Forcasting Bitcoin Price Using ARIMA Model

Longyin Poon

4/10/2018

Training dataset includes 230 weeks from 4/8/2013 to 8/21/2017

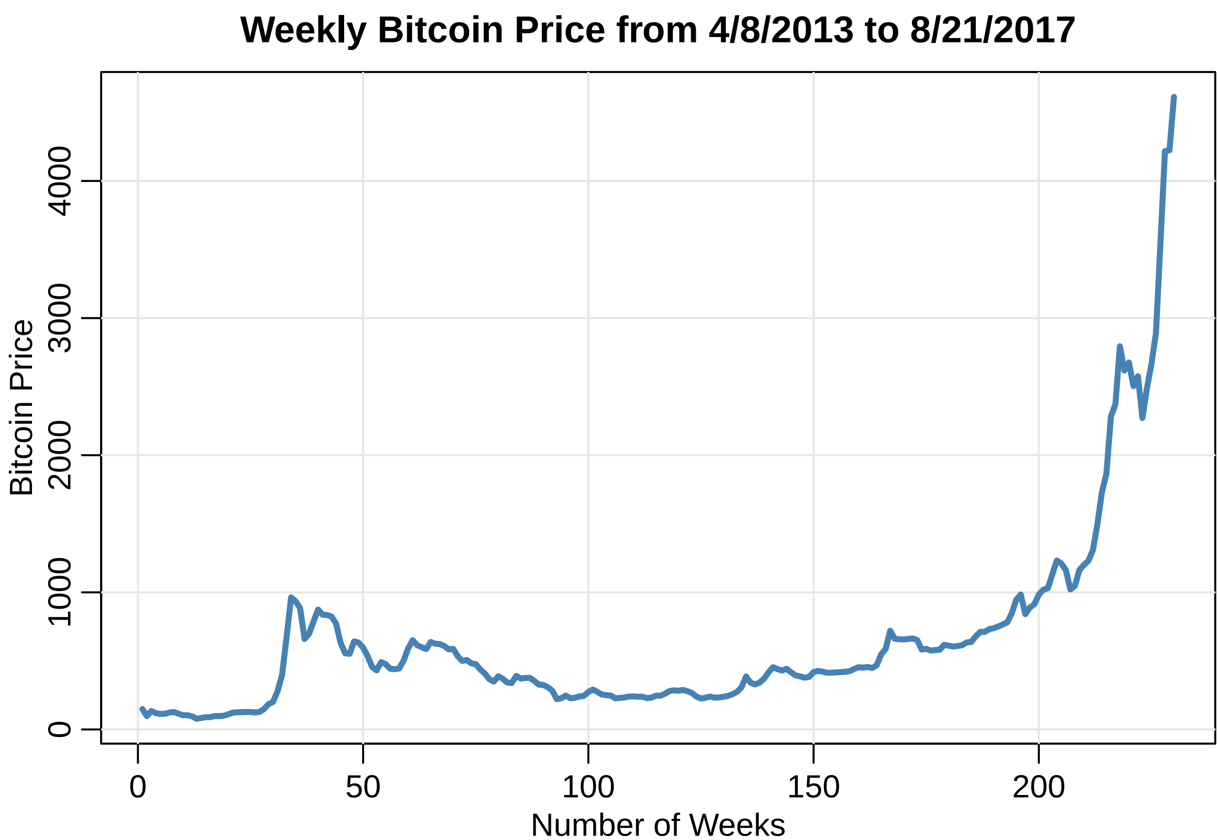
Testing dataset consists 30 weeks of data from 8/28/2017 to 3/26/2018

```r
# Load dataset
bitcoin_data <- read_csv("bitcoin_data.csv")
training <- bitcoin_data[1:230, ]
testing <- bitcoin_data[231:260, ]
```

Time Series Analysis

Statitonary Test

```r
# Statitonary Test
tsplot(x = training$week, y = training$bitcoin_price, ylab = "Bitcoin Price",
    xlab = "Number of Weeks", main = "Weekly Bitcoin Price from 4/8/2013 to 8/21/2017",
    col = "steelblue", lwd = 3)
```




```
adf.test(bitcoin_ts, alternative = "stationary")
```

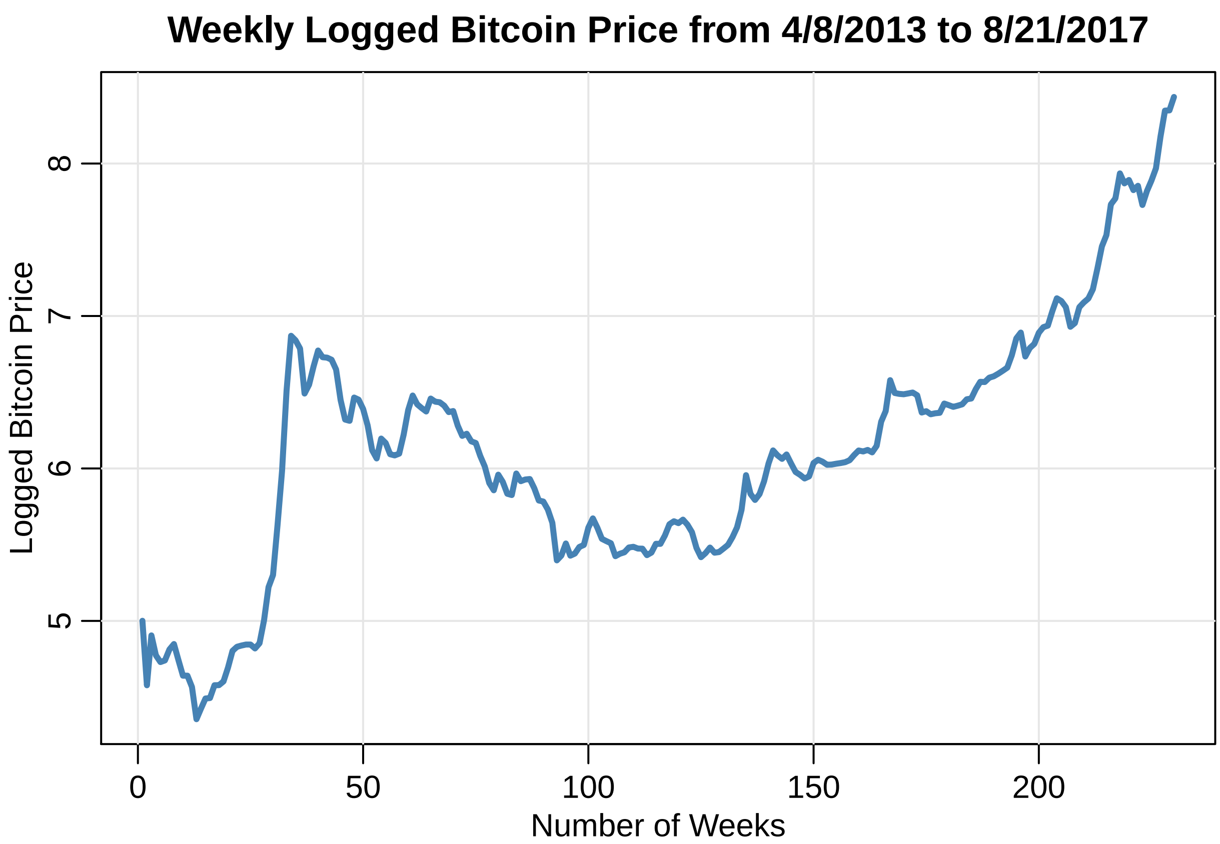
```
## Warning in adf.test(bitcoin_ts, alternative = "stationary"): p-value
## greater than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  bitcoin_ts
## Dickey-Fuller = 3.5991, Lag order = 6, p-value = 0.99
## alternative hypothesis: stationary
```

```
# Fail to reject the series is non-stationary with high
# p-value
```

Log transformation

```
training$log_price <- log(training$bitcoin_price)
tsplot(x = training$week, y = training$log_price, ylab = "Logged Bitcoin Price",
    xlab = "Number of Weeks", main = "Weekly Logged Bitcoin Price from 4/8/2013 to 8/21/2017",
    col = "steelblue", lwd = 3)
```



