**Financial Forecasting On Adult’s Income Prediction Models**

**Project Description**

Imagine you are a data analyst tasked with predicting the Income in United states across different demographic areas such as country, Gender, Race, Occupation and so on. You have access to a dataset containing detailed information on the different types of occupation, their native countries, and their Work Class they belong to. Your goal is to analyse this data and predict the user earning range of the user for providing the future Financial Investment plannings.

**Scenario 1: Personalized Investment Recommendations**

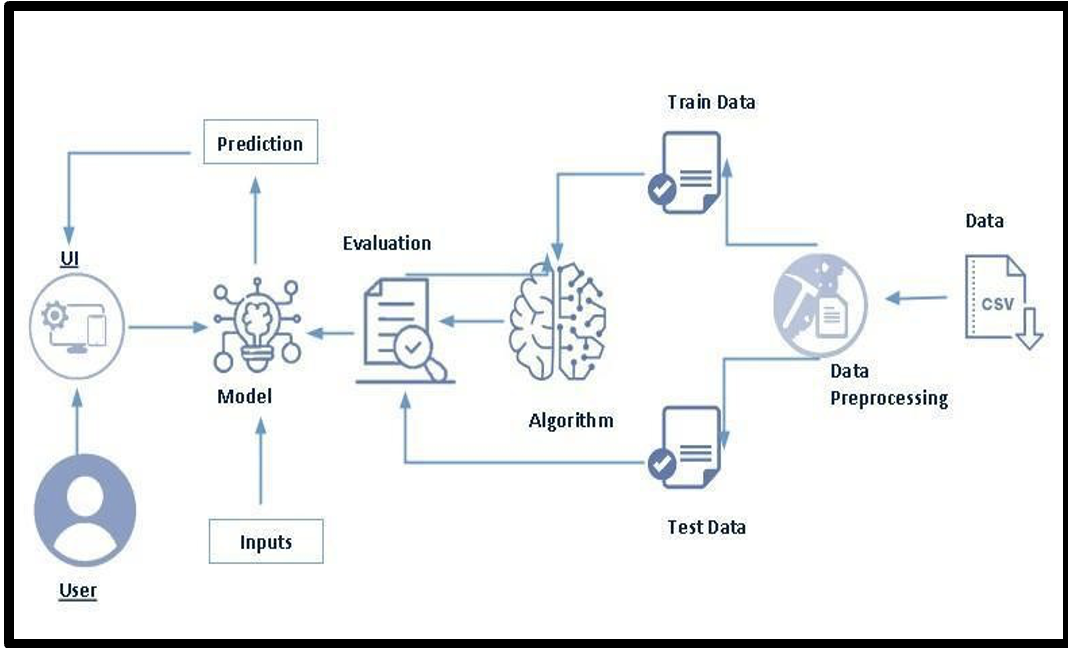
A financial advisory firm seeks to enhance its service by offering personalized investment recommendations to their clients. Understanding each client's financial capacity and investment needs is crucial for providing relevant and effective advice. By predicting an individual’s income using demographic and occupational data, the firm can categorize clients into different income brackets. This enables the firm to offer customized investment plans that align with the financial goals and capacities of each client, ensuring higher client satisfaction and better investment outcomes.

**Scenario 2: Targeted Marketing for Investment Products**

An investment company plans to launch a new premium investment product aimed at high-income individuals. The company needs to identify and target high-income individuals for their marketing campaigns to maximize the return on investment. Using the income prediction model, the company can analyse its customer database and segment it based on predicted income levels. This allows the marketing team to focus their efforts on individuals who are more likely to have an income exceeding $50,000, thereby increasing the efficiency and effectiveness of their marketing campaigns.

**Scenario 3: Optimizing Financial Planning Services**

A financial planning service aims to optimize its service offerings based on the income levels of its clients. Different income groups have varying financial planning needs and risk tolerance levels. The prediction model can help the service identify the income range of their clients. With this information, financial planners can develop and offer tailored financial plans that match the specific needs and risk profiles of different income groups, improving client satisfaction and financial outcomes.

**Technical Architecture:**

**Project Flow:**

* User interacts with the UI to enter the input.
* Entered input is analysed by the model which is integrated.
* Once model analyses the input the prediction is showcased on the UI

**To accomplish this, we have to complete all the activities listed below:**

* **Data Collection & Preparation** 
  + Collect the dataset
  + Data Preparation
* **Exploratory Data Analysis** 
  + Descriptive statistical
  + Visual Analysis
* **Model Building** 
  + Training the model in multiple algorithms
  + Testing the model
* **Performance Testing** 
  + Testing model with multiple evaluation metrics
  + Comparing model accuracy before & after applying hyperparameter tuning
* **Model Deployment** 
  + Save the best model
  + Integrate with Web Framework

**Prior Knowledge:**

You must have the prior knowledge of the following topics to complete this project.

* ML Concepts:
* Supervised learning: [https://www.javatpoint.com/supervised-machine-learning](http://www.javatpoint.com/supervised-machine-learning)
* Logistic Regression: https://www.geeksforgeeks.org/understanding-logistic-regression/
* Gradient boosting regressor: https://www.geeksforgeeks.org/ml-gradient-boosting/
* Decision Tree Classifier: https://www.geeksforgeeks.org/decision-tree/
* Flask Basics: [https://www.youtube.com/watch?v=lj4I\_CvBnt0](http://www.youtube.com/watch?v=lj4I_CvBnt0)

**Project Structure**

|  |  |
| --- | --- |
|  | * We are building a flask application which needs HTML pages stored in the Template folder and python script app.py for scripting * dtree.joblib is our saved model. Further we will use this model for flask integration. * Training folder contains a model training file. |

**Milestone 1: Data Collection & Preparation**

ML depends heavily on data. It is the most crucial aspect that makes algorithm training possible. So, this section allows you to download the required dataset.

**Activity 1: Collect the Dataset**

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc. In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the dataset.

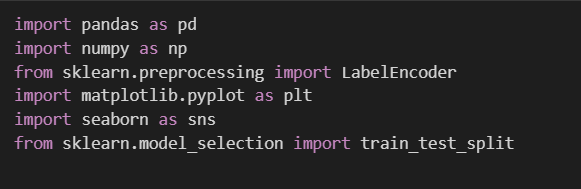
Dataset: [LINK](https://www.kaggle.com/datasets/lovishbansal123/adult-census-income/data)

As the dataset is downloaded. Let us read and understand the data properly with the help of some visualisation techniques and some analysing techniques.

**Note:** There are a number of techniques for understanding the data. But here we have used some of it. In an additional way, you can use multiple techniques.

