**Analysis of the Bitcoin Price**

**Introduction**

Bitcoin, Cryptocurrency has grown exponentially in value, resulting in it being the frequent topic of conversation and news as its value has grown in last few years. The focus of this analysis is to try and predict Bitcoin daily close price. Although there are different techniques to model market data this analysis will focus on regression and time series models.

**Data Description**

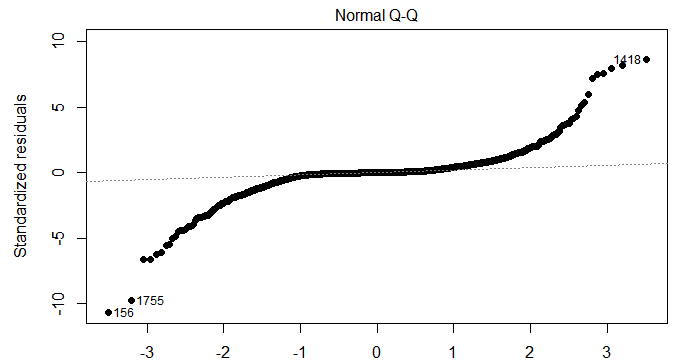
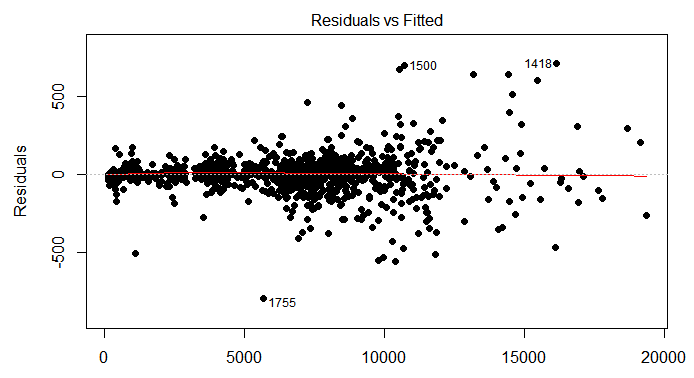
For this analysis two types of data were collected BTC market data (source: investing.com) and Bitcoin mining data (Source: https://www.quandl.com/data/BITCOINWATCH/MINING-Bitcoin-Mining-Statistics). Market data includes the close price which we are attempting to predict along with other market specific data such as Open and close. The Bitcoin mining data includes information about the Bitcoin network such as hash rate and difficulty but it also includes some market data including total BTC blocks and Market cap. The specifics of the requested analysis will determine which variables were used. The complete list of Data Variables can be found in the appendix.

# **Objective 1**

**Exploratory Analysis**

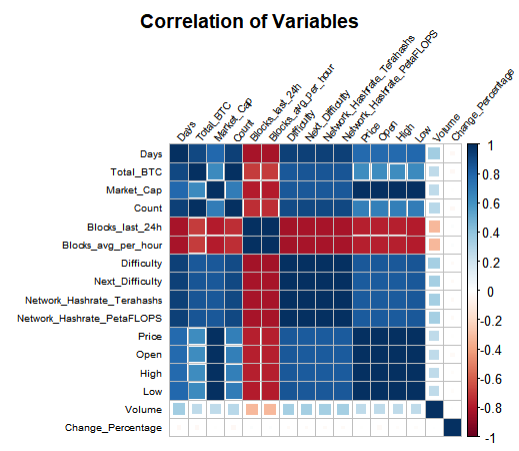
Data Transformation

Judging by the scatter plot and q-q plot matrices below we decided to proceed without any transformation of data to use with linear regression. Full output can be found in the appendix.



Correlation of variables

Judging by correlation plot, the variables closely related to Bitcoin Price are Days, Total\_BTC, Market\_Cap, Count (Total number of Bitcoins ), Difficulty, Next\_Difficulty,Network\_Hashrate\_Terahashs,Network\_Hashrate\_PetaFLOPS,Price,Open,High and Low.



Problem Statement

To find best predictive model for estimating the Bitcoin price using linear regression for Observation 1 task. We are using variables identified as correlated in the exploratory analysis to build best model for the price of Bitcoin. To accomplish this, we will use two different methods of variable selection: **Custom-Built** using regression model and **LASSO** regression. LASSO regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean.

Model Selection: Custom-Built

The first model selection, we have created a custom model based on the correlation matrices with 8 parameters, providing an Adjusted R2 of 0.9995 and optimal value of CV Press of 12958128 and AIC of 18172.36

Residuals:

Min 1Q Median 3Q Max

-759.70 -7.35 0.23 15.20 689.36

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -33.9197469501325 54.6928733742204 -0.620 0.5352

Market\_Cap 0.0000000108281 0.0000000006290 17.215 <2e-16 \*\*\*

Network\_Hashrate\_Terahashs 0.0000158631536 0.0000289343700 0.548 0.5836

Difficulty -0.0000000001290 0.0000000002016 -0.640 0.5225

Total\_BTC 0.0000044433812 0.0000042846902 1.037 0.2999

Days -0.0346772747405 0.0144270493628 -2.404 0.0164 \*

Open -0.4509278232801 0.0148638986047 -30.337 <2e-16 \*\*\*

High 0.7404810209817 0.0162649693953 45.526 <2e-16 \*\*\*

Low 0.5341945679153 0.0128827032967 41.466 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 89.04 on 1528 degrees of freedom

Multiple R-squared: 0.9995, Adjusted R-squared: 0.9995

F-statistic: 3.564e+05 on 8 and 1528 DF, p-value: < 2.2e-16

AIC(lm\_FITTrain) :18172.36

PRESS(lm\_FITTrain): 12958128

Assumptions

|  |  |
| --- | --- |
| **Normality**:  Judging from the scatter plot and q-q plot,  there is slight evidence against normality. However, due  to the large sample size, it is not an issue.  **Linear Trend**: Judging by the predicted value plot, there  is no visual evidence against linear trend.  **Constant Standard Deviation**: There is no significant  visual evidence of heteroscedasticity.  **Independence**: We assume the observations are independent  **Influential Points**: Judging from the Cooks D chart, it appears that there are some observations that are  outside of the normal(1418 and 1755), however they do not pose an issue. |  |

Parameter Interpretation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Beta | Estimate Beta (Value) | Significant in this model at alpha = 0.05 | p-value | Analysis |
| Intercept | β0 | -34 | No | 0.54 | When all other variables are negligible the mean Bitcoin Price will be decreased by 34.  (It means that the mean effect of all omitted variables may not be important, however, that does not mean that constant should be taken out because it does two other things in an equation. It is a garbage term and it forces the residuals to have a zero mean) |
| Market\_Cap | β1 | 0.000000011 | Yes | 7.7e-61 | When all other variables are held constant, mean Bitcoin Price increase by 0.000000011 for 1 unit increase of Market\_CAP. |
| Network\_Hashrate\_Terahashs | β2 | 0.000016 | No | 0.58 | This is insignificant for interpretation. |
| Difficulty | β3 | -0.00000000013 | No | 0.52 | This is insignificant for interpretation. |
| Total\_BTC | β4 | 0.0000044 | No | 0.3 | This is insignificant for interpretation. |
| Days | β5 | -0.035 | Yes | 0.016 | When all other variables are held constant, mean Bitcoin Price decreases by 0.035 for 1 day increase |
| Open | β6 | -0.45 | Yes | 1.2e-158 | When all other variables are held constant, mean Bitcoin Price decreases by 0.45 for 1 unit increase of Open Price |
| High | β7 | 0.74 | Yes | 1.1e-286 | When all other variables are held constant, mean Bitcoin Price increases by 0.74 for 1 unit increase of High Price |
| Low | β8 | 0.53 | Yes | 2e-252 | When all other variables are held constant, mean Bitcoin Price increases by 0.53 for 1 unit increase of Low Price |

Model Selection: LASSO

The second model selection, we have created LASSO regression model based with all the parameters, providing an Adjusted R2 of 0.9995 and optimal value of CV Press of 163347531 and AIC of 15824

Assumptions

|  |  |
| --- | --- |
| **Normality**:  Judging from the scatter plot and q-q plot,  there is slight evidence against normality. However, due  to the large sample size, it is not an issue.  **Linear Trend**: Judging by the predicted value plot, there  is no visual evidence against linear trend.  **Constant Standard Deviation**: There is no significant  visual evidence of heteroscedasticity.  **Independence**: We assume the observations are independent  **Influential Points**: Judging from the Cooks D chart, it appears that there are some observations that are  outside of the norm(1418 and 1755), however they do not pose an issue. |  |

Parameter Interpretation

Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. As you can see from the glmnet output it has shrunk to 3 variables excluding the intercept.

|  |  |
| --- | --- |
| (Intercept) 75.70328971338677  (Intercept) .  Days .  Date .  Total\_BTC .  Market\_Cap 0.00000001371174  Count .  Blocks\_last\_24h .  Blocks\_avg\_per\_hour .  Difficulty .  Next\_Difficulty .  Network\_Hashrate\_Terahashs .  Open .  High 0.50214346128092  Low 0.24621234362938  Volume .  Change\_Percentage . | Some information I couldn’t get through r lang glmnet, so I have got it through SAS |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable | Beta | Estimate Beta (Value) | Significant in this model at alpha = 0.05 | p-value | Analysis |
| Intercept | β0 | 77 | Yes |  | When all other variables are negligible the mean Bitcoin Price will be increased by 76.  (It means that the mean effect of all omitted variables may not be important, however, that does not mean that constant should be taken out because it does two other things in an equation. It is a garbage term and it forces the residuals to have a zero mean) |
| Market\_Cap | β1 | 0.00000001371174 | Yes |  | When all other variables are held constant, mean Bitcoin Price increase by 0.00000001371174 for 1 unit increase of Market\_CAP. |
| High | β7 | 0.50 | Yes |  | When all other variables are held constant, mean Bitcoin Price increases by 0.50 for 1 unit increase of High Price |
| Low | β8 | 0.25 | Yes |  | When all other variables are held constant, mean Bitcoin Price increases by 0.25 for 1 unit increase of Low Price |

Cross Validation

|  |  |
| --- | --- |
| When we ran the predictions with both Custom Fit and LASSO model prediction fit perfectly with actuals. This phenomenon is due to direct correlation of Price with Market Cap, Low , High and Open. | By removing highly correlated variables like Market Cap, Low , High and Open, this is how trend looks like. |
|  |  |

Final Model Comparisons:

|  |  |  |  |
| --- | --- | --- | --- |
| **Predictive Models** | **Adjusted R2** | **CV PRESS** | **AIC** |
| CUSTOM | .9995 | 12958128 | 18172 |
| LASSO | .9992 | 163347531 | 15824 |

Comparing both the models, they both are same with a very minute upper hand with CUSTOM model with the lower CV PRESS and AIC.

**Conclusion**

Further we will extent the model by calculating the Market Cap for today by taking yesterday’s Price and on though regression.

# **Objective 2**

For objective 2 we choose to go with Time series with the goal of predicting close price per day. Time series was chosen as the method for Objective 2 due to the nature of the data collected. Given the market data and Bitcoin network data was collected on a daily basis and was already in a time series format going with that type of analysis made sense.

**Time Series Analysis**

The complete data set was first plotted out along with ACF and PACF plots to check for autocorrelation. The ACF plot shows constant autocorrelation as the plot begins to tail off the further out the lag. The Partial ACF shows the greatest autocorrelation at lag 1. For comparison a log transformation of the price data was performed and looked to

A screenshot of a cell phone

Description automatically generatedA picture containing screenshot

Description automatically generatedA screenshot of a social media post

Description automatically generated

*Durbin-Watson Test*

To confirm that autocorrelation is indeed happening a Durbin-Watson test was performed. Ho: The residuals are not autocorrelated Ha: The residuals show to be autocorrelated. At significance level of alpha = .05 we reject the null hypothesis and conclude that indeed the residuals are autocorrelated.

A screenshot of a cell phone

Description automatically generated

*Correlation with Price*

The correlation of price with other variables was also check in this part of the analysis. As Identified in the previous analysis there are several variables that correlate closely with Price. Market cap was closely correlated and chosen for further analysis

A picture containing circuit, computer

Description automatically generated

*Market Cap as predictor*

Market cap was found to be too closely correlated with Price and after further thought on this predictor given that Market cap is a product of total blocks \* close price using this as a predictor would not make sense in a real world situation.

A close up of a map

Description automatically generated

**Model**

Instead of using all data available a one year window of the total data was used to construct the following models with a prediction window of 10 days.

Time Series Model 1 (no Predictors): This model did not do a very good job of predicting with an AIC =5126.66 and a RMSE of 329.47.

A picture containing knife

Description automatically generatedA picture containing knife

Description automatically generatedA screenshot of a cell phone

Description automatically generated

A close up of a map

Description automatically generated

**Conclusion of TS analysis:**

During the analysis phase several different approaches were taken including adding predictors and even log transforming the price data in an attempt to get a better result. The predictors that were tried were count, market cap, and difficulty. None of these predictors added to the TS model. Lessons learned from the time series analysis include focusing on a time period that makes sense. During my original analysis all the data was used which included a lot of early data that was probably no longer relevant. The model could be improved by adding predictors but given the data set the most relevant predictors that would improve the model such as Market cap, open, high, and low were too closely correlated with the Price and would not be good predictors in a real-world situation.

**Final Conclusion of Analysis:**

After comparing the regression model and the TS model the regression model looks to do a better job of predicting price based on the data that we collected. The nature of this data is very much time series in nature though and given a more robust set of predictor variables the TS model could probably be improved upon. Given the volatile nature of the Bitcoin market it is very difficult to build to predict price that could be used in real world situation. The market and network data that we collected did not have many variables that ended up being useful in our models. If we were to attempt this analysis again we would most likely look to source other types of data that might be more useful in constructing our models than Bitcoin Network data. Some alternatives to BTC network data include S&P 500 data and/or data around Bitcoin social sentiment.

