# How can machine learning help stock investment?

Xin Guo
Email: guoxin@stanford.edu

## 1 Introduction

The million-dollar question for stock investors is if the price of a stock will rise or not. The fluctuation of stock market is violent and there are many complicated financial indicators. Only people with extensive experience and knowledge can understand the meaning of the indicators, use them to make good prediction to get fortune. Most of other people can only rely on lucky to earn money from stock trading. Machine learning is an opportunity for ordinary people to gain steady fortune from stock market and also can help experts to dig out the most informative indicators and make better prediction.

The purpose of the present project is to investigate the modeling of stock price movement trend and build up models to predict if the close price of a stock rises or falls on the next trading day. The problem belongs to a classification problem.

The input to my algorithm includes: 1. moving average of historical close prices; 2. trading volume, and open, highest, lowest, and close prices of the present trading day; 3. financial indicators, e.g., DecisionPoint Price Momentum Oscillator (PMO), Money Flow Index (MFI), Percentage Price Oscillator (PPO), and et al. [3]; and 4. self-developed price movement trend based on local Taylor expansion and spline fitting. Totally, there are 253 features in the initial feature bag. I then trained logistical regression, support vector machine (SVM), and Random Forest models to predict the close price rises or falls on the next trading day, e.g., 1 means stock price will rise on the next trading day, $-1$ means stock price will fall.

In this project, I used Python (e.g., SciPy [2]) and MongoDB [1] for calculation and data storage.

## 2 Related Work

Stock prediction is a complicated and challenging problem. Most researchers focus on stock selection problem and the prediction of stock return.

Refenes et al. [9] applied neural networks to predict stock performance. They found that even simple neural learning procedures showed better prediction accuracy than classic statistical techniques, e.g., multiple linear regression. They also claimed that with careful network design, model performance can be further improved.

Levin [7] designed a multilayer feedforward neural networks to select stocks. He showed that his model can make good prediction even if data is contaminated by large ratio of noise.

Ghosn and Bengio [5] also investigated artificial neural networks to predict future returns of stocks. With a serials of experiments, they concluded that artificial neural networks have the best performance, when the neural networks for different stocks do not share any parameter or only share some parameters. In another word, to get the best prediction, one always needs to train model specifically for each stock and there is no universal model for all the stocks with the best performance.

Tsai et al. [10] examined the performance of classifier ensembles on the prediction of stock return and made comparison with single classifiers, i.e., neural networks, decision trees, and logistic regression. They studied the impact of different types of classifier ensembles and majority voting and bagging. They concluded that in general, classifier ensembles perform better than single classifier.

Leung et al. [6] compared the forecasting performance of classification models to predict the direction of index return and level estimation models to predict the value of the return. They concluded that classification models always perform better than level estimation models. They also showed that the forecasting from classification model can be used to develop trading strategies for more trading profits.

## 3 Dataset and Features

The data for the present project was downloaded from `quandl.com` through API. The initial data includes daily stock open, highest, lowest, and close prices, volume, dividend, and split ratio. I chose to focus on 4 stocks with the longest data history, i.e., American Airline (AA), General Electric (GE), Hewlett-Packard Company (HPQ), and International Business Machines Corporation (IBM). Each of the stocks has 13534-day data from January 2, 1962 to October 21, 2015. For cross validation purpose, I used the first 70% samples as training data set, and the last 30% samples as validation data set.

The following steps were performed for data cleaning and normalization:

1. I first cleaned the data to remove the entry with incomplete or invalid information. For example, some entries have 0 volume meaning that on that day, there is no trading at all. Since the entries with incomplete/invalid information are few, the cleaning should not have impact on the modeling. The data was saved into a MongoDB, i.e., each stock as a collection and the trading information on one single day as a document.
2. I developed a Python code to calculate a serial of technical indicators, e.g., accumulation/distribution line, Aroon, Average directional index, BandWidth, %B indicator, and et al. [3] The moving average of historical prices and some self-developed indicators are also calculated. In total, there are 253 features in the feature bag.
3. Since the magnitudes of features range from $O(0.1)$ to $O(10^5)$, to avoid the model is dominant by large magnitude features, all the features are normalized with their mean and standard deviation as the following formula,

