Low Level Design (LLD) SHIPMENT PRICING PREDICTION



**Project By:**

Deepraj Vadhwane

Shanu Singh

Roushan Singh

Kaushlendra kumar

Kiran Raghuwanshi

Farheen Syed

Hari Prasad Kale

## Document Version control

| **Date issued** | **Version** | **Description** | **Author** |
| --- | --- | --- | --- |
| 17-02-2023 | 1 | Initial LLD | **Deepraj Vadhwane** |
| 20-02-2023 | 2 | Initial LLD | Roushan Singh |
|  |  |  |  |

Contents

[Document Version Control 2](#_heading=h.gjdgxs)

[Abstract 4](#_heading=h.30j0zll)

1. [Introduction 5](#_heading=h.1fob9te)

[1.2 Scope 5](#_heading=h.3znysh7)

1. [Architecture 6](#_heading=h.2et92p0)
2. [Architecture Description 7](#_heading=h.tyjcwt)
   1. [Loading and Reading Dataset 7](#_heading=h.3dy6vkm)
   2. [Data Cleaning 8](#_heading=h.1t3h5sf)
   3. [Exploratory Data Analysis (EDA) 9](#_heading=h.4d34og8)
   4. [Data Preprocessing 10](#_heading=h.2s8eyo1)
   5. [Feature Engineering 11](#_heading=h.17dp8vu)
   6. [Model Selection 11](#_heading=h.3rdcrjn)
   7. [Model Evaluation 11](#_heading=h.26in1rg)
   8. Model Deployment 11
   9. Deployment 12

[4. Unit Test Cases 13](#_heading=h.lnxbz9)

**ABSTRACT**

The goal of this project is to develop a model that can accurately predict consignment price based on a variety of factors. The market for supply chain analytics is expected to experience significant growth over the next few years, as organizations increasingly recognize the benefits of being able to forecast future events with a high degree of certainty. By accurately predicting supply chain pricing, supply chain leaders can address challenges, reduce costs, and improve service levels.

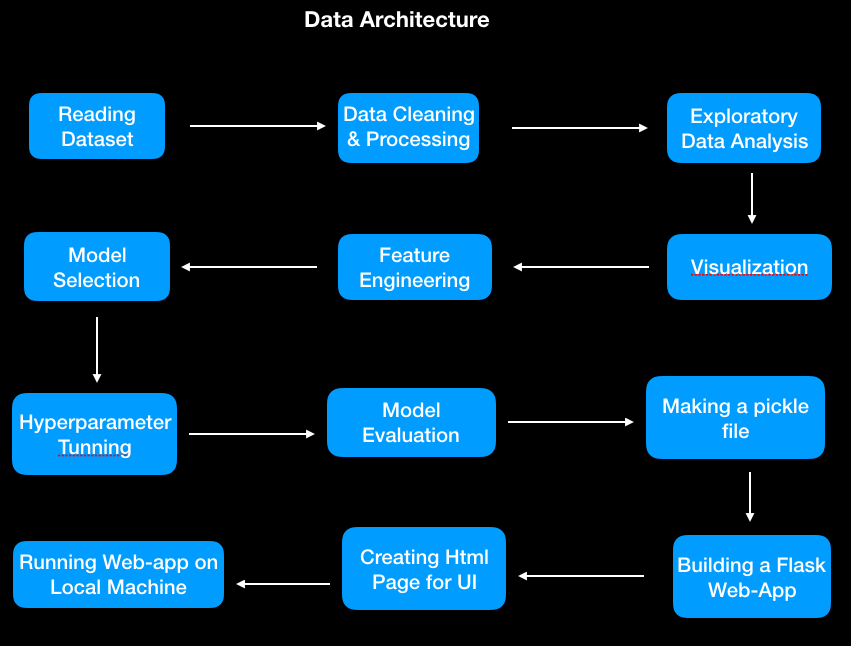
# Introduction

* 1. **Why this Low-Level Design Document?**

The goal of the Low-level design document (LLDD) is to give the internal logic design of the actual program code for the Shipment Pricing Prediction. LLDD 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. 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:-**

# Architecture Description

* 1. **Loading and Reading Dataset**

The primary source of data for this project from Kaggle.The dataset is comprised of 10,324 records with 33 attributes. The data is in a structured format. The data is stored locally and is exported as a CSV file to be used for Data Pre-processing and Model Training.



