Machine Learning in Statistical Arbitrage

Xing Fu, Avinash Patra

December 11, 2009

Abstract

We apply machine learning methods to obtain an index arbitrage strategy. In particular, we employ linear regression and support vector regression (SVR) onto the prices of an exchange-traded fund and a stream of stocks. By using principal component analysis (PCA) in reducing the dimension of feature space, we observe the benefit and note the issues in application of SVR. To generate trading signals, we model the residuals from the previous regression as a mean reverting process. At the end, we show how our trading strategies beat the market.

1 Introduction

In the field of investment, statistical arbitrage refers to attempting to profit from pricing inefficiencies identified through mathematical models. The basic assumption is that prices will move towards a historical average. The most commonly used and simplest case of statistical arbitrage is *pairs trading*. If stocks P and Q are in the same industry or have similar characteristics, we expect the returns of the two stocks to track each other. Accordingly, if P_t and Q_t denote the corresponding price time series, then we can model the system as

$$\frac{dP_t}{P_t} = \alpha dt + \beta \frac{dQ_t}{Q_t} + dX_t,$$

where X_t is a *mean reverting Ornstein-Uhlenbeck stochastic process*.

In many cases of interest, the drift α is small compared to the fluctuations of X_t and can therefore be neglected. The model suggests a contrarian investment strategy in which we go long one dollar of stock P and short beta dollars of stock Q if X_t is small and, conversely, go short P and long Q if X_t is large.

Opportunities for this kind of pairs trading depend upon the existence of similar pairs of assets, and thus are naturally limited. Here, we use a natural extension of pairs trading called *index arbitrage*, as described in [1]. The trading system we develop tries to exploit discrepancies between a target asset, the iShares FTSE/MACQ traded in the London Stock Exchange, and a synthetic artificial asset represented by a data stream obtained as a linear combination of a possibly large set of explanatory streams assumed to be correlated with the target stream. In our case, we use the stock prices of the 100 constituents of FTSE 100 Index to reproduce the target asset.

We start with linear regression on the 100 constituents and take the window as 101 days from April to September 2009. After extracting significant factors by Principal Component Analysis, we calibrate the model cutting the window size down to 60 days. Closed prices of each asset are used. The emphasis here is to decompose the stock data into systematic and idiosyncratic components and statistically model the idiosyncratic part. We apply both Linear Regression and Supported Vector Regression, and collect the residual that remains after the decomposition is done. Finally, we study the mean-reversion using auto-regression model in the residuals to generate trading signals for our target asset.

2 Linear Regression

We consider the data of 101 days from Apr.28 to Sep.18, 2009. The reason we choose this time period is that given 100 feature variables, we need at least 101 observations to train the 101 parameters in the linear regression model. To avoid overfitting parameters, we only use 101 training examples.

Denote P_t to be target asset and Q_{it} to be the ith stock constituent at time t. The linear model can be written as

$$P_t = \theta_0 + \sum_{i=1}^{100} \theta_i Q_{it}.$$

Implementing the ordinary least squares algorithm in MATLAB, we get parameters θ and training errors, i.e. residuals.

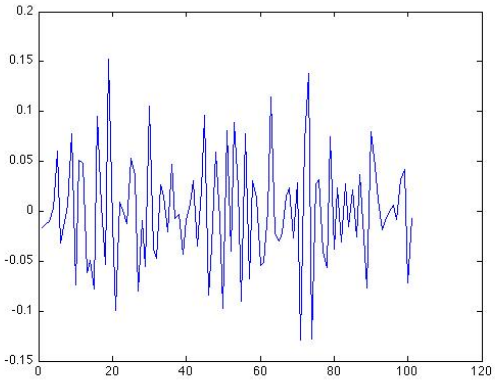

Figure 1: Residuals of linear regression over 100 constituents

From Figure 1, we see that the empirical error is acceptable. However, when we test this linear model using 30 examples not in the training set, i.e. prices of each asset from Sep. 21 to Oct. 30, 2009, the generalization error is far from satisfying, as shown in Figure 2. This implies that we are facing an overfitting problem, even though we've used the smallest possible training set. There are so many parameters in the linear model, which gets us into this problem. Hence, we apply Principal Component Analysis to reduce the dimension of the model.

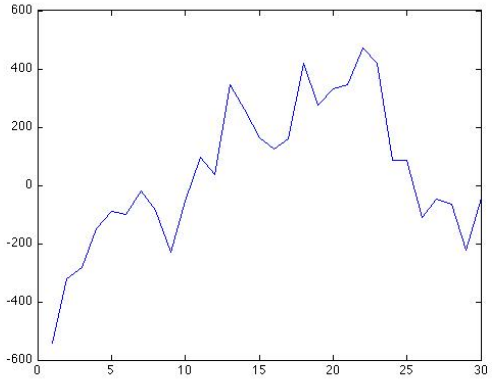

Figure 2: Generalization error on 30 testing days

3 Principal Component Analysis (PCA)

Now we apply PCA to analyze the 100 stocks. The estimation window for the correlation matrix is 101 days. The eigenvalues at the top of the spectrum which are isolated from the bulk spectrum are obviously significant. The problem becomes evident by looking at eigenvalues of the correlation matrix in Figures 3. Apparently, the

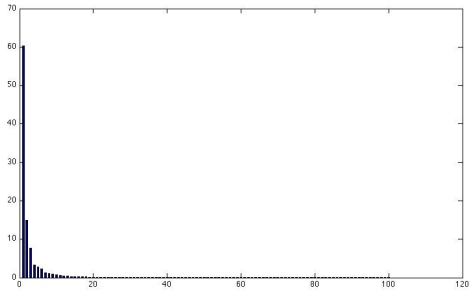

Figure 3: eigenvalue of the correlation matrix

eigenvalues within the first twenty reveal almost

all information of the matrix. Now we apply validation rules to find how many principal components to use that can give us the least generalization error. Given that the dimension of the model is reduced, we reset our window size to 60 days to avoid overfitting problem. After running multivariate linear regression within the first 20 components using 60 days training set, we find that the first 12 components give the smallest generalization error on the 30 testing days. The out-of-sample residual is shown in Figure 4.

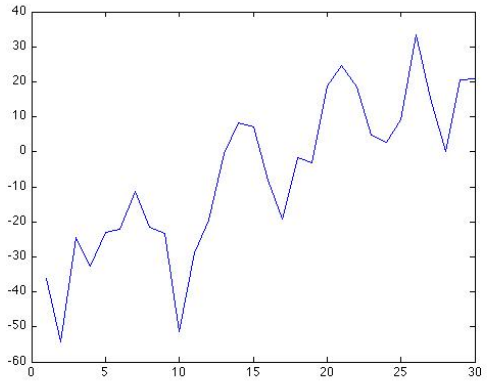

Figure 4: Generalization error on the first 12 components over 30 out-of-sample days

We take a quick look back at the empirical error of these 12 components over 60 training days. From Figure 5, we see that the residuals are not as satisfying as in Figure 1, in terms of magnitude, but residuals here successfully reveal the trend of residuals using 100 constituents. Thus, by applying PCA to reduce the dimension of the model, we avoid overfitting parameters. We will use the in-sample residuals from regression on principal components to generate trading signals in a following section.

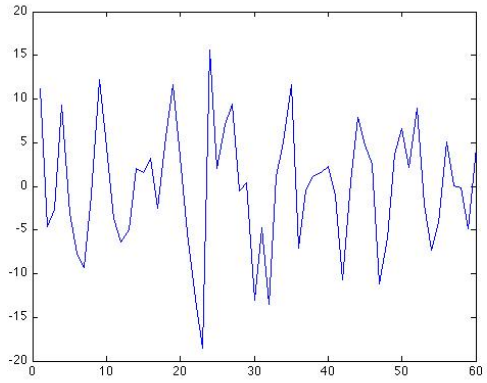

Figure 5: Empirical error on the first 12 components over 60 training days

4 Support Vector Regression (SVR)

We apply SVR on the 12 feature attributes obtained through PCA. We use a Gaussian kernel and empirically decide on the kernel bandwidth, cost and epsilon (slack variable) parameters. (We used SVM and Kernel Methods MATLAB toolbox, available online for implementation). We do not notice any improvement in this approach,as seen in the plots of training error and test error, Figure 6 and Figure 7. A main issue here is to be able to determine appropriate SVR parameters.

5 Mean Reverting Process

We want to model the price of the target asset such that it accounts for a drift which measures systematic deviations from the sector and a price fluctuation that is mean-reverting to the overall industry level. We construct a trading strategy when significant fluctuations from equilibrium are observed.

We introduce a parametric mean reverting model for the asset, the Ornstein-Uhlembeck

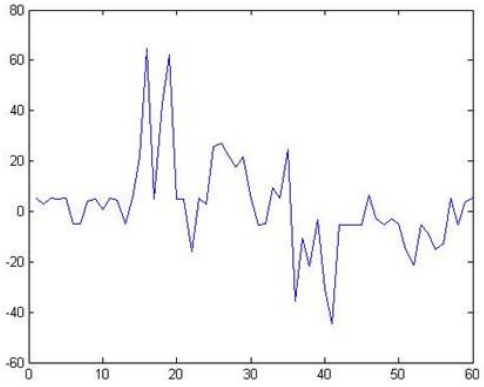

Figure 6: Empirical error on the first 12 components over 60 training days using SVR

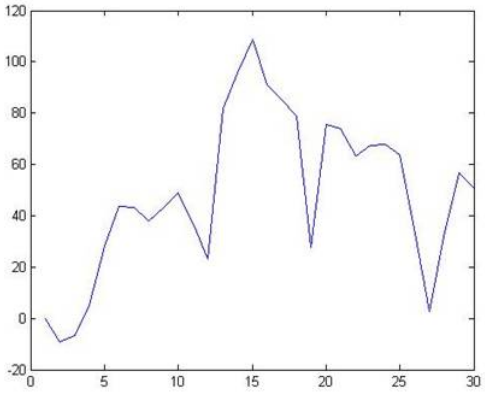

Figure 7: Generalization error on the first 12 components over 30 testing days using SVR

process,

$$dX(t) = \kappa(m - X(t))dt + \sigma dW(t), \quad \kappa > 0.$$

The term $dX(t)$ is assumed to be the increment of a stationary stochastic process which models price fluctuations corresponding to idiosyncratic fluctuations in the prices which are not reflected in the industry sector, i.e. the residuals from linear regression on principal components in the previous section. Note that the increment $dX(t)$ has unconditional mean zero and conditional mean equal to

$$E\{dX(t)|X(s), s \leq t\} = \kappa(m - X(t))dt.$$

The conditional mean, i.e. the forecast of expected daily returns, is positive or negative according to the sign of $m - X(t)$.

This process is stationary and can be estimated by an auto-regression with lag 1. We use the residuals on a window of length 60 days, assuming the parameters are constant over the window. In fact, the model is shown as

$$X_{t+1} = a + bX_t + \epsilon_{t+1}, \quad t = 1, 2, \cdots, 59,$$

where we have

$$\begin{aligned} a &= m(1 - e^{-\kappa\Delta t}) \\ b &= e^{-\kappa\Delta t} \\ Var(\epsilon) &= \sigma^2 \frac{1 - e^{-\kappa\Delta t}}{2\kappa}. \end{aligned}$$

Hence, we get

$$\begin{aligned} \kappa &= -\log(b) \times 101 = 178.2568 \\ m &= \frac{a}{1-b} = -0.0609 \\ \sigma &= \sqrt{\frac{Var(\epsilon) \cdot 2\kappa}{1 - b^2}} = 135.9869 \\ \sigma_{eq} &= \sqrt{\frac{Var(\epsilon)}{1 - b^2}} = 7.2021. \end{aligned}$$

6 Signal Generation

We define a scalar varaible called the s-score

$$s = \frac{X(t) - m}{\sigma_{eq}}.$$

The s-score measures the distance of the cointegrated residual from equilibrium per unit standard deviation, i.e. how far away a given stock is from the theoretical equilibrium value associated with our model. According to empirical studies in this field, our basic trading signal based on mean-reversion is

1. Buy iShare FTSE if $s < -1.25$
2. Sell iShare FTSE if $s > 1.25$
3. Close short position in iShare FTSE if $s < 0.75$
4. Close long position in iShare FTSE if $s > -0.5$

The rationale for opening trades only when the s-score s is far from equilibrium is to trade only when we think that we detected an anomalous escursion of the co-integration residual. We then need to consider when we close trades. Closing trades when the s-score is near zero makes sense, since we expect most stocks to be near equilibrium most of the time. Thus, our trading rule detects stocks with large excursions and trades assuming these excursions will revert to the mean.

Figure 8 shows the signals we generate over 60 days.

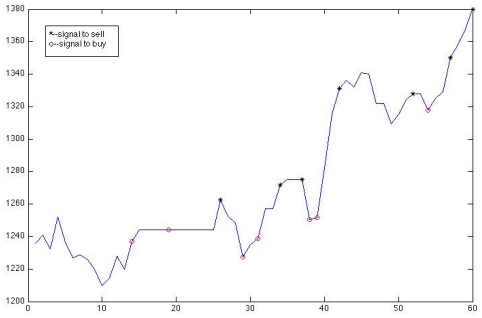

Figure 8: Trading signals over 60 days

7 Conclusion

We note that PCA helped in getting rid of overfitting problem with 100 feature attributes, while conducting linear regression. However, we see that to effectively apply Support Vector Regression, a technique to learn SVR-parameters might have to be developed. We see on testing (over 60 days data) that using our strategy based on the trading signals clearly makes a trading profit as on average we purchase iShare FTSE when it is "low" and sell it when it is "high". On the other hand, in the future, idiosyncratic factors behaving erratically (such as occurrence of some specific unforeseen event) might lead to the index systematically under-performing or doing much better than the "significant factors" found from PCA and this could seriously undermine the effectiveness of our approach. To achieve a systematic method, online learning would be a good way to try out, in order to update our feature set with the latest information.

References

[1] G. Montana, K. Triantafyllopoulos and T. Tsagaris, *Flexible least squares for temporal data mining and statistical arbitrage,* Expert Systems with Applications, Vol. 36, Issue 2, Part 2, pp. 2819-2830, 2009.

[2] A. Marco and Jeong-Hyun Lee , *Statistical Arbitrage in the U.S. Equities Market*, Social Science Research Network, 2008.

[3] T. Fletcher, *Support Vector Macines Explained*, 2009