*Code with output*

1. Linear, Multiple and Quadratic Regression

**Code**

#install ggiraphExtra package

library(ggplot2)

library(plyr)

library(ggiraphExtra)

report\_r=read.csv("D://sem3 materials//labs//R programming//lab\_project//water\_data.csv")

report\_r=report\_r[,c(4:11)]

report\_r=na.omit(report\_r)

#DO

lm1<-lm(DO~Temp+BOD,data=report\_r)

summary(lm1)

a=data.frame(Temp=29.2,BOD=1.5)

result=predict(lm1,a)

result[1]

ggPredict(lm1,se=TRUE,interactive=TRUE)

lm\_do=lm(DO~.,data=report\_r)

summary(lm\_do)

a=data.frame(Temp=29.2,PH=6.3,CONDUCTIVITY=100,BOD=1.5,NITRATE=0.1,F\_COLIFORM=7942,T\_COLIFORM=13575)

result=predict(lm\_do,a)

result[1]

#CONDUCTIVITY

#linear

lm2<-lm(report\_r$CONDUCTIVITY~report\_r$PH)

summary(lm2)

a=data.frame(PH=7.5)

result=predict(lm2,a)

result[1]

ggplot(report\_r,aes(y=CONDUCTIVITY,x=PH))+geom\_point()+geom\_smooth(method="lm")

#Polynomial(Quadratic) regression

x1<-(report\_r$PH)^2

qm<-lm(report\_r$CONDUCTIVITY~x1+report\_r$PH)

summary(qm)

a=data.frame(PH=7.5)

result=predict(qm,a)

result[1]

ggplot(report\_r,aes(y=CONDUCTIVITY,x=PH^2+PH))+geom\_point()+geom\_smooth(method="lm")

#NITRATE

#multiple regression

lm3<-lm(report\_r$NITRATE~report\_r$F\_COLIFORM+report\_r$T\_COLIFORM)

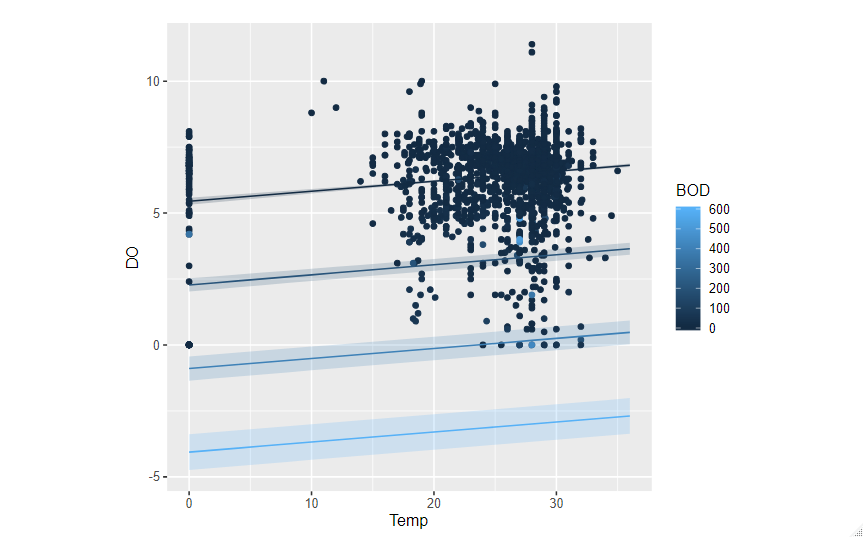
summary(lm3)

