ST5218

Machine Learning in Stock Trend Prediction

Tian Maoshan	A0129002X
Hu Peiran	A0128954U
Wang Luzhou	A0128979E
Liu Enzhi	A0128944W

Preface

This project is taken by our 4-member group. Each of us is responsible for one particular algorithm. And other common tasks including stock data pre-processing, indicator calculating and report writing are finished together.

- **Tian Maoshan** focuses on Linear Discriminant Analysis.
- **Liu Enzhi** focuses on Quadratic Discriminant Analysis.
- **Hu Peiran** focuses on Logistic Regression.
- **Wang Luzhou** focuses on Support Vector Machine.

Contents

Abstract

Machine learning, an interdisciplinary subject, has been widely used in probability theory, statistics and data mining. It specially study computing simulation of data modelling to achieve learning purpose, therefore human being can acquire more new knowledge or skills and reorganize the existing knowledge structures to continuously improve our performance. Moreover, machine learning is a fundamental way to make computer intelligent, which is also the core of artificial intelligence. The practical application of machine learning is wildly used across all areas of artificial intelligence by induction.

In order to facilitate discussion and progress estimation data, it is necessary to provide the definition of machine learning, even if the definition is incomplete and inadequate. Arthur Samuel gave the definition of machine learning as a "Field of study that gives computers the ability to learn without being explicitly programmed" in 1959 (Phil Simon, 2013).

Machine learning is closely related field to statistic, which focuses on prediction based on known features from the training data. There are four mainly specific algorithm of machine learning, including constructing conditional probability by regression analysis and statistical classification, constructing the probability density function by regenerating model, similarly inference techniques and the optimal method. We focus on four constructing conditional probability algorithm in this report, such as Linear Discriminant Algorithm (LDA), Quadratic Discriminant Algorithm (QDA), Logistic Regression (LR) and Support Vector Machine (SVM).

1. Introduction

1.1 Linear Discriminant Analysis

Linear discriminant analysis (LDA) is a machine learning method used in statistics and pattern recognition, which is a generalization of Fisher's linear discriminant. The purpose of LDA is to find a linear combination of variables that characterizes as two or more groups of objects. The basic idea of LDA is to sample high-dimensional pattern and project it to the optimal discriminant vector space, in order to achieve feature space dimension effect of the classification of extracted and compressed information. After the sample projection, LDA ensures that the model has minimal space distance within each class and maximum space distance between classes in the new sub-sample space, in other words, the model has the best separability in the space. Accordingly, it is an effective feature extraction method. LDA is used when the measurements depend on independent variables with continuous observations.

To be more specific, LDA seeks to reduce dimensionally while preserving as much of the class discriminatory information as possible. Moreover, LDA is closely related to linear regression analysis, and expressed as a linear combination of one categorical dependent variable and other continuous independent variables. In our data, we express one of L1-L50 as categorical dependent variable, which means up or down discrimination of the stock price after one to fifty days, and let y=1 be the up of stock price and y=0 be the down of stock price. Therefore, we can divide out data into two classes, class 1 is the situation when y=1 and the class 2 is when y =0. In addition, we express column open, high, low, close, MA5, MA10, MA20, MA50, NC2, NC3, NC5, NC10, EPS, PE, HH2, HH3, LL2, LL3 as our continuous independent variables in our dataset.

In this case, we get a set of 5-dimensional samples, N_1 of which belong to class 1($y = 1$) and N_2 to class 2(y=0). Now we need to obtain a scalar y by projecting the samples onto a line $y = w^T x$, and we would like to select the line that maximizes the separability of the scalars among all the possible lines as below.

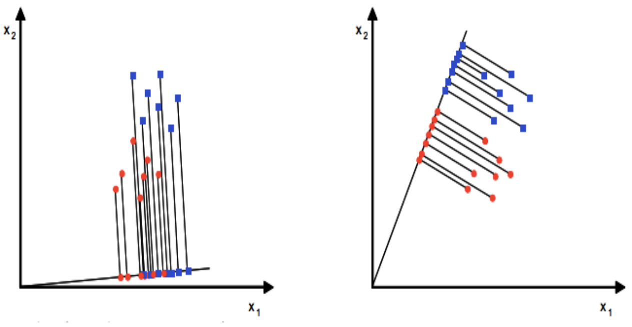

Figure 1 Separability of the Scalars

We need to define a measure of separation, in order to find a good projection vector. Then we seek to find the mean vector of each class in x-space and y-space as

$$\mu_i = \frac{1}{N_i}\sum_{x\in w_i} x$$

and

$$\widetilde{\mu_i} = \frac{1}{N_i}\sum_{y\in w_i} y = \frac{1}{N_i}\sum_{x\in w_i} w^T x = w^T \mu_i$$

In next step, we could choose the distance between the projected means as our objective function

$$J(w) = |\tilde{\mu}_1 - \tilde{\mu}_2| = |w^T(\mu_1 - \mu_2)|$$

But, the distance between projected means is not a good measure of separation, because it does not account for the standard deviation within classes.

Therefore, Fisher suggested maximizing the difference between the means, which is normalized by a measure of the in-class scatter. For each class, we also define the scatter, which is an equivalent of the variance, as

$$\tilde{s}_i^2 = \sum_{y \in w_i} (y - \tilde{\mu}_i)^2$$

In addition, the quantity $(\tilde{s}_1^2 + \tilde{s}_2^2)$ is called the within-class scatter of projected data.

Fisher defines the linear discriminant function $w^T x$ that maximizes the criterion function as

$$J(w) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{s_1^2 + s_2^2}$$

Hence, we are seeking for a projection model where data from the same class are projected very close to each other, and at the same time, the projected means are as farther apart as possible.