$$f = \frac{f - mean\ value\ of\ f}{standard\ deviation\ of\ f}. \quad (1)$$

| AA | GE | HPQ | IBM |
|---|---|---|---|
| 0.0084 | 0.0033 | 0.0079 | 0.0080 |

**Table 1:** Price-difference boundary from K-means classification.

Since my objective is to predict price rises or falls, to label data samples, I first calculated price difference by subtracting the present close price from the next trading day close price. I labeled my data with two methods.

1. (Movement trend 1): The first method is to label a sample as 1 if price difference is positive, i.e., the price raises on the next trading day and $-1$ if price difference is negative, i.e., the price falls.
2. (Movement trend 2): The second method is to use K-means to classify the price difference into two groups and label the group with large price difference as 1 and the other group as $-1$.

Here, the second method with K-means model is equivalent to find a new price-difference boundary to label the data, whereas the price-difference boundary in the first method is 0. Table 1 lists the price-difference boundary of the second method.

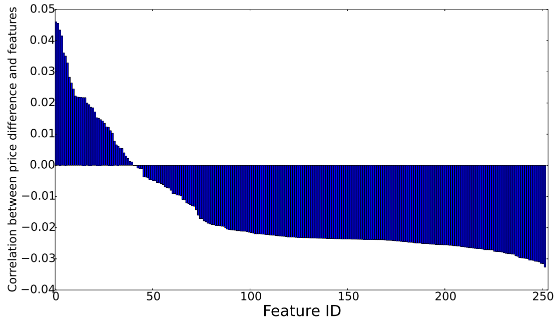

**Figure 1:** Correlation between features and price difference of AA.

To understand the data, I checked the correlation between the features and price difference. Figure 1 shows this correlation of AA as an example. In general, there is no obvious strong correlation between features and price difference. For the result shown here, the correlation ranges from $-0.03$ to 0.04. Most of the correlation is negative and their magnitude is about 0.02. The correlation between the features and price difference of the rest stocks showed the similar variation and range.

# 4 Model Quality Assessment Method

For two-class classification problem, based on confusion matrix, i.e., the number of true positive (TP), true negative (TN), false positive (FP), and false negative (FN), many metrics are defined to assess the performance of a model. However, most of metrics are not reliable. For example, accuracy will yield misleading results if the number of samples in different classes are quite different; F1 scores only considers TP but no TN.

In the present project, I used Mattews correlation coefficient (MCC) to assess the performance of models [8]. MCC is defined as

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}. \tag{2}$$

As shown, MCC takes into account all the components in confusion matrix and is a general balance measurement regardless the sample number variation of different classes. MCC can be considered as a "correlation" between predicted value and true value, i.e.,

1. $MCC = 1$ means all predictions are right.
2. $MCC = 0$ means model prediction is no better than random prediction.
3. $MCC = -1$ means no prediction is right.

Therefore, the objective of the project is to find the models with the highest MCC value.

## 5 Models

In this project, I applied logistical regression, SVM, and random forest models. Due to the space limitation, I briefly introduce them below.

1. Logistical regression is a linear model for classification. The hypothese is written as

   $$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}. \tag{3}$$

   To improve the performance of the model, two cost functions are considered, i.e., L2 penalized cost function:

   $$\min_{\theta, C} \frac{1}{2}\theta^T\theta + C\Sigma_{i=1}^n \log[\exp(-y_i\theta^T x_i)+1], \tag{4}$$

   and L1 penalized cost function:

   $$\min_{\theta, C} \|\theta\|_1 + C\Sigma_{i=1}^n \log[\exp(-y_i\theta^T x_i)+1]. \tag{5}$$

