A PROJECT REPORT ON

TAILORED DIET RECOMMENDATION SYSTEM USING KNN

Submitted in partial fulfilment of the Requirement for the award of the degree of

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

Submitted by:

Project Associates	Regd. No
M. Madhavan	21095A3204
J. Mahesh	20091A3225
B. S. Rohith Kumar	20091A3238

Under the Guidance of

Supervisor
Mr. B. Bhaskara Rao M. Tech, (Ph. D)
Associate Professor, Dept. of CSE(DS)

Co-Supervisor
Dr. S. Karimulla Basha Ph. D
Assistant Professor, Dept. of CSE(DS)

(ESTD-1995)

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)
RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
Affiliated to JNTUA - Anantapuramu, Approved by AICTE - New Delhi,
Accredited by NBA - New Delhi, Accredited by NAAC with A+ Grade – New Delhi
NANDYAL – 518501 Dist. A.P.

YEAR: 2020 - 2024

RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY
(AUTONOMOUS)
Approved by AICTE, New Delhi; Affiliated to JNTUA-Ananthapuramu,
Accredited by NBA (6-Times); Accredited by NAAC with 'A+' Grade (Cycle-3),
New Delhi;
World Bank Funded Institution; Nandyal (Dist)-518501, A.P

(ESTD – 1995)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

CERTIFICATE

This is to certify that **M. MADHAVAN** (21095A3204), **J. MAHESH** (20091A3225), and **B. S. ROHITH KUMAR** (20091A3238) of IV- B. Tech II- Semester, have carried out the major project work entitled "**TAILORED DIET RECOMMENDATION SYSTEM USING KNN**" under the supervision and guidance of **Mr. B. Bhaskara Rao,** Associate Professor, CSE(DS) Department, and **Dr. S. Karimulla Basha,** Assistant Professor, CSE(DS) Department, in partial fulfillment of the requirements for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering (Data Science)** from **Rajeev Gandhi Memorial College of Engineering & Technology (Autonomous)**, Nandyal is a bonafied record of the work done by them during 2023-2024.

Project Supervisor
Mr. B. Bhaskara Rao M. Tech, (Ph. D)
Associate Professor, Dept. of CSE(DS)

Head of the Department
Mr. B. Bhaskara Rao M. Tech, (Ph. D)
Associate Professor, Dept. of CSE(DS)

Co-Supervisor
Dr. S. Karimulla Basha Ph .D
Assistant Professor, Dept. of CSE(DS)

External Examiner

Place: Nandyal
Date:

CANDIDATE'S DECLARATION

We hereby declare that the work done in this project entitled **"TAILORED DIET RECOMMENDATION SYSTEM USING KNN"** submitted towards completion of the major project in IV- Year II- Semester of B. Tech CSE (Data Science) at the **Rajeev Gandhi Memorial College of Engineering & Technology (Autonomous)**, Nandyal. It is an authentic record of our original work done under the esteemed guidance of **Mr. B. Bhaskara Rao,** Associate Professor, Department of **Computer Science and Engineering (Data Science)**, and **Dr. S. Karimulla Basha,** Assistant Professor, Department of **Computer Science and Engineering (Data Science),** RGMCET(Autonomous), Nandyal.

We have not submitted the matter embodied in this report for the award of any other Degree in any other institutions for the academic year 2023-2024.

By

M. MADHAVAN
21095A3204
J. MAHESH
20091A3225
B. S. ROHITH KUMAR
20091A3238

Dept. of CSE(DS),
RGMCET(Autonomous).

Place: Nandyal
Date:

ACKNOWLEDGEMENT

We manifest our heartier thankfulness of your contentment over our project guide **Mr. B. Bhaskara Rao,** Associate Professor of Computer Science Engineering (Data Science) department, and **Dr. S. Karimulla Basha,** Assistant Professor of Computer Science Engineering (Data Science) department, with whose adroit concomitance the excellence has been exemplified in bringing out this project to work with artistry.

We express our gratitude to **Mr. B. Bhaskara Rao garu,** Head of the Department of Computer Science Engineering (Data Science) department, and all the **Teaching Staff Members** of the Computer Science Engineering (Data Science) of Rajeev Gandhi Memorial College of Engineering and Technology for providing continuous encouragement and cooperation at various steps of our project successful.

Involuntarily, we are perspicuous to divulge our sincere gratefulness to our Principal, **Dr. T. Jaya Chandra Prasad garu,** who has been observed posing valiance in abundance towards our individuality to acknowledge our project work tangentially.

At the outset, we thank our honourable **Chairman Dr. M. Santhi Ramudu garu**, for providing us with exceptional faculty and moral support throughout the course.

Finally, we extend our sincere thanks to all the non-teaching **Staff Members** of the CSE(DS) Department who have co-operated and encouraged us in making our project successful.

Whatever one does, whatever one achieves, the first credit goes to the **Parents** be it not for their love and affection, nothing would have been responsible. We see in every good that happens to us their love and blessings.

BY

M. Madhavan (21095A3204)
J. Mahesh (20091A3225)
B. S. Rohith Kumar (20091A3238)

ABSTRACT

In recent times, there has been an increase in the incidence of diseases affecting people across various age groups, including both the elderly and younger populations. Recognizing this trend, the World Health Organization underscores the significance of adopting proper nutrition for improved health outcomes. In response to this imperative, a personalized food recommendation system has been crafted to cater to individual dietary requirements and preferences. This system draws upon an expansive dataset sourced from Kaggle's "Food.com - Recipes and Reviews," which is freely available under the CC0: Public Domain license. Comprising a vast collection of 522,517 recipes spanning 312 categories, this dataset offers a diverse array of nutritional information to support the development of tailored dietary recommendations

KNN emerges as the most effective algorithm for our personalized food recommendation system, outperforming others based on rigorous evaluation measures such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) Score. Its superiority can be attributed to its ability to accurately predict optimal food choices for individuals, leveraging the diverse nutritional information contained in the extensive dataset from Kaggle's "Food.com - Recipes and Reviews."

The personalized food recommendation system not only tailors suggestions to individual dietary needs and preferences but also introduces a novel feature that recommends food items based on users' Body Mass Index (BMI) and Basal Metabolic Rate (BMR). This personalized approach takes into account users' unique physical characteristics, ensuring that the recommendations align closely with their individual nutritional requirements.

Through comprehensive visualizations of nutrient values, ingredient lists, and preparation instructions, users are empowered to make informed decisions about their diet, thereby supporting their overall health and well-being. By prioritizing the accuracy and relevance of recommendations, our research contributes to the advancement of personalized diet recommendation systems, offering users valuable and customized suggestions to achieve their dietary goals and preferences.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES **PAGE NO**

LIST OF TABLES **PAGE NO**

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

Our project stands out from existing applications in several key aspects. Firstly, while many existing systems focus solely on personalized dietary recommendations, our approach goes beyond by incorporating a content-based recommendation system using a vast dataset from Food.com, which contains over 500,000 recipes. This allows for a broader range of dietary options to be considered, catering to diverse tastes and preferences.

Unlike many existing applications, our project harnesses the power of a content-based recommendation system, drawing from a vast dataset comprising over 500,000 recipes sourced from Food.com. This extensive collection enables a more comprehensive analysis of dietary options, accommodating diverse tastes and preferences. Moreover, our project sets itself apart by employing a range of machine learning models, including KNN, RF, DT, and LR, to evaluate and optimize recommendation accuracy.

One of the unique features of our project is the integration of Basal Metabolic Rate (BMR) into the recommendation process, allowing for the creation of a generalized diet recommendation system. By considering individual metabolic needs, our system offers tailored suggestions that cater to a broader audience, enhancing its applicability and effectiveness.

In summary, our project distinguishes itself through its multifaceted approach, leveraging a vast recipe dataset, employing diverse machine learning techniques, and incorporating BMR-based recommendations. These elements collectively contribute to a more robust and adaptable system, poised to address the diverse needs of users and make a meaningful impact in the realm of personalized nutrition.

1.1 Problem Statement

In today's health-conscious society, providing individuals with personalized dietary recommendations that cater to their unique preferences and nutritional needs remains a significant challenge. Existing methods often lack accuracy and overlook individual variations, resulting in less effective guidance. This project aims to address this issue by developing a Tailored Diet Recommendation System that leverages data-driven insights. By offering customized dietary plans, the system empowers individuals to achieve a balanced relationship between their body and plate, promoting enhanced well-being and overall health.

1.2 Objectives

Our project aims to create a Personalized Diet Recommendation System using the extensive "Food.com - Recipes and Reviews" dataset. The system suggests recipes that cater to individual dietary needs and preferences. Users receive recommendations based on diverse nutritional information, including ingredient lists, nutrient values, and preparation instructions to facilitate optimal meal planning.

To determine the most effective model for food recommendation, we tested various machine learning algorithms such as K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR). The models were evaluated using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) Score.

We introduced a feature that provides diet recommendations based on an individual's Basal Metabolic Rate (BMR). This allows for personalized dietary suggestions tailored to each user's unique physical characteristics. The addition of BMR-based recommendations enhances the system's overall effectiveness.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

Title: Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model

- **Author:** C. Iwendi et al.
- **Description:** The paper provides dietary recommendations for patients using machine learning and deep learning models for optimized patient health management.

Title: Diet Recommendation System based on Different Machine Learners

- **Author:** Shah, Degadwala, and Vyas
- **Description:** The majority of individuals are frantically trying to reduce weight, gain weight, or keep their health in check.

Title: Person—personalized expert recommendation system for optimized nutrition

- **Author:** Chen et al.
- **Description:** The main barrier of con nesting genetic information to personalized diets is the complexity of data and the scalability of the applied systems, optimized nutrition , consumer personalized food item filtering and recommendation.

