19MCEC08_DecisionTree

1 Decision Tree

[7]:
```
import pandas as  pd
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics

w=pd.read_csv('slr.csv')
X=w.iloc[:,:-1]
y=w.iloc[:,-1:]
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
labelencoder_Y=LabelEncoder()
y=labelencoder_Y.fit_transform(y)
onehotencoder=OneHotEncoder(categorical_features='all')
X=onehotencoder.fit_transform(X).toarray()

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=1)
tree=DecisionTreeClassifier(criterion='entropy',max_depth=2)
tree.fit(X,y)
y_pred=tree.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test,y_pred))

from sklearn.metrics import␣
 ↪classification_report,confusion_matrix,accuracy_score,recall_score,precision_score,f1_score
print("\nConfusion Matrix:")
print(confusion_matrix(y_test,y_pred))
print("\nClassification report:")
print(classification_report(y_test,y_pred))
print("\nAccuracy Score:{0}".format(accuracy_score(y_pred,y_test)))
print("\nPrecision Score:{0}".
 ↪format(precision_score(y_pred,y_test,average=None)))
print("\nRecall Score:{0}".format(recall_score(y_pred,y_test,average=None)))
print("F1 score:{0}".format(f1_score(y_pred,y_test,average=None)))
```

```
Accuracy: 0.6666666666666666
```

```
Confusion Matrix:
[[1 0]
 [1 1]]

Classification report:
              precision    recall  f1-score   support

           0       0.50      1.00      0.67         1
           1       1.00      0.50      0.67         2

    accuracy                           0.67         3
   macro avg       0.75      0.75      0.67         3
weighted avg       0.83      0.67      0.67         3


Accuracy Score:0.6666666666666666

Precision Score:[1.  0.5]

Recall Score:[0.5 1. ]
F1 score:[0.66666667 0.66666667]

D:\Users\kshitij\Anaconda3\lib\site-packages\sklearn\preprocessing\label.py:235:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
D:\Users\kshitij\Anaconda3\lib\site-
packages\sklearn\preprocessing\_encoders.py:441: DeprecationWarning: The
'categorical_features' keyword is deprecated in version 0.20 and will be removed
in 0.22. The passed value of 'all' is the default and can simply be removed.
  DeprecationWarning)
D:\Users\kshitij\Anaconda3\lib\site-
packages\sklearn\preprocessing\_encoders.py:441: DeprecationWarning: The
'categorical_features' keyword is deprecated in version 0.20 and will be removed
in 0.22. The passed value of 'all' is the default and can simply be removed.
  DeprecationWarning)
```

Analysis:- The decision tree algorithm forms a node, where the most important attribute is placed at the root node. It starts at the root node and work our way down the tree by following the corresponding node that meets our condition.This process continues until a leaf node is reached. Here we used tennis dataset and used decision tree classifier.

2 Grid Search in Decision Tree

[14]:
```
from sklearn.model_selection import GridSearchCV
grid_values = {'criterion': ['gini', 'entropy'],'max_depth':[1,2,3,4,5]}
grid_clf_acc = GridSearchCV(tree,param_grid = grid_values,cv = 4,scoring =␣
 ↪'accuracy')
grid_clf_acc.fit(X_train, y_train)

#Best Score with Grid Search
print("Best score is",grid_clf_acc.best_score_)

#Best params with Grid Search
print("Best parameters are",grid_clf_acc.best_params_)

dtc=DecisionTreeClassifier(criterion='gini',max_depth=3)
dtc.fit(X,y)
y_pred_dtc = dtc.predict(X_test)

print("\nConfusion Matrix : ")
print(confusion_matrix(y_test,y_pred_dtc))
print("\nClassification Report : ")
print(classification_report(y_test,y_pred_dtc))
print("\nAccuracy Score : {0}".format(accuracy_score(y_pred_dtc,y_test)))
print("Recall Score : {0}".format(recall_score(y_pred_dtc,y_test,average=None)))
print("Precision Score : {0}".
 ↪format(precision_score(y_pred_dtc,y_test,average=None)))
print("F1 Score : {0}".format(f1_score(y_pred_dtc,y_test,average=None)))
```

```
Best score is 0.7272727272727273
Best parameters are {'criterion': 'entropy', 'max_depth': 2}

Confusion Matrix :
[[1 0]
 [1 1]]

Classification Report :
              precision    recall  f1-score   support

           0       0.50      1.00      0.67         1
           1       1.00      0.50      0.67         2

    accuracy                           0.67         3
   macro avg       0.75      0.75      0.67         3
weighted avg       0.83      0.67      0.67         3


Accuracy Score : 0.6666666666666666
```

```
Recall Score : [0.5 1. ]
Precision Score : [1.  0.5]
F1 Score : [0.66666667 0.66666667]

D:\Users\kshitij\Anaconda3\lib\site-
packages\sklearn\model_selection\_search.py:813: DeprecationWarning: The default
of the `iid` parameter will change from True to False in version 0.22 and will
be removed in 0.24. This will change numeric results when test-set sizes are
unequal.
  DeprecationWarning)
```

[12]:
```
#Checking for worst parameters
combinations=[]
accuracies=[]
criterion = ['gini', 'entropy']
max_depth = [1,2,3,4,5]
for i in criterion:
    for j in max_depth:
        tree = DecisionTreeClassifier(criterion=i,max_depth=j)
        tree.fit(X_train, y_train)
        y_pred_train = tree.predict(X_train)
        combinations.append([i,j,accuracy_score(y_train,y_pred_train)])
        accuracies.append(accuracy_score(y_train,y_pred_train))
print("\n\n")

print("Criterion \t \t Max Depth \t \t Accuracy \n")
for i in combinations:
    print("   {0} \t \t {1} \t \t {2} ".format(i[0],i[1],i[2]))

print("\n\n")

print("Lowest accuracy: {0}".format(min(accuracies)))

print("\n")

index=-1

for i in range(0,len(accuracies)):
    if accuracies[i]==min(accuracies):
        index=i
print("Worst Parameters : \nCriterion : {0} , Max Depth : {1} ".
 ↪format(combinations[index][0],combinations[index][1]))
```

```
Criterion                Max Depth                Accuracy
```

```
gini                1               0.6363636363636364
gini                2               0.9090909090909091
gini                3               0.9090909090909091
gini                4               1.0
gini                5               1.0
entropy             1               0.6363636363636364
entropy             2               0.9090909090909091
entropy             3               0.9090909090909091
entropy             4               1.0
entropy             5               1.0
```

```
Lowest accuracy: 0.6363636363636364
```

```
Worst Parameters :
Criterion : entropy , Max Depth : 1
```

Analysis:- Using Grid SearchCV we try to find appropriate hyper parameters that would provide us with optimal result.Here we took criterion and max depth as two parameters.The set of keys in the dictionary are gone through in the GridSearchCV() process and gives us the best score and parameters.Best and worst case both are displayed.