Final Project

Mai Castellano, Shannon Long, Zahlen Zbinden

2024-03-09

Introduction

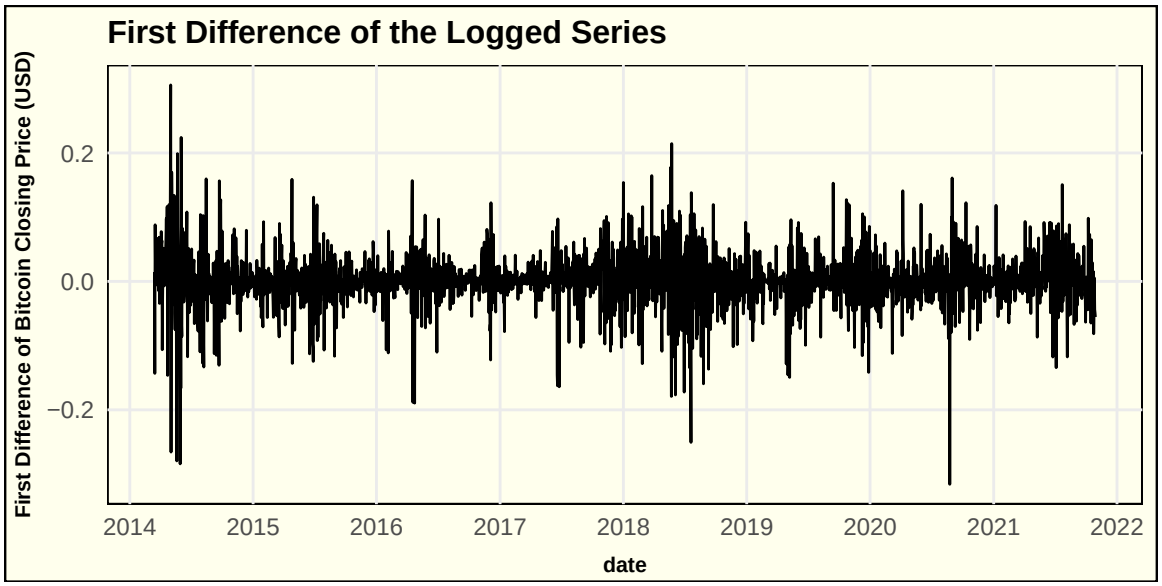
Bitcoin is the first and most recognized decentralized cryptocurrency. Due to the decentralized nature and lower risks, this currency has gained immense popularity. Forecasting patterns of this stock is a skill that could yield substantial rewards. Time series analysis utilizing autoregressive integrated moving average (ARIMA) or Box - Jenkins modeling is a powerful tool to understand the complex nature of historic Bitcoin data to make future predictions. The goal of this study is to predict future closing exchange prices for Bitcoin using ARIMA Time Series modeling. It should be noted that since the data is daily, our model will only be relevant to predicting a few days into the future. When forecasted out to long the model will return to an average value as expected.

Methods

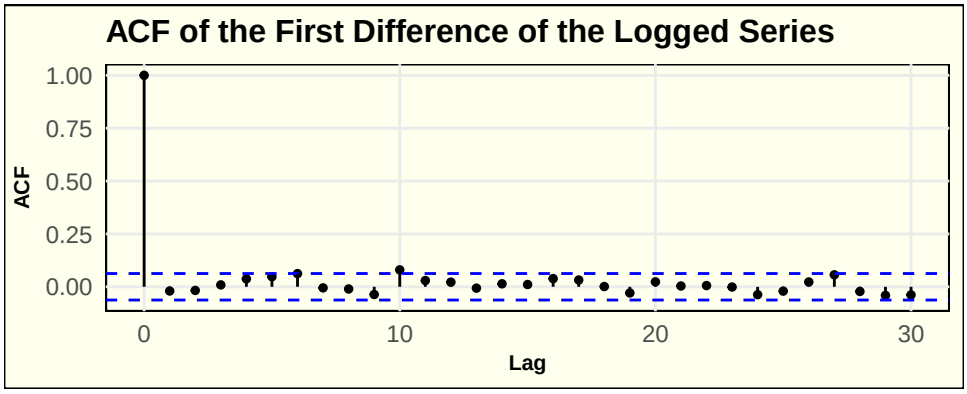
Daily Bitcoin activity data was obtained, including information on daily closing price, open price, high price, and low price. Comparing these features, it became evident that all attributes are very similar. This suggests that only one feature was needed for consideration, as they are all likely to produce comparable results. Therefore, the daily closing price was selected due to its traditional significance in financial analysis and its reflection of market sentiment at the close of each trading day.

