Exploring the World of Cryptocurrency

Predicting Bitcoin Prices with Multiple Linear Regression

Joseph Bajin (NetID : jbaji2)

Jacob Rettig (NetID : jrettig2)

Yash Lakhani (NetID : yashl3)

Brad Ballard (NetID : bjb3)

08/04/2018

- Introduction
  - Why this Topic?
- Methods
  - Loading all necessary Libraries.
  - Data Acquisition
    - Acquire Data from Multiple Sources - (Quandl and Yahoo Finance)
    - Data Extraction from Multiple Data Sources
    - Clean-up Column Names for Readability
    - Merge Data Sources together into a Single Dataset
  - Investigating the Predictors
    - Overview of the Dataset
    - Closing Price of Gold (GLD)
    - Closing Price of S&P500 Index (Market Proxy) (SPY)
    - Closing CBOE Volatility Index (Volatility Implied by S&P 500 Index Options) (VIX)
    - Yield of US 10 Year Treasury Bond
    - Exchange Rate of Bitcoin (BTC) to USD
    - Current Bitcoin Hash Rate
    - Currency Pairs
- Results
  - Building the Model
    - Splitting up the data - Training/Testing
    - Additive Model and Testing Assumptions
    - Removing influential data points
    - Alternative models
  - Test Validation
    - Testing Our Models - RMSE
    - Visualizing the Train and Test Data Sets
- Discussion
  - Outcomes and Observations

Introduction

Why this Topic?

The formation of our group hinged upon our interest in economics and finance, as well as our collective academic and/or professional experience in these fields.

Our aim for this project is to be able to predict within a reasonable amount of confidence the price of Bitcoin. Many see Bitcoin as an asset class that is bought when the market is bearish. This is analogous to how Gold and other commodities are utilized, as a ‘safe haven’ asset which are used to hedge systemmatic risk in the marketplace. Data Points are collected to measure different aspects of the economy that express current market conditions: such as foreign exchange rates, market closes, commodity prices, and Bitcoin prices. Using these data points, a model to predict the price of Bitcoin will be trained and then we aim to run our model against a test set to see how it performs.

We wanted to stay away from financial models that have been heavily documented in order to keep this as an exploratory data analysis task. This allows for greater excitement if we are able to discover a ‘useful’ model as Bitcoin has become one of the ‘hottest’ investment vehicles today and continues to garner media-wide attention.

Equipped with the knowledge gained over the semester, we feel well prepared in order to understand how a predictive model is used by financial analysts and is a major driving force for our interest, given a relatively novel and volatile market of cryptocurrency. In addition to this, we hope that each member of the group can use our findings outside this class as well. Whether it is for professional or personal use, we can utilize our findings from this project regardless of whether it is revolutionary or simply an interesting conversation over a summer cocktail.

Our inspiration comes from an Economics paper, citing from the Universitat de Barcelona : Exploring the determinants of Bitcoin’s price: an application of Bayesian Structural Time Series (https://arxiv.org/pdf/1706.01437.pdf). This paper broke down a lot of the pure concepts that allowed us to bring together the concepts we learned as well as provide background information to understand how these models are determined.

Methods

Loading all necessary Libraries.

```
library(Quandl)
library(readr)
library(lmtest)
library(leaps)
library(tibbletime)
library(dplyr)
library(tidyr)
library(caret)
library(knitr)
library(ggplot2)
library(GGally)
library(gridExtra)
library(ggthemes)
library(MASS)
library(faraway)
Quandl.api_key("esqkikVR6svTWMGWBWgv")
```

The major libraries needed will be loaded from here to be able to be used throughout the document. This will include the libraries to visualize the data `ggplot2` , read input from a csv files `readr` , and obtain finanical data `Quandl` .

In order to use Quandl (https://www.quandl.com/sign-up-modal?defaultModal=showSignUp), a free API account must be setup to obtain an API Key. This key is per user and is needed to obtain the free datasources found through-out the rest of the document.

Data Acquisition

Acquire Data from Multiple Sources - (Quandl and Yahoo Finance)

```
bonds = Quandl("USTREASURY/YIELD",start_date="2016-01-01", end_date="2018-07-01")
bitcoin=Quandl("BITFINEX/BTCUSD", start_date="2016-01-01", end_date="2018-07-01")
vix = Quandl("CBOE/VIX", start_date="2016-01-01", end_date="2018-07-01")
bitcoin_hash = Quandl("BCHAIN/HRATE", start_date="2016-01-01", end_date="2018-07-01")
spy = read_csv("spy.csv")
gld = read_csv("gld.csv")
## Currency Pairs
currency_pairs = c("BOE/XUDLSFD","BOE/XUDLJYD","FRED/DEXUSUK","FRED/DEXUSEU")
currency = Quandl(currency_pairs, start_date="2016-01-01", end_date="2018-07-01")
```

The data for this model was obtained from multiple sources. The initial goal was to use Quandl to obtain all the data needed, but after further research it was determined that they did not provide this type of data under their free api usage. This required that that data be obtained from outside sources. The datasets for `GLD` and `SPY` were obtained from Yahoo! Financial using a python library (fix_yahoo_finance (https://pypi.org/project/fix-yahoo-finance/)) to extract both GLD and SPY daily close prices from its API and write it to a file. There was no such easily obtainable option within R.

The remainder of the data was obtained from multiple different sources within the Quandl module. This included data provided by the `Bank of England`, `United States Department of Treasury`, `Chicago Board Options Exchange`, and `Bitcoin` exchanges. More information about Quandl datasources can be found at the Quandl Website (https://www.quandl.com/search?filters=%5B%22Free%22%5D).

All time periods for this data were aligned for *January 1st, 2016* through *July 1st, 2018*.

Data Extraction from Multiple Data Sources

```
bonds = subset(bonds, select = c("Date","10 YR"))
bitcoin = subset(bitcoin, select = c("Date","Last"))
vix = subset(vix, select = c("Date","VIX Close"))
spy = subset(spy, select = c("Date","Close"))
gld = subset(gld, select = c("Date","Close"))
```

The data from the multiple sources included additional information that was not needed within the model. As it was not needed only the necessary columns were extracted. This reduced the size of the dataset dramatically and allowed the speed of the model to be efficient.

It was determined that only the `Date` and the particular value of the respective asset was necessary. The `Date` is used as a key to link all the different datasets together by a common value.

Clean-up Column Names for Readability

```
colnames(bonds)[2] = "10yrbond"
colnames(bitcoin)[2] = "bitcoin_close"
colnames(vix)[2] = "vix_close"
colnames(spy)[2] = "spy_close"
colnames(gld)[2] = "gld_close"
colnames(currency)[2] = "usd_chf"
colnames(currency)[3] = "usd_jpy"
colnames(currency)[4] = "usd_gbp"
colnames(currency)[5] = "usd_eur"
colnames(bitcoin_hash)[2] = "bitcoin_hash"
```

As the data came from multiple sources, there was not a unified naming convention for the data. Renaming the columns provided an easier way to not only look at the data but to reference it within the model. Moreoever, we were able to derive a common understanding of what each column represented.

Merge Data Sources together into a Single Dataset

```
## Basic Merge
merge1 = merge(bonds, bitcoin, by="Date")
merge2 = merge(vix, spy, by="Date")
merge3 = merge(gld, currency, by="Date")

## Merge of Merges
merge4 = merge(merge3, bitcoin_hash, by="Date")
merge5 = merge(merge2, merge1, by="Date")

## Final Merge
data = merge(merge4, merge5, by="Date")
```

This brought all the different datasets into a single source. It was combined by using the `Date` column as the key from each data source. The final result has 613 records. This represents data for *January 1st, 2016* through *July 1st, 2018*.

Investigating the Predictors

Overview of the Dataset

The Predictors represented measurements of different aspects of the economy that express current market conditions. These predictors are indicative of how many active investors view Bitcoin as an asset class that can be used to counter bearish market conditions. Due to the fact that investing is a global activity, there are obviously countless other variables that coult be considered; however, we beleive that this representation provides a reputable pulse on the global economy from an investment perspective.

In the below table, the output of the `Column Clean-up` and `Merge` can be seen as well as the respective description of each predictor.

Variables	Type	Description
bitcoin_usd	numerical(response)	Exchange Rate of Bitcoin (BTC) to USD
10yrbond	numerical	Yield of US 10 Year Treasury Bond

Variables	Type	Description
vix_close	numerical	Closing CBOE Volatility Index (Volatility Implied by S&P 500 Index Options)
spy_close	numerical	Closing Price of S&P500 Index (Market Proxy)
gld_close	numerical	Closing Price of Gold
usd_chf	numerical	USD and Swiss Franc (CHF) Exchange Rate
usd_jpy	numerical	USD and Japanese Yen (JPY) Exchange Rate
usd_gbp	numerical	USD and Pound (GBP) Exchange Rate
usd_eur	numerical	USD and Euro (EUR) Exchange Rate
bitcoin_hash	numerical	Current Bitcoin Hash Rate

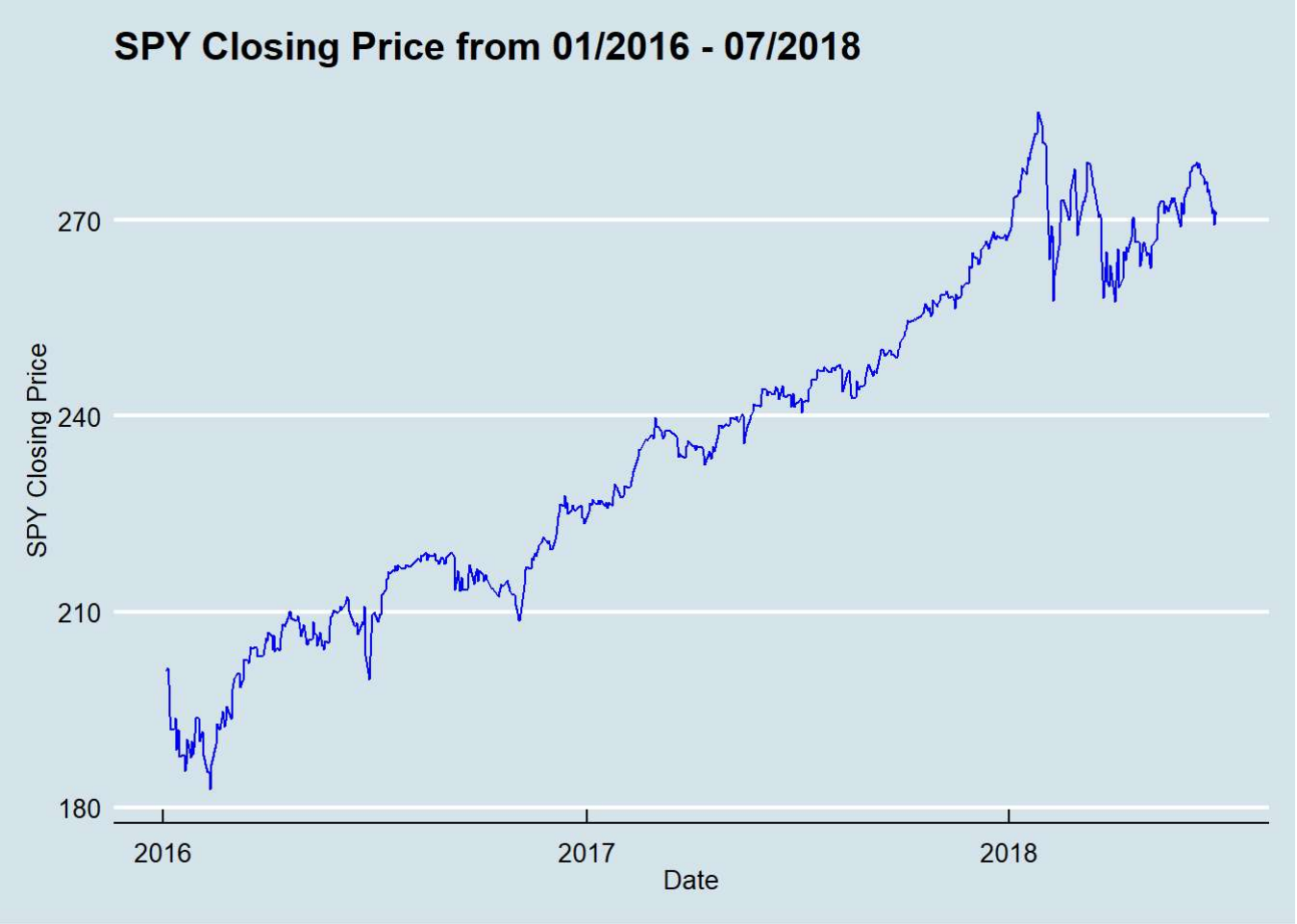
Closing Price of Gold (GLD)

```
gld_plot = data.frame(yvalue = data$gld_close, xvalue=data$Date)
ggplot(data = gld_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "GLD Closing Price from 01/2016 - 07/2018", x = "Date", y = "GLD Closing Price")
 + theme_economist()
```

Gold is often considered the primary investment class to hedge against systematic market risks. There also appears to be a possible non linear trend, which we can assess in transformations.

