**CODE:**

**library(psych) # For visualization**

**library(DataExplorer)**

**library(car)**

**library(lmtest)**

**library(ModelMetrics)**

**library(MASS) # For multiple linear regression**

**library(glmnet) # For Lasso and Ridge regression**

**# Import data**

**le\_data <- read.csv("C:/Users/HP/Downloads/student-mat.csv")**

**# Perform EDA**

**# Understand data structure**

**str(le\_data)**

**# Missingness analysis**

**summary(le\_data)**

**plot\_missing(le\_data)**

**# Understand distributions and correlations**

**plot\_correlation(le\_data)**

**# Data partition**

**set.seed(1234)**

**le\_mixed <- le\_data[order(runif(nrow(le\_data))),]**

**le\_training <- le\_mixed[1:floor(0.7\*nrow(le\_mixed)),]**

**le\_testing <- le\_mixed[(floor(0.7\*nrow(le\_mixed))+1):nrow(le\_mixed),]**

**# Simple Linear Regression (using one feature for demonstration, e.g., G1)**

**le\_lm\_simple <- lm(G3 ~ G1, data = le\_training)**

**summary(le\_lm\_simple)**

**# Model diagnostics**

**par(mfrow=c(2,2))**

**plot(le\_lm\_simple)**

**# Evaluate model performance**

**le\_lm\_simple\_pred <- predict(le\_lm\_simple, newdata = le\_testing)**

**mse(le\_testing$G3, le\_lm\_simple\_pred)**

**# Multiple Linear Regression**

**le\_lm\_full <- lm(G3 ~ ., data = le\_training)**

**summary(le\_lm\_full)**

**# Ensure compatibility of variables between training and testing sets**

**common\_vars <- intersect(names(coef(le\_lm\_full)), names(le\_testing))**

**# Build a formula with common variables only**

**formula\_full <- as.formula(paste("G3 ~", paste(common\_vars[-1], collapse = " + ")))**

**# Refit the full model with common variables**

**le\_lm\_full <- lm(formula\_full, data = le\_training)**

**summary(le\_lm\_full)**

**# Predict and evaluate the full model**

**le\_lm\_full\_pred <- predict(le\_lm\_full, newdata = le\_testing)**

**mse\_full <- mse(le\_testing$G3, le\_lm\_full\_pred)**

**print(paste("MSE of Full Multiple Linear Regression: ", mse\_full))**

**# Select the best features using stepwise selection**

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

**# Build the reduced model based on selected features**

**selected\_vars <- names(coef(le\_step))**

**formula\_reduced <- as.formula(paste("G3 ~", paste(selected\_vars[-1], collapse = " + ")))**

**# Ensure variables used in the model are consistent**

**missing\_vars <- setdiff(selected\_vars[-1], colnames(le\_testing))**

**if (length(missing\_vars) > 0) {**

**cat("The following variables are missing in the testing data: ", missing\_vars, "\n")**

**# Remove missing variables from the formula**

**selected\_vars <- setdiff(selected\_vars, missing\_vars)**

**formula\_reduced <- as.formula(paste("G3 ~", paste(selected\_vars[-1], collapse = " + ")))**

**cat("Updated formula: ", formula\_reduced, "\n")**

**}**

**# Build the reduced model with the updated formula**

**le\_lm\_red <- lm(formula\_reduced, data = le\_training)**

**summary(le\_lm\_red)**

**# Model diagnostics for the reduced model**

**par(mfrow=c(2,2))**

**plot(le\_lm\_red)**

**# Use the reduced model for predictions**

**le\_lm\_red\_pred <- predict(le\_lm\_red, newdata = le\_testing)**

**# Check if predictions are generated correctly**

**head(le\_lm\_red\_pred)**

**# Evaluate the reduced model performance**

**mse\_reduced <- mse(le\_testing$G3, le\_lm\_red\_pred)**

**print(paste("MSE of Reduced Multiple Linear Regression: ", mse\_reduced))**

**# Prepare data for Lasso and Ridge regression**

**x\_train <- model.matrix(G3 ~ .-1, data = le\_training)**

**y\_train <- le\_training$G3**

**x\_test <- model.matrix(G3 ~ .-1, data = le\_testing)**

**y\_test <- le\_testing$G3**

**# Lasso Regression**

**lasso\_model <- glmnet(x\_train, y\_train, alpha = 1)**

**cv\_lasso <- cv.glmnet(x\_train, y\_train, alpha = 1)**

**best\_lambda\_lasso <- cv\_lasso$lambda.min**

**lasso\_pred <- predict(lasso\_model, s = best\_lambda\_lasso, newx = x\_test)**

**mse\_lasso <- mse(y\_test, lasso\_pred)**

**print(paste("MSE of Lasso Regression: ", mse\_lasso))**

**# Ridge Regression**

**ridge\_model <- glmnet(x\_train, y\_train, alpha = 0)**

**cv\_ridge <- cv.glmnet(x\_train, y\_train, alpha = 0)**

**best\_lambda\_ridge <- cv\_ridge$lambda.min**

**ridge\_pred <- predict(ridge\_model, s = best\_lambda\_ridge, newx = x\_test)**

**mse\_ridge <- mse(y\_test, ridge\_pred)**

**print(paste("MSE of Ridge Regression: ", mse\_ridge))**