Predicting Equity Returns from Securities Data with Minimal Rule Generation

Chidanand Apté

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, NY 10598
apte@watson.ibm.com

Se June Hong

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, NY 10598
hong@watson.ibm.com

Abstract

Based on our experiments with financial market data, we have demonstrated that the domain can be effectively modeled by classification rules induced from available historical data for the purpose of making gainful predictions for equity investments, and that new techniques developed at IBM Research, including minimal rule generation (R-MINI) and contextual feature analysis, are robust enough to consistently extract useful information from noisy domains such as financial markets. We will briefly introduce the rationale for our rule minimization technique, and the motivation for the use of contextual information in analyzing features. We will then describe our experience from several experiments with the S&P 500 data, showing the general methodology, and the results of correlations and managed investment based on classification rules generated by R-MINI. We will sketch how the rules for classifications can be effectively used for numerical prediction, and eventually to an investment policy. Both the development of robust "minimal" classification rule generation, as well as its application to the financial markets, are part of a continuing study.

Keywords
Financial Markets Data Mining, DNF Minimal Rule Generation, Contextual Feature Analysis, Predictive Performance Evaluation, Investment Portfolio Management

1 Introduction

There is currently a surge of interest in financial markets data mining. Large amounts of historical data is available for this domain in machine readable form. Analyses of this data for the purpose of abstracting and understanding market behavior, and using the abstractions for making predictions about future market movements, is being seriously explored [AI on Wall St., 1991, AI on Wall St., 1993]. Some firms have also deployed data mining analytical methods for actual investment portfolio management [Barr and Mani, 1993]. We report here on our recent experiments with applying classification rule generation to S&P 500 data.

The R-MINI rule generation system can be used for generating "minimal" classification rules from tabular data sets where one of the columns is a "class" variable and the remaining columns are "independent" features. The data set is completely discretized by a feature discretization subsystem prior to rule generation. The feature discretization performs feature ranking as well as the conversion of numerical valued features into discretized features using a optimal cutting algorithm. Once rule generation is completed, the R-MINI system can be used for classifying unseen data sets and measuring the performance using various error metrics.

The R-MINI rules are in Disjunctive Normal Form (DNF). There have been many approaches to generating DNF rules from data. These include [Michalski *et al.*, 1986, Clark and Niblett, 1989, Weiss and Indurkhya, 1993] which work in principle by iteratively forming one rule at a time to cover some examples from the training data which are removed from consideration before repeating the iteration. The other primary approach [Pagallo, 1989, Quinlan, 1993] is decision tree based, i.e., a decision tree is created that classifies the training examples, and the rules are then derived selecting and pruning the paths from the root to the leaf nodes.

While the R-MINI approach to generating classification rules is similar to the former, it differs from both approaches in its primary goal, which is to strive for a "minimal" rule set that is complete and consistent with the training data. Completeness implies that the rules cover all of the examples in the training data while consistency implies that the rules cover no counter-examples for their respective intended classes. Others too have argued for generating complete and consistent classification models before applying error minimizing pruning processes [Breiman *et al.*, 1984, Weiss and Kulikowski, 1991]. The R-MINI system goes beyond current technology in its attempt to generate "minimal" complete and consistent rules. The merits of striving for minimality have been well discussed [Blumer *et al.*, 1989, Rissanan, 1989]. Minimality of the representation will favor accuracy and interpretability.

2 Minimal Rule Generation

The R-MINI rule generation technique works with training data in which all features are categorical in nature. All numeric features are therefore discretized by a feature analysis and discretization sub-system prior to rule generation. The rule generation technique is based upon a highly successful heuristic minimization technique that was used for minimizing large switching functions (MINI [Hong *et al.*, 1974]). Similar heuristic minimization techniques have been developed for a commercially available switching function minimization package, ESPRESSO [Brayton *et al.*, 1984]. The core heuristics used in the MINI system for achieving minimality consists of iterating (for a reasonable number of rounds) over three key sub-steps:

1. Generalization step, EXPAND, which takes each rule in the current set (initially each example is a rule) and opportunistically generalizes it to remove other rules that are subsumed.

