Empirical Asset Pricing via Machine Learning

He Weiwei

wheal@connect.ust.hk

Li Xintong

xlifw@connect.ust.hk

Ma Jingkun

jmabg@connect.ust.hk

Xu Tongcan

txuav@connect.ust.hk

Abstract

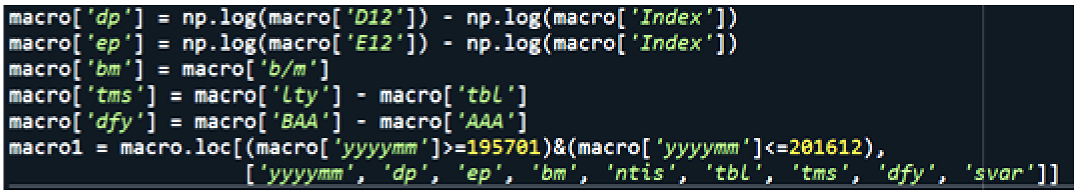
We replicate part of the content of the Empirical Asset Pricing via Machine Learning by Dr. Xiu Dacheng of Chicago Booth. We chose 6 models to see their performance in empirical asset pricing which are: OLS,ElasticNet, Random Forest, Principal component Regression Gradient Boosting and Neural Networks.

1. Data preprocessing

First, we download 94 firm characteristics data and process them. We believe that if all the features are selected, which will consume a large amount of memory and affect the running speed of the program, and are not conducive to the establishment of the model. At the same time, we found that there are a lot of null values in some firm characteristics, and we have reason to believe that these characteristics are not very practical in the model building process. In addition, too many missing values may cause significant deviations in the results.

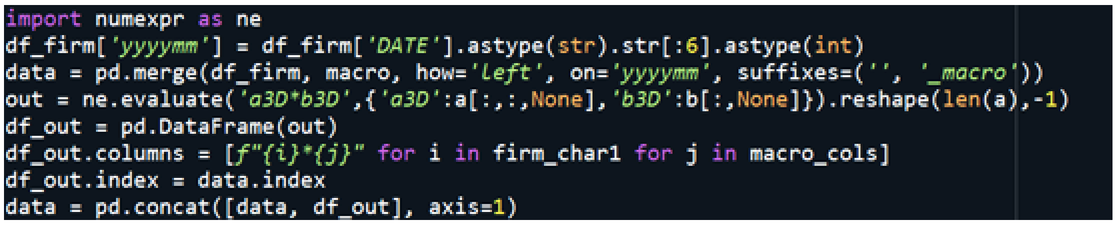
By constraining the period of the data (1957-2016), we obtained a total of 3,762,139 rows of samples. Therefore, we judge that if the missing value of a company feature exceeds 800,000, it is enough to cause a significant deviation in the result. We removed these features, left 19 company features, and used function. fillna( ) to process the remaining data in the actual data cleaning process.

Next, we obtained the macro data. We also processed the overall data according to the definition detailed in Welch and Goyal (2008) and finally constructed eight macroeconomic predictors.



```
macro['dp'] = np.log(macro['D12']) - np.log(macro['Index'])
macro['ep'] = np.log(macro['E12']) - np.log(macro['Index'])
macro['bm'] = macro['b/m']
macro['tms'] = macro['lty'] - macro['tbl']
macro['dfy'] = macro['BAA'] - macro['AAA']
macro1 = macro.loc[(macro['yyyymm']>=195701)&(macro['yyyymm']<=201612),
                   ['yyyymm', 'dp', 'ep', 'bm', 'ntis', 'tbl', 'tms', 'dfy', 'svar']]
```

Finally, we calculate the interaction between stock level characteristics and macroeconomic state variables based on the description in the paper. We multiply the remaining firm feature with the macro predictors corresponding to the time. Note that here we first connect the two data packets through the year and month characteristics to ensure one-to-one data correspondence. The total number of covariates is 19× (8+1) =171.



```
import numexpr as ne
df_firm['yyyymm'] = df_firm['DATE'].astype(str).str[:6].astype(int)
data = pd.merge(df_firm, macro, how='left', on='yyyymm', suffixes=('', '_macro'))
out = ne.evaluate('a3D*b3D',{'a3D':a[:,:,None],'b3D':b[:,None]}).reshape(len(a),-1)
df_out = pd.DataFrame(out)
df_out.columns = [f"{i}*{j}" for i in firm_char1 for j in macro_cols]
df_out.index = data.index
data = pd.concat([data, df_out], axis=1)
```

In addition, we found that in the firm data,'sic2' represents the firm's industry dummies, which means that we can use get_dummies() to convert them during the specific model building process.

