MachineLearny McMachineLearningFace Final Report

Nikolai Lipscomb, Nail Karabay, Max Klevgard, Keliang Gao

It is **ok** to use an anonymized version of this report as an example of a project for future classes.

Summary

Financial markets are one of the most fascinating inventions of our time. They have had a significant impact on many areas like business, education, jobs, technology and thus on the economy (Hiransha et al. 2018; Shah et al. 2019).

Over the years, investors and researchers have been interested in developing and testing models of stock price behavior (Fama 1995). However, analyzing stock market movements and price behaviors is extremely challenging because of the markets dynamic, nonlinear, nonstationary, nonparametric, noisy, and chaotic nature (Abu-Mostafa and Atiya 1996). According to Zhong and Enke (2017), stock markets are affected by many highly interrelated factors that include economic, political, psychological, and company-specific variables.

Technical and fundamental analysis are the two main approaches to analyze the financial markets (Park and Irwin 2007; Nguyen et al. 2015). To invest in stocks and achieve high profits with low risks, investors have used these two major approaches to make decisions in financial markets (Arévalo et al. 2017).

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable.

In this study, the authors collected dataset of three years. The first two years are assigned to the training set, and the last year to the test set.

Different reliable sources like blockchain.com, coinmarketcap, yahoofinance, and investing.com are adopted. Data was compiled and cleaned by Nail.

Multiple predictor variable are adopted in this study, totally there are 37 variables. Below are details about the response and examples of some predictors.

Response: BTC Price: BTC Price in $ (Note: for autoregressive purposes, BTC Price on day n can be used as a predictor for BTC price on day n+1, which will reduce the number of observations in a sequence by one).

BTC in circulation: The total number of bitcoins that have already been mined.

Market Cap: The total USD value of bitcoin supply in circulation.

Total Trade (USD): The total USD value of trading volume on major bitcoin exchanges.

Block Size(MB): The total size of all block headers and transactions.

Avg Block Size(MB): The average block size in MB.

Avg # of Transactions: The average number of transactions per block.

Median Transaction Time: The median time for a transaction to be accepted into a mined block.

Hash Rate: The estimated number of tera hashes per second the Bitcoin network is performing.

Difficulty: A relative measure of how difficult it is to find a new block.

The classes of models were developed to attempt to predict stock. These include the classical linear model and its variations (Ridge, LASSO, Elastic Net), ARIMA models (both with just the response, then incorporating covariates), and Hidden Markov Model Regression.

Linear Model

For variable preselection: first, we found a correlation matrix between all predictors. If the absolute value of correlation amount between two predictors is bigger than α, we removed one of them from the model randomly. We followed this procedure for α=1, 0.9, 0.8, 0.7, 0.6, and 0.5 by applying 10-fold CV to find linear regression model. Results can be seen in Table 1 where q represents the number of predictors left after remove process is done.

α	1	0.9	0.8	0.7	0.6	0.5
q	34	24	18	10	9	8
Test MSE	1984061.0	149975.7	927081.2	697708.1	901448.5	144406703.0
Training MSE	118397.3	126930.2	132385.5	136909.2	137598.4	3641373.0
Adjusted R^2	0.99249	0.99206	0.99179	0.99160	0.99157	0.77728

Table 1. Results of linear regression models for different α values

As expected, best training MSE and adjusted R^2 are obtained when all predictors are used in the model. Best test MSE is obtained when α=0.9. It is hard to tell that there is a trend with α values. However we can say that while decreasing α after α=0.7, the training MSE is worsened clearly. Adjusted R^2 doesn't say anything meaningful in this case.

For subset selection: first, we tried to apply best subset selection method but it is impossible to calculate computationally. Then we built regression models from a set of candidate predictor variables by entering predictors based on p values and AIC (Akaike Information Criteria), in a stepwise manner until there is no variable left to enter any more which is called forward stepwise regression(FSR). Also we built regression models from a set of candidate predictor variables by removing predictors based on p values and AIC, in a stepwise manner until there is no variable left to remove any more which is called backward stepwise regression (BSR). Lastly we built regression models from a set of candidate predictor variables by entering and removing predictors based on p values and AIC, in a stepwise manner until there is no variable left to enter or remove any more which is called stepwise regression(SR). Results can be seen in Table 2 where q represents the number of predictors left after stepwise selection is applied. Best test MSE is obtained with the model which is found by FSR (AIC) or SR (AIC). It is hard to say why those

methods dominates the others in this case. Also, this model also has better test MSE than the test MSE of the best model which is found by applying variable pre-selection method.

