In [71]:

```
#Paul Galvez
#Part 2 Week 7
#DSC 550
#Date: 4/28/23
```

In [72]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import pickle
from sklearn.utils import shuffle
```

In [73]:

```
#Import the data as a data frame and ensure it is loaded correctly.

df2 = pd.read_csv('mushrooms.csv')
df2
```

Out[73]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	nu
0	p	x	s	n	t	p	f	c	n	k	...	s	w	w	p	w	
1	e	x	s	y	t	a	f	c	b	k	...	s	w	w	p	w	
2	e	b	s	w	t	l	f	c	b	n	...	s	w	w	p	w	
3	p	x	y	w	t	p	f	c	n	n	...	s	w	w	p	w	
4	e	x	s	g	f	n	f	w	b	k	...	s	w	w	p	w	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
8119	e	k	s	n	f	n	a	c	b	y	...	s	o	o	p	o	
8120	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	n	
8121	e	f	s	n	f	n	a	c	b	n	...	s	o	o	p	o	
8122	p	k	y	n	f	y	f	c	n	b	...	k	w	w	p	w	
8123	e	x	s	n	f	n	a	c	b	y	...	s	o	o	p	o	

8124 rows × 23 columns

In [74]:

```
df2.head()
```

Out[74]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring numbe
0	p	x	s	n	t	p	f	c	n	k	...	s	w	w	p	w	
1	e	x	s	y	t	a	f	c	b	k	...	s	w	w	p	w	
2	e	b	s	w	t	l	f	c	b	n	...	s	w	w	p	w	
3	p	x	y	w	t	p	f	c	n	n	...	s	w	w	p	w	
4	e	x	s	g	f	n	f	w	b	k	...	s	w	w	p	w	

5 rows × 23 columns

In [75]:

```
df2.shape
```

Out[75]:

```
(8124, 23)
```

In [76]:

```
df2.describe()
```

Out[76]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color
count	8124	8124	8124	8124	8124	8124	8124	8124	8124	8124	...	8124	8124	8124	8124	8124
unique	2	6	4	10	2	9	2	2	2	12	...	4	9	9	1	4
top	e	x	y	n	f	n	f	c	b	b	...	s	w	w	p	w
freq	4208	3656	3244	2284	4748	3528	7914	6812	5612	1728	...	4936	4464	4384	8124	7924

4 rows × 23 columns

In [77]:

```
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder,OneHotEncoder
```

In [78]:

```
label_encoder = preprocessing.LabelEncoder()
```

In [79]:

```
df2['class'] = label_encoder.fit_transform(df2['class'])
```

In [80]:

```
df2_cols = list(df2.columns)
```

In [81]:

```
#Convert the categorical features (all of them) to dummy variables.

for i in range (len(df2_cols)):
    df2[df2_cols[i]] = LabelEncoder().fit_transform(df2[df2_cols[i]])
```

In [82]:

```
#Convert the categorical features (all of them) to dummy variables.
#Also dropped the class column to make sure there are 22 columns not 23

print(df2.head)
y=df2['class']
df2=df2.drop(['class'], axis=1)
```

```
<bound method NDFrame.head of       class  cap-shape  cap-surface  cap-color  bruises  odor  \
0         1          5            2          4        1     6   
1         0          5            2          9        1     0   
2         0          0            2          8        1     3   
3         1          5            3          8        1     6   
4         0          5            2          3        0     5   
...     ...        ...          ...        ...      ...   ...   
8119      0          3            2          4        0     5   
8120      0          5            2          4        0     5   
8121      0          2            2          4        0     5   
8122      1          3            3          4        0     8   
8123      0          5            2          4        0     5   

      gill-attachment  gill-spacing  gill-size  gill-color  ...  \
0                   1             0          1           4  ...   
1                   1             0          0           4  ...   
2                   1             0          0           5  ...   
3                   1             0          1           5  ...   
4                   1             1          0           4  ...   
...               ...           ...        ...         ...  ...   
8119                0             0          0          11  ...   
8120                0             0          0          11  ...   
8121                0             0          0           5  ...   
8122                1             0          1           0  ...   
8123                0             0          0          11  ...   

