A Profitable Approach to Security Analysis Using Machine Learning: An Application to the Prediction of Market Behavior Following Earnings Reports

I. Background

Four times per year, companies release earnings reports. Often the day following such reports, the stocks of these companies make significant moves either up or down from the resulting information. Such moves are often attributed to a large surprise between the analyst estimates for the stock and the actual earnings amounts. Unfortunately for the average trader, these jumps usually occur in a very short window of time following the earnings release, making it difficult to profit on either bullish or bearish earnings. Since earnings reports are not usually released during market hours, a method for determining which stocks will continue to move either up or down throughout the day would allow a trader to create a strategy that takes actual earnings information into account without being limited by this extremely short window of opportunity. Further, with the hundreds of earnings reports that are released each day, this type of method could parse and interpret an enormous amount of information—far beyond the capability of a single person. This study endeavors to create such a method that could utilize publicly available stock data in order to make trading recommendations for the entire day following the release of earnings reports.

The traditional method that is used to measure the strength of earnings reports is earnings per share (EPS)—or the ratio of a company's profit to the number of outstanding shares of its common stock [5]. *Outstanding shares of common stock* simply represent the number of shares of a company that are held publicly [5]. Many fundamental financial analysts often give their own predications for what a given company's EPS will be in any given quarter, and as such, data concerning EPS estimates and actual realizations are much less sparse than other measurements of earnings strength (one such possibility could be company revenue). In addition, EPS has an intuitive interpretation which makes its use easy—EPS simply represents the average dollar

amount of total gain or loss of publicly owned stock during a specified time period. Thus, this project will attempt to learn patterns of market behavior following EPS news reports by examining behavior of past data. Further, this newly learned information will be used to create a machine-learning based algorithm for trading following earnings reports. In turn, this discovery will provide a foundation for algorithms that can process the information of hundreds of earnings reports each day in a meaningful manner.

II. Data Collection

The first step of the project was to determine an appropriate data set. Earnings reports are released quarterly, so each observation was chosen to describe the earnings release for a single stock. In addition, for each observation, the only attributes that were considered were required to be freely and publicly available on the internet. To reduce variability in the observations, the data only include those reports that released just quarterly earnings and not annual earnings (this could be relaxed for future research). The actual attributes that were chosen to describe a single observation in the data set are shown in Appendix A. Past observations containing all of the attributes listed in Appendix A were obtained by matching data from the Thompson Reuters I/B/E/S and Compustat Databases—both available freely to the MIT community through Wharton Research Data Services [10]. The I/B/E/S database contained all information about past analyst EPS estimates and actual EPS realizations necessary for computing items 5-7 in Appendix A. The Compustat database contained the close, high, low, and open prices for each stock by day, as well as daily volume (number of stocks traded) and shares outstanding, used to calculate items 8-19 and 22-24, while items 20-21 required both databases.

Many of the attributes were dependent upon the number of shares available for any given stock. Unfortunately, the number of shares can be chosen arbitrarily for an initial public offering, meaning that many attributes in the aforementioned data set did not have a standard basis for comparison across the observations. Even though the number of shares can be chosen arbitrarily, the market capitalization of a stock is absolute. Thus, any attribute that is expressed as some ratio "per share" was transformed into a ratio "per dollar of Market Capitalization" by simply dividing the current values by the $ClosePrior$ value in Appendix A, thus fixing the problem of standardization. For example, suppose that two different observations are being compared, one for stock A and the other for stock B. Let stock A have a $ClosePrior$ value of \$50 and a $MarketCap$ value of \$100, and stock B a $ClosePrior$ value of \$25 but a $MarketCap$ value of \$200. Suppose that the actual realized EPS for these observations were \$2/share for stock A and \$1/share for stock B. By definition, market capitalization is the number of outstanding shares multiplied by the price per share [5], meaning that stock A has 2 shares, and stock B has 8 shares. Thus, stock A has a total earnings (profit) of \$4, and stock B has a total earnings of \$8, both of which are 4% of their respective market capitalizations.