***Appendix A –Objective1 Code***

***---***

***title: "Bitcoin Data Analysis"***

***author: "Rajesh satluri"***

***date: "2/11/2020"***

***output: html\_document***

***---***

***```{r setup, include=FALSE}***

***knitr::opts\_chunk$set(echo = TRUE)***

***```***

***## R Markdown***

***This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.***

***When you click the \*\*Knit\*\* button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:***

***# Cleaning up dataset***

***```{r cars}***

***library(knitr)***

***library(pander)***

***library(tidyverse)***

***library(broom)***

***library(scatterplot3d)***

***library(DataCombine)***

***library(corrplot)***

***library(caret)***

***library(kableExtra)***

***library(lubridate)***

***library(stringr)***

***bc\_data <- read.csv(file.choose(),header = TRUE)***

***bc\_data\_test <- read.csv(file.choose(),header = TRUE)***

***colnames(bc\_data) # get column names***

***summary(bc\_data) # to find null values in each column***

***#Update Column names***

***bc\_data <- bc\_data %>%***

***rename(***

***Days = Days,***

***Total\_BTC = Total.BTC,***

***Market\_Cap = Market.Cap,***

***# Transactions\_last\_24h = Transactions.last.24h,***

***# Transactions\_avg\_per\_hour = Transactions.avg..per.hour,***

***# Bitcoins\_sent\_last\_24h = Bitcoins.sent.last.24h,***

***# Bitcoins\_sent\_avg\_per\_hour = Bitcoins.sent.avg..per.hour,***

***Blocks\_last\_24h = Blocks.last.24h,***

***Blocks\_avg\_per\_hour = Blocks.avg..per.hour,***

***Next\_Difficulty = Next.Difficulty,***

***Network\_Hashrate\_Terahashs = Network.Hashrate.Terahashs,***

***Network\_Hashrate\_PetaFLOPS = Network.Hashrate.PetaFLOPS,***

***Volume = Vol.,***

***Change\_Percentage = Change..,***

***)***

***bc\_data$Date <- as.Date(bc\_data$Date)***

***bc\_data$Price <- as.numeric(bc\_data$Price)***

***#Update Column names***

***bc\_data\_test <- bc\_data\_test %>%***

***rename(***

***Days = Days,***

***Total\_BTC = Total.BTC,***

***Market\_Cap = Market.Cap,***

***# Transactions\_last\_24h = Transactions.last.24h,***

***# Transactions\_avg\_per\_hour = Transactions.avg..per.hour,***

***# Bitcoins\_sent\_last\_24h = Bitcoins.sent.last.24h,***

***# Bitcoins\_sent\_avg\_per\_hour = Bitcoins.sent.avg..per.hour,***

***Blocks\_last\_24h = Blocks.last.24h,***

***Blocks\_avg\_per\_hour = Blocks.avg..per.hour,***

***Next\_Difficulty = Next.Difficulty,***

***Network\_Hashrate\_Terahashs = Network.Hashrate.Terahashs,***

***Network\_Hashrate\_PetaFLOPS = Network.Hashrate.PetaFLOPS,***

***Volume = Vol.,***

***Change\_Percentage = Change..,***

***)***

***bc\_data\_test$Date <- as.Date(bc\_data\_test$Date)***

***bc\_data\_test$Price <- as.numeric(bc\_data\_test$Price)***

***#bc\_data$Days <- 1:nrow(bc\_data)***

***ggplot(bc\_data, aes(bc\_data$Date, bc\_data$Price)) +***

***geom\_point(color="firebrick") +***

***ggtitle('BitCoin Value vs. Time') +***

***theme(plot.title = element\_text(size=20, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Date", y="USD")+***

***theme(axis.text.x=element\_text(angle=50, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey90'))***

***```***

***#Summary of the Bitcoin Dataset***

***##Correlation between variables***

***```{r}***

***cor <- cor(bc\_data[,c(1,3:17)]) #selecthead()ing variables to include in correlation analysis***

***colnames(cor) <- c("Days","Total\_BTC", "Market\_Cap","Count","Blocks\_last\_24h","Blocks\_avg\_per\_hour","Difficulty","Next\_Difficulty","Network\_Hashrate\_Terahashs","Network\_Hashrate\_PetaFLOPS","Price","Open","High","Low","Volume","Change\_Percentage")***

***rownames(cor) <-c("Days","Total\_BTC", "Market\_Cap","Count","Blocks\_last\_24h","Blocks\_avg\_per\_hour","Difficulty","Next\_Difficulty","Network\_Hashrate\_Terahashs","Network\_Hashrate\_PetaFLOPS","Price","Open","High","Low","Volume","Change\_Percentage")***

***corrplot(cor, method = "square", tl.srt = 50, tl.col = "black", tl.cex = 0.6, title = "Correlation of Variables", mar=c(0,0,1,0))***

***```***

***The variables closest related to Price are Days,Total\_BTC, hightly related to Market\_Cap, Count(Total number of BitCoins),Difficulty, Next\_Difficulty,Network\_Hashrate\_Terahashs,Network\_Hashrate\_PetaFLOPS,Price,Open,High and Low***

***##Market Price vs. Market Cap***

***```{R}***

***ggplot(bc\_data, aes(bc\_data$Market\_Cap, bc\_data$Price)) +***

***geom\_point(color="firebrick") +***

***ggtitle('Bitcoin Market Capitalization vs. Bitcoin Market Price') +***

***theme(plot.title = element\_text(size=19.5, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Market Cap (USD)", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=50, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***stat\_smooth(method = "lm", formula = y ~ x, col = "yellow")***

***```***

***##Regression Model Summary***

***```{R}***

***lm\_FitMarketCap<-lm(bc\_data$Price~bc\_data$Market\_Cap)***

***panderOptions("digits", 2)***

***pander(lm\_FitMarketCap, caption = "Linear Model: Market Price ~ Market Capitalization")***

***R1=summary(lm\_FitMarketCap)$r.squared***

***cat("R-Squared = ", R1)***

***```***

***As you can see above Market capital is significant to Bitcoin Market Price***

***##Residuals Plots for Market Price vs Market Capitalization***

***```{R}***

***plot(lm\_FitMarketCap, pch=20, which=1)***

***```***

***Residual Plot appears to be scatterd across the x-axis eventhough there is more concentration in the starting and middle***

***##Difficulty vs. Market Price***

***```{R}***

***ggplot(bc\_data, aes(bc\_data$Difficulty, bc\_data$Price)) +***

***geom\_point(color="firebrick") +***

***ggtitle('Difficulty vs. Market Price') +***

***theme(plot.title = element\_text(size=19.5, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Difficulty", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=90, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***stat\_smooth(method = "lm", formula = y ~ poly(x,2), col = "yellow")***

***```***

***##Regression Model Summary***

***```{R}***

***lm\_FitDifficulty<-lm(bc\_data$Price~poly(bc\_data$Difficulty,2))***

***panderOptions("digits", 2)***

***pander(lm\_FitDifficulty, caption = "Linear Model: Market Price ~ Difficulty squared")***

***R2=summary(lm\_FitDifficulty)$r.squared***

***cat("R-Squared = ", R2)***

***```***

***Polynomial transformation fixed heteroscedasticity to some extent. And it looks much better as compared to log and no transformation.***

***##Residuals***

***```{R}***

***plot(lm\_FitDifficulty, pch=16, which=1)***

***```***

***##Hash Rate vs. Market Price***

***```{R}***

***ggplot(bc\_data, aes(bc\_data$Network\_Hashrate\_Terahashs, bc\_data$Price)) +***

***geom\_point(color="firebrick") +***

***ggtitle('Bitcoin Hash Rate vs. Market Price') +***

***theme(plot.title = element\_text(size=19.5, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Tera Hash Rate", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=90, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***stat\_smooth(method = "lm", formula = y ~ poly(x,2), col = "yellow")***

