Compressive-strength of concrete

November 6, 2022

This project is based on alanysing the dependence of Compressive strength on Various Input Factors of Concrete Formation.

Objective: To evaluate how Compressive Strength depends on the 7 input variables. To construct an appropriate model by method of regression. Improvising the model based on the validity of the assumptions. To fit an appropriate model to the data and draw inference.

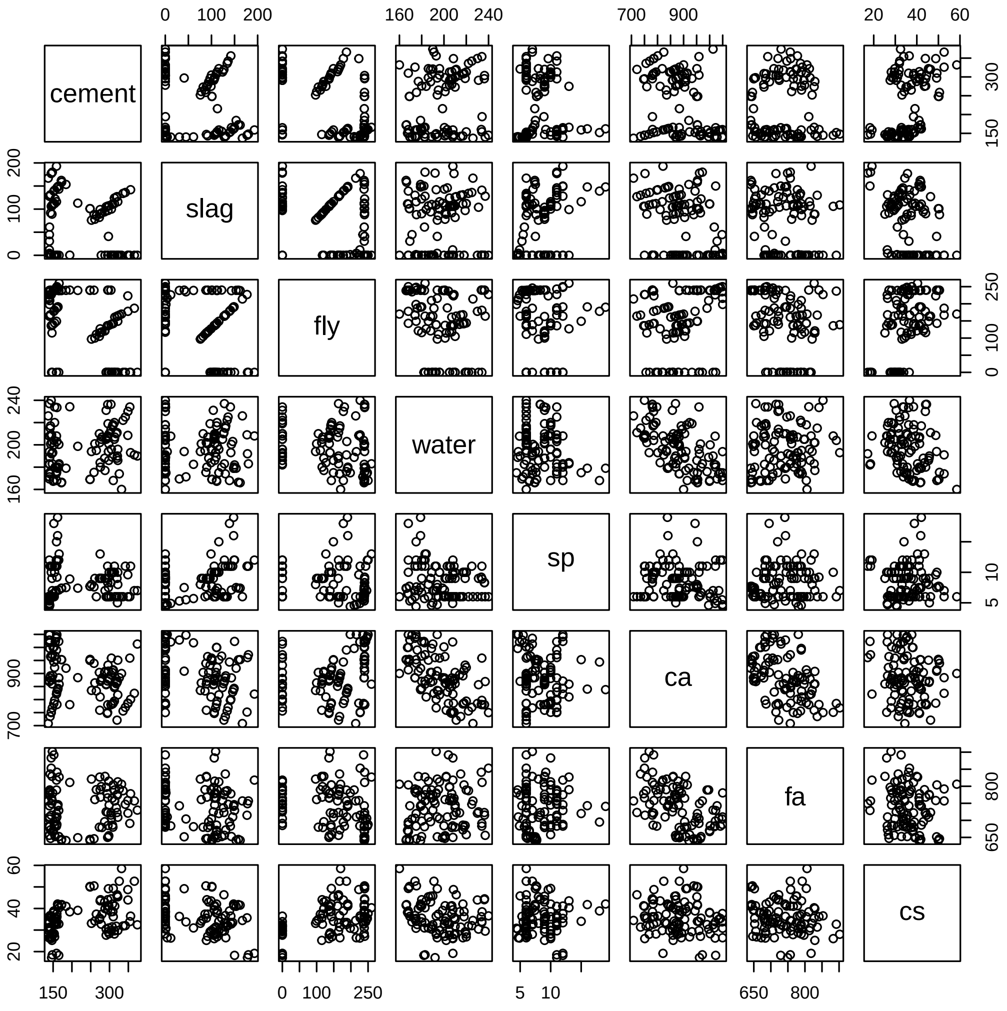
1. **Cement** -- Cement is a binder, a substance used for construction that sets, hardens, and adheres to other materials to bind them together. It mixed with other matreials produces mortar, concrete, etc.
2. **Slag_Cement** -- Slag cement is most widely used in concrete, either as a separate cementitious component or as part of a blended cement. It works synergistically with portland cement to increase strength, reduce permeability, improve resistance to chemical attack and inhibit rebar corrosion.
3. **Fly_Ash** -- Fly ash use in concrete improves the workability of plastic concrete, and the strength and durability of hardened concrete.
4. **Water** -- Input Variable
5. **Superplasticizer** -- Superplasticizers, also known as high range water reducers, are additives used in making high strength concrete. Plasticizers are chemical compounds that enable the production of concrete with approximately 15% less water content. Superplasticizers allow reduction in water content by 30% or more.
6. **Coarse_Aggregate(ca)** -- Coarse aggregates are defined as any material greater than 4.75 mm. Aggregates make up 60-80% of the volume of concrete and 70-85% of the mass of concrete. Aggregate is also very important for strength, thermal and elastic properties of concrete, dimensional stability and volume stability.
7. **Fine_Aggregate(fa)** -- Fine aggregates are usually sand or crushed stone that are less than 9.55mm in diameter. functionality same as course aggr.

All variables are in the unit **kg/m^3**

Compressive_Strength Test pieces are used to measure the compressive strength of concrete. The test pieces generally consist of cylinders 16 cm in diameter and 32 cm in height that are manufactured in cardboard moulds. The test pieces are subjected to a crushing force (generally 28 days after manufacture) and the force required to break the specimen is measured. The ratio between the crushing force and the cross-sectional area of the test piece gives the compressive strength of the concrete. It is measured to check whether the strength of the concrete has reached the requirement of specified strength.

- Scatter plots of each covariate and the response is drawn to visualise the relationship.
- Then to show the relationship between the response and each covariate after elemination of the effect of other covariates, we used partial residual plot and added variable plot. They resembled to have linear relationship with the response variable.
- Q-Q plot is drawn and from this we can infer that the normality assumption is appropriate. Also Shapiro-Wilk normality test is used to confirm that normality holds.
- Then I have detected the outliers and the influential observations and excluded the observations from the data using cook's distance. Then I checked the appropriateness of the assumptions in the linear model. First, homoskedusticity of the residuals I have used Bruesh pagan test Then for normality of the residuals I have used kolmogorov Smirnoff test. Then for independence of the residuals I have used Durbin Watson test. Then for checking the multicolinearity in the data I have evaluated the Variance inflation factors. As there were multicolinearity in the data so I have used both lasso and ridge regression. Then i have assessed the risk for using both model. The lasso has lesser risk. Hence I took that model as my final model.

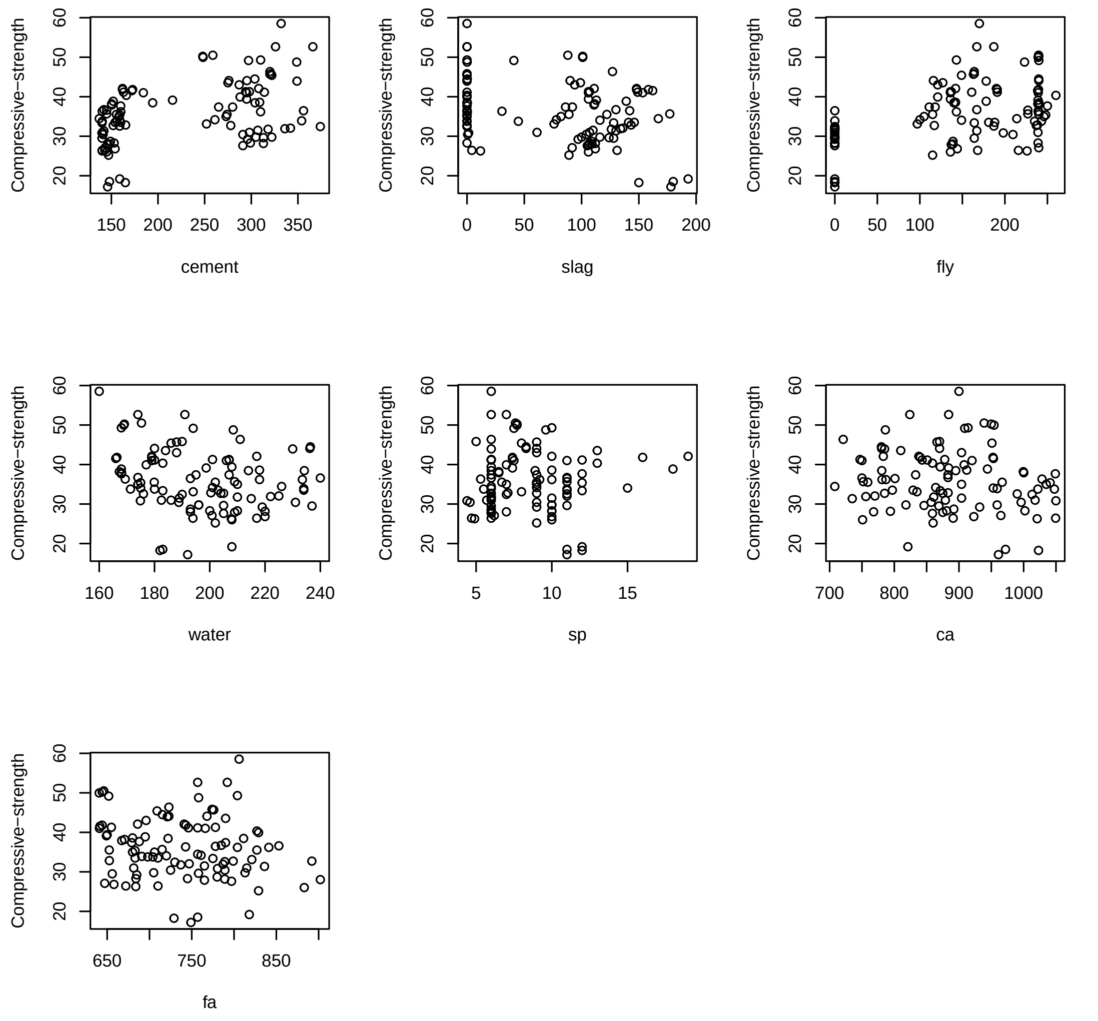
```
data=read.csv("D:\\DU\\new project arijit\\slump_test.csv",header=TRUE)
data=data[,-1]
colnames(data)=c("cement","slag","fly","water","sp","ca","fa","slump","flow","cs")
dt=data
head(dt)

  cement slag fly water sp  ca  fa slump flow    cs
1    273   82 105   210  9 904 680    23 62.0 34.99
2    163  149 191   180 12 843 746     0 20.0 41.14
3    162  148 191   179 16 840 743     1 20.0 41.81
4    162  148 190   179 19 838 741     3 21.5 42.08
5    154  112 144   220 10 923 658    20 64.0 26.82
6    147   89 115   202  9 860 829    23 55.0 25.21

pairs(dt[,-c(8,9)])
```



- We can not observe linear relation between these response and the predictors using scatter plot.