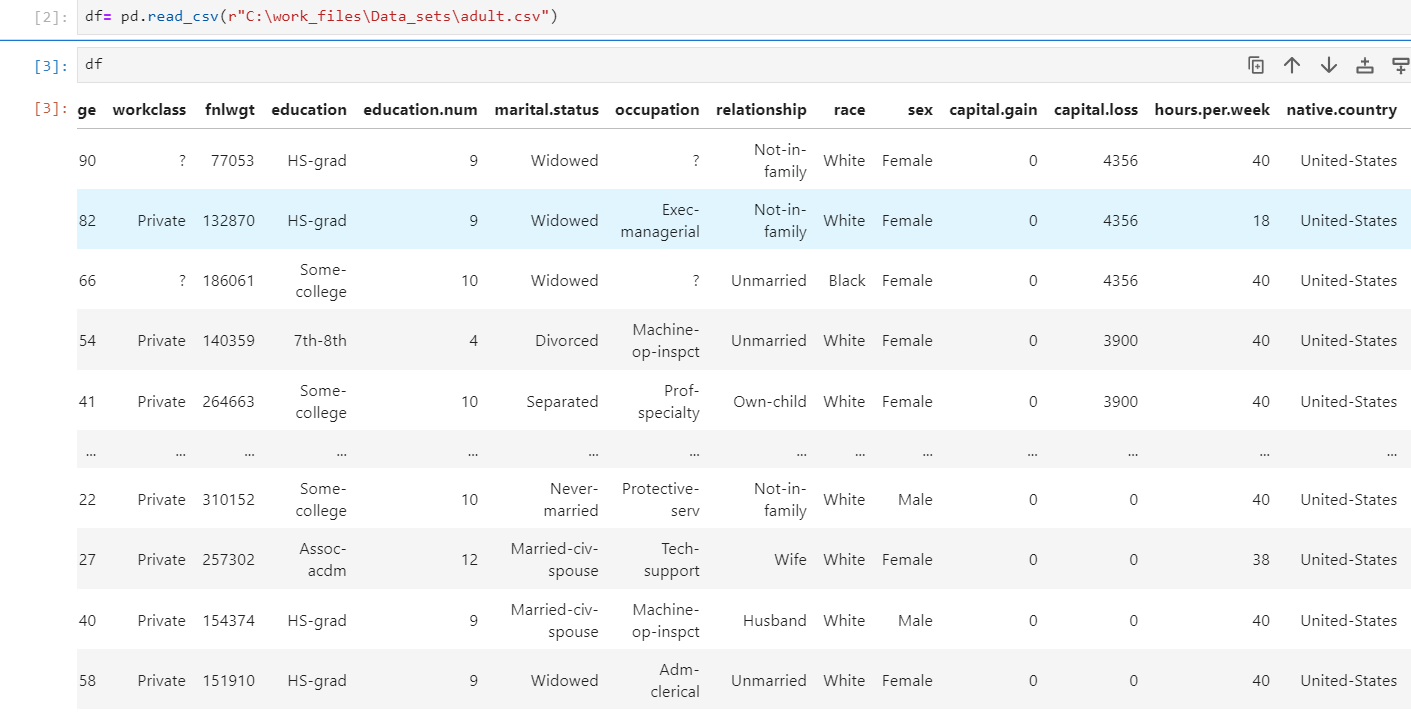
**Activity 1.1: Importing the libraries**

Import the necessary libraries as shown in the image.



## Activity 1.2: Read the Data set

Our dataset format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas. In pandas we have a function called read\_csv() to read the dataset. As a parameter we have to give the directory of the csv file.



**Activity 2: Data Preparation**

As we have understood how the data is, let's pre-process the collected data.

The download data set is not suitable for training the machine learning model as it might have so much randomness and noise .so we need to clean the dataset properly in order to fetch good results.

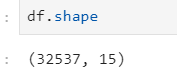
This activity includes the following steps.

* Handling missing values
* Handling categorical data
* Handling Outliers

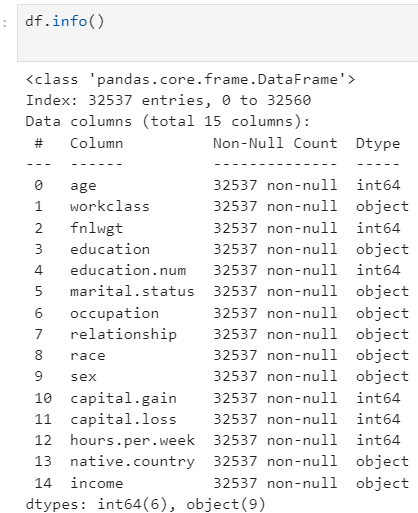
Note: These are the general steps of pre-processing the data before using it for machine learning. Depending on the condition of your dataset, you may or may not have to go through all these steps.

**Activity 2.1: Handling Missing Values**

Let’s find the shape of our dataset first. To find the shape of our data, the df. shape method is used. To find the data type, df.info () function is used.



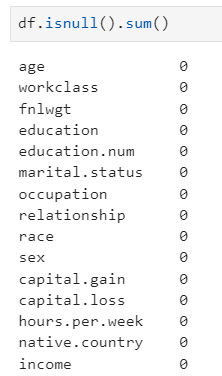
Above Figure Describes the Shape of the Dataset i.e, there are 32537 rows and 15 columns including the Target column as well.



df.info() provides the information about the column’s datatype and provides the count of non-null values in the column is concerned.

Dataset do not have any missing values.

For checking the null values, df.isnull() function is used. To sum those null values, we use .sum() function. From the below image we found that there no null values present in our dataset:



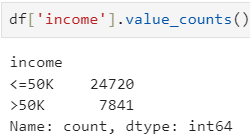
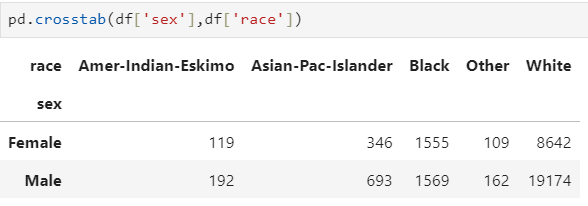
From the above figure we can observe that there are no null values present in the dataset.

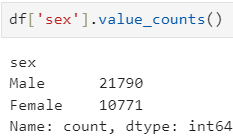
**Activity 2.2: Handling Categorical Values:**

There are multiple categorical columns present in the dataset, they are Work class, Sex, Race, Education, Occupation, Relationship, Martial Status and native country.

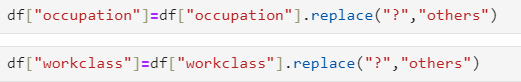
As we know they are no missing values/ null values present in the dataset. We need to know their number of categories present in each column with their counts.

There are several operations to find different insights using categorical values some of the functions are **value\_counts**, **cross\_tab(),** **mode**, **replacement of values.**



There are few unknown values in the occupation and Work class column representing in “?” so we need to treat them by replaceing them with the general category as “Others” .



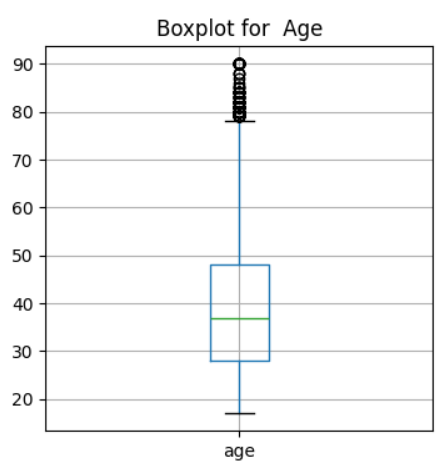
At last we have all the values regarding the categorical column. For the further process process of model building we have to encode the categorical values with numerical values we have used label encoding the process of label encoding can be obtained form the below snippet of code.



**Activity 2.3: Treating Outliers:**

Outliers are the abnormal data which are away from the range of the distribution of the data of each column in the data. Here we have the box plot to find whether the Outliers present or not

but we cannot know how many Outliers present in the Age column.