Closing Price of S&P500 Index (Market Proxy) (SPY)

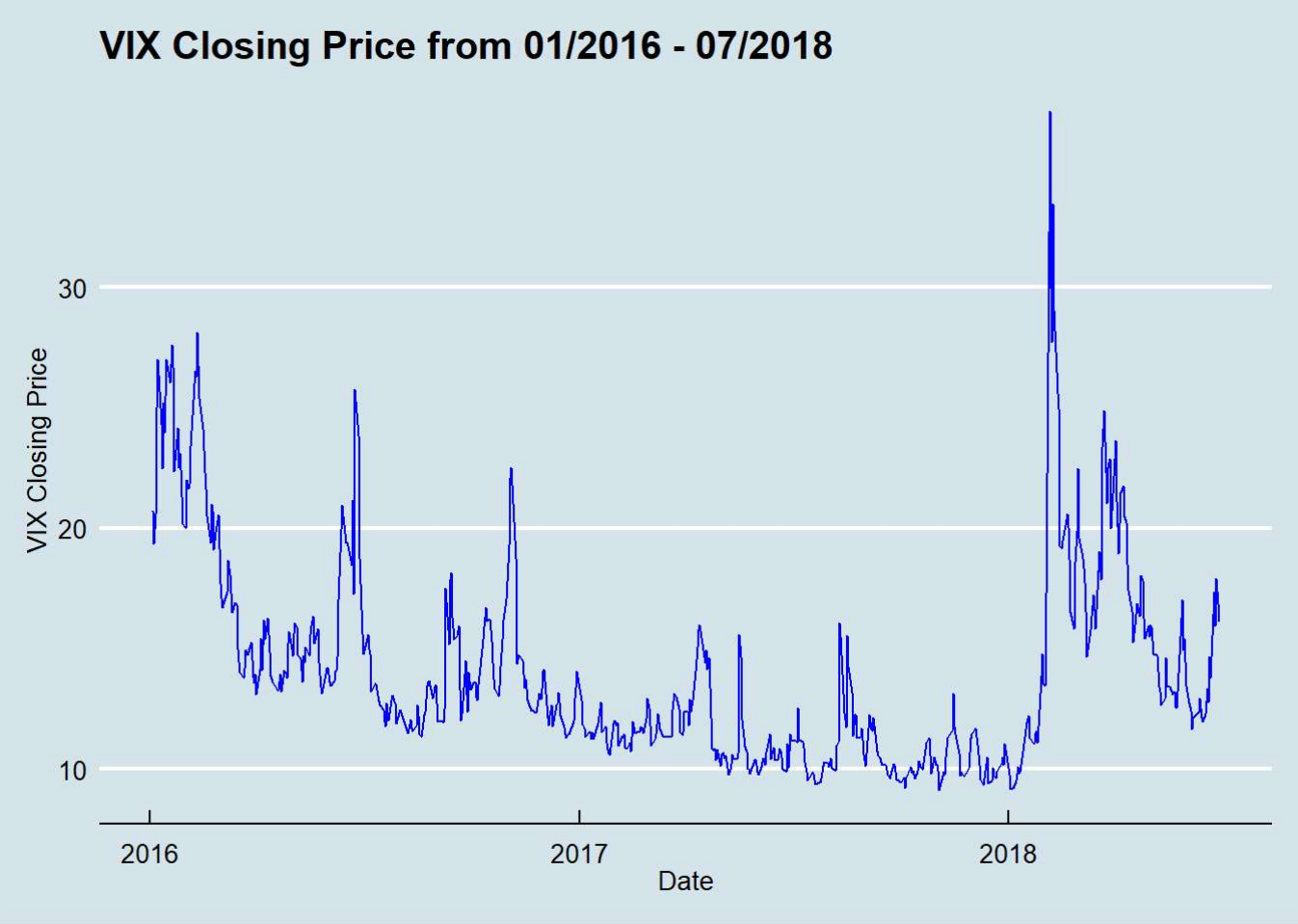
```
spy_plot = data.frame(yvalue = data$spy_close, xvalue=data$Date)
ggplot(data = spy_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "SPY Closing Price from 01/2016 - 07/2018", x = "Date", y = "SPY Closing Price")
 + theme_economist()
```



The `SPY` is an ETF (Exchange Traded Fund) that represents the S&P500 which is a measurement of the 500 largest American companies and one of the best indices to track US Market growth.

Closing CBOE Volatility Index (Volatility Implied by S&P 500 Index Options) (VIX)

```
vix_plot = data.frame(yvalue = data$vix_close, xvalue=data$Date)
ggplot(data = vix_plot, aes(x = xvalue, y = yvalue)) +
    geom_line(color='blue') +
  labs(title = "VIX Closing Price from 01/2016 - 07/2018", x = "Date", y = "VIX Closing Price")
 + theme_economist()
```

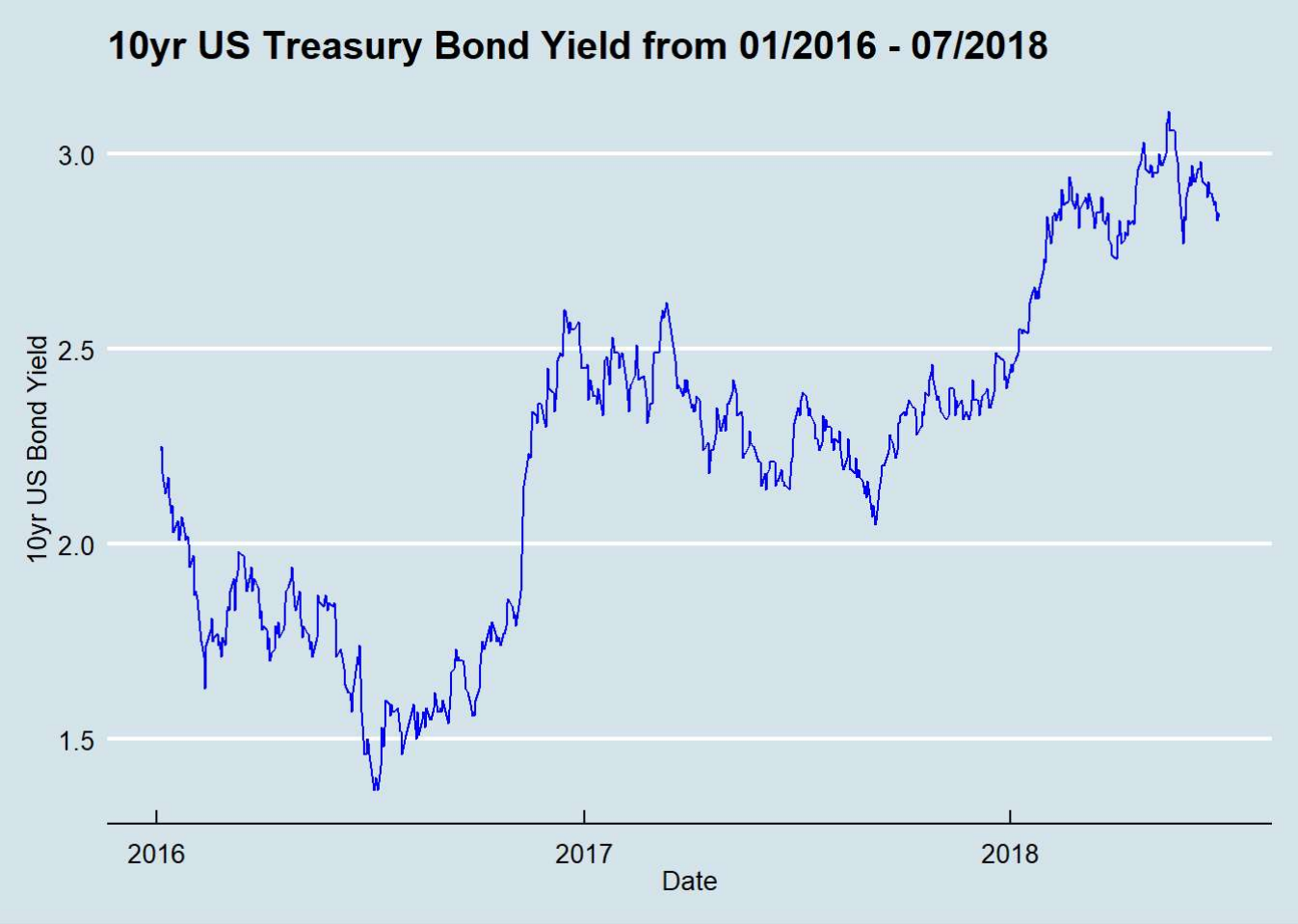


The VIX ETF (the Volatility Index) is a measurement of volatility in the stock market. Meaning, as trading activity, either bearish or bullish increases, the volatility of the market increases.

This possibly has potential to have a polynomial transformation added.

Yield of US 10 Year Treasury Bond

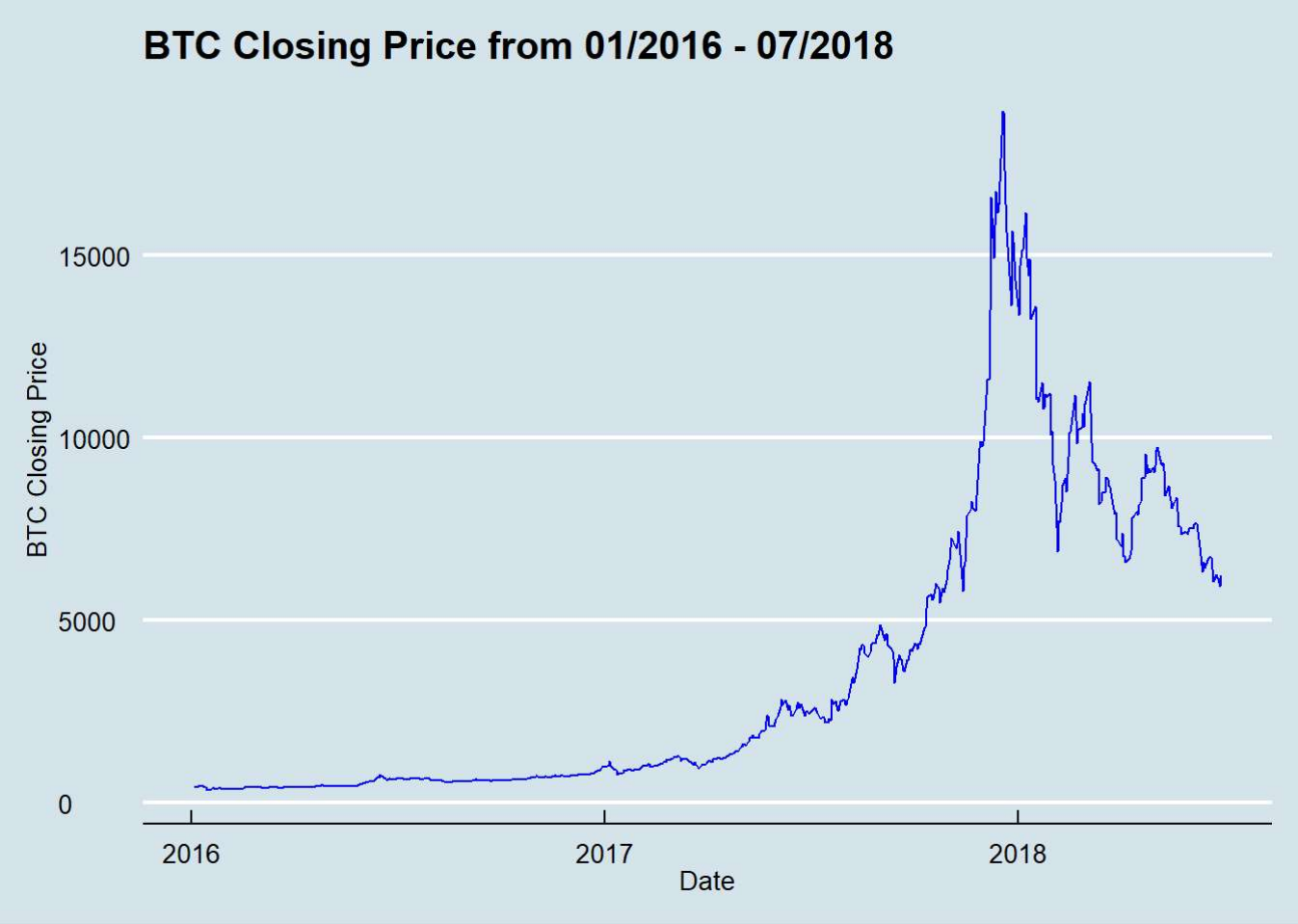
```
bond_plot = data.frame(yvalue = data$`10yrbond`, xvalue=data$Date)
ggplot(data = bond_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "10yr US Treasury Bond Yield from 01/2016 - 07/2018", x = "Date", y = "10yr US Bo
nd Yield")  + theme_economist()
```



Along with gold, US Treasury notes are considered the ‘gold-standard’ in systemmatic market hedging. The 10yr US Treasury bond is supported by the United States Government which gives it a level of security that may not exist in other asset classes.

