

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")
```

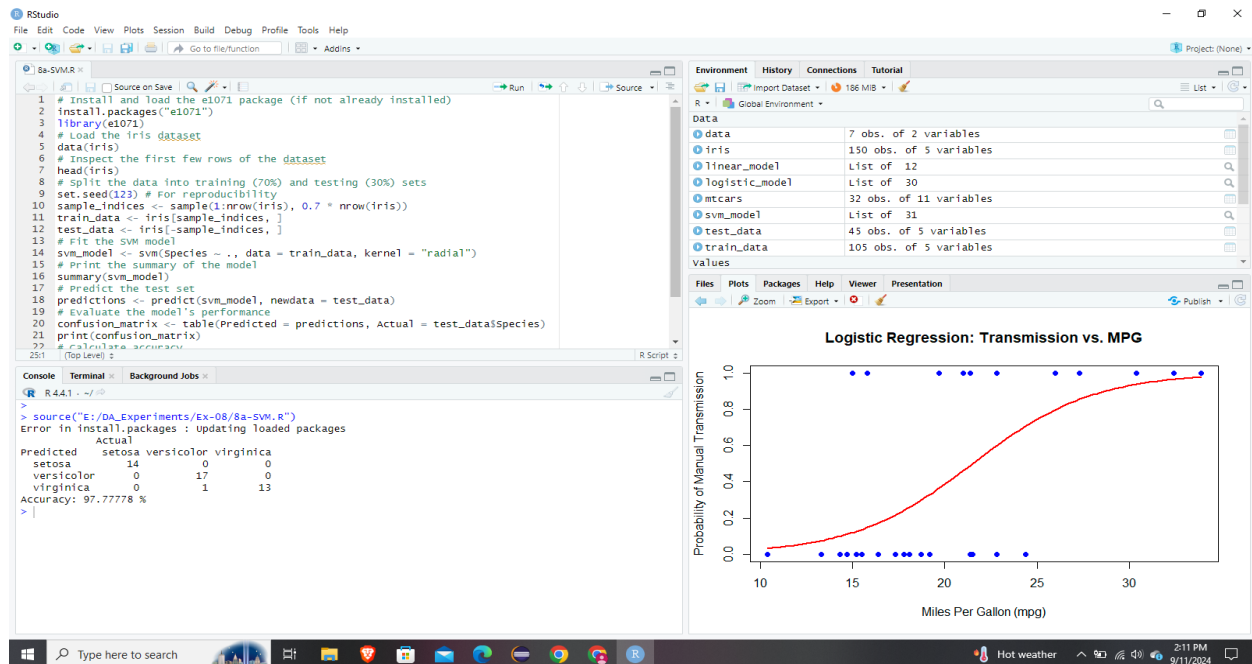
Decision Tree.R:

```
# 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, ]
```

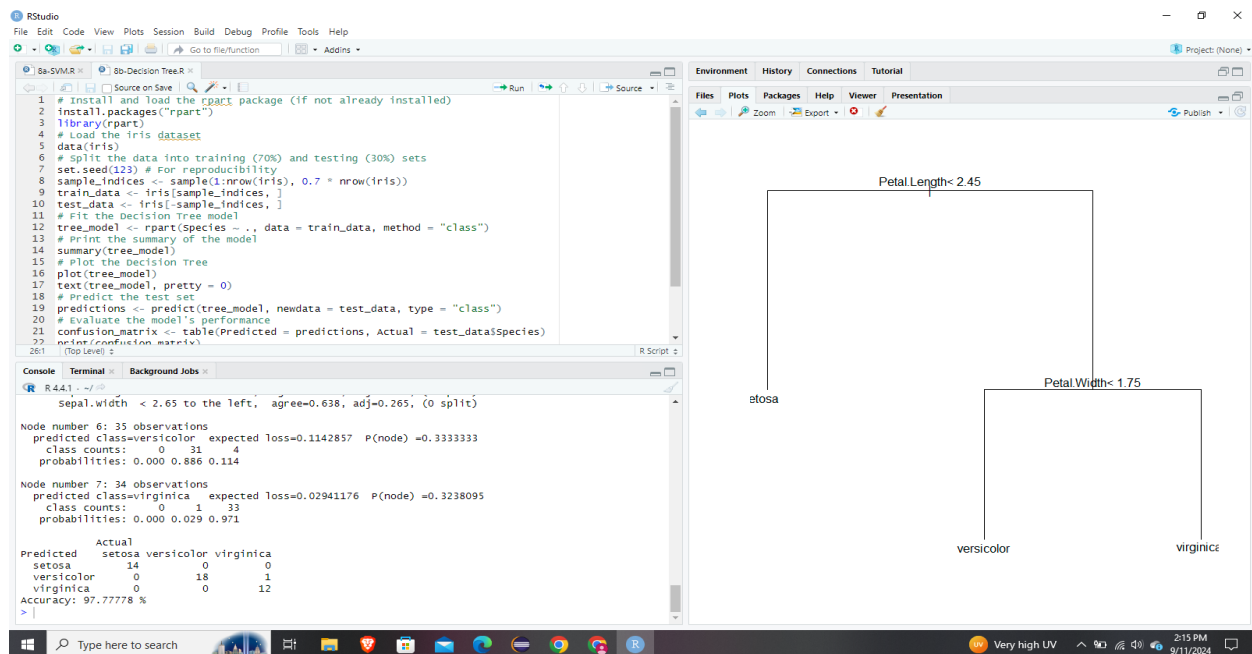
```
test_data <- iris[-sample_indices, ]
# Fit the Decision Tree model
tree_model <- rpart(Species ~ ., data = train_data, method = "class")
# Print the summary of the model
summary(tree_model)
# Plot the Decision Tree
plot(tree_model)
text(tree_model, pretty = 0)
# Predict the test set
predictions <- 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:



Decision tree:



RESULT:

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