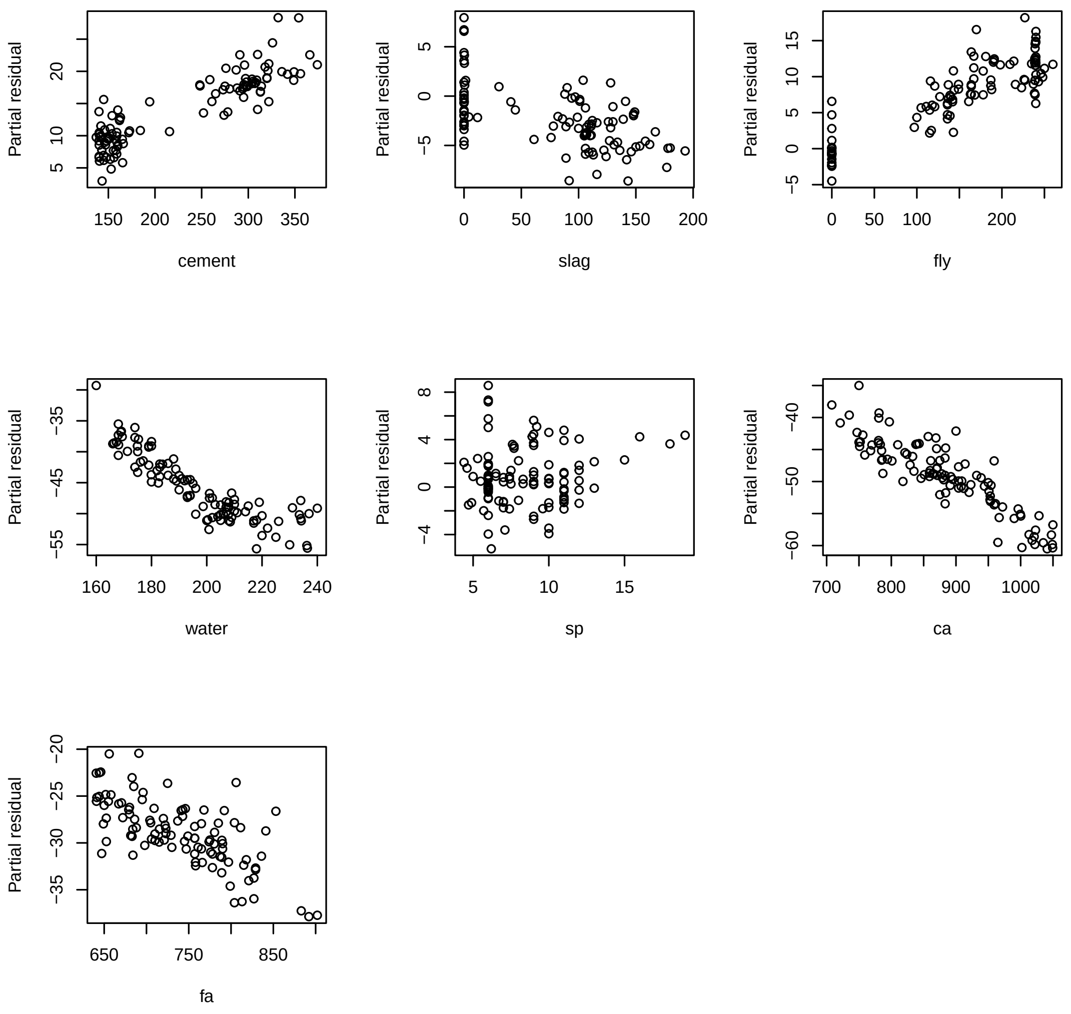
```
model1=lm(cs~.-(slump+flow),dt)
e=model1$residuals
b=coefficients(model1)
```

```r
##PHASE !
par(mfrow=c(3,3))
for(i in 1:7)
{
  plot(dt[,i],dt$cs,xlab=colnames(data)[i],ylab="Compressive-strength")
}
```

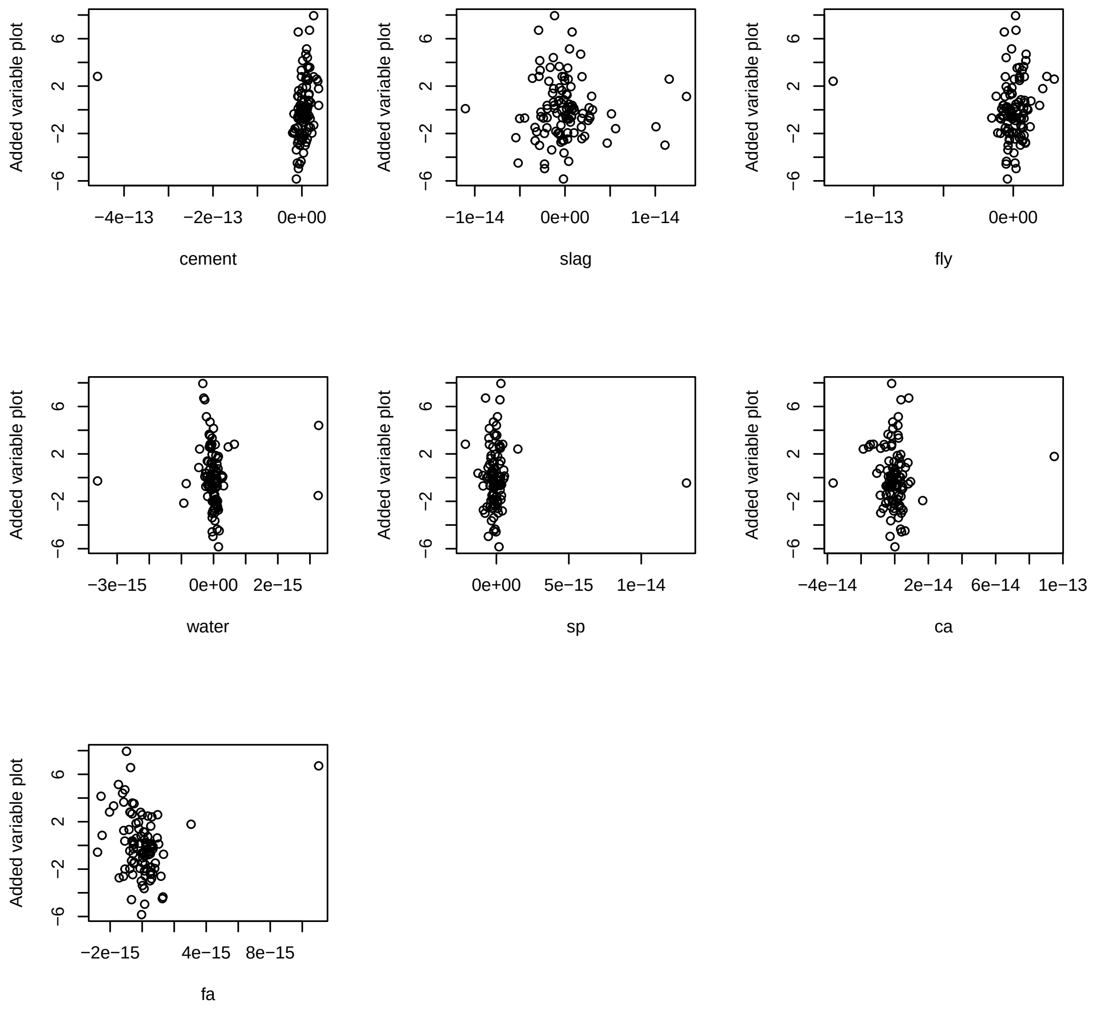


- There isn't any clear linear relationship between the predictor and the response variables.

```r
#---Partial residual plot--
par(mfrow=c(3,3))
for(i in 1:7)
{
  plot(dt[,i],b[i+1]*dt[,i]+e,xlab=colnames(data)[i],ylab="Partial residual")
}
#-Added variable plot--
par(mfrow=c(3,3))
```

