtained a percentage R^2 of 0.46, with the final model complexity being 4 principal components 11. From the training results of the time series, the overall model complexity (the number of selected components) is relatively low. However, there were significant fluctuations during the periods 1990-1995 and 2005-2010, which may be closely related to economic events at those times.

2.5 PLS

Different from PCR, PLS chooses W by letting the j-th new factor be

$$w_j = \arg\max w\,\mathrm{Cov}^2(Y, XW), \quad \text{s.t.} \quad w'w = 1, \quad \mathrm{Cov}\,(Xw, Xwl) = 0, \quad l = 1, 2, \ldots, j-1$$

To optimize the number of components, we set a search range of 1 to 100, ultimately selecting 1 as the optimal count.

In the PLS analysis, the final model yielded an R^2 of 0.74, with an optimal model complexity of a single principal component. Based on the time series training results, the model's complexity is relatively low and significantly lower than that of the PCR model12. This may be attributed to PLS's specific approach to decomposing the dependent variable Y. Additionally, a noticeable increase in model complexity was observed for top stocks between 2013 and 2015. When combined with feature importance rankings, this suggests that the increase may relate to changes in key variables during those years.

2.6 Generalized Linear Model

Compared to the OLS model, the Generalized Linear Model (GLM) makes predictions by performing a K-term spline series expansion on the predictors. In this paper, we set (K = 3). Additionally, we employed the Huber robustness modification as the prediction target and used group lasso as the penalization function. The group lasso has the following form:

$$g(z;\theta,p(\cdot)) = \sum_{j=1}^{p} p(z_j)'\theta_j, \phi(\theta;\lambda,K) = \lambda \sum_{j=1}^{P} \left(\sum_{k=1}^{K} \theta_{j,k}^2 \right)^{1/2}$$

we expand it to (1, z, z^2) and adjust the parameter (λ) within the range of (1×10^{-4}, 1×10^{-6}). We found that the optimal (λ) is 0.0001, with the final R^2 of the model being -9.39. The complexity of the model fluctuated significantly over time, which may be due to the inclusion of group lasso as a penalized function in the GLM, causing some feature coefficients to approach zero 13.

2.7 Random Forest

Random Forest 3a is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes or the mean prediction of the individual trees. We think it can capture feature interactions effectively through its random selection of features at each split, allowing the model to explore various combinations of features. Therefore, we used the original dataset (i.e., containing only stock-level features, SIC2 code, and macro factors) rather than an interaction-transformed dataset for training.

(a) Random Forest (b) XGBoost

Figure 3: Structure of Random Forest and XGBoost

For our Random Forest model, the hyperparameters were selected based on the supplementary material 4:

To assess the complexity of the model and the importance of different features, we generated a graph showing the top 20 feature importances (also include top-1000 stocks and bottom-1000 stocks) 14. We use the depth of each tree in the forest to represent model complexity. To evaluate the time-varying model complexity, we choose the tree depth of the random forest at each training iteration.

2.8 XGBoost

In consideration of the relatively slow training speed of traditional Gradient Boosting Regression Trees (GBRT) models, we opted to substitute GBRT with XGBoost 3b in this study. XGBoost has several advantages over traditional GBRT, mainly including regularization. XGBoost includes L1 and L2 regularization terms in its objective function to prevent overfitting. The regularization term is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

where:

- γ is the minimum loss reduction required to make a further partition on a leaf node of the tree,
- λ is the L2 regularization term on weights,
- T is the number of leaves in the tree,
- w_j is the score on the j-th leaf.

For our XGBoost model, the following hyperparameters were chosen based on preliminary experiments and the supplementary material 4:

We calculate the variable importance 16 for the top20 feature importance in this model (also include top-1000 stocks and bottom-1000 stocks). The feature importance are normalized to sum to one. To evaluate the time-varying model complexity, we use the tree depth of the XGBoost model at each training iteration.

