Adult Census Income

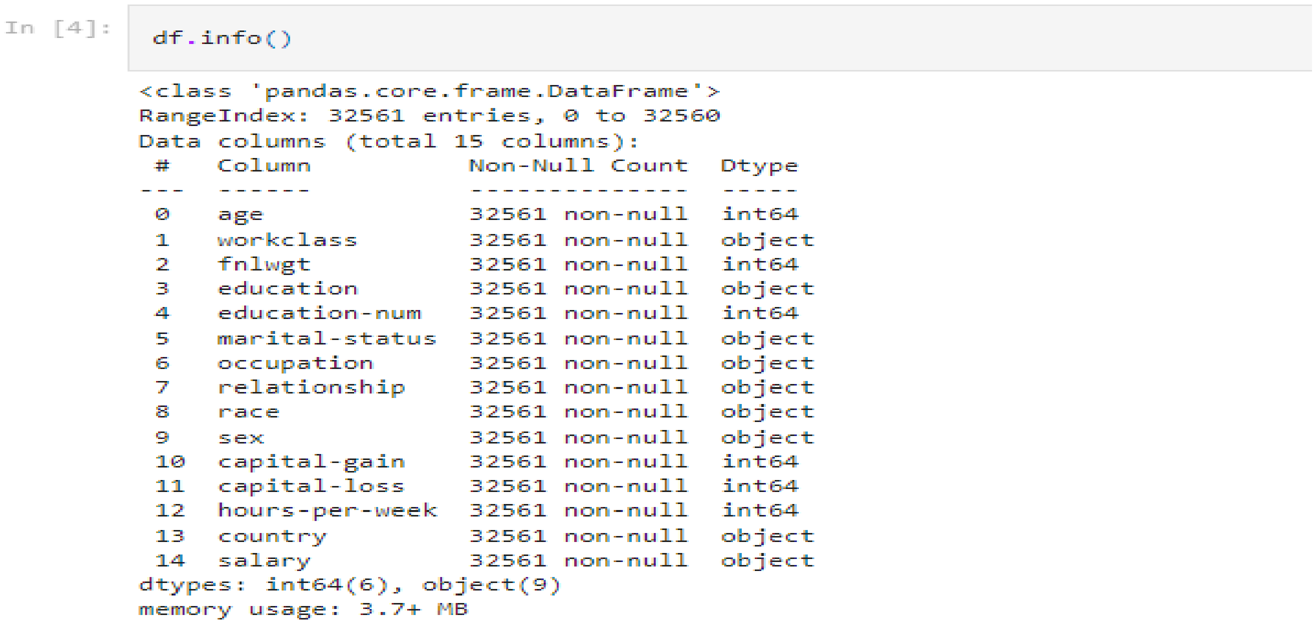
Wireframe Documentation

Homepage

1 Data Preparation

We have to find null values , Outliers , Categorical features: -

1.1. We find null values present in dataset or not

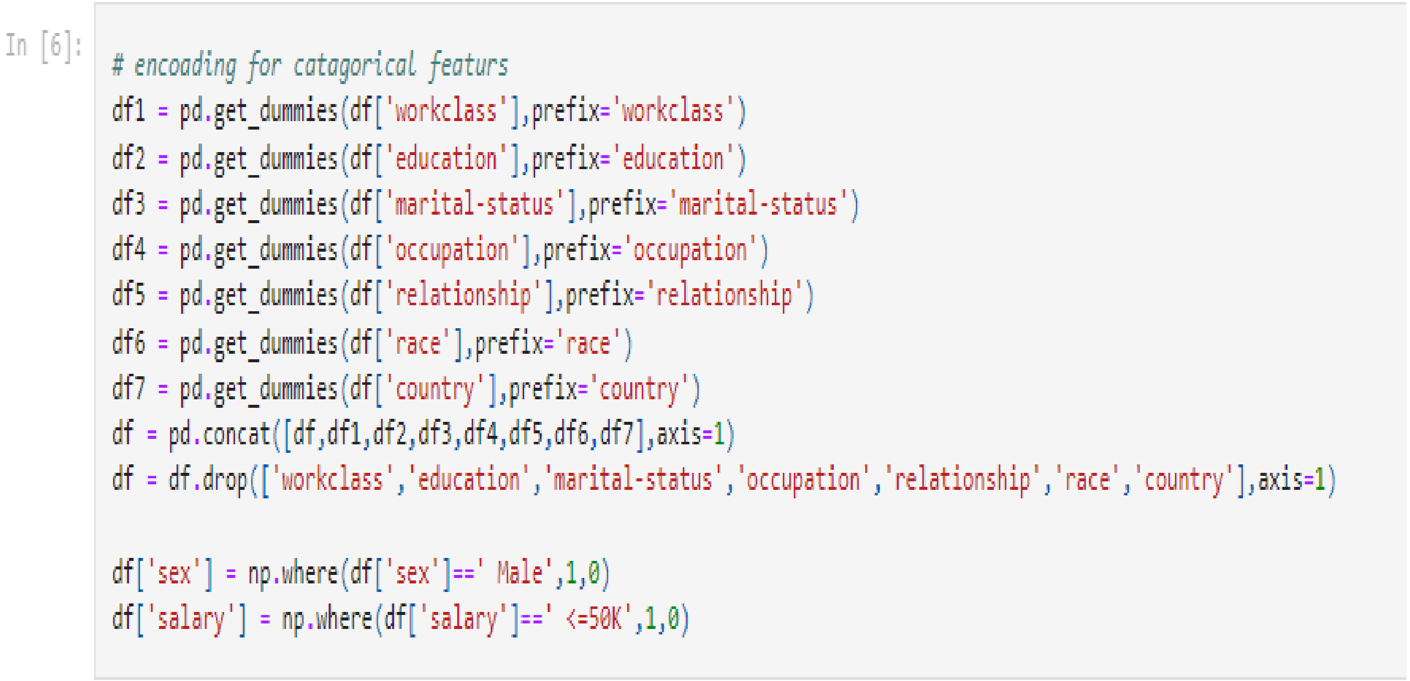


```
In [4]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   age             32561 non-null  int64 
 1   workclass       32561 non-null  object
 2   fnlwgt          32561 non-null  int64 
 3   education       32561 non-null  object
 4   education-num   32561 non-null  int64 
 5   marital-status  32561 non-null  object
