In [6]:
```
# Author : Sayali Kudale
#importing required libraries
import pandas as pd
import scipy.io
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix
from sklearn import metrics
import time
```

In [2]:
```
#load USPS Dataset
mat = scipy.io.loadmat('Data/USPS_all.mat')

#get dataset in dataframes
df_features = pd.DataFrame(mat['fea'])
df_gnd = pd.DataFrame(mat['gnd'])
```

In [3]:
```
#Data splitting into training and testing

train_features= df_features.iloc[:7291]
test_features= df_features.iloc[-2007:]
train_decisions= df_gnd.iloc[:7291]
test_decisions= df_gnd.iloc[-2007:]
```

Parameters info:

max_Depth: decides how deeper is the tree.

min_samples_split : this parameter decides minimum number of samples required to split an internal node

max_features: minimum number of features to be consider for a best split

min_samples_leaf: this decides minimum number of samples required to be at the leaf node(i.e at the base of the tree)

In [113]:
```
#feature scaling

sc_X = StandardScaler()
train_features = sc_X.fit_transform(train_features)
test_features = sc_X.transform(test_features)


def decisionTreeClassifier(criterion="gini",max_depth=None,max_features=
None,min_samples_split=2,min_samples_leaf=1):

    classifier = DecisionTreeClassifier(criterion = criterion,max_depth=
max_depth,max_features=max_features,min_samples_split=min_samples_split,
min_samples_leaf=min_samples_leaf)
    trainStart = time.time()

    classifier = classifier.fit(train_features.astype(int),train_decisio
ns.astype(int))

    trainEnd = time.time()

    training_time=trainEnd-trainStart

    #prediction
    y_pred = classifier.predict(test_features)


    # test accuracy
    cm= confusion_matrix(test_decisions, y_pred)

    accuracy=metrics.accuracy_score(test_decisions,y_pred)

    graph=print_Tree(classifier)

    return training_time,cm,accuracy, graph
```

In [85]:
```
# Print the decision Tree


from sklearn.tree import export_graphviz
from sklearn.externals.six import StringIO
from IPython.display import Image
import pydotplus

def print_Tree(classifier):
    dot_data=StringIO()
    feature_cols = list(range(0, 255))

    export_graphviz(classifier, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True,feature_names = feature_cols,cla
ss_names=['1','2','3','4','5','6','7','8','9','10'])

    graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
    return graph
```

Decision Tree Classifer with below parameters:

split criterion - Entropy

max_depth - 5

max_features - 200

min_samples_split - 10

min_samples_leaf- 10

In [98]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier('entropy',5,200,1
0,10)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.1506969928741455
Confusion Matrix:
 [[330   2   3   4   3   1   8   0   6   2]
 [  1 252   6   0   5   0   0   0   0   0]
 [ 43   7 100   7   4   2   8   6  18   3]
 [ 35   6  15  71   5  17   2   5   7   3]
 [  1  11  19   0 136   0   0  22   4   7]
 [ 33   4   7  10   3  77  17   3   4   2]
 [  2  11  17   1   0   5 131   0   2   1]
 [  3   2   8   0   4   0   0 124   1   5]
 [  4  20   9  16  13   6  10  11  70   7]
 [  0   9   5   0  32   0   0  19   0 112]]
Accuracy Score of decision tree classifier 0.699053313403
```

In [99]:
```
# Print the decision Tree

Image(graph.create_png())
```

Out[99]:

Decision Tree Classifer with increased depth:

split criterion - Entropy

max_depth - 10

max_features - 200

min_samples_split - 20

min_samples_leaf- 15

In this sample we increased the depth of the tree to 10. If we improve the depth the accuracy of the model

In [104]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier('entropy',10,200,
20,15)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.19078612327575684
Confusion Matrix:
 [[323   1   4   0   6  16   2   1   4   2]
 [  2 254   6   0   1   0   0   1   0   0]
 [ 14  13 124   2  12  12   3  13   4   1]
 [ 12   6   7  79   4  49   2   3   3   1]
 [  2  13  14   4 138  12   1  13   0   3]
 [ 14   8   8   1   6 119   3   1   0   0]
 [ 10   7   6   0   9  12 125   0   1   0]
 [  6   6   2   0   5   1   1 116   0  10]
 [ 10   7   9   2  33  19   2   4  59  21]
 [  2  13   1  12  27   7   0  15   0 100]]
Accuracy Score of decision tree classifier 0.715994020927
```

In [105]:
```
# Print the decision Tree
Image(graph.create_png())
```

Out[105]:

Decision Tree Classifer: max feature value

split criterion - Entropy

max_depth - 5

max_features - 250

min_samples_split - 10

In [106]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier('entropy',5,250,1
0,10)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.15143990516662598
Confusion Matrix:
 [[300   2  32   2   3   1   8   0   9   2]
 [  1 252   6   0   5   0   0   0   0   0]
 [ 12   7 145   2   4   2   8  13   4   1]
 [  5   6  28  69   5  17   2   3  27   4]
 [  0  11  27   0 136   0   0  17   1   8]
 [ 26   4  11   7   3  77  17   2   8   5]
 [  2  11  19   0   0   5 131   1   1   0]
 [  0   2  10   1   4   0   0 114   1  15]
 [  9  20  19   2  13   6  10   5  76   6]
 [  0   9   7   0  32   0   0  16   0 113]]
Accuracy Score of decision tree classifier 0.704035874439
```

In [107]:
```
# Print Tree

Image(graph.create_png())
```

Out[107]:

Decision Tree Classifer: gini criterion

split criterion - gini

max_depth - 5

max_features - 250

min_samples_split - 10

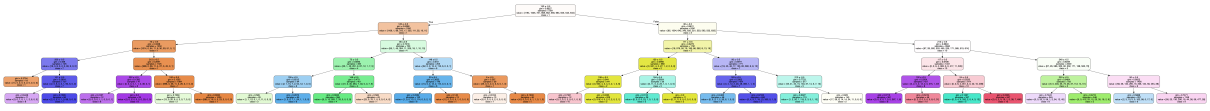
min_samples_leaf- 10

In [108]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier('gini',5,250,10,1
0)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.14022588729858398
Confusion Matrix:
 [[304   5  32   1   1   4   4   3   5   0]
 [  4 227   0   0   0   0   2   1   0  30]
 [ 18  15  93   2  11   8   2  21  25   3]
 [ 10   4  17  66   2  28   1   4  32   2]
 [  3  10   1   4 115  15  12  18   3  19]
 [ 22   8  11   0   3 108   2   1   3   2]
 [  3   4   4   0   4  15 132   6   2   0]
 [  3   4   2   0   3   0   0 127   1   7]
 [  9  12   5   0   7  20   0   7  79  27]
 [  3   9   0  17  15   1   0  23   0 109]]
Accuracy Score of decision tree classifier 0.677628300947
```

In [110]:
```
Image(graph.create_png())
```

Out[110]:



Decision Tree Classifer: min samples split 10

split criterion - Entropy

min_samples_split - 10

In [117]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier(criterion='entrop
y',min_samples_split=10)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.3564128875732422
Confusion Matrix:
 [[321   1   8  14   3   3   7   0   0   2]
 [  1 240   2   5  14   0   1   0   1   0]
 [ 14   4 144  16   6   1   0  10   3   0]
 [ 10   2  17  94   2  22   2   6   4   7]
 [  6   7   7   4 155   5   0  10   3   3]
 [ 14   3   6  13   3 101  12   3   3   2]
 [  6   3   8   2   2  15 130   3   1   0]
 [  2   0   3   7   6   0   0 116   8   5]
 [  8  17  16  10  13   7   5   9  74   7]
 [  0  10   1   6  10   0   0  17   1 132]]
Accuracy Score of decision tree classifier 0.750871948181
```

In [118]:
```
Image(graph.create_png())
```

Out[118]:

Decision Tree Classier eith below parameters:

split criterion - Entropy

max_depth - 5

min_samples_split - 10

In [119]:
```
trainingTime,cm,accuracy,graph= decisionTreeClassifier(criterion='entrop
y',max_depth=5,min_samples_split=10)
print("Training time : ", trainingTime)

print("Confusion Matrix: \n",cm)

print("Accuracy Score of decision tree classifier", accuracy)
```

```
Training time :  0.15929102897644043
Confusion Matrix:
 [[300   2  32   2   3   1   8   0   9   2]
 [  1 252   1   5   5   0   0   0   0   0]
 [ 15   2 143   4   4   2   8  15   4   1]
 [  5   4  26  71   5  17   2   5  27   4]
 [  4   7  25   2 136   0   0  17   1   8]
 [ 27   3  11   7   3  77  17   2   8   5]
 [  2  11  18   1   0   5 131   1   1   0]
 [  0   0   4   7   4   0   0 116   1  15]
 [ 11  15  16   5  13   6  10   8  76   6]
 [  0   9   2   5  32   0   0  16   0 113]]
Accuracy Score of decision tree classifier 0.705032386647
```

In [120]:
```
Image(graph.create_png())
```

Out[120]: