**SALES FORECASTING**

AIM OF PROJECT – To forecast the sales of next 30 days

PRODUCT- Sales (tourism)

TOOLS- Excel and R

TECHNIQUE- analyzing the time series data

* Preprocessed the actual data with Excel (with formatting the dates) and R
* Aggregated the sales day wise with the mean value
* Looking for forecasting methods to fit

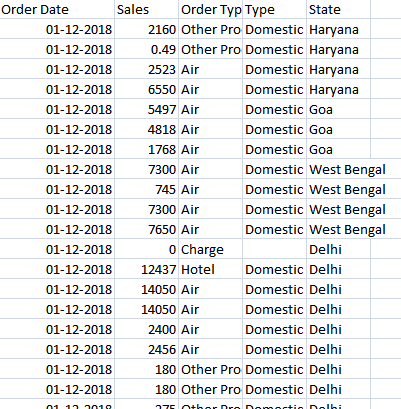
Actual Data

**Composition of data :** Order date, sales, state, order type and type

Order date had some observations with time so, they were formatted in excel and some in R.

NAs were removed.

Formatted data attached with the folder.



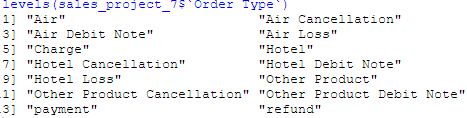
Reading the excel file:

library(readxl)

sales\_project\_7 <- read\_excel("C:/Users/KISHOR/Desktop/Sales\_forecast/sales\_project 7.xlsx")

View(sales\_project\_7)

Levels in order type

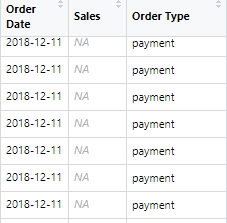


**Bringing the date into as.Date format**:

sales\_project\_7$`Order Date` <- as.Date(sales\_project\_7$`Order Date`, "%Y%m%d")

sapply(sales\_project\_7,class)

Some part of the data also have NAs in sales :



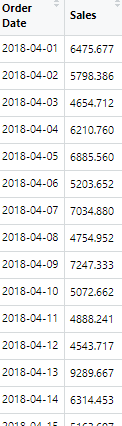
Replacing such NAs wit zeroes: sales\_new[is.na(sales\_new)] <- 0

Sales (average) calculated :

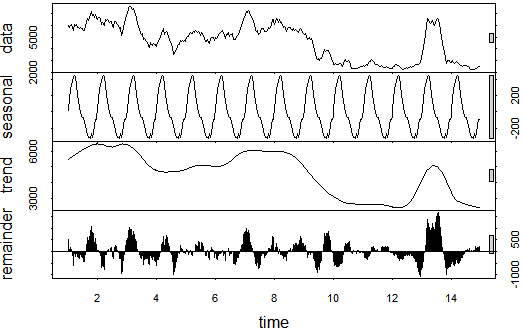
sales\_new <- data.table(sales\_new)

sales\_new <- sales\_new[, mean(Sales), by = `Order Date`]

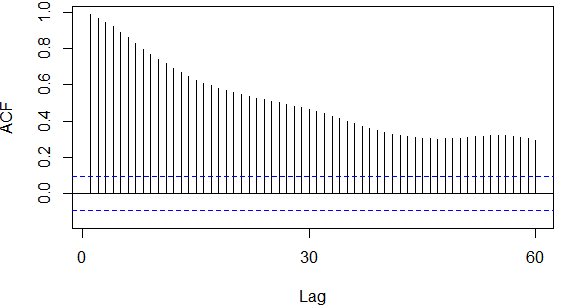
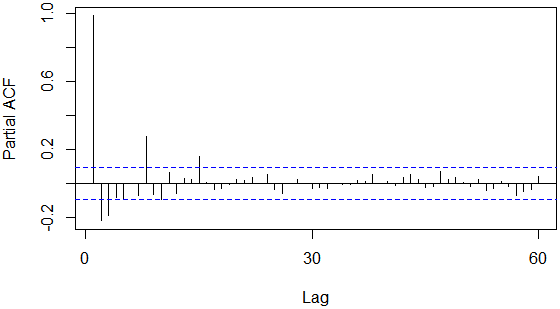
Final data:



Data visualization



ACF and PACF plots on cleaned data:



#ARIMA model

library(ggplot2)

library(forecast)

library(tseries)

ggplot(sales\_new,aes(`Order Date`,Sales))+geom\_line()+scale\_x\_date('month')+ylab("Sales")+xlab("")

sales\_ts = ts(sales\_new[,c('Sales')])

#removing outliers as there are sales amount that are beynd 50000

sales\_new$clean\_sales = tsclean(sales\_ts)

ggplot() +geom\_line(data = sales\_new, aes(x = `Order Date`, y = clean\_sales)) + ylab('Cleaned Sales Data')

#plotting the moving average for weekly and monthly using cleaned data

sales\_new$sale\_ma = ma(sales\_new$clean\_sales,order = 7)

sales\_new$sale\_ma30 = ma(sales\_new$clean\_sales,order = 30)

#plotted MAs with actual data

ggplot() +

geom\_line(data = sales\_new, aes(x = `Order Date`, y = clean\_sales, colour = "Sales")) +

geom\_line(data = sales\_new, aes(x = `Order Date`, y = sale\_ma, colour = "Weekly Moving Average")) +

geom\_line(data = sales\_new, aes(x = `Order Date`, y = sale\_ma30, colour = "Monthly Moving Average")) +

ylab('Sales')

#deseasonliaze data of the weekly moving average series for simplicity

count\_ma = ts(na.omit(sales\_new$sale\_ma),frequency = 30)

decomp = stl(count\_ma,s.window = "periodic")

deseasonal\_sales = seasadj(decomp)

plot(decomp)

#decomp1 = stl(count\_ma,s.window = "periodic",allow.multiplicative.trend=TRUE)

#checking stationary with Dickey\_Fuller test

adf.test(count\_ma, alternative = "stationary")#p-value =0.3355(non -stationary)

Acf(count\_ma,main='')#geometric(many lags)

Pacf(count\_ma,main='')#lags at 1,2,3,8,15

#differencing is needed to make the data stationary

#first taking d=1

count\_d1 = diff(deseasonal\_sales,differences = 1)

plot(count\_d1)

adf.test(count\_d1,alternative = "stationary")#p-value = 0.01 (<0.05)

#so data has now become stationary

#plotting acf and pacf on the differenced series

Acf(count\_d1,main='Acf for differenced series')

#lags at 1..7

Pacf(count\_d1,main='Pacf for differenced series')

#lags at 1,2,3,..,7,14

hold <- window(ts(deseasonal\_sales),start = 390)

fit\_no\_holdout <- arima(ts(deseasonal\_sales[-c(390:419)]),order=c(1,1,7))

fcast\_no\_holdout <- forecast(fit\_no\_holdout,h=29)

plot(fcast\_no\_holdout,main="")

lines(ts(deseasonal\_sales))

#the forecast doesnt match the actual data

#trying auto arima in the whole data including seasonality

fit\_w\_seasonality = auto.arima(deseasonal\_sales, seasonal=TRUE)

fit\_w\_seasonality

#log likelihood=-2743.34 AIC=5506.69 AICc=5507.23 BIC=5547.04

#forecasting for the next 30 days

seas\_fcast <- forecast(fit\_w\_seasonality,h=30)

plot(seas\_fcast,main = "")