 6   occupation      32561 non-null  object
 7   relationship    32561 non-null  object
 8   race            32561 non-null  object
 9   sex             32561 non-null  object
 10  capital-gain    32561 non-null  int64 
 11  capital-loss    32561 non-null  int64 
 12  hours-per-week  32561 non-null  int64 
 13  country         32561 non-null  object
 14  salary          32561 non-null  object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

There is no any null values

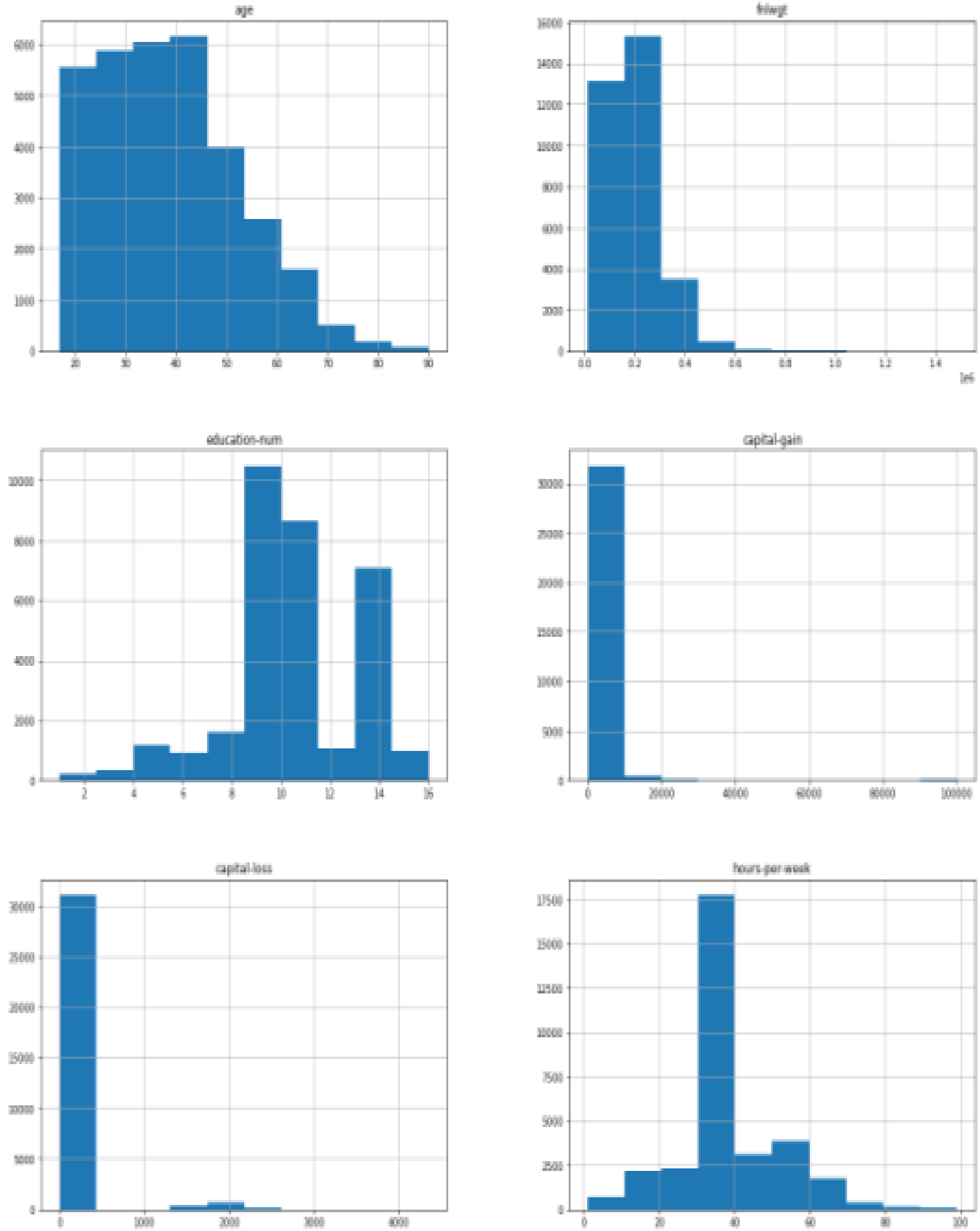
1.2. We find Categorical features present in dataset and applying encoding .