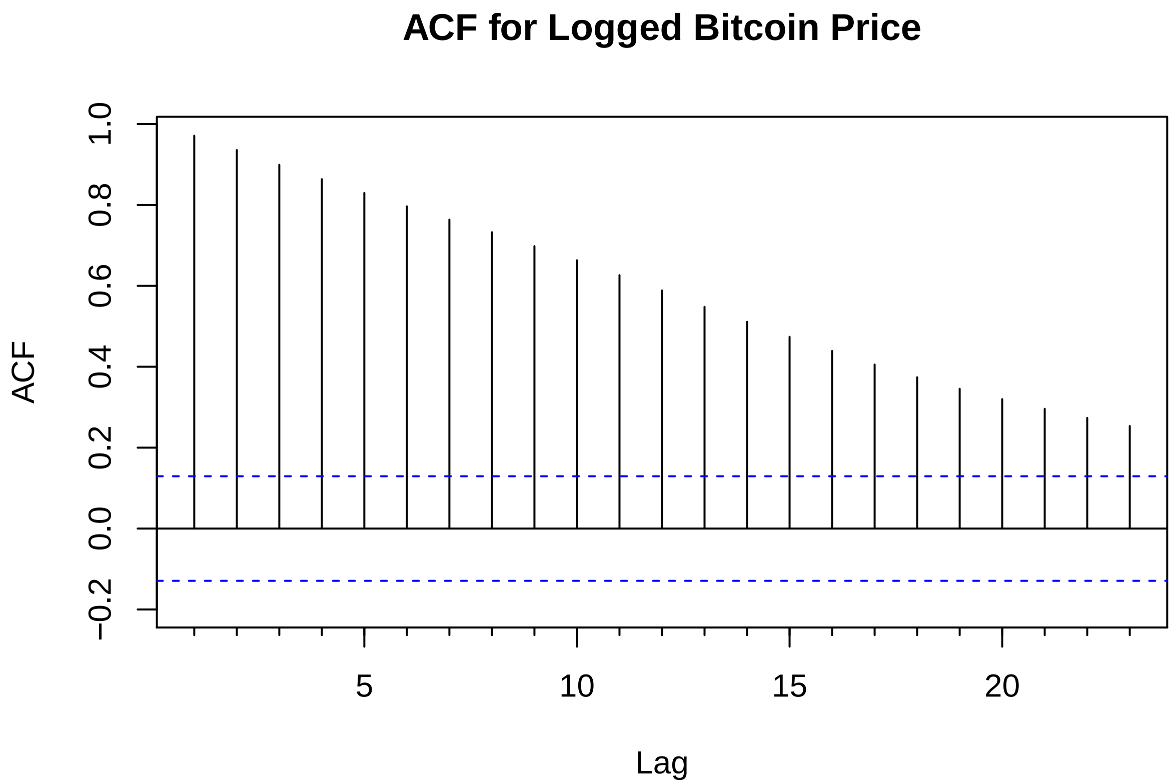

```r
# The augmented Dickey-Fuller test
adf.test(training$log_price, alternative = "stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  training$log_price
## Dickey-Fuller = -1.3664, Lag order = 6, p-value = 0.8421
## alternative hypothesis: stationary
```

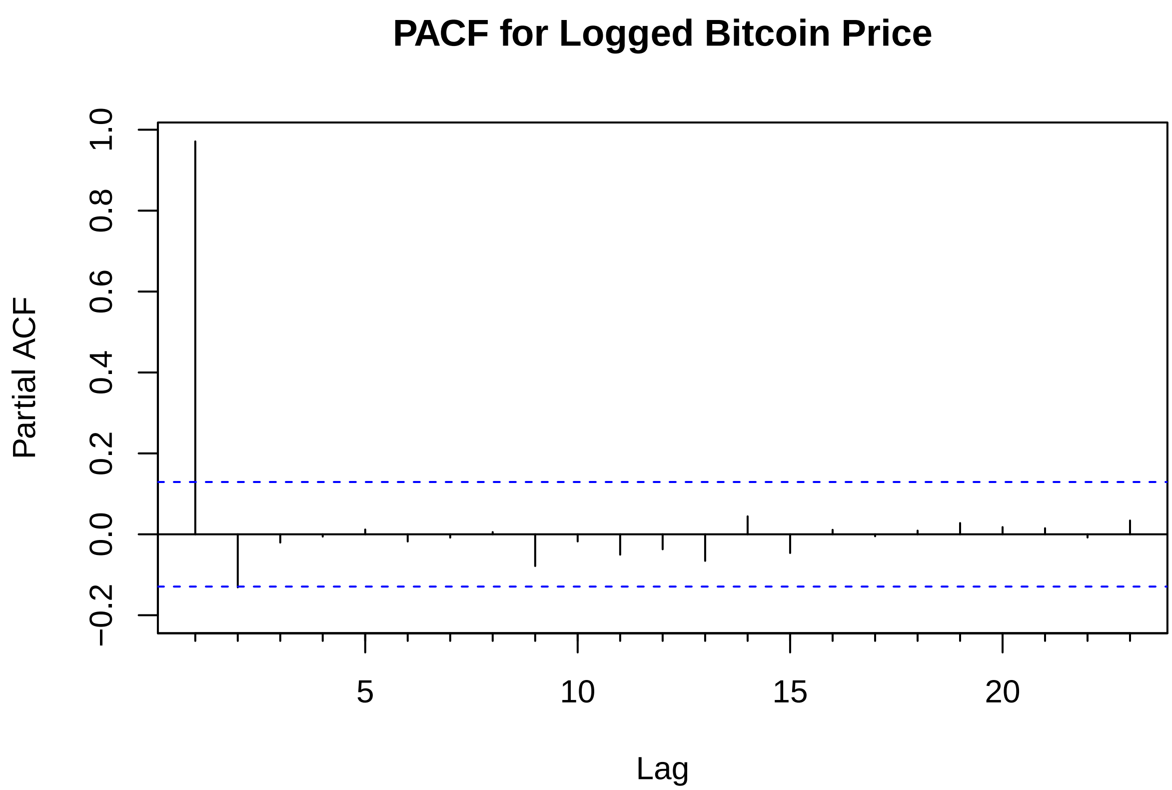
```r
# Fail to reject the series is non-stationary but p-value is
# lower
```

ACF nd PACF for logged Bitcoin Price

```r
Acf(training$log_price, main = "ACF for Logged Bitcoin Price")
```



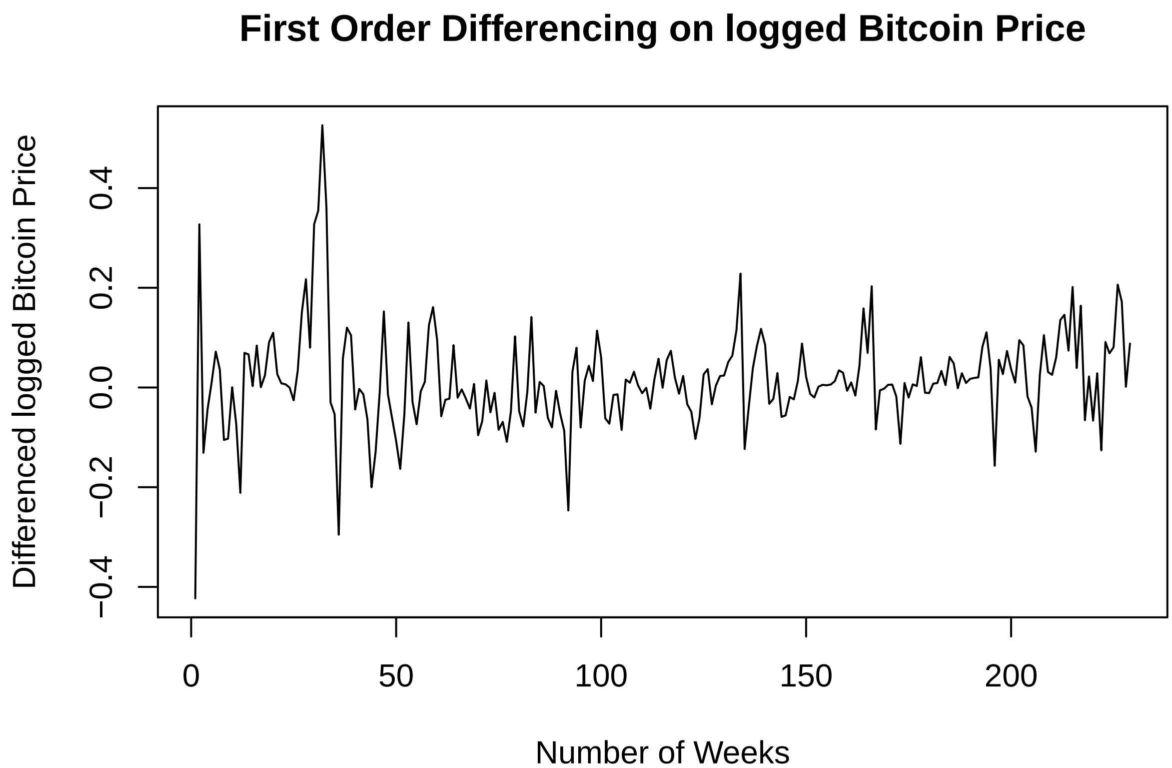

```r
# Significant autocorrelations with many lags caused by carry
# over from earlier lags
Pacf(training$log_price, main = "PACF for Logged Bitcoin Price")
```



```r
# A spike at 1 in PACF meaning lag 1 alone explained most of
# the information
```

First order differencing on logged Bitcoin Price

```r
diff_log_price <- diff(training$log_price)
plot(diff_log_price, type = "l", xlab = "Number of Weeks", ylab = "Differenced logged Bitcoin Price",
    main = "First Order Differencing on logged Bitcoin Price")
```




```r
adf.test(diff_log_price, alternative = "stationary")
```

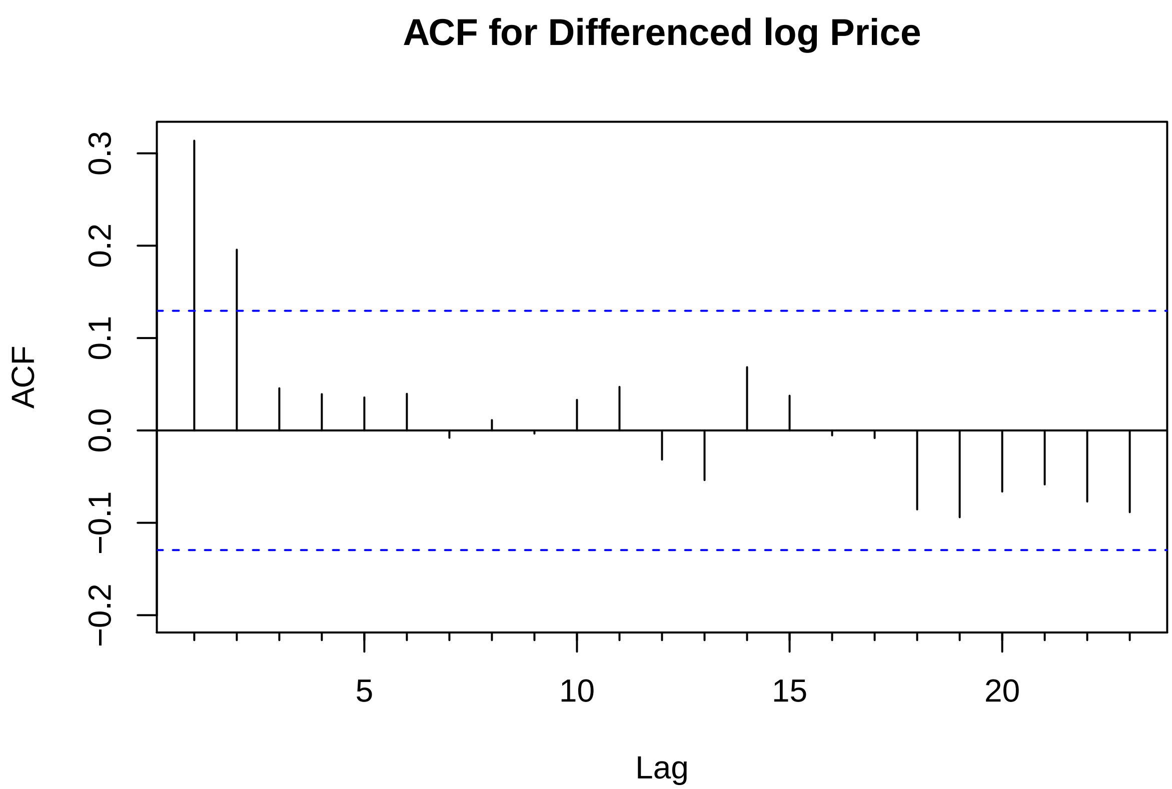
```
## Warning in adf.test(diff_log_price, alternative = "stationary"): p-value
## smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_log_price
## Dickey-Fuller = -4.9599, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

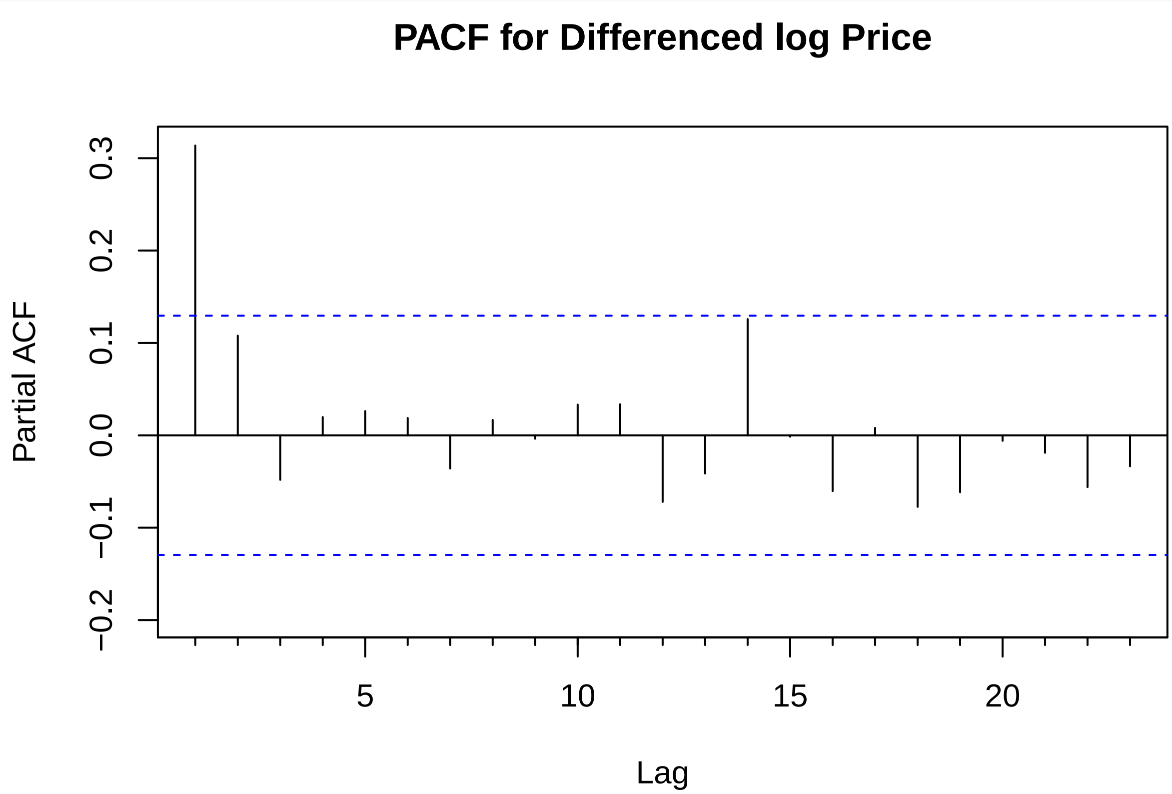
```r
# Reject H0 with low p-value. The series is stationary
```

ACF and PACF for Differenced log Bitcoin Price

```r
# ACF and PACF for differenced series to determine p and q
# for ARIMA model
Acf(diff_log_price, main = "ACF for Differenced log Price")
```



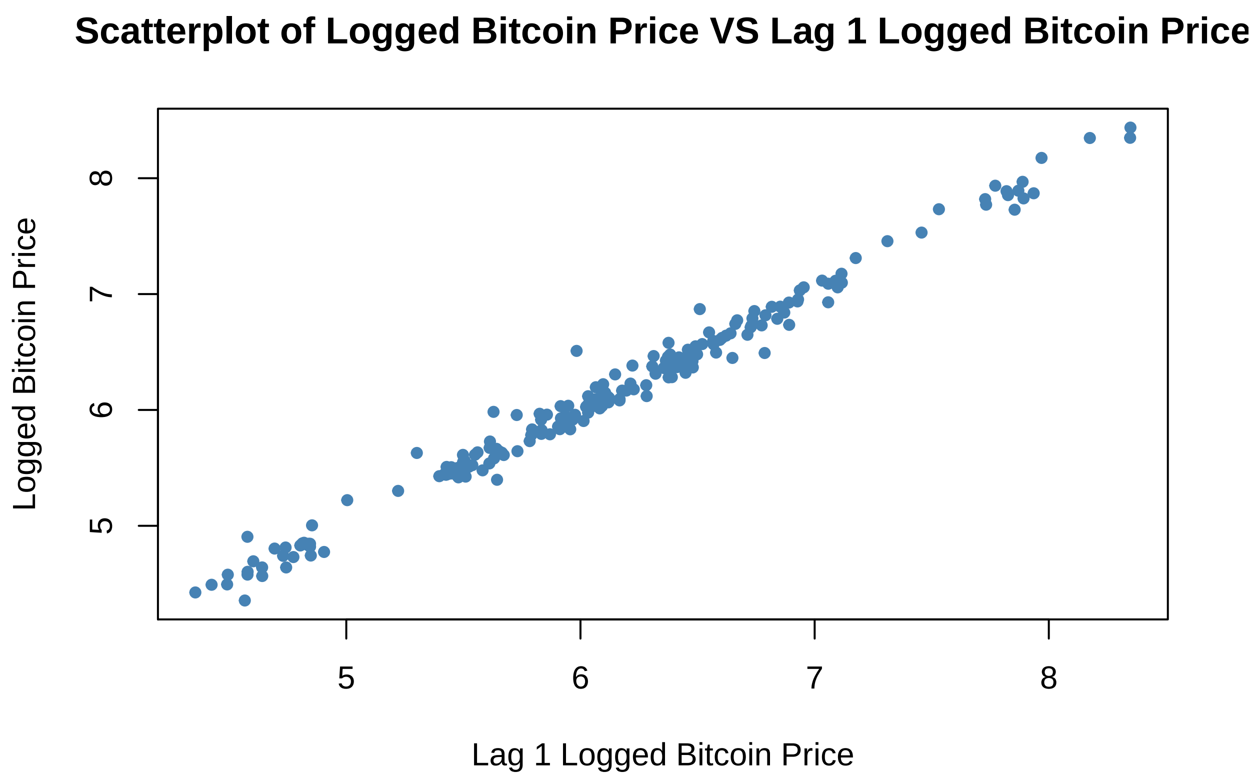

```r
# Significant spike at lag 1 and lag 2. Consider MA(2) model
Pacf(diff_log_price, main = "PACF for Differenced log Price")
```




```r
# Significant spike at lag 1 and close to significant for lag
# 2. Consider AR(1) or AR(2) model
```

Scatterplot of logged Bitcoin Price against its lags

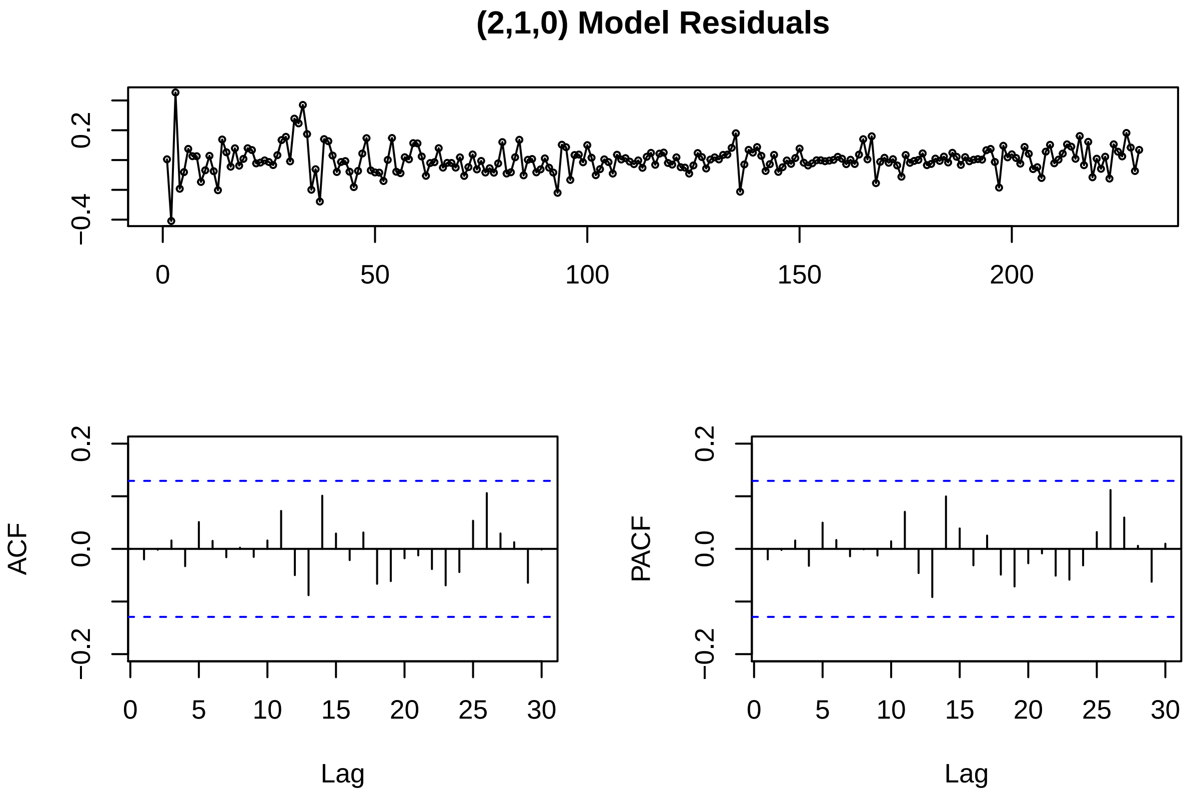
```
plot(x = lag(training$log_price[2:230]), y = training$log_price[2:230],
    ylab = "Logged Bitcoin Price", xlab = "Lag 1 Logged Bitcoin Price",
    pch = 19, cex = 0.7, col = "steelblue", main = "Scatterplot of Logged Bitcoin Price VS Lag 1 Logged
```



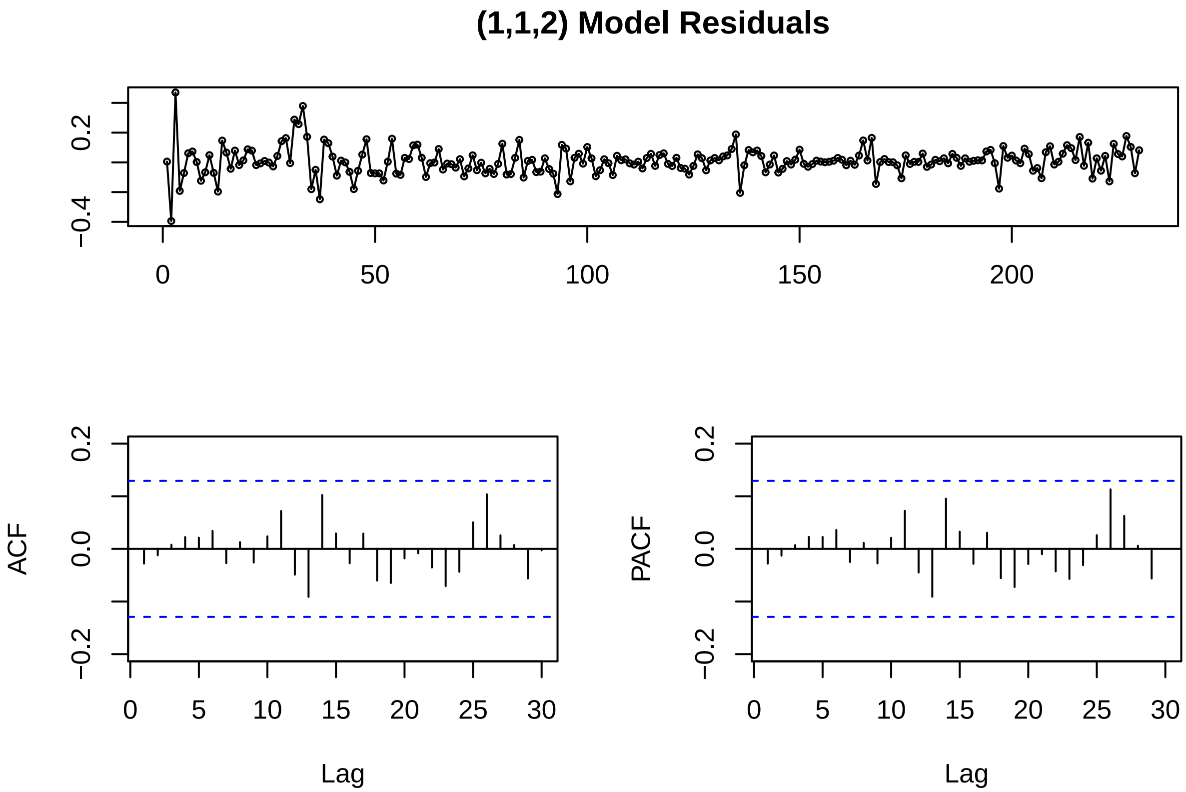

```
# Linear pattern suggesting the first order autoregressive
# model is appropriate.
```

ARIMA Model

```
## Series: training$log_price
## ARIMA(2,1,1) with drift
##
## Coefficients:
##             ar1     ar2     ma1   drift
##         -0.3461  0.3146  0.6548  0.0142
## s.e.     0.4411  0.1298  0.4607  0.0099
##
## sigma^2 estimated as 0.008916:  log likelihood=217.43
## AIC=-424.85   AICc=-424.59   BIC=-407.69
```

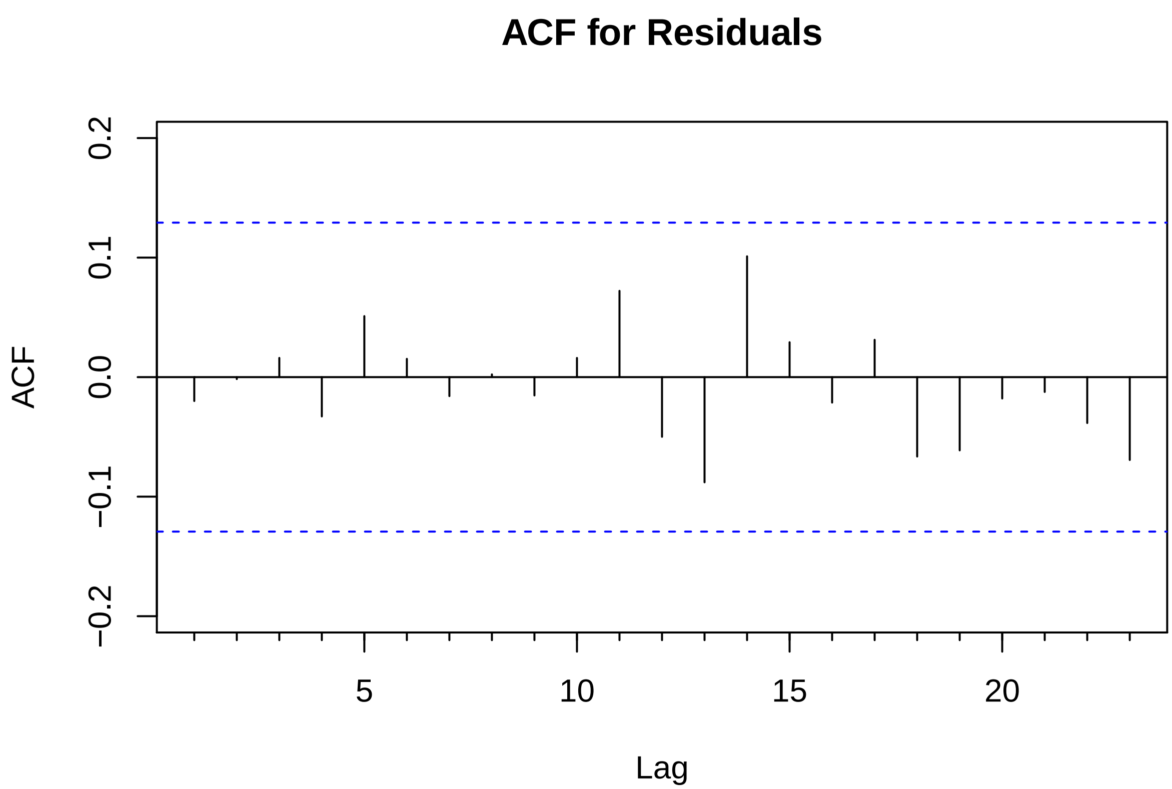



```
## Series: training$log_price
## ARIMA(1,1,2)
##
## Coefficients:
##          ar1     ma1     ma2
##       0.1657  0.1532  0.1692
## s.e.  0.4284  0.4225  0.1589
##
## sigma^2 estimated as 0.008973:  log likelihood=216.19
## AIC=-424.38   AICc=-424.2   BIC=-410.64
```

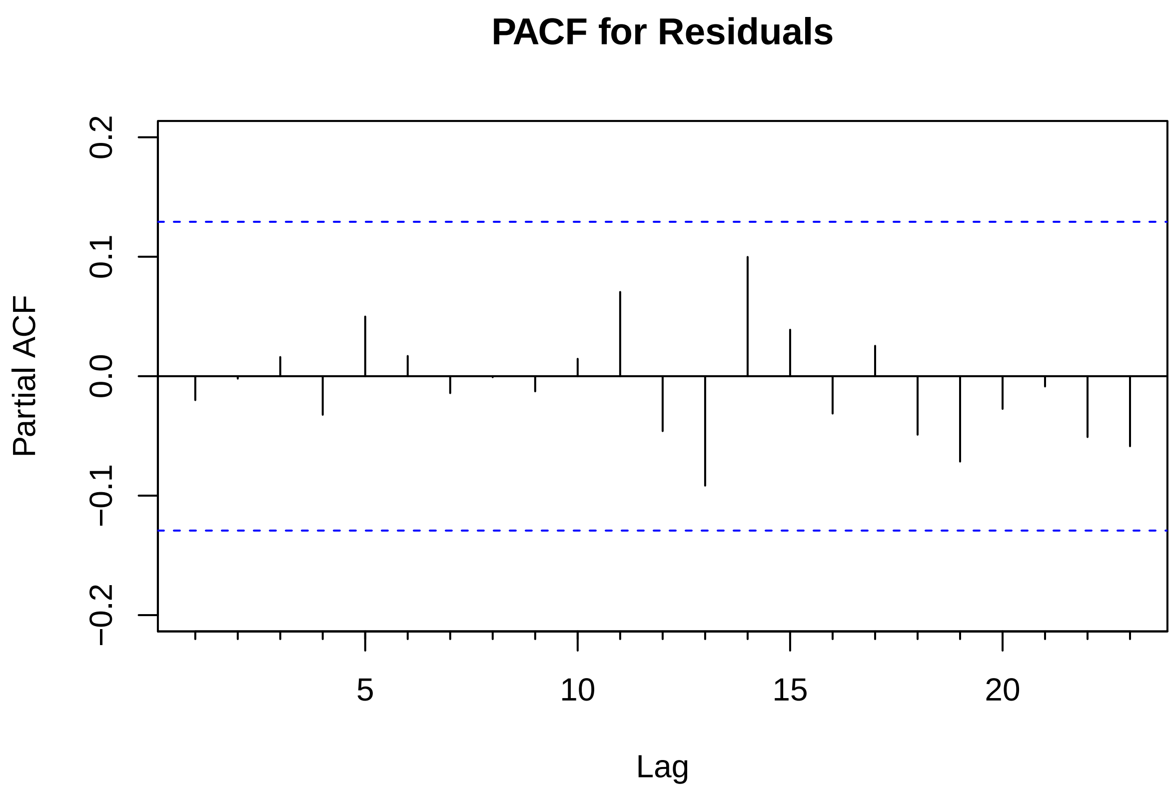


Check Residuals for model validity

```
res <- fit$residuals
Acf(res, main = "ACF for Residuals")
```




```r
Pacf(res, main = "PACF for Residuals")
```




```r
# Based on ACF and PACF, residuals are randomly distributed.

Box.test(res, type = "Ljung", lag = 1)
```

```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 0.093094, df = 1, p-value = 0.7603
```

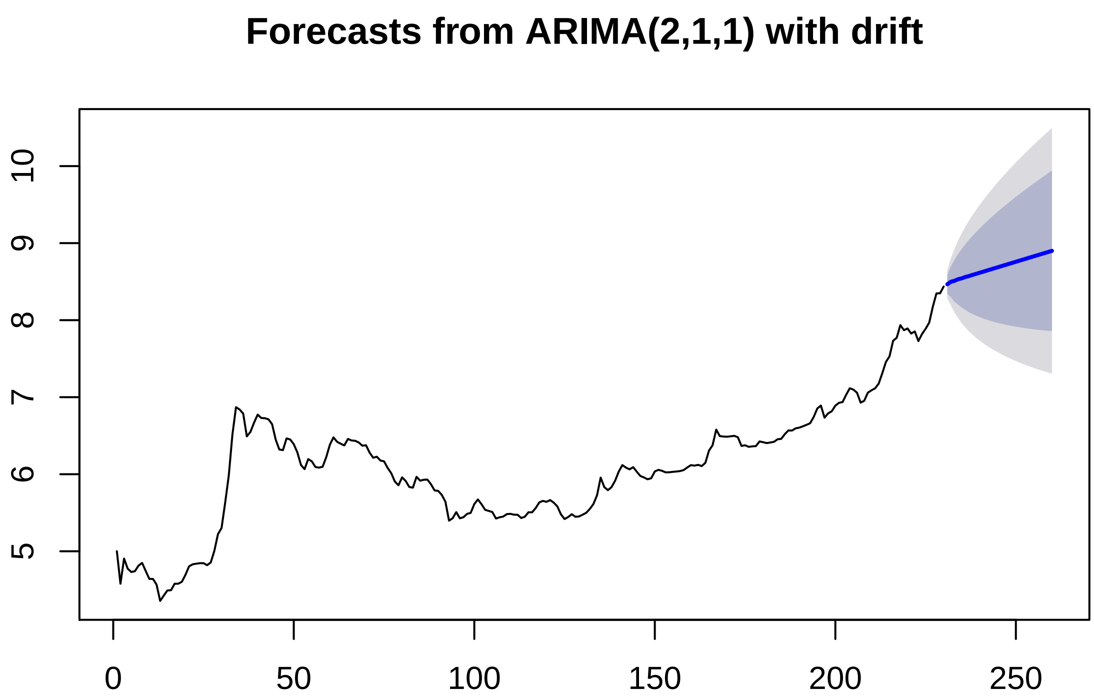
```r
# P-value is high so we fail to reject H0: The residuals are
# independent. Box-Ljung test confirms our results.
```

Antilog function to reverse log transformation

```r
# Anti log function
antilog <- function(lx, base) {
    lbx <- lx/log(exp(1), base = base)
    result <- exp(lbx)
    result
}
```

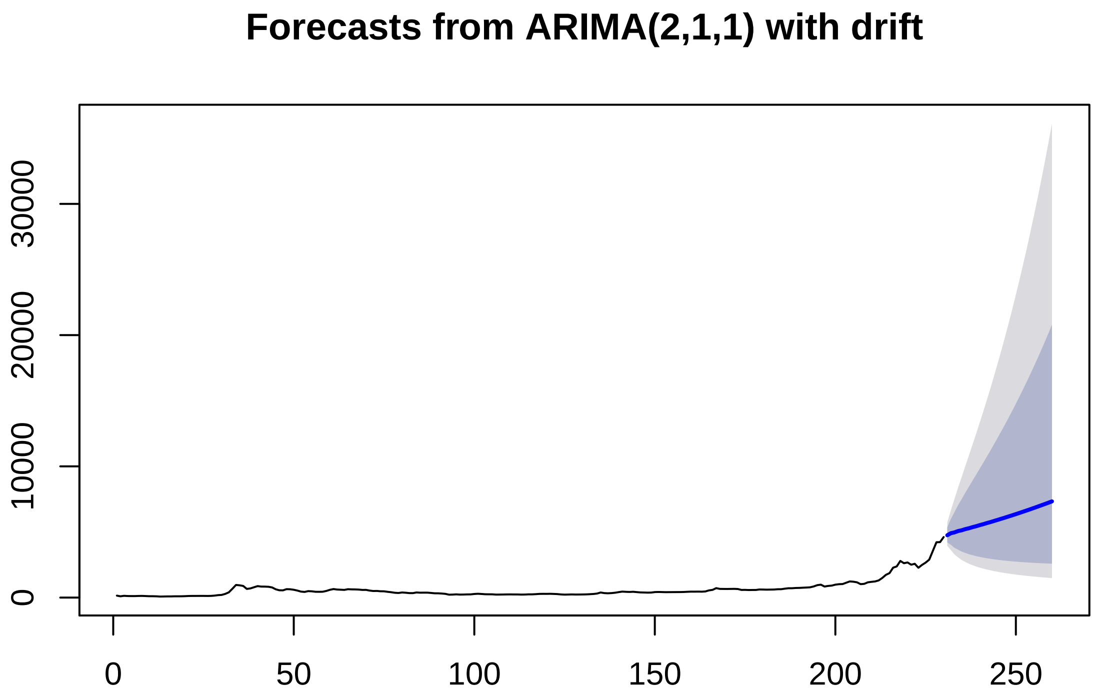
Forecast

```r
fcast <- forecast(fit, h = 30)
plot(fcast)
```




```r
# Take the exponent
fcast$mean <- antilog(fcast$mean)
fcast$lower <- antilog(fcast$lower)
fcast$upper <- antilog(fcast$upper)
fcast$x <- antilog(fcast$x)
```

```r
plot(fcast)
```



Compare prediction and actual bitcoin price for 30 weeks

```r
actual <- testing$bitcoin_price
pred <- fcast$mean[1:30]
error <- abs(actual - pred)/actual
prediction_table <- cbind(actual, pred, error)
colnames(prediction_table) <- c("Actual", "Prediction", "%Change")
prediction_table <- as.data.frame(prediction_table)
prediction_table$Week <- seq(1:30)
prediction_table
```

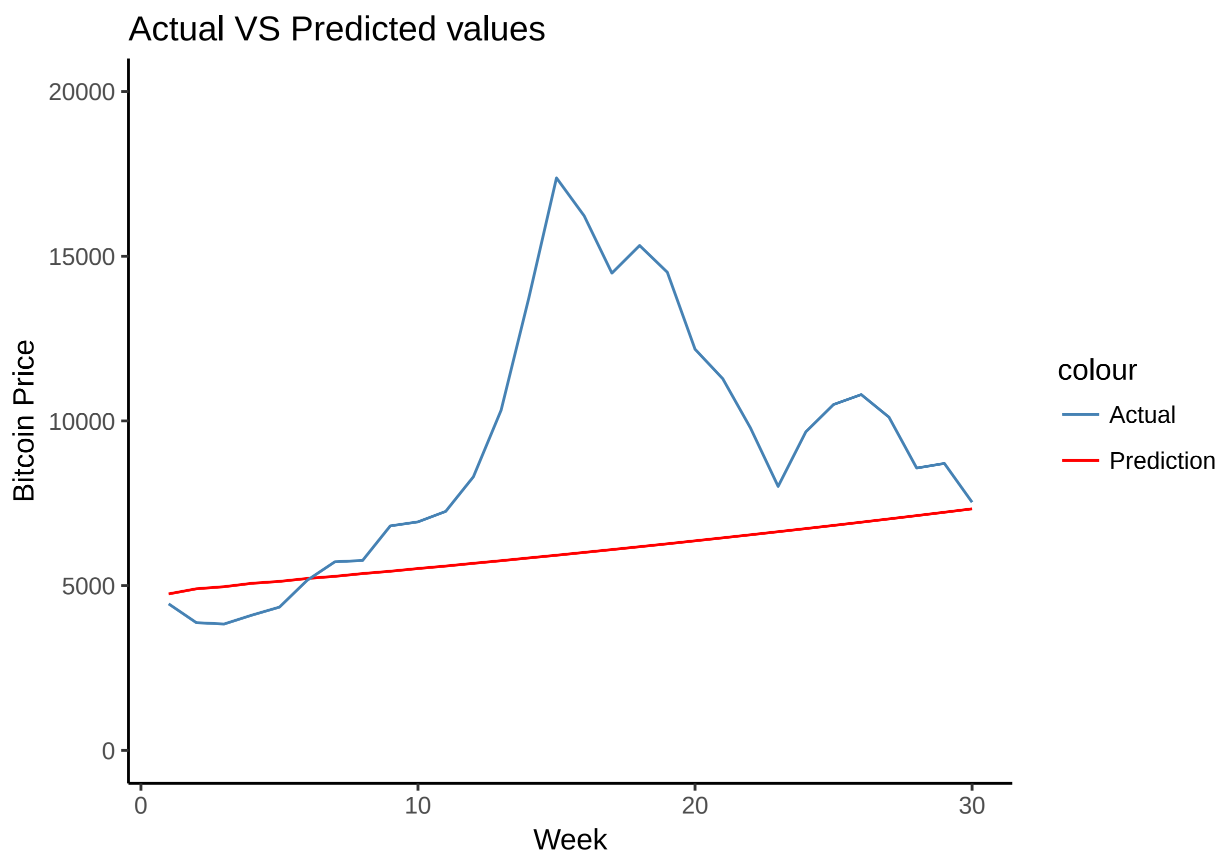
```
##       Actual Prediction    %Change Week
## 1   4448.186   4749.975 0.06784529    1
## 2   3877.361   4905.766 0.26523342    2
## 3   3836.227   4968.332 0.29510882    3
## 4   4104.402   5070.842 0.23546417    4
## 5   4349.547   5129.876 0.17940472    5
## 6   5163.845   5218.222 0.01053046    6
## 7   5724.994   5283.266 0.07715767    7
## 8   5764.160   5367.087 0.06888661    8
## 9   6814.830   5437.888 0.20205077    9
## 10  6936.828   5520.471 0.20417927   10
## 11  7254.876   5595.853 0.22867686   11
## 12  8305.702   5678.743 0.31628382   12
## 13 10326.294   5757.848 0.44240904   13
## 14 13732.345   5841.911 0.57458749   14
## 15 17376.360   5924.226 0.65906407   15
## 16 16223.990   6009.992 0.62956138   16
```

```
## 17 14485.893   6095.232 0.57922979   17
## 18 15325.235   6183.044 0.59654490   18
## 19 14512.732   6271.070 0.56789182   19
## 20 12177.336   6361.160 0.47762307   20
## 21 11282.490   6451.919 0.42814763   21
## 22  9788.082   6544.454 0.33138545   22
## 23  8014.081   6637.945 0.17171468   23
## 24  9670.893   6733.058 0.30378110   24
## 25 10498.607   6829.314 0.34950288   25
## 26 10800.558   6927.115 0.35863358   26
## 27 10114.764   7026.186 0.30535342   27
## 28  8570.413   7126.775 0.16844444   28
## 29  8711.170   7228.726 0.17017740   29
## 30  7532.565   7332.195 0.02660049   30
```

```r
mean(prediction_table$`%Change`)
```

```
## [1] 0.3097158
```

```r
ggplot() + geom_line(aes(x = Week, y = Prediction, color = "steelblue"),
    prediction_table) + geom_line(aes(x = Week, y = Actual, color = "red"),
    prediction_table) + ylim(0, 20000) + ylab("Bitcoin Price") +
    scale_color_manual(labels = c("Actual", "Prediction"), values = c("steelblue",
        "red")) + theme_classic() + ggtitle("Actual VS Predicted values")
```



Add Google Trend As a Predictor

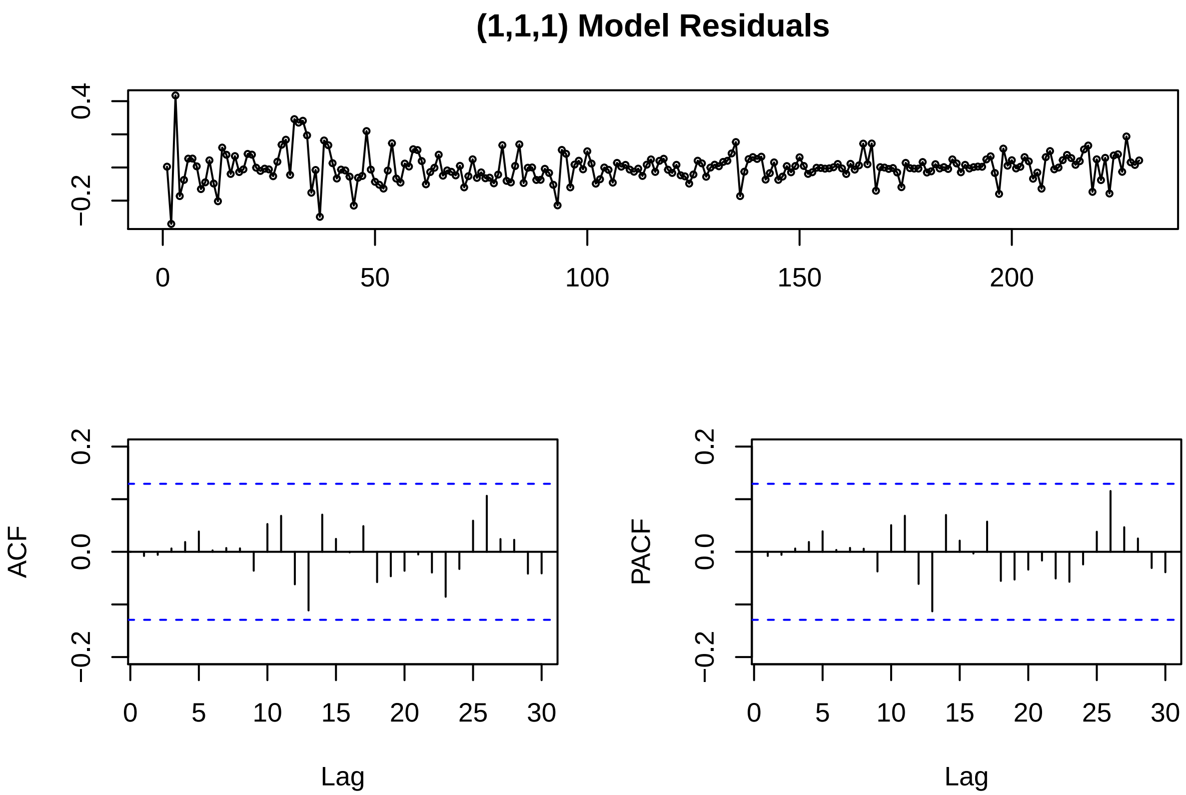
```
tsdata <- ts(training)
```

```
## Warning in data.matrix(data): NAs introduced by coercion

## Warning in data.matrix(data): NAs introduced by coercion
```

```
fit_trend <- auto.arima(training$log_price, xreg = training$google_trend)
fit_trend  #AIC=-405.75   AICc=-405.52   BIC=-387.97
```

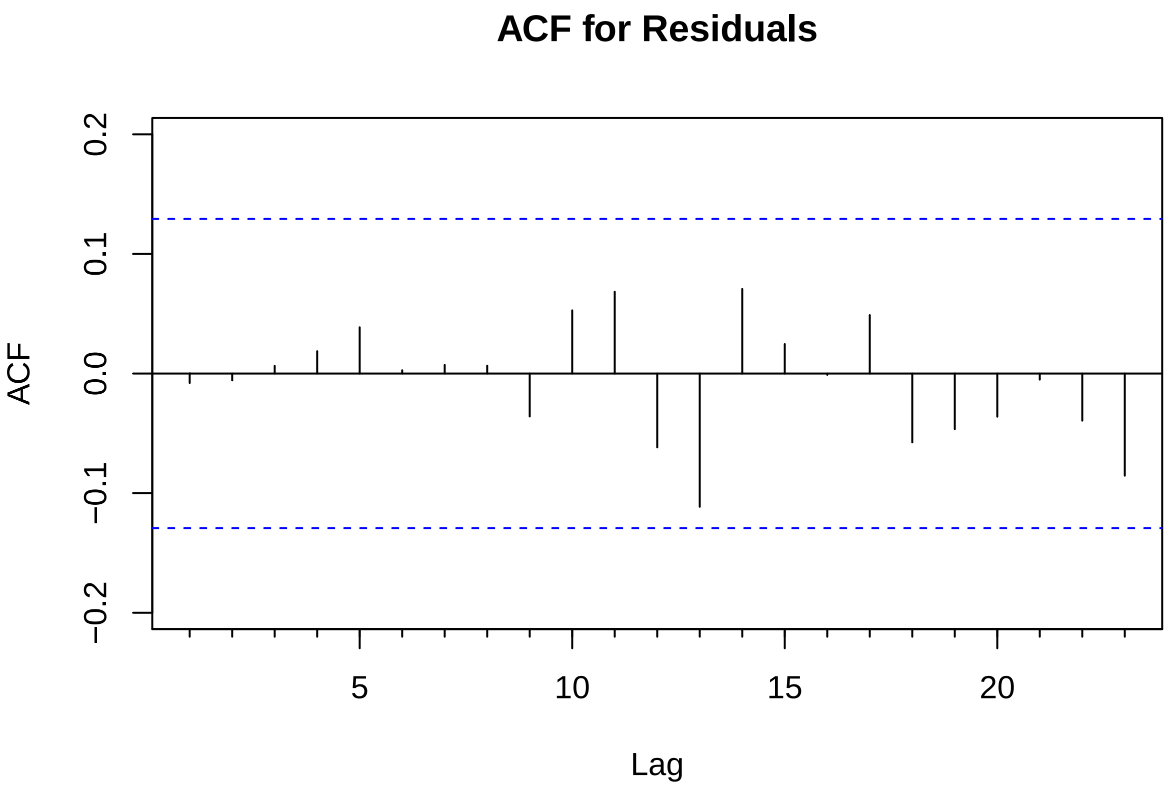
```
## Series: training$log_price
## Regression with ARIMA(2,1,1) errors
##
## Coefficients:
##           ar1     ar2     ma1   drift    xreg
##       -0.0393  0.1705  0.2730  0.0142  0.0166
## s.e.   0.7104  0.1895  0.7203  0.0088  0.0046
##
## sigma^2 estimated as 0.008456:  log likelihood=224.04
## AIC=-436.08   AICc=-435.7   BIC=-415.48
```

```
# Lower AIC, AICc and BIC compare with model without google
# trend

tsdisplay(residuals(fit_trend), lag.max = 30, main = "(1,1,1) Model Residuals")
```

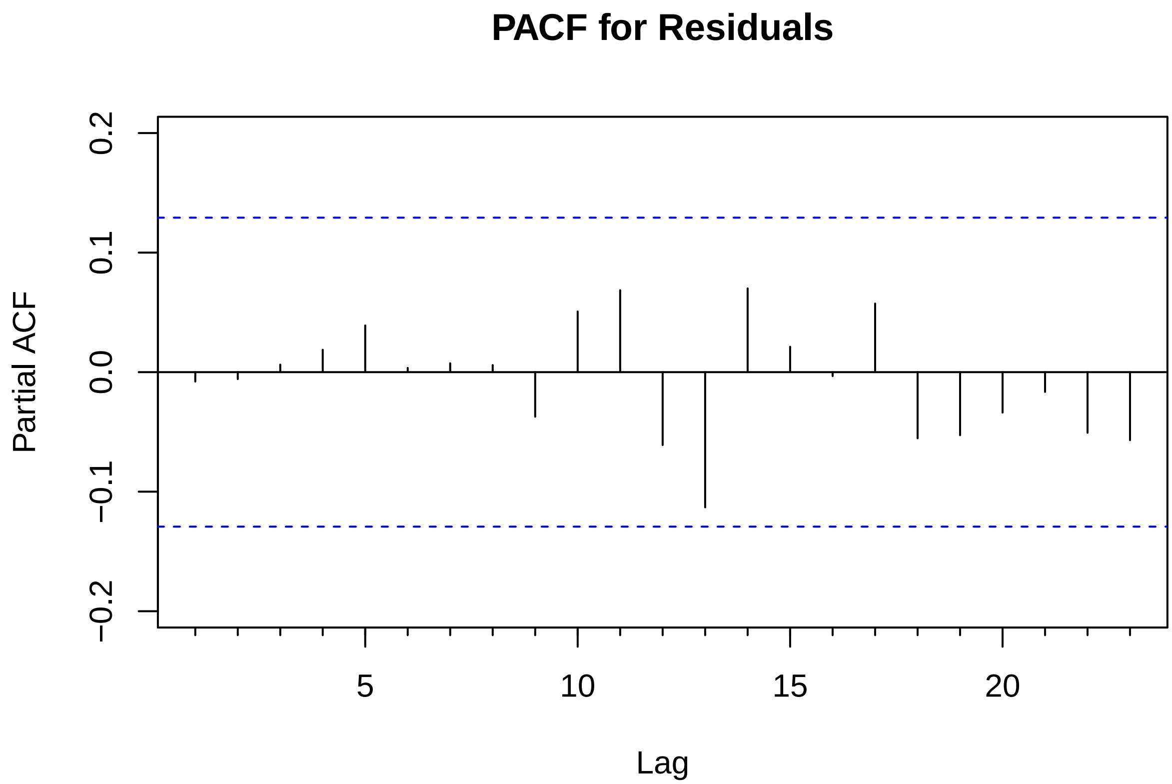


Check Residuals for model validity

```r
res1 <- fit_trend$residuals
Acf(res1, main = "ACF for Residuals")
```




```r
Pacf(res1, main = "PACF for Residuals")
```




```
# Based on ACF and PACF, residuals are randomly distributed.

Box.test(res1, type = "Ljung", lag = 1)
```

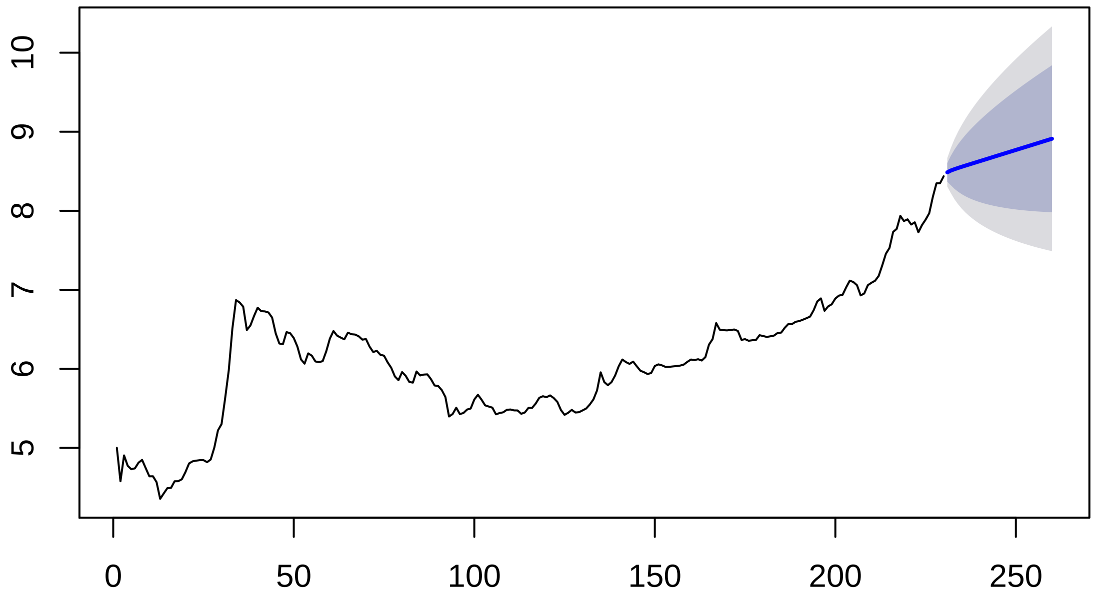
```
##
##  Box-Ljung test
##
## data:  res1
## X-squared = 0.014416, df = 1, p-value = 0.9044
```

```
# P-value is high so we fail to reject H0: The residuals are
# independent. Box-Ljung test confirms our results.
```

Google Trend ARIMA Model Forcast

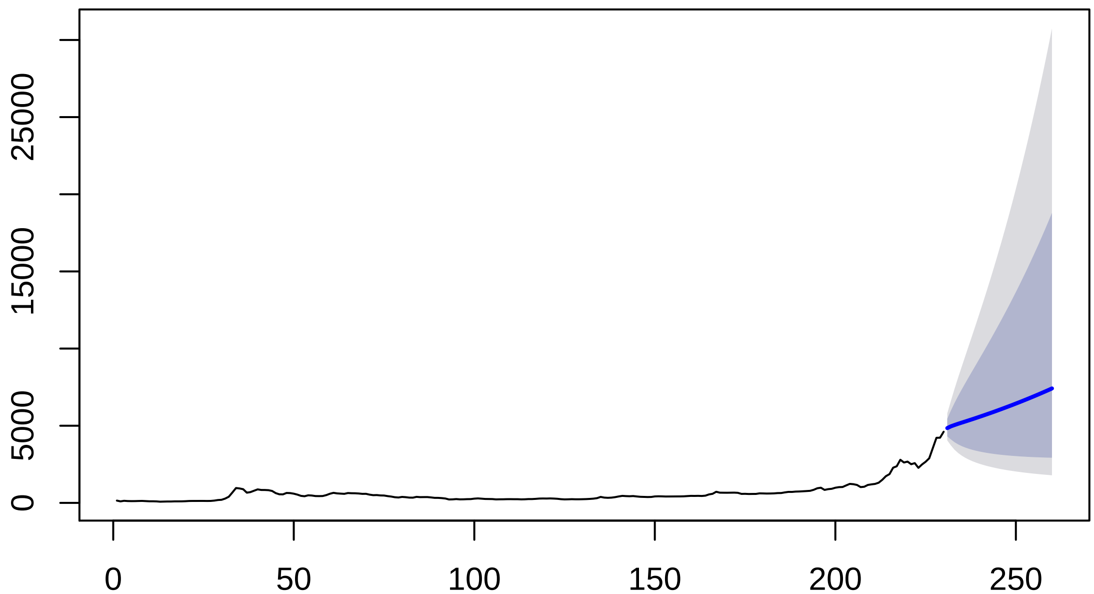
```
# Forecast
tsdata_trend <- ts(testing$google_trend)

fc.c2 <- forecast(tsdata_trend, h = 30)

newxreg <- as.matrix(fc.c2$mean)

fcast_trend <- forecast(fit_trend, xreg = newxreg)

plot(fcast_trend)
```

Forecasts from Regression with ARIMA(2,1,1) errors




```r
# Take the exponent
fcast_trend$mean <- antilog(fcast_trend$mean)
fcast_trend$lower <- antilog(fcast_trend$lower)
fcast_trend$upper <- antilog(fcast_trend$upper)
fcast_trend$x <- antilog(fcast_trend$x)


plot(fcast_trend)
```

Forecasts from Regression with ARIMA(2,1,1) errors



Compare prediction and actual bitcoin price for 30 weeks

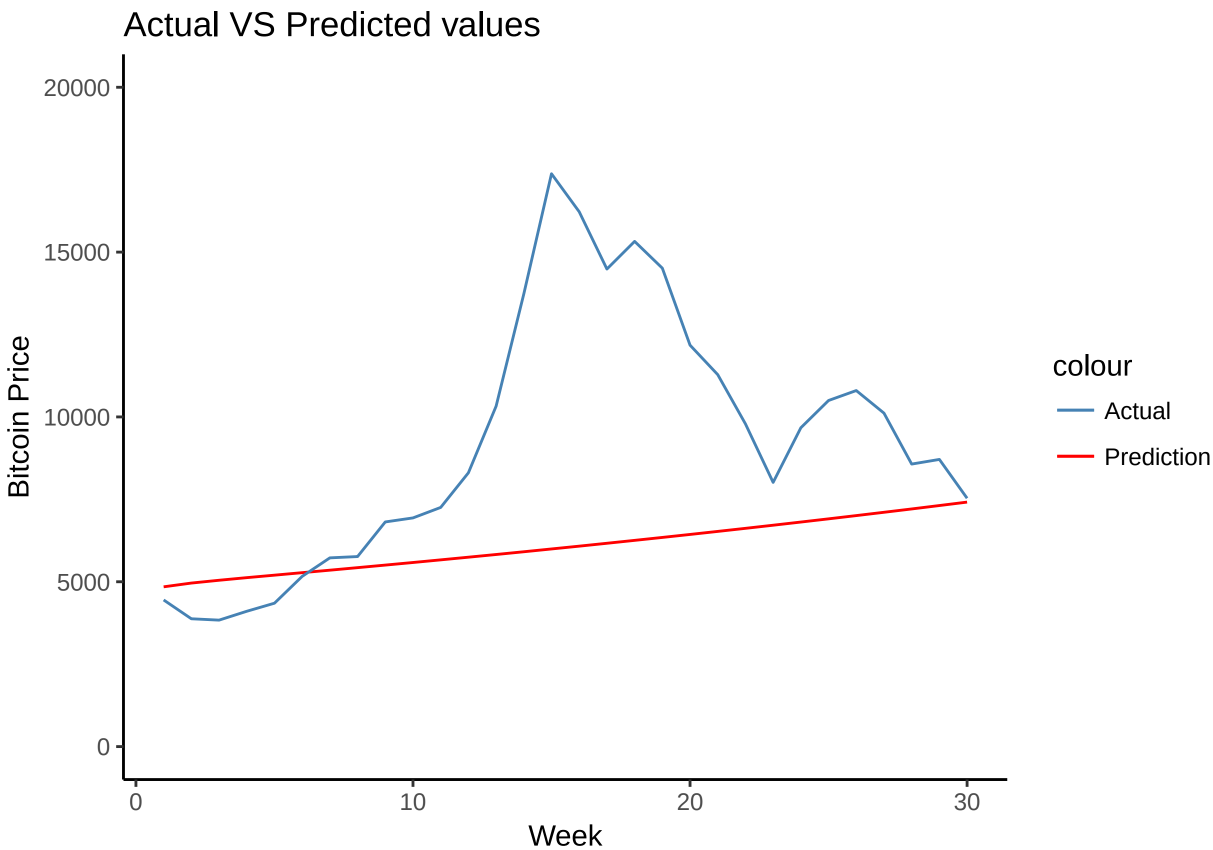
```
actual <- testing$bitcoin_price
pred1 <- fcast_trend$mean[1:30]
error1 <- abs(actual - pred1)/actual
prediction_table1 <- cbind(actual, pred1, error1)
colnames(prediction_table1) <- c("Actual", "Prediction", "%Change")
prediction_table1 <- as.data.frame(prediction_table1)
prediction_table1$Week <- seq(1:30)
prediction_table1
```

```
##       Actual Prediction    %Change Week
## 1   4448.186   4846.434 0.08953034    1
## 2   3877.361   4960.340 0.27930843    2
## 3   3836.227   5045.311 0.31517517    3
## 4   4104.402   5124.725 0.24859233    4
## 5   4349.547   5200.129 0.19555662    5
## 6   5163.845   5275.623 0.02164628    6
## 7   5724.994   5351.330 0.06526878    7
## 8   5764.160   5427.980 0.05832249    8
## 9   6814.830   5505.579 0.19211786    9
## 10  6936.828   5584.268 0.19498243   10
## 11  7254.876   5664.057 0.21927582   11
## 12  8305.702   5744.983 0.30830857   12
## 13 10326.294   5827.061 0.43570647   13
## 14 13732.345   5910.311 0.56960655   14
## 15 17376.360   5994.750 0.65500541   15
## 16 16223.990   6080.396 0.62522191   16
## 17 14485.893   6167.265 0.57425721   17
## 18 15325.235   6255.375 0.59182521   18
## 19 14512.732   6344.743 0.56281538   19
## 20 12177.336   6435.389 0.47152739   20
## 21 11282.490   6527.329 0.42146376   21
## 22  9788.082   6620.583 0.32360768   22
## 23  8014.081   6715.170 0.16207860   23
## 24  9670.893   6811.107 0.29571060   24
## 25 10498.607   6908.416 0.34196836   25
## 26 10800.558   7007.114 0.35122665   26
## 27 10114.764   7107.223 0.29734170   27
## 28  8570.413   7208.761 0.15887821   28
## 29  8711.170   7311.751 0.16064654   29
## 30  7532.565   7416.212 0.01544676   30
```

```
mean(prediction_table1$`%Change`)
```

```
## [1] 0.3067473
```

```
ggplot() + geom_line(aes(x = Week, y = Prediction, color = "steelblue"),
    prediction_table1) + geom_line(aes(x = Week, y = Actual,
    color = "red"), prediction_table1) + ylim(0, 20000) + ylab("Bitcoin Price") +
    scale_color_manual(labels = c("Actual", "Prediction"), values = c("steelblue",
        "red")) + theme_classic() + ggtitle("Actual VS Predicted values")
```



Accuracy of all ARIMA Model

```
# Accuracy
accuracy(prediction_table$Actual, prediction_table$Prediction)
```

```
##                 ME     RMSE      MAE       MPE     MAPE
## Test set -3183.876 4757.224 3468.105 -50.97236 56.66672
```

```
accuracy(prediction_table1$Actual, prediction_table1$Prediction)
```

```
##                 ME    RMSE      MAE      MPE     MAPE
## Test set -3111.502 4708.59 3423.035 -49.1853 55.34531
```