Project Report for CIS 419/519
Stock Recommendations using Machine Learning

Abstract

In this report, we present a simplified stock recommendation system. This system is based on patterns found from the price history of a list of equities. In developing the system, various machine learning techniques were tested to try to predict which stocks should be bought or sold to maximize the percent return going forward. We chose to use neural networks, support vector machines with a gaussian kernel, and naive bayes as our main machine learning techniques. We tried a few additional models as well, such as boosted decision trees, but not as extensively. We trained each model on past data through a backtesting module and then presented each model with new data in order to make predictions. In addition, we utilized our module to test several different combinations of features and compared the accuracy, precision, and recall of various models.

1. Introduction

Many financial institutions and investors are wary of trading in the stock market since the financial collapse in 2008. Almost all financial institutions have turned to machine learning or other algorithmic techniques in order to reduce the risk in trading. These techniques are used both in high-frequency trading and longer-term trades based on a buy and hold strategy. Our system is targeted as a longer-term trade recommendation system, avoiding the real-time constraints of high-frequency algorithms. We attempt to solve a small subset of this larger financial landscape, making good short-term predictions for buying or selling stocks that the average investor might be able to utilize.

The principle of our trading strategy is based on the idea that price movements in a stock form technical patterns that represent the sentiment of a particular equity over time. These patterns can help predict the future price movement of a stock, which if known, can improve the overall return for an investor. In order to simplify our recommendation model we have limited our selection to the stocks in the Dow 30 and only look at closing price for the stocks in our list as the basis of our predictions. Also, in order to avoid the complication of taxes and trading costs we decided to subtract a flat 0.5% from each trade made.

1.1. Model assumptions

In our implementation, we make several assumptions to simplify the constraints. We believe that while these assumptions necessarily make the algorithms we develop less applicable to the real world, they still remain close to from reality. We believe that some of our models could potentially be used by individuals in the real world at some rate of success, but at their own risk. We first assume that the trader has stock in whichever stocks our model was trained on. This allows us to to predict to buy, sell, or hold regardless of whether or not the individual actually owns any stock at that current point in time. Our model can be viewed as the global prediction of whether to buy, sell, or hold on a certain day and is not related or tied to a specific portfolio.

An additional assumption will be that the closing price reflects the sentiment of that day's price action enough to be considered the price for the entire day. This has the added benefit of avoiding intraday fluctuations in the stock market. We will also ignore all transaction fees and taxes associated with trading on the stock market and simply use a flat fee of 1.0%.

1.2. Definition

Our models were compared on several different characteristics, including many base machine learning metrics. In addition, we calculated the percent return that our model would have produced if it was followed every day of the test. This was calculated so that we could compare it to the actual return of the stock over the same period of time. That way, we could empirically see that our model was outperforming the average case. The formula for calculating the percent return is below:

$$PercentReturn = \frac{(V_{now} - V_{beginning})}{V_{beginning}}$$

University of Pennsylvania, CIS 419/519 Course Project.

2. Methodology

2.1. Source of data

The dataset used was downloaded from Wharton Research Data Services (WRDS). The stocks used are the stocks that comprise the Dow30. The Dow30 is INSERT TEXT HERE. For each stock, we pulled several features, including closing price, high, low, and volume. We pulled stock data from January 1, 1983 to December 31, 2013 to give us a window of 30 years. WRDS does not currently store the data we were utilizing for 2014.

The closing price is the INSERT TEXT HERE. The high and low are the INSERT TEXT HERE (which is ask and which is bid). The volume is the INSERT TEXT HERE.

The features we used could also be pulled in different time frames. For instance, we could pull yearly, monthly, or daily records for closing price. We could even pull hourly records on the high and low features. CAN VOLUME BE PULLED HOURLY? We pulled each of the different time frames in order to try and find the best time frame in addition to the best feature set.

2.2. Trading strategy

The strategy each of our models is designed for is determining exactly when it is time to buy and sell stocks as well as when to simply hold steady. Therefore, we have a choice of three classes to assign to each time frame in our model. Our model assigns its choice based on the threshold of the transaction fee of 1.0% we assigned above. If the model is expecting a change of more than 1.0% either way, the model will choose to buy or sell accordingly. Otherwise, it will predict to hold. This decision is because it does not make sense to make a trade when the percentage difference is less than the fee to make the transaction. We then compare the model's choice with the actual best decision, which is calculated with the actual percent change in the same manner, in order to determine if the model's choice was correct. All predictions, including hold, are counted as successful "trades" if they match the actual best decision. The number of successful trades divided by the number of possible successful trades is how we obtain our accuracy. Calculations for precision and recall are related to the number of successful trades as well.

2.3. Backtesting

We have written a backtesting module to look at different time frames and feature sets in our historical trades data. The backtesting module also allows us to choose where our time frame begins and ends, i.e. if we only wanted to use 2013 data for our training and testing. After researching how it is done in the real world, we have also implemented pattern recognition in our backtesting code to confirm that there are trends we could exploit.

After some runs with different inputs with the backtesting module, we began to deduce that the best time frame for training and testing was daily data, not monthly or yearly. The monthly or yearly data simply produced very bland patterns when compared to the patterns produced by daily data. BETTER EXPLANATION. We decided we would utilize daily data when training our models in order to achieve the best, and more importantly, most realistic data.

Section 3.5 shows some example patterns that our system found with the current pattern in cyan. The other lines are past patterns that our system found that are potential matches and the dots to the right represent outcomes for each past pattern. The rightmost dots are the actual outcome and average of the past patterns.

2.4. Training and Analysis Overview

We trained and analyzed the performance of each of the following models: naive bayes, SVM with gaussian kernel, and neural network. Each model was trained with the help of the backtesting module. MORE INFO HERE OR IS ABOVE ENOUGH? Each of the models were trained on the same stocks, time frame, and features.

We define a trade as a prediction. Every day a prediction must be made, whether to buy, sell, or hold the stock. Whether or not this prediction is the correct prediction is the heart of our model. We are attempting to predict the correct trade as often as possible based on previous data in the model.

Our models utilize the percent change of the various features they are given in order to determine whether to buy, sell, or hold. SHOULD WE SAY THE THRESHOLD AND STUFF, REALLY DUMBS IT DOWN. BOTH HERE AND IN TRADING STRATEGY. SHOULD SAY WE USE THE PERCENTAGE CHANGE TO DETERMINE WHAT THE BEST DECISION WAS. SHOW FIGURES HERE.

We used several metrics to evaluate our model, including accuracy, precision, and recall. In addition to those base metrics, we also calculated the percent return as defined above in order to better determine the financial success of our model. For example, our model could predict very well, but perhaps only gain a little on each correct prediction and lose a lot on each incorrect prediction. Therefore, percent return provides a way to verify that our correct predictions are more than just correct, they need to also be profitable. We can determine if our percent return is successful by comparing it with the stock's average actual percent return over the same period. If we outperform the actual average percent return, we consider our model to be successful.

For each model, our main focus was on model accuracy and maximizing the percent return. We wanted to produce a model that could not only beat the average return, but produce the best return possible given its training and input data.

FOCUSED ON MAXIMIZING PERCENT RETURN several features including the price and volume history of a list of equities could talk about data normalization

either here or before this we need to talk about how all models had the best success with a daily time frame and a single feature, but mention we tried combinations of them. Talk about how for the analysis in this paper, we used the last 100 days of 2013.

talk about how we found that all 3 classifiers did the best with just closing price - maybe because some features are not independent in finance, didn't help us to have additional features, only contributed to overfitting

talk about the most important models either here or above

talk about being trained on ony one stock at a time, model doesn't handle multiple stocks well

future work - try and predict the percent change so traders know if it's worth the risk as opposed to predicting what to do

is it worse than random guessing even though we have three options - maybe we could talk about how our predictions aren't statistically significant

3. Model Analysis

3.1. Naive Bayes

The first machine learning classifier we analyzed was naive bayes. Through research into past papers on the topic, it appeared many people had mixed results when trying to use naive bayes to predict. We did not expect naive bayes to perform well, but we did expect it to perform better than random guessing as it was still being trained on actual data. We tested multiple time frames with our naive bayes implementation, including yearly, monthly, and daily data. We also tested many different combinations of features in order to see if different feature sets changed how the model performed.

