Blog is generally on ‘Census Income Project’ an evaluation project as part of the PG Data Science curriculum in on **Data trained** academy  walks you through each and every step in detail and helps us to understand the whole ML model building process . So, let’s get started.

Problem Statement



This data was extracted from the [1994 Census bureau database](http://www.census.gov/en.html) by Ronny Kohavi and Barry Becker (Data Mining and Visualization, Silicon Graphics). A set of reasonably clean records was extracted using the following conditions: ((AAGE>16) && (AGI>100) && (AFNLWGT>1) && (HRSWK>0)). ***The prediction task is to determine whether a person makes over $50K a year*.**.

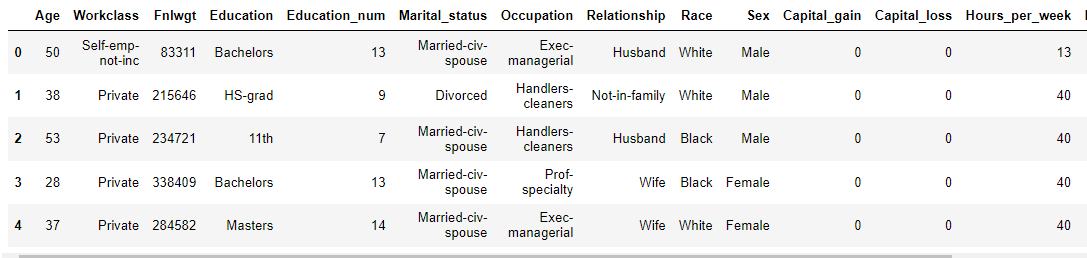
**Datasets**

We will be using a single dataset , which we will be used for Feature engineering and then divided to Train data and Test data, which can be downloaded from kaggle.com or below

laod Files:

* <https://raw.githubusercontent.com/dsrscientist/dataset1/master/census_income.csv>

**The dataset sample:**



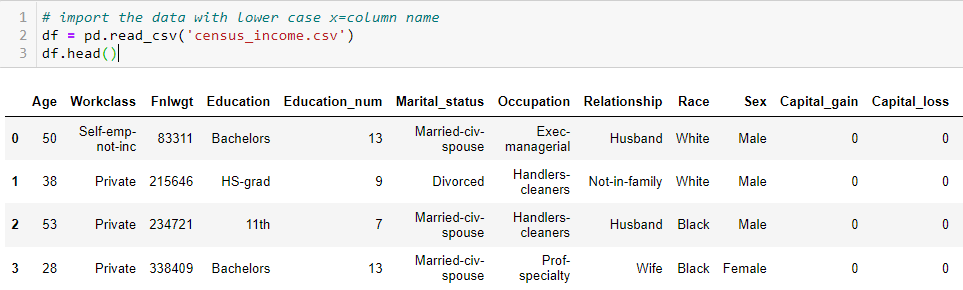
Screenshot of the  **dataset with** (32560 rows and 15 columns): this datarefers to that portion of data used to fit a model.

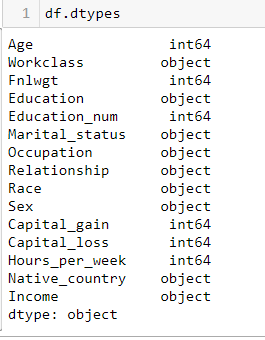
The data is consisting of both numerical & categorical also we can observed some special character are also used the data transformation on the data is required before applying it to our model