Title: A Food Recommender System Considering Nutritional Information and User Preferences

- **Author:** Yera Toledo et al
- **Description:** A food recommender system that considers both nutritional information and user preferences, how individuals make dietary choices, individuals according to their physical, physiological data, and further personal information.

Title: Machine learning: Algorithms, real world applications and research directions - SN computer science

- **Author:** Sarker, I.H et al.
- **Description:** Recommendation systems, offering valuable insights into machine learning algorithms.

Title: Human-Behaviour Based Personalized Meal Recommendation and Menu Planning Social System

- **Author:** Sa Islam et al.
- **Description:** A meal recommendation system based on human behavior analysis, transforming the way dietary suggestions are made. Authors used an hierarchical ensemble method applied to predict affectivity upon multiple feature extraction methods and Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) is used to generate a food list based on the predicted affectivity.

2.1 Overview of related work

The paper discusses a variety of existing dietary recommendation systems that serve different purposes and cater to diverse user needs. For example, one approach leverages machine learning in the context of the Internet of Medical Things (IoMT) to provide patient-specific diet recommendations. This method uses real-time health data to offer tailored nutrition plans for individuals based on their current health status and objectives.

Another system provides diet recommendations tailored to individual requirements. These personalized recommendations take into account users' specific needs, preferences, and health conditions to offer diets that are best suited for them. This approach ensures that dietary plans are aligned with the user's objectives and constraints, potentially improving adherence and outcomes.

The paper also mentions a system that employs machine learning and big data to generate meal recommendations based on human behaviour. By analyzing patterns in individuals' eating habits and preferences, this approach offers suggestions that align with users' tastes and lifestyles. Additionally, other systems provide personalized expert nutrition recommendations by combining nutritional information with user preferences, further enhancing the quality and relevance of dietary guidance.

These different methodologies demonstrate the versatility and innovation within the field of dietary management. By utilizing advanced technologies such as machine learning and big data analytics, these systems aim to provide personalized and comprehensive nutritional advice to support individuals in achieving their health and wellness goals.

2.1.1 How our model different from existing base paper

Our model differs from the base paper in several key aspects. While the base paper focused solely on patients, our project extends its scope to cater to the dietary needs and preferences of every individual, encompassing a broader audience. However, it's important to note that our model, while striving to address the needs of diverse demographics, may encounter limitations in recognizing variations in dietary preferences and cultural nuances, particularly across different countries and cultures like India. Additionally, the dataset and approach we employ vary slightly from those of the base paper. While the authors of the base paper considered a classification model using algorithms such as logistic regression, naïve bayes, Recurrent Neural Network (RNN), Multilayer Perceptron (MLP), Gated Recurrent Units (GRU), and Long Short-Term Memory (LSTM), we utilize a different set of machine learning algorithms, including K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR), for predicting optimal food choices.

2.1.2 Existing System

Existing diet recommendation systems in the market leverage a variety of approaches and technologies to provide users with personalized dietary guidance. Many of these systems aim to support individuals in making healthier food choices and achieving their nutritional goals, taking into account their health conditions, preferences, and lifestyle factors.

One common approach is the use of rule-based systems that provide recommendations based on predefined dietary guidelines and nutritional principles. These systems often utilize databases of food items and their nutrient contents to suggest meals or diets that meet specific criteria, such as calorie limits or nutrient balance. While these systems can be effective, they may lack the ability to tailor recommendations precisely to each individual's unique needs.

Another popular approach involves machine learning and artificial intelligence (AI) algorithms. These systems analyse user data such as dietary preferences, health information, and activity levels to generate personalized diet plans. Machine learning models can identify patterns and trends in large datasets, enabling them to provide more targeted and relevant dietary advice. AI-powered systems can also adapt to changing user needs over time, making them more flexible and responsive.

Some systems focus on integrating with wearable devices and health tracking apps to gather real-time data on users' physical activity, sleep, and other health metrics. This data can be used

to adjust dietary recommendations dynamically, ensuring that users receive the most up-to-date advice based on their current health status.

While there are various diet recommendation systems available in the market, they each have their own strengths and limitations. The most effective systems combine rule-based logic, machine learning, and user data to offer personalized and comprehensive dietary guidance that aligns with users' health and wellness goals.

2.1.3 Disadvantages of Existing System

Existing diet recommendation systems in the market, while useful, have certain disadvantages that can impact the effectiveness and user satisfaction of these systems. One of the key issues is the lack of personalization. Many systems rely on generalized dietary guidelines or rule-based approaches that may not account for individual differences in metabolism, health conditions, and preferences. This can lead to recommendations that are not fully tailored to the user's unique needs, reducing the likelihood of successful dietary adherence.

Another disadvantage is the reliance on static data. Some diet recommendation systems use outdated databases of food items and nutritional information, leading to potentially inaccurate or incomplete recommendations. Additionally, these systems may not account for changes in a user's health status or lifestyle over time, resulting in less relevant and effective advice as the user's circumstances evolve.

The user experience of existing systems can also be hindered by limited interaction and feedback capabilities. Many systems do not provide adequate explanations for their recommendations or offer opportunities for users to input their preferences and receive real-time adjustments. This can lead to frustration and reduced engagement, as users may not understand the rationale behind the advice they receive or feel that their needs are being met.

Lastly, some systems may lack integration with other health and wellness tools, such as wearable devices or fitness apps, which could provide valuable data for more comprehensive and dynamic recommendations. This lack of connectivity can limit the ability of the system to offer a holistic approach to health and nutrition, potentially diminishing its overall effectiveness.

2.1.4 Proposed System

The proposed system introduces a tailored diet recommendation approach using machine learning models, which include K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR). These models are carefully chosen to provide accurate predictions of nutrient values in food items based on their nutritional content. The system's objective is to identify the most effective model for offering precise dietary suggestions to users, considering their unique needs and preferences.

The system utilizes a comprehensive dataset that comprises 522,517 recipes across 312 categories, providing a broad base for analyzing and recommending diverse food items. Features such as Calories, FatContent, SaturatedFatContent, SodiumContent, CholesterolContent, CarbohydrateContent, FiberContent, SugarContent, and ProteinContent are used to train the machine learning models. By understanding the relationships between these nutritional features, the models can predict optimal nutrient values for various food items, resulting in highly personalized and precise dietary recommendations.

One notable strength of the proposed system is its rigorous evaluation of model performance using metrics such as Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2). These metrics allow the system to assess the accuracy and reliability of each model by comparing predicted data with actual data. This approach ensures that the system delivers high-quality, personalized dietary advice while minimizing errors.

The proposed system's personalized diet recommendation capabilities offer users tailored guidance that aligns with their health and nutrition goals. By considering user preferences and dietary objectives, the system delivers actionable recommendations that support users in making informed dietary choices. This personalized approach, combined with the system's robust use of machine learning models and comprehensive dataset, positions it as an innovative solution in the domain of personalized nutrition.

2.1.5 Advantages of proposed system

The proposed system offers several advantages over existing diet recommendation systems, focusing on personalization and user-specific recommendations. By using machine learning models such as K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR), the system can provide precise and accurate dietary suggestions

tailored to each user's unique nutritional needs and preferences. These models are trained on a variety of nutritional features, enabling the system to learn patterns and relationships in the data to predict optimal nutrient values for each food item.

One significant advantage of the proposed system is its use of comprehensive evaluation metrics, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) scores, to assess the performance of each model. This meticulous evaluation ensures that the system is capable of accurately predicting nutritional content while minimizing errors. As a result, the system can optimize recommendations and provide reliable dietary advice to individual users.

Additionally, the proposed system's approach leverages a large dataset of recipes and nutritional content, which allows for a more diverse and robust set of dietary recommendations. This dataset, sourced from a reputable platform, includes a wide variety of food items across numerous categories, offering users a comprehensive and extensive selection. By incorporating user preferences and dietary objectives, the proposed system can deliver personalized and actionable guidance, supporting users in achieving their health and nutrition goals.

2.2 Key Concepts and Technologies:

1. Content-Based Filtering: Content-based filtering methods are used to recommend food items based on their nutritional content and relevance to the user's preferences and dietary requirements.

2. Machine Learning Algorithms: Various machine learning algorithms such as K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR) are employed to predict optimal food choices and provide personalized recommendations.

3. Data Preprocessing and Feature Engineering: Preprocessing techniques and feature engineering methods are applied to prepare the dataset for training machine learning models.

4. Model Selection: Based on the nature of the problem and the characteristics of the dataset, appropriate machine learning algorithm is selected for training from the machine learning algorithms

5. Training: Once the model and features are defined, the training process begins. During training, the model learns from the training data by adjusting its internal parameters to minimize the error between predicted and actual outcomes.

6. Tailored Diet Recommendation Systems: This is used to take the input from the user to recommend the food items. The user input includes calories, fat content, saturated fat content, cholesterol content, sodium content, carbohydrate content, fibre content, sugar content, and protein content. Based on these input data further process and recommendation will be generated.

7.Generalized Diet Recommendation Systems: Personalized diet recommendation is focused to take input from the user to recommend generalized diet, the input includes age, height, weight, gender, and activity level and weight maintenance goals.

CHAPTER-3

METHODOLOGY

3. METHODOLOGY

3.1 Research Design

This above architecture shows to be a system for generating recommendations, possibly for a product or content recommendation service.

Data Collection: This is the process of gathering data that will be used to train the recommendation models. The data could come from source Kaggle.

Data Preprocessing: Once the data is collected, it needs to be cleaned and pre-processed to make it suitable for training the models. This could involve handling missing values, normalizing numerical data, encoding categorical data, and other similar tasks.