***```***

***##Regression Model Details***

***```{R}***

***lm\_FitTerahashrate<-lm(bc\_data$Price~poly(bc\_data$Network\_Hashrate\_Terahashs,2))***

***panderOptions("digits", 2)***

***pander(lm\_FitTerahashrate, caption = "Linear Model: Market Price ~ Hash Rate")***

***R5=summary(lm\_FitTerahashrate)$r.squared***

***cat("R-Squared = ", R5)***

***```***

***##Residuals***

***```{R}***

***plot(lm\_FitTerahashrate, pch=16, which=1)***

***```***

***#What is the signifigance of these variables to the market price?***

***##All variables***

***```{R}***

***lnFITALL <- lm(bc\_data$Price~., bc\_data)***

***panderOptions("digits", 2)***

***pander(lnFITALL, caption = "Linear Model: Market Price vs. All Variables")***

***Rb1=summary(lnFITALL)$r.squared***

***cat("R-Squared = ", Rb1)***

***```***

***##Residuals***

***```{R}***

***plot(lnFITALL, pch=16)***

***```***

***Most of the volume is located in the first half of x-region, but the trend line relatively flat.***

***##Highly Correlated Variables only***

***```{R}***

***ln\_FITHighlyCorelated <- lm(bc\_data$Price~bc\_data$Market\_Cap+bc\_data$Network\_Hashrate\_Terahashs+bc\_data$Difficulty+bc\_data$Total\_BTC+bc\_data$Days+bc\_data$Open+bc\_data$High+bc\_data$Low, bc\_data)***

***#ln\_FITHighlyCorelated <- lm(bc\_data$Price~bc\_data$Market\_Cap+poly(bc\_data$Network\_Hashrate\_Terahashs,2)+bc\_data$Network\_Hashrate\_#PetaFLOPS+bc\_data$Difficulty+bc\_data$Total\_BTC+bc\_data$Days+bc\_data$Days+bc\_data$Open++bc\_data$High+bc\_data$Low, bc\_data)***

***panderOptions("digits", 2)***

***pander(ln\_FITHighlyCorelated, caption = "Linear Model: Market Price vs. Highly Correlated Variables")***

***Rb2=summary(ln\_FITHighlyCorelated)$r.squared***

***cat("R-Squared = ", Rb2)***

***```***

***It appears that all of the highly correlated vairables to Market Price (Market Cap, Hash Rate, BTC Difficulty, Miners Revenue, and Estimated Transaction Volume USD) are significant.***

***##Residuals***

***```{R}***

***plot(ln\_FITHighlyCorelated, pch=16)***

***```***

***No dispersion and a flat line. Best model so far.***

***#Train Model and Test***

***##Creating the Training Subset and Test Subset***

***```{R}***

***set.seed(1)***

***bc\_traindata.index<-sample(1:nrow(bc\_data),0.70\*nrow(bc\_data), replace=FALSE)***

***bc\_traindata <- bc\_data[bc\_traindata.index, ]***

***bc\_testdata <- bc\_data[-bc\_traindata.index,]***

***```***

***##Training and Testing***

***```{R}***

***#Using the same model settings as above for lmfit7.***

***#lm\_FITTrain <- lm(Price~Market\_Cap+Network\_Hashrate\_Terahashs+Difficulty+Total\_BTC+Days, bc\_traindata)***

***lm\_FITTrain = lm(Price~Network\_Hashrate\_Terahashs+Difficulty+Total\_BTC+Days, bc\_traindata)***

***panderOptions("digits", 2)***

***pander(lm\_FITTrain, caption = "Linear Model: Market Price vs. Highly Correlated Variables")***

***summary(lm\_FITTrain)***

***Rb2=summary(lm\_FITTrain)$r.squared***

***cat("R-Squared = ", Rb2)***

***#PRESS(lm\_FITTrain)***

***bc\_testdata$predict1 <- predict(lm\_FITTrain,bc\_testdata)***

***bc\_data\_test$predict1 <- predict(lm\_FITTrain,bc\_data\_test)***

***ggplot(bc\_testdata, aes(bc\_testdata$Days)) +***

***geom\_point(aes(y=bc\_testdata$Price),color="Firebrick") +***

***geom\_line(aes(y=bc\_testdata$predict1), color="Blue")+***

***ggtitle('BTC Prediction vs. Actuals') +***

***theme(plot.title = element\_text(size=16, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Days", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=90, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***labs(title = paste("BTC Linear Regression Model Prediction vs. Actuals",***

***"\n\nAdj R2 = ",signif(summary(lm\_FITTrain)$adj.r.squared, 5),***

***" P =",signif(summary(lm\_FITTrain)$coef[2,4], 2)))***

***ggplot(bc\_data\_test, aes(bc\_data\_test$Days)) +***

***geom\_point(aes(y=bc\_data\_test$Price),color="Firebrick") +***

***geom\_line(aes(y=bc\_data\_test$predict1), color="Blue")+***

***ggtitle('BTC Prediction vs. Actuals') +***

***theme(plot.title = element\_text(size=16, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Days", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=90, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***labs(title = paste("BTC Linear Regression Model Prediction vs. Actuals",***

***"\n\nAdj R2 = ",signif(summary(lm\_FITTrain)$adj.r.squared, 5),***

***" P =",signif(summary(lm\_FITTrain)$coef[2,4], 2)))***

***```***

***Model does a very good job tracking the test set.***

***More work to be done....***

***Days,Total\_BTC, hightly related to Market\_Cap, Count(Total number of BitCoins),Difficulty, Next\_Difficulty,Network\_Hashrate\_Terahashs,Network\_Hashrate\_PetaFLOPS,Price,Open,High and Low***