Exchange Rate of Bitcoin (BTC) to USD

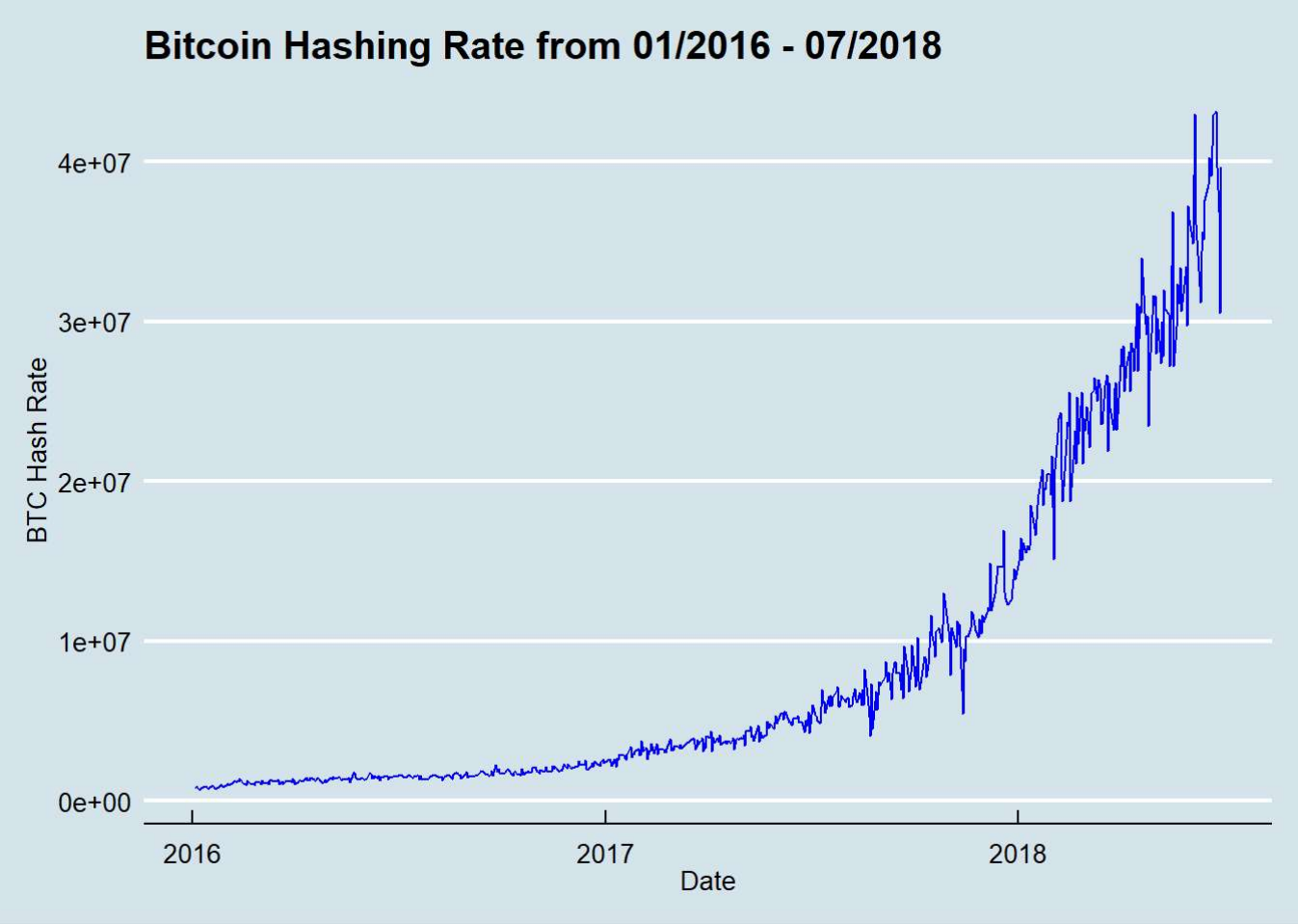
```
bitcoin_plot = data.frame(yvalue = data$bitcoin_close, xvalue=data$Date)
ggplot(data = bitcoin_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "BTC Closing Price from 01/2016 - 07/2018", x = "Date", y = "BTC Closing Price")
 + theme_economist()
```



The response of this model. The above is the graphical representation of the USD / Bitcoin exchange rate.

Current Bitcoin Hash Rate

```
btchash_plot = data.frame(yvalue = data$bitcoin_hash, xvalue=data$Date)
ggplot(data = btchash_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "Bitcoin Hashing Rate from 01/2016 - 07/2018", x = "Date", y = "BTC Hash Rate")
 + theme_economist()
```

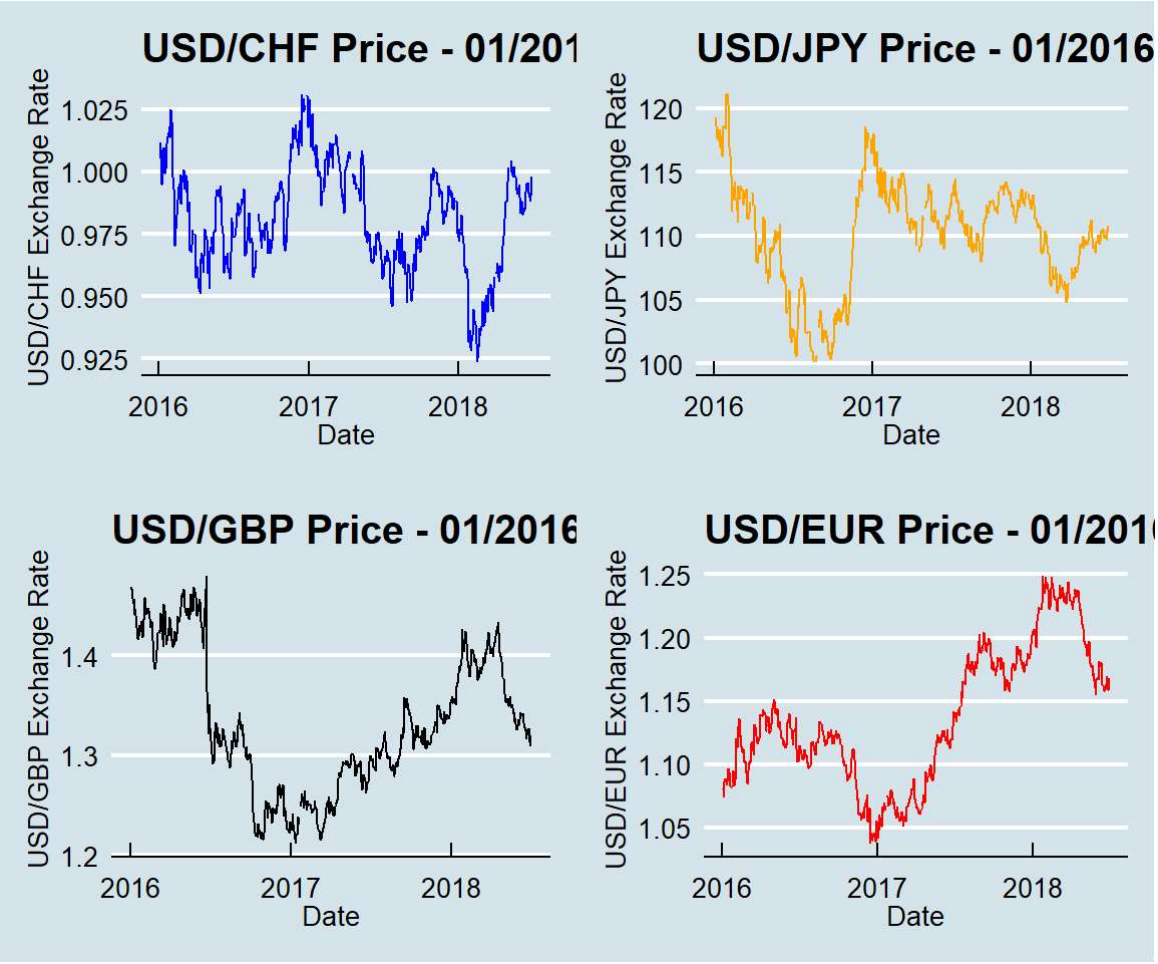


This is the amount of cycles necessary for computers to mine (create) new Bitcoins in the system. The higher the hash rate, the harder the systemic algorithm becomes thus resulting in increased difficulty in 'mining' additional Bitcoins.

We can see a log transformation would assist in linearizing this exponential trend, we will address this in our transformations section.

Currency Pairs

```
chf_plot = data.frame(yvalue = data$usd_chf, xvalue=data$Date)
jpy_plot = data.frame(yvalue = data$usd_jpy, xvalue=data$Date)
gbp_plot = data.frame(yvalue = data$usd_gbp, xvalue=data$Date)
eur_plot = data.frame(yvalue = data$usd_eur, xvalue=data$Date)
p1 = ggplot(data = chf_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='blue') +
  labs(title = "USD/CHF Price - 01/2016 - 07/2018", x = "Date", y = "USD/CHF Exchange Rate") + t
heme_economist()
p2 = ggplot(data = jpy_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='orange') +
  labs(title = "USD/JPY Price - 01/2016 - 07/2018", x = "Date", y = "USD/JPY Exchange Rate") + t
heme_economist()
p3 = ggplot(data = gbp_plot, aes(x = xvalue, y = yvalue)) +
  geom_line(color='black') +
  labs(title = "USD/GBP Price - 01/2016 - 07/2018", x = "Date", y = "USD/GBP Exchange Rate") + t
heme_economist()
p4 = ggplot(data = eur_plot, aes(x = xvalue, y = yvalue)) +
   geom_line(color='red') +
  labs(title = "USD/EUR Price - 01/2016 - 07/2018", x = "Date", y = "USD/EUR Exchange Rate") + t
heme_economist()
grid.arrange(p1, p2, p3, p4, nrow = 2)
```



Exchange rates from the 4 major currency pairs compared against the USD. Those currency pairs are Swiss Franc (CHF), Japanese Yen (JPY), British Pound (GBP), and the Euro (EUR). The selection of currency pairs could be more expansive to improve our model, but this acts as a proxy for economic growth in other countries. This is because, devaluation is usually caused by poor economic conditions in that country. Recently, we see that post

Brexit, a political change radically changed the future trade relations with the EU and the UK. Therefore, we see a large drop in exchange rate between USD and GBP. This suggests there is speculation on the currency markets, based on economic and political changes.

Generally, most economic variables are compared in USD, as traditionally Oil prices, Commodities have a pegging currency which is USD. Therefore, our analysis can be done by comparing USD to other forms of currency to contrast with the exchange rate to Bitcoin.