**Python Coding**

**Step 1: Import the relevant libraries in Python and the dataset for excel.**





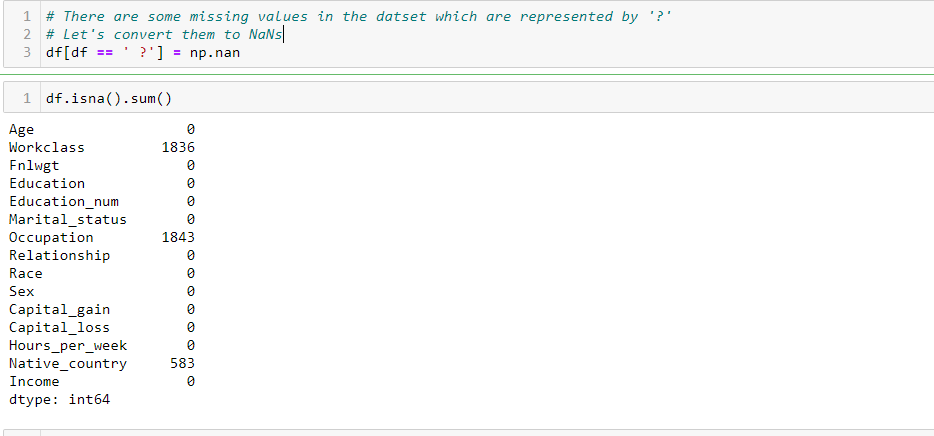


## Step 2: we will check for Null/Missing data

## 

Checking for Null or missing data in the system. This does not confirm whether the data cleansing will be required or not. This is just a upfront count of missing (Nan) values .

However while analysing the data we found values like “?” gives us a hint of missing values, which should be converted to NULL and then treated



Now we see around 1.8K+ data missing for Work class and Occupation.

Thus till now we have the below inference

The dataset has 32,561 observations with 14 Features and 1 Target Class.

Though the data didn’t had any nan values but we saw values like '?' which we assumed to missing values.

This missing values will be treated as nan values, the missing data less is around 5.7% of the total values in the column

Our objective is classify the target into two sets :

it has 2 classes:

1: greater than 50K

2: 50K or lower

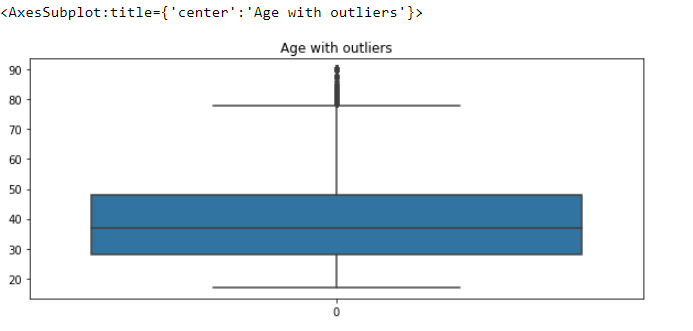
We need to it convert it into machine learning compatible values

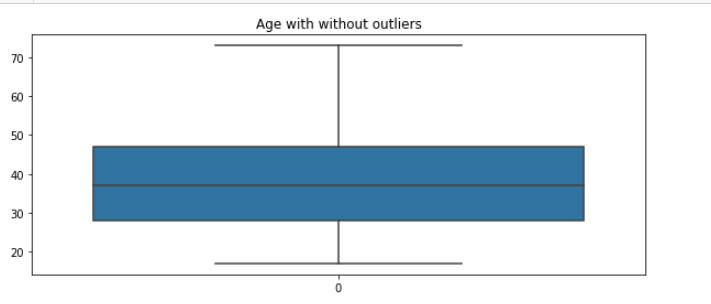
**Step 3: Feature Generation [Univariate/Bivariate]**

In this step the main objective is to work on the data set and perform some transformation such as creating different bins of particular columns ,clean the dirty data so that it can be used in our ML model . This step is very important as to build a model with high prediction score with a higher accuracy

**Age:**

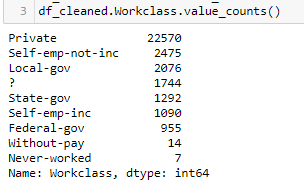
In the column ‘AGE’, we can see the data is numeric and continuous in nature. After plotting box plot , it seems it might have some outliers



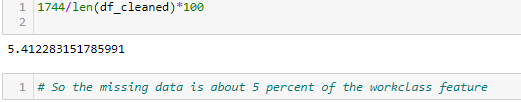
From the box plot we can interpret that the outliers may be present about 0.99 quantile. Let's remove those observations whose lies above 99% of all the ages.