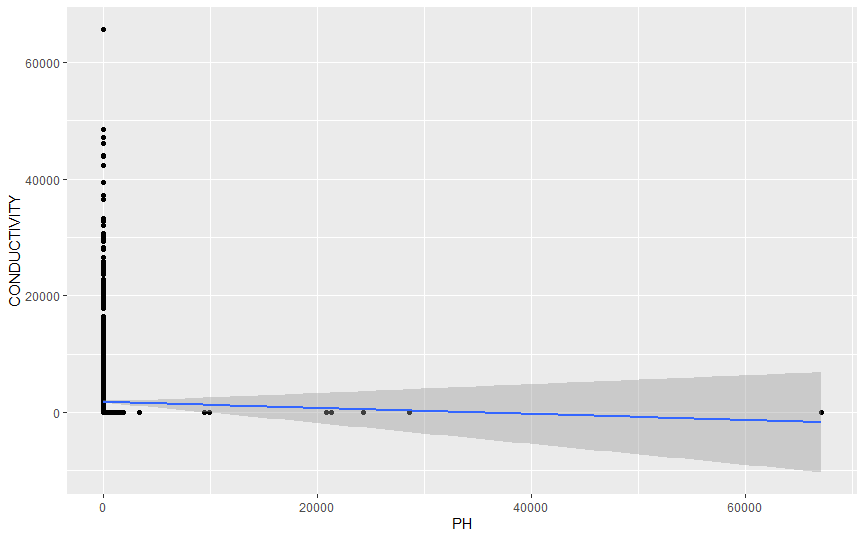
a=data.frame(F\_COLIFORM=11,T\_COLIFORM=27)

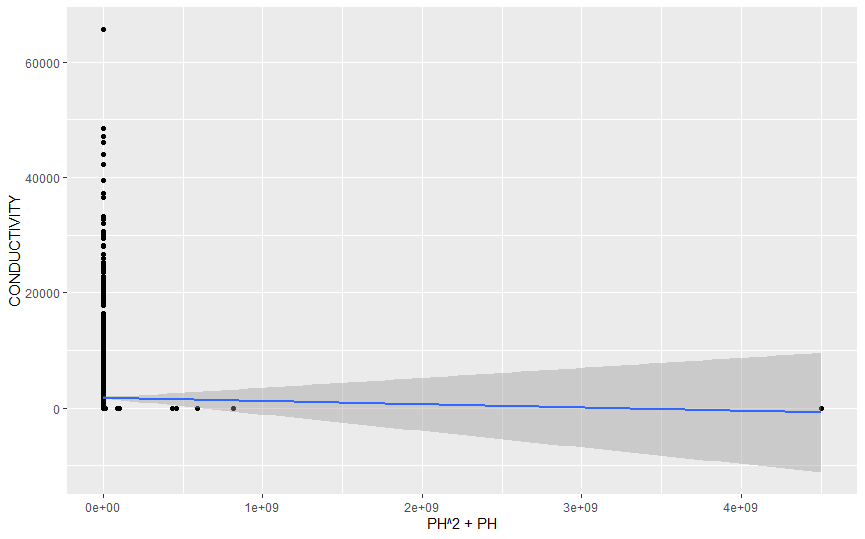
result=predict(lm3,a)

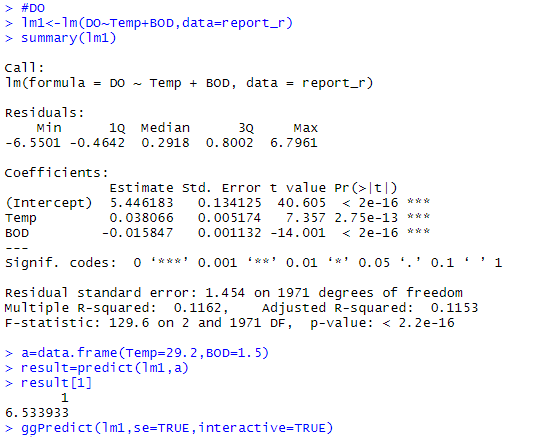
result[1]

**Output**

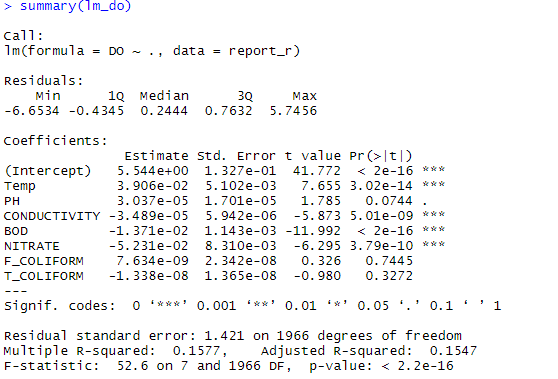


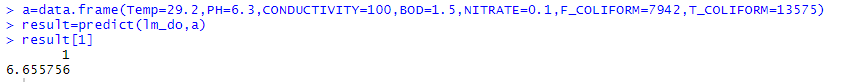


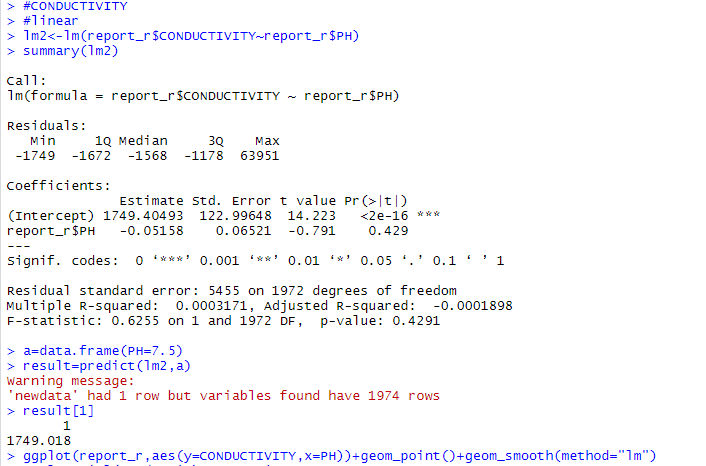


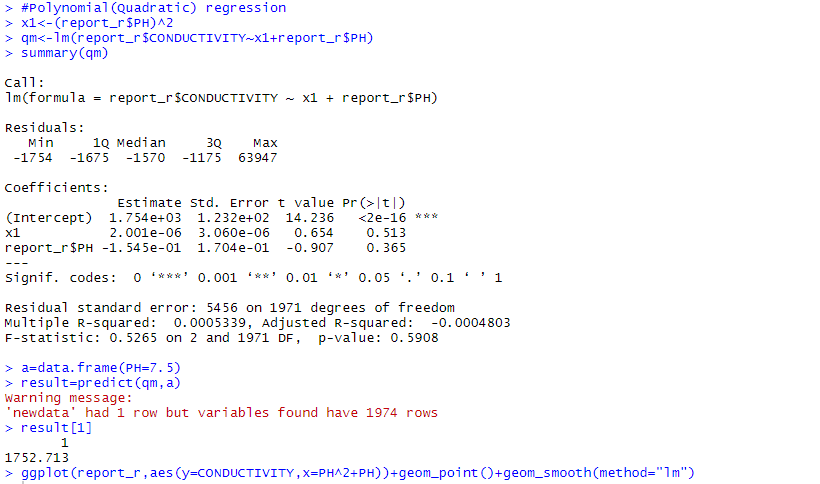


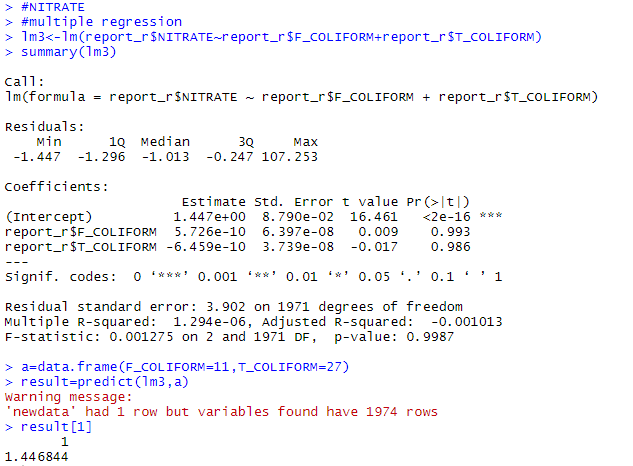
#calculating D.O.  considering all factors











