

**Ex No 8**

**Implement SVM/Decision tree classification techniques**

**AIM:**

To Implement SVM/Decision tree classification techniques using R.

**PROCEDURE:**

- Collect and load the dataset from sources like CSV files or databases.
- Clean and preprocess the data, including handling missing values and encoding categorical variables.
- Split the dataset into training and testing sets to evaluate model performance.
- Normalize or standardize the features, especially for SVM, to ensure consistent scaling.
- Choose the appropriate model: SVM for margin-based classification, Decision Tree for rule-based classification.
- Train the model on the training data using the 'fit' method.
- Make predictions on the testing data using the 'predict' method.
- Evaluate the model using metrics like accuracy, confusion matrix, precision, and recall.
- Visualize the results with plots, such as decision boundaries for SVM or tree structures for Decision Trees.
- Fine-tune the model by adjusting hyperparameters like `C` for SVM or `max\_depth` for Decision Trees.

**CODE:**

**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, ]
```

210701238

```
# 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")
```

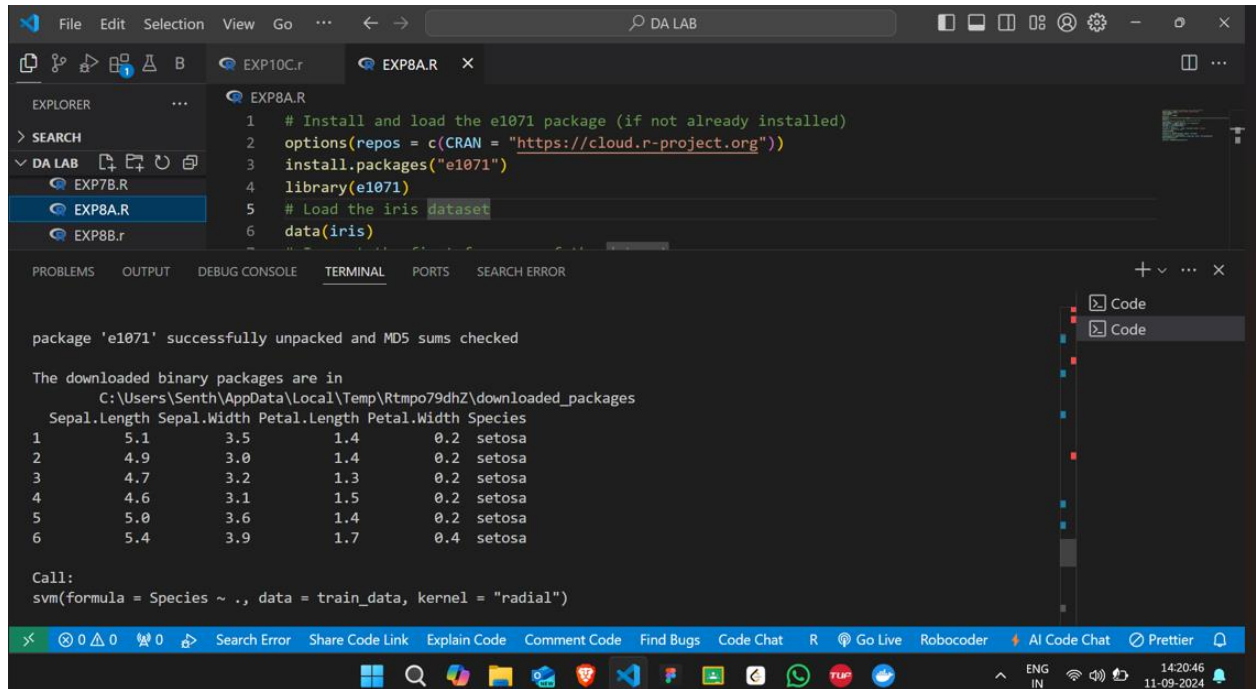
### **Decision Tree.R:**

```
# 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")
```

210701238

## OUTPUT:

## SVM in R:



The screenshot shows a Visual Studio Code editor with a file named `EXP8A.R` open. The code in the editor is as follows:

```
1 # Install and load the e1071 package (if not already installed)
2 options(repos = c(CRAN = "https://cloud.r-project.org"))
3 install.packages("e1071")
4 library(e1071)
5 # Load the iris dataset
6 data(iris)
```

The terminal window at the bottom displays the output of the R script. It shows that the `e1071` package was successfully installed and loaded. The output also displays the first six rows of the `iris` dataset, which are all `setosa` species. Finally, the `svm` function is called with the formula `Species ~ .` and the `train_data` dataset, using a `radial` kernel.

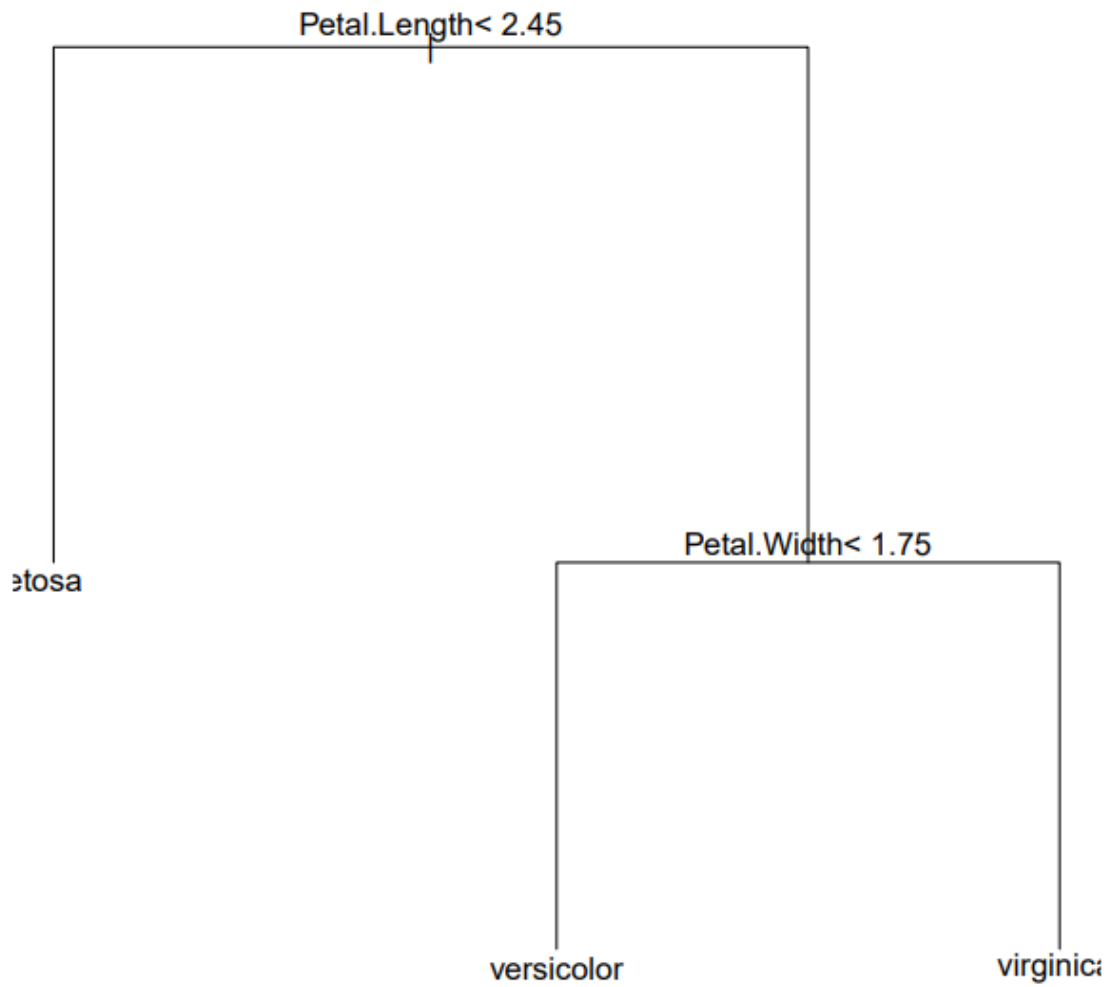
```
package 'e1071' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:\Users\Senth\AppData\Local\Temp\Rtmpo79dhZ\downloaded_packages
Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4          0.2  setosa
2          4.9         3.0          1.4          0.2  setosa
3          4.7         3.2          1.3          0.2  setosa
4          4.6         3.1          1.5          0.2  setosa
5          5.0         3.6          1.4          0.2  setosa
6          5.4         3.9          1.7          0.4  setosa

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

210701238

**Decision tree:**



210701238

**RESULT:**

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