```
In [6]: # encoading for catagorical featurs
df1 = pd.get_dummies(df['workclass'],prefix='workclass')
df2 = pd.get_dummies(df['education'],prefix='education')
df3 = pd.get_dummies(df['marital-status'],prefix='marital-status')
df4 = pd.get_dummies(df['occupation'],prefix='occupation')
df5 = pd.get_dummies(df['relationship'],prefix='relationship')
df6 = pd.get_dummies(df['race'],prefix='race')
df7 = pd.get_dummies(df['country'],prefix='country')
df = pd.concat([df,df1,df2,df3,df4,df5,df6,df7],axis=1)
df = df.drop(['workclass','education','marital-status','occupation','relationship','race','country'],axis=1)

df['sex'] = np.where(df['sex']==' Male',1,0)
df['salary'] = np.where(df['salary']==' <=50K',1,0)
```

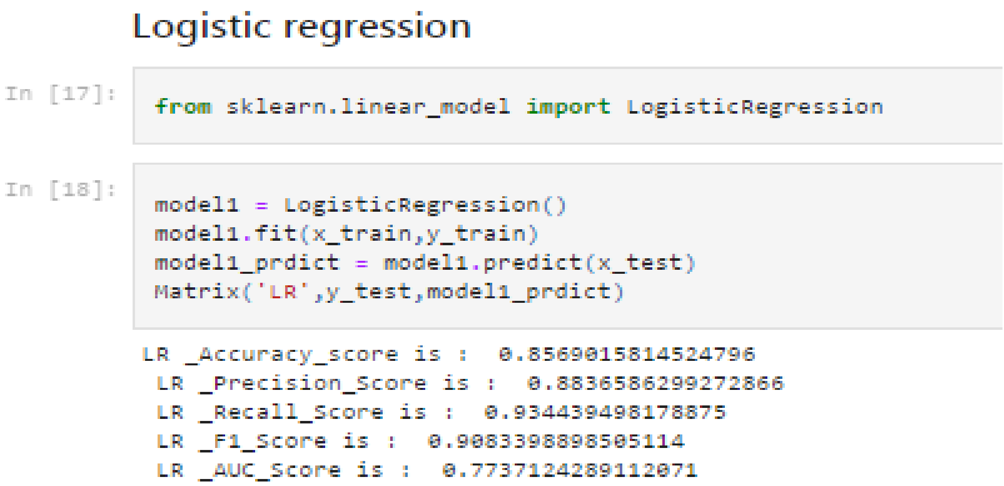
1.3. We find Outliers present in dataset or not



There is no any outlier

2 Model Building

2.1 Logistic regression



Logistic regression

In [17]:

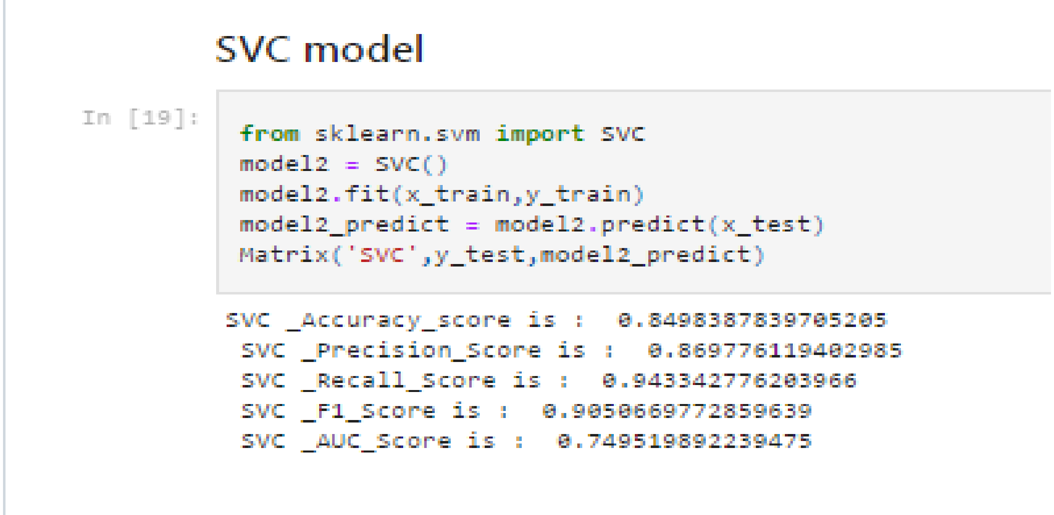
```
from sklearn.linear_model import LogisticRegression
```

In [18]:

```
model1 = LogisticRegression()
model1.fit(x_train,y_train)
model1_prdict = model1.predict(x_test)
Matrix('LR',y_test,model1_prdict)
```

```
LR _Accuracy_score is :  0.8569015814524796
 LR _Precision_Score is :  0.8836586299272866
 LR _Recall_Score is :  0.934439498178875
 LR _F1_Score is :  0.9083398898505114
 LR _AUC_Score is :  0.7737124289112071
```

2.2 SVC model



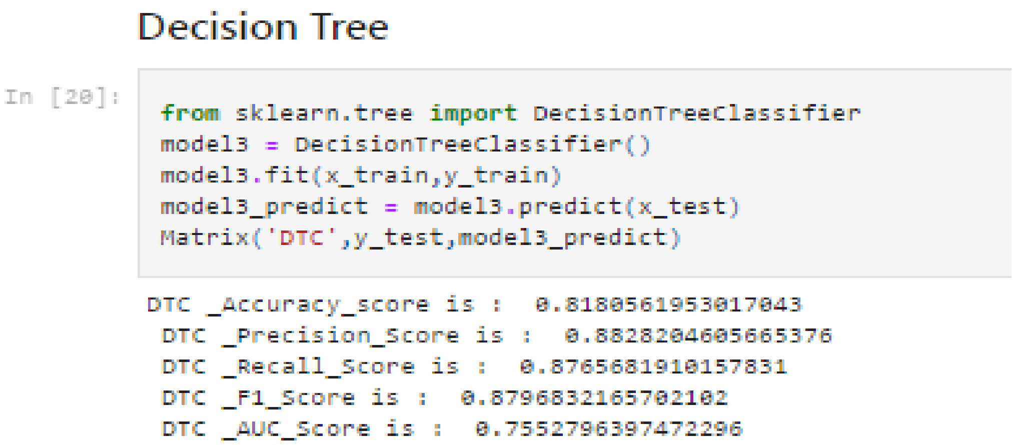
SVC model

In [19]:

```
from sklearn.svm import SVC
model2 = SVC()
model2.fit(x_train,y_train)
model2_predict = model2.predict(x_test)
Matrix('SVC',y_test,model2_predict)
```

```
SVC _Accuracy_score is :  0.8498387839705205
 SVC _Precision_Score is :  0.869776119402985
 SVC _Recall_Score is :  0.943342776203966
 SVC _F1_Score is :  0.9050669772859639
 SVC _AUC_Score is :  0.749519892239475
```

2.3 Decision Tree



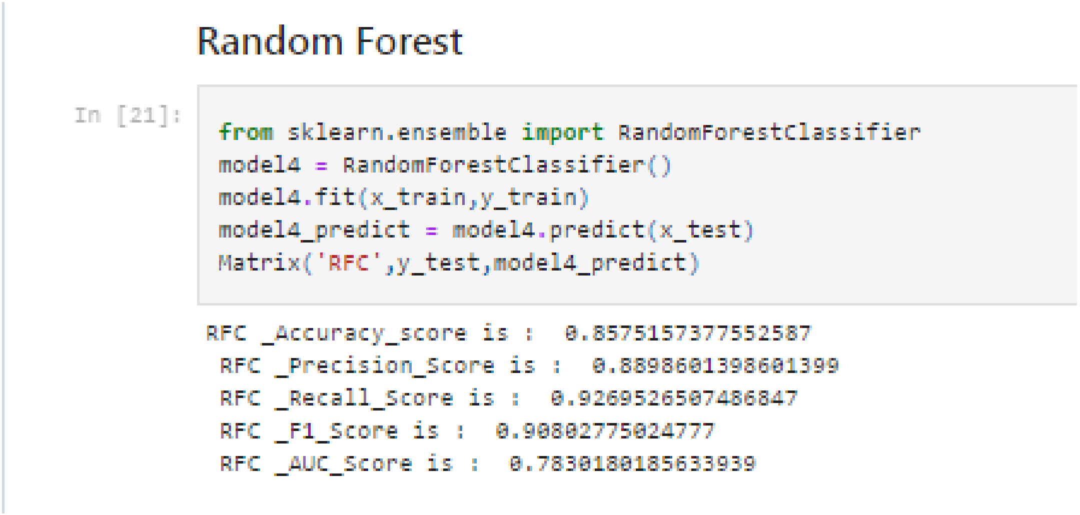
Decision Tree

In [20]:

```
from sklearn.tree import DecisionTreeClassifier
model3 = DecisionTreeClassifier()
model3.fit(x_train,y_train)
model3_predict = model3.predict(x_test)
Matrix('DTC',y_test,model3_predict)
```

```
DTC _Accuracy_score is :  0.8180561953017043
 DTC _Precision_Score is :  0.8828204605665376
 DTC _Recall_Score is :  0.8765681910157831
 DTC _F1_Score is :  0.8796832165702102
 DTC _AUC_Score is :  0.7552796397472296
```

2.4 Random Forest

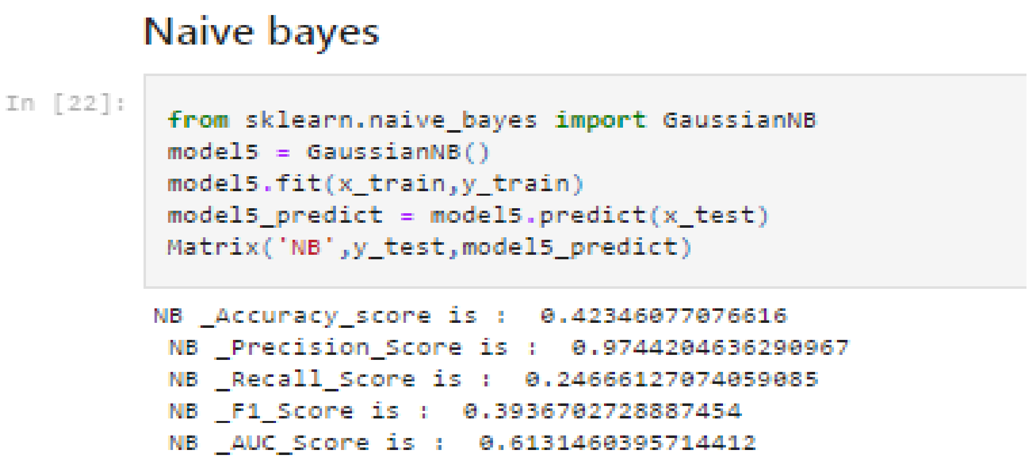


Random Forest