2. For SVM model, since the sample set is not linearly separable, the primal problem is:

   $$\min_{w,b,\zeta} \frac{1}{2}w^T w + C\Sigma_{i=1}^n \zeta_i, \tag{6}$$

   subject to $y_i(w^T\phi(x_i)+b \geq 1-\zeta_i$, $\zeta_i \geq 0, i = 1, ..., n$. Its dual is

   $$\max_\alpha \Sigma_i\alpha_i - \frac{1}{2}\Sigma_i\Sigma_j y_i y_j K(x_i, x_j), \tag{7}$$

   subject to $y^T\alpha = 0$, $0 \leq \alpha_i \leq C, i = 1, ..., n$. Here, $K(x_i, x_j) = \phi(x_i)^T\phi(x_j)$. I considered the following kernel functions:

   (a) Linear: $< x, x' >$

   (b) Polynomial: $(\gamma < x, x' > +r)^d$

   (c) RBF: $\exp(-\gamma|x-x'|^2$

   (d) Sigmoid: $\tanh(\gamma < x, x' > +r)$

3. Random forest classification model is an ensemble learning method for classification. Random forest model is to train several decision trees with a random subset of features (feature bagging) and a random sample with replacement of training sets [4]. The prediction of random forest models takes the average of all the decision tree prediction or the majority vote of all the decision trees in the model. Because random forest model takes the average result, it decreases the variance in decision trees prediction.

## 6 Feature Selection

As introduced in Sec. 3, there are 253 features in the initial feature bag. I wanted to know if all the features are important and if I can use only subset of features without losing prediction accuracy. In the present project, I applied two feature selection techniques, i.e., random forest feature selection and forward search.

### 6.1 Random forest feature selection

Random forest model can provide ranking scores for features. The larger the ranking score is, the more important the feature is. To obtain the ranking sore of a feature, one needs to: first, obtain the average value of out-of-bag error; second, permute the feature among the training data and obtain a second out-of-bag error; and the ranking score is proportional to the difference between the two out-of-bag errors.

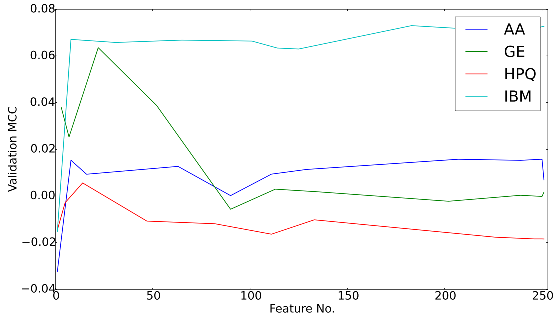

**Figure 2:** Variation of validation MCC values when features are added gradually in the order of feature ranking from random forest model.

To check if all the features are important, I gradually added features into feature bags according to their ranking scores (important feature is added first), trained the model, and obtained the MCC value of the validation sample set. Figure 2 plots the variation of validation MCC value when less important features are gradually added to the feature bag. It shows that in general, the validation MCC value first increases as the feature number increases; with some magic feature combination, the validation MCC value reaches its maximum; and then the validation MCC value decreases as the feature number increases. Comparison among the results of all the stocks shows that when the validation MCC value reaches its maximum, the feature combination is different for different stocks. For example, GE and HPQ only needs less than 50 features to reach the maximum validation MCC value, whereas AA and IBM needs much more than that. Based on this result, I chose the feature combination with the best validation MCC value to further train and finetune models. The feature number of the best candidate of each stock is listed in table 2 and table 3 for movement trend 1 and movement trend 2, respectively.

### 6.2 Forward search

Forward search is a feature selection algorithm to reduce the number of features. The search procedure of the forward search is introduced in lecture note and is not repeated here.

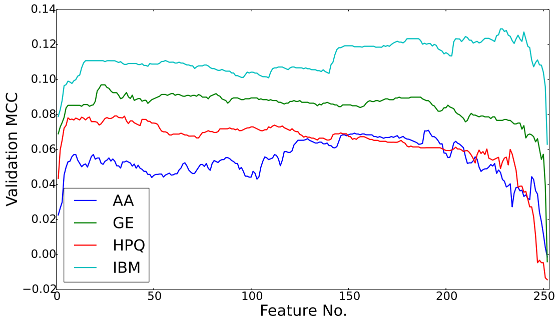

**Figure 3:** History of the best validation MCC values in forward search with logistical regression.

In the present project, I applied forward search with logistical regression. Figure 3 plots the history of the best validation MCC values in the forward search. In general, as the feature number increases, the best validation MCC value increases dramatically first, varies slowly and reaches a maximum value in the middle, and decreases rapidly when the feature bag is almost full. Similar to the random forest feature selection result shown in Sec. 6.1, the validation MCC value of different stocks reaches its maximum with different feature combinations. For each stock, I chose the feature combination with the best validation MCC value to further train and finetune models. The feature number of the best candidate of each stock is listed in table 2 and table 3 for movement trend 1 and movement trend 2, respectively.

## 7 Model Training and Result Discussion

In this section, I explained my procedure to train model and discussed the prediction result.

### 7.1 Grid search and model training

When I trained models, I applied grid search by searching a parameter space to find the model with the best performance. For example, for logistical regression, as introduced in Sec. 5, there are two types of cost functions and a parameter $C$. I trained model candidates on two grids: one is with L1 penalized cost function and $C$ values in $[0.001, 0.01, 0.1, 1, 10, 100]$ and the second one is with L2 penalized cost function and $C$ values in $[0.001, 0.01, 0.1, 1, 10, 100]$. After the best model candidate is found, I kept the cost function unchanged and search $C$ in a smaller range around the $C$ value of the best candidate; and repeated this process until the validation MCC value does not change much in the neighborhood of the current best $C$ value. Similar search procedure is also applied to SVM and random forest.

More specifically, for each stock, I first used all the features and trained logistical regression model with L2 penalized cost function and $C = 1$. This model is considered as a baseline model. Then I applied grid search technique to find the model with the best performance. The best model candidates by using all the features, features selected by random forest, and features selected by forward search are obtained.

### 7.2 Result and discussion

Table 2 and table 3 list the result of the baseline model and the best model for the sample set labeled by movement trend 1 and 2 (defined in Sec. 3), respectively. The best models trained with all the features, feature selected by random forest, and feature selected by logistical regression forward search are listed.

Comparing to the baseline model, model performance has been improved significantly with the help of grid search and feature selection techniques. For

| | baseline | All features | | RF feature selection | | | Forward search (LR) | |
|---|---|---|---|---|---|---|---|---|
| | Val. MCC | Model | Best Val. MCC | Model | Feature No. | Best Val. MCC | Feature No. | Best Val. MCC |
| AA | -0.0012 | RF | 0.052 | SVM (RBF) | 129 | 0.019 | 191 | **0.072** |
| GE | -0.0040 | SVM (lin.) | 0.062 | SVM (RBF) | 22 | **0.092** | 23 | 0.091 |
| HPQ | -0.014 | LR | 0.053 | SVM (lin.) | 47 | 0.053 | 30 | **0.071** |
| IBM | 0.063 | LR | 0.10 | SVM (RBF) | 183 | **0.11** | 228 | 0.084 |

**Table 2:** Baseline and best models for the movement trend 1 with feature numbers and best validation MCC.

| | baseline | All features | | RF feature selection | | | Forward search (LR) | |
|---|---|---|---|---|---|---|---|---|
| | Val. MCC | Model | Best Val. MCC | Model | Feature No. | Best Val. MCC | Feature No. | Best Val. MCC |
| AA | 0.0054 | LR | 0.051 | SVM (Sig.) | 43 | 0.079 | 120 | **0.11** |
| GE | 0.063 | LR | 0.13 | LR | 43 | 0.15 | 91 | **0.17** |
| HPQ | 0.15 | LR | 0.15 | LR | 149 | 0.16 | 36 | **0.17** |
| IBM | 0.069 | LR | 0.14 | LR | 85 | 0.10 | 97 | **0.15** |

**Table 3:** Baseline and best models for the movement trend 2 with feature numbers and best validation MCC.

example, for GE in table 2, the validation MCC value increases from $-0.004$ (baseline) to 0.092 (note that negative MCC value means the model prediction is worst than random prediction).

