Low Level Design

Adult Census Income Prediction

|  |  |
| --- | --- |
| Written By | Amanpreet Oberoi, Lovepreet Singh, Anjali Mishra |
| Document Version | 0.3 |
| Last Revised Date | 23 – Oct -2021 |

**Document Control**

### Change Record:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1 | 19 – Aug -  2021 | Amanpreet Oberoi,  Lovepreet Singh | Introduction & Architecture defined |
| 0.2 | 29 – Sep -  2021 | Lovepreet Singh | Architecture & Architecture Description appended and updated |
| 0.3 | 23 – Oct -  2021 | Amanpreet Oberoi, Anjali Mishra | Unit Test Cases defined and appended |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### Reviews:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reviewer** | **Comments** |
|  |  |  |  |

### Approval Status:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Review**  **Date** | **Reviewed By** | **Approved By** | **Comments** |
|  |  |  |  |  |

Contents

1. [Introduction 1](#_bookmark0)
   1. [What is Low-Level design document? 1](#_bookmark1)
   2. [Scope 1](#_bookmark2)
2. [Architecture 2](#_bookmark3)
3. [Architecture Description 2](#_bookmark4)
   1. [Data Collection 2](#_bookmark5)
   2. Import Data……………………………………………………………………………………………………………………2
   3. Data Validation………………………………………………………………………………………………………………3
   4. [Data Transformation 3](#_bookmark7)
   5. [Data Pre-processing 3](#_bookmark10)
   6. Best [Model Selection 3](#_bookmark12)
   7. [Hyper-Parameter Tuning 3](#_bookmark13)
   8. [Deployment 4](#_bookmark14)
   9. [User Data Input 4](#_bookmark15)
   10. User [Data Validation 4](#_bookmark16)
   11. [Make Predictions 4](#_bookmark17)
   12. [Display Predictions 4](#_bookmark18)
4. [Unit Test Cases 5](#_bookmark20)

# Introduction

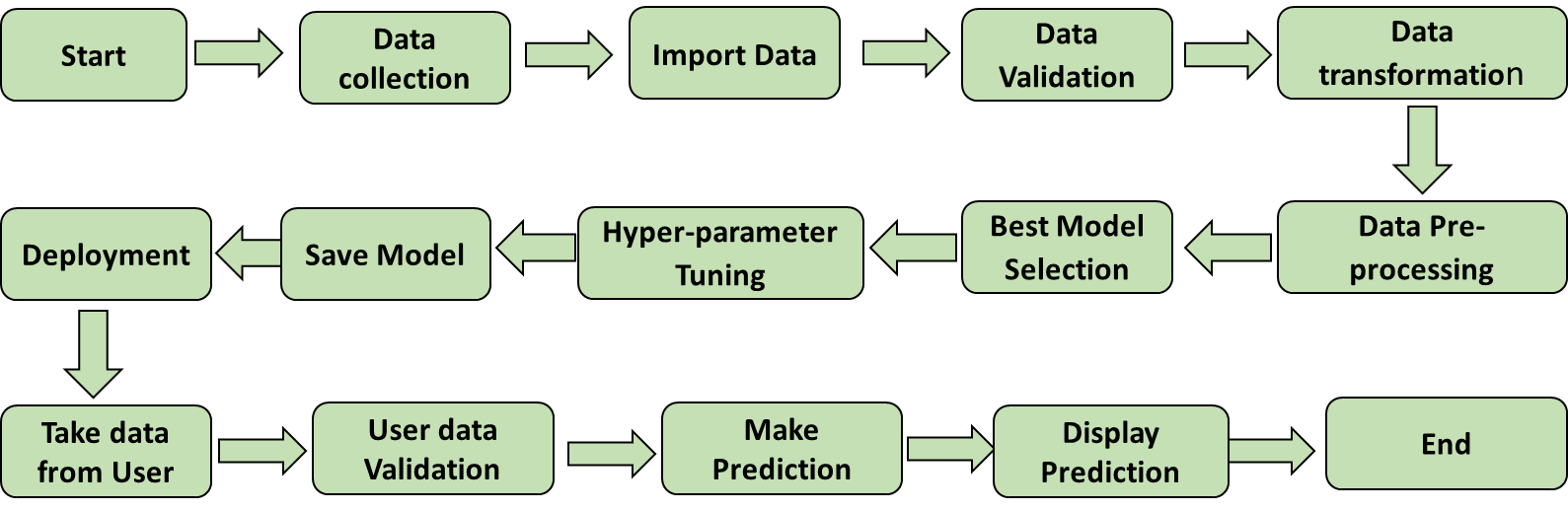
## What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Adult Census Income Prediction. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

## Scope

Low-level design (LLD) is a component-level design process that follows a step-by-Step [refinement](https://en.wikipedia.org/wiki/Refinement_(computing)) process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work

# Architecture



# Architecture Description

## Data Collection

As per data is provided on ineuron.ai according to the problem statement, so first we will download it on our local pc in the form of .csv file format from internship portal and save it in our workspace.

