
Comparison of Instance-Based Techniques for Learning to Predict Changes in Stock Prices

David B. LeRoux

LEROUX@CS.RUTGERS.EDU

Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, NJ 08854-8019 USA

Abstract

This paper is a practical guide to the application of instance-based machine learning techniques to the solution of a financial problem. A broad class of instance-based families is considered for classification using the WEKA software package. The problem selected for analysis is a common one in financial and econometric work: the use of publicly available economic data to forecast future changes in a stock market index. This paper examines various stages in the analysis of this problem including: identification of the problem, considerations in obtaining and preprocessing data, model and parameter selection, and interpretation of results. Finally, the paper offers suggestions of areas of future study for applying instance-based machine learning in the setting of solving financial problems.

1. Introduction

The fields of finance and economics provide a fertile area for application of machine learning techniques. Vast amounts of data measuring many aspects of the economy and financial markets are readily available. While many theories exist on the causal relationships among measurable quantities, generally the relationships are poorly understood and the predictive powers of the models are often weak. The problem is made more difficult because of its non-stationary nature. This is true not only because of changing underlying factors in the economy, but also because knowledge of relationships can provide opportunities to financially exploit the information, which in turn changes the environment. This is particularly true with respect to predicting future changes in stock market prices where public information is readily incorporated into current price levels.

Machine learning has the ability to deal with large amounts of data, to rapidly explore many potential models and to adapt to changing environments. This paper illustrates the use of machine learning techniques to solve a common problem in finance using publicly available data and the WEKA open source software (Witten & Frank, 1999).

2. Identification of the Problem

The problem examined here is the prediction of the future change in a stock market index based on information available at the time of the prediction. Specifically, the problem is to use publicly available monthly economic index data from the Federal Reserve to predict changes in the S&P 500 stock index. An important consideration in defining this problem is to ensure that the data used in predicting the change in the stock market index are publicly available prior to the beginning date of the period for which the stock market change is measured. For example, it would not be proper to use the change in a September employment index as an input to a model to predict the change in the October S&P index because the September employment index is not available until near the end of October and thus would not be available at the time needed to make the prediction. For this reason, the problem has been constructed to use data current as of time t to estimate the change in the stock index between times $t + 1$ month and $t + 2$ months.

The problem examined is a classification problem: Will the S&P index increase by more than typically in the month following the month when all of the data is available? In order to provide a maximally challenging binary feature for testing, the question is whether the index will increase by more than the median amount of increase during the period under consideration. By defining the problem this way, exactly half the observations are correctly classified TRUE and half FALSE, so that a constant decision rule can do no better than 50% accuracy.

3. Obtaining and Pre-processing Data

The data source for this problem is the FRED II economic database maintained by the Federal Reserve Bank of St. Louis. This data source contains over 1,000 economic time series in categories of interest rates, inflation indexes, employment rates, monetary aggregates, population data, productivity rates, and many other financial categories. The data may be downloaded for all indexes in a single zipped file and extracted in ASCII format with each series in a separate file.

The first step in preprocessing the data is to select which indexes to use and to combine them into a single file with their index dates synchronized. For this example analysis, 14 indexes were selected: 5 involving interest rates of different maturities and credit spreads (GS1, GS10, MORT, AAA, and BAA), 7 involving business conditions (INDPRO, BUSINV, ISRATIO, PPIENG, PPIACO, TCU, and NAPM) and 2 involving employment (AWHI and UNRATE). The indexes were chosen to be a broad mix of important and widely publicized economic data. All of these indexes are monthly and have been calculated for many years. The individual files were combined into a single file with one record for each month. Months prior to the date when all of the indexes became available were discarded, resulting in a full table of indexes from 1/1/1992 through 6/1/2003, or 138 months of complete data.

The WEKA software has the ability to calculate additional features by performing arithmetic operations on the existing features. This may be done on the Preprocessing window using the AttributeExpression Filter or it may be done prior to importing the data to WEKA. Since much of the information content in economic indexes comes from the change in index value rather than the simple magnitude of the index, an additional feature was added for each of the above 14 indexes to show the rate of change of the index from the prior month to the current month.

Finally, the values of the S&P 500 index were obtained from the Standard & Poor's website. For each month, m , in the feature database, the following values related to the S&P 500 index were added: index at $m+1$, index at $m+2$, change in the index from $m+1$ to $m+2$, and a "normalized up/down indicator." This last feature takes the value "up" if the index rises by more than the median historical increase during the month from $m+1$ to $m+2$ and "down" otherwise. Of course, for the up/down classification problem, the values of the index at $m+2$ must be excluded from the available features.

4. Model and Parameter Selection

Conceptually, instance-based or lazy-learner methods are simple, following the basic rule of classifying or estimating based on the correct values of similar instances in the known history. There are, however, many choices involved in selecting a particular instance-based model. In this section consideration is restricted to the k -nearest neighbor model as implemented in the WEKA software. Reference is made to the model and parameter selection options available in version 3.2.3 of WEKA. As will be seen, this software offers considerable flexibility and requires numerous choices in parameter selection. An active area of research is how to explore the parameter selection space in an efficient, automated way to identify the optimal parameters (Maron & Moore, 1997; Moore & Lee, 1994). Section 6 of this paper discusses models beyond those offered in WEKA.

4.1 Metric and Feature Selection

In order to base our estimation or classification on similar past observations, we must define "similar." This is

frequently the most important and difficult aspect in applying an instance-based model. A natural choice for similarity is to view each observation as a point in n -dimension space, where n is the number of features of each observation. Similarity can then be defined based on the distance between points using any metric, such as the Euclidean metric. There are several problems with this approach. First, without some form of normalization of the feature values, features with the largest values will dominate. The features may be normalized both as to size and dispersion, but this may not be optimal. Ideally, we would like to use training data to determine which factors are most important and to increase their importance in the similarity metric. A possible approach to this is discussed in Section 6. Without a method of automatically reducing the importance of features with little relevance, we are left with a problem related to the "curse of dimensionality." Factors with little relevance tend to increase the distance between points that are truly similar based on the factors that do matter. Without the ability to vary the weights on features within the similarity metric, we must restrict our features to those with high importance.

WEKA offers no choice of metric and uses the Euclidean metric applied to all numeric features. The feature values may be normalized either by applying a Normalization Filter before classification or by selecting normalization during classification. In either case the numeric attribute values are rescaled so that they range between 0 and 1. This is clearly better than using the raw data, which in this example can vary in size by several orders of magnitude between features. However, this form of normalization does not necessarily put all features on an equal footing. Suppose, for example, that the data of one feature has a single outlier. This form of normalization will put the outlier at either 0 or 1 and clump the other values near the other extreme. Care must be taken to eliminate outliers or use an external method to normalize data prior to importing it to WEKA.

WEKA offers a number of ways to identify the features that are most important to the task at hand. Upon selecting the AttributeSelection Filter on the Preprocessing screen, WEKA offers several Evaluators for selecting attributes. An alternative approach is to use the Linear Regression classifier to fit a linear model to the change in stock index variable and use one of the attribute selection methods offered there. The attributes chosen here have the most predictive value in a linear model estimating the change in future stock price, and so should be useful in the classification problem. This latter approach was used to limit the number of variables to the 8 with the most predictive value in a linear model. These are: GS1, MORT, INDPRO, TCU, UNRATE, SP, BUSINV_change, and TCU_change.

4.2 Number of Neighbors

The selection of the number of neighboring points to use is based primarily on evaluation of the noisiness of the data and the heterogeneity of the underlying function. If there is significant noise in the data, choice of a larger k value will

cause the noise contributions to offset and will result in a better estimation. On the other hand, including more points broadens the radius of the neighborhood and includes points that are less representative of the point being classified.

Some insight into the relative importance of these two factors can be gained by considering the problem of estimating a continuous function, f , at x_o by using a simple average of the known values of f at k -nearest neighbors, x_1, x_2, \dots, x_k , where x_i are vectors of n feature values. Assume the feature values have been normalized and the exact function values are known at m training points uniformly distributed in the n -dimensional unit square. The average radius in n -dimension Euclidean space needed to obtain k points will be that of a sphere with volume equal to k/m . Using the formula for the volume of a sphere in n -space, $(1/(n/2)!) \pi^{(n/2)} r^n$ (in the case n is even) we can solve for the average radius required. If we have an estimate of the derivative of f in the region under consideration, we can use the radius to calculate an estimate of the error caused from the heterogeneity source. This, of course, will increase with k . The error from the noise source will decrease in proportion to $1/\sqrt{k}$ based on the Central Limit Theorem. An estimate of the proportionality constant may be obtainable through repeated sampling of f at a single point. Once formulae have been obtained for the two sources of error, choose a value for k that makes the two sources close to equal.

