Documentation for my Entry to The Winton Stock Market Challenge

Michael Furchtgott

Seattle, WA

[furchtgott@gmail.com](mailto:furchtgott@gmail.com)

January 31, 2016

First of all, I would like to say thank you to the folks at Winton for sponsoring this interesting and challenging data science contest! I would be thrilled to speak with the team at Winton to learn about how they are using data science and machine learning in the finance industry. The following is a brief summary of how I attempted to predict the stock movements.

1. Overview of my approach  
     
   My approach was not very complicated. In fact, the entry that got me to fourth place in the final standings did not use any ML or forecasting methods more sophisticated than least-squares regression with some crude weights. My script was a few dozen lines of R code with no add-on packages required.   
     
   The most important insight in my approach was an intuition about what is realistic, and what isn’t, in terms of predicting stock price movements. I drew some of this intuition from portfolio theory, which I’ll discuss below. I believe that my score also benefitted from some predictions about the test-set stocks’ relative weights, which were absent from the test data provided to the competitors.
2. Feature selection and extraction

The most important features in my model were Ret\_Minus1, Ret\_Minus2, and the stock-specific mean and variance for the first 119 minute-by-minute returns. This approach draws on a classic concept in finance theory: some stocks move with the overall market, and some stocks move against it, to varying degrees. When I regressed Ret\_Plus1 against these features in the training data, I produced a simple estimate for how stocks that went up X% yesterday should have gone up today. There wasn’t much historical data to work with, but it was still much better than nothing, and the short window of historical data was part of the challenge of this competition.  
  
I also tried explaining some of the additional cross-sectional variation in Ret\_Plus1 and Ret\_Plus2 by using the “feature” variables (Feature\_1 through Feature\_25). The inclusion of these additional features did not help my score on the leaderboard—perhaps due to overfitting—and I quickly abandoned that approach. However, Feature\_13 was useful because it was correlated (>0.4) with the weights in the training data.

1. Training methods

I ended up with a very simple script at the end but I tried more elaborate approaches along the way. These approaches included:

* 1. For each stock in the test set, constructing a portfolio of training-set stocks that tended to move with that test-set stock. I would then predict the return of the test-set stock using that smaller training-set portfolio, rather than based on the entire training set. I was surprised when this approach did not appear to improve my score.
  2. Estimating a beta for each stock in the test data, using the minute-by-minute returns. [A beta coefficient captures the extent to which a stock moves with or against the overall market.] This approach didn’t help my score either. I suspect this is due to the different time slices of the data (unbeknownst to me at the time), as well as to the high-frequency beta estimates being far noisier than beta estimates using daily data.

I evaluated my models using a simple approach with the training set. I randomly partitioned the training set into a sub-training set (80%) and a sub-test set (20%). I then used this ad hoc cross-validation approach to test out new models before fitting them to the real test data and submitting them to see my new score on the leaderboard.

When my model generated outliers among the predicted values, I mitigated the damage a few ways. One such way was to Winsorize the predicted values for the daily returns—that is, I truncated the distribution in order to bring the extreme predictions closer to the rest of the distribution. Another way was to take a weighted average of my regression-based stock-specific prediction, and the median return from the training data. This was the closest I came to estimate ensembles, essentially an ensemble of my own estimate and a trivial estimate.

1. Simple features and methods  
     
   I am relatively new to Kaggle competitions. I dabbled in the Springleaf Marketing Response competition last summer. I found out later that some of the top finishers had used models similar to mine, but they had been more computationally aggressive and had done more fine-tuning of their parameters. I drew the lesson that excellent execution on simple methods could get me most of the way toward the goal and would usually outperform complicated and unwieldy methods.

One thing I’ve learned as a data scientist is to think carefully about what it is that we’re trying to optimize. I paid close attention to the evaluation metric—in this case, the weighted sum of absolute deviations from the actual return. But the weights were absent from the test data that was presented. I estimated relative weights by using features from the training data. I also used ground-truth weights in fitting my linear model for daily return prediction, which likely helped my score.   
  
Overall, I used only a subset of features from the training data to fit my models: Ret\_Minus1, Ret\_Minus2, sum of minute returns, variance of minute returns, and Feature\_13. These features plus a few transformations got me almost all of the way there.

1. Interesting findings  
     
   I am surprised by how well my approaches worked even though there were major findings that I missed out on. In particular, after the contest ended, I read a contest administrator’s post on the forum explaining that the data were taken from different days. That might explain why my beta models performed so poorly. Yet my daily forecast estimates did reasonably well, considering they were built on similar logic as a CAPM beta model.  
     
   Another interesting finding is that for my training models, least-squares optimization methods outperformed WLAD methods. I was surprised by this, since WLAD was the evaluation metric on the test data.
2. Background

I work as a Data Scientist for Microsoft, in a group that has an emphasis on A/B testing. I’m an economist by background; I completed my PhD in Economics, with a focus on corporate finance, at UC-San Diego in 2013. I come from a stats background that emphasizes hypothesis testing, inference properties of estimators, and causal identification more than pattern recognition, algorithmic feature engineering, and prediction. I’m working on complementing that skill set with industry experience, coding skills, understanding of databases, and knowledge of ML methods.   
  
I had two advantages in this particular Kaggle challenge. First, I was able to draw on a background in financial modeling and portfolio theory, as well as on my experience working with messy financial data. While I did not end up using any advanced financial models in the end, I was able to avoid getting bogged down with approaches that were destined not to work. Second, coming from an economics and finance background, I’m trained to use approaches driven by theories or hypotheses. These approaches sacrifice predictive power in some settings but are less likely to result in overfitting in settings like this one.

In retrospect, I wish I had invested more time in the final weeks of the competition. I was very busy after the holidays with work, travel, and other commitments, and it looks like I underestimated my own chances of finishing as a top contestant. But I was surprised and thrilled to learn that I had finished very close to the top!