Research Question

This project will answer the research question: Does the daily closing price of bitcoin depend on the volume of bitcoin traded that day?

The hypothesis that this project will investigate is: the volume of bitcoins traded daily will affect the final daily closing price. The null hypothesis in this case is, H_0 the final closing price of bitcoin does not depend on the volume of bitcoin trades that day. The alternative hypothesis, H_a is that the final closing price does depend on the volume traded. The significance level for this project will be set at α = 0.05. (McNeese, 2020)

Why study bitcoin? According to (Ullah, 2021) "The bitcoin market cap is over 1 trillion USD with 6350% growth from 2017"

The phenomenal growth of the bitcoin market has made it a topic of great interest. People from all over the world have invested in bitcoin, and several banks and large companies are now heavily involved in the bitcoin market. Finding a factor that could potentially indicate which way the market will swing would be of great use.

Step One: Install the Necessary Packages

In [16]:
```
install.packages("corrplot")
```

```
package 'corrplot' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ContactTracer\AppData\Local\Temp\RtmpmmHeZA\downloaded_packages
```

In [17]:
```
library(corrplot)
```

```
Warning message:
"package 'corrplot' was built under R version 3.6.3"
corrplot 0.84 loaded
```

In [18]:
```
install.packages("PerformanceAnalytics")
```

```
package 'PerformanceAnalytics' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ContactTracer\AppData\Local\Temp\RtmpmmHeZA\downloaded_packages
```

In [19]:
```
library(PerformanceAnalytics)
```

```
Warning message:
"package 'PerformanceAnalytics' was built under R version 3.6.3"
Loading required package: xts

Warning message:
"package 'xts' was built under R version 3.6.3"
Loading required package: zoo

Warning message:
"package 'zoo' was built under R version 3.6.3"

Attaching package: 'zoo'


The following objects are masked from 'package:base':

    as.Date, as.Date.numeric



Attaching package: 'xts'


The following objects are masked from 'package:dplyr':

    first, last



Attaching package: 'PerformanceAnalytics'


The following object is masked from 'package:graphics':

    legend
```

In [30]:

```
library(data.table)
```

```
Warning message:
"package 'data.table' was built under R version 3.6.3"

Attaching package: 'data.table'


The following objects are masked from 'package:xts':

    first, last


The following objects are masked from 'package:dplyr':

    between, first, last


The following object is masked from 'package:purrr':

    transpose
```

In [1]:
```
library(tidyverse)
```

```
Warning message:
"package 'tidyverse' was built under R version 3.6.3"
-- Attaching packages --------------------------------------------------------------------
------------ tidyverse 1.3.0 --

v ggplot2 3.3.2     v purrr   0.3.4
v tibble  3.0.4     v dplyr   1.0.2
v tidyr   1.1.2     v stringr 1.4.0
v readr   1.4.0     v forcats 0.5.0

Warning message:
"package 'ggplot2' was built under R version 3.6.3"
Warning message:
"package 'tibble' was built under R version 3.6.3"
Warning message:
"package 'tidyr' was built under R version 3.6.3"
Warning message:
"package 'readr' was built under R version 3.6.3"
Warning message:
"package 'purrr' was built under R version 3.6.3"
Warning message:
"package 'dplyr' was built under R version 3.6.3"
Warning message:
"package 'stringr' was built under R version 3.6.3"
Warning message:
"package 'forcats' was built under R version 3.6.3"
-- Conflicts -----------------------------------------------------------------------------
------- tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

In [51]:
```
install.packages("caTools")
```

```
package 'caTools' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ContactTracer\AppData\Local\Temp\RtmpmmHeZA\downloaded_packages
```

In [52]:
```
install.packages("ROCR")
```

```
package 'ROCR' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\ContactTracer\AppData\Local\Temp\RtmpmmHeZA\downloaded_packages
```

In [54]:
```
library(caret)
```

```
Warning message:
"package 'caret' was built under R version 3.6.3"
Loading required package: lattice

Warning message:
"package 'lattice' was built under R version 3.6.3"

Attaching package: 'caret'


