In [12]:
```
import numpy as np
import pandas as pd
from sklearn.impute import SimpleImputer
import collections
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
```

In [14]:
```
#import the file using pandas
df=pd.read_csv('train.csv')
print('Shape of the data',df.shape)
print()
print(df.head())
```

```
Shape of the data (9557, 143)

             Id      v2a1  hacdor  rooms  hacapo  v14a  refrig  v18q  v18q1  \
0  ID_279628684  190000.0       0      3       0     1       1     0    NaN
1  ID_f29eb3ddd  135000.0       0      4       0     1       1     1    1.0
2  ID_68de51c94       NaN       0      8       0     1       1     0    NaN
3  ID_d671db89c  180000.0       0      5       0     1       1     1    1.0
4  ID_d56d6f5f5  180000.0       0      5       0     1       1     1    1.0

   r4h1  ...  SQBescolari  SQBage  SQBhogar_total  SQBedjefe  SQBhogar_nin  \
0     0  ...          100    1849               1        100             0
1     0  ...          144    4489               1        144             0
2     0  ...          121    8464               1          0             0
3     0  ...           81     289              16        121             4
4     0  ...          121    1369              16        121             4

   SQBovercrowding  SQBdependency  SQBmeaned  agesq  Target
0         1.000000            0.0      100.0   1849       4
1         1.000000           64.0      144.0   4489       4
2         0.250000           64.0      121.0   8464       4
3         1.777778            1.0      121.0    289       4
4         1.777778            1.0      121.0   1369       4

[5 rows x 143 columns]
```

In [15]:
```
df.isnull().sum()
```

Out[15]:
```
Id                     0
v2a1                6860
hacdor                 0
rooms                  0
hacapo                 0
                    ...
SQBovercrowding        0
SQBdependency          0
SQBmeaned              5
agesq                  0
Target                 0
Length: 143, dtype: int64
```

In [16]:
```
null_columns=df.columns[df.isnull().any()]
df[null_columns].isnull().sum()
```

Out[16]:
```
v2a1         6860
v18q1        7342
```

```
rez_esc      7928
meaneduc        5
SQBmeaned       5
dtype: int64
```

In [17]:
```
print ('Percentage of null values in v2a1 : ', df['v2a1'].isnull().sum()/df.shape[0]
print ('Percentage of null values in v18q1 : ', df['v18q1'].isnull().sum()/df.shape[
print ('Percentage of null values in rez_esc : ', df['rez_esc'].isnull().sum()/df.sh
print ('Percentage of null values in meaneduc : ', df['meaneduc'].isnull().sum()/df.
print ('Percentage of null values in SQBmeaned : ', df['SQBmeaned'].isnull().sum()/d
```

```
Percentage of null values in v2a1 :  71.7798472323951
Percentage of null values in v18q1 :  76.82327090091033
Percentage of null values in rez_esc :  82.95490216595167
Percentage of null values in meaneduc :  0.05231767290990897
Percentage of null values in SQBmeaned :  0.05231767290990897
```

In [18]:
```
#Percentage of null values in v2a1, v18q1, rez_esc is more than 50%. So, these colum
df= df.drop(['v2a1','v18q1','rez_esc'],axis=1)
print(df.shape)
```

```
(9557, 140)
```

In [19]:
```
#Imputing the meaneduc & SQBmeaned coumns
imp = SimpleImputer(missing_values=np.nan, strategy='median')
imp.fit(df[['meaneduc','SQBmeaned']])
df[['meaneduc','SQBmeaned']]=imp.transform(df[['meaneduc','SQBmeaned']])
df[['meaneduc','SQBmeaned']].isnull().sum()
```

Out[19]:
```
meaneduc     0
SQBmeaned    0
dtype: int64
```

In [20]:
```
df= df.drop(['Id'],axis=1)
df.describe(include='O')
```

Out[20]:

	idhogar	dependency	edjefe	edjefa
count	9557	9557	9557	9557
unique	2988	31	22	22
top	fd8a6d014	yes	no	no
freq	13	2192	3762	6230

In [21]:
```
df.dependency = df.dependency.replace(to_replace=['yes','no'],value=[0.5,0]).astype(
```

In [22]:
```
med_1=np.median(df.edjefe[df.edjefe.isin(['yes','no'])==False].astype('float'))
df.edjefe= df.edjefe.replace(to_replace=['yes','no'],value=[med_1,0]).astype('float'
```

