Assignment: Decision Tree Classification on PlayTennis Dataset

Aim

To implement a Decision Tree classifier (ID3 algorithm with entropy) on the PlayTennis dataset, visualize the decision tree, predict on a test sample, and analyze the results.

Lab Setup

- Software: Python 3.x environment (Google Colab, Jupyter Notebook, or local IDE)
- Required libraries:
  - pandas
  - scikit-learn
  - matplotlib
- Installation command (if needed):

  ```
  bash
  pip install pandas scikit-learn matplotlib
  ```
-

Input

- The PlayTennis dataset with the following features and target:
  - Features: Outlook, Temperature, Humidity, Wind
  - Target: PlayTennis (Yes or No)
- Test sample with attributes: Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong

Expected Output

- Decision tree visualization showing splits based on entropy.
- Prediction of PlayTennis decision for the test sample.
- Encoded categorical data used internally.

Example output:

```
text
Prediction for sample [Sunny, Cool, High, Strong]: ['No']
```

Theory / Algorithm

Decision Trees are supervised learning algorithms used for classification and regression. The ID3 algorithm builds the tree by selecting attributes that provide the highest information gain (based on entropy), recursively splitting the dataset until all data points are classified or stopping criteria are met.

Algorithm steps:

1. Calculate entropy of the target.
2. For each attribute, calculate information gain by splitting data.
3. Select attribute with highest gain to split the node.
4. Repeat recursively for each branch until pure subsets or other stopping conditions.
5. Use the tree to classify new data points by traversing splits.

Code

python

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier, plot_tree
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
# PlayTennis dataset
data = {
    'Outlook':    ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain', 'Overcast', 'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast', 'Overcast', 'Rain'],
    'Temperature':['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild', 'Cool', 'Mild', 'Mild', 'Mild', 'Hot', 'Mild'],
    'Humidity':   ['High', 'High', 'High', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'Normal', 'Normal', 'High', 'Normal', 'High'],
    'Wind':       ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong', 'Weak', 'Strong'],
    'PlayTennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No']
}
# Create DataFrame
df = pd.DataFrame(data)
# Encode categorical features and target
label_encoders = {}
for column in ['Outlook', 'Temperature', 'Humidity', 'Wind', 'PlayTennis']:
    le = LabelEncoder()
    df[column + '_enc'] = le.fit_transform(df[column])
    label_encoders[column] = le
# Features and target
feature_cols = ['Outlook_enc', 'Temperature_enc', 'Humidity_enc', 'Wind_enc']
X = df[feature_cols].values
y = df['PlayTennis_enc'].values
# Train Decision Tree (ID3 with entropy)
clf = DecisionTreeClassifier(criterion='entropy', random_state=42)
clf.fit(X, y)
# Visualize the tree
plt.figure(figsize=(12,8))
plot_tree(clf,
          feature_names=['Outlook', 'Temperature', 'Humidity', 'Wind'],
          class_names=label_encoders['PlayTennis'].classes_,
          filled=True)
plt.title("ID3 Decision Tree - PlayTennis Dataset")
plt.show()
# Test sample: Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong
test_sample = [
    label_encoders['Outlook'].transform(['Sunny'])[0],
    label_encoders['Temperature'].transform(['Cool'])[0],
    label_encoders['Humidity'].transform(['High'])[0],
    label_encoders['Wind'].transform(['Strong'])[0]
]
pred = clf.predict([test_sample])
print("Prediction for sample [Sunny, Cool, High, Strong]:", label_encoders['PlayTennis'].inverse_transform(pred))
```

Conclusion / Discussion

The Decision Tree classifier using the ID3 algorithm successfully learned rules to predict whether tennis will be played based on weather features. The decision tree visualization reveals how features like Outlook and Humidity split the data. The classifier correctly predicts the test sample with attributes (Sunny, Cool, High, Strong) as 'No', demonstrating the model’s applicability.

Actual Output

text

```
Prediction for sample [Sunny, Cool, High, Strong]: ['No']
```

Graph / Analysis

The plotted tree clearly shows nodes splitting on feature values with class distributions and entropy. This visual aids in understanding the classification logic and importance of different features. The tree confirms the model's decision boundaries to classify input conditions.