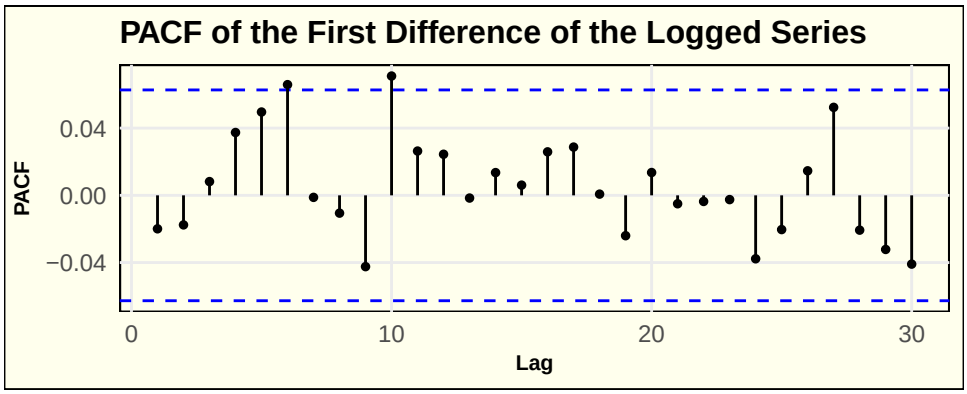
The daily Bitcoin time series is plotted against the closing price, demonstrating a non-stationary graph. Thus, both logarithmic and square root transformations were considered to stabilize the time series. Initially, both transformations helped address the issue of constant variance over time. However, the first-order differencing of the log-transformed series was more effective in de-trending the time series. Therefore, the log transformation was chosen for further analysis. Given the absence of seasonality, further differencing was deemed unnecessary. The first difference of the

log transformed series is shown below, indicating that the trend has been removed, and much more closely aligns with a stationary time series.



After removing the trend, the autocorrelation function (ACF) and partial autocorrelation function (PACF) were examined for the first difference of the logged series.





The ACF plot did not reveal any significant lags, with all lags falling within the confidence interval bounds. Moreover, no obvious pattern was observed in the ACF plot. However, the PACF showed significance values at lag 6 and 10, suggesting that a higher-order ARIMA(6,1,0) model with only Auto Regressive features should be considered.

Additionally, the forecast package in R was employed to explore other potential models. The autoarima function suggested an ARIMA(4,0,1) model, which, when considering the differenced time series it was built on, actually translated to an ARIMA(4,1,1) model. The function confirmed that there was no need to incorporate seasonal components, or further differencing..

As a result, three different models were considered: a higher-order autoregressive model, the suggested ARIMA model, and the suggested ARIMA model with an additional moving average order: ARIMA(6,1,0), ARIMA(4,1,1), ARIMA(4,1,2).

