Adaboosting Algoritham

Classification Census Dataset

Dataset: https://archive.ics.uci.edu/ml/datasets/census+income (https://archive.ics.uci.edu/ml/datasets/census+income)

For EDA an FE Please watch this post: https://www.linkedin.com/posts/murali-divya-teja-gummadidala-machine-learning_logisticsvm-assignment-activity-6993985726476378112-XTwa?utm_source=share&utm_medium=member_desktop (https://www.linkedin.com/posts/murali-divya-teja-gummadidala-machine-learning_logisticsvm-assignment-activity-6993985726476378112-XTwa?utm_source=share&utm_medium=member_desktop)

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [3]:
```
df = pd.read_csv(r'C:\Users\HP\Documents\ML-Assignment\ML-assingment\Classificati
df.head()
```

Out[3]:

	Unnamed: 0	age	workclass	fnlwgt	education	education-num	marital-status	occupation	capit g
0	0	3.663562	0.039864	11.258240	0	13	0.328092	0.115783	0.7347
1	1	3.912023	0.078038	11.330336	0	13	0.459937	0.124873	-0.0000
2	2	3.637586	0.753417	12.281393	1	9	0.136452	0.042075	-0.0000
3	3	3.970292	0.753417	12.366153	2	7	0.459937	0.042075	-0.0000
4	4	3.332205	0.753417	12.732011	0	13	0.459937	0.183747	-0.0000

5 rows × 23 columns

In [4]:
```
df.columns
```

Out[4]:
```
Index(['Unnamed: 0', 'age', 'workclass', 'fnlwgt', 'education',
       'education-num', 'marital-status', 'occupation', 'capital-gain',
       'capital-loss', 'hours-per-week', 'native-country', 'salary',
       'sex_ Male', 'race_ Asian-Pac-Islander', 'race_ Black', 'race_ Other',
       'race_ White', 'relationship_ Not-in-family',
       'relationship_ Other-relative', 'relationship_ Own-child',
       'relationship_ Unmarried', 'relationship_ Wife'],
      dtype='object')
```

In [5]:
```
df.drop(['Unnamed: 0'],axis=1,inplace= True)
```

In [6]:
```
df.head()
```

Out[6]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	capital-gain	capital-loss
0	3.663562	0.039864	11.258240	0	13	0.328092	0.115783	0.734717	-0.0
1	3.912023	0.078038	11.330336	0	13	0.459937	0.124873	-0.000000	-0.0
2	3.637586	0.753417	12.281393	1	9	0.136452	0.042075	-0.000000	-0.0
3	3.970292	0.753417	12.366153	2	7	0.459937	0.042075	-0.000000	-0.0
4	3.332205	0.753417	12.732011	0	13	0.459937	0.183747	-0.000000	-0.0

5 rows × 22 columns

In [7]:
```
df.isnull().sum()
```

Out[7]:
```
age                          0
workclass                    0
fnlwgt                       0
education                    0
education-num                0
marital-status               0
occupation                   0
capital-gain                 0
capital-loss                 0
hours-per-week               0
native-country               0
salary                       0
sex_ Male                    0
race_ Asian-Pac-Islander     0
race_ Black                  0
race_ Other                  0
race_ White                  0
relationship_ Not-in-family  0
relationship_ Other-relative 0
relationship_ Own-child      0
relationship_ Unmarried      0
relationship_ Wife           0
dtype: int64
```

In [8]:
```
df.describe().T
```

Out[8]:

	count	mean	std	min	25%	50%	75%	max
age	32561.0	3.589360	0.360379	2.833213	3.332205	3.610918	3.871201	4.499810
workclass	32561.0	0.581492	0.300678	0.000215	0.753417	0.753417	0.753417	0.753417
fnlwgt	32561.0	11.986407	0.583247	10.583043	11.676973	12.091537	12.376031	12.847089
education	32561.0	3.424465	3.453582	0.000000	1.000000	2.000000	5.000000	15.000000
education-num	32561.0	10.080679	2.572720	1.000000	9.000000	10.000000	12.000000	16.000000
marital-status	32561.0	0.339891	0.140320	0.000706	0.328092	0.328092	0.459937	0.459937
occupation	32561.0	0.111534	0.046773	0.000276	0.101195	0.115783	0.125887	0.183747
capital-gain	32561.0	0.061195	0.203023	-0.000000	0.000000	0.000000	0.000000	0.734738
capital-loss	32561.0	0.016342	0.073877	-0.000000	0.000000	0.000000	0.000000	0.350305
hours-per-week	32561.0	40.437456	12.347429	1.000000	40.000000	40.000000	45.000000	99.000000
native-country	32561.0	0.835528	0.254674	0.000031	0.913762	0.913762	0.913762	0.913762
salary	32561.0	0.240810	0.427581	0.000000	0.000000	0.000000	0.000000	1.000000
sex_ Male	32561.0	0.669205	0.470506	0.000000	0.000000	1.000000	1.000000	1.000000
race_ Asian-Pac-Islander	32561.0	0.031909	0.175761	0.000000	0.000000	0.000000	0.000000	1.000000
race_ Black	32561.0	0.095943	0.294518	0.000000	0.000000	0.000000	0.000000	1.000000
race_ Other	32561.0	0.008323	0.090851	0.000000	0.000000	0.000000	0.000000	1.000000
race_ White	32561.0	0.854274	0.352837	0.000000	1.000000	1.000000	1.000000	1.000000
relationship_ Not-in-family	32561.0	0.255060	0.435901	0.000000	0.000000	0.000000	1.000000	1.000000
relationship_ Other-relative	32561.0	0.030128	0.170942	0.000000	0.000000	0.000000	0.000000	1.000000
relationship_ Own-child	32561.0	0.155646	0.362525	0.000000	0.000000	0.000000	0.000000	1.000000
relationship_ Unmarried	32561.0	0.105832	0.307627	0.000000	0.000000	0.000000	0.000000	1.000000
relationship_ Wife	32561.0	0.048156	0.214099	0.000000	0.000000	0.000000	0.000000	1.000000

In [9]:
```
X = df.drop(['salary'],axis=1)
X.head()
```

Out[9]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	capital-gain	capital-loss
0	3.663562	0.039864	11.258240	0	13	0.328092	0.115783	0.734717	-0.0
1	3.912023	0.078038	11.330336	0	13	0.459937	0.124873	-0.000000	-0.0
2	3.637586	0.753417	12.281393	1	9	0.136452	0.042075	-0.000000	-0.0
3	3.970292	0.753417	12.366153	2	7	0.459937	0.042075	-0.000000	-0.0
4	3.332205	0.753417	12.732011	0	13	0.459937	0.183747	-0.000000	-0.0

5 rows × 21 columns

In [10]:
```
X.columns
```

Out[10]:
```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',
       'marital-status', 'occupation', 'capital-gain', 'capital-loss',
       'hours-per-week', 'native-country', 'sex_ Male',
       'race_ Asian-Pac-Islander', 'race_ Black', 'race_ Other', 'race_ White',
       'relationship_ Not-in-family', 'relationship_ Other-relative',
       'relationship_ Own-child', 'relationship_ Unmarried',
       'relationship_ Wife'],
      dtype='object')
```