1. Stepwise Regression(Also, forward selection and backward elimination)

**Code**

report\_r=read.csv("D://sem3 materials//labs//R programming//lab\_project//water\_data.csv")

report\_r=report\_r[,c(4:11)]

str(report\_r)

report\_r=na.omit(report\_r)#to remove null values nan

#both direction

Report\_model=step(lm(DO~.,data=report\_r),direction="both")

Report\_model

a=data.frame(Temp=29.2,PH=6.3,CONDUCTIVITY=100,BOD=1.5,NITRATE=0.1,F\_COLIFORM=7942,T\_COLIFORM=13575)

result=predict(Report\_model,a)

result

#forward direction

Report\_model\_for=step(lm(DO~1,data=report\_r),direction="forward",scope=~ Temp+PH+CONDUCTIVITY+BOD+NITRATE+F\_COLIFORM+T\_COLIFORM)

Report\_model\_for

a\_for=data.frame(Temp=29.2,PH=6.3,CONDUCTIVITY=100,BOD=1.5,NITRATE=0.1,F\_COLIFORM=7942,T\_COLIFORM=13575)

result\_for=predict(Report\_model\_for,a)

result

#backward direction

Report\_model\_back=step(lm(DO~.,data=report\_r),direction="backward")

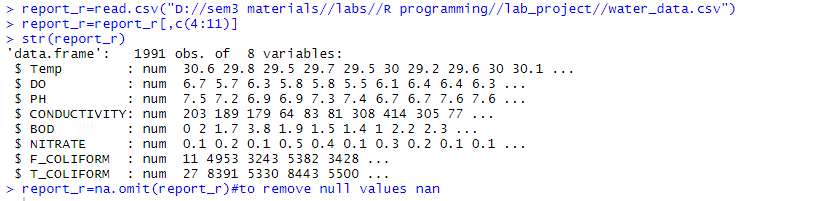
Report\_model\_back

a\_back=data.frame(Temp=29.2,PH=6.3,CONDUCTIVITY=100,BOD=1.5,NITRATE=0.1,F\_COLIFORM=7942,T\_COLIFORM=13575)

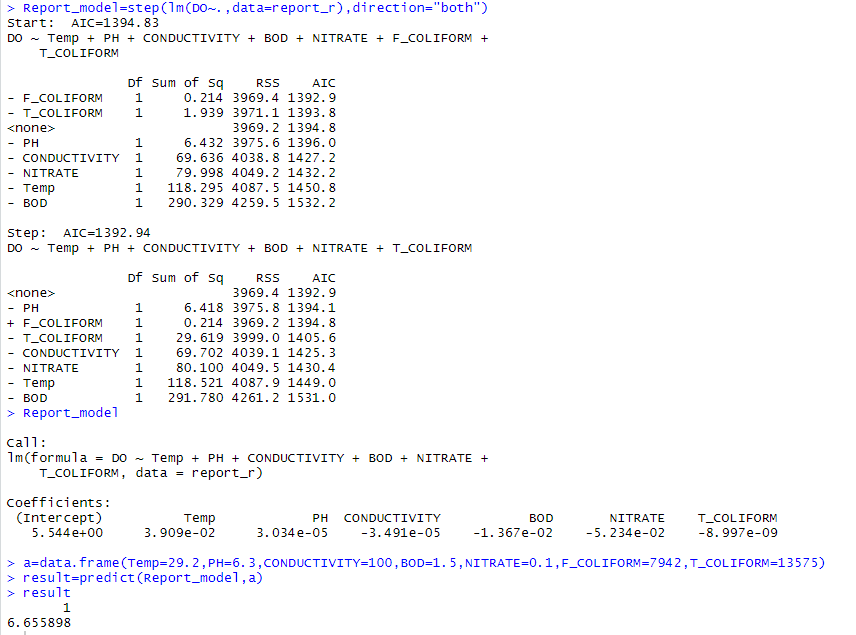
result\_back=predict(Report\_model\_back,a)

