Nasdaq 100 Futures

Forecasting Nasdaq 100 Futures using Time series methods in R

Karan Swamy

Abstract

The purpose of this project is to create the most accurate forecast with my current knowledge in order to forecast Nasdaq 100 futures stock prices. To start, I split the data into a training and testing set, so that we can train our data and test our predictions using the testing set. Next, I plotted the training data to look for trends and seasonality. Based on my observations I decided to perform a box-cox transformation in order to stabilize the variance. I also observed that the data had a trend and seasonality, which needed to be removed in order to perform proper forecasting. I removed the trend and seasonality by differencing the data twice. I then did the necessary tests and verified visually that the transformed data was stationary. Once the data was stationary with a stable variance it was time to analyze the ACF and PACF of the differenced data, to select an appropriate model. After picking two models and performing diagnostic checking on them, I concluded that a $SARIMA(0,1,0)(0,1,1)_{12}$ model was appropriate for forecasting. Finally, I forecasted values 12 months ahead of the training data and compared them to the true values of the data to ensure that our model was accurate. Based on my forecasts, I concluded that my model had a reasonably accurate prediction for the true Nasdaq 100 futures stock prices.

Introduction

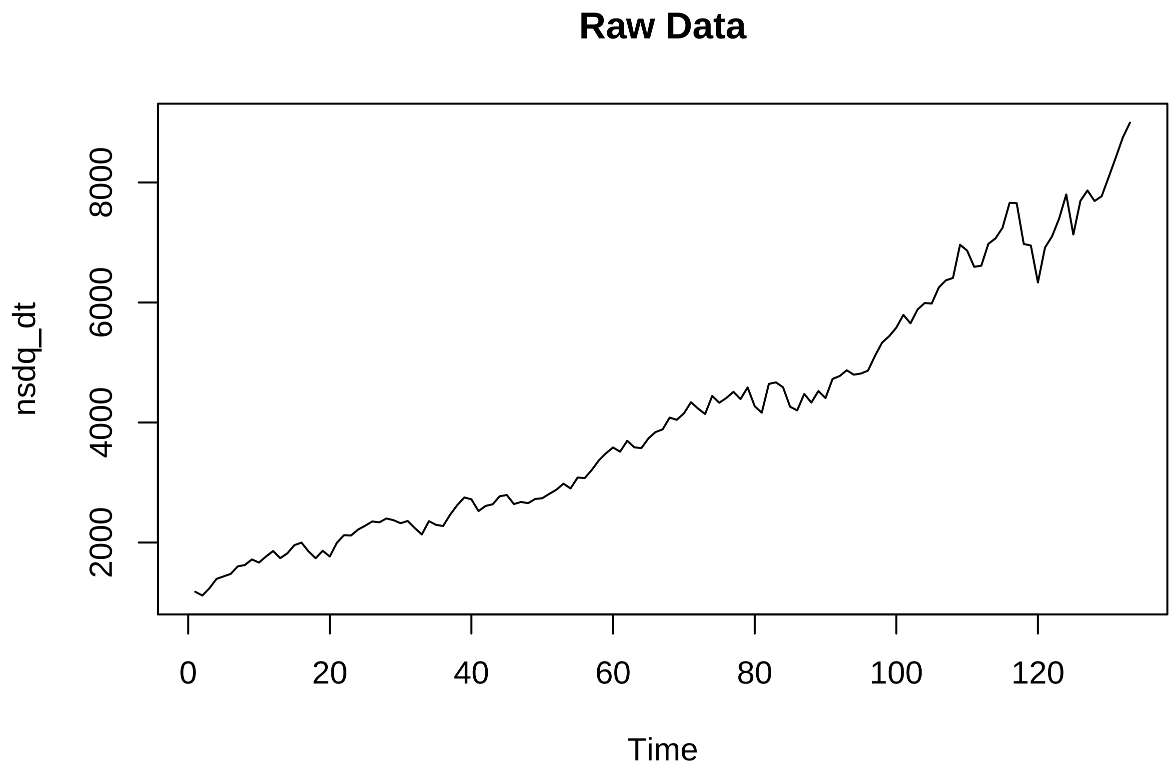
Nasdaq 100 Futures hold significant importance in the financial markets. They provide investors with exposure to the performance of major technology companies in the Nasdaq 100 Index, allowing them to benefit from their market performance. These futures also offer opportunities for portfolio diversification and risk management, enabling investors to spread risk and protect against potential downturns. Furthermore, Nasdaq 100 Futures contribute to market efficiency and price discovery, while providing traders with the ability to profit from both upward and downward price movements. Given this description, we can see the importance of the Nasdaq 100 Futures. In my time series analysis and forecasting, I will be using Nasdaq 100 Futures historical data for forecasting. This data contains monthly values from January 1st, 2009, to January 1st, 2020. I acquired this data from NASDAQS Futures Historical Price Data. Investing.com. (n.d.). https://www.investing.com/indices/nq-100-futures-historical-data.

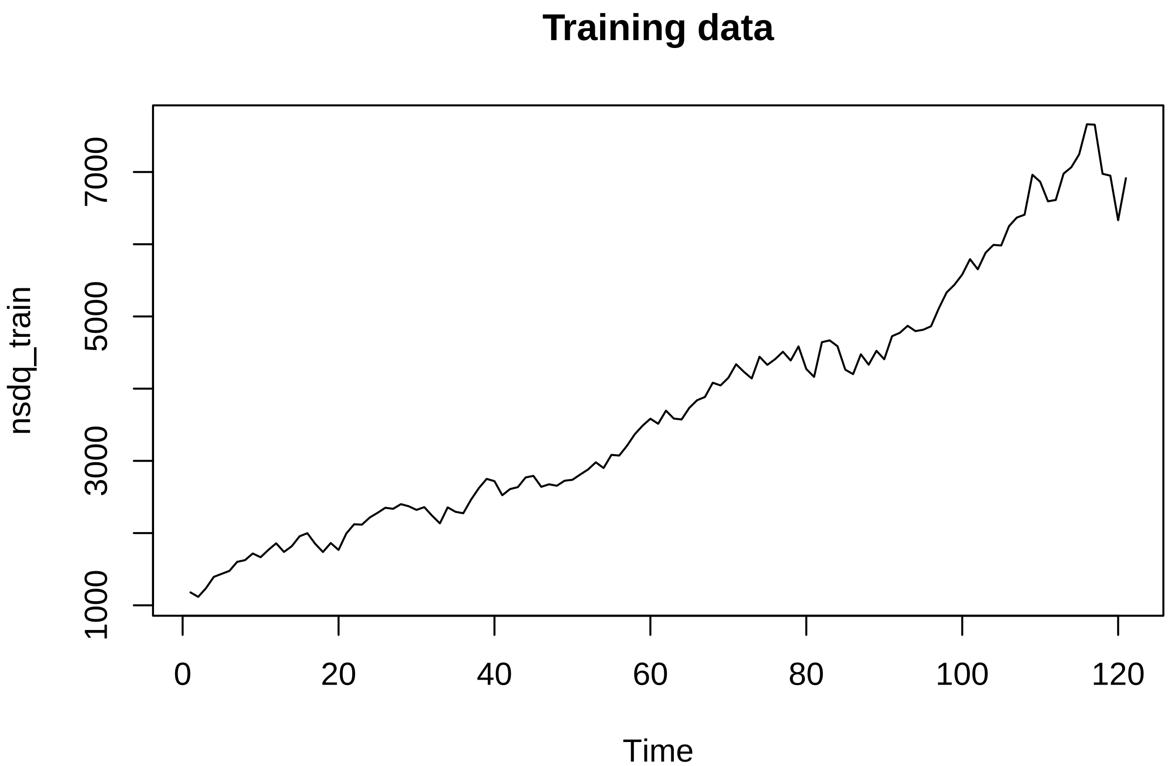
The first step of this project was splitting the data into training and testing sets. I cut off the last 12 observations of the data for the training set and used the last 12 observations of the data for the testing set. I then applied a box-cox transformation to the training data after observing a non-constant variance. I then successfully removed the trend and seasonality that I observed in the decomposed data. I ensured this data was stationary by observing the data's distribution and plotting the ACF and PACF. Next, I performed model selection by analyzing the ACF and PACF of the differenced data. I concluded that a $SARIMA(0,1,0)(0,1,1)_{12}$ model was the best model to use for forecasting, as it had the lowest Aikaike's Information Corrected Criterion (AICc), and passed all diagnostic checks such as the the Shapiro-Wilk test and Box-Ljung test. I then proceeded to perform forecasting with the chosen model. After predicting values for the next 12 months using the training data, I compared them with the actual values to validate the accuracy of our model. After analyzing the forecasts, I determined that our model had a reasonably precise prediction for the actual Nasdaq 100 futures stock prices.

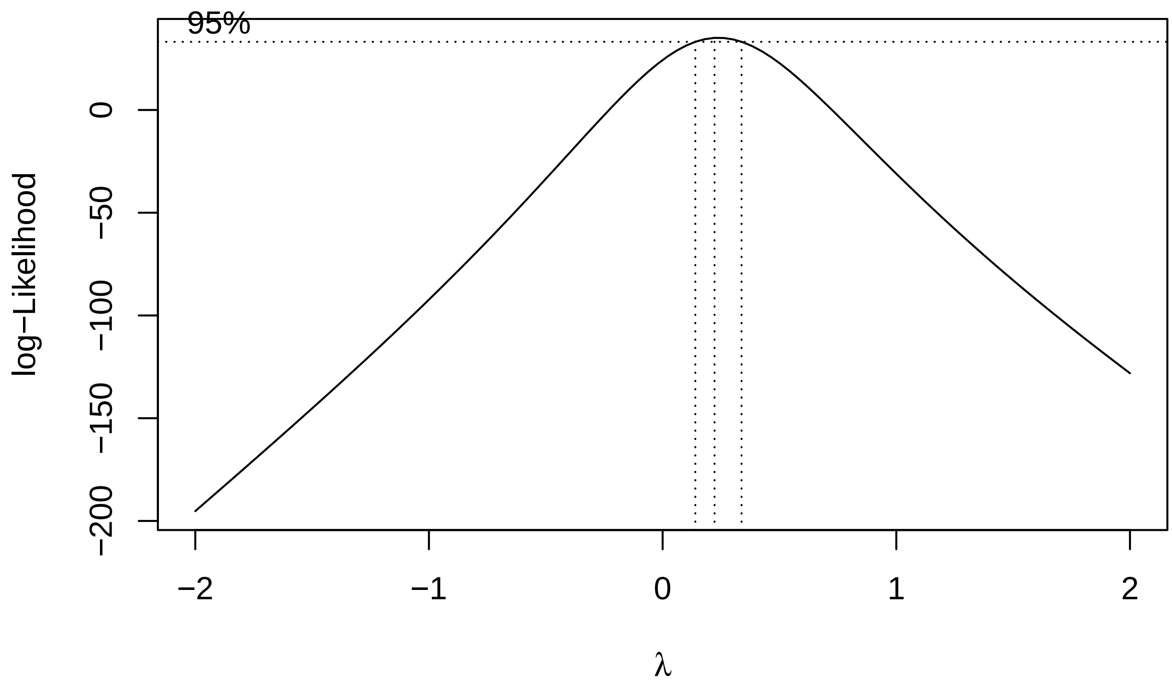
Time Series Analysis

Plotting Raw Data

To start let's plot our raw data in order to get an idea of the shape of the graph.