The following object is masked from 'package:purrr':
```

```
    lift
```

In [99]:
```
library(broom)
```

```
Warning message:
"package 'broom' was built under R version 3.6.3"
```

Step Two: Import and Prepare the Data

The data for this project comes from the website Kaggle.com (Bitcoin Historical Data, 2021)

One challenge in acquiring the data was finding a source that had both closing price and daily trading volume in the same dataset.

About the data: "Bitcoin is the longest running and most well known cryptocurrency, first released as open source in 2009 by the anonymous Satoshi Nakamoto. Bitcoin serves as a decentralized medium of digital exchange, with transactions verified and recorded in a public distributed ledger (the blockchain) without the need for a trusted record keeping authority or central intermediary. Transaction blocks contain a SHA-256 cryptographic hash of previous transaction blocks, and are thus "chained" together, serving as an immutable record of all transactions that have ever occurred. As with any currency/commodity on the market, bitcoin trading and financial instruments soon followed public adoption of bitcoin and continue to grow. Included here is historical bitcoin market data at 1-min intervals for select bitcoin exchanges where trading takes place." (Bitcoin Historical Data, 2021)

Advantages/Disadvantages of the selected data

Advantages

- The data was in an easy to work with format
- The data covered a large period of time
- The data features were well organized

Disadvantages

- The data contained a high number of missing values
- The dataset was Extremely large

The data will be imported using the read.csv function in R.

In [2]:
```
bit_value_USD <- read.csv("bit_value_USD.csv")
```

In [3]:
```
head(bit_value_USD)
```

A data.frame: 6 × 8

	Timestamp	Open	High	Low	Close	Volume_.BTC.	Volume_.Currency.	Weighted_Price
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1325317920	4.39	4.39	4.39	4.39	0.4555809	2	4.39
2	1325317980	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1325318040	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1325318100	NaN	NaN	NaN	NaN	NaN	NaN	NaN
5	1325318160	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	1325318220	NaN	NaN	NaN	NaN	NaN	NaN	NaN

The head function shows that several values are missing from the dataset. These missing values will be dropped using the drop_na() function in R.

In [4]:
```
bit_value_USD1 <-bit_value_USD %>% drop_na()
```

In [5]:
```
head(bit_value_USD1)
```

A data.frame: 6 × 8

	Timestamp	Open	High	Low	Close	Volume_.BTC.	Volume_.Currency.	Weighted_Price
	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	1325317920	4.39	4.39	4.39	4.39	0.4555809	2.00000	4.390000
2	1325346600	4.39	4.39	4.39	4.39	48.0000000	210.72000	4.390000
3	1325350740	4.50	4.57	4.50	4.57	37.8622972	171.38034	4.526411
4	1325350800	4.58	4.58	4.58	4.58	9.0000000	41.22000	4.580000
5	1325391360	4.58	4.58	4.58	4.58	1.5020000	6.87916	4.580000
6	1325431680	4.84	4.84	4.84	4.84	10.0000000	48.40000	4.840000

In [7]:
```
nrow(bit_value_USD1)
```

3484305

The dataset contains almost 3.5 million rows of data, as shown by the nrow() function in R.

The dataset will be trimmed down to 10000 rows of data. This is a large enough set to hopefully see any interesting patterns, without being so large as to be unwieldly. The smaple function will be used to select the 10000 rows for the reduced dataset.

In [11]:
```
bit_value_reduced <- bit_value_USD1[sample(1:nrow(bit_value_USD1), 10000), ]
```

In [12]:
```
summary(bit_value_reduced)
```