To solve this equation, we need to define some measures of the scatter in the feature space x and the original feature space. At first, let the within-class scatter matrix be $S_w = S_1 + S_2$ with

$$s_i = \sum\nolimits_{x \in w_i} (x - \mu_i)(x - \mu_i)^T$$

Thus we can get $\tilde{s}_1^2 + \tilde{s}_2^2 = w^T s_w w$

Next, we define $s_B(\mu_1 + \mu_2)(\mu_1 + \mu_2)^T$

Then we can express $(\tilde{\mu}_1 - \tilde{\mu}_2)^2 = w^T s_B w$

After the complex calculations, we find the optimum w* which satisfies all conditions as above

$$w^* = arg\,mas\,\left[\frac{w^T s_B w}{w^T s_w w}\right] = s_w^{-1}(\mu_1 + \mu_2)$$

Therefore, LDA is also known as Fisher's linear discriminant (1936). It is not a discriminant but rather a specific choice of direction for the projection of the high-dimensional data down to one dimension.

Although LDA is an efficient machine learning to model a statistic analysis, there are also some limitations of LDA method. Firstly, LDA just produces at most c-1 feature projections, where c is the number of class group. If the classification error estimates establish that more features are needed, therefore some other statistic mothers must be employed to provide those additional features. Secondly, LDA should assume unimodal Gaussian likelihoods, which is a parametric method. If the distributions are significantly non-Gaussian, the LDA projections may not preserve complex structure in the dataset need for classification. Finally, LDA will also fail if discriminatory information is not in the mean but in the variance of the data.

1.2 Quadratic Discriminant Analysis

In machine learning field and statistical classification methods, Quadratic Classifier is widely used to separate measurement of more than one classes of targeting objects and affairs with a quadric surface. As most of us know the traditional statistical classification method of linear classifier can hardly be precise when dealing with non-linear problems, which are much more common to meet in the real world. Thus a general version of linear classifier is invented, which is indeed quadratic classifier.

To know quadratic classifier better, let's consider it in a direct and mathematical way. Suppose we have a series of observations of one object or event, denoted by a vector **x**, in which each of the observations has a known type y. And **x** is regarded as the training set. We want to know, if given a new set of observations vector, how we can classify them into appropriate class. In the method of quadratic classifier, y is decided in a form of:

$$\mathbf{x}^{\mathbf{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathbf{T}}\mathbf{x} + c$$

As we can see, each of the observations consist of two kinds of measurement with three parameters. It is not difficult to see that a conic section, which can be either a line, circle, ellipse, parabola or hyperbola, is the surface that separates the classes. This model is more general than linear model and is able to represent a more complex separating surface.

Quadratic Discriminant Analysis, generally speaking, comes from Quadratic Classifier. In QDA, we have normality assumption for the model and the covariance of each class, different from LDA, do not have to be identical. Training data **x** can be expressed as:

$$\mathbf{x} = (\mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3} \dots \mathbf{X}_{\boldsymbol{n}})$$

Where *n* is the time index and at each time:

$$\mathbf{X}_{\boldsymbol{i}} = (d_1^i, d_2^i, d_3^i \dots d_m^i)^{\mathbf{T}}$$

Where *d* denotes the different data sets, or different features of stocks, and *m* is the number of features. Then we define a predictor function based on the data **X**:

$$\mathrm{P}(\mathbf{x}) = \begin{cases} 1, if\ goes\ up \\ 0, otherwise \end{cases}$$

That is, if the stock price is predicted to go up after some pre-determined days, say 20 days, we have P=1, otherwise P=0.

Under normal assumption, given every class *k* follows a Gaussian distribution and *p* is the dimension, we have multivariate Gaussian distribution:

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k{}^{-1}(x-\mu_k)}$$

Where Σ is the covariance matrix. Then we define the quadratic discriminant function to be:

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2} ln|\Sigma_k| - \frac{1}{2}(\boldsymbol{x} - \mu_k)^T \Sigma_k{}^{-1}(\boldsymbol{x} - \mu_k) + ln\,\pi_k$$

Where π is the prior probability. Our classification rule is chosen as follows, based on Bayesian probability theorem which says that we should pick a class that has the maximum posterior probability given the feature matrix **x**:

$$\hat{P}(\boldsymbol{x}) = arg\max_k \delta_k(\boldsymbol{x})$$

In other words, we need to find the class *k* such that the quadratic discriminant function is maximized. The decision boundaries are quadratic equations in **x**, as we defined in the quadratic classifier part.

1.3 Logistic Regression

Logistic regression is a special case of generalized linear model, a type of probabilistic statistical classification model, used for predicting the outcome of a categorical dependent variable based on one or more predictor variables. Logistic regression is focusing to measure the relationship between the dependent and independent variables by using probability scores as predicted values of dependent variable.

Generally, logistic regression is very frequently used to refer specifically to the problem in which the dependent variable is binary. In finance area, logistic regression can be applied to problems of classification predicting and estimation.

Definition of model

The main idea of logistic regression is the logistic function, or sigmoid function. For any negative or positive number input for this function, an output value would be between one and zero hence interpretable as a probability of dependent variable in model. The logistic function is as follow:

$$g(z) = \frac{e^z}{e^z + 1} = \frac{1}{e^{-z} + 1}$$

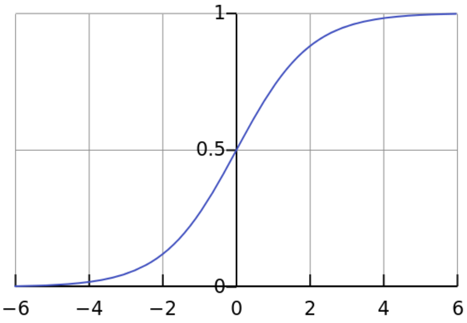

Figure 2 logistic function

In the logistic there is z as our decision bound, which can be defined as:

$$\theta_0 + \theta_1 x_1 +, \dots, +\theta_n x_n = \sum \theta_i x_i = \theta^T x$$

The decision bound is a sorter in the model separating the space into two areas, implied different value of dependent variable. And the predictor function as:

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

Where $h_\theta(x)$ can take 1 or 0 whose probabilities are as below:

$$P(y = 1|x;\theta) = h_\theta(x)$$

$$P(y = 0|x;\theta) = 1 - h_\theta(x)$$

Model fitting and predicting

In model fitting applied maximum likelihood estimation by establishing cost function:

$$Cost(h_\theta(x), y) = \begin{cases} -log\left(h_\theta(x)\right) & if\ y = 1 \\ -log\left(1 - h_\theta(x)\right) & if\ y = 0 \end{cases}$$

$$J(\theta) = \frac{1}{m}\sum Cost(h_\theta(x_i), y_i)$$

$$= -\frac{1}{m}\Big[\sum y_i\, log\left(h_\theta(x_i)\right)$$

$$+ (1 - y_i) log(1 - h_\theta(x_i))\Big]$$

According to predictor function we can derive the likelihood function: (and in log)

$$P(y|x;\theta) = (h_\theta(x))^y (1 - h_\theta(x))^{1-y}$$

$$L(\theta) = \prod P(y_i|x_i;\theta)$$

$$= \prod (h_\theta(x))^y (1 - h_\theta(x))^{1-y}$$

$$l(\theta) = \log\left(L(\theta)\right)$$

$$= \sum y_i\, log\left(h_\theta(x_i)\right)$$

$$+ (1 - y_i) log(1 - h_\theta(x_i))$$

Here we can calculate decision bound θ to maximize $l(\theta)$. After obtaining the decision bound we can apply the model to predict the future value, in which we can set predicted value equal 0 if $h_\theta(x) > \mathbf{0.5}$ and 1 otherwise.

1.4 Support Vector Machine

Support is supervised learning model, which is regarded as one of the most robust and precious machine learning methods. It can be classified into two types. One is Support Vector Classification (SVC), the other is the Support Vector Regression (SVR). SVM was proposed by Vapnik and is based on solid statistical theories. It performs much better than other machine learning algorithms when dealing with the problems with high dimensions or small samples.

Support Vector Classification

SVC is involved in this project. To be briefly, SVC is to find the maximum-margin hyperplane to separate the data set into two subsets. The hyperplane can be defined as

$$\boldsymbol{w}^T\boldsymbol{x} + b = \mathbf{0}$$

The distance from the sample x to the maximum-margin hyperplane is

$$r = \frac{\boldsymbol{w}^T\boldsymbol{x} + b}{||\boldsymbol{w}||}$$

where *r* is called geometrical margin and $\boldsymbol{w}^T\boldsymbol{x} + b$ is called the functional margin.

SVC aims to find the hyper plane to maximize the minimal *r* of all the sample points maximum. Without loss of generality, we can let functional margin equal to 1. Then we can skip the problem of finding the minimal *r* to make the optimal problem more simplified. The minimal *r* is converted to

$$r = \frac{1}{||\boldsymbol{w}||}$$

To make the functional margin positive, we assign a indicator variable *y* to each sample *x*, which satisfies

$$\begin{cases} y = \mathbf{1}, if\ \boldsymbol{w}^T\boldsymbol{x} + \mathrm{b} \geq 1 \\ y = -\mathbf{1}, if\ \boldsymbol{w}^T\boldsymbol{x} + \mathrm{b} \leq -1 \end{cases}$$

Then the product of *y* and functional margin is always positive.

The sample data points $\langle x_i, y_i \rangle$ which can make the inequality above satisfied are called **support vector**.

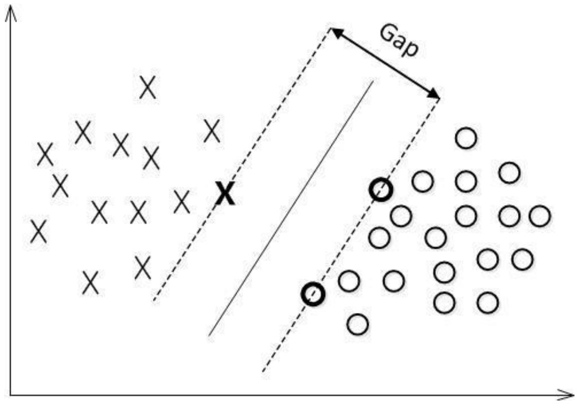

Figure 3: Support Vectors and the gap

The gap between the support vector located in the different side of the hyperplane is called the margin of separation ρ, which is the double of *r*

$$\rho = \frac{2}{||\boldsymbol{w}||}$$

The constrained optimization problem to find the maximum margin hyperplane is

$$\max_{w,b} \frac{2}{||\boldsymbol{w}||}$$

$$s.t.\, y_i(\boldsymbol{w}^T\boldsymbol{x} + b) \geq 1, i = 1,2, \ldots, n$$

The optimization problem above is equivalent to

$$\min_{\boldsymbol{w},b} \frac{1}{2}||\boldsymbol{w}||^2$$

$$s.t. y_i(\boldsymbol{w}^T\boldsymbol{x}+b) \geq 1, i = 1,2,\dots,n$$

By using the method of Lagrange multipliers, we can construct Lagrange function

$$L(\boldsymbol{w},b,\alpha) = \frac{1}{2}w^Tw - \sum\nolimits_{i=1}^{n}\alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{x_i}+b)-1]$$

where α_i is the Lagrange multiplier with respect to the i th inequality.

The problem can be described as

$$\min_{\boldsymbol{w},b}\max_{\alpha} L(\boldsymbol{w},b,\alpha)$$

The problem above is called primal problem.

Dual Problem

The dual problem of the primal problem is

$$\max_{\alpha}\min_{\boldsymbol{w},b} L(\boldsymbol{w},b,\alpha)$$

In SVM, we assume that there exists a hyperplane which can divide the data points into two parts (For linear inseparable problems, we can use kernel function to increase the dimensions to make the data linear separable, which will be introduced later). Therefore, the dual problem follows the strong duality. That is to say the primal problem and its dual problem with Karush-Kuhn-Tucker (KKT) complementary condition are equivalent.