      stalk-surface-below-ring  stalk-color-above-ring  \
0                            2                       7   
1                            2                       7   
2                            2                       7   
3                            2                       7   
4                            2                       7   
...                        ...                     ...   
8119                         2                       5   
8120                         2                       5   
8121                         2                       5   
8122                         1                       7   
8123                         2                       5   

      stalk-color-below-ring  veil-type  veil-color  ring-number  ring-type  \
0                          7          0           2            1          4   
1                          7          0           2            1          4   
2                          7          0           2            1          4   
```

```
3                          7          0           2            1          4
4                          7          0           2            1          0
...                      ...        ...         ...          ...        ...
8119                       5          0           1            1          4
8120                       5          0           0            1          4
8121                       5          0           1            1          4
8122                       7          0           2            1          0
8123                       5          0           1            1          4

      spore-print-color  population  habitat
0                     2           3        5
1                     3           2        1
2                     3           2        3
3                     2           3        5
4                     3           0        1
...                 ...         ...      ...
8119                  0           1        2
8120                  0           4        2
8121                  0           1        2
8122                  7           4        2
8123                  4           1        2

[8124 rows x 23 columns]>
```

In [83]:

```python
#Split the data into a training and test set.

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df2, y, test_size=0.3)
```

In [84]:

```python
#importing Decision Tree Classifier

from sklearn.tree import DecisionTreeClassifier
```

In [85]:

```python
#Fit a decision tree classifier on the training set.

clf_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=0)
```

In [86]:

```
#Fit a decision tree classifier on the training set.
#Fitting the model

clf_gini.fit(X_train, y_train)
```

Out[86]:

```
DecisionTreeClassifier(max_depth=3, random_state=0)
```

In [87]:

```
y_pred_gini=clf_gini.predict(X_test)
```

In [88]:

```
#Report the accuracy and create a confusion matrix for the model prediction on the test set.

from sklearn.metrics import accuracy_score
print('the accuracy score for the model is: {0:0.4f}'.format(accuracy_score(y_test, y_pred_gini)))
```

```
the accuracy score for the model is: 0.9569
```

In [89]:

```
#Report the accuracy and create a confusion matrix for the model prediction on the test set.

from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, y_pred_gini))
```

```
[[1199   73]
 [  32 1134]]
```

In [90]:

```
#Create a visualization of the decision tree. Setting the figure size for the visual

plt.figure(figsize=(15,10))
```

Out[90]:

```
<Figure size 1500x1000 with 0 Axes>

<Figure size 1500x1000 with 0 Axes>
```

In [91]:

```
from sklearn import tree
```

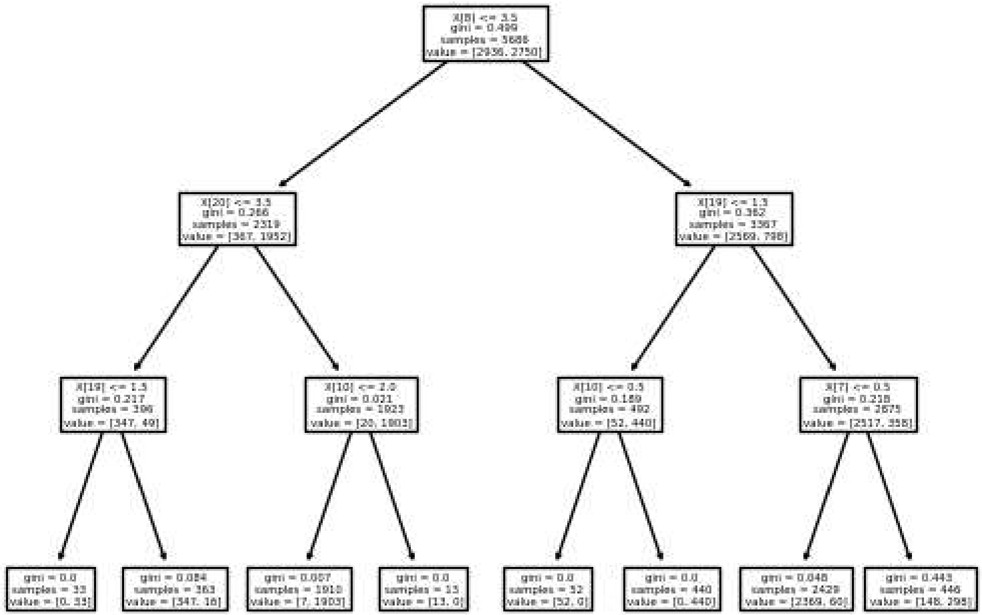
In [92]:

```
#the tree plot below shows the visual result for the model

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

Out[92]:

```
[Text(0.5, 0.875, 'X[8] <= 3.5\ngini = 0.499\nsamples = 5686\nvalue = [2936, 2750]'),
 Text(0.25, 0.625, 'X[20] <= 3.5\ngini = 0.266\nsamples = 2319\nvalue = [367, 1952]'),
 Text(0.125, 0.375, 'X[19] <= 1.5\ngini = 0.217\nsamples = 396\nvalue = [347, 49]'),
 Text(0.0625, 0.125, 'gini = 0.0\nsamples = 33\nvalue = [0, 33]'),
 Text(0.1875, 0.125, 'gini = 0.084\nsamples = 363\nvalue = [347, 16]'),
 Text(0.375, 0.375, 'X[10] <= 2.0\ngini = 0.021\nsamples = 1923\nvalue = [20, 1903]'),
 Text(0.3125, 0.125, 'gini = 0.007\nsamples = 1910\nvalue = [7, 1903]'),
 Text(0.4375, 0.125, 'gini = 0.0\nsamples = 13\nvalue = [13, 0]'),
 Text(0.75, 0.625, 'X[19] <= 1.5\ngini = 0.362\nsamples = 3367\nvalue = [2569, 798]'),
 Text(0.625, 0.375, 'X[10] <= 0.5\ngini = 0.189\nsamples = 492\nvalue = [52, 440]'),
 Text(0.5625, 0.125, 'gini = 0.0\nsamples = 52\nvalue = [52, 0]'),
 Text(0.6875, 0.125, 'gini = 0.0\nsamples = 440\nvalue = [0, 440]'),
 Text(0.875, 0.375, 'X[7] <= 0.5\ngini = 0.218\nsamples = 2875\nvalue = [2517, 358]'),
 Text(0.8125, 0.125, 'gini = 0.048\nsamples = 2429\nvalue = [2369, 60]'),
 Text(0.9375, 0.125, 'gini = 0.443\nsamples = 446\nvalue = [148, 298]')]
```



In [93]:

```
#Use a χ2-statistic selector to pick the five best features for this data
#(see section 10.4 of the Machine Learning with Python Cookbook).
```

In [94]:

```
#importing required libraries from sklearn

from sklearn.feature_selection import chi2
from sklearn.feature_selection import SelectKBest
```

In [95]:

```
#setting the top five best features

top_five = SelectKBest(score_func=chi2, k=5)
```

In [96]:

```
top_five.fit(df2.fillna(0), y)
```

Out[96]:

```
SelectKBest(k=5, score_func=<function chi2 at 0x0000024AE2A32040>)
```

In [97]:

```
#Which five features were selected in step 7? Hint: Use the get_support function.
#the top five fetures are listed below: 'bruises', 'gill-size', 'gill-color', 'stalk-root', 'ring-type'

df2.columns[top_five.get_support()].to_numpy()
```

Out[97]:

```
array(['bruises', 'gill-size', 'gill-color', 'stalk-root', 'ring-type'],
      dtype=object)
```

In [100]:

```
#Repeat steps 4 and 5 with the five best features selected in step 7.
```

In [101]:

```
nw_5=top_five.transform(df2)
nw_5=pd.DataFrame(nw_5)
```

In [102]:

```
#Train test and split the data with new variable nw_5 which represents the
#top five features.

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(nw_5, y, test_size=0.3)
```

In [103]:

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

In [104]:

```
clf_gini = DecisionTreeClassifier(criterion='gini', max_depth=3, random_state=0)
```

In [105]:

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

Out[105]:

```
DecisionTreeClassifier(max_depth=3, random_state=0)
```

In [106]:

```
Y_pred_gini=clf_gini.predict(X_test)
```

In [107]:

```
#The accuarcy score for the new model us below 0.5156

from sklearn.metrics import accuracy_score
print('the accuracy score for the model is: {0:0.4f}'.format(accuracy_score(y_test, y_pred_gini)))
```

```
the accuracy score for the model is: 0.5004
```

In [108]:

```
#the confusion matrix below for the new model for the top 5 features

from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test, y_pred_gini))
```

```
[[643 630]
 [588 577]]
```

In [109]:

```
#the accuarcy score in the second scenario is much lower because the number of features has been
#lowered to the top 5. We can also see the confusion matrix is different as well.
```