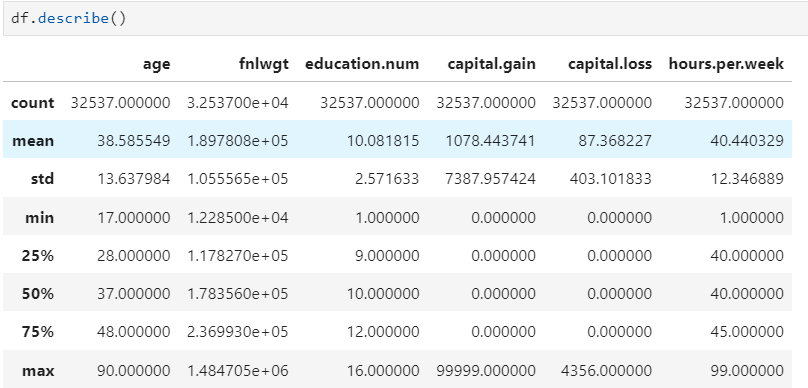


Actually we are preparing the model for the Adults income so adults age ranges from 20 to 90 but we must not consider the person whose age is 90 is not an outlier even though we need not to change or modify the Age column. It is observed that there is no data present below the lower limit of the data.

**Milestone 2: Exploratory Data Analysis**

**Activity 1: Descriptive Analysis**

Descriptive analysis involves examining fundamental characteristics of data using statistical methods. It provides insights into the mean, standard deviation, minimum, maximum, and percentile values of continuous features.

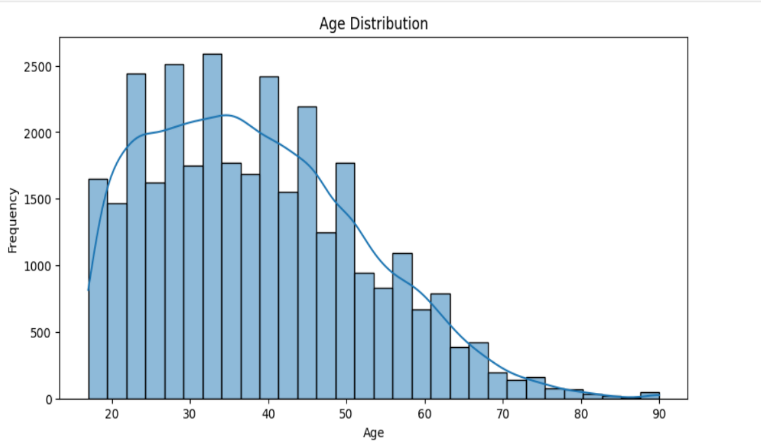
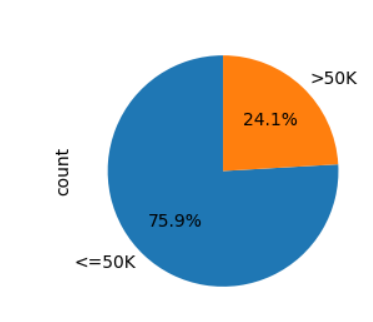


**Activity 2: Visual analysis**

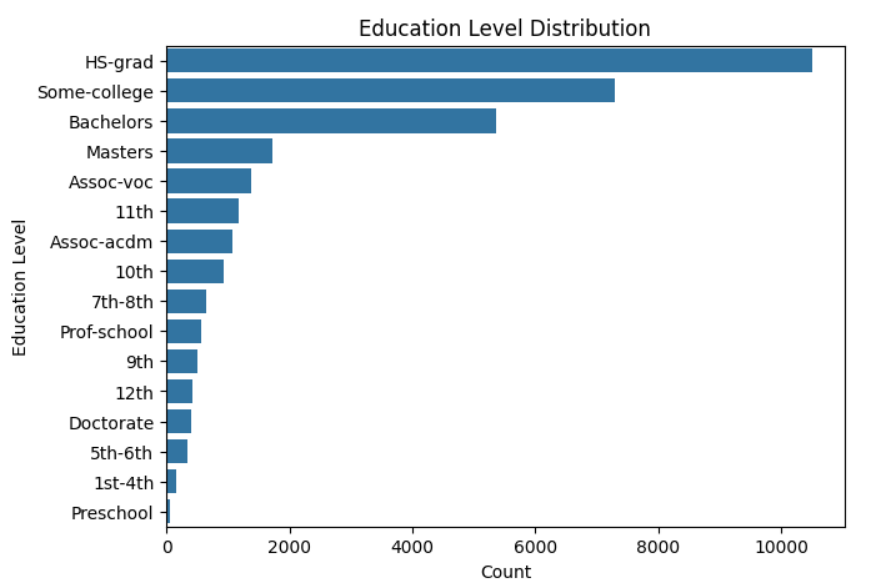
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

**Activity 2.1: univariate analysis**

In simple words, univariate analysis is understanding the data with single feature. Here we have displayed a histogram, Pie plot and Horizontal bar plot.

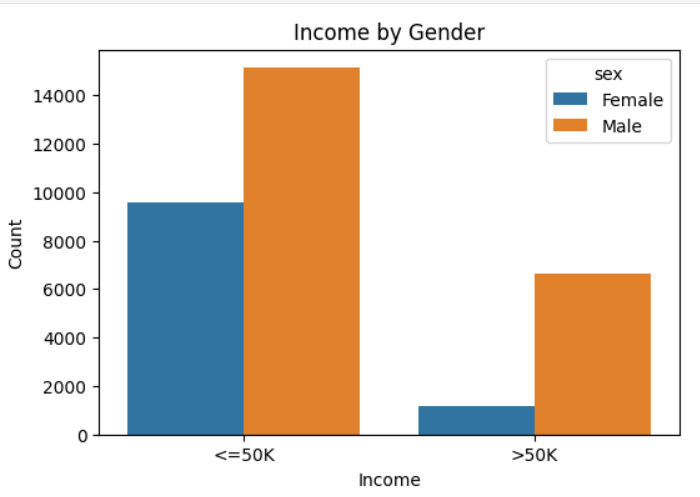
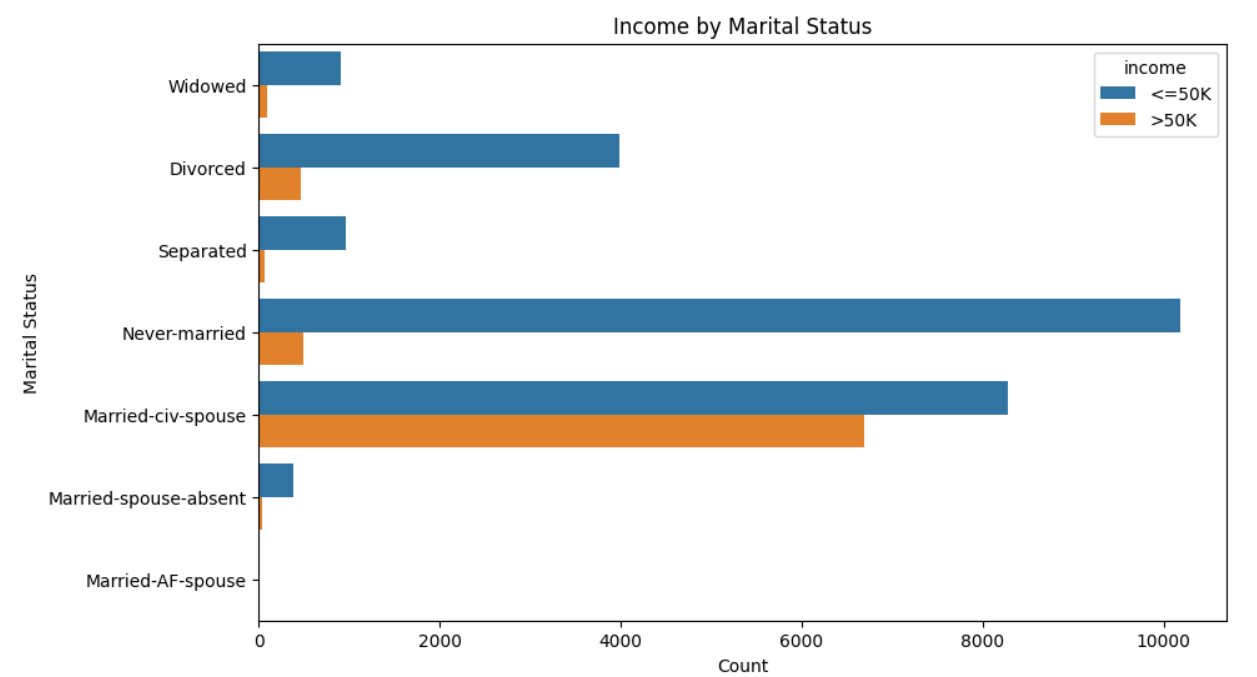
The histogram describes the distribution of Age in the dataset and a KDE plot is being added to find skewness of the age and range of distribution of ages. A pie plot is being displayed it represents the percentage of income earning by the Adults from the Data.



Above Horizontal histogram represents the educational Background of adults.

**Activity 2.2: Bivariate analysis**

Bivariate analysis is a statistical method that involves the analysis of two variables to determine the empirical relationship between them. Here we have grouped bar plots and Horizontal Grouped Bar charts.

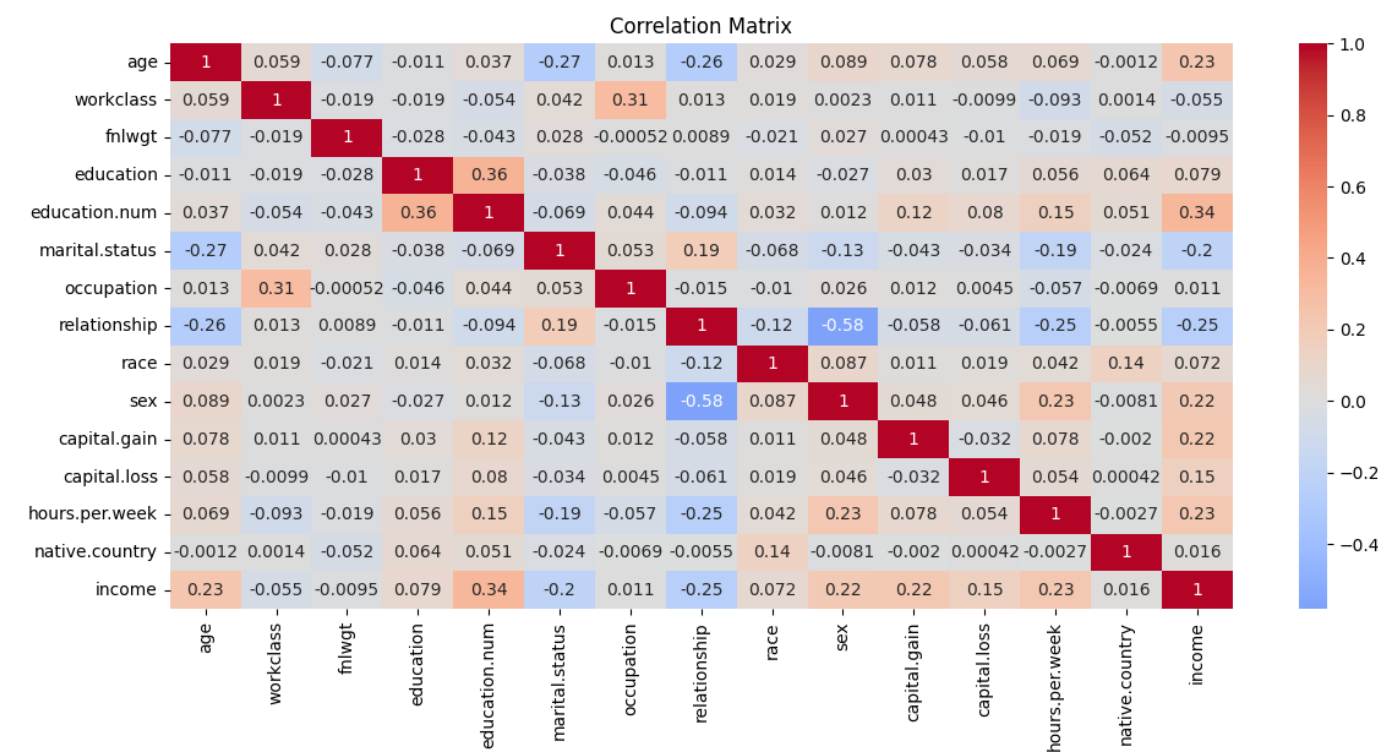
**** ****

Above Grouped bar char represents the income ranges with respect to the gender is concerned it provides the insights regarding the frequency of male earning on both income categories with represents Blue in colour and female with Orange in colour,

Horizontal Grouped Bar chart represents the earning ranges of by each category of marital status. from the chart blue coloured bar represents the income with <=50k and Orange coloured bar represent the income with >50k.

**Activity 2.3: Multi-variate analysis**

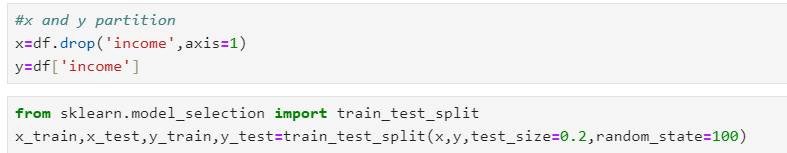
Multi-variate analysis is a statistical method that involves the analysis more than 2 variables to determine the empirical relationship among them. Here we have a heatmap representing the correlation among the variables in the Data.

****

**Activity 4: Splitting data into train and test**

Now let’s split the Dataset into train and test sets. First, split the dataset into x and y and then split the data set. “x” represents the whole data columns other than the target column,“y” represents the Target column in the dataset. We need to build the model by giving the training to the model and make the predictions on the test data.so we need to divide the whole dataset into training and testing data.

For splitting training and testing data we are using train\_test\_split () function from sklearn. As parameters, we are passing x, y, test\_size, random\_state.



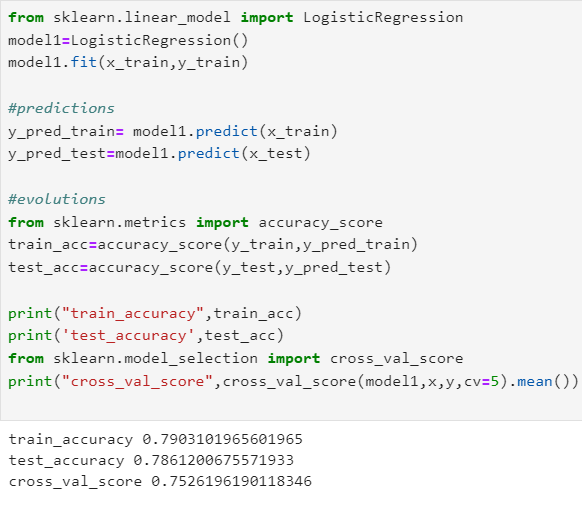
**Milestone 3: Model Building**

**Activity 1: Training and testing the models using multiple algorithms**

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying classification algorithms. The best model is saved based on its performance.

**Activity 2.1 Logistic Regression:**

Logistic Regression is the base line model where we import the Logistic Regression from the sklearn library and importing the linear\_model and assigning the model to the variable model1 and making the data to fit and moulding the data with respect to the training data.

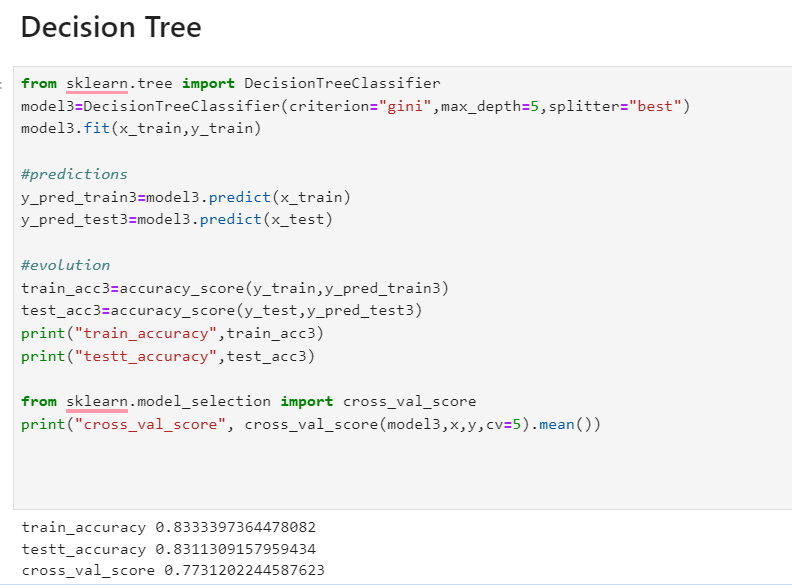


We make the predictions on the train and test data using predict function and assigning the predicted values to the y\_pred\_train and y\_pred\_test

We are calculating the accuracy scores on how the model is working on the data and we use cross-validations to reduce the Bias and trade condition to the dataset. Actually we are able to divide the dataset based on the chosen test size. If CV=4. then we are training the model with 75% of data and we are testing with 25% of data. If CV=5 then we are training the model with 80% of data and testing the model with 20% of data.

**Activity 2.2: Decision Tree Classifier:**

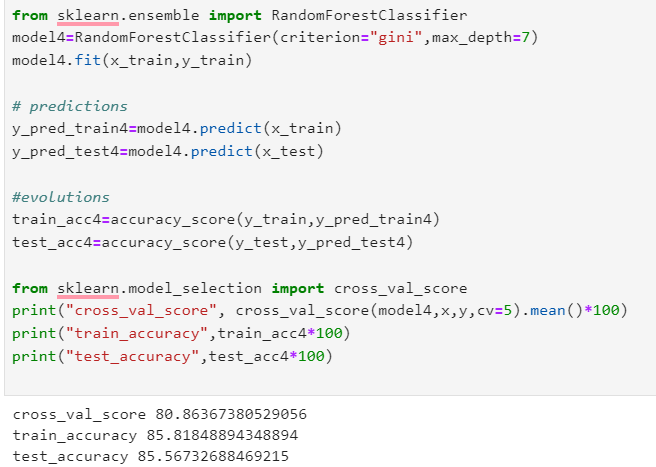
Decision Tree Classifier and regression algorithm is initialized and the training data is passed to the model and assigning the variable as model3 with the .fit() function. Test data is predicted with model3.predict() function and saved in a new variable. For evaluating the model, accuracy is calculated

****

At the end we need to observe the train accuracy and test accuracy with cross-validation score. There is a constrain that the difference of train and test accuracy should not be more than 5%and the difference of test accuracy and cross-validation should not be more than 5% the best model is fixed by this scenario.

## Activity 2.2 : Random-Forest Classifier

Random Forest algorithm is the classification and regressions algorithm initialized and training data is passed to the model and assigned to the variable as model4 with .fit() function. Test data is predicted with model4.predict() function and saved in a new variable. For evaluating the model accuracy is calculated. For the best obtaining of accuracy we use hyper parameter tuning to tune the model with the best hyper parameters using the Grid Search CV by choosing the best params we can able to get the best accuracy this method is known as Hyper parameter Tuning.



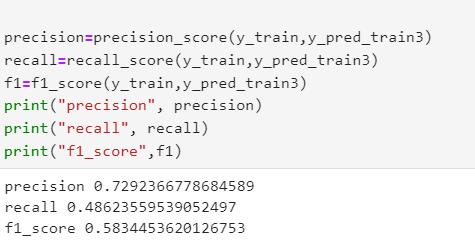
## Activity 2.3 : Ada Boost Classifier

Ada Boost Classifier algorithm is initialized and training data is passed to the model and assigned to the variable as model5 with .fit() function. Test data is predicted with model5.predict() function and saved in a new variable. For evaluating the model accuracy is calculated.



**Milestone 4: Performance Testing**

Under performance Testing we need to test the model’s accuracy with different testing Metrics like Precision, Recall and F1\_score. Below are the performance Metrics of final fixed model



# **Milestone 5: Model Deployment**

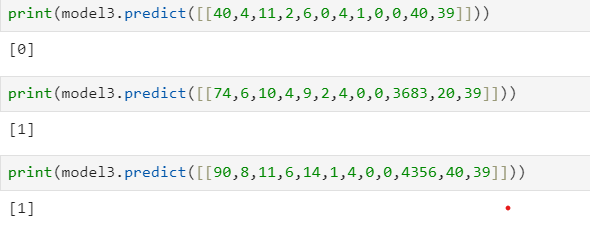
# **Activity 1: Save and load the best model**

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.

****

## Activity 2 : Test the model

Let’s test the model first in python notebook itself. As we have 7 features in this model, let’s check the output by giving all the inputs.



The predicted value and actual value are results same.

**Activity 3: Integrate with Web Framework**

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

* Building HTML Pages
* Building server-side script
* Run the web application

**Activity 3.1: Building Html Pages:**

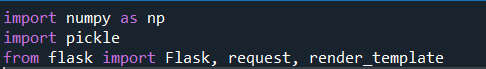
For this project create two HTML files namely

* index.html
* predict.html
* result.html

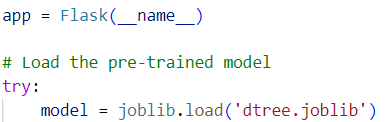
and save them in the templates folder.

**Activity 3.2: Build Python code:**

Import the libraries

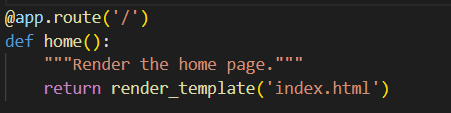


Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (\_\_name\_\_) as argument.



We render index.html for the displaying the web application , similarly we render the predict.html for the user input values of the forms to predict the income. Simultaneously we render result .html to display the result of the prediction value.

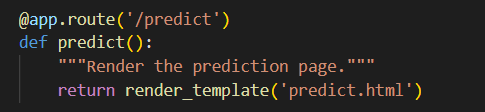
Render Index.html:



Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, ‘/’ URL is bound with the index.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered.