Model Training: After preprocessing, the data is used to train machine learning models. These models learn patterns from the data that can be used to make predictions or recommendations.

Model Evaluation: Once the models are trained, they need to be evaluated to ensure they are making accurate recommendation. Metrics are MAE, MSE, RMSE and R2.

Developed UI for Tailored Recommendation: This suggests that a user interface was developed to display personalized recommendations to users. These recommendations are likely based on the individual user's behaviour and preferences.

Recommendation: This is the output of the system - the actual recommendations that are shown to the user.

Developed UI for Generalized Recommendation: This suggests that a user interface was also developed to display general recommendations. These recommendations might not be personalized to the individual user, but rather based on overall trends or popular items.

3.2 Data Collection:

Data collected from Kaggle

Name of the Dataset: Food.com -Recipes and Reviews

License: CC0: Public Domain

SI. NO	FEATURES	TYPE
1	RecipeId	Numeric
2	Name	Categorical
3	AuthorId	Numeric
4	AuthorName	Categorical
5	CookTime	Numeric
6	PrepTime	Numeric
7	TotalTime	Numeric
8	DatePublished	Categorical
9	Descrption	Categorical
10	Images	Categorical
11	RecipeCategory	Categorical
12	Keywords	Categorical
13	RecipeIngredienstQuantites	Categorical
14	RecipeIngredientsParts	Categorical
15	AggregatedRating	Numeric
16	ReviewCount	Numeric
17	Calories	Numeric
18	FatContent	Numeric
19	SaturatedFatContent	Numeric
20	CholesterolContent	Numeric
21	SodiumContent	Numeric
22	CarbohydrateContent	Numeric
23	FiberContent	Numeric
24	SugarConent	Numeric
25	ProtteinContent	Numeric
26	RecipeServing	Numeric
27	RecipieYield	Categorical

Table 1: Original data set

The dataset contains:The recipes dataset contains 522,517 recipes from 312 different categories.Though this dataset provides information about each recipe like cooking times, servings, ingredients, nutrition, instructions, and more. For the project we will be choosing only the helpful features for the diet recommendation system, which are

SI. NO	FEATURE	TYPE
1	RecipeId	Numeric
2	Name	Categorical
3	CookTime	Numeric
4	PrepTime	Numeric
5	TotalTime	Numeric
6	RecipeIngredientParts	Categorical
7	Calories	Numeric
8	FatContent	Numeric
9	SaturatedFatContent	Numeric
10	CholesterolContent	Numeric
11	SodiumContent	Numeric
12	CarbohydrateContent	Numeric
13	FiberContent	Numeric
14	SugarContent	Numeric
15	ProteinContent	Numeric
16	RecipeInstructions	Categorical

Table 2: Modified Dataset

Dataset Description:

The Table 1 shows dataset utilized for this project is sourced from Kaggle and is known as "Food.com - Recipes and Reviews." This dataset is available under the CC0: Public Domain license. It comprises a vast collection of 522,517 recipes spanning 312 different categories. Each recipe entry provides comprehensive information such as cooking times, servings, ingredients, nutritional values, instructions, and more. However, for the purpose of this project, only select features relevant to the diet recommendation system have been chosen. Including the name of the recipe, recipe category(categorical type), and various nutritional components such as calories, fat content, saturated fat content, cholesterol content, sodium content,

carbohydrate content, fiber content, sugar content, and protein content. The name of the recipe serves for food item identification and prediction, while the recipe category and nutritional values are crucial for the recommendation system's functioning. The dataset's numeric features provide essential data for analysis and prediction, aiding in the development of a robust and effective diet recommendation system tailored to individual preferences and nutritional needs.

At application level using some of the existing helpful features in the dataset which includes CookTime, PrepTime, TotalTime, RecipeIngredientParts and RecipeInstructions.

3.3 Tools and Technologies Used

1. Domain : Health care and Nutrition
2. Front-End : StreamLit
3. Back-End : Python
4. Technical Domains : Machine Learning
5. Visualization Tools : Matplotlib, Streamlit echarts and Seaborn
6. Other Tools : GitHub
7. Testing : Pytest, Unit Testing, UI Testing
8. Execution tools : Docker, VS code
9. Deployment : Azure cloud

CHAPTER-4

SYSTEM IMPLEMENTATION

4. SYSTEM IMPLEMENTATION

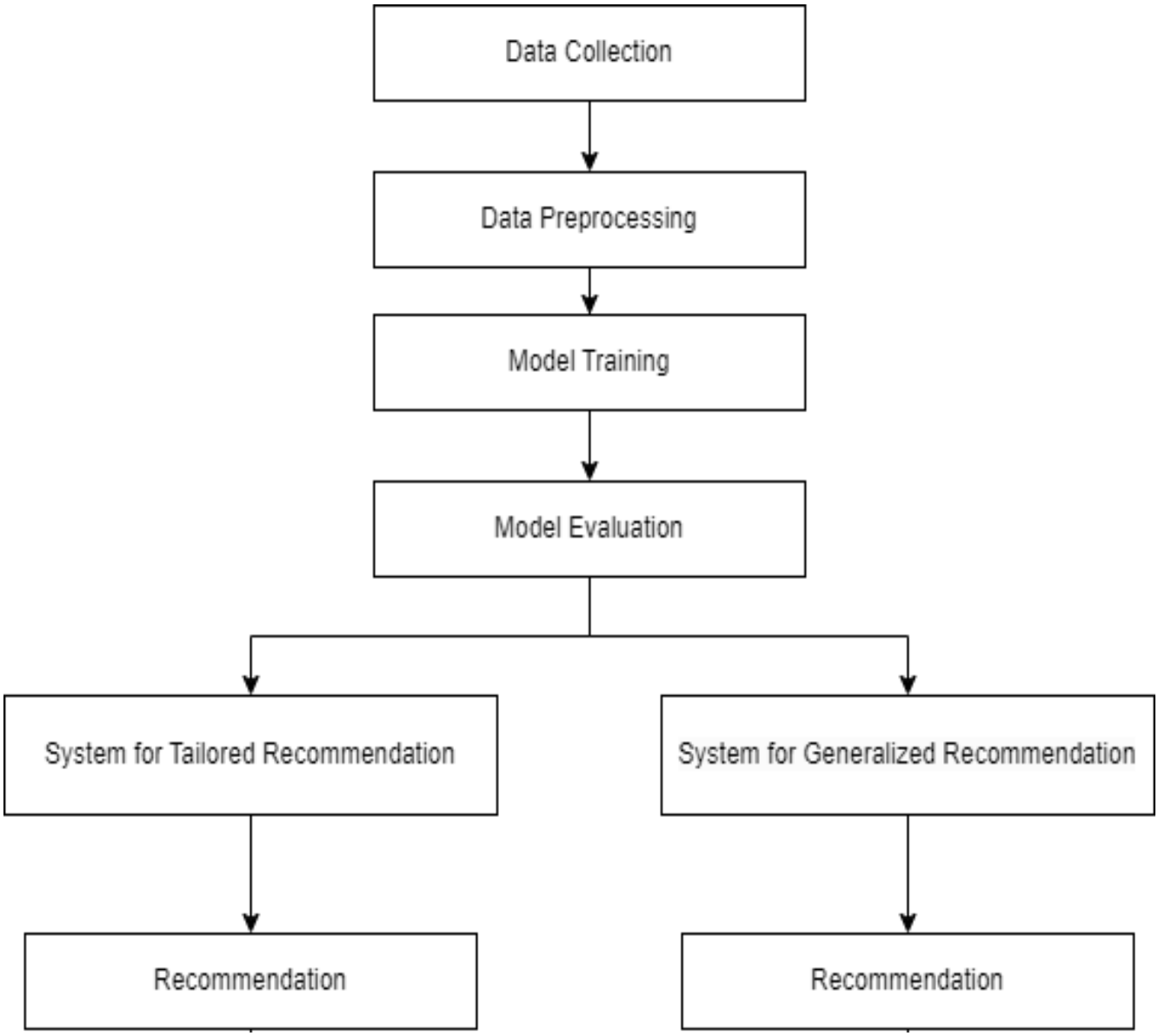


Figure 4.0.1: System Implementation.

This above architecture shows to be a system for generating recommendations, possibly for a product or content recommendation service.

Data Collection: This is the process of gathering data that will be used to train the recommendation models.

Data Preprocessing: Once the data is collected, it needs to be cleaned and pre-processed to make it suitable for training the models. This could involve handling missing values, normalizing numerical data, encoding categorical data, and other similar tasks.

Model Training: After preprocessing, the data is used to train machine learning models. These models learn patterns from the data that can be used to make predictions or recommendations.

Model Evaluation: Once the models are trained, they need to be evaluated to ensure they are making accurate predictions. This could involve using metrics such as precision, recall, F1 score, or ROC AUC score.

System for Tailored Recommendation: This stages states that that model and user interface was developed to display tailored recommendations to users. These recommendations are likely based on the individual user's behaviour and preferences.

Recommendation: This is the output of the system - the actual recommendations that are shown to the user.

System for Generalized Recommendation: This stages states that that model and user interface was also developed to display generalized recommendations. This takes input from the user on age, height, weight and gender and user preferences and recommends the diet recipes.

Data Pre-Processing

During the data pre-processing stage, our first step involved the removal of irrelevant features that held no significance for our project. These irrelevant features were identified and excluded, leaving behind only the essential attributes such as calories, fat content, saturated fat content, and others relevant to our dietary recommendation system. Additionally, we meticulously checked for and eliminated any duplicate values present in the dataset.