Figure 1 illustrates this tradeoff in the sample problem. Using only a few neighbors makes for poor predictions due to the noisiness and weak correlations of the underlying data. Choosing too many neighbors sweeps in examples that are not representative of the case at hand. The choice of 25 neighbors is used in the remainder of this analysis. While there may be some bias in choosing the k for which the accuracy is highest, the graph suggests that the improved accuracy is due to the error reduction effects rather than model selection bias.

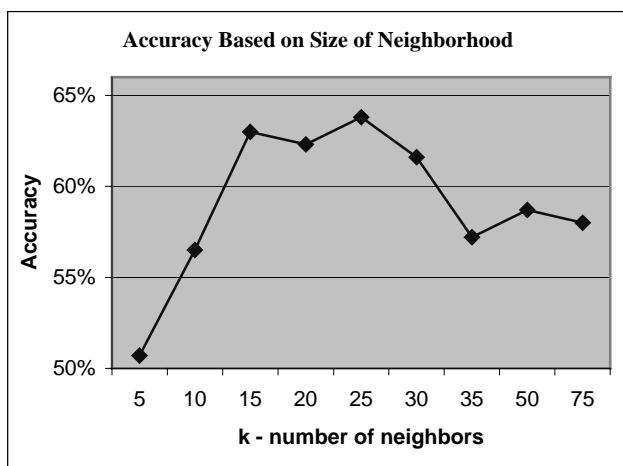


Figure 1. Accuracy rates for choices of number of neighbors.

4.3 Weighting of Neighbor Contributions

Once we have selected the nearest neighbors to the point under consideration, we must decide how to use their results

in estimating or classifying the current point. One possibility is to give equal weighting to each of the neighbors. It seems reasonable, however, to give more weight to the closer neighbors. WEKA allows three weighting choices: equal weighting, weights based on $1/d$ and weights based on $1-d$, where d is the distance from the neighbor to the point under consideration. Assuming the features have been normalized to the unit cube, the $1/d$ and $1-d$ weighting schemes produce weights in the ranges $[1/\sqrt[n]{n}, \infty]$ and $[0, 1]$ respectively, where n is the number of features. The $1/d$ weighting scheme will put virtually all the weight on a single very close point if one exists. This is not desirable in the case of noisy data as we have in the stock prediction example since we want to consider a large number of points to offset errors. The $1-d$ weighting produces the best results with the stock data.

5. Interpretation of Results

5.1 Testing Using WEKA

WEKA provides several options for testing the results of the model. You may test the model on the training data, although there is a clear bias in doing so. Alternatively, you may provide a separate dataset for testing, presumably independent of the training data, or withhold a specified portion of the data from the training and use it for testing. These approaches eliminate the bias but do so at a cost of reducing the data available for training. A third approach allowed by WEKA is "cross-validation" based on a user-selected number of "folds." The model is developed using all of the data, but for testing purposes an additional model is created for each fold by holding out a portion of the data for testing. For example, if the user indicates 10 folds, the model will be tested ten times by: 1) holding out 1/10 of the data, 2) developing a model for the remaining 9/10 of the data, and 3) testing the resulting model on the 1/10 withheld. The data withheld is selected at random from the data not yet tested, so at the conclusion all data will have been used as test data, but on models that are slightly different from the final model. This paper uses cross-validation with 10 folds to test the results of the stock model and to compare the results to alternative methods.

5.2 The Up/Down Classification Problem

In this problem we are to determine whether, based on the data available at month m , the stock index will increase by more than the median amount during the period $m+1$ to $m+2$. Based on this definition, exactly half of the data points correspond to TRUE and half to FALSE. Table 1 shows the results of using the k -nearest neighbor approaches. The base case is the approach developed in the prior sections. It is shown with modifications discussed above and using two algorithms designed to improve performance. The classification accuracy is shown along with the statistical significance at which we could reject the null hypothesis of the classification being random.

Table 1. Classification accuracies for several modifications of the k-nearest neighbor model. Base case uses 25 nearest neighbors, (1-d) metric and feature selection.