2. Model Performance

2.1 OLS and ElasticNet

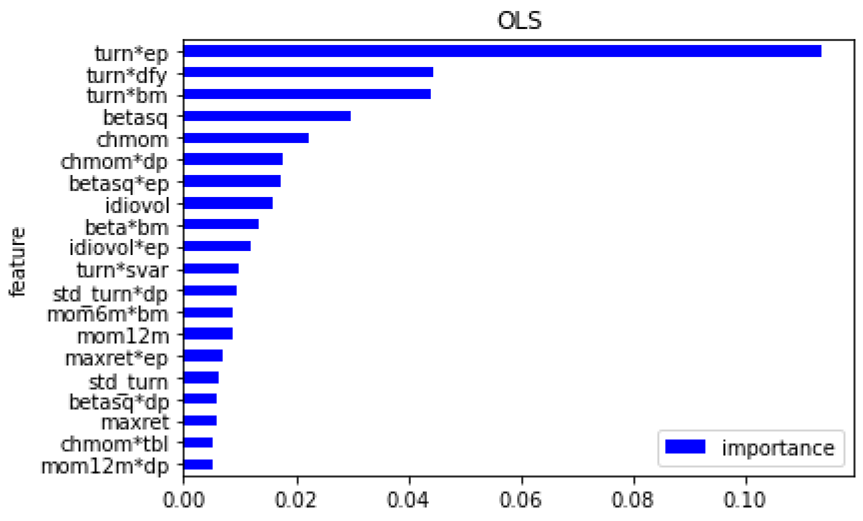
We use a simple linear predictive regression model: ordinary least squares (OLS) and elastic net as the starting point for learning more sophisticated methods.

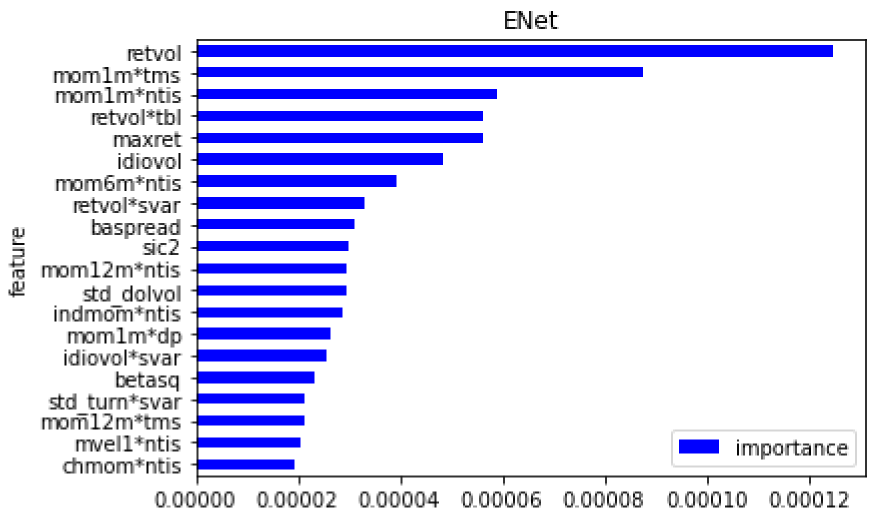
In the actual process, we use LinearRegression and ElasticNet. The two models are similar so we use it in the same way. We use the for loop to achieve recursive prediction and get R^2 using r2_score for each loop and finally we calculate the average score. Based on different size of company ,we choose top 1000 size of company, bottom 1000 size of company and all the company to train the model. For OLS,the result is -0.0546, -0.8976, -0.0239 and for elastic net the result is -0.0311, -0.0413, -0.0057.

```python
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
lr = LinearRegression()
lr.fit(x_train_std, y_train.astype('int'))
y_pred1 = lr.predict(x_test_std)
print('OLS accuracy:', r2_score(y_test,y_pred1))
PLS_score.append(r2_score(y_test,y_pred1))

from sklearn.linear_model import ElasticNet
ENreg = ElasticNet(alpha=1, l1_ratio=0)
ENreg.fit(x_train_std, y_train.astype('int'))
y_pred2 = ENreg.predict(x_test_std)
print('ElasticNet accuracy:', r2_score(y_test,y_pred2))
ENet_score.append(r2_score(y_test,y_pred2))
```

For the characteric importance of OLS and elastic net, we calculate the correlation coefficient of each feature.