result\_back

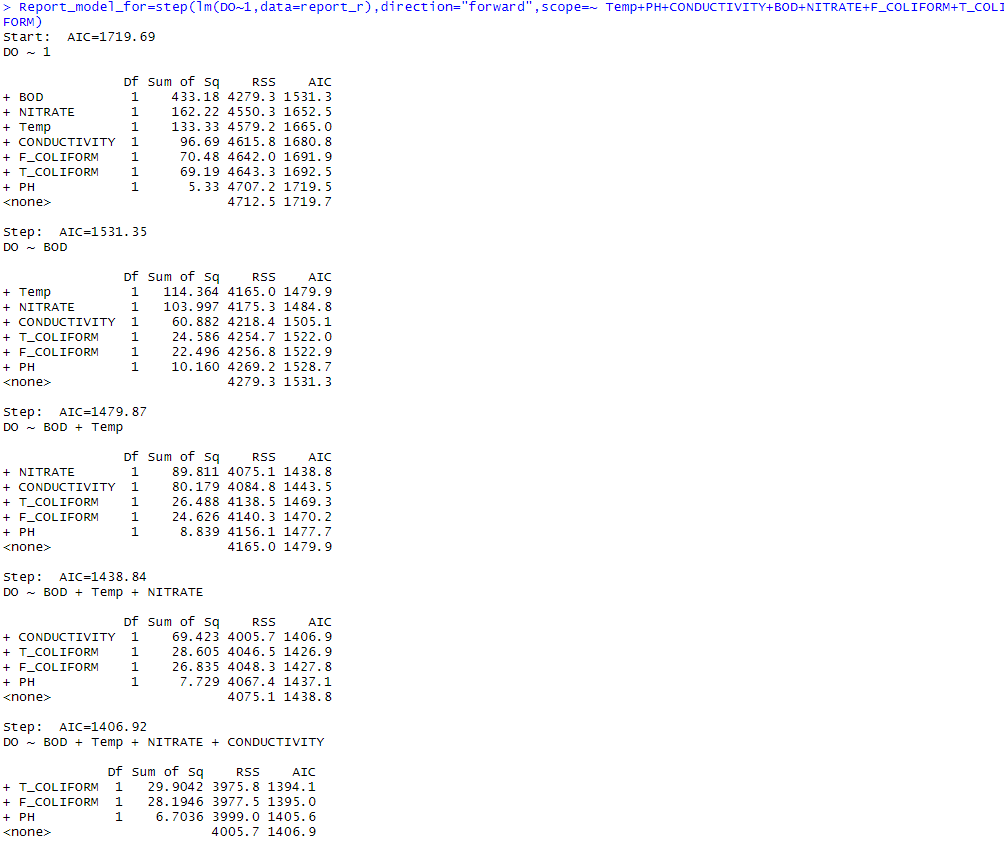
**Output**

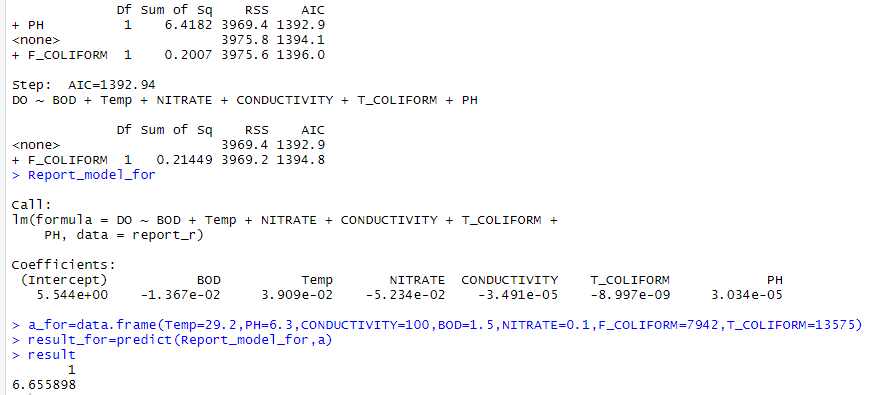


Stepwise regression

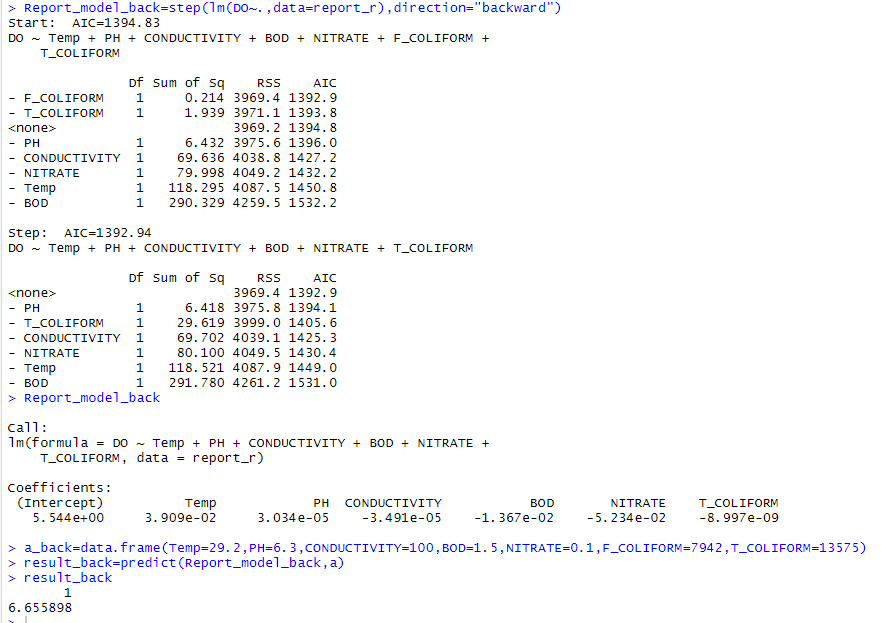


Forward selection





Backward elimination



1. Ridge, Lasso, and Elastic Net Regression

**Code**

library(glmnet)

library(tidyverse)

library(caret)

report\_r=read.csv("D://sem3 materials//labs//R programming//lab\_project//water\_data.csv")

report\_r=report\_r[,c(4:11)]

report\_r=na.omit(report\_r)

#partition

set.seed(103)#The seed number (n) you choose is the starting point used in the generation of a sequence of random numbers

ind=sample(2,nrow(report\_r),replace=T,prob=c(0.8,0.2))

ind

train=report\_r[ind==1,]

test=report\_r[ind==2,]

str(test)

#DO with BOD and temp

x=model.matrix(DO~BOD+Temp, train)[,-1]

y=train$DO

cv = cv.glmnet(x, y, alpha = 0)

cv$lambda.min

Ridge\_model = glmnet(x, y, alpha = 0, lambda = cv$lambda.min)

Ridge\_model

summary(Ridge\_model)

coef(Ridge\_model)

#predictions

x\_test = model.matrix(DO~BOD+Temp, test)[,-1]

result\_rid = Ridge\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

5.65026689+-0.01540126\*1.5+0.03136930\*29.2

cv = cv.glmnet(x, y, alpha = 1)

cv$lambda.min

Lasso\_model = glmnet(x, y, alpha = 1, lambda = cv$lambda.min)

Lasso\_model

summary(Lasso\_model)

coef(Lasso\_model)

#predictions

x\_test = model.matrix(DO~BOD+Temp, test)[,-1]

result\_las = Lasso\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

5.61717393+-0.01712410\*1.5+0.03311161\*29.2

Elastic\_net= train(DO~Temp+BOD,data=train,method="glmnet",trControl=trainControl("cv",number=10),tuneLength=10)

Elastic\_net\_model = glmnet(x, y, alpha = Elastic\_net$bestTune$alpha, lambda = Elastic\_net$bestTune$lambda)

summary(Elastic\_net\_model)

coef(Elastic\_net\_model)

#predictions

x\_test = model.matrix(DO~BOD+Temp, test)[,-1]

result\_elas = Elastic\_net\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

