

Implement SVM/Decision tree classification techniques**AIM:**

To Implement SVM/Decision tree classification techniques using R.

PROCEDURE:

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

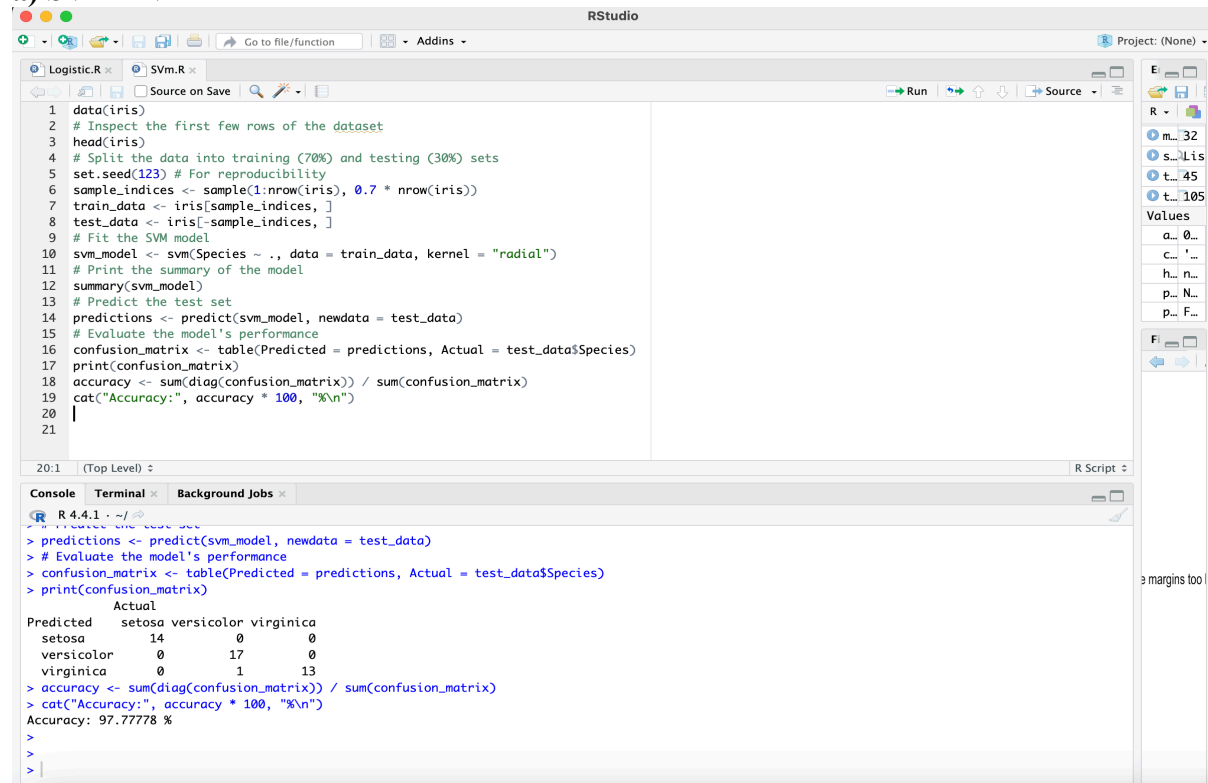
CODE:

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

OUTPUT:

a) SVM IN R



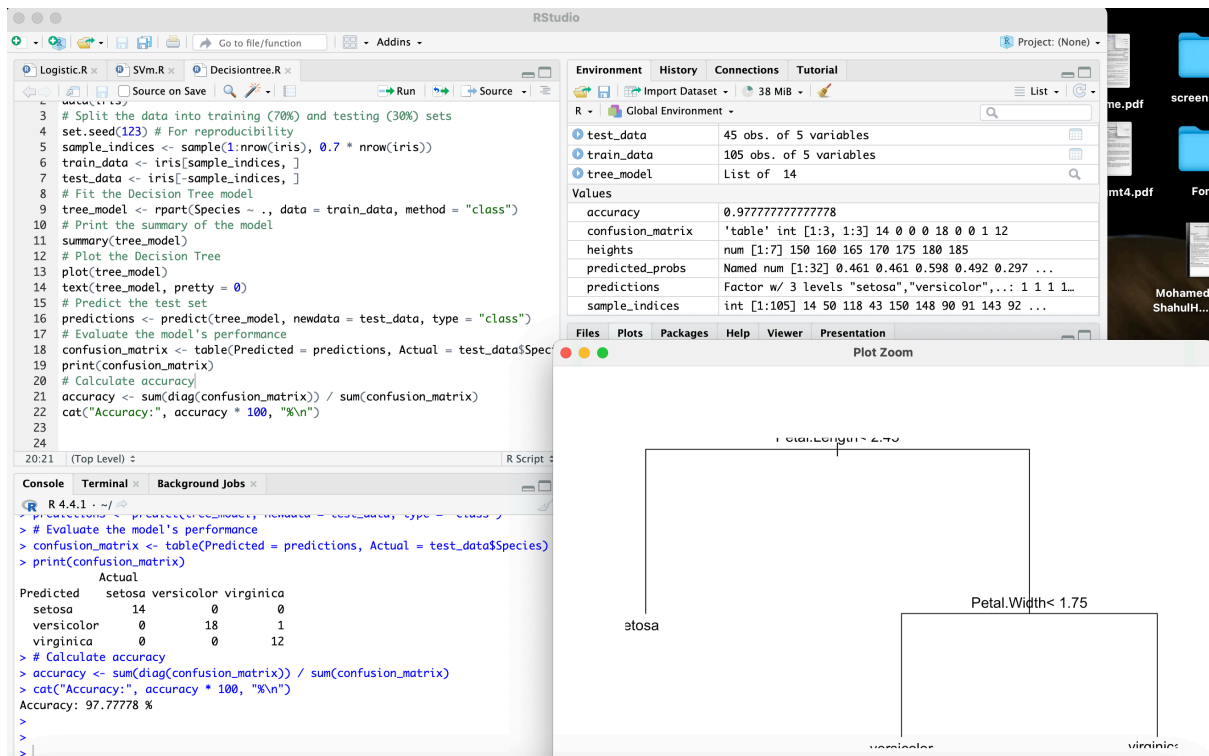
The screenshot shows the RStudio interface with a script editor and a console. The script editor contains R code for training and evaluating an SVM model on the iris dataset. The console shows the output of the code, including the confusion matrix and the calculated accuracy.

```
1 data(iris)
2 # Inspect the first few rows of the dataset
3 head(iris)
4 # Split the data into training (70%) and testing (30%) sets
5 set.seed(123) # For reproducibility
6 sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
7 train_data <- iris[sample_indices, ]
8 test_data <- iris[-sample_indices, ]
9 # Fit the SVM model
10 svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")
11 # Print the summary of the model
12 summary(svm_model)
13 # Predict the test set
14 predictions <- predict(svm_model, newdata = test_data)
15 # Evaluate the model's performance
16 confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
17 print(confusion_matrix)
18 accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
19 cat("Accuracy:", accuracy * 100, "%\n")
20
21
```

Console Output:

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

b)Decision tree:



RESULT:

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