In [1]:
```
"""
Author: Santpsh chidura
Date: 4th -March-2019

"""
```

Out[1]:
```
'\nAuthor: Santpsh chidura\nDate: 4th -March-2019\n\n'
```

In [2]:
```
# Importing required libraries
import numpy as np
import pandas as pd
```

In [3]:
```
# Reading the data set into data frame df
df=pd.read_csv('http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data',names=['buying','maint','doors','pe
```

In [4]:
```
# displaying data frame head- first 5 records
df.head()
```

Out[4]:

	buying	maint	doors	persons	lug_boot	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 [5]:

```
# Displaying data frame information -structure - data type of each column
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1728 entries, 0 to 1727
Data columns (total 7 columns):
buying      1728 non-null object
maint       1728 non-null object
doors       1728 non-null object
persons     1728 non-null object
lug_boot    1728 non-null object
safety      1728 non-null object
class       1728 non-null object
dtypes: object(7)
memory usage: 94.6+ KB
```

In [6]:

```
# verifying any null values available in data frame
df.isnull().values.any()
```

Out[6]:

```
False
```

In [7]:

```
# describing the data frame
df.describe()
```

Out[7]:

	buying	maint	doors	persons	lug_boot	safety	class
count	1728	1728	1728	1728	1728	1728	1728
unique	4	4	4	3	3	3	4
top	high	high	5more	4	med	high	unacc
freq	432	432	432	576	576	576	1210

In [8]:

```
#all are category
df['buying'],uniq = pd.factorize(df['buying'])
df['maint'],uniq = pd.factorize(df['maint'])
df['doors'],uniq = pd.factorize(df['doors'])
df['persons'],uniq = pd.factorize(df['persons'])
df['lug_boot'],uniq = pd.factorize(df['lug_boot'])
df['safety'],uniq = pd.factorize(df['safety'])
```

In [9]:

```
df['class'],uniq_class=pd.factorize(df['class'])
```

In [10]:

```
df.head()
```

Out[10]:

	buying	maint	doors	persons	lug_boot	safety	class
0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0
2	0	0	0	0	0	2	0
3	0	0	0	0	1	0	0
4	0	0	0	0	1	1	0

In [11]:

```
df['class'].value_counts()
```

Out[11]:

```
0    1210
1     384
3      69
2      65
Name: class, dtype: int64
```

In [12]:

```
X = df.iloc[:,:-1]
y = df.iloc[:,-1]
```

In [13]:

```
# Importing sklearn librarires for further processing
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from time import time
from operator import itemgetter
```

In [14]:

```
# creating train and test data sets
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.3, random_state=1234,stratify=y)
```

In [15]:

```
def GridSearch_BestParam(X, y, clf, param_grid,cv=10):
    grid_search = GridSearchCV(clf,
                              param_grid=param_grid,
                              cv=cv)
    start= time()
    grid_search.fit(X,y)
    top_params=grid_search.best_params_
    return top_params
```

In [16]:

```
y_test.value_counts()
```

Out[16]:

```
0    363
1    115
3     21
2     20
Name: class, dtype: int64
```

In [17]:

```
param_grid_dt={'criterion':['gini','entropy'],
              'min_samples_split':[5,10,20,30,40],
              'max_depth':[2,3,5,7,9,15,20],
              'min_samples_leaf':[1,5,10,20,25,30]}
```

In [18]:

```
# creating Decesion tree model as model_dt
model_dt=DecisionTreeClassifier()
```

In [19]:

```
top_paramtrs=GridSearch_BestParam(X_train,y_train,model_dt,param_grid_dt,cv=10)
```

```
In [20]: print(top_paramtrs)

            {'criterion': 'entropy', 'max_depth': 15, 'min_samples_leaf': 1, 'min_samples_split': 5}
```

```
In [21]: best_dt=DecisionTreeClassifier(criterion='entropy',max_depth=15,min_samples_leaf=1,min_samples_split=5)
```

```
In [22]: best_dt.fit(X_train,y_train)

Out[22]: DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=15,
                     max_features=None, max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=5,
                     min_weight_fraction_leaf=0.0, presort=False, random_state=None,
                     splitter='best')
```

```
In [23]: best_dt.score(X_train,y_train)

Out[23]: 0.9909015715467329
```

```
In [24]: y_pred = best_dt.predict(X_test)
```

```
In [25]: from sklearn import metrics
```

```
In [26]: print(metrics.accuracy_score(y_test,y_pred))

            0.9556840077071291
```

```
In [27]: (y_test != y_pred).sum()

Out[27]: 23
```

```
In [28]: uniq_class

Out[28]: Index(['unacc', 'acc', 'vgood', 'good'], dtype='object')
```