Furthermore, we conducted a thorough examination for null values within the dataset. Various techniques were employed to handle these null values effectively, including replacing them with appropriate values, mean imputation, median imputation, frequency imputation, or removing the null data altogether. Outliers, which are data points that significantly deviate from the rest of the dataset, were also addressed during this stage. Boxplot analysis was utilized to detect outliers, and appropriate methods were employed to handle them.

Moreover, different features within the dataset exhibited varying scales of values. To ensure uniformity and facilitate the training and testing processes, we applied the Standard Scalar

technique for normalization. This technique rescales the features to have a mean of 0 and a standard deviation of 1, making it easier to compare and analyse the data. Through these meticulous steps in data pre-processing, we ensured that our dataset was cleaned, standardized, and ready for further analysis and model development.

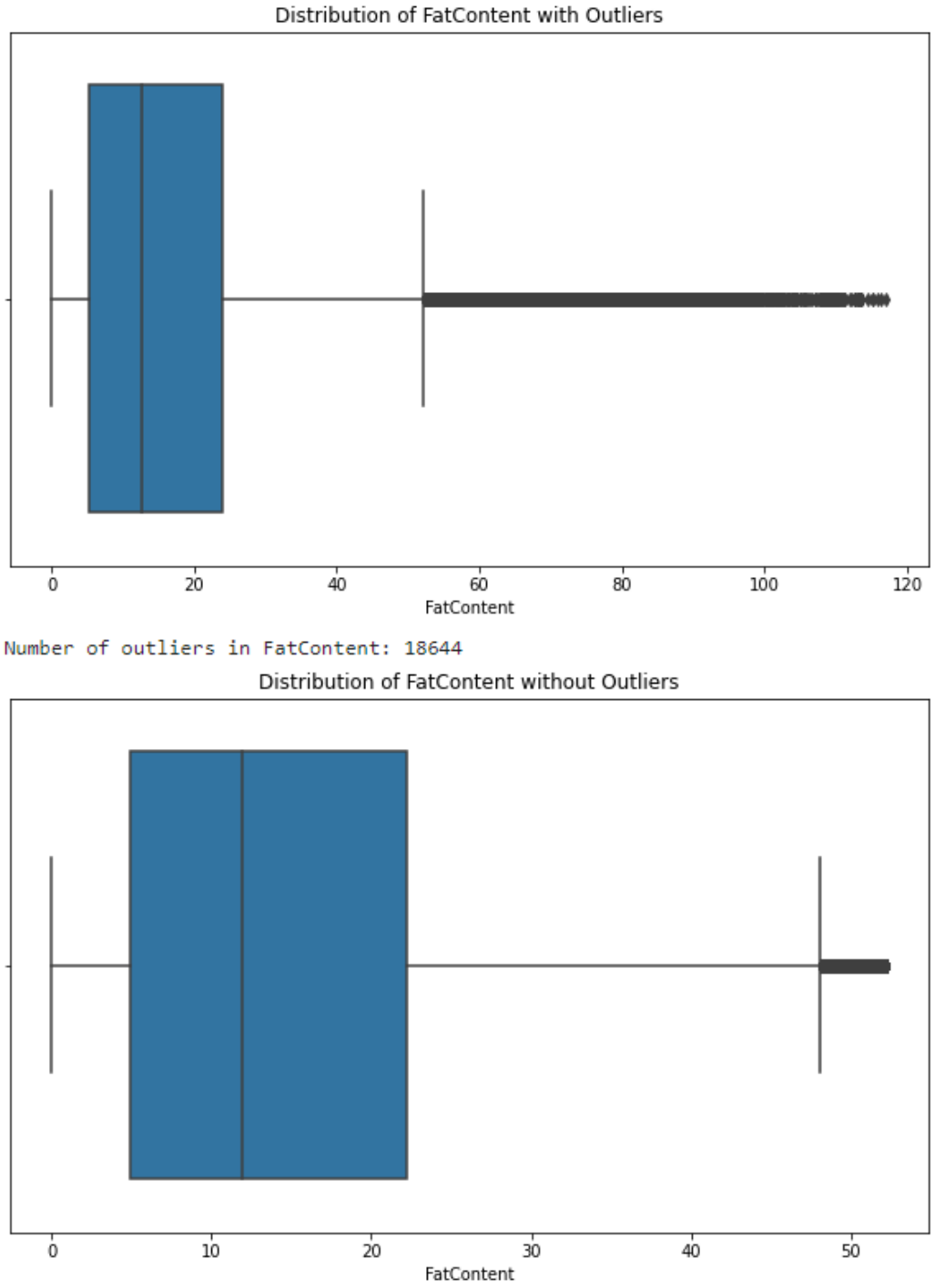
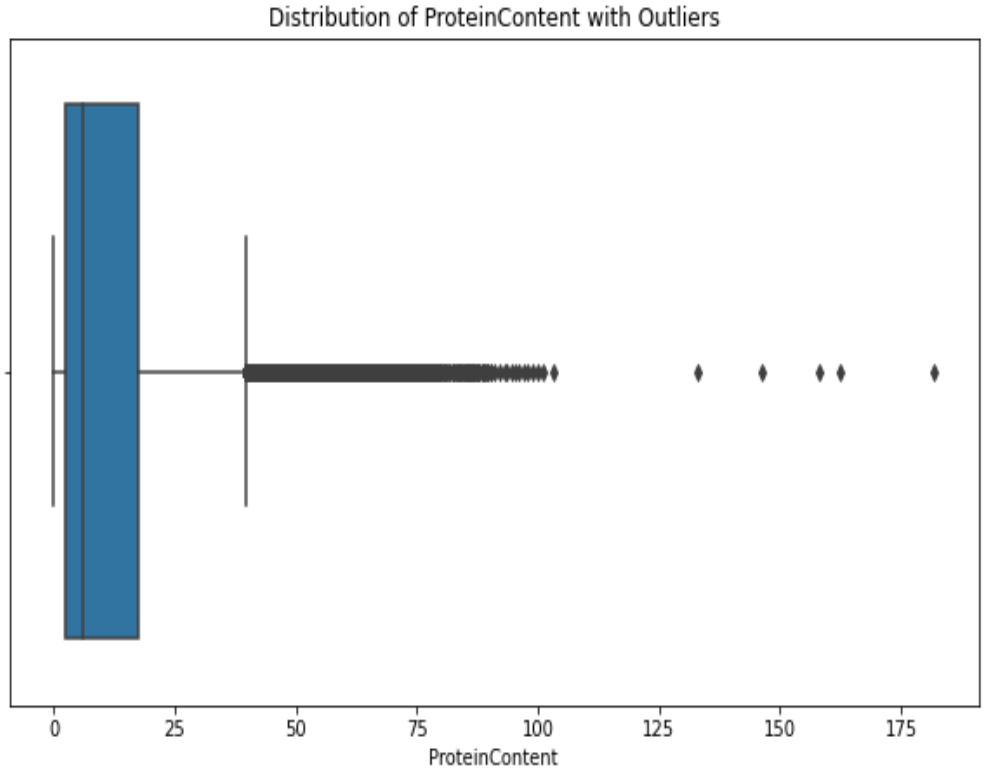


Figure 4.0.2 : Removing outlier




```
Number of outliers in ProteinContent: 13774
```

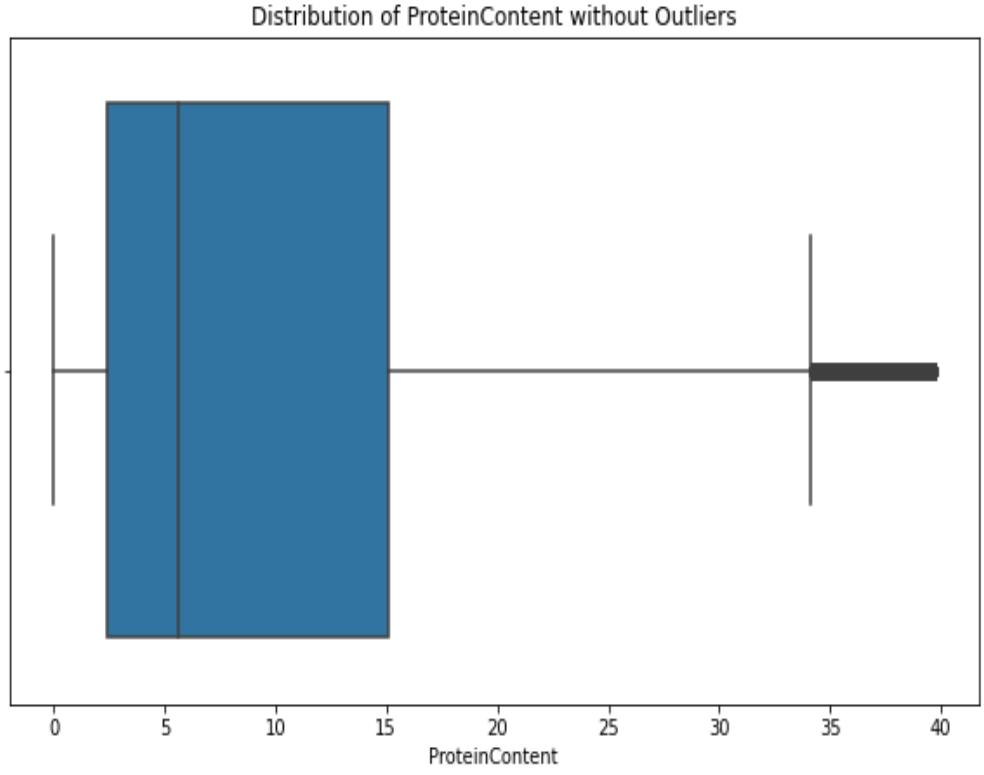


Figure 4.0.2 : Removing Outliers

Model Selection

Proposed Algorithms:

K-Nearest Neighbors (KNN): K-Nearest Neighbors (KNN) is a non-parametric algorithm used for classification and regression tasks. In the context of diet recommendation system, KNN works by identifying the k-nearest neighbors to a given data point based on a similarity measure (e.g., Euclidean distance). In the case of dietary recommendations, it is a simple yet effective algorithm, particularly useful when the underlying data has a clear structure and distinct clusters.

Random Forest (RF): Random Forest (RF) is an ensemble learning method that operates by constructing a multitude of decision trees during training and outputting the mode (classification) or mean prediction (regression) of the individual trees. RF is well suited for handling high-dimensional data and mitigating overfitting. In the context of our diet recommendation system, RF can capture complex relationships between dietary factors and health outcomes, providing robust recommendations based on these relationships.

Decision Tree (DT): Decision Trees (DT) are tree-like structures where each internal node represents a "decision" based on a feature, each branch represents the outcome of that decision, and each leaf node represents the final decision or prediction. DT is a simple and interpretable algorithm that can handle both categorical and numerical data. In the context of diet recommendation, DT can be used to create a hierarchy of decision rules based on nutritional content, dietary preferences, and health objectives to recommend suitable diets.

Linear Regression (LR): Linear Regression involves modeling the relationship between a dependent variable and one or multiple independent variables through a linear approach. It assumes a linear relationship between the variables and aims to find the best fitting line to describe this relationship. In the context of our diet recommendation system, Linear Regression can help identify linear dependencies between dietary factors (e.g., calorie intake, fat content) and health outcomes (e.g., weight loss, cholesterol levels). It provides insights into how changes in dietary habits may impact health, allowing for personalized recommendations aligned with specific health goals.

Training and Testing:

For this Project we are taking four machine learning models which can best serve the purpose of diet recommendation, which includes K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR). Our objective is to determine the most effective model for predicting nutrient values in food items based on their nutritional content. We divided our dataset into training and testing sets, utilizing 70% of the data for training and 30% for testing. The features used for training include Calories, FatContent, SaturatedFatContent, SodiumContent, CholesterolContent, CarbohydrateContent, FiberContent, SugarContent, and ProteinContent. Each model was trained using these features to learn the combinations and patterns in the data, aiming to predict the nearest nutritional values for a given food item. Each model was trained using 70% of the dataset, with the nine features mentioned earlier as inputs. During training, the models learned the relationships between the nutritional features to predict the nearest nutrient values for a given food item. The remaining 30% of the dataset was used for testing the trained models. The models predicted the nutrient values for the test data, which were then compared with the actual values to evaluate the performance of each model. Subsequently, a variety of evaluation metrics including Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R2) are employed to assess the performance of each algorithm by comparing the predicted data by the model against actual data present the data. This meticulous evaluation ensures that our system can accurately predict nutritional content while minimizing errors and optimizing recommendations for individual users. With this approach our diet recommendation system, ensuring reliable and accurate dietary suggestions tailored to each user's unique needs and preferences.

4.1 Workflow

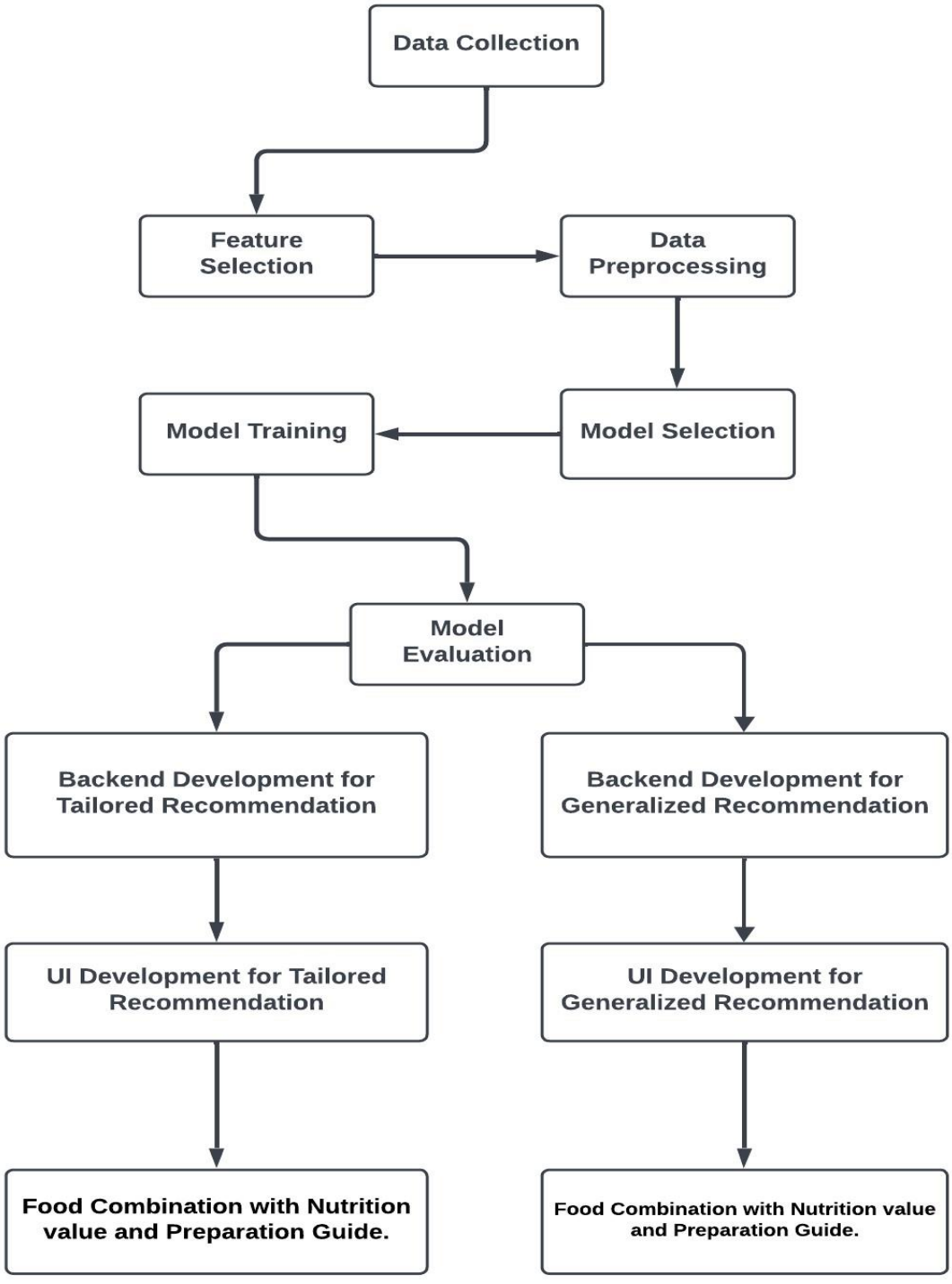


Figure 4.1.1: Workflow

our project is a diet recommendation system that includes both tailored and generalized recommendations. Here's a detailed description of the system's diagram:

Feature Selection: In this step, relevant features are selected for the machine learning models. These features may include various nutritional values, dietary restrictions, and other factors related to food and health. The features are selected based on their relevance to the problem and their potential to improve the performance of the machine learning models.

Data Collection: After selecting the features, data is collected from various sources, such as nutrition databases, health surveys, and user input. The data may include information about different types of food, their nutritional values, and other relevant factors. The data is collected in a structured format that can be used to train the machine learning models.

Data Preprocessing: The collected data is pre-processed to ensure it is in a suitable format for training the models. This may involve cleaning the data, removing outliers, and normalizing the values. Data preprocessing is an essential step in the machine learning pipeline, as it can significantly impact the performance of the models.

Model Training: The pre-processed data is then used to train machine learning models for both tailored and generalized recommendations. The models are trained using various algorithms, such as decision trees, neural networks, or clustering algorithms. The training process involves optimizing the model parameters to minimize the error between the predicted and actual values.

Model Selection: After training the models, the best-performing models are selected based on various evaluation metrics, such as accuracy, precision, and recall. The selection process involves comparing the performance of different models and choosing the one that performs the best on the validation dataset.

Model Evaluation: The selected models are evaluated to ensure they meet the desired performance criteria. This may involve testing the models on a separate dataset or using cross-validation techniques. The evaluation process is essential to ensure that the models are robust and generalize well to new data.

Backend Development for Tailored Recommendation: The backend for the tailored recommendation system is developed to provide personalized dietary recommendations based on individual user data. This may involve integrating the selected machine learning models with a web application or mobile app. The backend development includes implementing the necessary infrastructure to handle user requests, process data, and generate recommendations.

Backend Development for Generalized Recommendation: The backend for the generalized recommendation system is developed to provide dietary recommendations based on general

nutritional guidelines. This may involve integrating the selected machine learning models with a web application or mobile app. The backend development includes implementing the necessary infrastructure to handle user requests, process data, and generate recommendations.

UI Development for Tailored Recommendation: The user interface for the tailored recommendation system is developed to allow users to input their individual data and receive personalized dietary recommendations. The UI development includes implementing the necessary frontend components, such as forms, buttons, and input fields, to allow users to interact with the system.

UI Development for Generalized Recommendation: The user interface for the generalized recommendation system is developed to allow users to access dietary recommendations based on general nutritional guidelines. The UI development includes implementing the necessary frontend components, such as menus, lists, and tables, to display the recommendations to the users.

Food Combination with Nutrition Value and Preparation Guide: The system also includes a guide for food combinations with nutrition values and preparation instructions. This guide can be used to provide additional context and information to users about the recommended diets. The guide may include information about the nutritional values of different food combinations, as well as instructions on how to prepare them. The guide can be implemented as a separate module within the system or integrated into the user interface.

4.2 Architecture Overview:

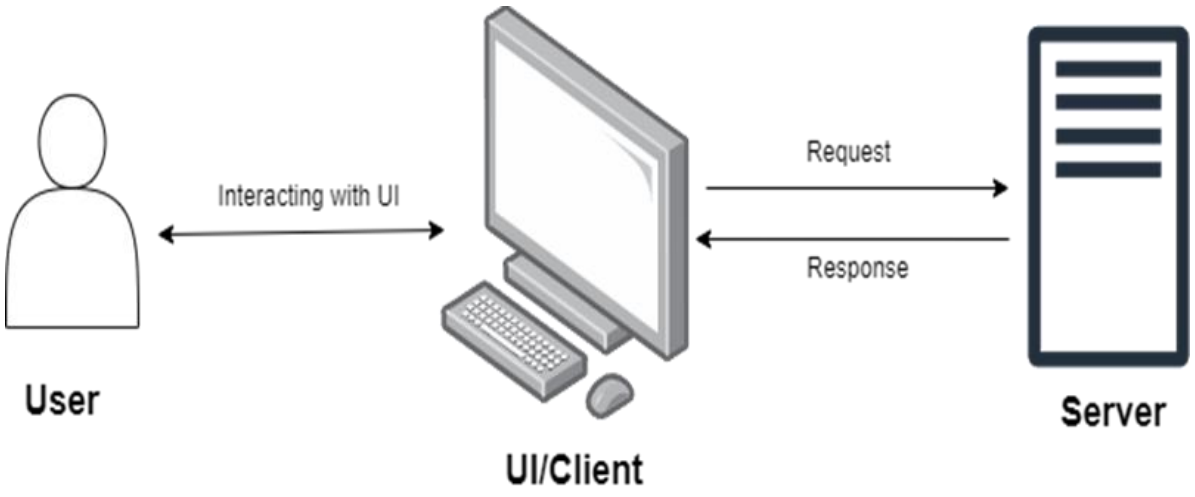


Figure 4.2.1: Client Server architecture

The architecture depicted in Figurer 1 illustrates the client-server model, a commonly employed system in modern computing environments. In this system, users interact with a graphical user interface (UI) to communicate with a server, enabling various functionalities and services.

At the user's end, the UI provides a platform for inputting preferences and pertinent information. Users utilize the UI to tailor their requests and provide context for the server's operations. Upon receiving user inputs, the UI initiates a request to the server, transmitting the gathered information. This request is then processed by the server, which performs a series of operations based on the received data.

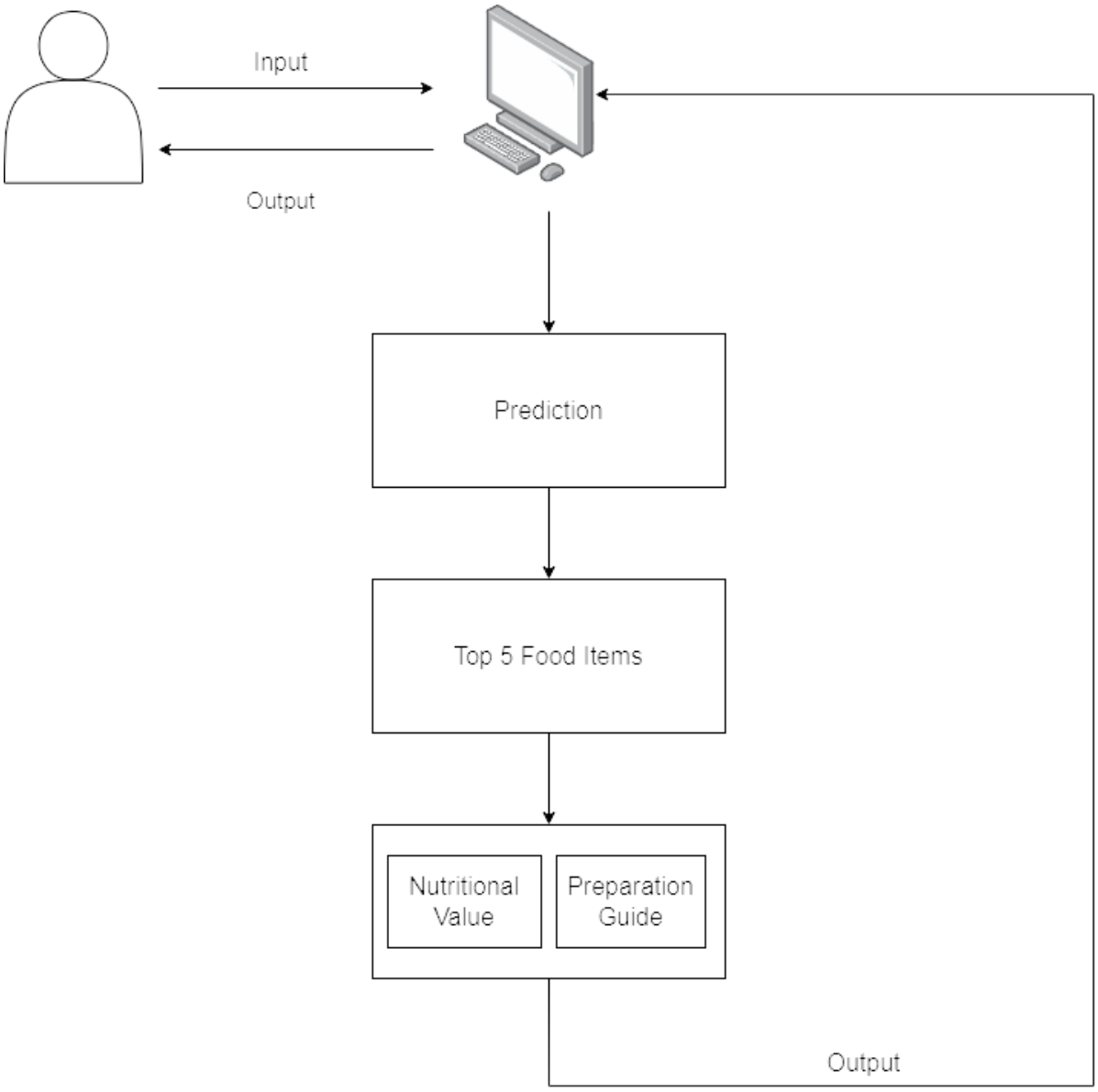


Figure 4.2.2: Tailored Recommendation System

The architecture illustrated in Figure 3 outlines the components and functionalities of a Tailored Diet Recommendation System, designed to assist users in making informed dietary choices based on their nutritional requirements and preferences.

Input:

This section encompasses a list of types of nutritional information that the system can process. These include essential nutritional components such as calories, fat content, saturated fat content, cholesterol content, sodium content, carbohydrate content, fiber content, sugar content, and protein content. Users input their dietary preferences and constraints related to these nutrients to guide the recommendation process.

Prediction:

Based on the input nutrient values provided by the user, the internal diet recommendation system utilizes algorithms and models to make predictions. The system analyses the input data and generates recommendations for food items that align with the user's nutritional requirements. The predicted food items are selected based on the nearest values of the specified nutrients, aiming to provide tailored suggestions that meet the user's dietary goals.

Nutritional Value and Preparation Guide:

In the predicted food items, the system not only presents the nutritional values but also provides guidance on the preparation of the recommended food items. Users can access detailed information on the nutritional composition of each food item, including macro and micronutrient content. Additionally, the system offers instructions and tips on how to prepare the recommended meals, ensuring that users have the necessary information to incorporate them into their diet effectively.

Output:

The output section of the architecture encompasses the recommended food items along with their nutritional value and preparation processes. Users receive a curated list of top 5 food items that best match their dietary requirements and preferences. Each recommended food item is accompanied by comprehensive nutritional information, enabling users to make informed decisions about their diet. Furthermore, the system provides clear and concise instructions on how to prepare each recommended meal, empowering users to implement dietary changes with confidence.

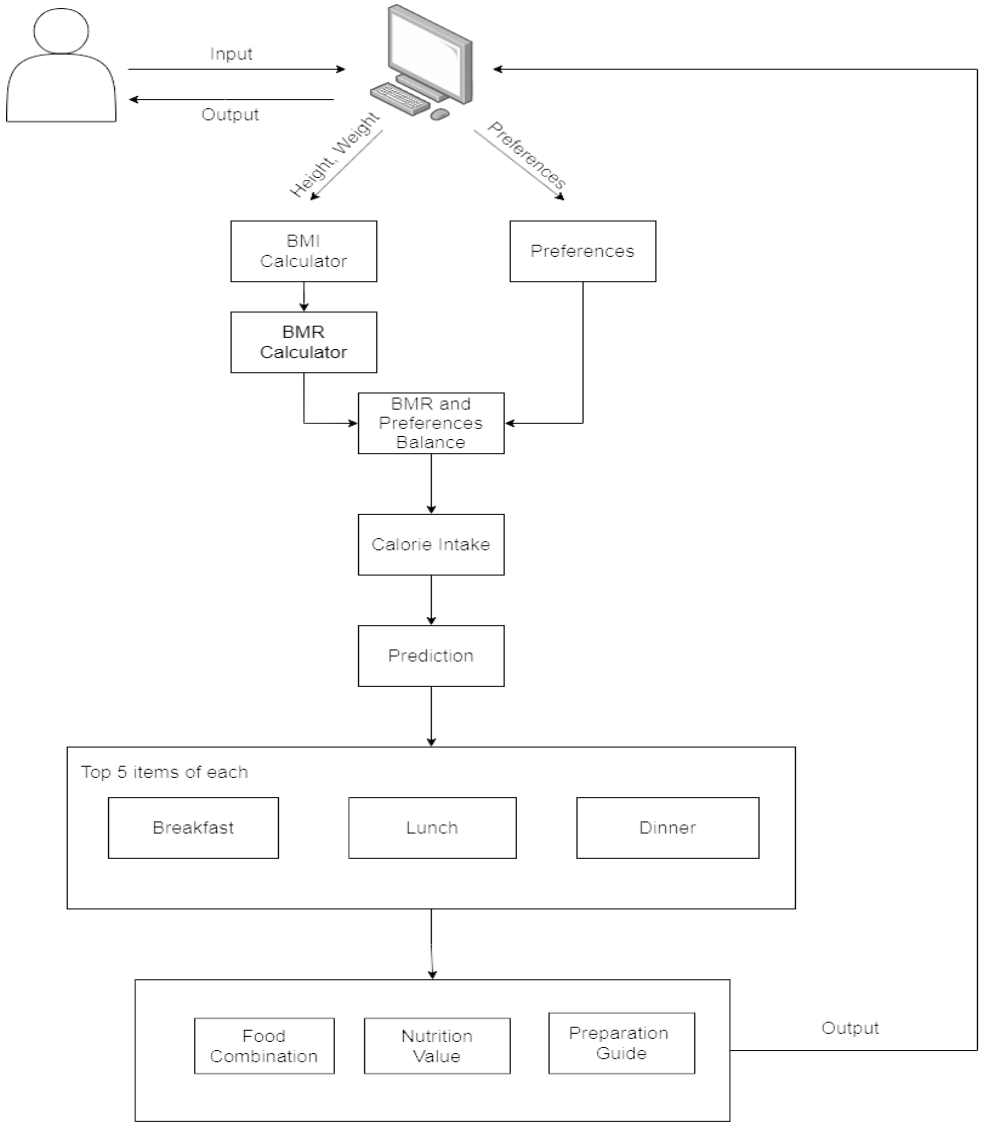


Figure 4.2.3: Generalized Recommendation System

The architecture depicted in Figure 4 represents a generalized diet recommendation system that utilizes various user inputs and preferences to offer personalized recommendations for food combinations and nutrition. Here's a detailed description of each component:

Input:

This section encompasses the user's information such as age, height, weight, gender, activity level, and weight loss goal. These inputs serve as the foundation for the system's calculations and recommendations.

BMI Calculator:

This component calculates the user's Body Mass Index (BMI) based on the provided input. The BMI is a measure of body fat based on height and weight, and it helps determine the user's weight category (underweight, normal weight, overweight, or obese). The BMI calculation guides the system in providing appropriate recommendations tailored to the user's weight status.

$$BMI = \frac{Weight\ in\ Kilogram}{Height\ in\ Meters^2}$$

BMR Calculator:

The Basal Metabolic Rate (BMR) calculator computes the user's BMR, which represents the minimum number of calories required to maintain bodily functions at rest. It takes into account factors such as age, height, weight, gender, and activity level. The BMR calculation forms the basis for determining the user's calorie needs and assists in generating personalized nutrition recommendations.

$$\textbf{MEN} = (10 \times \text{weight [kg]}) + (6.25 \times \text{height [cm]}) - (5 \times \text{age[yrs]}) + 5$$

$$\textbf{WOMEN} = (10 \times \text{weight [kg]}) + (6.25 \times \text{height [cm]}) - (5 \times \text{age[yrs]}) - 161$$

Preferences:

This section includes the user's preferences, such as activity level and weight loss goal. These preferences play a crucial role in tailoring the system's recommendations to align with the user's specific dietary objectives and lifestyle choices.

BMR and Preferences Balance:

The system combines the user's BMR calculation with their preferences, such as weight loss goals, to generate personalized nutrition recommendations. This component ensures that the recommendations are in line with the user's energy requirements and dietary objectives.

Calorie Intake:

Based on the user's BMR and weight loss goal, this section calculates the user's daily calorie intake. This information serves as a guideline for determining the appropriate portion sizes and

food combinations that meet the user's calorie needs while supporting their weight management goals.

Prediction:

This section predicts the user's food item combinations based on their BMR and preferences balance. It utilizes machine learning algorithms to generate recommendations that optimize nutrient intake and align with the user's dietary preferences and goals.

Meal Categories:

The system includes different meal categories (Breakfast, Lunch, and Dinner), allowing users to input the food items they consume for each meal. This feature enables users to track their dietary intake and receive tailored recommendations for each mealtime.

Food Combination with Nutrition and Preparation guide:

This section provides recommendations for food combinations along with detailed nutrition information based on the user's input and preferences. Additionally, it offers guidance on the preparation of recommended food items, ensuring that users have the necessary information to incorporate them into their diet effectively.

Output:

The output section displays the results of the system, including the user's BMI, calorie intake, and recommended food combinations with nutrition information and preparation instructions. This comprehensive output empowers users to make informed dietary decisions and maintain a balanced and healthy diet.

Overall, the generalized diet recommendation system architecture outlined in Figureure 3 aims to leverage user inputs, calculations, and preferences to deliver personalized nutrition recommendations that support the user's health and wellness goals effective tool for managing dietary intake and promoting healthy eating habits.

4.3 Components Design:

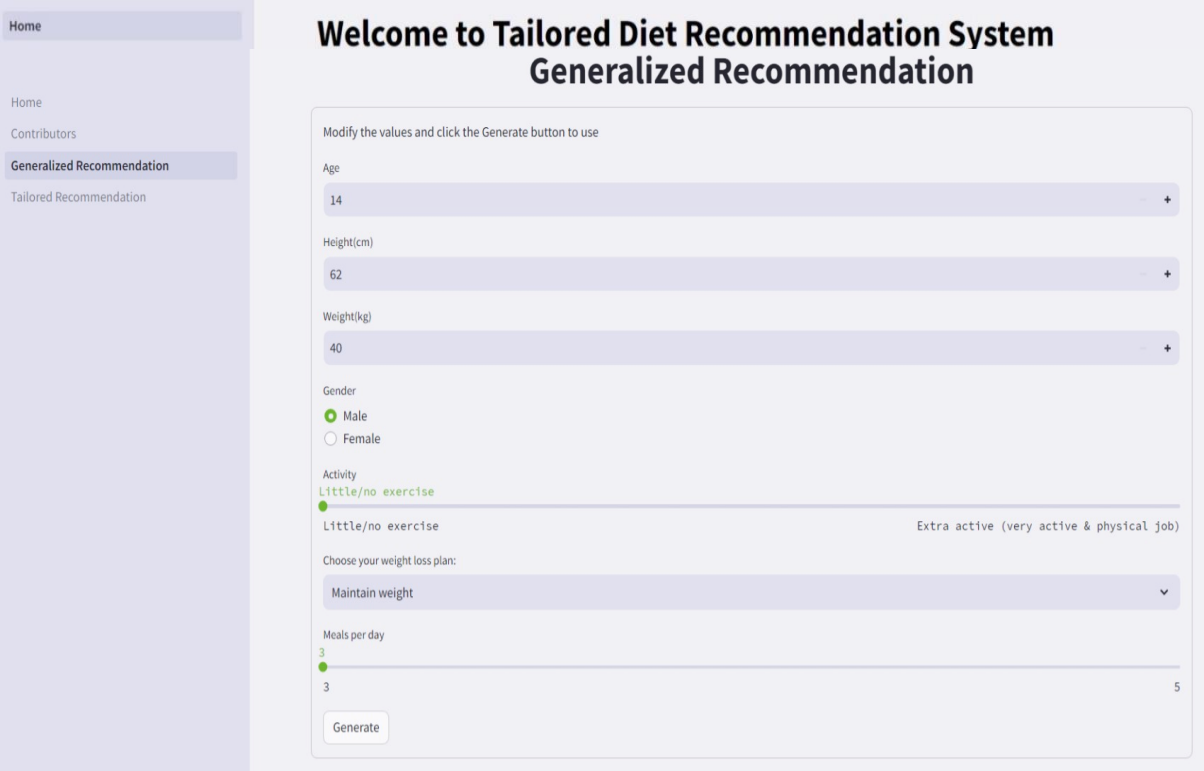


Figure 4.3.1.: Home Page

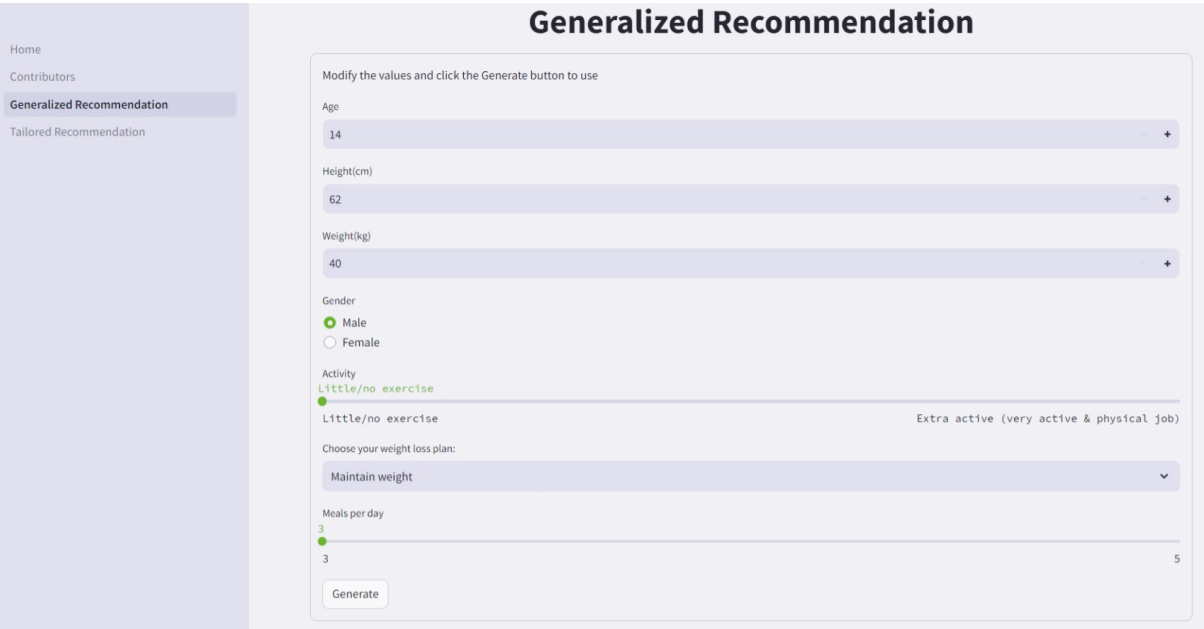


Figure 4.3.2: Generalized Recommendation

BMI CALCULATOR

Body Mass Index (BMI)

24.24 kg/m²

Normal

Healthy BMI range: 18.5 kg/m² - 25 kg/m².