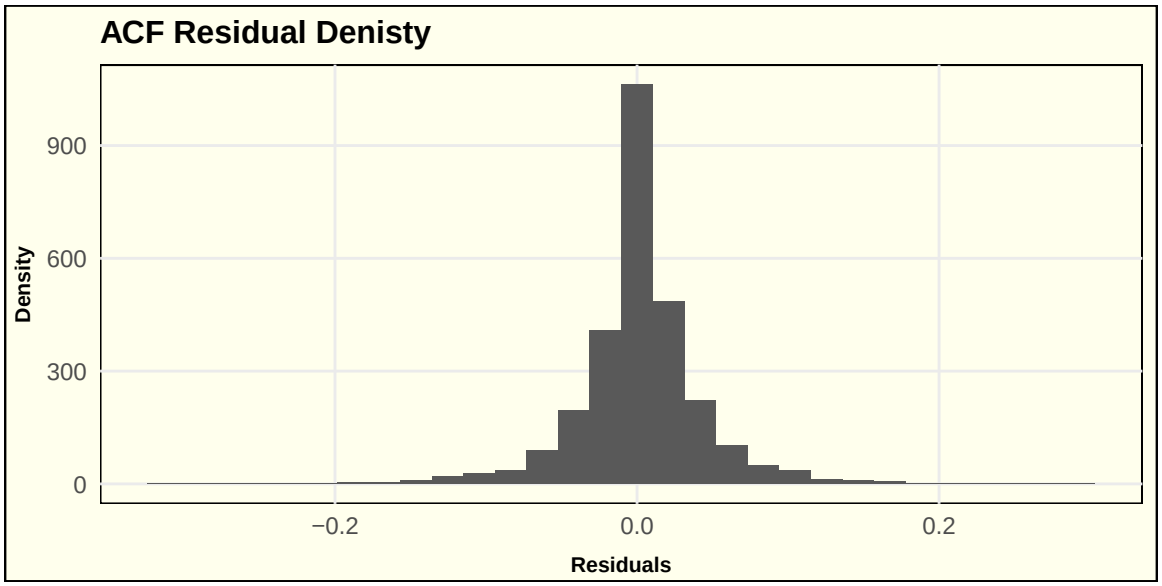
Here we can see the AIC values for the 3 chosen models.

Model Fitting AIC Results

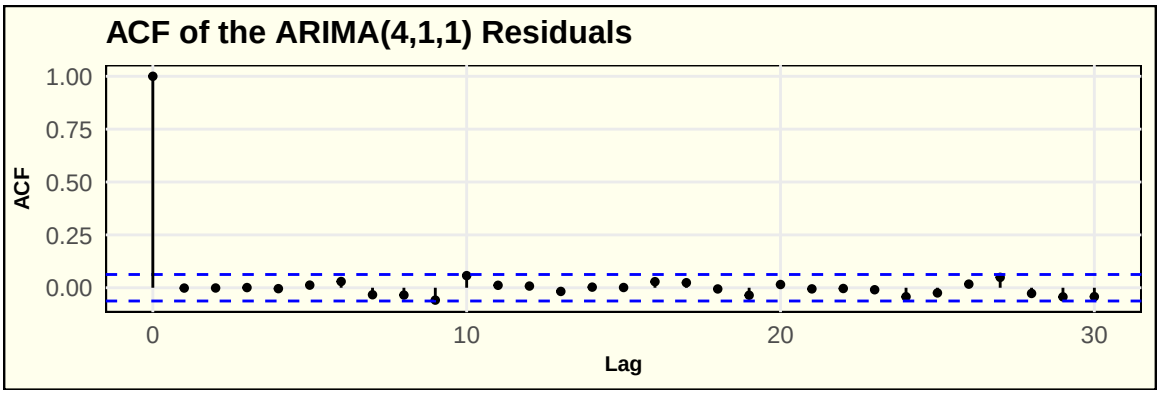
Model	AIC
ARIMA(6,1,0)	-9679.165
ARIMA(4,1,1)	-9677.434
ARIMA(4,1,2)	-9674.832

