

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, ]

# 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 in R:**

```

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 (Top Level)

```

Number of Fisher Scoring iterations: 5

Predicted	Actual
setosa	setosa
versicolor	versicolor
virginica	virginica

Accuracy: 97.77778 %

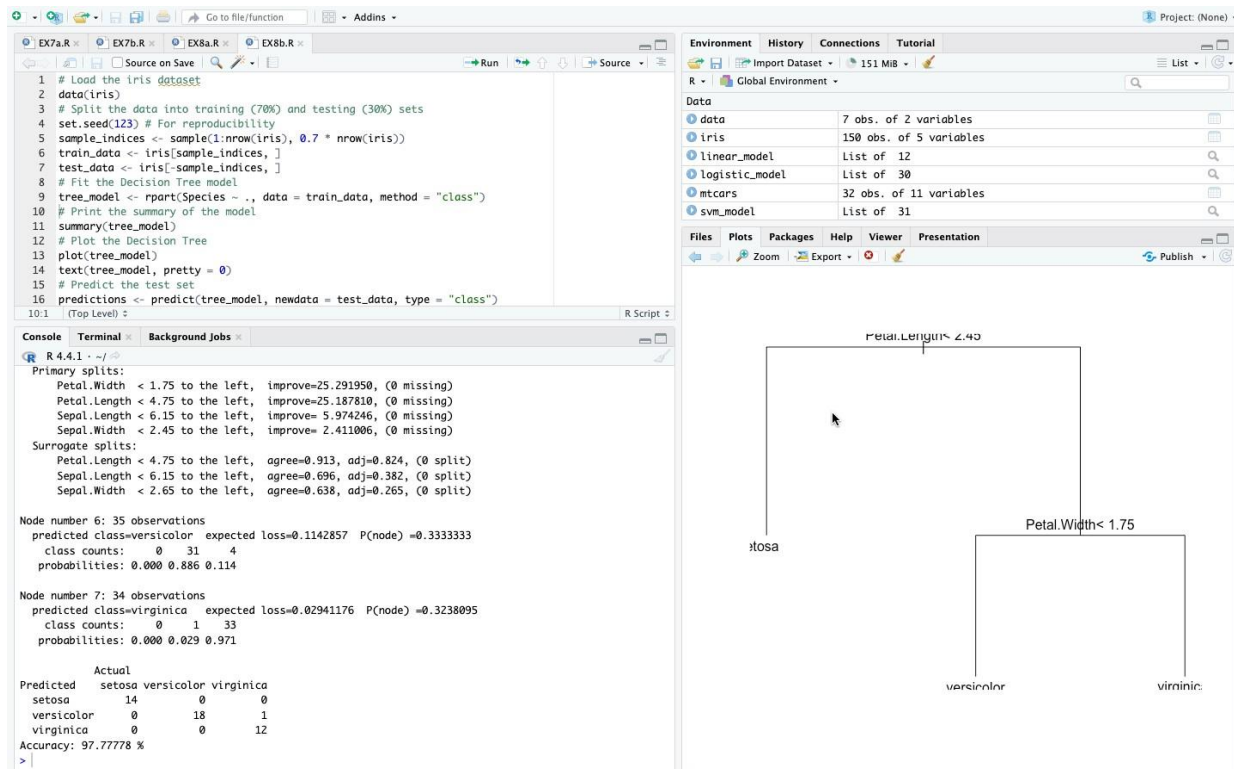
Environment: Global Environment

Object	Class	Size
data	data.frame	7 obs. of 2 variables
iris	data.frame	150 obs. of 5 variables
linear_model	list	12
logistic_model	list	30
mtcars	data.frame	32 obs. of 11 variables
svm_model	svm	31
test_data	data.frame	45 obs. of 5 variables
train_data	data.frame	105 obs. of 5 variables
tree_model	list	14

Values

Object	Value
accuracy	0.977777777777778
confusion_matrix	'table' int [1:3, 1:3] 14 0 0 0 17 1 0 0 13
heights	num [1:7] 150 160 165 170 175 180 185
predicted_probs	Named num [1:32] 0.461 0.461 0.598 0.492 0.297 ...
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 ...
weights	num [1:7] 55 60 62 68 70 75 80

Decision tree:



RESULT:

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