***A PROJECT ON***

# “CENSUS INCOME PREDICITON”

SUBMITTED IN

PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE COURSE OF

DIPLOMA IN BIG DATA ANALYSIS



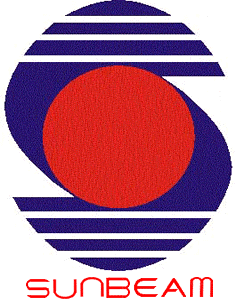
**SUNBEAM INSTITUTE OF INFORMATION TECHNOLOGY, PUNE**

Submitted By:

Sarvesh Watkar (83900)

Kartik Sarode (83833)

**Mr.Nitin Kudale Mrs.Manisha Hingne** Centre Coordinator Course Coordinator



**CERTIFICATE**

This is to certify that the project work under the title “CENSUS INCOME PREDICITON” is done by Dattatray Hake & Udit Deshmukh in partial fulfillment of the requirement for award of Diploma in Big Data Analysis Course.

Mr. Aniket P Mrs. Manisha Hingne

**Project Guide** **Course Coordinator**

Date:

# ACKNOWLEDGEMENT

A project usually falls short of its expectation unless aided and guided by the right persons at the right time. We avail this opportunity to express our deep sense of gratitude towards Mr. Nitin Kudale (Center Coordinator, SIIT, Pune) and Mrs. Manisha Hingne (Course Coordinator, SIIT ,Pune) and Project Guide Mr. Aniket P.

We are deeply indebted and grateful to them for their guidance, encouragement and deep concern for our project. Without their critical evaluation and suggestions at every stage of the project, this project could never have reached its present form.

Last but not the least we thank the entire faculty and the staff members of Sunbeam Institute of Information Technology, Pune for their support.

Sarvesh Watkar

DBDA March 2024 Batch, SIIT Pune

Kartik Sarode

DBDA March 2024 Batch, SIIT Pune

**TABLE OF CONTENTS**

1. **Introduction**
   1. Introduction And Objectives
   2. Why this problem needs To be Solved?
   3. Dataset Information

## Problem Definition and Algorithm

* 1. Problem Definition
  2. Algorithm Definition

## Experimental Evaluation

* 1. Methodology/Model
  2. Exploratory Data Analysis

## Results And Discussion

1. **GUI**
2. **GitHub link**

## 7. Conclusion

**8. Future Scope**

* + 1. **Introduction:**
       1. **Introduction And Objectives:**

The Census Income Prediction Machine Learning project focuses on predicting an individual's income level based on their demographic and employment-related attributes. This type of prediction is critical in various fields, including economic planning, social services, and market research. By leveraging machine learning techniques, this project aims to analyze and predict whether a person earns more than a specified threshold income, often referred to as the "target income."

The dataset commonly used for such predictions is derived from census data, which includes a range of features such as age, work class, education level, occupation, hours worked per week, and marital status. Machine learning models can process this data to identify patterns and relationships between these features and the income levels.

The economic well-being of a Nation is highly driven by the income of the residents.

Countless decisions in private and public sectors are based on Census data. Census data is the backbone of the democratic system of government, highly affecting the economic sectors. Census-related figures are used to distribute the federal funding by the government into different states and localities.

Not only the above, the census data is also used for post census population estimates and projections, economic and social science research, and many other such applications. Hence, the importance of this data and its correct predictions is very clear to us.

Data has always been the backbone of many important decisions. When an assumption is backed up by facts and numbers, the chances of incorrectness and bad decisions decrease.

**Objective:**

## Why this problem needs to be Solved?

**Problem Statement:**

The above introduction had an aim to increase the awareness about how the income factor actually has an impact not only on the personal lives of people, but also an impact on the nation and its betterment. We will today have a look on the data extracted from the 1994 Census bureau database, and try to find insights about how different features have an impact on the income of an individual. Though the data is quite old, and the insights drawn cannot be directly used for derivation in the modern world, but it would surely help us to analyse what role different features play in predicting the income of an individual.

The primary objectives of the Census Income Prediction Machine Learning project are:

1. **Data Exploration and Preparation:**
   * **Understand and Preprocess the Data:** Explore the dataset to understand its structure, identify missing values, and perform data cleaning. This involves normalizing or standardizing data, encoding categorical variables, and handling missing values.
   * **Feature Selection:** Determine the most relevant features that contribute to predicting income levels and remove irrelevant or redundant features.
2. **Model Building:**
   * **Choose and Implement Machine Learning Models:** Apply various machine learning algorithms such as Logistic Regression, Decision Trees, Random Forests, Gradient Boosting, and Neural Networks to the dataset.
   * **Train and Validate Models:** Split the data into training and testing sets. Train models on the training set and validate their performance on the testing set using appropriate metrics like accuracy, precision, recall, and F1-score.
3. **Model Evaluation and Optimization:**
   * **Assess Model Performance:** Evaluate the models using metrics like confusion matrix, ROC curve, and AUC (Area Under the Curve) to understand their predictive performance.
   * **Hyperparameter Tuning:** Optimize model performance by adjusting hyperparameters and employing techniques like cross-validation.
4. **Deployment and Interpretation:**
   * **Deploy the Best Model:** Implement the best-performing model in a real-world scenario or a simulated environment.
   * **Interpret Results:** Provide insights and interpretations of the model’s predictions. This includes understanding which features are most influential in predicting income levels and how the model’s predictions can be utilized.
5. **Ethical Considerations and Bias Mitigation:**
   * **Address Ethical Issues:** Ensure that the model does not propagate biases related to gender, race, or other sensitive attributes. Implement strategies to mitigate any identified biases.
   * **Transparency and Accountability:** Document the model's decision-making process and make it transparent to stakeholders to build trust and ensure accountability.

This outline provides a comprehensive overview of the key aspects and goals of a Census Income Prediction Machine Learning project. The ultimate aim is to develop a robust model that accurately predicts income levels while considering ethical implications and practical applications.

## Data Information.

## Description of every column: -

**The Dataset:**

The dataset provided to us contains 48800 rows, and 14 different independent features. We aim to predict if a person earns more than 50k$ per year or not. Since the data predicts 2 values (>50K or <=50K), this clearly is a classification problem, and we will train the classification models to predict the desired outputs.

Mentioned below are the details of the features provided to us, which we will be feeding to our classification model to train it.

1. Age — The age of an individual, this ranges from 19 to 65.

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

3. Fnlwgt — The weight assigned to the combination of features (an estimate of how many people belong to this set of combination)

4. Education — Highest level of education

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

6. Marital\_Status — Represents the category assigned on the basis of marriage status of a person

7. Occupation — Profession of a person

8. Relationship — Relation of the person in his family

9. Race — Origin background of a person

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 — Country to which a person belongs

Output:

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

## Train.csv

It has seven columns.

Following are the columns that has been Dropped:

1) Workclass

2) Final Weight

3) Race

4) Native Country

5) Income

**Test.csv:** is same as train.csv except it does not have ‘Income’ Column.

## Problem Definition and Algorithm:

* + - 1. **Problem Definition**

**Problem Statement:**

The above introduction had an aim to increase the awareness about how the income factor actually has an impact not only on the personal lives of people, but also an impact on the nation and its betterment. We will today have a look on the data extracted from the 1994 Census bureau database, and try to find insights about how different features have an impact on the income of an individual. Though the data is quite old, and the insights drawn cannot be directly used for derivation in the modern world, but it would surely help us to analyse what role different features play in predicting the income of an individual.

## Algorithm Definition:

**Logistic regression: It** is a widely used statistical model for binary classification tasks, where the goal is to predict the probability of an outcome that can take one of two values, typically 0 or 1. Unlike linear regression, which predicts continuous outcomes, logistic regression uses a logistic function (also known as the sigmoid function) to map the predicted values to a probability between 0 and 1.

**Random forest:** is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

**k-Nearest Neighbors (k-NN)** : Algorithm is a simple, instance-based learning method used for classification and regression tasks. It works by finding the "k" closest data points (neighbors) to a given input point and then predicting the label based on the majority class (in classification) or the average value (in regression) of these neighbors.

The distance between points is usually measured using metrics like Euclidean distance. k-NN is intuitive and easy to implement but can be computationally expensive, especially with large datasets, as it requires storing all data points and calculating distances for each prediction.

**Decision Tree:** algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data(training data).In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

**Naive Bayes**: It is a simple, yet powerful, supervised machine learning algorithm used primarily for classification tasks. It is based on Bayes' Theorem, which describes the probability of an event occurring given prior knowledge of conditions related to the event. The "naive" part refers to the assumption that all features are independent of each other, which rarely holds true in real life, but the model still performs well in practice. Naive Bayes is particularly useful for text classification, such as spam detection, due to its efficiency and ability to handle large datasets.

**Support Vector Machine (SVM):** It is a supervised learning algorithm used mainly for classification. It works by finding the optimal hyperplane that best separates data points into different classes. This hyperplane is chosen to maximize the margin, the distance between it and the nearest data points from each class, called support vectors. SVM can handle non-linear data by applying kernel functions, which transform the data into a higher-dimensional space where separation is easier. It's effective for high-dimensional data but may require careful parameter tuning for optimal performance.

**CatBoost**: It is a high-performance gradient boosting algorithm developed by Yandex, designed for handling categorical data more effectively than many other machine learning models. It builds an ensemble of decision trees, combining the predictions of many models to improve accuracy. CatBoost stands out for its ability to automatically handle categorical features without requiring extensive preprocessing, such as one-hot encoding. It also reduces overfitting through techniques like ordered boosting and has built-in support for handling missing values. CatBoost is known for its ease of use, fast training, and strong performance in various machine learning tasks, including ranking, classification, and regression.

**XGBoost:** or extreme gradient boosting is one of the well-known [gradient](https://analyticsindiamag.com/gradient-descent-everything-you-need-to-know-with-implementation-in-python/) [boosting](https://analyticsindiamag.com/gradient-descent-everything-you-need-to-know-with-implementation-in-python/) techniques (ensemble) having enhanced performance and speed in tree- based (sequential decision trees) machine learning algorithms. XGBoost was created by Tianqi Chen and initially maintained by the Distributed (Deep) Machine Learning Community (DMLC) group. It is the most common algorithm used for applied machine learning in competitions and has gained popularity through winning solutions in structured and tabular data. It is open- source software. Earlier only [python and R packages](https://analyticsindiamag.com/python-vs-scala-for-apache-spark/) were built for XGBoost but now it has extended to Java, Scala, Julia and other languages as well.

**AdaBoost:** It short for Adaptive Boosting, is an ensemble learning algorithm that combines multiple weak learners, typically decision trees with a single split (stumps), to create a strong classifier. The key idea behind AdaBoost is to focus on the data points that are hardest to classify correctly. During each iteration, it assigns higher weights to misclassified data points so that the next learner focuses more on those challenging cases. The final model is a weighted sum of the weak learners, where more accurate learners have greater influence. AdaBoost is effective in improving the accuracy of weak models but can be sensitive to noisy data and outliers.

## Experimental Evaluation:

* + - 1. **Methodology:**

The census income prediction project uses machine learning algorithms to predict if an individual's income is more than $50,000 per year based on socio-economic and demographic factors. The project trains the algorithms on census data to create an accurate income prediction model. Predictive modeling algorithms are a set of statistical techniques and mathematical equations that use historical data to predict future behavior or outcomes. These algorithms can be used to build predictive models that can identify patterns in data, forecast future trends, and make data-driven decisions

## Loading in raw data and Importing Libraries

## import pandas as pd

## import numpy as np

## import matplotlib.pyplot as plt

## import seaborn as sns

df = pd. read\_csv(‘Census\_Income.csv’)

df. head()

df.describe

## Preprocessing:

## Recoginzing Datatypes of each column: df.info()

## We have to Deal with two types of values :

## 

## Numerical [ Fill with mean values]

## Categorical Values [Fill with mode values]

## Categorical Values [Fill with mode values]

## df['Workclass'] = df['Workclass'].str.replace('?', df['Workclass'].mode()[0])

## df['Workclass'].value\_counts()

## df['Occupation'] = df['Occupation'].str.replace('?', df['Occupation'].mode()[0])

## df['Occupation'].value\_counts()

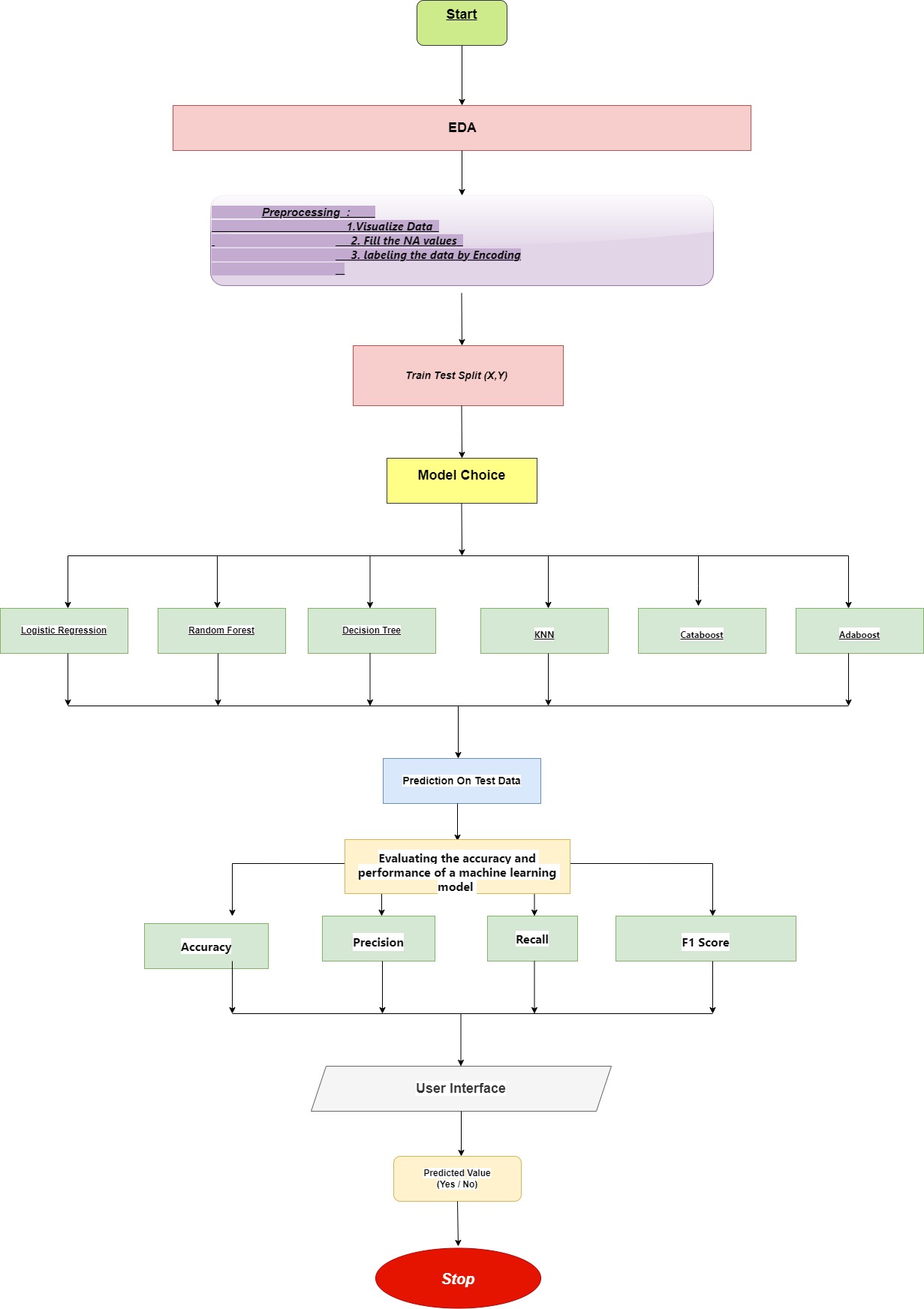
## df['Native Country'] = df['Native Country'].str.replace('?',df['Native Country'].mode()[0])