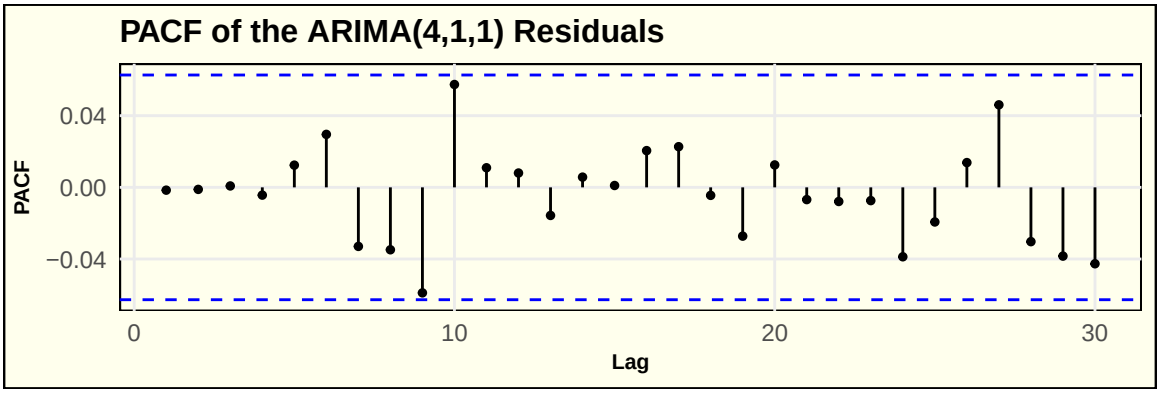
Upon comparing the AIC, it was noted that all three models exhibited very similar values. Although the ARIMA(6,1,0) model had the lowest AIC value among the three, the ARIMA(4,1,1) model with a slightly higher AIC value was chosen for prediction due to its simplicity, thereby avoiding potential overfitting.

A key indicator that a model is well fitted is inspecting the residuals. If the residuals are not normally distributed then the model is overestimating or underestimating. We want to see a normal distribution in the residuals to ensure that the model is correctly capturing the variance in the data. Below we can see that the residuals are normally distributed.



Both the ACF and PACF of the residuals resembled white noise, with all lags falling within the confidence intervals bounds, suggesting that the model adequately captured the trend and fluctuations in the data.

