EX.NO:8 **210701183**

Implement SVM/Decision tree classification technique

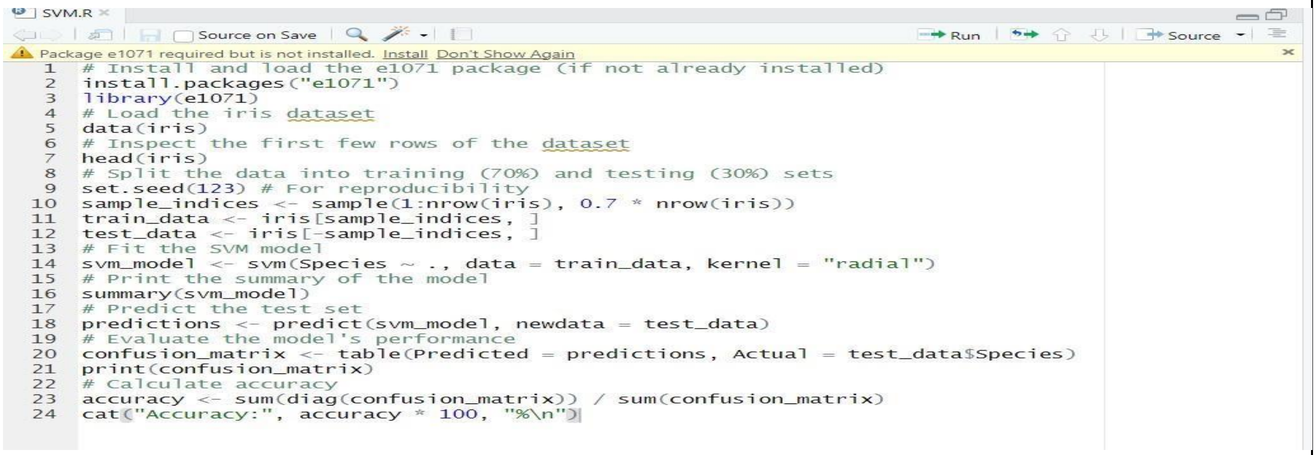
AIM:

To Implement SVM and Decision tree classification techniques using R programming in R Studio.

a) SVM IN R

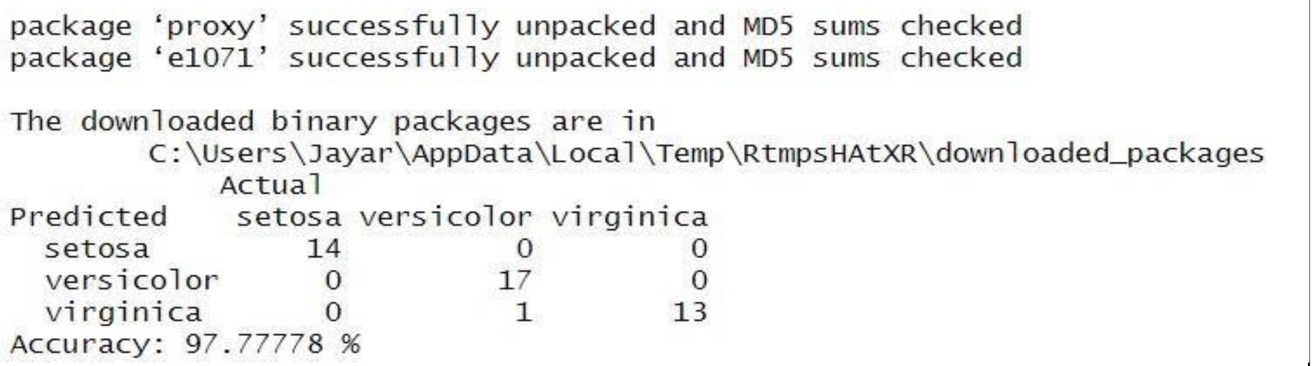
```
# Install and load the e1071 package (if not already installed)
install.packages("e1071") library(e1071)

# Load the iris dataset data(iris)

# Inspect the first few rows of the dataset head(iris)

# Split the data into training (70%) and testing (30%) sets
set.seed(123)  # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]   test_data <- iris[-
sample_indices, ]

# Fit the SVM model svm_model <- svm(Species ~ ., data =
train_data, kernel = "radial")

# Print the summary of the model   summary(svm_model)

# Predict the test set predictions <- predict(svm_model,
newdata = test_data)

# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)

# Calculate accuracy accuracy <-
sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")
```

OUTPUT:



Package e1071 required but is not installed. Install Don't Show Again

```
# Install and load the e1071 package (if not already installed)
install.packages("e1071")
library(e1071)
# Load the iris dataset
data(iris)
# Inspect the first few rows of the dataset
head(iris)
# Split the data into training (70%) and testing (30%) sets
set.seed(123) # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]
test_data <- iris[-sample_indices, ]
# Fit the SVM model
svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")
# Print the summary of the model
summary(svm_model)
# Predict the test set
predictions <- predict(svm_model, newdata = test_data)
# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")
```

