▾ CE066-ML-LAB4-Jaydeep Mahajan

Task 1: Try the algo on Same Whether dataset- LabelEncoding of features: and Train test Division 90%-10%

```python
from sklearn.tree import DecisionTreeClassifier
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn import metrics


#Predictor variables
Outlook = ['Rainy', 'Rainy', 'Overcast', 'Sunny', 'Sunny', 'Sunny', 'Overcast',
           'Rainy', 'Rainy', 'Sunny', 'Rainy','Overcast', 'Overcast', 'Sunny']
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 = ['False', 'True', 'False', 'False', 'False', 'True', 'True',
        'False', 'False', 'False', 'True', 'True', 'False', 'True']

#Class Label:
Play = ['No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No',
'Yes', 'Yes', 'Yes', 'Yes', 'Yes', 'No']


#creating labelEncoder
le = preprocessing.LabelEncoder()

# Converting string labels into numbers.
Outlook_encoded = le.fit_transform(Outlook)
Outlook_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Outllok mapping:",Outlook_name_mapping)

Temperature_encoded = le.fit_transform(Temperature)
Temperature_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Temperature mapping:",Temperature_name_mapping)

Humidity_encoded = le.fit_transform(Humidity)
Humidity_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Humidity mapping:",Humidity_name_mapping)

Wind_encoded = le.fit_transform(Wind)
Wind_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Wind mapping:",Wind_name_mapping)

Play_encoded = le.fit_transform(Play)
Play_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
print("Play mapping:",Play_name_mapping)

print("\n\n")
print("Weather:"  Outlook encoded)
```

```
print( weather:  ,OutLook_encoded)
print("Temerature:" ,Temperature_encoded)
print("Humidity:" ,Humidity_encoded)
print("Wind:" ,Wind_encoded)
print("Play:" ,Play_encoded)
```

```
↳   Outllok mapping: {'Overcast': 0, 'Rainy': 1, 'Sunny': 2}
     Temperature mapping: {'Cool': 0, 'Hot': 1, 'Mild': 2}
     Humidity mapping: {'High': 0, 'Normal': 1}
     Wind mapping: {'False': 0, 'True': 1}
     Play mapping: {'No': 0, 'Yes': 1}


     Weather: [1 1 0 2 2 2 0 1 1 2 1 0 0 2]
     Temerature: [1 1 1 2 0 0 0 2 0 2 2 2 1 2]
     Humidity: [0 0 0 0 1 1 1 0 1 1 1 0 1 0]
     Wind: [0 1 0 0 0 1 1 0 0 0 1 1 0 1]
     Play: [0 0 1 1 1 0 1 0 1 1 1 1 1 0]
```

```
features = tuple(zip(Outlook_encoded ,Temperature_encoded ,Humidity_encoded ,Wind_encoded)
print("Features : ",features)
```

```
↳   Features :  ((1, 1, 0, 0), (1, 1, 0, 1), (0, 1, 0, 0), (2, 2, 0, 0), (2, 0, 1, 0), (2
```

```
x_train, x_test, y_train, y_test = train_test_split(features, Play_encoded, test_size=0.1,
```

```
clf = DecisionTreeClassifier(criterion="entropy")
clf.fit(x_train, y_train)
y_pred = clf.predict(x_test)
print("Accuracy: ",metrics.accuracy_score(y_test, y_pred))
```

```
↳   Accuracy:  0.5
```

```
y_pred_example = clf.predict([[1,2,1,0],[2,0,0,1]])
print("y predicted : ",y_pred_example)
```

```
↳   y predicted :  [0 0]
```

```
#create confusion matrix
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred)
```

```
↳   array([[0, 0],
            [1, 1]])
```

```
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
precision = precision_score(y_test,y_pred)
recall = recall_score(y_test,y_pred)
print('precision: {}'.format(precision))
print('recall: {}'.format(recall))
```

```
precision: 1.0
recall: 0.5
```

```python
from sklearn.tree import export_graphviz
export_graphviz(clf,out_file='tree_entropy.dot',
                feature_names=['outlook','temperature','humidity','wind'],
                class_names=['play_no','play_yes'],
                filled=True)

#Convert to png
from subprocess import call
call(['dot', '-Tpng', 'tree_entropy.dot', '-o', 'tree_entropy.png', '-Gdpi=600'])

#Display in python
import matplotlib.pyplot as plt
plt.figure(figsize = (9, 12))
plt.imshow(plt.imread('tree_entropy.png'))
plt.axis('off');
plt.show();
```

outlook <= 0.5

value = [5, 7]
class = play_yes

True

False

entropy = 0.0
samples = 3
value = [0, 3]
class = play_yes

wind <= 0.5
entropy = 0.991
samples = 9
value = [5, 4]
class = play_no

outlook <= 1.5
entropy = 0.918
samples = 6
value = [2, 4]
class = play_yes

entropy = 0.0
samples = 3
value = [3, 0]
class = play_no

temperature <= 0.5
entropy = 0.918
samples = 3

entropy = 0.0
samples = 3