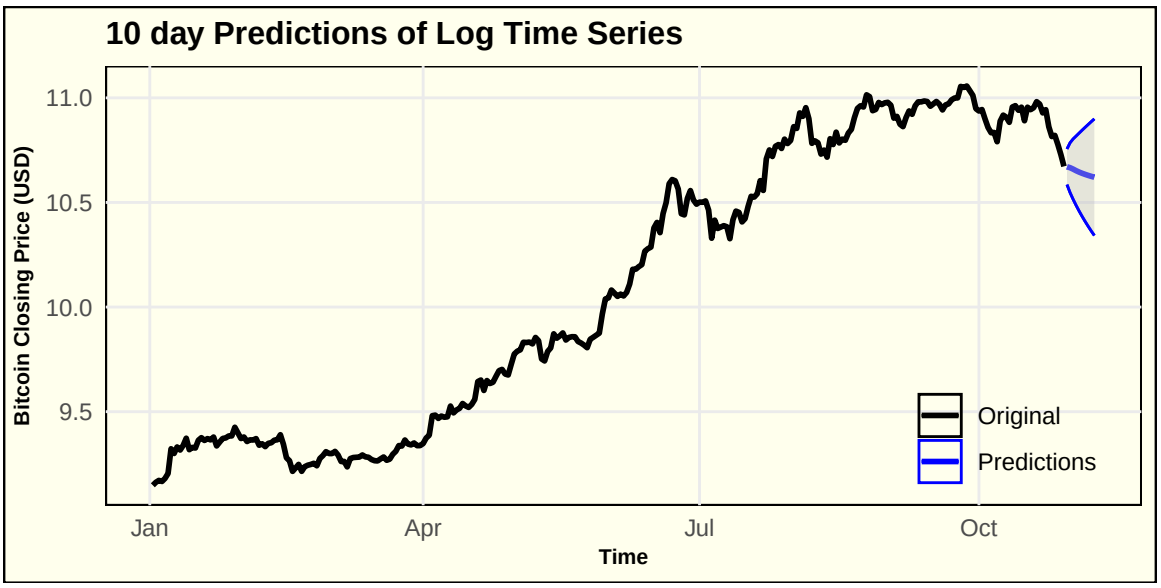




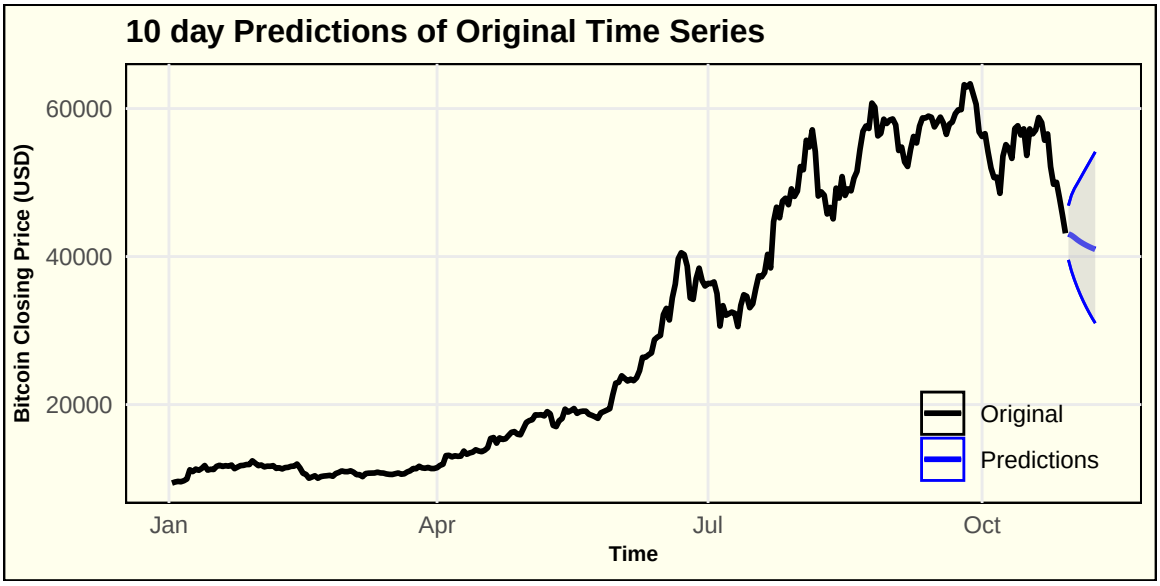
Overall, the chosen model was deemed appropriate for prediction purposes.

Results

Using the fitted ARIMA(4,1,1) model, we forecasted Bitcoin's log transformed closing exchange price for 10 days using the `predict` function.



The above figure shows that the Bitcoin log closing exchange prices will decrease in the next 10 day period. The 95% confidence intervals for the predicted values are shaded in gray.



The data was exponentially transformed to acquire the predicted closing exchange prices of the raw, original data. The figure above shows that over a 10 day period the closing price of Bitcoin with decrease from $43,145.35 to $40,973.47 between October 29, 2021 to November 8, 2021. The 95% confidence region for this prediction is very wide, ranging from $31,006.68 to $54,143.61 predicted closing prices on November 8, 2021.

Discussion

Many methods were employed during this analytical dive into the world of bitcoin. We transformed the time series to stablize the variance and the trend. We compared multiple ARIMA models and evaluated the AIC and complexity of each model to decide on our final model, ARIMA(4,1,1). This model has a higher order autoregressive part, a lower order moving average, and one difference of the time series.

Due to the natural volatility of Bitcoin and stock prices in general it is hard to capture all of the variance of a data set and correctly predict future values. This is enhanced by the fact that Bitcoin prices are influenced by human nature, news, the job market, and social hype. All of these factors can't be taken into account when building an ARIMA model, this is why the 95% confidence interval on our predictions is so wide, and we should only use this model to predict out a few day into the future.

If this model were to be used it would need to be updated constantly after employing it, using the predicted value and the actual value of the previous day to enhance our next prediction, in this manner we could use the model to more accurately predict values a few days in advance.

Overall time series data that contain many fluctuations, data points, and is highly volatile is hard to predict. Through this process we have learned more about data and how ARIMA methods can be utlized to build models to predict future values.

