Low Level Design (LLD)

Restaurant Rating Prediction for Zomato.

Basawanayya Hiremath

Contents

1. Introduction

Restaurant industries phase is shifted from dine in or take out to order online. This shift is due to entrepreneurship and technology. Any Hotel or Restaurant can register on platform like Zomato, Swiggy etc., eventually they can grow business by providing good service. This platform provide direct feedback to restaurant from customers. The objective of this project is to analysis and predict the Rating of the restaurant from given dataset. The overall analysis and Dominos vs Pizza Hut are compared. To predict the rating linear regression, Random forest and XGBoot are tested.pr

1.1 Low Level Design

(LLD) is like detailing the HLD. It defines the actual logic for each and every component of the system. Class diagrams with all the methods and relation between classes comes under LLD. Programs specs are covered under LLD.

1.2 Scope

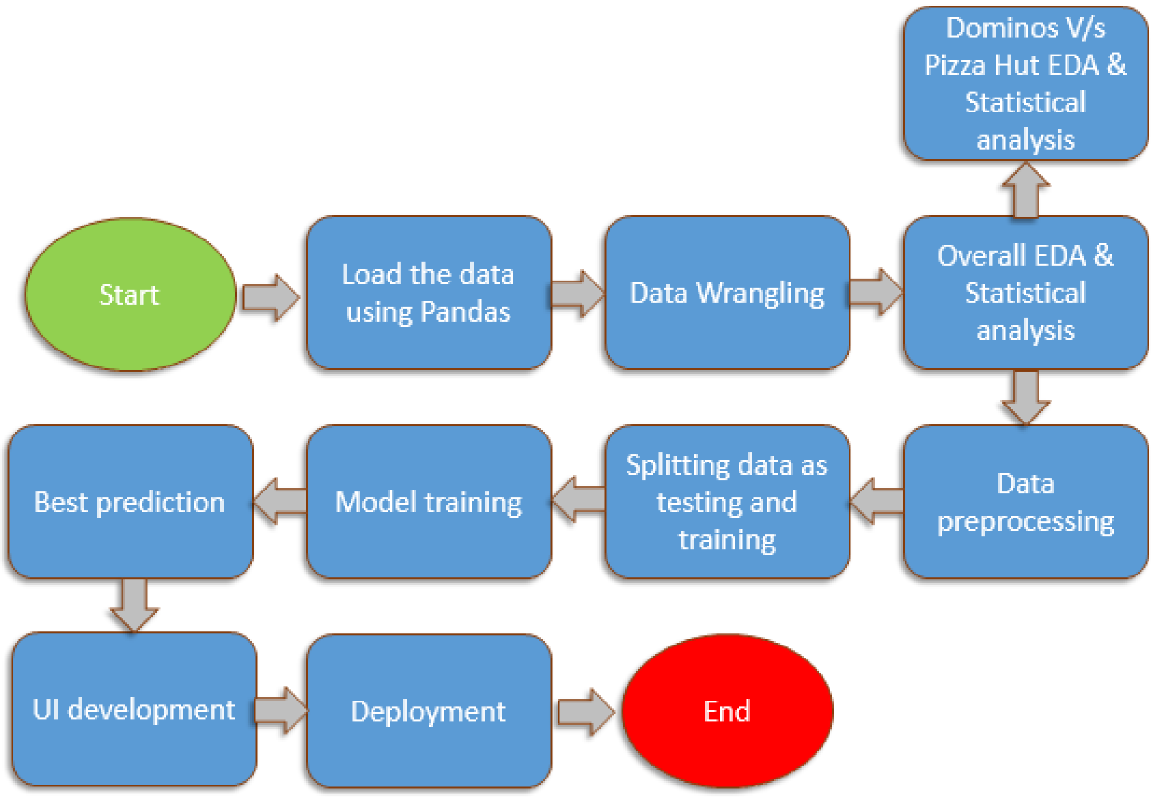
The goal of LLD or a low-level design document (LLDD) is to predict restaurant rating based on certain given features. The code is written in Python language and deployed in local host. The interface will have some features that will predict the rating.

The main goal of this project is to perform extensive Exploratory Data Analysis (EDA) on the Zomato Dataset and build an appropriate Machine Learning Model that will help various Zomato Restaurants to predict their respective Ratings based on certain features.

- Overall EDA of the data
- Dominos Vs Pizza Hut
- Model building
- Deployment

The scope of the project is to visualize and compare the data with other restaurant to adapt and to build UI to show the Rating as a output based on certain features like location, votes, cost for two people, restaurant type etc.

