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.

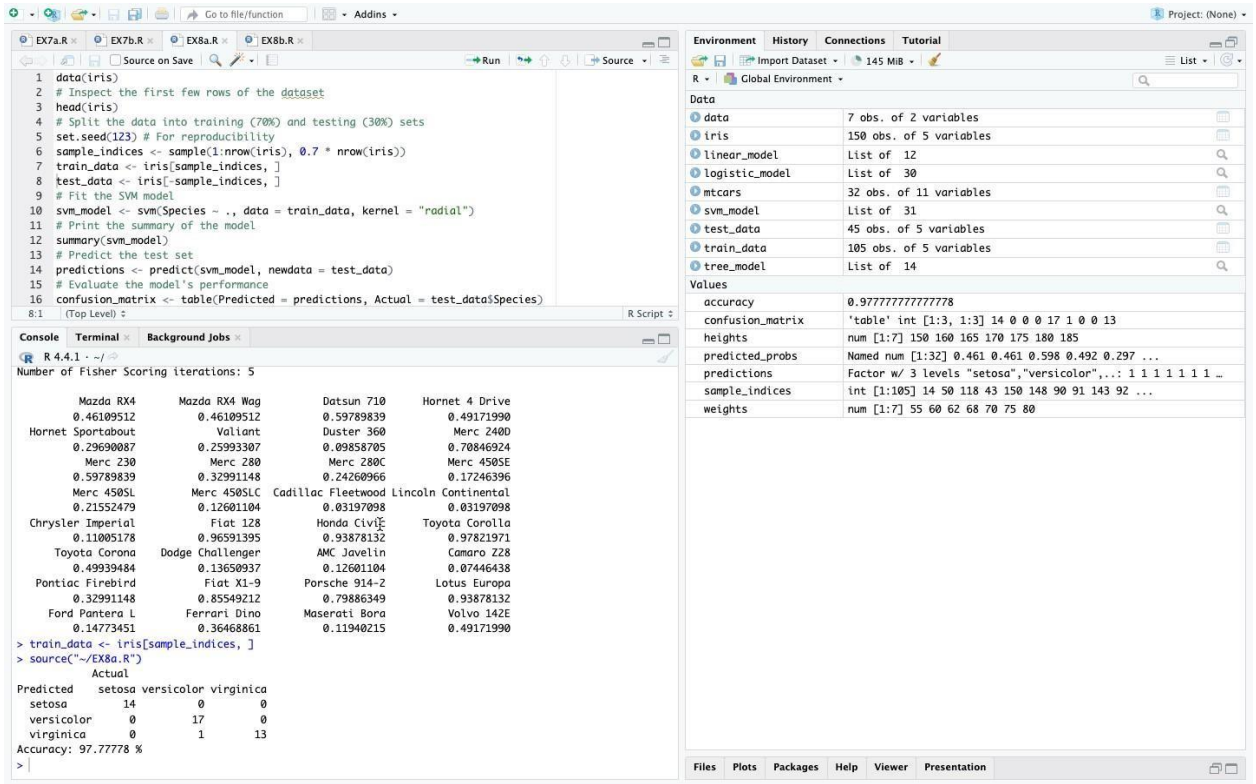
CODE:

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:**

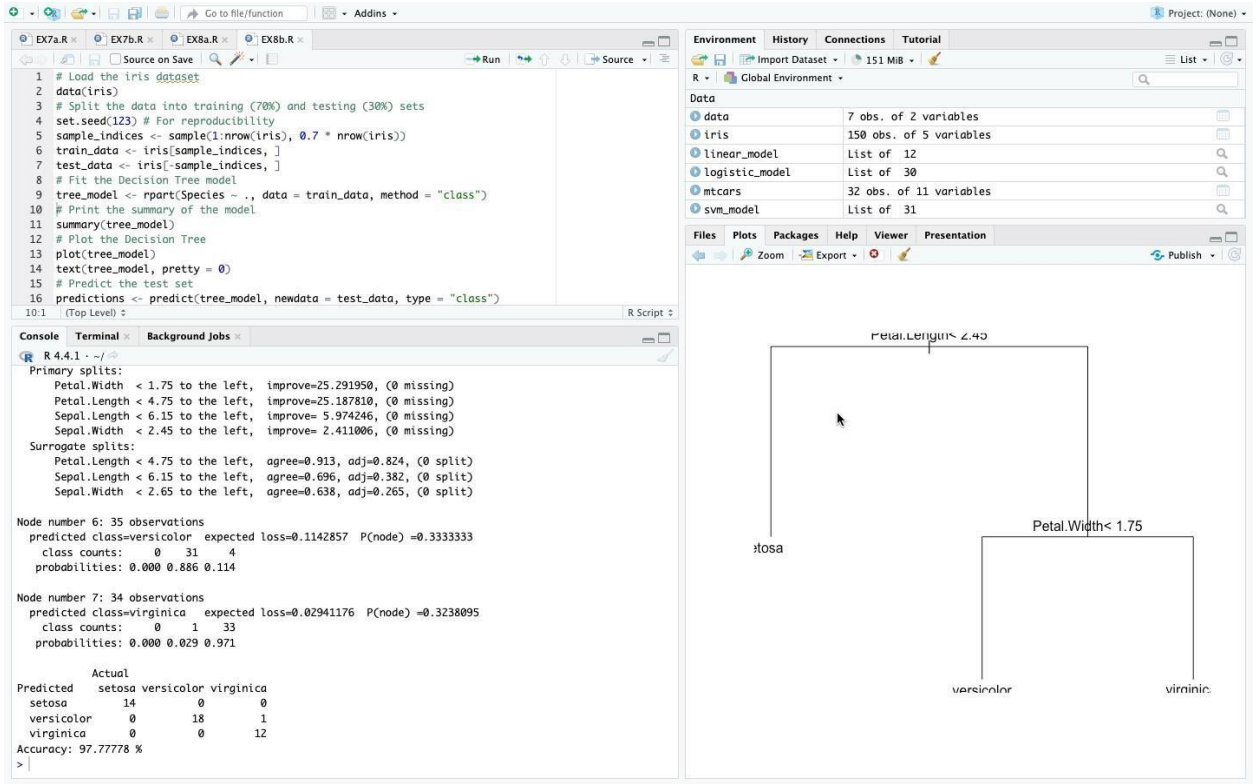


```
data(iris)
# Inspect the first few rows of the dataset
head(iris)
# Split the data into training (70%) and testing (30%) sets
set.seed(123) # For reproducibility
sample_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_indices, ]
test_data <- iris[-sample_indices, ]
# Fit 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)
```

```
Number of Fisher Scoring iterations: 5

          Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive
         0.46109512          0.46109512          0.59789839          0.49171990
  Hornet Sportabout             Valiant          Duster 360           Merc 240D
         0.29690087          0.25993307          0.09858705          0.70846924
           Merc 230            Merc 280           Merc 280C           Merc 450SE
         0.59789839          0.32991148          0.24260966          0.17246396
          Merc 450SL         Merc 450SLC  Cadillac Fleetwood Lincoln Continental
         0.21552479          0.12601104          0.03197098          0.03197098
  Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla
         0.11005178          0.96591395          0.93878132          0.97821971
       Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28
         0.49939484          0.13650937          0.12601104          0.07446438
   Pontiac Firebird           Fiat X1-9       Porsche 914-2        Lotus Europa
         0.32991148          0.85549212          0.79886349          0.93878132
      Ford Pantera L        Ferrari Dino       Maserati Bora          Volvo 142E
         0.14773451          0.36468861          0.11940215          0.49171990
> train_data <- iris[sample_indices, ]
> source("~/EX8a.R")
            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= 2.411006, (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, Implement SVM and Decision tree classification techniques has been successfully executed.