DECISION TREE CLASSIFIER USING GINI INDEX AS IMPURITY MEASURE

1. Load the German Credit card dataset

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable

2. Create a Pandas Frame for this file and explore its content

```
import pandas as pd
df=pd.read_csv('/content/German Credit Data (1).csv')
```

3. Print the first five records and first 7 columns

```
df.iloc[0:5,0:7]
```

	checkin_acc	duration	credit_history	amount	savings_acc	present_emp_since	inst_r
0	A11	6	A34	1169	A65	A75	
1	A12	48	A32	5951	A61	A73	
2	A14	12	A34	2096	A61	A74	
3	A11	42	A32	7882	A61	A74	

4. Print the first five records and remaining columns

```
df.iloc[0:5,7:15]
```

	personal_status	residing_since	age	inst_plans	num_credits	job	status
0	A93	4	67	A143	2	A173	0
1	A92	2	22	A143	1	A173	1
2	A93	3	49	A143	1	A172	0
3	A93	4	45	A143	1	A173	0
4	A93	4	53	A143	2	A173	1

5. Few of the columns are categorical and are infered as objects. Ex: checkin_acc. Print all unique values of this column

```
df['checkin_acc'].unique()
```

```
array(['A11', 'A12', 'A14', 'A13'], dtype=object)
```

6. Encode all categorical features using one-hot encoding. A feature with n values is encoded using (n-1) values, retaining the first one (drop_first = True)

```
x_features = list(df.columns)
x_features.remove('status')
encoded_df =pd.get_dummies(df[x_features],drop_first=True)
print(list(encoded_df.columns))
```

```
['duration', 'amount', 'inst_rate', 'residing_since', 'age', 'num_credits', 'checkin_acc_A12', 'checkin_acc_A13', 'checkin_acc_A14', 'cr
```

7. Make independent features of the encoded frame as X and column 'status' as dependent feature.

```
X=encoded_df
Y=df['status']
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-274ff8870b66> in <module>
----> 1 X=encoded_df
      2 Y=df['status']

NameError: name 'encoded_df' is not defined
```

SEARCH STACK OVERFLOW

8. Divide data into 70% training and 30% as testing.

```
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=42)
```

9. Train a decision tree model using Gini INdex and depth of 3

```
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier(criterion='entropy',max_depth=3)
clf.fit(X_train,Y_train)
```

```
DecisionTreeClassifier(criterion='entropy', max_depth=3)
```

10. Make predictions on test/validation data

```
pred_y=clf.predict(X_test)
```

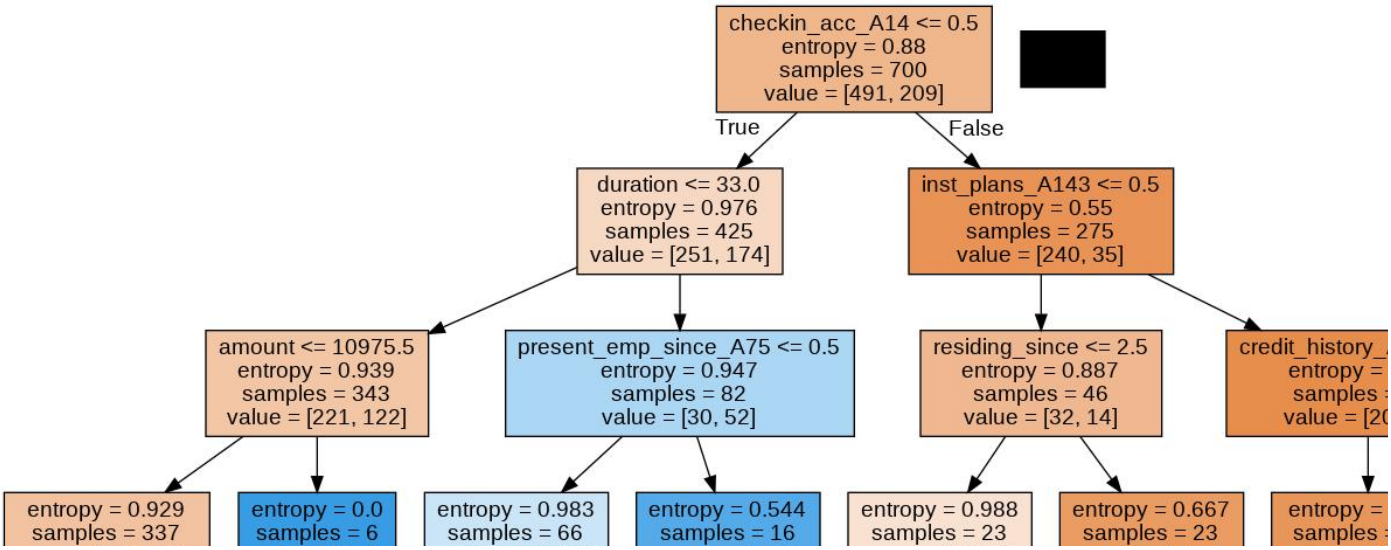
11. Print the confusion matrix, accuracy and AUC score of this model on test set

```
from sklearn import metrics
print("Confusion Matrix is\n",metrics.confusion_matrix(pred_y,Y_test))
print("Accuracy is\n",metrics.accuracy_score(pred_y,Y_test))
print("AUC score is \n",metrics.roc_auc_score(pred_y,Y_test))
```

```
Confusion Matrix is
 [[195  71]
 [ 14  20]]
Accuracy is
 0.7166666666666667
AUC score is
 0.6606590004422821
```

12. Visualize the tree using grapghviz and pydotplus libraries

```
from sklearn.tree import export_graphviz
import pydotplus as plot
from IPython.display import Image
export_graphviz(clf,out_file="tree.odt",feature_names=X_train.columns,filled=True)
graph=plot.graphviz.graph_from_dot_file("tree.odt")
graph.write_jpg("tree.png")
Image(filename="tree.png")
```


checkin_acc_A14 <= 0.5
entropy = 0.88
samples = 700
value = [491, 209]
True
False
duration <= 33.0
entropy = 0.976
samples = 425
value = [251, 174]
inst_plans_A143 <= 0.5
entropy = 0.55
samples = 275
value = [240, 35]
amount <= 10975.5
entropy = 0.939
samples = 343
value = [221, 122]
present_emp_since_A75 <= 0.5
entropy = 0.947
samples = 82
value = [30, 52]
residing_since <= 2.5
entropy = 0.887
samples = 46
value = [32, 14]
credit_history_
entropy =
samples =
value = [2
entropy = 0.929
samples = 337
entropy = 0.0
samples = 6
entropy = 0.983
samples = 66
entropy = 0.544
samples = 16
entropy = 0.988
samples = 23
entropy = 0.667
samples = 23
entropy =
samples =