Results

Building the Model

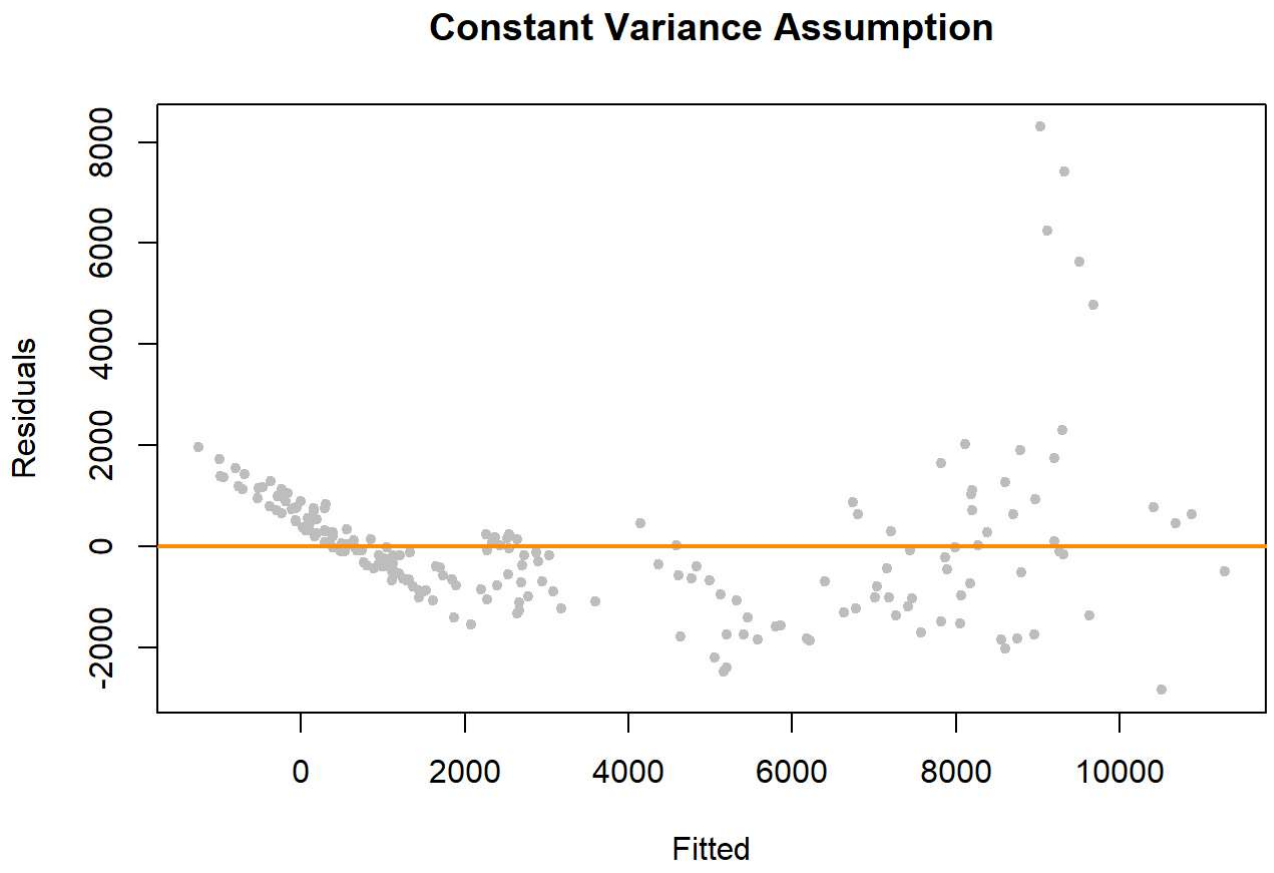
Splitting up the data - Training/Testing

```
data = na.omit(data)
set.seed(20160924)
trn_idx = sample(nrow(data), 200)
data_trn = data[trn_idx, ]
data_tst = data[-trn_idx, ]
```

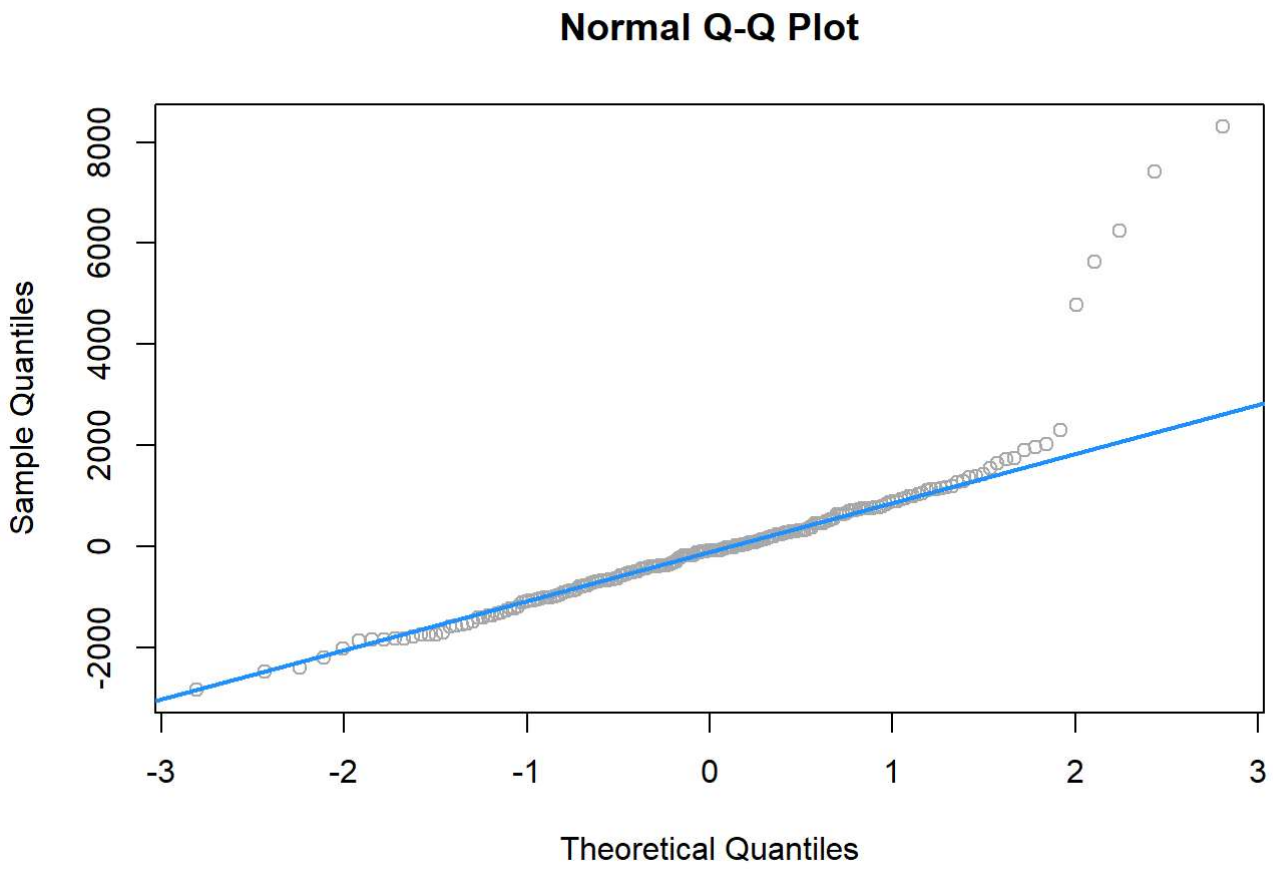
This breaks up the data into a training and testing set. The training set will include 200 rows of data, while the testing dataset will include 401.

Additive Model and Testing Assumptions

```
calc_loocv_rmse = function(model) {
  sqrt(mean((resid(model) / (1 - hatvalues(model))) ^ 2))
}

btc_lm = lm(bitcoin_close ~ . -Date ,data=data_trn)
cd_btc_lm_add = cooks.distance(btc_lm)
large_cd_btc = cd_btc_lm_add > 4 / length(cd_btc_lm_add)
plot(fitted(btc_lm), resid(btc_lm), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Constant Variance Assumption")
abline(h = 0, col = "darkorange", lwd = 2)
```




```
qqnorm(resid(btc_lm), col = "darkgrey")
qqline(resid(btc_lm), col = "dodgerblue", lwd = 2)
```



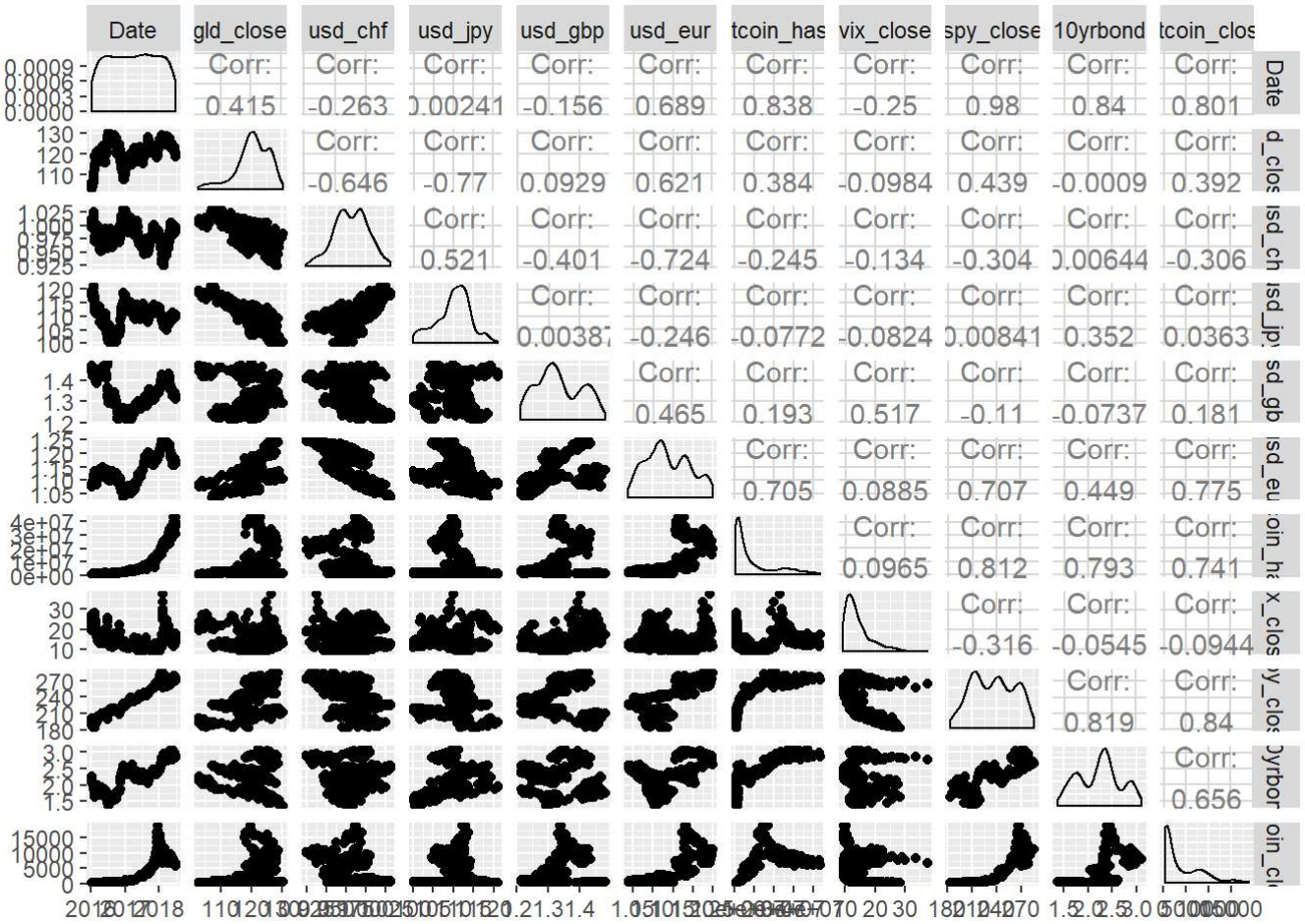
- The R^2 value is: **0.857**
- The Adjusted R^2 value is: **0.8502**
- The p-value of the Shapiro test is: **4.857810^{-15}**
- The p-value of the BP test is: **2.387110^{-5}**
- The LOOCV RMSE value is: **1496.3611**
- The number of Large Influential Data Points value is: **7**

We are already gain high predictive power, but we seem to be violating constant variance and normality assumptions. Therefore, we will first address collinearity and then apply transformations to the response and predictors.

There are also a few data points that appear to be influencing the data. However, it is difficult to simply remove them given this is a financial dataset. In an ideal world, we can identify what's a random shock and what is an outlier. Theoretically, is there a difference and whether it should be removed or our model should be able to identify such datapoints with a certain amount of accuracy. We can continue this in our further discussion later in the report. Systemic market shocks can be predicted, but they very rarely are predicted in practice but rather in hindsight. As they say with hindsight you '20-20' vision, and such is true with data analysis tasks as well.

Colinearity

```
#pairs(data, pch=1, col = "dodgerblue", upper.panel = NULL)
ggpairs(data)
```




```
library(faraway)
vif(btc_lm)
```

```
##    gld_close      usd_chf      usd_jpy      usd_gbp      usd_eur bitcoin_hash
##        7.876        4.040        7.413        3.948       12.956        9.268
##    vix_close    spy_close   `10yrbond`
##        3.827       35.401       14.407
```

Usually a heuristic can be applied to to determine whether multicollinearity issues exists. We can use the value of 10 to identify whether the variance inflation factor is too high for a particular predictor. In that case, we see an large collinearity issue with spy_close, 10yrbond and possibly our exchange rate of usd_eur could be highly correlated with another currency pair such as usd_gbp (due to current political changes between England and the EU).

We define R_j^2 to be the proportion of observed variation in the j^{th} predictor explained by the other predictors. In other words R_j^2 is the multiple R-Squared for the regression of x_j on each of the other predictors. Therefore, we are regression on the predictor itself with every other predictor, to see how much of the variance it is explained by the other variables, which is a pseudo definition of collinearity.

Let us remove these three identified variables, with possible collinearity issues one by one.

```
btc_lm_vif1 = lm(bitcoin_close ~ gld_close + usd_chf + usd_jpy + usd_gbp + usd_eur + bitcoin_has
h + vix_close + `10yrbond`, data=data_trn)
vif(btc_lm_vif1)
```

```
##    gld_close      usd_chf      usd_jpy      usd_gbp      usd_eur bitcoin_hash
##        6.785        4.039        7.113        2.699        7.920        7.487
##    vix_close   `10yrbond`
##        1.781        8.086
```

Therefore, removing SPY has improved our model in terms of colinearity and thus would reduce the variability of our estimation.

```
spy_colinear = lm(spy_close ~ gld_close + usd_chf + usd_jpy + usd_gbp + usd_eur + bitcoin_hash +
vix_close + `10yrbond`, data = data_trn)
btc_colinear = lm(bitcoin_close ~ gld_close + usd_chf + usd_jpy + usd_gbp + usd_eur + bitcoin_ha
sh + vix_close + `10yrbond`, data = data_trn)

cor(resid(spy_colinear), resid(btc_colinear))
```

```
## [1] 0.5185
```

This puts us in a difficult position, that this provides us with predictive power but introduces colinearity issues. We can consider removing other factors first.

```
usd_eur_colinear = lm(usd_eur ~ gld_close + usd_chf + usd_jpy + usd_gbp + spy_close + bitcoin_ha
sh + vix_close + `10yrbond`, data = data_trn)
btc_colinear = lm(bitcoin_close ~ gld_close + usd_chf + usd_jpy + usd_gbp + spy_close + bitcoin_
hash + vix_close + `10yrbond`, data = data_trn)
cor(resid(usd_eur_colinear), resid(btc_colinear))
```

```
## [1] 0.259
```

```
bond_colinear = lm(`10yrbond` ~ gld_close + usd_chf + usd_jpy + usd_gbp + spy_close + bitcoin_ha
sh + vix_close + usd_eur, data = data_trn)
btc_colinear = lm(bitcoin_close ~ gld_close + usd_chf + usd_jpy + usd_gbp + spy_close + bitcoin_
hash + vix_close + usd_eur, data = data_trn)
cor(resid(bond_colinear), resid(btc_colinear))
```

```
## [1] -0.2338
```

```
btc_lm_vif2 = lm(bitcoin_close ~ spy_close + gld_close + usd_chf + usd_jpy + bitcoin_hash + vix_
close, data=data_trn)
vif(btc_lm_vif2)
```

```
##    spy_close    gld_close      usd_chf      usd_jpy bitcoin_hash    vix_close
##        8.823        5.922        1.903        4.834        6.092        2.487
```

```
summary(btc_lm_vif2)$adj.r.squared
```

```
## [1] 0.7715
```

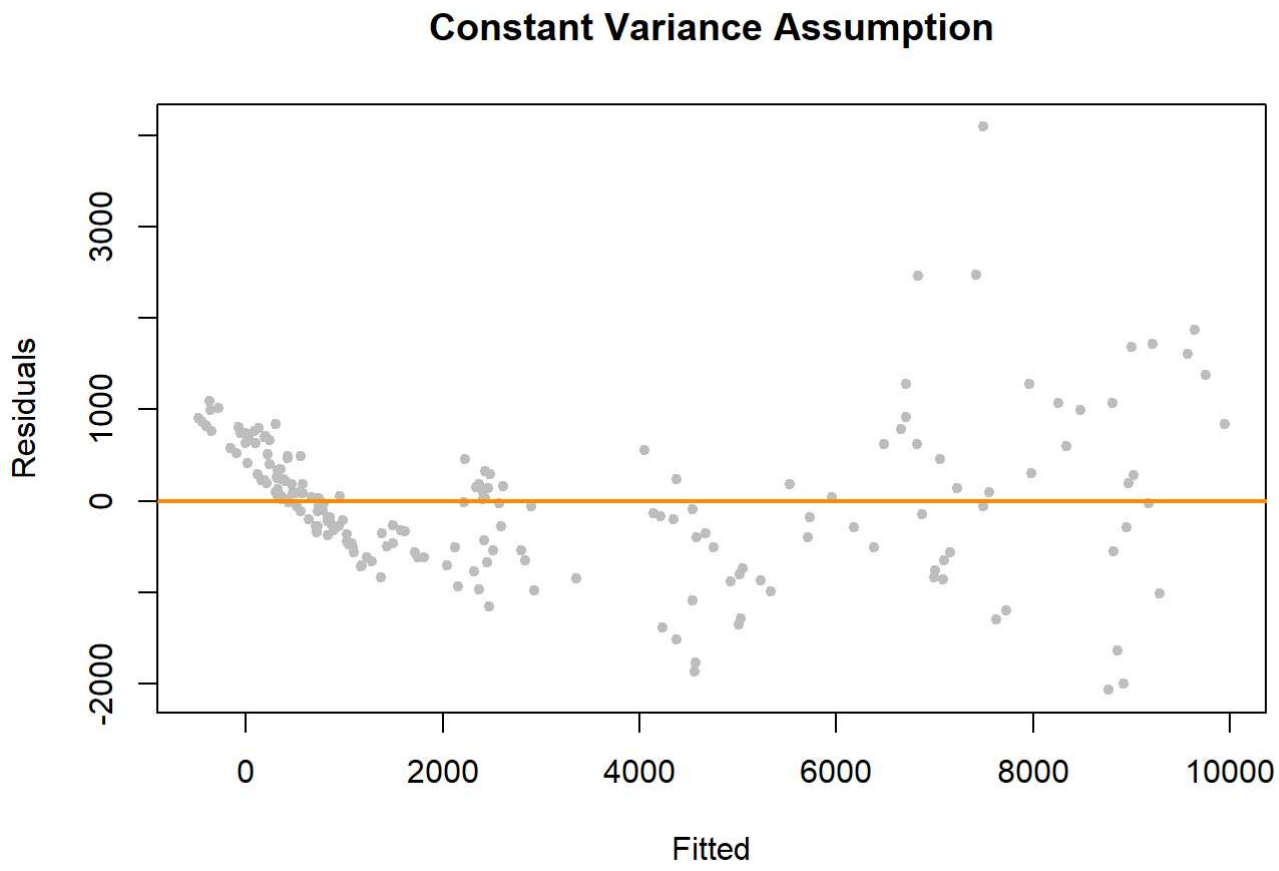
We are sacrificing prediction for improved estimation variability. This will be a theme given our dataset. We will test both approaches.

Removing influential data points

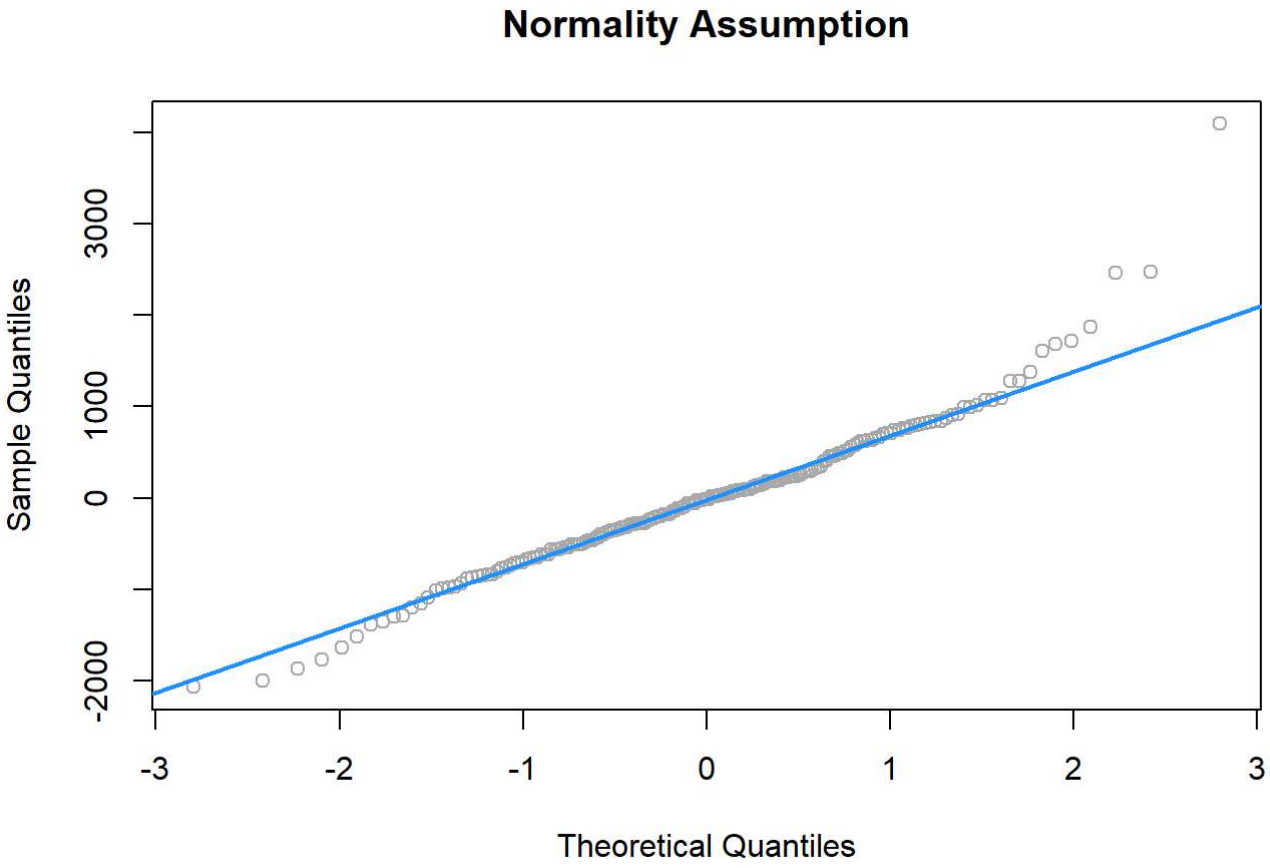
```
btc_lm_fix = lm(bitcoin_close ~ . -Date ,data=data_trn,
                    subset = cd_btc_lm_add <= 4 / length(cd_btc_lm_add))

summary(btc_lm_fix)
```

```
##
## Call:
## lm(formula = bitcoin_close ~ . - Date, data = data_trn, subset = cd_btc_lm_add <=
##     4/length(cd_btc_lm_add))
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -2068   -496    -15    454   4092
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.14e+05   9.17e+03  -12.45  < 2e-16 ***
## gld_close     2.41e+01   3.07e+01    0.78     0.43
## usd_chf       4.46e+04   6.13e+03    7.27  9.8e-12 ***
## usd_jpy      -1.22e+00   3.71e+01   -0.03     0.97
## usd_gbp       7.95e+03   1.78e+03    4.47  1.4e-05 ***
## usd_eur       2.27e+04   4.19e+03    5.41  2.0e-07 ***
## bitcoin_hash -7.41e-05   1.70e-05   -4.36  2.2e-05 ***
## vix_close     2.45e+02   3.07e+01    7.98  1.5e-13 ***
## spy_close     1.43e+02   1.44e+01    9.96  < 2e-16 ***
## `10yrbond`   -8.74e+02   5.39e+02   -1.62     0.11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 822 on 183 degrees of freedom
## Multiple R-squared:  0.936,  Adjusted R-squared:  0.933
## F-statistic:  299 on 9 and 183 DF,  p-value: <2e-16
```

```
plot(fitted(btc_lm_fix), resid(btc_lm_fix), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Constant Variance Assumption")
abline(h = 0, col = "darkorange", lwd = 2)
```




```
qqnorm(resid(btc_lm_fix), main="Normality Assumption", col = "darkgrey")
qqline(resid(btc_lm_fix), col = "dodgerblue", lwd = 2)
```



- The R^2 value is: **0.9364**
- The Adjusted R^2 value is: **0.8502**
- The p-value of the Shapiro test is: **9.725810^{-6}**
- The p-value of the BP test is: **5.871610^{-6}**
- The LOOCV RMSE value is: **851.3005**

Removing of the influencal data points begins to make the model look a lot better. The R^2 is much more in-line with what we would want to see. However, we will still apply transformations before we continue with model selection.

