Machine Learning application — Census Income Prediction

Problem Definition:

This problem revels an aim to increase about how the income factor impact personal lives of people and also impact on the nation. This data is extracted from the 1994 census bureau database and try to find out how different features have an impact on income of an individual, this dataset is quite old thought it surely help us and analyse to predicting the income of an individual.

This are the following features are provided, which will be build to classification model to train

1. Age — **The age of an individual**

2. Workclass — **The class of work to which an individual belongs.**

3. Fnlwgt — **The weight assigned to the features**

4. Education — **Education level**

5. Education\_num — **Number of years taken for education**

6. Marital\_Status — **This category define the marriage status of a person**

7. Occupation **— Profession of a person**

8. Relationship — **Relation of the person in his family**

9. Race — **Origin background**

10. Sex — **Gender of a person**

11. Capital\_gain — **Capital gained by a person**

12. Capital\_loss **— Loss of capital for a person**

13. Hours\_per\_week — **Number of hours for which an individual works per week**

14. Native\_Country — **a person belongs to which country**

Target:

Income — **The target variable, which predicts if the income is higher or lower than 50K$.**

# Data Analysis

# This is the first step to check information about data

**Age 0**

**Workclass 0**

**Fnlwgt 0**

**Education 0**

**Education\_num 0**

**Marital\_status 0**

**Occupation 0**

**Relationship 0**

**Race 0**

**Sex 0**

**Capital\_gain 0**

**Capital\_loss 0**

**Hours\_per\_week 0**

**Native\_country 0**

**Income 0**

**dtype: int64**

we can observe the data set belongs to mix of categorical and numerical format and there is no missing value present.

**Checking Outlier-** now is to detect the outliers in dataset, best way to interpret outliers is using boxplot for better visualization.

**Skewness-** now proceed with skewness in our data, which allow to fit data with symmetric distribution. We transforming data set for removing skewness using yeo johnson method

**Pre-processing pipeline**

Classification model needs input as float/ int, do not work with string data so we are using “LabelEncoder” to encode our categorical columns.

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

census['Workclass']=le.fit\_transform(census['Workclass'])

census['Education']=le.fit\_transform(census['Education'])

census['Marital\_status']=le.fit\_transform(census['Marital\_status'])

census['Occupation']=le.fit\_transform(census['Occupation'])

census['Relationship']=le.fit\_transform(census['Relationship'])

census['Native\_country']=le.fit\_transform(census['Native\_country'])

census['Race']=le.fit\_transform(census['Race'])

census['Sex']=le.fit\_transform(census['Sex'])

census['Income']=le.fit\_transform(census['Income'])

**Building Machine Learning Models**

Now we are proceed to main step of machine learning , fitting the model and finding the output.

We are fitting the dataset with multiple model to compare the performance of all model and select the best model.

from sklearn.linear\_model import LogisticRegression

from sklearn.svm import SVC

from sklearn.tree import DecisionTreeClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import GradientBoostingClassifier

from sklearn.ensemble import AdaBoostClassifier

**from sklearn.model\_selection import train\_test\_split**

**from sklearn.metrics import confusion\_matrix,classification\_report,accuracy\_score**

gb.fit(x\_train,y\_train)

gb.score(x\_train,y\_train)

predgb=gb.predict(x\_test)

print(accuracy\_score(y\_test,predgb))

print(confusion\_matrix(y\_test,predgb))

print(classification\_report(y\_test,predgb))

0.8613329238329238

[[4655 281]

[ 622 954]]

precision recall f1-score support

0 0.88 0.94 0.91 4936

1 0.77 0.61 0.68 1576

accuracy 0.86 6512

macro avg 0.83 0.77 0.80 6512

weighted avg 0.86 0.86 0.86 6512

**Concluding Remarks**

This is the end of our process and we have successfully trained our model to predict the income of an individual with 86% of accuracy.