Training data
nsdq_train
1000
3000
5000
7000
0
20
40
60
80
100
120
Time

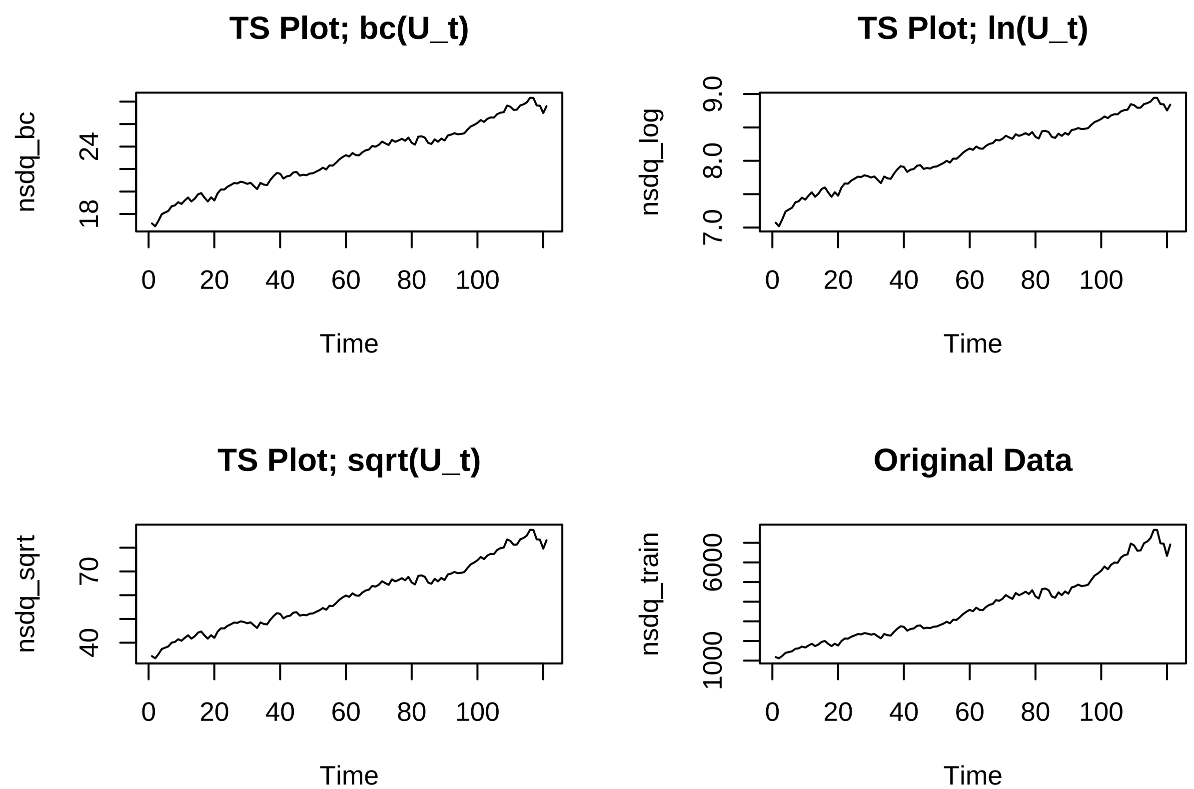



```
## [1] "Lambda = 0.222222"
```

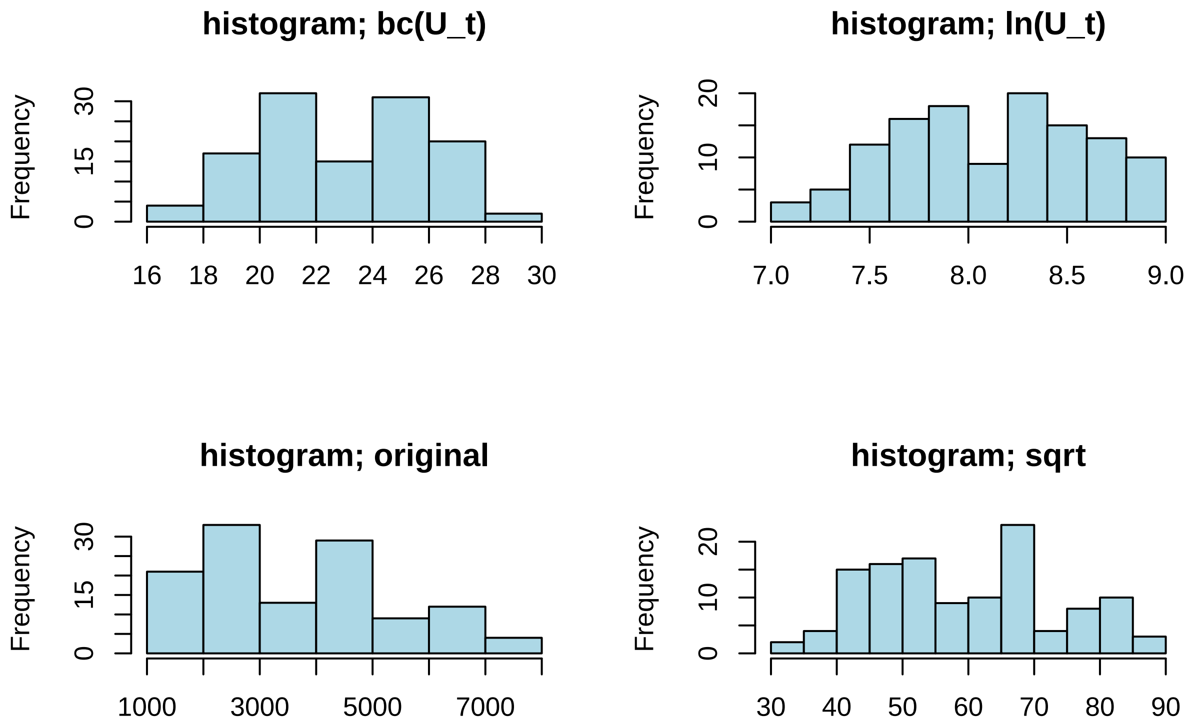
The above Boxcox graph suggests a transformation since 1 is not in the confidence interval for lambda. $\lambda = 0.222222$. So, a suitable transformation is $Y_t = \frac{1}{\lambda}(X_t^{\lambda} - 1) \implies Y_t = \frac{1}{0.222222}(X_t^{0.222222} - 1)$. The Boxcox transformation can be compared with the log and square root transformation.

```
##
##  Augmented Dickey-Fuller Test
##
## data:  nsdq_train
## Dickey-Fuller = -1.4554, Lag order = 4, p-value = 0.8028
## alternative hypothesis: stationary
```

From the Augmented Dickey-Fuller Test we can see that the $p - value = 0.8028 > 0.05$. So, the data is not stationary.

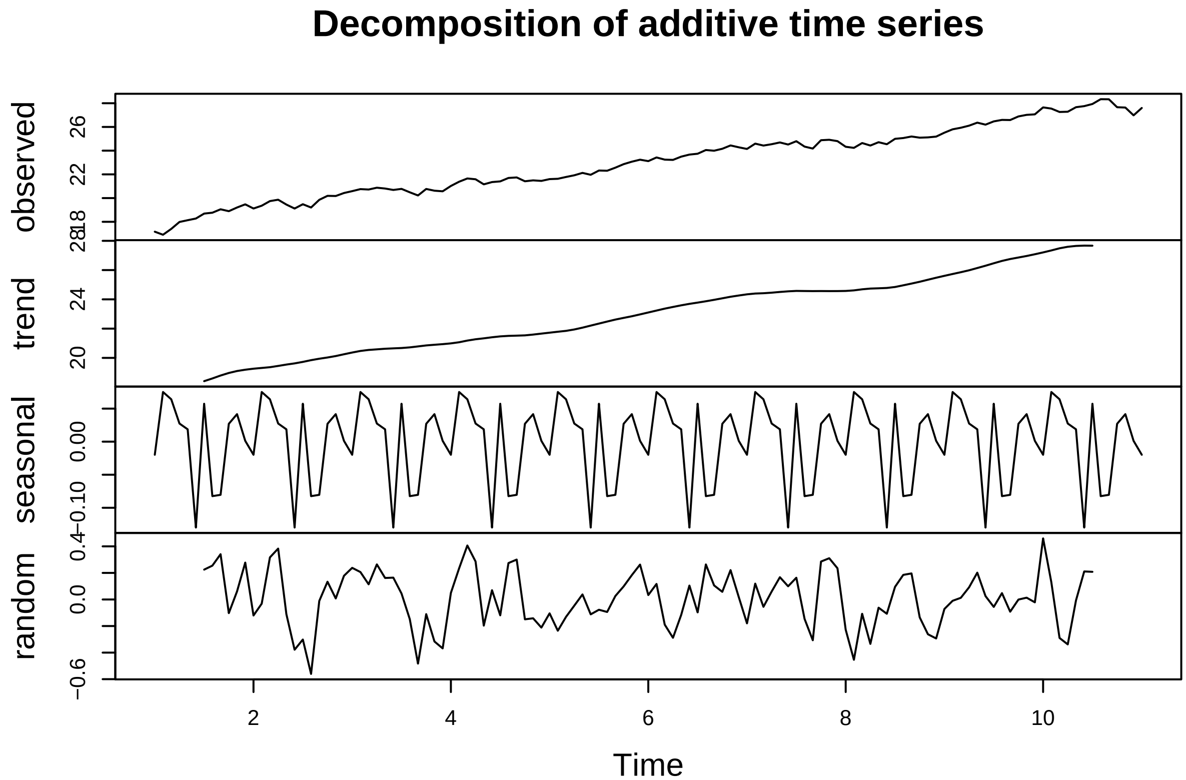


From the time series plots above, the boxcox and log transformations have more stable variances.



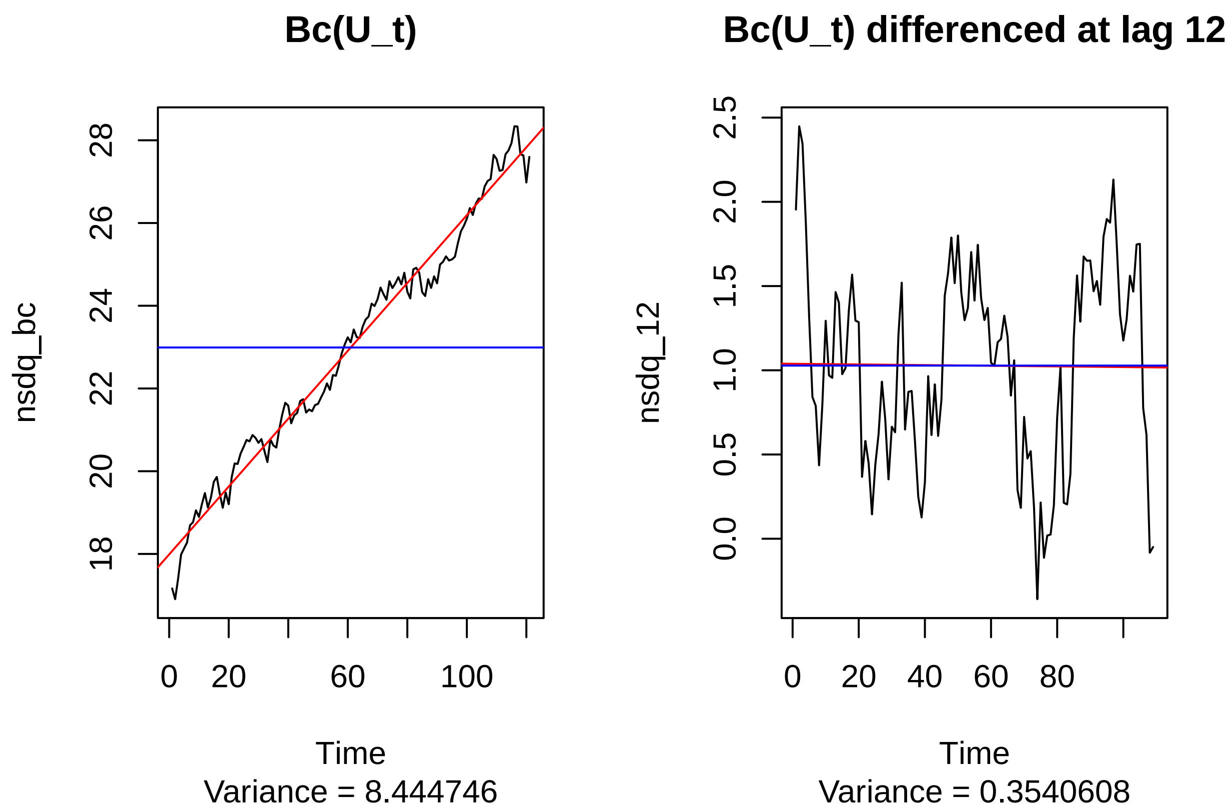
Both the boxcox and log transformation resemble the normal distribution. The log transformation is left skewed, whereas the boxcox transformation resembles the normal distribution the most and is also the most symmetric. So, the boxcox transformation is the most appropriate transformation.