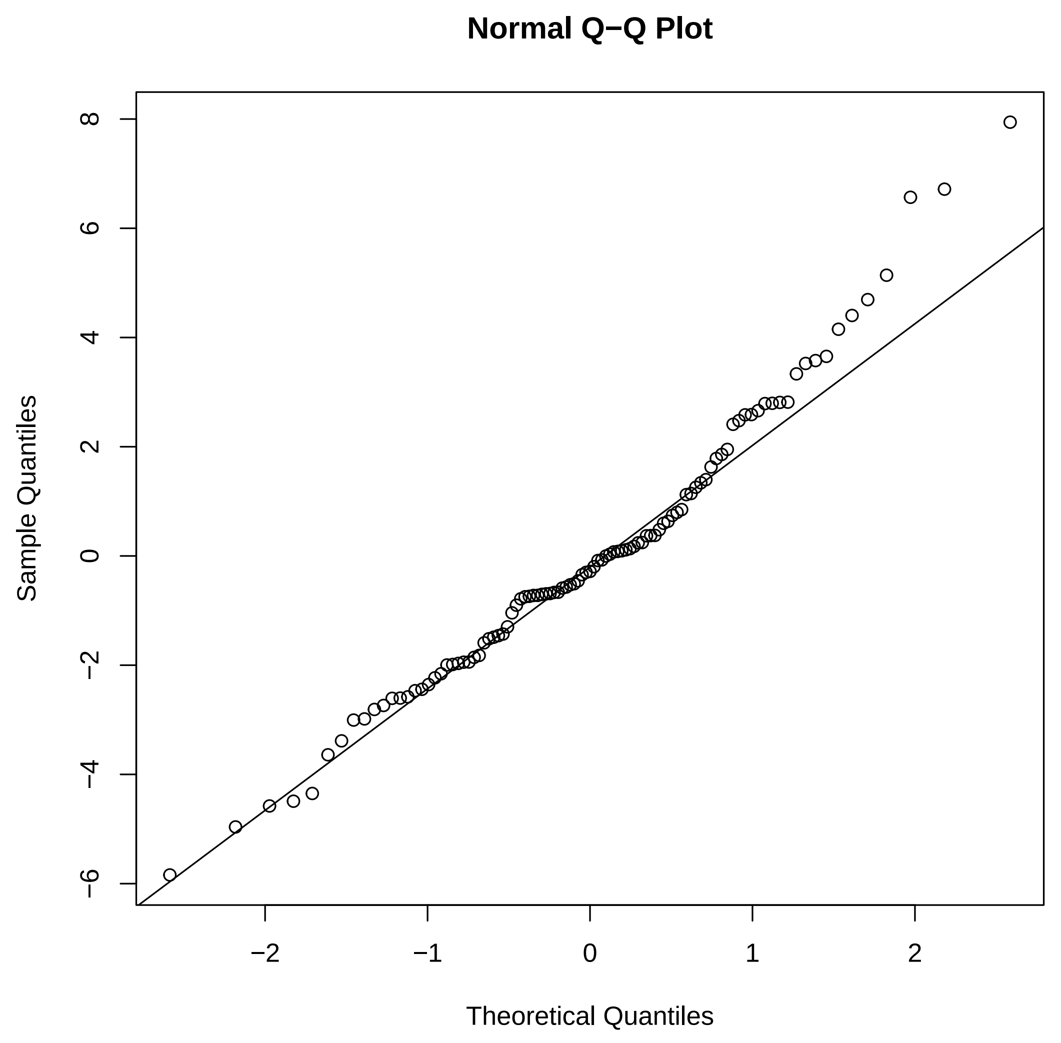


```
dt1=dt[-c(8,9)]
for(i in 1:7)
{
  plot(lm(dt1[,i]~.-(dt1$cs),data=dt1)$residuals,lm(cs~.-(dt1[,i]),data=dt1)$residuals,xlab=colnam
}
```



- Hence They resembled to have linear relationship with the response variable.

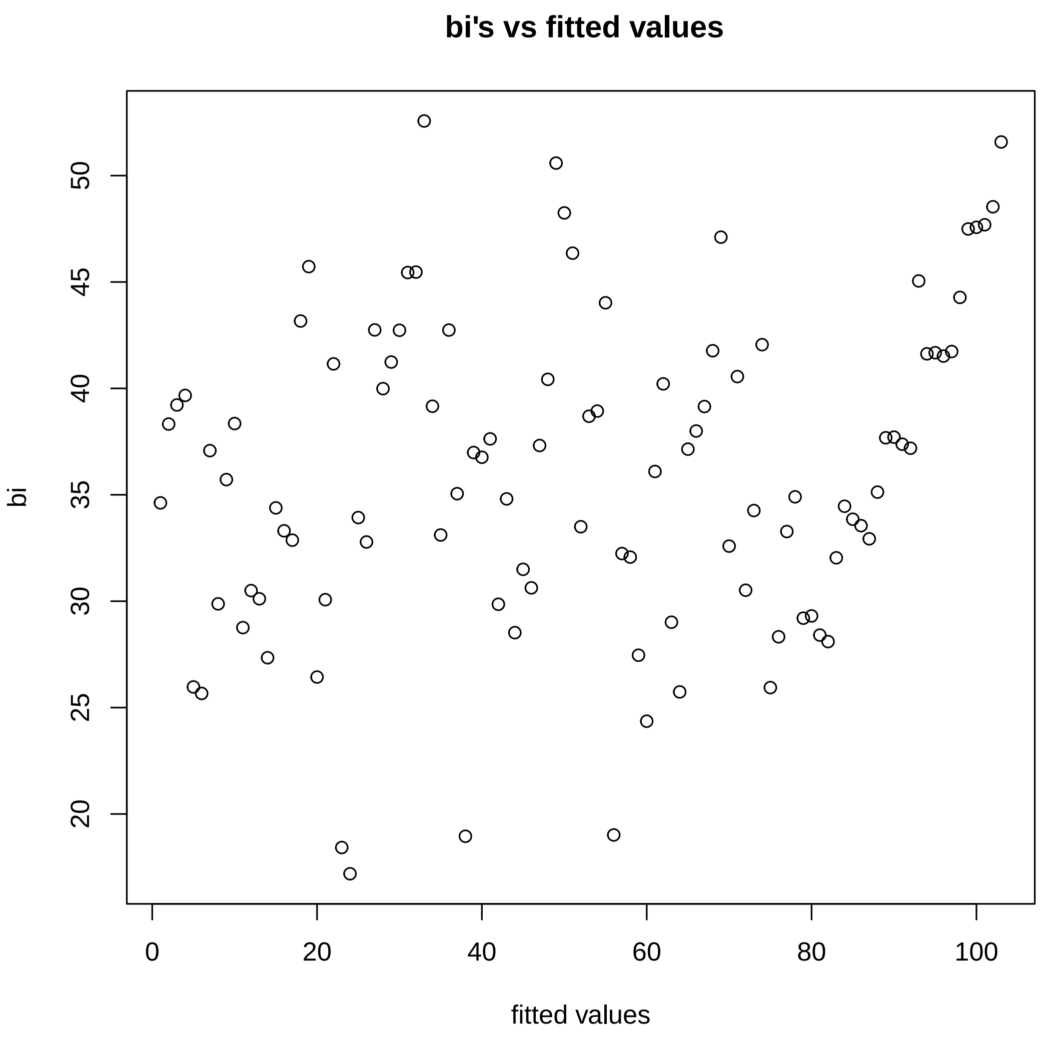
```
##Normality Test
par(mfrow=c(1,1))
qqnorm(e)
qqline(e)
```



```
library(dplyr)

Warning:  package 'dplyr' was built under R version 4.1.3

Attaching package:  'dplyr'
The following objects are masked from 'package:stats':
      filter, lag
The following objects are masked from 'package:base':
      intersect, setdiff, setequal, union

shapiro.test(e)


Shapiro-Wilk normality test

data:  e
W = 0.97486, p-value = 0.0467
```

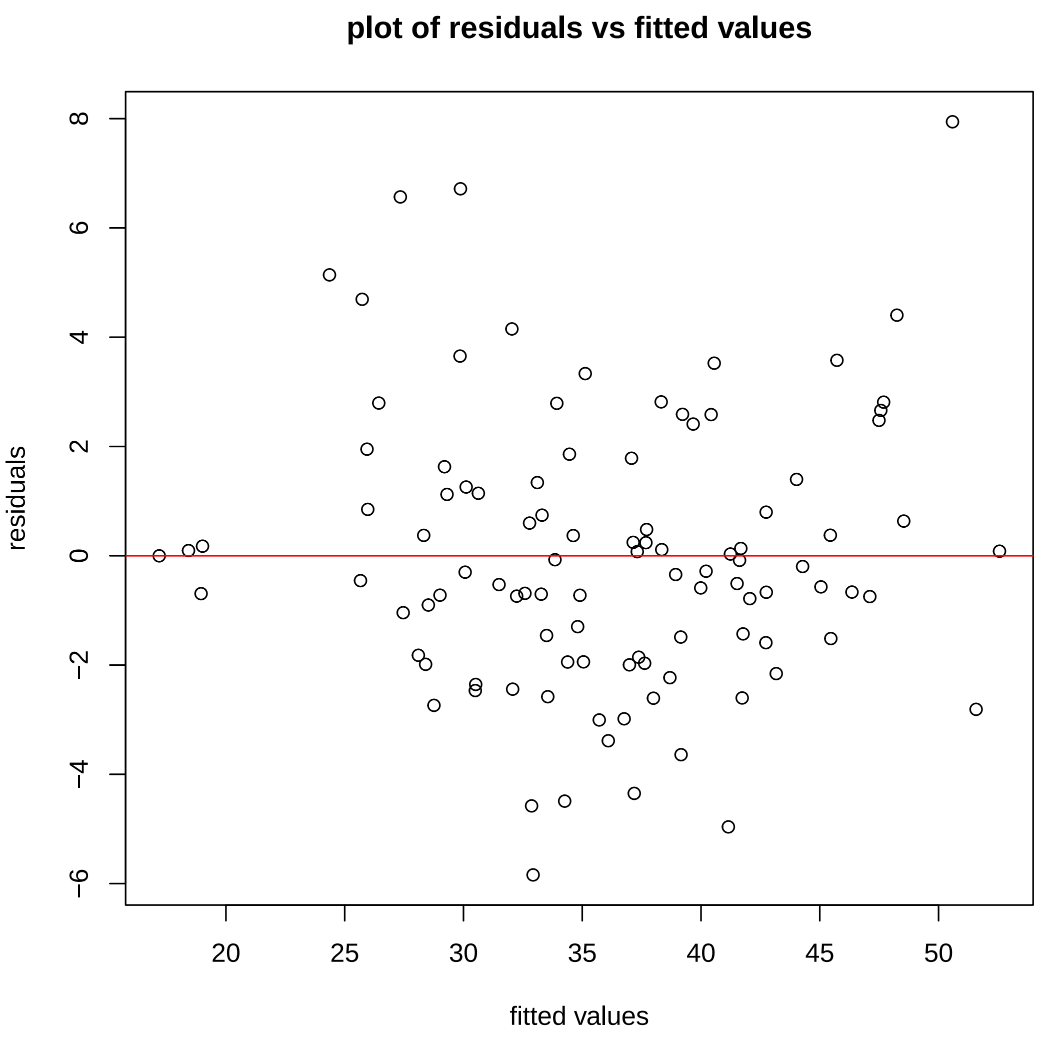
- From QQ plot normality assumption is not satisfied.but From the output of Shapiro-Wilk Test, we see that the p-value is 0.0467. The Null Hypothesis is accepted at level 0.01.Hence we can conclude that the Normality assumptions of the errors is valid.

```
##HETEROSCED-------- eta bad----------
X=as.matrix(dt[,-(8:10)])
n=nrow(X)
X=cbind(rep(1,n),X)
H=X%*%solve(t(X)%*%X)%*%t(X)
b=c()
for(i in 1:n)
{
  bi=(e[i]^2)/(1-H[i,i]) #
}
yh=model1$fitted.values
plot(yh,b,xlab="fitted values",ylab="bi",main=" bi's vs fitted values")
```



```
plot(yh,e,xlab="fitted values",ylab="residuals",main="plot of residuals vs fitted values")
abline(h=0,col="red")
#------------etota obdhi bad-------------
#bp test
library(lmtest)

Warning:  package 'lmtest' was built under R version 4.1.3
Loading required package:  zoo
Warning:  package 'zoo' was built under R version 4.1.3

Attaching package:  'zoo'
```

