Predict the Criminal

Import Libraries

In [29]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
```

Get the Data

In [30]:
```
train = pd.read_csv('criminal_train.csv')
test = pd.read_csv('criminal_test.csv')
```

In [31]:
```
train.head()
```

Out[31]:

	PERID	IFATHER	NRCH17_2	IRHHSIZ2	IIHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	I
0	25095143	4	2	4	1	3	1	1	1
1	13005143	4	1	3	1	2	1	1	1
2	67415143	4	1	2	1	2	1	1	1
3	70925143	4	0	2	1	1	1	1	1
4	75235143	1	0	6	1	4	1	1	1

5 rows × 72 columns

In [32]:
```
test.head()
```

Out[32]:

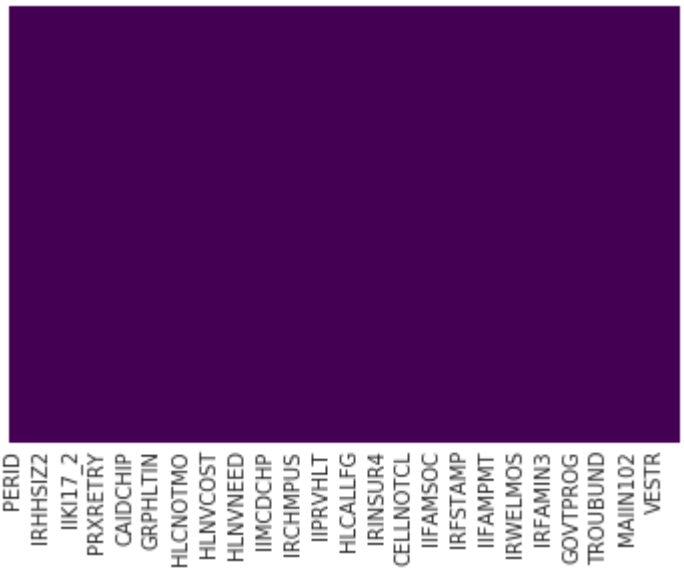
	PERID	IFATHER	NRCH17_2	IRHHSIZ2	IIHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	I
0	66583679	4	0	4	1	2	1	1	1
1	35494679	4	0	4	1	1	1	1	1
2	79424679	2	0	3	1	2	1	1	1
3	11744679	4	0	6	1	2	1	1	1
4	31554679	1	0	4	1	3	1	1	1

5 rows × 71 columns

Exploratory Data Analysis

In [33]:
```
sns.heatmap(train.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

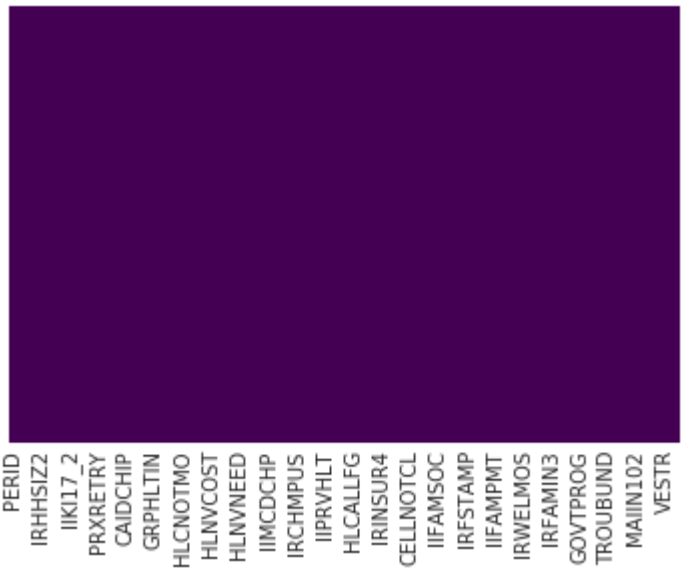
Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ef1bf160>



