EXPERIMENT-3

AIM:

Make a decision tree classifier using sklearn.

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [ ]:
```
```

In [2]:
```
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [3]:
```
import warnings

warnings.filterwarnings('ignore')
```

In [4]:
```
data = 'car_evaluation.csv'

df = pd.read_csv(data, header=None)
```

In [5]:
```
df
```

Out[5]:

	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
...	...	...	...	...	...	...	...
1723	low	low	5more	more	med	med	good
1724	low	low	5more	more	med	high	vgood
1725	low	low	5more	more	big	low	unacc
1726	low	low	5more	more	big	med	good
1727	low	low	5more	more	big	high	vgood

1728 rows × 7 columns

In [6]:
```
df.shape
```

Out[6]:
```
(1728, 7)
```

In [7]:
```
# preview the dataset

df.head()
```

Out[7]:

	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 [8]:
```
col_names = ['buying', 'maint', 'num_doors', 'num_persons', 'boot_space', 'safet


df.columns = col_names

col_names
```

Out[8]:
```
['buying',
 'maint',
 'num_doors',
 'num_persons',
 'boot_space',
 'safety',
 'class']
```

In [9]:
```
df.head()
```

Out[9]:

	buying	maint	num_doors	num_persons	boot_space	safety	class
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 [10]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1728 entries, 0 to 1727
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   buying       1728 non-null   object
 1   maint        1728 non-null   object
 2   num_doors    1728 non-null   object
 3   num_persons  1728 non-null   object
 4   boot_space   1728 non-null   object
 5   safety       1728 non-null   object
 6   class        1728 non-null   object
dtypes: object(7)
memory usage: 94.6+ KB
```

In [11]:
```
col_names = ['buying', 'maint', 'num_doors', 'num_persons', 'boot_space', 'safet


for col in col_names:

    print(df[col].value_counts())
```

```
buying
vhigh    432
high     432
med      432
low      432
Name: count, dtype: int64
maint
vhigh    432
high     432
med      432
low      432
Name: count, dtype: int64
num_doors
2        432
3        432
4        432
5more    432
Name: count, dtype: int64
num_persons
2       576
4       576
more    576
Name: count, dtype: int64
boot_space
small    576
med      576
big      576
Name: count, dtype: int64
safety
low     576
med     576
high    576
Name: count, dtype: int64
class
unacc    1210
acc       384
good       69
vgood      65
Name: count, dtype: int64
```

In [12]:
```
df['class'].value_counts()
```

Out[12]:
```
class
unacc    1210
acc       384
good       69
vgood      65
Name: count, dtype: int64
```

In [13]:
```
df.isnull().sum()
```

Out[13]:
```
buying         0
maint          0
num_doors      0
num_persons    0
boot_space     0
safety         0
class          0
dtype: int64
```

In [14]:
```
X = df.drop(['class'], axis=1)

y = df['class']
```

In [15]:
```
pip install scikit-learn
```

```
Requirement already satisfied: scikit-learn in c:\users\vrajc\miniconda3\envs\py3
12\lib\site-packages (1.5.1)
Requirement already satisfied: numpy>=1.19.5 in c:\users\vrajc\miniconda3\envs\py
312\lib\site-packages (from scikit-learn) (2.0.1)
Requirement already satisfied: scipy>=1.6.0 in c:\users\vrajc\miniconda3\envs\py3
12\lib\site-packages (from scikit-learn) (1.14.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\vrajc\miniconda3\envs\py
312\lib\site-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\vrajc\miniconda3
\envs\py312\lib\site-packages (from scikit-learn) (3.5.0)
Note: you may need to restart the kernel to use updated packages.
```

In [16]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, rand
```

In [17]:
```
X_train.shape, X_test.shape
```

Out[17]:
```
((1157, 6), (571, 6))
```

In [18]:
```
X_train.dtypes
```

Out[18]:
```
buying         object
maint          object
num_doors      object
num_persons    object
boot_space     object
safety         object
dtype: object
```

In [19]:
```
X_train.head()
```

Out[19]:

	buying	maint	num_doors	num_persons	boot_space	safety
48	vhigh	vhigh	3	more	med	low
468	high	vhigh	3	4	small	low
155	vhigh	high	3	more	small	high
1721	low	low	5more	more	small	high
1208	med	low	2	more	small	high

In [20]:
```
pip install category_encoders
```

