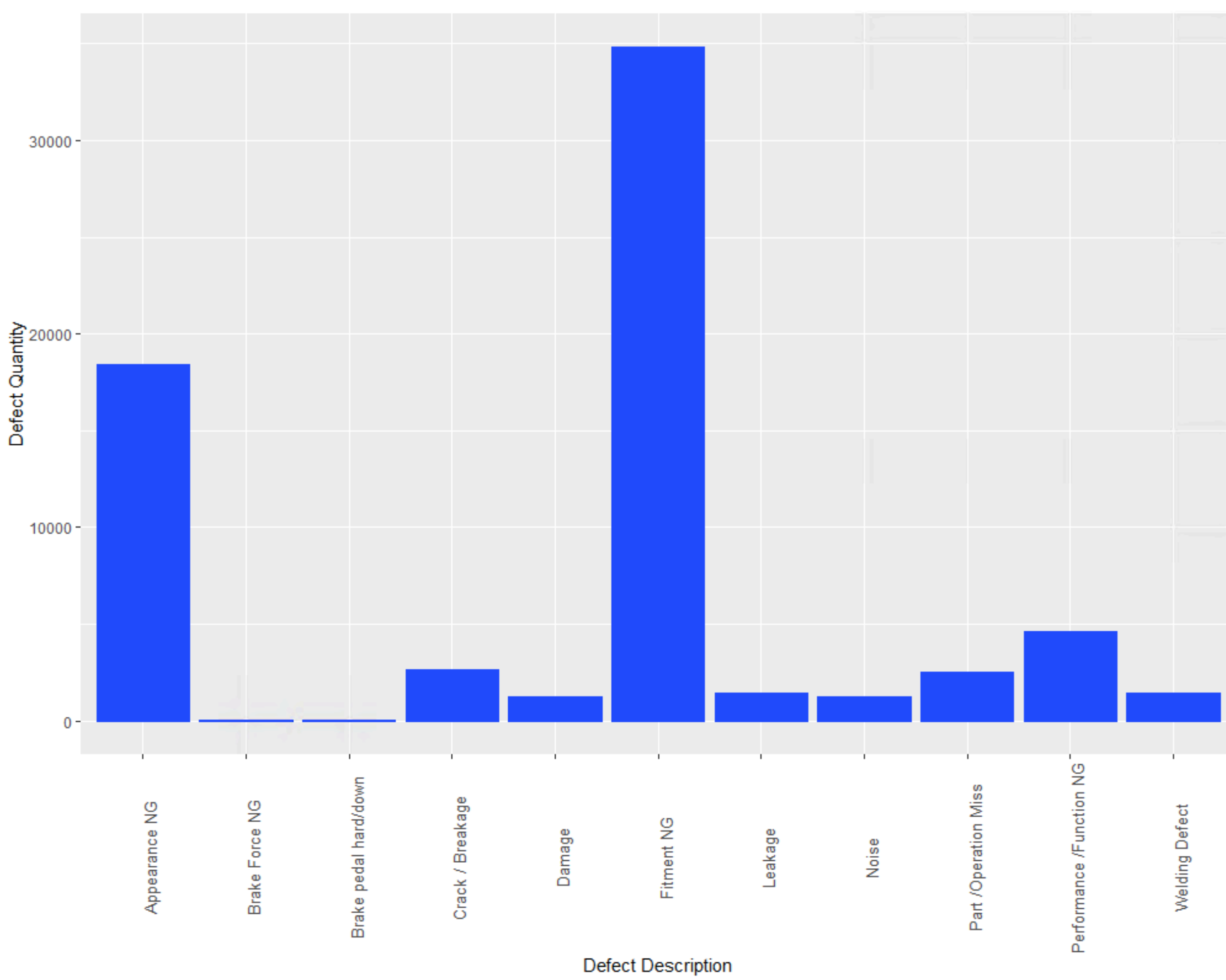
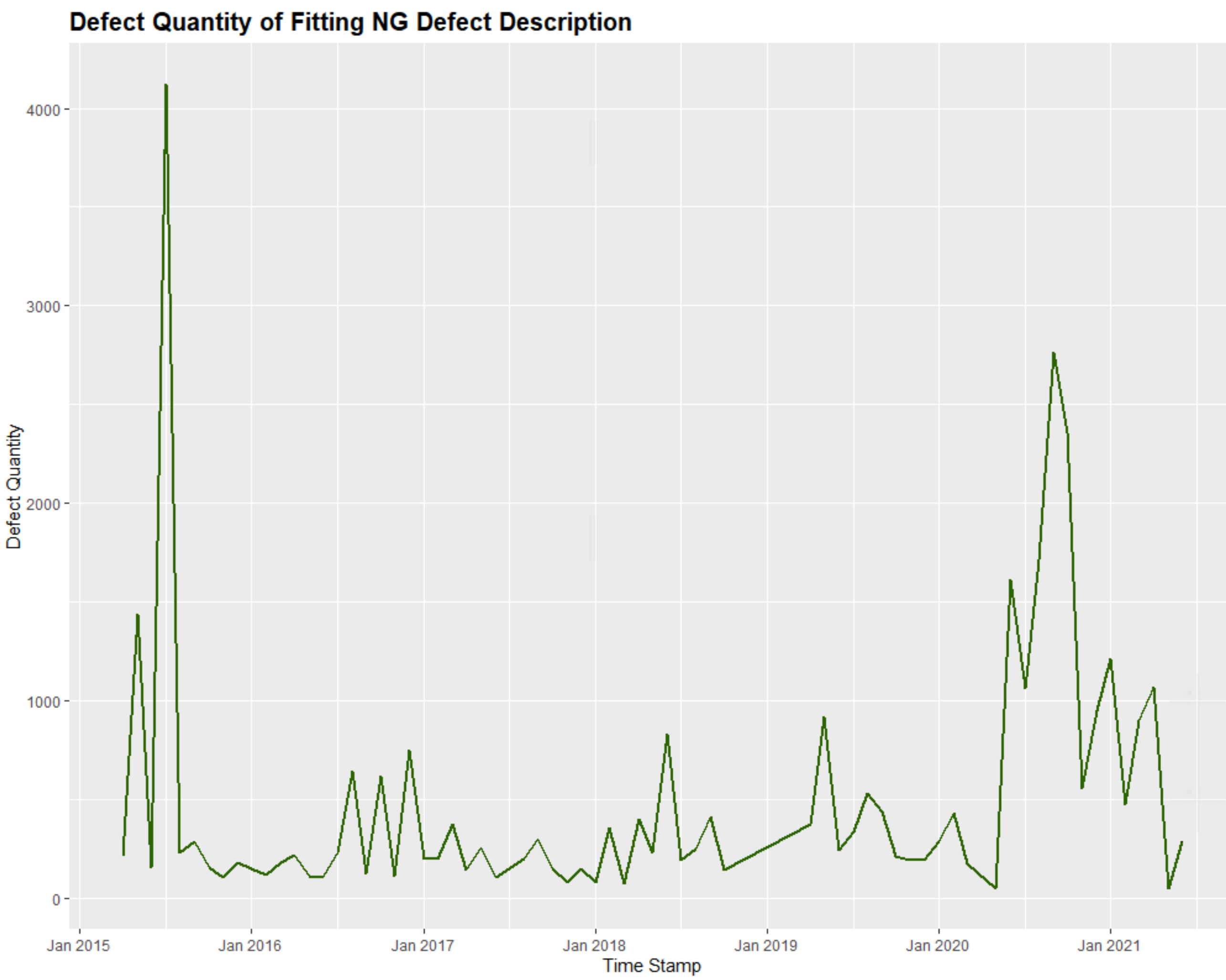
Line Defect Quantity Time Series Analysis

Introduction

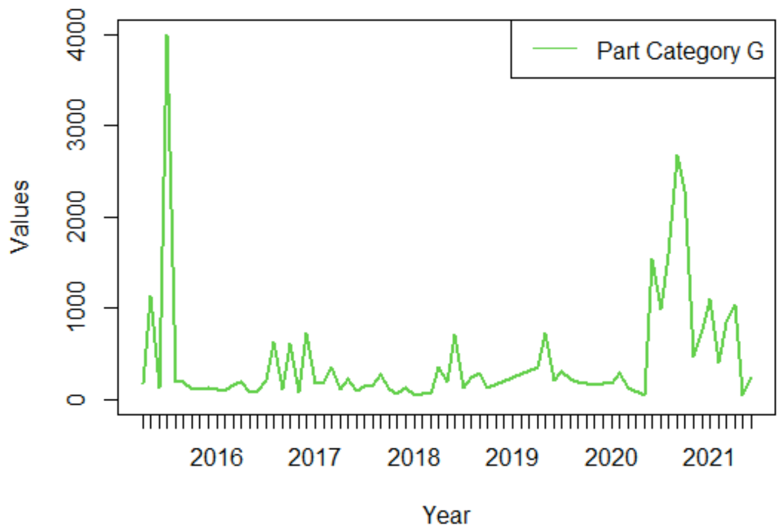
The goal of this time series is to forecast line defect quantities for 9 months starting from October 2020 to June 2021. To start with, datasets from fiscal year 2015 have been combined and all the important features have been considered like Defect Quantity, Date of Punching, Defect Description,Part Category, Rep Defect, Area, Sub Defect Description, Defect Category, 4M, Non-Adherance With MIS-P. Next, we check which type of defect description gives the highest defect quantity.

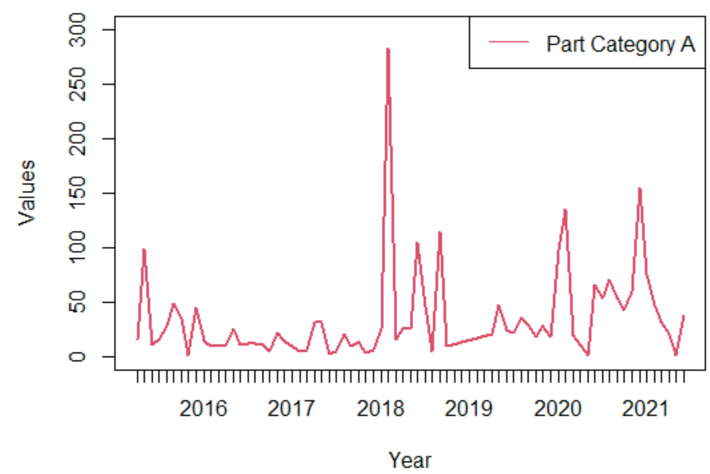


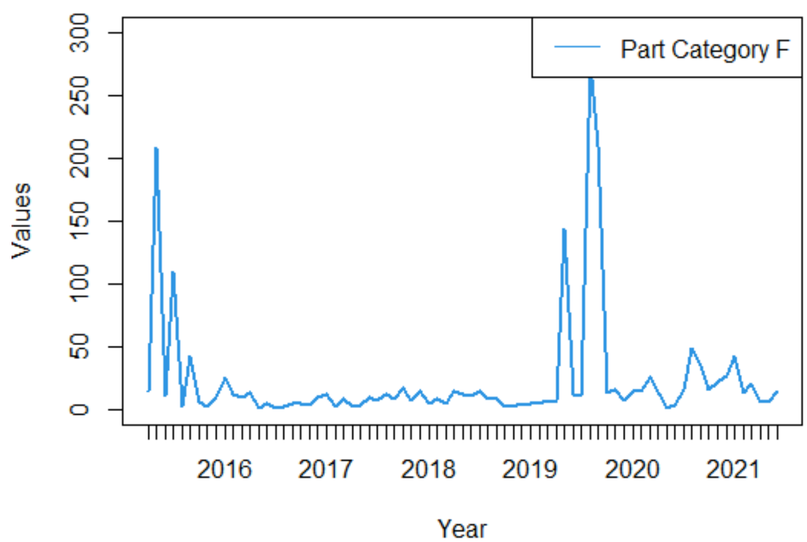
We find that Fitment NG type of defect description gives the highest line defect quantity. Hence this is the most significant defect description type and thus we will select only those observations whose defect description. Next we aggregate all the daily line defect quantities for Fitment NG into monthly quantities and plot its time series.