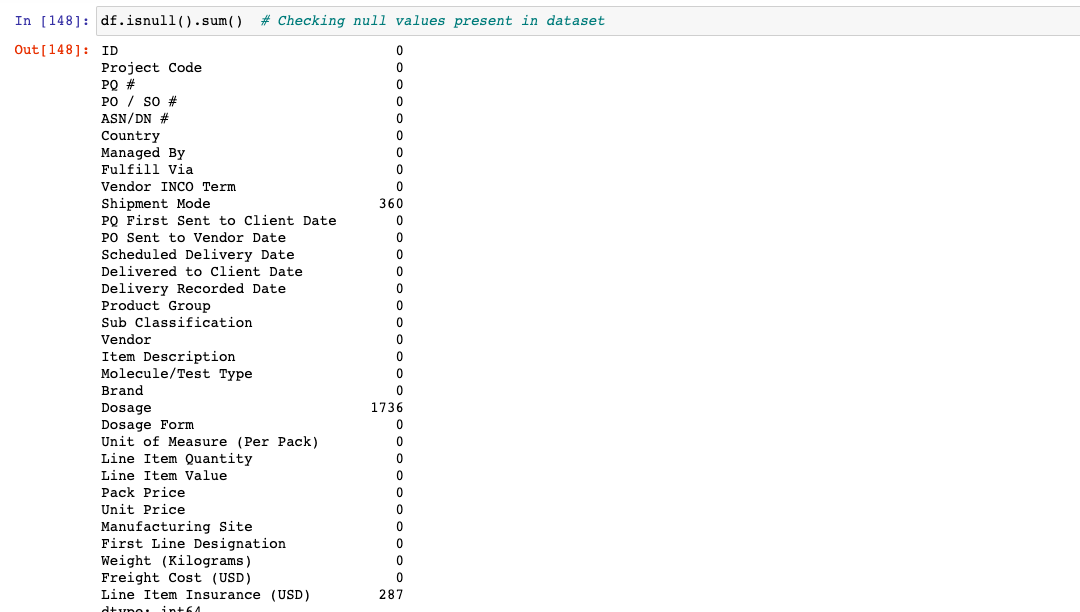


* 1. **Data Cleaning**

Data cleaning is the process of detecting and correcting errors, inconsistencies, or missing values in a dataset. It is an essential step in data pre-processing and is often the most time-consuming part of the data analysis process.

There are many different techniques for cleaning data, depending on the nature of the dataset and the types of errors it contains. Some common techniques include:

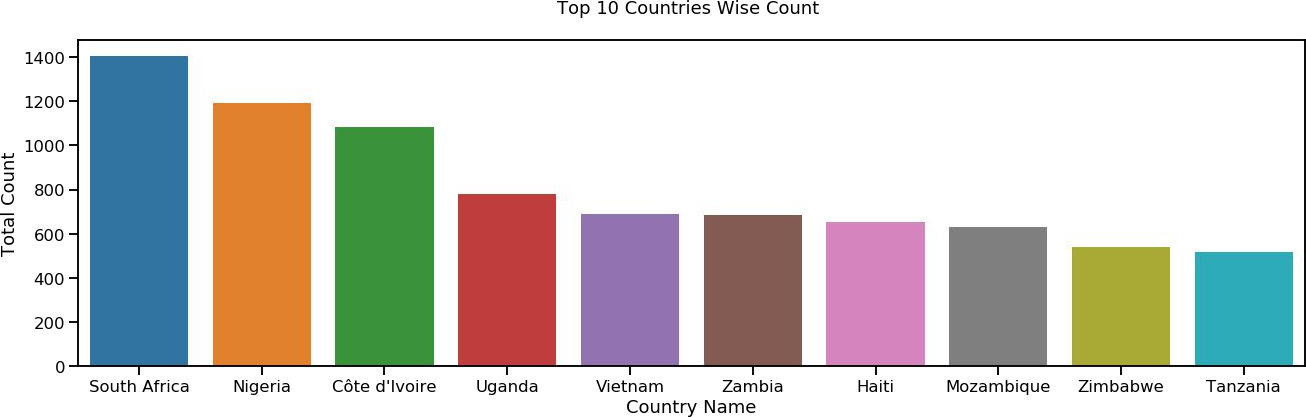
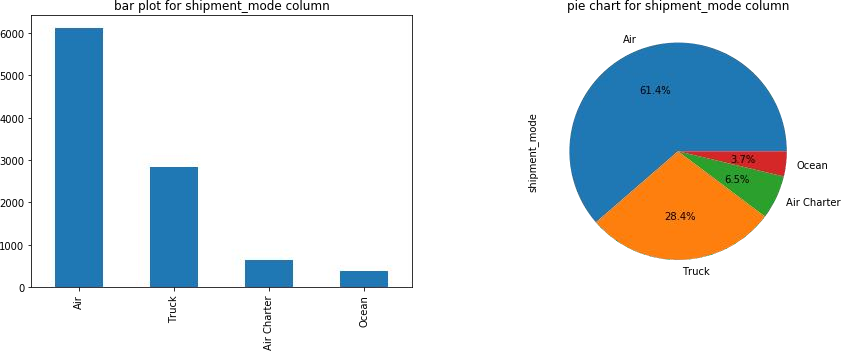
* Identifying and removing duplicates
* Handling missing values (e.g., replacing them with a default value or dropping them)
* Correcting data formatting issues (e.g., inconsistent date formats)
* It is important to carefully inspect the data and carefully considering the appropriate cleaning techniques to apply, as these can significantly affect the quality and usefulness of the data.



