air\_quality<-read.csv(choose.files())

library(psych)

library(DataExplorer)

library(car)#scatterplot, vif

library(lmtest) #Autocorrelation

library(MASS) #Step-wise regression

library(Metrics) #Loss/Cost function

library(glmnet)

library(dplyr)

plot\_missing(air\_quality)

str(air\_quality)

summary(air\_quality)

air\_quality$Date<-as.factor(air\_quality$Date)

air\_quality$Time<-as.factor(air\_quality$Time)

plot\_missing(air\_quality)

str(air\_quality)

plot\_histogram(MLA1)

plot\_correlation(MLA1)

plot\_density(MLA1)

#Data Partitioning

aq\_mixed<-air\_quality[order(runif(9357)),]

aq\_training<-aq\_mixed[1:6550,]

aq\_testing<-aq\_mixed[6550:9357,]

#Full Model

aq\_lm\_full<-lm(CO.GT.~.,data=aq\_training)

summary(aq\_lm\_full)

#Best Fit

aq\_step<-stepAIC(aq\_lm\_full,direction="backward")

#Reduced Model

aq\_reduced<-lm( CO.GT. ~ Date + Time + NMHC.GT. + PT08.S2.NMHC. + NOx.GT. + PT08.S3.NOx. +

NO2.GT. + PT08.S4.NO2. + PT08.S5.O3. + T + RH + AH , data=aq\_training)

summary(aq\_reduced)

#LASSO REGRESSION

# Load required libraries

library(glmnet)

library(dplyr)

# Create Model matrix (including dummy variables for State)

X <- model.matrix(CO.GT. ~ ., data = air\_quality)

Y <- air\_quality$CO.GT.

# Define the lambda sequence (regularization parameter)

lambda <- 10^seq(10, -2, length = 100)

# Split the data into training and testing sets

set.seed(567)

part <- sample(2, nrow(X), replace = TRUE, prob = c(0.7, 0.3))

X\_train <- X[part == 1, ] # Training Data

X\_test <- X[part == 2, ] # Testing Data

Y\_train <- Y[part == 1] # Training Data

Y\_test <- Y[part == 2] # Testing Data

# Fit Lasso regression model

lasso\_reg <- glmnet(X\_train, Y\_train, alpha = 1, lambda = lambda)

# Perform cross-validation to select the best lambda (minimizing MSE)

lasso\_reg\_cv <- cv.glmnet(X\_train, Y\_train, alpha = 1)

# Get the optimal lambda

best\_lambda <- lasso\_reg\_cv$lambda.min

print(paste("Optimal Lambda (min MSE):", best\_lambda))

# Predict on the testing set using the best lambda

lasso\_pred <- predict(lasso\_reg, s = best\_lambda, newx = X\_test)

# Calculate Mean Squared Error (MSE)

mse <- mean((Y\_test - lasso\_pred)^2)

print(paste("Mean Squared Error (MSE):", mse))

# Calculate R-squared value

sst <- sum((Y\_test - mean(Y\_test))^2)

sse <- sum((Y\_test - lasso\_pred)^2)

r2 <- 1 - (sse / sst)

print(paste("R-squared (R^2):", r2))

# Summary of Lasso regression model

summary(lasso\_reg)

# Predict on the testing set using the best lambda

lasso\_pred <- predict(lasso\_reg, s = best\_lambda, newx = X\_test)

# Create a data frame for actual and predicted values

predictions <- data.frame(Actual = Y\_test, Predicted = lasso\_pred)

#RIGDE REGRESSION

library(dplyr)

library(glmnet)

#View the first few rows of the dataset

head(air\_quality)

#Create Model matrix (including dummy variables for State)

X <- model.matrix(CO.GT. ~ ., data = air\_quality)

Y <- air\_quality$CO.GT.

#Define the lambda sequence

lambda<-10^seq(10,-2,length=100)

print(lambda)

#Split the data into training and validation sets

set.seed(567)

part<-sample(2, nrow(X), replace=TRUE, prob=c(0.7,0.3))

X\_train<-X[part==1, ] #Training Data

X\_test<-X[part==2, ] #Testing Data

Y\_train<-Y[part==1] #Training Data

Y\_test<-Y[part==2] #Testing Data

#Perform Ridge Regression

ridge\_reg<-glmnet(X\_train, Y\_train, alpha=0, lambda=lambda)

summary(ridge\_reg)

#Find the best lambda via cross-validation

ridge\_reg1<-cv.glmnet(X\_train, Y\_train, alpha=0)

bestlam<-ridge\_reg1$lambda.min

print(bestlam)

#Predict on the validation set

ridge.pred<-predict(ridge\_reg, s=bestlam, newx=X\_test)

#Calculate the mean squared error

mse<-mean((Y\_test - ridge.pred)^2)

print(paste("Mean Squared Error: ",mse))

#Calculate R squared Value

sst<-sum((Y\_test - mean(Y\_test))^2)

sse<-sum((Y\_test - ridge.pred)^2)

r2<- 1 - (sse/sst)

print(paste("R square: ",r2))