Method	FSR (p)	BSR (p)	SR (p)	FSR (AIC)	BSR (AIC)	SR (AIC)
q	24	26	12	15	24	15
Test MSE	3674789.0	1137984.0	232843.8	121054.1	1359378.0	121054.1
Training MSE	286185.3	118868.4	125702.7	124535.3	119293.6	124535.3
Adjusted R^2	0.98210	0.99254	0.99227	0.99231	0.99254	0.99231

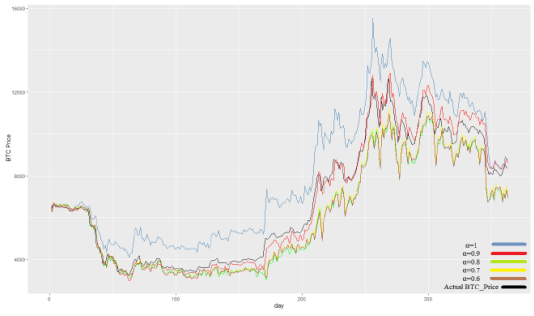
Table 2. Results of linear regression models for stepwise methods

We found ridge, lasso and elastic.net models by changing alpha values in glmnet. First, we calculated lambda value which gives smallest MSE for each method. Then with that lambda we constructed regression models. Results can be seen in the Table 3. There is a considerable improvement in MSPE of Lasso compared to MSPE of Ridge. Lasso makes coefficient of only 9 predictors nonzero. Although it is far more better than result of ridge, it is dominated by best models of variable-pre selection and subset selection. We improved that result by applying Elastic.net which makes coefficient of 13 predictors nonzero. Although it dominates the best result of variable-pre selection method, it is dominated by the best model of subset selection. Lastly, we did Principle Component Regression (PCR). It gives terrible MSPE in this problem. Overall, the best model is obtained by applying subset selection.

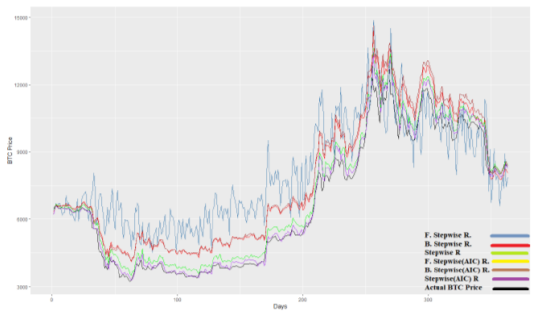
Method	Ridge	Lasso	Elastic.Net	PCR
MSPE	2090701.0	232409.6	140375.3	995458.4

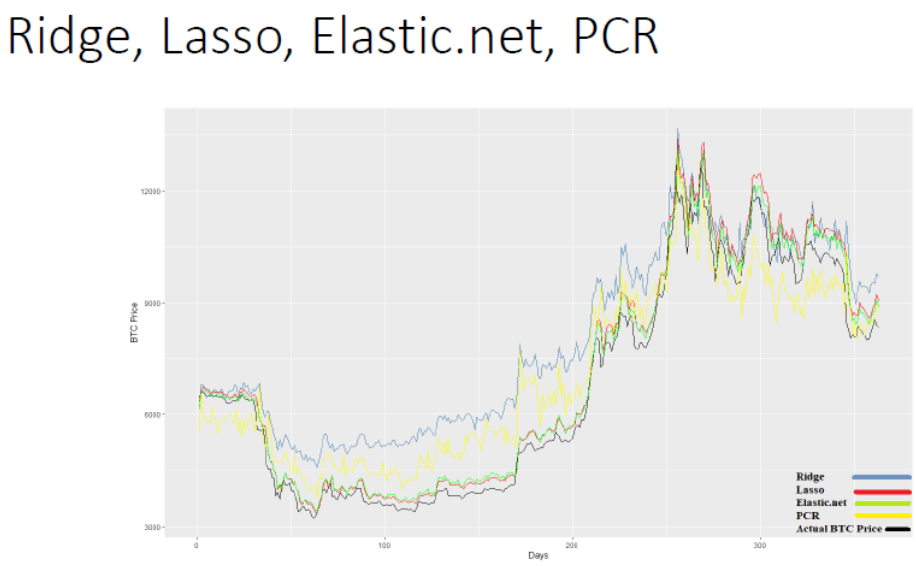
Table 3. Results of regression models for different methods

Variable Pre-Selection



Subset Selection





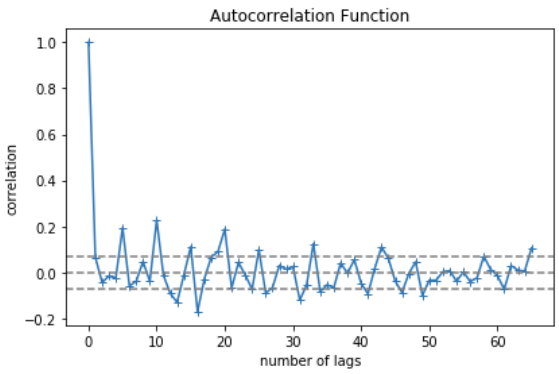
These graphs present prediction results from various models (black is actual test data).