2.9 Neutral Network

Our final model utilizes neural networks 4. Neural networks have demonstrated remarkable effectiveness in addressing complex problems across various applications. Through multiple hidden layers and nonlinear activation functions, neural networks can uncover intricate, latent patterns within data. Therefore, we chose to leverage a neural network to tackle our problem. The structure of our neural network is as follows 5:

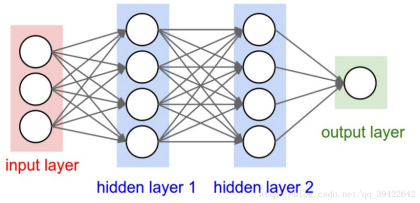


Figure 4: NN model

Model	Architecture
NN1	Single hidden layer of 32 neurons
NN2	Two hidden layers with 32 and 16 neurons
NN3	Three hidden layers with 32, 16, and 8 neurons
NN4	Four hidden layers with 32, 16, 8, and 4 neurons
NN5	Five hidden layers with 32, 16, 8, 4, and 2 neurons

Figure 5: NN Model Architectures

Neural networks have a wide range of parameters and various selection options. Based on the characteristics of our data and the problem we aim to solve, we have chosen the following neural network parameters 1:

where

$$\text{ReLU}(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{otherwise} \end{cases} \tag{1}$$

The penalized loss function can be defined as follows:

$$L = L_{\text{main}}(y, \hat{y}) + \lambda_1 L_{\text{regularization}} \tag{2}$$

where:

$$L_{\text{main}}(y, \hat{y}) = \text{MSE}(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, L_{\text{regularization}} = \sum_{j=1}^{m}\|w_j\|_1 = \sum_{j=1}^{m}\sum_{k=1}^{p_j}|w_{jk}|$$

Parameters	Values
Activation Function	ReLU Function
Loss Function	Penalized L2 objective function (L1 penalty $\lambda_1 \in (10^{-5}, 10^{-3})$)
Learning Rate	$LR \in \{0.001, 0.01\}$
Batch Size	10000
Epochs	100
Patience	5
Optimizer	Adam

Table 1: NN Model Parameters and Values

We train the neural network models recursively, expanding the training dataset by one year with each iteration. For each expanded dataset, we train a new model and tune 17 its parameters using the corresponding validation set. After identifying the optimal parameters, we re-train the model on the combined training and validation sets and proceed with prediction and evaluation. This process provides the best parameters for each iteration, minimizing validation loss. According to the R^2 results, NN3 consistently outperforms the other neural network models in most cases.

We believe this is because a smaller number of layers may lead to underfitting, while a larger number of layers can result in overfitting. NN3 achieves the right balance with three hidden layers, making it most suitable for our application.

Additionally, we observed that as time progresses and the training set grows, the model's performance fluctuates significantly, occasionally producing extreme outliers. This may be due to the increased noise in the data as the time range expands. For time series data in finance, a larger time range often introduces more noise, and the correlation between distant data points diminishes, ultimately disrupting the model's predictions for future data.

For feature importance 19, we choose to use the coefficient of the first linear layer of each NN model as the value of each feature's importance and calculate the importance ratio of each feature. As shown in the figure below, the feature importance of NN1-NN5 is very close, among which `pchsale` and `rd` rank first and second, followed by `sic2, std` and `bm`.

3 Model Analysis

3.1 Model Performance

As mentioned previously, nearly all models have exhibited poor performance over the last five years, making it unsurprising that the yearly 2 and monthly 3 R^2 are negative. Consequently, we focus our comparison on global R^2 6. We observe that, except for the R^2 of OLS3, all other models yield positive values. This highlights the accuracy of our model.

3.2 Feature Importance

For OLS and ENet, industry classification is of great importance. Different industries have different economic cycles and driving factors, and our model can help identify trends and opportunities within the industry. In addition, we find that some features are of great importance in all models, such as momentum factor `mom1m`, `mom12m`, which indicates that the market has a certain trend effect, our model can use the price trend to capture market inertia, and fundamental factors (`bm`, `ep`, `tb`, `sp`), which means our model can evaluate the intrinsic value of assets according to the company's financial status, and find out the undervalued stocks. Emotional factor (`ms`) can reflect irrational market behavior and capture short-term opportunities, and risk factor (`retrol`) can reduce potential losses 22.

