

**Exp.No: 8**

## **Implement SVM/Decision tree classification techniques**

### **AIM:**

To Implement SVM/Decision tree classification techniques using R.

### **PROCEDURE:**

- Load the dataset from sources such as CSV files or databases using suitable libraries.
- Perform data cleaning and preprocessing, including addressing missing values and encoding categorical variables.
- Divide the dataset into training and testing sets to effectively assess model performance.
- Normalize or standardize the features, particularly for SVM, to ensure uniform scaling across inputs.
- Select the appropriate model: use SVM for margin-based classification or Decision Tree for rule-based classification tasks.
- Train the model on the training data using the fit method.
- Generate predictions on the testing data with the predict method.
- Evaluate the model's performance using metrics such as accuracy, confusion matrix, precision, and recall.
- Visualize the outcomes with relevant plots, like decision boundaries for SVM or tree structures for Decision Trees.
- Tune the model by adjusting hyperparameters, such as C for SVM or max\_depth for Decision Trees.

### **PROGRAM:**

#### **SVM.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")
```

**Decision Tree.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:****SVM :**

The screenshot displays the RStudio interface with a script editor, environment pane, and console output.

**Script Editor (Exp 8 a SVM.R):**

```

1 # Install and load the e1071 package (if not already installed)
2 install.packages("e1071")
3 library(e1071)
4 # Load the iris dataset
5 data(iris)
6 # Inspect the first few rows of the dataset
7 head(iris)
8 # Split the data into training (70%) and testing (30%) sets
9 set.seed(123) # For reproducibility
10 sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
11 train_data <- iris[sample_indices, ]
12 test_data <- iris[-sample_indices, ]
13 # Fit the SVM model
14 svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")
15 # Print the summary of the model
16 summary(svm_model)
17 # Predict the test set
18 predictions <- predict(svm_model, newdata = test_data)
19 # Evaluate the model's performance
20 confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
21 print(confusion_matrix)
22 # Calculate accuracy
23 accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
24 cat("Accuracy:", accuracy * 100, "%\n")
25
26

```

**Environment Pane:**

Object	Class	Size
iris	Data	150 obs. of 5 variables
svm_model	List	of 31
test_data	Data	45 obs. of 5 variables
train_data	Data	105 obs. of 5 variables

**Values:**

accuracy	0.977777777777778
confusion_matrix	'table' int [1:3, 1:3] 14 0 0 17 1 0 0 13
predictions	Factor w/ 3 levels "setosa", "versicolor", ...: 1 1 1 1 1 ...
sample_indices	int [1:105] 14 50 118 43 150 148 90 91 143 92 ...

**Console Output:**

```

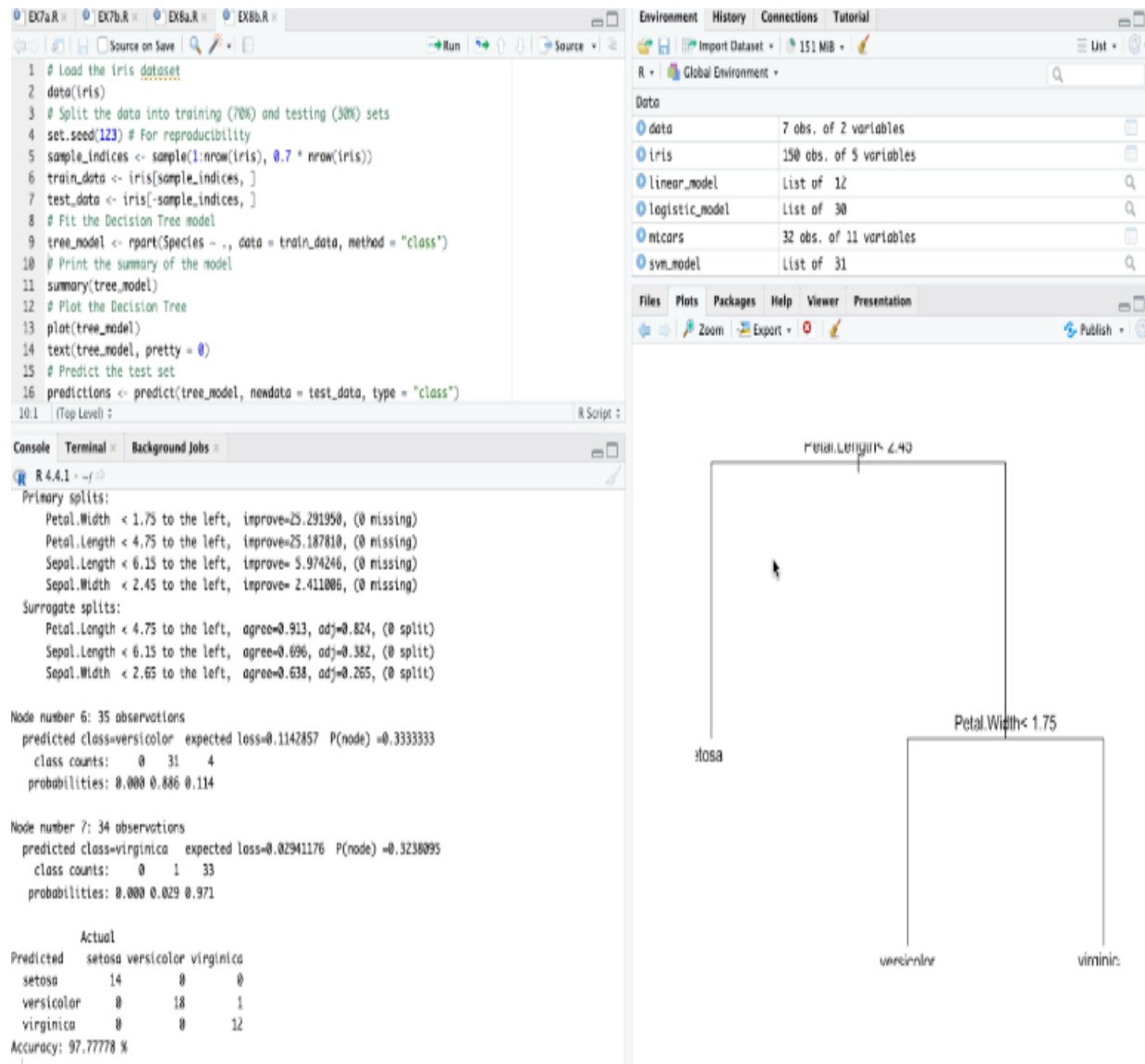
R 4.4.1 - C:\Users\shrin\OneDrive\Desktop\
Current type: application/x-rscript; charset=utf-8; length=6536 bytes (6536 bytes)
downloaded 656 KB

package 'e1071' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
  C:\Users\shrin\AppData\Local\Temp\Rtmpc7iUEu\downloaded_packages
Actual
Predicted  setosa versicolor virginica
setosa     14         0         0
versicolor  0         17        0
virginica   0          1        13
Accuracy: 97.77778 %
>

```

## Decision tree:



## RESULT:

Thus SVM and Decision tree classification techniques has been successfully executed.