Customer Personality Analysis

Low Level Design

Domain: Customer relationship

Creator: Vinayaka Uppar

Date: 07.02.2024

Document Version Control

Date issued	Version	Description	Author
JAN 06, 2023	1.1	First Draft	Vinayaka.uppar
JAN 16, 2024	1.2	Added the models	Vinayaka.uppar

Contents

Introduction 3
What is Low-Level Design Document? 3
Scope 3
Architecture 4
Architecture Description 5
Data Preparation 5
Data Description 5
Data Preprocessing 5
Exploratory Data Analysis 5
Feature Engineering 6
Model Development 6
Model implementation 6
Hyper-parameter Tuning 6
Model Evaluation 6
Deployment 7
Designing UI with Anvil 7
Designing a server 7
Code deployment on cloud 7
Deployment Process 7
Unit cases 8

Introduction

What is Low-Level Design Document?

A Low-Level Design (LLD) Document in software development is a comprehensive and detailed guide that provides an in-depth view of the technical aspects and internal workings of a system. It goes beyond the high-level architecture presented in High-Level Design (HLD) and breaks down the system into smaller, manageable components. In the context of the "Facebook Post Status Prediction" project, the LLD would elaborate on how each module functions, the specific algorithms used for sentiment analysis and emotion prediction, the detailed structure of the database, and the interactions between different components. It acts as a roadmap for developers, offering a clear understanding of the system's intricacies and facilitating the smooth implementation of the project.

Scope

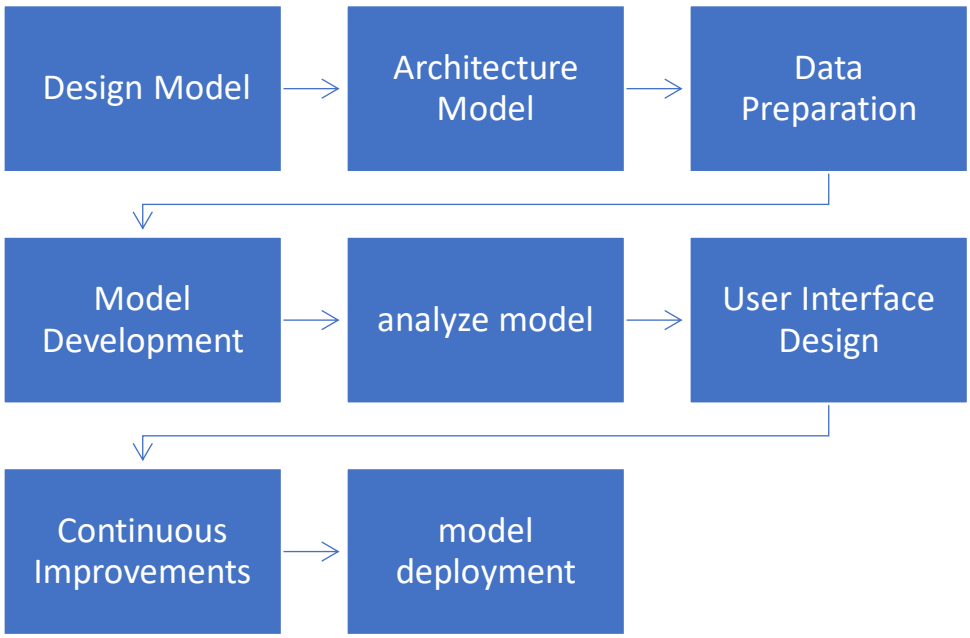
The scope of the Customer Personality Analysis project encompasses understanding customer sentiments and behaviors to inform business strategies. It involves collecting and analyzing customer data such as feedback, reviews, and actions to derive insights into their preferences and attitudes towards products or services. The project aims to develop machine learning models for sentiment analysis and behavioral pattern recognition. Key components include data collection, cleaning, feature engineering, model training, and performance evaluation. The project's focus is to provide actionable insights to businesses for improving customer engagement and satisfaction, ultimately driving business growth and success.

Architecture

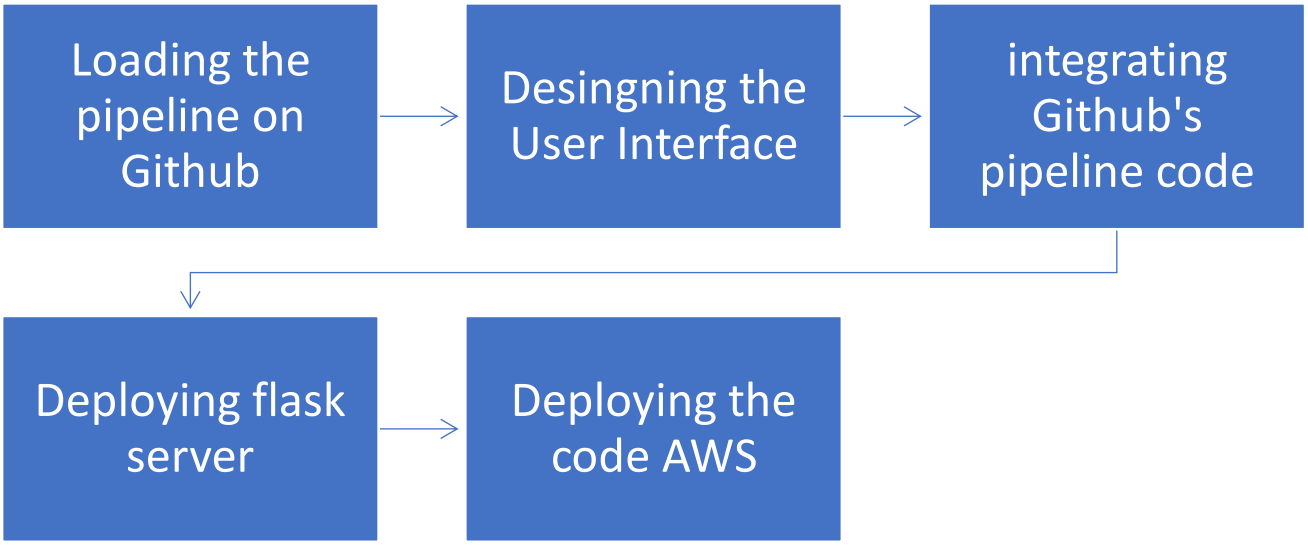
Data Preparation

Model Development

Deployment



Deployment



Architecture Description

Data Preparation

Data Description

Data Preprocessing

Data preprocessing in the Customer Personality Analysis project involves cleaning and transforming raw data into a format suitable for analysis. This includes handling missing values, removing duplicates, and standardizing data formats. Text data may undergo tokenization, stop word removal, and stemming or lemmatization to normalize text features. Numeric data may require scaling or normalization to ensure all features contribute equally to the analysis. Preprocessing techniques aim to enhance the quality and reliability of the data, reducing noise and improving the performance of machine learning models. Overall, effective data preprocessing is essential for accurate analysis and meaningful insights into customer behavior and sentiments.

Exploratory Data Analysis

Exploratory Data Analysis (EDA) in the Customer Personality Analysis project involves visually and statistically exploring the dataset to understand its structure, patterns, and relationships. It includes techniques such as summary statistics, data visualization (e.g., histograms, scatter plots, and heatmaps), and correlation analysis. EDA helps identify trends, outliers, and potential issues in the data, guiding feature selection and preprocessing decisions. By gaining insights into the dataset's characteristics, EDA lays the foundation for building predictive models and extracting meaningful insights about customer behavior and preferences, ultimately contributing to informed business decisions and strategy formulation.

Feature Engineering

Feature Engineering in the Customer Personality Analysis project involves transforming raw data into informative features that enhance the predictive power of machine learning models. This process includes creating new features, such as sentiment scores from text data or derived variables from existing ones, to capture meaningful patterns and relationships. Feature engineering aims to improve model performance by providing relevant input variables that better represent the underlying characteristics of the data. By selecting, transforming, and augmenting features strategically, this step enhances the model's ability to extract insights and make accurate predictions about customer personality traits and behaviors.

Model Development

Model implementation

Model implementation in the Customer Personality Analysis project involves deploying the trained machine learning model to make predictions on new data. This process includes integrating the model into the production environment, where it can receive input data, perform inference, and output predictions. Model implementation ensures that the insights gained from the analysis can be effectively utilized in real-world scenarios, such as identifying customer preferences or behavior patterns. By operationalizing the model, businesses can leverage its predictive capabilities to make informed decisions and optimize their strategies for customer engagement and satisfaction.

Hyper-parameter Tuning

Hyper-parameter tuning is the process of optimizing the settings of a machine learning algorithm to enhance its performance. It involves adjusting parameters that are not learned during training, such as the learning rate or the number of estimators in a model.

Model Evaluation

Model evaluation assesses the performance of a machine learning model on unseen data to determine its effectiveness and generalization capability. Common

evaluation metrics include accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC).

Deployment

Designing UI

Designing the user interface (UI) involves creating an intuitive and visually appealing interface for users to interact with the customer personality analysis system. It includes elements such as layout design, color schemes, typography, and interactive components like buttons and forms

Designing a server

Designing a server involves configuring and setting up the infrastructure required to host the customer personality analysis application. This includes selecting appropriate server hardware, operating system, and networking components

Code deployment on cloud

Code deployment on the cloud involves transferring the application code and associated resources from a local development environment to a cloud platform for hosting and execution. This process typically includes packaging the application code into deployable artifacts, configuring cloud services such as virtual machines or containers, and orchestrating the deployment process using tools like AWS CodeDeploy or Azure DevOps.

Deployment Process

The deployment process on the cloud involves several steps to ensure the successful hosting and execution of the application. Initially, the application code and associated resources are packaged into deployable artifacts. Then, cloud services such as AWS Elastic Beanstalk or Azure App Service are configured to provide the necessary infrastructure for hosting the application. Next, the artifacts are uploaded to the cloud platform, and deployment configurations are specified, including scaling options and monitoring settings. Finally, the deployment process is initiated, during which the cloud platform orchestrates the deployment, manages resource allocation, and ensures that the application is accessible to users. This cloud-based deployment process enhances

performance by leveraging scalable infrastructure, automated deployment mechanisms, and streamlined management capabilities,

Unit cases

Test Case: Cloud Deployment

Description: This test case verifies the successful deployment of the customer personality analysis application on a cloud platform.

Pre-requisite:

Valid cloud account credentials.

Access to the cloud deployment environment.

Deployable artifacts of the application (e.g., Docker image, application code).

Test Steps:

Access Cloud Console:

Open the web browser.

Navigate to the cloud provider's console.

Log in using valid credentials.

Select Deployment Service:

Choose the deployment service (e.g., AWS Elastic Beanstalk, Azure App Service).

Create Deployment Environment:

Click on the option to create a new deployment environment.

Select the appropriate configuration (e.g., region, platform).

Provide a unique name for the environment.

Upload Artifacts:

Upload the deployable artifacts (e.g., Docker image, application code).

Configure any necessary environment variables or settings.

Configure Deployment Options:

Specify deployment options such as auto-scaling, load balancing, and monitoring settings.

Review and confirm the deployment configuration.

Initiate Deployment:

Click on the deploy button to initiate the deployment process.

Monitor the deployment progress through the cloud console.

Expected Result:

The deployment process completes successfully without errors.

The deployed application is accessible via the provided URL.

The application functions as expected, allowing users to analyze customer personality based on input data.

The deployed application scales automatically to handle varying loads and remains available and responsive.