Train data do not have any Null values

In [34]:
```
sns.heatmap(test.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

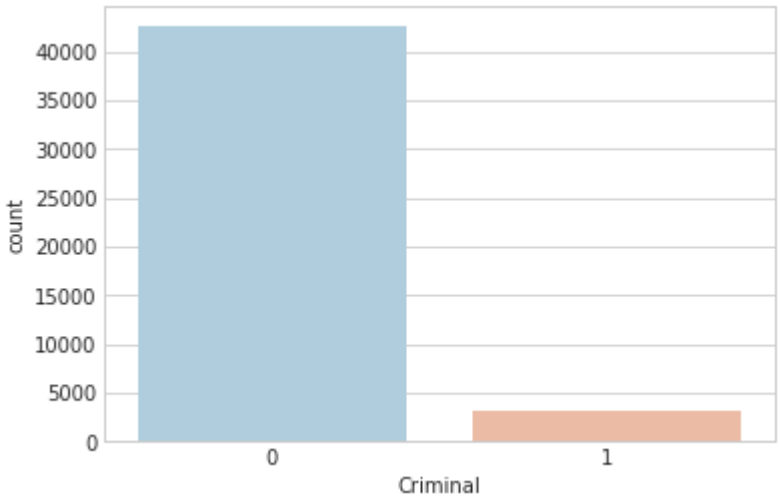
Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ef1e1a20>



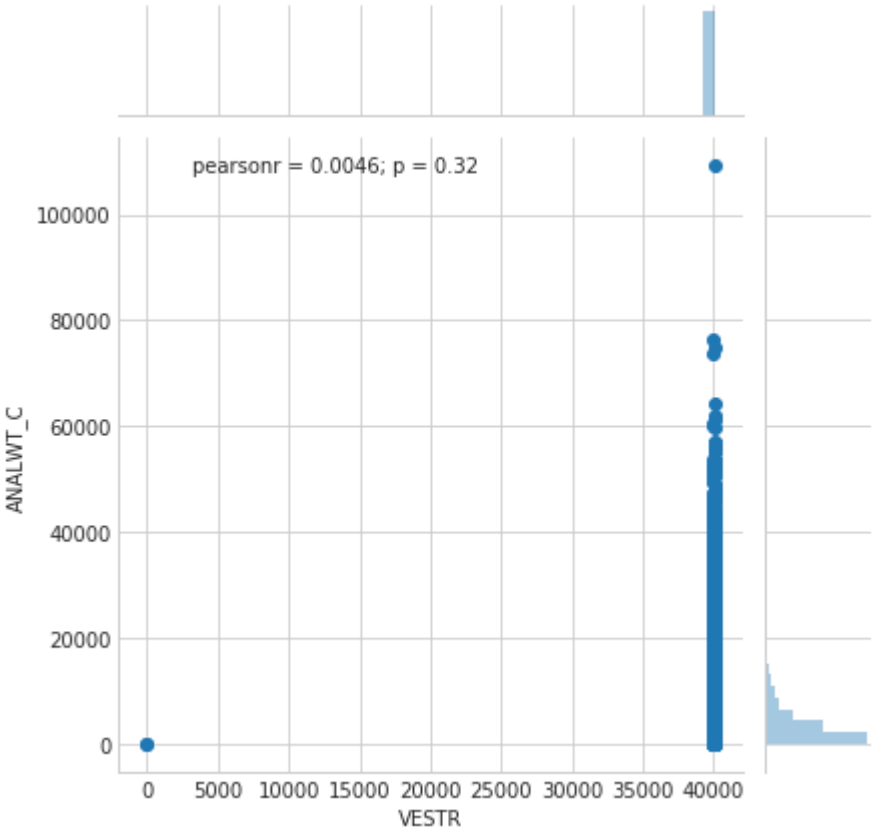
test data do not have any Null values

In [35]:
```
sns.set_style('whitegrid')
sns.countplot(x='Criminal',data=train,palette='RdBu_r')
```

Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ee817240>

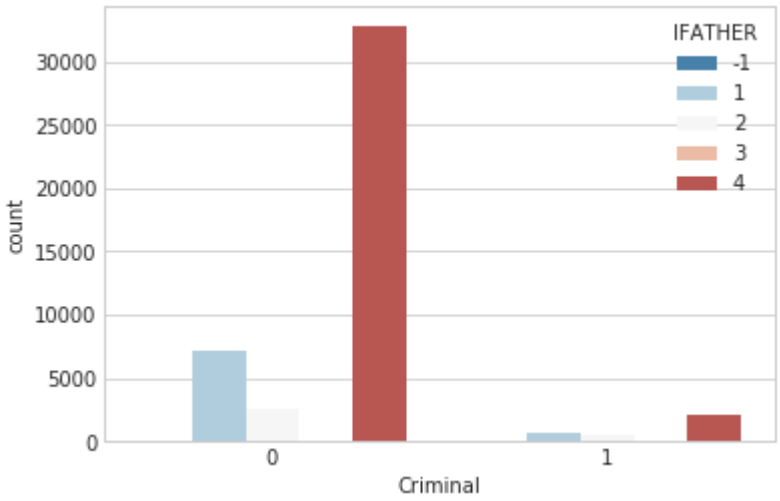


In [36]:
```
sns.jointplot(x="VESTR", y="ANALWT_C", data=train);
```



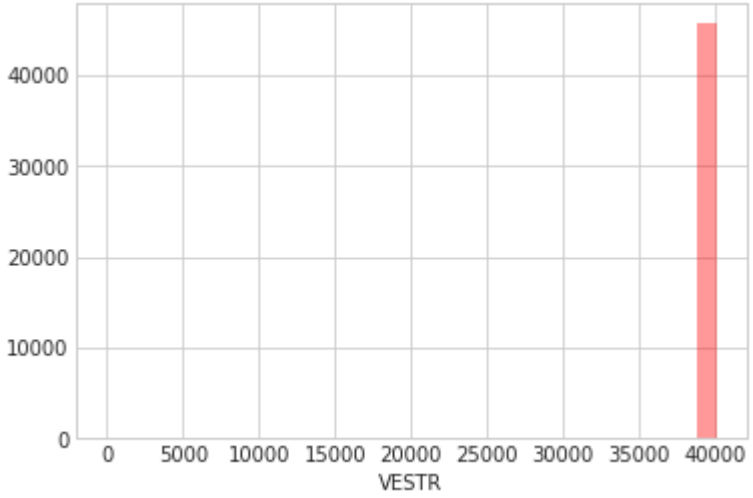
In [37]:
```
sns.set_style('whitegrid')
sns.countplot(x='Criminal',hue='IFATHER',data=train,palette='RdBu_r')
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ee6509e8>