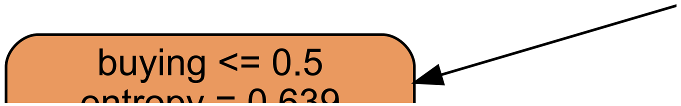
In [29]:

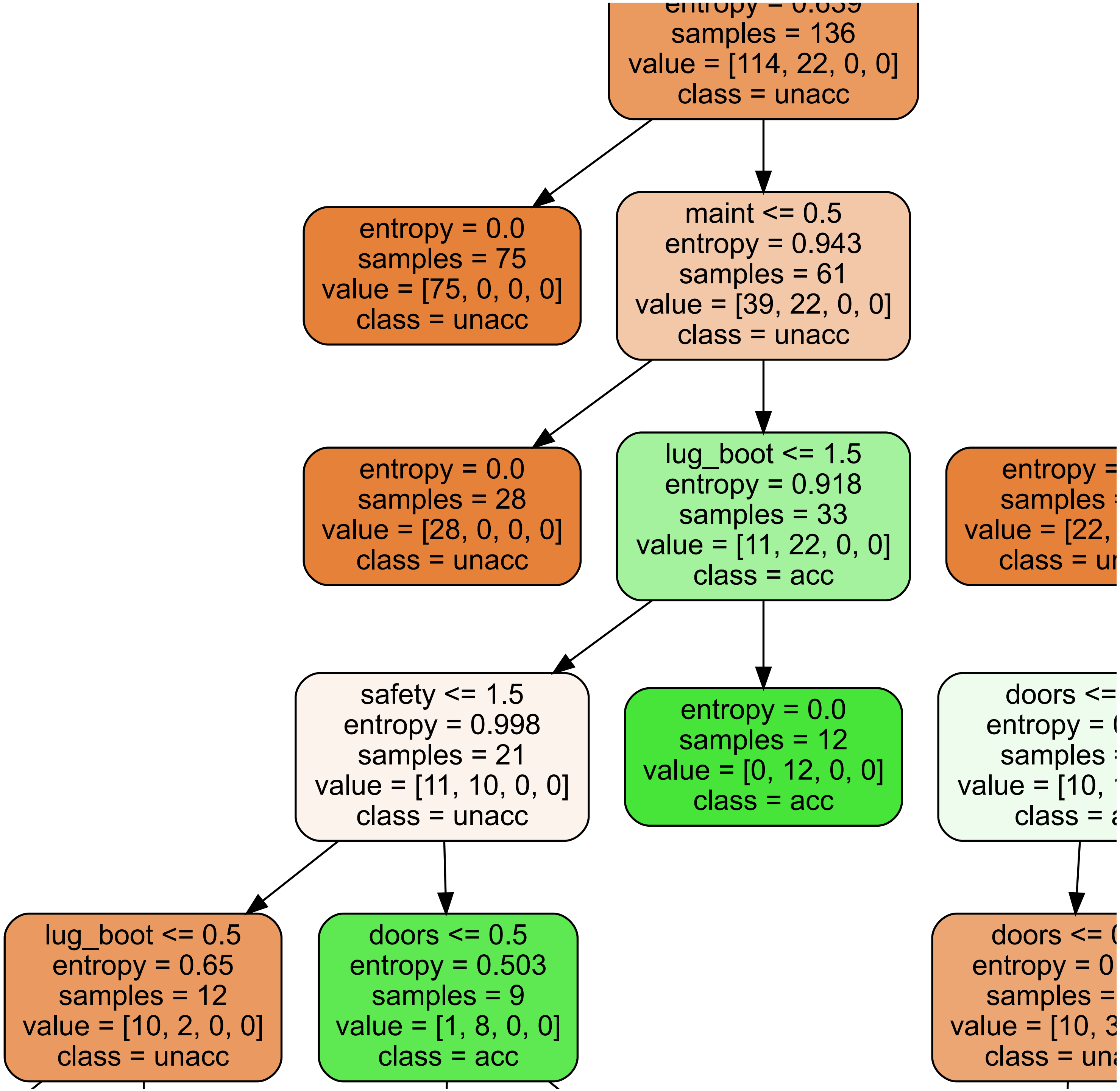
```
import graphviz
from sklearn import tree
feature_names = X.columns

dot_data = tree.export_graphviz(best_dt, out_file=None, filled=True, rounded=True,
                                feature_names=feature_names,class_names=uniq_class)
graph = graphviz.Source(dot_data)
```