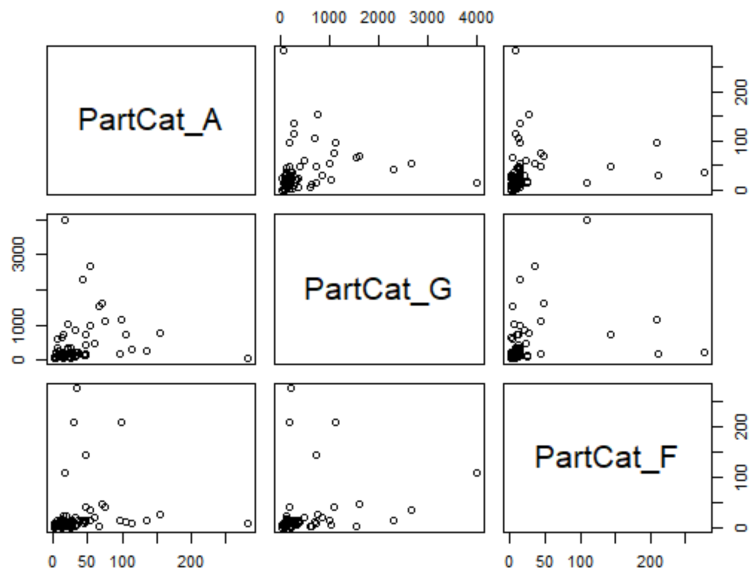
Other than a uniform pattern throughout, we observe big spikes for June 2016 and August 2020. Thus we will contemplate where these spikes come from by examining our features. Thus, we will run Boruta function which is used to study feature importance for the features in deterring line defect quantity except for Date of Punching. After running this function, it only considers Part Category an important feature in determining line defect quantity. We have three type of Part Categories- Part Category G,A,F. After doing some data cleaning for each part category and grouping defect quantity for each part category, we plot times series for individual categories. We find that a majority of Fitment NG defects are coming from Part Category G.



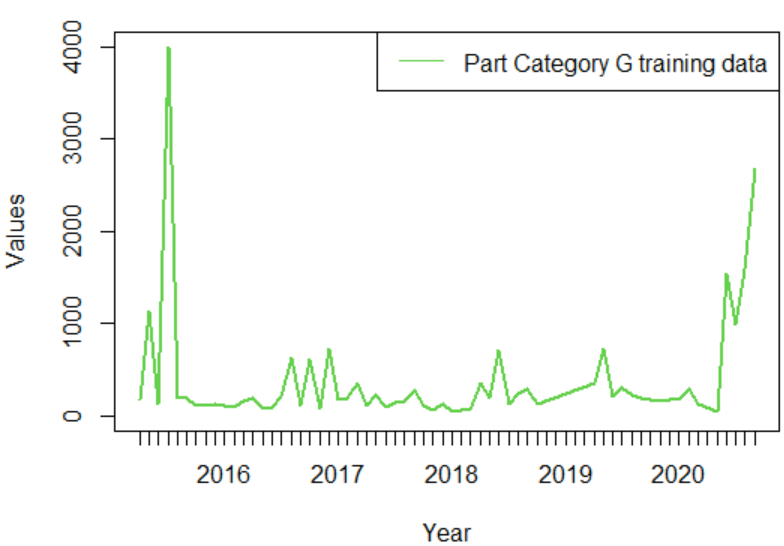


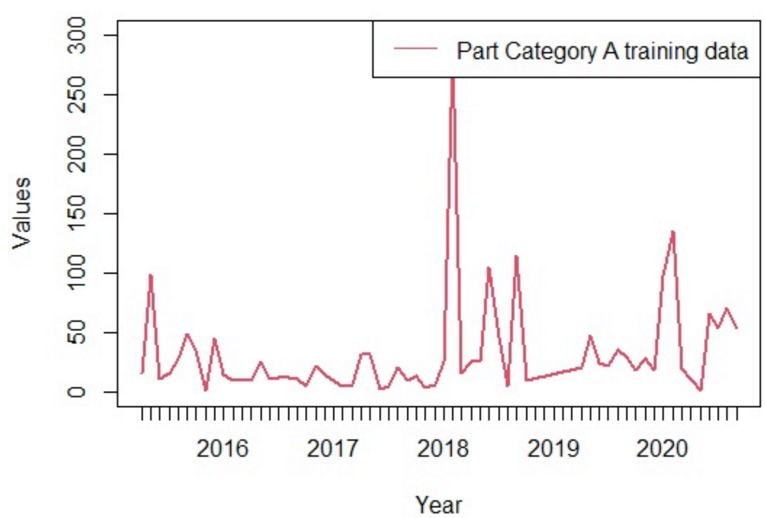


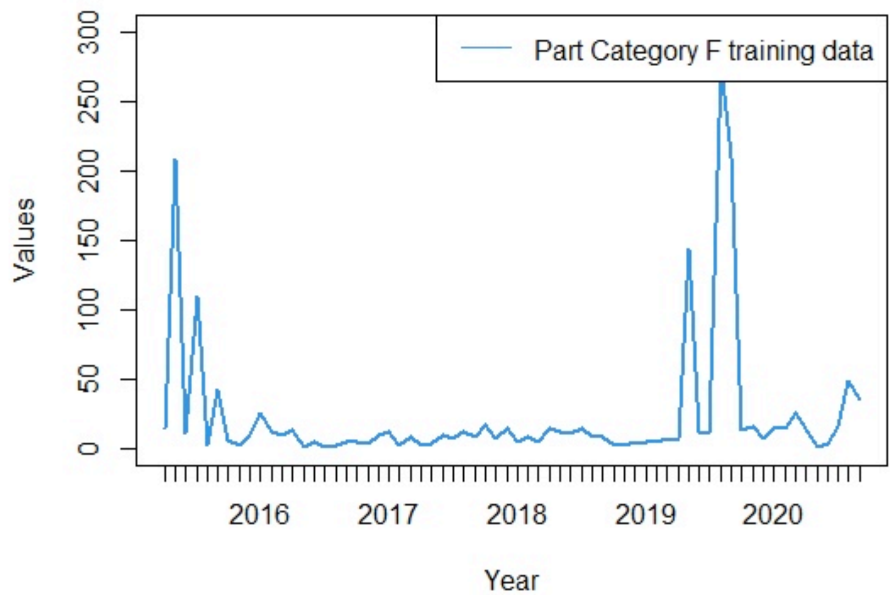
Additionally, we will also plot a correlation plot for our part categories to see if increase or decrease in defect quantity for one part category is related with another. Here, we observe a very weak positive correlation between every two categories. Hence it can be concluded that defect quantities for each part category are not related with one another



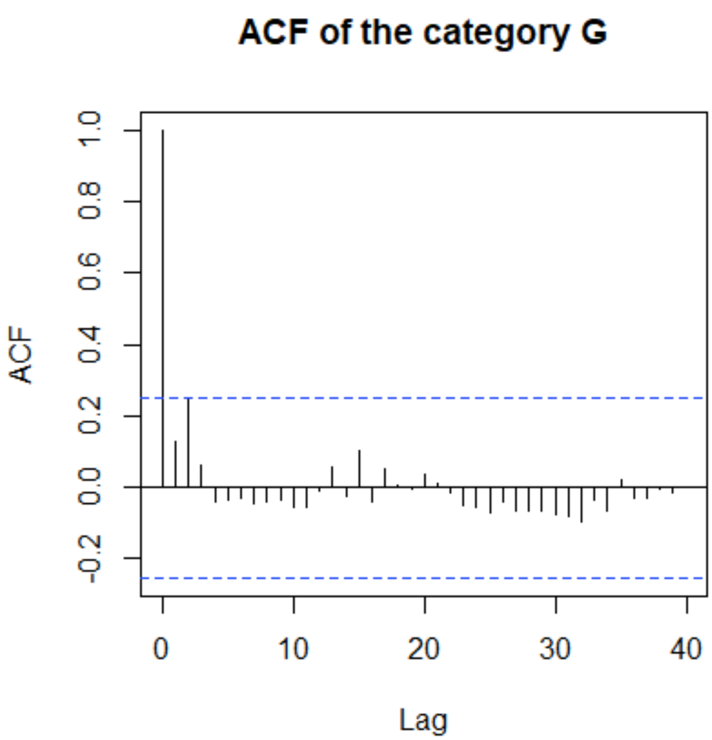
Next we will divide our data into training and testing data for each part category and plot our training data for each part category