Differentiating the Lagrange function with respect to $\boldsymbol{w}$ and *b,* we can obtain

$$\begin{cases} \boldsymbol{w} = \sum\nolimits_{i=1}^{n}\alpha_i y_i x_i \\ \sum\nolimits_{i=1}^{n}\alpha_i y_i = 0 \end{cases}$$

Substituting the equations above into the dual problem, the dual problem is converted to

$$\max_{\alpha} W(\alpha) = \sum\nolimits_{i=1}^{n}\alpha_i$$

$$-\frac{1}{2}\sum\nolimits_{i=1}^{n}\sum\nolimits_{i=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j$$

$$s.t.\sum\nolimits_{i=1}^{n}\alpha_i y_i = 0$$

$$\alpha_i \geq 0, i = 1,2,\dots,n$$

with KKT condition

$$\alpha_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)-1] = 0, i = 1,2,\dots,n$$

α_i is nonzero for support vectors and for all the other data points, α_i is zero.

After the optimal α_i^* is determined, the optimal weight vector is

$$\boldsymbol{w}^{*T} = \sum\nolimits_{i=1}^{n}\alpha_i^* y_i x_i$$

Then we can obtain corresponding optimal b^*

$$b^* = \mathbf{1} - |\boldsymbol{w}^{*T}\boldsymbol{x_i}|$$

Kernal Function

The kernal function is a commonly technique to solve linear inseparable problems. The kernal function transforms the data into a feature with higher dimension in order to make problems linearly separable by inner production.

Given $\boldsymbol{\phi}: \boldsymbol{X} \to \boldsymbol{H}$ which denotes a nonlinear transformation from input space *X* to feature space *H,* the data is mapped into *H* with higher dimension. Then we can use the SVC method to find the optimal hyperplane. The optimal hyperplane is

$$\boldsymbol{w}^{\phi T}\phi(\boldsymbol{x}) + b = 0$$

Similarly the optimal weight vector is

$$\boldsymbol{w}^{\phi *} = \sum\nolimits_{i=1}^{n} \alpha_i^* y_i \phi(\boldsymbol{x}_i)$$

Kernal function is the inner product of the mapping vectors of two sample points. It is denoted as

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi^T(\boldsymbol{x}_i)\phi(\boldsymbol{x}_j)$$

Then we can obtain the hyperplane function

$$\sum\nolimits_{i=1}^{n} \alpha_i^* y_i K\phi(\boldsymbol{x}_i, \boldsymbol{x}) = 0$$

Actually, the linear inseparable issue is a very common problem. If we simply map the original data to the higher dimension feature space, the dimension of the feature space will increase rapidly. Sometimes it will be even mapped into infinite dimension space. Kernel function aims to solve this problem. Though it also transform the lower dimension space into higher dimension space, it can make all the computation done in the lower dimension space and the calculated result is equivalent to the result computed in the higher dimension.

2. Choice of Stock Market

There are many different stock and markets in the world, and therefore we should clarify some criteria to decide which market and stock we ought to analyze.

Firstly the target market and stock should be large enough so that they cannot be manipulated easily. Secondly, the market and stock should already exist for a long time so that we have enough data to mine. Thirdly, the data should be accessible easily and not expensive to get so that the analysis is practical. Also, the market we choose cannot be efficient in general so that we have more opportunity to make a profit overriding the index.

After considering many potential options, we finally decided to choose China's stock market. A representation of Shanghai Stock Exchange, Shanghai Pudong Development Bank, listing as the first stock on the market, is chosen to be our target stock.

3. Data Collection and Processing

We use Shanghai Pudong Development Bank (SH600000) as the underlying stock. The original time series data and financial data are collected from the stock database. We have also calculated some popular technical indicators. In this case the available input parameters and the corresponding abbreviations in the data set are as follow.

3.1 Daily information

Daily time series data includes open price, highest price (HIGH), lowest price (LOW), close price (CLOSE), volume (VOLUME) and volume percentage change (P_VC) in every trading day.

3.2 Moving average

5, 7, 10, 20, 50, 60 and 65 days moving average close price are involved, the abbreviations for them in the data set are MA5, MA7, MA10, MA20, MA50, MA60 and MA65.

3.3 Net change of close price

We calculated the net change (NC[*n*]) and corresponding percentage net change (P_NC[*n*]) of previous 1, 2, 3, 5 and 10 day close price. The *n* in the abbreviation means the length of the chosen period.

3.4 Financial data

Financial data including the earning per share (EPS) and P/E ratio (PE).

3.5 SSEC

SSEC is short for Index of Shanghai Stock Exchange Composite. We also calculate its net daily change (DIFFSSEC) and percentage change (P_DIFFSSEC).

3.6 ADX

Average Directional Movement index of SSEC (ADX) is a technical indicator measuring trend strength in a series of prices. The computation process is very complex, we don't illustrate the formula here.

3.7 B/B index

Bull/Bear index (BB) ranges from -1 to 1, to imply the big market trend, calculated by standardize ADX values. B/B index lager than 0 implies a bullish market otherwise a bearish market.

3.8 Indicator of Continuous gains/looses

We consider that indicators of 2 or 3 days' continuous gains or losses as input parameters, which is a Boolean value. The abbreviation is HH[*n*] for increasing price or LL[*n*] for decreasing price. The *n* is 2 or 3.

If the stock satisfies the certain condition, the corresponding indicator is 1. Otherwise, it will be set to be 0.

3.9 RAVI

The full name of RAVI is Region Activity identification, which is the absolute value of the present that the difference of 7 day simple moving average and 65 days simple moving average divided by 65 days simple moving average. Compared with the reference value of 3 percent, if RAVI is less than 3%, the stock is in the fluctuated period. Otherwise the stock is in the rising trend or falling trend. The formula is as follows

$$RAVI = \left|\frac{100 \times (7MA - 65MA)}{65MA}\right|$$

We also consider another indicator similar to this, which just replace *7MA* with *20MA* and *65MA* with *60MA*. We simply call it *DIFFMA*.