Comparing the baseline model with the best model using all the features of HPQ and IBM in table 2 and all the stocks in table 3, we can see the grid search technique helps to find better model and largely improves model performance. I also found that the best model candidate is much different among the four stocks. In table 3, even if all the best models using all the features are logistical regression, their actual model forms (i.e., cost function and $C$ value) are quite different (the information on detailed model form is omitted here due to space limitation). This result is consistent with previous research that to have the best prediction, one needs to train models for each stock instead of train one model using the data of all stocks [5].

In general, the best models with features selected by random forest and forward search have better performance than those with all the features. That means each stocks may be only sensitive to certain feature combination. The models in table 3 have better performance than those in table 2. Hence, a good data label algorithm is important to the prediction accuracy.

| Metric | Baseline model | Best model |
|---|---|---|
| F1 | 0.345 | 0.399 |
| recall | 0.391 | 0.419 |
| precision | 0.309 | 0.381 |
| accuracy | 0.546 | 0.615 |

**Table 4:** Validation F1 score, recall, precision, and accuracy of the baseline and best model of AA in table 3.

To further show model performance improvement, in table 4, I listed the validation F1 score, recall, precision, and accuracy of the baseline and best models of AA in table 3 as an example (the definitions of those metrics are omitted here, due to space limitation). Grid search and feature selection techniques improve not only MCC value but also other metrics derived from confusion matrix.

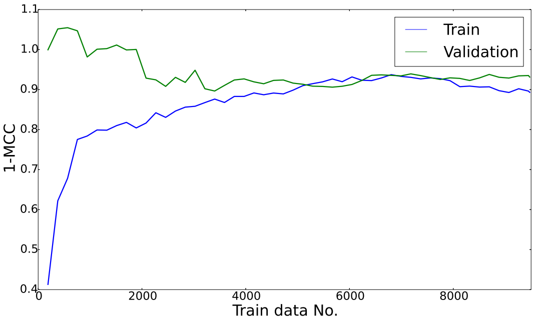

**Figure 4:** Learning curve of the best logistical regression model for IBM.

In figure 4, I showed the learning curve of the best logistical regression model for IBM with samples labeled by movement trend 2. The training and validation error converges to a similar value. If we want to further improve the model performance, we need more informative features.

## 8 Summary and Future Work

In summary,

1. Models with best validation MCC are built up based on current feature set with the help of grid search, random forest feature selection, and forward search techniques. MCC and other metrics are significantly improved.
2. Stock prediction is quite feature and stock dependent. Different feature subsets and different models are best for different stocks.
3. A good classification model to label sample may help to increase prediction accuracy.
4. For more accurate prediction, more features are needed to provide more useful information.

## References

[1] Mongodb. `https://www.mongodb.com`.

[2] Scikit-learn. `http://scikit-learn.org/`.

[3] Technical indicators and overlays. `http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators`.

[4] Breiman, L. Random foests. *Machine Learning 45* (2001), 5–32.

[5] Ghosn, J., and Bengio, Y. Multi-task learning for stock selection. In *NIPS*. 1997.

[6] Leung, M. T., Daouk, H., and Chen, A.-S. Forecasting stock indices: a comparison of classification and level estimation model. *International Journal of Forecasting 16* (2000), 173–190.

[7] Levin, A. U. Stock selection via nonlinear multi-factor models. In *NIPS*. 1996.

[8] Matthews, B. W. Comparison of the predicted and observed secondary structure of T4 phage lsozyme. *Biochimica et Biophysica Acta 405* (1975), 442–451.

[9] Refenes, A. N., Zapranis, A., and Francis, G. Stock performance modeling using neural networks: a comparative study with regression models. *Neural Networks 7* (1994), 375–388.

[10] Tsai, C.-F., Lin, Y.-C., Yen, D. C., and Chen, Y.-M. Predicting stock returns by classifier ensembles. *Applied Soft Computing 11* (2011), 2452–2459.