iNeuron

Restaurant Rating Prediction

Written By	M V S K Karthik
Document Version	0.1
Last Revised Date	29 – Nov-2022

Document Control

Change Record:

Version	Date	Author	Comments
0.1	29 – Nov-2022	Karthik	Introduction & Architecture defined

Approval Status:

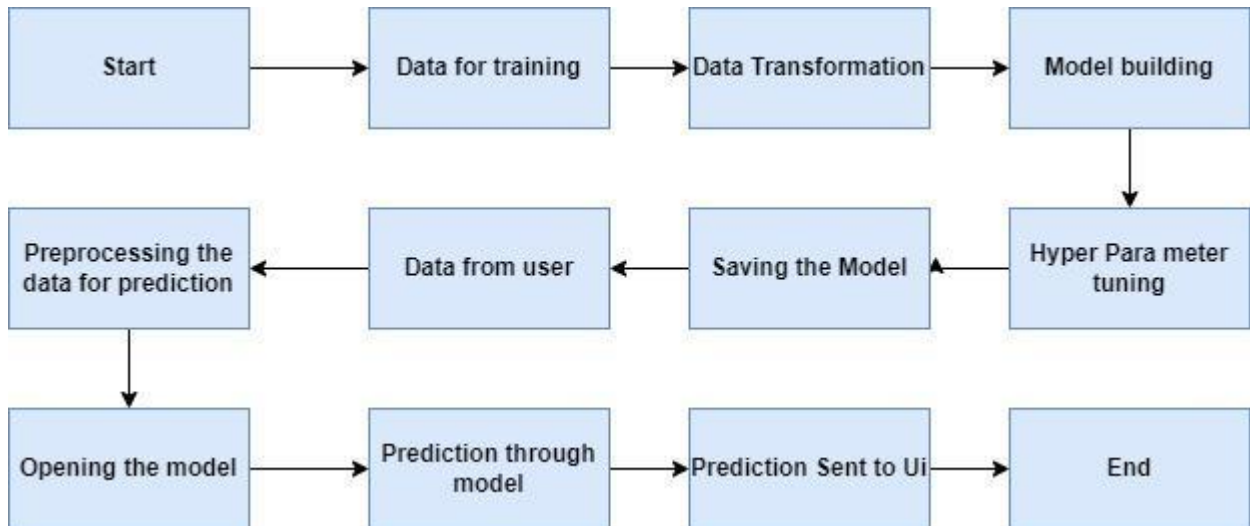| Version	Review Date	Reviewed By	Approved By	Comments

Contents

1. Introduction

1.1. What is a 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 Restaurant rating 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.

1.2. Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement 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

2. Architecture

```
┌──────────────┐      ┌──────────────┐      ┌──────────────────┐      ┌──────────────┐
│    Start     │─────▶│Data for training│──▶│Data Transformation│────▶│Model building│
└──────────────┘      └──────────────┘      └──────────────────┘      └──────────────┘
                                                                              │
                                                                              ▼
┌──────────────────┐  ┌──────────────┐      ┌──────────────┐      ┌──────────────────┐
│Preprocessing the │◀─│Data from user│◀─────│Saving the Model│◀───│ Hyper Para meter │
│data for prediction│ └──────────────┘      └──────────────┘      │     tuning       │
└──────────────────┘                                              └──────────────────┘
         │
         ▼
┌──────────────────┐  ┌──────────────────┐  ┌──────────────────┐  ┌──────────────┐
│Opening the model │─▶│Prediction through│─▶│Prediction Sent to Ui│─▶│     End      │
│                  │  │      model       │  └──────────────────┘  └──────────────┘
└──────────────────┘  └──────────────────┘
```

3. Architecture Description

3.1. Data for Training

The data set contains 51717 rows and 17 columns in a CSV file. Each column represents a different feature such as restaurant name, rest type, online order, etc.

3.2. Data Transformation

In the Transformation Process, we will read our data set through pandas to make it into a data frame and as per the findings of our EDA, we'll start data preprocessing which are as follows.
- first, we'll remove unnecessary columns such as url, name, address, phone, dish liked, review list, listed_in(city)
- after that, we'll remove duplicate columns.
- Next, we'll remove null values from the dataset.
- next, we'll rename several columns as per our needs.
- next, we'll change the datatype of the cost feature to float
- we'll format our rating column.
- next, we'll convert our categorical features to numerical features.
- after that, we'll separate feature and label columns for training

3.3. Model building

Transformed data is sent for the model building where we'll create different models and try to find out which model is giving the best accuracy.

3.4. Saving the Model

After finding out the best model we can save that model into a model directory for prediction.

3.5. Data from User

from here our prediction pipeline will start where we'll ask the user to provide certain data about the restaurant he wants to predict the rating.

3.6. Data preprocessing

We'll capture the data from Ui and we'll make some preprocessing parts such as converting categorical data into numerical

3.7. Model opening

we'll load our trained model for prediction.

3.8. Prediction through model

After loading the model we'll pass the processed data to the model. and it'll give us the final prediction.

3.9. Prediction on UI

the final prediction will be sent to UI.

Deployment

We will be deploying the model to Streamlit.