Netflix (NFLX) Stock Price Time Series

Sharon Nguyen

Abstract

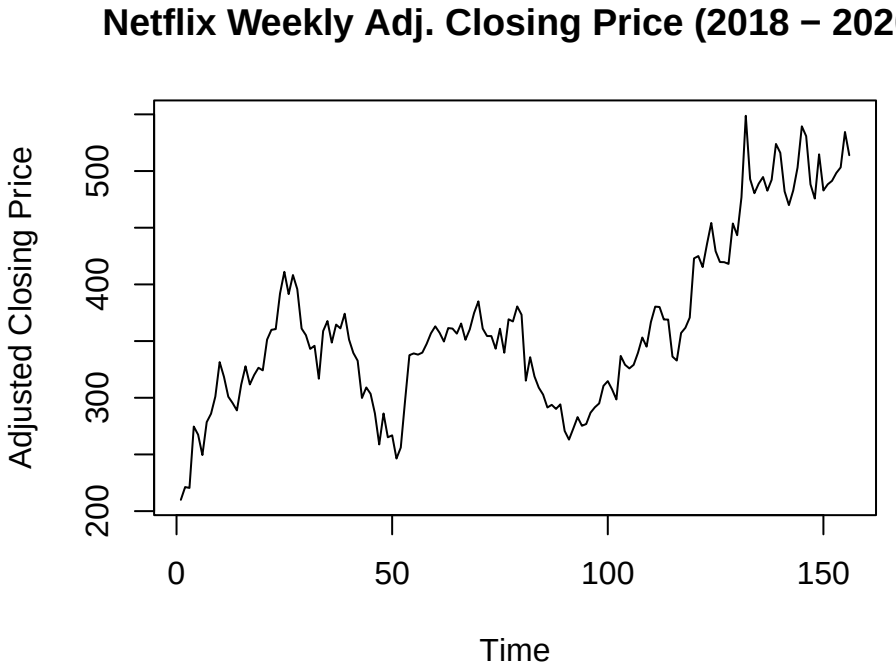
If the COVID-19 pandemic never happened what would the Netflix stock price in 2020 look like? The unforeseen pandemic led to an enexpected stock market crash. To discover the predicted 2020 growth and actual 2020 stock price of NFLX, I used 2018-2019 weekly NFLX stock adjusted closing prices to predict 2020's outcome. The time series is fitted with an ARIMA model (autoregressive integrated moving average model) and the last 9 weeks of 2019 was used as a test set. The model predicted the 9 weeks within the 95% confidence interval. It was also used to predict the next 52 weeks of 2020. The predictions are all within the 95% confidence interval. However, due to the randomness and unpredictability of the stock market the model is only sufficient for short term use.

Introduction

Netflix is one of the four prominent American technology companies. By creating a time series model that can predict the future of Netflix, we can explore the growth or decline of Netflix. This can have a massive effect on not only the stock market but also on society. Currently, there are 192.95 million paid Netflix subscribers as of the second quarter of 2020 (Statista). When investors, traders, and analyst are deciding what stocks are a good investment we can consider the predicted stock prices as part of our decision.

This data set contains Weekly Netflix Stock Adjusted Closing Prices from 2018-2020 and aims to forecast 2020 Adjusted Closing Price for Netflix. On 20 February 2020, stock markets across the world crashed after the growing instability due to the COVID-19 pandemic. This forecast shows the predicted Weekly NFLX Adjusted Closing Price for 2020 if the stock market crash never happened and compares it with the actual Weekly NFLX Adjusted Closing Price for 2020 during the unexpected crash. This expresses the unknown variables that can affect a time series like stocks. I first made a data split between training set and test set. This allows me to use 2018 and part of 2019 data to create models on the training data and then testing its accuracy on the test data. I analyzed autocorrelation and partial autocorrelation plots in RStudio to find the best moving average or autoregressive orders. Also, I used Box-Cox transformations to explore the effect it has on normality of the data. I then tested differencing to out rule trends and seasonality of the data. Furthermore, there was model diagnostics for checking and evaluating the models using Akaike's Information Criteria, numerous tests for normality, and tests for stationarity. The final model predicted the last 9 weeks of 2019 NFLX Stock Adjusted Closing Prices and were within the 95% confidence interval. The 2020 predicted data was also within a 95% confidence interval, however, the interval was very large and was not able to predict stock growth over the weeks. The interval is quite large due to the nature of stocks and how weekly data can also be a bit noisy. Stocks are daily data and averaging it to weekly data can just remove some noise. This also does not account for the volatility of the stock market.

The dataset was downloaded from Yahoo Finance which tracks the daily stock market. I used RStudio software for data transformations, plotting, and some data cleaning. Also, I used it for model building and model diagnostics. The report is fully written in Rmarkdown. Furthermore, Microsoft Excel was used for some quick data cleaning and removal of unneeded variables from the Yahoo Finanace dataset. The RStudio libraries used were dplyr, ggplot2, lubridate, MASS, forecast, astsa, and MuMIn.

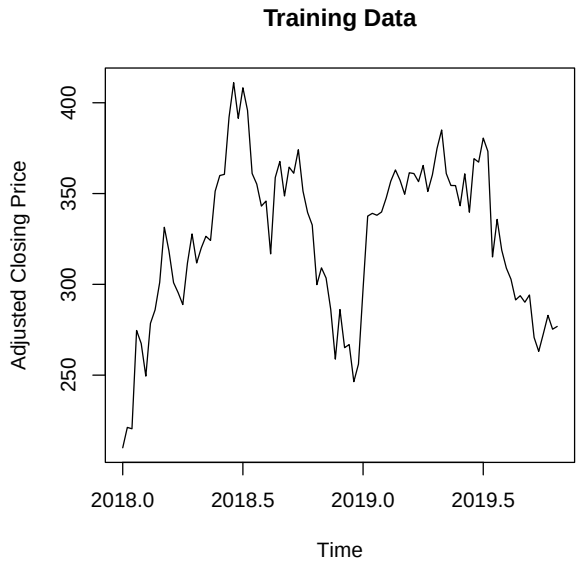


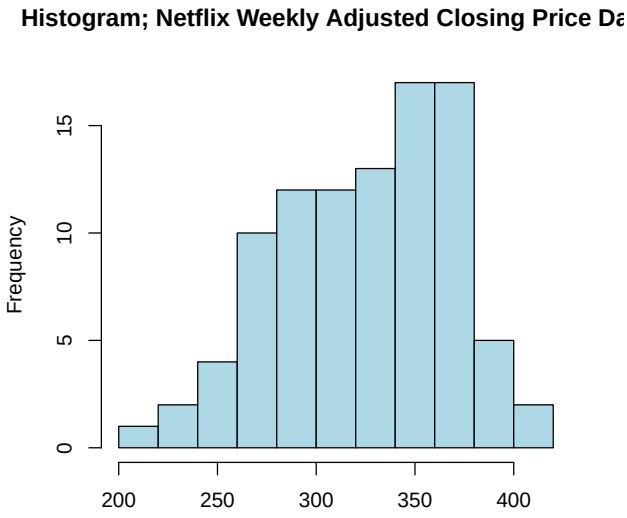
There seems to be an upward trend at the beginning of the year and a downward trend starting at the middle of the year for 2018-2019. In 2020, there seems to be a continuous upward trend.