```
  Timestamp             Open              High              Low         
Min.   :1.326e+09   Min.   :    4.76   Min.   :    4.76   Min.   :    4.76  
1st Qu.:1.429e+09   1st Qu.:  437.57   1st Qu.:  438.00   1st Qu.:  437.15  
Median :1.502e+09   Median : 2844.78   Median : 2845.42   Median : 2842.78  
Mean   :1.494e+09   Mean   : 4603.12   Mean   : 4606.17   Mean   : 4599.76  
3rd Qu.:1.557e+09   3rd Qu.: 8157.73   3rd Qu.: 8165.81   3rd Qu.: 8154.09  
Max.   :1.609e+09   Max.   :28256.82   Max.   :28262.22   Max.   :28241.46  
    Close          Volume_.BTC.      Volume_.Currency. Weighted_Price   
Min.   :    4.76   Min.   :   0.0000   Min.   :      0   Min.   :    4.76  
1st Qu.:  437.90   1st Qu.:   0.3822   1st Qu.:    433   1st Qu.:  437.77  
Median : 2844.92   Median :   1.9597   Median :   3214   Median : 2844.57  
Mean   : 4602.91   Mean   :   9.1424   Mean   :  33118   Mean   : 4602.99  
3rd Qu.: 8161.07   3rd Qu.:   7.4277   3rd Qu.:  21198   3rd Qu.: 8159.78  
Max.   :28250.05   Max.   :2939.3562   Max.   :3781851   Max.   :28246.94  
```

The data will next be checked for correlations betwen the variables by creating a correlation matrix.

In [13]:
```
cor_mat <- cor(bit_value_reduced)
```

In [14]:
```
cor_mat
```

A matrix: 8 × 8 of type dbl

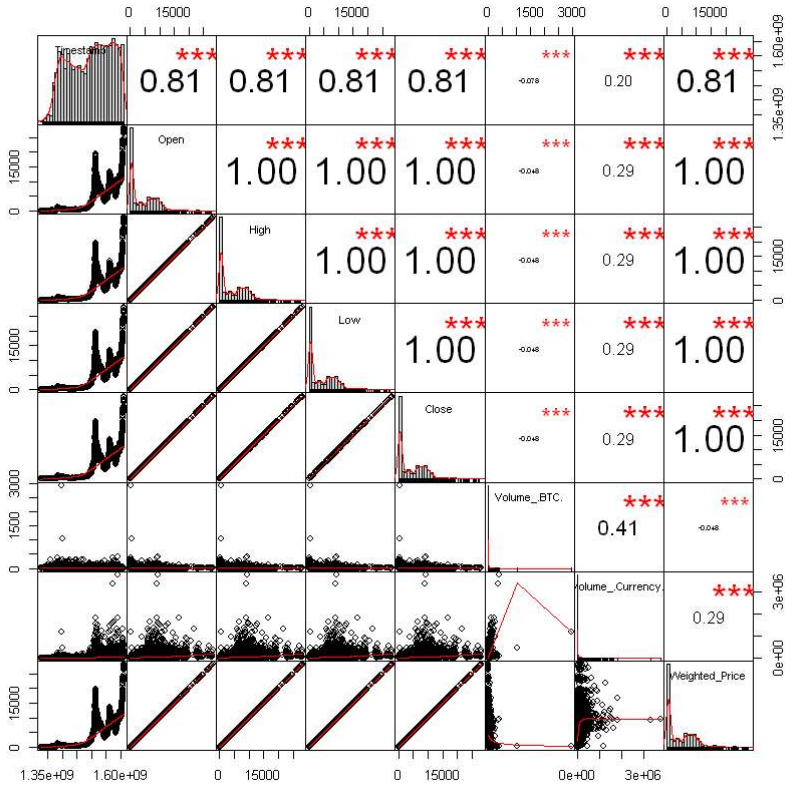
	Timestamp	Open	High	Low	Close	Volume_.BTC.	Volum
Timestamp	1.00000000	0.80573889	0.80561801	0.80590779	0.80577236	-0.07765982	
Open	0.80573889	1.00000000	0.99999895	0.99999869	0.99999824	-0.04822468	
High	0.80561801	0.99999895	1.00000000	0.99999776	0.99999874	-0.04805297	
Low	0.80590779	0.99999869	0.99999776	1.00000000	0.99999895	-0.04847836	
Close	0.80577236	0.99999824	0.99999874	0.99999895	1.00000000	-0.04828131	
Volume_.BTC.	-0.07765982	-0.04822468	-0.04805297	-0.04847836	-0.04828131	1.00000000	
Volume_.Currency.	0.20114881	0.28963192	0.29010355	0.28895801	0.28950020	0.40539005	
Weighted_Price	0.80576285	0.99999923	0.99999922	0.99999935	0.99999932	-0.04828516	

Looking at the output of the cor function there is a small negative correlation betwen closing price and volume traded. This implies that the price of bitcoin finishes lower on days in which a higher volume of coins are traded.

In [79]:

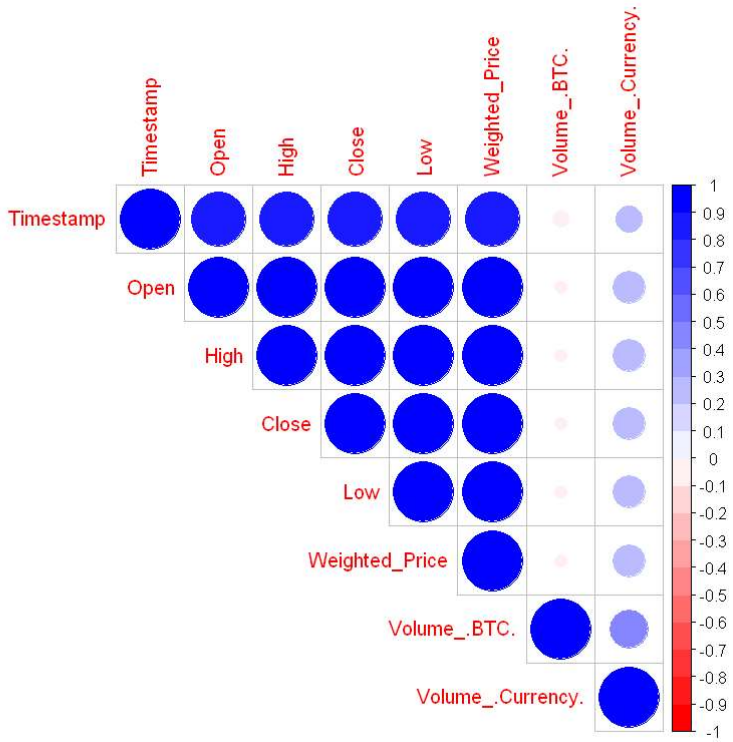
```
chart.Correlation(bit_value_reduced, histogram=TRUE, pch=3)
```