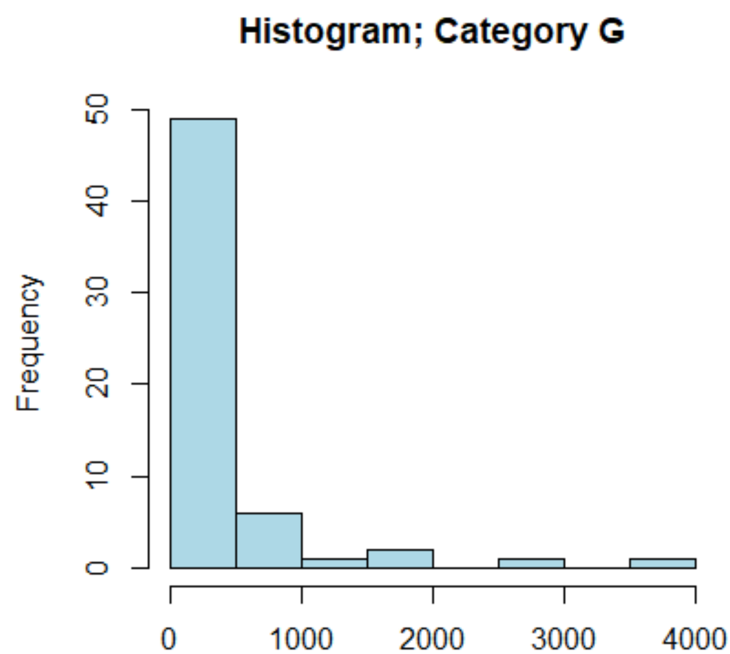


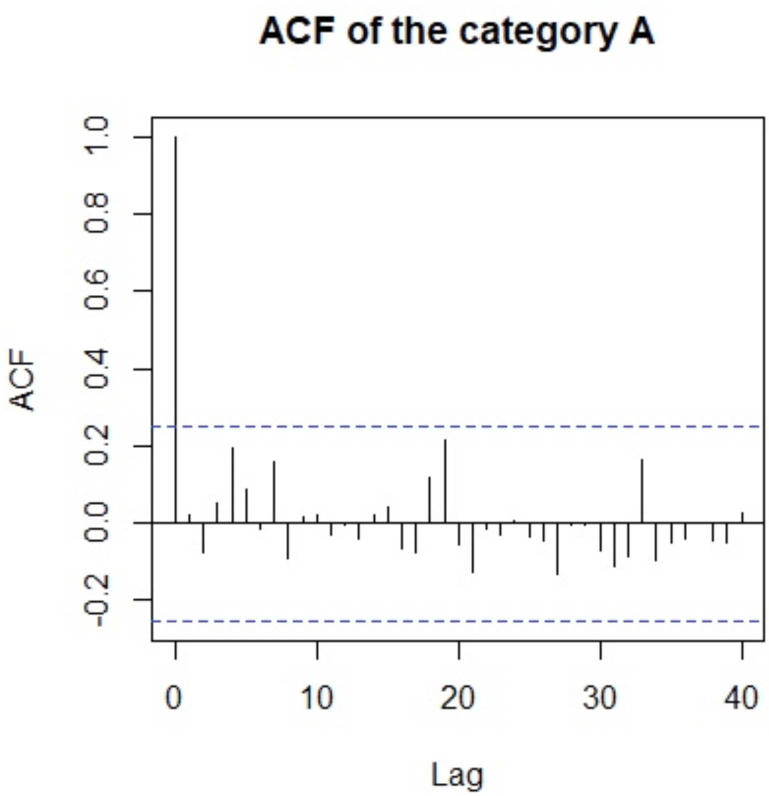


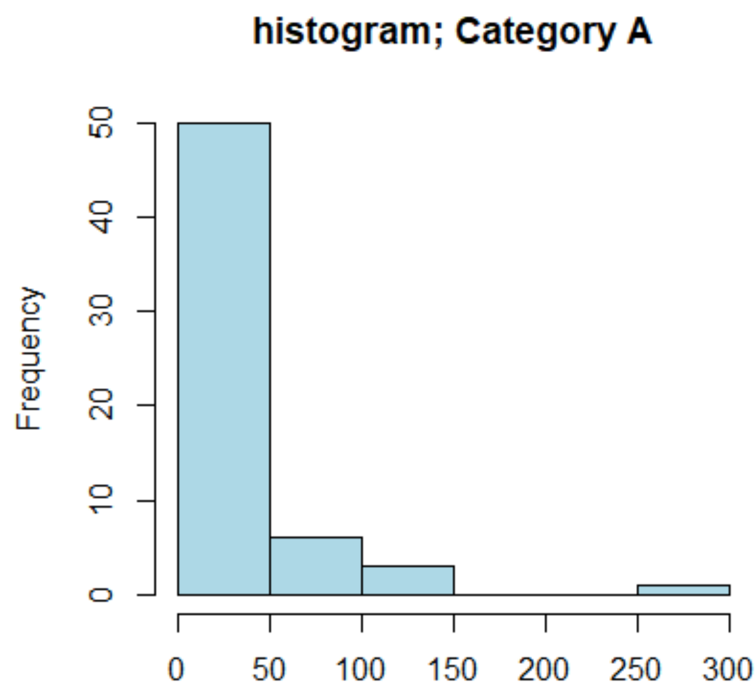


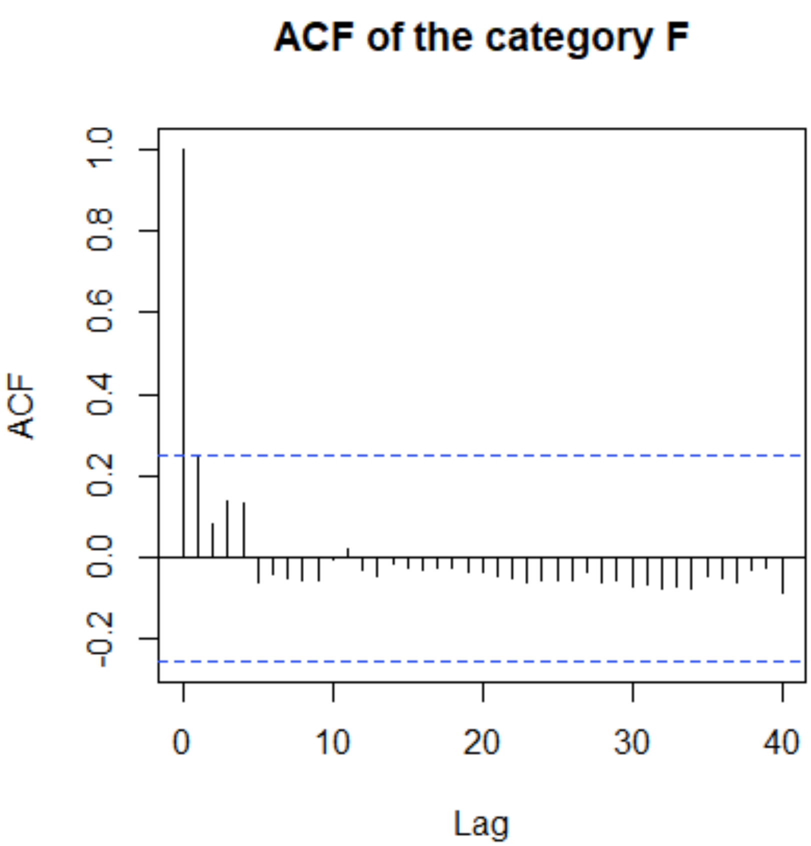
Next, we will plot histograms and autocorrelation function plots for each category to see if transformation is required. We see that histogram plots of category G, A, and F are badly skewed towards left and autocorrelation function plots look periodic.

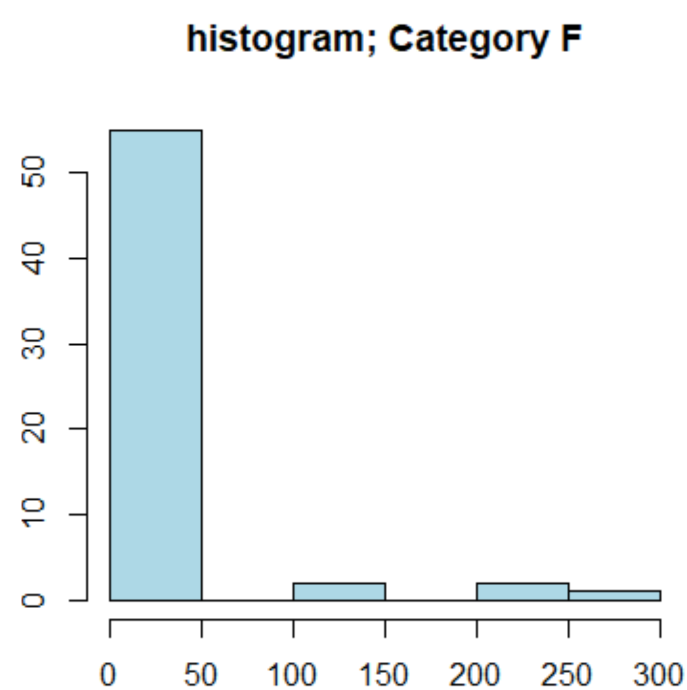




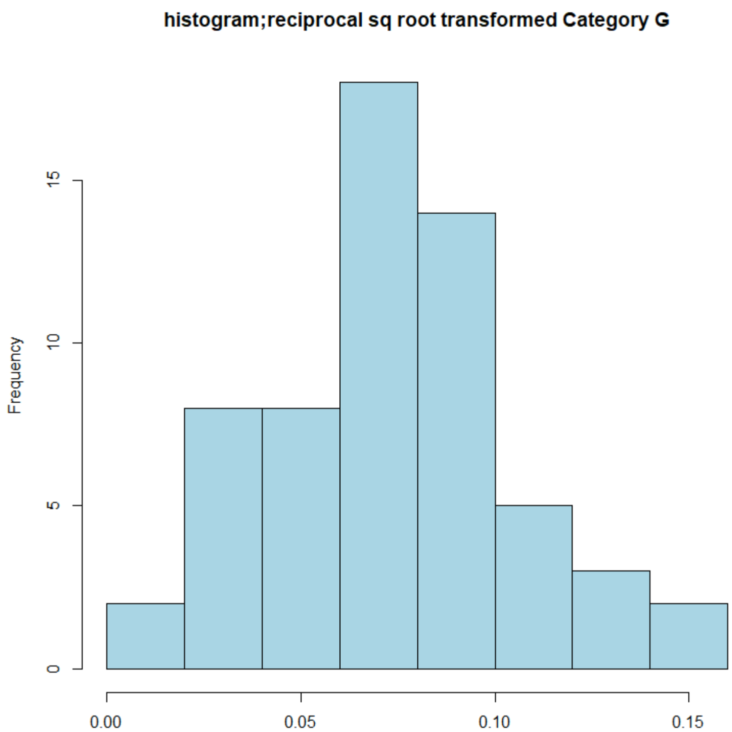

ACF of the category A
ACF
Lag
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0
10
20
30
40

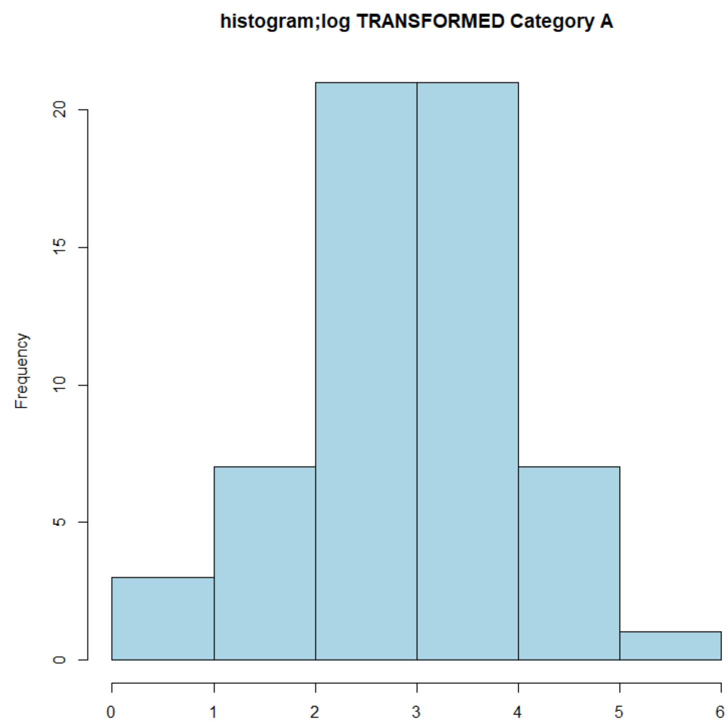

histogram; Category A
Frequency
50
40
30
20
10
0
0
50
100
150
200
250
300

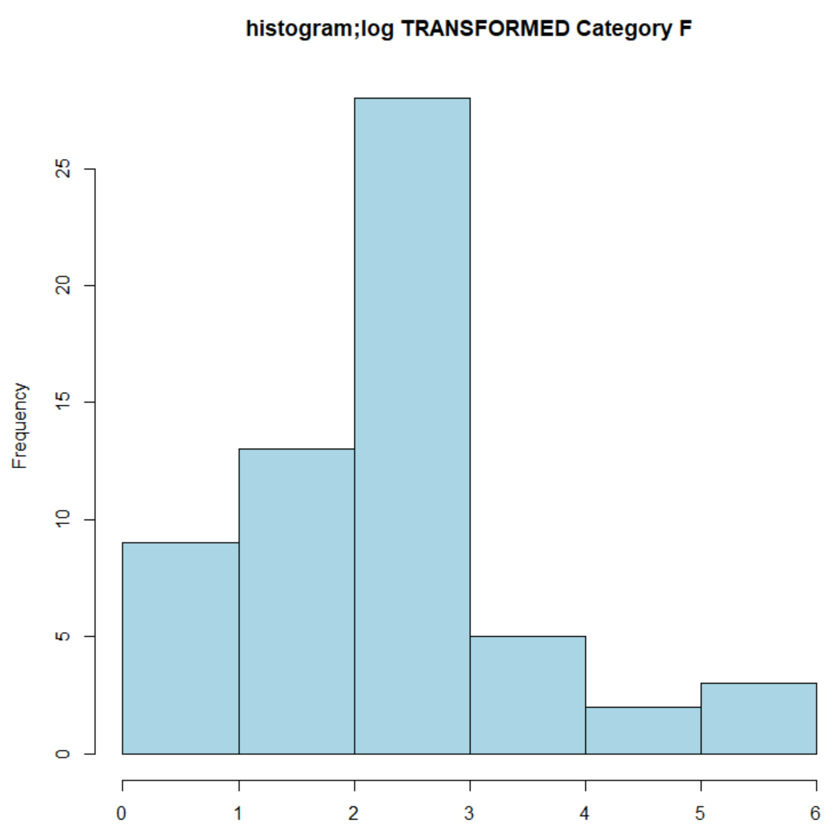

ACF of the category F
ACF
Lag
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
0
10
20
30
40


histogram; Category F
50
40
30
20
10
0
0
50
100
150
200
250
300

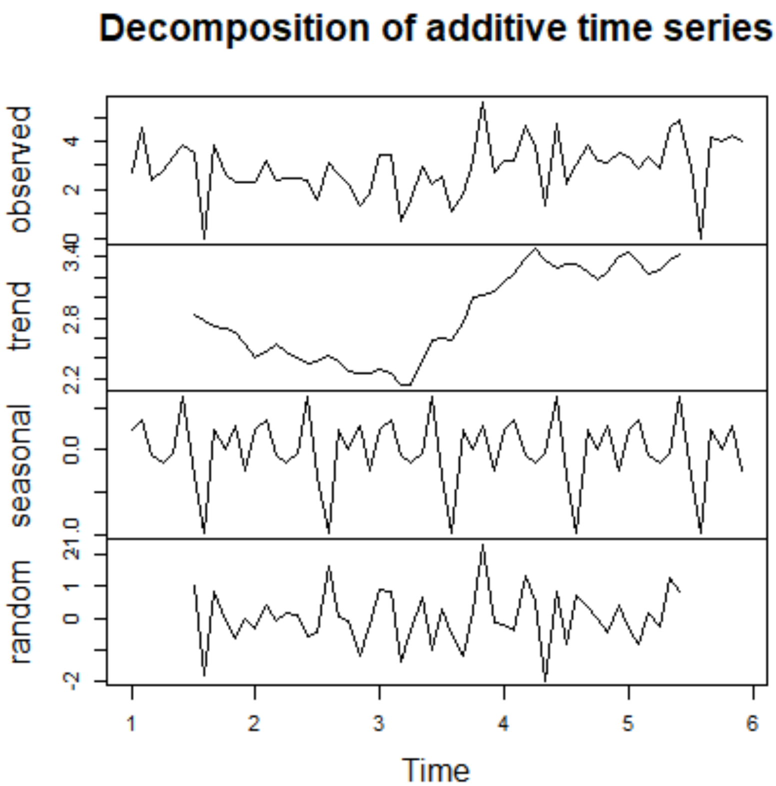
To make the data stationary for all categories, we will transform our our original data for all the categories. To see which which transformation is required, we will plot boxcox plots and check the lambda values to do respective transformations. After doing the transformations, the data for all the categories look much more symmetric.