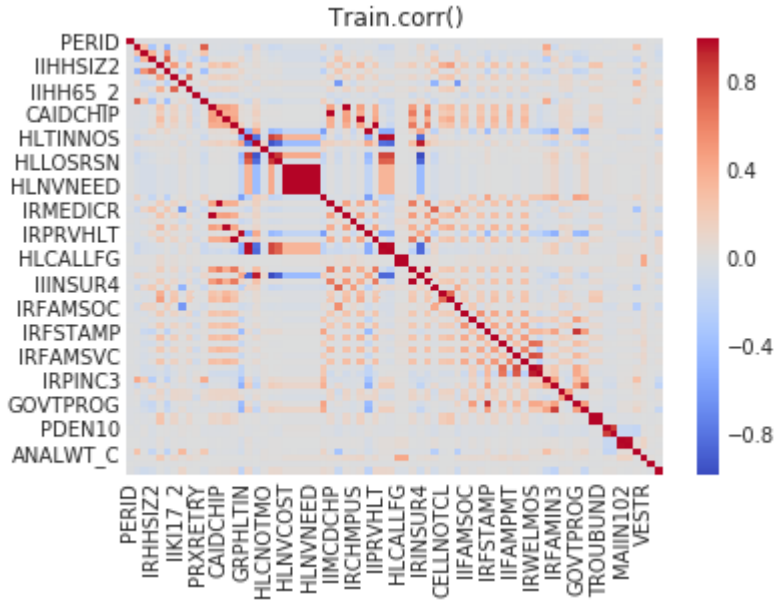
In [38]:
```
sns.distplot(train['VESTR'],bins=30,kde=False,color='red')
```

Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ee5d8c88>



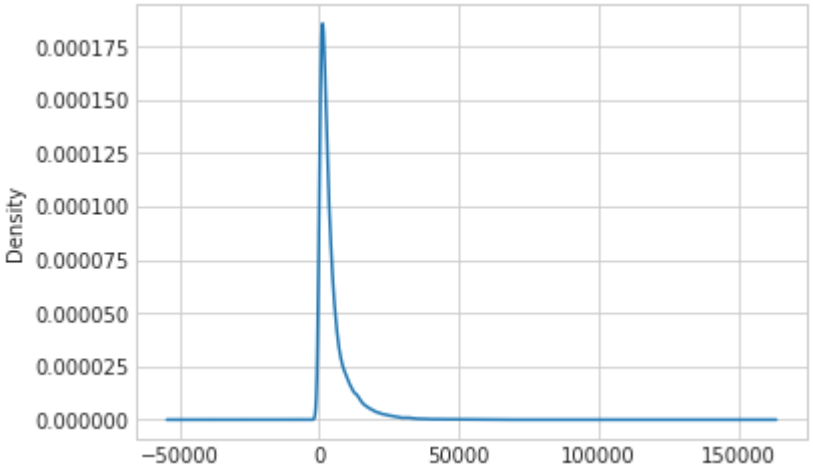
In [39]:
```
sns.heatmap(train.corr(),cmap='coolwarm')
plt.title('Train.corr()')
```

Out[39]: Text(0.5,1,'Train.corr()')



In [40]:
```
train['ANALWT_C'].plot.kde()
```

Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc8ee403048>



Data Cleaning

Drop unnecessary column PERID

In [41]:
```
train.drop("PERID",axis=1,inplace=True)

train.drop("IIHH65_2",axis=1,inplace=True)
test.drop("IIHH65_2",axis=1, inplace=True)
```

In [42]:
```
train.drop("HLCALL99",axis=1,inplace=True)
test.drop("HLCALL99",axis=1, inplace=True)
```

In [43]:
```
#train['ANALWT_C'] = train['ANALWT_C'].astype(int)
#test['ANALWT_C']    = test['ANALWT_C'].astype(int)
```

In [44]:
```
train.drop("IIFSTAMP",axis=1,inplace=True)
test.drop("IIFSTAMP",axis=1, inplace=True)
```

In [ ]:
```

```

In [45]:
```
train.drop("MAIIN102",axis=1,inplace=True)
test.drop("MAIIN102",axis=1, inplace=True)
```

In [46]:
```
train.drop("HLNVREF",axis=1,inplace=True)
test.drop("HLNVREF",axis=1, inplace=True)
```

In [ ]:
```

```

In [47]:
```
train.drop("IIHHSIZ2",axis=1,inplace=True)
test.drop("IIHHSIZ2",axis=1, inplace=True)
```

In [ ]:
```

```

In [53]:
```
train.head()
```

Out[53]:

	IFATHER	NRCH17_2	IRHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	PRXRETRY	PRXYDA
0	4	2	4	3	1	1	99	99
1	4	1	3	2	1	1	99	99
2	4	1	2	2	1	1	99	99
3	4	0	2	1	1	1	99	99
4	1	0	6	4	1	1	99	1

5 rows × 65 columns

In [54]:
```
##

train.loc[ train['VESTR'] <= 40010, 'VESTR'] = 0
train.loc[(train['VESTR'] > 40010) & (train['VESTR'] <= 40025), 'VEST
R'] = 1
train.loc[(train['VESTR'] > 40025) & (train['VESTR'] <= 40040), 'VEST
R'] = 2
train.loc[ train['VESTR'] > 40040, 'VESTR'] = 3
test.loc[ test['VESTR'] <= 40010, 'VESTR'] = 0
test.loc[(test['VESTR'] > 40010) & (test['VESTR'] <= 40025), 'VESTR']
 = 1
test.loc[(test['VESTR'] > 40025) & (test['VESTR'] <= 40040), 'VESTR']
 = 2
test.loc[test['VESTR'] > 40040, 'VESTR'] = 3
```