```
Warning message in breaks[-1L] + breaks[-nB]:
"NAs produced by integer overflow"
```



In [43]:

```
col<- colorRampPalette(c("red", "white", "blue"))(20)
corrplot(cor_mat, method="circle", type="upper", order="hclust", col=col)
```



The above graphs both confirm what was shown by the correlation matrix, there is little to no correlation between volume traded and daily closing price.

Step Three: Fitting the models

The timestamp column of the dataset does not contain useful information seeing as closing price happens at a fixed time everyday. This column will be dropped from the dataset as it's inclusion offers no benefit.

In [22]:
```
data_clean <- select(bit_value_reduced, -1)
```

Next the data will be normalized such that all values are between 0 and 1. (Mulani, 2021)

In [64]:
```
data_norm <- preProcess(as.data.frame(data_clean), method=c("range"))
```

In [65]:
```
data_scaled.df <- predict(data_norm, as.data.frame(data_clean))
```

R does not like parenthesis in column names and so those names will be fixed.

In [66]:
```
```

```
setnames(data_scaled.df, old=c("Volume_.BTC.","Volume_.Currency."), new=c("volume_coin"
```

In [67]:

```
head(data_scaled.df)
```

A data.frame: 6 × 7

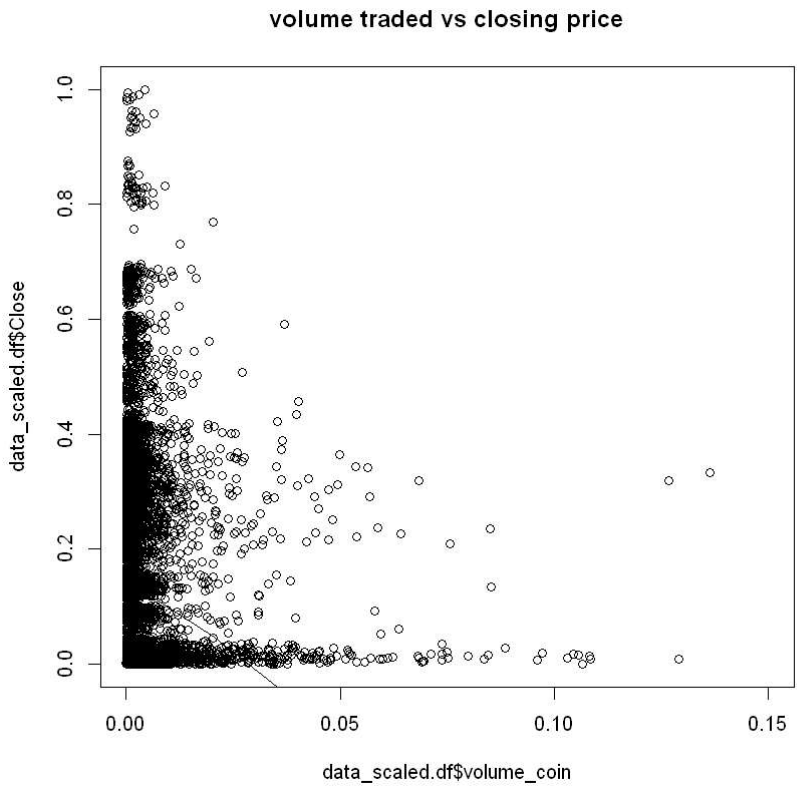
	Open	High	Low	Close	volume_coin	volume_currency	Weighted
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
2620776	0.190657602	0.190621167	0.190717754	0.190659752	9.026664e-06	3.788837e-05	0.1907
326786	0.022555523	0.022551213	0.022178229	0.022454717	2.955550e-02	1.455415e-02	0.0222
530477	0.021995564	0.022065678	0.022004342	0.022075185	2.051797e-03	9.990695e-04	0.0220
804769	0.008759361	0.008757687	0.008750314	0.008747653	1.290845e-03	2.529589e-04	0.0087
2499962	0.122084549	0.122111825	0.122084734	0.122107792	1.265007e-03	3.395703e-03	0.1221
295495	0.027652143	0.027784875	0.027667185	0.027658771	2.818166e-03	1.727847e-03	0.0277

The dataset is now fully clean and prepared for use.

Next a scatterplot of volume traded vs closing price will be created in order to see, in greater detail, if any relationship exists between the variables.

In [81]:

```
scatter.smooth(x=data_scaled.df$volume_coin, y=data_scaled.df$Close, xlim = c(0,0.15),
```

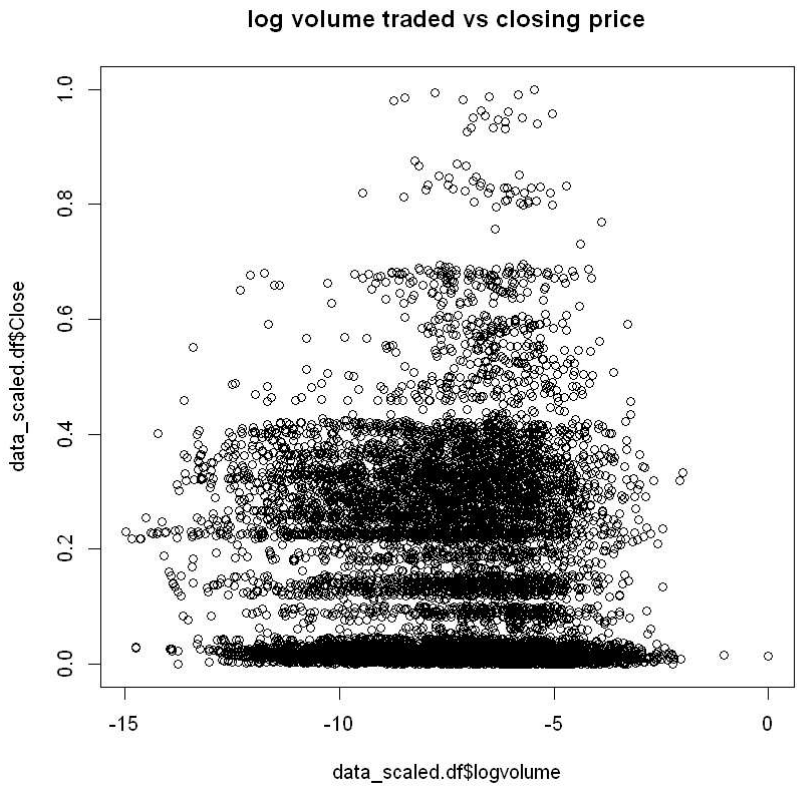


Looking at the graph of closing price versus volume traded, it is clear that no linear relationship exists between the quantities. Even though no linear relationship is apparent in the above graph, a linear model will still be fit to the data as a way to check goodness of fit.

In the case in which no linear relationship exsits between the variables one approach is to transform one of the variables and then check if a linear relationship exists between the transformed variable and the non-transformed variable. (Log in R - Transforming Your Data, 2019)

In [82]:
```
data_scaled.df$logvolume=log(data_scaled.df$volume_coin)
```

In [87]:
```
plot(x=data_scaled.df$logvolume, y=data_scaled.df$Close, ylim=c(0,1), main="log volume
```



The graph of the log of the volume traded vs the closing price shows much more of a linear trend, upwards and to the right. This transformation should produce better results than the original untransformed data would have.

Creating the Linear Regression model:

Before a linear model can be fit to the data the assumptions of linear regression must be met. (Zach, 2021)

- Linear relationship: There exists a linear relationship between the independent variable, x, and the dependent variable, y.
- Independence: The residuals are independent. In particular, there is no correlation between consecutive residuals in time series data.
- Homoscedasticity: The residuals have constant variance at every level of x.
- Normality: The residuals of the model are normally distributed.

Advantages and disadvantages of linear regression (Kumar et al., 2019):

Advantages

- Linear Regression performs well when the dataset is linearly separable. We can use it to find the nature of the relationship among the variables.
- Linear Regression is easier to implement, interpret and very efficient to train.

Disadvantages

- Main limitation of Linear Regression is the assumption of linearity between the dependent variable and the independent variables. In the real world, the data is rarely linearly separable. It assumes that there is a straight-line relationship between the dependent and independent variables which is incorrect many times.
- Prone to outliers: Linear regression is very sensitive to outliers (anomalies). So, outliers should be analyzed and removed before applying Linear Regression to the dataset.

The data will be split into a training and a testing set to allow for testing of the models. (Sharma, 2020)

In [92]:
```
sample_size = round(nrow(data_scaled.df)*.70)
index <- sample(seq_len(nrow(data_scaled.df)), size = sample_size)
```

In [93]:
```
train <- data_scaled.df[index, ]
test <- data_scaled.df[-index, ]
```

In [95]:
```
linearMod_train <- lm(High ~ volume_coin, data=train)
print(linearMod_train)
```