In [11]:
```
y = df['salary']
y.head()
```

Out[11]:
```
0    0
1    0
2    0
3    0
4    0
Name: salary, dtype: int64
```

In [12]:
```
from sklearn.model_selection import train_test_split
```

In [13]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.33,random_state=
```

In [14]:
```
X_train.shape
```

Out[14]:
```
(21815, 21)
```

In [15]:
```
y_train.shape
```

Out[15]:
```
(21815,)
```

In [16]:
```
from sklearn.ensemble import AdaBoostClassifier
```

In [17]:
```
adb = AdaBoostClassifier()
```

In [18]:
```
adb.fit(X_train,y_train)
```

Out[18]:
```
AdaBoostClassifier
AdaBoostClassifier()
```

In [19]:
```
adb.score(X_train,y_train)
```

Out[19]:
```
0.855787302314921
```

In [20]:
```
y_prd_adb = adb.predict(X_test)
y_prd_adb
```

Out[20]:
```
array([0, 0, 1, ..., 0, 0, 0], dtype=int64)
```

In [21]:
```
from sklearn.metrics import confusion_matrix, classification_report, accuracy_sc
```

In [22]:
```
cm = confusion_matrix(y_test,y_prd_adb)
```

In [23]:
```
cm
```

Out[23]:
```
array([[7719,  477],
       [1016, 1534]], dtype=int64)
```

In [24]:
```
print(classification_report(y_test,y_prd_adb))
```

```
              precision    recall  f1-score   support

           0       0.88      0.94      0.91      8196
           1       0.76      0.60      0.67      2550

    accuracy                           0.86     10746
   macro avg       0.82      0.77      0.79     10746
weighted avg       0.86      0.86      0.86     10746
```

In [25]:
```
print(accuracy_score(y_test,y_prd_adb))
```

```
0.8610645821701098
```

In [26]:
```
from sklearn.metrics import roc_auc_score,roc_curve
```

In [27]:
```
y_prd_adb_prb = adb.predict_proba(X_test)[:,1]
y_prd_adb_prb
```

Out[27]:
```
array([0.47908405, 0.49763667, 0.5024228 , ..., 0.49157928, 0.49773317,
       0.4741657 ])
```

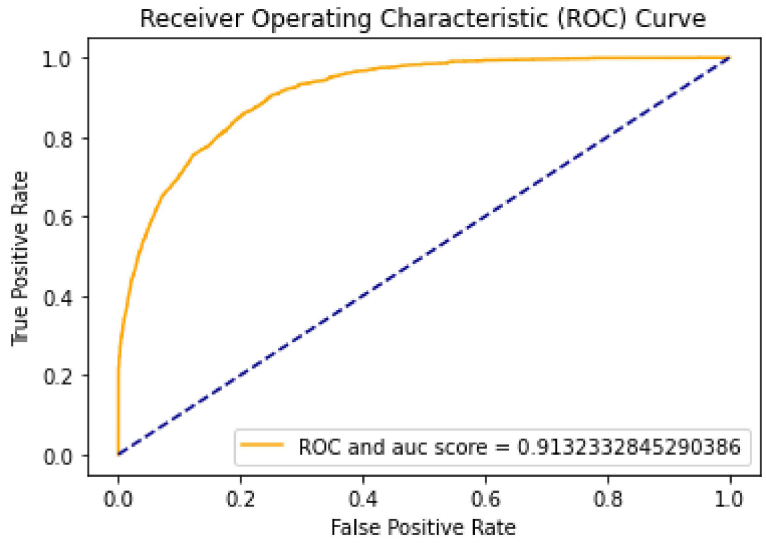
In [28]:
```
fpr_a,tpr_a,ther_a = roc_curve(y_test,y_prd_adb_prb)
```

In [29]:
```
auc_a = roc_auc_score(y_test,y_prd_adb_prb)
auc_a
```

Out[29]:
```
0.9132332845290386
```

In [30]:
```
def plot_roc_curve(fpr, tpr,auc):
    plt.plot(fpr, tpr, color='orange', label='ROC and auc score = '+str(auc))
    plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend()
    plt.show()
```

In [31]:
```
plot_roc_curve(fpr_a,tpr_a,auc_a)
```



get more accuracy we can do Hyperparametre tuening

In [32]:
```
from sklearn.model_selection import RandomizedSearchCV
```

In [33]:
```
grid_param = {
    'n_estimators': [50,70,90,120,150],
    'learning_rate':[0.001,0.01,0.1,0.5]
}
```

In [34]:
```
rsearch = RandomizedSearchCV(estimator=adb,param_distributions=grid_param,cv=3,ve
```

In [35]:
```
rsearch.fit(X_train,y_train)
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

