This paper was presented at the IASTED Conference on Modeling and Simulation, Benaldema, Spain, June 2005.

Modeling the United States Stock Market with Kernel Regression

John R. Wolberg, Technion – Israel Institute of Technology
jwolber@attglobal.net

Ronen Kimche, Intel Corporation
Ronen.kimchi@intel.com

David Aronson, Baruch College - City University of New York
aronson@mindspring.com

Keywords: Statistical Modeling, Simulation Tools, Kernel Regression, Stock Market

Abstract

In late 2000 we initiated a project to develop a short term system for modeling the United States stock market. All stocks listed on the NYSE, AMEX and NASDAQ exchanges were included in the database. The modeling technique was kernel regression, a technique we had previously used in an earlier venture. The system was based upon predictions four days into the future. Nine technical indicators were considered with varying time windows from 5 to 30 days. Models were created from combinations of up to three dimensions of the indicator space and the best five models were chosen based upon a learning period of about one year. Each model gave a score for every active stock in the database and an average score for the best five models was used to sort the stocks. Initially every second day a list of the top and bottom 20 stocks was created. Eventually we created a daily list. Longs were chosen from the top list and shorts were chosen from the bottom list. Actual trading commenced in May 2001 and many lessons were learned as we gained experience. In this paper the method of kernel regression as applied to stock market modeling is described. The paper also discusses some of the technical problems that must be solved to develop a working system. Results from actual trading are also included in the paper.

The Database

The raw data required to generate the database used for modeling and predictions is daily open, close, high, low and volume data for all stocks listed on the major United States exchanges: the NYSE, the AMEX and the NASDAQ. Data is available from 1984 and is updated daily. The early data was used in the initial modeling process and was used to set the basic parameters of the ongoing stock ranking algorithm. The raw data is used to generate a file that includes all entries for all stocks in a specified period for specific dates within the period. To avoid serial correlation, data is grouped by dates so that each entry for a specific stock within the period is totally independent from the other entries of the same stock. For example, if we are considering an N day return on the stock, then we look at the return from the open on day 1 to the close on day N, the open on day N+1 to the close on day 2N, etc. Clearly, if we start another data set on day 2 (i.e., open on day 2, close on day N+1, open on day N+2, etc.) this data is different then the first data set but not entirely independent. However, it can be used to make decisions at the opening of trading on days 2, N+2, 2N+2, etc. In fact, we can have N such data sets. Only on day N+1 do we return to entries from the original data set. What this implies is even if we model based upon an N day return, we can generate predictions daily and the predictions will be based upon a data set of independent entries for each stock.

The Stock Ranking Algorithm

The target variable that was chosen after analyzing the data back to 1984 was a 4 day return (relative to the S&P index). We also employed a moving data window for model adaptation with 36 dates as the modeling period and 18 dates as the test period. The time period covered by 36 dates is 36*4 = 144 trading dates plus 72 testing dates. On any given date only every 4th date within the applicable period is used. The method of model selection is described below in the section on Kernel Regression. Once the models are selected, they are used for generating predictions for a further 18 dates (spanning an additional period of 72 trading dates). All data in the 36+18 = 54 modeling and test dates are used to make the first ranking. At the end of day 4, the returns for that date are known and thus the 2nd rankings are based upon 55 dates. The 18th rankings in the period are based upon 54+17 = 71 dates. After completion of the 18th ranking in the 72 date period, the time window for the modeling process is moved ahead 18 four-day periods (i.e., 72 trading dates) and the process is repeated.

One might ask "why not use all the data in the database from 1984 to make the predictions?" The answer is that the dynamics of the markets change over time. So what was a reasonable model several years ago might not be relevant today. To try to capture the dynamic nature of the markets, a reasonably short period of time should yield better results. Our initial analysis of the data back to 1984 led us to the choice of the 36 date modeling and 18 date test periods with an 18 date usage of this data for generating rankings. The modeling plus test period covers 216 trading dates. There are about 22 trading dates per month so about 10 months of data is used to create models and then the models are used for an additional 18*4 / 22 months (i.e., about 3 months).

Another interesting issue is the amount of entries in the file used to generate the rankings. Currently the number of stocks included in the database is approximately 7000. So initially the first ranking is based upon approximately 54 * 7000 = 378,000 entries. The 18th ranking is based upon approximately 71 * 7000 = 497,000 entries. The number of stocks varies from day to day as new companies emerge and older companies merge or are delisted for any number of reasons. Thus the number of entries for each date in the database is also dynamic. Stock splits are another problem that must be handled to make reasonable predictions. For example, if a stock splits 2 for 1, then the price drops to approximately one-half of the pre-split price. The system must identify the splits and not treat them as true massive drops in prices.

It is important to note that the target variable used for modeling was based upon the 4 day returns relative to the S&P index and not just the 4 day returns. What one hopes to achieve is identification of stocks that will out-perform or under-perform the market. Use of returns relative to the market enables a market-neutral long-short approach to investing. Theoretically, the returns achieved with such a strategy should be relatively independent of the market as a whole. By removing the 4 day return of the S&P index from the 4 day returns of the stocks in the database, the average return in the database will be close to zero. Thus if the predictions are correct, positive values only imply that the stocks will out-perform the market. If the market is rising, the stocks will rise more than the market. If the market falls, the stocks will show greater returns than the market. These returns might be positive or might just be less negative. For stocks that are predicted to under-perform the market, if the predictions are correct, if the markets are rising, the stocks will rise less or might even fall. If the markets are falling, the stocks should fall more.

Kernel Regression

Kernel regression is one class of data modeling methods that fall within the broader category of *smoothing* methods. The general purpose of smoothing is to find a line or surface which exhibits the general behavior of a dependent variable (lets call it *Y*) as a function of one or more independent variables (lets call them *X* variables). For this particular application the *Y* is defined as the 4 day return relative to the S&P index. For this application 9 technical variables were the independent variables used to model *Y*. No attempt is made to fit *Y* exactly at every point. If there is only one independent variable, then the resulting *smoothing* is an undulating line. If there is more than one independent variable, the *smoothing* is a surface. Smoothing methods that are based upon a mathematical equation to represent the line or surface are called parametric methods. On the other hand, data driven methods that only smooth the data without trying to find a single mathematical equation are called nonparametric methods. Kernel regression is a nonparametric smoothing method for data modeling.

The distinguishing feature of kernel regression methods is the use of a *kernel* to determine a weight given to each data point when computing the smoothed value at any point on the surface. There are many ways to choose a kernel. Wolfgang Hardle reviews the relevant literature in his book on this subject [1]. Another overview of the subject by A. Ullah and H. D. Vinod is included in the Handbook of Statistics [2].

When using data to create models, it is useful to separate the data into several categories:

1) *Learning* data
2) *Testing* data
3) *Evaluation* data (i.e., Reserved data for final evaluation)

The usual strategy is to divide the data with *nlrn*, *ntst* and *nevl* points assigned to the three data sets. For the stock market modeling application, the values of *nlrn*, *ntst* and *nevl* were based upon *nlrndates*, *ntstdates* and *nevldates.* The independent variables are called "candidate predictors". For various subspaces of the 9 dimensional candidate predictor space, the *nlrn* learning points are used to make predictions for the *ntst* testing points and then some measure of performance is computed. One iterates through the various subspaces following a searching strategy. For this particular application all one-dimensional, two-dimensional and three-dimensional subspaces of the nine-dimensional candidate predictor space are considered. The five best models are then used with all the *nlrn* and *ntst* data points to make predictions using the *nevl* evaluation data points. Each of the five models gives a score to each stock in the evaluation data set. The stocks are then ranked by the average score from the five models. The use of 5 models rather than the single best model is consistent with the notion that averaging independent forecasts will produce a combined forecast with less error than any individual model. One then compares the actual average *N* day return for the top stocks versus the bottom stocks. Over the analysis period starting in 1984, the top stocks significantly out-performed the bottom stocks in the ranking lists. The dates were advanced *nevldates* and the process was repeated. Values of *nlrndates*, *ntstdates* and *nevldates* and the value of *N* (the number of trading dates used to determine the returns) were selected using this procedure. This process should not be repeated too often because if is repeated many times the probability of achieving a high measure of performance purely by chance increases dramatically.