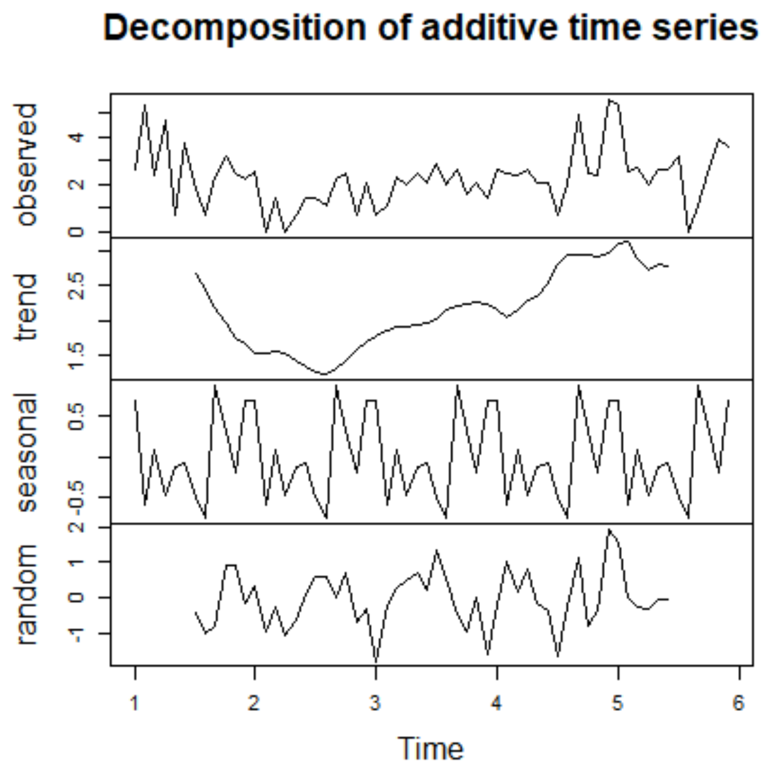


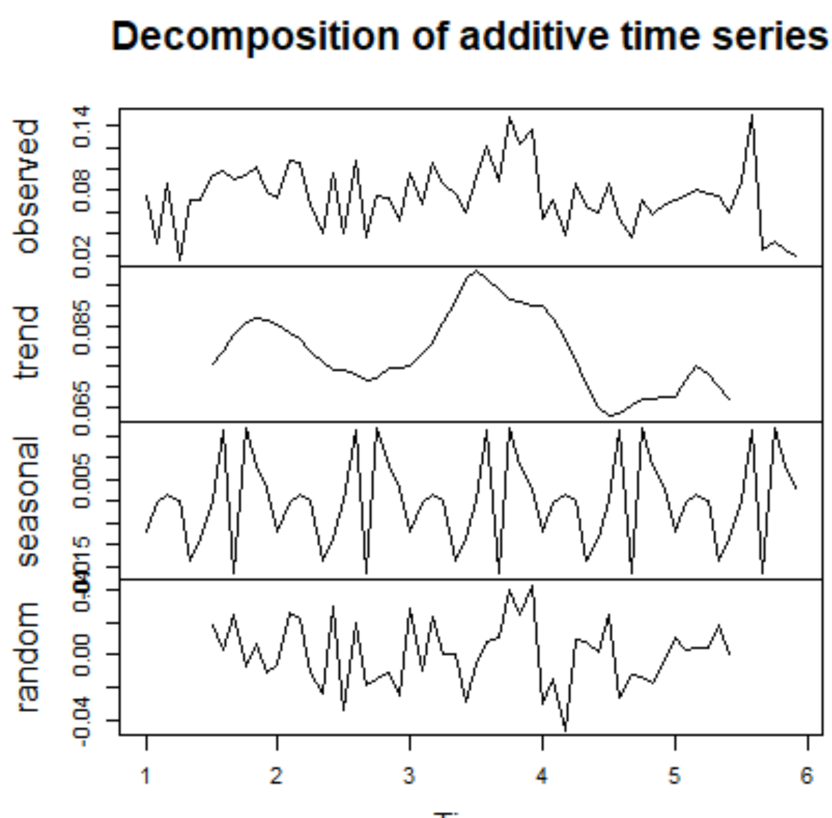




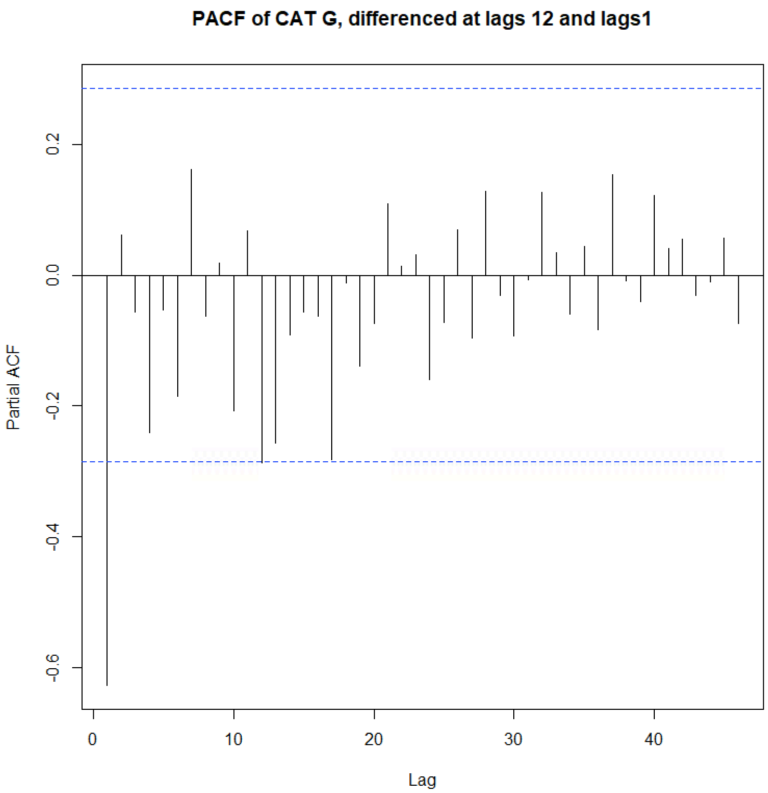
We will also plot decomposition graphs for all the categories to see if we have any seasonality and linear trend in our data. We observe that there is a weak trend and seasonality in all the decomposition plots. Hence we will difference at lag 12 to remove seasonality and lag 1 to remove trend from our time series.

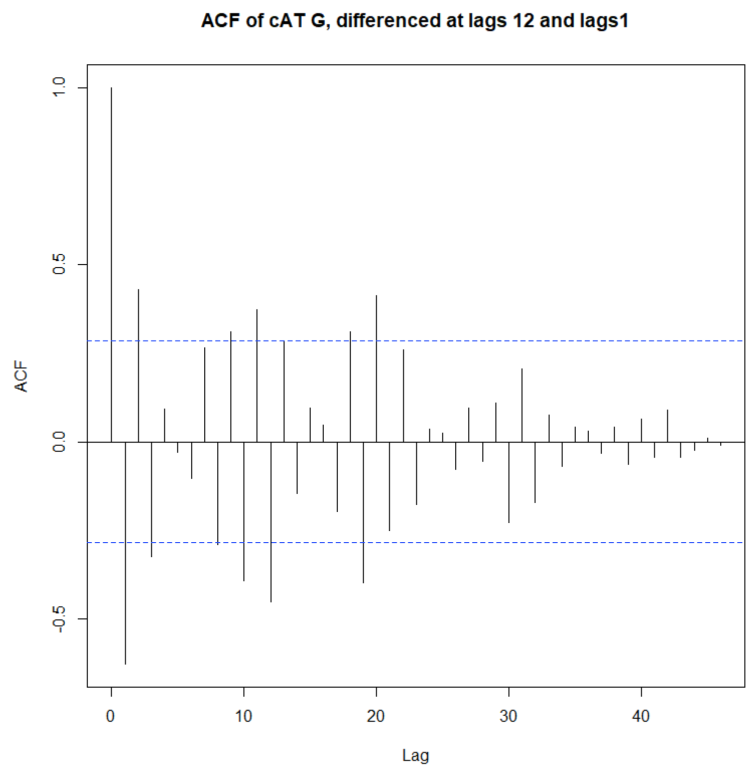




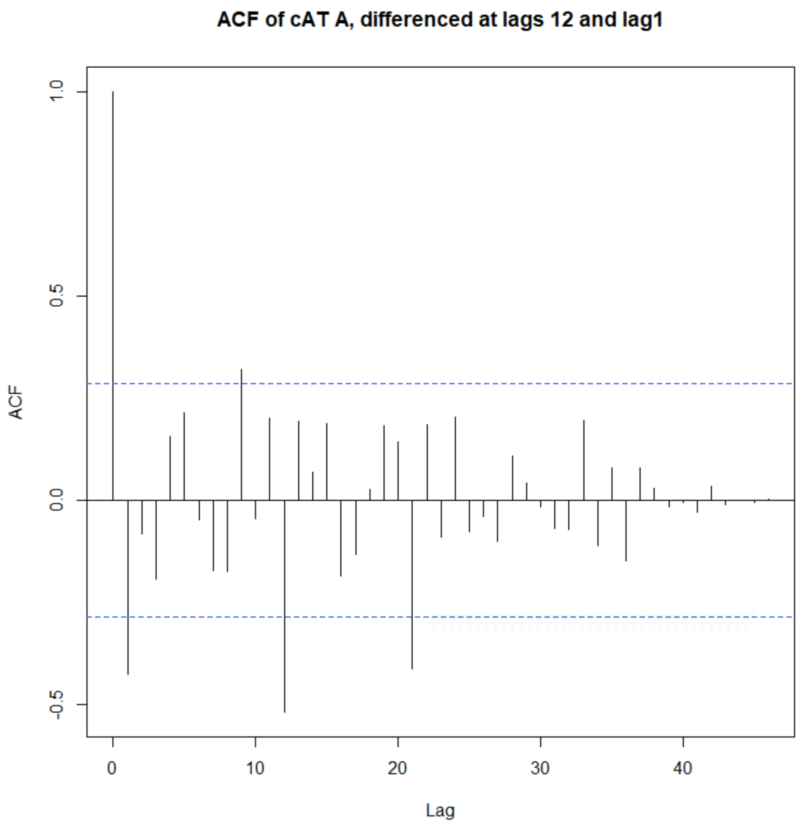


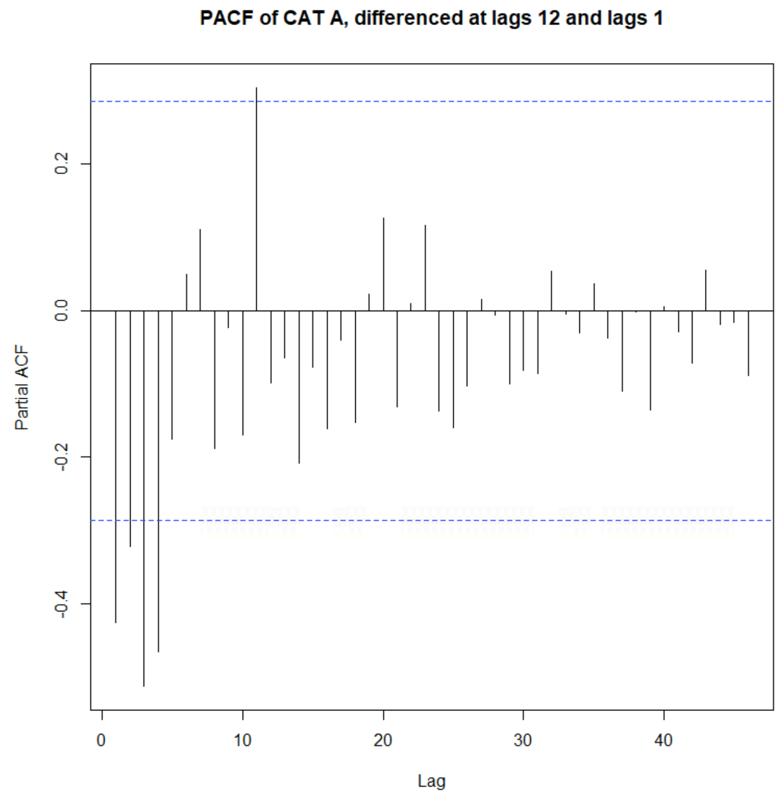
After removing the seasonality and trend from all the categories, we will plot autocorrelation and partial autocorrelation functions for our categories to select values of p, q, Q, and P for our categories. We observe that PACF and ACF of category G has peaks at lags1 and lags 1,2,3,10,11,12,18,19,20. Hence the value of P = 0, p = 1, q = 1,2,3, Q = 1,2