**Workclass:**

These columns categorical data where it shows the vairpus job sectors



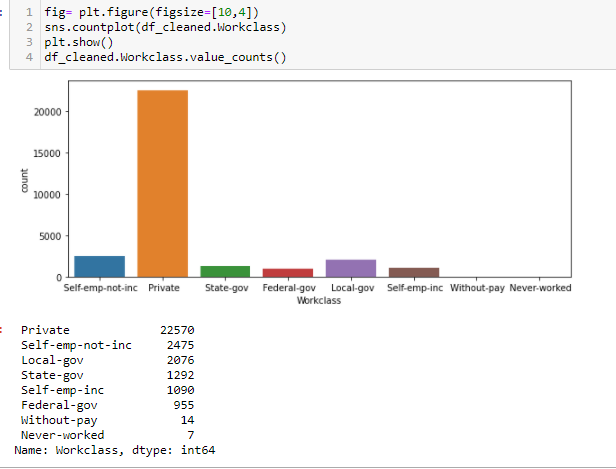
We can see that we have unknown characters “?” which count for 5% of total Workclass data



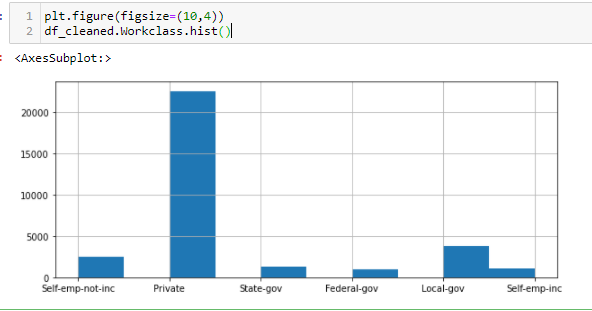
Thus we impute it with the Local gov data



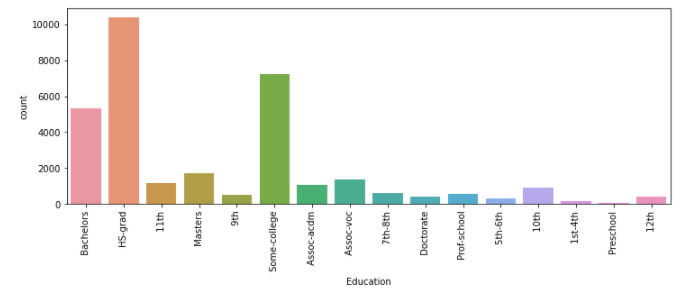
Post



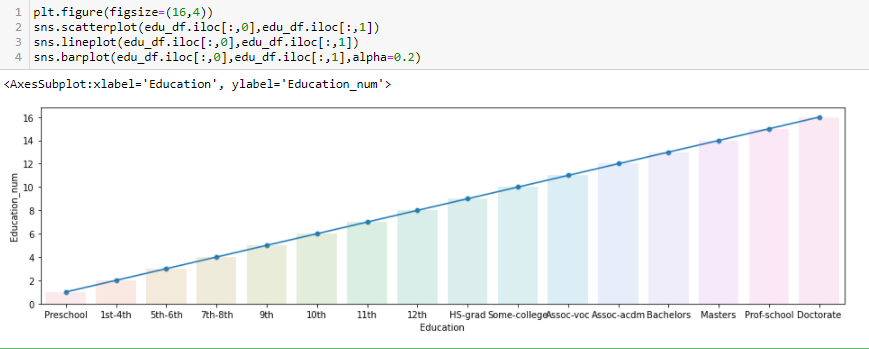
The attributes of the workclass columns: Without pay and never worked are too less to positively impact the data. Thus removing those values would be a better option for our modelling

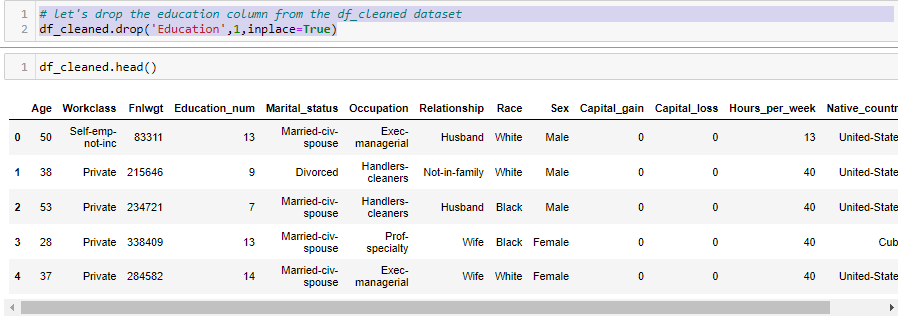


**Education:**



We can see that the education number represents the ordinal encoding of the education feature

