DEVYANI BEOHAR

J068

In [5]:

EXP 9 Decision Tree with Cross Validation and GridSearchCV

```
In [1]:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import os
print(os.getcwd())
%matplotlib inline
C:\Users\USER
In [2]:
df=pd.read csv("C:\\Users\\USER\\Desktop\\Data Science\\SEM 5\\ML\\car evaluation.csv",h
eader=None)
df.head()
Out[2]:
      0
           1 2 3
                     4
                         5
                               6
0 vhigh vhigh 2 2 small low unacc
1 vhigh vhigh 2 2 small med unacc
2 vhigh vhigh 2 2 small high unacc
3 vhigh vhigh 2 2 med
                       low unacc
4 vhigh vhigh 2 2 med med unacc
In [3]:
col names=['buying','maint','doors','persons','lug boot','safety','class']
df.columns=col names
col names
Out[3]:
['buying', 'maint', 'doors', 'persons', 'lug boot', 'safety', 'class']
In [4]:
df.head()
Out[4]:
  buying maint doors persons lug_boot safety
                                          class
0
   vhigh vhigh
                  2
                         2
                              small
                                     low unacc
        vhigh
                  2
                         2
   vhigh
                              small
                                     med unacc
   vhigh vhigh
                  2
                         2
                              small
                                     high unacc
2
    vhigh vhigh
                  2
                         2
                               med
                                     low unacc
                  2
                         2
   vhigh vhigh
                               med
                                     med unacc
```

for i in col names:

```
print(df[i].value_counts())
         432
vhigh
high
         432
med
         432
low
         432
Name: buying, dtype: int64
vhigh 432
        432
high
med
        432
low
        432
Name: maint, dtype: int64
5more 432
3
        432
2
        432
        432
Name: doors, dtype: int64
       576
more
        576
4
       576
Name: persons, dtype: int64
       576
big
        576
small
       576
med
Name: lug boot, dtype: int64
     576
high
        576
med
low
       576
Name: safety, dtype: int64
unacc 1210
acc
         384
         69
good
vgood
          65
Name: class, dtype: int64
In [6]:
df.shape
Out[6]:
(1728, 7)
In [7]:
X=df.drop(['class'],axis=1)
y=df['class']
In [8]:
from sklearn.model selection import train test split
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.3, random_state=42)
In [10]:
from sklearn.preprocessing import OrdinalEncoder
enc=OrdinalEncoder()
X train = enc.fit transform(X train)
X test = enc.transform((X test))
Gini index as criterion
In [11]:
```

```
from sklearn.tree import DecisionTreeClassifier
```

```
In [12]:
clf_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=42)
```

```
clf_gini.fit(X_train,y_train)
Out[12]:
DecisionTreeClassifier(max depth=3, random state=42)
In [13]:
y pred=clf gini.predict(X test)
In [14]:
from sklearn.metrics import accuracy score
print(f'Model with gini index gives an accuracy of: {accuracy score(y test, y pred)}')
Model with gini index gives an accuracy of: 0.7572254335260116
In [15]:
from sklearn import tree
plt.figure(figsize=(15,8))
tree.plot tree(clf gini, feature names=['buying', 'maint', 'doors', 'persons', 'lug boot', 'saf
ety'],
                 class names=list(set(y train)),filled=True)
plt.show()
                               persons <= 0.5
                                gini = 0.452
                              samples = 1209
                          value = [266, 50, 852, 41]
                              class = unacc
                                            safety \leq 0.5
                     gini = 0.0
                                            gini = 0.571
                  samples = 393
                                           samples = 816
                value = [0, 0, 393, 0]
                                      value = [266, 50, 459, 41]
                   class = unacc
                                            class = unacc
                   maint <= 2.5
                                                                     safety \leq 1.5
                                                                      gini = 0.42
                   gini = 0.627
                   samples = 273
                                                                    samples = 543
              value = [147, 21, 64, 41]
                                                                value = [119, 29, 395, 0]
                   class = vgood
                                                                     class = unacc
       gini = 0.613
                                gini = 0.498
                                                          gini = 0.0
                                                                                  gini = 0.59
      samples = 202
                                samples = 71
                                                        samples = 274
                                                                                samples = 269
  value = [114, 21, 26, 41]
                            value = [33, 0, 38, 0]
                                                     value = [0, 0, 274, 0]
                                                                            value = [119, 29, 121, 0]
       class = vgood
                               class = unacc
                                                        class = unacc
                                                                                 class = unacc
In [17]:
print(f'Training set score: {clf gini.score(X train, y train)}')
print(f'Test set score: {clf_gini.score(X_test,y_test)}')
Training set score: 0.7775020678246485
Test set score: 0.7572254335260116
```

Entropy as criterion

```
In [18]:
```

```
clf entropy=DecisionTreeClassifier(criterion='entropy', max depth=3, random state=42)
clf entropy.fit(X train, y train)
```

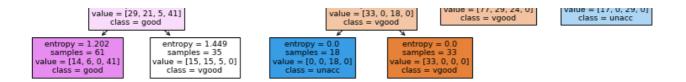
Out[18]:

DecisionTreeClassifier(criterion='entropy', max depth=3, random state=42)

```
In [19]:
y pred=clf entropy.predict(X test)
In [20]:
from sklearn.metrics import accuracy score
print(f'Model with gini index gives an accuracy of: {accuracy score(y test, y pred)}')
Model with gini index gives an accuracy of: 0.7572254335260116
In [21]:
plt.figure(figsize=(15,8))
tree.plot tree(clf entropy, feature names=['buying', 'maint', 'doors', 'persons', 'lug boot', '
safety'],
                class names=list(set(y train)), filled=True)
plt.show()
                              persons <= 0.5
                              entropy = 1.192
                              samples = 1209
                         value = [266, 50, 852, 41]
                              class = unacc
                                           safety \leq 0.5
                   entropy = 0.0
                                          entropy = 1.458
                  samples = 393
                                          samples = 816
               value = [0, 0, 393, 0]
                                     value = [266, 50, 459, 41]
                  class = unacc
                                           class = unacc
                   maint <= 2.5
                                                                   safety \leq 1.5
                  entropy = 1.667
                                                                   entropy = 1.04
                  samples = 273
                                                                   samples = 543
              value = [147, 21, 64, 41]
                                                              value = [119, 29, 395, 0]
                  class = vgood
                                                                   class = unacc
     entropy = 1.653
                              entropy = 0.996
                                                       entropy = 0.0
                                                                              entropy = 1.385
      samples = 202
                               samples = 71
                                                      samples = 274
                                                                              samples = 269
  value = [114, 21, 26, 41]
                           value = [33, 0, 38, 0]
                                                    value = [0, 0, 274, 0]
                                                                          value = [119, 29, 121, 0]
                               class = unacc
                                                       class = unacc
                                                                               class = unacc
      class = vgood
In [22]:
print(f'Training set score: {clf entropy.score(X train, y train)}')
print(f'Test set score: {clf entropy.score(X test, y test)}')
Training set score: 0.7775020678246485
Test set score: 0.7572254335260116
Grid Search CV
In [23]:
from sklearn.model selection import GridSearchCV
In [24]:
params={'criterion':['gini','entropy'],'max depth':list(range(3,7)),'min samples split':
         'min samples leaf':list(range(3,7)),'max leaf nodes':list(range(3,12))}
In [25]:
decision tree=DecisionTreeClassifier()
dt=GridSearchCV(decision tree,params,cv=10,scoring='accuracy')
```