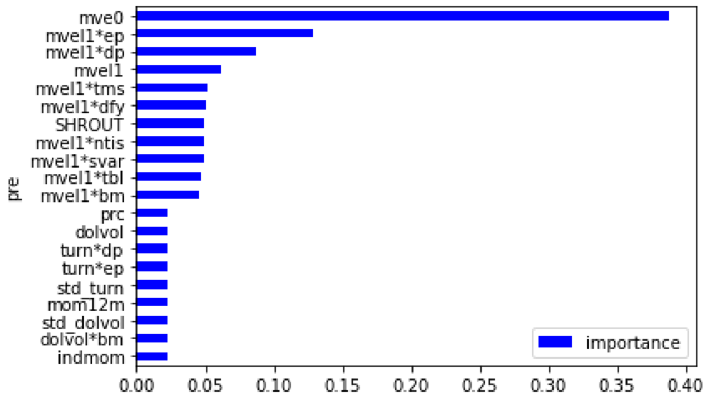


2.2 PCR

PCR is the combination of PCA and regression. The first step is doing CPA to convert the original variable to the main components. We convert all the variables to 10 components.

All the information can be compressed into these 10 components and then I do the regression to see the in-sample R^2 which is around 0.12. Then I do the recursive procedure to do the validation part and test part. In addition, we still choose different dataset to train the model and see the out-of-sample of R^2. Based on different size of company ,we choose top 1000 size of company, bottom 1000 size of company and all the company to train the model and the result is 0.1402, 0.1463, 0.2191.

The variable importance is below.



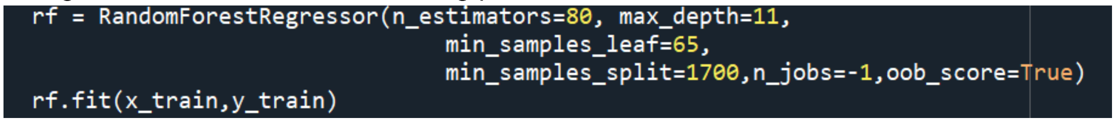
2.3 RF and GBRT

As two similar modeling methods, RF and GBRT are composed of multiple trees and determine the final result. But there are also differences between them. For example, RF is not sensitive to outliers, while GBRT is sensitive to outliers. For the result, RF uses majority voting, while GBRT uses accumulation or weighted accumulation. At the same time, RF improves performance by reducing model variance, while GBRT enhances performance by reducing model deviation.

In the actual process, we use RandomForestRegressor and GradientBoostingRegressor in scikit-learn to model the data.

With regard to these two models, we use a similar approach.

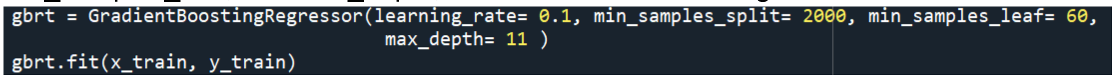
For RF, we use the validation sample to adjust the model parameters, mainly adjusting the parameters n_estimators, max_depth, min_samples_leaf, and min_samples_split. Through tuning, we have selected the following parameters.



```
rf = RandomForestRegressor(n_estimators=80, max_depth=11,
                           min_samples_leaf=65,
                           min_samples_split=1700,n_jobs=-1,oob_score=True)
rf.fit(x_train,y_train)
```

During the final operation of the model, we use the for loop to achieve recursive prediction and use r2_score () to record the R^2 under each loop, and finally take the average. Through each cycle, the training set is increased by one year from the earliest 18-year sample, and the validation sample remains unchanged for 12 years and then predicts the result of the following year. At the same time, similar to other models, we calculate the R^2 value of top size company and bottom size company separately at same time. In the end, we get R^2 as 0.0734, 0.0949, 0.0680 respectively.