To further investigate feature stability across different models, we analyzed the marginal relationship between the top four continuous features—`rd` (R&D expenditure), `turn` (share turnover), `sin` (sin stocks), and `acc` (working capital accruals)—and expected return in all models 21. Firstly, we observe that for `rd`, when values fall below -0.1 or exceed 0.85, there is an associated increase in future returns, a trend particularly pronounced in the XGB model. Conversely, for `turn` and `sin`, all models

	OLS3	ENet	PCR	PLS	GLM	RF	XGBoost	NN1	NN2	NN3	NN4	NN5
All	-1.06	0.68	0.46	0.74	-2.50	6.13	0.71	0.32	1.08	1.54	1.33	0.54
Top	4.02	0.90	0.46	-0.38	-9.39	6.26	-4.20	2.42	3.54	3.86	2.72	1.35
Bottom	3.86	0.28	0.11	-0.46	-7.66	6.19	3.39	2.89	2.37	1.99	3.08	0.64

Figure 6: Global R^2 (percentage of R^2_{oos})

consistently indicate an almost zero relationship with future returns. Secondly, for the `acc` feature, which holds high importance in the PLS, PCA, eNet, and NN models, we observe that future returns tend to decrease as acc increases.

We also calculate an importance measure for each macroeconomic predictor, normalized to sum to one within each model 7. Most models consistently identify `dp` (dividend-price ratio) as a significant predictor, especially in the ENet and PCR models, where it has considerable weight. In contrast, `ntis` (net equity expansion) generally has lower importance, with only minor influence across most models. Tree-based models, such as RF and XGBoost, show a notable emphasis on `tbl` (Treasury bill rate) and `svar` (stock variance), highlighting these variables more than other models. Linear and generalized linear models, such as OLS and GLM, heavily favor the bond market variables `tbl` and `bm` (book-to-market ratio), emphasizing these over other predictors. Neural networks (NN1 to NN5) assign relatively uniform importance across all eight indicators, likely due to their architecture, which allows for capturing complex interactions without overfitting individual predictors. This distribution across indicators suggests that neural networks balance multiple predictors, rather than focusing heavily on any single one.

3.3 Comparison with the original paper

There are some differences between our results and the original paper, and the possible reasons are as follows:

1. The data fluctuates from year to year, so there may be some differences in the original data we use;

2. In terms of data preprocessing, there are some differences between us and the original paper. For example, we used mode filling for `nan` filling of `sic2`, while the original paper did not specify the filling method;

3. The parameters are not exactly the same every year. It is difficult for us to ensure that the model parameters are completely consistent with the original paper, but we have tried our best to adjust the parameters to make the model better;

Feature	OLS	PLS	PCR	ENet	GLM	RF	XGBoost	NN1	NN2	NN3	NN4	NN5
dp	17.64	0.07	50.84	77.44	6.31	6.12	6.65	20.59	16.93	15.20	14.89	14.12
ep	22.28	0.09	39.98	14.37	0.30	5.08	12.08	18.74	16.14	14.59	14.47	13.86
bm	24.55	0.35	7.67	8.19	28.23	14.67	7.70	13.17	12.80	12.62	13.00	12.89
ntis	11.36	8.97	0.17	0.00	6.42	3.33	4.93	9.47	10.82	11.53	11.55	11.84
tbl	10.77	3.81	0.86	0.00	23.57	44.84	30.21	10.17	11.05	11.75	11.80	11.95
tms	10.13	9.78	0.28	0.00	6.44	4.11	7.96	9.44	10.79	11.45	11.44	11.82
dfy	2.60	21.54	0.16	0.00	7.68	6.08	10.99	9.29	10.75	11.43	11.43	11.77
svar	0.68	55.39	0.03	0.00	21.05	15.78	19.47	9.13	10.72	11.42	11.42	11.75

