ARCHITECTURE DOCUMENT

Adult Census Income Prediction

Written By	Saakar Sengar And Shreyansh Jain
Document Version	0.3
Last Revised Date	17 - 01 -2022

Contents

Contents

1. Introduction

1.1. Why this Document?

The purpose of this document is to present a detailed description of the Adult Census Income Prediction System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the higher management for its approval.

The main objective of the project is classified the income category of either greater than 50K Dollars or less equal to 50K Dollar category of the person by using classification based Supervised Machine Learning algorithms.

1.2. Scope

This software system will be a Web application, this system will be designed to predict whether a person has an income of more than 50K Dollars a year or not. This is basically a binary classification problem where a person is classified into the >50K group or <=50K group.

2. Technical Specifications

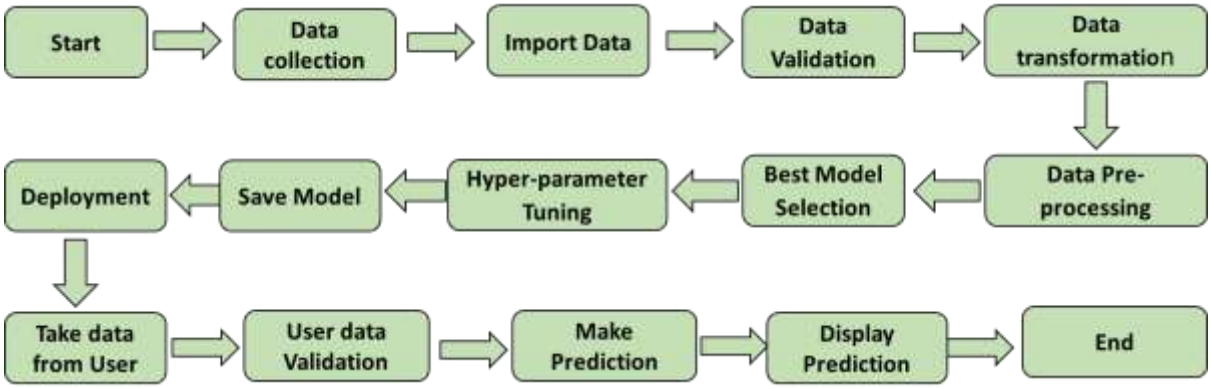
2.1. Logging

We should be able to log every activity done by the user:

- The System identifies at what step logging required
- The System should be able to log each and every system flow.
- Developers can choose logging methods.

- You can choose database logging/ File logging as well.
- A System should not be hung even after using so many loggings. Logging is just because we can easily debug issues so logging is mandatory to do.

3. Architecture



4. Architecture Description

4.1. 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.

4.2. Import Data

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

4.3. Data Validation

3.3.1. Number of Columns – Validation of number of columns present in the files.

3.3.2. Name of Columns - The name of the columns is validated and should be the same as given in the schema file.

3.3.3. Data type of columns - The data type of columns is given in the schema file.

3.3.4. Null values in columns - If any of the columns in a file have all the values as NULL or missing.

4.4. Data Transformation

3.4.1. 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.

3.4.2. Replacing unknown entities – if any entities are present is in unknown data type, we will convert it into missing value (NAN).

3.4.3. Separated Dependent and Independent Variables – Separated all independent variables from the dependent Variable.

4.5. Data Pre-processing

3.5.1. Missing Values Imputation – all the missing values are present in categorical column will be replaced with MODE values.

3.5.2. Categorical Variables handling – all the categorical variables will be transformed by using Encoding.

3.5.3. Outlier handling – outliers computation and handling in the Numerical variables, if needed.

3.5.4. Imbalanced Dataset –if Dataset is highly imbalanced then first we balanced it by using some kind of Over Sampler.

3.5.5. 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.

4.6. 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.

4.7. Hyper-Parameter Tuning

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

4.8. 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.

4.9. 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.

4.10. Data Validation

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

4.11. Make Prediction

Here Predictions are done on Data given by the user.