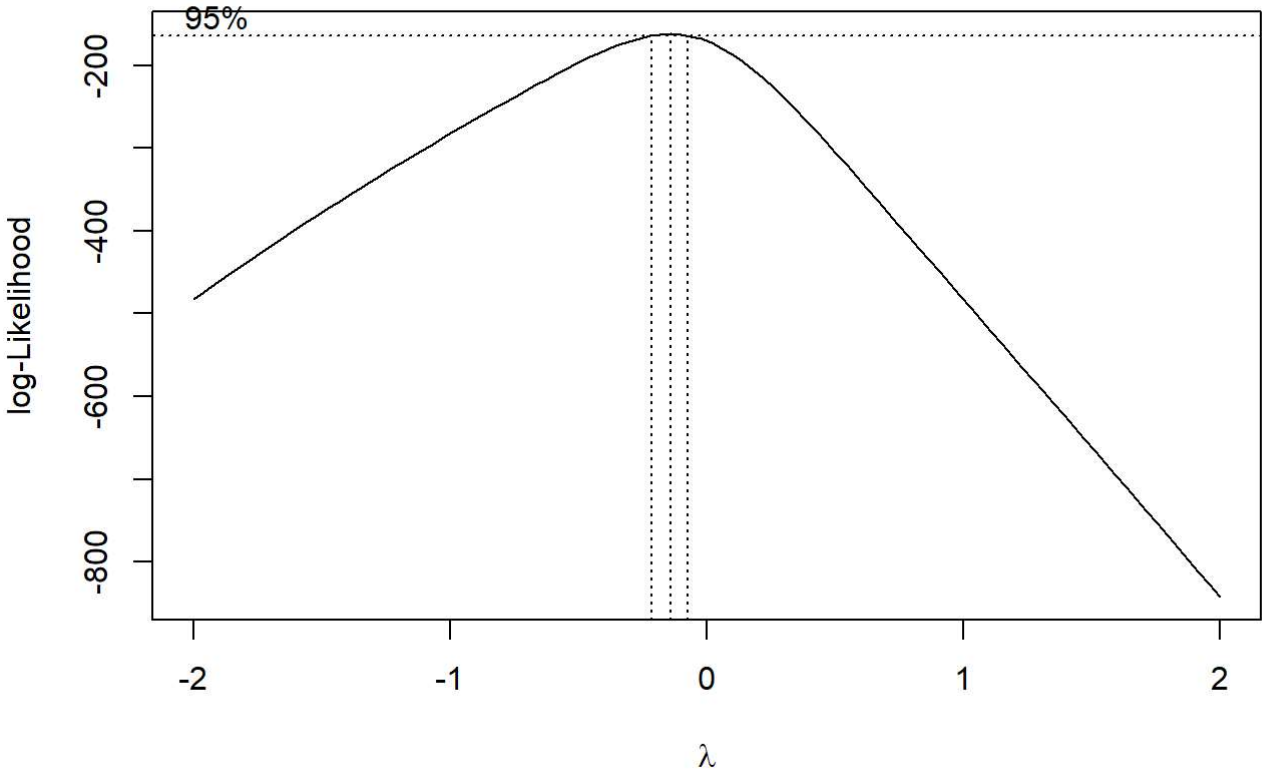
We should try to look at what else we can do. Both with and without the removal of data.

Transformations

Repsonse Transformation

We can attempt to apply a variance stabilizing transformation by using Box Cox.

```
boxcox(bitcoin_close ~ . -Date ,data=data_trn)
```



We see that either a log transformation $\lambda = 0$ or a slightly lower $\lambda = -0.2$, could improve our testing of assumptions.

```
btc_cox = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ . -Date, data = data_trn)
btc_log = lm(log(bitcoin_close) ~ . -Date, data = data_trn)
```

```
summary(btc_cox)$adj.r.squared
```

```
## [1] 0.9798
```

```
summary(btc_log)$adj.r.squared
```

```
## [1] 0.9787
```

```
bptest(btc_log)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  btc_log
## BP = 46, df = 9, p-value = 6e-07
```

```
bptest(btc_cox)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  btc_cox
## BP = 45, df = 9, p-value = 1e-06
```

```
shapiro.test(resid(btc_log))
```

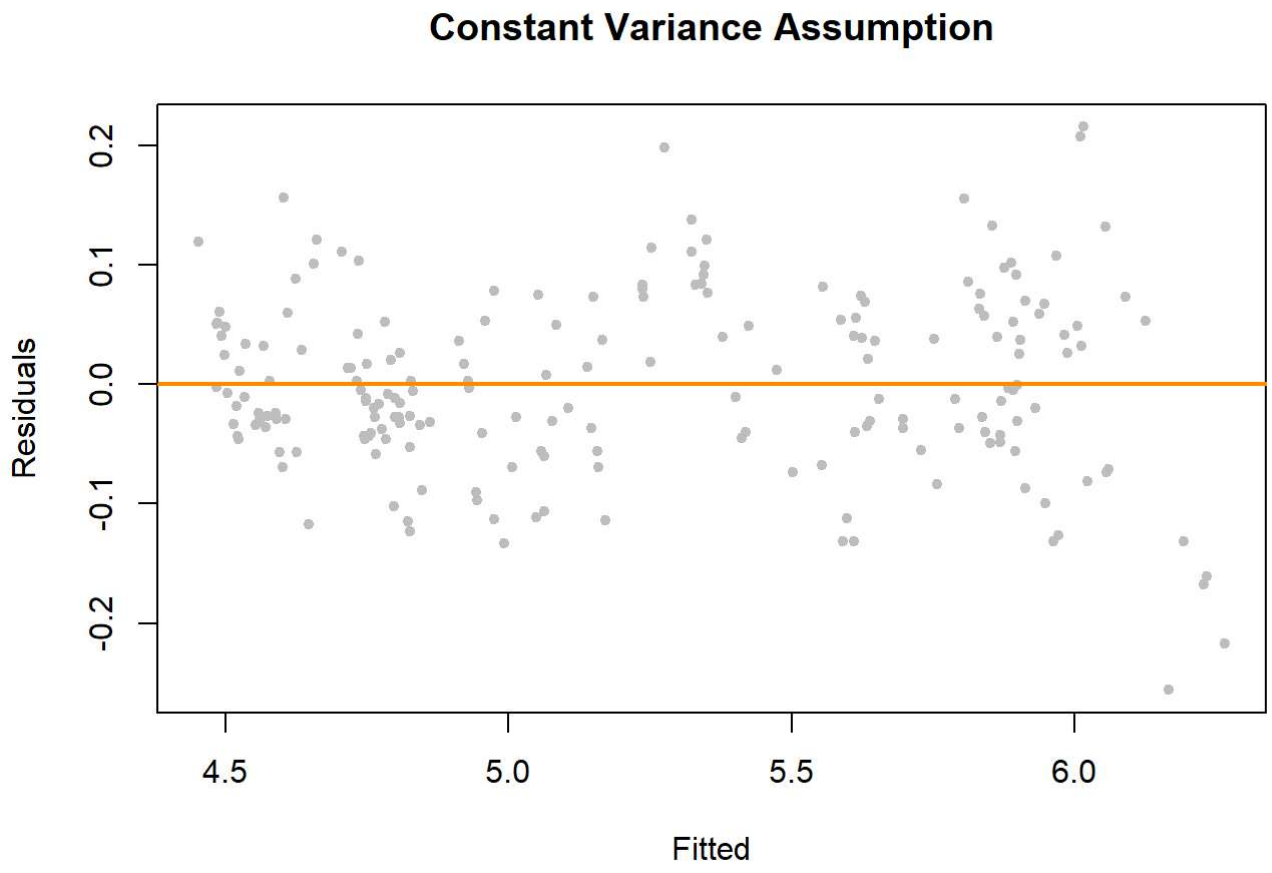
```
##
##  Shapiro-Wilk normality test
##
## data:  resid(btc_log)
## W = 0.99, p-value = 0.06
```

```
shapiro.test(resid(btc_cox))
```

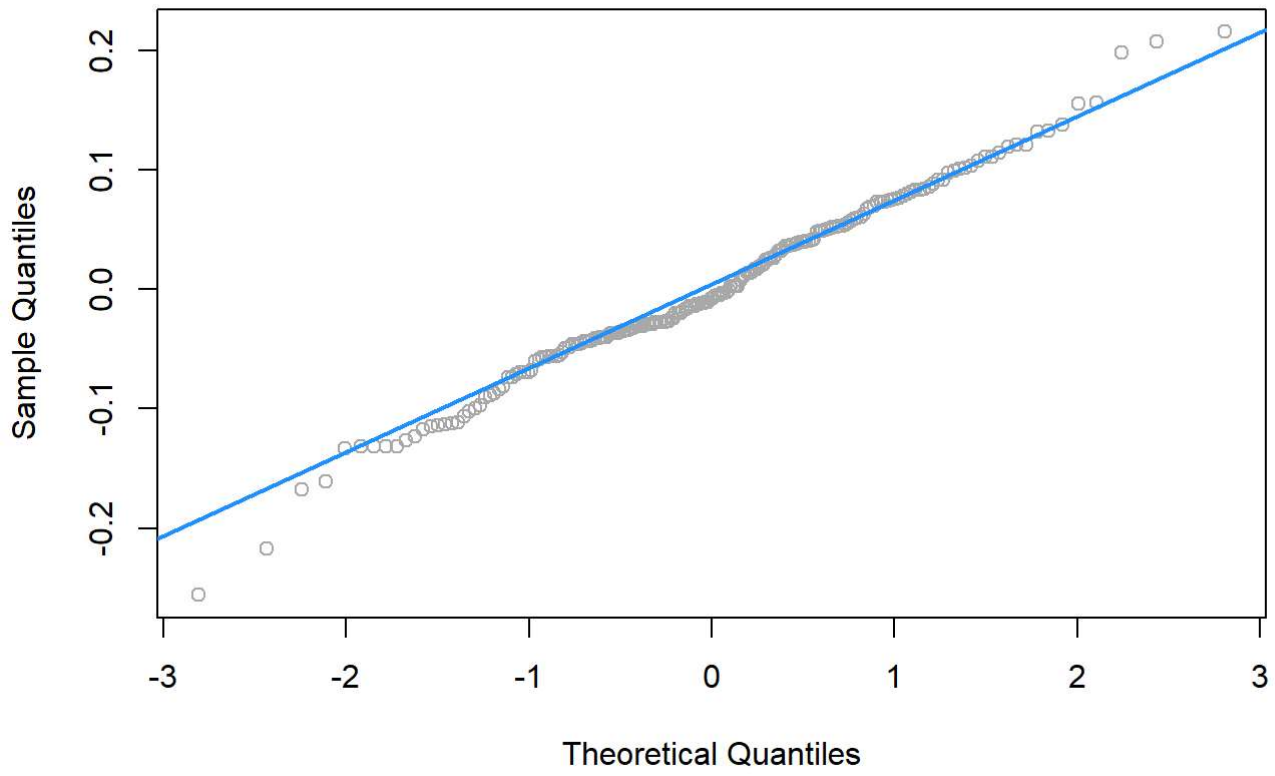
```
##
##  Shapiro-Wilk normality test
##
## data:  resid(btc_cox)
## W = 0.99, p-value = 0.3
```

Therefore, we see a large improvement in terms of our normality assumption with this response transformation.

```
plot(fitted(btc_cox), resid(btc_cox), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Constant Variance Assumption")
abline(h = 0, col = "darkorange", lwd = 2)
```




```
qqnorm(resid(btc_cox), main="Normality Assumption", col = "darkgrey")
qqline(resid(btc_cox), col = "dodgerblue", lwd = 2)
```



Let us try with our improved mdoel according to colinearity.

```
btc_cox_2 = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ spy_close + gld_close + usd_chf + usd_jpy
 + bitcoin_hash + vix_close -Date, data = data_trn)

bptest(btc_cox_2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  btc_cox_2
## BP = 52, df = 6, p-value = 2e-09
```

```
shapiro.test(resid(btc_cox_2))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(btc_cox_2)
## W = 0.99, p-value = 0.5
```

We can consider both models.

