**Low Level Design (LLD)**

Rental Bike Share Prediction

**Document Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date Issued** | **Version** | **Description** | **Author** |
| 04/08/2022 | 1 | Initial LLD – V1.0 | A. Jana |
|  |  |  |  |
|  |  |  |  |

Contents

[1 Introduction 4](#_Toc110541172)

[1.1 What is Low-Level design document? 4](#_Toc110541173)

[1.2 Scope 4](#_Toc110541174)

[2 Architecture 4](#_Toc110541175)

[3 Architecture Description 4](#_Toc110541176)

[3.1 Data Description 4](#_Toc110541177)

[3.2 Data Transformation 5](#_Toc110541178)

[3.3 Data Insertion into Database 5](#_Toc110541179)

[3.4 Export Data from Database 5](#_Toc110541180)

[3.5 Data Pre-processing 5](#_Toc110541181)

[3.6 Data Clustering 5](#_Toc110541182)

[3.7 Model Building 6](#_Toc110541183)

[3.8 Data from User 6](#_Toc110541184)

[3.9 Data validation 6](#_Toc110541185)

[3.10 User Data Inserting into Database 6](#_Toc110541186)

[3.11 Data Clustering 6](#_Toc110541187)

[3.12 Model Call for Specific Cluster 6](#_Toc110541188)

[3.13 Saving Output in Database 6](#_Toc110541189)

[3.14 Deployment 6](#_Toc110541190)

[4 Unit Test Cases 6](#_Toc110541191)

# 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 Food Recommendation System. 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

A picture containing diagram

Description automatically generated

# Architecture Description

## Data Description

There are two data files present in the dataset.

* hour.csv: bike sharing counts aggregated on hourly basis. Records: 17379 hours
* day.csv - bike sharing counts aggregated on daily basis. Records: 731 days

Both hour.csv and day.csv have the following fields, except hr which is not available in day.csv

* + instant: record index
  + dteday: date
  + season: season (1: springer, 2: summer, 3: fall, 4: winter)
  + yr: year (0: 2011, 1: 2012)
  + mnth: month (1 to 12)
  + hr: hour (0 to 23)
  + holiday: weather day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)
  + weekday: day of the week
  + workingday: if day is neither weekend nor holiday is 1, otherwise is 0.
  + weathersit:

1: Clear, Few clouds, Partly cloudy, Partly cloudy

2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist

3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds

4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog

* + temp: Normalized temperature in Celsius. The values are divided to 41 (max)
  + atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)
  + hum: Normalized humidity. The values are divided to 100 (max)
  + windspeed: Normalized wind speed. The values are divided to 67 (max)
  + casual: count of casual users
  + registered: count of registered users
  + cnt: count of total rental bikes including both casual and registered

## Data Transformation

We’ll check if the input files are in csv format or not, we’ll transformation the data into csv format if necessary.

## Data Insertion into Database

1. Database Creation and connection - Create a database with name passed. If the database is already created, open the connection to the database.
2. Table creation in the database.
3. Insertion of files in the table

## Export Data from Database

The data in a stored database is exported as a CSV file to be used for Data Pre-processing and Model Training.

## Data Pre-processing

Data Pre-processing steps we could use are Null value handling, unnecessary column removal, Imbalanced data set handling, Handling columns with standard deviation zero or below a threshold, etc.

## Data Clustering

K-Means algorithm will be used to create clusters in the pre-processed data. The optimum number of clusters is selected by plotting the elbow plot. The idea behind clustering is to implement different algorithms to train data in different clusters. The K-means model is trained over preprocessed data and the model is saved for further use in prediction.

## Model Building

If clusters are created, we will find the best model for each cluster. For each cluster, algorithms will be passed with the best parameters derived from Grid-Search. We will calculate the RMSE and R2 scores for models and select the model with the best score. Similarly, the models will be selected for each cluster. All the models for every cluster will be saved for use in prediction.

If clusters are not used, we’ll simply find the best parameters derived from Grid-Search. We will calculate the R2 scores for models and select the model with the best score.

## Data from User

Here we will collect seasonal data such as season, weather situation, as well as daily/hourly data such as month, day of the week, temperature, humidity, windspeed from the user.

## Data validation

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

## User Data Inserting into Database

Collecting the data from the user and storing it into the database. The database can be either MySQL or Cassandra DB.

## Data Clustering

The model created during training will be loaded, and if clustering is used, clusters for the user data will be predicted.

## Model Call for Specific Cluster

If clustering is used, based on the cluster number, the respective model will be loaded and will be used to predict the data for that cluster.

## Saving Output in Database

After calling model Output will be recommended, this output will be saved in Database, and it will be used to show the same Output if other users provide the same data.

## Deployment

Deployment will be done in one of these platforms

A picture containing logo

Description automatically generated

# 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.  2. | Application URL is accessible  Application URL is deployed | Application URL should load completely for the user when URL is accessed |
| Verify whether user can see input field after opening URL | 1. | Application is accessible | User should be able to see input fields after opening URL |
| Verify whether user can edit all the input fields | 1. | Application is accessible | User should be able to edit all the input fields |
| Verify whether user has options to filter the inputs fields | 1. | Application is accessible | User should filter the options of 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 can see the output after submitting the inputs | 1. | Application is accessible | User should get outputs after submitting the inputs |
| Verify whether KPIs modify as per the user inputs | 1. | Application is accessible | KPIs should modify as per the user inputs for the user's health |
| Verify whether the KPIs indicate details of the result | 1. | Application is accessible | The KPIs should indicate details of the result |