This example shows that between stocks, differences in EPS do not imply differences in profit or market capitalization. In fact, the percentage increase of total assets in terms of market capitalization is the true desired measure of absolute earnings. Using the definition of market capitalization, it is easy to see that this value is simply the EPS divided by $ClosePrior$ (which returns 4% for both A and B). This idea generalizes for all of the other "per share" ratios (items 5, 6, and 8-10 in Appendix A), which allows for complete standardization by simply dividing these attributes by $ClosePrior$. Volume also suffers from the arbitrary number of shares, but has units of shares/day rather than dollars/share. Thus, to standardization of volume involves

multiplying by $ClosePrior$ rather than dividing, giving the total absolute number of dollars traded per day, as well as standard deviation in dollars, for that stock (this must be done for items 15-18). By performing these transformations, the attributes can be compared on an absolute basis between observations.

III. Methodology

The first objective was to use the completely cleaned, standardized dataset to predict the values of $ReturnPost$ for out-of-sample observations. Successful completion of this objective would immediately open the doors for many successful trading strategies. Naturally, this can be framed as a regression problem, where the dependent variable $ReturnPost$ is regressed on the independent variables 1-21 in Appendix A. However, regression requires real-valued independent variables, so the categorical attributes in Appendix A were first converted to binary attributes before running regression. In addition, there were a total of 91,726 collected observations of quarterly earnings release data, which meant that each of the 13 sectors had about 7,056 observations—more than enough data to allow separate regression analysis by sector. Another added benefit of splitting sectors is that the industries within each sector are unique. Thus, controlling by sector vastly reduces the number of possible industries in each of the 13 data sets from over 200 to around 15-40 per data set. For simplicity, the data and analysis for this project only come from the Capital Goods sector.

For this regression, the methods employed included Classification and Regression Trees (CART), Random Forests, additive nonparametric regression, and backpropagation neural network regression. The reader is referred to [11], [2], [3], and [4], respectively, to obtain background pertaining to these methods. Due to the large number of binary variables and

nonlinearities in the data, multiple linear regression and its variants (such as Ridge Regression) were not considered.

For any given sector, each method was given the same training and testing data. The training set for each sector was created by randomly selecting 70% of the data in a manner that kept approximately the same distribution of $ReturnPost$ values in both the training and test sets. All models were implemented in R using packages rpart (CART) [8], randomForest (Random Forests) [1], mgcv (additive nonparametric regression) [9], and nnet (backpropagation neural networks) [7]. Each algorithm was trained carefully to reduce the chance of overfitting. For CART, this involved varying the complexity parameter in R to be one of 0.00005, 0.0001, 0.0005, 0.001, 0.005, and 0.5 (see [8] for information about the complexity parameter). For Random Forests, the number of decision trees was set to 500 to make sure that every input observation was predicted enough times by multiple trees, which helps to prevent overfitting.

Additive nonparametric regression was implemented using the package *mgcv* in R, which has a built-in method called "generalized cross-validation" that helps to decrease the chance of overfitting [3]. Thus, no extra controls were implemented for this method. The next method, backpropagation neural networks, could have the problem of "memorizing" the data without learning general patterns if either the number of nodes in the hidden layer or the number of learning iterations were too large. However, these parameters must also be large enough to allow the network to learn patters from the data. Thus, the number of nodes in the hidden layer were varied from 3 to 15, and the number of learning iterations was varied from 10 to 100, with the best model being chosen via 10-fold cross validation on the training data using the package *e1071* in R (see [4] for neural network theory).

Unfortunately, these regressions performed poorly. As a result, another easier problem was developed to accomplish the high-level goal of finding a profitable trading strategy. This involved creating a classifier that predicted the general direction of price movement for a stock during the session following earnings releases. However, it was not immediately clear how to define the "general direction of price movement." This ambiguity stemmed from the fact that in the absence of a good regression price predictor, different trading strategies motivated creation of different classification levels. While there were many possible strategies from which to choose, four simple strategies were constructed as motivation for classifiers in a fashion that brought to light a few important insights about the data. These strategies were compared against a basic naïve strategy to illustrate the improvement gained by using machine learning techniques. These are described here as strategies (1)-(4) and "naïve" (variables in italics are defined in Appendix A).

(1) The first strategy considered was to purchase on open and sell on close. Intuitively, this strategy motivated a classifier which labeled "1" if the closing price is predicted to be above some threshold value a (i.e. $ReturnPost \geq a$), and "-1" otherwise.

(2) Another slightly more complicated strategy involved setting both a limit sell order and a stop sell order at specified threshold values. This ensured that the stock was sold if either a desired amount of gain or a maximum acceptable loss were attained. In the event that neither level was reached, the stock was assumed to be sold on close. This strategy motivated a classifier which labeled "1" if $MaxReturnPost \geq a$ and $MinReturnPost \geq b$ for some real thresholds $a > 0, b < 0$ (recall that $MinReturnPost$ is non-positive), and "-1" otherwise.

(3) A variant of the same strategy that was considered was to only have a stop sell order but no limit order, so that stock would be sold before close only if the maximum allowable loss was attained during the day. This strategy motivated the same classifier as used in (2) because an investor would want to be confident that his/her choices would likely have a high maximum possible return with a minimum return that was no worse than some acceptable loss.

(4) The final motivational strategy was derived from a "long straddle" in options trading, which benefits when the underlying stock price moves a large amount in either direction. In this case, a sensible classification would be to give a label of "1" if $MaxReturnPost > 0.5$ or $MinReturnPost < 0.5$, and "-1" otherwise. (Note that the threshold values were preset in this strategy because option pricing data was not a part of this dataset, so an appropriate objective function that maximized profit could not be constructed.)

(5) (naïve) This strategy simply assumed that a trader would buy if the actual EPS from the earnings report was greater than the consensus mean analyst EPS estimate. Thus, its corresponding classifier labeled "1" when $Actual > MeanEstimate$ and "-1" otherwise. This was provided to illustrate the benefit of using machine-learning based classification methods over simple, non-technical strategies.

This method of choosing classifiers was named *strategy-based classification*. For the sake of brevity, the analysis of (1)-(3) was performed on the assumption of buying long at open rather than selling short, although the methodology that would be employed in the latter case is completely analogous to the former. The analysis was performed by maintaining an equal proportion of both classes in the training and testing sets, so the separation of the data differed

slightly for the different classifiers. Though this introduced a small amount of variability into the problem, it ensured that more appropriate classifiers were developed. The threshold values for strategies (1)-(3) were then chosen as follows. Let X_k be the set of observations x in the test data such that the classifier associated with strategy k assigned a label of "1" to x, and let r_x be the return obtained if the stock associated with observation x was purchased on open and sold according to strategy k during the day following the earnings release, for all $x \in X_k$. Then the threshold values a or a, b (depending on the strategy) were chosen so that the resulting classifiers maximized the objective function $\prod_{x \in X_k} r_x$, representing the total return that would be obtained by investing all money sequentially in each of the observations during the sessions following the releases of earnings. This objective function worked well because it appropriately modeled the compromise between having a high geometric average return and creating a classifier that offered a decent amount of trading opportunities (recall that trading only occurred if an observation had a predicted label of "1").

Comparison of the results of the various classifiers was then used to provide valuable insights into the structure of price movements following earnings releases. This contribution aided in determining which of the aforementioned strategies was best, assuming the values of a and b created near-optimal evaluations of the objective function. It is important to note that the goals here were deeper than simply finding the most appropriate classification algorithm for this data. Also, fitting the values of a and b required constructing a very large amount of classifiers. Thus, Random Forests were chosen to perform all of the classifications as they have been shown to work very well on large datasets (on par with boosting) while also being very computationally efficient [2].

Following this analysis, some of the insights obtained suggested that regression on a subset of the original data set could be more effective than the regression that was previously performed on the entire data set (see analysis and results section). To test this hypothesis, the training set was divided into two sets of observations based on whether or not $ReturnPost \geq 0.01$. In each set, Random Forest regression was applied to train a predictive model for $ReturnPost$. For the test set, the classifier for strategy (1) was then applied to predict whether or not $ReturnPost \geq 0.01$. For those that were predicted to have $ReturnPost \geq 0.01$, the first regression model trained on observations that actually had $ReturnPost \geq 0.01$ was applied, otherwise the second regression model was utilized. For this analysis, Random Forests were the only regression method considered, although future research should include other methods as well. The reason for this was similar to why Random Forests were also used for the classifications—the purpose of this analysis was simply to see if this form of regression could provide any sort of benefit. Thus, time was spent in testing this method rather than comparing the different regression algorithms. In addition, an added benefit of Random Forests was that they did very well at finding nonlinear trends in the data without overfitting because they classified based on linear combinations of the underlying decision-tree classifications.

IV. Analysis and Results

The first objective, trying to predict the values of $ReturnPost$ from the attributes, did not yield quality results for any of the models considered. There were four measures of performance that were employed to analyze these methods. The first of these, least squares error of the test set residuals (LSE), provided a relative measure of the performance between algorithms. To gain insight into the absolute performance of each method, the actual and

predicted values of the test set were compared in a scatterplot (see Figure B.1 in Appendix B for one example of these scatterplots, done for additive nonparametric regression). In such a plot, a good predictor should have most of its data points clustered along the $y = x$ line through the origin. To see if this was the case, the actual $ReturnPost$ values were regressed on the predicted $ReturnPost$ using linear regression, and the resulting R^2, slope, and intercept values were reported in table 1. These provide absolute measures of performance where a good predictor should have this R^2 value close to 1, a slope of approximately 1, and an intercept through the origin.

Method	Best LSE on test set	R^2	Slope	Intercept
CART	9.315226	Not defined (best tree had same prediction for all data)	Not defined (best tree had same prediction for all data)	Not defined (best tree had same prediction for all data)
Random Forests	7.304142	0.05453	1.258	-0.000251
Additive Nonparametric Regression	4.423454	0.0001189	0.0612	0.03258
Backpropagation Neural Networks	9.315226	Not defined (best network had same prediction for all data)	Not defined (best network had same prediction for all data)	Not defined (best tree had same prediction for all data)

Table 1—Results of nonlinear regression methods when predicting $ReturnPost$

Looking at table 1, there were two methods for which the best model in terms of LSE simply predicted a constant value for each input observation—CART and backpropagation neural networks. Thus, no unique regression line could be drawn for these, although clearly predicting the same value for any observation was not a good predictor. In addition, though Random Forests and additive nonparametric regression do not predict a single constant value of

$ReturnPost$ for any observation, the statistics of the linear regression of actual versus predicted showed that these methods also performed poorly. Both of these latter methods had R^2 values well below 0.1 which meant that the predictors did very little to explain the behavior of the actual $ReturnPost$ values. Also, both had slope and intercept values that were clearly different than 1 and 0, respectively, when considering that the $ReturnPost$ values have magnitudes on average around 0 to 0.06. These statistics suggested that the data may have had such a large amount of noise that quality predictions were not possible.

Despite the poor performance of the regression methods for determining $ReturnPost$, the strategy-based classification methods were much more successful at capturing patterns in the data. Recalling the structure of these methods, a value of "1" meant that the stock was expected to perform in a manner that was likely to make the strategy profitable, and "-1" in a manner that was likely to be unprofitable. These methods were compared on four measures of performance—percent of the test data classified correctly, percent predicted to be "1," percent of those predicted to be "1" that were correct, and percent of those predicted to be "-1" that were correct. Of these, the most important two were the percent predicted to be "1" and the percent of those predicted to be "1" that were correct. This was due to the construction of the classifier, which was designed to indicate "1" when there was high confidence that its respective strategy would work. Thus, more "1" values meant more opportunities to trade. This also implied that an error of predicting a "1" instead of a "-1" was very bad, for this implied that a loss would be incurred. The results of this comparison, as well as the threshold values which maximized the objective function $\prod_{x \in X_k} r_x$ on the test set (see Methodology), are summarized in table 2.

Strategy	% classified correctly	% of values Predicted as "1"	% of Predicted "1" values that were correct	% of Predicted "-1" values that were correct	Best fitted threshold values
1	70.96%	20.02%	74.90%	69.98%	$a = 0.01$
2	77.88%	10.67%	81.08%	77.50%	$a = 0.05$ and $b = -0.125$
3	73.27%	31.18%	72.13%	73.79%	$a = 0.035$ and $b = -0.175$
4	77.35%	61.29%	77.42%	77.23%	$a = 0.05$ and $b = -0.05$ (not fitted—no option pricing data)
Naïve	53.05%	63.55%	51.13%	56.38%	N/A