5.68355435+-0.01546892\*1.5+0.03006168\*29.2

#comparision

data.frame(RMSE = RMSE(result\_rid, test$DO),Rsquare = R2(result\_rid, test$DO))

data.frame(RMSE = RMSE(result\_las, test$DO),Rsquare = R2(result\_las, test$DO))

data.frame(RMSE = RMSE(result\_elas, test$DO),Rsquare = R2(result\_elas, test$DO))

#DO with all variables

x=model.matrix(DO~., train)[,-1]

y=train$DO

cv = cv.glmnet(x, y, alpha = 0)

cv$lambda.min

Ridge\_model = glmnet(x, y, alpha = 0, lambda = cv$lambda.min)

Ridge\_model

summary(Ridge\_model)

coef(Ridge\_model)

#predictions

x\_test = model.matrix(DO~., test)[,-1]

result\_rid = Ridge\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

6.340615e+00 + 1.397444e-04\*29.2 + 8.815697e-08\*6.3 + -1.464299e-07\*100 + -6.658398e-05\*1.5 + -2.395029e-04\*0.1 + -1.273718e-10\*7942 + -7.772523e-11\*13575

cv = cv.glmnet(x, y, alpha = 1)

cv$lambda.min

Lasso\_model = glmnet(x, y, alpha = 1, lambda = cv$lambda.min)

Lasso\_model

summary(Lasso\_model)

coef(Lasso\_model)

#predictions

x\_test = model.matrix(DO~., test)[,-1]

result\_las = Lasso\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

6.396498431+-0.008600281\*1.5

Elastic\_net= train(DO~.,data=train,method="glmnet",trControl=trainControl("cv",number=10),tuneLength=10)

Elastic\_net\_model = glmnet(x, y, alpha = Elastic\_net$bestTune$alpha, lambda = Elastic\_net$bestTune$lambda)

summary(Elastic\_net\_model)

coef(Elastic\_net\_model)

#predictions

x\_test = model.matrix(DO~., test)[,-1]

result\_elas = Elastic\_net\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

6.391311540+-0.007765156\*1.5

#comparision

data.frame(RMSE = RMSE(result\_rid, test$DO),Rsquare = R2(result\_rid, test$DO))

data.frame(RMSE = RMSE(result\_las, test$DO),Rsquare = R2(result\_las, test$DO))

data.frame(RMSE = RMSE(result\_elas, test$DO),Rsquare = R2(result\_elas, test$DO))

#NITRATE

x1=model.matrix(NITRATE~F\_COLIFORM+T\_COLIFORM, train)[,-1]

y1=train$DO

cv = cv.glmnet(x1, y1, alpha = 0)

cv$lambda.min

Ridge\_model <- glmnet(x1, y1, alpha = 0, lambda = cv$lambda.min)

Ridge\_model

summary(Ridge\_model)

coef(Ridge\_model)

#predictions

x\_test = model.matrix(NITRATE~F\_COLIFORM+T\_COLIFORM, test)[,-1]

result\_rid = Ridge\_model %>% predict(x\_test) %>% as.vector()

cv = cv.glmnet(x1, y1, alpha = 1)

cv$lambda.min

Lasso\_model <- glmnet(x1, y1, alpha = 1, lambda = cv$lambda.min)

Lasso\_model

summary(Lasso\_model)

coef(Lasso\_model)

x\_test = model.matrix(NITRATE~F\_COLIFORM+T\_COLIFORM, test)[,-1]

result\_las = Lasso\_model %>% predict(x\_test) %>% as.vector()

Elastic\_net= train(NITRATE~F\_COLIFORM+T\_COLIFORM,data=train,method="glmnet",trControl=trainControl("cv",number=10),tuneLength=10)

Elastic\_net\_model <- glmnet(x1, y1, alpha = Elastic\_net$bestTune$alpha, lambda = Elastic\_net$bestTune$lambda)

summary(Elastic\_net\_model)

coef(Elastic\_net\_model)

result\_elas = Elastic\_net\_model %>% predict(x\_test) %>% as.vector()