Out[35]:

RandomizedSearchCV

```
RandomizedSearchCV(cv=3, estimator=AdaBoostClassifier(), n_jobs=-1,
                   param_distributions={'learning_rate': [0.001, 0.01, 0.1,
                                                          0.5],
                                        'n_estimators': [50, 70, 90, 120, 15
0]},
                   verbose=1)
```

estimator: AdaBoostClassifier

AdaBoostClassifier

```
AdaBoostClassifier()
```

In [36]:
```
print(rsearch.best_params_)
```

```
{'n_estimators': 90, 'learning_rate': 0.5}
```

In [37]:
```
adb_p = AdaBoostClassifier(n_estimators=90,learning_rate=0.5)
```

In [38]:
```
adb_p.fit(X_train,y_train)
```

Out[38]:

AdaBoostClassifier

```
AdaBoostClassifier(learning_rate=0.5, n_estimators=90)
```

In [39]:
```
adb_p.score(X_train,y_train)
```

Out[39]:
```
0.8559706623882649
```

In [40]:
```
y_prd_bp = adb_p.predict(X_test)
```

In [41]:
```
print(accuracy_score(y_test,y_prd_bp))
```

```
0.860785408524102
```

In [42]:
```
print(classification_report(y_test,y_prd_bp))
```

```
              precision    recall  f1-score   support

           0       0.88      0.94      0.91      8196
           1       0.76      0.60      0.67      2550

    accuracy                           0.86     10746
   macro avg       0.82      0.77      0.79     10746
weighted avg       0.85      0.86      0.85     10746
```

In [43]:
```
y_prd_adb_prb = adb_p.predict_proba(X_test)[:,1]
y_prd_adb_prb
```

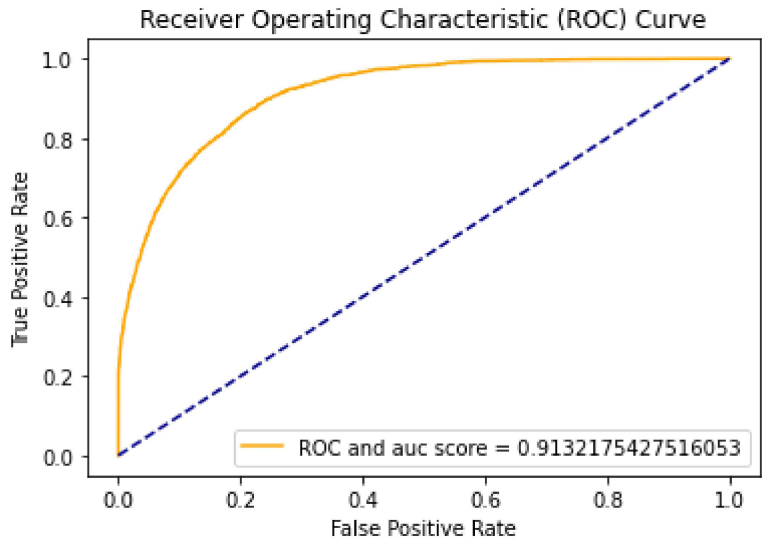
Out[43]:
```
array([0.47447542, 0.49921235, 0.50175257, ..., 0.49218003, 0.49828127,
       0.47720076])
```

In [44]:
```
fpr_a,tpr_a,ther_a = roc_curve(y_test,y_prd_adb_prb)
```

In [45]:
```
auc_a = roc_auc_score(y_test,y_prd_adb_prb)
auc_a
```

Out[45]:
```
0.9132175427516053
```

In [46]:
```
plot_roc_curve(fpr_a,tpr_a,auc_a)
```



In [ ]:
```
```