***##Mutate Data in Percent Change***

***```{R, message=F, warning=F}***

***#library(zoo)***

***bc\_data$AverageTotalBitcoins<-bc\_data$Total\_BTC***

***bc\_data$AverageMarketCap<-bc\_data$Market\_Cap***

***bc\_data$AverageCount<-bc\_data$Count***

***bc\_data$AverageDifficulty<-bc\_data$Difficulty***

***bc\_data$AverageNetworkHashrateTera<-bc\_data$Network\_Hashrate\_Terahashs***

***bc\_data\_Ave<- subset(bc\_data,bc\_data$AverageTotalBitcoins>0)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, PercentChangePrice=(bc\_data\_Ave$Price-lag(bc\_data\_Ave$Price))/lag(bc\_data\_Ave$Price)\*100)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, AverageTotalBitcoins=(bc\_data\_Ave$AverageTotalBitcoins-lag(bc\_data\_Ave$AverageTotalBitcoins))/lag(bc\_data\_Ave$AverageTotalBitcoins)\*100)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, AverageMarketCap=(bc\_data\_Ave$AverageMarketCap-lag(bc\_data\_Ave$AverageMarketCap))/lag(bc\_data\_Ave$AverageMarketCap)\*100)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, AverageCount=(bc\_data\_Ave$AverageCount-lag(bc\_data\_Ave$AverageCount))/lag(bc\_data\_Ave$AverageCount)\*100)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, AverageDifficulty=(bc\_data\_Ave$AverageDifficulty-lag(bc\_data\_Ave$AverageDifficulty))/lag(bc\_data\_Ave$AverageDifficulty)\*100)***

***bc\_data\_Ave<-mutate(bc\_data\_Ave, AverageNetworkHashrateTera=(bc\_data\_Ave$AverageNetworkHashrateTera-lag(bc\_data\_Ave$AverageNetworkHashrateTera))/lag(bc\_data\_Ave$AverageNetworkHashrateTera)\*100)***

***bc\_data\_Ave <- subset(bc\_data\_Ave, !is.na(bc\_data\_Ave$PercentChangePrice))***

***ggplot(bc\_data\_Ave, aes(bc\_data\_Ave$Date, bc\_data\_Ave$PercentChangePrice)) +***

***geom\_point(color="firebrick") +***

***ggtitle('BTC Percent Change vs. Time') +***

***theme(plot.title = element\_text(size=20, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Date", y="Pct. Change")+***

***theme(axis.text.x=element\_text(angle=50, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))***

***```***

***##Correlation Analysis of Transformed Variables***

***```{R}***

***cor2 <- cor(bc\_data\_Ave[c(18:23)]) #lecthead()ing variables to include in correlation analysis***

***cor2 <- cor(BTCm11[c(26:30)]) #selecting variables to include in correlation analysis***

***colnames(cor2) <- c("% Change Transx Volume", "% Change Miner Rev.","% Change Market Cap", "% Change Market Price","Change Total Coins")***

***rownames(cor2) <- c("% Change Transx Volume", "% Change Miner Rev.","% Change Market Cap","% Change Market Price", "Change Total Coins")***

***corrplot(cor2, method = "square", tl.srt = 50, tl.col = "black", tl.cex = 0.6, title = "Correlation of Variables", mar=c(0,0,1,0))***