Now let's produce a decomposition of the boxcox transformed data, $boxcox(U_t)$, to check for seasonality and a trend.

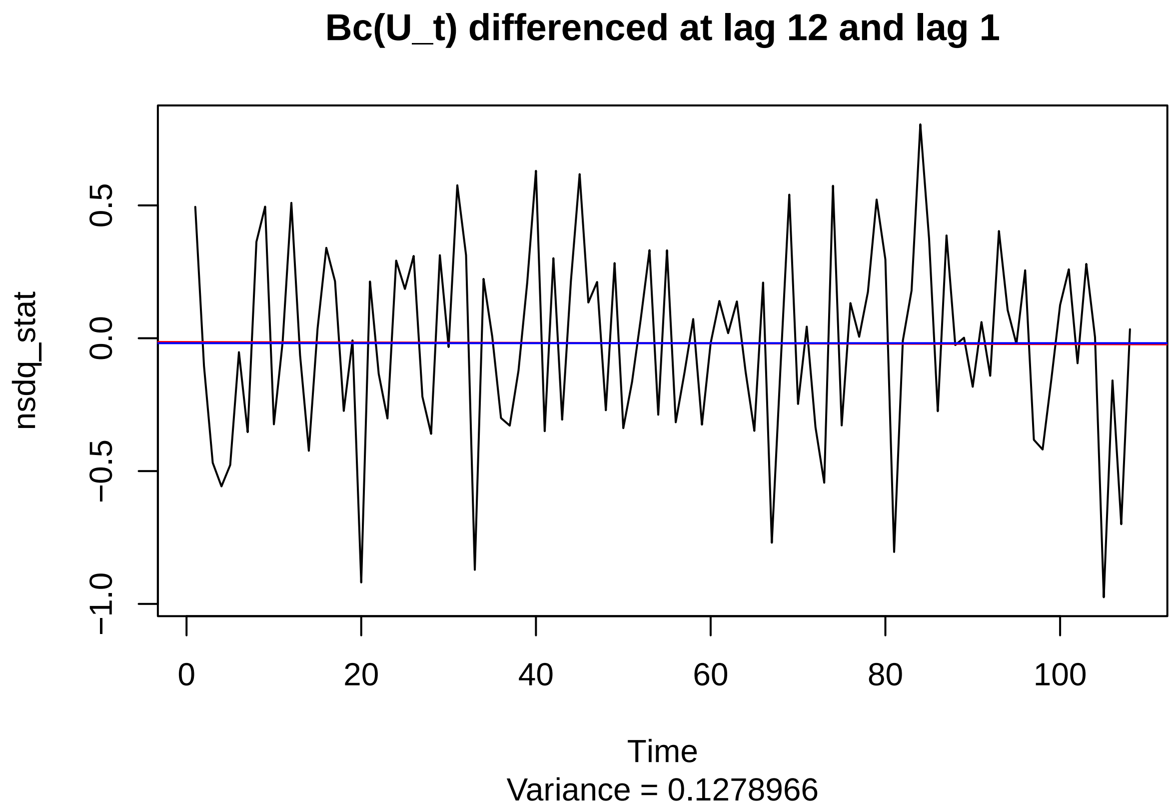


From the decomposition above, there seems to be an upward trend and seasonality. So, the trend and seasonality in the data need to be removed before the time series data can be forecasted.

Differencing



After differencing at lag 12 there is no longer seasonality and the variance has reduced. However there still seems to be a slight trend. To get rid of this trend we can difference again at lag 1.



After differencing at lag 12 and then at lag 1, the data is no longer seasonal and there is no apparent trend in the data. In addition, the variance has also reduced significantly after differencing.

Now let's confirm our data is stationary using the Augmented Dickey Fuller test.

Augmented Dickey Fuller test for boxcox transformed data:

```
##
##  Augmented Dickey-Fuller Test
##
## data:  nsdq_bc
## Dickey-Fuller = -3.0892, Lag order = 4, p-value = 0.1241
## alternative hypothesis: stationary
```

At the $\alpha = 0.05$ significance level, the data is not stationary as the p value is larger than 0.05.

```
##
##  Augmented Dickey-Fuller Test
##
## data:  nsdq_12
## Dickey-Fuller = -2.5141, Lag order = 4, p-value = 0.3634
## alternative hypothesis: stationary
```

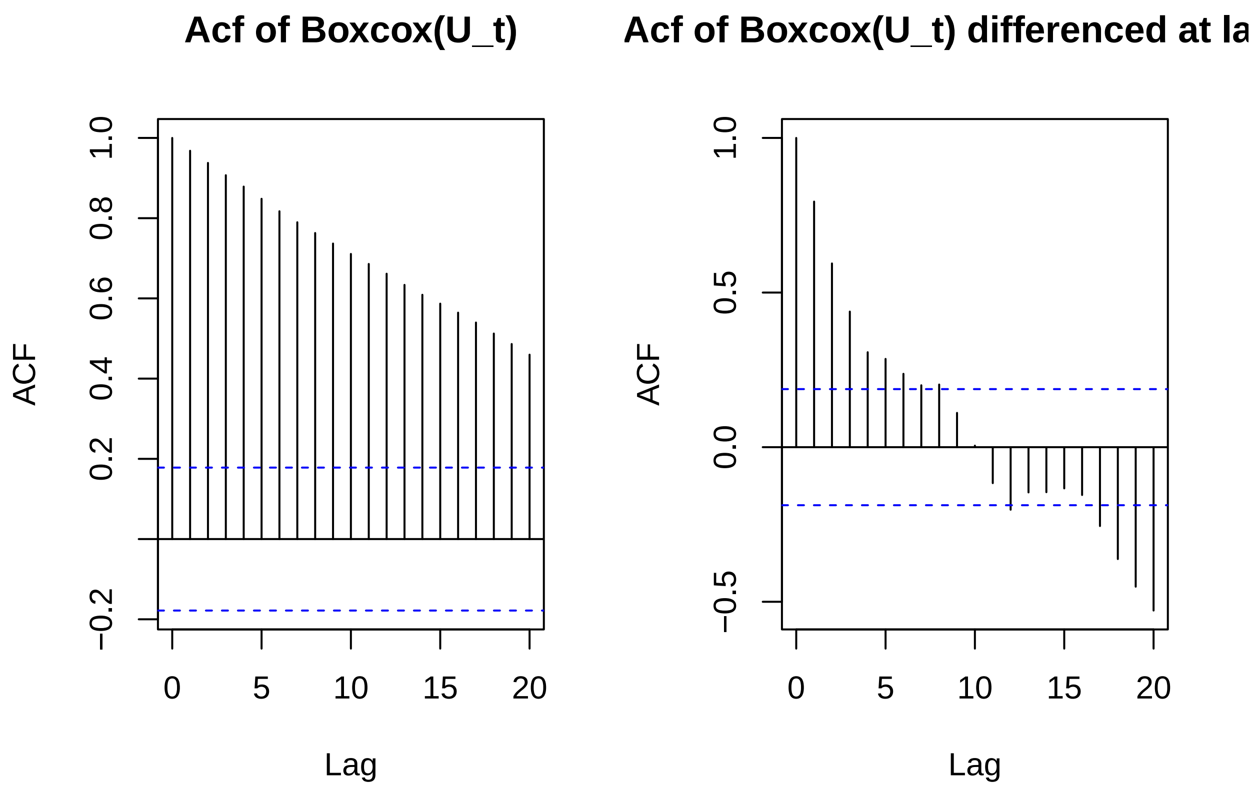
At the $\alpha = 0.05$ significance level, the data is still not stationary as the p value is larger than 0.05.

```
##
##  Augmented Dickey-Fuller Test
```