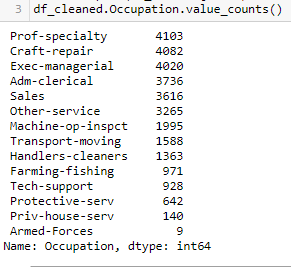
Most of the population in the dataset are high school graduates follows by population studying in some-college. This is a salary prediction dataset and there are some observations in the dataset which are potential not capable of having a income. So we will drop this column

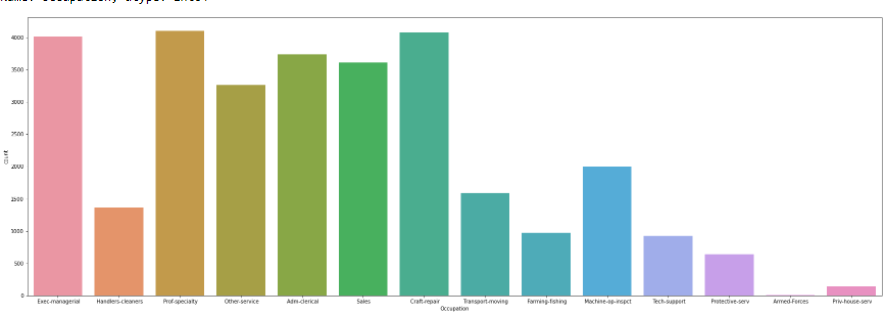


=

**Occupation:**

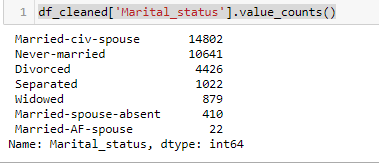
The occupation of people varies from a lot of jobs ranging from Armed forces to prove house helps. Most population in mainly involved in clerical jobs

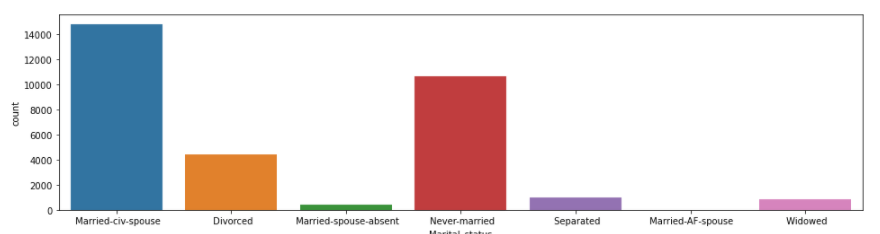


****

**Maritial-Status**

The Marital status of people varies from happily married to divorced

****

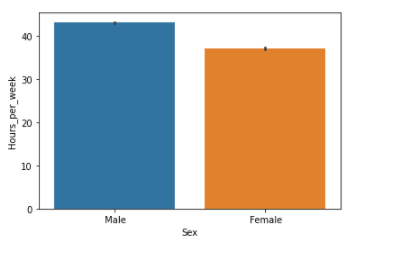
****

**Bivariate Analysis:**

Bivariate analysis is a kind of statistical analysis when two variables are observed against each other. One of the variables will be dependent and the other and vice versa

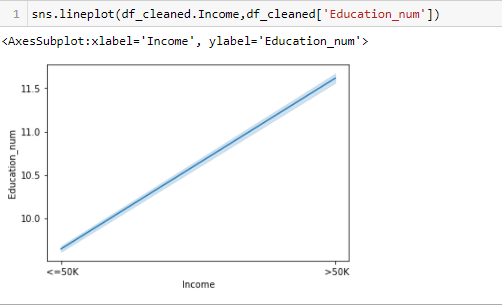
**Gender vs Hours per week:**

When we plot a graph between the Gender and Hours per week, we find that males are working more hours as compared to females



