Clustering-for-Customer-Segmentation-Understanding

Low Level Design

Contents

1. Introduction

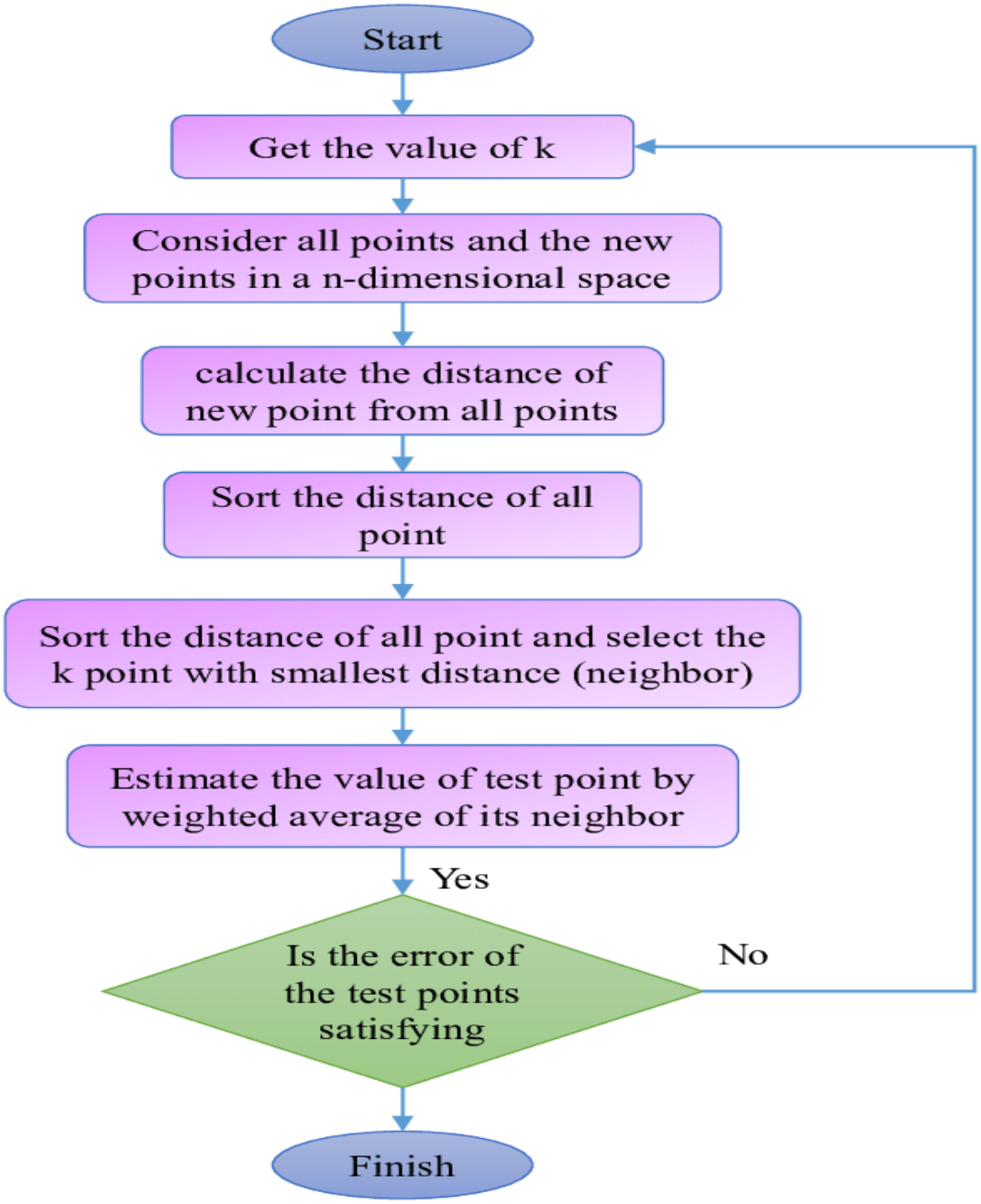
1.1 What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Clustering-for-Customer-Segmentation-Understanding. 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-bystep 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

2. Architecture



3. Architecture Description

3.1. Data Description

The sample dataset summarizes the usage behaviour of about 9,000 active credit cards. Holders during the last 6 months. The file is at the customer level, with 18 behavioural variables.

- Variables of the Dataset
- Balance
- Balance Frequency
- Purchases
- One-off Purchases
- Instalment Purchases
- Cash Advance
- Purchases Frequency
- One-off Purchases Frequency
- Purchases Instalment Frequency
- Cash Advance Frequency
- Cash Advance
- TRX Purchases
- TRX Credit Limit
- Payments
- Minimum Payments
- PRC
- Full Payment
- Tenure Cluster

From this dataset, we need to calculate some patterns, as it is an unsupervised method, so we don't know what to calculate exactly.

The steps to be followed for implementation are given below:

3.2. Data Pre-processing

- **Importing Libraries**

  We will import the libraries for our model, which is part of data pre-processing. The **numpy** we have imported for the performing mathematics calculation, **matplotlib** is for plotting the graph, and **pandas** are for managing the dataset.

- **Importing the Dataset:**

  Next, we will import the dataset that we need to use. So here, we are using the Customer_Data.csv dataset.

- **Extracting Independent Variables**

  Here we don't need any dependent variables for the data pre-processing step as it is a clustering problem and we have no idea what to determine.

3.3. Finding the optimal number of clusters using the elbow method

In the second step, we will try to find the optimal number of clusters for our clustering problem. So, as discussed above, we are going to use the elbow method for this purpose.

As we know, the elbow method uses the WCSS concept to draw the plot by plotting WCSS values on the Y-axis and the number of clusters on the X-axis. So we are going to calculate the value for WCSS for different k values ranging from 1 to 10. In our graph the elbow point is at 5. So the number of clusters here will be 5.

3.4. Data from User

Here we will collect all data from user such as Variables of the Dataset, Balance, Balance Frequency, Purchases, One-off Purchases, Instalment Purchases, Cash Advance Purchases Frequency, One-off Purchases Frequency, Purchases Instalment Frequency, Cash Advance Frequency, Cash Advance, TRX Purchases, TRX Credit Limit, Payments, Minimum Payments, PRC, Full Payment, and Tenure Cluster.

3.5. Data Validation

Here Data Validation will be done, given by the user

3.6. Data Clustering

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

3.7. Model Call for Specific Cluster

Based on the cluster number, the respective model will be loaded and will be used to predict/Recommend the data for that cluster.

3.8. Recipe Recommendation & Saving Output in Database

After calling model Recipe/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.

3.9. Deployment

Building the **Streamlit App**. Creating the Streamlit UI. Loading the Saved Model & Making Real-Time Predictions. Deploying Machine Learning Models with Python and Streamlit.