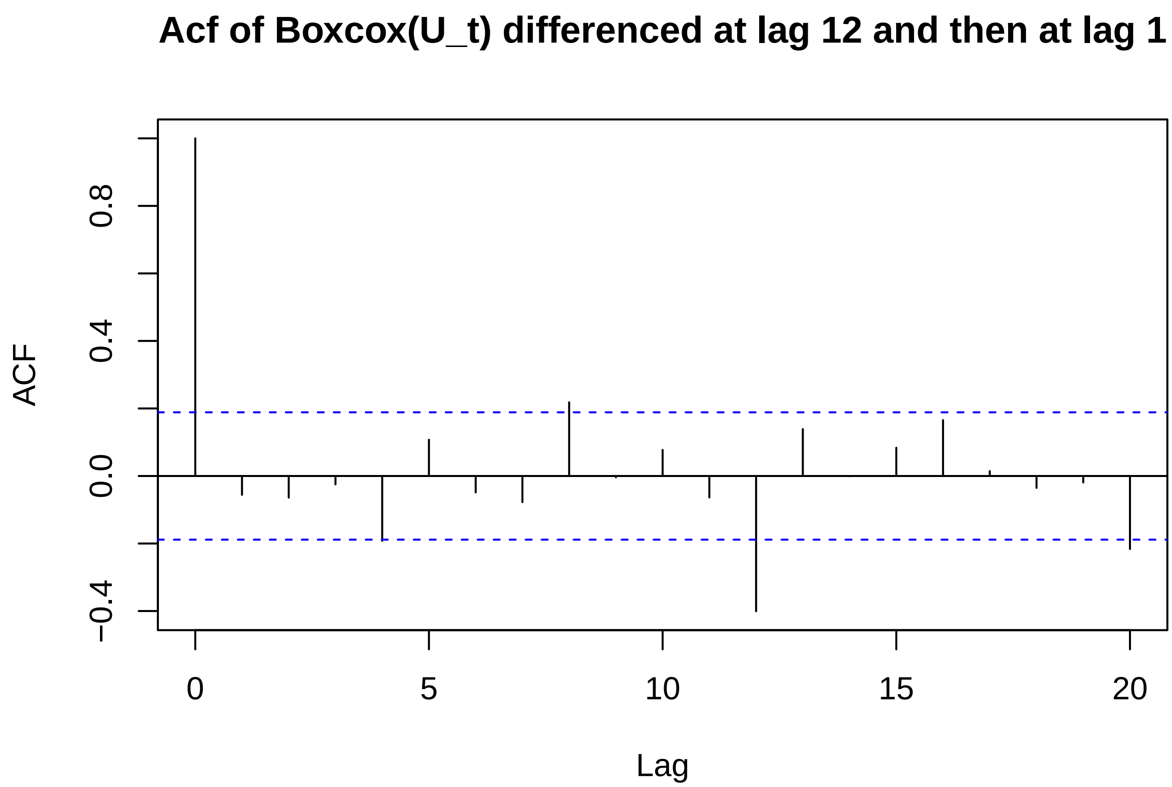
```
##
## data:  nsdq_stat
## Dickey-Fuller = -4.8838, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

At the $\alpha = 0.05$ significance level, the data is finally stationary as the p value is less than 0.05.

Now let's look at the ACFs before and after differencing:

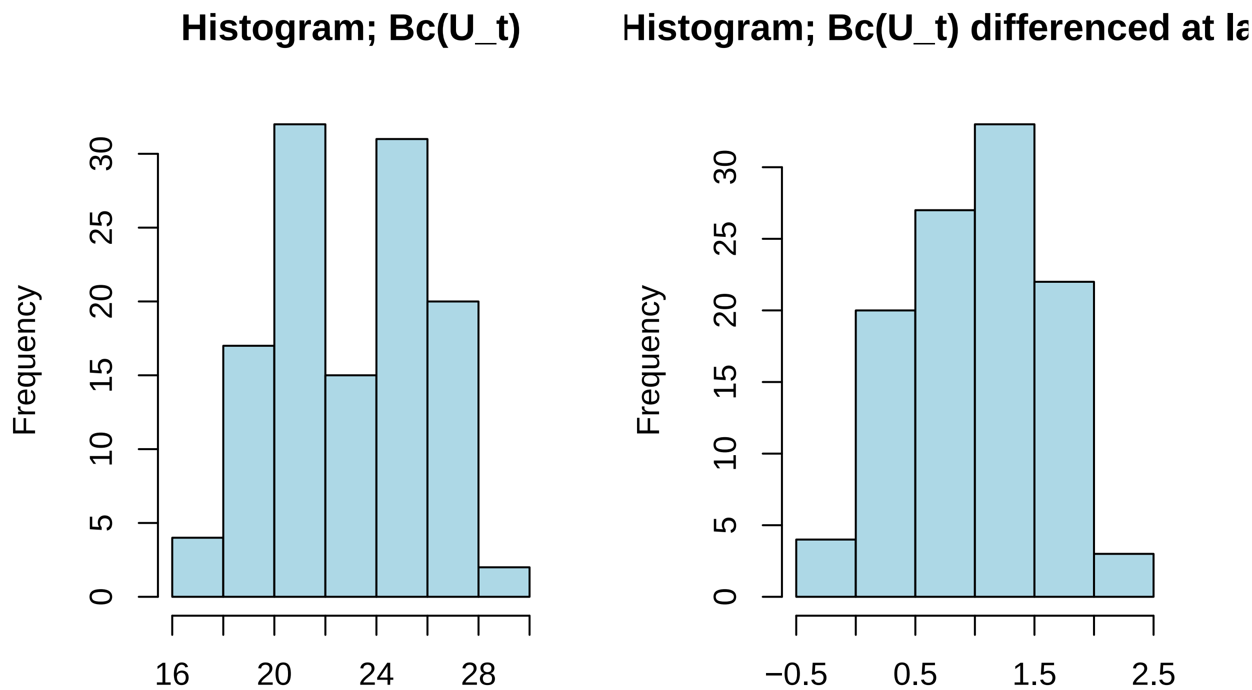


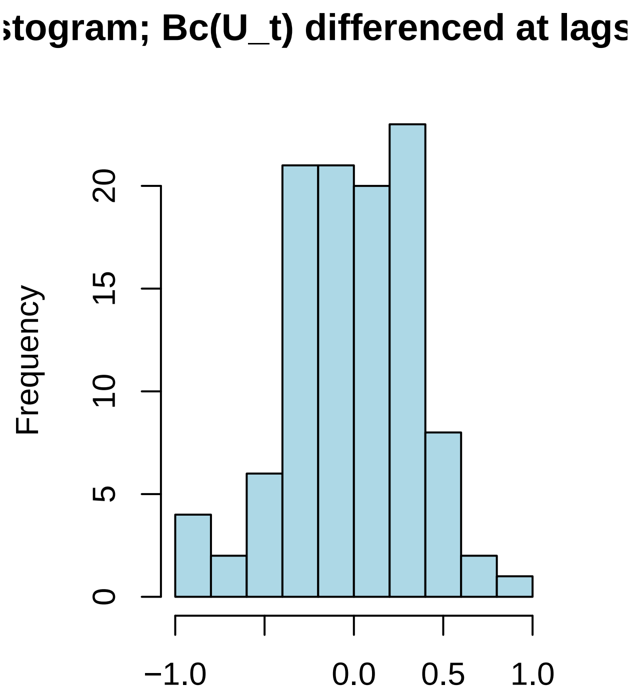
The ACF on the left decays slow decay and there is clearly seasonality. The seasonality has reduced after differencing at lag 12. So, let's difference again at lag 1.



The ACF quickly decays to zero, therefore corresponding to a stationary process with no trend or seasonality.

Now let's look at the distributions of our differenced data.

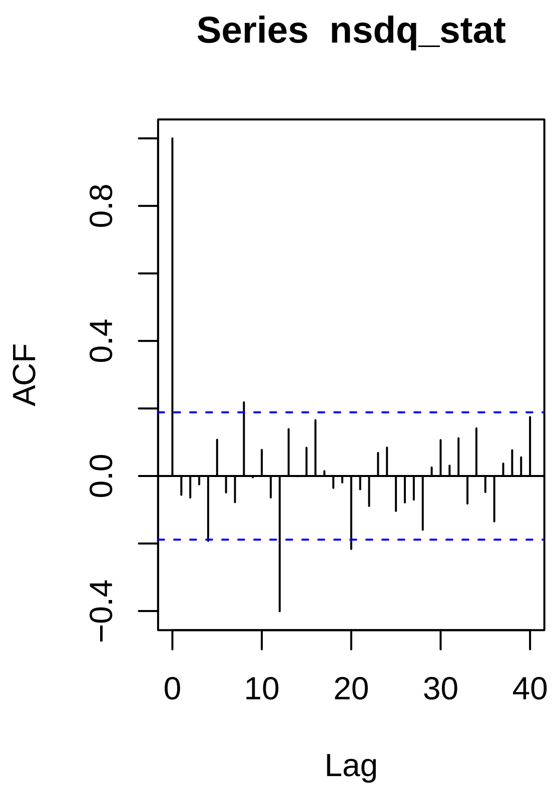

Histogram; Bc(U_t)
Histogram; Bc(U_t) differenced at la
Frequency
0
5
10
15
20
25
30
16
20
24
28
Frequency
0
5
10
15
20
25
30
−0.5
0.5
1.5
2.5

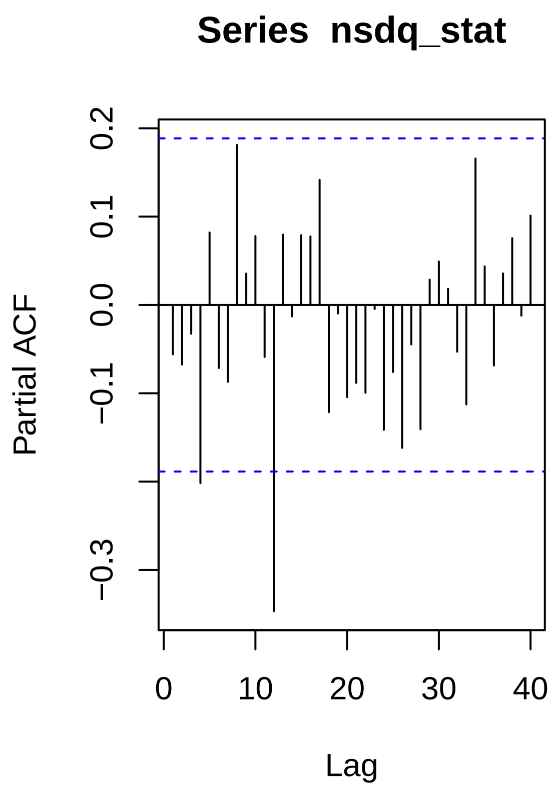


Comparing these distributions, we can see that the distribution of the twice differenced data follows a more Gaussian distribution compared to the other histograms. In addition, its ACF follows a more stationary process as mentioned. Therefore, it's an appropriate to use the twice differenced data to build our model, $\nabla_1\nabla_{12}boxcox(U_t)$.

Model Selection

Now that the data has been transformed and made stationary, let's try selecting a suitable model to forecast the data. This can be done by analyzing the ACF and PACF of the new data.





From the ACF plot there seems to be significant lags which go outside the confidence interval at lags 0,8,12,20.

From the PACF plot there seems to be significant lags which go outside the confidence intervals at lags 4, 12.

With this information let's try to come up with some candidate models. Since there was seasonality in our data a SARIMA model for $boxcox(U_t)$ is appropriate.

Let's start with modeling the seasonal part of our model, (P, D, Q):

- We applied one seasonal differencing, so D = 1 at lag = 12.
- The ACF shows a small peaks at lag = 8 and lag = 20, and a large peaks at lag = 0 and lag = 12. A good choice for the moving average (MA) part of the model is Q = 0 or Q = 1.
- The PACF shows a small peaks at lags 4 and a strong peak at lag 12. A good choice for the autoregressive(AR) part of the model is P = 0 or P = 1.

Now, let's set up the non-seasonal part of our model, (p, d, q):

- We also differenced at lag 1 to remove the trend, so d = 1.
- ACF shows peaks at lag = 8 and lag = 12. A good choice for the moving average (MA) part is q = 0 or q = 4.
- PACF has a small peak at lag 4. A good choice for the autoregressive(AR) part is p = 0 or p = 4.

Now, let's test these models and choose one with the lowest Second-order Akaike Information Criterion (AICc).

We'll be testing SARIMA : P = 0 or 1; D = 1; Q = 0 or 1; p = 0 or 4; d = 1; q = 0 or 4; with s = 12; where s is the period.

```
##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 0), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          sar1     sma1
##       -0.0192  -0.7443
## s.e.   0.1515   0.1489
##
## sigma^2 estimated as 0.08479:  log likelihood = -24.99,  aic = 55.99
```

```
##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 0), seasonal = list(order = c(1, 1, 1), period = 12),
##     fixed = c(0, NA), method = "ML")
##
## Coefficients:
##       sar1     sma1
##          0  -0.7562
## s.e.     0   0.1155
##
## sigma^2 estimated as 0.08469:  log likelihood = -25,  aic = 54
```

```
## [1] 56.08839
```

```
## [1] 54.10436
```

```
##
## Call:
## arima(x = nsdq_bc, order = c(4, 1, 0), seasonal = list(order = c(1, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1      ar2      ar3      ar4    sar1     sma1
##       -0.0872  -0.1250  -0.0408  -0.1813  0.0258  -0.8034
## s.e.   0.0970   0.1018   0.1022   0.1087  0.1500   0.1608
##
## sigma^2 estimated as 0.07985:  log likelihood = -22.75,  aic = 59.5
```

```
##
## Call:
## arima(x = nsdq_bc, order = c(4, 1, 0), seasonal = list(order = c(1, 1, 1), period = 12),
##     fixed = c(0, NA, 0, NA, 0, NA), method = "ML")
##
## Coefficients:
##       ar1      ar2  ar3      ar4  sar1     sma1
##         0  -0.1186    0  -0.1767     0  -0.7779
## s.e.    0   0.1019    0   0.1061     0   0.1257
##
## sigma^2 estimated as 0.08113:  log likelihood = -23.23,  aic = 54.47
```

```
## [1] 60.23423
```

```
## [1] 55.20344

##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 0), seasonal = list(order = c(1, 1, 0), period = 12),
##     method = "ML")
##
## Coefficients:
##          sar1
##       -0.4812
## s.e.   0.0901
##
## sigma^2 estimated as 0.09926:  log likelihood = -30.09,  aic = 64.17

## [1] 64.20825

##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 0), seasonal = list(order = c(0, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          sma1
##       -0.7562
## s.e.   0.1155
##
## sigma^2 estimated as 0.08469:  log likelihood = -25,  aic = 54

## [1] 54.03628

##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 4), seasonal = list(order = c(0, 1, 1), period = 12),
##     method = "ML")
##
## Coefficients:
##          ma1      ma2      ma3      ma4     sma1
##       -0.074  -0.1317  -0.0259  -0.1418  -0.7859
## s.e.   0.100   0.1018   0.1134   0.1002   0.1270
##
## sigma^2 estimated as 0.08039:  log likelihood = -22.93,  aic = 57.86

