YumTum Project Report (Outline)

1. Project Description (1-2 Pages) (10 points) - Samarth

Overview: A recommendation-based application with AWS using Databricks to create jupyter notebook scripts to create the pipeline for the predictions.

As the size of the data is massive we decided to use databricks to connect to AWS and create the resources we will need for our project.

A databricks workspace is created that connects to AWS and creates a Stack in AWS CloudFormation. The stack creates the resources in S3, lambda, and EC2 that are used by the cluster and data warehouse in databricks for our application.

Databricks can take care of the storage and database, and the scaling of the model.

Problem Statement: To build a restaurant recommendation system based on:
1) Customer Details such as
    - Id
    - Location
    - Previous Orders
    - Gender
    - language
2) Location details such as
    - Customer Location
    - Restaurant Location
    - Delivery Time and Cost
    - Feasibility
3) Restaurant Details such as
    - Cuisines
    - Discounts (Normal and Promo Code)
    - Ratings

Data Sources: Dataset Link (Kaggle)
Github: https://github.com/Bryan-Az/YumTum

Requirements

Data Requirements

Our goal is to create a restaurant recommendation system by the name “YumTum”. This system does not sell food, but it provides recommendations to restaurants that exist within a user’s given location using features within the restaurant recommendation system.

As recommendations must be calculated on a user-by-user basis, the types of data that are used should contain a complete representation for the users.

Luckily, the dataset we are using has been prepared and connects users to the user’s customer details, geographical and delivery details, and restaurant details.

To make great recommendations, we would need a large volume and variety of data. A large volume of data will provide us with low bias and high variance. To balance, we have done careful preparation of the data and ran stratified sampling of the data for more carefully crafted and personalized recommendation.

The types of data were also taken into consideration when preparing the data in the cleaning and preprocessing stages. Our dataset contained multiple features of varying data types, including categorical, numerical, string, date, and further, included metrics like ‘discount percentage’ that were included in the raw data.

Technical Requirements

- Tools: Databricks, Kaggle (Data Source), Github (Central Repo)
- Database: AWS S3, Databricks Cloud SQL
- Software: Pyspark, Scikit-Learn, Python

3. Knowledge Discovery in Databases (KDD)(EDA) (15 points)

Selection: Identify relevant data sources and subsets of data.

Preprocessing: Detail cleaning and normalization steps.

Transformation: Explain how data is transformed into a suitable format for mining.

Data Mining: Describe the algorithms and techniques used to extract patterns.
Interpretation/Evaluation: Discuss how results are interpreted and evaluated against objectives.

Link: Exploratory Data Analysis

Overview

Step	Dataframe (test, train)	Details
1 Feature Selection	Reduced	
2 Pre-processing	Reduced, Clean	
3 Pre-processing	Feature Engineered	
4 Feature Engineering	Full Feature Engineered	
5 Feature Engineering	Customer Feature Engineered	
6 Feature Selection	Location Density Feature Engineered	
7 Feature Selection	Cuisine Feature Engineered	
8 Pre-processing	Pre-processed	
9 Pre-processing	Encoded	
10 Pre-processing	Normalized	

4. Feature Engineering (30 points)

Steps 4 & 5
Features engineered: Distance between Customer and Restaurant (Haversine Distance), Day-wise Open Hours, Customer's Multiple Orders History, Customer's Use of Multiple Locations, Cuisine Diversity Score

5. High-Level Architecture Design (10 points)
System Overview: Present a bird’s eye view of the system architecture.
Data Flow: Show how data moves through the system.
Components: Identify major components like databases, processing units, etc.

Step 1: test_full.csv & train_full
Step 1 - > Step 2: (test|train)_reduced
Step 2 - > Step 3: (test|train)_reduced_clean
Step 3 - > Step 4: (test|train)_feature_engineered
Step 4 - > Step 5: (test|train)_full_feature_engineered
New Table
Step 5 - > Step 6: (test|train)_customer_feature_engineered
New Table
Step 6 -> Step 7: (test|train)_location_density_feature_engineered
Step 7 -> Step 8: test|train_cuisine_feature_engineered
New Table
Step 8 -> Step 9: test|train_preprocessed
Step 9 -> Step 10: test|train_encoded
Step 10 -> Step 11: test|train_normalized

6. Data Flow Diagram & Component Level Design (5 points)
Data Flow Diagram: Visual representation of data flow through the system.

Component Interaction: How different components interact with each other.

7. Sequence or Workflow (5 points)
Step-by-Step Process: Outline the sequential steps from data ingestion to output.
Automation: Mention any automation in the workflow.

8. Data Science Algorithms & Features Used (20 points)
Algorithms: Detail the algorithms used and justify their selection.
Features: Discuss the features used by these algorithms.

9. Interfaces – RESTful & Server-Side Design (10 points)
RESTful APIs: Design of APIs for data exchange.
Server-Side Logic: Describe the server-side processing and logic.

10. Client-Side Design (20 points)
User Interface: Design and usability of the client-side application.
Client-Server Interaction: How the client interacts with the server.

11. Testing (Data Validation/nFold) (25 points)

Validation Techniques: Discuss data validation techniques used.
Cross-Validation: Explain the n-fold cross-validation process.

12. Model Deployment (25 points)
Deployment Strategy: Discuss how the model will be deployed (cloud, on-premises, etc.).
Monitoring and Maintenance: How the model will be monitored and maintained post-deployment.

13. High-Performance Computing (HPC) (20 points)
Usage: Explain how HPC is used for data processing or model training.
Benefits: Discuss the benefits of using HPC in your project.

14. Documentation (10 points)
Code Documentation: Detail the documentation provided for the codebase.
System Documentation: Manuals and guides for understanding the system.

15. Design Patterns Used (10 points)
Software Design Patterns: Describe any design patterns (MVC, Singleton, etc.) used in the project.

16. AutoML or Serverless AI (15 points)
AutoML: If used, explain how AutoML has been leveraged.
Serverless AI: Discuss the use of serverless computing in the project.

17. Data Engineering (Bonus) (30 points)
Data Pipelines: Design and implementation of data pipelines.
ETL Processes: Details on Extract, Transform, and Load processes.
Data Storage: Discuss the data storage solutions used.

18. Active Learning or Feedback Loop (10 points)
Active Learning: How the model learns from new data over time.
Feedback Loop: Mechanisms for incorporating user or system feedback.

19. Interpretability of the Model (20 points)
Transparency: Methods used to make the model's decisions understandable.
Techniques: Techniques like LIME, SHAP, or others are used for interpretability.