2. Specialization step, REDUCE, which takes each rule in the current set and specializes it to the most specific rule necessary to continue covering only the unique examples it covers.

3. Reformulation step, RESHAPE, which transforms a pair of rules into another pair for all pairs that can be thus transformed.

The R-MINI rule generation technique is in principle based upon this approach. Experiments to date indicate that it is quite robust in its minimality. Since the rule generation relies on iterative improvements, one can potentially use as much computing time as is affordable. In practice, we have observed that R-MINI starts converging in 5-6 iterations on most well known test data sets as well as on some of the specific real applications in which we have been using the system.

R-MINI has been applied to several real data sets, up to those with a few hundred features and tens of thousands of examples. Preliminary evaluations suggest that complete and consistent full cover rule sets that result from applying other known techniques can be several times larger. Initial

benchmarking studies have also indicated that the predictive power of R-MINI's rule sets is always ahead of the "best" DNF rule sets generated by other well known methods. An in-depth detailed discussion of the rule generation component of R-MINI appears in [Hong, 1993].

3 Contextual Feature Analysis

As mentioned in the previous section, R-MINI rule generation requires all features to be in categorical form, and hence the reason for discretizing all numerical features employing a feature analysis and discretization sub-system. There is also another important reason for applying this step prior to rule generation. Classification model generators will typically work only as well as the quality of the features from which they are trying to generate a model. Poor features will almost always result in weakly performing classification models.

Various approaches have been used to alleviate this problem. The decision tree based methods have relied on information theoretic measures (such as the "entropy" and "gini" functions) to determine the best feature to use at each node while expanding the decision tree [Breiman *et al.*, 1984]. This basic principle may be thought of as a 1-level lookahead algorithm that determines the best feature to use at a node based on how well the feature partitions the training examples into their respective classes. Variants of this method include 2-level and more lookahead methods as well as employing simple conjuncts of features (instead of single features) as decision tests for nodes.

One well known method used in a DNF rule generator is backtracking based local optimization [Weiss and Indurkhya, 1993]. This method works in principle by attempting to constantly improve the performance of a rule while being constructed by swapping member tests (features) with new tests. Although this method appears more powerful than the decision tree methods, it may not perform well in the presence of extremely large numbers of features.

The R-MINI system employs a contextual feature analyzer that simultaneously ranks the features in terms of their classificatory power as well as determining the "optimal" number of cuts for each numerical feature for discretization so as to maximize that feature's ability to discriminate. Features are ranked based upon merits that are computed for each of them. Merits are computed by taking for each example in a class a set of "best" counter examples, and accumulating a figure for each feature that is a function of the example-pair feature values. Dynamic programming is then used for producing optimum cuts [Aggarwal *et al.*, 1993] for the numeric variables by simultaneously looking at all numerical features and their value spans. This process is iteratively repeated until a reasonable level of convergence emerges in the merits and the cut values. In comparison to the tree based methods, the R-MINI contextual feature analyzer may be thought of as a full-level lookahead feature analyzer. It will not suffer from falling into false local minima because of its ability to analyze merits of features in a global context. An in-depth discussion of contextual feature analysis appears in [Hong, 1994].

4 Experiments with S&P 500 Data

In cooperation with the IBM Retirement Fund group, we are undertaking a study to determine the feasibility of applying DNF rule generation technology to managing equity investment portfolios. Initial results appear quite promising.

All our experiments have been conducted with S&P 500 data, for a contiguous period of 78 months. The data spans 774 securities (S&P deletes and adds new securities to its 500 index over time, so that the index continues to reflect the true market capitalization of large cap. firms). The data comprises of 40 variables for each month for each security. The type of information conveyed through these variables is both fundamental (company performance data) as well as technical (stock performance data). Some of the variables provide trend information (ranging from month-to-month trends to 5-yearly trends). With the exception of one variable, the industry sector identifier, which is categorical, all the rest are numerical.