In [55]:
```
train.head()
```

Out[55]:

	IFATHER	NRCH17_2	IRHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	PRXRETRY	PRXYDA
0	4	2	4	3	1	1	99	99
1	4	1	3	2	1	1	99	99
2	4	1	2	2	1	1	99	99
3	4	0	2	1	1	1	99	99
4	1	0	6	4	1	1	99	1

5 rows × 65 columns

In [ ]:
```

```

In [56]:
```
#'PRXRETRY','PRXYDATA','GRPHLTIN','HLTINNOS','HLCNOTMO','HLCLAST','HL
LOSRSN','HLLOSRSN','HLNVCOST','HLNVCOST','HLNVREF','HLNVNEED','HLNVSO
R','IROTHHLT','HLCALLFG','HLCALL99','IRWELMOS'
```

In [ ]:
```
train.loc[ train['HLCALL99'] <= 1, 'HLCALL99'] = 0
train.loc[(train['HLCALL99'] > 1) & (train['HLCALL99'] <= 10), 'HLCAL
L99'] = 1
train.loc[ train['HLCALL99'] > 10, 'HLCALL99'] = 2
test.loc[ test['HLCALL99'] <= 1, 'HLCALL99'] = 0
test.loc[(test['HLCALL99'] > 1) & (test['HLCALL99'] <= 10), 'HLCALL9
9'] = 1
test.loc[test['HLCALL99'] > 10, 'HLCALL99'] = 2

train.loc[ train['HLCALLFG'] <= 1, 'HLCALLFG'] = 0
train.loc[(train['HLCALLFG'] > 1) & (train['HLCALLFG'] <= 10), 'HLCAL
LFG'] = 1
train.loc[ train['HLCALLFG'] > 10, 'HLCALLFG'] = 2
test.loc[ test['HLCALLFG'] <= 1, 'HLCALLFG'] = 0
test.loc[(test['HLCALLFG'] > 1) & (test['HLCALLFG'] <= 10), 'HLCALLF
G'] = 1
test.loc[test['HLCALLFG'] > 10, 'HLCALLFG'] = 2
```

```
train.loc[ train['IROTHHLT'] <= 1, 'IROTHHLT'] = 0
train.loc[(train['IROTHHLT'] > 1) & (train['IROTHHLT'] <= 10), 'IROTH
HLT'] = 1
train.loc[ train['IROTHHLT'] > 10, 'IROTHHLT'] = 2
test.loc[ test['IROTHHLT'] <= 1, 'IROTHHLT'] = 0
test.loc[(test['IROTHHLT'] > 1) & (test['IROTHHLT'] <= 10), 'IROTHHL
T'] = 1
test.loc[test['IROTHHLT'] > 10, 'IROTHHLT'] = 2


train.loc[ train['HLNVSOR'] <= 1, 'HLNVSOR'] = 0
train.loc[(train['HLNVSOR'] > 1) & (train['HLNVSOR'] <= 10), 'HLNVSO
R'] = 1
train.loc[ train['HLNVSOR'] > 10, 'HLNVSOR'] = 2
test.loc[ test['HLNVSOR'] <= 1, 'HLNVSOR'] = 0
test.loc[(test['HLNVSOR'] > 1) & (test['HLNVSOR'] <= 10), 'HLNVSOR']
= 1
test.loc[test['HLNVSOR'] > 10, 'HLNVSOR'] = 2

train.loc[ train['HLNVNEED'] <= 1, 'HLNVNEED'] = 0
train.loc[(train['HLNVNEED'] > 1) & (train['HLNVNEED'] <= 10), 'HLNVN
EED'] = 1
train.loc[ train['HLNVNEED'] > 10, 'HLNVNEED'] = 2
test.loc[ test['HLNVNEED'] <= 1, 'HLNVNEED'] = 0
test.loc[(test['HLNVNEED'] > 1) & (test['HLNVNEED'] <= 10), 'HLNVNEE
D'] = 1
test.loc[test['HLNVNEED'] > 10, 'HLNVNEED'] = 2

train.loc[ train['HLNVREF'] <= 1, 'HLNVREF'] = 0
train.loc[(train['HLNVREF'] > 1) & (train['HLNVREF'] <= 10), 'HLNVRE
F'] = 1
train.loc[ train['HLNVREF'] > 10, 'HLNVREF'] = 2
test.loc[ test['HLNVREF'] <= 1, 'HLNVREF'] = 0
test.loc[(test['HLNVREF'] > 1) & (test['HLNVREF'] <= 10), 'HLNVREF']
= 1
test.loc[test['HLNVREF'] > 10, 'HLNVREF'] = 2

train.loc[ train['HLNVCOST'] <= 1, 'HLNVCOST'] = 0
train.loc[(train['HLNVCOST'] > 1) & (train['HLNVCOST'] <= 10), 'HLNVC
OST'] = 1
train.loc[ train['HLNVCOST'] > 10, 'HLNVCOST'] = 2
test.loc[ test['HLNVCOST'] <= 1, 'HLNVCOST'] = 0
test.loc[(test['HLNVCOST'] > 1) & (test['HLNVCOST'] <= 10), 'HLNVCOS
T'] = 1