```
Call:
lm(formula = High ~ volume_coin, data = train)

Coefficients:
(Intercept)  volume_coin
     0.1658      -0.5737
```

In [96]:
```
summary(linearMod_train)
```

```
Call:
lm(formula = High ~ volume_coin, data = train)

Residuals:
     Min       1Q   Median       3Q      Max
-0.16577 -0.14862 -0.04735  0.12427  0.83664

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.165808   0.002104  78.817  < 2e-16 ***
volume_coin -0.573666   0.138454  -4.143 3.46e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1721 on 6998 degrees of freedom
```

```
Multiple R-squared:  0.002447,  Adjusted R-squared:  0.002305
F-statistic: 17.17 on 1 and 6998 DF,  p-value: 3.463e-05
```

In [97]:
```
linearMod_test <- lm(High ~ volume_coin, data=test)
print(linearMod_test)
```

```
Call:
lm(formula = High ~ volume_coin, data = test)

Coefficients:
(Intercept)  volume_coin
     0.1636      -1.1631
```

In [98]:
```
summary(linearMod_test)
```

```
Call:
lm(formula = High ~ volume_coin, data = test)

Residuals:
     Min       1Q   Median       3Q      Max
-0.16317 -0.14502 -0.06311  0.12574  0.83335

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.163582   0.003301  49.549  < 2e-16 ***
volume_coin -1.163119   0.415236  -2.801  0.00513 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1681 on 2998 degrees of freedom
Multiple R-squared:  0.00261,   Adjusted R-squared:  0.002278
F-statistic: 7.846 on 1 and 2998 DF,  p-value: 0.005126
```

In [100…
```
tidy(linearMod_train)
```

A tibble: 2 × 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	0.1658084	0.002103727	78.816502	0.000000e+00
volume_coin	-0.5736662	0.138454460	-4.143357	3.462913e-05

In [101…
```
tidy(linearMod_test)
```

A tibble: 2 × 5

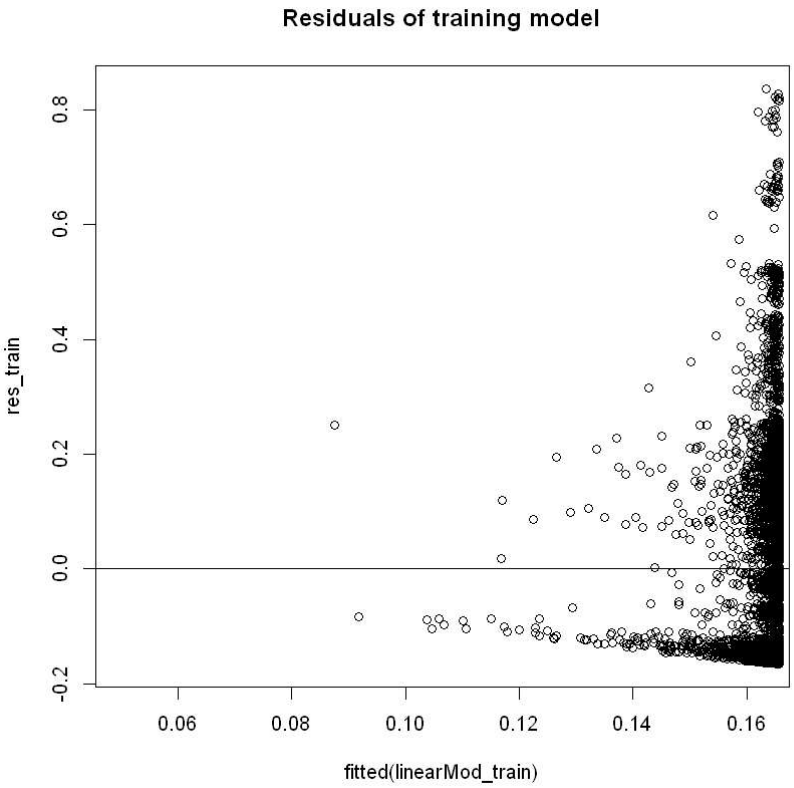
term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	0.1635821	0.003301394	49.549404	0.0000000
volume_coin	-1.1631185	0.415236032	-2.801102	0.0051255

To test the assumption that the residuals are normally distributed a plot of the residuals for the training/testing models will be created.

A line at the point (0,0) will be added. The expectation is that an equal number of residuals will be above and below the line. (Zach, 2021b)

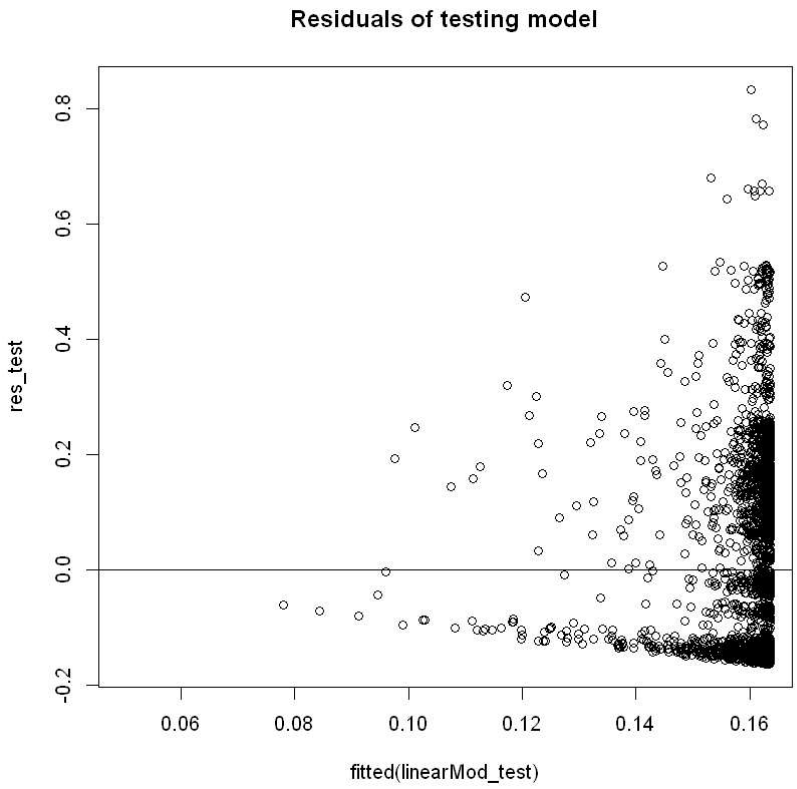
In [108…
```
res_train <- resid(linearMod_train)
```

In [118…
```
plot(fitted(linearMod_train), res_train, xlim = c(0.05,0.163), main="Residuals of train
abline(0,0)
```



In [119…
```
res_test <- resid(linearMod_test)
```

In [120…
```
plot(fitted(linearMod_test), res_test, xlim = c(0.05,0.163), main="Residuals of testing
abline(0,0)
```



In both graphs the points are fairly evenly distributed above and below the line. The assumption of normality of residuals seems to be upheld.

Next the metrics for the models will be calculated and compared. (Regression Model Accuracy Metrics: R-Square, AIC, BIC, Cp and More, 2018)

In [122…

