

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

To implement SVM/Decision tree classification techniques using RStudio.

Procedure:**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 two windows. The left window is RGui's console, displaying the output of the R code. It shows the number of classes (3), the levels (setosa, versicolor, virginica), the predicted vs actual confusion matrix, and the calculated accuracy (97.77778%). The right window is an R script editor showing the code used to train and evaluate the SVM model. The code includes installing the e1071 package, loading the iris dataset, splitting it into training and testing sets, fitting the SVM model with a radial kernel, and calculating the accuracy.

```
# RGui Console
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 %
> |

# RStudio Editor
# 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")
```

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)

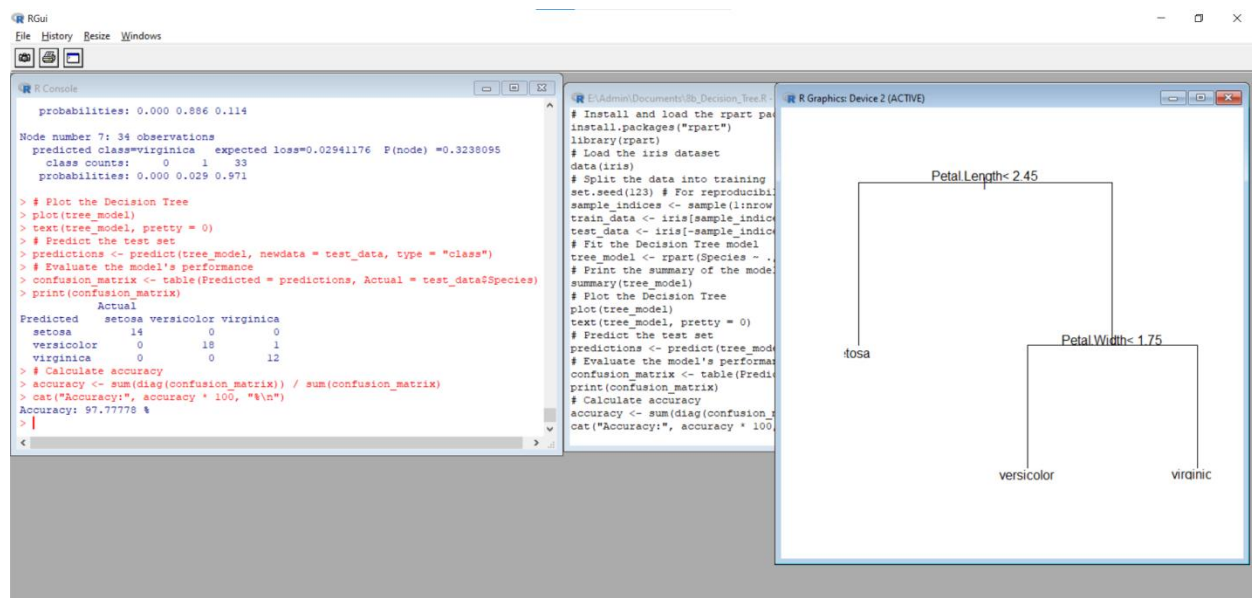
# Calculate accuracy

accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)

cat("Accuracy:", accuracy * 100, "%\n")

```

Output:



Result:

Thus, SVM/Decision tree classification techniques using R and RStudio was implemented successfully.