Replicating Comparative Automated Bitcoin Trading Strategies

Writing sample from CFRM 521: Machine Learning for Finance

1 Introduction

1.1 Our Approach and Previous Work

The purpose of this paper is to replicate and validate the findings reported in a paper on machine learning algorithms for Bitcoin trading written by Hegazy and Mumford. We began by generating a dataset that is as close as possible to the original described in the paper and then we used the same methods used by the authors, including Weighted Linear Regression, Gaussian Discriminant Analysis, Logistic Regression, and Recurrent Reinforcement Learning and compared our results to those of the original paper.

1.2 Data Processing

To generate the same dataset that was used by Hegazy and Mumford, we downloaded price data from the same source, Bitcoincharts.com, and placed trades into 8-minute binds. If more than one trade existed inside of a bin, the price for the bin was found by weighting each trade by the number of shares sold at that price. The original study used two nearly consecutive 250-day windows, however, an exact start date of the dataset was not listed. In response, we made a visual approximation of the start date. There should be 45,000 observations in a 250-day window. However, in our dataset there were 12,500 observations, which suggests that there were many 8-minute windows without a trade. Although the Hegazy and Mumford paper does not discuss how the authors handled these cases, our replication attempts for Figure-2 strongly suggest that bins without trades were dropped.

Once we had a complete set of data, we attempted to smooth the data by implementing locally weighted linear regression. As a part of this procedure, the times of each of the 30 previous time bins were put in the form $(1, t_i)$ and concatenated to form a 2×30 matrix. Unfortunately, we were unable to replicate this smoothing procedure because the reference paper did not define an appropriate value for t_i.[1] Instead, we implemented a standard exponentially weighted moving average by using the stats package in R. The smoothing parameter was chosen to visually match the graph of the prices over time with the graph from the paper.[2] Figure 1 is a replication of the price chart from the original paper.

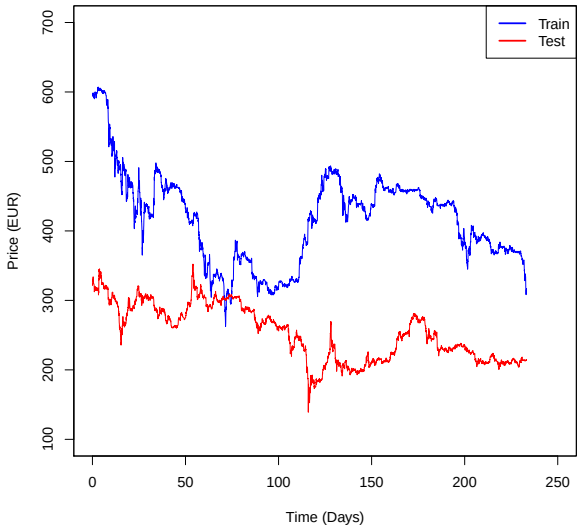

Figure 1: The binned and smoothed price data

1.3 Trading Approach and Metrics

We implemented the correct rate portion of this section. The trading metric was poorly specified in the original paper and we were unable to get it to work. With additional time, we could look more into implementing the CWC and profit but at the moment it is not clear exactly how to interpret q_i for this purpose.

The feature set $x^{(i)}$ used for all the algoritms is the first five derivatives of the price data. We show how we calculate the first derivative below.

```
derivs_ex <- dat %>%
dplyr::select(time, smooth) %>%
mutate(dt=c(NA,diff(time,diff=1))) %>%
mutate(one=c(NA,diff(smooth,diff=1))) %>%
mutate(one=one/dt)
```

2 Algorithms Examined

2.1 Weighted Linear Regression

The weighted linear regression was the simplest algorithm to implement. We applied the formula directly to the derivatives.

```
weighted <- derivs %>%
filter(time > 1.391e9) %>%
filter(time < 1.43135e9) %>%
mutate(q = 2/(1+exp(-one/mean(abs(one))))-1)
```

[1] The paper also defined an index j relative to the current index, so we know this is not a correct interpretation for t_i.

[2] Essentially, we decreased parameter until volotility spikes appeared to be the same magnitude.