The predictions are typically based upon algorithms of ***Order 0, 1, or* 2** [3]. The simplest kernel regression paradigm is what I will call the ***Order 0 algorithm***:

$$\boldsymbol{y_j} = \frac{\sum_{i=1}^{nlrn} \boldsymbol{w(x_i, x_j, k) Y_i}}{\sum_{i=1}^{nlrn} \boldsymbol{w(x_i, x_j, k)}} \qquad \textbf{(1)}$$

In the statistical literature this equation is often referred to as the *Nadaraya-Watson estimator* [1,2]. In this equation $\boldsymbol{y_j}$ is the value of $\boldsymbol{Y}$ computed for the jth test or evaluation point. Typically the weights $\boldsymbol{w}$ are computed using a Gaussian kernel:

$$\boldsymbol{w(x_i, x_j, k) = e^{-k D_{ij}^2}} \qquad \textbf{(2)}$$

The parameter $\boldsymbol{k}$ is called the *smoothing parameter* and $\boldsymbol{D_{ij}^2}$ is the squared distance between the learning and test points. If $\boldsymbol{k}$ is assigned a value of 0 then all points are equally weighted. As $\boldsymbol{k}$ increases, the nearer points are assigned greater weights relative to points further away from the jth test point. As $\boldsymbol{k}$ approaches infinity, the relative weight of the nearest point becomes infinitely greater relative to all other points. The values $\boldsymbol{Y_i}$ are the actual values of $\boldsymbol{Y}$ for the learning points.

Equation (1) is not useful for applications in which the value of *nlrn* is large. It implies that all learning points are used to compute a weighted average value at point $\boldsymbol{x_j}$. If all points are used, then the calculational complexity is enormous: O(*nlrn*ntst*). The approach followed in this study was to only use nearby points. In other words, Equation (1) is replaced by the following:

$$\boldsymbol{y}_j = \frac{\sum_{i=1}^{near} w(\boldsymbol{x}_i, \boldsymbol{x}_j, k)\boldsymbol{Y}_i}{\sum_{i=1}^{near} w(\boldsymbol{x}_i, \boldsymbol{x}_j, k)} \tag{3}$$

The index *near* is the number of nearest points as determined using an algorithm described in [3]. The actual points chosen are not the real nearest neighbors but a close approximation. Analysis showed that a value of ***k*** = 0 was a reasonable choice so that the kernel becomes one for the points included in the *near* list and zero for all other points. For every point in the test or evaluation data sets, the *near* points must be determined but this becomes a simple task when the learning data is organized in a suitable data structure. Using ***k*** = 0 Equation (3) can be further simplified:

$$\boldsymbol{y}_j = \frac{\sum_{i=1}^{near} \boldsymbol{Y}_i}{near} \tag{4}$$

Use of Equation (4) rather than Equation (1) resulted in a speedup of several orders of magnitude and the results were significantly better. Using nearest neighbors makes a lot more sense than eliminating the effect of distant points through the use of a large value of ***k***. The need for speed is quite essential. Using Equation (4) the daily compute time to generate the predictions and then the ranked list of stocks takes over an hour using a 400 megahertz computer. If Equation (1) were to be used, the time required would be more than the hours between market close and market open on the next day.

Simulation versus Trading

The project consisted of two distinct phases:

(1) The simulations of the market phase. This phase was required to set the system parameters and to answer the basic question: "should the system be profitable?"

(2) The real-time phase. Once trading commenced, this phase was initiated.

Once the parameters had been set, the same software was used to enter the second phase. All data points in the learning and test data sets are combined to produce a single learning data set. The evaluation data set includes a single date which is the current date. We use the Worden Brothers data service and data (i.e., open, close, high, low and volume) for every stock in the markets of interest. The data is available a few hours after the markets close. The generation of the predictions from the five models, computation of an average expected relative return, ranking of this average score and generatation of the top and bottom reports takes about an hour, so there is plenty of time before the markets open the next day to make trading decisions. Periodically the time window is moved forward, new models are selected and the process continues as before.

Trading Strategy

In May 2001 we started trading based upon the system described above. A list of the 20 top and bottom stocks based upon the ranked list was generated every day. Initially 5 stocks were chosen from the top list and five stocks were chosen from the bottom list. Long positions were taken on 5 stocks from the top list and short positions were taken on 5 stocks from the bottom list. Every 2 days another 5 longs were bought and 5 shorts were sold at the start of the day and 5 long and short positions were closed at the end of the day. We entered positions using market orders and closed them also using market orders. Eventually we switched to limit orders and therefore sometimes we didn't get all 5 longs and shorts. So the number of positions held during the trading days was usually less than the 10 longs and shorts that our initial strategy called for. We learned some very costly lessons during this period.