```
In [21]:
from sklearn.ensemble import RandomForestClassifier
model4 = RandomForestClassifier()
model4.fit(x_train,y_train)
model4_predict = model4.predict(x_test)
Matrix('RFC',y_test,model4_predict)
```

```
RFC _Accuracy_score is :  0.8575157377552587
 RFC _Precision_Score is :  0.8898601398601399
 RFC _Recall_Score is :  0.9269526507486847
 RFC _F1_Score is :  0.90802775024777
 RFC _AUC_Score is :  0.7830180185633939
```

2.5 Naive bayes

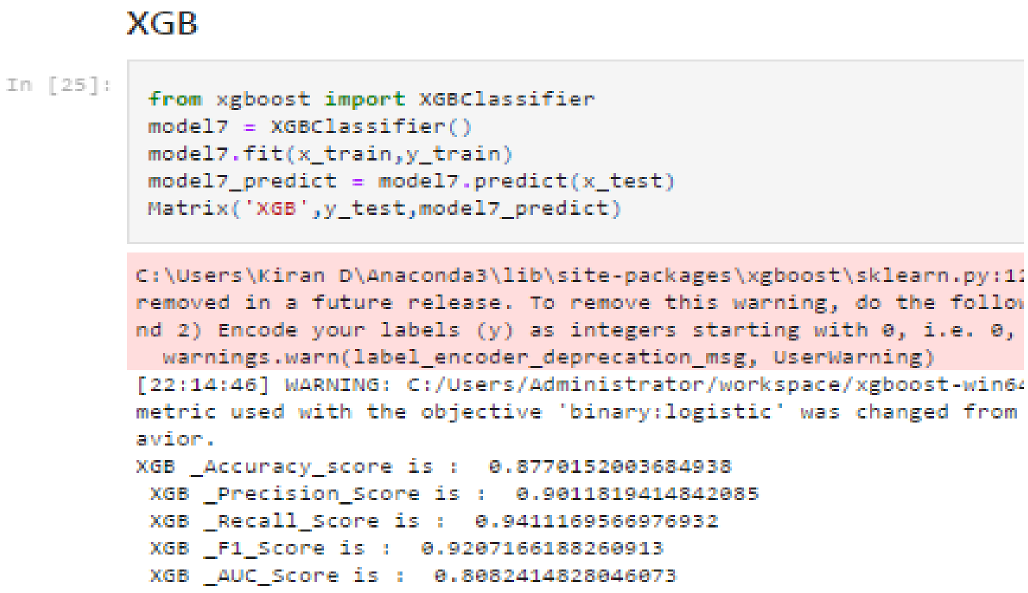


Naive bayes

```
In [22]:
from sklearn.naive_bayes import GaussianNB
model5 = GaussianNB()
model5.fit(x_train,y_train)
model5_predict = model5.predict(x_test)
Matrix('NB',y_test,model5_predict)
```