2. Architecture

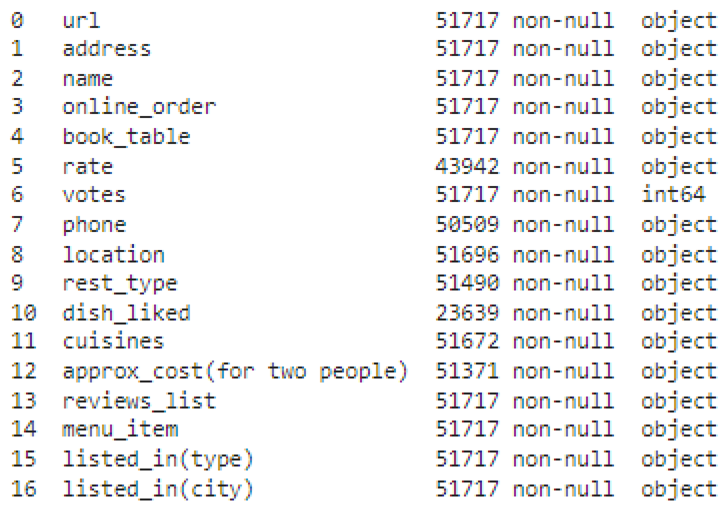

Dominos V/s Pizza Hut EDA & Statistical analysis
Start
Load the data using Pandas
Data Wrangling
Overall EDA & Statistical analysis
Best prediction
Model training
Splitting data as testing and training
Data preprocessing
UI development
Deployment
End

3. Data overview and Architecture Description

Data shape 51717*17

Data Description

The zomato dataset Bangalore contains 17 columns and 51717 rows. The dataset is available in kaggle link is given below. The dataset is in csv format.



```
0   url                          51717 non-null  object
1   address                      51717 non-null  object
2   name                         51717 non-null  object
3   online_order                 51717 non-null  object
4   book_table                   51717 non-null  object
5   rate                         43942 non-null  object
6   votes                        51717 non-null  int64
7   phone                        50509 non-null  object
8   location                     51696 non-null  object
9   rest_type                    51490 non-null  object
10  dish_liked                   23639 non-null  object
11  cuisines                     51672 non-null  object
12  approx_cost(for two people)  51371 non-null  object
13  reviews_list                 51717 non-null  object
14  menu_item                    51717 non-null  object
15  listed_in(type)              51717 non-null  object
16  listed_in(city)              51717 non-null  object
```

Data link, https://www.kaggle.com/himanshupoddar/zomato-bangalore-restaurants

3.1 Features: - url, address, name, online_order, book_table, rate, votes, phone, location, rest_type, dish_liked cuisines, approx_cost(for two people), reviews_list, menu_item, listed_in(type) and listed_in(city).

In the data we can see that only votes feature is int, all other are objects.

3.2 Data wrangling:- Some features required cleaning for example ‘rate’ feature is in format ‘4/5’ which is object, and contains entries like ‘NEW’, ‘-’ etc. this feature has to clean in order to analysis.

3.3 EDA: - overall EDA (uni-variate, bi-variate analysis) is performed on most of the features.

3.4 EDA of Dominos V/s Pizza Hut: - comparison is done on both Dominos and Pizza Hut by plotting several graphs and analysis.

3.5 Data Preprocessing: - In this step some features are dropped like phone number, location, url etc. After that featuring engineering is performed on object features. Here label encoding is used to transform the data.

3.6 Splitting data as training and testing: - Data is splitting as 80% and 20% as training and testing respectively.

3.7 Model training: - Various models like linear regression, random forest and XGboosting machine learning algorithms are used to produce best R2 score. Hyperparmeter is performed to increase the score. Best score model is chosen and converted and saved in pickle file.

3.8 UI development: - Flask and Html is used to design to be displayed in a web browser.

3.9 Deployment: - Model is deployed on local host.

4. 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 is 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 is loading 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 successfully login to the application	1. Application is accessible 2. User is signed up to the application	User should be able to successfully login to the application
Verify whether user is able to see input fields on logging in	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be able to see input fields on logging in

Verify whether user is able to edit all input fields	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be presented with recommended results on clicking submit
Verify whether the recommended results are in accordance to the selections user made	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	The recommended results should be in accordance to the selections user made
Verify whether user has options to filter the recommended results as well	1. Application is accessible 2. User is signed up	User should have options to filter the recommended results as well