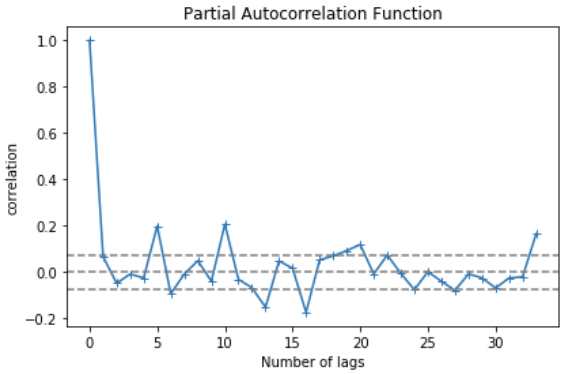
Univariate ARIMA Model

This model is used by fitting previous values of a time series - in this case the bitcoin price - to current values to help better understand the data and predict future prices. The AR part of ARIMA indicates that the changing variable of interest (BTC Price) is regressed on its own lagged values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred. The I (for "integrated") indicates that the data values have been replaced with the difference between their values and the previous values. The importance of each of these features to the ARIMA model is so that the model fits the data as best as possible for prediction.

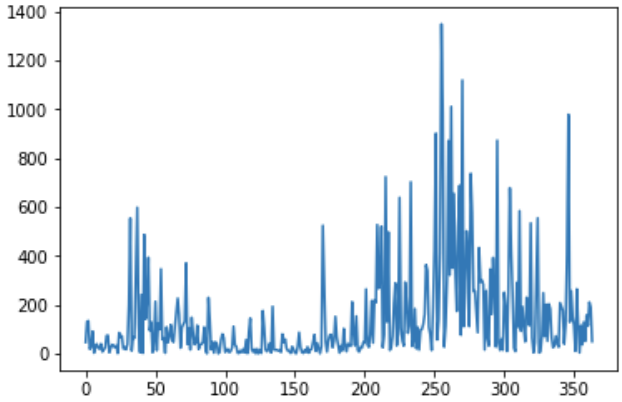
The ARIMA model uses three parameters: p, d, and q. When deciding "d", the amount of differencing needed for the time series data, we use the Augmented Dickey-Fuller (ADF) test. This tests if the data is stationary or non-stationary. Stationary data exhibits a similar mean and variance over time, which is required to perform the ARIMA process. For deciding p and q, we use the PACF and ACF tests respectively to find the optimal lag to use for the model. The p value is the lag window for the actual BTC price, while the q value is the lag window for the error terms.

Based on ACF and PACF, we determine the best model uses lag 1.





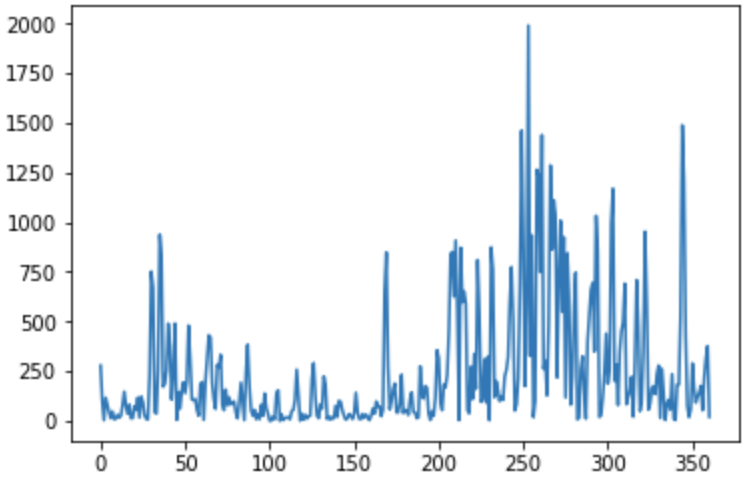
Average error across Test data is approximately 143.2573 with the absolute prediction errors across the one-year test set shown below.