##
## Call:
## arima(x = nsdq_bc, order = c(0, 1, 4), seasonal = list(order = c(0, 1, 1), period = 12),
##     fixed = c(0, NA, 0, NA, NA), method = "ML")
##
## Coefficients:
##       ma1      ma2  ma3      ma4     sma1
##         0  -0.1334    0  -0.1425  -0.7723
## s.e.    0   0.0995    0   0.0949   0.1223
##
## sigma^2 estimated as 0.0814:  log likelihood = -23.28,  aic = 54.56
```

```
## [1] 58.3787
```

```
## [1] 55.08153
```

From the above results the the three models with the lowest AICc's are:

Model A : $SARIMA(0,1,0)(0,1,1)_{12}$, with AICc = 52.0404

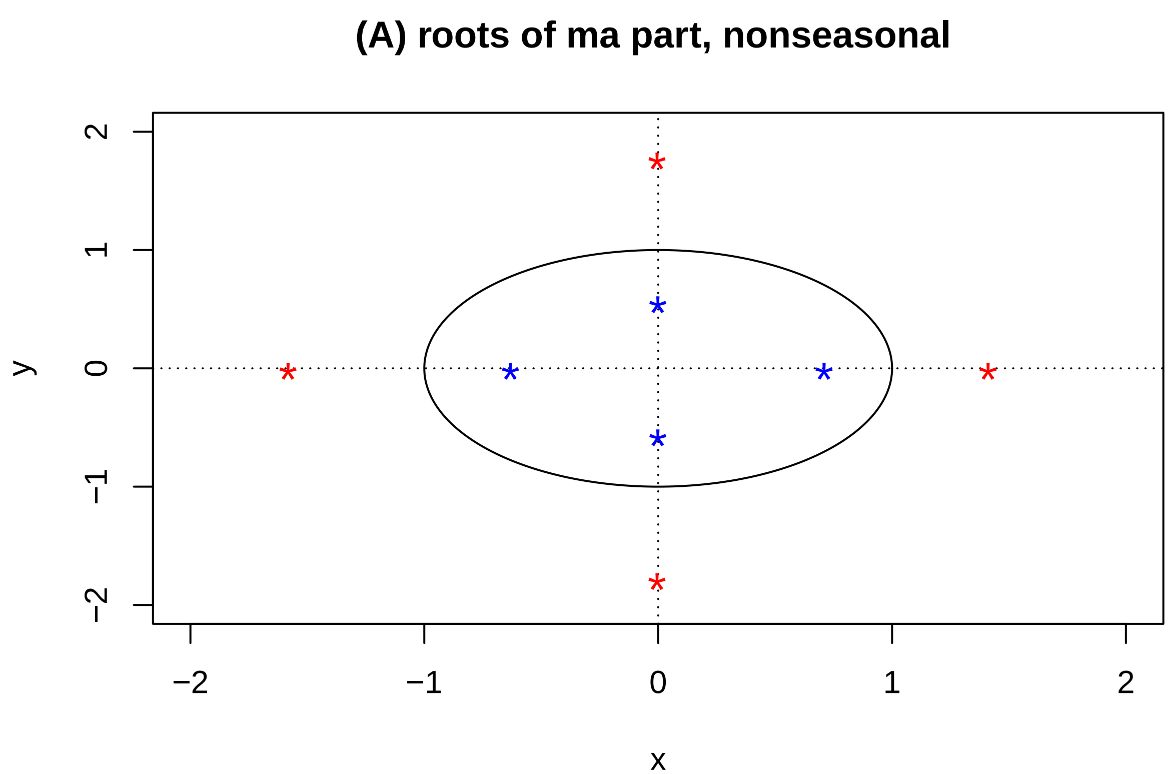
Model B : $SARIMA(0,1,0)(1,1,1)_{12}$, with AICc = 54.10098

Model C : $SARIMA(0,1,4)(0,1,1)_{12}$, with AICc = 56.44519

Invertibility

Model A is invertible since $|\Theta = -0.7562| < 1$. Model B is invertible since $|\Theta = -0.7443| < 1$

Checking Invertibility of Model C



Let's perform unit root tests for Model C to ensure invertibility. The red asterisks denote the roots, and the blue asterisks denote the inverse roots. From the above results, , all roots are outside the unit circle. For the seasonal Moving Average(MA) part $|\Theta = -0.7859| < 1$. So, Model C is invertible.

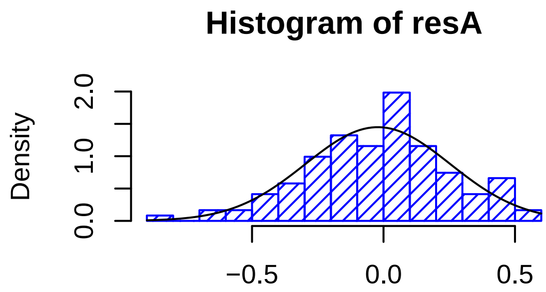
All 3 models are invertible. However, for diagnostic checking, we are going to move forward with Model A and B because of the principle of parsimony which states that a simpler model with fewer parameters is favored over more complex models with more parameters, provided the models fit the data similarly well.

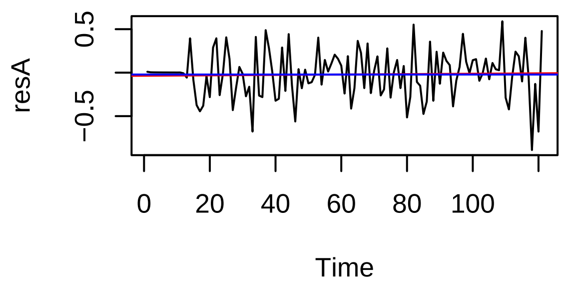
Diagnostic Checking

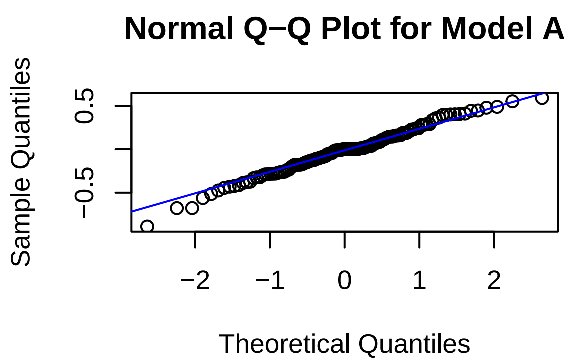
Diagnostic Checking for Model A

Recall that Model A is a $SARIMA(0,1,0)(0,1,1)_{12}$, with AICc = 52.0404.
We can write this model as: $\nabla_1\nabla_{12}boxcox(U_t) = (1 - 0.7562_{(0.1155)}B^{12})Z_t$, $\hat{\sigma}_z^2 = 0.08469$, where B is the backshift operator.

Let's check the distribution of the residuals.

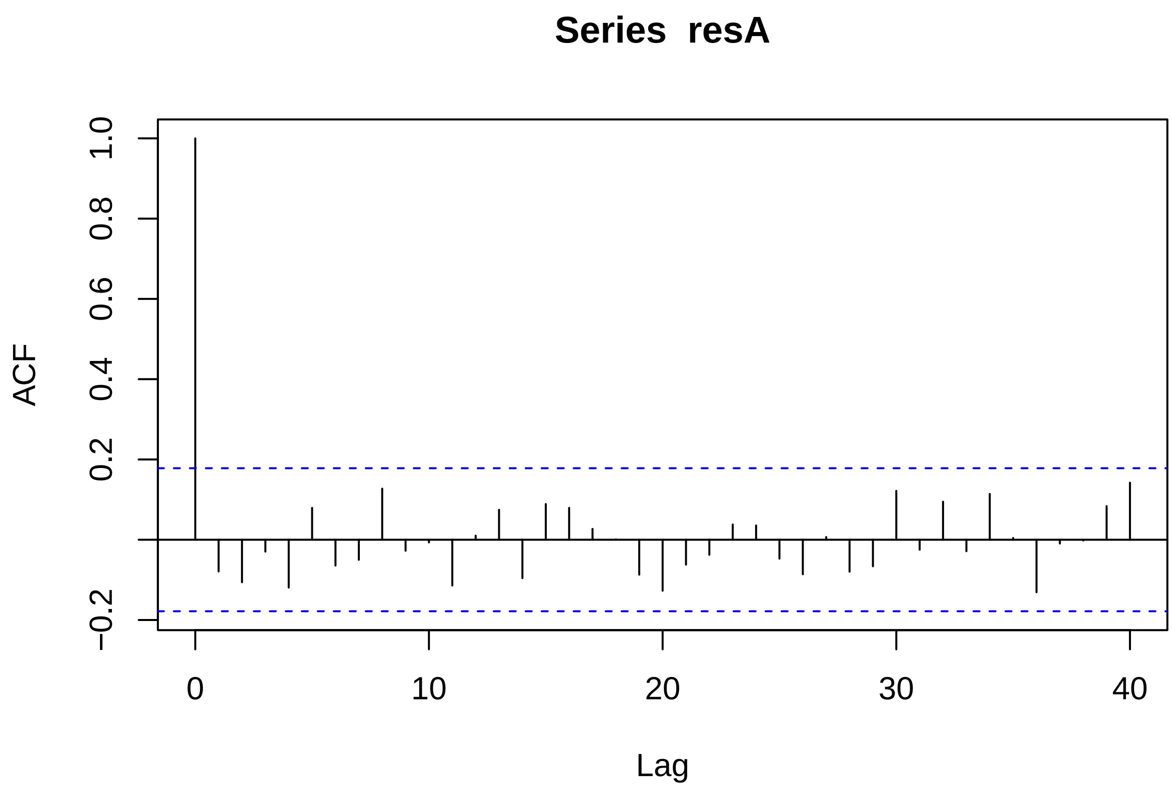


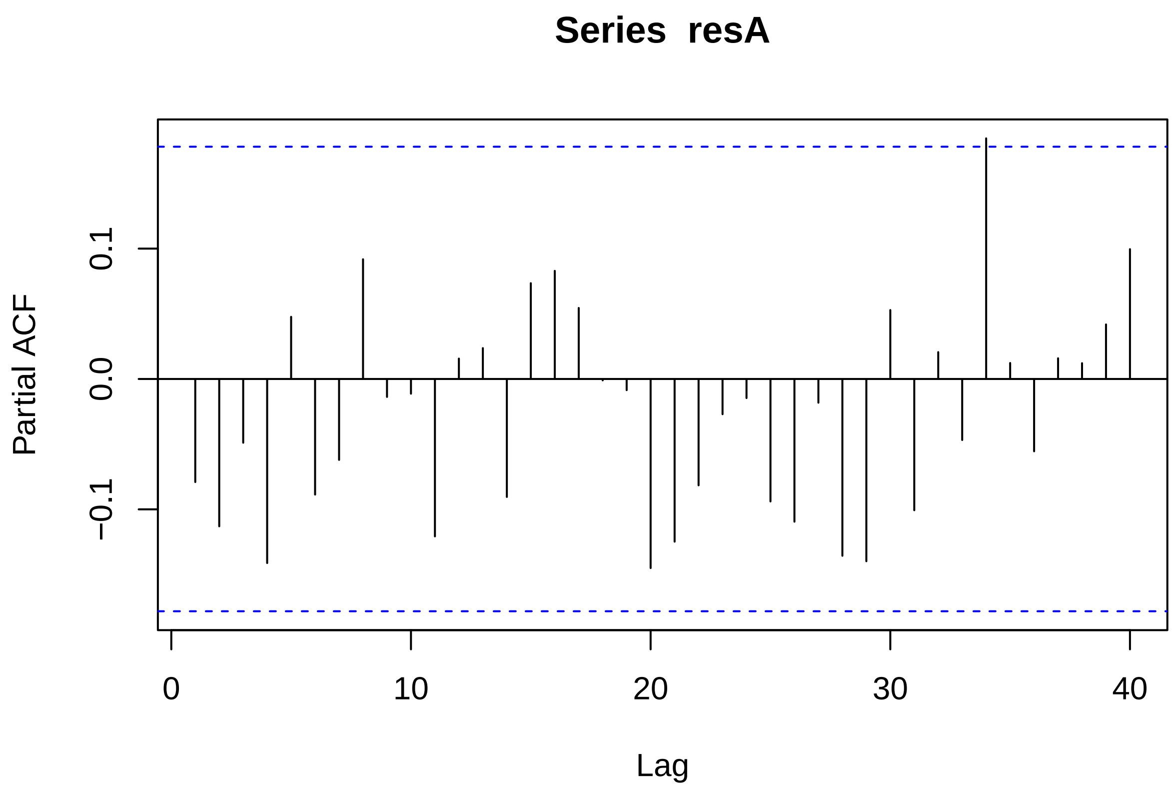




From the above graphs we can see there is no trend nor seasonality, and constant variance. The distribution is roughly normal, and the Q-Q plot looks good.

Now, let's check the ACF and PACF of the residuals.


Series resA
ACF
1.0
0.8
0.6
0.4
0.2
−0.2
0
10
20
30
40
Lag



All of the ACF and PACF residuals are within confidence intervals and can be counted as zeros.

Let's now run the following tests:

- Shapiro-Wilk normality test
- Box-Pierce Test
- Box-Ljung
- Mc-Leod Li test