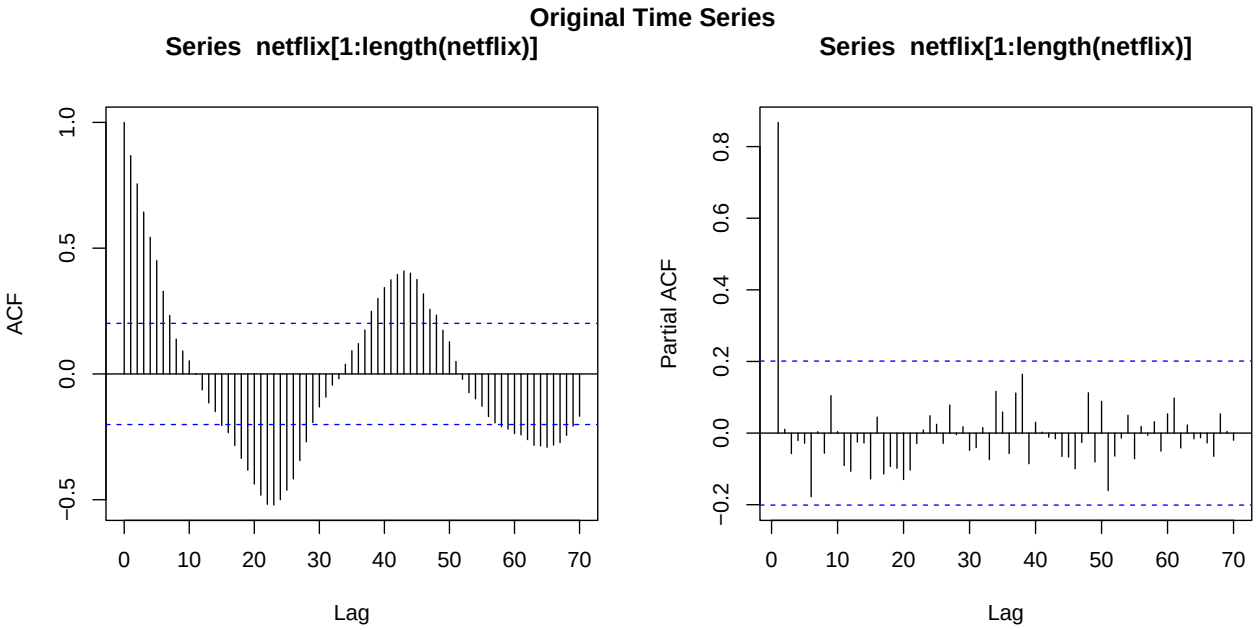
Train and Test Set Split

I partitioned the dataset into two parts for model training and model validation. The training set is used to train our models (2018-2019- first 95 weeks) and the test set (2019- last 9 weeks and 2020- 52 weeks) is used to test our model forecasts.

Netflix Weekly Adj. Closing Price (2018 - 2019) Training Set: 95 Observations





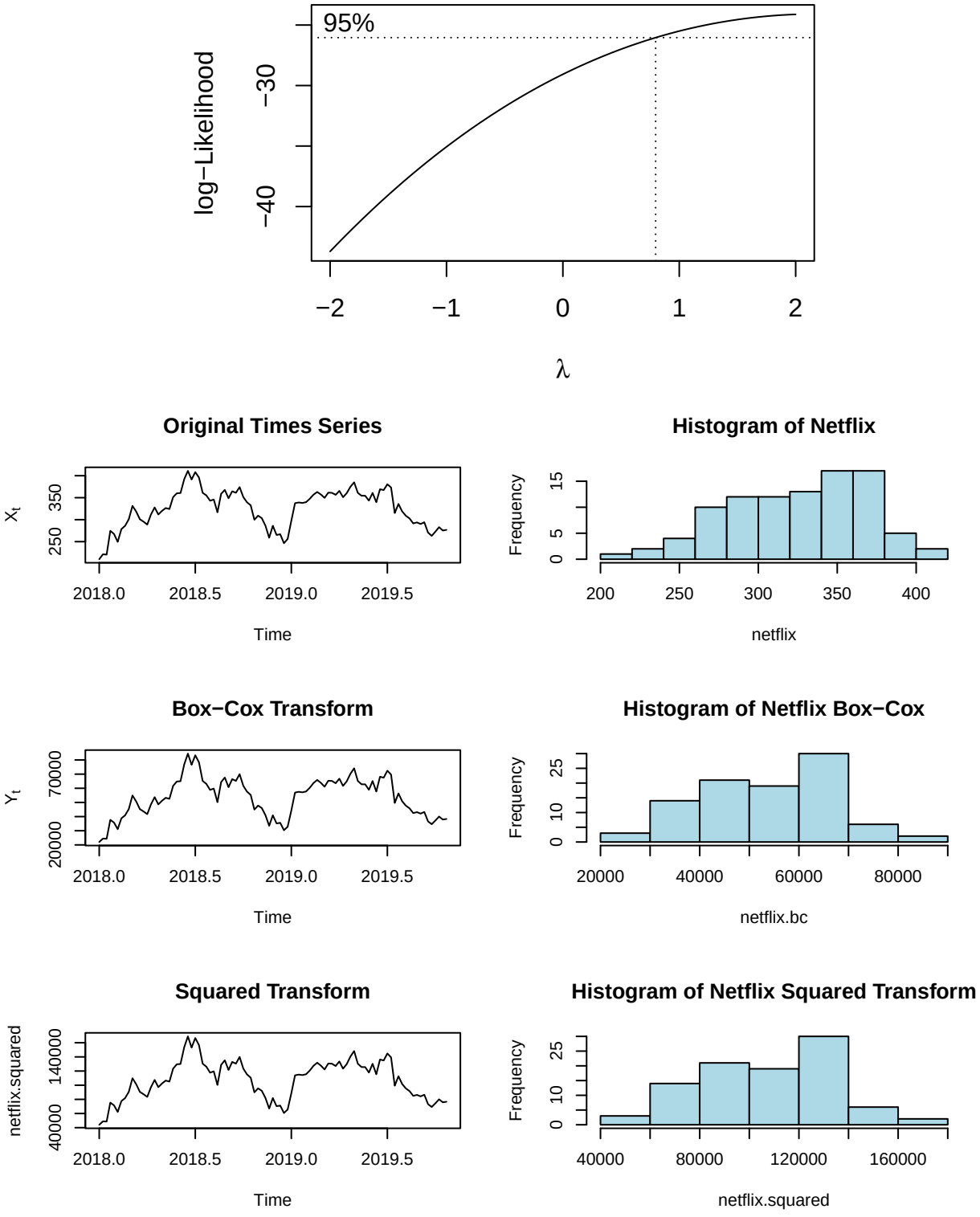


The histogram is not normally distributed and it is left-skewed.

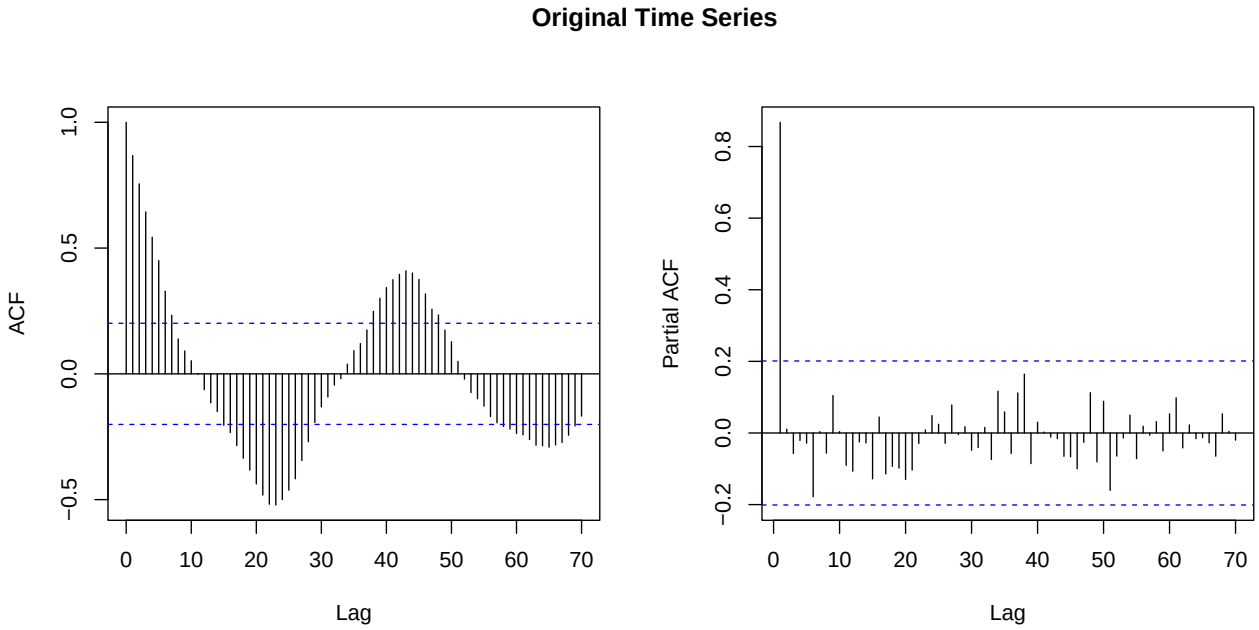
The ACF remains large and periodic. It does not decrease to zero quickly which implies non-stationary data. We may need to difference the data to get stationarity. To stabilize the variance and try to make it normal, we can transform the data using Box-Cox transformation. To remove the seasonality and trend we can difference. We can check if first lag differencing makes the data more stationary and decreases its variance. If we continue differencing and the variance increases than we have overdifferenced and should go back to the previous differenced data.