```
Requirement already satisfied: category_encoders in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (2.6.3)
Requirement already satisfied: numpy>=1.14.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (2.0.1)
Requirement already satisfied: scikit-learn>=0.20.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (1.5.1)
Requirement already satisfied: scipy>=1.0.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (1.14.0)
Requirement already satisfied: statsmodels>=0.9.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (0.14.2)
Requirement already satisfied: pandas>=1.0.5 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (2.2.2)
Requirement already satisfied: patsy>=0.5.1 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from category_encoders) (0.5.6)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from pandas>=1.0.5->category_encoders) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from pandas>=1.0.5->category_encoders) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from pandas>=1.0.5->category_encoders) (2024.1)
Requirement already satisfied: six in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from patsy>=0.5.1->category_encoders) (1.16.0)
Requirement already satisfied: joblib>=1.2.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from scikit-learn>=0.20.0->category_encoders) (1.4.2)
Requirement already satisfied: threadpoolctl>=3.1.0 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from scikit-learn>=0.20.0->category_encoders) (3.5.0)
Requirement already satisfied: packaging>=21.3 in c:\users\vrajc\miniconda3\envs\py312\lib\site-packages (from statsmodels>=0.9.0->category_encoders) (24.1)
Note: you may need to restart the kernel to use updated packages.
```

In [21]:
```
import category_encoders as ce

encoder = ce.OrdinalEncoder(cols=['buying', 'maint', 'num_doors', 'num_persons',

X_train = encoder.fit_transform(X_train)

X_test = encoder.transform(X_test)
```

In [22]:
```
X_train.head()
```

Out[22]:

	buying	maint	num_doors	num_persons	boot_space	safety
48	1	1	1	1	1	1
468	2	1	1	2	2	1
155	1	2	1	1	2	2
1721	3	3	2	1	2	2
1208	4	3	3	1	2	2

In [23]:
```
X_test.head()
```

Out[23]:

	buying	maint	num_doors	num_persons	boot_space	safety
599	2	2	4	3	1	2
1201	4	3	3	2	1	3
628	2	2	2	3	3	3
1498	3	2	2	2	1	3
1263	4	3	4	1	1	1

In [24]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [25]:
```
clf_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=0)


# fit the model
clf_gini.fit(X_train, y_train)
```

Out[25]:
```
DecisionTreeClassifier(max_depth=3, random_state=0)
```

In [26]:
```
y_pred_gini = clf_gini.predict(X_test)
```

In [27]:
```
from sklearn.metrics import accuracy_score

print('Model accuracy score with criterion gini index: {0:0.4f}'. format(accurac
```

```
Model accuracy score with criterion gini index: 0.8021
```

In [28]:
```
y_pred_train_gini = clf_gini.predict(X_train)

y_pred_train_gini
```

Out[28]:
```
array(['unacc', 'unacc', 'unacc', ..., 'unacc', 'unacc', 'acc'],
      dtype=object)
```

In [29]:
```
print('Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train, y_
```

```
Training-set accuracy score: 0.7865
```

In [30]:
```
# print the scores on training and test set

print('Training set score: {:.4f}'.format(clf_gini.score(X_train, y_train)))