In [23]:
```
med_2=np.median(df.edjefa[df.edjefa.isin(['yes','no'])==False].astype('float'))
df.edjefa= df.edjefa.replace(to_replace=['yes','no'],value=[med_2,0]).astype('float'
```

In [24]:
```
df.describe(include='O')
```

Out[24]:

	idhogar
count	9557
unique	2988
top	fd8a6d014
freq	13

In [25]:
```
print(df.idhogar.nunique())
```

```
2988
```

In [26]:
```
df.Target.value_counts()
import collections
print(df.shape)
collections.Counter(df['Target'])
```

```
(9557, 139)
```

Out[26]:
```
Counter({4: 5996, 2: 1597, 3: 1209, 1: 755})
```

In [27]:
```
poverty_level=(df.groupby('idhogar')['Target'].nunique()>1).index
print(poverty_level)
```

```
Index(['001ff74ca', '003123ec2', '004616164', '004983866', '005905417',
       '006031de3', '006555fe2', '00693f597', '006b64543', '00941f1f4',
       ...
       'ff250fd6c', 'ff31b984b', 'ff38ddef1', 'ff6d16fd0', 'ff703eed4',
       'ff9343a35', 'ff9d5ab17', 'ffae4a097', 'ffe90d46f', 'fff7d6be1'],
      dtype='object', name='idhogar', length=2988)
```

In [28]:
```
no_head=(df.groupby('idhogar')['parentesco1'].sum()==0).index
display(no_head)
```

```
Index(['001ff74ca', '003123ec2', '004616164', '004983866', '005905417',
       '006031de3', '006555fe2', '00693f597', '006b64543', '00941f1f4',
       ...
       'ff250fd6c', 'ff31b984b', 'ff38ddef1', 'ff6d16fd0', 'ff703eed4',
       'ff9343a35', 'ff9d5ab17', 'ffae4a097', 'ffe90d46f', 'fff7d6be1'],
      dtype='object', name='idhogar', length=2988)
```

In [29]:
```
target_mean=df.groupby('idhogar')['Target'].mean().astype('int64').reset_index().ren
df=df.merge(target_mean,how='left',on='idhogar')
df.Target=df.Target_mean
df.drop('Target_mean',axis=1,inplace=True)
df.head()
```

Out[29]:

	hacdor	rooms	hacapo	v14a	refrig	v18q	r4h1	r4h2	r4h3	r4m1	...	SQBescolari	SQBage
0	0	3	0	1	1	0	0	1	1	0	...	100	1849
1	0	4	0	1	1	1	0	1	1	0	...	144	4489
2	0	8	0	1	1	0	0	0	0	0	...	121	8464
3	0	5	0	1	1	1	0	2	2	1	...	81	289
4	0	5	0	1	1	1	0	2	2	1	...	121	1369

5 rows × 139 columns

In [30]:
```
df.shape
```

Out[30]:
```
(9557, 139)
```

In [31]:
```
df= df.drop(['idhogar'],axis=1)
df.shape
```

Out[31]:
```
(9557, 138)
```

In [32]:
```
x=df.drop(['Target'],axis=1)
print('shape of the x',x.shape)
y=df.Target
print('shape of the y',y.shape)
```

```
shape of the x (9557, 137)
shape of the y (9557,)
```

In [33]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=10)
rfc = RandomForestClassifier(criterion= 'gini',n_estimators=100)
rfc.fit(x_train,y_train)
pred=rfc.predict(x_test)
```

In [34]:
```
print('Accuracy score: ', accuracy_score(pred,y_test))
print()
print('Confusion matrix: ', confusion_matrix(pred,y_test))
print()
print('Classification report: ', classification_report(pred,y_test))
```

```
Accuracy score:  0.928347280334728

Confusion matrix:  [[ 136    2    1    2]
 [   3  266    3    4]
 [   0    3  170    1]
 [  30   46   42 1203]]

Classification report:                precision    recall  f1-score   support

           1       0.80      0.96      0.88       141
           2       0.84      0.96      0.90       276
           3       0.79      0.98      0.87       174
           4       0.99      0.91      0.95      1321

    accuracy                           0.93      1912
   macro avg       0.86      0.95      0.90      1912
weighted avg       0.94      0.93      0.93      1912
```

In [ ]:
```
```