**Census Income Prediction**

**Problem Definition**

The dataset contains features such as education, working class, age, occupation, marital status, relationship etc. We have to build a classification model based on the features to predict person's yearly income is above $50k or not.

**Data Analysis**

The dataset contains 32561 rows and 15 columns are present where one columns is the result column. In the dataset, input variables are a blend of numerical and categorical types, where the non-numerical columns are represented using strings. The categorical variables need to be convert into numerical using encoding techniques. Also we can see that the target variable i.e. income is having string datatype. Income will also be need to be label encoded with 0 as the income less than 50K and 1 as the income greater than 50k. As the count value of income for value count for greater than 50k is very less than the count of value less than 50k, hence it shows the imbalanced data classification.

In the dataset, there are some there are some categorical data and numerical data, the numerical data contains age which is the age of a person, fnlwgt is sampling weight, education-num is the total number of education in years, capital-gain is the capital from the source, and capital-loss is the loss of capital.

In the dataset, we noticed that none of the numerical features are having missing values.

In the dataset for categorical attributes, missing values are marked with a ‘?’ character. To remove the missing values or ‘?’ we will need to use Imputer by which we can replace those values with mode, or else if the value count of ‘?’ is low then we can delete those rows from the dataset.

Dataset values workclass having 1836 values represented by ‘?’ are present, 1843 values represented by ‘?’ are present in occupation and 583 values represented by ‘?’ are present in native country. And dataset having no null values.

**EDA**

As the dataset contains numerical and categorical values so we will handle the numerical and categorical values differently, numerical values need to be scaled, whereas for categorical columns we need to fill the missing values and then encode the categorical values into numerical values.

From the observation of categorical attributes by using countplot of seaborn:

* Most of the person are having salary less than 50k and a very few are having salary more than 50k, i.e. 24720 person are having salary less than 50k and 7841 person having salary greater than 50k.
* Our dataset has 24720 people earning <=50K i.e. 76% and remaining 24% earns more than 50K.
* Gender: People with gender male is more in our dataset compare to female. Out of total male 30% of them earn salary more than 50K while less than 15% female earn more than 50K. 89% female earn less than 50K.
* workclass: Most of the people are working with private sectors and there are very few to work for State-Gov, Federal-Gov, Self-Emp and Local-Gov. More than 75% people are working for Private companies.
* There are same values are present in dataset which can we combined to one value. 'Self-emp-not-inc' and 'Self-emp-inc' can be combine to Self Employed also 'Local-gov', 'State-gov' and 'Federal-gov' can be combined to Government employee. After combining we can see now we have more than 10% of people are working as government employee and 10% people are working self-employed.
* Education: In the education, most of the people are HS-Grad, Bachelors and College went student. In education attribute, we can combine 1st-4th, 5th-6th, 7th-8th as elementary group and 9th, 10th, 'HS-grad','11th','12th' as graduate group.
* marital.status: Married-civ-spouse count is very high, also many people who are not married yet are having more count and many people with divorced has observed. Also 41% of married people seem to earn salary greater than 50K.

**Pre-processing Pipeline**

The workclass, occupation and native country contains values ‘?’ character, to replace those character first we will replace those character with ‘NaN’, after replacing all character present in workclass, occupation and native.country we will use mode technique to replace all NaN value from the columns. After removing those value we will see there are no character and null value are present.

For numerical data none of column are having null values, and categorical columns were having character present which has been removed using mode.

As the dataset contains numerical and categorical values we need to use Scaling technique and Encoding technique as all the machine learning models expect numerical values only. First we will use Encoding technique to encode all categorical data into numerical data, as we are having more unique values in categorical we have to think of encoding technique to be implement. As unique value is more than 2, we cannot use OneHot Encoder we will use Label Encoder to transform the column from categorical datatype to numerical datatype. To achieve this we need to import Label encoder from sklearn libraries after implementing and initializing Label Encoder will can encode those categorical data. To do so, we will create a list variable which will contains all attribute which has to be encode, after initializing list element we will add all categorical columns into that list by taking the attributes having datatype as ‘object’ which we will obtain from information of dataset, after adding all categorical attributes into a list we can run it into a for loop, which will transform all categorical column into numerical columns.

After the encoding techniques, we can see the correlation of matrix using the heatmap of sns, from where we can observed that income has strong correlation with age, also age having strong correlation with Hours per week and also strong correlation is present between capital gain and hours per week, and moderate correlation with capital loss and age.

