Octane Rating Project's R – Code

Group – V:
Goda Venkata Adithya Tarun (MD2106)
Mainack Paul (MD2111)
Vicky Gupta (MD2129)

Data Input

```
Data=read.csv(file="path of the file")
library(car)
A1=Data1[,2]          #Extracting the variable
A2=Data1[,3]          #Extracting the variable
A3=Data1[,4]          #Extracting the variable
A4=Data1[,5]          #Extracting the variable
OR=Data1[,6]          #Extracting the variable
```

Scatter Plots before the removal of outliers

```
scatterplot(OR~A1)   #Plotting
scatterplot(OR~A2)   #Plotting
scatterplot(OR~A3)   #Plotting
scatterplot(OR~A4)   #Plotting
```

Least Squares Fit

```
Xtilda = cbind(Data$A1,Data$A2,Data$A3,Data$A4)
Xstar = cbind(
        (Data$A1-mean(Data$A1))/((sum((Data$A1-mean(Data$A1))^2))^0.5),
        (Data$A2-mean(Data$A2))/((sum((Data$A2-mean(Data$A2))^2))^0.5),
        (Data$A3-mean(Data$A3))/((sum((Data$A3-mean(Data$A3))^2))^0.5),
        (Data$A4-mean(Data$A4))/((sum((Data$A4-mean(Data$A4))^2))^0.5))
X = cbind(rep(1,82),Xtilda)            #regression matrix
Y = Data$Response                      #Response variable
n=82 ; p=5                             #Data size and no. of parameters
model_old = lm(Response~A1+A2+A3+A4,Data)
s = summary(model_old); s              #summary of the fitted regression model
TSS = sum((Y-mean(Y))^2)               #Total sum of squares
RSS = TSS*(1-s$r.squared)              #Residual sum of squares
betacap = s$coefficients[,1]           #fitted regression coefficients
S2 = RSS/(n-p)                         #unbiased estimator of variance of errors
Ycap = as.vector(X%*%betacap)          #fitted values
e = Y-Ycap                             #residuals
ols_plot_obs_fit(model_old)            #Fitting of actual vs fitted for OR
```

F-test for testing $\beta_3 = 0$

```
A = matrix(c(0,0,0,1,0),nrow=1,ncol=5)
q = nrow(A)
F=((t(A%*%betacap))%*%solve(A%*%solve(t(X)%*%X)%*%(t(A)))%*%(A%*%betacap))/(q*S2)
TabF = qf(0.95,q,n-p)
if(F>TabF)
{print("A3 variable is significant by F-test")}
else
{print("A3 variable is insignificant by F-test")}
```

Normality Assumption before checking the outliers

```
library("olsrr")
ols_plot_resid_qq(model_old)          #Q-Q plot of Residuals

library("olsrr")
ols_plot_resid_hist(model_old)        #plotting histogram of residuals

s = shapiro.test(e)
if(s$p.value>0.05)
{print("We do not reject the null hypothesis at 5% level of significance")}
else{print("We reject H0 at 5% level of significance")}
```

Heteroscedasticity before checking the outliers

```
b = e^2/(1-hii)
plot(Ycap,b,main = "bi'svs.Fitted Values",xlab = "Fitted Values",ylab = "b")

r = e/sqrt(S2*(1-hii))
plot(Ycap,r^2,main="ri^2's vs. Fitted Values",xlab="Fitted Values",ylab="r")

ols_plot_resid_fit(model_old)          # Residual Plot

install.packages("lmtest")             #install this library is not installed yet
library(lmtest)                        #Load the lmtest library
bptest(model_old)                      #Command to run the BP Test
```

Auto – Correlation of errors before removal of outliers

```
acf(e,xlab="Time Lag",ylab="ACF",main="Correlogram")

num = 0; for (i in 2:n) { num = num+(e[i]-e[i-1])^2 }
d = num/sum(e^2); d

cor(Xstar)                             #R code for correlation matrix
```

Multicollinearity before the removal of outliers

```
install.packages("GGally")                #install GGally package if not installed yet
library("GGally")                         #Load the GGally library
ggpairs(Data[,-c(1,6)])                   #This generates scatter plot of all thevariables

install.packages("ppcor")                 #install this library if not yetinstalled
library(ppcor)
pcor(Data[,-c(1,6)],method="pearson")     #Calculating Partial Correlation Matrix

vif(model_old)

l = eigen(t(Xstar)%*%Xstar)$values
sqrt(max(l)/min(l))
```

Partial Residual Plots before the removal of outliers

```
library(conf)
crPlots(model_old)
```

Added variable Plots before the removal of outliers

```
ols_plot_added_variable(model_old)
```

Goodness of fit before the removal of outliers

```
model_old = lm(OR~A1+A2+A3+A4)
summary(model_old)
```

Testing for significance of parameters before the removal of outliers

```
model_old = lm(OR~A1+A2+A3+A4)
summary(model_old)
```