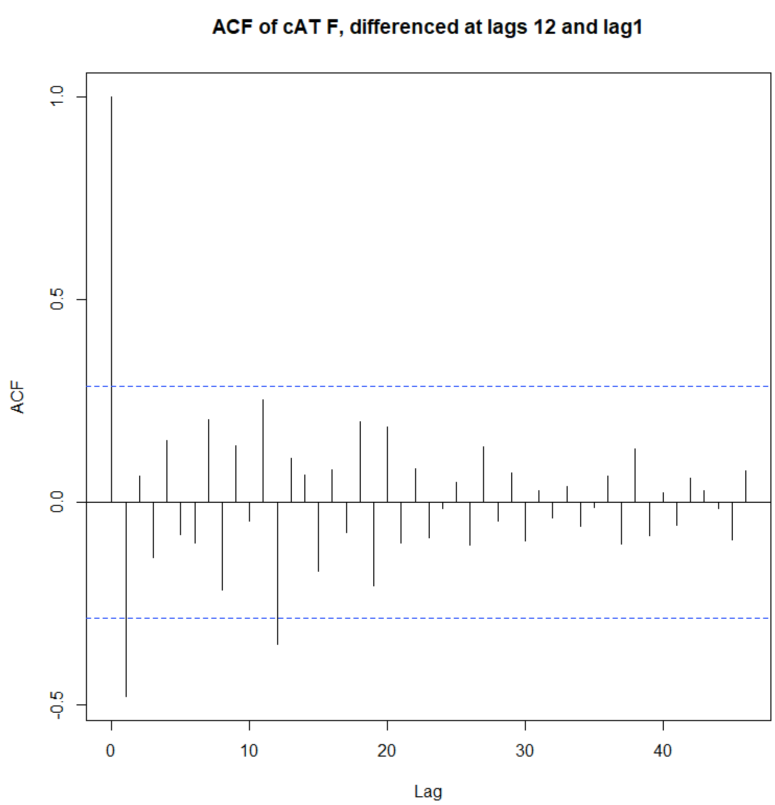


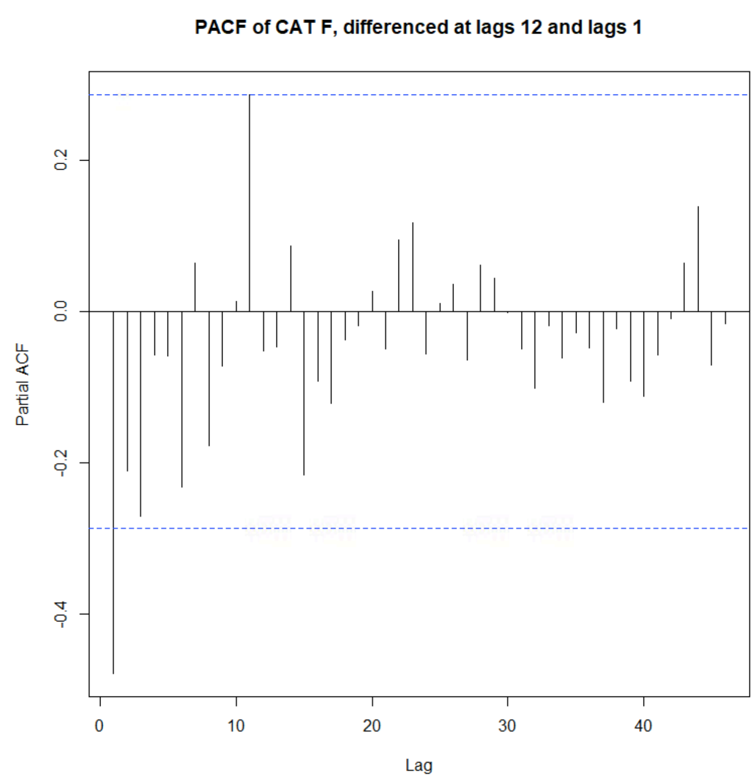
For category A, ACF has peaks at lags 1,9,12,21. Hence choice of q = 1,9 and Q = 1,2. For PACF, we have peaks at lags 1,2,3,4,12. Hence p = 1,2,3,4 and P=12





For category F, ACF has peaks at lag1,12. Hence choice of q = 1 and Q=1. For PACF, there are peaks at lag1. Hence choice of p = 1 and P=0.





By the principle of parsimony, we will select the models out of our candidate models that give us the lowest AICc. Hence we select

model1.2 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order = c(1,1,3),seasonal=list(order=c(0,1,1),period = 12), method="ML") for Category G

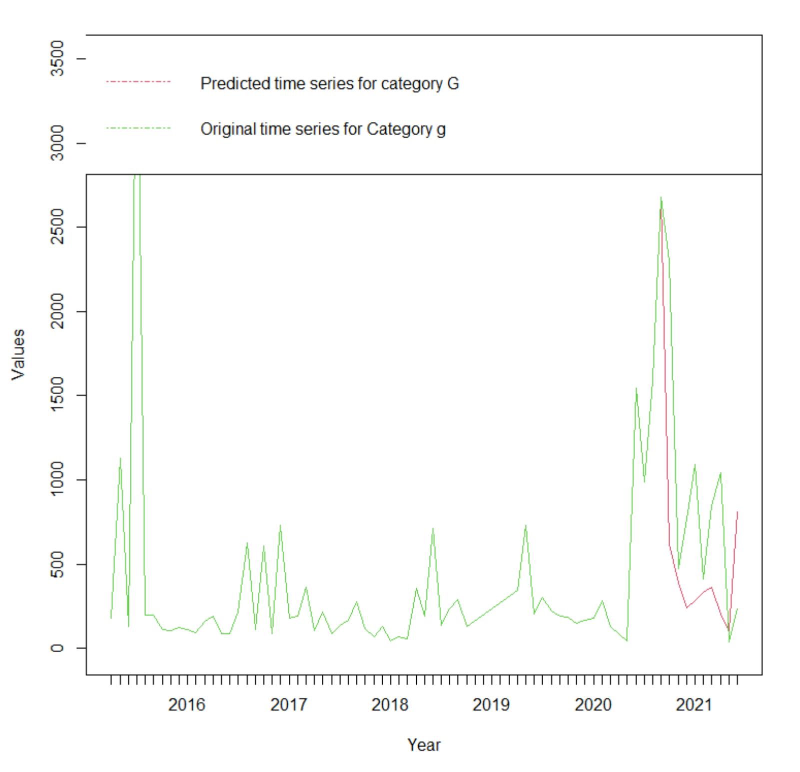
model2.3 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order = c(4,1,1),seasonal=list(order=c(1,1,1),period=12),method="ML") for Category A

model3 =arima(data_line.FitNG.PartCat.train$PartCat_F_log_trans, order = c(1,1,1),seasonal=list(order=c(0,1,1),period=12),method="ML") for Category F

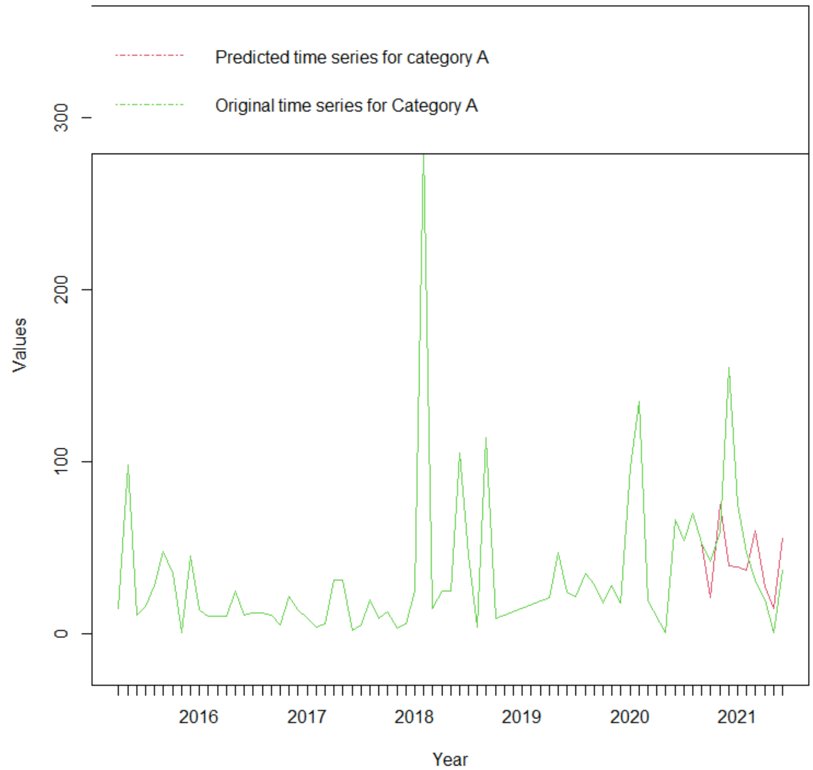
All these selected models pass invertibility and stationarity tests as well as residuals tests such as Box-Pierce Test, Ljung-Box, McLeoad-Li Test. Hence these models can be used for forecasting.

When forecasting line defect quantities for 9 months starting from October 2020 to June 2021, we compare our observations with the test data. We use Mean Absolute Percentage Error (MAPE) to compare our results as it ignores the scaling as for category G we have results in thousands whereas for categories A and F, the results are in tens. Using something as Mean Absolute error (MAE) would hive biased results for category G as it incorporates scaling.

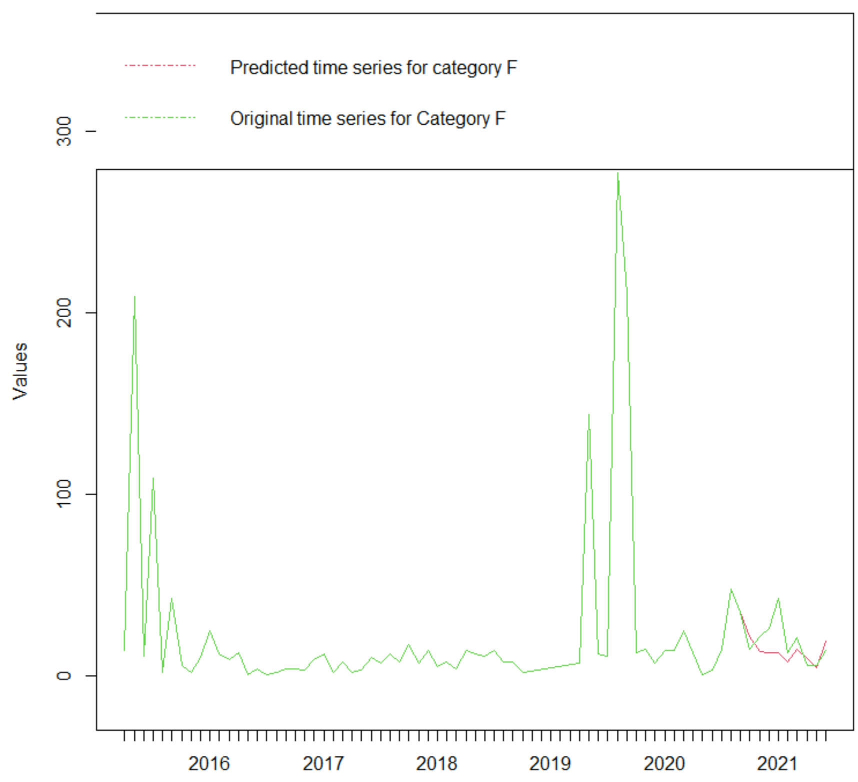
On comparing our predicted values with test values for all the categories we get 55.28 as MAPE for category G after removing the outliers.



On comparing our predicted values with test values for all the categories we get 38.57 as MAPE for category A after removing the outliers.



On comparing our predicted values with test values for all the categories we get 36.67 as MAPE for category F after removing the outliers.



Hence, we observe a signifiant deviance in the predicted values from the test values due to the fact that the original time series didn’t show a great degree of trend and seasonality. Therefore, accurate predictions cannot be easily found.