Appendix

```
#load libraries
library(ggplot2)
library(tidyverse)
library(forecast)
library(lubridate)
library(RColorBrewer)
library(gt)
# set modifiable common theme for all plots
com_theme <- function(...) {
  theme(
    plot.title=element_text(size=12, face="bold"),
    axis.title=element_text(size=8, face="bold"),
    axis.title.x=element_text(vjust=-1),
    axis.title.y=element_text(vjust=2),
    panel.grid.minor=element_blank(),
    panel.background=element_rect(fill="#FFFFED"),
    plot.background=element_rect(fill="#FFFFED")
  ) +
  theme(...)
}
# Load the data
#df <- read.csv("/Users/mai/OSU/ST566 - Time Series/Final_Project/Data/BTC.csv")
df <- read.csv("D:/RepoMan/osu/data/BTC_USD_Price_Prediction_Data.csv")
# Convert to datetime object, change name formating, and remove uneeded currency column
df <- df |>
  mutate(date = as.Date(Date, format="%Y-%m-%d"))  |>
  rename(
    close="Closing.Price..USD.",
    open_24h="X24h.Open..USD.",
    high_24h="X24h.High..USD.",
    low_24h="X24h.Low..USD.",
  ) |>
  select(-Currency)
#select only closing price and Date
df <- df |>
        select(date, close)
# Extract start date and end date from the dataframe
start_date <- min(df$date)
end_date <- max(df$date)
# Create differenced data frame for plotting
log_dif_df <- diff(log(df$close)) |>
            as_tibble() |>
            mutate(
              date=seq(
                start_date + days(1),
                end_date,
                by="days"
              )
            ) |>
            rename(close="value")
```

```r
# plot of closing price over time
df |>
  ggplot(aes(x=date, y=close)) +
    geom_line() +
    labs(
      title="Bitcoin Closing Price Over Time",
      x="Date",
      y="Closing Price (USD)"
    ) +
    scale_x_date(date_labels="%Y", date_breaks="1 year") +
    theme_minimal() +
    com_theme()
```

```r
#Plot time series of log closing price
df |>
  mutate(close=log(close)) |>
  ggplot(aes(x=date, y=close)) +
    geom_line() +
    labs(
      title="Log Transformation of the Closing Price"
    ) +
    scale_x_date(date_labels="%Y", date_breaks="1 year") +
    theme_minimal() +
    com_theme()
```

```r
# Plot First difference of the logged series
log_dif_df |>
  ggplot(aes(x=date, y=close)) +
    geom_line() +
    labs(
      title="First Difference of the Logged Series",
      y = 'Bitcoin Closing Price (USD)'
    ) +
    scale_x_date(date_labels="%Y", date_breaks="1 year") +
    theme_minimal() +
    com_theme()
```

```r
# Square root transformation
df |>
  mutate(close=sqrt(close)) |>
  ggplot(aes(x=date, y=close)) +
    geom_line() +
    labs(
      title="Square Root Transformation of the Closing Price"
    ) +
    scale_x_date(date_labels="%Y", date_breaks="1 year") +
    theme_minimal() +
    com_theme()
```

```r
# Plot of the First Difference of the Square Root Series
dif_sqrt_df <- diff(sqrt(df$close)) |>
```

```
            as_tibble() |>
            mutate(
              date = seq(
                start_date + days(1),
                end_date,
                by="days"
              )
            ) |>
            rename(close="value")
# First difference of the square root series
dif_sqrt_df |>
  ggplot(aes(x=date, y=close)) +
    geom_line() +
    labs(
      title="First Difference of the Square Root Series"
    ) +
    scale_x_date(date_labels="%Y", date_breaks="1 year") +
    theme_minimal() +
    com_theme()
```

```
# log_dif_df will be shortened to ldd
acf_ldd <- acf(log_dif_df$close, lag.max=30, plot=FALSE)
ci <- qnorm((1 - .05) / 2) / sqrt(length(acf_ldd$n.used))
# plot the ACF of first dif log time series
acf <- acf_ldd$acf |>