* 1. Import Data

For further process we will create a python module with the help of Pandas library to import the data on our workspace.

## Data Validation

* + 1. Number of Columns – Validation of number of columns present in the files.
    2. Name of Columns - The name of the columns is validated and should be the same as given in the schema file.
    3. Data type of columns - The data type of columns is given in the schema file.
    4. Null values in columns - If any of the columns in a file have all the values as NULL or missing.

## Data Transformation

* + 1. Extra Spaces – If some extra spaces are present in the some of the object (string) kind entities of the dataset which can hamper the work while doing pre-processing or time of training. So first we will remove these spaces.
    2. Replacing unknown entities – if any entities are present is in unknown data type, we will convert it into missing value (NAN).
    3. Separated Dependent and Independent Variables – Separated all independent variables from the dependent Variable.

## Data Pre-processing

* + 1. Missing Values Imputation – all the missing values are present in categorical column will be replaced with MODE values.
    2. Categorical Variables handling – all the categorical variables will be transformed by using Encoding.
    3. Outliers handling – outliers computation and handling in the Numerical variables, if needed.
    4. Imbalanced Dataset –if Dataset is highly imbalanced, than first we balanced it by using some kind of Over Sampler.
    5. Feature Scaling – all numeric Variables are must be on different scale, so we will scale down all the features on same scale with the help of some kind of Scaler.

## Best Model Selection

After the Pre-processing is got completed, we will go for the Model training approach to find the best model for our dataset use the various binary classification algorithms like “Logistic Regression” , “Random Forest Classifier”, “Gaussian-NB”, "XGBoost Classifier" and etc. For every model tuned algorithms are used. We will calculate the Accuracy-score for all the models and select the model with the best score.

## Hyper-Parameter Tuning

After selection of best model, we will be passed with the best parameters derived. We will calculate the Accuracy-score ant try to maintain it high. And then model will be saved for use in Recommendation.

## Deployment

After Training and model Selection we will go for the Deployment of the project for this we will create a web application by using “Flask Web Application framework” for the backend development.

HTML and CSS will be used for the Frontend development.

Flask is used for the backend, but it makes use of a templating language called Jinja2 which is used to create HTML markup formats that are returned to the user via an HTTP request. More on that in a bit.

After creation of web application “Heroku” will be used as a platform as a service (PaaS) to build, run, and operate applications entirely on the cloud.

## Data from User

## Here we will collect data from user such as user’s 'age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss', 'hours-per-week', 'workclass', 'education', 'marital-status', 'occupation', 'relationship', 'race', 'sex' and 'country’ with the help of HTML form input.

## Data Validation

Here Data Validation will be done, given by the user.

## Make Prediction

Here Predictions are done on Data given by the user.

## Display Prediction

Here Income Predictions will be displayed on web application at user interface.

# Unit Test Cases

|  |  |  |
| --- | --- | --- |
| **Test Case Description** | **Pre-Requisite** | **Expected Result** |
| Verify whether the Application URL is  accessible to the user | 1. Application URL  should be defined | Application URL should be  accessible to the user |
| Verify whether the Application loads completely for the user when the URL is accessed | 1. Application URL is accessible 2. Application is deployed | The Application should load completely for the user when the URL is accessed |
| Verify whether the User is able to sign  up in the application | 1. Application is  accessible | The User should be able to sign up  in the application |
| Verify whether user is able to edit all input fields | 1. Application is accessible | User should be able to edit all input fields |
| Verify whether user gets Submit button to submit the inputs | 1. Application is accessible | User should get Submit button to submit the inputs |
| Verify whether user is presented with Predictions results on clicking  submit | 1. Application is accessible | User should be presented with Predicted results on clicking  submit |
| Verify whether the predicted results are in accordance to the selections user made | 1. Application is accessible | The predicted results should be in accordance to the selections user made |