Low Level Design

Restaurant Rating Prediction

REVISION NUMBER – 1.2

Last date of Revision: 01/07/2022

Authored by: Utkarsh Yeole
Ritik Ratnawat
Vedant Deshmukh

Document Version Control

Date	Version	Description	Author
30/06/2022	1.0	Abstract Introduction Architecture	Utkarsh Yeole
31/06/2022	1.1	Architectural Design	Utkarsh Yeole
01/07/2022	1.2	Deployment Unit test Cases	Utkarsh Yeole

Contents

Abstract

The basic idea of analyzing the Zomato dataset is to get a fair idea about the factors affecting the establishment of different types of restaurants at different places in Bengaluru, aggregate rating of each restaurant, Bengaluru being one such city has more than 12,000 restaurants with restaurants serving dishes from all over the world. With each day new restaurants opening the industry hasn't been saturated yet and the demand is increasing day by day. Bengaluru being an IT capital of India, most of the people here are dependent mainly on the restaurant food as they don't have time to cook for themselves. With such an overwhelming demand for new restaurants, it has become important to study the ratings of restaurants.

1. Introduction

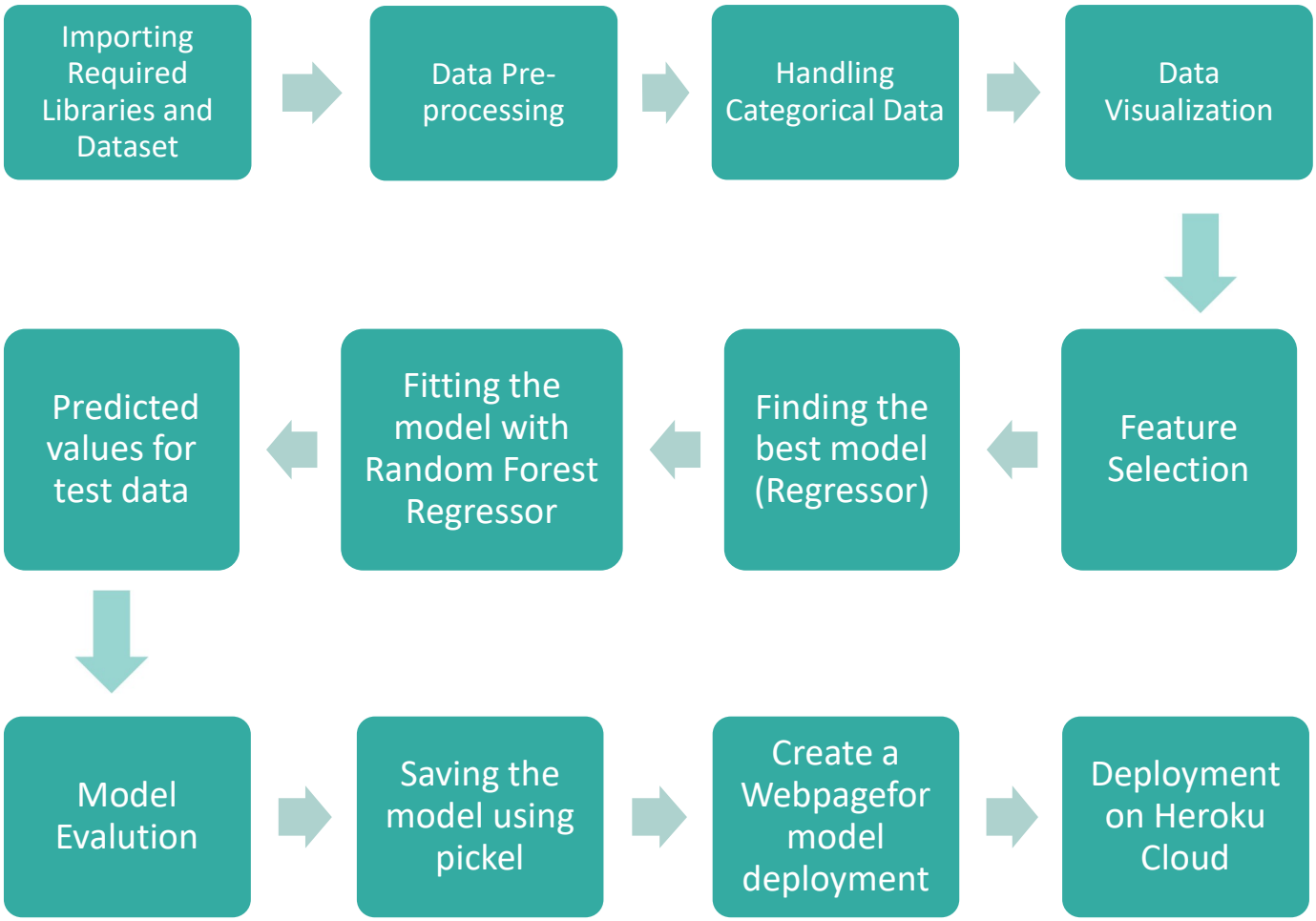
1.1 What is LLD Document ?