  as_tibble() |>
  mutate(lag=seq(0, 30, 1)) |>
  ggplot(aes(x=lag, y=V1)) +
    geom_point(size=1) +
    geom_hline(yintercept=ci, linetype="dashed", col="blue") +
    geom_hline(yintercept=-ci, linetype="dashed", col="blue") +
    geom_segment(
      aes(
        x=lag,
        xend=lag,
        y=0,
        yend=V1
      )
    ) +
    labs(
      title="ACF of the First Difference of the Logged Series",
      x="Lag",
      y="ACF"
    ) +
    theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
# PACF of the first difference of the logged series
pacf_ldd <- pacf(log_dif_df$close, lag.max=30, plot=FALSE)
#make pacf plot
pacf <- pacf_ldd$acf |>
```

```
  as_tibble() |>
  mutate(lag=seq(1, 30, 1)) |>
  ggplot(aes(x=lag, y=V1)) +
    geom_point(size=3) +
    geom_hline(yintercept=ci, linetype="dashed", col="blue") +
    geom_hline(yintercept=-ci, linetype="dashed", col="blue") +
    geom_segment(
      aes(
        x=lag,
        xend=lag,
        y=0,
        yend=V1
      )
    ) +
    labs(
      title="PACF of the First Difference of the Logged Series",
      x="Lag",
      y="PACF"
    ) +
    theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
# Plot Both ACF and PACF
acf
```

```
pacf
```

```
#Make empty table for AIC values of tested ARIMA models
fit_results <- setNames(
  data.frame(matrix(ncol=2, nrow=0)),
  c("Model", "AIC")
)
#Create dataframe with log transformed closing price
df_log <- df |>
            mutate(close=log(close))
#ARIMA (6,1,0)
fit_arima_610 <- arima(df_log$close, order=c(6,1,0))
#add AIC results of ARIMA(6,1,0) to table
fit_results <- rbind(
  fit_results,
  data.frame(
    Model="ARIMA(6,1,0)",
    AIC=fit_arima_610$aic
  )
)
#ARIMA (4,1,1)
fit_arima_411 <- arima(df_log$close, order=c(4,1,1))
#Add AIC results of arima(4,1,1) to table
fit_results <- rbind(
  fit_results,
  data.frame(
```

```
    Model="ARIMA(4,1,1)",
    AIC=fit_arima_411$aic
  )
)
#ARIMA (4,1,2)
fit_arima_412 <- Arima(df_log$close, order=c(4,1,2))
#add aic value for ARIMA (4,1,2)
fit_results <- rbind(
  fit_results,
  data.frame(
    Model="ARIMA(4,1,2)",
    AIC=fit_arima_412$aic
  )
)
#Show all AIC results of ARIMA models
fit_results |>
  gt() |>
  tab_header(title="Model Fitting AIC Results")
#Residuals of ARIMA(4,1,1)
res_411 <- fit_arima_411$residuals
#ACF values of residuals
res_acf <- acf(res_411, lag.max=30, plot=FALSE)
#PACF values of residuals
res_pacf <- pacf(res_411, lag.max=30, plot=FALSE)
res_ci <- qnorm((1 - .05) / 2) / sqrt(length(res_acf$n.used))
# Histogram of residuals - check normality assumption
res_411 |>
  as_tibble() |>
  ggplot(aes(x=x)) +
    geom_histogram() +
    labs(
      title="ACF Residual Denisty",
      x="Residuals",
      y="Density"
    ) +
    theme_minimal() +
    com_theme()
```

```
# ACF of residuals
acf_res <- res_acf$acf |>
  as_tibble() |>
  mutate(lag=seq(0, 30, 1)) |>
  ggplot(aes(x=lag, y=V1)) +
    geom_point(size=1) +
    geom_hline(yintercept=res_ci, linetype="dashed", col="blue") +
    geom_hline(yintercept=-res_ci, linetype="dashed", col="blue") +
    geom_segment(
      aes(
        x=lag,
        xend=lag,
        y=0,
        yend=V1
```

```
      )
    ) +
    labs(
      title="ACF of the ARIMA(4,1,1) Residuals",
      x="Lag",
      y="ACF"
    ) +
    theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
# PACF of residuals
pacf_res <- res_pacf$acf |>
  as_tibble() |>
  mutate(lag=seq(1, 30, 1)) |>
  ggplot(aes(x=lag, y=V1)) +
    geom_point(size=1) +
    geom_hline(yintercept=res_ci, linetype="dashed", col="blue") +
    geom_hline(yintercept=-res_ci, linetype="dashed", col="blue") +
    geom_segment(