MODEL	ACCURACY
BASE CASE	63.8%
NO FEATURE SELECTION	61.6%
1/D METRIC	60.9%
WITH ADABOOST	63.8%
WITH BAGGING	63.0%

All of the models provide results that are statistically significant at low alpha levels ($<.01$). Limiting the model to a small number of key features improves the results, as does using the (1-d) weighting in the metric. Neither AdaBoost nor Bagging improved the results of the simple k-nearest neighbor classifier.

Figure 2 shows a comparison of classification accuracy among various classifiers provided in WEKA. For each, the comparison is made based on using all feature data as well as selecting the key features.

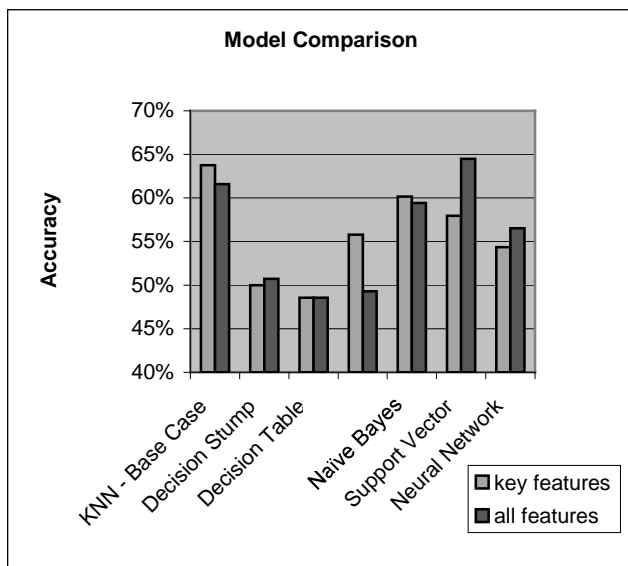


Figure 2. Accuracy comparison of alternative classifiers provided by WEKA. Default parameters were used in the models other than KNN.

The simple k-nearest neighbor approach compares favorably to the alternative WEKA classifiers on the stock classification problem. The Support Vector Machine did perform slightly better than KNN when presented with all of the features and appears to be able to avoid the problems of too many features encountered by KNN.

Overall, it appears that the k-nearest neighbor approach applied to economic indexes is able to find information helpful in predicting stock market direction. Further study is needed before concluding that this information could be used in a successful trading system, since the magnitudes of gain and loss were not considered in this analysis.

6. Areas for Future Study

6.1 Evolving Metrics and Automatic Scaling of Features

One significant limitation of the WEKA software is the lack of flexibility in defining the metric. Even with more flexibility, it is difficult to know in advance which metric to use. An area of future study is the use of training data to learn an appropriate metric. One approach is to use the Euclidean metric except to apply a weight or "scaling factor" to the square of the difference of each feature. During repeated training cycles on the same data, the weights would be gradually adjusted to improve the similarity metric. If a training example is misclassified, the weights of the features with the largest differences would be increased and weights of the features with small differences would be decreased. Suggestions along these lines have been made by Witten & Frank (1999).

6.2 Using Time in the Similarity Metric

The non-stationary nature of this problem indicates a need to incorporate the time dimension in the similarity metric. This analysis experimented with using the month sequence number as a feature, but a more explicit consideration of time may be needed (McCallum, 1995).

6.3 Stock Index Prediction

This data was also used to estimate the changes of stock index values for periods in the future, rather than the simple up/down classification. The results showed statistically significant correlations with actual changes. An area for further study is whether k-nearest neighbor methods lead to better estimates than simple linear regression. Another area to explore is whether using local linear regression on the k-nearest neighbors leads to an even better fit (Atkeson, Moore & Schaal, 1997).

References

- Atkeson, C. G., Moore, A. W., and Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review*, pp. 11-73.
- Maron, O., and Moore, A. W. (1997). The racing algorithm: model selection for lazy learners. *Artificial Intelligence Review*, pp. 193-225.
- McCallum, A. K. (1995). Instance-based utility distinctions for reinforcement learning with hidden state. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 387-395. San Francisco, CA. Morgan Kaufmann.
- Moore, A. W., and Lee, M. (1994). Efficient algorithms for minimizing cross validation error. *Proceedings of the 11th International Conference on Machine Learning*, pp. 190-198. Morgan Kaufmann.
- Witten, I. H., and Frank, I. (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.