Other observation with income:

* Income is highly positively correlated with education.num.
* Income is highly negatively correlated with maritial.status.
* Income is moderate correlated with sex, capital.gain, capital.loss and hours.per.week

Now to transform numerical data, we can use another technique which called as Standard Scaler technique, it will transform our data such that the distribution will have a mean value 0 and standard deviation of 1.

Before doing this we will remove the outliers present in the dataset by using using zscore and threshold values as 3. After implementing zscore we can see the data shape will get reduce as it has removed all the outliers present from dataset, now we can processed with other scaling technique.

After removing the outliers we will split the data into x and y. To split the data we will take all attributes except income into x and income attribute into y. Now we can check for the skewness of the data. To see the skewness of x variable we will use a skew function i.e. skew(). This function will show us the skewness present in the dataset. After applying the function we can see our data is highly skewed for capital.loss, race, capital.gain, workclass and native.country because it is having the skewness value greater than 0.5 and -0.5, so we cannot process the data for prediction with skewness, which will result into poor prediction of model. To remove the skewness of data we will use square technique, we will remove skewness from the attributes having skewness greater than 0.5 and -0.5 by taking it into a loop. After looping x variable we can see our skewness has been reduced within the range. So that we can process with Scaling. After removing the outliers and skewness from dataset then we will use Standard Scaler to scale our data, for this we will include StandardScaler library from sklearn.preprocessing. To scale the data we will pass our x variable with the object of standard scaler to the function fit(), StandardScaler.fit(x), and will again pass it to transform method StandardScaler.transform(x) and will assign to our variable x again.

Now as we have scaled our data we can implement PCA technique. The main idea of **principal component analysis** (**PCA**) is to reduce the dimensionality of a data set as consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the dataset, up to the maximum extent. As we are having 14 attributes in x variable we will get highly correlated attributes using PCA. PCA has to be import from the sklearn.decomposition library. In the PCA we have to pass n\_components, it will return us attribute with respect to n\_components. To do so we will create an object of PCA as pca with 10 components and will pass our x parameter into fit\_transform() function, and will reassign to our variable x. Now we have processed out data and we are ready to build our model, let’s prepare the data for the model and build it.

**Building Machine Learning Models**

Now we will build our model with train\_test\_split and will calculate accuracy score for each model individually and will store it into list. To prepare the model and calculate the accuracy score we will use a for loop to test our model for each random state, to do this we will create a function name as main\_fun() where we will initialize accu\_score variable with initial value as 0, this variable will contain the max accuracy score that will get from the accuracy\_score function in the loop. Then we will run a ‘for’ loop with the random state range from 42 to 101.

Inside the for loop we will divide our data into train and test using train\_test\_split, where we will pass data size of 0.2 and random state values will be coming from for loop and will store train and test values in train\_x, train\_y, test\_x, test\_y. After this we will calculate accuracy score for each model and will compare each accuracy score with the previous accuracy\_score until each random state get execute and then max accuracy score will get return.

**Preparing model**

Let’s start by evaluating a mixture of machine learning models on the dataset.

We will evaluate the following machine learning models on the dataset:

* Decision Tree
* Logistic Regression
* KNN
* Random Forest (RF)
* AdaBoost Classifier
* Naïve Bayes Classifier

To run all models in one go we will create a model list names as models. In the model list we will add each model one by one with their parameters and the name of the model with their instance or object. The advantage of creating a list is, we do not have to run same function for each model and can obtain all accuracy score of the models into a single variable. As we have create a function name main\_fun() so we can pass our list directly to the function. And function will start doing it’s work by running through all random state for each model and will return max accuracy score of each model to a variable.

After getting all accuracy scores of models we will create one data frame to compare accuracy score of model in descending order of accuracy score.

After plotting of data frame we can notice that Logistic Regression has performed better than other model on random state of 45 and gave accuracy score of 84. We also plot a graph to show the visualization of accuracy score.

**Saving a model**

To save our model we will use our best accuracy score model LogisticRegression with random state 45. By using train test split we will calculate accuracy score along with confusion matrix and classification report. After this we will store our model for future use. We will save our model in pickle by importing joblib from sklearn.externals.

**Conclusion**

In this project, we've overcome missing values and applied a data transformation technique which is Label Encoder to convert categorical data into numerical data, also used scaling technique to scale our data and PCA to get the attributes for data prediction. To make new prediction of income of person based on attributes our model in very much accurate and gave the good prediction that we can say by accuracy score.