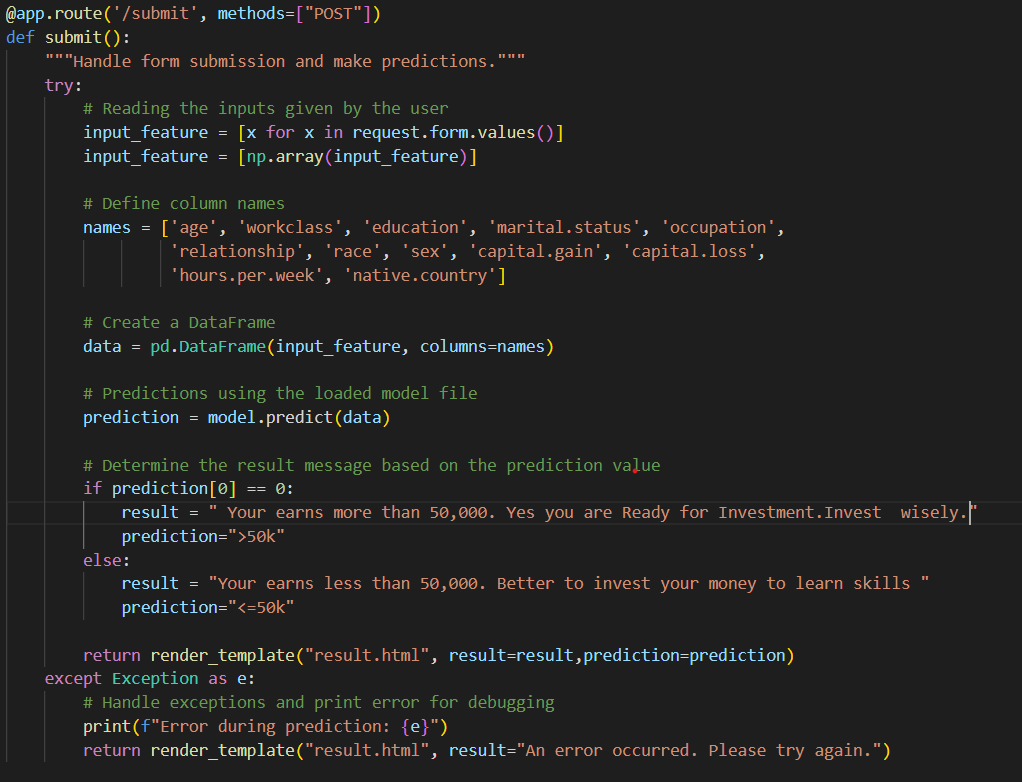
Render Predict.html:



In the predict.html where we provide the user inputs in the form for the prediction of income

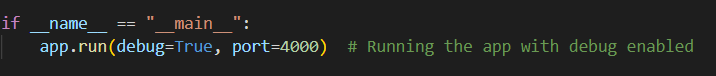
Whenever you enter the values from the html page the values can be retrieved using POST and GET Methods.

Retrieving the value from UI:



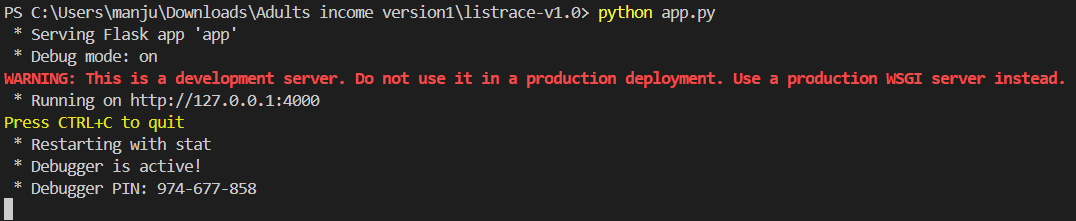
Here we are routing our app to conditional statement. This will retrieve all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

Main Function:



**Activity 3.3: Run the web application**

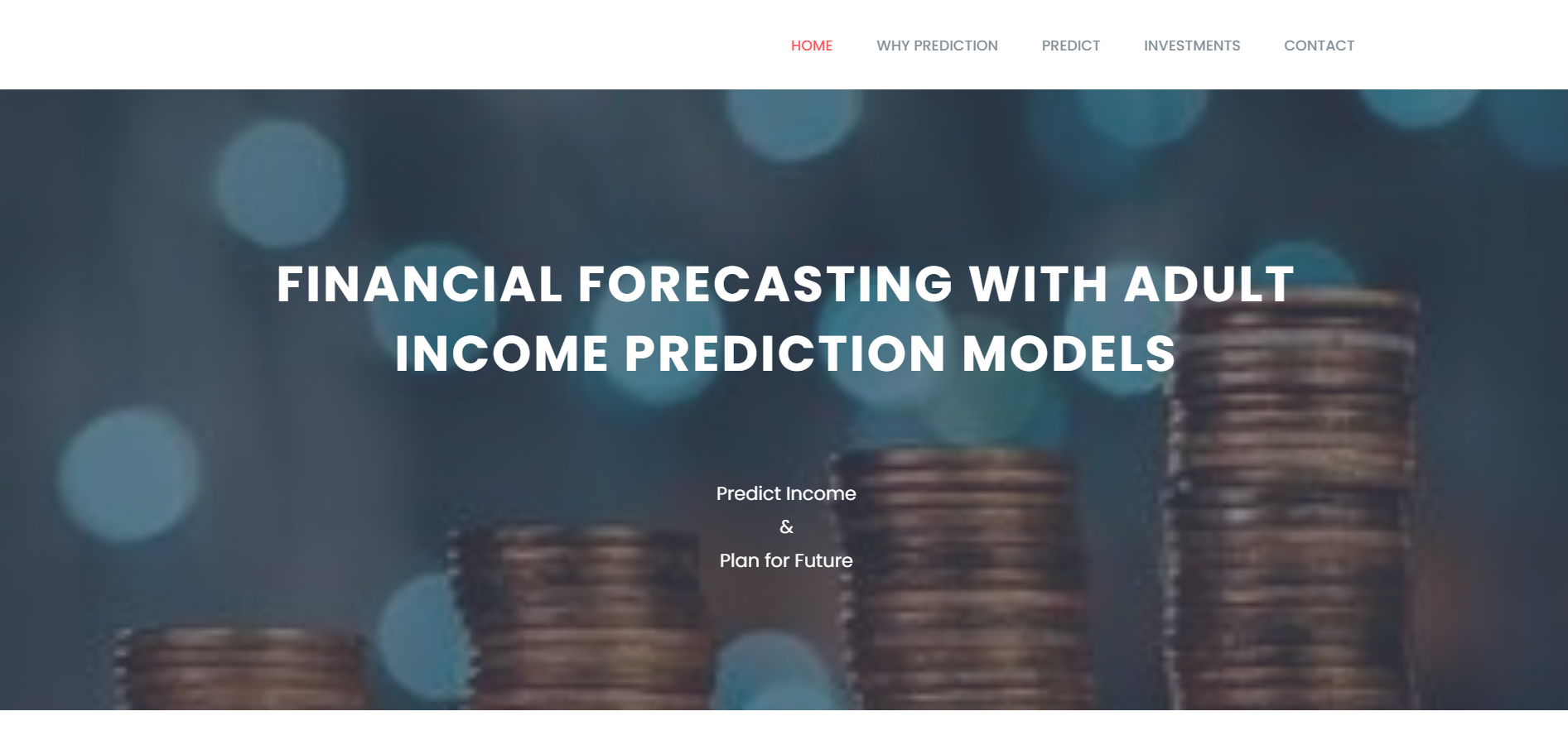
* Open vs code application in the search menu.
* Navigate to the folder where your flask folder of your files exist.
* Click on the view button in the vs code nav bar and click on the terminal option in the dropdown menu.
* Now type “app.py” command
* You will have a link displayed in the terminal as “<http://127.0.0.1:4000> “.
* Double click on the link then you will be navigated to the web application.
* Click on the predict button in the nav bar , enter the inputs, click on the predict button, and see the result/prediction in the result.html..