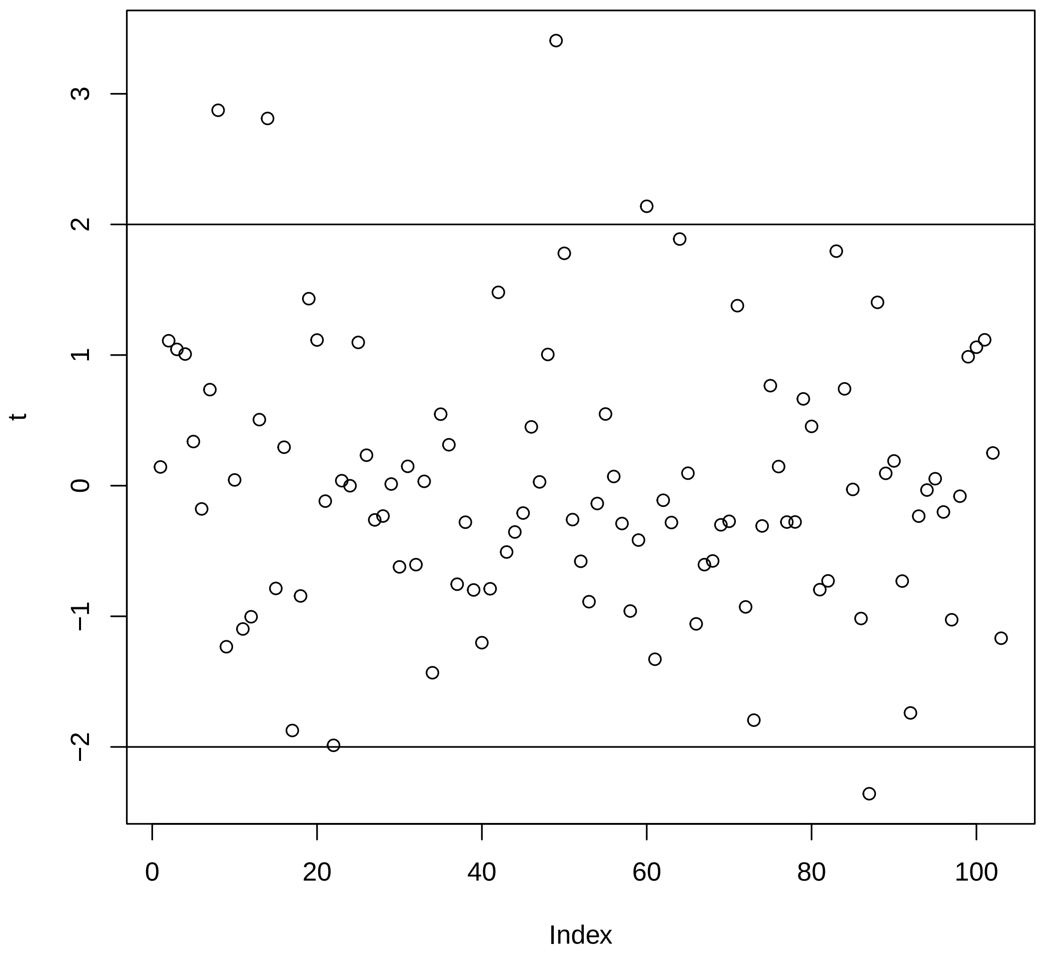
```
The following objects are masked from 'package:base':
      as.Date, as.Date.numeric
```



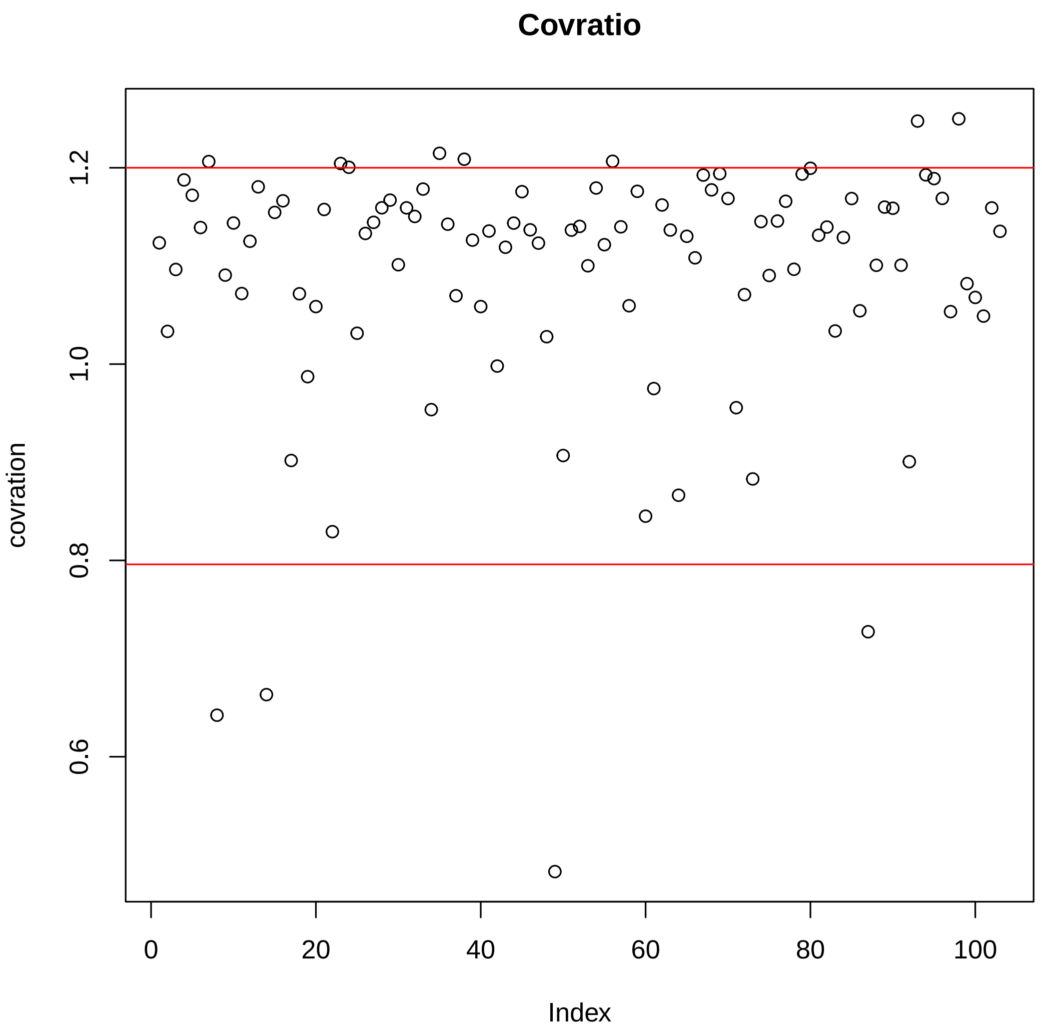
```
bptest(model1)


studentized Breusch-Pagan test

data:  model1
BP = 8.791, df = 7, p-value = 0.268

##godf q test
gqtest(model1)


Goldfeld-Quandt test

data:  model1
GQ = 0.7494, df1 = 44, df2 = 43, p-value = 0.8279
alternative hypothesis: variance increases from segment 1 to 2
```

- The p value of BP test is 0.6584. So we can conclude that the residuals are homoscedastic. Same conclusion from gq test.

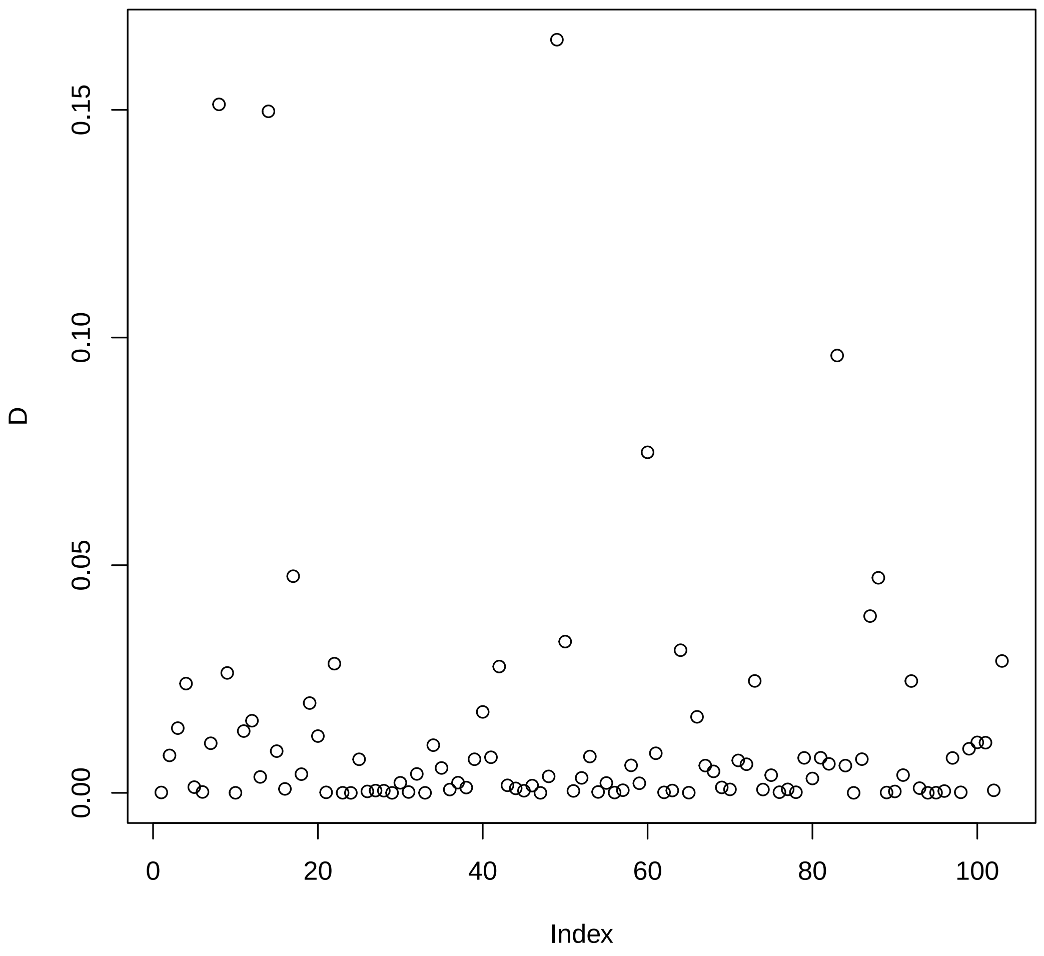
```
#--Outlier detection--
A=dt[,-c(8,9)]
r=c();t=c()
hat=H
p=ncol(X)
S=sqrt(sum(e^2)/(n-p))    #this is MSE
for(i in 1:n)
{
   r[i]=(e[i]/sqrt(1-hat[i,i]))/S  #this is studentised residual
   t[i]=r[i]*sqrt((n-p-1)/(n-p-r[i]^2)) # externally studentized residual
}
plot(t)
abline(h=-2)
abline(h=2)
```



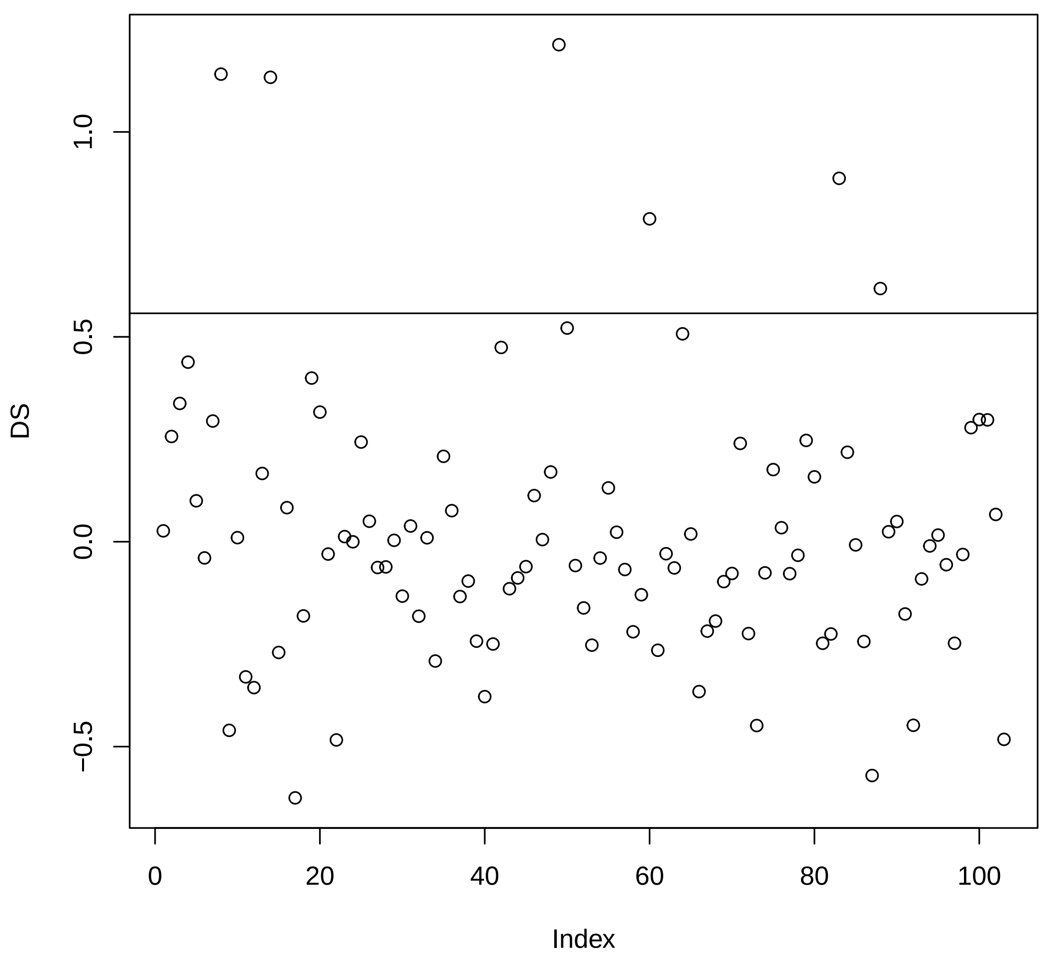
```
which(abs(t)>2)

[1]  8 14 49 60 87

#---covratio---
cr=c()
for(i in 1:n)
{
   cr[i]=1/(((((n-p-1)/(n-p))+((t[i]^2)/(n-p)))^p)*(1-hat[i,i]))
}
which(abs(cr-1)>3*p/n)

[1]  8 14 49 87 93 98

plot(cr,ylab="covration",main="Covratio")
abline(h=1.2,col="red")
abline(h=0.796,col="red")
```



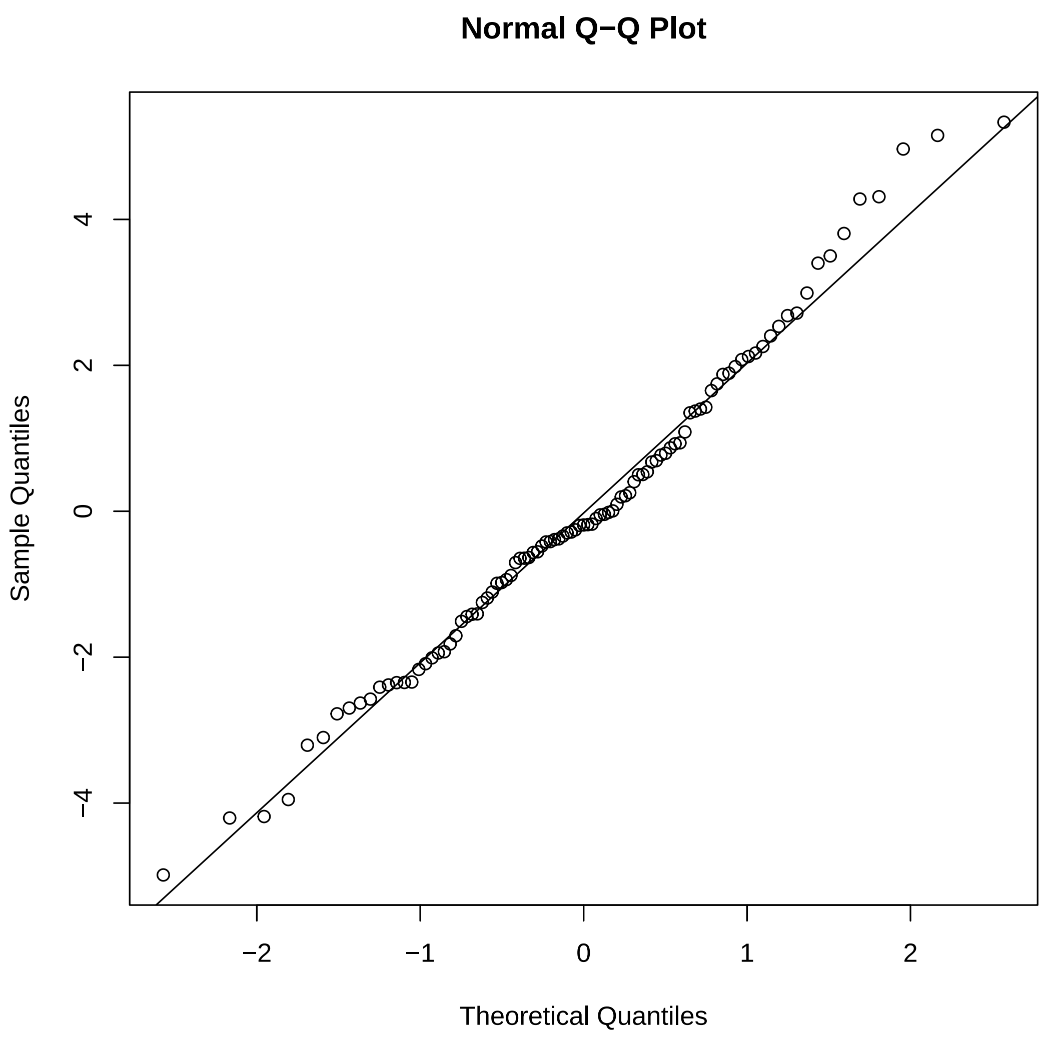
```
#--cooks Distance--
D=c()
for(i in 1:n)
{
   D[i]=((r[i]^2)*hat[i,i])/(p*(1-hat[i,i]))
}
f=qf(0.10,p,n-p)
which(D>1)

integer(0)

plot(D)
abline(h=f)
```



```
#--DFFITSi--
DS=c()
for(i in 1:n)
{
   DS[i]=t[i]*sqrt(hat[i,i]/(1-hat[i,i]))
}
plot(DS)
abline(h=2*sqrt(p/n))
```



