```
import requests
import pandas as pd
from sklearn import metrics
from sklearn.metrics import confusion_matrix

%matplotlib inline

url = 'https://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data'
data = pd.read_csv(url, names=['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class'])
data = data.drop(['persons'], axis=1)
data
```

```
print(data['buying'].unique())
print(data['class'].unique())

     ['vhigh' 'high' 'med' 'low']
     ['unacc' 'acc' 'vgood' 'good']
```

```
from sklearn.preprocessing import OrdinalEncoder
buying_price_category = ['low', 'med', 'high', 'vhigh']
maint_cost_category = ['low', 'med', 'high', 'vhigh']
doors_category = ['2', '3', '4', '5more']
person_capacity_category = ['2', '4', 'more']
lug_boot_category = ['small', 'med', 'big']
safety_category = ['low', 'med', 'high']
class_category = ['unacc', 'acc', 'vgood', 'good']
all_categories = [buying_price_category, maint_cost_category,doors_category,lug_boot_category,safety_category,class_category]
oe = OrdinalEncoder(categories= all_categories)

tf_data = oe.fit_transform( data)
X = tf_data[:,1:]
y = tf_data[:,0]

case = [['med', 'high', '4', 'big', 'high', 'good']]
tf_case = oe.fit_transform(case)
```

```
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=125)
```

```
from sklearn.tree import DecisionTreeClassifier
DT_classifier = DecisionTreeClassifier( criterion='gini' , max_depth= 5)
DT_classifier.fit(X_train, y_train)
y_pred = DT_classifier.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(metrics.classification_report(y_test, y_pred))
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
    [[19 45 36 38]
     [18 26 30 37]
     [ 4 22 56 60]
     [ 0 13 59 56]]
                  precision    recall  f1-score   support

             0.0       0.46      0.14      0.21       138
             1.0       0.25      0.23      0.24       111
             2.0       0.31      0.39      0.35       142
             3.0       0.29      0.44      0.35       128

        accuracy                           0.30       519
       macro avg       0.33      0.30      0.29       519
    weighted avg       0.33      0.30      0.29       519

    Accuracy: 0.302504816955684
```

```
case_pred = DT_classifier.predict(tf_case[:,1:])
print(buying_price_category[int(case_pred[0])])
```

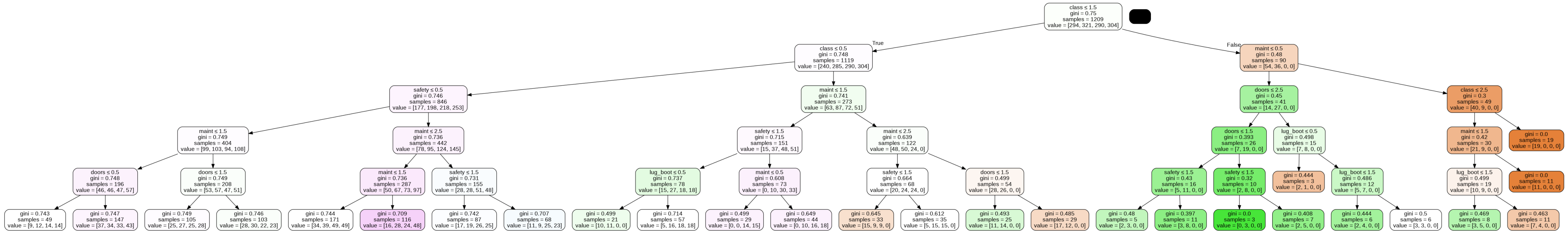
```
    low
```

```
from sklearn.tree import export_graphviz
from IPython.display import Image
import pydotplus

names = list(data.columns)
names.remove(names[0])

dot_data = StringIO()
export_graphviz(DT_classifier, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = names)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
```

```
Image(graph.create_png())
```



[redacted]

✓ 2s completed at 9:37 AM