DSO 530

PROJECT REPORT

ALGORITHMIC TRADING

PREPARED BY	EMAIL
DIAN (DONNY) YANG	dianyang@usc.edu
IMANI MUFTI	imufti@usc.edu
ROMAN SIELEWICZ	sielewic@usc.edu
MING-HSIEN SANDRA CHEN	minghsic@usc.edu
VISHAL CHOLLERA	vcholler@usc.edu

TABLE OF CONTENTS

▪ UNDERSTANDING OF THE PROBLEM

As one of the fundamental factors behind the emergence of FinTech, algorithmic trading has transformed the legacy practices of Wealth Management. In 2019 alone, the volume of trades executed algorithmically amounted to $11 billion, and this industry is forecasted to continue growing 11% annually (2019-2024). However, the methods to accurately assess risk, identify high-value opportunities and optimize timing are highly complex and create great challenges for professionals across industry and academia.

The focus is to develop a robust model that attempts to accurately forecast returns for a sample of 50 securities comprising the SSE 50 index. The problem assumes that we have $1,000,000 which we can invest in varying amounts in any stock in our sample. Each buy/sell transaction incurs a transaction cost of 0.065% times the share price at time-period *t*. The objective is to maximize profits through simulating trades for securities from day 505 to day 756. Throughout our process, we engineered features, applied a range of statistical learning approaches, and tuned a final model, which we used as the basis of decision-making for our trading algorithm.

▪ DATA DESCRIPTION

The data provided consists of daily stock data of 50 companies which are constituents of the SSE 50 index. The period of the data is from Day 1 to Day 756 (in the years 2017-2019). There are two types of data provided for each stock:

1. **Raw Data** (from Day 1 – Day 756)
   - ***Open*****:** Opening price of the stock that on day *n*.
   - ***High*****:** Highest price of the stock on day *n*.
   - ***Low*****:** Lowest price of the stock on day *n*.
   - ***Volume*****:** Volume of shares traded on day *n*.
   - ***Adjusted Close*****:** Closing price of the stock on day *n* after considering dividends and stock splits.

2. **Engineered Features** (from Day 17 – Day 756)
   - ***RSI lag*****:** Momentum indicator between 1 to 100 which measures the magnitude of recent price changes to evaluate overbought or oversold conditions in stock price.
   - ***fastK lag*****:** Momentum indicator between 1 to 100 which compares the closing price of a stock to a range of its prices over a certain period of time.
   - ***AD lag*****:** Determines the flow of money into or out of a stock. It ranges between -1 to 1 where negative indicates outflow and positive indicates inflow.
   - ***OBV lag*****:** Trading momentum indicator that uses volume flow to predict changes in stock price.
   - ***PROC lag*****:** 1-day lag of the rate of change in stock price.
   - ***MA 5 lag*****:** 5-day moving average
   - ***MA 15 lag*****:** 15-day moving average
   - ***Volume lag*****:** 1-day lag of the volume of shares traded
   - ***day 5 return lag*****:** 1-day lag of the 5-day return
   - ***day 15 return lag*****:** 1-day lag of the 15-day return

In addition, we engineered additional features to enhance our final model's performance.

3. **Additional Engineered Features**

- ***MA 5 return*****:** Moving average of the 5-day return %
- ***MA 15 return*****:** Moving average of the 15-day return %
- ***MSTD 5 return*****:** Moving standard deviation of the 5-day return %
- ***MSTD 15 return*****:** Moving standard deviation of the 15-day return %
- ***MA dir flag*****:** Directional flag indicating whether the 5-day and 15-day moving averages are moving in the same direction. 1 if they're moving in the same direction, 0 otherwise. (Binary)

▪ ATTEMPTED MODELING APPROACHES

Moving Simple Linear Regression of Adjusted Closing price

In this modeling approach, we used the closing prices of the previous N days to predict the closing price 2 days from the present. Essentially, this modeling approach employed a moving training window of N days, where N is selected by choosing the number of training days that minimized mean actual percent error in the validation set (day 357 to day 504). An example of inputs to this model would be the number of days into the training period (e.g. [0, 1, 2, 3, 4,] when N = 5) and the corresponding adjusted closing prices on those days of the training window. We then made a prediction for time N+1, based on the linear trend of the closing prices in the previous days. This model had two major weaknesses. The first is that it is slow to respond to changes in trends, as each new point only shifts the slope of the trend slightly. The second is that although it generated a high R-squared and low prediction error, it predicts closing price, not return, which is ultimately the variable of interest. Due to this issue, the predictions for expected return were not accurate enough to generate a profit during the trading period.

Multiple Linear Regression using only provided variables to predict 5-day return

In this modeling approach, we utilized only the regressors provided in the datasets to predict 5 day return. This model was unsuccessful largely due to the forecast period being too long, and the 5 day percent return varying too wildly to be accurately predicted by our model. In addition, the lack of moving average and moving standard deviation variables limited the prediction accuracy of this model. Ultimately, this model spurred us to add more engineered variables and use a shorter forecast horizon for predicted return.