The data is skewed with variance non constant so we need to try a Box-Cox transformation.

Box-Cox Transformation

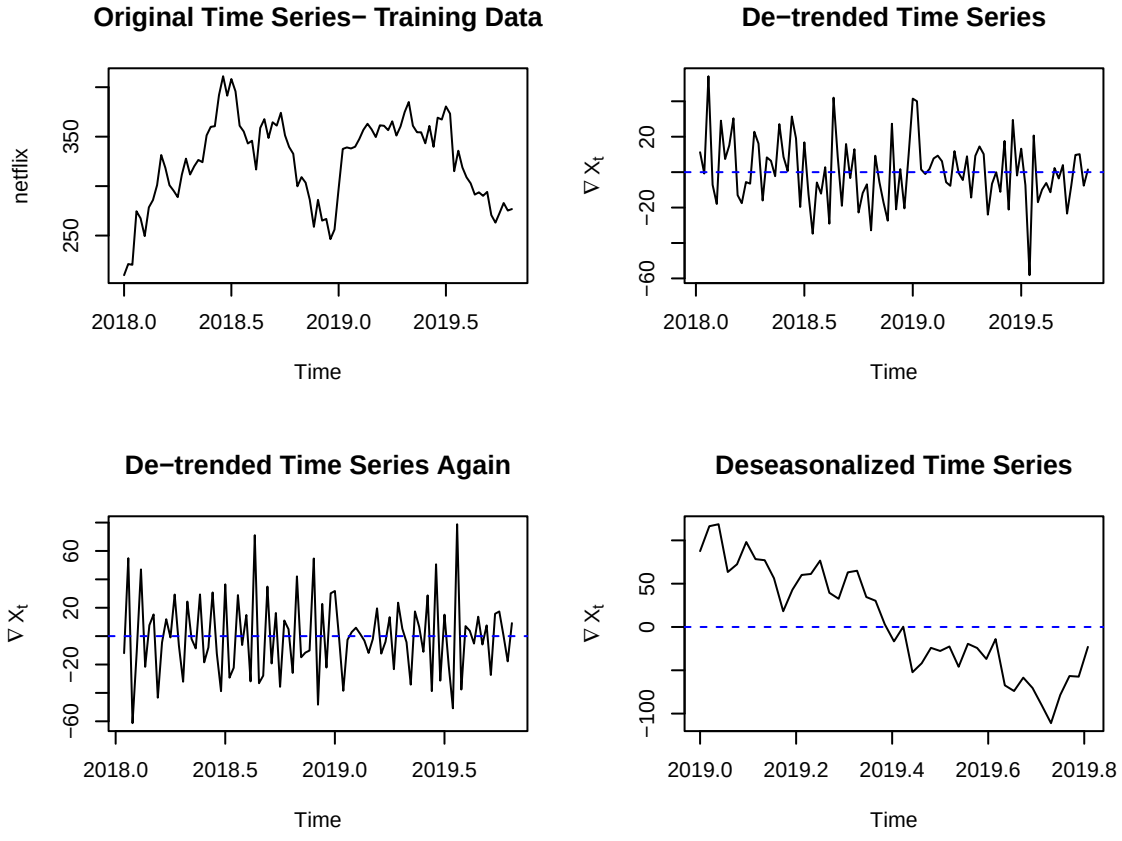


We transformed the data by squaring the data. We are going to choose our original data because the transformations did not change the time series plots that much. The Box Cox Transformed Histogram may be a bit more normal but it is hard to tell so I chose the original data.

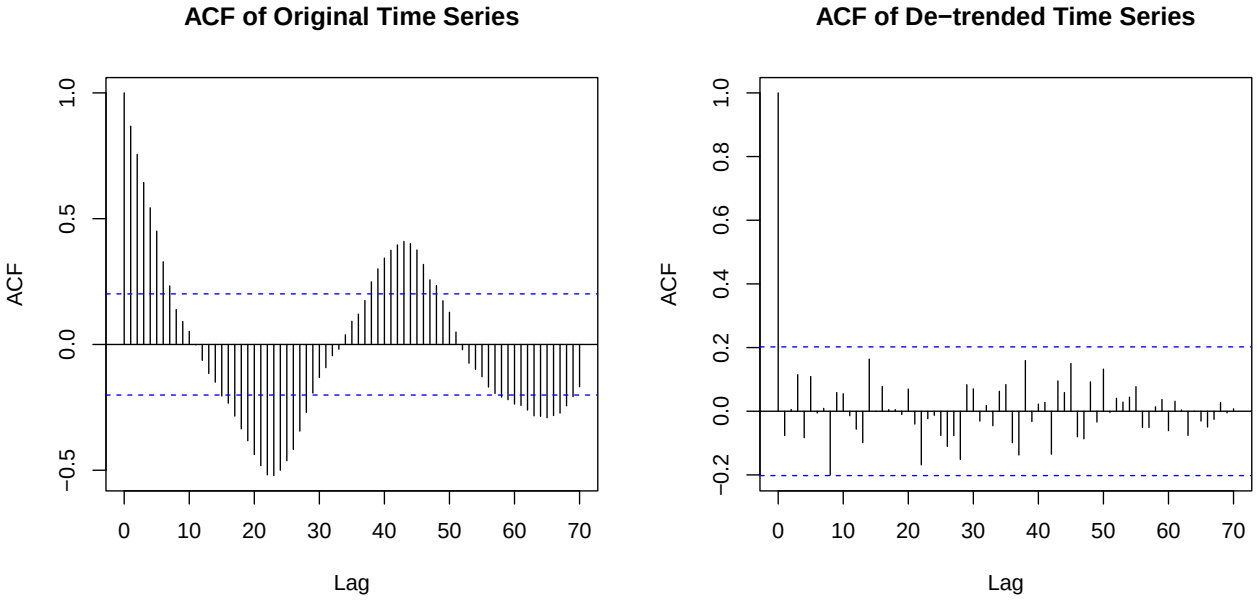


There seems to be cyclical behavior in the ACF of the transformed data. There are significant correlations with values moving proportionally. It could be moving proportionally every 52 lags because we know that this is Weekly Netflix Stock Adjusted Closing Prices for 2018-2019. Therefore, we can say that the period of the seasonal component is given by d = 52.

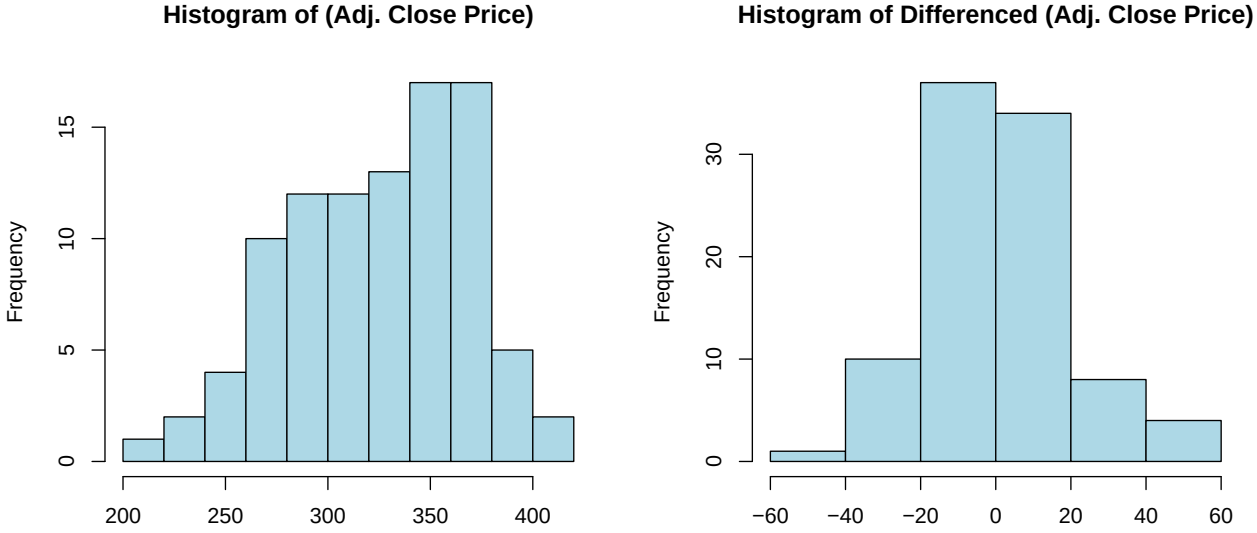
Differencing



We know that Netflix stock prices have a trend. After we difference at lag = 1 to remove the trend, we can see that the data is more stationary compared to the original data plot. This is a good choice because differencing at lag = 1 decreased our variance from Var = 1911.074 to Var = 343.6827. Differencing at lag 1 again only increased variance to Var = 746.7033. Differencing at lag 52 also increased variance. Rather than a seasonal component the time series may have a cyclical component.