```
glance(linearMod_train) %>%
  dplyr::select(adj.r.squared, sigma, AIC, BIC, p.value)
```

A tibble: 1 × 5

adj.r.squared	sigma	AIC	BIC	p.value
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0.002304636	0.1720887	-4767.296	-4746.735	3.462913e-05

In [123…

```
glance(linearMod_test) %>%
  dplyr::select(adj.r.squared, sigma, AIC, BIC, p.value)
```

A tibble: 1 × 5

adj.r.squared	sigma	AIC	BIC	p.value
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
0.002277619	0.1681399	-2180.124	-2162.104	0.0051255

Interpretation of results:

In both cases the p-value was significantly less than the value of alpha. In this case we would reject the null hypothesis and accept the alternative hypothesis. The linear model shows a definite relationship between volume traded and closing price.

Creating the Logistic Regression model

Advantages and disadvantages of logistic regression

Advantages of Logistic Regression

- Logistic regression is easier to implement, interpret, and very efficient to train.
- It makes no assumptions about distributions of classes in feature space.
- Good accuracy for many simple data sets and it performs well when the dataset is linearly separable.

Disadvantages of Logistic Regresion

- It can only be used to predict discrete functions. Hence, the dependent variable of Logistic Regression is bound to the discrete number set.
- Non-linear problems can't be solved with logistic regression because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.
- Logistic Regression requires average or no multicollinearity between independent variables.

In [102…

```
log_train <- glm(High ~ volume_coin, data=train, family = "binomial")
```

```
Warning message in eval(family$initialize):
"non-integer #successes in a binomial glm!"
```

In [103…

```
summary(log_train)
```

```
Call:
glm(formula = High ~ volume_coin, family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.6091  -0.4972  -0.1360   0.3089   1.9116