## df['Native Country'].value\_counts()

## df['Income'] = df['Income'].str.replace('.', '', regex=False)

## df['Income'].value\_counts()

## Flow Diagram:



**2.2 Exploratory Data Analysis**

Contents of the article

The following information/steps will be covered further in the article –

1. Exploratory data analysis

2. Data modeling

3. Outlier detection and skewness treatment

4. Encoding the data — Label Encoder

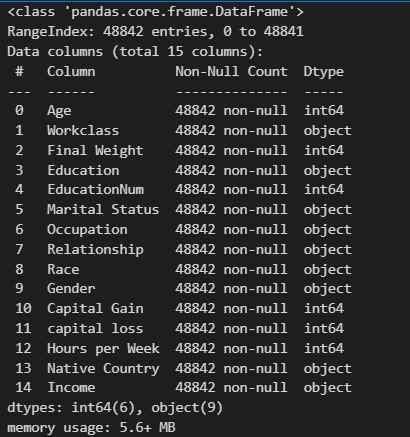
5. Scaling the data — Standard scaler

6. Fitting the machine learning models

7. Cross-validation of the selected data

1. Saving the final model and prediction using saved model
   * 1. **Exploratory Data Analysis**

The first step that we do is to check the information about our data. We see the results shown in the image below:



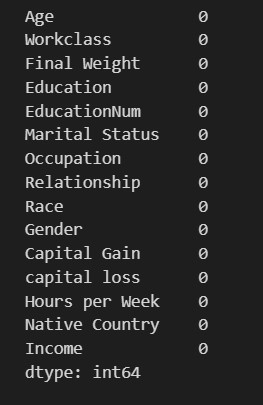
We see that we have a mixture of categorical and numeric columns. We have 6 integer columns and 9 object type columns. We observe that the count of entries is 48800 for all columns, hence no NaN values are present in our dataset.

We confirm this assumption using data.isnull().sum() command –

But, while having a close look at our dataset, we observe some of the values as ‘?’, which represent missing values.

Hence, we deduce that there are some values in our data set which need to be treated.

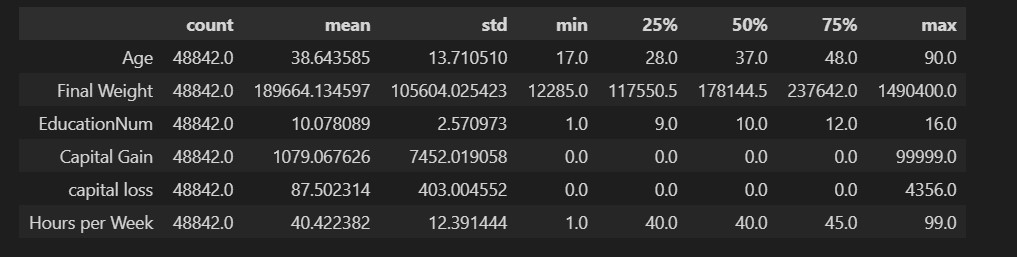
We also check the numerical statistics of our data using data.describe() command –



But, while having a close look at our dataset, we observe some of the values as ‘?’, which represent missing values.

Hence, we deduce that there are some values in our data set which need to be treated.

We also check the numerical statistics of our data using data.describe() command –



Following observations are made in this step –

- The age column has a range of 17 to 90.

- The fnlwgt column has a minimum value of 12285 and maximum value of 1484705

- The education number has a range of 1 to 16

- The capital gain starts from 0 and ends at 99999

- The capital loss starts at 0 and ends at 4356

- Hours per week range between 1–99.

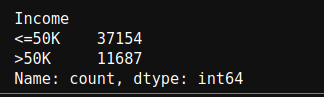
- There are outliers expected in Capital gain column as the values till 75% are 0. Same is the case with capital loss as well.

- The fnlwgt column also has a huge difference between 75% values and the max value. There is a chance of getting outliers here.

Further, we have a look at our dataset and explore the data in various columns, one by one –

**· Income column:**

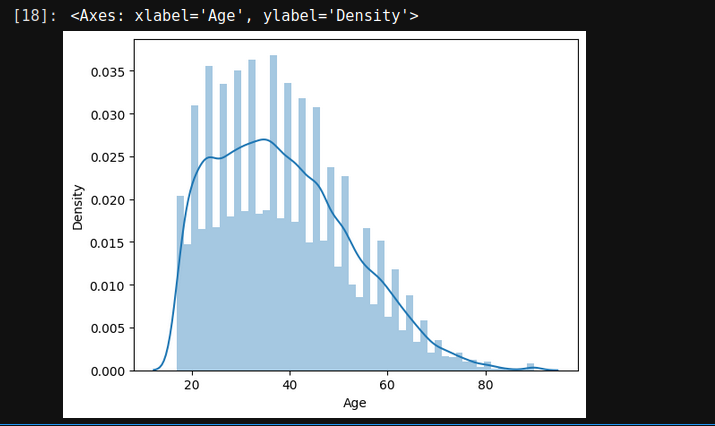
The income column is our target variable with 2 values — ‘<=50K’ and ‘>50K’. The count of these values is 37154 and 11687 respectively, suggesting that people with income higher than 50K are significantly less, and our data set is imbalanced considering the target variable.



**· Age column:**

The data in age column has a minimum value 17 and max value 90.

We create a distribution plot for the age column –

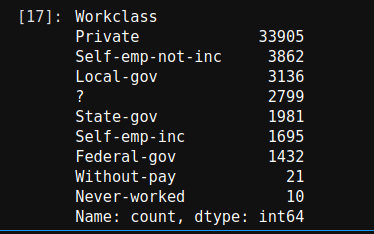


We observe that our data is has right skewness, with majority of the ages falling in the 20–50. The count keeps on decreasing as the age increases.

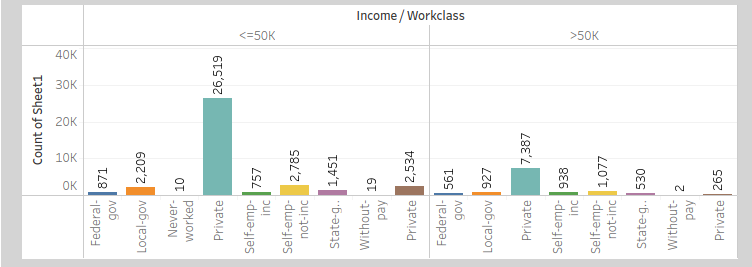
We also observe that we do not have any null values in the age column.

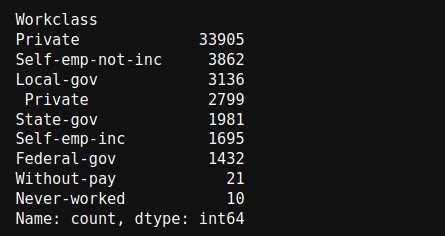
**· Workclass column:**

While checking the unique values for workclass, we see that we have 7 different types of values, along with some missing values represented by ‘?’. The count of null values is 1836, which is around 5% of the data.



We observe that majority of the people belong to ‘Private’ sector workclass.

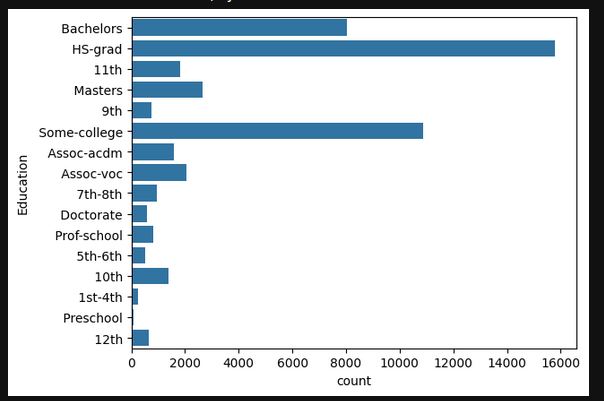




We also make an interesting discovery here — The values where ‘Workclass’ is missing, also has ‘Occupation’ missing!

· Education column

The ‘Education’ column has 16 different categories available. Majority of these categories belong to ‘School’ type (different classes are divided into multiple categories)

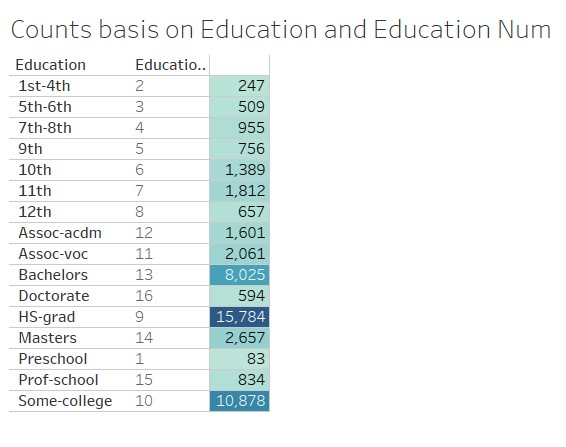


We observe no missing values in this column, and also find out that majority of the people have education level as ‘HS-grad’, followed by ‘Some-college’ and ‘Bachelors’.

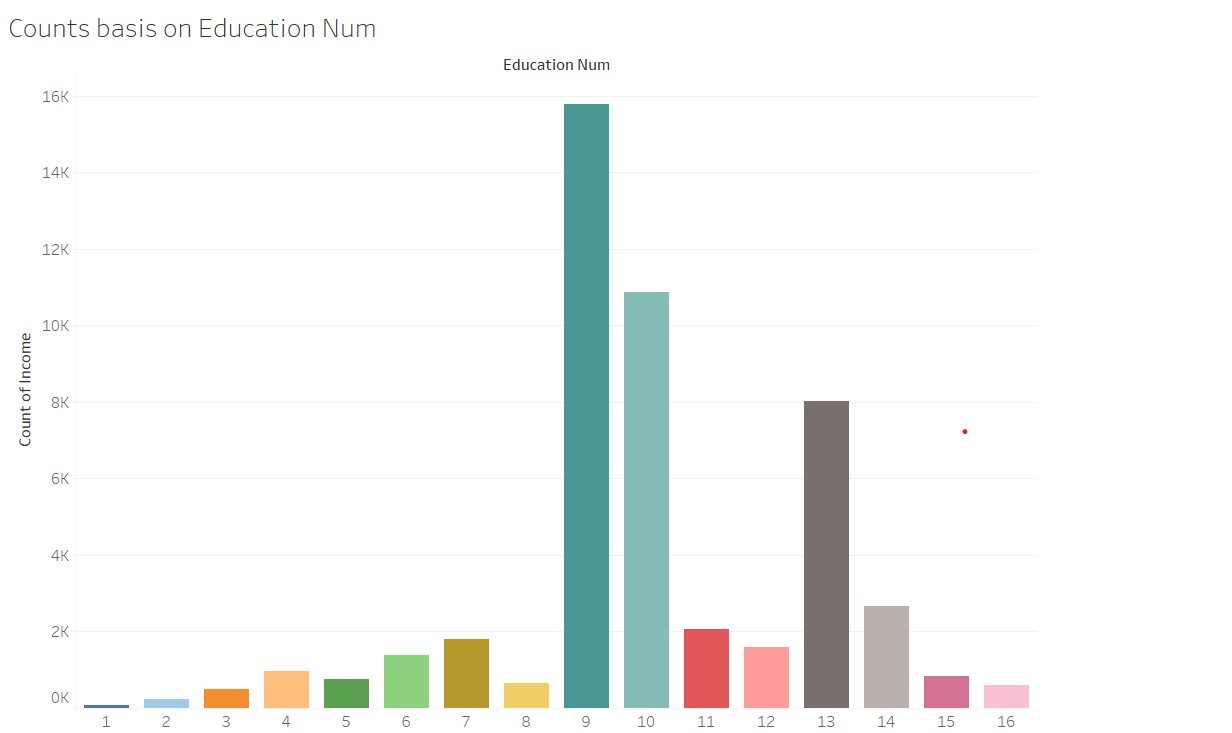
**· Education-num column:**

The education number is the number of years for which a person received education. This is an ordinal column, which contains 16 different values. When we check the division of ‘Education\_num’ column, we observe that the count of ‘Education’ column and ‘Education\_num’ is exactly same! Which means, the ‘Education\_num’ column is providing same information as ‘Education’ column, but in a numeric manner!

We can map the ‘Education’ level and ‘Education\_num’ columns by keeping the information side-by-side as is displayed below:



The count chart for ‘Education\_num’ is as follows –

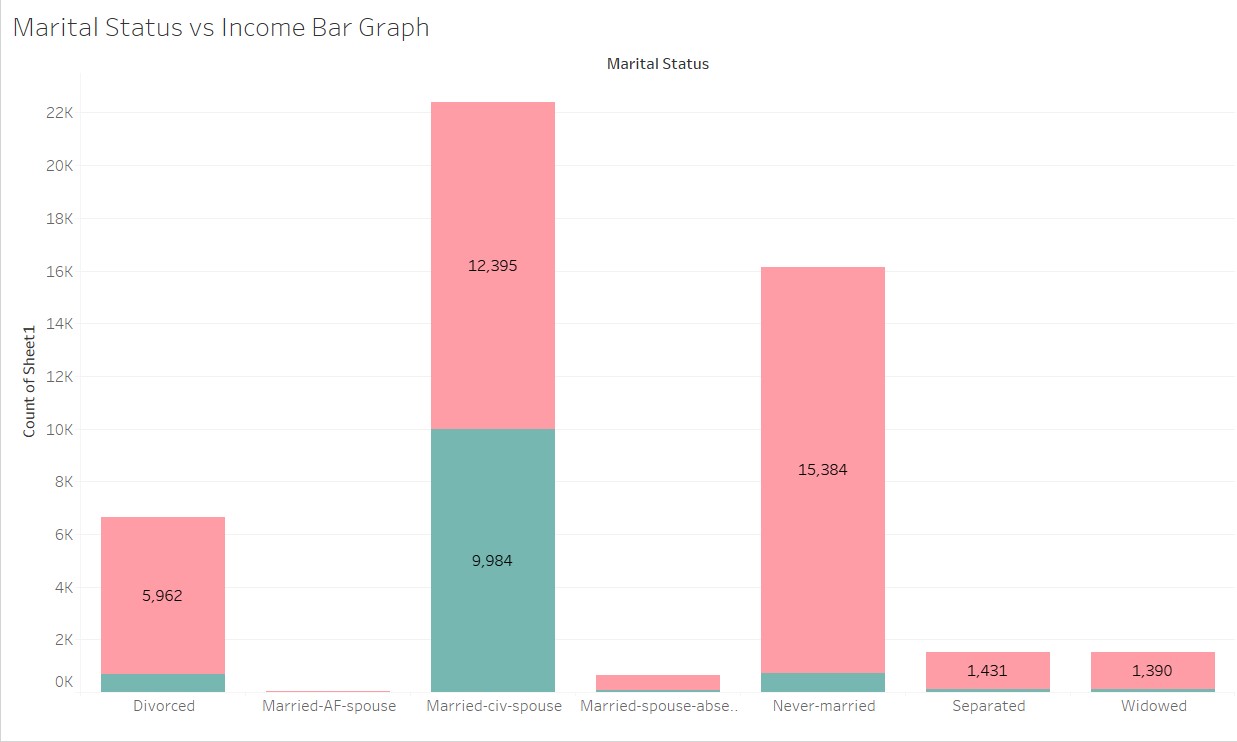


**· Marital\_Status column:**

The ‘Marital\_Status’ column has 7 different categories available, and has no missing values.

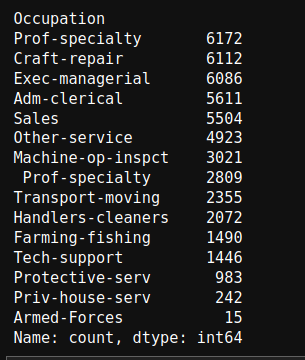
Majority of the people have ‘Marital\_Status’ as ‘Married-civ-spouse’, and least have ‘Married-AF-spouse’.

Count of ‘Never-married’ is also quite high.



**· Occupation column:**

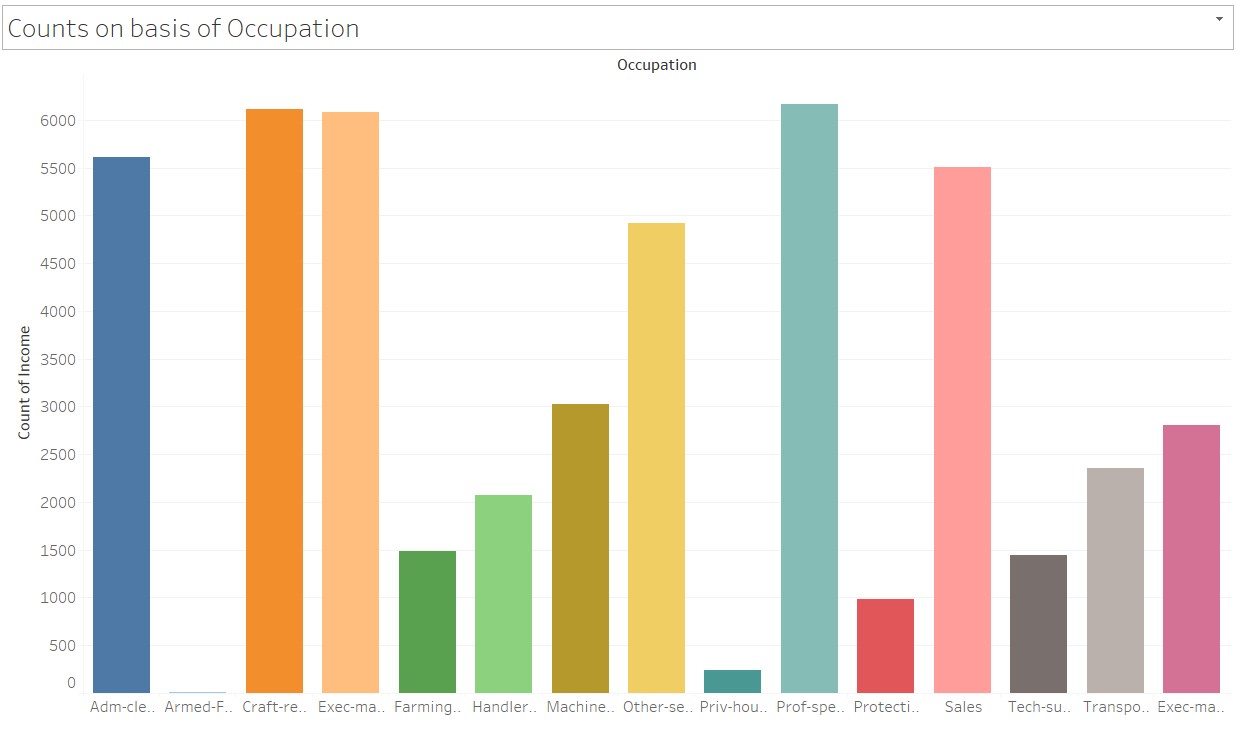
The occupation column contains 14 different categories, and have missing values represented by ‘?’ (which we have already observed, and combined with ‘Workclass’ column).



But the count of missing values is slightly higher than ‘Workclass’ column

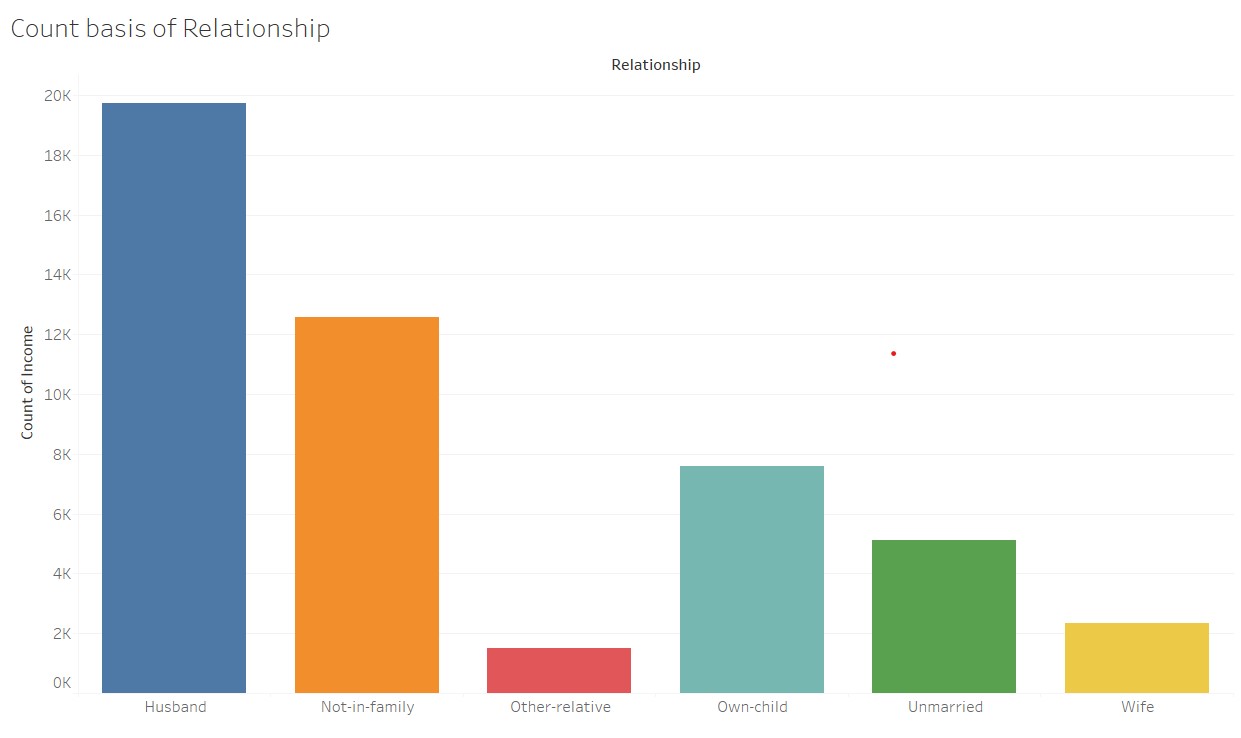
We try to find out the extra rows where the ‘Occupation’ is missing –

We observe that in case the ‘Workclass’ is ‘Never-worked’, then also the ‘Occupation’ is missing.



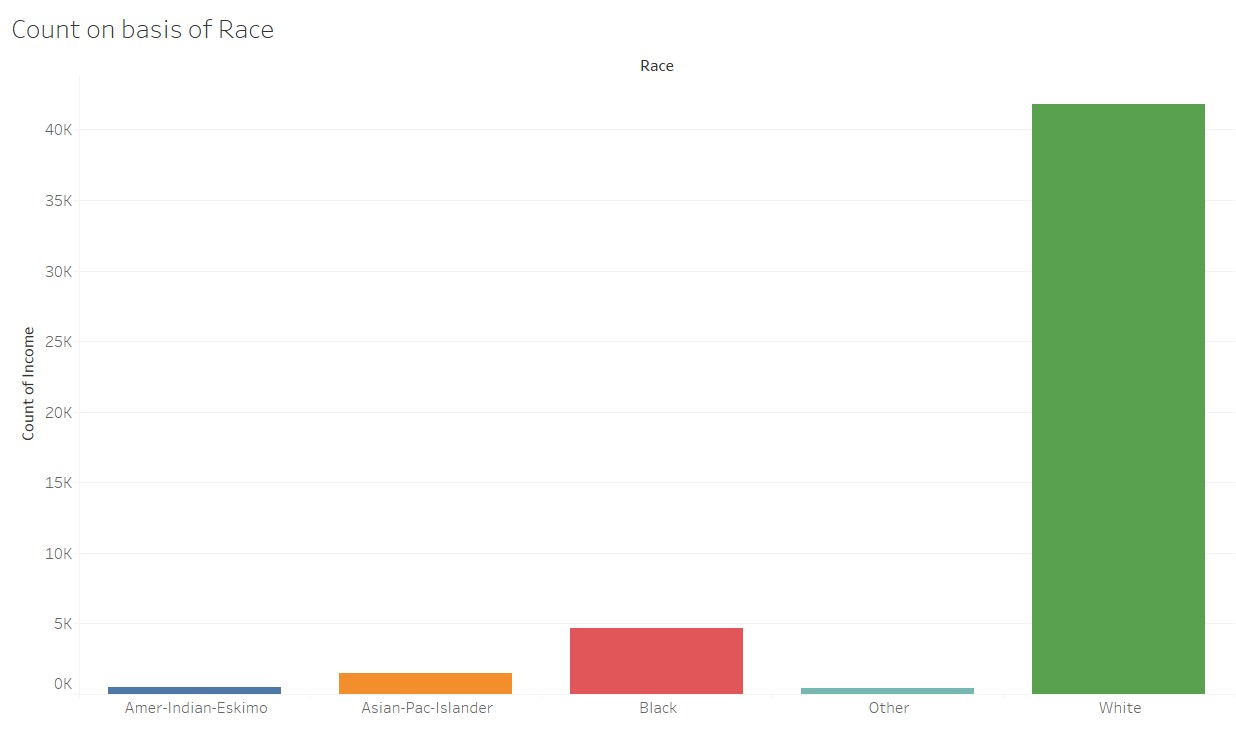
**· Relationship column:**

The relationship column contains 6 different types of values, with highest number set for ‘Husband’ and lowest for ‘Other-relative’. The column does not have any missing value.



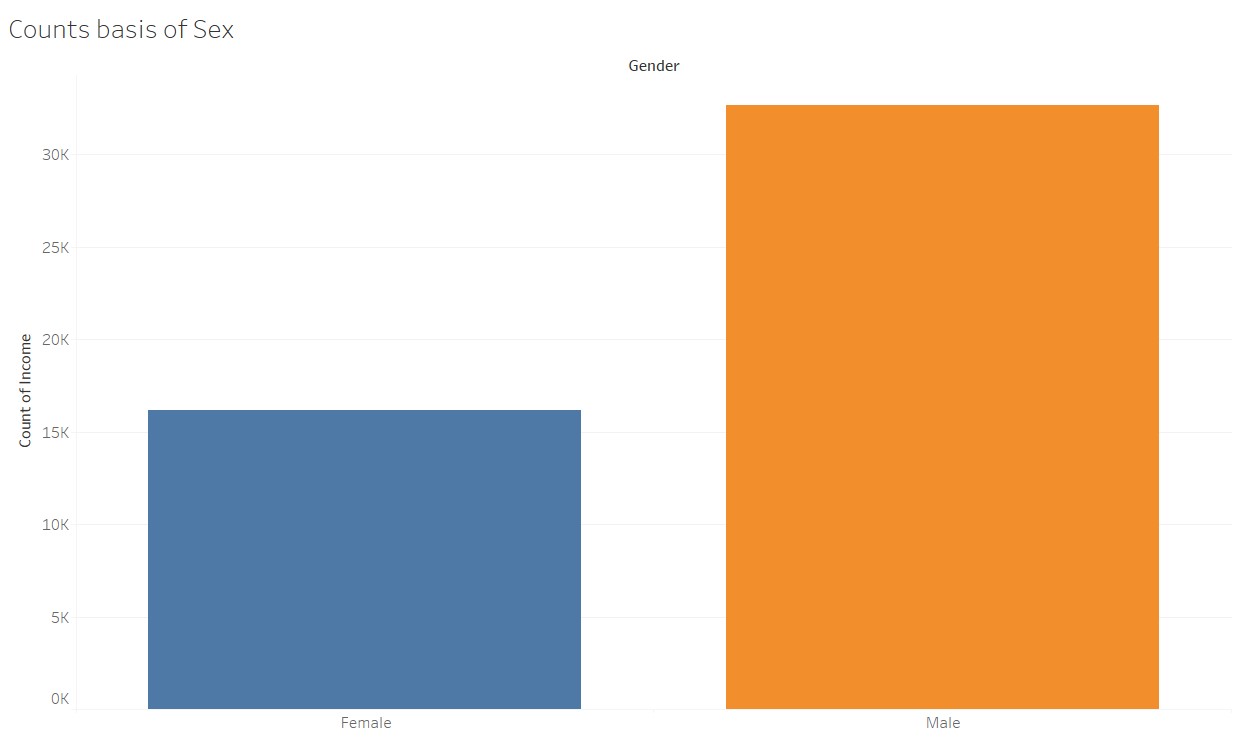
**· Race column:**

The Race column has 5 different categories, and no missing data. Highest number of people have race as ‘White’ (significantly high numbers).



**· Sex column:**

The ‘Sex’ column has 2 categories — Male and Female, where number of males are almost double to number of females. Missing values are not found in this column.



**· Capital gain column:**

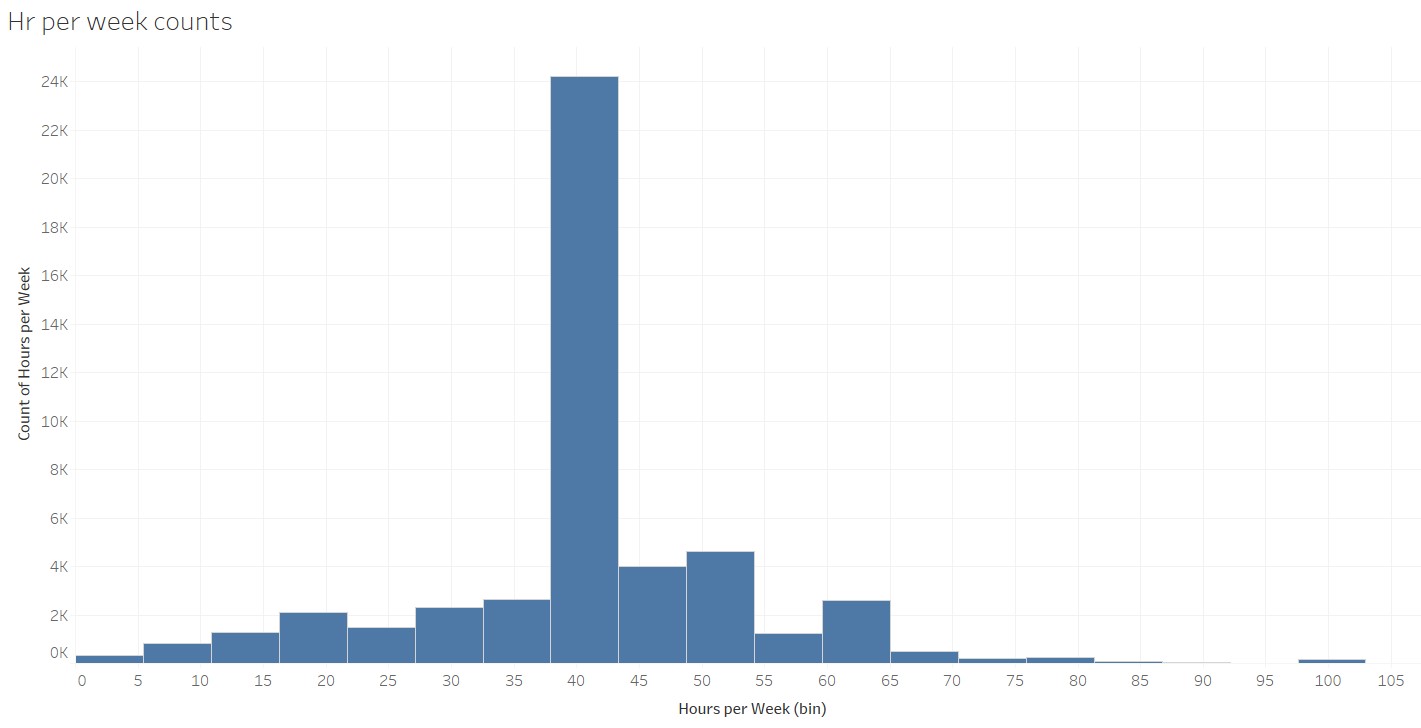
‘Capital\_gain’ column is a numeric column, with majority of the values set as 0. The distribution plot for ‘Capital\_gain’ column is highly right skewed.

**· Capital Loss column:**

The ‘Capital\_loss’ column also has majority of the values set as 0, similar to ‘Capital\_gains’. The data is highly right skewed in this case as well.

**· Hours per week column:**

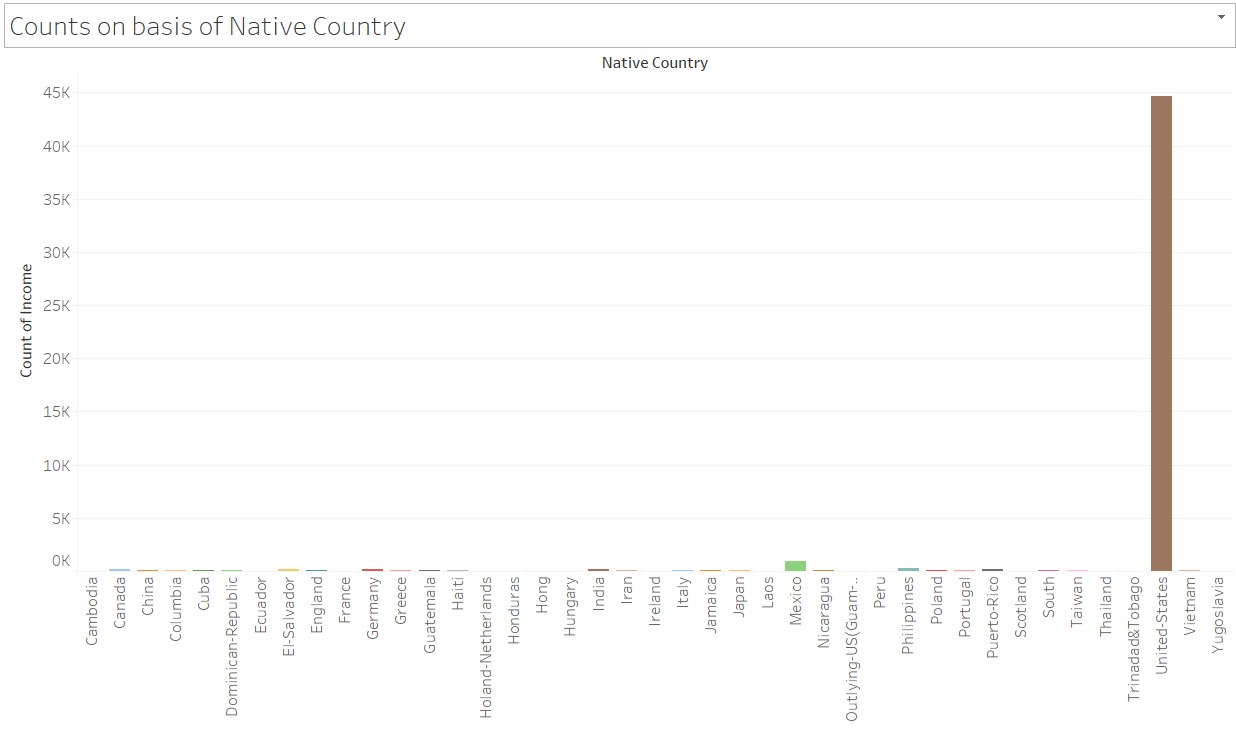
The hours per week column has values scattered over a range of 1–99. The column does not have any missing values. Majority of the values have data near 40 hours and hence a high peak can be observed in the below distribution plot –



**· Native\_country column:**

The Native\_country column contains the highest count set to ‘United-States’, and rest of the rows contain quite few numbers (highest count after US is 643).

We further check how the income gets impacted due to the features which we just explored.



Looking at the graphs above, we make the following conclusions –

1. A person has high chance of earning >50K in case his/her race is ‘White’/’Asian-pac-islander’.

2. Males have a higher chance of earning more than 50K, than females.

3. Ratio of people earning more than 50K is higher in case Workclass is ‘Self-emp-inc’.

4. People with education level as ‘Masters/Doctorate/Prof-school’ have higher ratios of >50K earning, than <=50K. Bachelors degree also has around 10:7 ratio of <=50K : >50K.

5. If the relationship in family is either ‘Husband/Wife’, the chances of earning more than 50K is high.

6. From the scatterplot between age, hours\_per\_week and income, we observe that a person needs to be >30 to be earning more than 50K, else needs to work at least 60 hours\_per\_week to earn >50K.

If we check the correlation between the numeric columns, we observe that –

Income has 34% correlation with ‘Education\_num’, 23% correlation with ‘hours\_per\_week’ and ‘age’, and 22% correlation with ‘Capital\_gain’. The correlations are moderate.

**Data modeling:**

We now proceed to an important part of our process — data modeling. Based on our analysis above, we will fill the missing values in our data, and group certain categories logically, to allow our model to learn better.

**Replacing missing values –**

We choose to replace the missing values with the mode of the data, i.e. the most frequently occurring values.

Hence, we replace ‘?’ is ‘Workclass’ column by ‘Private’, ‘Occupation’ column by ‘Prof-speciality’ and ‘Native\_country’ by ‘United\_States’.

Combining the data logically to reduce categories -

**Further, we combine the data in various columns –**

**· Workclass –**

We put ‘Never-worked’ and ‘Without-pay’ in one category, we classify ‘State-gov’ and ‘Local-gov’ as ‘Gov’, and we add ‘Self-emp-not-inc’ in ‘Private’ category, since the distributions are similar.

We obtain 5 different categories in Workclass column

**· Education –**

We combine all the columns relevant to schools in ‘School’ category, put ‘Doctorate’ and ‘Prof school’ in a single category ‘Doctorate’, ‘Assoc-acdm’ and ‘Assoc-voc’ in one category ‘Assoc’, and ‘HS-Grad’ and ‘Some-college’ in one category ‘College’.

We now obtain 6 categories of education which we feed to our machine learning model.

**· Marital Status -**

We combine ‘Divorced’, ‘Married-spouse-absent’, ‘Separated’, ‘Widowed’ and ‘Married-AF-Spouse’ to one category and name it as ‘No spouse’.

We now obtain 3 categories.

**· Relationship column -**

We combine ‘Not-in-family’, ‘Own-child’, ‘Unmarried’ and ‘Other-relative’ columns to a single category looking at the distributions, and name is as ‘Other’.

**· Race column -**

We combine the categories ‘Amer-Indian-Eskimo’ and ‘Other’ to ‘Others’ category, since they have similar distributions.

Our data categorization is now done. The next step is to identify if our data has any outliers, and deal with them.

‘Z-score (also called a standard score) gives an idea of how far from the mean a data point is. But more technically it’s a measure of how many standard deviations below or above the population mean a raw score is.’

After calculating the number of rows containing outliers, we find that we would lose 2733 rows taking a threshold value of zscore as 3. This data has 1546 rows with census income less than 50K$ and 1187 rows with higher income than 50K$. Since our dataset is already imbalanced, losing this number of rows with further increase the imbalance, and would be a significant loss if we consider the rows with income higher than 50K$.

Hence, we keep the rows and proceed with the next steps.

**Skewness treatment:**

We now proceed with treating skewness in our data, which allows us to fit our data in a symmetric distribution, which further allows our model to learn better.

The skewness of the data without any transformation is –

We treat ‘Fnlwgt’, ‘Capital\_gain’ and ‘Capital\_loss’ column for skewness, and use square-root transform and cube-root transform methods (since we cannot apply log and boxcox transform to columns where 0 values are present)

The final skewness that we receive after multiple transformations (sqrt transform for Fnlwgt column, and cbrt transform for Capital\_gain and Capital\_loss columns twice) –

We further proceed to next steps since the skewness does not get further decreased.

**Encoding the data:**

Since majority of the classification models need input as ‘int/float’, and do not work on ‘string’ data, we encode our categorical columns using ‘Label Encoder’

The final dataset looks something like –

**Scaling the data:**

The next step is to bring the data to a common scale, since there are certain columns with very small values and some columns with high values. This process is important as values on a similar scale allow the model to learn better.

We use standard scaler for this process –

‘StandardScaler follows Standard Normal Distribution (SND). Therefore, it makes mean = 0 and scales the data to unit variance’

Fitting data into classification models

We now proceed to the main step of our machine learning, fitting the model and predicting the outputs. We fit the data into multiple classification models to compare the performance of all models and select the best model –

We use the below mentioned code snipped to fit the data into ML models and predict the output –

We achieve the best results using — ‘Support Vector Classifier’, which provides an accuracy of 85% on test data. Below is the classification report for the SVC model –

We further try fitting the data to classification models to check how our ensemble models perform on the given dataset.

As a result, we observe that Random Forest Classifier over fits the train set, and gives a 99% accuracy on train data.

We see that the Gradient Boosting Classifier gives us an accuracy of ~87% (higher than SVC), and the f1-score, recall and precision scores also improve. Hence we choose ‘Gradient boosting classifier’ as our final model, and proceed with hypertuning the model. But before this, we perform k-folds cross validation on our dataset.

**Cross validation:**

The goal of cross-validation is to test the model’s ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias and to give an insight on how the model will generalize to an independent dataset (i.e., an unknown dataset, for instance from a real problem).

We obtain the following results using cross-validation –

This helps us interpret that the model is not overfitting and will perform well for new data that we feed to our model. We now proceed with hypertuning the model, using GridSearch CV.

**Hypertuning the model:**

We receive the best params for our model, which result in a best score of 87.7%. We increased our model accuracy by 1% using hypertuning.

We now save the model with best parameters that we identified using GridSearch, and create an object for our model using ‘Joblib’.

AUC ROC curve

AUC — ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0’s as 0’s and 1’s as 1’s.

We draw the AUC-ROC curve to obtain the following output –

The area under the ROC curve represents the ability of our model to predict correct values, and the curve that we got is quite a good score.

**Conclusion:**

We further proceed to test the object that we saved using joblib, and create a dataframe of predicted values –

Following are the results that we achieve, with an accuracy of 87.7%.

This marks the end of our process; we have successfully trained our model to predict the income of a person, with an accuracy of ~88%.

We moved step by step, analyzing, cleaning and modeling the data, and applied various machine learning models to achieve the desired predictions. We also tuned the model to improve the accuracy, and were able to achieve a model with quite a good accuracy.

## Results and discussion:

## def train\_model\_lg():

## from sklearn.linear\_model import LogisticRegressionCV

## model = LogisticRegressionCV(max\_iter = 1000)

## model.fit(x\_train,y\_train)

## return model

## models = [

## (train\_model\_lg(),'Logistic Regression']

## def evaluate\_model(model, model\_name):

## from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score

## 

## y\_pred = model.predict(x\_test)

## y\_true = y\_test

## accuracy = accuracy\_score(y\_true, y\_pred)\*100

## precision = precision\_score(y\_true, y\_pred)\*100

## recall = recall\_score(y\_true, y\_pred)\*100

## f1 = f1\_score (y\_true, y\_pred) \*100

## return model\_name, accuracy, precision, recall, f1

## data = []

## for (model, model\_name,\_) in models:

## data.append(evaluate\_model(model, model\_name))

## result = pd.DataFrame(data, columns=['Algorithm', 'Accuracy', 'Precision', 'Recall', 'f1'])

## result:

## Algorithm Accuracy Precision Recall f1\n"

## Logistic Regression: 82.483007 72.682661 44.867048 55.483871\n",

## GUI:

GUI is made using Flask framework. **Flask** is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

1. **GitHub Link:**

https://github.com/kartiksarode83833/DBDA\_Project.git

## 7. Conclusion:

The Census Income Prediction model aims to accurately forecast an individual's income level based on various demographic and employment-related attributes. Through a rigorous methodology involving data collection, preprocessing, model selection, evaluation, and deployment, the project achieves the following:

**Model Effectiveness:**

**Performance Metrics:** The selected model(s) demonstrate strong predictive performance with high accuracy, precision, recall, and F1 scores. The ROC-AUC score indicates the model’s effectiveness in distinguishing between income categories.

Feature Insights: Key features, such as education, occupation, and hours worked per week, prove to be significant predictors of income, providing valuable insights into income determinants.

Data Handling:

**Preprocessing:** Effective handling of missing values, categorical encoding, and feature scaling ensures the data is well-prepared for modeling. Feature engineering enhances the model’s ability to learn from the data.

Bias Mitigation: Careful consideration of ethical issues helps minimize biases related to sensitive attributes such as race and gender, ensuring fair and equitable predictions.

Model Deployment:

**Operational Readiness:** The model is successfully deployed in a production environment, capable of generating real-time or batch predictions as required. APIs and integration interfaces are developed to facilitate deployment.

Ongoing Monitoring: Continuous performance monitoring and periodic retraining ensure the model remains accurate and relevant over time.

Ethical and Practical Considerations:

**Fairness and Transparency:** The model addresses ethical considerations by evaluating and mitigating biases, ensuring that predictions are fair and do not disproportionately impact any particular group. Clear documentation and stakeholder communication enhance transparency and trust in the model.

In conclusion, the Census Income Prediction model provides a robust and fair solution for predicting income levels based on census data. By leveraging comprehensive data preprocessing, advanced machine learning algorithms, and ethical practices, the project delivers valuable insights and practical applications that can benefit various stakeholders, including policymakers, businesses, and researchers.

**8. Future Scope:**

* Building such predictive models can help us better understand the population of a country as well as the various factors affecting the growth in the economy.
* Governments can understand such factors and improve upon them leading to the growth of the country.
* We have a large enough dataset, so we can use neural networks such as an artificial neural network to build a model which can result in better performance.