Unfortunately, we were never able to achieve statistically significant results with our naive bayes implementation. We averaged right around 51% accuracy and a 16% return for the model, which more or less equates to random guessing. The percent return also hovered right around the average return for the Dow30 over the same time frame, so we had not achieved a better than average return either.

NEED TO TALK ANYMORE ABOUT THIS

For comparative purposes to the other more successful models analyzed below, here is the naive bayes model's performance on each of the thirty stocks we analyzed.

NAIVE BAYES TABLE

ENDING?

3.2. SVM (gaussian)

The second machine learning classifier we analyzed was a support vector machine with a gaussian kernel. Research into using support vector machines in the past had given us insight into how successful these models could be. We expected the svm to perform very well on the data and to provide us with a better return on the Dow30 than the average. After tuning the parameters with a lengthy grid search, we found that our optimal svm had very respectable predictive power.

On average over the thirty stocks, our svm model predicted the correct trade with an accuracy of 67.97%. We also averaged a 34.93% return over the thirty stocks, meaning that we more than doubled the average percent return over the same time frame. It is important o note that the predictions and percent return are based on a simplified model, without factoring in transaction costs. However, these returns are still much higher than average, and could be useful for many traders on the stock market.

NEED TO TALK MORE ABOUT THIS? WHY WAS SVM SUCCESSFUL? WHY DID WE THINK THE SVM WOULD BE SUCCESSFUL?

The results for each of the Dow30 stocks are displayed below.

SVM TABLE

The svm accuracy does fluctuate quite a bit between some of the stocks, having higher predictive power for some (MRK, MCD) and less for others (TRV, V). Through some deeper digging, this appears to be the result of the svm performing better on more stable stocks (segements such as drugs, food, and retail) when compared to more volatile stocks (segements such as insurance and finance). This was an interesting corollary we saw to our main findings.

IS COROLLARY THE CORRECT WORD?

3.3. Neural network

The final machine learning classifier we analyzed was a neural network. There was some papers we found analyzing neural network performance, but we mainly decided to try it as many quantitative finance teams in industry utilize neural networks in order to make predictions on current market trends. Faced with this knowledge, we expected the neural netowrk to perform well and outperform the Dow30

average percent return.

TALK ABOUT BACKPROP THING WE FOUND AND RANDOM RULE MICHAEL WENT WITH

On average over the thirty stocks, our neural network performed the best of the three models we analyzed. Our neural network had an average trade prediction accuracy of 68.81% and an average 34.66% return. We more than doubled the average return of the Dow30 over the same time period. The results here are slightly more optimistic than they should be, as our model was trained and predicted on a reduced model with a fixed transaction cost. However, the model still performs very well and would definitely prove very helpful for traders looking for an edge.

WE SHOULD SAY SVM DID BETTER BECAUSE IT HAD A SLIGHTY HIGHER PERCENT RETURN, BUT THE DIFFERENCE BETWEEN THE TWO WAS NOT THAT IMPORTANT OR SIGNIFICANT

NEED TO TALK MORE ABOUT THIS? WHY WAS NN SUCCESSFUL? WHY DID WE THINK THE NN WOULD BE SUCCESSFUL?

NN 100 epochs, ADDITIONAL DON'T HELP TOO MUCH, MUST ALREADY BE CONVERGING

The results for the neural network on all thirty stocks are printed below.

NN TABLE

MAYBE COMBINE OBSERVATIONS INTO OWN SUBSECTION The neural network prediction accuracy fluctuates quite a bit on for differing stocks. The model, like the svm, had higher predictive accuracy for more stable stocks and performed worse (sometimes even worse than random guessing) for more volatile stocks.

3.4. Overall comparison

We had a large amount of success with two of the three models we trained, which is exactly what we expected to see. We did not predict that our successful models would perform as well as they did. Both the svm and neural network had a high percent return and accuracy. The svm performed slightly better in terms of percent return, while the neural network had a slightly higher accuracy. Both outperformed the Dow30 average percent return, meaning we can empirically state that our implementations are successful and would be useful for traders looking for an edge.

MAYBE PUT OBSERVATIONS FROM INDIVIDUAL SECTIONS HERE.

For a quick summary, here is the average performance of each of the three models over the thirty stocks.

SUMMARY TABLE

3.5. Machine learning framework

At this point, we've also set up another framework that allows us to easily evaluate different classifiers, different amounts of data, and different features easily. This will help us decide which classifiers, features, and data to use to train each respective machine learning model for best results.

We've also reviewed numerous published papers to understand what existing work has been done, both successful and unsuccessful, which will ultimately aid in narrowing the scope on our final evaluation and help us determine best train and test practices.

3.6. Classifier Evaluation

So far we've evaluated a boosted decision tree and SVMs with polynomial, gaussian, and cosine similarity kernels using our framework. We've evaluated small subsets of data and different combinations of features for our preliminary trials in order to get some idea of performance without investing too much time. So far the gaussian kernel is performing the best of our set of tested classifiers, but requires additional tuning to be a good predictor in general. The boosted decision tree and other SVM kernels performed about the same as random guessing.

3.7. Future Work

Next, we plan to evaluate additional classifiers, including a neural network and naive bayes, with similar baseline variables for data and features.

As a stretch goal, we are also still trying to find additional sources of data to incorporate additional economic features into our dataset.

3.8. Figures

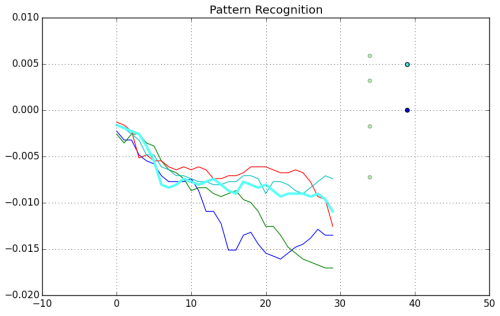

Figure 1. Examples of downward trend reversals

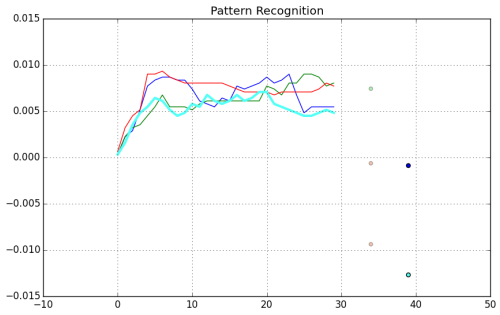

Figure 2. Examples of upward trend reversal

Acknowledgments

We would like to thank Eric Eaton and the TA's for their help with gathering data and getting started.

References

Chase Lochmiller, Yuan Chen. Predicting short term stock returns. Technical report, Stanford University, 2013.

Chenxu Shao, Zheming Zheng. Algorithmic trading using machine learning techniques. Technical report, Department of Management Science and Engineering, 2013.

Lin, Hui. Feature investigation for stock market prediction. Technical report, Stanford University, 2013.

Shunrong Shen, Haomiao Jiang, Tongda Zhang. Stock market forecasting using machine learning algorithms. Technical report, Department of Electrical Engineering, Stanford University, 2012.

Tianxin Dai, Arpan Shah, Hongxia Zhong. Automated stock trading using machine learning algorithms. Technical report, Department of Management Science and Engineering, 2012.