```
##
##  Shapiro-Wilk normality test
##
## data:  resA
## W = 0.9894, p-value = 0.4758
```

```
##
##  Box-Pierce test
##
## data:  resA
## X-squared = 9.1374, df = 10, p-value = 0.5191
```

```
##
##  Box-Ljung test
##
## data:  resA
## X-squared = 9.7682, df = 10, p-value = 0.4611
```

```
##
##  Box-Ljung test
##
## data:  resA^2
## X-squared = 18.872, df = 11, p-value = 0.06342
```

All the tests reveal a p-value higher than 0.05, meaning model A passes all tests.

Let's check if our residuals follow Gaussian White Noise by using yule-walker estimation.

```
##
## Call:
## ar(x = resA, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.07578
```

Our fitted residuals correspond to AR(0), which is a white noise process.

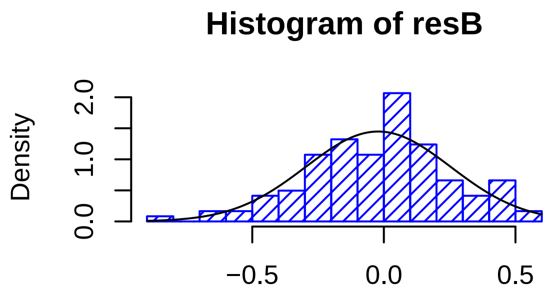
From the above results Model A passes the diagnostic checking stage. So, we can forecast future values with this model.

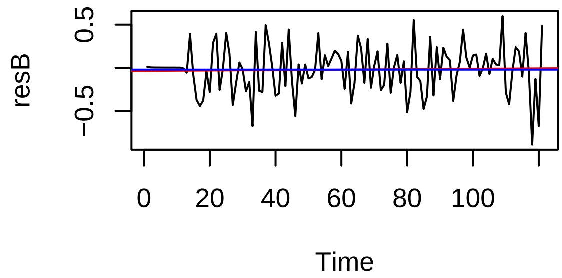
Diagnostic Checking for Model B

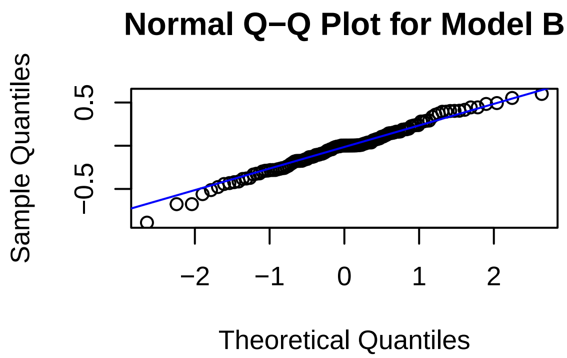
Recall that Model B is a $SARIMA(0,1,0)(1,1,1)_{12}$, with AICc = 54.10098

We can write this model as: $(1 - 0.0192_{(0.1515)}B^{12})\nabla_1\nabla_{12}boxcox(U_t) = (1 - 0.7443_{(0.1489)}B^{12})Z_t$, $\hat{\sigma}_z^2 = 0.08479$, where B is the backshift operator.

Let's check the distribution of the residuals.

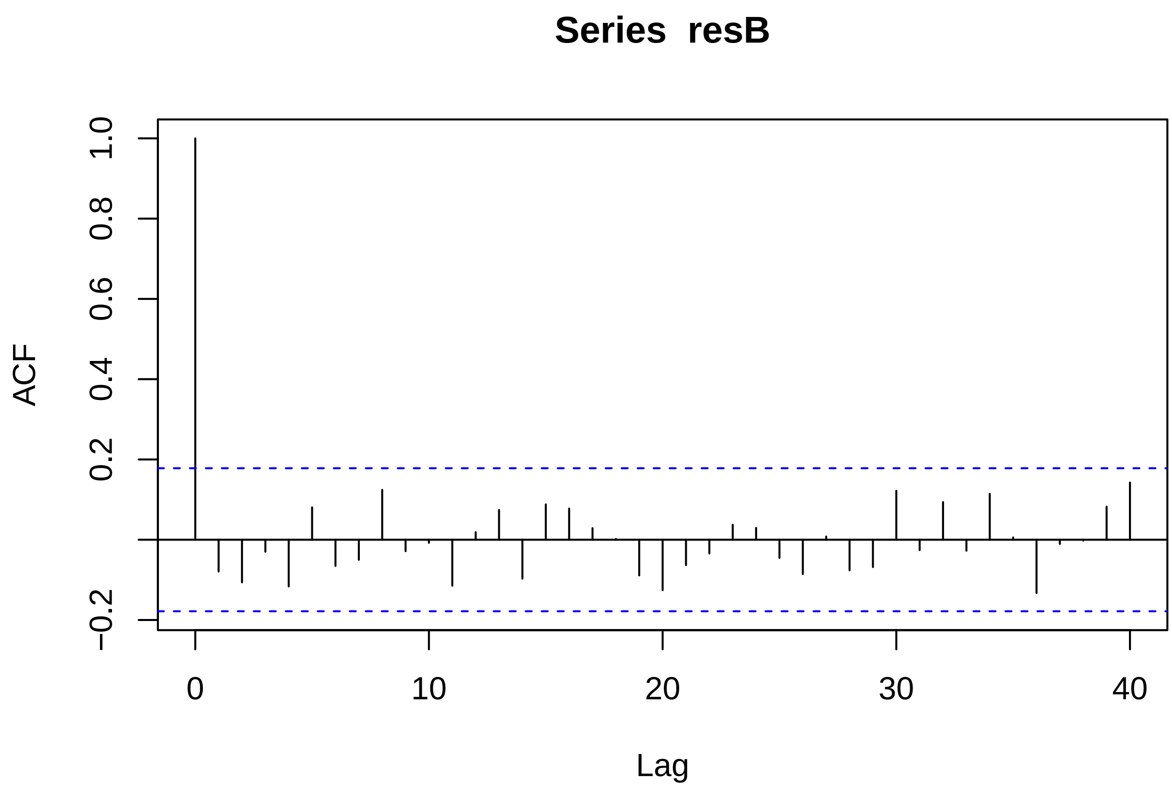


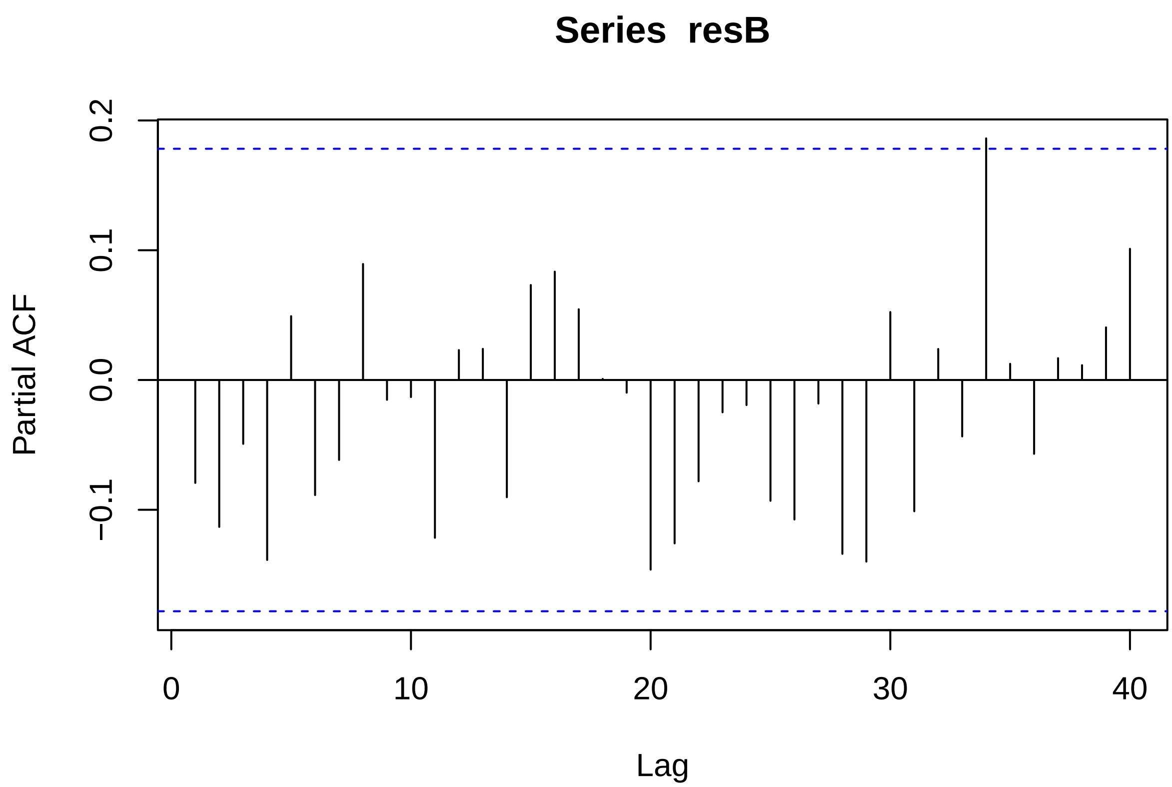




From the above graphs we can see there is no trend nor seasonality, and constant variance. The distribution is roughly normal, and the Q-Q plot looks good.

Now, let's check the ACF and PACF of the residuals.


Series resB
ACF
1.0
0.8
0.6
0.4
0.2
−0.2
0
10
20
30
40
Lag



All of the ACF and PACF residuals are within confidence intervals and can be counted as zeros.

Let's now run the following tests:

- Shapiro-Wilk normality test
- Box-Pierce Test
- Box-Ljung
- Mc-Leod Li test

```
##
##  Shapiro-Wilk normality test
##
## data:  resB
## W = 0.98969, p-value = 0.5012
```

```
##
##  Box-Pierce test
##
## data:  resB
## X-squared = 9.0406, df = 9, p-value = 0.4335
```

```
##
##  Box-Ljung test
##
## data:  resB
## X-squared = 9.6648, df = 9, p-value = 0.3783
```

```
##
##  Box-Ljung test
##
## data:  resB^2
## X-squared = 19.142, df = 11, p-value = 0.05859
```

All the tests reveal a p-value higher than 0.05, meaning model B passes all tests.

Let's check if our residuals follow Gaussian White Noise by using yule-walker estimation.