Table 2—Performance of strategy-based classification methods

The arithmetic and geometric means for strategies (1)-(3), as well as the naïve strategy, are shown in table 3 (strategy 4 is not included as actual pricing data for options were not included in this data set—this was left for future research). From these data, it can be seen that strategies (3), (4), and the naïve strategy would provide the most opportunities to trade, even though (1) and (2) had higher average returns. Taking all of this data into account, the objective function determined that strategy (3) should produce the most profit among the stock trading strategies (recall that strategy 4 could not be evaluated for profitability since option pricing was not in the data set). In addition, (1)-(3) all outperformed the naïve strategy of simply investing in those stocks for which the actual return was higher than the consensus analyst return, which suggests that employing machine learning was useful. The fact that this naïve strategy performed so poorly as compared to the other methods suggests the difference between consensus analyst EPS estimates and actual EPS values may not have had as significant of an effect on the price shift following earnings releases as originally hypothesized. This evidence supports recent findings that managers purposefully distort the perceived success of their

companies in order to receive positive differences between actual and estimated EPS values, because this would cause EPS surprises to be less influential in the market [6].

Strategy	Arithmetic Average Return on Test Set	Geometric Average Return on Test Set
1	3.214%	3.038%
2	3.274%	3.179%
3	2.305%	2.018%
Naïve	0.358%	0.180%

Table 3—Average returns of strategy-based classification methods on the test set

Strategy-based classification was also used to improve the regression results obtained at the outset of the study (see methodology for a description of how this was done). The results of this regression for Random Forests are shown in table 4. Comparing these results to those obtained earlier, the LSE obtained by separating the data first via classification actually increased with this method, suggesting that the data separation may have caused the new regression method to perform even worse than the original ones. However, further analysis showed more promising results. The R^2 value of the actual $ReturnPost$ values regressed on the predicted $ReturnPost$ values was much higher in this method, and almost above 0.1. In addition, the slope and intercept of this regression line were much closer to 1 and 0, respectively, than any of the former methods employed. This suggests that while there was more total error in the prediction, separating the data using classification prior to performing regression allowed the methods to better capture the trends that existed in the data, even if only slightly. Figure B.2 in Appendix B contains a scatterplot of actual versus predicted, which verifies this analysis.

Best a	Best LSE on test set	R^2	Slope	Intercept
-0.05	10.0157	0.0934	0.777	-0.00145

Table 4— Results of Random Forest regression derived from strategy-based classification when predicting $ReturnPost$

V. Conclusions and Further Research

The overall objective of creating a potentially profitable machine-learning based trading strategy to capture information in earnings reports was accomplished throughout the course of this study. Although regression techniques proved to be unsuccessful in predicting the difference between open and close prices during the trading session following earnings releases, strategy-based classification was able to find some underlying patterns that generalized to out-of-sample test data. The success of these methods imply that very often a high push in momentum the day following earnings reports can be predicted before the trading session begins. However, these conclusions do come with some reservations. Although the prediction algorithms were trained on data that was completely separate from the test data, they were all drawn at random. Thus, some data points in the testing set may have temporally occurred before data points in the training set. It would be useful to see how training the algorithm on all data before a given time would perform when predicting later observations. It is expected that this type of study should not drastically change the results because the data used were not true time series data. However, to be safe, the strategies should be tested both analytically as well as in a paper trading account before being implemented. Also, it is not immediately clear how much transactions costs would decrease the mean returns of each strategy—another item to be investigated via paper trading.

Future research into this area may provide interesting new statistical insights. The most natural extensions of the work done in this study would involve using strategy-based classification to create analogous short-selling strategies, as well broadening the dataset beyond the Capital Goods sector. In addition, the prediction ability of the classifier designed to aid a straddle option strategy seemed to be very strong, so analysis into option prices on the day following earnings reports could potentially be very useful. The strategy-based classification

also provided a promising new outlook for how to perform regression on noisy data, which could be eventually refined and generalized to other data sets. Outside of regression, this study focused on how to successfully optimize threshold values for strategy-based classification rather than comparing the utility of different learning algorithms. Thus, a future project that extends the Random Forest techniques used here to other classifiers could provide more insight into the data. In light of the success of strategy-based classification, as well as the prospects for further research in this area, a bright future exists for the use of machine learning to analyze financial earnings.

Appendix A—Dataset Attributes

Prices in dollars and EPS values are quarterly.

Attribute ID	Attribute	Description	Possible Values
1	Ticker	Official Ticker for the observed stock	Text, ~5,000 analyzed
2	Sector	Stock sector of the observation	Text, 13 possible sectors
3	Industry	Stock industry of the observation	Text, over 200 possibilities
4	Month	Month of earnings release	1-12
5	Actual	Actual EPS for the observation quarter	Real values
6	MeanEstimate	Consensus mean EPS estimate for the observation quarter	Real values
7	BeforeMkt	TRUE if the earnings release occurred before the market opened, FALSE if after the market closed	Boolean TRUE or FALSE
8	ClosePrior	Closing price the trading session immediately before the earnings release	Positive real values
9	Close10	Mean closing price of the 10 days immediately before the earnings release	Positive real values
10	Close60	Mean closing price of the 60 days immediately before the earnings release	Positive real values
11	MeanIntra10	Arithmetic mean percent return if stock was purchased on open and sold on close for the 10 days immediately before the earnings release	Real values
12	MeanIntra60	Arithmetic mean percent return if stock was purchased on open and sold on close for the 60 days immediately before the earnings release	Real values
13	SDIntra10	Sample standard deviation of the percent return if stock was purchased on open and sold on close for the 10 days immediately before the earnings release	Positive real values
14	SDIntra60	Sample standard deviation of the percent return if stock was purchased on open and sold on close for the 60 days immediately before the earnings release	Positive real values

15	Vol10	Mean number of shares traded per day for the 10 days immediately before the earnings release	Positive real values
16	Vol60	Mean number of shares traded per day for the 60 days immediately before the earnings release	Positive real values
17	SDVol10	Standard deviation of the number of shares traded per day for the 10 days immediately before the earnings release	Positive real values
18	SDVol60	Standard deviation of the number of shares traded per day for the 60 days immediately before the earnings release	Positive real values
19	MarketCap	Market capitalization of the stock the session immediately before the earnings release	Positive real values
20	NormProfit	$Actual/ClosePrior$	Real values
21	NormSurprise	$(Actual - MeanEstimate)/ClosePrior$	Real values
22	ReturnPost	Realized return if stock was purchased on open and sold on close the day following the release	Real values
23	MinReturnPost	Realized return if stock was purchased on open and sold at the low value the day following the release	Non-positive real values
24	MaxReturnPost	Realized return if stock was purchased on open and sold at the high value the day following the release	Non-negative real values

Appendix B—Actual vs. Predicted Figures

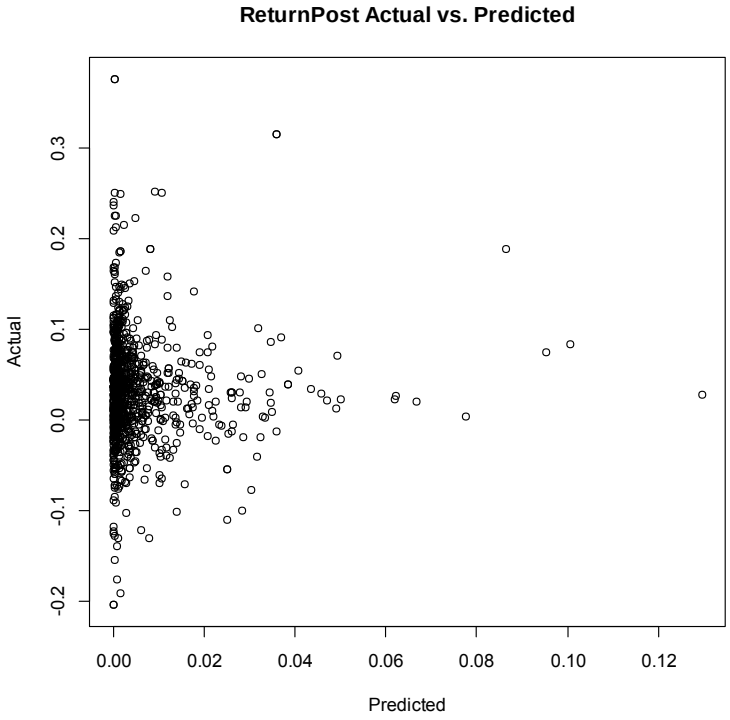

Figure B.1—Actual vs. predicted scatterplot for additive nonparametric regression.