test.loc[test['HLNVCOST'] > 10, 'HLNVCOST'] = 2

train.loc[ train['HLLOSRSN'] <= 1, 'HLLOSRSN'] = 0
train.loc[(train['HLLOSRSN'] > 1) & (train['HLLOSRSN'] <= 10), 'HLLOS
RSN'] = 1
train.loc[ train['HLLOSRSN'] > 10, 'HLLOSRSN'] = 2
test.loc[ test['HLLOSRSN'] <= 1, 'HLLOSRSN'] = 0
test.loc[(test['HLLOSRSN'] > 1) & (test['HLLOSRSN'] <= 10), 'HLLOSRS
N'] = 1
test.loc[test['HLLOSRSN'] > 10, 'HLLOSRSN'] = 2
```

```
train.loc[ train['HLCLAST'] <= 1, 'HLCLAST'] = 0
train.loc[(train['HLCLAST'] > 1) & (train['HLCLAST'] <= 10), 'HLCLAS
T'] = 1
train.loc[ train['HLCLAST'] > 10, 'HLCLAST'] = 2
test.loc[ test['HLCLAST'] <= 1, 'HLCLAST'] = 0
test.loc[(test['HLCLAST'] > 1) & (test['HLCLAST'] <= 10), 'HLCLAST']
= 1
test.loc[test['HLCLAST'] > 10, 'HLCLAST'] = 2

train.loc[ train['HLCNOTMO'] <= 1, 'HLCNOTMO'] = 0
train.loc[(train['HLCNOTMO'] > 1) & (train['HLCNOTMO'] <= 10), 'HLCNO
TMO'] = 1
train.loc[ train['HLCNOTMO'] > 10, 'HLCNOTMO'] = 2
test.loc[ test['HLCNOTMO'] <= 1, 'HLCNOTMO'] = 0
test.loc[(test['HLCNOTMO'] > 1) & (test['HLCNOTMO'] <= 10), 'HLCNOTM
O'] = 1
test.loc[test['HLCNOTMO'] > 10, 'HLCNOTMO'] = 2

train.loc[ train['PRXRETRY'] <= 1, 'PRXRETRY'] = 0
train.loc[(train['PRXRETRY'] > 1) & (train['PRXRETRY'] <= 10), 'PRXRE
TRY'] = 1
train.loc[ train['PRXRETRY'] > 10, 'PRXRETRY'] = 2
test.loc[ test['PRXRETRY'] <= 1, 'PRXRETRY'] = 0
test.loc[(test['PRXRETRY'] > 1) & (test['PRXRETRY'] <= 10), 'PRXRETR
Y'] = 1
test.loc[test['PRXRETRY'] > 10, 'PRXRETRY'] = 2

train.loc[ train['PRXYDATA'] <= 1, 'PRXYDATA'] = 0
train.loc[(train['PRXYDATA'] > 1) & (train['PRXYDATA'] <= 10), 'PRXYD
ATA'] = 1
train.loc[ train['PRXYDATA'] > 10, 'PRXYDATA'] = 2
test.loc[ test['PRXYDATA'] <= 1, 'PRXYDATA'] = 0
test.loc[(test['PRXYDATA'] > 1) & (test['PRXYDATA'] <= 10), 'PRXYDAT
A'] = 1
test.loc[test['PRXYDATA'] > 10, 'PRXYDATA'] = 2

train.loc[ train['GRPHLTIN'] <= 1, 'GRPHLTIN'] = 0
train.loc[(train['GRPHLTIN'] > 1) & (train['GRPHLTIN'] <= 10), 'GRPHL
TIN'] = 1
train.loc[ train['GRPHLTIN'] > 10, 'GRPHLTIN'] = 2
test.loc[ test['GRPHLTIN'] <= 1, 'GRPHLTIN'] = 0
test.loc[(test['GRPHLTIN'] > 1) & (test['GRPHLTIN'] <= 10), 'GRPHLTI
N'] = 1
test.loc[test['GRPHLTIN'] > 10, 'GRPHLTIN'] = 2

train.loc[ train['HLTINNOS'] <= 1, 'HLTINNOS'] = 0
train.loc[(train['HLTINNOS'] > 1) & (train['HLTINNOS'] <= 10), 'HLTIN
NOS'] = 1
train.loc[ train['HLTINNOS'] > 10, 'HLTINNOS'] = 2
test.loc[ test['HLTINNOS'] <= 1, 'HLTINNOS'] = 0
test.loc[(test['HLTINNOS'] > 1) & (test['HLTINNOS'] <= 10), 'HLTINNO
S'] = 1
test.loc[test['HLTINNOS'] > 10, 'HLTINNOS'] = 2

train.loc[ train['IRWELMOS'] <= 1, 'IRWELMOS'] = 0
train.loc[(train['IRWELMOS'] > 1) & (train['IRWELMOS'] <= 10), 'IRWEL
MOS'] = 1
```

```
train.loc[ train['IRWELMOS'] > 10, 'IRWELMOS'] = 2
test.loc[ test['IRWELMOS'] <= 1, 'IRWELMOS'] = 0
test.loc[(test['IRWELMOS'] > 1) & (test['IRWELMOS'] <= 10), 'IRWELMO
S'] = 1
test.loc[test['IRWELMOS'] > 10, 'IRWELMOS'] = 2
```

In [59]:
```
train.loc[ train['ANALWT_C'] <= 10000, 'ANALWT_C'] = 0
train.loc[(train['ANALWT_C'] > 10000) & (train['ANALWT_C'] <= 20000),
 'ANALWT_C'] = 1
train.loc[(train['ANALWT_C'] > 20000) & (train['ANALWT_C'] <= 30000),
 'ANALWT_C'] = 2
train.loc[(train['ANALWT_C'] > 30000) & (train['ANALWT_C'] <= 40000),
 'ANALWT_C'] = 3
train.loc[(train['ANALWT_C'] > 40000) & (train['ANALWT_C'] <= 50000),
 'ANALWT_C'] = 4
train.loc[(train['ANALWT_C'] > 50000) & (train['ANALWT_C'] <= 60000),
 'ANALWT_C'] = 5
train.loc[(train['ANALWT_C'] > 60000) & (train['ANALWT_C'] <= 70000),
 'ANALWT_C'] = 6
train.loc[(train['ANALWT_C'] > 70000) & (train['ANALWT_C'] <= 80000),
 'ANALWT_C'] = 7
train.loc[(train['ANALWT_C'] > 80000) & (train['ANALWT_C'] <= 90000),
 'ANALWT_C'] = 8
train.loc[ train['ANALWT_C'] > 90000, 'ANALWT_C'] = 9


test.loc[ test['ANALWT_C'] <= 10000, 'ANALWT_C'] = 0
test.loc[(test['ANALWT_C'] > 10000) & (test['ANALWT_C'] <= 20000), 'A
NALWT_C'] = 1
test.loc[(test['ANALWT_C'] > 20000) & (test['ANALWT_C'] <= 30000), 'A
NALWT_C'] = 2
test.loc[(test['ANALWT_C'] > 30000) & (test['ANALWT_C'] <= 40000), 'A
NALWT_C'] = 3
test.loc[(test['ANALWT_C'] > 40000) & (test['ANALWT_C'] <= 50000), 'A
NALWT_C'] = 4
test.loc[(test['ANALWT_C'] > 50000) & (test['ANALWT_C'] <= 60000), 'A
NALWT_C'] = 5
test.loc[(test['ANALWT_C'] > 60000) & (test['ANALWT_C'] <= 70000), 'A
NALWT_C'] = 6
test.loc[(test['ANALWT_C'] > 70000) & (test['ANALWT_C'] <= 80000), 'A
NALWT_C'] = 7
test.loc[(test['ANALWT_C'] > 80000) & (test['ANALWT_C'] <= 90000), 'A
NALWT_C'] = 8
test.loc[test['ANALWT_C'] > 90000, 'ANALWT_C'] = 9
```