#single value prediction

6.347061e+00 + -1.977305e-08\*27

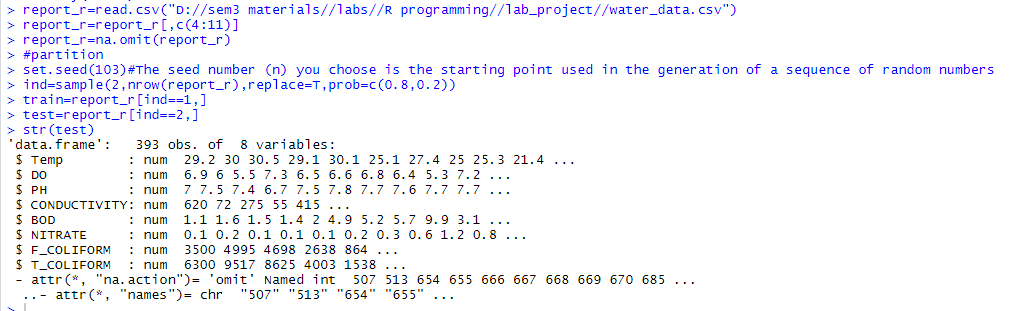
#comparision

data.frame(RMSE = RMSE(result\_rid, test$NITRATE),Rsquare = R2(result\_rid, test$NITRATE))

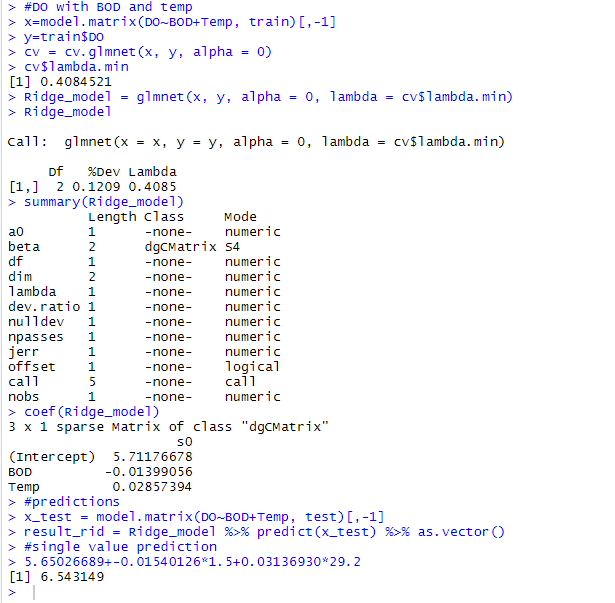
data.frame(RMSE = RMSE(result\_las, test$NITRATE),Rsquare = R2(result\_las, test$NITRATE))

data.frame(RMSE = RMSE(result\_elas, test$NITRATE),Rsquare = R2(result\_elas, test$NITRATE))

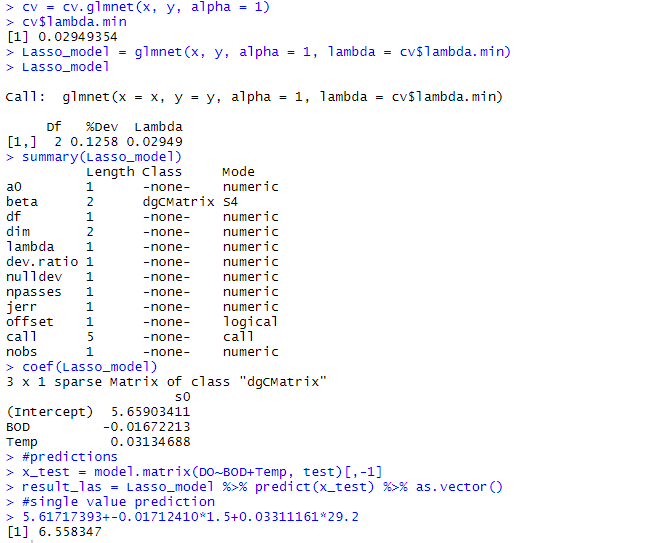
**Output**



Ridge Model



Lasso Model



Elastic net Model

