Low Level Design (LLD)

Concrete Compressive strength Prediction

|  |  |
| --- | --- |
| Written by | Ninad Uday Karlekar |
| Document version | 1 |
| Last revised date | 06-Jan-2024 |

**Document Version Control**

**Change record:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1 | 06/01/2024 | Ninad Uday Karlekar | Introduction and architecture defined |
|  |  |  |  |

**Reviews:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reviewer** | **Comments** |
| 0.2 | 06/01/2024 | Ninad Uday Karlekar | Architecture implemented completely, Version Control and Unit Test Cases to be added |

**Approval status:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Review Date** | **Reviewed By** | **Approved By** | **Comments** |
|  |  |  |  |  |

Table of Contents

[**1.** **Introduction** 4](#_Toc155456149)

[**1.1.** **What is Low-Level design document?** 4](#_Toc155456150)

[**1.2.** **Scope** 4](#_Toc155456151)

[**2.** **Architecture** 4](#_Toc155456152)

[**3.** **Architecture Description** 4](#_Toc155456153)

[**3.1.** **Data ingestion** 4](#_Toc155456157)

[**3.2.** **Data pre-processing** 4](#_Toc155456158)

[**3.3.** **Model building** 4](#_Toc155456159)

[**3.4.** **Model evaluation** 4](#_Toc155456160)

[**3.5.** **Saving the best models** 4](#_Toc155456161)

[**3.6.** **Model deployment** 4](#_Toc155456162)

[**4.** **Unit Test Cases** 4](#_Toc155456163)

# **Introduction**

## **What is Low-Level design document?**

The purpose of a Low-Level Design Document (LLD) is to provide an internal logical blueprint of the Concrete Compressive Strength Prediction System's actual program code. LLD outlines class diagrams, methods, and relationships between classes, along with program specifications. It serves as a guide for programmers to directly implement the program based on the details provided in the document.

## **Scope**

Low-Level Design (LLD) is a detailed design process focusing on individual components. It involves a step-by-step refinement process, addressing data structures, software architecture, source code, and performance algorithms. This approach is instrumental in defining data organization, a crucial aspect initiated during requirement analysis and further refined in the course of data design work.

# **Architecture**

A diagram of a model

Description automatically generated

# **Architecture Description**



## **Data Description**

The Concrete Compressive Strength dataset, sourced from Kaggle here, consists of 1030 instances with 9 features, providing insights into factors influencing concrete compressive strength.

URL:- <https://www.kaggle.com/datasets/elikplim/concrete-compressive-strength-data-set>

**Variable Information:**

|  |  |
| --- | --- |
| Name | Dtype |
| cement | float64 |
| blast\_furnace\_slag | float64 |
| fly\_ash | float64 |
| water | float64 |
| superplasticizer | float64 |
| coarse\_aggregate | float64 |
| fine\_aggregate | float64 |
| age | int64 |
| concrete\_compressive\_strength | float64 |

## **Data ingestion**

* The data for this project was loaded from a CSV file named "concrete\_data.csv" located in the "../dataset" directory using pandas.
* The data contains features related to concrete composition and the corresponding compressive strength.

## **Data pre-processing**

* The loaded data underwent preprocessing steps to ensure its suitability for machine learning modeling.
* This included handling missing values (if any) using appropriate techniques like imputation or deletion.
* Feature scaling was applied to numerical features to bring them to a common range, improving the performance of some machine learning algorithms.

## **Model building**

## **Model evaluation**

## **Saving the best models**

## **Model deployment**

# **Unit Test Cases**