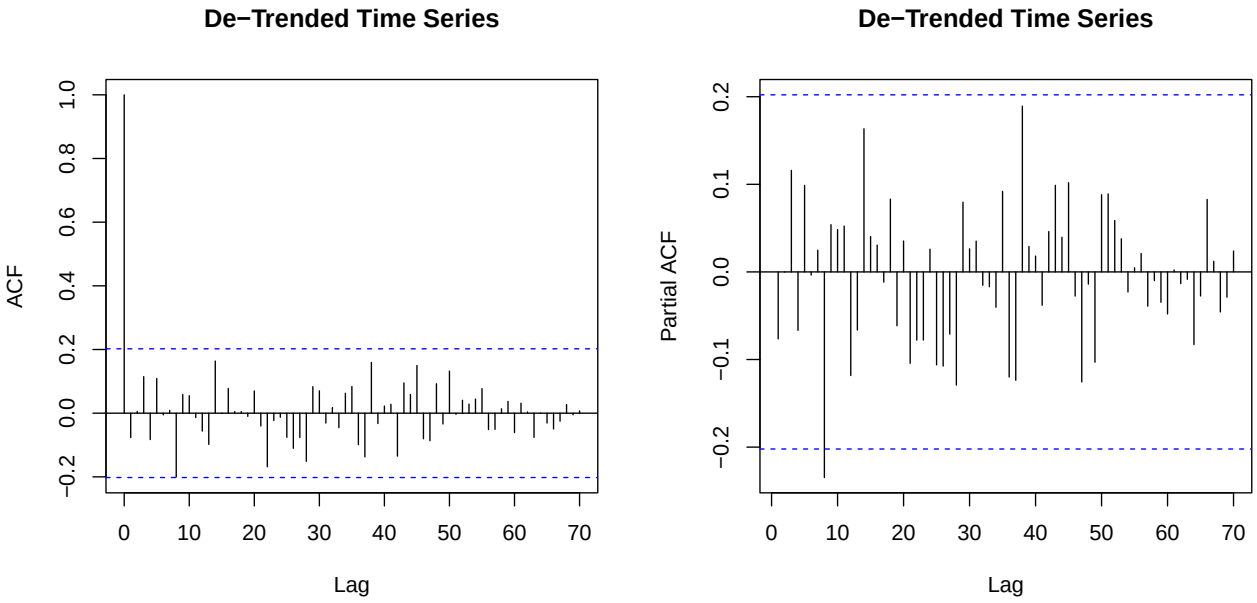
The original data was non-stationary as shown by the ACF remaining large and periodic. It does not decrease to zero quickly which implies non-stationary data. After differencing at lag = 1 to remove the trend component it drops to zero quickly now which means the data is stationary. There seems to be no significant lags meaning that all the lags = 0 after differencing at lag = 1.



The histogram of differenced adjusted close price looks more normally distributed in comparison to the histogram of the original adjusted close price data. From the histogram we can see that the differenced data looks more symmetric.

Model Building

ACF and PACF of de-trended time series $$



For differencing we have d = 1 because we difference at lag = 1 to remove the trend component which made stationary data. The ACF plot shows no significant lags meaning that all the lags = 0 after differencing at lag = 1. The 95% confidence interval is a very conservative estimate so we can consider lag 8 = 0. This means we have moving average model of MA(0). The PACF plot shows a significance at lag 8 which means we have an autoregressive model of AR(8). We can also consider where autoregressive model of AR(0).

Given this information we have a possible ARIMA(p, d, q) model of ARIMA$(8, 1, 0)$,ARIMA$(0, 1, 0)$, or AR(1).

Model Estimation

```
##    p q     AICc
## 1  0 0 816.8718
## 2  1 0 818.4323
## 10 0 1 818.4448
## 3  2 0 820.5669
## 11 1 1 820.5673
## 13 3 1 821.2375
## 4  3 0 821.4166
## 12 2 1 822.5225
## 5  4 0 823.1871
## 14 4 1 823.5155
## 6  5 0 824.4966
## 9  8 0 825.9517
```

After fitting different ARIMA models using maximum likelihood estimate and comparing the model fits using AICc we can choose 3 models with the smallest AICc. These models would be ARIMA$(0, 1, 0)$, AR(1), ARIMA$(8, 1, 0)$. We don't consider p and q for the ARIMA model where the coefficients are within the 95% confidence interval in the ACF/PACF plots. The models suggested with AIC(C) are similar to models suggested by ACF/PACF.

Model ARIMA$(0, 1, 0)$ is random walk , AR(1), ARIMA$(8, 1, 0)$, SAR$(1)_8$.

Model A: ARIMA$(0, 1, 0)$

```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(0, 1, 0), method = "ML")
##
##
## sigma^2 estimated as 340.5:  log likelihood = -407.41,  aic = 816.83
```

Model B: AR(1)

```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(1, 0, 0), method = "ML")
##
## Coefficients:
##          ar1  intercept
##       0.9341   306.4462
## s.e.  0.0395    26.4567
##
## sigma^2 estimated as 331:  log likelihood = -411.44,  aic = 828.87
```

Model C: ARIMA$(8, 1, 0)$

```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(8, 1, 0), method = "ML")
##
## Coefficients:
##           ar1      ar2     ar3      ar4     ar5     ar6     ar7      ar8
##       -0.0517  -0.0055  0.1362  -0.0910  0.1279  0.0075  0.0091  -0.2519
## s.e.   0.0992   0.0991  0.1028   0.1042  0.1028  0.1045  0.1043   0.1034
##
## sigma^2 estimated as 307.3:  log likelihood = -402.9,  aic = 823.81
```

Model D: SAR$(1)_8$

```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(0, 0, 0), seasonal = list(order = c(1,
##     0, 0), period = 8), method = "ML")
##
## Coefficients:
##         sar1  intercept
##       0.2123   322.3652
## s.e.  0.1236     5.7531
##
## sigma^2 estimated as 1828:  log likelihood = -491.75,  aic = 989.5
```

New Model C: ARIMA$(8,1,0)$ fixed

```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(8, 1, 0), transform.pars = FALSE,
##     fixed = c(0, 0, 0, 0, 0, 0, 0, NA), method = "ML")
##
## Coefficients:
##       ar1  ar2  ar3  ar4  ar5  ar6  ar7      ar8
##         0    0    0    0    0    0    0  -0.2189
## s.e.    0    0    0    0    0    0    0   0.1048
##
## sigma^2 estimated as 324.2:  log likelihood = -405.3,  aic = 814.6
```

Model Selection

Stationary/Invertible Checks Random Walk, ARIMA$(0,1,0)$, is nonstationary.

AR(1) models are always invertible. This is a nonstationary AR(1) process with parameter $|\phi| > 1$.

ARIMA$(8,1,0)$ is AR(p) is always invertible by its construction $Z_t = \phi(B)X_t$. AR(p) has MA(∞) representation when the roots of the polynomial $\phi(z)$ lie outside of the unit circle (z^* is a root of $\phi(z)$ if $\phi(z^*) = 0$. Condition: $|z^*| > 1$) which is true for our AR(8).

Random walk and the limiting AR(1) case sways me away from those two models. SAR$(1)_8$ has the highest AIC from the models I chose, thus, we're not going to use that model. We're going to compare the ARIMA$(8,1,0)$ model and the ARIMA$(8,1,0)$ fixed model.

Model Diagnostics

Check: We make the assumption that Z_t is a white noise $Z_t \sim \mathbf{WN}(0, \sigma^2)$.