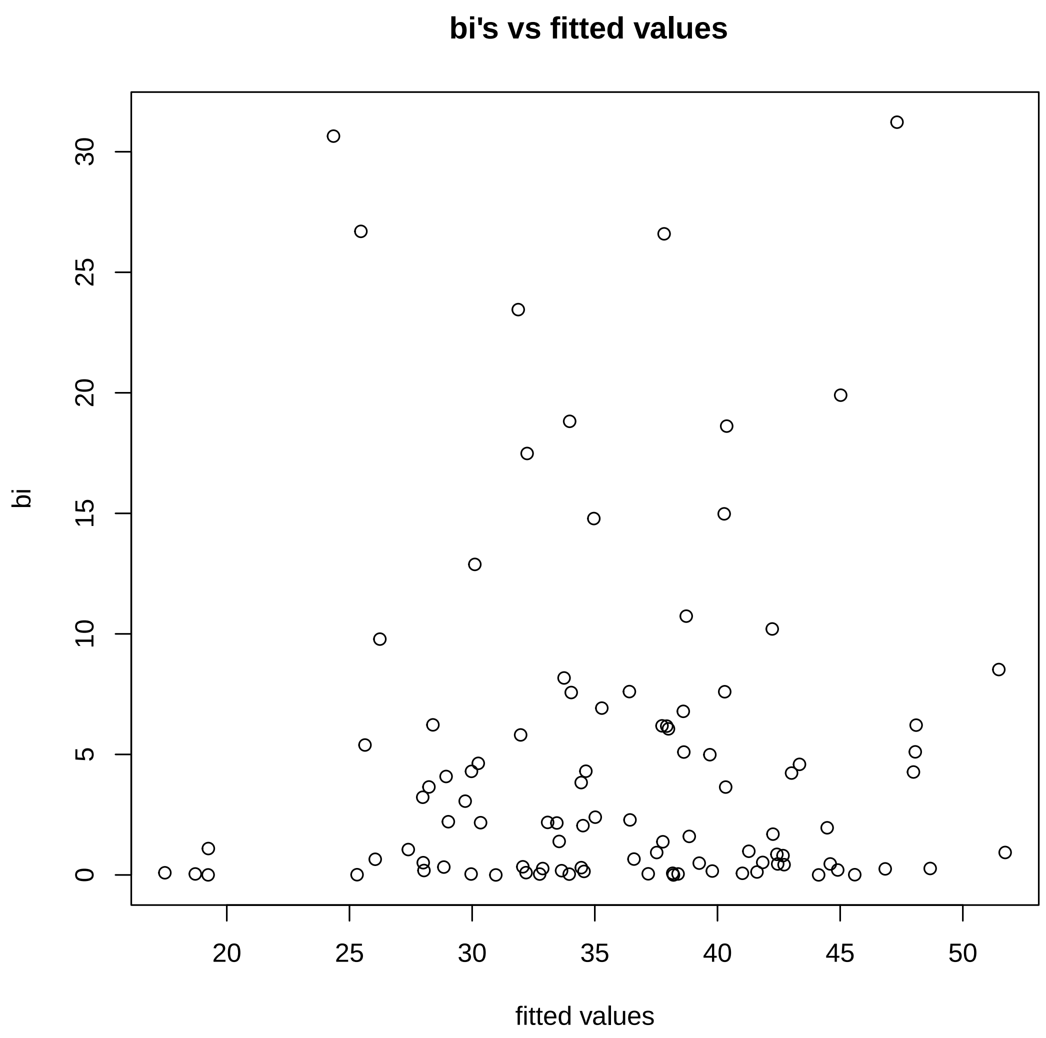
```
which(abs(DS)>2*sqrt(p/n))

[1]  8 14 17 49 60 83 87 88

##truncating the outliers
outliers=c(8,14,49,87)
dt1=dt[-outliers,]
model2=lm(cs~.-(slump+flow),dt1)
e=model2$residuals
##Normality Test
qqnorm(e)
qqline(e)
```



```
library(dplyr)
shapiro.test(e)


Shapiro-Wilk normality test

data:  e
W = 0.9863, p-value = 0.3995
```

- Hence normality assumption is now satisfied.

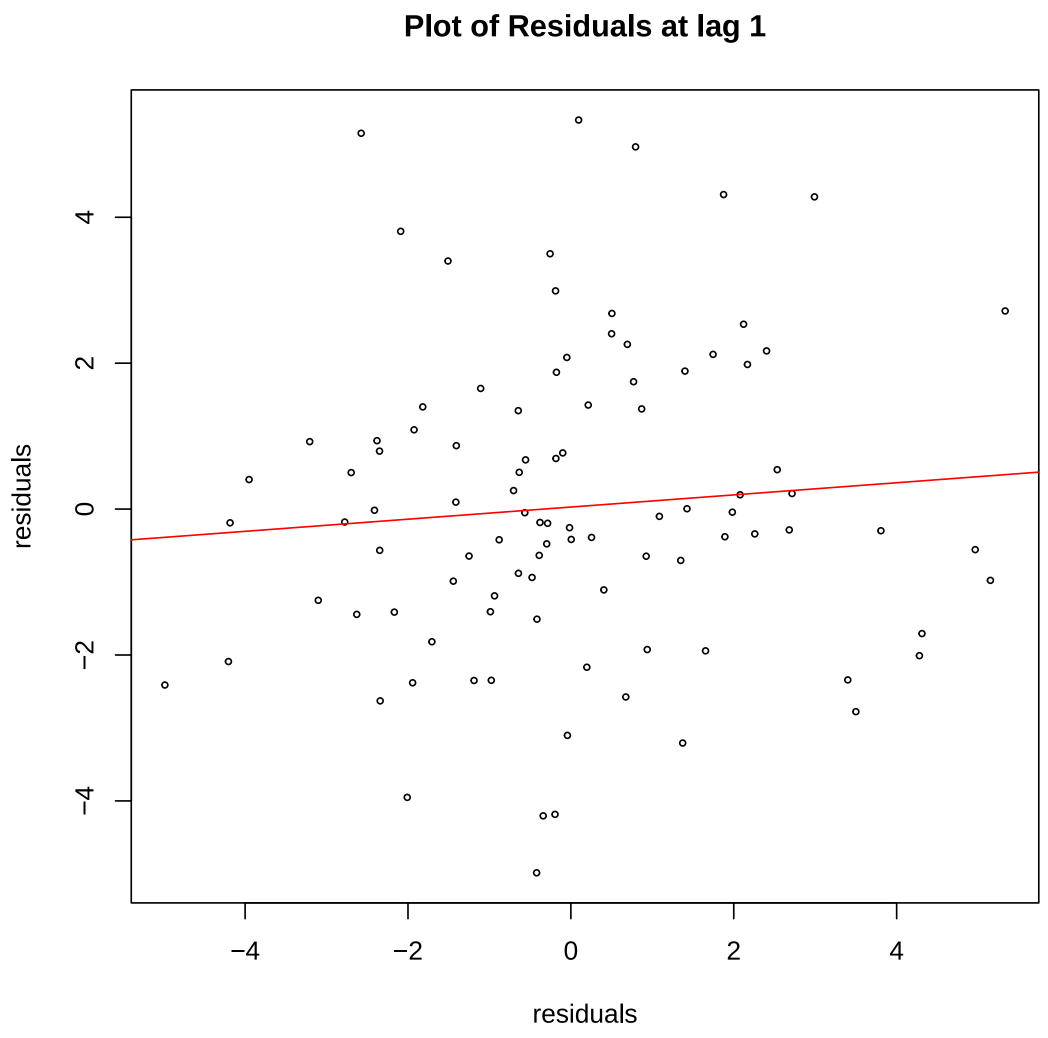
```
#HETEROSCED-----eta bad-----------
X1=as.matrix(dt1[,-(8:10)])
n=nrow(X1)
X1=cbind(rep(1,n),X1)
H1=X1%*%solve(t(X1)%*%X1)%*%t(X1)
b=c()
for(i in 1:n)
{
  b[i]=(e[i]^2)/(1-H1[i,i])
}
yh=model2$fitted.values
plot(yh,b,xlab="fitted values",ylab="bi",main=" bi’s vs fitted values")
```



```
#--------------etota bad-------------------

library(lmtest)
bptest(model2)


studentized Breusch-Pagan test

data:  model2
BP = 4.7668, df = 7, p-value = 0.6884

gqtest(model2)


Goldfeld-Quandt test

data:  model2
GQ = 1.3749, df1 = 42, df2 = 41, p-value = 0.155
alternative hypothesis: variance increases from segment 1 to 2
```