K-Nearest Neighbors Regression

In the K-nearest neighbors regression approach, we treat the Adjusted Closing Price of each stock as the response variable while using the rest of the variables as predictors along with the calculated percent change of High, Low, Open, and Close from the previous day. The models for each of the stocks were trained using 70% of the whole dataset for a specific stock and validated on the rest of the 30%. The parameter K, number of nearest observations that are in the past day, for each of the stock models was chosen using cross validation with cv of 5. In our first attempt using K-nearest neighbors regression, the data was not scaled or standardized, causing the predictions to look very much like random predictions. Scaling the data greatly improved the performance of the majority of the models. While the K-nearest neighbors regression worked well for some of the stocks (about one third of the 50 stocks) in predicting the next day's Adjusted Closing Price, it performed poorly

for the rest — the minimum RMSE was approximately 0.323 while the maximum RMSE was approximately 2888.582.

ARIMA model

Time series forecasting is commonly modeled using Auto Regressive Integrated Moving Average (ARIMA) as it usually captures the different temporal structures in the time series data well. Therefore, our team has decided to explore this option in predicting the Adjusted Closing Price of each stock. In the ARIMA models, we used Adjusted Closing Price to predict future Adjusted Closing price. We looked at five stocks from the dataset and followed a tutorial to plot the ACF and PACF graphs to decide on the parameters that ARIMA takes in. Since none of our team members has experience developing ARIMA models in Python, our team has decided to move away from this approach to focus on developing our final predictive model for each stock using alternative machine learning techniques.

Lasso Regression Model

In this modeling approach, we used a rolling-window Lasso regression model to predict future 1-day return % using a long training window, the eleven given engineered features, and five additional engineered features (also used in our final model, see below for details). However, the Lasso regression models resulted in predicted 1-day return values near zero for the most test days, rendering this modeling technique unusable. We believe that the sparse models encouraged by the Lasso technique removed important regressors from the models and led to the models having very poor predictive power, with low variance but high bias.

▪ FINAL PREDICTIVE MODEL

After exploring and evaluating the various predictive models described in the prior section, our team ultimately deployed a rolling-window multivariate linear regression model to predict future 1-day return % as the response. The regressors of the model are as follows:

```
regressors = ['volumelag', 'rsilag', 'fastKlag', 'fastDlag', 'ADlag',
'OBVlag', 'MA5lag', 'MA15lag', 'day5Returnlag', 'day15Returnlag', 'PROClag',
'ma5_return', 'ma15_return', 'mstd5_return', 'mstd15_return', 'ma_dir_flag']
```

In addition to the eleven provided feature engineered variables, we added five additional engineered variables to our model: Moving Average (5 Day) of 1-Day Return %, Moving Average (15 Day) of 1-Day Return %, Moving Standard Deviation (5 Day) of 1-Day Return %, Moving Standard Deviation (15 Day) of 1-Day Return %, and Moving Average Direction Flag (i.e. if MA5Day and MA15Day are moving in the same direction then 1 else 0). Since these additional input variables are aggregations of recent historical 1-day returns, they provide valuable explanatory power when predicting our response, future 1-day return %.

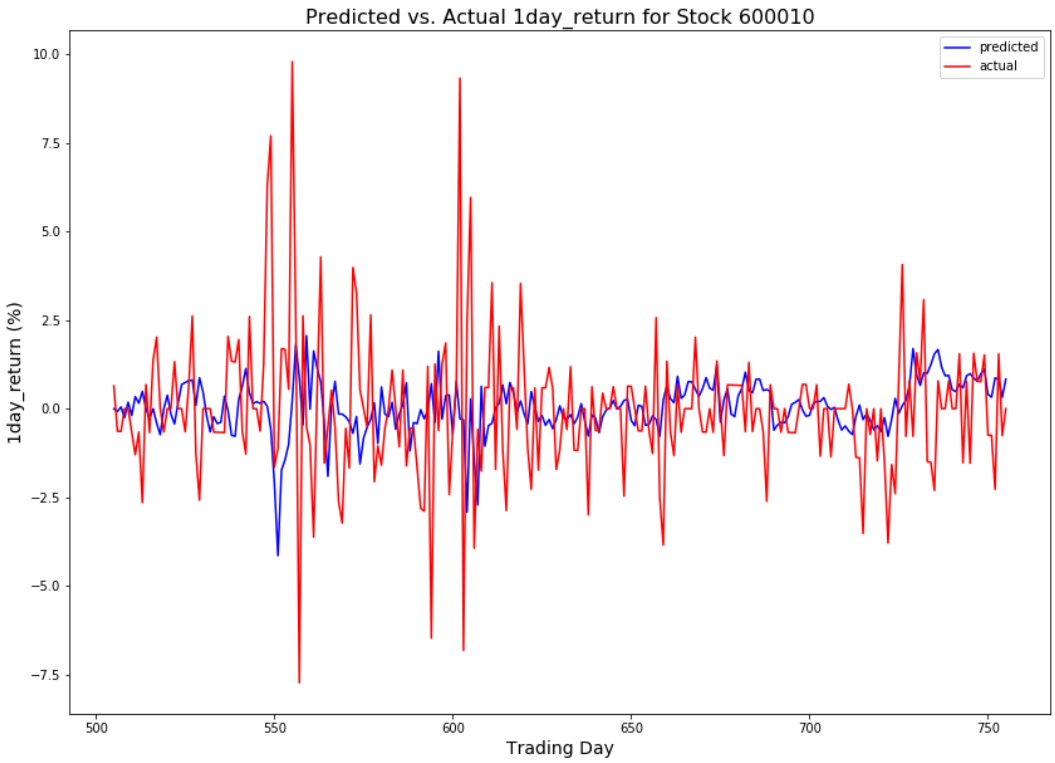
In order to make predictions for our stock portfolio during our trading window, we dynamically generated a separate regression model for each stock on each trading day. In other words, our algorithm produced fifty separate linear regression models for each trading day. The training window immediately preceded the day we make predictions for and would slide forward through time to