* 1. **Exploratory Data Analysis (EDA)**

Exploratory Data Analysis refers to the critical process of performing initial investigations on data to discover patterns, spot anomalies, test hypotheses and to check assumptions with the help of summary statistics and graphical representations.

* **Bar charts**: Show the distribution of a categorical variable or the comparison of multiple variables.
* **Histograms**: Shows the distribution of a continuous variable.
* **Scatter plots**: Shows the relationship between two continuous variables.
* **Box plots**: Shows the distribution of a continuous variable by displaying the median, quartiles, and outliers.
* **Heat maps**: Shows the relationship between two categorical variables or the distribution of a continuous variable over two categorical variables.
* **Pie charts**: Shows the proportion of a categorical variable.



* 1. **Data Pre-Processing**

Before building any model, it is crucial to perform data pre-processing to feed the correct data to the model to learn and predict. Model performance depends on the quality of data fed to the model to train.

This process includes

* Data transformation: Modifying the data to fit the requirements of the analysis or model being used.
* Data scaling: Normalizing the data by scaling it to a specific range.
* Data reduction: Reducing the number of features or samples in the dataset to improve the efficiency of the analysis or model.
* Outlier Treatment: Outliers Detection and Removal
  1. **Feature Engineering**

Feature engineering is the process of designing and creating new features or variables from existing data to improve the performance of a machine-learning model. It is a key step in the data preprocessing process, as the quality and relevance of the features can significantly affect the model's ability to learn and generalize.

Some common techniques for feature engineering include

* Feature selection: Selecting a subset of the most relevant features from the data.
* Feature extraction: Creating new features from existing data by applying transformations or combining existing features.
* Feature creation: Generating new features based on domain knowledge or by applying algorithms to the data.
  1. **Model Selection**

In This step we apply different machine learning algorithms to our processed data and select the model with the best results for further Hyper-parameter tuning to make the best possible model that can be made for accurate and correct prediction.

Model building is an iterative process, and the specific steps and techniques used will depend on the nature of the data and the goals of the analysis.

* 1. **Model Evaluation**

We evaluate our regression model based on performance metrics such as R- square, Adjusted R-square, and Root mean square values. We can also compare the model's predictions to the actual values using a scatter plot.

* 1. **Deployment Process**

To deploy a model, we used the following steps:

1. Save the trained model as a pickle file using Python's pickle library.
2. Create a Flask app in Python, which will act as the server for your model.
3. Define the routes for the Flask app, which will determine the behaviour of the server when it receives different HTTP requests.
4. In the routes, you can load the pickle file and use it to make predictions based on the input received in the request.
5. We created HTML templates to display the results of the predictions on a website.
6. Test the Flask app using Postman or a similar API testing tool to ensure it is working correctly.

**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. | 1. Application URL should be accessible to the user. |
| Verify whether the application  loads completely for the user when the URL is accessed | 1. Application URL is accessible 2. The application URL is   deployed. | Application URL should load  completely for the user when  the URL is accessed. |
| Verify whether the user can see the input field after opening URL | 1. Application is accessible | User should be able to see input fields after opening the URL |
| Verify whether the user can edit all  the input fields | 1. Application is accessible | User should be able to edit all the input fields |
| Verify whether the user has options  to filter the input fields | 1. Application is accessible | User should filter the options of input fields |
| Verify whether the user gets submit  button to submit the inputs | 1. Application is accessible | User should get submit button to  submit the inputs |
| Verify whether the user can see the  output after submitting the inputs | 1. The application is accessible | User should get outputs after submitting the inputs |