3.10 Stochastic Indicator

Stochastic Momentum Indicator is momentum oscillator intended to help determine the strength of price trends and to highlight potential short-term market overbought and oversold levels. It has two main variables (K and D), defined at each time t as follow:

$$K_t(n) = \frac{P_t - P_t^{min}(n)}{P_t^{max}(n) - P_t^{min}(n)}$$

$$D(n) = \sum\nolimits_{i=1}^{3} \frac{K_{t-i}(n)}{3}, n \le 3$$

Where $P_t^{min}(n)$ and $P_t^{max}(n)$ are the minimum and maximum price during the last n days. In this case n equals to 3.

3.11 RSI

RSI is short for Relative Strength Index, which is a momentum oscillator used to compare the magnitude of it recent losses to help determine the overbought and oversold conditions.

The formula is as follow

$$RSI(n) = 100 - \frac{100}{1 + RS(n)}$$

Where $RS = \frac{Average\ Gains}{Average\ Losses}$

We consider RSI with value of *n* in 6, 12 and 24.

3.12 MACD

MACD is short for Moving Average Convergence Divergence, which is a very common technical indicator regarded as one of the most reliable indicators in the stock market. It is a trend following momentum indicator that exhibits the relation between two distinct moving averages. The computation steps for MACD is shown below.

$$EMA_t(n) = \alpha \times P_t + (1 - \alpha) \times EMA_{t-1}(n)$$

where $\alpha = \frac{2}{n+1}$

$$DIF_t = EMA_t(12) - EMA_t(26)$$

$$DEA_t = DEA_{t-1} \times 0.8 + DIF_t \times 0.2$$

Then we can obtain the MACD

$$MACD_t = DIF_t - DEA_t$$

Based on the large number of our experiments, different machine learning methods may choose different parameters as its input parameters to make the methods perform better.

3.13 Prediction label (dependent variable) selection

As many machine learning models being tested, diffident prediction time period would be calculated and applied to optimize model accuracy. We define prediction label(x) as the indicator for whether x-day-later stock price would be larger than price today, if so label is set one, otherwise it's zero. Using different time period is not only for models optimization, we also consider in economic views that the stock price needs some time to response for relative information, not immediately.

4. Trading Strategy

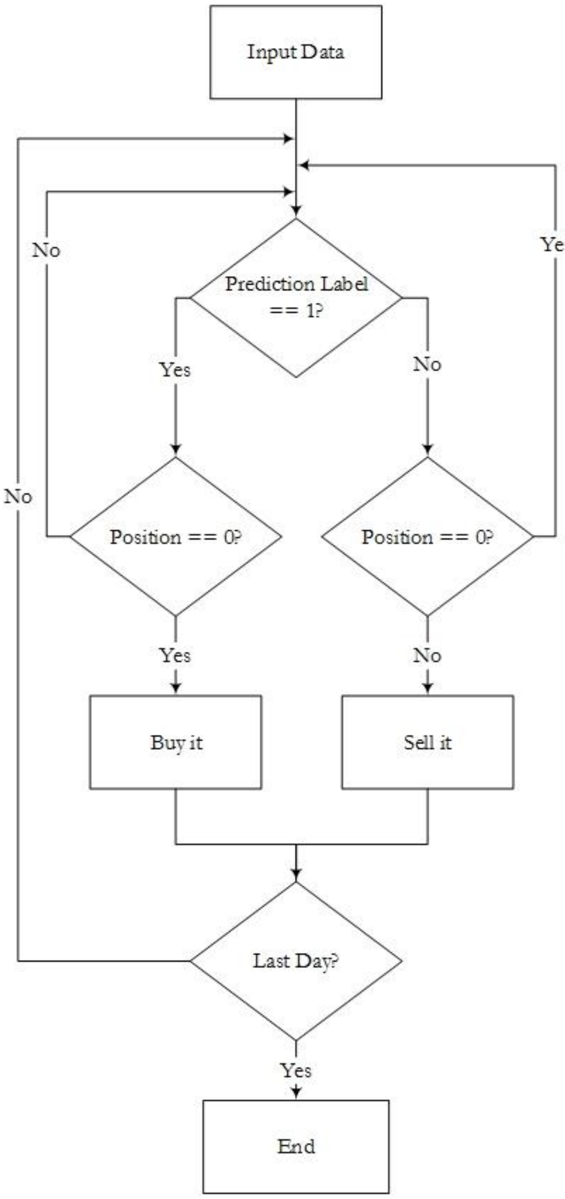

Figure 4 Flow chart of the trading strategy

Our trading strategy is closely related to the result of prediction. We use the T+1 strategy (i.e. we can only do one trading operation in one day) and make the decision at the last moment of the trading session in the afternoon. Thus the enter price or sell price is always close to the close price of the underlying stock in the corresponding day. We assume that we can buy the stock with the close price immediately when we finished the data analysis on today's close price.

We test on the predictions based on different period ahead. According to the test performance, we will choose the prediction with the highest accuracy.

The trading period in this simulation is from 22nd August 2013 to 6th March 2015. We assume that we have the initial wealth of ￥1 at the first time. Flow chart of the trading strategy is as shown in previous page.

5. Implementation and Results

5.1 Linear Discriminant Analysis

Firstly we use LDA/Fisher's linear discriminant as introduced at 1.1 section into the practical stock market and fix our dataset into the LDA method and use programming R to calculate the relative formula and get some plots to predict out model.

Firstly, after plenty of complex selecting calculations, all continuous independent variables are chosen by testing each possible independent variables to maximize the account balance. We finally get 5 independent variables, LL3, HH2, HH3, PE and MACD to construct the linear discriminant function in this case.

Then from programming R output, we can plot three graphs to analysis stock marketing by linear discriminant analysis (LDA) method.

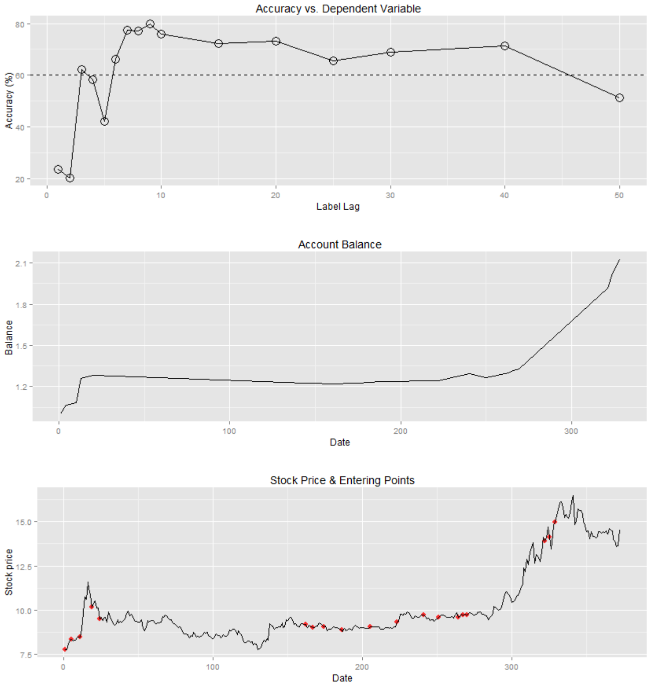

Figure 5 Linear Discriminant Analysis Result

From figure above, after 9 days we get the maximal accuracy 80% between training data and testing data in first graph, hence we choose L9 used for our objective dependent variable in LDA model. Then at the end of the year, we get maximal account balance which is more than 210% as the original balance from the second graph in figure 2. Comparing to the third graph in figure 2 which is the trend of real stock price, our account balance curve fixes the retracement of real stock price by using LDA method.

5.2 Logistic Regression

5.2.1 Features selection for logistic regression

We are focusing on the linear-relationship independent variables in logistic regression because logistic model is a simple, linear model. According to the results of previous experiments in which including variable significance and AIC value, we find that it seems close price, moving average price series, EPS, PE, B/B indicator and SSEC index are more likely significant (here 5% as significance level).

5.2.2 Early implementation

In firstly team discussion we establish training set from 4 Jan 2010 to 21 Aug 2013 (866 data points totally) and 22 Aug 2013 to 6 Mar 2015 as test set (372 data points totally). The result of logistic regression turn out not very positive. As can be seen as below, the independent variables include trading volume, 50 and 60 day moving average price, DIFFMA, EPS, PE, B/B indicator and SSEC index.

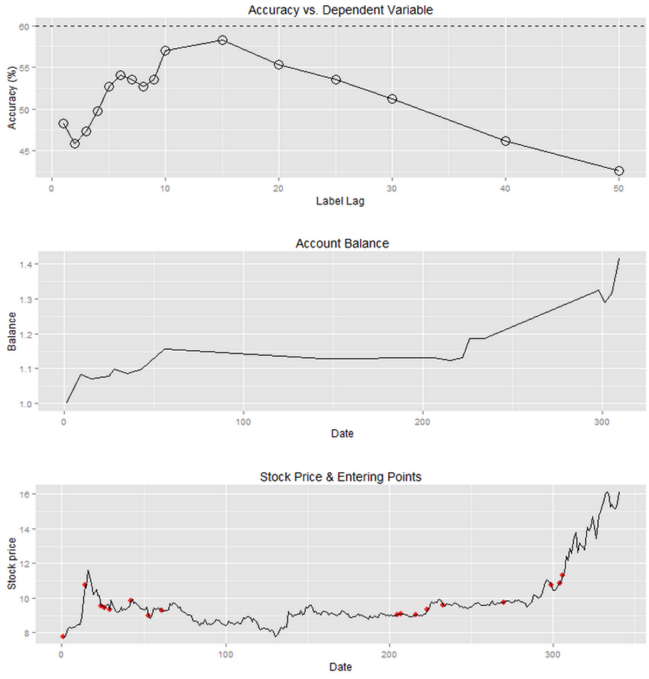

Figure 6 Logistic Regression Result 1

The best accuracy is 56.76% (by back test with training set) on time period 15 days with total profit 40% according to our trading strategy. As we can see in chart Stock Price & Entering Points, holding this stock all time would obtain more than 100% profit and in the meantime, growth of SSEC is 61.40%, which means there's lot of room for improvement of logistic regression.

5.2.3 Redefine model and discussion

Training and test set selection

One reason of low profit might be the training set from 4 Jan 2010 to 21 Aug 2013 is more likely bearish market however in the test set the SSEC growth is more than 61.40% which is a totally diffident market. In the above chart it's very clear entering points mostly in shocking or volatile market and returns are better than SSEC index and stock. But in bullish market, the strategy hardly gives entering points, missing large profit.

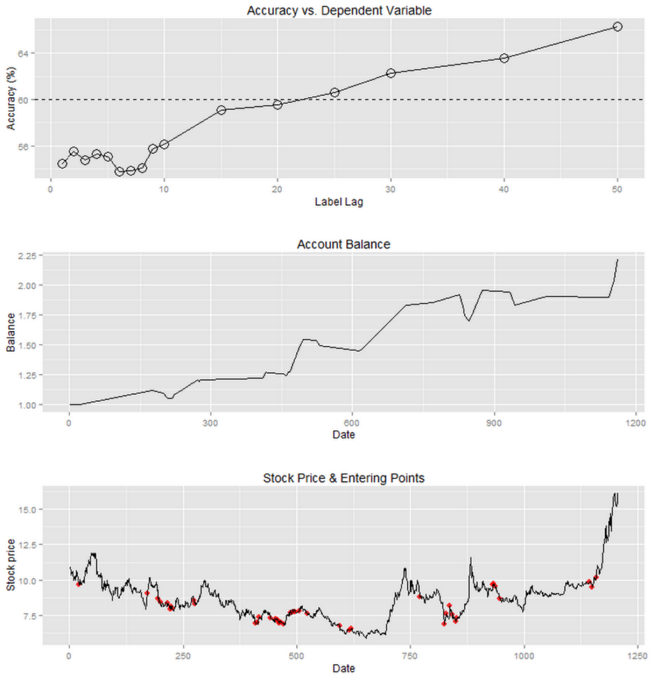

Figure 7 Logistic Regression Result (refined model)

To solve this problem we swap over training and test set which contains both bullish and bullish trend. The result turns out good performance in bullish market but not in bearish. Instead of swapping over, we directly change the training set from 1 Feb 2011 to 28 Feb 2014 including shocking, volatile downside and bullish market. The result with all data as test set, 50-day label is as above (Figure 7). The profit is 220.8% and profit in last 3 trades (in bullish market) is more than 38.0%, though this return is less than one of SSEC index, really better than previous.

Time period selection

Generally the accuracy of logistic regression is much larger following increasing of time period, however more precise entering points, meaning more profit we can get with less time period. So we should find a balance for time period and profit. In logistic regression model, profit changes little when time period more than 20. According to above analysis we finally choose time period 50 days.

5.3 Quadratic Discriminant Analysis

After selecting tens of different permutations of different data set, we finally decided to use the following data to do QDA classification: volume, EPS, continuous decreasing days, continuous increasing days, difference of moving average and PE. And the result is shown as following:

As we can see in the figure 8, the accuracy reached its peak, which is 70%, at the 15th day. Trading with the corresponding strategy, our balance almost doubled during the test days, while the stock price doubled during the same, too. Thus we cannot say that trading with this strategy is better than the simplest buy-and-hold strategy.

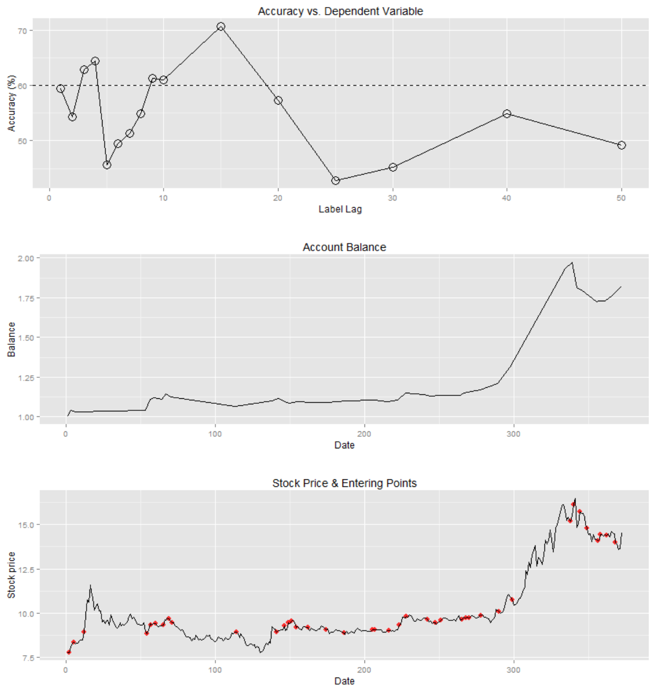

Figure 8 Quadratic Discriminant Analysis Result

5.4 Support Vector Machine

Based on many experiments, SVM does not perform very well if we input some absolute values such as SSEC and MA50.

Thus we skip all the absolute values and mainly consider the percentage parameters, difference parameters and Boolean parameters. The input parameters chosen for SVM are C_MA5, C_MA10, LL2, HH2, K, D, ADX, BB, DIFFMA, MACD, RAVI, RSI6, RSI12, RSI24 and P_VC.

The parameters for SVM is as follow.

Predictor	SVM
Kernel function	Polynomial
Number of features	15
Cost coefficient	0.5
Gamma	1/15

The results are shown below in figure 9.

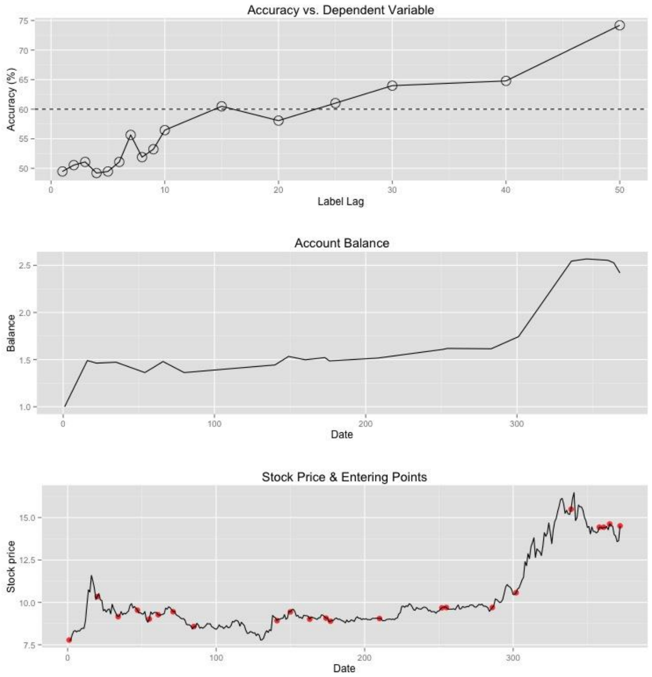

Figure 9 Support Vector Machine Result

From the first figure, we can see the precise rate has an increasing trend when the time window for prediction increases. Though the accuracy is only around 50% for short time window prediction, the highest accuracy is 74.2% for the 50 days ahead prediction.

The balance curve based on our trading strategy is shown in second figure. We can see final balance is 2.419. If we hold the stock during the whole period, the final balance is 1.865. Therefore our trading strategy outweighs the performance of the stock.

