

DATE:**IMPLEMENT SVM/DECISION TREE CLASSIFICATION TECHNIQUES****AIM:**

To implement SVM/Decision tree classification techniques.

PROGRAM CODE:**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")
```

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:

SVM in R:

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for loading the Iris dataset, splitting it into training and testing sets, training an SVM model with a radial kernel, and evaluating its performance.
- Environment:** Shows the objects in the global environment:

Object	Details
iris	150 obs. of 5 variables
svm_model	List of 31
test_data	45 obs. of 5 variables
train_data	105 obs. of 5 variables
- Values:** Displays the results of the model evaluation:

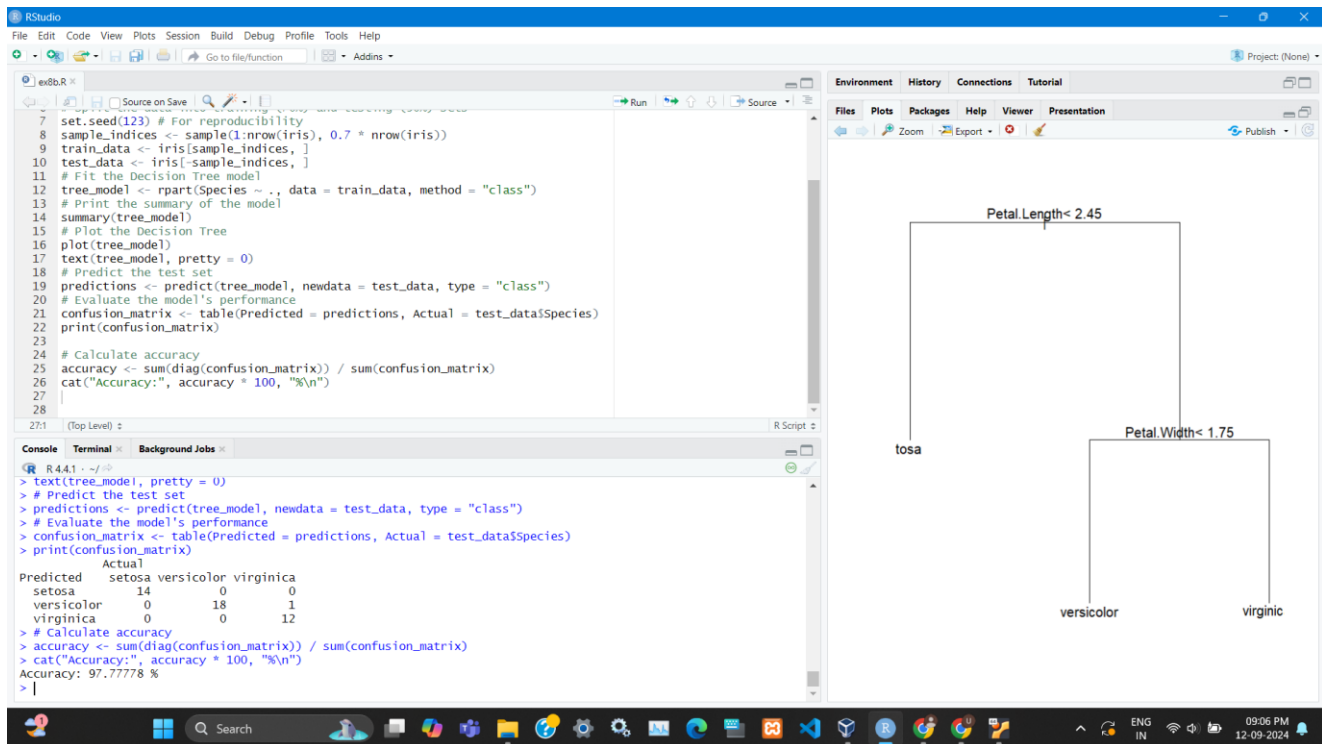
Variable	Value
accuracy	0.977777777777778
confusion_matrix	'table' int [1:3, 1:3] 14 0 0 0 17 1 0 0 13
predictions	Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 ...
sample_indices	int [1:105] 14 50 118 43 150 148 90 91 143 92 ...
- Console:** Shows the execution of the code, including the confusion matrix and the final accuracy 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
> # Calculate accuracy
> accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
> cat("Accuracy:", accuracy * 100, "%\n")
Accuracy: 97.77778 %

```
- Files:** Lists installed and available packages, including 'cli', 'corplot', 'cpp11', 'e1071', 'glue', 'igraph', 'lifecycle', 'magrittr', 'mvtnorm', 'pkgconfig', 'proxy', 'RColorBrewer', 'rlang', and 'vctrs'.

Decision Tree in R:



RESULT:

Thus the implementation of SVM/Decision tree classification techniques done successfully.