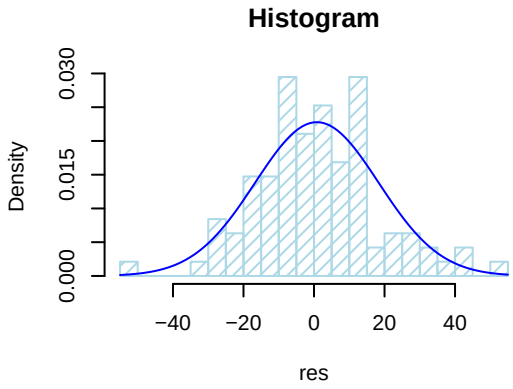
Model C: ARIMA$(8,1,0)$ AIC: 823.81

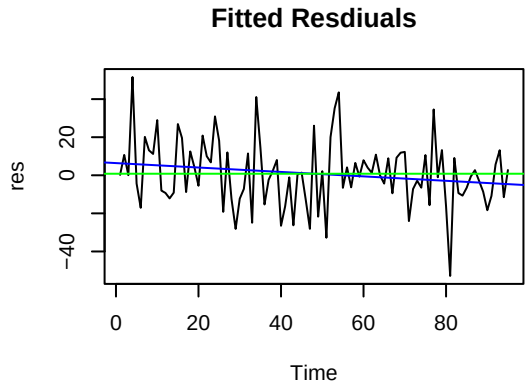
```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.98543, p-value = 0.3769
```

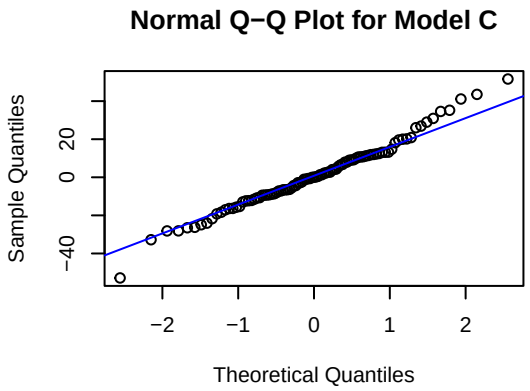
```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 1.1438, df = 2, p-value = 0.5644
```

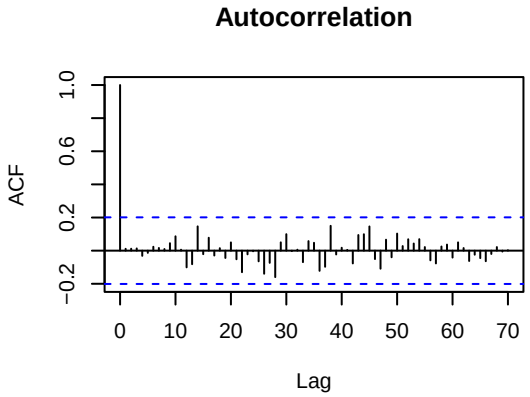
```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 1.2866, df = 2, p-value = 0.5255
```

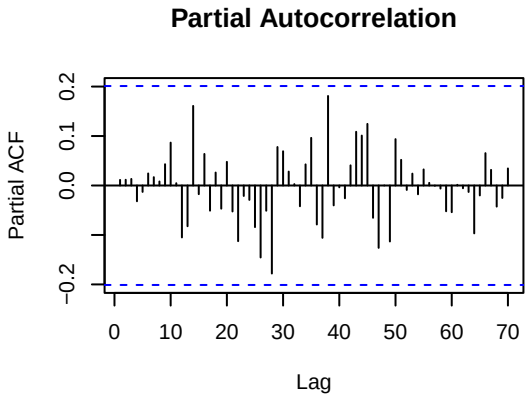
```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 3.1259, df = 10, p-value = 0.9783
```

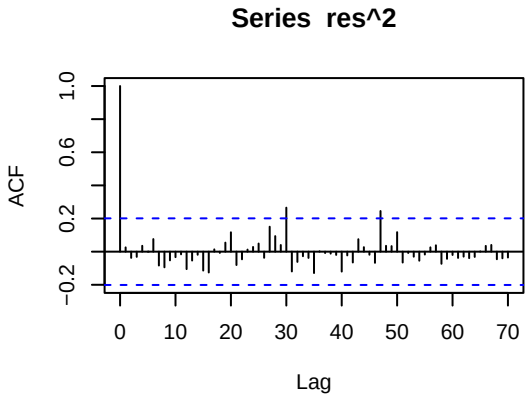













```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  306.6
```

Formula: $\nabla_1 X_t = -0.0517X_{t-1} - 0.0055X_{t-2} + 0.1362X_{t-3} - 0.0910X_{t-4} + 0.1279X_{t-5} + 0.0075X_{t-6} + 0.0091X_{t-7} - 0.2519X_{t-8} + Z_t$

From the diagnostics we test the assumption Z_t is white noise. We see that the statement is true for the model ARIMA$(8, 1, 0)$. The residuals closely resemble white noise and appear to be stationary. There is no apparent trend. There are no changes in variance. The histogram and qq-plot appear to be normal. For the Portmanteau Tests, they test the null hypothesis that the model is normal and all the tests pass because the p-values are larger than 0.05. We fail to reject the null hypothesis that Model C is normally distributed.

The ACF and PACF of the residuals also shows that they are within confidence intervals. At all lags it is equal to 0. The residuals are all independent. The ACF of the residuals squared though has lags that do not equal 0.

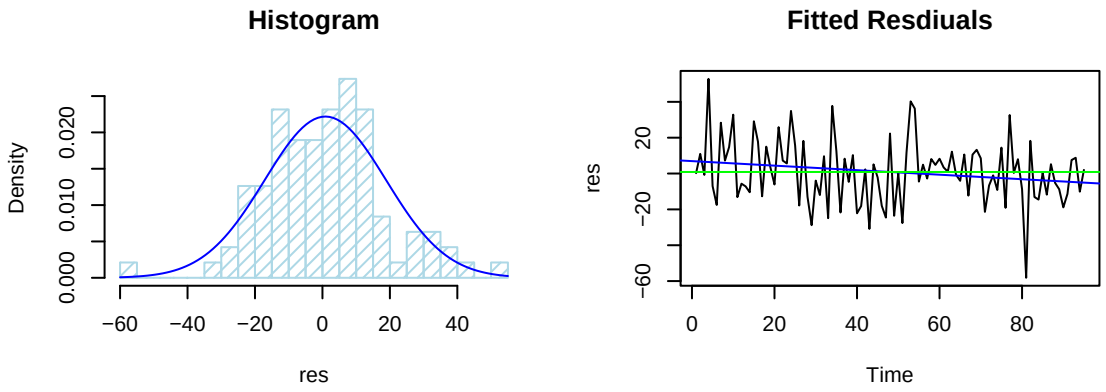
New Model C: ARIMA$(8, 1, 0)$ **fixed** AIC: 814.6

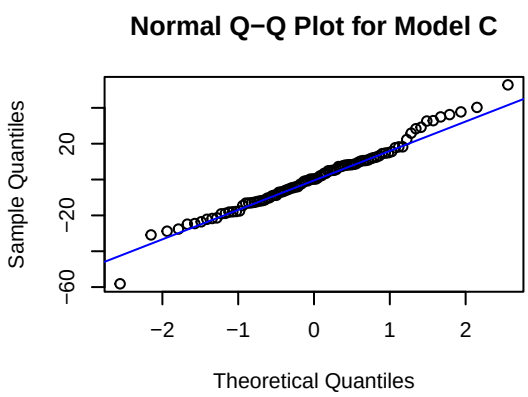
```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.98372, p-value = 0.2877
```

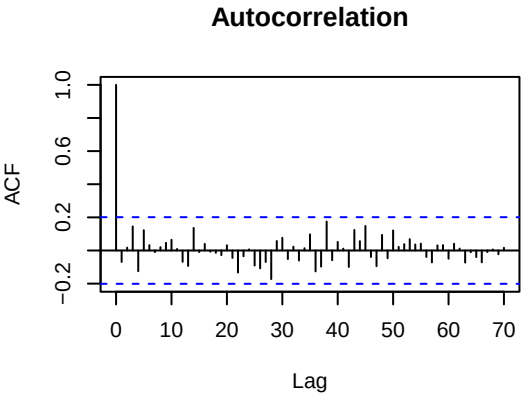
```
##
##  Box-Pierce test
##
## data:  res
## X-squared = 6.1471, df = 9, p-value = 0.7251
```

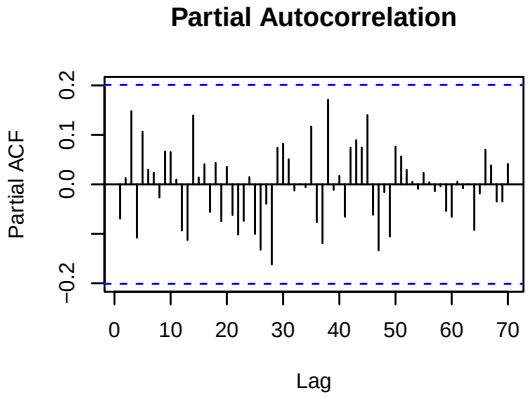
```
##
##  Box-Ljung test
##
## data:  res
## X-squared = 6.5768, df = 9, p-value = 0.6811
```