Multivariate ARIMA Model

In the multivariate ARIMA model, sometimes called the ARIMAX model, we assume that the time series in question (BTC Price) is affected by another time series occuring simultaneously. To see whether or not another time series has a significant effect on the bitcoin price, we used the Granger Causality test. This test determines whether one time series is useful in forecasting another. In this case, we examined different time series (such as mining difficulty, S&P 500 index, average # of transactions, etc.) to see if they can help better predict future prices for BTC. We needed to make all of the column data stationary, so that we do not get any misleading results from the Granger Causality Test. We ensured that each column is individually differenced the correct amount of times (such that Augmented Dickey Fuller test passes to that 1% significance threshold) so the causality test can be performed properly. The gives us an array that contains the amount of times each column was differenced, and it will also change the training data so that it contains the new differenced data. Looking through the information from the tests, we can identify that Total Trade (USD), Cost per Transaction, XRP Price, and XRP Volume are the best time series correlations to BTC Price.

Average error across Test data is approximately 244.3112 with the absolute prediction errors across the one-year test set shown below.



We can see that using the Total Trade (USD) time series in the multivariate ARIMA model does not perform as well as the original univariate ARIMA model. This trend also continues with the other three datasets of Cost per Transaction, XRP Price, and XRP Volume. Comparing solely the multivariate ARIMA models, Total Trade (USD) performed the best with a test MSE of 606040.805, followed by Cost per Transaction, XRP Price, then XRP Volume, with test MSEs of 60834.406, 60846.867, and 61540.787 respectively.
Overall, we see that the simple univariate ARIMA model works the best. This result makes sense for a couple of reasons. First, in the multivariate model, it makes sense that variables such as XRP Price and Volume would not be very indicative of bitcoin price considering that it is a much less used cryptocurrency comparatively. If anything, BTC price would probably have a bigger effect on the prices of other, less traded, options like XRP and Litecoin. Secondly, bitcoin prices are still being adopted into society at a slow rate, and the idea that other time series data is indicative of its future price contradicts the belief that BTC is too volatile and unstable to predict.

Hidden Markov Model

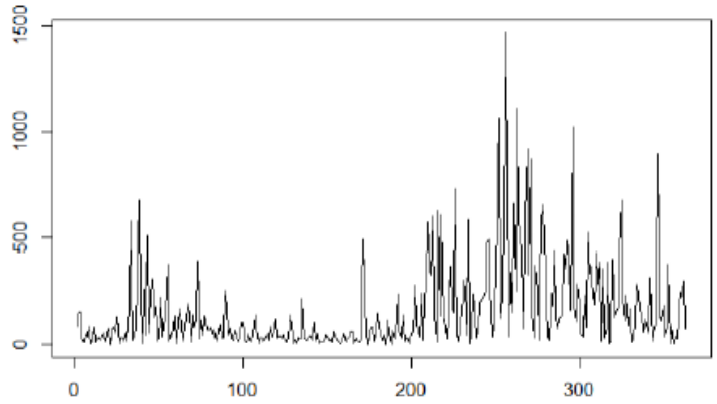
In the Hidden Markov Model, we make use of the theory developed in Zucchini, Macdonald, and Langrock (2016). A (first-order) Hidden Markov Model is fit to the training data bitcoin price using expectation maximization of the data likelihood. Estimates provided include an initial distribution for the Markov chain δ, transition probability matrix P, and parameters for the mixture distributions which are assumed to all by Gaussians. BIC criteria were used to select 8 states as the optimal number of states. Estimates are seen below.

$$\delta = [0, 1, 0, 0, 0, 0, 0, 0]$$

$$\mathbf{P} = \begin{bmatrix} 0.994 & 0 & 0 & 0 & 0.006 & 0 & 0 & 0 \\ 0 & .921 & 0 & 0 & 0 & 0 & .079 & 0 \\ 0 & 0 & .983 & 0 & .017 & 0 & 0 & 0 \\ 0.019 & 0 & 0 & .962 & 0 & 0 & 0 & .019 \\ 0 & 0 & .016 & 0 & .984 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & .924 & 0 & .076 \\ 0 & 0 & 0 & 0 & 0 & 0.028 & .972 & 0 \\ 0 & 0 & 0 & 0.028 & 0 & .014 & 0 & .958 \end{bmatrix}$$

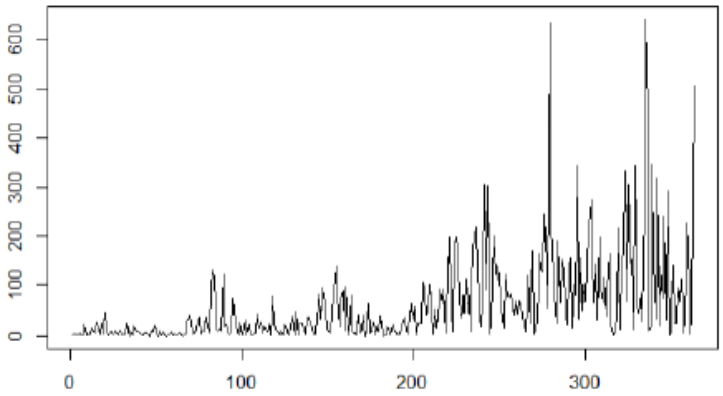
The Viterbi algorithm is also implemented to "cluster" or assign classes to the states in the training set, then a distinct linear model is trained for each of the eight subsets of the training data. The linear models are distinct in terms of their parameter estimates; however, the set of predictors used is the same for each with selection based on the smallest subset of significant predictors across the 8 models. Using the forward recursive form of the likelihood, we are able to determine probabilities for future states and develop predictions based on a probability-weighted average of the states' linear model predictions.

Below are the absolute residuals from the prediction compared against the true prices from the test set. The average absolute prediction error is approximately 150.



Conclusions

While some predictions appear to be quite spectacular since the range of bitcoin price is in the tens of thousands, we became aware of a simple explanation for the "prediction power" of our models. Consider the simple prediction rule: $\hat{Y}_t = \hat{Y}_{t-1}$; that is, use yesterday's price to predict today's price. The average error from this "silly" prediction rule is around 60 on average with the absolute prediction errors for the test set displayed below.



Even the Hidden Markov Model, univariate ARIMA, and better linear models, which appear to work very well from a cursory inspection, performed poorly in comparison. This is not too much of a surprise considering the volatile nature of currencies, and Bitcoin itself is an extreme example of volatility. Of course, the field of currency speculation would be "solved" if such volatility were not an issue. Future research considerations include mining better predictors such as text data in news and social media regarding bitcoin (positive vs. negative tweets, for example). Other future considerations include

diversifying the types of parametric distributions within a hidden Markov model, since only Gaussians were used in this analysis.

Division of Labour

Coordinating weekly assignments/meetings: Nikolai

Weekly Journals: Nikolai (Nail if Nikolai unavailable)

Data Preparation: Nail

Linear Models: Nail (R)

ARMA/ARIMA Models: Max (Python)

Hidden Markov Model: Nikolai (R)

Presentation: Keliang (MSPP)

Presentation Speakers: Everyone

Final Report Author: Everyone submits individual summaries: Keliang (Intro, summary, and references), Nail (LMs), Max (ARIMAs), Nikolai (HMM). Final report is compiled and edited by Nikolai.

References

Abu-Mostafa, Yaser S., and Amir F. Atiya. 1996. Introduction to Financial Forecasting. Applied Intelligence 6: 205-13.

Arévalo, Rubén, Jorge García, Francisco Guijarro, and Alfred Peris. 2017. A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. Expert Systems with Applications 81: 177-92.

Cryer Jonathan D., Chan Kung-Sik. 2009. Time Series Analysis with Applications in R. 2nd Edition, Springer Texts in Statistics.

Dev Shah, Haruna Isah, and Farhana Zulkernine, Stock Market Analysis: A Review and Taxonomy of Prediction Techniques, Int. J. Financial Stud. 2019, 7, 26; doi:10.3390/ijfs7020026.

Fama, Eugene F. 1995. Random walks in stock market prices. Financial Analysts Journal 51: 75-80. [CrossRef]

Hiransha, M., E. A. Gopalakrishnan, Vijay Krishna Menon, and Soman Kp. 2018. NSE stock market prediction using deep-learning models. Procedia Computer Science 132: 1351-62.

Nguyen, Thien Hai, Kiyoaki Shirai, and Julien Velcin. 2015. Sentiment Analysis on Social Media for Stock Movement Prediction. Expert Systems with Applications 42: 9603-11.

Park,Cheol-Ho, and ScottH.Irwin. 2007. What do we know about the profitability of technical analysis? Journal of Economic Surveys 21: 786-826.

Zucchini Walter, MacDonald Iain L., Langrock Roland, 2016. Hidden Markov Models for Time Series: An Introduction Using R, 2nd Edition, CRC Press.

Zhong, Xiao, and David Enke. 2017. Forecasting daily stock market return using dimensionality reduction. Expert Systems with Applications 67: 126-39.