For GBRT, we first adjusted the parameters learning rate, min_samples_split, min_samples_leaf, and max_depth, and selected the following results.



```
gbrt = GradientBoostingRegressor(learning_rate= 0.1, min_samples_split= 2000, min_samples_leaf= 60,
                                 max_depth= 11 )
gbrt.fit(x_train, y_train)
```

Similarly, we use the for loop to finally obtain the following result. The R^2 of all companies, top-size1000 company, and bottom-size1000 company are 0.0551, 0.0658 and 0.0125 respectively.

2.4 Neural Networks

According to the paper, there are 5 neural networks to be built: 1) shallowest neural network has a single hidden layer of 32 neurons, which we denoted NN1. 2) NN2 has two hidden layers with 32 and 16 neurons, respectively; 3) NN3 has three hidden layers with 32, 16, and 8 neurons, respectively; 4) NN4 has four hidden layers with 32, 16, 8, and 4 neurons, respectively; 5) NN5 has five hidden layers with 32, 16, 8, 4, and 2 neurons, respectively.

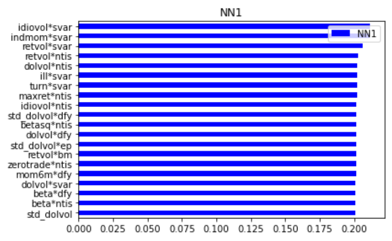
We test the R^2 result on the validation set to see which optimizer('lbfgs','sgd','adam') to choose for the final forecast. According to the result shown in the table below, 'sgd' gives us the largest R^2. So we chose 'sgd' which is stochastic gradient descent for the test set.

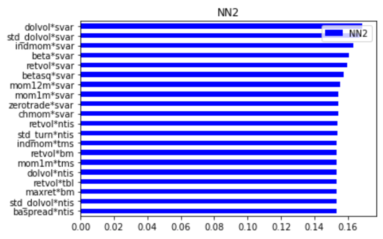
nn1			nn2			nn3			nn4			nn5		
lbfgs	sgd	adam	lbfgs	sgd	adam	lbfgs	sgd	adam	lbfgs	sgd	adam	lbfgs	sgd	adam
-0.272	-0.362	-1.182	-0.169	-0.07	-0.523	-0.104	-0.037	-0.715	-0.123	-0.07	-0.145	-0.228	-0.017	-0.627
-0.812	-0.414	-4.663	-1.174	-0.292	-1.267	-0.774	-0.047	-0.662	-0.746	-0.005	-0.121	-0.005	-0.024	-0.787
-0.648	-0.634	-2.628	-3.423	-0.905	-1.534	-1.488	-0.021	-1.322	-0.074	-0.116	-0.331	-0.033	-0.104	-0.684
-0.78	-0.429	-7.821	-2.345	-0.281	-1.41	-3.777	-0.094	-0.485	-2.235	-0.044	-0.446	-0.103	-0.07	

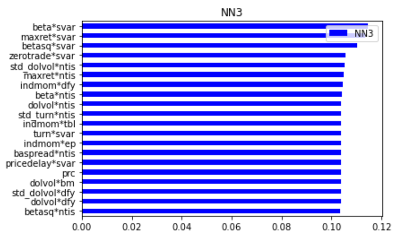
For neural network model, the out sample R2 for each kind of model in all stock and top-1,000-market size stocks and bottom-1,000-market size stocks respectively is shown below.

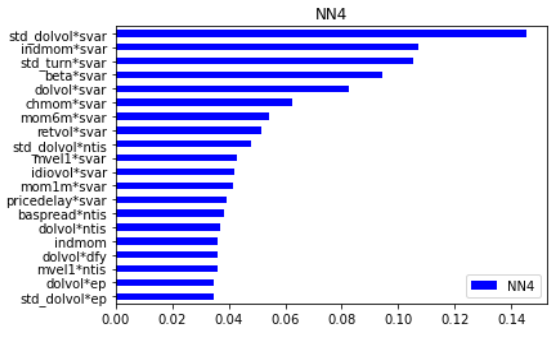
oos R^2	All	Top	Button
nn1	-2.308	-9.866	-1.020
nn2	-2.378	-5.666	-2.917
nn3	-2.438	-1.923	-1.492
nn4	-1.144	-2.687	-0.869
nn5	-0.002	-0.773	-0.221

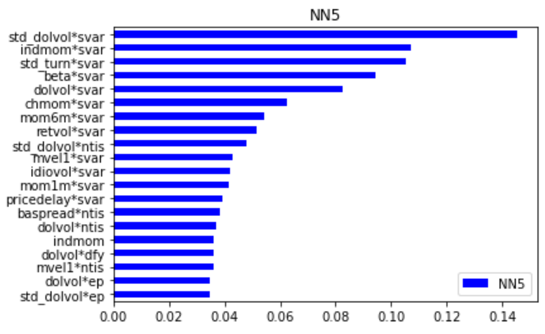
For the feature importance of the neural network model, we calculate the R^2 for NN model while setting the values of variable values to 0 except 1 variable to calculate the feature importance of the variable.