Appendix

```
#To read the excel file
install.packages("readxl")
library(readxl)
setwd("C:/Users/Administrator/Downloads")
data_line01 = read_excel("FY15-16.xlsx")
data_line02 = read_excel("FY16-17.xlsx")
data_line03 = read_excel("FY17-18.xlsx")
data_line04 = read_excel("FY18-19.xlsx")
data_line04.1 = read_excel("FY19-20.xlsx")
data_line05 = read_excel("FY20-21.xlsx")
data_line06 = read_excel("FY21-22.xlsx")

data_line1 = subset(data_line01,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line2 = subset(data_line02,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line3 = subset(data_line03,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line4 = subset(data_line04,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line4.1 = subset(data_line04.1,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line5 = subset(data_line05,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))
data_line6 = subset(data_line06,select = c("Defect Qty","Dt of Punching","Defect Desc","Part Cat","Rep Defect","Area","Sub Defect Desc","Def Cat","4M","Non-Adherance With MIS-P"))

data_line = rbind(data_line1,data_line2,data_line3,data_line4,data_line4.1,data_line5,data_line6)
data_line
```

```
#To change the column names to remove space
colnames(data_line)[1] = "DefQty"
colnames(data_line)[2] = "DtPunching"
colnames(data_line)[3] = "DefectDesc"
colnames(data_line)[4] = "PartCat"
colnames(data_line)[5] = "RepDefect"
colnames(data_line)[7] = "SubDefectDesc"
colnames(data_line)[8] = "DefCat"
colnames(data_line)[9] = "M4"
colnames(data_line)[10] = "Non-Adherance_With_MIS-P"
```

```
data_line
colnames(data_line)

#To extract months from our Date of Punching
DtPunching = as.Date(data_line$DtPunching,"%Y-%m-%d")
data_line$DtPunching = DtPunching

## To make new data-frame Defect-Qty, Dt of Punching, Defect Description
data_line_new = data.frame(data_line$DtPunching,data_line$DefQty,data_line$DefectDesc)
colnames(data_line_new) = c('DtPunching','DefQty','DefectDesc')
data_line_new
data_line.month = as.Date(data_line_new$DtPunching,format = "%Y-%m-%d")
data_line.month
data_line_new$TimeStamp = format(data_line.month,"%Y-%m")
data_line_new
data_line_new.Ddesc.TStamp =aggregate(DefQty ~ DefectDesc + TimeStamp, data =
data_line_new, FUN = sum, na.rm = TRUE)
data_line_new.Ddesc.TStamp
data_line_new.Ddesc = aggregate(DefQty ~ DefectDesc, data = data_line_new, FUN = sum,
na.rm = TRUE)
data_line_new.Ddesc
```

```
# Next we will plot Bar-Plots to show which Defect Description gives us
#maximum values such that we pick the one with the highest or second highest values

ggplot(data=data_line_new.Ddesc, aes(x=DefectDesc, y=DefQty)) +
  geom_bar(stat="identity", color="blue", fill="blue") +
  theme(axis.text.x = element_text(angle = 90, size = 10))+
  labs(y= "Defect Quantity", x = "Defect Description")


# Based on the Bar-plot we will select Fitment-NG's Defect Quantities




#data_line_new.AppNG =
data_line_new.Ddesc.TStamp[data_line_new.Ddesc.TStamp$DefectDesc == "Appearance
NG",c("TimeStamp", "DefQty")]
#rownames(data_line_new.AppNG) = data_line_new.AppNG$TimeStamp #Indexing as Time
Stamps
#data_line_new.AppNG$TimeStamp = as.yearmon(data_line_new.AppNG$TimeStamp, "%Y-
%m")


library("zoo")
data_line_new.FitNG = data_line_new.Ddesc.TStamp[data_line_new.Ddesc.TStamp$DefectDesc
== "Fitment NG",c("TimeStamp", "DefQty")]
rownames(data_line_new.FitNG) = data_line_new.FitNG$TimeStamp #Indexing as Time
Stamps
data_line_new.FitNG$TimeStamp = as.yearmon(data_line_new.FitNG$TimeStamp, "%Y-%m")




install.packages("ggplot2")
library(ggplot2)
#ts1 = ggplot(data_line_new.AppNG, aes(x=TimeStamp, y=DefQty))+geom_line(na.rm=TRUE)
+
  #ggtitle("Defect Quantity of Appearance NG Defect Description") +
  #xlab("Time Stamp") + ylab("Defect Quantity") +
  #theme(plot.title = element_text(lineheight=.20, face="bold",
```

```
                          #size = 15))+ geom_line(color = "#FC4E07", size = 1)
#ts1
ts2 = ggplot(data_line_new.FitNG, aes(x=TimeStamp, y=DefQty))+geom_line(na.rm=TRUE) +
  ggtitle("Defect Quantity of Fitting NG Defect Description") +
  xlab("Time Stamp") + ylab("Defect Quantity") +
  theme(plot.title = element_text(lineheight=.20, face="bold",
                          size = 15))+ geom_line(color = "dark green", size = 1)

ts2

data_line
data_line.FitNG = data_line[data_line$DefectDesc=="Fitment NG",]




## FEATURE SELECTION STEPS
install.packages('Boruta')
install.packages('mlbench')
install.packages('caret')
install.packages('randomForest')

library(Boruta)
library(mlbench)
library(caret)
library(randomForest)



# Data
str(data_line.FitNG)

#To remove NA rows
data_line.FitNG = na.omit(data_line.FitNG)


# Feature Selection
set.seed(11)
boruta = Boruta(DefQty~.,data=data_line.FitNG,doTrace=2,maxRuns=100)
print(boruta)

#Tentative feature fix
```

```
bor = TentativeRoughFix(boruta)
print(bor)
attStats(boruta)

#Based on our Feature selection we will select feature Part category

data_line.FitNG.Part_A = aggregate(DefQty ~ PartCat+DtPunching, data = data_line.FitNG,
FUN = sum, na.rm = TRUE)
data_line.FitNG.Part_A = data_line.FitNG.Part_A[data_line.FitNG.Part_A$PartCat=='A',]
data_line.FitNG.Part_A$TimeStamp =
format(as.Date(data_line.FitNG.Part_A$DtPunching,format = "%Y-%m-%d"),"%Y-%m")
data_line.FitNG.Part_A$TimeStamp = as.yearmon(data_line.FitNG.Part_A$TimeStamp, "%Y-
%m")
data_line.FitNG.Part_A = aggregate(DefQty ~ TimeStamp, data =data_line.FitNG.Part_A , FUN
= sum, na.rm = TRUE)
colnames(data_line.FitNG.Part_A)[2] = 'DefQty.Part_A'


data_line.FitNG.Part_F = aggregate(DefQty ~ PartCat+DtPunching, data = data_line.FitNG,
FUN = sum, na.rm = TRUE)
data_line.FitNG.Part_F = data_line.FitNG.Part_F[data_line.FitNG.Part_F$PartCat=='F',]
data_line.FitNG.Part_F$TimeStamp =
format(as.Date(data_line.FitNG.Part_F$DtPunching,format = "%Y-%m-%d"),"%Y-%m")
data_line.FitNG.Part_F$TimeStamp = as.yearmon(data_line.FitNG.Part_F$TimeStamp, "%Y-
%m")
data_line.FitNG.Part_F = aggregate(DefQty ~ TimeStamp, data =data_line.FitNG.Part_F , FUN
= sum, na.rm = TRUE)
colnames(data_line.FitNG.Part_F)[2] = 'DefQty.Part_F'

data_line.FitNG.Part_G = aggregate(DefQty ~ PartCat+DtPunching, data = data_line.FitNG,
FUN = sum, na.rm = TRUE)
data_line.FitNG.Part_G = data_line.FitNG.Part_G[data_line.FitNG.Part_G$PartCat=='G',]
data_line.FitNG.Part_G$TimeStamp =
format(as.Date(data_line.FitNG.Part_G$DtPunching,format = "%Y-%m-%d"),"%Y-%m")
data_line.FitNG.Part_G$TimeStamp = as.yearmon(data_line.FitNG.Part_G$TimeStamp, "%Y-
%m")
data_line.FitNG.Part_G = aggregate(DefQty ~ TimeStamp, data =data_line.FitNG.Part_G , FUN
= sum, na.rm = TRUE)
colnames(data_line.FitNG.Part_G)[2] = 'DefQty.Part_G'

data_line.FitNG.Part_F$TimeStamp==data_line.FitNG.Part_G$TimeStamp
data_line.FitNG.Part_A$TimeStamp==data_line.FitNG.Part_G$TimeStamp
data_line.FitNG.Part_F$TimeStamp==data_line.FitNG.Part_A$TimeStamp
```

```
new_row1 = c(NA,NA)


data_line.FitNG.Part_F<- rbind(data_line.FitNG.Part_F[1:55, ],
              new_row1 ,
               data_line.FitNG.Part_F[-(1:55), ])

data_line.FitNG.Part_F[56,1] = as.yearmon('2020-05', "%Y-%m")
data_line.FitNG.Part_F[56,2] = 1


data_line.FitNG.Part_A<- rbind(data_line.FitNG.Part_A[1:7, ],
                         new_row1 ,
                         data_line.FitNG.Part_A[- (1:7), ])


data_line.FitNG.Part_A[8,1] = as.yearmon('2015-11', "%Y-%m")
data_line.FitNG.Part_A[8,2] = 1

data_line.FitNG.Part_F$TimeStamp==data_line.FitNG.Part_G$TimeStamp
data_line.FitNG.Part_A$TimeStamp==data_line.FitNG.Part_G$TimeStamp
data_line.FitNG.Part_F$TimeStamp==data_line.FitNG.Part_A$TimeStamp


data_line.FitNG.PartCat =
data.frame(data_line.FitNG.Part_F$TimeStamp,data_line.FitNG.Part_A$DefQty.Part_A,data_lin
e.FitNG.Part_G$DefQty.Part_G,data_line.FitNG.Part_F$DefQty.Part_F)
colnames(data_line.FitNG.PartCat) = c("TimeStamp","PartCat_A","PartCat_G","PartCat_F")
rownames(data_line.FitNG.PartCat) = data_line.FitNG.PartCat$TimeStamp

#After all the data-cleaning, we will analyze two time series
#1data_line.FitNG.PartCat
#2data_line.nonAdh (later)

plot(data_line.FitNG.PartCat$TimeStamp,
    data_line.FitNG.PartCat$PartCat_A,
    type = "l",ylim = c(- 0, 300),
    col = 2,lwd=2,
    xlab = "Year",
    ylab = "Values")
legend("topright",
      "Part Category A",
```

```
       lty = 1,
       col = 2)


plot(data_line.FitNG.PartCat$TimeStamp,
     data_line.FitNG.PartCat$PartCat_G,
     type = "l",ylim = c( 0, 4000),
     col = 3,lwd=2,
     xlab = "Year",
     ylab = "Values")
legend("topright",
       "Part Category G",
       lty = 1,
       col = 3)

 plot(data_line.FitNG.PartCat$TimeStamp,
     data_line.FitNG.PartCat$PartCat_F,
     type = "l",ylim = c(- 0, 300),
     col = 4,lwd=2,
     xlab = "Year",
     ylab = "Values")
legend("topright",
       "Part Category F",
       lty = 1,
       col = 4)



#Let's plot pair-plots to check if the defect quantities for different part types are
#are correlated


# Equivalent with a formula
pairs(~ PartCat_A+PartCat_G+PartCat_F, data = data_line.FitNG.PartCat)


#There seems to be a weak positive correlation between all the part categories
#for all the months



#Our first step would be dividing the data into training and test set
```

```
data_line.FitNG.PartCat.train = data_line.FitNG.PartCat[c(1:60),]
data_line.FitNG.PartCat.test = data_line.FitNG.PartCat[c(61:69),]


plot(data_line.FitNG.PartCat.train$TimeStamp,
    data_line.FitNG.PartCat.train$PartCat_A,
    type = "l",ylim = c(- 0, 300),
    col = 2,lwd=2,
    xlab = "Year",
    ylab = "Values")
legend("topright",
      "Part Category A training data",
      lty = 1,
      col = 2)


plot(data_line.FitNG.PartCat.train$TimeStamp,
    data_line.FitNG.PartCat.train$PartCat_G,
    type = "l",ylim = c( 0, 4000),
    col = 3,lwd=2,
    xlab = "Year",
    ylab = "Values")
legend("topright",
      "Part Category G training data",
      lty = 1,
      col = 3)

plot(data_line.FitNG.PartCat.train$TimeStamp,
    data_line.FitNG.PartCat.train$PartCat_F,
    type = "l",ylim = c(- 0, 300),
    col = 4,lwd=2,
    xlab = "Year",
    ylab = "Values")
legend("topright",
      "Part Category F training data",
      lty = 1,
      col = 4)


fit <- lm(data_line.FitNG.PartCat.train$PartCat_G ~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_G)))
abline(fit, col="red")
abline(h=mean(data_line.FitNG.PartCat$PartCat_G), col="blue")
```

```
hist(data_line.FitNG.PartCat.train$PartCat_G, col="light blue", xlab="", main="Histogram;
Category G")
acf(data_line.FitNG.PartCat.train$PartCat_G,lag.max=40, main="ACF of the category G")

#Data is highly non-stationary because the histogram is badly skewed towards left



#lambda = -1. is a reciprocal transform.
#lambda = -0.5 is a reciprocal square root transform.
#lambda = 0.0 is a log transform.
#lambda = 0.5 is a square root transform.
#lambda = 1.0 is no transform.

install.packages('MASS')
library(MASS)
