Exp.No: 8

Implement SVM/Decision tree classification techniques

AIM:

To Implement SVM/Decision tree classification techniques using R.

PROCEDURE:

- Load the dataset from sources such as CSV files or databases using suitable libraries.
- Perform data cleaning and preprocessing, including addressing missing values and encoding categorical variables.
- Divide the dataset into training and testing sets to effectively assess model performance.
- Normalize or standardize the features, particularly for SVM, to ensure uniform scaling across inputs.
- Select the appropriate model: use SVM for margin-based classification or Decision Tree for rule-based classification tasks.
- Train the model on the training data using the fit method.
- Generate predictions on the testing data with the predict method.
- Evaluate the model's performance using metrics such as accuracy, confusion matrix, precision, and recall.
- Visualize the outcomes with relevant plots, like decision boundaries for SVM or tree structures for Decision Trees.
- Tune the model by adjusting hyperparameters, such as C for SVM or max_depth for Decision Trees.

PROGRAM:

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
```

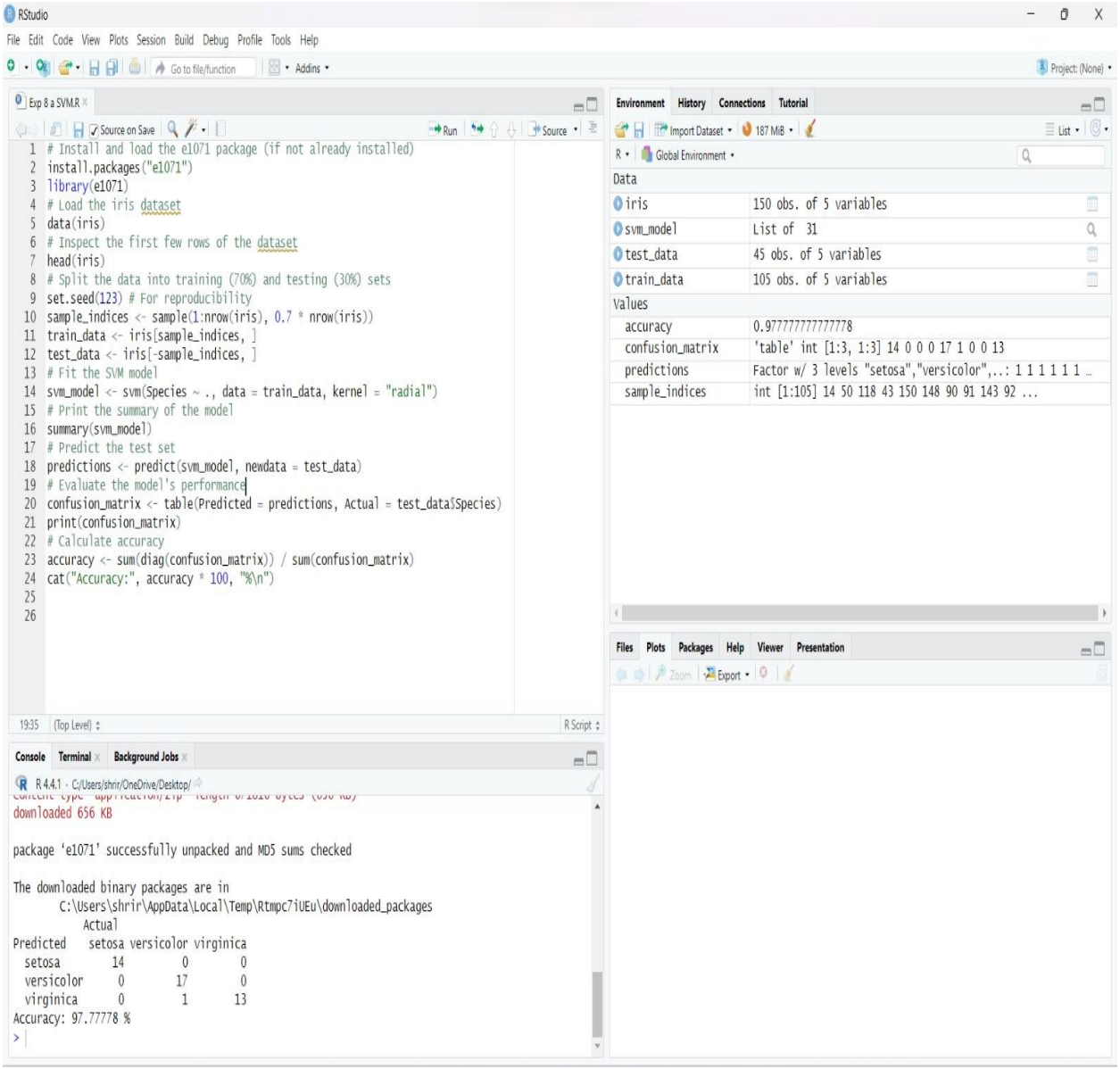
```
svm_model <- svm(Species ~ ., data = train_data, kernel = "radial")
# Print the summary of the model
summary(svm_model)
# Predict the test set
predictions <- predict(svm_model, newdata = test_data)
# Evaluate the model's performance
confusion_matrix <- table(Predicted = predictions, Actual = test_data$Species)
print(confusion_matrix)
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy * 100, "%\n")
```

Decision Tree.R:

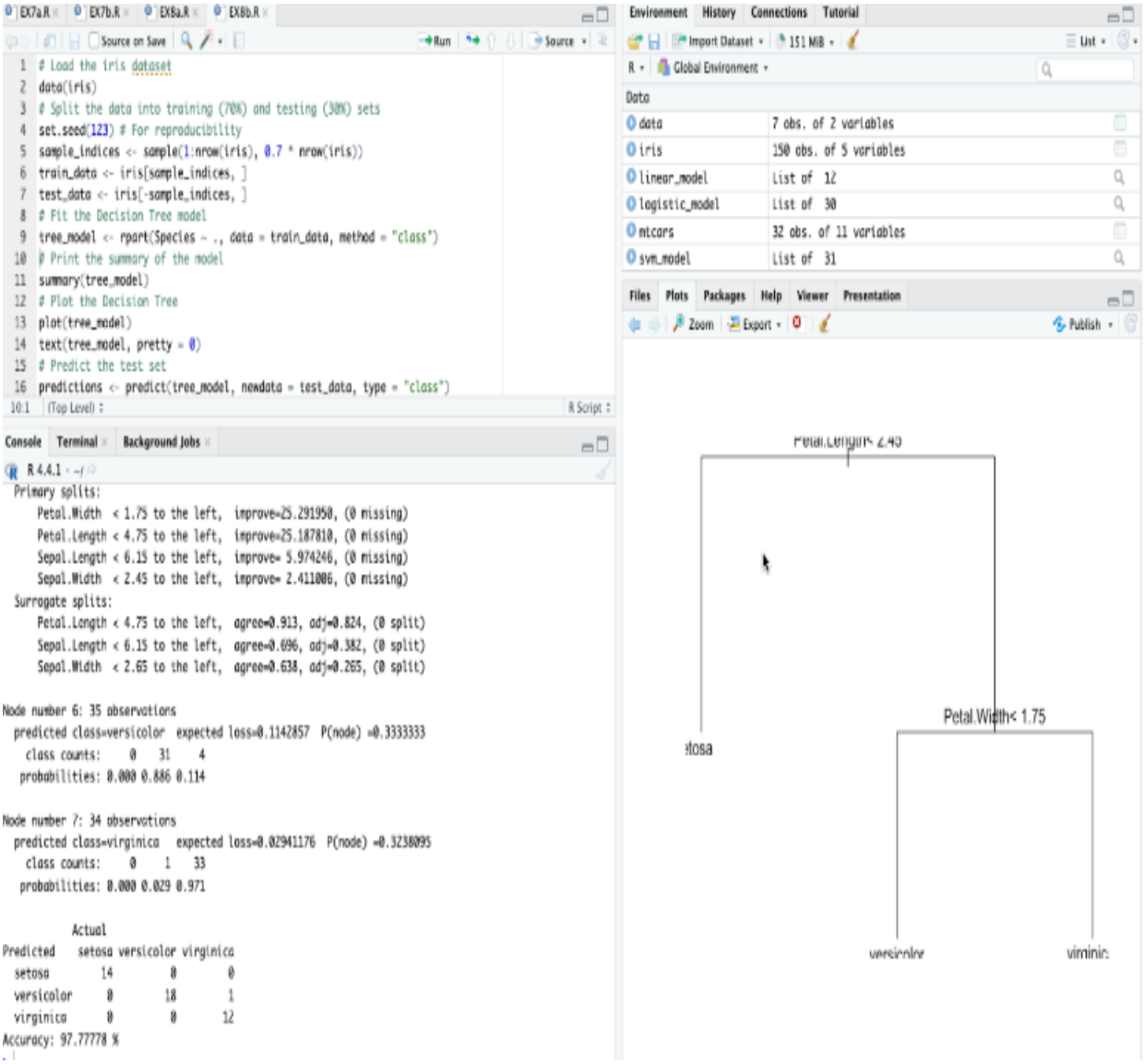
```
# Install and load the rpart package (if not already installed)
install.packages("rpart")
library(rpart)
# Load the iris dataset
data(iris)
# Split the data into training (70%) and testing (30%) sets
set.seed(123) # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data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


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Exp 8 a SVM.R
Source on Save
Run
Source
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)
22 # Calculate accuracy
23 accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
24 cat("Accuracy:", accuracy * 100, "%\n")
25
26
19:35 (Top Level)
R Script
Environment History Connections Tutorial
Import Dataset
187 MiB
List
R
Global Environment
Data
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
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 ...
Files Plots Packages Help Viewer Presentation
Zoom
Export
Console Terminal Background Jobs
R 4.4.1 · C:/Users/shrir/OneDrive/Desktop/
downloaded 656 KB
package 'e1071' successfully unpacked and MD5 sums checked
The downloaded binary packages are in
C:\Users\shrir\AppData\Local\Temp\Rtmpc7iUEu\downloaded_packages
Actual
Predicted setosa versicolor virginica
setosa 14 0 0
versicolor 0 17 0
virginica 0 1 13
Accuracy: 97.77778 %
>

Decision tree:



```
# 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")
```

```
Primary splits:
    Petal.Width  < 1.75 to the left,  improve=25.291950, (0 missing)
    Petal.Length < 4.75 to the left,  improve=25.187810, (0 missing)
    Sepal.Length < 6.15 to the left,  improve= 5.974246, (0 missing)
    Sepal.Width  < 2.45 to the left,  improve= 2.411086, (0 missing)
Surrogate splits:
    Petal.Length < 4.75 to the left,  agree=0.913, adj=0.824, (0 split)
    Sepal.Length < 6.15 to the left,  agree=0.696, adj=0.382, (0 split)
    Sepal.Width  < 2.65 to the left,  agree=0.638, adj=0.265, (0 split)

Node number 6: 35 observations
  predicted class=versicolor  expected loss=0.1142857  P(node) =0.3333333
    class counts:     0    31     4
   probabilities: 0.000 0.886 0.114

Node number 7: 34 observations
  predicted class=virginica   expected loss=0.02941176  P(node) =0.3238095
    class counts:     0     1    33
   probabilities: 0.000 0.029 0.971

          Actual
Predicted    setosa versicolor virginica
  setosa         14          0         0
  versicolor      0         18         1
  virginica       0          0        12
Accuracy: 97.77778 %
```


RESULT:

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