- Hence homoscedasticity assumption is satisfied.

```r
##AUTOCORR
e=model2$residuals
n=length(e)
plot(e[-1],e[-n],cex=0.5,xlab="residuals",ylab="residuals",main="Plot of Residuals at lag 1")
abline(lm(e[-n]~e[-1]),col="red")
```



```r
cor(e[-n],e[-1])

[1] 0.08391541

library(car)

Loading required package:  carData
Warning:  package 'carData' was built under R version 4.1.3

Attaching package:  'car'
The following object is masked from 'package:dplyr':
      recode

durbinWatsonTest(model2,max.lag=1)

 lag Autocorrelation D-W Statistic p-value
   1      0.08314867      1.816476   0.246
 Alternative hypothesis: rho != 0
```

- Comment: From the above plot, no significant correlation is visible among the residuals (considering lag1). Also the autocorrelation coefficient obtained from these residuals is 0.13 which is not very significant.
- in the test, the p-value is 0.246. Hence the null hypothesis gets accepted at level α= 0.05. We get no significant evidence in favour of the autocorrelation from the data.

```
#---Outlier detection-
r1=c()
for(i in 1:n)
{
  r1[i]=(e[i]/sqrt(1-H1[i,i]))/(sum(e^2)/(n-p))
}
D1=c()
for(i in 1:n)
{
   D1[i]=((r1[i]^2)*H1[i,i])/(p*(1-H[i,i]))
}
which(D1>f)

integer(0)
```

- No observation can be suspected to be outlier.

```
##MULTICOLLINEARITY
library(car)
vif(model2)

   cement      slag       fly     water        sp        ca        fa
46.329160 53.742391 56.771578 28.653220  2.123119 86.568206 47.066556

M=as.matrix(cor(X1[,-1]))
library(pracma)

Warning:  package 'pracma' was built under R version 4.1.3

Attaching package:  'pracma'
The following object is masked from 'package:car':
      logit

cond(M)

[1] 723.0177

#VIF of ca is highest  model_3=lm(cs~.-(slump+flow+ca),dt1)
vif(model_3)

Error in vif(model_3):  object 'model_3' not found

cond(as.matrix(cor(X1[,-c(1,7)])))

[1] 8.618534

# the condition became very small  pairs(X1[,-1],pch=20)
round(cor(X1[,-1]),2)

       cement  slag   fly water    sp    ca    fa
cement   1.00 -0.23 -0.47  0.26 -0.10 -0.35  0.06
slag    -0.23  1.00 -0.35  0.00  0.28 -0.24 -0.16
fly     -0.47 -0.35  1.00 -0.25 -0.15  0.20 -0.32
water    0.26  0.00 -0.25  1.00 -0.15 -0.63  0.12
sp      -0.10  0.28 -0.15 -0.15  1.00 -0.10  0.07
ca      -0.35 -0.24  0.20 -0.63 -0.10  1.00 -0.47
fa       0.06 -0.16 -0.32  0.12  0.07 -0.47  1.00
```

- from this matrix , the off-diagonal element are not very high , hence we can say the pairwise correlations are not very significant.
- Observation : We observe that in the data VIF for Coarse Aggr. is extremely large. We can recalculate the measures of multicollinearity by deleting this covariate. The obtained values of VIF's and Condition Number are as follows
- Comment : We observe that after eliminating the covariate Coarse Aggr. all the VIF's and Condition Number decrease considerably. Since this is not enough evidence to delete the covariate from our model , we would try to find a suitable linear model for our data by some statistically efficient methods such as Stepwise Regression , and Lasso Regression . We would compare the methods to select a single suitable model.

```
### stepwise
intercept_only=lm(cs~1,data=dt1)
all_v=lm(cs~.-(slump+flow),data=dt1)
stepwise=step(intercept_only,direction="both",scope=formula(all_v))

Start:  AIC=402.81
cs ~ 1

         Df Sum of Sq    RSS    AIC
+ fly     1   1300.18 4374.3 379.05
+ cement  1   1081.05 4593.4 383.89
+ slag    1    573.52 5100.9 394.26
+ fa      1    279.24 5395.2 399.82
+ water   1    274.65 5399.8 399.90
+ ca      1    151.13 5523.3 402.14
<none>                5674.5 402.81
+ sp      1      3.43 5671.0 404.75

Step:  AIC=379.05
cs ~ fly

         Df Sum of Sq    RSS    AIC
+ cement  1    3172.4 1201.8 253.15
+ ca      1     395.5 3978.8 371.67
+ slag    1     146.4 4227.9 377.68
<none>                4374.3 379.05
+ water   1      62.8 4311.4 379.62
+ fa      1      30.5 4343.8 380.36
+ sp      1      11.9 4362.4 380.78
- fly     1    1300.2 5674.5 402.81

Step:  AIC=253.15
cs ~ fly + cement

         Df Sum of Sq    RSS    AIC
+ water   1     321.7  880.1 224.31
+ slag    1     283.7  918.2 228.50
+ sp      1     220.6  981.2 235.07
<none>                1201.8 253.15
+ ca      1      11.5 1190.3 254.20
+ fa      1       0.1 1201.8 255.15
- cement  1    3172.4 4374.3 379.05
- fly     1    3391.6 4593.4 383.89

Step:  AIC=224.31
cs ~ fly + cement + water
```

```
         Df Sum of Sq    RSS    AIC
+ ca      1     309.8  570.3 183.35
+ slag    1     275.3  604.8 189.16
+ sp      1     147.3  732.8 208.17
<none>                 880.1 224.31
+ fa      1       2.3  877.8 226.05
- water   1     321.7 1201.8 253.15
- fly     1    3028.8 3908.9 369.91
- cement  1    3431.3 4311.4 379.62

Step:  AIC=183.35
cs ~ fly + cement + water + ca

         Df Sum of Sq    RSS    AIC
+ fa      1    104.05  466.2 165.41
+ slag    1     80.96  489.3 170.19
+ sp      1     50.63  519.7 176.15
<none>                 570.3 183.35
- ca      1    309.82  880.1 224.31
- water   1    620.05 1190.3 254.20
- cement  1   2759.08 3329.4 356.03
- fly     1   2916.36 3486.6 360.60

Step:  AIC=165.41
cs ~ fly + cement + water + ca + fa

         Df Sum of Sq     RSS    AIC
+ sp      1     26.81  439.42 161.54
+ slag    1     12.51  453.72 164.71
<none>                 466.23 165.41
- fa      1    104.05  570.28 183.35
- ca      1    411.61  877.84 226.05
- water   1    721.54 1187.77 255.99
- fly     1   2052.36 2518.59 330.40
- cement  1   2244.27 2710.50 337.67

Step:  AIC=161.54
cs ~ fly + cement + water + ca + fa + sp

         Df Sum of Sq     RSS    AIC
<none>                 439.42 161.54
+ slag    1      0.08  439.33 163.52
- sp      1     26.81  466.23 165.41
- fa      1     80.24  519.66 176.15
- ca      1    291.57  730.99 209.93
- water   1    540.03  979.45 238.89
- fly     1   1982.01 2421.43 328.50
- cement  1   2190.96 2630.38 336.70

##put trace=0 to skip step details
summary(stepwise)


Call:
lm(formula = cs ~ fly + cement + water + ca + fa + sp, data = dt1)

Residuals:
    Min      1Q  Median      3Q     Max 
-5.0170 -1.3946 -0.2051  1.3686  5.3623 
```