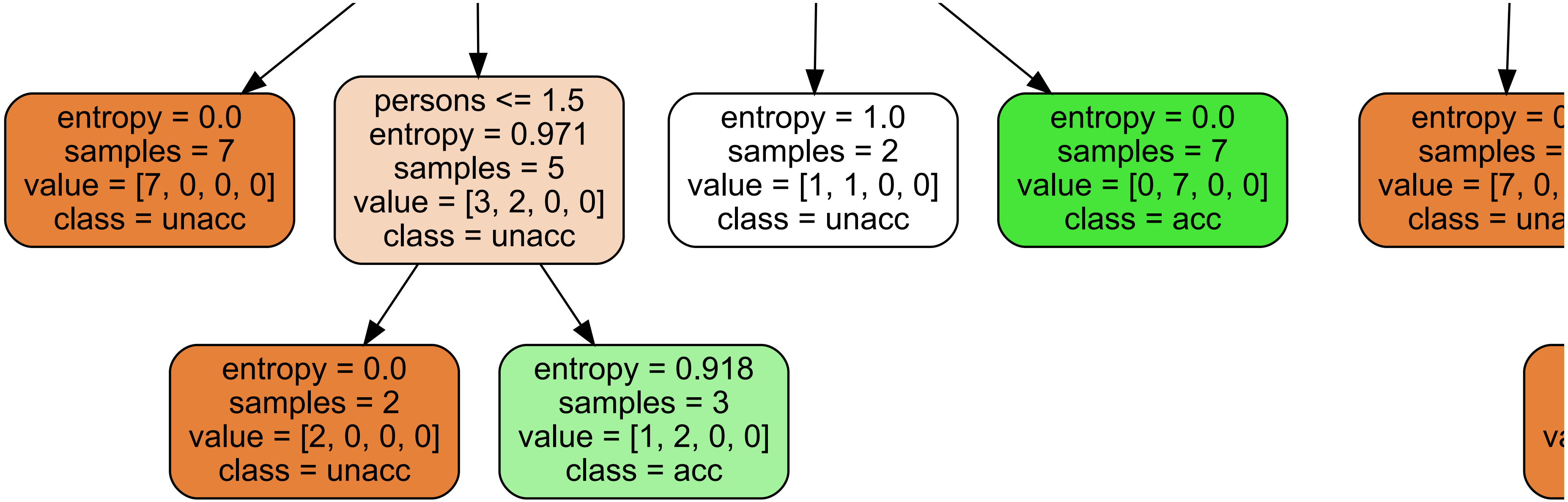
In [30]:

```
graph
```

Out[30]:




entropy = 0.639
samples = 136
value = [114, 22, 0, 0]
class = unacc
entropy = 0.0
samples = 75
value = [75, 0, 0, 0]
class = unacc
maint <= 0.5
entropy = 0.943
samples = 61
value = [39, 22, 0, 0]
class = unacc
entropy = 0.0
samples = 28
value = [28, 0, 0, 0]
class = unacc
lug_boot <= 1.5
entropy = 0.918
samples = 33
value = [11, 22, 0, 0]
class = acc
entropy =
samples =
value = [22,
class = ur
safety <= 1.5
entropy = 0.998
samples = 21
value = [11, 10, 0, 0]
class = unacc
entropy = 0.0
samples = 12
value = [0, 12, 0, 0]
class = acc
doors <=
entropy = (
samples =
value = [10,
class = a
lug_boot <= 0.5
entropy = 0.65
samples = 12
value = [10, 2, 0, 0]
class = unacc
doors <= 0.5
entropy = 0.503
samples = 9
value = [1, 8, 0, 0]
class = acc
doors <= (
entropy = 0
samples =
value = [10, 3
class = una


entropy = 0.0
samples = 7
value = [7, 0, 0, 0]
class = unacc
persons <= 1.5
entropy = 0.971
samples = 5
value = [3, 2, 0, 0]
class = unacc
entropy = 1.0
samples = 2
value = [1, 1, 0, 0]
class = unacc
entropy = 0.0
samples = 7
value = [0, 7, 0, 0]
class = acc
entropy = 0
samples =
value = [7, 0,
class = una
entropy = 0.0
samples = 2
value = [2, 0, 0, 0]
class = unacc
entropy = 0.918
samples = 3
value = [1, 2, 0, 0]
class = acc
va

In [ ]: