Low Level Design

**Customer Segmentation System**

|  |  |
| --- | --- |
| Attributes | Values |
| Written By | Thomas Antony |
| Document Version 0.3 | 0.1 |
| Last Revised Date | 29 – Feb -2024 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1 |  | Thomas | First Draft |
|  |  |  |  |
|  |  |  |  |

**Document Control**

Table of Contents

[1. Data Ingestion Module: 4](#_Toc160181876)

[1.1 Responsibilities: 4](#_Toc160181877)

[1.2 Libraries Used: 4](#_Toc160181878)

[2. Data Validation Module: 4](#_Toc160181879)

[2.1 Responsibilities: 4](#_Toc160181880)

[2.2 Libraries Used: 4](#_Toc160181881)

[3. Data Transformation Module: 5](#_Toc160181882)

[3.1 Responsibilities: 5](#_Toc160181883)

[3.2 Libraries Used: 5](#_Toc160181884)

[3.3 Description 5](#_Toc160181885)

[4. Customer Segmentation Module/Model Training Module: 5](#_Toc160181886)

[4.1 Responsibilities: 5](#_Toc160181887)

[4.2 Libraries Used: 5](#_Toc160181888)

[**4.**3 Description 5](#_Toc160181889)

[5. Model Evaluation Module: 5](#_Toc160181890)

[5.1 Responsibilities: 5](#_Toc160181891)

[5.2 Libraries Used: 5](#_Toc160181892)

[6. API Module/Prediction Module: 6](#_Toc160181893)

[7. Key Technologies: 7](#_Toc160181894)

[8. Hardware Requirements: 7](#_Toc160181895)

[9. Event Logging Module 7](#_Toc160181896)

[10. Error Handling 7](#_Toc160181897)

[ Purpose: 7](#_Toc160181898)

[ Sub-Components: 7](#_Toc160181899)

[Conclusion: 8](#_Toc160181900)

## 1. Data Ingestion Module:

## 1.1 Responsibilities:

* **Collect Data:**
  + Utilize the AWS SDK (boto3) to connect to the specified Amazon S3 bucket.
  + Download transaction and customer data to the local environment.
* **Data Versioning:**
  + Integrate DVC to manage data versioning.
  + Use DVC commands to track changes in the ingested data.

### 1.2 Libraries Used:

* **boto3:** For interacting with AWS S3.
* **DVC:** For data version control.

## 2. Data Validation Module:

### 2.1 Responsibilities:

* **Validate Data Structure:**
  + Define a schema for the expected data format and features.
  + Implement validation checks to ensure the ingested data adheres to the predefined structure.

### 2.2 Libraries Used:

* **pandas:** For handling and validating data.

## 3. Data Transformation Module:

### 3.1 Responsibilities:

* **Transform Data:**
  + Preprocess and transform the raw data into a format suitable for machine learning models.
  + Handle any missing values, outliers, or categorical encoding as necessary.

### 3.2 Libraries Used:

* **pandas:** For data manipulation and transformation.
* **Sklearn:** For Scaling and dimension reduction

### 3.3 Description

Data has been scaled to a similar unit and dimensions of the data has been reduced using PCA and created a preprocessor object that can be used to preprocess the input data from users

## 4. Customer Segmentation Module/Model Training Module:

### 4.1 Responsibilities:

* **Apply K-Means Clustering:**
  + Use scikit-learn to apply K-Means clustering on preprocessed data.
  + Generate clustering rules for each customer.

### 4.2 Libraries Used:

* **scikit-learn:** For K-Means clustering

### **4.**3 Description

Trained Model Has been saved for predicting user inputs in the model training artifact

## 5. Model Evaluation Module:

### 5.1 Responsibilities:

* **Evaluate Clustering Model:**
  + Implement a module to evaluate the performance of the clustering model.
  + Used silhouette score for tracking performace.
* **MLflow Integration:**
  + Utilize MLflow to log experiments and keep track of model performance.

## 5.2 Libraries Used:

* **scikit-learn:** For evaluating clustering performance.
* **MLflow:** For experiment tracking and model versioning.

### 6. API Module/Prediction Module:

**6.1 Responsibilities:**

* **Create RESTful API:**
  + Develop API endpoints using Flask/FastAPI.
  + Implement routes for real-time customer segment predictions.
* **HTML Form for User Inputs:**
  + Designed an HTML form for users to input customer data.
* **Retraining Endpoint:**
  + Include an endpoint (**/train**) for retraining the entire application if model degradation is identified.

**6.2 Libraries Used:**

* **Flask/FastAPI:** For API development.
* **HTML:** For simple Webpage design.

## 7. Key Technologies:

**7.1 Clustering Algorithm:**

* **K-Means:** For customer segmentation.

**7.2 Programming Languages:**

* **Python:** For data processing, clustering, and API development.
* **HTML/CSS:** For simple API UX.

**7.3 Tools and Libraries:**

* **Numpy, Pandas:** For data manipulation.
* **Scikit-learn:** For clustering and potential model training.
* **DVC:** For data version control.
* **MLflow:** For experiment tracking and model versioning.
* **Flask/FastAPI:** For API development.

## 8. Hardware Requirements:

* **PC:** The system is designed to run on a standard personal computer.

## 9. Event Logging Module

* **Purpose:**
  + Log every event within the system for debugging and monitoring.
* **Sub-Components:**
  + Custom Exception Handling
  + Logging Methods (Database/File)
  + Non-blocking Logging

## 10. Error Handling

### Purpose:

* + Display explanations for encountered errors.

### Sub-Components:

* + - Custom Exception Function
    - Error Logging

## Conclusion:

**In conclusion, the Low-Level Design (LLD) intricately defines the implementation details for each module in the Customer Segmentation and Classification project. The detailed breakdown provides a roadmap for developers to follow during the actual development phase. Here are additional aspects and considerations to enhance the conclusion:**

**1.Scalability:**

* + **Considerations for scalability have been embedded in the design, allowing the system to handle increasing volumes of data seamlessly. The modular architecture enables easy integration of additional features or enhancements to accommodate future business requirements.**

**2.Flexibility and Customization:**

* **The design emphasizes flexibility, allowing for easy customization based on evolving business needs. Developers can extend or modify modules to incorporate new algorithms, data sources, or even experiment with alternative clustering techniques.**

**3.Continuous Integration and Deployment (CI/CD):**

* **Incorporating CI/CD practices into the development pipeline to automate testing, integration, and deployment processes. This ensures a more efficient and reliable development lifecycle, reducing manual errors and promoting consistent releases.**

**4.Monitoring and Logging:**

* **The Event Log section highlights the importance of detailed logging for system events. Integrating comprehensive monitoring tools can provide real-time insights into system performance, helping to identify potential bottlenecks or anomalies.**

**4.User Experience (UX) Improvements:**

* **Acknowledge that the project could benefit from UX improvements. Future iterations could focus on enhancing the user interface for a more intuitive and visually appealing experience, possibly incorporating user feedback for continuous improvement.**