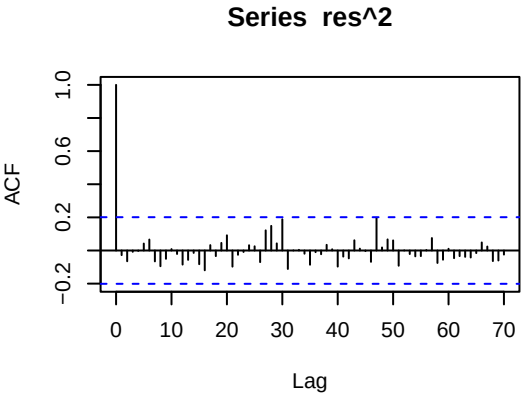
```
##
##  Box-Ljung test
##
## data:  res^2
## X-squared = 2.8173, df = 10, p-value = 0.9854
```












```
##
## Call:
## ar(x = res, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  323.4
```

Formula: $\nabla_1 X_t = -0.2189 X_{t-8} + Z_t$

From the diagnostics we test the assumption Z_t is white noise. We see that the statement is true for the model ARIMA$(8,1,0)$ fixed. The residuals closely resemble white noise and appear to be stationary. There is no apparent trend. There are no changes in variance. The histogram and qq-plot appear to be normal. For the Portmanteau Tests, they test the null hypothesis that the model is normal and all the tests pass

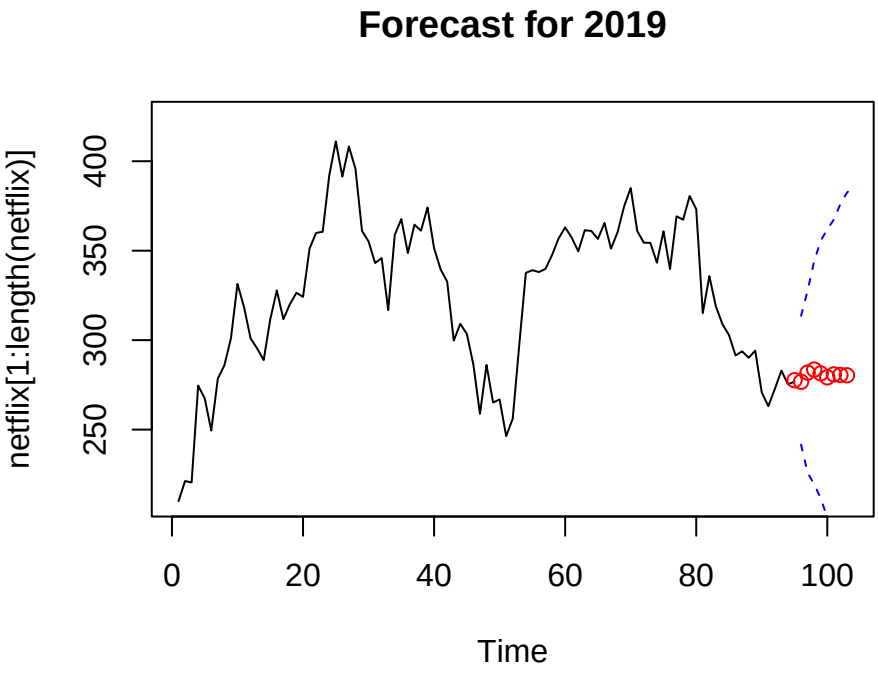
because all the p-values are larger than 0.05. We fail reject the null hypothesis that Model C fixed is normally distributed. This one also has the lowest AIC value.

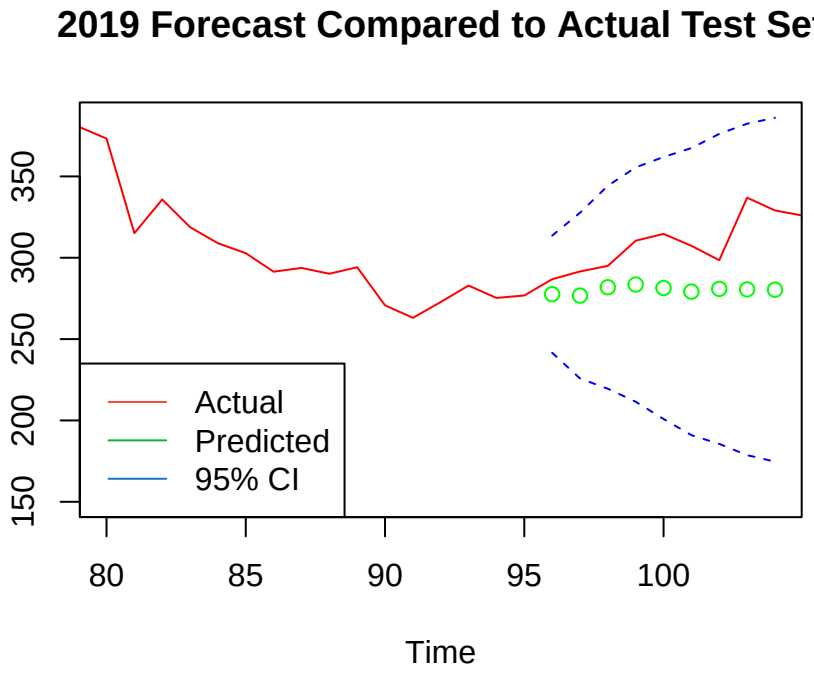
The ACF and PACF of the residuals also shows that they are within confidence intervals. At all lags it is equal to 0. The residuals are all independent. The ACF of the residuals has lags all equal to 0.

Final Model

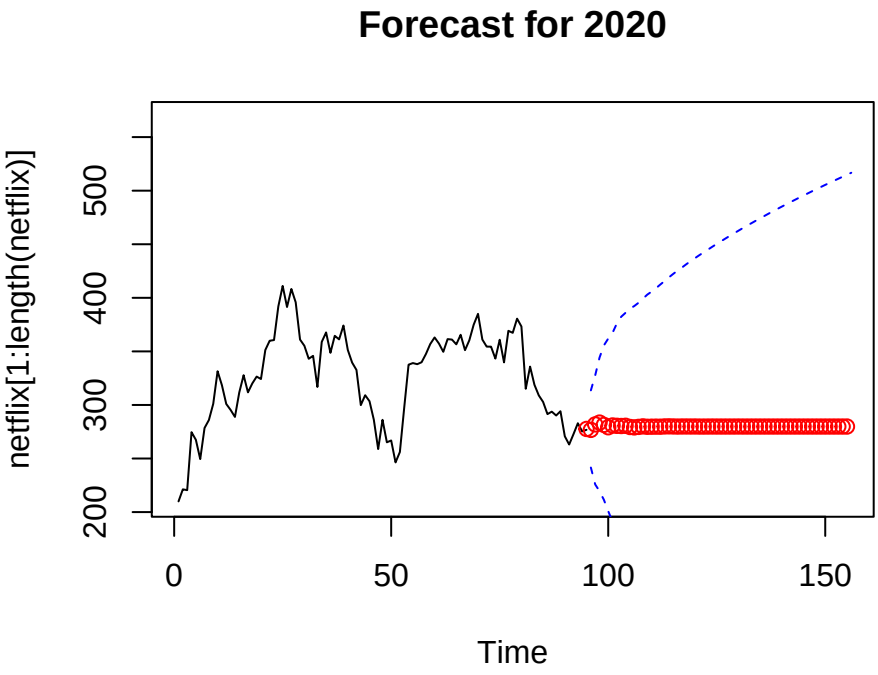
```
##
## Call:
## arima(x = netflix[1:length(netflix)], order = c(8, 1, 0), transform.pars = FALSE,
##     fixed = c(0, 0, 0, 0, 0, 0, 0, NA), method = "ML")
##
## Coefficients:
##       ar1  ar2  ar3  ar4  ar5  ar6  ar7      ar8
##         0    0    0    0    0    0    0  -0.2189
## s.e.    0    0    0    0    0    0    0   0.1048
##
## sigma^2 estimated as 324.2:  log likelihood = -405.3,  aic = 814.6
```