dt.fit(X train, y train)

```
dt.best_score_
Out[25]:
0.8461088154269971
In [26]:
dt.best params
Out[26]:
{'criterion': 'entropy',
  'max depth': 6,
  'max_leaf_nodes': 11,
  'min_samples_leaf': 3,
 'min samples split': 3}
In [27]:
y_pred=dt.predict(X_test)
In [28]:
print(f'Model with Decision Tree gives an accuracy of: {accuracy score(y test, y pred)}')
Model with Decision Tree gives an accuracy of: 0.861271676300578
After GridSearchCV
In [29]:
dt=DecisionTreeClassifier(criterion='entropy', max depth=6, max leaf nodes=11, min samples l
eaf=3,
                                         min samples split=3, splitter='best')
In [30]:
dt.fit(X train,y train)
Out[30]:
DecisionTreeClassifier(criterion='entropy', max_depth=6, max_leaf_nodes=11,
                                    min samples leaf=3, min_samples_split=3)
In [31]:
plt.figure(figsize=(15,8))
tree.plot tree(dt,feature names=['buying','maint','doors','persons','lug boot','safety'],
                        class names=list(set(y train)), filled=True)
plt.show()
                                                           persons <= 0.5
entropy = 1.192
                                                       samples = 1209
value = [266, 50, 852, 41]
                                                                       safety <= 0.5
entropy = 1.458
                                                 entropy = 0.0
                                              samples = 393
value = [0, 0, 393, 0]
class = unacc
                                                                   samples = 816
value = [266, 50, 459, 41]
                                                                                                 safety <= 1.5
entropy = 1.04
samples = 543
value = [119, 29, 395, 0]
                                          maint <= 2.5
                                      entropy = 1.667
samples = 273
value = [147, 21, 64, 41]
                                          class = vgood
                              buving \leq = 0.5
                                                                                                                lua boot \leq 1.5
                                                     entropy = 0.996
                          entropy = 1.653
samples = 202
value = [114, 21, 26, 41]
                                                                                                            entropy = 1.385
samples = 269
value = [119, 29, 121, 0]
                                                   samples = 71
value = [33, 0, 38, 0]
                                                                                           mples = 274
= [0, 0, 274, 0]
                                                      class = unacc
                              class = vgood
                                                                                                  buying <= 2.5
entropy = 1.433
samples = 176
value = [94, 29, 53, 0]
                                          buying \leq 2.5
                                                                                                                            entropy = 0.84
                                      entropy = 1.859
samples = 147
value = [62, 21, 23, 41]
                                                                                                                             samples = 93
se = [25, 0, 68, 0]
                                                                                                                          value
                                                                                                         = vgood
                  lug_boot <= 1.5
entropy = 1.748
samples = 96
                                                                  maint \leq = 0.5
                                                                                                                entropy = 0.95
samples = 46
                                                                                         entropy = 1.38
```



Cross Validation

```
In [32]:
```

```
from sklearn.model_selection import cross_val_score
```

In [33]:

```
score=cross_val_score(dt,X_train,y_train,cv=10,scoring='accuracy')
score.mean()
```

Out[33]:

0.8461088154269971

In [34]:

```
print(f'Training set score: {dt.score(X_train,y_train)}')
print(f'Test set score: {dt.score(X_test,y_test)}')
```

Training set score: 0.858560794044665 Test set score: 0.861271676300578

In [35]:

```
from sklearn.metrics import confusion_matrix, classification_report
cm=confusion_matrix(y_test, y_pred)
```

In [36]:

```
print(cm)

[[ 78     0     32     8]
    [ 16     0     0     3]
    [ 13     0     345     0]
    [     0     0      24]]
```

In [37]:

```
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
acc good unacc vgood	0.73 0.00 0.92 0.69	0.66 0.00 0.96 1.00	0.69 0.00 0.94 0.81	118 19 358 24
accuracy macro avg weighted avg	0.58 0.83	0.66 0.86	0.86 0.61 0.84	519 519 519

C:\Users\USER\anaconda3\lib\site-packages\sklearn\metrics_classification.py:1221: Undefi
nedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels wi
th no predicted samples. Use `zero_division` parameter to control this behavior.
 _warn_prf(average, modifier, msg_start, len(result))