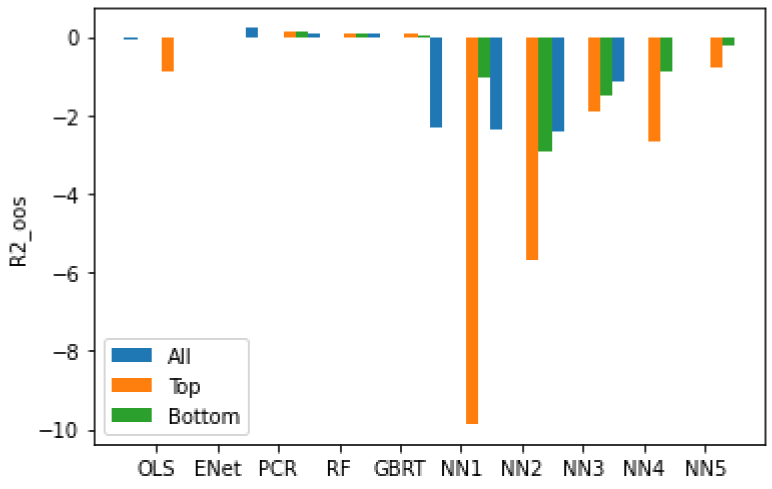


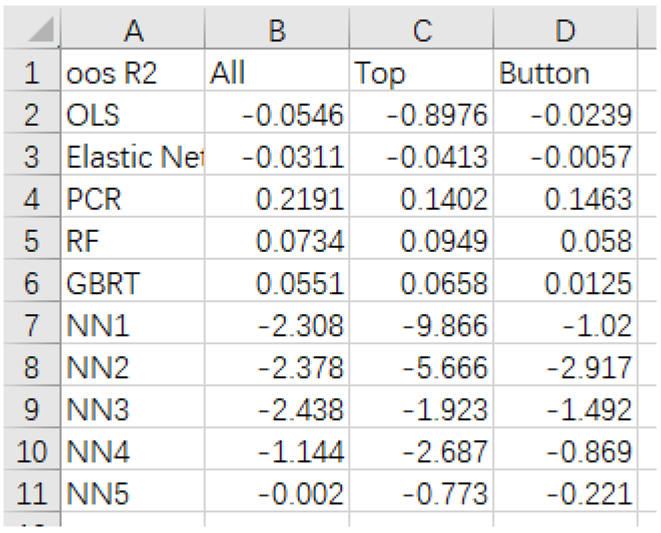




3. Conclusion and Contribution

Finally we combined all R^2 together for 6 models. It seems that PCR performs better than other models.





	A	B	C	D
1	oos R2	All	Top	Button
2	OLS	-0.0546	-0.8976	-0.0239
3	Elastic Net	-0.0311	-0.0413	-0.0057
4	PCR	0.2191	0.1402	0.1463
5	RF	0.0734	0.0949	0.058
6	GBRT	0.0551	0.0658	0.0125
7	NN1	-2.308	-9.866	-1.02
8	NN2	-2.378	-5.666	-2.917
9	NN3	-2.438	-1.923	-1.492
10	NN4	-1.144	-2.687	-0.869
11	NN5	-0.002	-0.773	-0.221

The contribution of each person is as follows.

Name	Contribution
Li Xintong	Neural Network: model and report
Ma Jingkun	PCR: model and report
He Weiwei	OLS and ElasticNet:model and report
Xu TongCan	RF and GBRT:model and report ; data preprocessing

4. Reference

[1] Empirical Asset Pricing via Machine Learning,Shihao Gu, Bryan Kelly, Dacheng Xiu
The Review of Financial Studies, Volume 33, Issue 5, May 2020, Pages 2223–2273