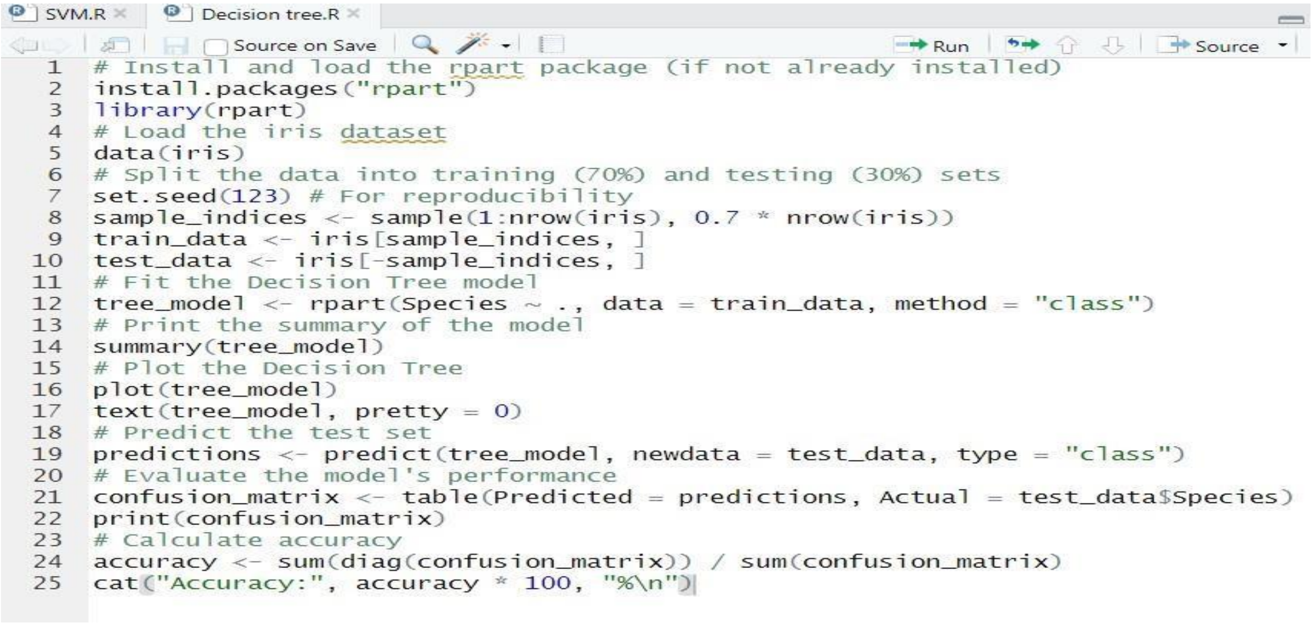


```
package ‘proxy’ successfully unpacked and MD5 sums checked
package ‘e1071’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\Jayar\AppData\Local\Temp\RtmpsHAtXR\downloaded_packages
            Actual
Predicted    setosa versicolor virginica
  setosa         14          0         0
  versicolor      0         17         0
  virginica       0          1        13
Accuracy: 97.77778 %
```

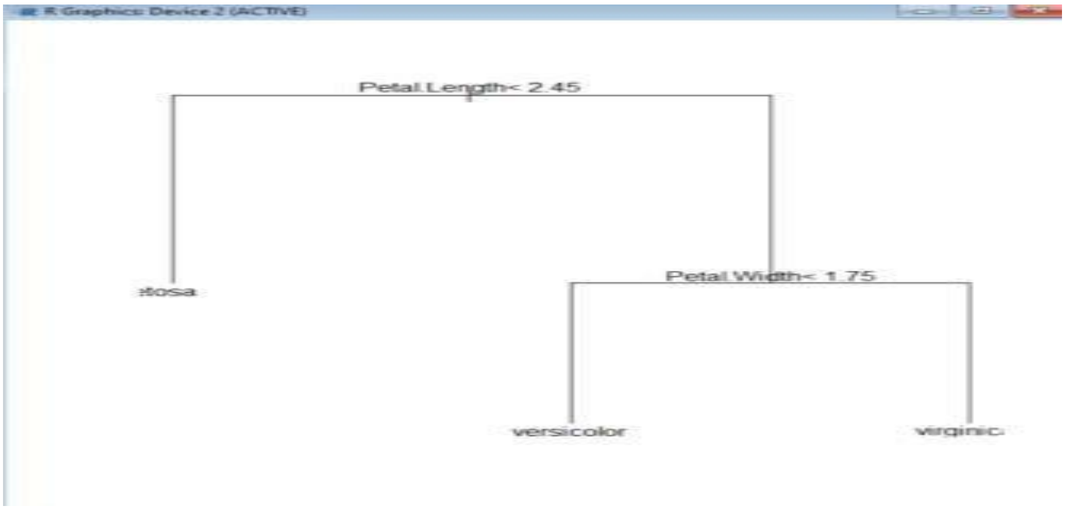
b) Decision tree in R

```
# Install and load the rpart package (if not already installed)
install.packages("rpart") library(rpart)

# Load the iris dataset data(iris)

# Split the data into training (70%) and testing (30%) sets
set.seed(123)  # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]   test_data <- iris[-
sample_indices, ]

# Fit the Decision Tree model tree_model <- rpart(Species ~
., data = train_data, method = "class")

# Print the summary of the model  summary(tree_model)

# Plot the Decision Tree
plot(tree_model) text(tree_model,
pretty =
0)

# Predict the test set predictions <- predict(tree_model,
newdata = test_data, type = "class")

# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)

# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)  cat("Accuracy:",
accuracy * 100, "%\n")
```

OUTPUT:



```
# Install and load the rpart package (if not already installed)
install.packages("rpart")
library(rpart)
# Load the iris dataset
data(iris)
# Split the data into training (70%) and testing (30%) sets
set.seed(123) # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]
test_data <- iris[-sample_indices, ]
# Fit the Decision Tree model
tree_model <- rpart(Species ~ ., data = train_data, method = "class")
# Print the summary of the model
summary(tree_model)
# Plot the Decision Tree
plot(tree_model)
text(tree_model, pretty = 0)
# Predict the test set
predictions <- predict(tree_model, newdata = test_data, type = "class")
# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")
```



RESULT:

Thus, the Implementation SVM/Decision tree classification techniques using R programming in R Studio.