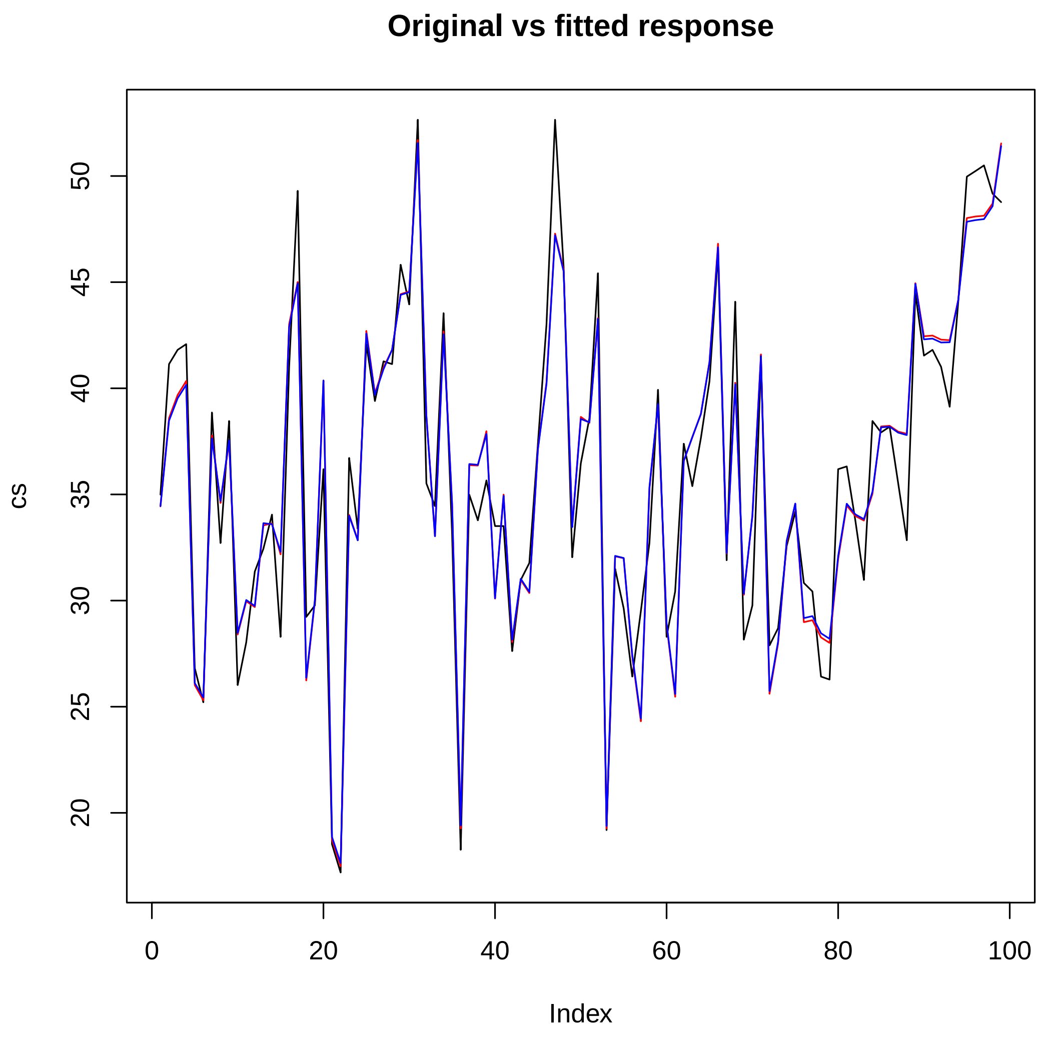
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 84.683607   9.792252   8.648 1.59e-13 ***
fly          0.070088   0.003441  20.371  < 2e-16 ***
cement       0.078586   0.003669  21.418  < 2e-16 ***
water       -0.178503   0.016787 -10.633  < 2e-16 ***
ca          -0.033718   0.004315  -7.813 8.79e-12 ***
fa          -0.019066   0.004652  -4.099 8.94e-05 ***
sp           0.207774   0.087694   2.369   0.0199 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.185 on 92 degrees of freedom
Multiple R-squared:  0.9226,Adjusted R-squared:  0.9175
F-statistic: 182.7 on 6 and 92 DF,  p-value: < 2.2e-16

y_pred=stepwise$fitted.values
mad_step=mean(abs(y_pred-dt1$cs))
```

- COMMENT : By stepwise regression, we obtain the fitted regression line as — Yi = 81.1 + 0.07xi3 + 0.08xi1 − 0.18xi4 − 0.03xi6 − 0.02xi7 + 0.25xi5, i = 1(1)97 R 2 = 0.9262 and Cross validation error of the model is 7.52 and the mean absolute deviation is 1.71

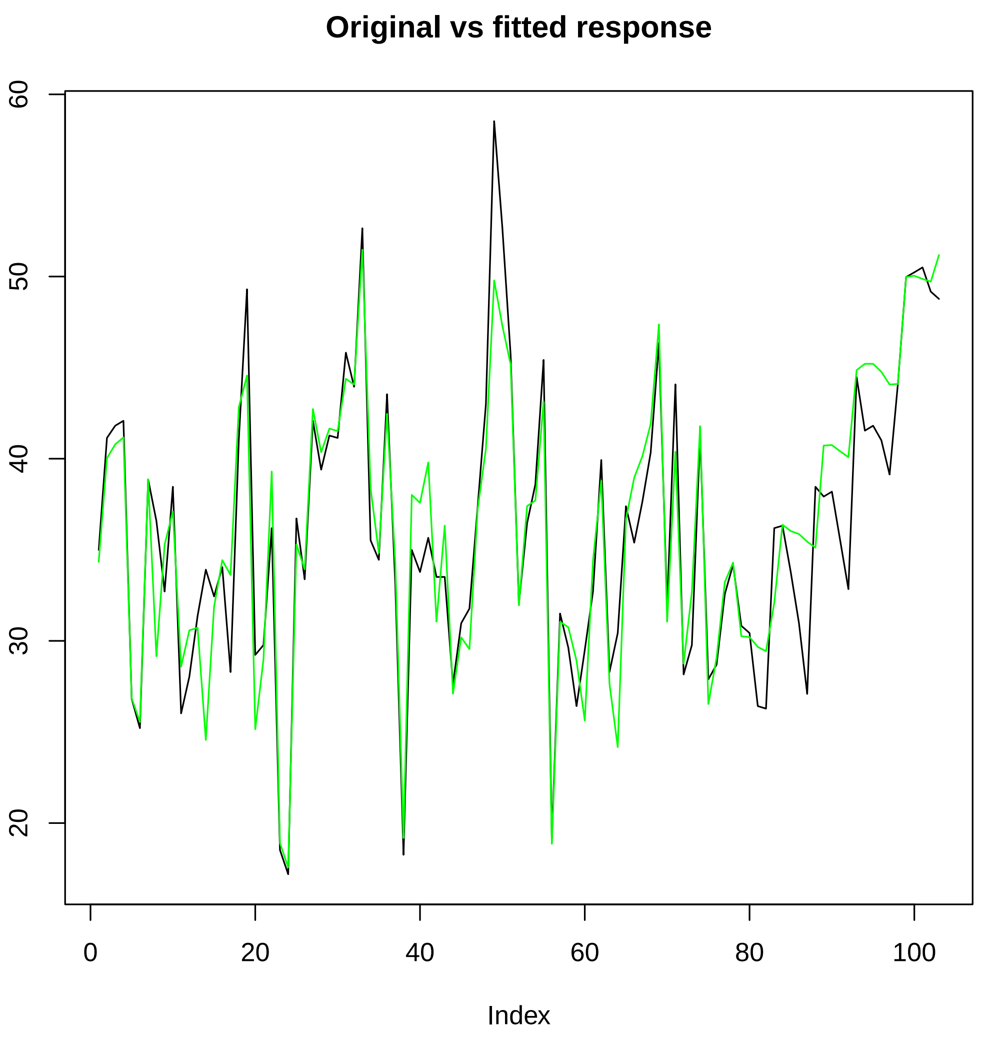
```
#lasso regression
library(caret)

Warning:  package 'caret' was built under R version 4.1.3
Loading required package:  ggplot2
Warning:  package 'ggplot2' was built under R version 4.1.3
Loading required package:  lattice

X=dt1[,-(8:10)]
scaled=preProcess(X,method = "center")
X=predict(scaled,X)
X=as.matrix(X)
dt1=as.data.frame(dt1)
library(pracma)
library(glmnet)

Warning:  package 'glmnet' was built under R version 4.1.3
Loading required package:  Matrix
Warning:  package 'Matrix' was built under R version 4.1.3

Attaching package:  'Matrix'
The following objects are masked from 'package:pracma':
      expm, lu, tril, triu
Loaded glmnet 4.1-4

cs=dt1$cs
u1=cv.glmnet(X,cs,alpha = 1, family = 'gaussian')
k1=0.02912
lasso_reg=glmnet(X,cs,alpha = 1, family = 'gaussian', lambda = k1)
y_lasso=lasso_reg$a0+X%*%lasso_reg$beta
mad_lasso=mean(abs(y_lasso-dt1$cs))
m=0
for(i in 1:n)
{
        lasso_reg1=glmnet(X[-i,],cs[-i],alpha = 1, family = 'gaussian', lambda = k1)
        m=m+(cs[i]-lasso_reg1$a0-t(X[i,])%*%lasso_reg1$beta)^2
}
cv_lasso=m/n
library(boot)
```

```

Attaching package:  'boot'
The following object is masked from 'package:lattice':
      melanoma
The following object is masked from 'package:pracma':
      logit
The following object is masked from 'package:car':
      logit
```

```
model_step1=glm(cs~.-(slag+slump+flow),data=dt)
cv=cv.glm(dt,model_step1)$delta
cv_step=cv[1]
#cross validation error of model obtained by stepwise model selection
#comparison b/w the models
plot(cs,type="l",main="Original vs fitted response")
lines(y_pred,col="red")
lines(y_lasso,col="blue")
```



```
#--Least median square--
library(MASS)


Attaching package:  'MASS'
The following object is masked from 'package:dplyr':
      select

dt=read.csv("D:\\DU\\new project arijit\\slump_test.csv",header=TRUE)
dt=dt[,-1]
colnames(dt)=c("cement","slag","fly","water","sp","ca","fa","slump","flow","cs")
dt=as.matrix(dt[,-c(8,9)])
model_lms=lqs(dt[,8]~dt[,1]+dt[,2]+dt[,3]+dt[,4]+dt[,5]+dt[,6]+dt[,7],method = "lms")
y_lms=model_lms$fitted.values
plot(dt[,8],type="l",ylab="",main="Original vs fitted response")
lines(y_lms,col="green")
```



```
m1=0;coe=c()
for(i in 1:n)
{
  model_lms1=lqs(dt[-i,8]~dt[-i,1]+dt[-i,2]+dt[-i,3]+dt[-i,4]+
                              dt[-i,5]+dt[-i,6]+dt[-i,7],method = "lms")
  coe=model_lms1$coefficients
  m1=m1+(cs[i]-coe[1]-t(X[i,])%*%lasso_reg$beta)^2
}
cv_lms=m1/103
```

- The fitted regression is given by, Yi = 78.79 + 0.08xi1 + 0.07xi3 − 0.178xi4 + 0.24xi5 − 0.032xi6 − 0.016xi7, i = 1(1)97 The cross validation error of the model is 4.726 and the mean absolute deviation is 1.56