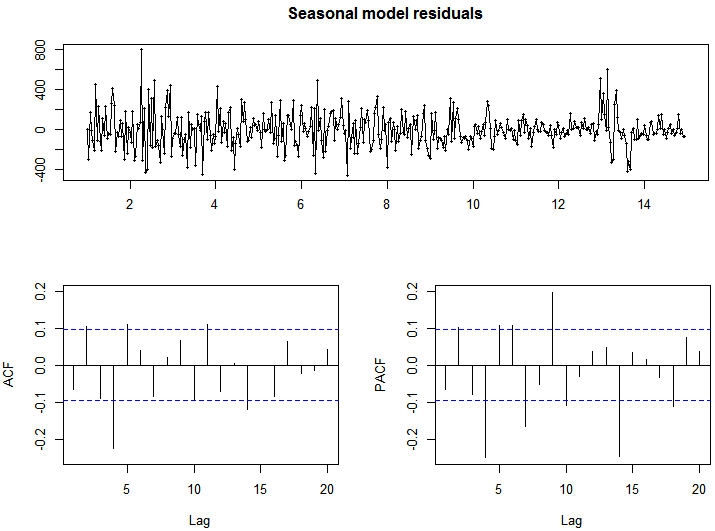
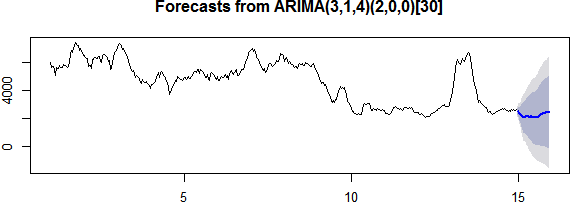
accuracy(fcast\_no\_holdout,hold)

#375.6383 13.959980

#ARIMA(3,1,4)(2,0,0)[30]

tsdisplay(residuals(fit\_w\_seasonality),lag.max = 20,main = "Seasonal model residuals")

#its found that the acf and pacf plots are showing lags at many places



#SIMPLE EXPONENTIAL SMOOTHING

library(tidyverse)

sales\_ts = ts(sales\_new[,c('Sales')])

#cleaning data

sales\_new$clean\_sales = tsclean(sales\_ts)

sales\_ses <- sales\_new[,-c(1,2)]

sale.train <- window(sales\_ses,end = 395)

sale.test <- window(sales\_ses,start = 396)

ses.sale <- ses(sale.train,alpha = .2,h=30)

autoplot(ses.sale)

#diiferencing the data

sale.diff <- diff(sale.train)

autoplot(ts(sale.diff))

ses.sale.diff <- ses(sale.diff,alpha= .2,h=30)

autoplot(ses.sale.diff)

#created differenced test data

sale.diff.test <- diff(sale.test)

accuracy(ses.sale.diff,sale.diff.test)#1126.444,120.1252

#797.5764 , 121.1201

#494.0383 , 770.8394

#identify optimal alpha value

alpha <- seq(.01,.99,by=.01)

RMSE <- NA

for(i in seq\_along(alpha)){

fit <- ses(sale.diff,alpha=alpha[i],h=30)

RMSE[i] <- accuracy(fit,sale.diff.test)[2,2]

}

#convert to a data frame and idenitify min alpha value

alpha.fit <- data.frame(alpha,RMSE)

alpha.min <- filter(alpha.fit,RMSE==min(RMSE))

#alpha = 0.04,RMSE= 1126.046

#0.04 ,797.0132

#0.01 ,464.3777

# plot RMSE vs. alpha

ggplot(alpha.fit, aes(alpha, RMSE)) +

geom\_line() +

geom\_point(data = alpha.min, aes(alpha, RMSE), size = 2, color = "blue")

#refit model with alpha = .01

ses.sale.opt <- ses(sale.diff, alpha = .01, h = 30)

# performance eval

accuracy(ses.sale.opt, sale.diff.test)

#1126.046,106.4788

#464.3777,151.2718

# plotting results

p1 <- autoplot(ses.sale.opt)

p2 <- autoplot(ts(sale.diff.test)) +

autolayer(ses.sale.opt, alpha = .4) +

ggtitle("Predicted vs. actuals for the test data set")

gridExtra::grid.arrange(p1, p2, nrow = 1)

#so its seen SES doesnt perform well

#HOLT’S METHOD

holt.sale <- holt(sale.train,h=30)

autoplot(holt.sale)

holt.sale$model

#Smoothing parameters:

#alpha = 0.1928 , 0.2169 , 0.3159

#beta = 1e-04

#AIC AICc BIC

#6464.118 6464.309 6482.960

#6329.324 6329.515 6348.166

#7886.640 7886.794 7906.534

accuracy(holt.sale,sale.test)