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.58812    0.03503 -45.342  < 2e-16 ***
volume_coin -14.86085    5.45466  -2.724  0.00644 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 1559.5  on 6999  degrees of freedom
Residual deviance: 1549.8  on 6998  degrees of freedom
AIC: 3423.2

Number of Fisher Scoring iterations: 5
```

In [104…]
```
log_test <- glm(High ~ volume_coin, data=test, family = "binomial")
```

```
Warning message in eval(family$initialize):
"non-integer #successes in a binomial glm!"
```

In [105…]
```
summary(log_test)
```

```
Call:
glm(formula = High ~ volume_coin, family = "binomial", data = test)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-0.5973  -0.4926  -0.1854   0.3137   1.8885

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.62871    0.05393 -30.203   <2e-16 ***
volume_coin -10.25929    8.02021  -1.279    0.201
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 649.93  on 2999  degrees of freedom
Residual deviance: 648.09  on 2998  degrees of freedom
AIC: 1432.8

Number of Fisher Scoring iterations: 4
```

In [106…]
```
tidy(log_train)
```

A tibble: 2 × 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-1.588116	0.03502554	-45.341656	0.000000000
volume_coin	-14.860852	5.45466185	-2.724431	0.006441231

In [107…]
```
tidy(log_test)
```

A tibble: 2 × 5

term	estimate	std.error	statistic	p.value
<chr>	<dbl>	<dbl>	<dbl>	<dbl>
(Intercept)	-1.628714	0.05392533	-30.203133	2.154498e-200
volume_coin	-10.259286	8.02021219	-1.279179	2.008341e-01

Interpretation of results:

In the case of the logistic regression model the p-value for the testing datsaset was less than α while the p-value for the testing dataset was greater than α. For the training dataset we would reject the null hypothesis, but for the testing dataset we would accept the null hypothesis. This results shows that the logistic regression model is a poor choice for this particular problem. Logistic regression is typically best suited for binary classification problems, problems where the output is either a 1 or a 0. In this problem the values were scaled to be between 1 and 0, but they could take on any value within that range. This prolem was not a binary classification problem and logisitic regression was clearly lacking in terms of analyzing this dataset.

Conclusion:

The goal of this project was to show that it is possible to create a model that can predict the closing price of bitcoin knowing the volume of trades that day. The linear model was successful in this capacity, but the logistic regression model was not. The recommendation based on this analysis would be, for any company interested in investing in bitcoin now would be to use a linear regression model to analyze the data. Once a significantly lower volume day was identified then money could be invested with the idea that as the volume increases the closing price will increase as well.

Directions for future study:

- Rather then study the effects of volume traded on closing price a study could be done on what factors effect the daliy high/low price. Knowing when a high or low price would happen would be extremely useful for predicting when to invest.
- This analysis, using a linear regression model, to predict closing price could be applied to different stocks or commodities to see if any interesting patterns exist there.

References:

Ullah. (2021, March 16). 21 Stats About The Global Bitcoin Market. Https://Finance.Yahoo.Com/. https://finance.yahoo.com/news/21-stats-global-bitcoin-market

Bitcoin Historical Data. (2021, January 4). Kaggle. https://www.kaggle.com/mczielinski/bitcoin-historical-data

Mulani, S. (2021, January 5). How to Normalize data in R [3 easy methods]. JournalDev. https://www.journaldev.com/47850/normalize-data-in-r

Log in R - Transforming Your Data. (2019, September 17). ProgrammingR. https://www.programmingr.com/tutorial/log-in-r/

Sharma. (2020, May 16). Six Amazing Function To Create Train Test Split In R. R Statistics Blog. https://rstatisticsblog.com/data-science-in-action/data-preprocessing/six-amazing-function-to-create-train-test-split-in-r/

Z. (2021b, March 18). How to Create a Residual Plot in R. Statology. https://www.statology.org/residual-plot-r/

Regression Model Accuracy Metrics: R-square, AIC, BIC, Cp and more. (2018, March 11). Articles - STHDA. http://www.sthda.com/english/articles/38-regression-model-validation/158-regression-model-accuracy-metrics-r-square-aic-bic-cp-and-more/

D., & McNeese, B. (2020, April 25). Hypothesis Testing. BPI Consulting. https://www.spcforexcel.com/knowledge/basic-statistics/hypothesis-testing

Kumar, N., Kumar, N., & Profile, V. M. C. (2019). Advantages and Disadvantages of Linear Regression in Machine Learning. The Professionals Point. https://theprofessionalspoint.blogspot.com/2019/05/advantages-and-disadvantages-of-linear.html