Predictor Transformation

Our first predictor that we noticed had a possible non linear trend, was bitcoin hash rate. Let us attempt to add a log transformation to our model.

```
btc_cox_3 = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ spy_close + gld_close + usd_chf + usd_jpy
 + log(bitcoin_hash) + vix_close -Date, data = data_trn)

bptest(btc_cox_3)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  btc_cox_3
## BP = 30, df = 6, p-value = 4e-05
```

```
shapiro.test(resid(btc_cox_3))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(btc_cox_3)
## W = 0.99, p-value = 0.07
```

We sacrifice normality for an improve constant variance.

```
summary(btc_cox_3)$adj.r.squared
```

```
## [1] 0.9478
```

We also sacrifice predictive ability.

```
btc_cox_4 = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ spy_close + poly(gld_close, 2, raw=TRUE)
 + usd_chf + usd_jpy + log(bitcoin_hash) + poly(vix_close, 2, raw=TRUE) -Date, data = data_trn)

bptest(btc_cox_4)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  btc_cox_4
## BP = 40, df = 8, p-value = 3e-06
```

```
shapiro.test(resid(btc_cox_4))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(btc_cox_4)
## W = 0.98, p-value = 0.02
```

This only seems to worsen our testing of assumptions. Therefore, it appears our transformation of bitcoin hash may be the only one that improves our constant variance assumptions.

Alternative models

Evaluating different methods to find a model that best represents the data is essential. Removing data isn't the only option and it needs to be evaluated along with other models before making a decision.

AIC/BIC Testing

```
n = length(resid(btc_lm))
btc_lm_backware_aic = step(btc_lm, direction = "backward", trace = 0 )
btc_lm_backware_bic = step(btc_lm, direction = "backward", k = log(n), trace = 0)
```

Currency Interactions and GLD^2

```
btc_lm_new = lm(bitcoin_close ~ gld_close^2 + (usd_chf * usd_jpy * usd_gbp * usd_eur) + bitcoin_
hash + vix_close + spy_close + `10yrbond` -Date, data = data_trn)
```

No Currency Values

```
btc_lm_new2 = lm(bitcoin_close ~ gld_close  + bitcoin_hash + vix_close + spy_close + `10yrbond`
-Date, data = data_trn)
```

Currency changes only CHF and EUR

```
btc_lm_new3 = lm(bitcoin_close ~ gld_close + usd_chf + usd_eur + bitcoin_hash + vix_close + spy_
close + `10yrbond` -Date, data = data_trn)
```

Currency changes only CHF and EUR and Log(bitcoin_hash)

```
btc_lm_new4 = lm(bitcoin_close ~ gld_close + usd_chf + usd_eur + log(bitcoin_hash) + vix_close +
spy_close + `10yrbond` -Date, data = data_trn)
```

Selected Model with AIC all Interactions and Log Transform

```
btc_wildcard = lm(log(bitcoin_close) ~ (.)^2 -Date, data = data_trn)
aic_full_int = step(btc_wildcard, direction='backward', trace=0)
summary(aic_full_int)$adj.r.squared
```

```
## [1] 0.9921
```

```
bptest(aic_full_int)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  aic_full_int
## BP = 44, df = 38, p-value = 0.2
```

```
shapiro.test(resid(aic_full_int))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(aic_full_int)
## W = 0.98, p-value = 0.01
```

```
coef(aic_full_int)
```

```
##             (Intercept)               gld_close                 usd_chf
##               -4.047e+02                1.982e+00                3.968e+02
##                 usd_jpy                  usd_gbp                  usd_eur
##                7.195e+00               -1.877e+02                4.991e+02
##             bitcoin_hash               vix_close                spy_close
##                7.997e-06               -2.271e+00               -3.487e+00
##              `10yrbond`            usd_jpy:Date            usd_gbp:Date
##              -6.588e+01               -3.319e-04                1.098e-02
##            usd_eur:Date       bitcoin_hash:Date          vix_close:Date
##              -2.458e-02               -4.619e-10                2.222e-04
##          spy_close:Date         `10yrbond`:Date       gld_close:usd_chf
##               1.760e-04                4.631e-03               -1.498e+00
##       gld_close:usd_gbp     gld_close:vix_close     gld_close:spy_close
##              -2.097e-01               -2.665e-03               -8.547e-04
##         usd_chf:usd_jpy         usd_chf:usd_gbp         usd_chf:usd_eur
##              -1.957e+00                5.862e+01               -5.557e+01
##    usd_chf:bitcoin_hash       usd_chf:vix_close         usd_jpy:usd_eur
##               7.355e-07               -1.584e+00               -1.234e+00
##    usd_jpy:bitcoin_hash       usd_jpy:vix_close       usd_jpy:spy_close
##              -7.616e-09                9.590e-03                7.581e-03
##         usd_gbp:usd_eur       usd_gbp:vix_close       usd_gbp:spy_close
##               1.510e+01               -2.792e-01               -1.420e-01
##      usd_gbp:`10yrbond`       usd_eur:spy_close      usd_eur:`10yrbond`
##              -7.891e+00                1.849e-01                2.402e+01
## bitcoin_hash:`10yrbond`    vix_close:`10yrbond`    spy_close:`10yrbond`
##               8.578e-08               -1.644e-01               -1.266e-01
```

Models from reducing colinearity and transformations

```
btc_cox = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ . -Date, data = data_trn)
btc_cox_2 = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ spy_close + gld_close + usd_chf + usd_jpy
 + bitcoin_hash + vix_close -Date, data = data_trn)
btc_cox_3 = lm((((bitcoin_close ^ -0.1) - 1) / -0.1) ~ spy_close + gld_close + usd_chf + usd_jpy
 + log(bitcoin_hash) + vix_close -Date, data = data_trn)
```

Model Comparison

	R^2	Normality	Constant Variance	RMSE (LOOCV)
AIC	0.8513	0.0000	0.0000	1483.7988
BIC	0.8513	0.0000	0.0000	1483.7988
No Currency	0.7798	0.0000	0.0001	1827.7446
Only CHF and EUR	0.8443	0.0000	0.0000	1547.8493
Only CHF,EUR & Log(bitcoin_hash)	0.8505	0.0000	0.0000	1517.9594
Box Cox (1)	0.9807	0.2952	0.0000	0.0803
Box Cox (2)	0.9554	0.4810	0.0000	0.1191
Box Cox (3)	0.9494	0.0681	0.0000	0.1266
Full Interactive AIC with Log Transform	0.9936	0.0115	0.2482	0.1230

Test Validation

Testing Our Models - RMSE

	Train RSME	Test RSME	Difference Between Train and Test
Basic Model	1418	1698	-280.4
Influential Removed	1521	1909	-388.1
BIC	1420	1708	-288.5
AIC	1420	1708	-288.5
GLD Interaction	1143	1630	-486.7
No Currency	1759	2086	-326.7
Only CHF and EUR	1479	1726	-246.5
Only CHF,EUR & Log(bitcoin_hash)	1450	1702	-251.9
Box Cox (1)	5061	5620	-559.6
Box Cox (2)	5061	5620	-559.6
Box Cox (3)	5061	5620	-559.6
Full Interactive with Log Transform	5061	5618	-557.6

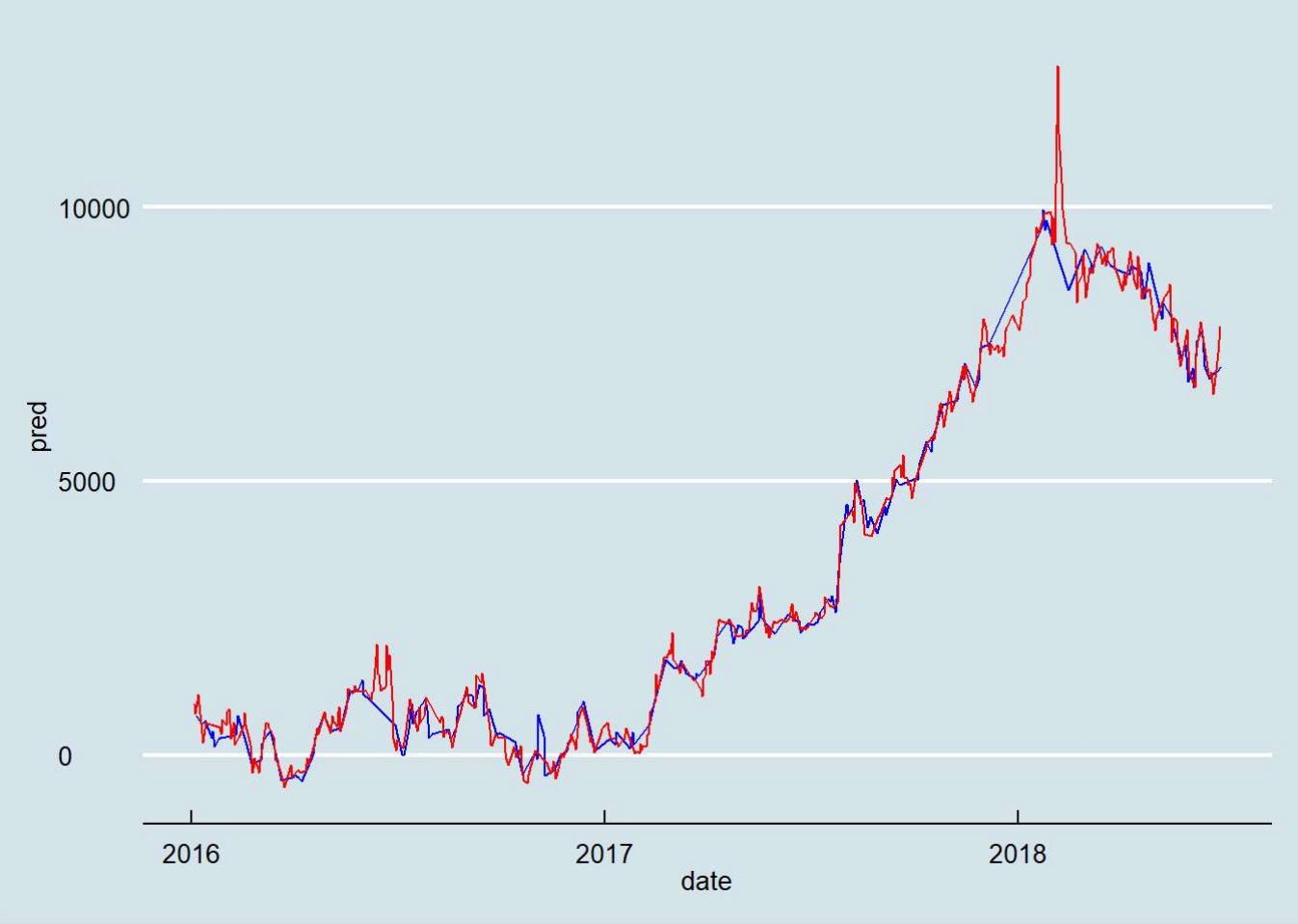
Visualizing the Train and Test Data Sets

Basic Model

```
trn_df = data.frame(pred = fitted(btc_lm), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```

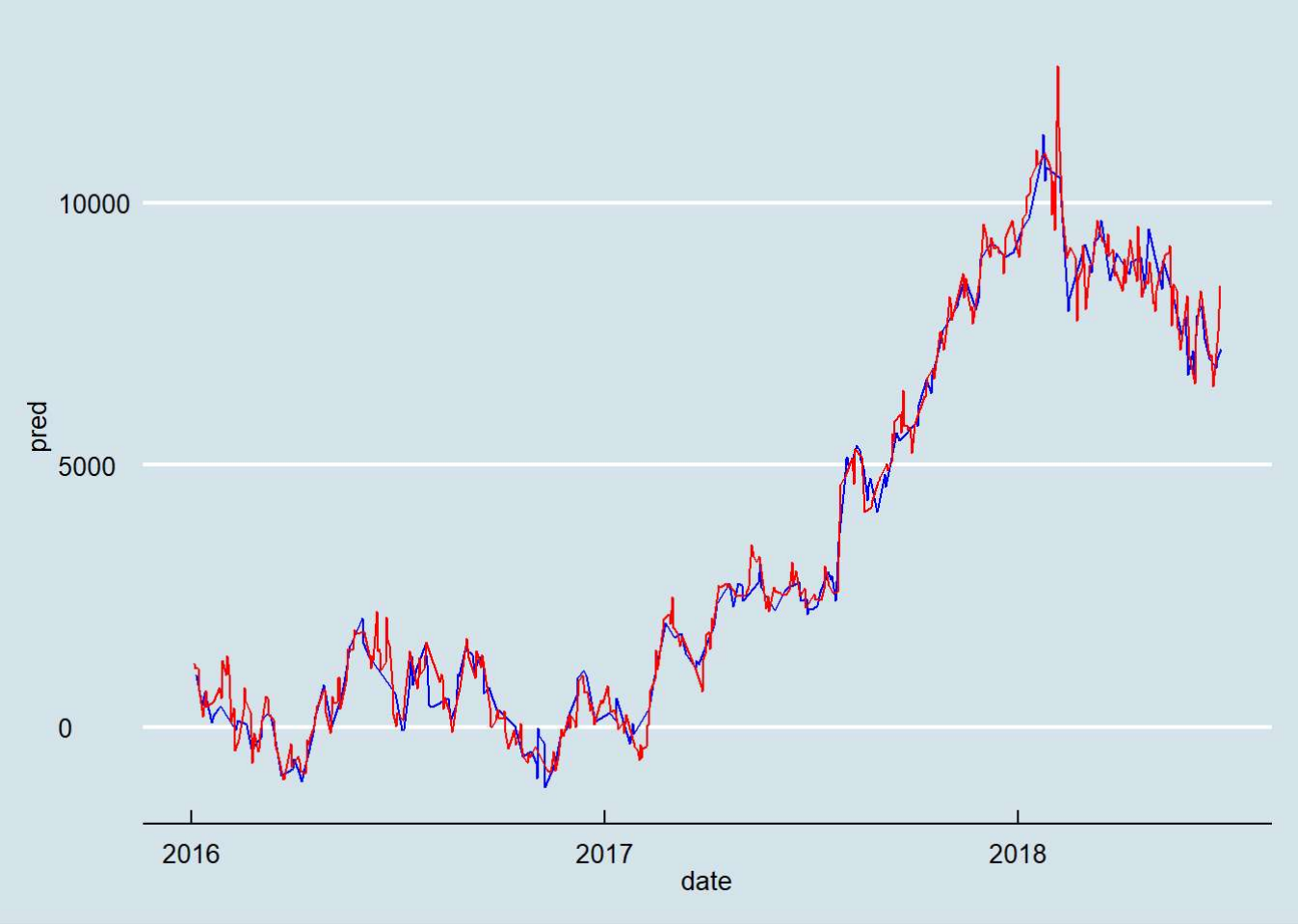
Influential Data Values removed