The main goal of the LLD document is to give the internal logic design of actual code implementation and supply the outline of the machine learning model and its implementation. Additionally, it provides the description how our project will designed end - to - end.

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



3. Architecture Design

This project is designed to make an interface for the User to predict the rating of restaurant.

3.1 Data Collection

The data for this project is collected from the Kaggle Dataset, the URL for the dataset is https://www.kaggle.com/datasets/himanshupoddar/zomato-bangalore-restaurants?resource=download

3.2 Data Description

The dataset contains 17 variables all of which were scrapped from the Zomato website. The dataset contains details of more than 50,000 restaurants in Bengaluru in each of its neighborhood. The total size of dataset is approximately 547 MB.

Variable	Type	#Unique Values	Description
url	object	51,717	contains the url of the restaurant in the zomato website
address	object	11,495	contains the address of the restaurant in Bengaluru
name	object	8,792	contains the name of the restaurant
online_order	category	2	whether online ordering is available in the restaurant or not
book_table	category	2	table book option available or not
rate	object	64	contains the overall rating of the restaurant out of 5
votes	int	2328	contains total number of rating for the restaurant as of the above mentioned date
phone	object	64	contains the phone number of the restaurant
location	category	93	contains the neighborhood in which the restaurant is located
rest_type	category	93	restaurant type
dish_liked	object	5271	dishes people liked in the restaurant
cuisines	object	2723	food styles, separated by comma
approx_cost(for two people)	float	70	contains the approximate cost for meal for two people
reviews_list	object	22513	list of tuples containing reviews for the restaurant, each tuple consists of two values, rating and review by the customer
menu_item	object	9098	contains list of menus available in the restaurant
listed_in(type)	category	7	type of meal
listed_in(city)	category	30	contains the neighborhood in which the restaurant is listed

3.3 Importing data into Database

Created associate API for the transfer of the info into the Cassandra info, steps performed are:

- Connection is created with the info.
- Created a info with name ZomatoInfo.
- cqlsh command is written for making the info table with needed parameters.
- And finally, a cqlsh command is written for uploading the Knowledge Set into data table by bulk insertion.

3.4 Exporting Data from Database

In the above created API, the download URL is also being created, which downloads the data into a csv file format.

3.5 Data Preprocessing

- Checked for info of the Dataset, to verify the correct datatype of the Columns.
- Checked for Null values, because the null values can affect the accuracy of the model.
- Converted all the illegal values into legal values.
- Performed Labeled encoding and One hot Encoding on the desired columns.
- Checking the distribution of the columns to interpret its importance.
- Now, the info is prepared to train a Machine Learning Model.

3.6 Model Creation

The Pre - processed info is now envisioned and drawn insights helps us to select the feature that improves the accuracy of the model. The info is randomly used for modelling with different machine learning algorithms to create a model to predict the Flight ticket price. After performing on different algorithms, we use Random Forest Regression to create a model and then also perform Hyperparameter Tuning to improve the accuracy of the model.

3.7 Data from User

The data from the user is retrieved from the created HTML web page.

3.8 Data Validation

The data provided by the user is then being processed by app.py file and validated. The validated data is then sent to the prepared model for the prediction.

4. Deployment

The tested model is then deployed to Heroku. So, users can access the project from any internet devices.

4.1 Unit test case

Test Case Description	Pre-Requisites	Expected Results
Verify whether the Webpage is accessible to the User or not.	Webpage URL should be defined.	Webpage should be accessible to the User.
Verify whether the Webpage is completely loads for the User or not	1. Webpage URL is accessible. 2. Webpage is deployed.	The Webpage should be completely loads for the User when it is accessed.
Verify whether the User is able to enter data in input fields or not.	1. Webpage URL is accessible. 2. Webpage is deployed. 3. Webpage input fields are editable.	The User is able to enter data in input fields.
Verify whether the User is able to submit details or not.	1. Webpage URL is accessible. 2. Webpage is deployed. 3. Webpage input fields are editable.	The User is able to submit details to process.
Verify whether the User gets recommended results on submitting the details or not.	1. Webpage URL is accessible. 2. Webpage is deployed. 3. Webpage input fields are editable	The User gets recommended results on submitting the details.