Outliers, High – Leverage and Influential Points

```
H = X%*%solve(t(X)%*%X)%*%t(X) #hat matrix
hii = numeric(length=0)
for (i in 1:n) { hii[i] = H[i,i]} #hat matrix diagonal
cases_1 = numeric(length=0) #cases where hi > 2p/n
for (i in 1:n) { if (hii[i]>2*p/n) {cases_1[length(cases_1)+1] = i} }
plot(hii)
lines(x=1:82,y=rep(2*p/n,82),col="red",lty=2)
points(x=cases_1,hii[cases_1],col="red",text(cases_1,hii[cases_1]),
labels=as.character(cases_1),pos=2,ces=0.9)
```

```
Si2 = (1/(n-p-1))*((n-p)*S2-(e^2)/(1-hii))   #Studentized Residuals
ti = e/((Si2*(1-hii))^0.5)
cases_2 = numeric(length=0)
for (i in 1:n) { if (abs(ti[i])>2) {cases_2[length(cases_2)+1] = i} }
ols_plot_resid_stud_fit(model_old)


library("olsrr") #install olsrr package if it is not yet installed               #dffits
ols_plot_dfbetas(model_old)
ols_plot_dffits(model_old)
cases_3 = numeric(length=0)
for(i in1:n){if(abs(dffits(model_old)[i])>2*((p/n)^0.5)){cases_3[length(cases_3)+1]=i}}


cr = covratio(model_old)  #covratio
cases_4 = numeric(length=0)
for (i in 1:n) {if (abs(cr[i]-1)>3*(p/n)) {cases_4[length(cases_4)+1] = i}}
plot(1:82,cr,xlab = "Case Number",ylab = "COVRATIO",main = "COVRATIO Plot")
lines(x = 1:82,y = rep(1+3*p/n,82),col="red",lty=2)
lines(x = 1:82,y = rep(1-3*p/n,82),col="red",lty=2)
points(cases_4,cr[cases_4],col="red",text(cases_4,cr[cases_4],
labels=as.character(cases_4),pos=2,cex = 0.9))


Di = cooks.distance(model_old)  #cook's distance
cases_5 = numeric(length=0)
for (i in 1:n) { if (Di[i]>4/n) {cases_5[length(cases_5)+1] = i} }
ols_plot_cooksd_chart(model_old)


all_cases = c(cases_1, cases_2, cases_3, cases_4, cases_5)           #Outlier Testing
suspicious_cases = c(44, 52, 71, 72, 73, 75, 76, 77, 82)
k = length(suspicious_cases)
tmp1 = X[-suspicious_cases,] ; tmp2 = X[suspicious_cases,]
newX = rbind(tmp1,tmp2)
tmp3 = Y[-suspicious_cases] ; tmp4 = Y[suspicious_cases]
newY = c(tmp3,tmp4)
newH = newX%*%solve(t(newX)%*%newX)%*%t(newX)
H22 = newH[(n-k+1):n,(n-k+1):n]
newe = (diag(n)-newH)%*%newY
e2 = as.matrix(newe[(n-k+1):n])
num = (t(e2)%*%solve(diag(k)-H22)%*%e2)
den = (RSS-(t(e2)%*%solve(diag(k)-H22)%*%e2))
outF = ((n-p-k)/k)*(num/den)
newTabF = qf(0.95,k,n-p-k)
if(outF>newTabF){print("The suspicious points are outliers by F-test")}
```

else{print("The suspicious points are not outliers by F-test")}

Scatter Plots after the removal of outliers

```
data=read.csv(file="path of the file")
AM1=data1[,2]                #Extracting the variable
AM2=data1[,3]                #Extracting the variable
AM3=data1[,4]                #Extracting the variable
MCR=data1[,5]                #Extracting the variable
OR_N=data1[,6]               #Extracting the variable
scatterplot(OR_N~AM1)        #Plotting
scatterplot(OR_N~AM2)        #Plotting
scatterplot(OR_N~AM3)        #Plotting
scatterplot(OR_N~MCR)        #Plotting
```

Least Squares Fit after the removal of outliers

```
model_new=lm(OR_N~AM1+AM2+AM3+MCR) #New fitted model
summary(model_new)
OR_hat= 98.045624-0.108919*AM1--0.143369*AM2 -
0.046249*AM3+1.976704*MCR      # Fitted Values
ols_plot_obs_fit(model_new)
```

Normality assumption after the removal of outliers

```
ols_plot_resid_qq(model_new)        #Remember to use the updated data as in the abovesection
ols_plot_resid_hist(model_new)


s = shapiro.test(e) #residuals of the updated model
if(s$p.value>0.05)
{print("We do not reject the null hypothesis at 5% level of significance")}
else{print("We reject H0 at 5% level of significance")}
```

Heteroscedasticity after the removal of outliers

```
b = e^2/(1-hii) #Remember to use the updated data without outliers
plot(Ycap,b,main = "bi's vs. Fitted Values",xlab = "Fitted Values",ylab ="b")


r = e/sqrt(S2*(1-hii))
plot(Ycap,r^2,main="ri^2's vs. Fitted values",xlab="Fitted Values",ylab="r")

ols_plot_resid_fit(model_new)

install.packages("lmtest")           #install this library is not installed yet
library(lmtest)                      #Load the lmtest library
bptest(model_new)                         #Command to run the BP Test
```

Autocorrelation of errors after the removal of outliers

```
library(DescTools)
d=DurbinWatsonTest(model_new)$statistic ; d


library(lmtest)
bgtest(OR_N~AM1+AM2+AM3+MCR)
```

Multicollinearity after the removal of outliers

```
Data=Data[-c(44,52,71,72,73,75,76,77,82),]          #Removing outliersfrom data
ggpairs(Data[,-c(1,6)])                             #This generates scatter plot ofall the variables

pcor(Data[,-c(1,6)],method="pearson")               #Calculating Partial Correlation Matrix

library(car)
model_new = lm(OR_N~AM1+AM2+AM3+MCR,Data)
vif(model_new)
l = eigen(t(Xstar)%*%Xstar)$values                  #use updated Xstar in computation
sqrt(max(l)/min(l))
```

Partial Residual Plots after the removal of outliers

```
crPlots(model_new)
```

Added Variable plots after the removal of outliers

```
ols_plot_added_variable(model_new)
```

Model Selection

```
selection=ols_step_all_possible(model_new)
print(selection)
plot(selection)
```

Testing for significance of model parameters

```
model_new = lm(OR_N~AM1+AM2+AM3+MCR)
summary(model_new)
```