```
NB _Accuracy_score is :  0.42346077076616
 NB _Precision_Score is :  0.9744204636290967
 NB _Recall_Score is :  0.24666127074059085
 NB _F1_Score is :  0.3936702728887454
 NB _AUC_Score is :  0.6131460395714412
```

2.6 XGB

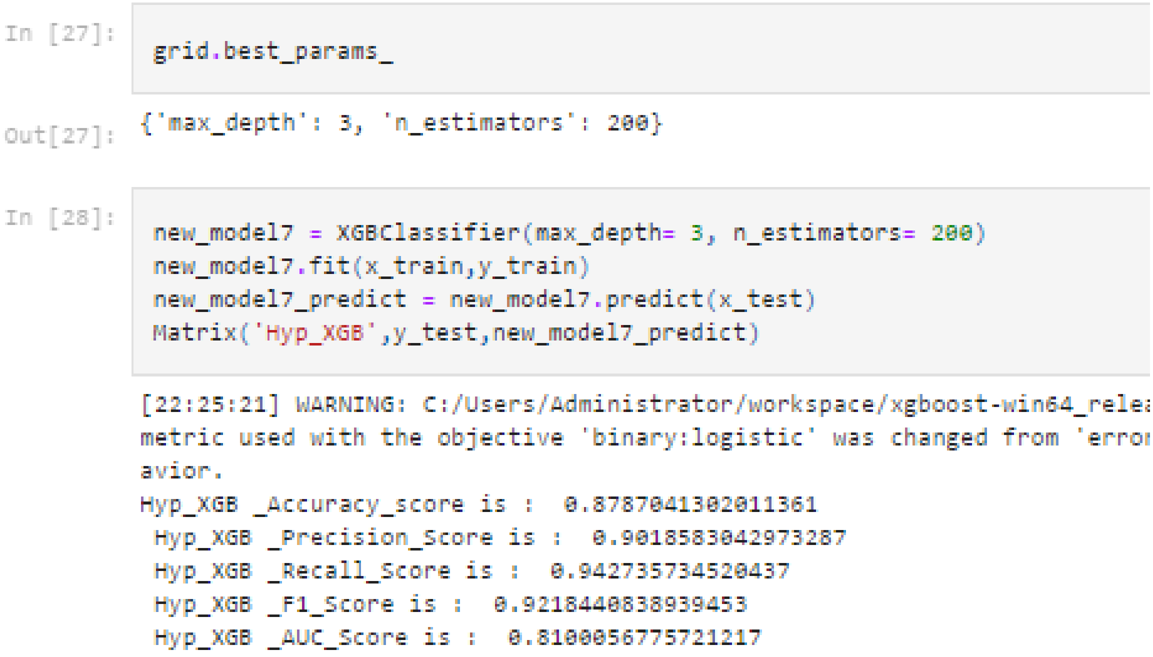


XGB

```
In [25]:
from xgboost import XGBClassifier
model7 = XGBClassifier()
model7.fit(x_train,y_train)
model7_predict = model7.predict(x_test)
Matrix('XGB',y_test,model7_predict)
```

```
C:\Users\Kiran D\Anaconda3\lib\site-packages\xgboost\sklearn.py:1
removed in a future release. To remove this warning, do the follo
nd 2) Encode your labels (y) as integers starting with 0, i.e. 0,
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[22:14:46] WARNING: C:/Users/Administrator/workspace/xgboost-win6
metric used with the objective 'binary:logistic' was changed from
avior.
XGB _Accuracy_score is :  0.8770152003684938
 XGB _Precision_Score is :  0.9011819414842085
 XGB _Recall_Score is :  0.9411169566976932
 XGB _F1_Score is :  0.9207166188260913
 XGB _AUC_Score is :  0.8082414828046073
```

2.7 Hyperparameter XGB



```
In [27]:  grid.best_params_

Out[27]:  {'max_depth': 3, 'n_estimators': 200}
```

```
In [28]:  new_model7 = XGBClassifier(max_depth= 3, n_estimators= 200)
          new_model7.fit(x_train,y_train)
          new_model7_predict = new_model7.predict(x_test)
          Matrix('Hyp_XGB',y_test,new_model7_predict)

          [22:25:21] WARNING: C:/Users/Administrator/workspace/xgboost-win64_rele
          metric used with the objective 'binary:logistic' was changed from 'erro
          avior.
          Hyp_XGB _Accuracy_score is :  0.8787041302011361
           Hyp_XGB _Precision_Score is :  0.9018583042973287
           Hyp_XGB _Recall_Score is :  0.942735734520437
           Hyp_XGB _F1_Score is :  0.9218440838939453
           Hyp_XGB _AUC_Score is :  0.8100056775721217
```

3 Model Selection

With high accuracy score we select Hyperparameter XGB model