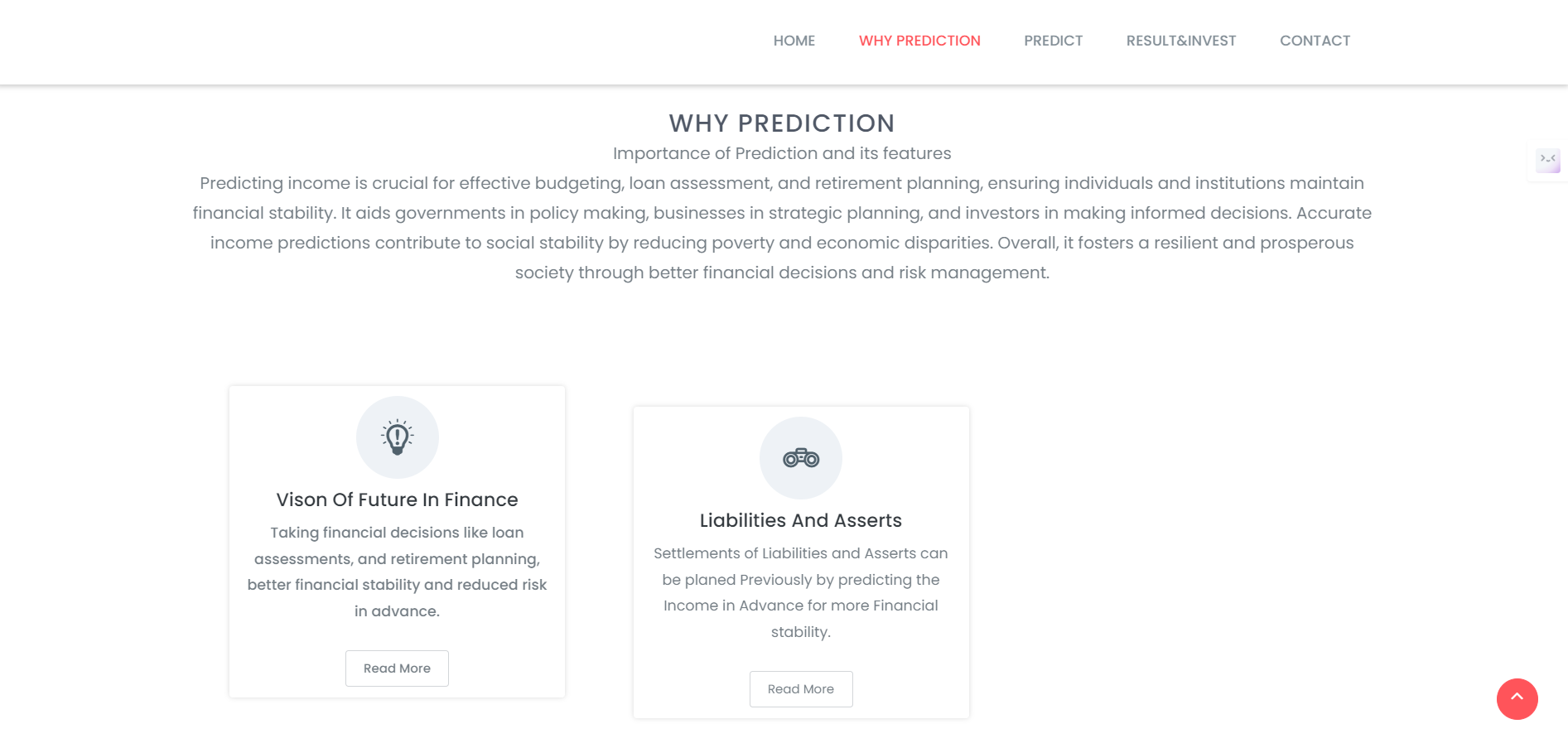


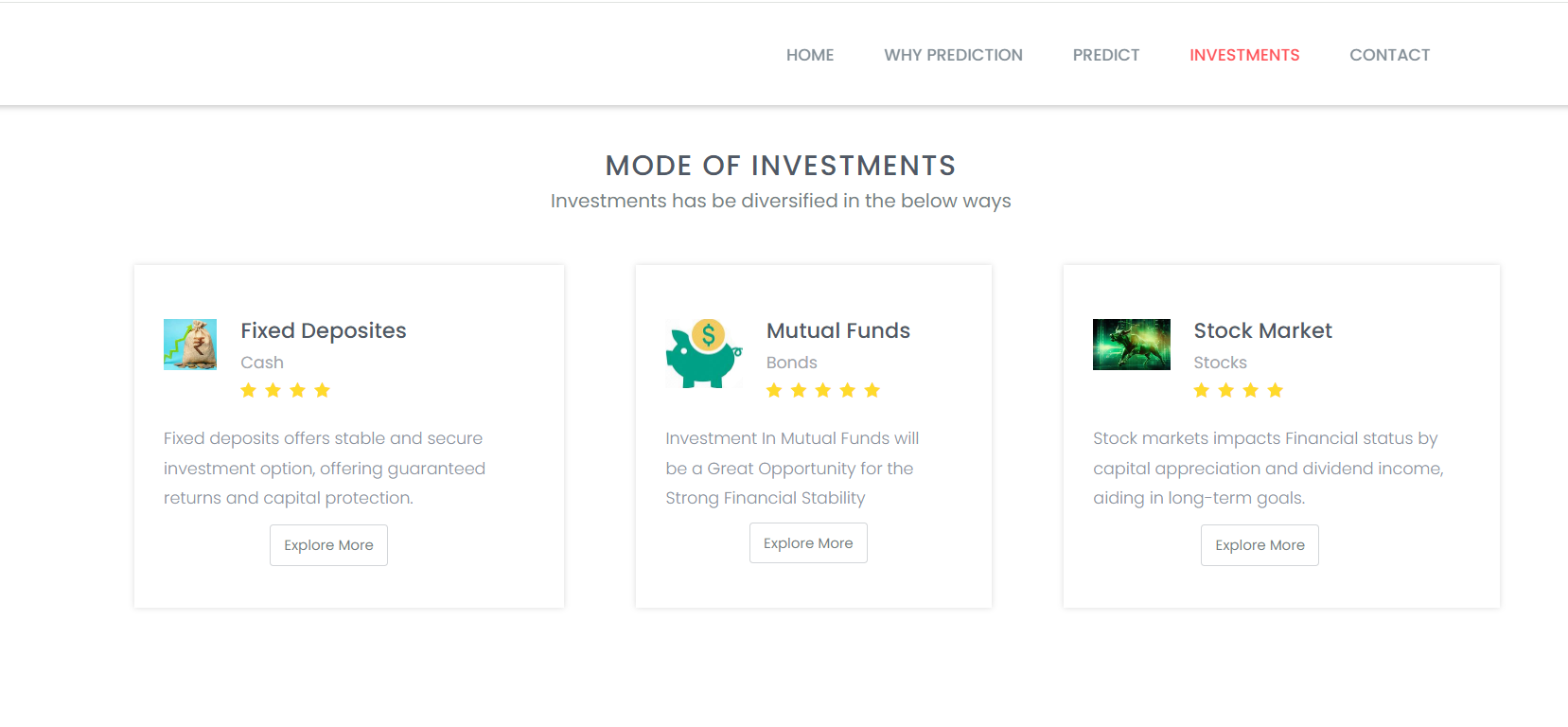
Now, Go the web browser and write the localhost URL (http://127.0.0.1:4000) to get the below results

Results:

1. Index page (Index.html)

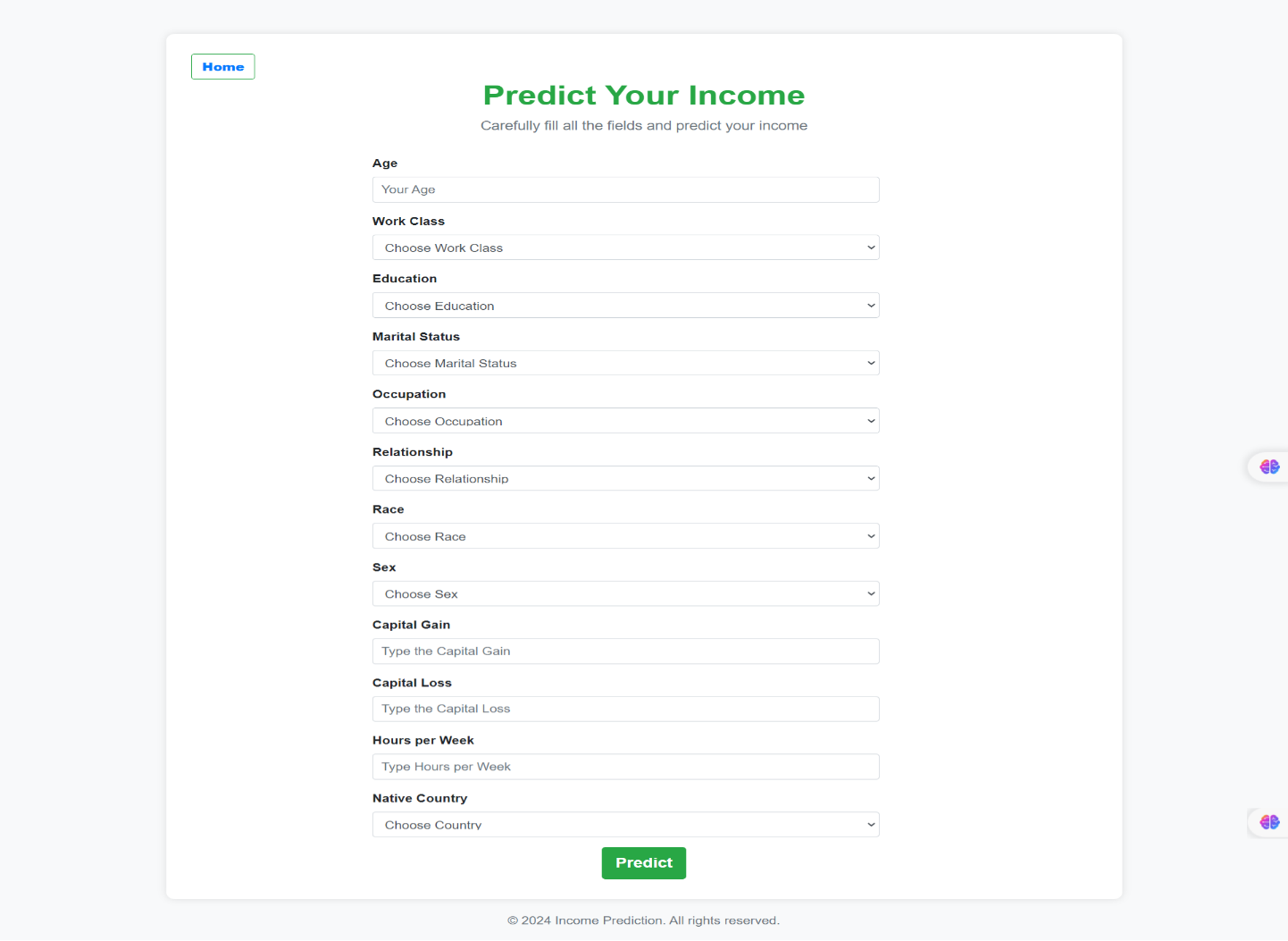






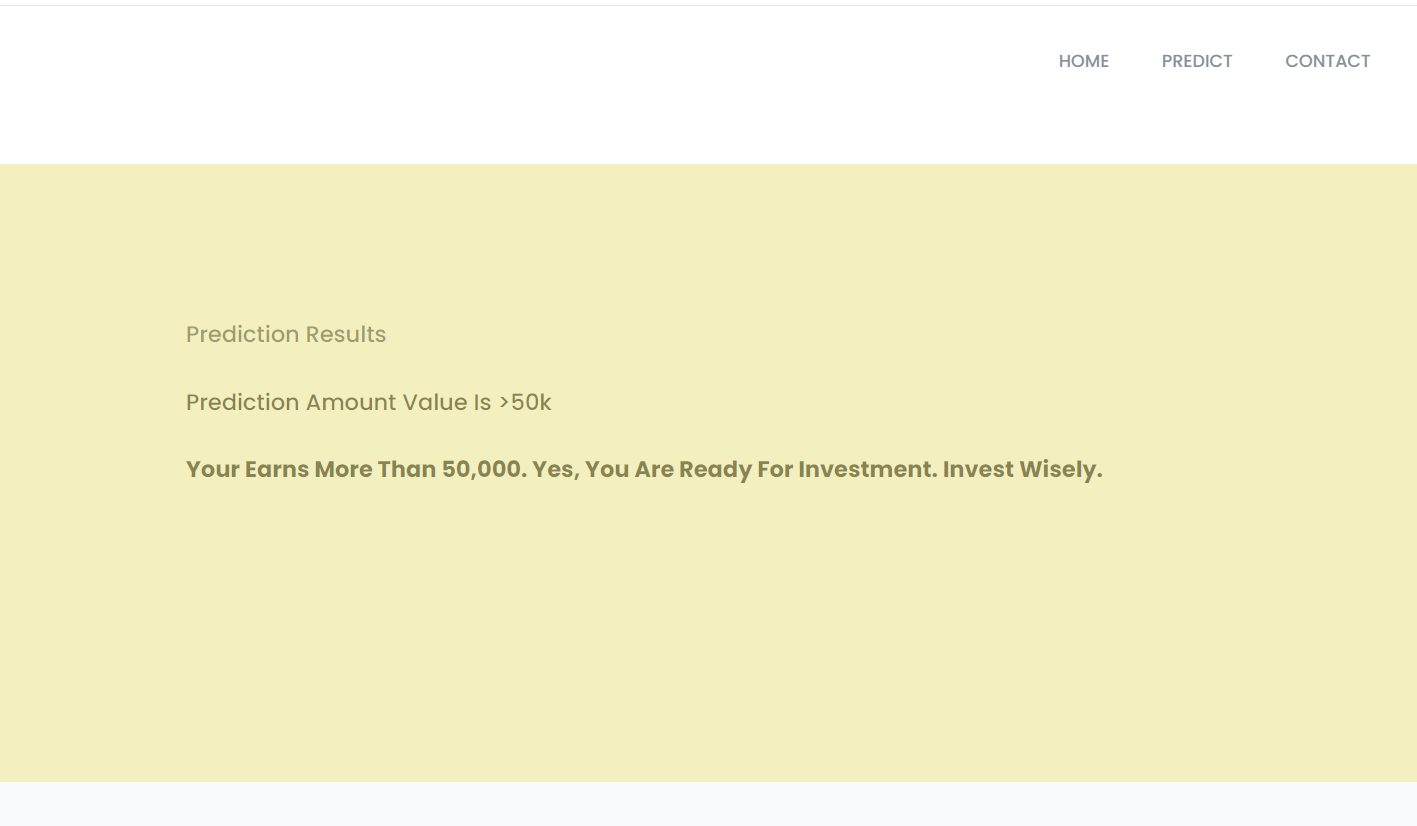
If you click on the predict on the nav bar we can able to navigate to predict.html below figure represents the UI of predict.html page

b) Prediction page (Predict.html)



By providing the inputs by the user and click on the predict button you will be navigating to the result.html it displays the result. Based on the result you can navigate to the home and click on the investments in the nav bar of index.html then you can have an info about different mode of investments.

Result.html:

****