#381.6096 13.25246

# identify optimal beta parameter

beta <- seq(.0001, .5, by = .001)

RMSE <- NA

for(i in seq\_along(beta)) {

fit <- holt(sale.train, beta = beta[i], h = 30)

RMSE[i] <- accuracy(fit, sale.test)[2,2]

}

# convert to a data frame and idenitify min beta value

beta.fit <- data\_frame(beta, RMSE)

beta.min <- filter(beta.fit, RMSE == min(RMSE))

#beta min = 0.278 ,0.279, 0.174

# plot RMSE vs. beta

ggplot(beta.fit, aes(beta, RMSE)) +

geom\_line() +

geom\_point(data = beta.min, aes(beta, RMSE), size = 2, color = "blue")

# new model with optimal beta

holt.sale.opt <- holt(sale.train, h = 30, beta = 0.174)

# accuracy of first model

accuracy(holt.sale, sale.test)

#2204.517,31.71329

#381.6096, 13.25246

# accuracy of new optimal model

accuracy(holt.sale.opt, sale.test)

#1797.488,30.67075

# 1996.238,31.50727

#339.4273, 11.32062

# on whole data

model <- holt(sales\_ses,h = 30, beta = 0.174)

plot(model)

p1 <- autoplot(holt.sale) +

ggtitle("Original Holt's Model")

p2 <- autoplot(holt.sale.opt) +

ggtitle("Optimal Holt's Model")

gridExtra::grid.arrange(p1, p2, nrow = 1)

#HOLT WINTER’S METHOD

autoplot(decompose(sales\_new))

sale.hw <- ets(sale.train,model="ZZZ")

#model formed M,N,N

autoplot(forecast(sale.hw))

summary(sale.hw)

checkresiduals(sale.hw)

sale.f1 <- forecast(sale.hw,h=30)

# check accuracy

accuracy(sale.f1, sale.test)

#1860.623,21.07005

# 1644.195,20.55384

#481.4886 18.03020

#plotting

autoplot(sale.f1)

FINAL MODEL

#Final model holts(optimal)

#since it has low RMSE and MAPE value

#339.4273, 11.32062

# model on whole cleaned data

model <- holt(sales\_ses,h = 30, beta = 0.174)

fore\_sale <- data.frame(model)

plot(model)

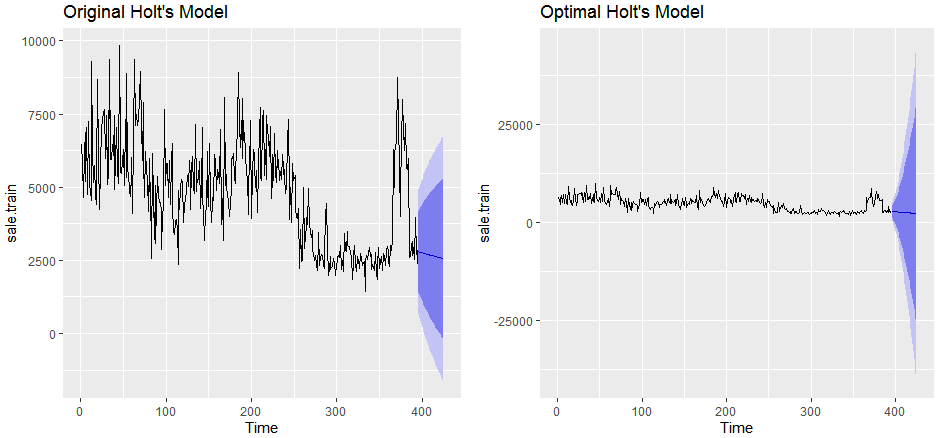
#create dates for next 30 days

startDate <- as.Date("2019-06-11")

xm <- seq(startDate,by = "1 day", length.out = 30)

xm

fore\_sale <- cbind(xm,fore\_sale)



**Forecasted Sales(of cleaned data)**