Also available for each monthly stream of data for a security is the monthly total return for that security, where the variables values are all at the beginning of a month while the monthly total return (stock change + dividends) is at the end of the month. From this 1-month return variable, one can compute 3-month, 6-month, as well as 12-month returns, for each security for each month. One can also compute the difference between these returns and the capitalization weighted mean as well as simple mean for each of the returns. Thus, if one can envision the basic 40 variable set as the "features", then we have available several ways to assign classes to each of the examples (a monthly stream of feature values for a security) by picking from one of the computed returns.

We have conducted a series of compute-intensive classification experiments with this data, using different ways to assign class labels as well as different ways to partition the data. We will focus in the rest of this paper on one particular study, which attempts to generate rules for classifying examples based upon the differential between monthly return and simple mean of monthly returns for a given stream of data. The idea here is to use these rules to predict the differential for unseen data for the following year(s) and utilize the predictions in a portfolio management scheme for maximizing the investment returns. The portfolio management strategy strives to remain constantly above the average market return, and therefore the use of the differential as a class label. A positive differential merely implies a return that is higher than the market average. The actual return could be positive or negative.

4.1 Generating Classification Rules for Equity Returns

Class	Returns	Year 1	Year 2	Year 3	Year 4
C0	< -6	880	857	559	674
C1	≥ -6 & < -2	1101	997	1188	936
C2	≥ -2 & < 2	1347	1180	1533	1295
C3	≥ 2 & < 6	874	883	977	1015
C4	≥ 6	699	808	560	847

Table 1: Number of S&P 500 data examples per class for years 1-4

There are several issues at hand for determining how much data to choose for generating DNF classification rules for this domain. A routinely used approach would be to hide a portion of the data, ranging from 10-30%, and generate rules from the remaining "training" data, and evaluate their performance on the hidden "test" data. However, that approach is not adequate for the financial markets domain. There is a strong time-dependent behavior in the securities market which needs to be accounted for. An accepted practice is to use the "sliding window" approach, in

which the data is laid out in temporal sequence, and the classification generation and performance evaluation experiments are repeatedly performed on successive sets of training and test data. This method can be used for determining whether the performance of a particular approach withstands the time-dependent variations that are encountered as one moves from set to set.

Adopting this latter methodology in on of our experiments, we chose to generate classification rules from a consecutive 12 months of data, and tested the performance of those rules on the following sets of 12 month streams. The idea here was to evaluate the rate of decline in the predictive power of classification rules as one moved forward in time. Once this rate is known, one can establish a policy of re-generating the rules once every "n" years from the immediate past data so as to continue holding up the predictive performance. Our data provided us with over 6 consecutive streams of 12 month data. We are conducting our experiments from the earliest point onwards, i.e., generate classification rules from the earliest available 12 month data (year 1), apply those rules to year 2, year 3, etc. until the performance becomes unacceptable, say at year "n". Then re-generate classification rules from the 12-month data for year "n-1", and repeat the process.

For the class label, we chose the differential between the next month's 1-month total return and the S&P 500 average 1-month return. This label is essentially a numerical valued assignment. We further discretized this assignment by attempting to emulate typical security analysts' categorization method for stocks, which would include the range "strongly performing", "moderately performing", "neutral", "moderately underperforming", and "strongly underperforming". Based upon preliminary analysis of the data distribution, we chose to assign the cut-point -6, -2, +2, and +6. That is, all examples who had a class label value of 6% or more were put in one class (the "strongly performing" class), all examples with class label values of 2% or more and less than 6% were put in another class (the "moderately performing" class) and so on. Using this class partitioning scheme, Table 4.1 illustrates how the first 4 years of data break up by class. Note that although 12 months worth of data for 500 securities should translate to about 6000 examples, the actuals vary for each time period because we chose to discard examples which had one or more missing values for the 40 features. The actual examples that we worked with are 4901 for year 1, 4725 for year 2, 4817 for year 3, and 4767 for year 4.

Before proceeding to apply R-MINI's feature analysis and discretization step, we tried to carefully adjust the features based upon discussions with the domain experts. As we pointed out in the previous section, the quality of features is of extreme importance in ensuring the quality of the generated classification model. Some of our adjustments to the raw data included the normalization of some features and the inclusion of additional trend indicating features. This preprocessing usually results in the transformation of input raw features into a new set of features, and cannot be done in the absence of domain experts. However, if this expertise is available, then utilizing it to refine the features is always very desirable. Once these transformations were made, we applied R-MINI's feature discretization step to the data for Year 1, which corresponds to 4901 examples. The result of this step is the assignment of merits to all the features and the assignment of cut points to the numerical features. Table 4.1 illustrates the merit and cut assignment for this experiment. Note that features with a 0 value for cutpoints indicates that the feature is categorical. Also, we chose the merit values to discard certain features from the rule generation step. These features appear the lower end of the table. Their cut points are not important, since they do not play a subsequent role in the classification experiments, and are therefore not indicated.

Using the selected features and fully discretized data values, we then apply R-MINI's rule generation step to the training data, which is now essentially a 5-class problem with 4901 examples

Feature	Merit	Cut Points
ret1by1m	322	5
monthr12	277	6
pr2	230	4
ret1mret1	229	4
ret1	228	4
pr1	223	4
pr3	216	4
pr6	201	4
valueprice	197	3
veer	152	1
cflowprice	131	1
earntr	122	2
epsprice	121	2
bookprice	115	1
eps5	112	2
aprice	111	1
hsg	110	2
per	105	1
cap	105	1
beta	105	1
eps12price	100	2
roe	94	1
rinveq	91	2
sects	90	0
fund	78	2
peg	77	1
odl	67	2
yld	66	1
salshr	61	4
epsvar	61	1
das	58	xx
derat	52	xx
gprat	48	xx
qualty	42	xx
ccr	40	xx
ass	40	xx
growth	29	xx
cne	21	xx

Table 2: Feature Merits and Cut Points for Year 1 Data

and 30 features. Since R-MINI uses a randomization process in its minimization phase, we run R-MINI several times (typically 5-6) on the same data set, and go with the smallest rule set that was generated. In this particular case, the smallest rule set size was 569. That is, 569 rules completely and consistently classified the 4901 training examples. Table 4.1 illustrate just 2 of these rules, where the first rule, Rule 1, is for Class 0, which corresponds to "strongly underperforming" and the second rule, Rule 481, is for Class 4, which corresponds to "strongly performing".

4.1.1 Rule-based Regression

To be able to precisely quantify the predictive performance, especially from an investment management point of view, it is necessary that the classification rules predict the actual return, and not the discretized class segments. We have developed a metric for assigning numeric predictions for the R-MINI classification rules. While primarily motivated by the current set of experiments,

```
Rule 1:
    monthr12: NOT ( 5.50 ≤ X < 9.50; )
    aprice: ( X < 43.19; )
    beta: ( X < 1.10; )
    epsprice: NOT ( 0.05 ≤ X < 0.06; )
    eps5: ( 3.98 ≤ X )
    peg: ( 1.54 ≤ X )
    pr3: NOT ( 0.93 ≤ X < 1.01; 1.07 ≤ X < 1.17; )
    valueprice: ( 0.86 ≤ X )
    veer: ( -2.37 ≤ X )
    ret1mret1: ( -10.85 ≤ X )
    ret1by1m: ( 1.03 ≤ X )
Then ⟹ C0

Rule 481:
    beta: ( 1.10 ≤ X )
    cap: ( X < 2743.50; )
    epsvar: ( 6.25 ≤ X )
    hsg: NOT ( 4.89 ≤ X < 9.45; )
    peg: ( X < 1.54; )
    pr1: ( X < 1.10; )
    pr2: NOT ( 1.06 ≤ X < 1.14; )
    pr3: NOT ( 0.93 ≤ X < 1.01; )
    pr6: NOT ( 0.91 ≤ X < 1.02; )
    rinveq: NOT ( 5.63 ≤ X < 10.64; )
    valueprice: NOT ( 0.70 ≤ X < 0.86; )
    veer: NOT ( -2.37 ≤ X < 0.84; )
    ret1mret1: NOT ( -10.85 ≤ X < -4.05; )
    ret1by1m: ( X < 1.03; 1.06 ≤ X < 1.06; )
Then ⟹ C4
```

Table 3: Examples of R-MINI Classification Rules Generated from Year 1 Data

it is conceivable that this approach could be used in any domain where it is required to predict numerical values. In a sense, this metric extends our R-MINI classification system for applications in non-linear multi-variate regression.

What we do is associate with each rule three parameters; μ, the mean of all actual class values (in this case, the differential between 1-month total return and mean S&P 500 1-month total return) of training examples covered by that rule; σ, the standard deviation of these values; and N, the total number of training examples covered by that rule.

When a rule set of this nature is applied to hidden "test" data, we have two metrics for predicting the numeric class value for each example in the test data. In the simple averaging approach, we compute for each example the simple average of μ of all rules that cover it as its predicted value (assigning a prediction of 0.0 if no rules cover it). In the weighted average approach, we compute and assign a prediction of the weighted average of $\frac{\sqrt{N}}{\sigma}\mu$ of all rules that cover it (assigning a prediction of 0.0 if no rules cover it). In general, we have noticed that the weighted average approach leads to smoother correlations between predicted and actual values.

4.2 Investment Portfolio Management with R-MINI Rules

To evaluate the performance of the generated rules, we applied them to subsequent year data. For example, rules generated from data for months 1-12, after some minimal pruning, are applied to

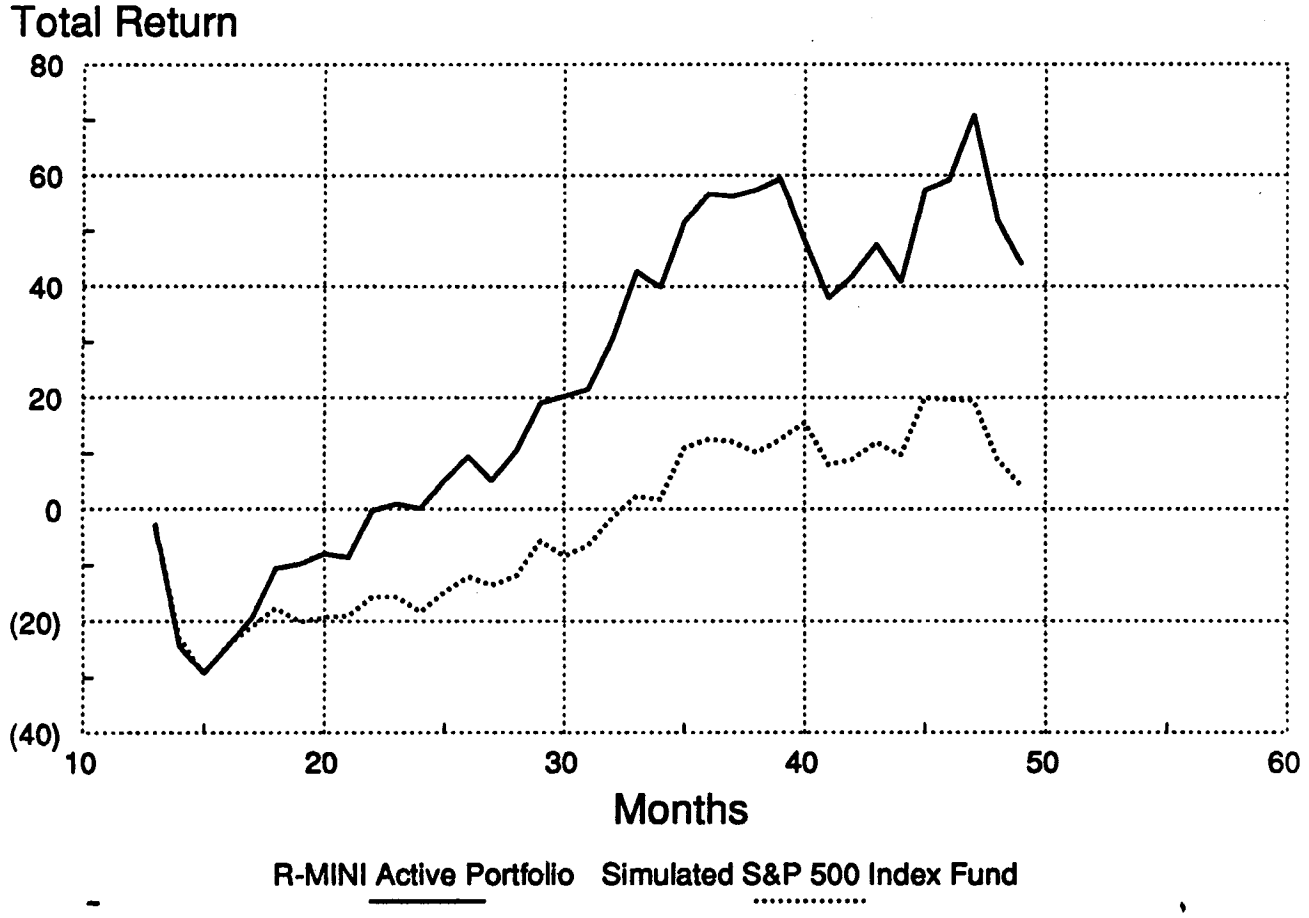

Figure 1: Comparing Total Returns of Simulated S&P 500 Index Fund Portfolio and R-MINI Rule Based Active Portfolio

the the following two years of data. We pruned out rules that cover 3 or less examples, since they are assumed to be covering the noise component in the data. For the remaining set of rules, we computed the μ, σ, and N for each rule, and then applied them to data for months 13-48.

For effective comparison of how these rules would perform if realistically used, we constructed a portfolio management scheme based upon these rules, and compared it to a simulated S&P 500 index fund performance. A index fund is passive in nature, and all that a S&P 500 index fund does is to constantly try and reflect the make up of the S&P 500 index, by investing in those companies in proportion of their capitalization[1]. In contrast to this passively managed approach, a portfolio management scheme based upon R-MINI rules will need to be highly active, since every month the rules will be making predictions which will need to be acted upon. What this active management policy does in principle is to start out with a investment which reflects the S&P 500 index fund,

[1]The simulation of the passive index fund was performed on only that subset of the available S&P 500 that did not have missing information, i.e., the data in Table 4.1. This simulation may not correspond exactly to the real S&P 500 performance, although it is very close.

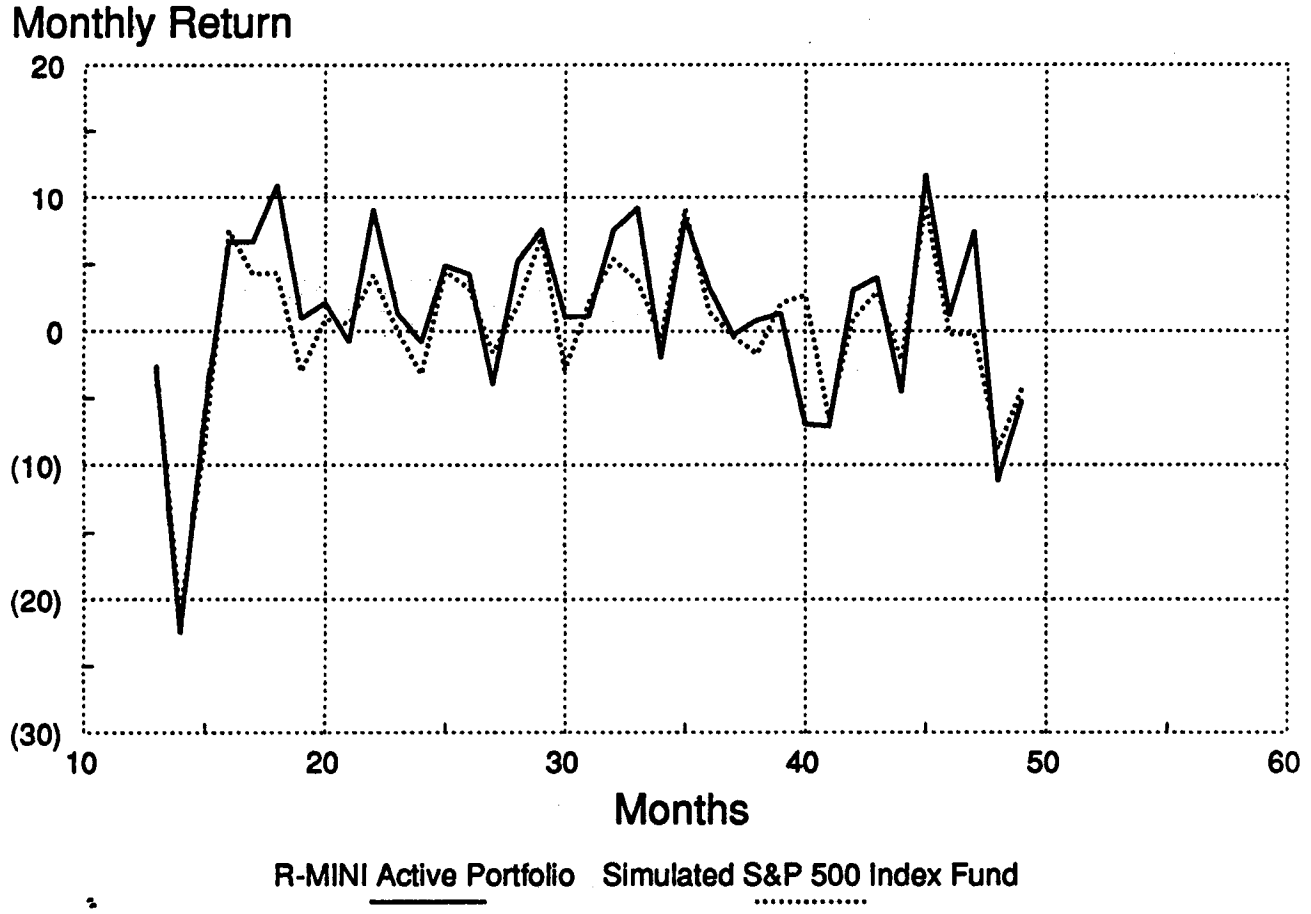

Figure 2: Comparing Monthly Returns of Simulated S&P 500 Index Fund Portfolio and R-MINI Rule Based Active portfolio

but then make trades every month based upon the rule predictions. One strategy that we have shown to be successful is as follows:

1. Generate rules (and use only those that cover > 3 examples).
2. Start with $1 million S&P 500 index portfolio.
3. Execute monthly action at the end of month as follows:
   (a) Update portfolio value for each equity position based upon the month's actual total return for that equity.
   (b) Apply rules to month-end data for making predictions.
   (c) Sort predictions in descending order.
   (d) Sell bottom 50% of sorted list provided the values are less than 4%.

(e) Buy top 5% of sorted list provided the values are greater than 4%, in equal amounts using funds available from the sale.

The buy and sell cutoff points and thresholds (50%, 5%, 4%, 4%) are parameters that can be adjusted for controlling the behavior of the portfolio. For example, they can be adjusted to make the portfolio "aggressive" or "conservative". Aggressive portfolios are characterized by high turnover and large positions in limited equities. Conservative portfolios hold relatively larger number of equities and trade less. The buy/sell cutoff points and thresholds for our investment portfolio simulator can be varied to achieve different behaviors on this spectrum.

For the above settings, Figures 1 and 2 illustrate how our active portfolio performs against a passive portfolio, on a monthly basis as well as a cumulative basis. We can see that the active management portfolio returned 44% return for months 13-48, as compared to a 4% return using a passive indexed portfolio. On the other hand, we can also see that the active management portfolio returned 56% return for months 13-36, as compared to a 12% return using a passive indexed portfolio. One could argue that the rules that were generated from data for months 1-12 held up well for months 13-36, but started weakening out thereafter. One therefore will need re-generation of rules from data for months up to 36 for applying to months 37 onwards, and repeating the process at the end of every two years. Our full performance evaluation exercise therefore will be done using the regime of generating rules for years 1, 3, and 5. The rules from year 1 will be used for making predictions for years 2-3, year 3 rules will be used for predicting returns for years 4-5, and year 5 rules will be used for years 6 and beyond.

Although we make a few simplifying assumptions when constructing the portfolio management simulations, we feel that they will not adversely affect the comparison. For example, we ignored the issue of fees and transaction costs of trading securities. However, these costs apply both to the passive as well as active portfolios. We expect that with the inclusion of such costs, the narrowing of the gap between the two will be a function of the portfolio settings. For a "conservative" setting, the gap may narrow, while it may still remain sizable for an "aggressive" setting. To be realistic and take all such factors into consideration, we are developing a more powerful portfolio management strategy that will compensate for additional constraining factors such as these.

5 Discussion

We can draw two key conclusions based upon our experiments. First, the S&P 500 data, as characterized by the features illustrated in Table 2, seem to provide adequate information for useful classification rule generation. Second, our techniques and methodology have the ability to extract this information from what is well known to be noise prone data.

The application of DNF classification rules in non-linear multi-variate regression applications is in itself another interesting direction to explore. The advantages of using DNF rules for these applications is clear; they provide a superior level of representation and interpretability in contrast to black-box style mathematical functions. Expert analysts can examine and understand these rules, and potentially even hand-edit them for improved performance.

We have demonstrated the predictive power of R-MINI's minimal rule generation philosophy in conjunction with its contextual feature analysis. We have observed that the R-MINI generated rules, when embedded in an appropriate portfolio management scheme, can outstrip passive index funds in performance.

We are also beginning to gain insight into temporal longevity of classification rules generated from historical data in the financial markets. Our initial experiments have clearly illustrated the nature of decay in predictive performance as one goes out further into time. We are nearing the completion of our "sliding window" rule generation and performance evaluation, and the promising results continue to hold. We have begun to explore options for embedding our methodology into an actual deployment.

References

[Aggarwal *et al.*, 1993] A. Aggarwal, B. Schieber, and T. Tokuyama. Finding a Minimum Weight K-link Path in Graphs with Monge Property and Applications. Technical report, IBM Research Division, 1993.

[AI on Wall St., 1991] *Artificial Intelligence Applications on Wall Street.* IEEE Computer Society, 1991.

[AI on Wall St., 1993] *Artificial Intelligence Applications on Wall Street.* Software Engineering Press, 1993.

[Barr and Mani, 1993] D. Barr and G. Mani. Neural Nets in Investment Management: Multiple Uses. In *Artificial Intelligence Applications on Wall Street*, pages 81–87, 1993.

[Blumer *et al.*, 1989] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension. *JACM*, 36:929–965, 1989.

[Brayton *et al.*, 1984] R. Brayton, G. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli. *Logic Minimization Algorithms for VLSI Synthesis.* Kluwer Academic Publishers, 1984.

[Breiman *et al.*, 1984] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth, Monterrey, CA., 1984.

[Clark and Niblett, 1989] P. Clark and T. Niblett. The CN2 Induction Algorithm. *Machine Learning*, 3:261–283, 1989.

[Hong *et al.*, 1974] S.J. Hong, R. Cain, and D. Ostapko. MINI: A Heuristic Algorithm for Two-Level Logic Minimization. *IBM Journal of Research and Development*, 18(5):443–458, September 1974.

[Hong, 1993] S.J. Hong. R-MINI: A Heuristic Algorithm for Generating Minimal Rules from Examples. Technical Report RC 19145, IBM Research Division, 1993.

[Hong, 1994] S.J. Hong. Use of Contextual Information for Feature Ranking and Discretization. Manuscript in preparation, 1994.

[Michalski *et al.*, 1986] R. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. In *Proceedings of the AAAI-86*, pages 1041–1045, 1986.

[Pagallo, 1989] G. Pagallo. Learning DNF by Decision Trees. In *Proceedings of the Eleventh IJCAI*, pages 639–644, 1989.

[Quinlan, 1993] J.R. Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[Rissanan, 1989] J. Rissanan. Stochastic Complexity in Statistical Inquiry. *World Scientific Series in Computer Science*, 15, 1989.

[Weiss and Indurkhya, 1993] S. Weiss and N. Indurkhya. Optimized Rule Induction. *IEEE EXPERT*, 8(6):61–69, December 1993.

[Weiss and Kulikowski, 1991] S.M. Weiss and C.A. Kulikowski. *Computer Systems That Learn.* Morgan Kaufmann, 1991.