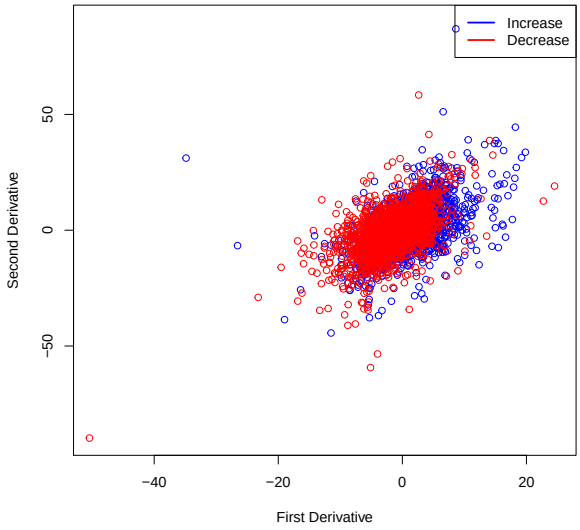

Figure 2: Spread of first and second derivatives for use in GDA

The end result is superior to what the paper found for their correct rate:

	Train & Test
New	0.6425264
Old	0.4883

2.2 Gaussian Determinant Analysis

To perform GDA we replicated the approach from the paper almost exactly[3] instead of using a built-in R package. When we replicated Figure 2 we noted that there was not a vertical line of points where the derivative was equal to zero, which suggests that the empty bins were not filled using last observation carried forward. Because the derivative graph depends on the smoothing parameter and derivative units, we did not expect the graphs to be identical. Surprisingly, the shape of the two graphs were quite different even when we adjusted the smoothing parameter. We do not think we made a mistake in calculating the derivatives so we suspect that the differing smoothing functions are the cause.

The GDA code is too long to display in the paper; our implementation can be viewed in the accompanying code file. We again failed to replicate the original results:

	Train	Test
New	0.6583505	0.6136865
Old	0.5128	0.5234

2.3 Logistic Regression

In the original paper the authors use a stochastic gradient descent to find the estimates for the logistic regression. We used the built-in glm() function in R to attempt to achieve the same result.

```
fit1 <- glm(y~one+two+three+four+five,
  family=binomial(link="logit"),data=train)
```

The end result did not match the original paper for correct rate:

	Train	Test
New	0.6405401	0.6025863
Old	0.5372	0.5504

2.4 Recurrent Reinforcement Learning (RRL)

We attempted to implement Recurrent Reinforcement Learning (RRL) using the methods outlined in the paper. As mentioned by the authors, bitcoin could not be shorted at the time, so the implementation varied slightly from a traditional approach to RRL trading algorithms. The basic idea of RRL is to maximize the Sharpe ratio for a given Bitcoin price.

Typically in RRL, gradient ascent is required to be derive the from the data. Normally, you could find $dF_t/d\theta$ from the collective time series data. The authors of this paper instead chose to use online learning to approximate $dF_t/d\theta$. This approach uses the previous partial derivative value dF_{t-1}/dw.

We attempted to implement the "Trader Function" (F_t), and all gradient equations as closely to the paper as possible. This was more difficult than anticipated as the paper had several errors in the gradient equations (Eq. 22). After implementation, testing and modifying the process several times, we were able to find a correct rate which closely matched that found by the authors. We found the correct rate to be 0.521 for the training period and 0.564 for the testing period.

We again attempted to implement the CWC and profit ratio estimate functions but found them to grossly miss-estimate the values compared to the paper. We chose to omit this portion from the results and code appendix.

The main discrepancy between our results and those from the paper, agian could have resulted from the variation in how derivatives were calculated. This is because RRL is very sensitive to small changes in the derivatives. Like in other models, the authors did

[3]There was a typo in equation 8 where the exponential was missing a negative sign.

not mention how many of the five features (derivatives) they implemented. We chose to use two since the CR estimates most closely matched the results presented in the paper.

We also found that if the time-series data is not smoothed enough then the derivatives will be too noisy to effectively perform RRL. Our implementation of RRL was affected by the sparse information presented in the paper on how pre-processing was performed.

3 Analysis: Comparing Algorithm Performance

Clearly we were unable to replicate the results from any algorithm reported in the Hegazy and Mumford paper. Our first observation in our analysis is that Figure 2 is strikingly different from that in the original paper. Since the derivatives are the inputs to all of the algorithms it is not surprising that the results are very different. It is unclear why the derivatives are so different but we suspect at least part of the difference is the smoothing function. One interesting issue to note about the derivatives plot in the original paper is that there appears to be a distinct diagonal line through the data, which is absent in our plot. This could be a feature of the smoothing function used by Hegazy and Mumford, or it could suggest a mistake.

Another issue to note about their paper is that it consistently has the test dataset outperform the training dataset. In our analysis we are unable to replicate this feature in any algorithm, despite trying a wide range of smoothing parameters.

We concluded that our attempt to replicate the findings of the Hegazy and Mumford paper could have been greatly improved if their paper had more clearly described the exact procedures used in the analysis. Attempting to replicate the orginal results entailed substantial guesswork and so it is unsurprising that we could not replicate the reported findings. If we had more time we might have reached out to the authors to better understand their approach.