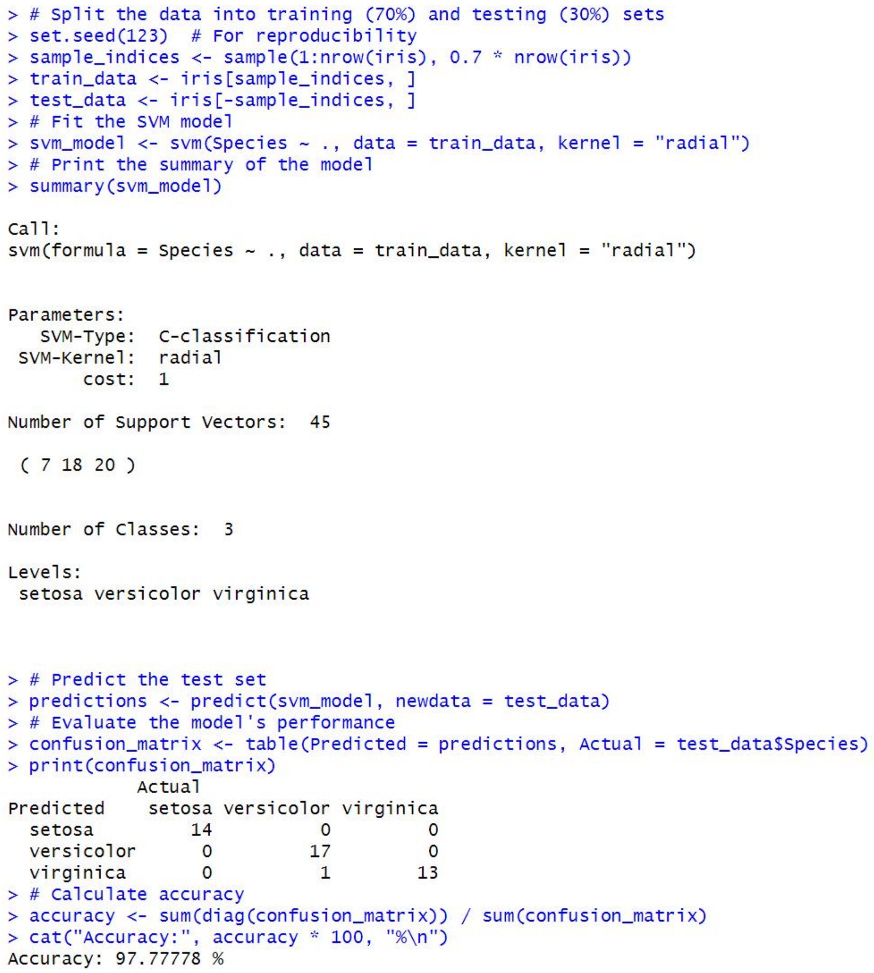
Implement SVM and Decision Tree Classification Techniques

SVM:

Code:

```
# 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")
```

Output:



```
> # 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)

Call:
svm(formula = Species ~ ., data = train_data, kernel = "radial")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  45

 ( 7 18 20 )


Number of Classes:  3

Levels:
 setosa versicolor virginica



> # 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)
            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 %
```

Decision Tree:

Code:

```
# 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)

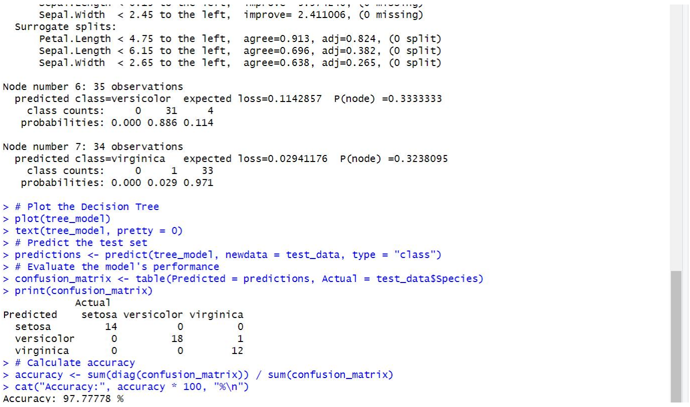
print(confusion_matrix)

Calculate accuracy

accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)

cat("Accuracy:", accuracy * 100, "%\n")

Output:

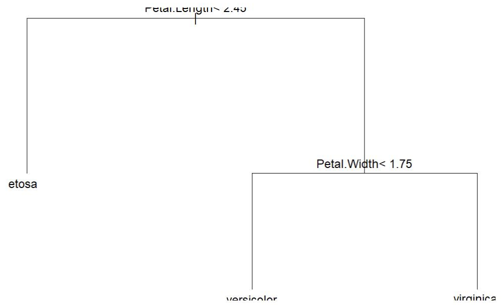


```
      Sepal.Width  < 2.45 to the left,  improve= 2.411006, (0 missing)
  Surrogate splits:
      Petal.Length < 4.75 to the left,  agree=0.913, adj=0.824, (0 split)
      Sepal.Length < 6.15 to the left,  agree=0.696, adj=0.382, (0 split)
      Sepal.Width  < 2.65 to the left,  agree=0.638, adj=0.265, (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

> # 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)
            Actual
Predicted    setosa versicolor virginica
  setosa         14          0         0
  versicolor      0         18         1
  virginica       0          0        12
> # Calculate accuracy
> accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
> cat("Accuracy:", accuracy * 100, "%\n")
Accuracy: 97.77778 %
```



Petal.Length< 2.45
etosa
Petal.Width< 1.75
versicolor
virginica