```r
#Install necessary packages
install.packages('xgboost')
install.packages('caret')
install.packages('e1071')

#Load required libraries
library(xgboost)
library(caret)
library(e1071)

#Load the Iris dataset
data <- iris
head(data)
summary(data)
dim(data)

#Pre-processing : Remove Missing Values
sum(is.na(data))
iris <- na.omit(data)
sum(is.na(iris))

#Pre-processing : Normalization or Scaling
preproc <- preProcess(iris[, -5], method = c("center", "scale"))
iris[, -1] <- predict(preproc, iris[, -5])
set.seed(123)  # For reproducibility

#Splitting
splitIndex <- createDataPartition(data$Species, p = 0.8, list = FALSE)
training_data <- data[splitIndex, ]
testing_data <- data[-splitIndex, ]
nrow(training_data)
nrow(testing_data)

#Extract independent variables (features) and dependent variable (target) for training set
X_train <- data.matrix(training_data[, -5])
y_train <- training_data[, 5]

#Extract independent variables (features) and dependent variable (target) for testing set
X_test <- data.matrix(testing_data[, -5])
y_test <- testing_data[, 5]

#Convert the train and test data into xgboost matrix type
xgboost_train <- xgb.DMatrix(data = X_train, label = y_train)
xgboost_test <- xgb.DMatrix(data = X_test, label = y_test)

#Train an xgboost model using the training data
model <- xgboost(data = xgboost_train,max.depth = 3,nrounds = 50)
#Display a summary of the trained model
summary(model)

#Use the trained model to make predictions on the test data
predictions <- predict(model, xgboost_test)
# Display the predicted values
print(predictions)

#Set predicted values greater than 3 to 3
predictions[(predictions > 3)] <- 3

#Convert predicted values to factors
predictions <- as.factor((levels(y_test))[round(predictions)])
print(predictions)

#Create a confusion matrix to evaluate the model performance
confusionMatrix(predictions,y_test)
```