***```***

***#LASSO***

***```{R}***

***#install.packages("glmnet")***

***#library(glmnet)***

***x <- model.matrix(Price~., bc\_traindata)[,-12]***

***y <- bc\_traindata[,c(12)]***

***xtest <- model.matrix(Price~., bc\_testdata)[,-12]***

***ytest <- bc\_testdata[,c(12)]***

***#plot(fit.lasso, xvar="lambda", label=TRUE)***

***#lambda <- 10^seq(10, -2, length = 100)***

***lassoreg1 <- cv.glmnet(x, y, family = "gaussian",type.measure = "default", alpha = 1,nlambda=100)***

***plot(lassoreg1)***

***bestlam <- lassoreg1$lambda.min***

***lasso\_reg <- glmnet(x, y, alpha = 1, lambda = lassoreg1$lambda.1se)***

***lasso\_reg$beta[,1]***

***lasso.pred <- predict(lasso\_reg, s = bestlam, newx = xtest[,-(17:18)])***

***m<-mean((lasso.pred-ytest)^2)***

***m***

***lasso.coef <- predict(lasso\_reg, type = 'coefficients', s = bestlam)***

***lasso.coef***

***ggplot(bc\_data\_test, aes(bc\_data\_test$Days)) +***

***geom\_point(aes(y=bc\_data\_test$Price),color="Firebrick") +***

***geom\_line(aes(y=bc\_data\_test$predict1), color="Blue")+***

***ggtitle('BTC Prediction vs. Actuals') +***

***theme(plot.title = element\_text(size=16, face="bold",***

***margin = margin(10, 0, 10, 0)))+***

***labs(x="Days", y="Market Price (USD)")+***

***theme(axis.text.x=element\_text(angle=90, vjust=0.5)) +***

***theme(panel.background = element\_rect(fill = 'grey75'))+***

***labs(title = paste("BTC Linear Regression Model Prediction vs. Actuals",***

***"\n\nAdj R2 = ",signif(summary(lm\_FITTrain)$adj.r.squared, 5),***

***" P =",signif(summary(lm\_FITTrain)$coef[2,4], 2)))***

***panderOptions("digits", 2)***

***pander(lasso\_reg, caption = "Linear Model: Market Price vs. Highly Correlated Variables")***

***summary(lasso\_reg)***

***Rb21=summary(lasso\_reg)$r.squared***

***cat("R-Squared = ", Rb21)***

***```***

**Objective 2 Code:**

**```{r}**

**library(tseries)**

**library(forecast)**

**library(car)**

**library(TSstudio)**

**library(tidyverse)**

**library(knitr)**

**```**

**```{r}**

**#setwd("/Users/matt/Desktop/SMU/Applied Stats/ProjectDetails/AppliedStats")**

**BitcoinDataMergeV2 <- read.csv("BitcoinDataMergeV2.csv", header = TRUE)**

**attach(BitcoinDataMergeV2)**

**```**

**Add a date column per Project requirements**

**```{r}**

**colnames(BitcoinDataMergeV2)[1] <- "Time"**

**```**

**Plot Bitcoin Price data over time**

**```{r, plot}**

**p <- ggplot(BitcoinDataMergeV2, aes(x=Time, y=Price)) +**

**geom\_line() +**

**xlab("Time") + ylab("Bitcoin Price") + ggtitle("Bitcoin Price Over Time")**

**p**

**```**

**```{r}**

**attach(BitcoinDataMergeV2)**

**acf(Price)**

**```**

**```{r}**

**pacf(Price)**

**```**

**Run durbin watson determin if serial corrolation exists**

**```{r}**

**durbinWatsonTest(lm(Price~1), max.lag = 10)**

**```**

**Convert to time series**

**```{r}**

**bitcoinTS\_Price <- ts(BitcoinDataMergeV2$Price, freq=365)**

**```**

**Decompose Time Series**

**```{r}**

**bitcoin\_tsDecomp <- decompose(bitcoinTS\_Price, type = 'multiplicative')**

**plot(bitcoin\_tsDecomp)**

**```**

**See what data has correlation**

**```{r}**

**library(corrplot)**

**library(purrr) #keep**

**#function to create corrolation heatmap**

**correlator <- function(df){**

**df %>%**

**keep(is.numeric) %>%**

**tidyr::drop\_na() %>%**

**cor %>%**

**corrplot(addCoef.col = "white", number.digits = 2,**

**number.cex = .5, method = "pie",**

**order = "hclust",**

**tl.srt = 45, tl.cex = .8)**

**}**

**correlator(BitcoinDataMergeV2)**

**```**

**Log Transform price data to compare**

**```{r}**

**BitcoinDataMergeV2$LogPrice <- log(Price)**

**```**

**Plot Log transformed Bitcoin Price data over time**

**```{r, plot}**

**p <- ggplot(BitcoinDataMergeV2, aes(x=Time, y=LogPrice)) +**

**geom\_line() +**

**xlab("Time") + ylab("Bitcoin Price") + ggtitle("Log transformed Bitcoin Price Over Time")**

**p**

**```**

**```{r}**

**acf(LogPrice)**

**```**

**```{r}**

**pacf(LogPrice)**

**```**

**```{r}**

**bitcoinTS\_LogPrice <- ts(BitcoinDataMergeV2$LogPrice, freq=1)**

**```**

**```{r}**

**bitcoinTS\_Price <- ts(BitcoinDataMergeV2$Price, freq=30)**

**```**

**Decompose Time Series**

**```{r}**

**bitcoin\_tsDecomp <- decompose(bitcoinTS\_Price, type = 'multiplicative')**

**plot(bitcoin\_tsDecomp)**

**```**

**Analysis of "Total BTC" as a Price Predictor**

**```{r}**

**plot(BitcoinDataMergeV2$Total.BTC,BitcoinDataMergeV2$Price,xlab="Total BTC")**

**Pred\_TotalBTC<-lm(BitcoinDataMergeV2$Price~ poly(BitcoinDataMergeV2$Total.BTC,degree = 3,raw = TRUE))**

**#Pred\_TotalBTC2<-lm(LogPrice~ poly(`Total BTC`,degree = 2,raw = TRUE))**

**abline(lm(BitcoinDataMergeV2$Price~BitcoinDataMergeV2$Total.BTC))**

**lines(smooth.spline(BitcoinDataMergeV2$Price,predict(Pred\_TotalBTC)), col = 'blue', lwd = 3)**

**text(16000000,5,paste("Cor=",round(cor(BitcoinDataMergeV2$Price,BitcoinDataMergeV2$Total.BTC),2)))**

**```**

**Polynomial model ^3**

**```{r}**

**summary(Pred\_TotalBTC)**

**```**

**Polynomial model ^2**

**```{r}**

**summary(Pred\_TotalBTC2)**

**```**

**Analysis of "Difficulty" as a Price Predictor**

**```{r}**

**plot(BitcoinDataMergeV2$Difficulty,BitcoinDataMergeV2$Price,xlab="Total BTC")**

**Pred\_Difficulty<-lm(BitcoinDataMergeV2$Price~ BitcoinDataMergeV2$Difficulty)**

**abline(lm(BitcoinDataMergeV2$Price~BitcoinDataMergeV2$Difficulty))**

**lines(smooth.spline(BitcoinDataMergeV2$Difficulty,predict(Pred\_Difficulty)), col = 'blue', lwd = 3)**

**text(800000000000,5,paste("Cor=",round(cor(BitcoinDataMergeV2$Price, BitcoinDataMergeV2$Difficulty),2)))**

**```**

**summary of Difficulty simple linear model**

**```{r}**

**DifficultyLM <- lm(LogPrice~Difficulty)**

**summary(DifficultyLM)**

**```**

**Summary of Difficulty polynomial fit ^2**

**```{r}**

**summary(Pred\_Difficulty)**

**```**

**Compare Difficulty simple model to polynomial model: P <.05 there is a significant difference between these models we'll include the ploynomial set in the predictors**

**```{r}**

**anova(DifficultyLM,Pred\_Difficulty)**

**```**

**Analysis of "Count" as a Price Predictor**

**Conculsion: Count and Total BTC are not statistically different but count gets a slightly better r2 so we will use count as a predictor in our model instead of Total BTC**

**```{r}**

**plot(BitcoinDataMergeV2$Count,BitcoinDataMergeV2$Price,xlab="Total BTC")**

**Pred\_Count2<-lm(BitcoinDataMergeV2$Price~ poly(BitcoinDataMergeV2$Count,degree = 2,raw = TRUE))**

**Pred\_Count3<-lm(BitcoinDataMergeV2$Price~ poly(Count,degree = 3,raw = TRUE))**

**abline(lm(BitcoinDataMergeV2$Price~BitcoinDataMergeV2$Count))**

**lines(smooth.spline(BitcoinDataMergeV2$Count,predict(Pred\_Count3)), col = 'blue', lwd = 3)**

**text(450000,5,paste("Cor=",round(cor(BitcoinDataMergeV2$Price,Count),2)))**

**```**

**```{r}**

**summary(Pred\_Count3)**

**```**

**Analysis of "MarketCap" as a Price Predictor**

**```{r}**

**attach(BitcoinDataMergeV2)**

**plot(BitcoinDataMergeV2$Market.Cap,Price,xlab="Total BTC")**

**Pred\_MC1<-lm(BitcoinDataMergeV2$Price~Market.Cap)**

**Pred\_MC2<-lm(BitcoinDataMergeV2$Price~ poly(Market.Cap,degree = 2,raw = TRUE))**

**Pred\_MC3<-lm(Price~ poly(Market.Cap,degree = 3,raw = TRUE))**

**abline(lm(Price~Market.Cap))**

**lines(smooth.spline(Market.Cap,predict(Pred\_MC1)), col = 'blue', lwd = 3)**

**text(800000000000,5,paste("Cor=",round(cor(Price,Market.Cap),2)))**

**```**

**```{r}**

**attach(BitcoinDataMergeV2)**

**plot(BitcoinDataMergeV2$Market.Cap,Price,xlab="Market Cap")**

**Pred\_MC1<-lm(BitcoinDataMergeV2$Price~Market.Cap)**

**Pred\_MC2<-lm(BitcoinDataMergeV2$Price~ poly(Market.Cap,degree = 2,raw = TRUE))**

**Pred\_MC3<-lm(Price~ poly(Market.Cap,degree = 3,raw = TRUE))**

**abline(lm(Price~Market.Cap))**

**lines(smooth.spline(Market.Cap,predict(Pred\_MC3)), col = 'blue', lwd = 3)**

**text(200000000000,3000,paste("Cor=",round(cor(Price,Market.Cap),2)))**

**```**

**```{r}**

**summary(Pred\_MC3)**

**```**

**```{r}**

**attach(BitcoinDataMergeV2)**

**plot(log(BitcoinDataMergeV2$Market.Cap),LogPrice,xlab="Total BTC")**

**Pred\_MC1<-lm(BitcoinDataMergeV2$LogPrice~Market.Cap)**

**Pred\_MC2<-lm(BitcoinDataMergeV2$LogPrice~ poly(Market.Cap,degree = 2,raw = TRUE))**

**Pred\_MC3<-lm(LogPrice~ poly(Market.Cap,degree = 3,raw = TRUE))**

**abline(lm(LogPrice~Market.Cap))**

**lines(smooth.spline(Market.Cap,predict(Pred\_MC3)), col = 'blue', lwd = 3)**

**text(800000000000,5,paste("Cor=",round(cor(LogPrice,Market.Cap),2)))**

**```**

**Create a matrix of predictors to use**

**```{r}**

**train\_preds <- as.matrix(cbind(Count[1:1467],Count[1:1467]^2,Count[1:1467]^3,Difficulty,Difficulty[1:1467]^2))**

**colnames(Preds)[2] <- c("Count^2")**

**colnames(Preds)[3] <- c("Count^3")**

**colnames(Preds)[5] <- c("Difficulty^2")**

**```**

**Split Time series**

**```{r}**

**Split\_Data <- ts\_split(ts.obj = bitcoinTS\_LogPrice, sample.out = 730)**

**train <- Split\_Data$train**

**test <- Split\_Data$test**

**length(train)**

**length(test)**

**```**

**```{r, like hw4}**

**test<-window(ts(BitcoinDataMergeV2$Price),start=2188)**

**train<-window(ts(BitcoinDataMergeV2$Price),start=1832, end = 2187)**

**predictor1\_train<-window(ts(BitcoinDataMergeV2$Count), end = 1467)**

**predictor1\_test <- window(ts(BitcoinDataMergeV2$Count), start = 1468)**

**predictor2\_train <- window(ts(BitcoinDataMergeV2$Difficulty), end = 1467)**

**predictor2\_test <- window(ts(BitcoinDataMergeV2$Difficulty), start = 1468)**

**predictor3\_train <- window(ts(BitcoinDataMergeV2$Market.Cap[1:1467]))**

**predictor3\_test <- window(ts(BitcoinDataMergeV2$Market.Cap), start = 1468)**

**predictor3quad\_train <-**

**#Preds <- as.matrix(cbind(predictor1,predictor1^2,as.numeric(as.character(predictor1^3)),predictor2,predictor2^2))**

**#colnames(Preds)<-c("Count","Count2","Count3","Difficulty","Difficulty2")**

**#Preds <- read.csv("Preds.csv")**

**#Preds <- as.matrix(Preds)**

**```**

**Split Predictors**

**```{r}**

**train\_preds <- slice(Preds2,1:1467)**

**test\_preds <- slice(Preds2,1468:2197)**

**#train\_prds <- as.matrix(train\_preds)**

**length(train\_preds)**

**length(test\_preds)**

**```**

**#Model 1 - no predictors**

**```{r}**

**BTC\_Model1<-auto.arima(train,stepwise=FALSE)**

**BTC\_Model1**

**tsdisplay(residuals(BTC\_Model1),lag.max=15,main="BTC\_Model1")**

**plot(forecast(BTC\_Model1,h=10))**

**#points(1:length(train),fitted(BTC\_Model1),type="l",col="blue")**

**points(1:length(train),fitted(BTC\_Model1),type="l",col="blue")**

**points(1:2197,BitcoinDataMergeV2$Price,type = 'l')**

**```**

**```{r}**

**cast.model1 <- forecast(BTC\_Model1,h = 10)**

**kable(accuracy(cast.model1))**

**```**