The red points in the third figure illustrates the time and price we enter the stock market.

6. Conclusion

In this project, we applied machine learning techniques in predicting the stock price trend of a single stock. In a nutshell, we can summarize our findings into several aspects:

Various learning methods were used for the prediction and we found that linear discriminant model could provide the highest predicting accuracy of 80%, when we predicted the stock price trend on a long-term basis of 9 days.

Our trading strategy turned out to perform quite well and achieve a very positive result, outperforming the corresponding stock index significantly. Based on the liner discriminant model, if we choose to buy the stock when the system gives a buying signal and sell the stock when it gives a selling signal, namely '1' and '0' signals, we expect to make a profit of 210% in nearly 2 years.

As a limitation, we only consider one stock in one equity market so that more test on different stocks should be per-formed to see the robustness of our system. A more general predictor can be developed for the market.

Another limitation of our sys-tem is that we did not diversify our risk into multiple stocks. A portfolio instead of a single stock may reduce the retracement of out P&L curve. Also, we did not take into consideration the transaction cost, which is not negligible in the real market. For further system development, those factors mentioned above are all worth considering when evaluating the strategy's effectiveness.

Reference

[1] Xindong Wu, Vipin Kumar. The Top Ten Algorithms in Data Mining [M]. Chapman and Hall/CRC. 2009

[2] HUI Xiao-feng, WU Ya-jun. Research on Simple Moving Average Trading System Based on SVM [I]. 2012 International Conference on Management Science & Engineering (19th), September 20-22, 2012

[3] Jan Ivar Larsen. Predicting Stock Price Using Technical Analysis and Machine Learning [D]. NTUT. Jun, 2010

[4] Drew Conway, John Myles White. Machine Learning for Hackers [M]. O'Reilly Media. Feb, 2012

[5] Peter Harrington. Machine Learning in Action [M]. Manning Publications. Apr, 2012

[6] Phil Simon. Too Big to Ignore: The Business Case for Big Data [M]. Wiley. Mar, 2013

Appendix

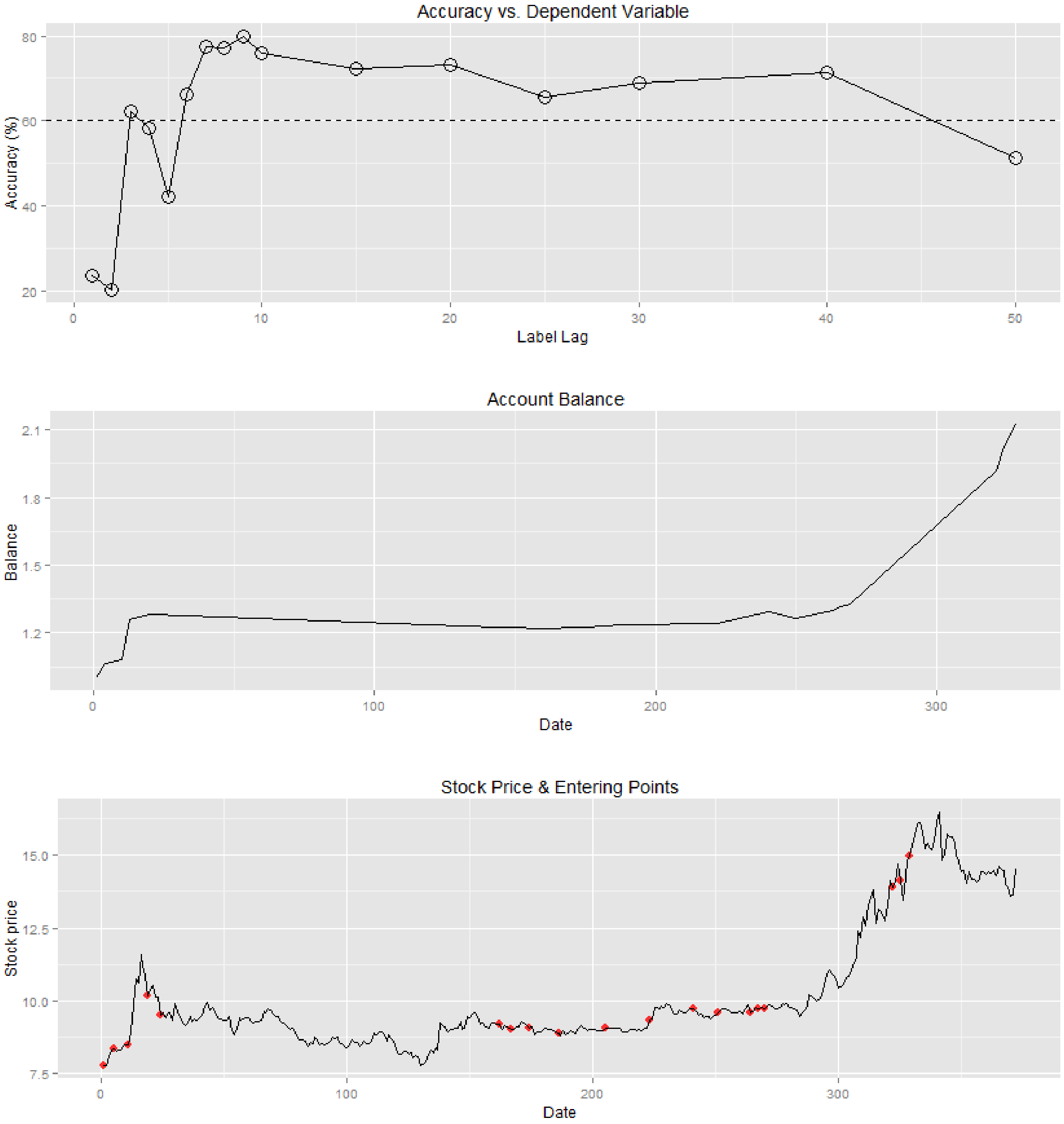

Figure 10 large image of liner discriminant model result