```
##
## Call:
## ar(x = resB, aic = TRUE, order.max = NULL, method = c("yule-walker"))
##
##
## Order selected 0  sigma^2 estimated as  0.07587
```

Our fitted residuals correspond to AR(0), which is a white noise process.

From the above results Model B passes the diagnostic checking stage. So, we can forecast future values with this model.

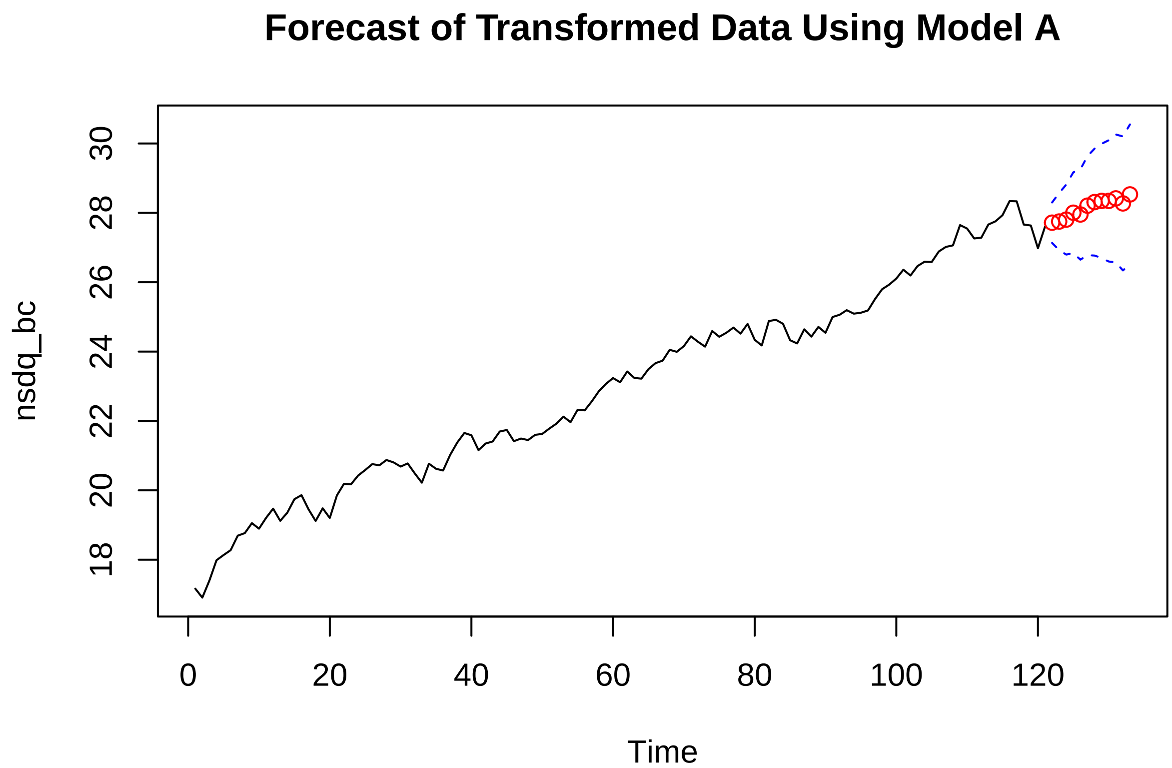
Summary Of Time Series Analysis

After transforming and differencing our data, checking for stationarity, picking candidate models from the ACFs and PACFs of our data, conducting unit root tests for invertibility, and performing diagnostic checking on our data, we can that Model A and Model B are suitable for forecasting. However, Model A has a lower AICc value and one less parameter than Model B. So, using the principle of parsimony and the AICc value, we will use Model A for forecasting.

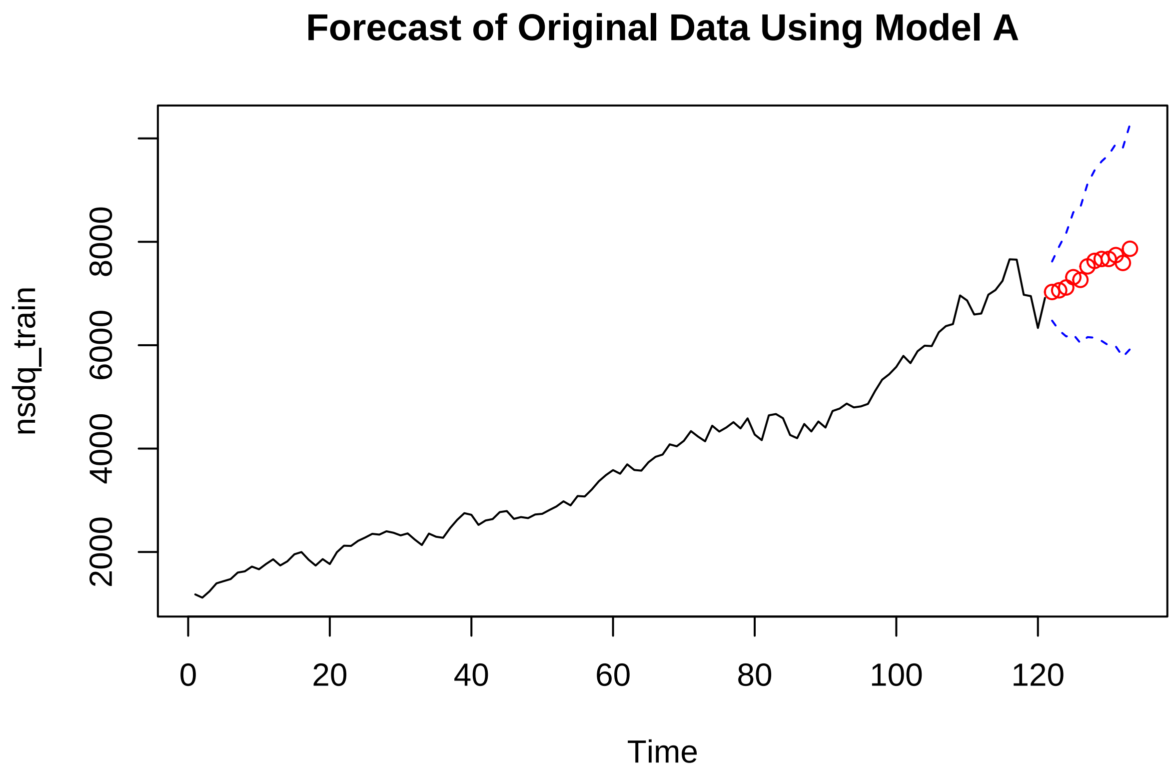
Model A is modeled as $SARIMA(0,1,0)(0,1,1)_{12}$, and can be written as $\nabla_1\nabla_{12}boxcox(U_t) = (1 - 0.7562_{(0.1155)}B^{12})Z_t$, where B is the backshift operator.

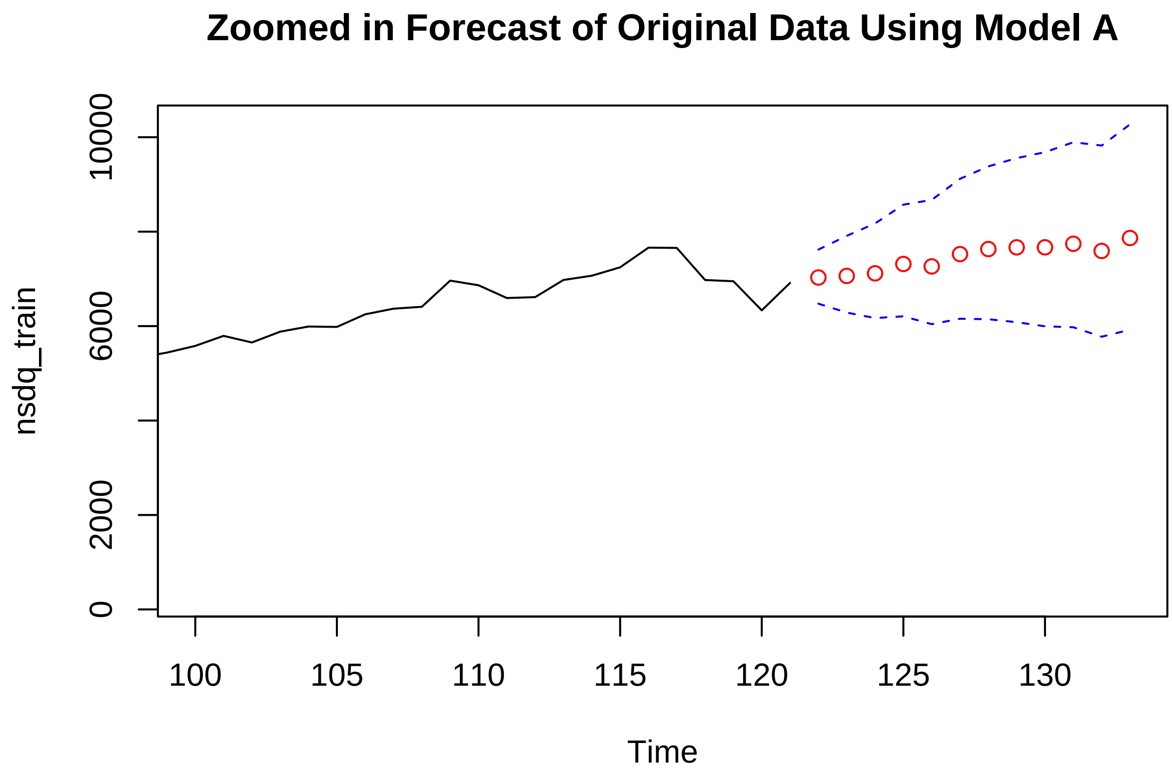
Forecasting

Let's forecast the next 12 observations, which are months. First, let's forecast values for the box-cox transformed data. Then, let's forecast values for the training data and see how these values performed using our testing data.

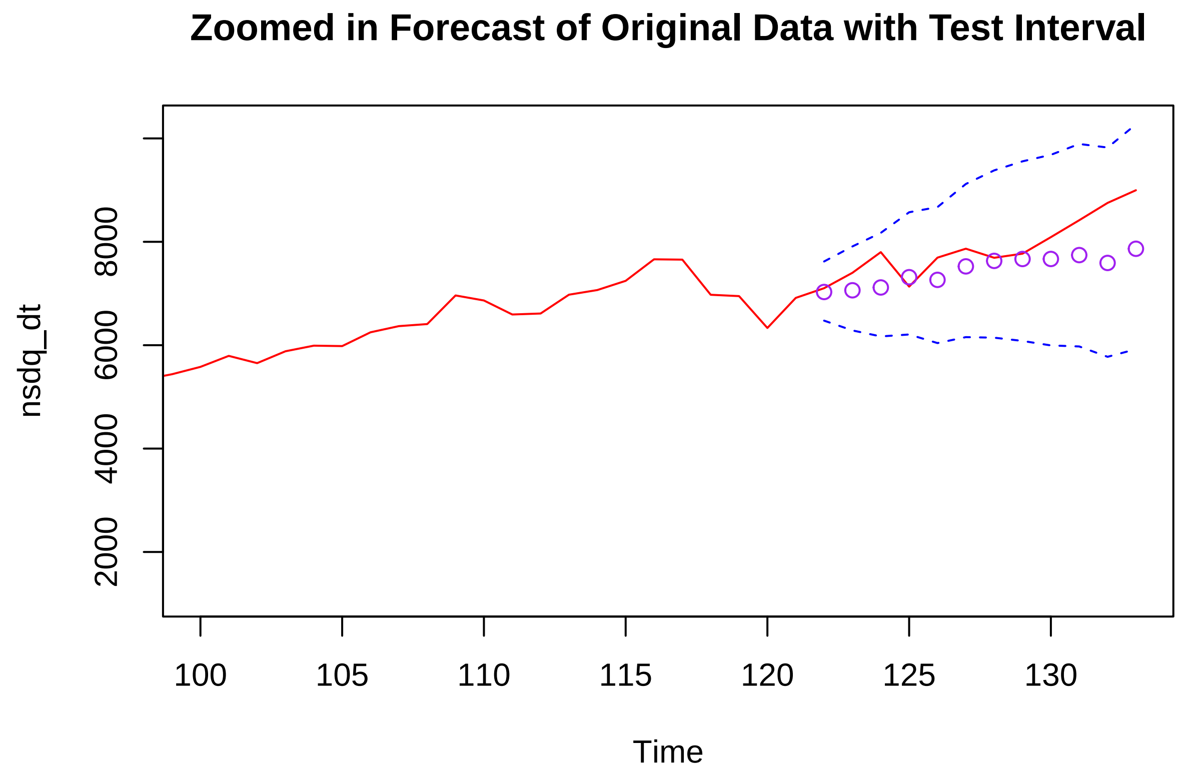


Forecast of Transformed Data Using Model A
nsdq_bc
18
20
22
24
26
28
30
0
20
40
60
80
100
120
Time


Forecast of Original Data Using Model A
nsdq_train
2000
4000
6000
8000
0
20
40
60
80
100
120
Time



Zoomed in Forecast of Original Data Using Model A
nsdq_train
0
2000
6000
10000
100
105
110
115
120
125
130
Time



I forecasted 12 observations(months) using the confidence intervals of Model A. We can see that the original line is within the confidence intervals of the forecasted values, and the points seem to do a moderately good job at following the test set.

Conclusion

Since we are predicting the stock market, it is very difficult to have near perfect forecasting. By transforming the data and applying practical time series methods to the data, we were able to do a reasonable job at forecasting future prices of the Nasdaq 100 Futures stock. Although the forecasted values are not perfect, investors can get a general idea of the movement of the stock and make investment decisions from the Model A, denoted as the $SARIMA(0,1,0)(0,1,1)_{12}$ model, which can also be written as: $\nabla_1\nabla_{12}boxcox(U_t) = (1 - 0.7562_{(0.1155)}B^{12})Z_t$.

References

https://www.investopedia.com/ask/answers/what-do-sp-500-dow-and-nasdaq-futures-contracts-represent/

Data: https://www.investing.com/indices/nq-100-futures-historical-data

Appendix: All code for this report