On Sept 11, 2001 we had 10 long positions and only 3 short positions and when the World Trade Center and the Pentagon were attacked the market closed for several days. When the market reopened we lost all our profits from the initial months of trading and then some. We learned never to let the system get so unbalanced again. Not only did we keep the numbers of longs and shorts close, we also did volatility balancing. In the daily reports we include column that is a measure of volatility. This column lists a parameter that we call the Actual True Range and is a fractional measure of the volatility of each stock. Typically the stocks in the short list are more volatile than the stocks in the long list so the dollar amount for long stocks was typically greater than the dollar amount for the short stocks.

Another lesson learned was not to go into a stock when an announcement is to be made within the 4 day holding period. We had a few experiences in which bad news caused a large drop in some of our long positions. For shorts, the opposite sometimes occurred: good news for companies in which we held short positions caused losses in these short positions. We also had some very pleasant surprises due to news releases but we decided to research our choice of the longs and shorts and eliminate any stock in which an announcement would be forthcoming.

Another lesson learned was that a lot of time was required to run the system. Besides researching the choices taken from the lists, the actual trading based upon limit orders was time consuming. For a large hedge fund this is not a serious problem, but we were only interested in a proof-of-concept operation and therefore the operation was low-budget. None of us were working full time on this project so we decided to reduce the effort by holding fewer longs and shorts but larger positions in each one. This resulted in a much lower level of effort to run the system. We eventually generated daily lists and just went in and out of positions at will. Although the modeling was based upon a 4 day return we just used the lists for stock selection. Exiting a position was another matter. We sometimes held positions for a month or two if we felt that the position was worth holding. When the position was closed out we looked at the latest list to find a replacement. Sometimes we could not find a suitable replacement so our fall back position was to go either long or short using the QQQ NASDAQ 100 Tracking Stock. This price of the QQQ's rises or falls with the NASDAQ 100 index. What became clear was that we were no longer trying to trade based purely upon computer generated directives. We were using the computer output as a starting point for our own judgment.

Results and Conclusions

The system was used for actual trading from May 2001 through the end of 2002. Although 2002 was quite profitable when compared to the major indices, we decided to terminate trading due primarily to the time and aggravation involved in such a venture. None of us were willing to continue to put in the hours required to run the system. From a system that had been mechanical, it turned into a computer based stock selection system that required a lot of judgment. The portfolio return results are included in Table 1 and the system did outperform the market for both years. The year 2001 was a learning experience and although the first few months started well, after September 11th, the remainder of the year was painful. In 2002, the number of transactions was reduced considerably and the profits were made in the short transactions. A comparison of the 2 years is shown in Table 2. Actually, the profit for the short transactions was 13.4% but this was offset by a 1.6% loss due to the net loss in the long transactions. Even though over 63% of the long transactions were profitable, a few large losing transactions resulted in this long net loss.

	2001 (from May 15)	**2002**
Dow Jones	-7.8%	-16.8%
S & P 500	-7.1%	-23.4%
NASDAQ	-6.5%	-31.8%
Portfolio	**-2.9%**	**11.8%**

Table 1 – Summary of Trading Results

	2001 (from May 15)	**2002**
Total num of completed trades	519	97
Total num of profitable trades	282	70
Fraction of profitable trades	0.543	0.722
Total number of long trades	252	41
Fraction of profitable longs	0.478	0.634
Total number of short trades	266	56
Fraction of profitable shorts	0.605	0.786

Table 2 – Breakdown of Trades by Longs and Shorts

Bibliography

[1] W. Hardle, *Applied Nonparametric Regression*, Cambridge University Press, 1990.

[2] A. Ullah and H. D. Vinod, *General Nonparametric Regression Estimation and Testing in Econometrics*, Chapter 4, Handbook of Statistics 11, North Holland, 1993.

[3] J. R. Wolberg, *Expert Trading Systems: Modeling Financial Markets with Kernel Regression*, Wiley, 2000**.**