make predictions along the trading window. We set our training window length to be fifteen observations (in this case, *days*) for each regressor in our model. This results in:

*Training Window Length = 15 days per regressor * 16 regressors in model = 240 days*

We believe 240 days training window length is adequate to make predictions for the future return without overfitting our models. This overall methodology ensured that regression coefficients in each model are optimized for every individual stock, which is important because different stocks may have very different characteristics and levels of variation. In this way, we were more confident in the predictions we make for our overall portfolio.

To visualize an example output, the below chart plots predicted 1-day return in blue and actual 1-day return in red for Stock 600010 during the trading window.



Although the predicted values do not capture the extreme highs and lows of the actual return trend, which can be quite volatile, the shape of the prediction line can still be seen to follow the general shape of the actual return line. This validates that our methodology is performing the way we want it to.

▪ TRADING STRATEGY

As our regression models predicted future 1-day return % for each stock on each trading day, we employed a simple daily trading strategy to purchase and sell stocks in order to generate a strong, positive return on our investment. Our strategy was as follows:

i. Began with $1,000,000 of capital to invest.
ii. Predictive models generate predicted future 1-day return % for each stock on trading day 505.
iii. Rank-order stocks based on predicted future 1-day return %, highest to lowest.
iv. For stocks predicted to have a future 1-day return % greater than a buy threshold of 1.00% (also factoring in additional transaction costs to account for buy and sell costs), invest $100,000 into each of the highest-ranked stocks until cash is depleted.
v. Time moves forward one day. Predictive models generate predicted future 1-day return % for each stock on trading day 506.
vi. Rank-order stocks based on predicted future 1-day return%, highest to lowest.
vii. Sell the entire position on any stock that has a predicted 1-day return % of less than a sell threshold of -0.50%. Recoup cash.
viii. Invest $100,000 into each of the highest-ranked stocks with predicted 1-day return % greater than the buy threshold of 1.00% until cash is depleted.
ix. Repeat steps 5-8 for every trading day through day 755.
x. On day 756, sell entire positions on all stocks for cash.

We chose this trading strategy because we believed a simple trading algorithm based on the strength of our predictive models would give us a strong return. Daily trading was employed so that we would not miss out on the most up-to-date opportunities and risks based on the newest signals and information in the data. We chose the individual stock investment amount of $100,000 because it would typically allow us to invest in a maximum of ten stocks at a time. Spreading money across ten stocks ensured we would concentrate our capital on the most promising stocks while still achieving a level of risk diversification and reducing transaction costs to a degree. We decided to set our sell threshold (-0.50%) below zero but at a lower absolute level than our buy threshold (1.00%) in order to introduce a degree of short-term risk tolerance while favoring long-term profitability. By giving the algorithm a lower limit for maintaining our position, we created a bias for holding stocks, which turns out to be advantageous in this set of mostly upward trending assets.

▪ SUMMARY OF RESULTS

At the conclusion of our trading on day 756, we end up with $1,176,043.38, representing **a 17.6% return** on our original investment. Our **Sharpe ratio was 1.1721** and we had **132 profitable days** out of 250 total possible trading days, meaning 52.8% of trading days were positive. The mean trading RMSE for our predicted values was 1.89.

In the future, in order to improve upon our results even further, we should explore additional expert variables to include in our models, other statistical models and subset selection methods, and more sophisticated trading logic based on domain knowledge.