CONCRETE COMPRESSIVE STRENGTH PREDICTION

LOW LEVEL DESIGN

REVISION No. : 1.0
LAST DATE OF REVISION : MAY 2ND,2022

DOCUMENT VERSION CONTROL

DATE	VERSION	DESCRIPTION	AUTHOR
02-05-2022	1	INITIAL LLD	MOHD. USAMA

TABLE OF CONTENTS

1 Introduction

1.1 Why this 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 Concrete Compressive Strength 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. The 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.

.

1.3 Project Introduction

Structures have to face various types of static and dynamic loads and to bear such loads we need a concrete of sufficient strength to bear these kind of loads .A concrete of enough strength to bear these loads is very necessary to avoid any accidents and loss. The concrete strength depends upon the choice and quantity of it's components. It achieves around 99% of it's strength in 28 days. So to predict the strength of our concrete sample we have to wait for that much time which is not feasible.

Hence a good data driven system for predicting concrete strength can help to design a concrete of required strength very quickly by removing the time barrier and finding out the effects of it's components based on their quantity in it's mixture.

2 Problem Statement

Strength determines the lifespan of as structure. Hence, we have to analyze the dataset to predict the strength of concrete. The dataset we have contains the components of concrete which effects the strength of concrete the age of concrete by which concrete have achieved it’s strength

3 Dataset Information

Cement : Quantity of in Cement kg per cubic meter of concrete.

Blast Furnace Slag : Quantity of Slag in kg per cubic meter of concrete.

Fly Ash : Quantity of Fly Ash in kg per cubic meter of concrete.

Water : Quantity of Water in kg per cubic meter of concrete.

Superplasticizer : Quantity of Superplasticizer in kg per cubic meter of concrete.

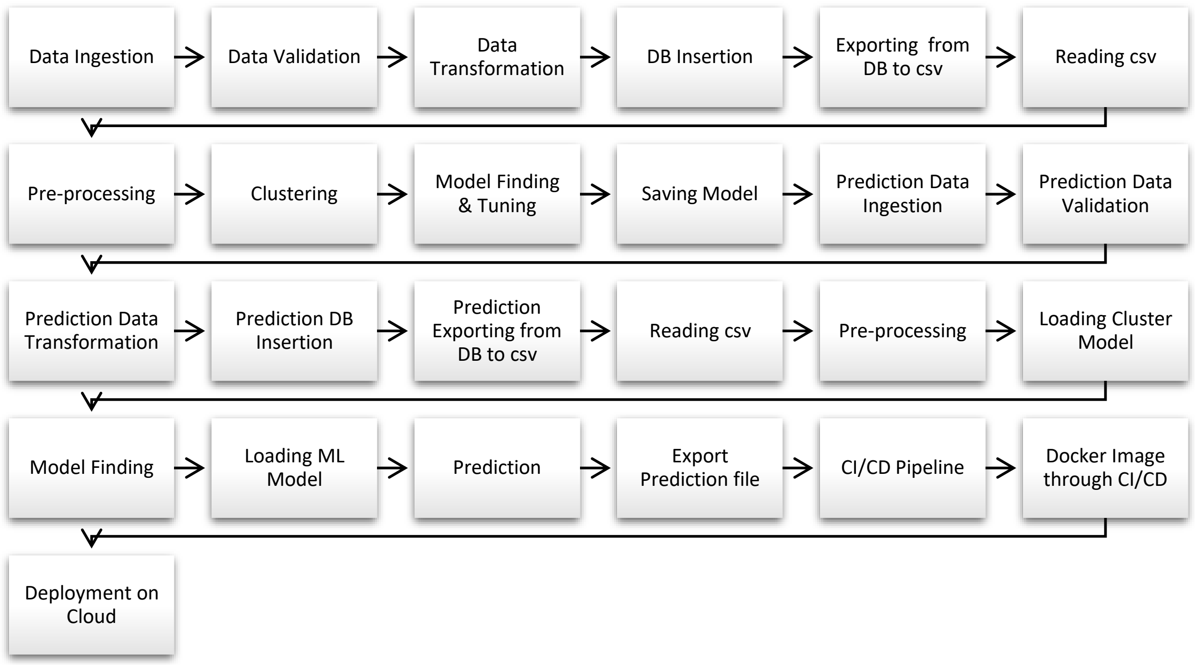
Coarse Aggregate : Quantity of C. Aggregate in kg per cubic meter of concrete.

Fine Aggregate : Quantity of F. Aggregate in kg per cubic meter of concrete.

Age : Duration at which concrete strength is tested.

Concrete Compressive Strength : Compressive strength in Megapascal which is achieved based on the above mentioned components and the duration at which it is tested.

4 Architecture



3.1 Architecture Description

- **Data Ingestion :** Dataset is taken from Kaggle link provided by I-Neuron
- **Data Validation :** Validation checks have been made like Name and Number of Columns, File name, Missing values in columns, etc.
- **Data Transformation :** Columns have been renamed to shorter name for easy access. No other necessary transformation required in the dataset.
- **DB Insertion :** After validation of the data and doing necessary transformations, the data is pushed to Cassandra Database. This is the last step in Training Raw Data Validation process.
- **Exporting from DB to csv :** Validated data that was pushed to Cassandran DB is retrieved into csv file. This csv file is used to train our model after necessary steps.
- **Reading csv :** The csv file that we received is now accessed using pandas
- **Data Pre-processing :** Dataset is checked for missing values, duplicate values, skewed data, outliers, scales, distributions, etc . We have found only duplicate rows are present which will not contribute anything for prediction, so have removed those rows. There are no other pre-processing steps required in this dataset.

- **Clustering :** Clustering of model is done to make the clusters of data having similarity between them.
- **Model Finding :** Model is trained based on each of the cluster dataset using appropriate ML algorithms and out of these models the best one is selected based on the R2 score for each of the clusters.
- **Model Tuning :** After finding out the best model for each of the clusters, we do hyperparameters tuning of those models to get better score. And then based on the best parameters, the model is trained.
- **Saving Model :** After training we save our model which can be used for predicion of unknown data.
- **Prediction Steps :** When we receive the new data from client for which we have to do predictions, we do all the Data Validation, Data Ingestion, Data Transformation, DB Operations, and Pre-processing steps, and then we will load our cluster model that he have created during Clustering step in Training and then using that model, it will make the clusters of prediction dataset and then based on that cluster number we use the cluster specific trained model to do the predictions of cluster and then we append those results into a single file which we have to save for Client.
- **CI/CD Pipeline :** Circle CI is used for the purpose of CI/CD operations which is used to create docker image for DockerHub and to deploy on cloud automatically just by committing our code to GitHub Repository.
- **Deployment :** The project is deployed to AWS and Heroku cloud platforms.