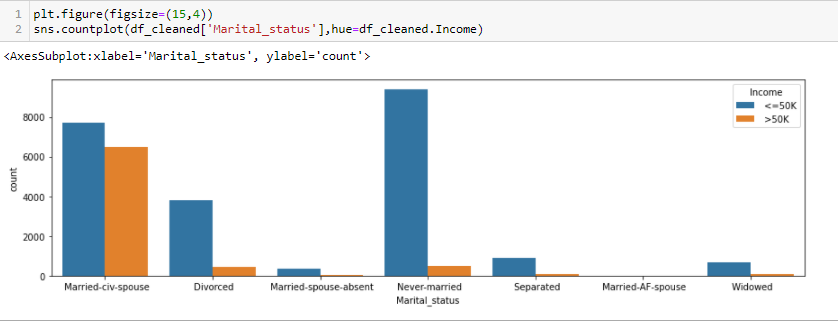
**Education Vs Income:**

Another plotting which we did wasplotting a line plot between the education and income .we found that they are directly

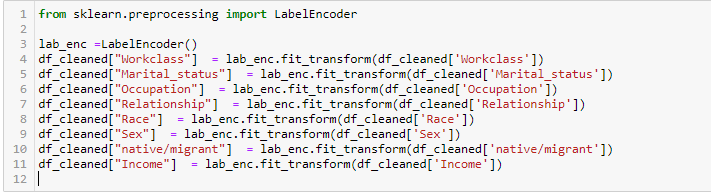


**Gender and Income :**

We observe that the Overall there is a Gender pay inequality over all



**Step 4: Prepare categorical variables for model using label encoder**

To convert categorical text data into model-understandable numerical data, we use the Label Encoder class. So all we have to do, to label encode a column is import the Label Encoder class from the sklearn library, fit and transform the column of the data, and then replace the existing text data with the new encoded data. For that we must segregate the Numerical and Categorical data 

**Step 5 :Split the dataset to X & Y for Training and Testing**

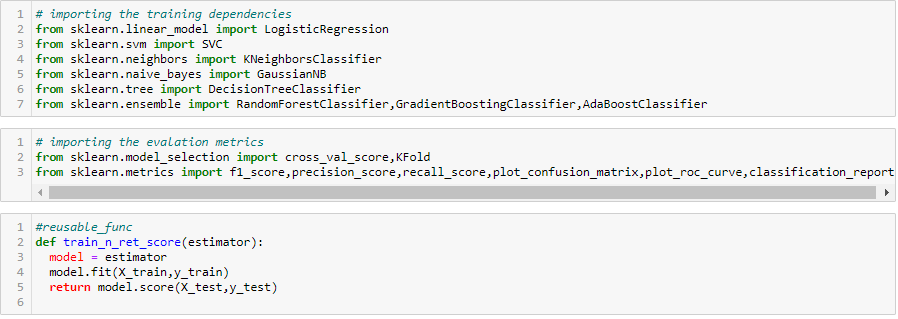
To train any model the data needs to be divided into feature data and label data. Then again it wilsplit to ttarin and test data .First it needs to be used to train the data and then the same model is used to predict for test data .It must be ensured that the Shape of these data split is in sync



**Step 6: Building Model and Train**

The main objective in this step is to develop a benchmark model that serves us as a baseline, upon which we will measure the performance of a better and more tuned algorithm. We will be using different Classification Models and compare t them to see which algorithm is giving better performance other and At the end we will combine all of them using Stacking and see how our model is predicting

FOR THAT WE WILL USE A Reusable function:



1. **LogisticRegression:**

**Score:** 0.8038201352964585

1. **Support Vector Classifier :**

**Score**: 0.8239156386788699

**3. DecisionTreeClassifier:**

**Score**: 0.7735773975328293

**4. KNeighborsClassifier:**

**Score :** 0.8078989255869479

**5.** **RandomForestClassifier**

**Score : 0.8152606446478313**

**6.** **AdaBoostClassifier(base\_estimator as LogisticRegression):**

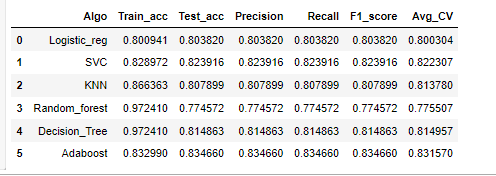
**Score :** 0. 0.7983485873458018

**Step 7: Cross Validation:**

Cross-validation is a re-sampling procedure used to evaluate machine learning models on a limited data sample.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.