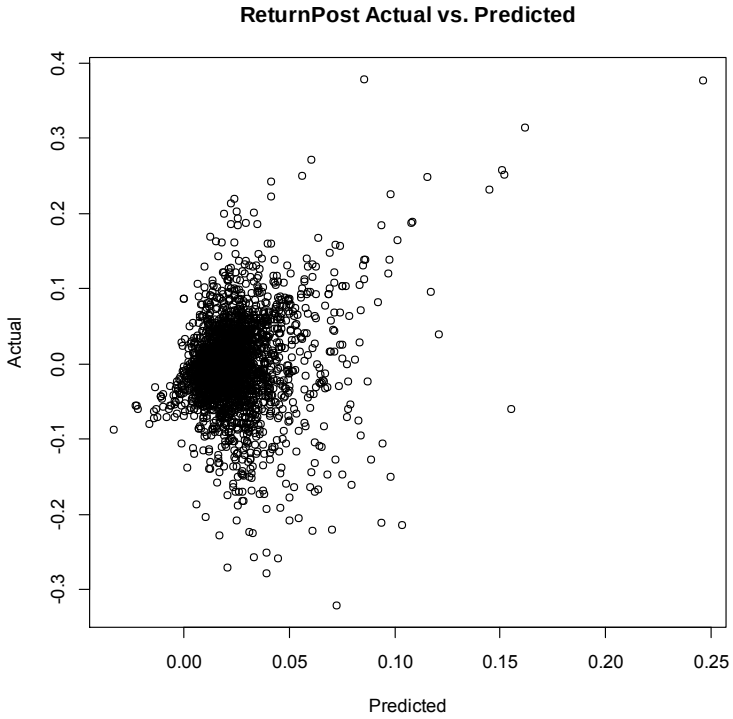

Figure B.2—Actual vs. predicted scatterplot for Random Forest regression derived from strategy-based classification.

References

[1] Breiman, Leo, et. al. "Breiman and Cutler's Random Forests for Classification and Regression." CRAN R Project. 15 February 2012. <http://cran.r-project.org/web/packages/randomForest/randomForest.pdf>.

[2] Breiman, Leo. "Random Forests." Machine Learning 45 (2001): 5-32.

[3] Fox, John. "Nonparametric Regression." CRAN R Project. January 2002. <http://cran.r-project.org/doc/contrib/Fox-Companion/appendix-nonparametric-regression.pdf>.

[4] Hecht-Nielsen, Robert. "Theory of the Backpropagation Neural Network." International Joint Conference on Neural Networks, 1989. vol. 1: 593-605.

[5] "Investopedia Dictionary." Investopedia.com. 13 November 2007. Available from <http://www.investopedia.com/dictionary/#axzz1v3szuOmA>.

[6] Matsumoto, Dawn A. "Management's Incentives to Avoid Negative Earnings Surprises." The Accounting Review 77.3 (2002): 483-514.

[7] Ripley, Brian. "Feed-Forward Neural Networks and Multinomial Log-Linear Models." CRAN R Project. 14 February 2012. < http://cran.r-project.org/web/packages/nnet/nnet.pdf>.

[8] Therneau, Terry M. and Beth Atkinson. "Recursive Partitioning." CRAN R Project. 4 March 2012. < http://cran.r-project.org/web/packages/rpart/rpart.pdf>.

[9] Wood, Simon. "Mixed GAM Computation Vehicle with GCV/AIC/REML Smoothness Estimation." CRAN R Project. 30 April 2012. < http://cran.r-project.org/web/packages/mgcv/mgcv.pdf>.

[10] "Wharton Research Data Services." Wharton—University of Pennsylvania. Available from <https://wrds-web.wharton.upenn.edu/wrds/>.

[11] Wu, Xindong and Vipin Kumar, eds. The Top Ten Algorithms in Data Mining. Boca Raton: Chapman and Hall/CRC Press. 2009.

Note: Wharton Research Data Services (WRDS) was used in preparing this paper. This service and the data available thereon constitute valuable intellectual property and trade secrets of WRDS and/or its third-party suppliers.

15.097 Prediction: Machine Learning and Statistics
Spring 2012