

Exp.No: 8**IMPLEMENT SVM/DECISION TREE CLASSIFICATION TECHNIQUES****AIM:**

To write an R code to implement SVM/decision tree classification techniques.

PROCEDURE:

1. Install and load the required packages (e1071 for SVM and rpart for Decision Tree) and load the iris dataset.
2. Split the dataset into training (70%) and testing (30%) sets using a reproducible random sampling method.
3. Fit the SVM model with a radial kernel using the training data, print the model summary, and evaluate its performance using a confusion matrix and accuracy calculation.
4. Fit the Decision Tree model using the rpart function with the training data, print the model summary, visualize the tree, and evaluate its performance using a confusion matrix and accuracy calculation.
5. Predict the test set results for both SVM and Decision Tree models and assess their accuracy.

PROGRAM CODE:**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:

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains R code for training an SVM model with a radial kernel on the iris dataset and predicting on test data.
- Console:** Displays the output of the SVM training, including parameters (SVM-Type: C-classification, SVM-Kernel: radial, cost: 1), the number of support vectors (45), and the number of classes (3).
- Environment:** Lists the objects in the global environment: data (7 obs. of 2 variables), iris (150 obs. of 5 variables), linear_model (List of 12), logistic_model (List of 30), mtcars (32 obs. of 11 variables), svm_model (List of 31), test_data (45 obs. of 5 variables), and train_data (105 obs. of 5 variables).
- Values:** Displays the results of the accuracy calculation and the confusion matrix. The accuracy is 0.977777777777778. The confusion matrix is a 3x3 table showing counts for setosa, versicolor, and virginica.
- Files:** Shows the system library with various R packages and their versions.

Console Output:

```
svm(formula = Species ~ ., data = train_data, kernel = "radial")

Parameters:
  SVM-Type:  C-classification
 SVM-Kernel: radial
      cost:  1

Number of Support Vectors: 45
( 7 18 20 )

Number of Classes: 3

Levels:
setosa versicolor virginica

> # 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)
      Actual
Predicted setosa versicolor virginica
setosa      14         0         0
versicolor  0         17         0
virginica   0          1        13

> # Calculate accuracy
> accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
> cat("Accuracy:", accuracy * 100, "%\n")
Accuracy: 97.77778 %
```

Environment:

Object	Details
data	7 obs. of 2 variables
iris	150 obs. of 5 variables
linear_model	List of 12
logistic_model	List of 30
mtcars	32 obs. of 11 variables
svm_model	List of 31
test_data	45 obs. of 5 variables
train_data	105 obs. of 5 variables

Values:

```
accuracy      0.977777777777778
confusion_matrix 'table' int [1:3, 1:3] 14 0 0 0 17 1 0 0 13
heights      num [1:7] 150 160 165 170 175 180 185
predicted_probs Named num [1:32] 0.461 0.461 0.598 0.492 0.297 ...
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 ...
weights       num [1:7] 55 60 62 68 70 75 80
```

Files:

Name	Description	Version
base	The R Base Package	4.4.1
BH	Boost C++ Header Files	1.84.0-0
BioCManager	Access the Bioconductor Project Package Repository	1.30.25
BiocParallel	Bioconductor facilities for parallel evaluation	1.38.0
BiocVersion	Set the appropriate version of Bioconductor packages	3.19.1
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-31
class	Functions for Classification	7.3-22
cli	Helpers for Developing Command Line Interfaces	3.6.3
cluster	"Finding Groups in Data": Cluster Analysis Extended	2.1.6

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)

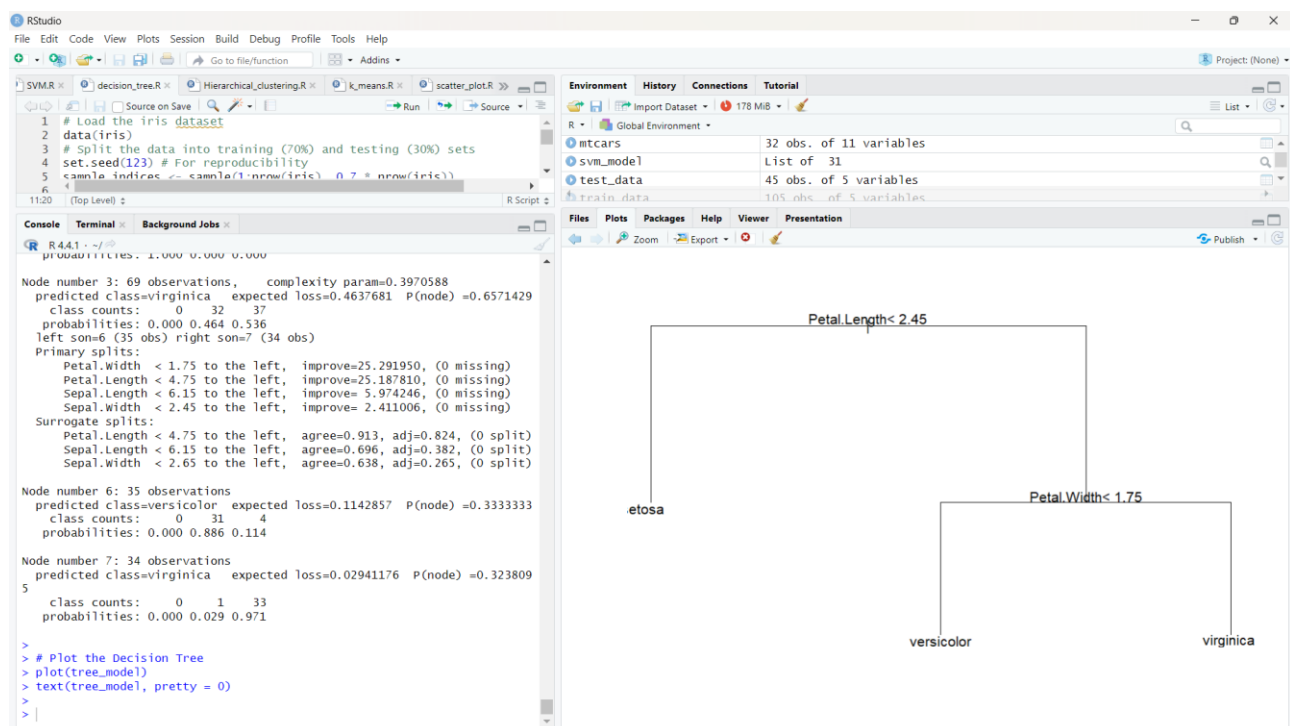
# Predict the test set
predictions <- predict(tree_model, newdata = test_data, type = "class")

# 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:



RESULT:

Thus the R program to implement SVM/decision tree classification techniques has been executed and verified successfully.