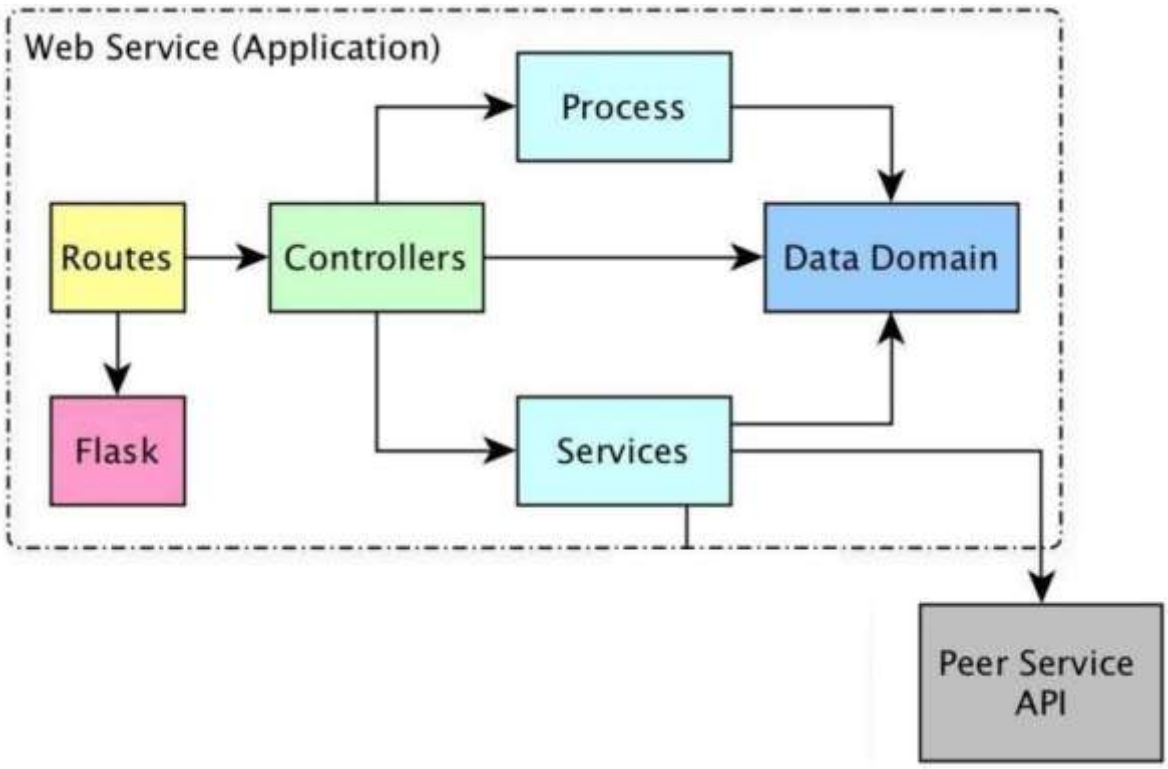
4.12. Display Prediction

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

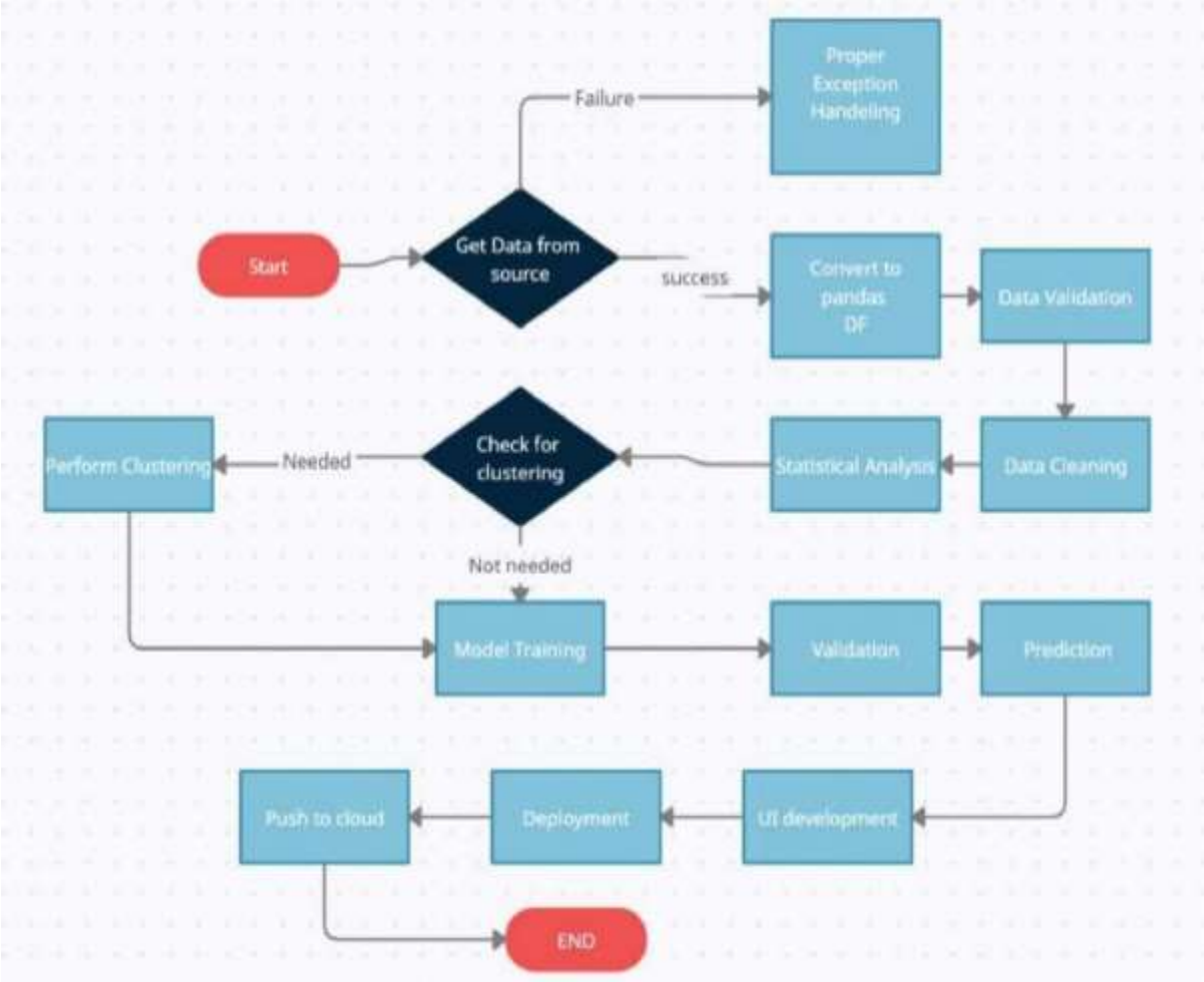
5. Workflows

5.1. Web Flow

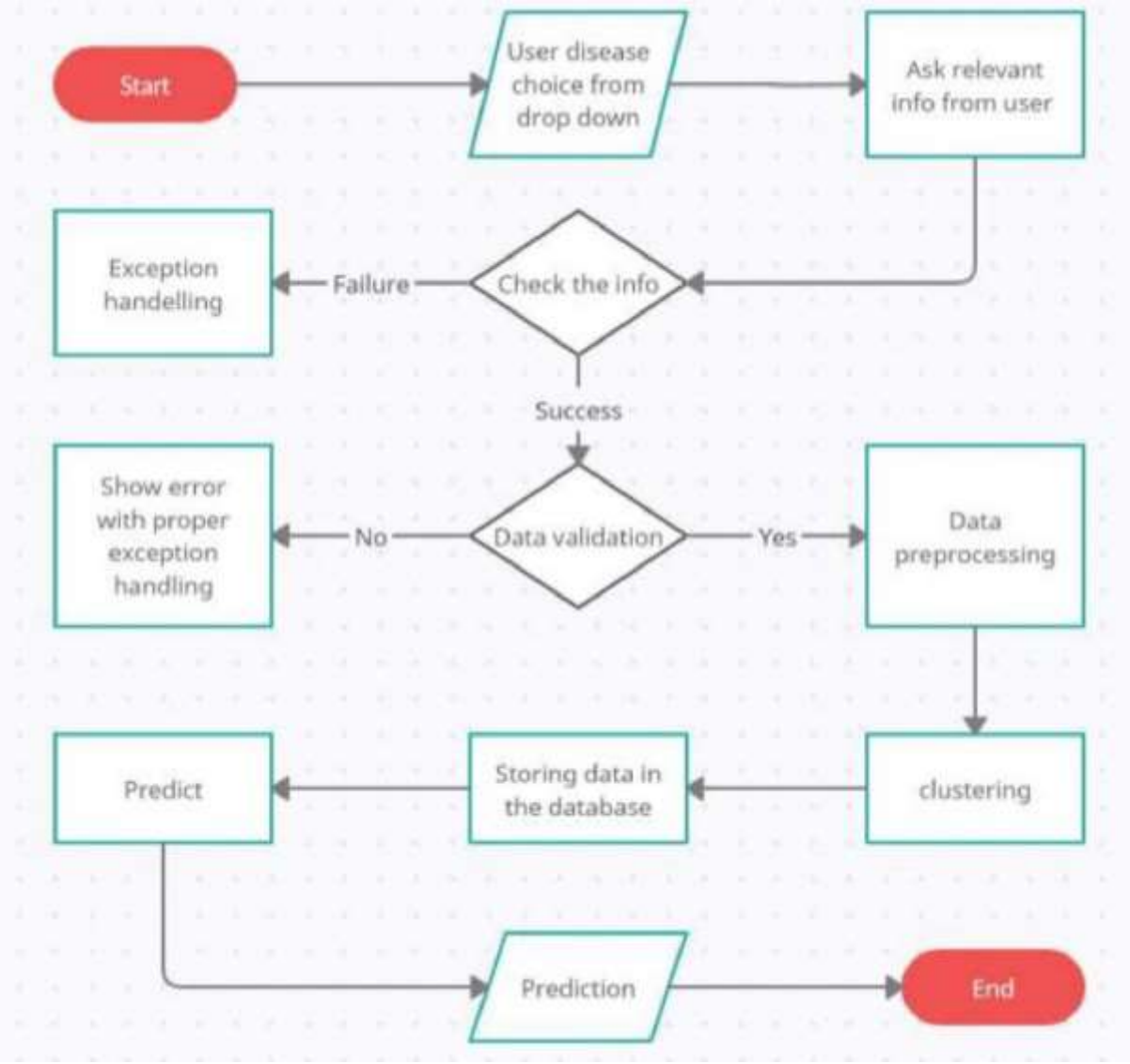
We have used Flask framework for this project so here is the Web Flow based Architecture of this flask project. We have integrated flight fare prediction model in this flask project .



5.2. Model Training Workflow



5.3. User Input Output Flow



5.4. User Interface

We have created an UI for user by using HTML and CSS.

The Adult Census Income Prediction model will classified the person into >50K group or <=50K group by using the input provided by the user and can get the idea person have income more than 50 K or not.

6. 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