```
knitr::opts_chunk$set(echo = FALSE)
options(rgl.useNULL = TRUE)
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
#Loading required Libraries
library(tsdl)
library(forecast)
library(tseries)
library(TTR)
library(seastests)
library(MASS)
library(ggplot2)
library(ggfortify)
library(qpcR)
# Read file and load into R
nsdq <- read.csv("Nasdaq_100.csv")
nsdq_dt <-  as.numeric(gsub(",","",nsdq$Price))

# Plot Raw Data
ts.plot(nsdq_dt, main = "Raw Data")

# Divide data into a training set and a test set
nsdq_train <- nsdq_dt[c(1:121)]
nsdq_test <- nsdq_dt[c(122:133)]

# Plot Training Data
ts.plot(nsdq_train, main = "Training data")
#Boxcox graph
bcTransform <- boxcox(nsdq_train~ as.numeric(1:length(nsdq_train)))
#bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
#lambda Value
lambda <- bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
sprintf("Lambda = %f",lambda)
#Checking the stationarity of the training data
adf.test(nsdq_train)

#Boxcox transformation
nsdq_bc <- (1/lambda)*(nsdq_train^lambda-1)
#Log transformation
nsdq_log <- log(nsdq_train)
#Square root transformation
nsdq_sqrt <- sqrt(nsdq_train)
par(mfrow = c(2, 2))
ts.plot(nsdq_bc, main = "TS Plot; bc(U_t)")
ts.plot(nsdq_log, main = "TS Plot; ln(U_t)")
ts.plot(nsdq_sqrt, main = "TS Plot; sqrt(U_t)")
ts.plot(nsdq_train, main = "Original Data")
#Comparing the histograms of the Boxcox, Log and Square root transformations to
#histogram of the training data
par(mfrow = c(2, 2))
hist(nsdq_bc, col="light blue", xlab="", main="histogram; bc(U_t)")
hist(nsdq_log, col="light blue", xlab="", main="histogram; ln(U_t)")
```

```
hist(nsdq_train, col="light blue", xlab="", main="histogram; original")
hist(nsdq_sqrt, col="light blue", xlab="", main="histogram; sqrt")
#Decomposition of the Boxcox transformed data
y <- ts(as.ts(nsdq_bc), frequency = 12)
decomp <- decompose(y)
plot(decomp)
#Differencing Boxcox transformed data at lag 12 once
nsdq_12 <- diff(nsdq_bc,12)

#Differencing again at lag 1
nsdq_stat <- diff(nsdq_12,lag=1,differences=1)

par(mfrow = c(1, 2))
plot.ts(nsdq_bc, main="Bc(U_t)", sub = "Variance = 8.444746")
fit <- lm(nsdq_bc ~ as.numeric(1:length(nsdq_bc)))
abline(fit, col="red")
abline(h=mean(nsdq_bc), col="blue")

plot.ts(nsdq_12, main="Bc(U_t) differenced at lag 12", sub = "Variance = 0.3540608")
fit1 <- lm(nsdq_12 ~ as.numeric(1:length(nsdq_12)))
abline(fit1, col="red")
abline(h=mean(nsdq_12), col="blue")
plot.ts(nsdq_stat, main="Bc(U_t) differenced at lag 12 and lag 1", sub = "Variance = 0.1278966")
fit <- lm(nsdq_stat ~ as.numeric(1:length(nsdq_stat)))
abline(fit, col="red")
abline(h=mean(nsdq_stat), col="blue")
#Adf test for boxcox transformed data
adf.test(nsdq_bc)
#Adf test for boxcox transformed data differenced at lag 12
adf.test(nsdq_12)
adf.test(nsdq_stat)

par(mfrow = c(1, 2))
#Acf of Boxcox transformed data
acf(nsdq_bc, main = "Acf of Boxcox(U_t)")

#Acf of data after differencing at lag 12 once
acf(nsdq_12, main = "Acf of Boxcox(U_t) differenced at lag 12" )

#Acf of data after differencing at lag 12 and at lag 1
acf(nsdq_stat, main = "Acf of Boxcox(U_t) differenced at lag 12 and then at lag 1" )
par(mfrow = c(1, 2))
hist(nsdq_bc, col="light blue", xlab="", main="Histogram; Bc(U_t)")
hist(nsdq_12, col="light blue", xlab="", main="Histogram; Bc(U_t) differenced at lag 12")
hist(nsdq_stat, col="light blue", xlab="", main="Histogram; Bc(U_t) differenced at lags 12 & 1")

par(mfrow = c(1, 2))
acf(nsdq_stat,lag.max=40)
pacf(nsdq_stat,lag.max=40)
arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,1),period=12),method = "ML")
arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,1),period=12),fixed = c(0,NA),method = "M
AICc(arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,1),period=12),method = "ML"))
AICc(arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,1),period=12),fixed = c(0,NA),metho
```

```
arima(nsdq_bc, order = c(4,1,0), seasonal = list(order = c(1,1,1),period=12),method = "ML")
arima(nsdq_bc, order = c(4,1,0), seasonal = list(order = c(1,1,1),period=12),fixed = c(0,NA,0,NA,0,NA),m
AICc(arima(nsdq_bc, order = c(4,1,0), seasonal = list(order = c(1,1,1),period=12),method = "ML"))
AICc(arima(nsdq_bc, order = c(4,1,0), seasonal = list(order = c(1,1,1),period=12),fixed = c(0,NA,0,NA,0
arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,0),period=12),method = "ML")
AICc(arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,0),period=12),method = "ML"))
arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(0,1,1),period=12),method = "ML")
AICc(arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(0,1,1),period=12),method = "ML"))
arima(nsdq_bc, order = c(0,1,4), seasonal = list(order = c(0,1,1),period=12),method = "ML")
arima(nsdq_bc, order = c(0,1,4), seasonal = list(order = c(0,1,1),period=12),fixed = c(0,NA,0,NA,NA),me
AICc(arima(nsdq_bc, order = c(0,1,4), seasonal = list(order = c(0,1,1),period=12),method = "ML"))
AICc(arima(nsdq_bc, order = c(0,1,4), seasonal = list(order = c(0,1,1),period=12),fixed = c(0,NA,0,NA,N
source("plot.roots.R")
plot.roots(NULL,polyroot(c(1, -0.074,-0.1317, -0.0259,-0.1418)), main="(A) roots of ma part, nonseasona
fitA <- arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(0,1,1),period=12),method = "ML")
resA <- residuals(fitA)
par(mfrow = c(2, 2))
hist(resA,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(resA)
std <- sqrt(var(resA))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(resA)
fittA <- lm(resA ~ as.numeric(1:length(resA))); abline(fittA, col="red")
abline(h=mean(resA), col="blue")
qqnorm(resA,main= "Normal Q-Q Plot for Model A")
qqline(resA,col="blue")
acf(resA, lag.max=40)
pacf(resA, lag.max=40)

shapiro.test(resA)
Box.test(resA, lag = 11, type = c("Box-Pierce"), fitdf = 1)
Box.test(resA, lag = 11, type = c("Ljung-Box"), fitdf = 1)
Box.test(resA^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)
ar(resA, aic = TRUE, order.max = NULL, method = c("yule-walker"))
fitB <- arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(1,1,1),period=12),method = "ML")
resB <- residuals(fitB)
par(mfrow = c(2, 2))
hist(resB,density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m <- mean(resB)
std <- sqrt(var(resB))
curve( dnorm(x,m,std), add=TRUE )
plot.ts(resB)
fittB <- lm(resB ~ as.numeric(1:length(resB))); abline(fittB, col="red")
abline(h=mean(resB), col="blue")
qqnorm(resB,main= "Normal Q-Q Plot for Model B")
qqline(resB,col="blue")
acf(resB, lag.max=40)
pacf(resB, lag.max=40)
shapiro.test(resB)
Box.test(resB, lag = 11, type = c("Box-Pierce"), fitdf = 2)
Box.test(resB, lag = 11, type = c("Ljung-Box"), fitdf = 2)
Box.test(resB^2, lag = 11, type = c("Ljung-Box"), fitdf = 0)
ar(resB, aic = TRUE, order.max = NULL, method = c("yule-walker"))
```

```r
fit.A<- arima(nsdq_bc, order = c(0,1,0), seasonal = list(order = c(0,1,1),period=12),method = "ML")
pred.tr <- predict(fit.A, n.ahead = 12)
U.tr= pred.tr$pred + 2*pred.tr$se
L.tr= pred.tr$pred - 2*pred.tr$se
ts.plot(nsdq_bc, xlim=c(1,length(nsdq_bc)+12), ylim = c(min(nsdq_bc),max(U.tr)),main = "Forecast of Tra
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(nsdq_bc)+1):(length(nsdq_bc)+12), pred.tr$pred, col="red")
library(forecast)
library(smooth)
pred.orig <- (pred.tr$pred * lambda + 1)^(1/lambda)
U= (U.tr * lambda + 1)^(1/lambda)
L= (L.tr* lambda + 1)^(1/lambda)
ts.plot(nsdq_train, xlim=c(1,length(nsdq_train)+12), ylim = c(min(nsdq_train),max(U)),main= "Forecast o
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(nsdq_train)+1):(length(nsdq_train)+12), pred.orig, col="red")

ts.plot(nsdq_train, xlim = c(100,length(nsdq_train)+12), ylim = c(250,max(U)),main="Zoomed in Forecast
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(nsdq_train)+1):(length(nsdq_train)+12), pred.orig, col="red")


ts.plot(nsdq_dt, xlim=c(100, length(nsdq_train)+ 12), ylim = c(min(nsdq_train),max(U)), col = "red", ma
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(nsdq_train)+1):(length(nsdq_train)+12), pred.orig, col=" purple")
#points((length(nsdq_train)+1):(length(nsdq_train)+12), nsdq_test, col = "black")
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
```