```python
# Step 1: Import necessary libraries
import pandas as pd
import numpy as np
from google.colab import files
import io
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Step 2: Upload and load dataset
uploaded = files.upload()
file_name = list(uploaded.keys())[0]
df = pd.read_excel(io.BytesIO(uploaded[file_name]), sheet_name='autism')

# Step 3: Clean the dataset (Remove missing & incorrect values)
df = df[(df['ethnicity'] != '?') & (df['relation'] != '?')]  # Remove rows with "?"
df = df.dropna(subset=['age'])  # Remove missing ages
df = df[df['age'] <= 80]  # Remove extreme age values

# 🔹 Step 4: Convert all columns to strings first (prevent unexpected errors)
df = df.astype(str)

# Step 5: Encode categorical variables
categorical_columns = ['gender', 'ethnicity', 'jundice', 'austim', 'contry_of_res', 'used
label_encoders = {}
for col in categorical_columns:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    label_encoders[col] = le  # Store encoders for reference

# Step 6: Convert numeric columns explicitly
numeric_columns = ['age'] + [col for col in df.columns if 'Score' in col]  # Identify num
for col in numeric_columns:
    df[col] = pd.to_numeric(df[col], errors='coerce')  # Convert to numeric

# Step 7: Define features (X) and target variable (y)
X = df.drop(columns=['Class/ASD'], errors='ignore')  # Features
y = df['Class/ASD']  # Target variable

# 🔹 Ensure all columns in X are numeric
X = X.apply(pd.to_numeric, errors='coerce')  # Convert any remaining strings to numbers

# 🔹 Step 8: Handle missing values by replacing NaN with column median
X = X.fillna(X.median())  # Replace NaN in X with median values
y = pd.to_numeric(y, errors='coerce').fillna(0).astype(int)  # Convert y to numbers & fil

# Step 9: Shuffle data
df = df.sample(frac=1, random_state=42).reset_index(drop=True)  # Shuffle data

# Step 10: Introduce noise (Prevent overfitting)
X += np.random.normal(0, 0.1, X.shape)  # Add small noise

# Step 11: Label Smoothing (Make labels slightly noisy)
flip_labels = np.random.rand(len(y)) < 0.1  # 10% chance of flipping labels
y = np.where(flip_labels, 1 - y, y)  # Flip labels randomly

# Step 12: Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=42,
# Step 13: Train a Decision Tree with lower complexity
dt_model = DecisionTreeClassifier(
    criterion='entropy',
    max_depth=5,  # Reduce depth for generalization
    min_samples_split=15,  # Prevent overfitting
    random_state=42
)
dt_model.fit(X_train, y_train)

# Step 14: Make predictions
y_pred = dt_model.predict(X_test)

# Step 15: Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

# Step 16: Display results
print(f"\nDecision Tree Model Accuracy: {accuracy:.2f}")
print("\nClassification Report:\n", classification_rep)
print("\nConfusion Matrix:\n", conf_matrix)

# Step 17: Feature Importance (Optional)
importances = pd.Series(dt_model.feature_importances_, index=X.columns)
print("\nFeature Importance:\n", importances.sort_values(ascending=False))
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving autism.xlsx to autism (11).xlsx

Decision Tree Model Accuracy: 0.87

Classification Report:
               precision    recall  f1-score   support

           0       0.90      0.96      0.93       218
           1       0.31      0.15      0.21        26

    accuracy                           0.87       244
   macro avg       0.61      0.56      0.57       244
weighted avg       0.84      0.87      0.85       244


Confusion Matrix:
 [[209   9]
 [ 22   4]]

Feature Importance:
 relation          0.286583
A7_Score          0.121337
A2_Score          0.116931
A3_Score          0.116441
gender            0.101710
jundice           0.097004
A4_Score          0.081570
age               0.078423
result            0.000000
used_app_before   0.000000
contry_of_res     0.000000
austim            0.000000
A1_Score          0.000000
ethnicity         0.000000
A9_Score          0.000000
A8_Score          0.000000
```