In [60]:
```
train.head()
```

Out[60]:

	IFATHER	NRCH17_2	IRHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	PRXRETRY	PRXYDA
0	4	2	4	3	1	1	99	99
1	4	1	3	2	1	1	99	99
2	4	1	2	2	1	1	99	99
3	4	0	2	1	1	1	99	99
4	1	0	6	4	1	1	99	1

5 rows × 65 columns

In [61]:
```
test.head()
```

Out[61]:

	PERID	IFATHER	NRCH17_2	IRHHSIZ2	IRKI17_2	IIKI17_2	IRHH65_2	PRXRETRY
0	66583679	4	0	4	2	1	1	99
1	35494679	4	0	4	1	1	1	99
2	79424679	2	0	3	2	1	1	99
3	11744679	4	0	6	2	1	1	99
4	31554679	1	0	4	3	1	1	99

5 rows × 65 columns

In [ ]:
```
```

In [ ]:
```
```

In [62]:
```
from sklearn.model_selection import train_test_split
X = train.drop('Criminal',axis=1)
y = train['Criminal']
```

In [63]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=
0.50)
```

In [64]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=100)
rfc.fit(X_train, y_train)
rfc_pred = rfc.predict(X_test)
rfc.score(X_train, y_train)
```

Out[64]: 0.99566910188547175

In [65]:
```
from sklearn.metrics import classification_report,confusion_matrix
print(classification_report(y_test,rfc_pred))
```

```
             precision    recall  f1-score   support

          0       0.97      0.98      0.97     21255
          1       0.66      0.55      0.60      1604

avg / total       0.94      0.95      0.95     22859
```

In [ ]:

In [ ]:

In [ ]:

Building a Model

Train-Test Split

Split the data into Training testing set

In [66]:
```
X_train = train.drop('Criminal', axis=1)
y_train = train['Criminal']
X_test = test.drop('PERID', axis=1)
```

In [67]:
```
# Logistic Regression
from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression()
logmodel.fit(X_train,y_train)
predictions = logmodel.predict(X_test)
logmodel.score(X_train,y_train)
```

Out[67]: 0.93833938492497482

Random Forest

Training and Predicting

We'll start training using Random Forest.

In [68]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [69]:
```
random_forest = RandomForestClassifier(n_estimators=150)
random_forest.fit(X_train, y_train)
```

Out[69]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion
='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=150, n_jobs=1,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [70]:
```
RFC_prediction = random_forest.predict(X_test)
```

In [71]:
```
random_forest.score(X_train, y_train)
```

Out[71]:
```
0.99400673695262265
```

In [72]:
```
random_forest.score(X_train, y_train)
```

Out[72]:
```
0.99400673695262265
```

Result file into .csv

In [ ]:
```
submission = pd.DataFrame({
    "PERID": test["PERID"],
    "Criminal": RFC_prediction,
    })
submission.to_csv('Result2.csv', index=False, columns=['PERID', 'Crim
inal'])
```

In [ ]:
```
result = pd.read_csv('Result.csv')
result.head()
```

In [ ]:
```

```