      aes(
        x=lag,
        xend=lag,
        y=0,
        yend=V1
      )
    ) +
    labs(
      title="PACF of the ARIMA(4,1,1) Residuals",
      x="Lag",
      y="PACF"
    ) +
    theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
# Plot both Residual ACF and PACF graphs
acf_res
```

```
pacf_res
```

```
# Forecast with predict
pred <- predict(fit_arima_411, n.ahead=10)
pred_start_date <- as.Date("2021-10-30")
pred_end_date <- pred_start_date + days(9)
#make data frame with predictions and original values to make graphics
pred_df <- data.frame("values"=as.matrix(pred$pred)) |>
            mutate(
              dataset="Predictions",
              date=seq(
                pred_start_date,
                pred_end_date,
```

```
                by="days"
              )
            ) |>
            rename(close="values")

original_df <- df_log |>
                as_tibble() |>
                mutate(
                  dataset="Original",
                  date=seq(
                    start_date,
                    end_date,
                    by="days"
                  )
                )
#combine original and prediction dataframes and filter to year 2021 data
plot_data <- rbind(original_df, pred_df) |>
              filter(date > as.Date("2021-01-01"))
#Add 95% confidence interval to plot dataframe
lower_ci = as.matrix(pred$pred-2*pred$se)
lower_ci = c(rep(NA,301),lower_ci)

upper_ci = as.matrix(pred$pred+2*pred$se)
upper_ci = c(rep(NA,301),upper_ci)

#add data columns to plot_data
plot_data <- plot_data |> mutate(lower_ci = lower_ci,
                    upper_ci = upper_ci)
#add the last observation of original dataset to try to make graphic more continuous
plot_data$lower_ci[301] = 10.67233
plot_data$upper_ci[301] = 10.67233
#Plot of log closing price predictions
plot_data |>
  ggplot(aes(x=date, y=close, col=dataset)) +
  labs(y = 'Bitcoin Closing Price (USD)',
       x = 'Time',
       col = '',
       title = '10 day Predictions of Log Time Series')+
  scale_color_manual(values = c('black','blue'))+
  geom_line(linewidth=1) +
  geom_line(aes(x = date, y = lower_ci)) +
  geom_line(aes(x = date, y = upper_ci)) +
  geom_ribbon(aes(x = date, ymin = lower_ci, ymax = upper_ci), fill = 'gray', alpha = 0.4) +
  guides(color=guide_legend(override.aes=list(fill=NA))) +
  theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
```

```
# Plot 10 day forecast of raw time series
plot_data |>
  ggplot(aes(x=date, y=exp(close), col=dataset)) +
  labs(y = 'Bitcoin Closing Price (USD)',
```

```
      x = 'Time',
      col = '',
      title = '10 day Predictions of Original Time Series')+
  scale_color_manual(values = c('black','blue'))+
  geom_line(linewidth=1) +
  geom_line(aes(x = date, y = exp(lower_ci))) +
  geom_line(aes(x = date, y = exp(upper_ci))) +
  geom_ribbon(aes(x = date,
                  ymin = exp(lower_ci),
                  ymax = exp(upper_ci)),
              fill = 'gray',
              alpha = 0.4) +
  guides(color=guide_legend(override.aes=list(fill=NA))) +
  theme_minimal() +
    com_theme(
      axis.title.x=element_text(vjust=1),
    )
```

```
#log closing price for predicted day 10
plot_data$close[311]
#original closing price predicted day 10
exp(plot_data$close[311])

#original predicted day 10 upper ci price
exp(plot_data$upper_ci[311])
#original predicted day 10 lower rci price
exp(plot_data$lower_ci[311])

#95% conf interval range
exp(plot_data$upper_ci[311]) - exp(plot_data$lower_ci[311])

#log closing price for last observed day
plot_data$close[301]
#original closing price for last observed day
exp(plot_data$close[301])
```