bcTransform <- boxcox(data_line.FitNG.PartCat.train$PartCat_G~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_G)))
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]


#We will use reciprocal square root transformation for Part Category G




data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt = 1/
sqrt(data_line.FitNG.PartCat.train$PartCat_G)


fit <- lm(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt ~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt)))
hist(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, col="light blue", xlab="", main="
histogram;reciprocal sq root transformed Category G  ")
acf(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt,lag.max=40, main="ACF of the
reciprocal sq root transformed G")




#####To check for transformation of Part A
```

```
library(MASS)
bcTransform <- boxcox(data_line.FitNG.PartCat.train$PartCat_A~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_A)))
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]


fit1 <- lm(data_line.FitNG.PartCat.train$PartCat_A ~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_A)))
hist(data_line.FitNG.PartCat.train$PartCat_A, col="light blue", xlab="", main="histogram;
Category A")
acf(data_line.FitNG.PartCat.train$PartCat_A,lag.max=40, main="ACF of the category A")

#Data is highly non-stationary because the histogram is badly skewed towards left


#By using different transformations we arrived at this one
data_line.FitNG.PartCat.train$PartCat_A_log_trans =
log(data_line.FitNG.PartCat.train$PartCat_A)


hist(data_line.FitNG.PartCat.train$PartCat_A_log_trans, col="light blue", xlab="",
main="histogram;log TRANSFORMED Category A")
acf(data_line.FitNG.PartCat.train$PartCat_A_log_trans,lag.max=40, main="ACF of the category
A log transformed")


#Part Category F


library(MASS)
bcTransform <- boxcox(data_line.FitNG.PartCat.train$PartCat_F~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_F)))
bcTransform$x[which(bcTransform$y == max(bcTransform$y))]
lambda=bcTransform$x[which(bcTransform$y == max(bcTransform$y))]


fit2 <- lm(data_line.FitNG.PartCat.train$PartCat_F ~
as.numeric(1:length(data_line.FitNG.PartCat.train$PartCat_F)))
```

```
hist(data_line.FitNG.PartCat.train$PartCat_F, col="light blue", xlab="", main="histogram;
Category F")
acf(data_line.FitNG.PartCat.train$PartCat_F,lag.max=40, main="ACF of the category F")



data_line.FitNG.PartCat.train$PartCat_F_log_trans =
log(data_line.FitNG.PartCat.train$PartCat_F)

hist(data_line.FitNG.PartCat.train$PartCat_F_log_trans, col="light blue", xlab="",
main="histogram;log TRANSFORMED Category F")
acf(data_line.FitNG.PartCat.train$PartCat_F_log_trans,lag.max=40, main="ACF of the category
F log transformed")





#To produce decomposition of our data
library(ggplot2)
y = ts(as.ts(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt),frequency=12)
decomp = decompose(y)
plot(decomp)
#Time Series part cat G is seasonal

library(ggplot2)
y = ts(as.ts(data_line.FitNG.PartCat.train$PartCat_A_log_trans),frequency=12)
decomp = decompose(y)
plot(decomp)
#Time Series part cat A is seasonal and has a linear trend


library(ggplot2)
y = ts(as.ts(data_line.FitNG.PartCat.train$PartCat_F_log_trans),frequency=12)
decomp = decompose(y)
plot(decomp)

#Time Series part cat F is seasonal and has a linear trend


#Difference at lag 12 to remove seasonality from Cat G and lag1 to remove trend
```

```
data_line.FitNG.PartCat.train.PartCat_G.lag12 =
diff(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt,lag=12)
plot.ts(data_line.FitNG.PartCat.train.PartCat_G.lag12,main="U_t differenced at lag12")
var(data_line.FitNG.PartCat.train.PartCat_G.lag12)
fit <- lm(data_line.FitNG.PartCat.train.PartCat_G.lag12 ~
as.numeric(1:length(data_line.FitNG.PartCat.train.PartCat_G.lag12)))
mean(data_line.FitNG.PartCat.train.PartCat_G.lag12)
abline(h=mean(data_line.FitNG.PartCat.train.PartCat_G.lag12), col="blue")
data_line.FitNG.PartCat.train.PartCat_G.stat =
diff(data_line.FitNG.PartCat.train.PartCat_G.lag12,lag=1)

#Diff at lag 12 to remove seasonality from A
data_line.FitNG.PartCat.train.PartCat_A.lag12 =
diff(data_line.FitNG.PartCat.train$PartCat_A_log_trans,lag=12)
plot.ts(data_line.FitNG.PartCat.train.PartCat_A.lag12,main="U_t differenced at lag12")
var(data_line.FitNG.PartCat.train.PartCat_A.lag12)
fit <- lm(data_line.FitNG.PartCat.train.PartCat_A.lag12 ~
as.numeric(1:length(data_line.FitNG.PartCat.train.PartCat_A.lag12)))
mean(data_line.FitNG.PartCat.train.PartCat_A.lag12)
abline(h=mean(data_line.FitNG.PartCat.train.PartCat_A.lag12), col="blue")


#Diff at lag 1 to remove trend
data_line.FitNG.PartCat.train.PartCat_A.stat =
diff(data_line.FitNG.PartCat.train.PartCat_A.lag12,lag=1)
plot.ts(data_line.FitNG.PartCat.train.PartCat_A.stat,main = "Cat A differenced at lag 1 and
lag12")


#Difference at lag 12 to remove seasonality from F
data_line.FitNG.PartCat.train.PartCat_F.lag12 =
diff(data_line.FitNG.PartCat.train$PartCat_F_log_trans,lag=12)
plot.ts(data_line.FitNG.PartCat.train.PartCat_F.lag12,main="U_t differenced at lag12")
var(data_line.FitNG.PartCat.train.PartCat_F.lag12)
fit <- lm(data_line.FitNG.PartCat.train.PartCat_F.lag12 ~
as.numeric(1:length(data_line.FitNG.PartCat.train.PartCat_F.lag12)))
mean(data_line.FitNG.PartCat.train.PartCat_F.lag12)
abline(h=mean(data_line.FitNG.PartCat.train.PartCat_F.lag12), col="blue")


#Difference at lag 1 to remove trend from F
data_line.FitNG.PartCat.train.PartCat_F.stat =
diff(data_line.FitNG.PartCat.train.PartCat_F.lag12,lag=1)
```

```
plot.ts(data_line.FitNG.PartCat.train.PartCat_F.stat,main = "Cat A differenced at lag 1 and lag12")


#ACFS AND PACFS plots of Cat G
hist(data_line.FitNG.PartCat.train.PartCat_G.stat, col="light blue", xlab="", main="histogram; U_t differenced at lags 12 and lag1")
hist(data_line.FitNG.PartCat.train.PartCat_G.stat, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m<-mean(data_line.FitNG.PartCat.train.PartCat_G.stat)
std<- sqrt(var(data_line.FitNG.PartCat.train.PartCat_G.stat))
curve( dnorm(x,m,std), add=TRUE )
acf(data_line.FitNG.PartCat.train.PartCat_G.stat, lag.max=60, main="ACF of cAT G, differenced at lags 12 and lags1")
pacf(data_line.FitNG.PartCat.train.PartCat_G.stat, lag.max=60, main="PACF of CAT G, differenced at lags 12 and lags1")



#ACFS AND PACFS plots of Cat A
hist(data_line.FitNG.PartCat.train.PartCat_A.stat, col="light blue", xlab="", main="histogram; CAT A differenced at lags 12 and lag 1")
hist(data_line.FitNG.PartCat.train.PartCat_A.stat, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m<-mean(data_line.FitNG.PartCat.train.PartCat_A.stat)
std<- sqrt(var(data_line.FitNG.PartCat.train.PartCat_A.stat))
curve( dnorm(x,m,std), add=TRUE )
acf(data_line.FitNG.PartCat.train.PartCat_A.stat, lag.max=60, main="ACF of cAT A, differenced at lags 12 and lag1")

pacf(data_line.FitNG.PartCat.train.PartCat_A.stat, lag.max=60, main="PACF of CAT A, differenced at lags 12 and lags 1")

#ACFS AND PACFS plots of Cat F
hist(data_line.FitNG.PartCat.train.PartCat_F.stat, col="light blue", xlab="", main="histogram; CAT F differenced at lags 12 and lag 1")
hist(data_line.FitNG.PartCat.train.PartCat_F.stat, density=20,breaks=20, col="blue", xlab="", prob=TRUE)
m<-mean(data_line.FitNG.PartCat.train.PartCat_A.stat)
std<- sqrt(var(data_line.FitNG.PartCat.train.PartCat_F.stat))
curve( dnorm(x,m,std), add=TRUE )
acf(data_line.FitNG.PartCat.train.PartCat_F.stat, lag.max=60, main="ACF of cAT F, differenced at lags 12 and lag1")
```

```
pacf(data_line.FitNG.PartCat.train.PartCat_F.stat, lag.max=60, main="PACF of CAT F,
differenced at lags 12 and lags 1")



#First we will do forecasting for Part G
#AFTER ANALYSING THE PACFS AND ACFS, we get q = 2, Q = 4, d = 0,
p=2,P=4,D=1,s=12
model1 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,1),seasonal=list(order=c(0,1,1),period = 12), method="ML")
model1.1 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,2),seasonal=list(order=c(0,1,1),period = 12), method="ML")
model1.2 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,3),seasonal=list(order=c(0,1,1),period = 12), method="ML")
model1.3 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,1),seasonal=list(order=c(0,1,2),period = 12), method="ML")
model1.4 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,2),seasonal=list(order=c(0,1,2),period = 12), method="ML")
model1.5 =arima(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt, order =
c(1,1,2),seasonal=list(order=c(0,1,2),period = 12), method="ML")




#To check invertiblity and stationarity of the model
install.packages('UnitCircle')
library('UnitCircle')

uc.check(pol=c(1,-.2311),plot_output=TRUE) ## non-Seasonal AR part
uc.check(pol=c(1, -.9450 ,0.4068,-.3903)) # Non - seasonal MR part
uc.check(pol=c(1,-.6789)) #seasonal MA part
#Change the coefficients

#Model 1 passes the stationarity and invertiblity test
#To test the residuals
res = residuals(model1.2)
hist(res,density=20,breaks=20,col="blue",xlab="",prob=TRUE)
m = mean(res)
std = sqrt(var(res))
curve(dnorm(data_line.FitNG.PartCat.train$PartCat_G,m,std),add=TRUE)
plot.ts(res)
fitt = lm(res~as.numeric(1:length(res)))
abline(fitt,col="red")
```

```
abline(h-mean(res),col="blue")
qqnorm(res,main="Normal Q-Q PLOT FOR MODEL 1")
qqline(res,col="blue")
##ACF AND PACF of the residuals and ACF of res^2
acf(res,lag.max=40)
pacf(res,lag.max=40)
acf(res^2,lag.max=40)

#Residual tests
Box.test(res,lag=8,type=c("Box-Pierce"),fitdf=4)#Box-Pierce Test
Box.test(res,lag=8,type=c("Ljung-Box"),fitdf=4)#Ljung-Box
Box.test(res^2,lag=8,type=c("Ljung-Box"),fitdf=0)#McLeoad-Li Test
shapiro.test(res)
ar(res,aic=TRUE,order.max=NULL,method=c("yule-walker"))


install.packages("forecast")
library(forecast)
forecast(model1.2)
library(forecast)
#To produce graph with 9 forecasts on transformed data:
pred.tr <- predict(model1.2, n.ahead = 9)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound of prediction interval
ts.plot(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt,xlim=c(1,length(data_line.FitNG.Part
Cat.train$TimeStamp)+9), ylim=c(-.02,0.28))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt)+1):
(length(data_line.FitNG.PartCat.train$PartCat_G_rec_sqrt)+9), pred.tr$pred, col="red")

#To produce graph with 9 forecasts on true  data:
pred.orig <- 1/pred.tr$pred^2
U= 1/U.tr^2
L= 1/L.tr^2
ts.plot(data_line.FitNG.PartCat.train$PartCat_G,
xlim=c(1,length(data_line.FitNG.PartCat.train$TimeStamp)+9), ylim = c(0,4000))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_G)+1):
(length(data_line.FitNG.PartCat.train$PartCat_G)+9), pred.orig, col="red")
```

```
#To zoom the graph, starting from entry 40:
ts.plot(data_line.FitNG.PartCat.train$PartCat_G, xlim =
c(58,length(data_line.FitNG.PartCat.train$PartCat_G)+9), ylim = c(-20,4000))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_G)+1):
(length(data_line.FitNG.PartCat.train$PartCat_G)+9),pred.orig , col="red")