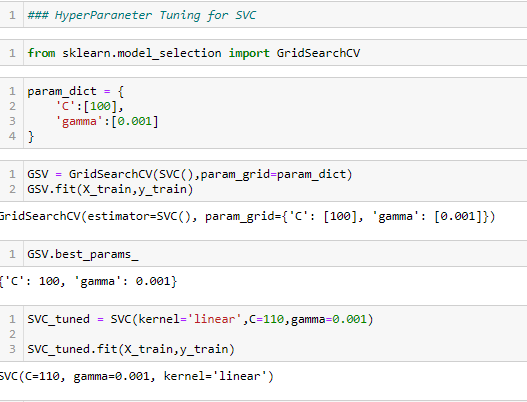


Thus we found the SVC to be apt . TO make it more suitable we will use Hyper parameterisation

**Step 8: Hyper parameter Tunning:**

Parameters which define the model architecture are referred to as **hyperparameters** and thus this process of searching for the ideal model architecture is referred to as hyperparameter tuning.

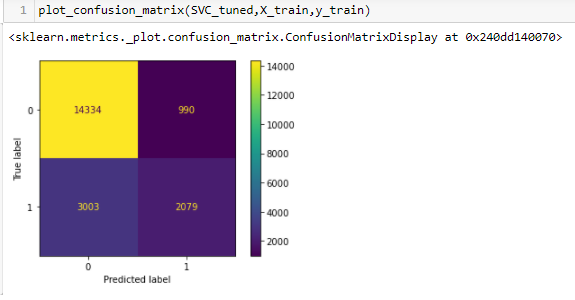
Implementing Hyperparamters we get the Best Parameters, we will rerun the training model with it.



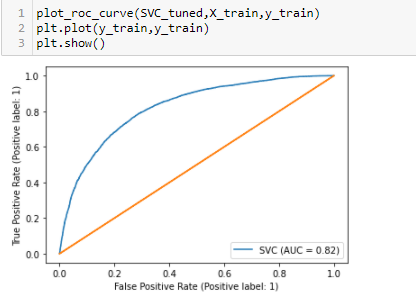
Once done the Test data will be run on the model and fine score is calculated: as 0.8066056506167927

**Step 10: Confusion Matrix and ROC**

A confusion matrix is **a technique for summarizing the performance of a classification algorithm**. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in your dataset. Confusion matrix goes deeper than classification accuracy by showing the correct and incorrect (i.e. true or false) predictions on each class



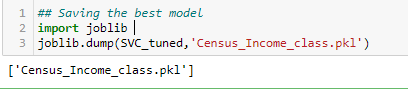
**ROC curve** summarizes the performance by combining confusion matrices at all threshold values



**Step 10: Saving Model for deployment**

Once done we will save the model for deployment using joblib or pickle libraries

****

****

**Final Conclusion**

In this type of problems , the Feature Engineering is the most important part. We need to handle the categorical and numerical data so that we do not lose any influential data and also how we build different ML models on the training dataset. We also check the Accuracy score of each model so that we can understand how it may perform in our test dataset . At last the Model is made more efficient by implementing Hyperparameter Tuning..

Please let me know your thoughts about this article and do comment if you face any issues while implementing it **Sources :**

1. [Cross-Validation. Validating your Machine Learning Models… | by Kurtis Pykes | Towards Data Science](https://towardsdatascience.com/cross-validation-c4fae714f1c5)
2. [Hyperparameter tuning for machine learning models. (jeremyjordan.me)](https://www.jeremyjordan.me/hyperparameter-tuning/)
3. [[ROC Curve and AUC — Explained. What they mean and when they are usefu | by Soner Yıldırım | Towards Data Science](https://towardsdatascience.com/roc-curve-and-auc-explained-8ff3438b3154)/](https://www.analyticsvidhya.com/)