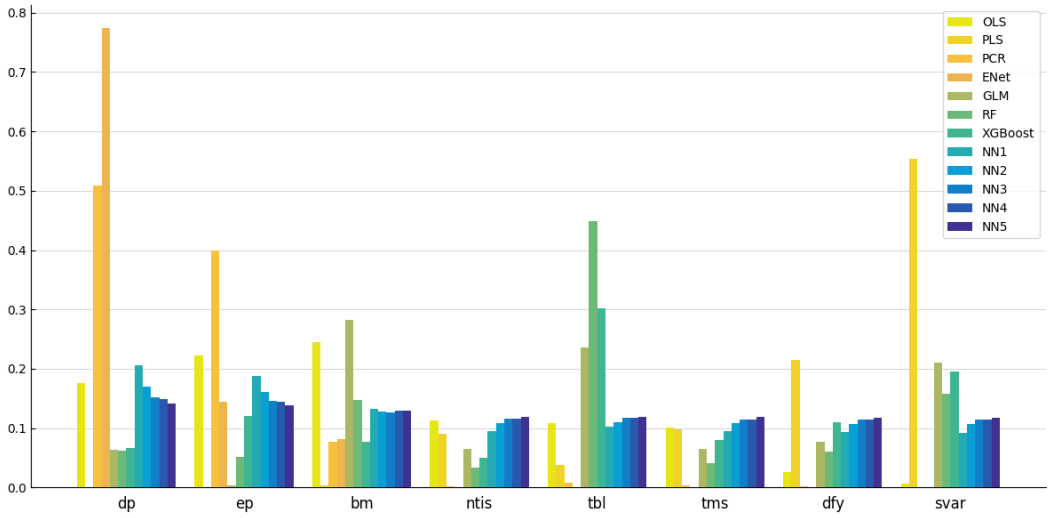


Figure 7: Variable importance for macroeconomic predictors (heatmap)

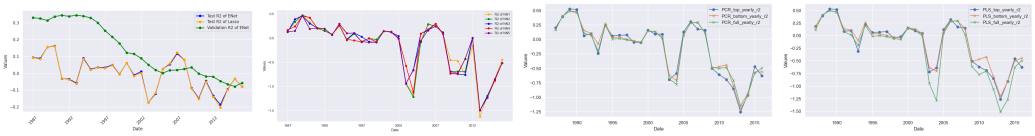
Nevertheless, we do not deny the validity of our own model, we believe that our data preprocessing method is more logical than the original paper, and at the same time, in some model effects, our model effect is even better than the original paper model effect.

4 Reflection

In the process of comparison, we found a lot of interesting phenomena.

1. The R^2 of the test dataset is gradually decreasing, which is shown in almost all models. We suspect that this phenomenon is because as the year progresses, the train dataset becomes larger, which also means that the noise becomes larger, leading to overfitting of the model.

2. Around 2003 and 2012, the R^2 of the test dataset showed a very obvious trough. We suspect this is because these particular years may have experienced significant market changes, economic fluctuations, or policy adjustments that led to market turbulence, and the model underperformed in these years. For example, in 2001, there was an economic recession in the United States, which affected the following years, in 2012, many European countries faced debt crises, which affected the global economy, and the discussion of the fiscal cliff in the United States led to increased market uncertainty 8.



(a) Yearly R^2 of ENet (b) Yearly R^2 of NN (c) Yearly R^2 of PCR (d) Yearly R^2 of PLS

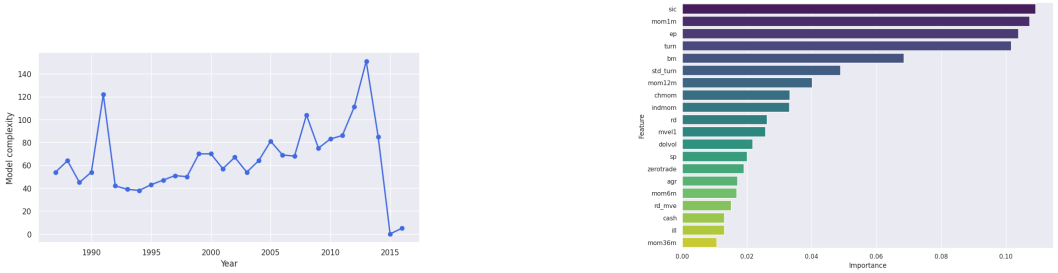
Figure 8: Comparison of Yearly R^2 across models

Appendix

A Model Performance

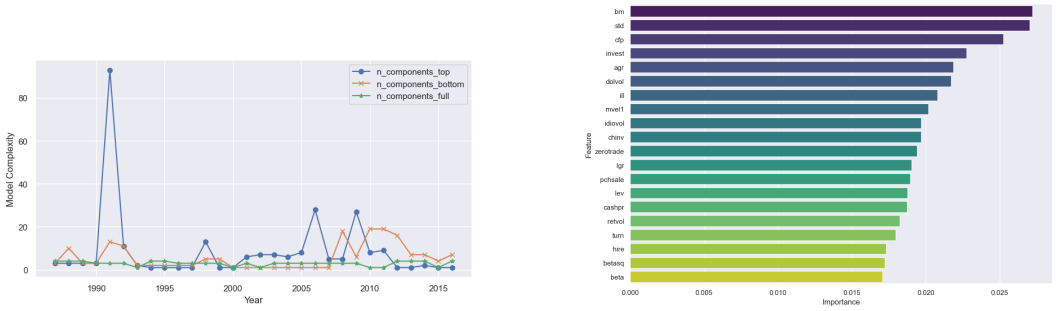
(a) Fullset (b) Topset

Figure 9: Top20 Feature Importance of OLS for fullset and Topset



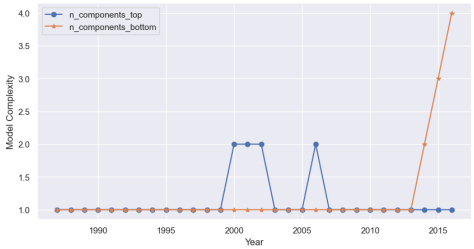
(a) Time-varying model complexity of ENet (b) Top20 Feature Importance of ENet

Figure 10: Performance of ENet

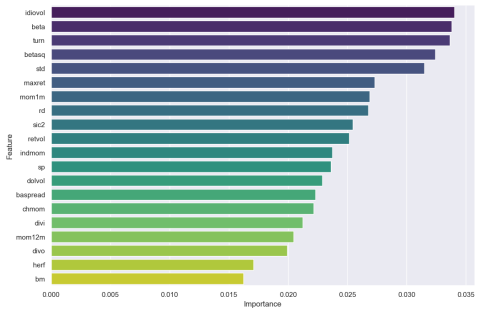


(a) Model complexity of PCR (b) Top20 Feature Importance of PCR

Figure 11: Performance of PCR

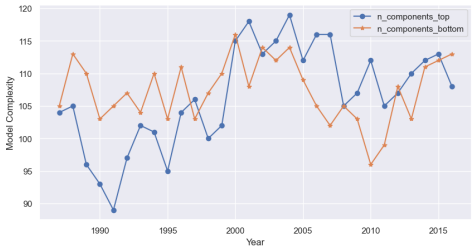


(a) Model complexity of PLS

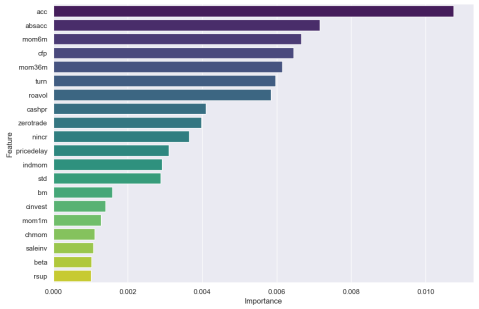


(b) Top20 Feature Importance of PLS

Figure 12: Performance of PLS

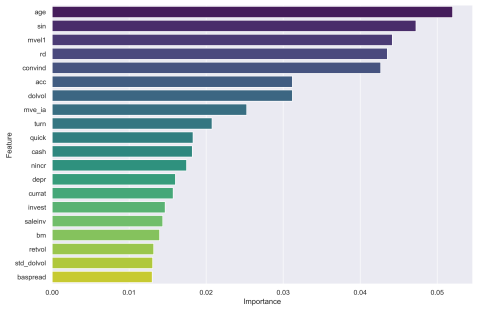


(a) Model complexity of GLM

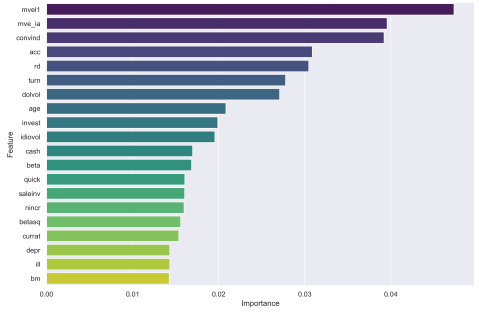


(b) Top20 Feature Importance of GLM

Figure 13: Performance of GLM

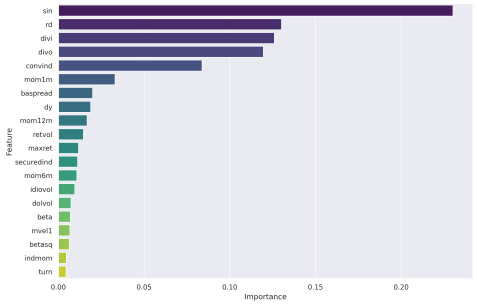


(a) Topset stocks

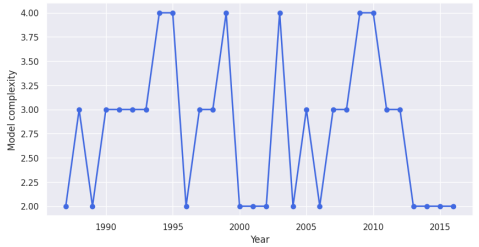


(b) Bottomset stocks

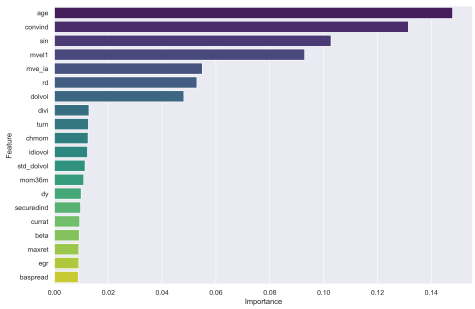
Figure 14: Variable importance for topset and bottomset stocks in RF



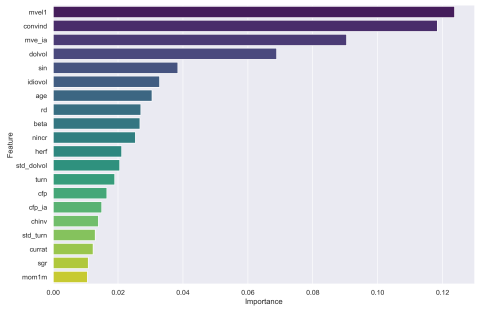
(a) Top20 feature importance



(b) Time-varying model complexity of XGBoost



(a) Topset stocks



(b) Bottomset stocks

Figure 16: Variable importance for topset and bottomset of XGBoost

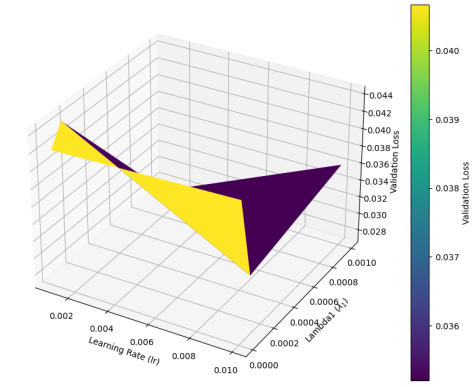


Figure 17: NN1 validation loss for parameter tuning

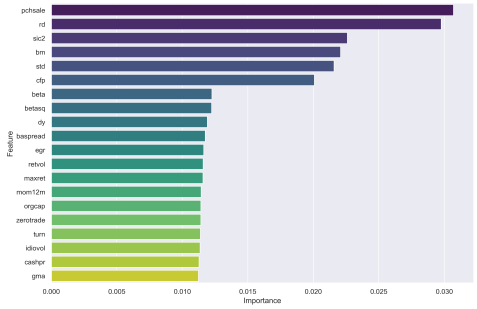
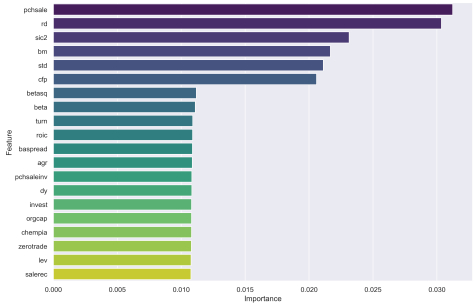
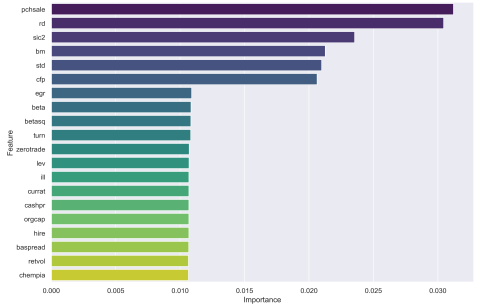