#To plot zoomed forecasts and true values (in ap):
ts.plot(data_line.FitNG.PartCat.train$PartCat_G, xlim =
c(58,length(data_line.FitNG.PartCat.train$PartCat_G)+9), ylim = c(-20,4000), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_G)+1):
(length(data_line.FitNG.PartCat.train$PartCat_G)+9), pred.orig, col="green")
points((length(data_line.FitNG.PartCat.train$PartCat_G)+1):
(length(data_line.FitNG.PartCat.train$PartCat_G)+9),data_line.FitNG.PartCat.test$PartCat_G ,
col="black")



traindf1 = data.frame(TimeStamp = data_line.FitNG.PartCat.train$TimeStamp, Quantity
=data_line.FitNG.PartCat.train$PartCat_G )
preddf1 =  data.frame(TimeStamp = data_line.FitNG.PartCat.test$TimeStamp, Quantity
=pred.orig)

df1 = rbind(traindf1,preddf1)


plot(df1$TimeStamp,                          # Draw first time series
    df1$Quantity,
    type = "l",
    col = 2,
    ylim = c(- 15, 3500),
    xlab = "Year",
    ylab = "Values")
lines(data_line.FitNG.PartCat$TimeStamp,                          # Draw second time series
     data_line.FitNG.PartCat$PartCat_G,
     type = "l",
     col = 3)
```

```
legend("topright",
       c("Predicted time series for category G", "Original time series for Category g"),
       lty = 4,
       col = 2:3)



data.frame(data_line.FitNG.PartCat.test$PartCat_G,pred.orig,abs(data_line.FitNG.PartCat.test$PartCat_G-pred.orig))

MEAN_ABS_ERR1 = sum(abs(data_line.FitNG.PartCat.test$PartCat_G-pred.orig))/9

accuracy(pred.orig,data_line.FitNG.PartCat.test$PartCat_G)


##Here we see a very high mean absolute error because the values are very large.
##Thus we should MEAN ABS PERCENTAGE ERROR

MEAN_ABS_PER_ERR1 = sum(abs(data_line.FitNG.PartCat.test$PartCat_G-pred.orig)/
data_line.FitNG.PartCat.test$PartCat_G*100)/9

##Removing the outliers as 7th, 8th, and 9th observations, our MAPE improves to 55.28


#################PREDICTION FOR CATEGORY A
##########################################

library(rgl)
library(qpcR)
model2 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(1,1,1),seasonal=list(order=c(1,1,1),period=12),method="ML")
model2.1 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(2,1,1),seasonal=list(order=c(1,1,1),period=12),method="ML")
model2.2 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(3,1,1),seasonal=list(order=c(1,1,1),period=12),method="ML")
model2.3 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(4,1,1),seasonal=list(order=c(1,1,1),period=12),method="ML")
model2.4 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(1,1,1),seasonal=list(order=c(1,1,2),period=12),method="ML")
model2.5 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(2,1,1),seasonal=list(order=c(1,1,2),period=12),method="ML")
model2.6 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(3,1,1),seasonal=list(order=c(1,1,2),period=12),method="ML")
```

```
model2.7 =arima(data_line.FitNG.PartCat.train$PartCat_A_log_trans, order =
c(4,1,1),seasonal=list(order=c(1,1,2),period=12),method="ML")




install.packages('UnitCircle')
library('UnitCircle')

uc.check(pol=c(1,-.2576,-.0105),plot_output=TRUE) ## non-Seasonal AR part
uc.check(pol=c(1, -.1143 ,0.4260)) # Non - seasonal MR part


res = residuals(model2.3)
hist(res,density=20,breaks=20,col="blue",xlab="",prob=TRUE)
m = mean(res)
std = sqrt(var(res))
curve(dnorm(data_line.FitNG.PartCat.train$PartCat_G,m,std),add=TRUE)
plot.ts(res)
fitt = lm(res~as.numeric(1:length(res)))
abline(fitt,col="red")
abline(h-mean(res),col="blue")
qqnorm(res,main="Normal Q-Q PLOT FOR MODEL 1")
qqline(res,col="blue")
##ACF AND PACF of the residuals and ACF of res^2
acf(res,lag.max=40)
pacf(res,lag.max=40)
acf(res^2,lag.max=40)

#Residual tests
Box.test(res,lag=8,type=c("Box-Pierce"),fitdf=7)#Box-Pierce Test
Box.test(res,lag=8,type=c("Ljung-Box"),fitdf=7)#Ljung-Box
Box.test(res^2,lag=8,type=c("Ljung-Box"),fitdf=0)#McLeoad-Li Test
shapiro.test(res)
ar(res,aic=TRUE,order.max=NULL,method=c("yule-walker"))


install.packages("forecast")
library(forecast)
forecast(model2.3)
library(forecast)
#To produce graph with 15 forecasts on transformed data:
pred.tr <- predict(model2.3, n.ahead = 9)
```

```
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound of prediction interval
ts.plot(data_line.FitNG.PartCat.train$PartCat_A_log_trans,xlim=c(1,length(data_line.FitNG.Part
Cat.train$TimeStamp)+9), ylim = c(0,10))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_A_log_trans)+1):
(length(data_line.FitNG.PartCat.train$PartCat_A_log_trans)+9), pred.tr$pred, col="red")


pred.orig <- exp(pred.tr$pred)
U= exp(U.tr)
L= exp(L.tr)
ts.plot(data_line.FitNG.PartCat.train$PartCat_A,
xlim=c(1,length(data_line.FitNG.PartCat.train$TimeStamp)+9), ylim = c(0,200))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_A)+1):
(length(data_line.FitNG.PartCat.train$PartCat_A)+9), pred.orig, col="red")



#To zoom the graph, starting from entry 40:
ts.plot(data_line.FitNG.PartCat.train$PartCat_A, xlim =
c(30,length(data_line.FitNG.PartCat.train$PartCat_A)+9), ylim = c(-2,200))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_A)+1):
(length(data_line.FitNG.PartCat.train$PartCat_A)+9), pred.orig, col="red")


#To plot zoomed forecasts and true values (in ap):
ts.plot(data_line.FitNG.PartCat.train$PartCat_A, xlim =
c(38,length(data_line.FitNG.PartCat.train$PartCat_A)+9), ylim = c(-2,200), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_A)+1):
(length(data_line.FitNG.PartCat.train$PartCat_A)+9), pred.orig, col="red")
points((length(data_line.FitNG.PartCat.train$PartCat_A)+1):
(length(data_line.FitNG.PartCat.train$PartCat_A)+9),data_line.FitNG.PartCat.test$PartCat_A ,
col="black")
```

```
data_line.FitNG.PartCat.train$TimeStamp
data_line.FitNG.PartCat.test$PartCat_A


data_line.FitNG.PartCat$TimeStamp


cbind(data_line.FitNG.PartCat.train$PartCat_G,pred.orig)


traindf2 = data.frame(TimeStamp = data_line.FitNG.PartCat.train$TimeStamp, Quantity
=data_line.FitNG.PartCat.train$PartCat_A )
preddf2 =  data.frame(TimeStamp = data_line.FitNG.PartCat.test$TimeStamp, Quantity
=pred.orig)

df1 = rbind(traindf2,preddf2)


plot(df1$TimeStamp,                          # Draw first time series
    df1$Quantity,
    type = "l",
    col = 2,
    ylim = c(- 15, 350),
    xlab = "Year",
    ylab = "Values")
lines(data_line.FitNG.PartCat$TimeStamp,                          # Draw second time series
     data_line.FitNG.PartCat$PartCat_A,
     type = "l",
     col = 3)
legend("topright",
      c("Predicted time series for category A", "Original time series for Category A"),
      lty = 4,
      col = 2:3)


data.frame(data_line.FitNG.PartCat.test$PartCat_A,pred.orig,abs(data_line.FitNG.PartCat.test$P
artCat_A-pred.orig))

MEAN_ABS_ERR2 = sum(abs(data_line.FitNG.PartCat.test$PartCat_A-pred.orig))/9
```

```
MEAN_ABS_PER_ERR2 = sum(abs(data_line.FitNG.PartCat.test$PartCat_A-pred.orig)/
data_line.FitNG.PartCat.test$PartCat_A*100)/9
```

```
##Removing the outliers (3rd, 6th, 8th observation) our MAPE improves to 38.57

#######Forecasting for Model F ##############

model3 =arima(data_line.FitNG.PartCat.train$PartCat_F_log_trans, order =
c(1,1,1),seasonal=list(order=c(0,1,1),period=12),method="ML")

res = residuals(model3)
hist(res,density=20,breaks=20,col="blue",xlab="",prob=TRUE)
m = mean(res)
std = sqrt(var(res))
curve(dnorm(data_line.FitNG.PartCat.train$PartCat_G,m,std),add=TRUE)
plot.ts(res)
fitt = lm(res~as.numeric(1:length(res)))
abline(fitt,col="red")
abline(h-mean(res),col="blue")
qqnorm(res,main="Normal Q-Q PLOT FOR MODEL 1")
qqline(res,col="blue")
##ACF AND PACF of the residuals and ACF of res^2
acf(res,lag.max=40)
pacf(res,lag.max=40)
acf(res^2,lag.max=40)

#Residual tests
Box.test(res,lag=8,type=c("Box-Pierce"),fitdf=2)#Box-Pierce Test
Box.test(res,lag=8,type=c("Ljung-Box"),fitdf=2)#Ljung-Box
Box.test(res^2,lag=8,type=c("Ljung-Box"),fitdf=0)#McLeoad-Li Test
shapiro.test(res)
ar(res,aic=TRUE,order.max=NULL,method=c("yule-walker"))

forecast(model3)
library(forecast)
#To produce graph with 9 forecasts on transformed data:
```

```
pred.tr <- predict(model3, n.ahead = 9)
U.tr= pred.tr$pred + 2*pred.tr$se # upper bound of prediction interval
L.tr= pred.tr$pred - 2*pred.tr$se # lower bound of prediction interval
ts.plot(data_line.FitNG.PartCat.train$PartCat_F_log_trans,xlim=c(1,length(data_line.FitNG.Part
Cat.train$TimeStamp)+9), ylim = c(-2,5))
lines(U.tr, col="blue", lty="dashed")
lines(L.tr, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_F_log_trans)+1):
(length(data_line.FitNG.PartCat.train$PartCat_F_log_trans)+9), pred.tr$pred, col="red")



#To produce graph with 9 forecasts on true data:
pred.orig <- exp(pred.tr$pred)
U= exp(U.tr)
L= exp(L.tr)
ts.plot(data_line.FitNG.PartCat.train$PartCat_F,
xlim=c(1,length(data_line.FitNG.PartCat.train$TimeStamp)+9), ylim = c(0,200))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_F)+1):
(length(data_line.FitNG.PartCat.train$PartCat_F)+9), pred.orig, col="red")



#To zoom the graph, starting from entry 40:
ts.plot(data_line.FitNG.PartCat.train$PartCat_F, xlim =
c(30,length(data_line.FitNG.PartCat.train$PartCat_F)+9), ylim = c(-50,300))
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_F)+1):
(length(data_line.FitNG.PartCat.train$PartCat_F)+9), pred.orig, col="red")


#To plot zoomed forecasts and true values (in ap):
ts.plot(data_line.FitNG.PartCat.train$PartCat_F, xlim =
c(38,length(data_line.FitNG.PartCat.train$PartCat_F)+9), ylim = c(-50,300), col="red")
lines(U, col="blue", lty="dashed")
lines(L, col="blue", lty="dashed")
points((length(data_line.FitNG.PartCat.train$PartCat_F)+1):
(length(data_line.FitNG.PartCat.train$PartCat_F)+9), pred.orig, col="red")
```

```
points((length(data_line.FitNG.PartCat.train$PartCat_F)+1):
(length(data_line.FitNG.PartCat.train$PartCat_F)+9),data_line.FitNG.PartCat.test$PartCat_F ,
col="black")
```

```
traindf3 = data.frame(TimeStamp = data_line.FitNG.PartCat.train$TimeStamp, Quantity
=data_line.FitNG.PartCat.train$PartCat_F )
preddf3 =  data.frame(TimeStamp = data_line.FitNG.PartCat.test$TimeStamp, Quantity
=pred.orig)

df1 = rbind(traindf3,preddf3)


plot(df1$TimeStamp,                          # Draw first time series
    df1$Quantity,
    type = "l",
    col = 2,
    ylim = c(- 15, 350),
    xlab = "Year",
    ylab = "Values")
lines(data_line.FitNG.PartCat$TimeStamp,                          # Draw second time series
     data_line.FitNG.PartCat$PartCat_F,
     type = "l",
     col = 3)
legend("topright",
      c("Predicted time series for category F", "Original time series for Category F"),
      lty = 4,
      col = 2:3)


data.frame(data_line.FitNG.PartCat.test$PartCat_F,pred.orig,abs(data_line.FitNG.PartCat.test$P
artCat_F-pred.orig))

MEAN_ABS_ERR3 = sum(abs(data_line.FitNG.PartCat.test$PartCat_F-pred.orig))/9

MEAN_ABS_PER_ERR3 = sum(abs(data_line.FitNG.PartCat.test$PartCat_F-pred.orig)/
data_line.FitNG.PartCat.test$PartCat_F*100)/9
```

##Removing observations 2,3, and 4th as outliers, we get MAPE as 36.67