```
data_trn_inf = data_trn[-which(rownames(data_trn) %in% rownames(as.data.frame(cd_btc_lm_add[large_cd_btc]))), ]

trn_df = data.frame(pred = fitted(btc_lm_fix), date=data_trn_inf$Date)
test_df = data.frame(pred = predict(btc_lm_fix,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```



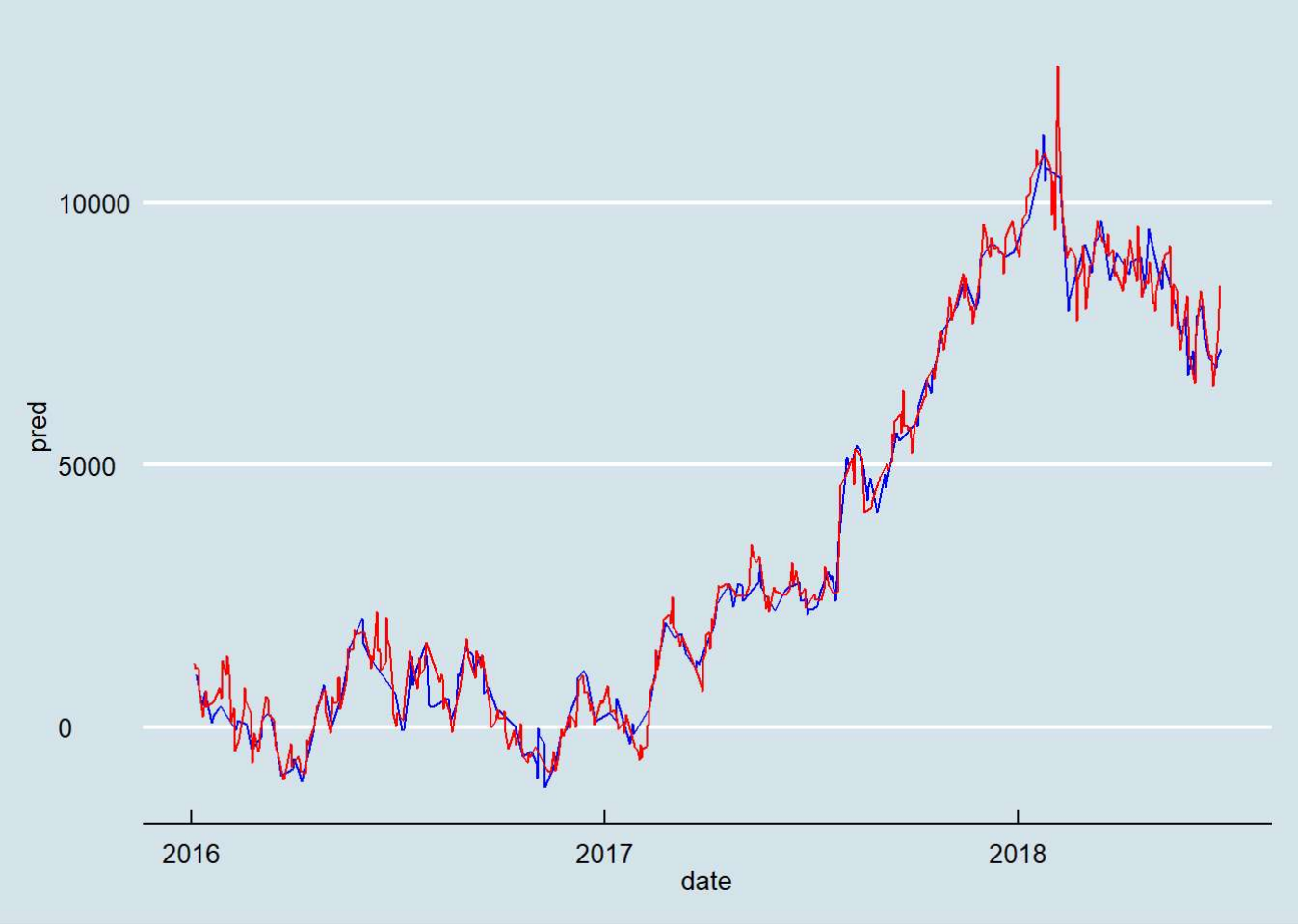
AIC

```
trn_df = data.frame(pred = fitted(btc_lm_backware_aic), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_backware_aic,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```



BIC

```
trn_df = data.frame(pred = fitted(btc_lm_backware_bic), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_backware_bic,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```



GLD Interaction

```
trn_df = data.frame(pred = fitted(btc_lm_new), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_new,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```

No Currency

```
trn_df = data.frame(pred = fitted(btc_lm_new2), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_new2,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```

Only CHF and EUR

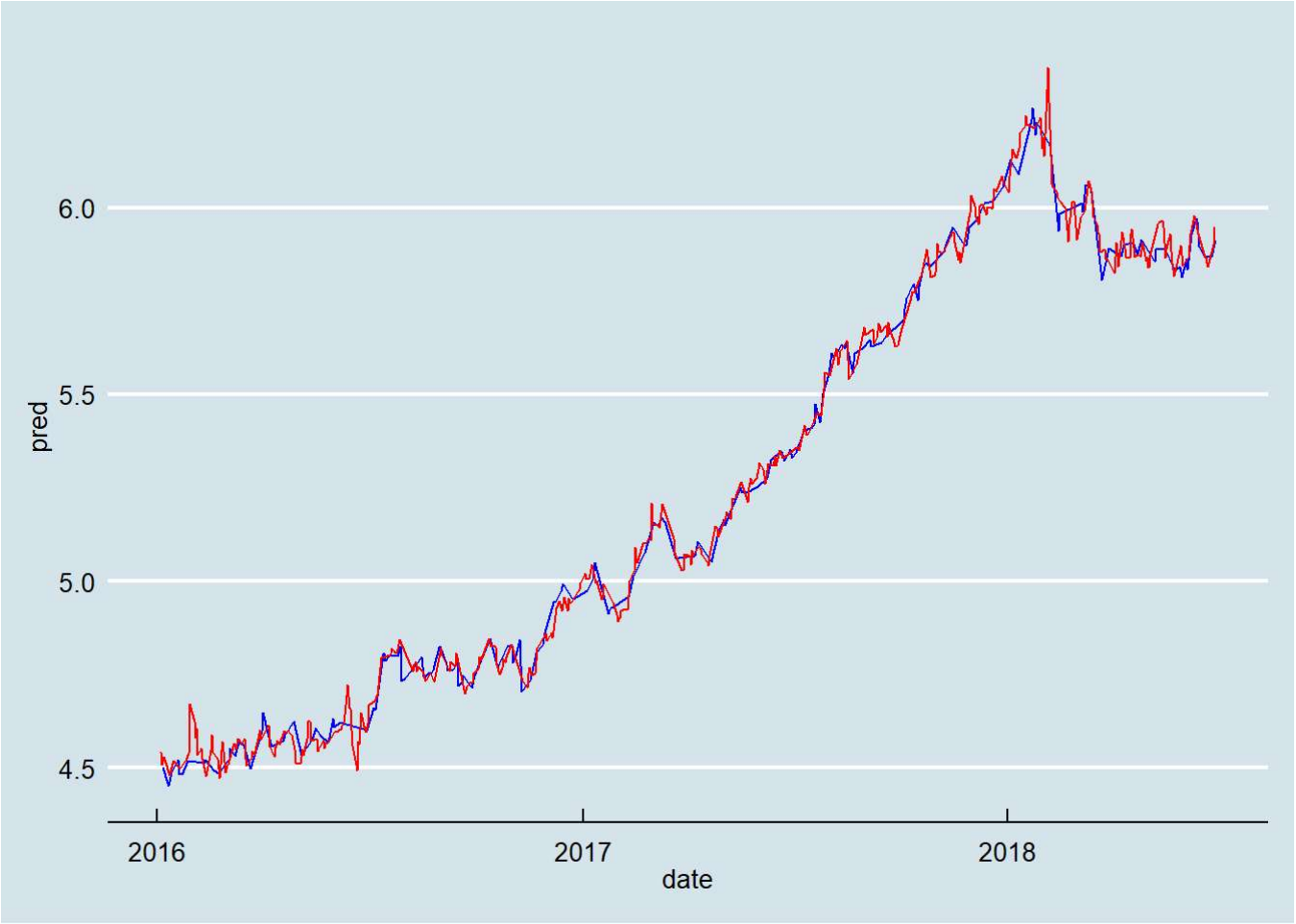
```
trn_df = data.frame(pred = fitted(btc_lm_new3), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_new3,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) +
 scale_colour_manual(values=c("blue","red")) + theme_economist()
```

Only CHF,EUR & Log(bitcoin_hash)

```
trn_df = data.frame(pred = fitted(btc_lm_new4), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_lm_new4,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```

Potential Selected Model (1)

```
trn_df = data.frame(pred = fitted(btc_cox), date=data_trn$Date)
test_df = data.frame(pred = predict(btc_cox,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```



Potential Selected Model (2)

```
trn_df = data.frame(pred = fitted(aic_full_int), date=data_trn$Date)
test_df = data.frame(pred = predict(aic_full_int,data_tst), date=data_tst$Date)

ggplot(data = trn_df, aes(x = date, y = pred)) +
  geom_line(color='blue') +
  geom_line(color='red',data = test_df, aes(x=date, y=pred)) + theme_economist()
```

Discussion

Outcomes and Observations

We were able to find a model that improved upon our assumptions or even failed to reject both our normality and constant variance tests and increase predictive power. We had to apply a variance stabilizing transformation, but we have normality and closer to constant variance. We still have issues with collinearity which need be addressed, but it's not as trivial as removing predictors, since they have real impact on our model.

However, the model that was able to fulfill these requirements was also the least interpretable. We would need real economic interpretability and they didn't seem to be required given our predictor set was quite small. The Box Cox model and full interactive log transformed (response) model also performed poorer in terms of $RMSE$. Therefore, in model building we would want lower variance in predictive power so we would still decide to select either of these, however we sacrifice on certain metrics which other models that have superior $RMSE$ would not necessarily be able to guarantee.

The theme of this class has been, 'All models are wrong, some are useful'. This summarizes what we discovered in this project. In reality, removing observations and apply other statistical techniques can be complex for financial datasets. Determining what is a random shock, or white noise is better dealt with by autoregressive models, and ARMA (ARIMA) models which have an element of autoregression which could be applied to create a stochastic differencing and then the moving average element to properly model white noise (random shocks).

We did however, find that economic indicators and financial markets can impact and help predict circumstances of the cryptocurrency market. Whilst, we are cherry picking the date range we still are confident that some of the factors identified can help our prediction of bitcoin pricing and possibly considering models in a non-multiple linear regression framework, and expanding to non-linear models and ARIMA models may have potential to be useful for bitcoin pricing models in the real world.