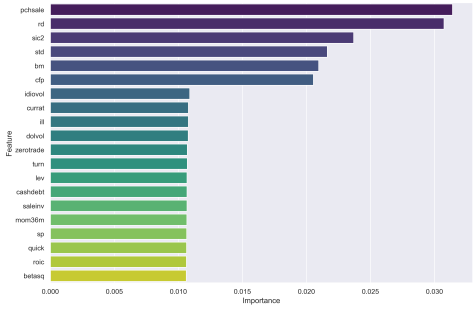
Figure 18: Top 20 Feature Importances of NN1



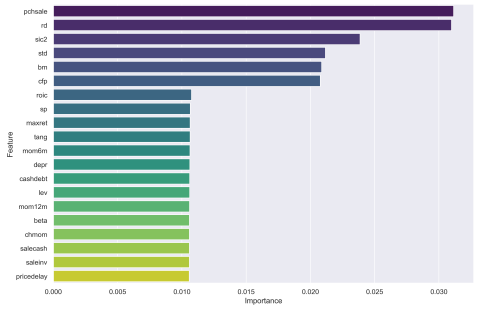
(a) Top 20 Feature Importances of NN2



(b) Top 20 Feature Importances of NN3



(c) Top 20 Feature Importances of NN4



(d) Top 20 Feature Importances of NN5

Figure 19: Feature Importance of NN2-5

B Monthly & yearly results

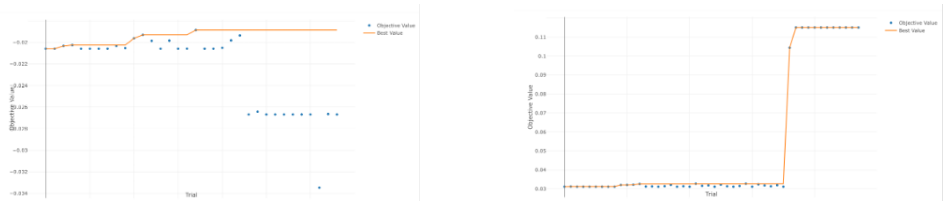
	OLS3	ENet	PCR	PLS	GLM	RF	XGBoost	NN1	NN2	NN3	NN4	NN5
All	-3.57	-0.58	-0.50	-0.50	-0.04	0.39	-7.52	0.11	1.19	3.21	2.47	1.03
Top	-6.81	-12.12	-0.63	-0.63	-0.21	0.46	-3.60	2.80	3.89	7.37	4.10	1.81
Bottom	-5.89	-12.63	-0.56	-0.45	0.15	0.12	7.30	2.11	1.67	4.61	3.25	2.10

Table 2: Yearly R^2(percentage of R^2_{oos})

	OLS3	ENet	PCR	PLS	GLM	RF	XGBoost	NN1	NN2	NN3	NN4	NN5
All	-3.10	-0.04	-1.97	-1.90	-0.83	-8.62	-7.04	-0.17	0.02	0.21	0.16	0.09
Top	-17.38	-28.78	-2.31	-2.36	-0.95	-11.17	-15.41	-0.04	0.11	0.31	0.34	0.45
Bottom	-14.94	-27.88	-2.19	-2.01	-0.86	-10.01	-0.78	0.03	0.54	1.34	0.43	0.37

Table 3: Monthly R^2(percentage of R^2_{oos})

C Parameter Tunning



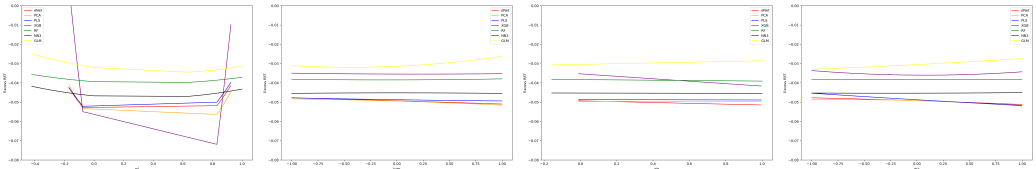
(a) from 1957 to 1987 (b) from 1957 to 1989

Figure 20: Hyperparameter Tuning Examples of ENet

Hyperparameter	**RF**	**XGBoost**
`max_depth`	[1, 2, 3, 4, 5, 6]	[2, 3, 4]
`max_features`	[3, 5, 10, 20]	-
`n_estimators`	300	[50, 200, 300]
`learning_rate`	-	[0.01, 0.1]

Table 4: Hyperparameter settings for RF and XGBoost

D Marginal Effect & Macro Features Effect



(a) rd (b) turn (c) sin (d) acc

Figure 21: Marginal Association between Return and Features

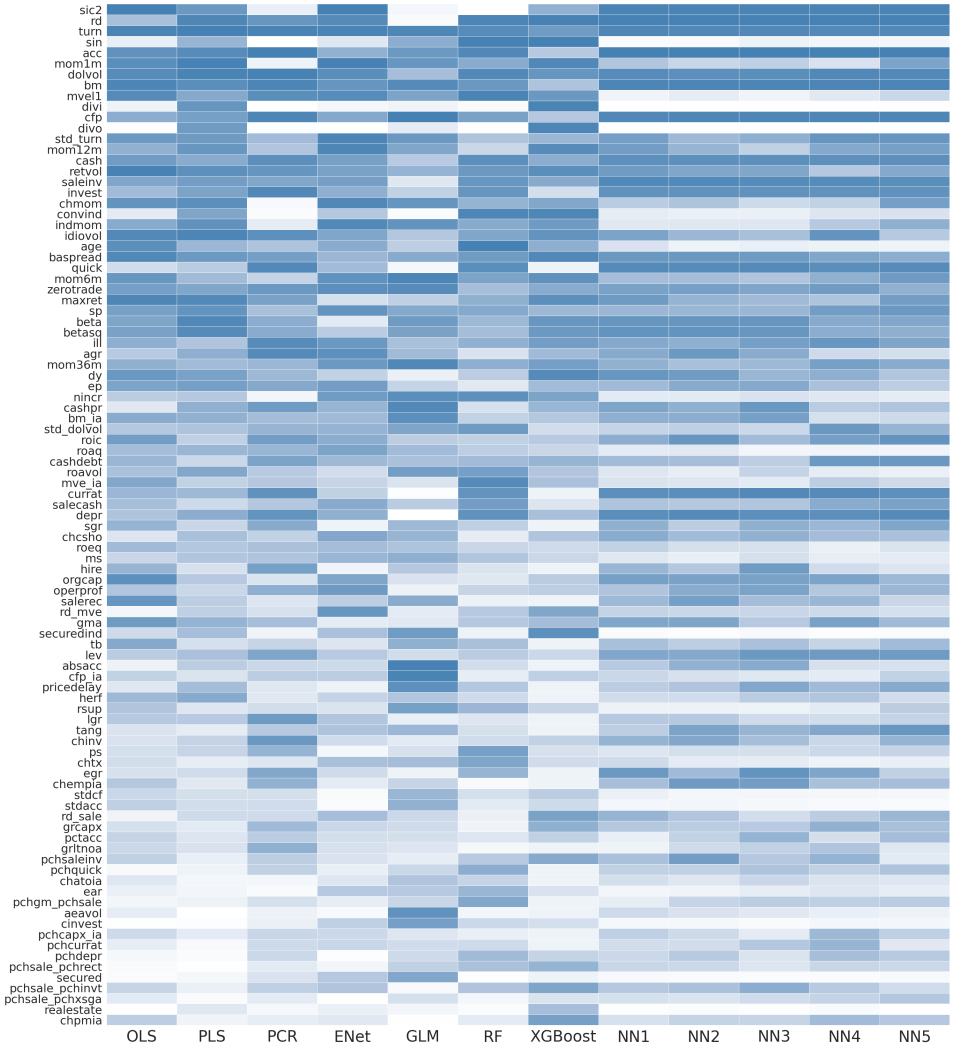


Figure 22: Characteristic Importance