

EXP 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:

a) SVM.R:

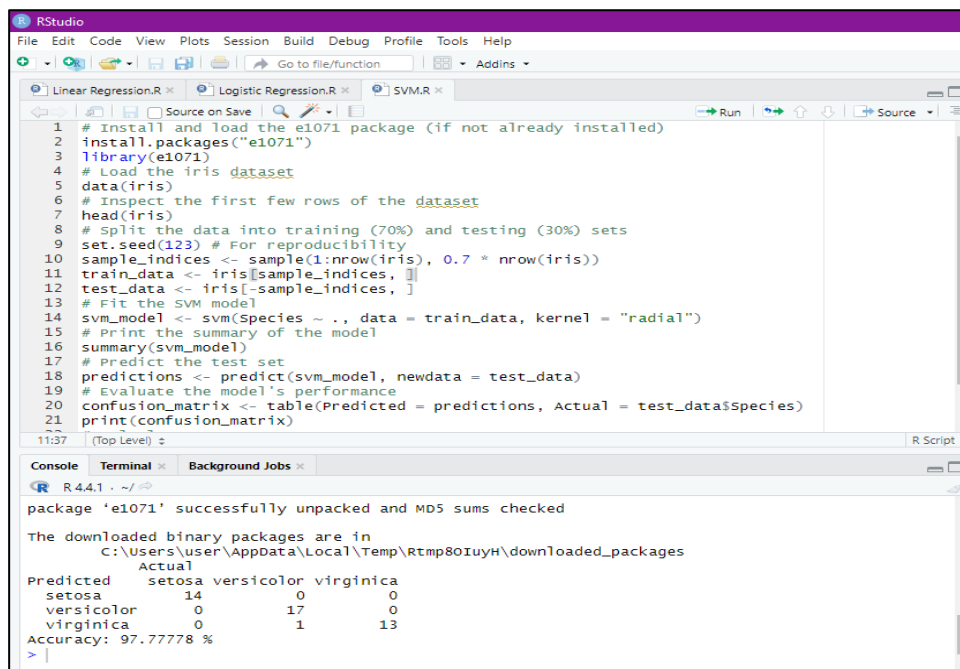
```
# Install and load the e1071 package (if not already installed)
install.packages("e1071")
library(e1071)
# Load the iris dataset
data(iris)
head(iris)
# Split the data into training (70%) and testing (30%) sets
set.seed(123) # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
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)
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.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:



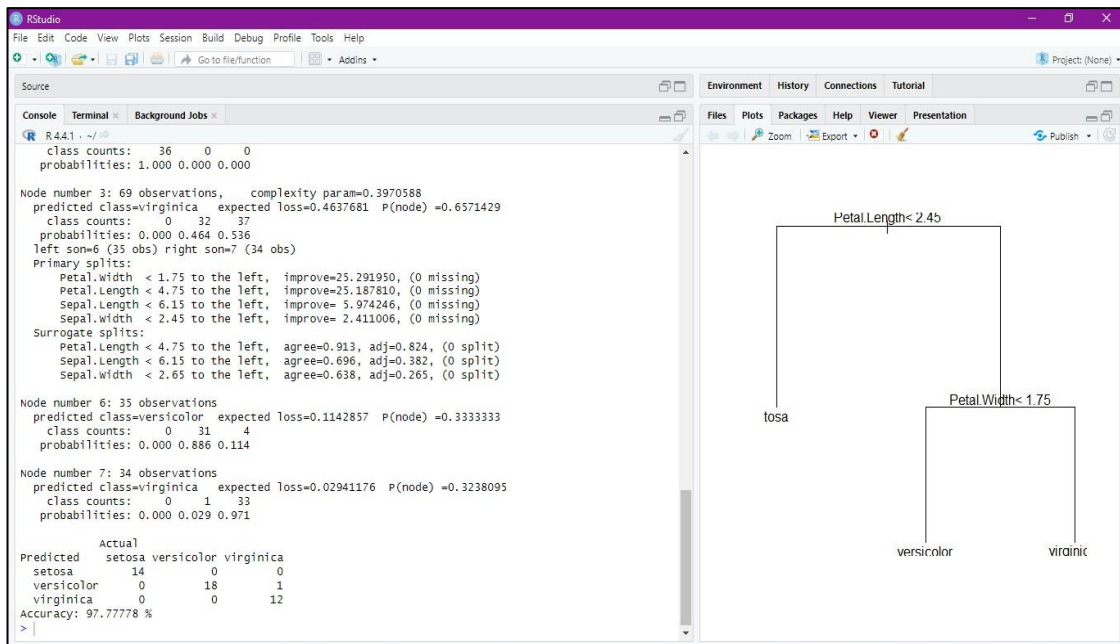
The screenshot shows the RStudio interface with a script editor containing R code for SVM classification. The code includes installing the 'e1071' package, loading the 'iris' dataset, splitting it into training and testing sets, fitting an SVM model with a radial kernel, and evaluating its performance. The console output shows the successful installation of 'e1071', the path to the downloaded packages, and the resulting confusion matrix and accuracy.

```
1 # Install and load the e1071 package (if not already installed)
2 install.packages("e1071")
3 library(e1071)
4 # Load the iris dataset
5 data(iris)
6 # Inspect the first few rows of the dataset
7 head(iris)
8 # Split the data into training (70%) and testing (30%) sets
9 set.seed(123) # For reproducibility
10 sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
11 train_data <- iris[sample_indices, ]
12 test_data <- iris[-sample_indices, ]
13 # Fit the SVM model
14 svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")
15 # Print the summary of the model
16 summary(svm_model)
17 # Predict the test set
18 predictions <- predict(svm_model, newdata = test_data)
19 # Evaluate the model's performance
20 confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
21 print(confusion_matrix)
```

Console Output:

```
R 4.4.1 ~ /
package 'e1071' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\User\AppData\Local\Temp\Rtmp80IuyH\downloaded_packages
      Actual
Predicted setosa versicolor virginica
setosa    14         0          0
versicolor 0         17         0
virginica  0          1         13
Accuracy: 97.7778 %
> |
```

Decision tree:



RESULT:

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