Rakshita jadav
1nt23is407

PROGRAM 4

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score
import numpy as np

# Load the dataset
df = pd.read_csv("zoo.csv")

# Preprocess the data
X = df.drop(columns=['animal_name', 'class_type'])
y = df['class_type']

# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and train the Decision Tree model
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)

# Generate confusion matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)

# Calculate accuracy, precision, and recall
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average=None)
recall = recall_score(y_test, y_pred, average=None)

# Print results
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision per class: {precision}")
print(f"Recall per class: {recall}")

# Class-wise precision and recall
classes = np.unique(y)
for i, label in enumerate(classes):
    print(f"Class {label} - Precision: {precision[i]:.2f}, Recall: {recall[i]:.2f}")
```

```
Confusion Matrix:
 [[12  0  0  0  0  0  0]
 [ 0  2  0  0  0  0  0]
 [ 0  0  0  0  1  0  0]
 [ 0  0  0  2  0  0  0]
 [ 0  0  0  0  0  0  0]
 [ 0  0  0  0  0  3  0]
 [ 0  0  0  0  0  0  1]]
Accuracy: 0.95
Precision per class: [1. 1. 0. 1. 0. 1. 1.]
Recall per class: [1. 1. 0. 1. 0. 1. 1.]
Class 1 - Precision: 1.00, Recall: 1.00
Class 2 - Precision: 1.00, Recall: 1.00
Class 3 - Precision: 0.00, Recall: 0.00
Class 4 - Precision: 1.00, Recall: 1.00
Class 5 - Precision: 0.00, Recall: 0.00
Class 6 - Precision: 1.00, Recall: 1.00
Class 7 - Precision: 1.00, Recall: 1.00
/home/admn/jpyter/notebookenv/lib/python3.12/site-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is il
l-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/home/admn/jpyter/notebookenv/lib/python3.12/site-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Recall is ill-d
efined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```