print('Test set score: {:.4f}'.format(clf_gini.score(X_test, y_test)))
```

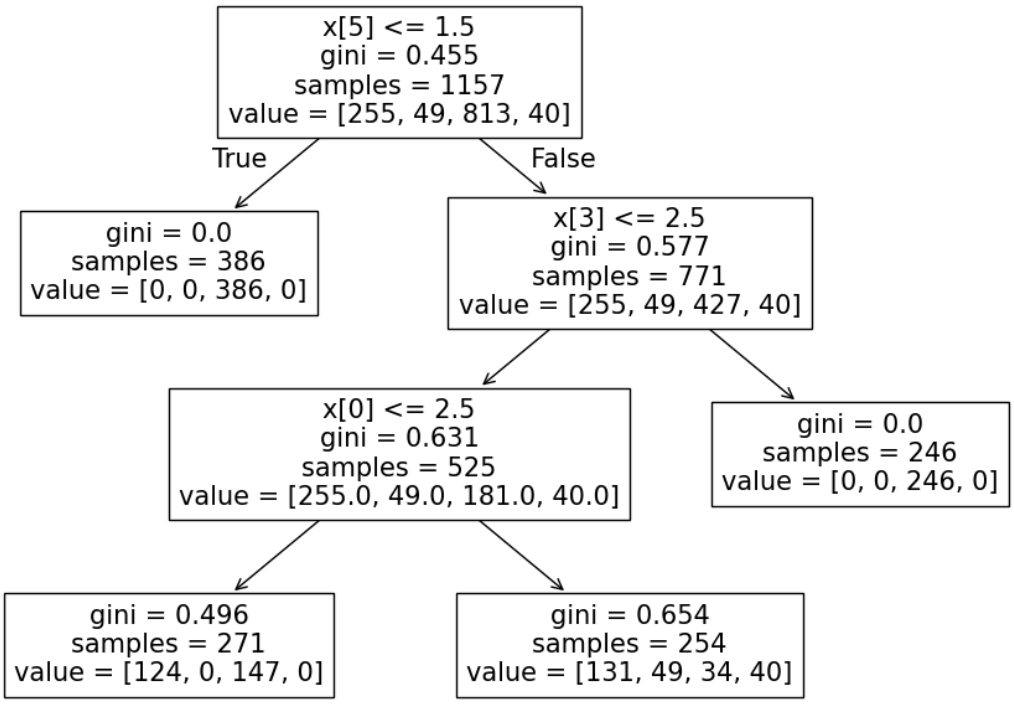
```
Training set score: 0.7865
Test set score: 0.8021
```

In [31]:
```
plt.figure(figsize=(12,8))

from sklearn import tree

tree.plot_tree(clf_gini.fit(X_train, y_train))
```

Out[31]:
```
[Text(0.4, 0.875, 'x[5] <= 1.5\ngini = 0.455\nsamples = 1157\nvalue = [255, 49,
813, 40]'),
 Text(0.2, 0.625, 'gini = 0.0\nsamples = 386\nvalue = [0, 0, 386, 0]'),
 Text(0.30000000000000004, 0.75, 'True  '),
 Text(0.6, 0.625, 'x[3] <= 2.5\ngini = 0.577\nsamples = 771\nvalue = [255, 49,
427, 40]'),
 Text(0.5, 0.75, '  False'),
 Text(0.4, 0.375, 'x[0] <= 2.5\ngini = 0.631\nsamples = 525\nvalue = [255.0, 4
9.0, 181.0, 40.0]'),
 Text(0.2, 0.125, 'gini = 0.496\nsamples = 271\nvalue = [124, 0, 147, 0]'),
 Text(0.6, 0.125, 'gini = 0.654\nsamples = 254\nvalue = [131, 49, 34, 40]'),
 Text(0.8, 0.375, 'gini = 0.0\nsamples = 246\nvalue = [0, 0, 246, 0]')]
```



In [32]:
```
clf_en = DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=0


# fit the model
clf_en.fit(X_train, y_train)
```

Out[32]:
```
DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=
0)
```

In [33]:
```
y_pred_en = clf_en.predict(X_test)
```

In [34]:
```
from sklearn.metrics import accuracy_score

print('Model accuracy score with criterion entropy: {0:0.4f}'. format(accuracy_s
```

```
Model accuracy score with criterion entropy: 0.8021
```

In [35]:
```
y_pred_train_en = clf_en.predict(X_train)
```

```
y_pred_train_en
```

Out[35]:
```
array(['unacc', 'unacc', 'unacc', ..., 'unacc', 'unacc', 'acc'],
      dtype=object)
```

In [36]:
```
print('Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train, y_
```

```
Training-set accuracy score: 0.7865
```

In [37]:
```
# print the scores on training and test set

print('Training set score: {:.4f}'.format(clf_en.score(X_train, y_train)))

print('Test set score: {:.4f}'.format(clf_en.score(X_test, y_test)))
```

```
Training set score: 0.7865
Test set score: 0.8021
```

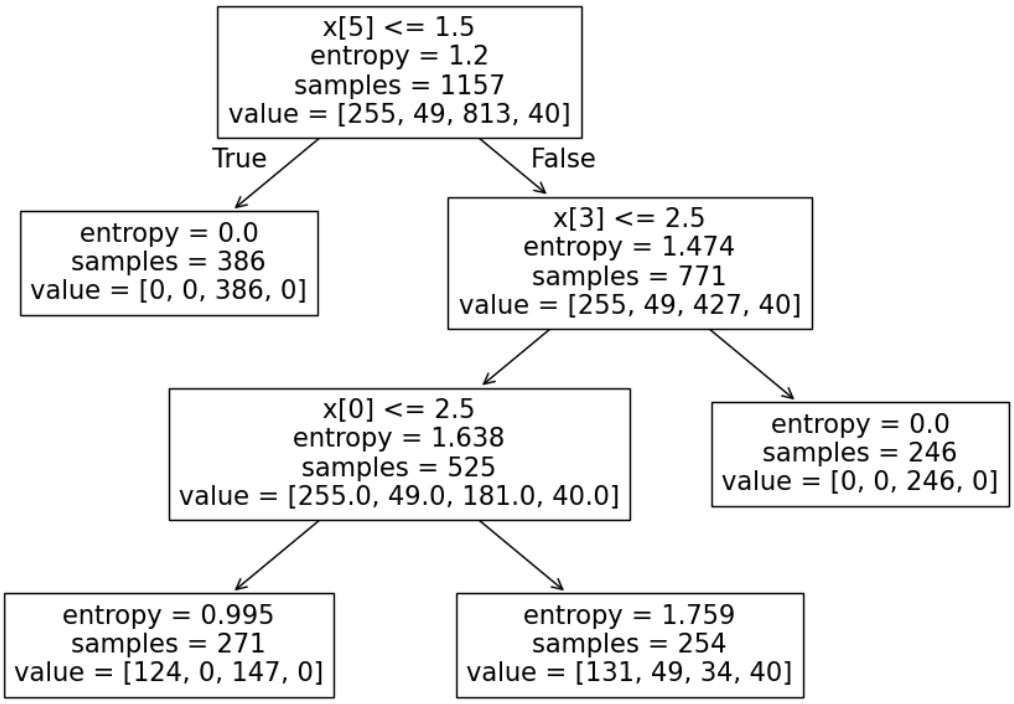
In [38]:
```
#plotting decision tree built on entropy

plt.figure(figsize=(12,8))

from sklearn import tree

tree.plot_tree(clf_en.fit(X_train, y_train))
```

Out[38]:
```
[Text(0.4, 0.875, 'x[5] <= 1.5\nentropy = 1.2\nsamples = 1157\nvalue = [255, 4
9, 813, 40]'),
 Text(0.2, 0.625, 'entropy = 0.0\nsamples = 386\nvalue = [0, 0, 386, 0]'),
 Text(0.30000000000000004, 0.75, 'True  '),
 Text(0.6, 0.625, 'x[3] <= 2.5\nentropy = 1.474\nsamples = 771\nvalue = [255, 4
9, 427, 40]'),
 Text(0.5, 0.75, '  False'),
 Text(0.4, 0.375, 'x[0] <= 2.5\nentropy = 1.638\nsamples = 525\nvalue = [255.0,
49.0, 181.0, 40.0]'),
 Text(0.2, 0.125, 'entropy = 0.995\nsamples = 271\nvalue = [124, 0, 147, 0]'),
 Text(0.6, 0.125, 'entropy = 1.759\nsamples = 254\nvalue = [131, 49, 34, 40]'),
 Text(0.8, 0.375, 'entropy = 0.0\nsamples = 246\nvalue = [0, 0, 246, 0]')]
```



In [39]:
```
# Confusion Matrix

from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_test, y_pred_en)

print('Confusion matrix\n\n', cm)
```

```
Confusion matrix

[[ 73   0  56   0]
 [ 20   0   0   0]
 [ 12   0 385   0]
 [ 25   0   0   0]]
```

In [40]:
```
from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred_en))
```

```
              precision    recall  f1-score   support

         acc       0.56      0.57      0.56       129
        good       0.00      0.00      0.00        20
       unacc       0.87      0.97      0.92       397
       vgood       0.00      0.00      0.00        25

    accuracy                           0.80       571
   macro avg       0.36      0.38      0.37       571
weighted avg       0.73      0.80      0.77       571
```