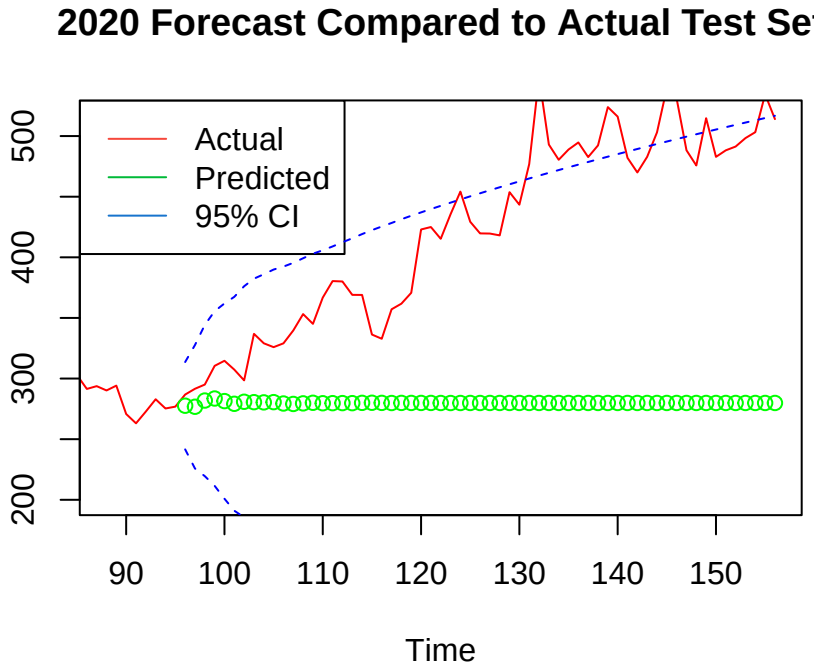
Data Forecasrting





This is the last 9 weeks of 2019 predicted. The values are all within the confidence interval but the predictions are not very exact with the acutal time series.





This is the last 9 weeks of 2019 predicted + 52 weeks of 2020 predicted. The values are all within the confidence interval but the predictions are not very exact with the actual time series. The actual time series for 2020 even falls out of the confidence interval. We can see how unpredictable conditions can affect the stock market and change the path of the time series rapidly. We can also see that as more data is predicted, the model has a hard time predicting any growth. The model is mostly good for short term predicting, if that, because stock prices are so unpredictable.

Conclusion

My final model for forecasting Netflix Stock Adjusted Closing Stock Price is model ARIMA$(8, 1, 0)$ (fixed):

$$\nabla_1 X_t = -0.2189 X_{t-8} + Z_t$$

My predictions are not too far off from the actual data short term. They all lie within the 95% confidence interval. However, the confidence interval gets quite large and we can see that in 2020 with the sudden change in stock prices, the model predicts poorly as even the actual data starts to leave the 95% confidence interval. However, because this is stock data it is important to consider the fact that I averaged this data into weekly data. Daily data predictions would be much noisier and hard to forecast. Due to unknown circumstances, we cannot predict stock prices accurately long term such as with the unexpected stock market crash of 2020 due to COVID-19, "Random walk theory of stocks suggests that changes in stock prices have the same distribution and are independent of each other" (Investopedia). The theory means that we assume past movement or trends of a stock cannot be used for future predictions because stock prices are random and unpredictable. This makes it difficult to predict stock prices long term. This is proven by the sudden crash in the market after the growing instability due to the COVID-19 pandemic. I compared the predicted 2020 Weekly NFLX Adjusted Closing Price based on the best model found for the time series based on 2018-2019 and compared it to the actual 2020 Weekly NFLX Adjusted Closing Stock Price data. Notice, how the confidence interval is very large and the predictions become very inaccurate as more weeks are trying to be predicted. The model cannot predict the NFLX stock price growth. The stock market is volatile and hard to predict long term and time series models on NFLX stock can only be useful short term due to the nature of the market and unforeseen circumstances.

Throughout the writing of this report, I have received a great deal of support and assistance. I sincerely thank Professor Feldman and Youhong Lee whose guidance was invaluable in formulating this time series report.

References

Data: https://finance.yahoo.com/quote/NFLX/history?p=NFLX

Lecture Material: Professor Raya Feldman

https://www.investopedia.com/terms/r/randomwalktheory.asp

https://www.statista.com/statistics/250934/quarterly-number-of-netflix-streaming-subscribers-worldwide/

Appendix: Include your full code with comments.

```
knitr::opts_chunk$set(echo = FALSE,
                      results = 'hide')
library(dplyr)
library(ggplot2)
library(lubridate)
library(MASS)
library(forecast)
library(astsa)
library(MuMIn)
# Load data
NFLX.csv <- read.csv("NFLX.csv")
NFLX.csv <- NFLX.csv %>% dplyr::select(Date, Adj.Close, Week)

# Change Date Variable to Date data type
NFLX.csv$Date <- as.Date(NFLX.csv$Date, format= "%m/%d/%y")

# Check Variable data types
sapply(NFLX.csv, class)
# Remove 2020 data
# I removed the 2020-2022 data because of the pandemic changes.
NFLX <- NFLX.csv[c(1:104),]

# 2020 Data
NFLX2020 <- NFLX.csv[-c(1:104),]
plot.ts(NFLX.csv$Adj.Close,
        main = "Netflix Weekly Adj. Closing Price (2018 - 2020)",
        ylab = "Adjusted Closing Price")
par(mfrow=c(1, 2))

plot.ts(NFLX$Adj.Close,
        main = "Netflix Weekly Adj. Closing Price (2018 - 2019)",
        ylab = "Adjusted Closing Price")

# Added trend to data plot
fit <- lm(NFLX$Adj.Close ~ as.numeric(1:length(NFLX$Adj.Close)))
```

```
abline(fit, col="blue")

# Added mean (constant) to data plot
abline(h=mean(NFLX$Adj.Close), col="green")

# Add legend to plot
legend("topright", legend=c("Trend", "Mean"),
       col=c("blue", "green"), lty=1, cex=0.8)

# Plot data with Weeks on x-axis
ts.adj.close <- ts(NFLX$Adj.Close, start = c(2018, 1), frequency = 52)
ts.plot(ts.adj.close,
        main = "Raw Data",
        ylab= 'Adjusted Closing Price')
# Partition dataset to two parts for model training and model validation
# Training Dataset
NFLX.train <- NFLX[c(1:95),]

# Testing Dataset
NFLX.test <- NFLX[-c(1:95),]
par(mfrow=c(1, 2))
netflix <- ts(NFLX.train[,2], start = c(2018, 1), frequency = 52)
netflix.test <- ts(NFLX.train[,2], start = c(2019, 10), frequency = 52)

# Plot
ts.plot(netflix,
        main = "Training Data",
        ylab = "Adjusted Closing Price")

# Check normality with Histogram
hist(netflix,
     col="light blue",
     xlab ="",
     main = "Histogram; Netflix Weekly Adjusted Closing Price Data")
# Plot ACF and PACF of Original Data
op = par(mfrow = c(1,2))
acf(netflix[1:length(netflix)], lag.max=70)
pacf(netflix[1:length(netflix)], lag.max=70)
title("Original Time Series", line=-1, outer=TRUE)
t = 1:length(netflix)
fit = lm(netflix ~ t)
bcTransform = boxcox(netflix ~ t, plotit = TRUE)

lambda = bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
netflix.bc = (1/lambda) * (netflix^lambda - 1)

# Power 2 Transform
netflix.squared = netflix^2
# Compare transforms
op= par(mfrow=c(3,2))
ts.plot(netflix, main = "Original Times Series", ylab = expression(X[t]))
hist(netflix,
     col="light blue",
```

```
     main = "Histogram of Netflix")

ts.plot(netflix.bc, main = "Box-Cox Transform", ylab = expression(Y[t]))
hist(netflix.bc,
     col="light blue",
     main = "Histogram of Netflix Box-Cox")

ts.plot(netflix.squared, main = "Squared Transform")
hist(netflix.squared,
     col="light blue",
     main = "Histogram of Netflix Squared Transform")
op = par(mfrow = c(1,2))
acf(netflix[1:length(netflix)], lag.max = 70, main="")
pacf(netflix[1:length(netflix)], lag.max = 70, main="")
title("Original Time Series", line = -1, outer=TRUE)
op = par(mfrow = c(3,2))
# Original Data
plot.ts(netflix,
        main = "Original Time Series- Training Data")
# Difference at lag = 1 to remove trend component
netflix1 <- diff(netflix, 1)
plot.ts(netflix1, main = "De-trended Time Series", ylab = expression(nabla~X[t]))
abline(h = 0, lty = 2, col = "blue")

# Difference at lag = 1 one more time
netflix2 <- diff(netflix1, 1)
plot.ts(netflix2, main = "De-trended Time Series Again", ylab = expression(nabla~X[t]))
abline(h = 0, lty = 2, col = "blue")

# Difference at lag = 52 to remove seasonal component
netflix52 <- diff(netflix, 52)
plot.ts(netflix52, main = "Deseasonalized Time Series", ylab = expression(nabla~X[t]))
abline(h = 0, lty = 2, col = "blue")

# # Difference at lag = 1 to remove trend component
# net <- diff(netflix52, 1)
# plot.ts(net, main = "Deseasonalized/De-trended Time Series",
#         ylab = expression(nabla~X[t]))
# abline(h = 0, lty = 2, col = "blue")
# Check Variance
var(netflix)
var(netflix1)
var(netflix2)
var(netflix52)
# var(net)
op = par(mfrow = c(1,2))
acf(netflix[1:length(netflix)], lag.max = 70, main = "ACF of Original Time Series")
acf(netflix1[1:length(netflix1)], lag.max = 70, main="ACF of De-trended Time Series")
op = par(mfrow = c(1,2))
hist(netflix,
     col="light blue",
     xlab ="",
     main="Histogram of (Adj. Close Price)")
```

```r
hist(netflix1,
     col="light blue",
     xlab ="",
     main="Histogram of Differenced (Adj. Close Price)")
op = par(mfrow = c(1,2))
acf(netflix1[1:length(netflix1)], lag.max = 70, main = "De-Trended Time Series")
pacf(netflix1[1:length(netflix1)], lag.max = 70, main = "De-Trended Time Series")
# Candidate models:
df <- expand.grid(p=0:8, q=0:1)
df <- cbind(df, AICc=NA)
# Compute AICc:

for (i in 1:nrow(df)) {
  try(arima.obj <- arima(netflix, order=c(df$p[i], 1, df$q[i]), method="ML",
                         optim.control= list(maxit = 1000)))

  if (!is.null(arima.obj)) { df$AICc[i] <- AICc(arima.obj) }
  # print(df[i, ])
}

# Complete AICc Table Sorted from Descending AICc
df_ordered <- df[order(df$AICc),]
df_ordered[1:12,]
# ARIMA(0,1,0)
fit <- arima(netflix[1:length(netflix)], order=c(0,1,0), method="ML")
fit
abs(polyroot(c(1,fit$coef))) > 1
# AR(1)
fit2 <- arima(netflix[1:length(netflix)], order=c(1,0,0), method="ML")
fit2
abs(polyroot(c(1,fit2$coef))) > 1
# ARIMA(8,1,0)
fit3 <- arima(netflix[1:length(netflix)], order=c(8,1,0), method="ML")
fit3
abs(polyroot(c(1, fit3$coef))) > 1
# SAR(1)_8
fit5 <- arima(netflix[1:length(netflix)], order = c(0, 0, 0),
              seasonal = list(order=c(1, 0, 0), period = 8), method="ML")
fit5
# sarima(netflix[1:length(netflix)],
#                p = 1, d = 0, q = 0,
#                P = 0, D = 0, Q = 0,
#                S = 8)
abs(polyroot(c(1, fit5$coef))) > 1
# ARIMA(8,1,0) fixed
fit4 <- arima(netflix[1:length(netflix)], order=c(8,1,0),
              fixed=c(0,0,0,0,0,0,0,NA), method="ML",
              transform.pars = FALSE) # change coefficients to 0 if within 95% CI
fit4
abs(polyroot(c(1, fit4$coef))) > 1
# Check Roots
abs(polyroot(c(1, fit3$coef))) > 1
```

```r
abs(polyroot(c(1, fit4$coef))) > 1
par(mfrow=c(3,2))
# Test for independence of residuals
res <- residuals(fit3)

# Histogram
hist(res, density=20, breaks=20, prob=TRUE, main = "Histogram", col="light blue")
curve(dnorm(x, mean(res), sqrt(var(res))), add=TRUE, col="blue")

# Plot
ts.plot(res, main="Fitted Resdiuals")
fitt <- lm(res~as.numeric(1:length(res))); abline(fitt, col="blue")
abline(h=mean(res), col="green")

# q-q plot
qqnorm(res, main="Normal Q-Q Plot for Model C")
qqline(res,col ="blue")

# acf
acf(res, lag.max=70, main = "Autocorrelation")
# pacf
pacf(res, lag.max=70, main = "Partial Autocorrelation")

# Test for normality of resdiuals
shapiro.test(res)

Box.test(res, lag = 10, type="Box-Pierce", fitdf=8)
Box.test(res, lag = 10, type="Ljung-Box", fitdf=8)
Box.test(res^2, lag = 10, type="Ljung-Box", fitdf=0)

acf(res^2, lag.max = 70)
ar(res,aic=TRUE, order.max=NULL, method=c("yule-walker"))

# Add overall title
# title("Fitted Residuals Diagnostics", outer=TRUE)
par(mfrow=c(3,2))
# Test for independence of residuals
res <- residuals(fit4)

# Histogram
hist(res, density=20, breaks=20, prob=TRUE, main = "Histogram", col="light blue")
curve(dnorm(x, mean(res), sqrt(var(res))), add=TRUE, col="blue")

# Plot
ts.plot(res, main="Fitted Resdiuals")
fitt <- lm(res~as.numeric(1:length(res))); abline(fitt, col="blue")
abline(h=mean(res), col="green")

# q-q plot
qqnorm(res, main="Normal Q-Q Plot for Model C")
qqline(res,col ="blue")

# acf
```

```
acf(res, lag.max=70, main = "Autocorrelation")
# pacf
pacf(res, lag.max=70, main = "Partial Autocorrelation")

# Test for normality of resdiuals
shapiro.test(res)

Box.test(res, lag = 10, type="Box-Pierce", fitdf=1)
Box.test(res, lag = 10, type="Ljung-Box", fitdf=1)
Box.test(res^2, lag = 10, type="Ljung-Box", fitdf=0)

acf(res^2, lag.max = 70)
ar(res,aic=TRUE, order.max=NULL, method=c("yule-walker"))

# Add overall title
# title("Fitted Residuals Diagnostics", outer=TRUE)
# Final model:
fit <- arima(netflix[1:length(netflix)], order=c(8,1,0),
             fixed=c(0,0,0,0,0,0,0,NA), method="ML", transform.pars = FALSE)
fit
forecast(fit)
# Predict 9 future observations and plot
par(mfrow=c(1, 1))
mypred <- predict(fit, n.ahead = 9)
U <- mypred$pred + 2*mypred$se
L <- mypred$pred - 2*mypred$se
ts.plot(netflix[1:length(netflix)], xlim=c(1, length(netflix1)+9),
        ylim = c(min(netflix), max(U)*1.1),main="Forecast for 2019")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(netflix1)+1):(length(netflix1)+9), mypred$pred, col="red")
# forecasts and true values
ts.plot(NFLX.csv[1:length(NFLX.csv)], xlim = c(80,length(netflix)+9),
        ylim = c(150, max(U)), col="red",
        main="2019 Forecast Compared to Actual Test Set")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(netflix)+1):(length(netflix)+9), mypred$pred, col="green")
legend("bottomleft", legend=c("Actual", "Predicted", "95% CI"), col=2:4, lty=1)
mypred <- predict(fit, n.ahead = 61)
U <- mypred$pred + 2*mypred$se
L <- mypred$pred - 2*mypred$se
ts.plot(netflix[1:length(netflix)], xlim=c(1, length(netflix1)+61),
        ylim = c(min(netflix), max(U)*1.1),main="Forecast for 2020")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(netflix1)+1):(length(netflix1)+61), mypred$pred, col="red")
# forecasts and true values
ts.plot(NFLX.csv[1:length(NFLX.csv)], xlim = c(88,length(netflix)+61),
        ylim = c(200, max(U)), col="red",
        main="2020 Forecast Compared to Actual Test Set")
legend("topleft", legend=c("Actual", "Predicted", "95% CI"), col=2:4, lty=1)
lines(U, col="blue", lty="dashed")
```

```
lines(L, col="blue", lty="dashed")
points((length(netflix)+1):(length(netflix)+61), mypred$pred, col="green")
```