CALORIES CALCULATOR

The results show a number of daily calorie estimates that can be used as a guideline for how many calories to consume each day to maintain, lose, or gain weight at a chosen rate.

Maintain weight	Mild weight loss	Weight loss	Extreme weight loss
1916 Calories/day	1724 Calories/day	1532 Calories/day	1149 Calories/day
↓ -0 kg/week	↓ -0.25 kg/week	↓ -0.5 kg/week	↓ -1 kg/week

DIET RECOMMENDATOR

Recommended recipes:

BREAKFAST	LUNCH	DINNER
Halibut and Bean Stew	Modern Venison Roast	Grilled Tuna With White Bean and Charred Onion Salad
Stove Top Tuna Noodle With Orzo (Reduced Fat)	Venison Soup	Venison Soup
Healthy Spaghetti and Meatballs	Poisson En Papillote, Victorian Style	Venison Black Bean Chili
Mexican Taquitos (Tuna)	Maltese Baked Tuna	Tuna and White Bean Salad
Lomo Saltado (Vegan)	Tuna Potato Spicy Scratch	Tuna and White Bean Salad

Figure 4.3.3: Output of Generalized Recommendation

Figure 4.3.4: Output graph of Generalized Recommendation

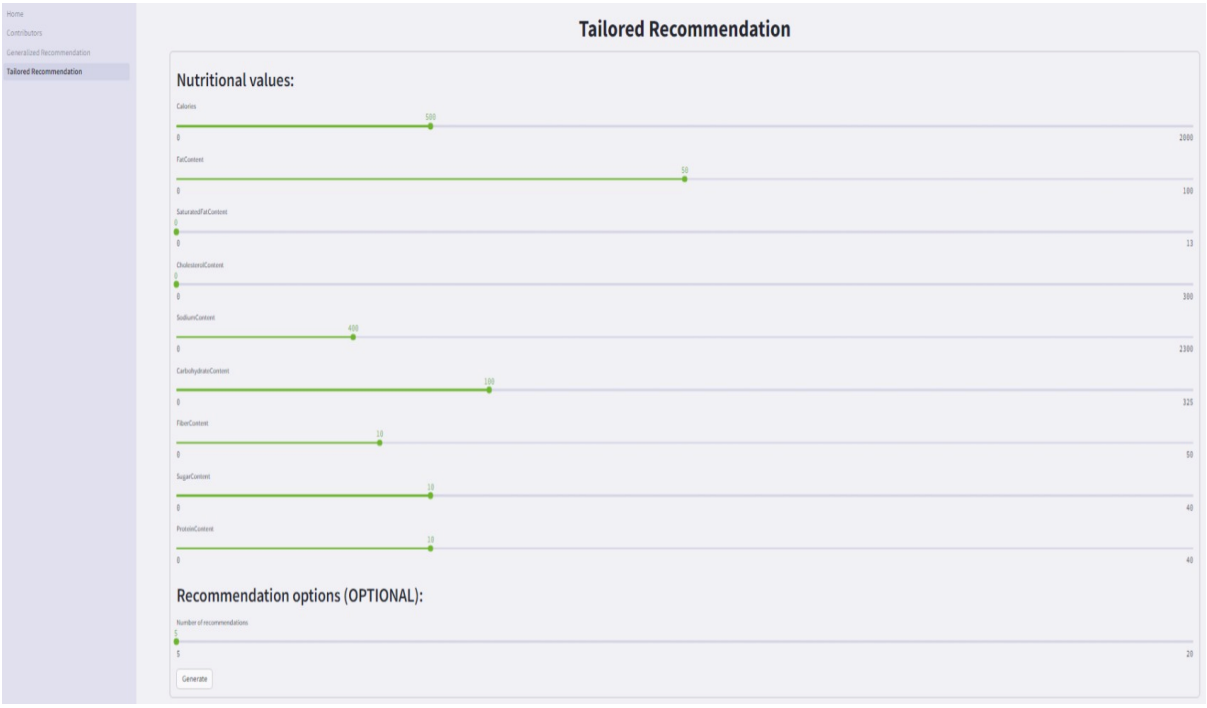


Figure 4.3.5: Tailored Recommendation

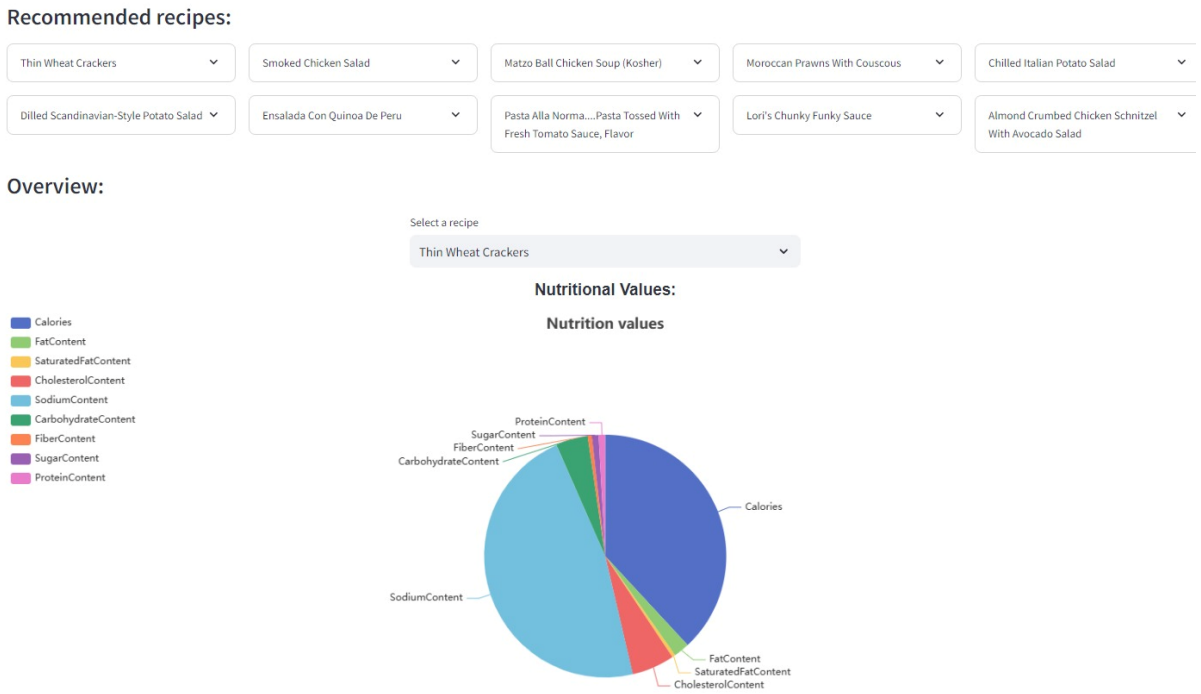


Figure 4.3.6: Output and Graphs of Tailored Recommendation

4.4 Development Environment:

What is Docker?

Docker is currently one of the most widely used containerization platforms. It provides a way to package, distribute, and run applications within containers. Docker enables developers to build, ship, and run applications and services in isolated environments, making it easier to deploy and manage software across different computing environments.

Advantages of Docker:

1. Portability: Docker containers encapsulate everything an application needs to run, including libraries, dependencies, and configure ration files. This makes it easy to deploy applications consistently across different environments, from development to production.

2. Lightweight: Docker containers share the host operating system's kernel and only contain the application and its dependencies, making them lightweight and efficient in terms of resource utilization.

3. Isolation: Docker containers provide process isolation, ensuring that applications running in containers do not interfere with each other. This isolation enhances security and stability, especially in multi-tenant environments.

4. Scalability: Docker's architecture allows applications to scale horizontally by running multiple instances of containers across different hosts. Docker Swarm and Kubernetes are popular tools for orchestrating containerized applications and managing their scalability.

5. Speed: Docker containers start quickly and have low overhead, enabling fast development cycles and efficient resource utilization in production environments.

6. Flexibility: Docker supports a wide range of programming languages and frameworks, making it suitable for building and deploying diverse types of applications, from web services to data processing pipelines.

What is Streamlit?

Streamlit is a Python library used for building interactive web applications and data dashboards with minimal effort. It allows developers and data scientists to create and share data-driven applications quickly, without needing expertise in web development or frontend technologies. Streamlit simplifies the process of turning data scripts into shareable web apps, enabling rapid prototyping, experimentation, and deployment.

Advantages of Streamlit:

- Simplicity
- Rapid Prototyping
- Interactive Visualizations
- Compatibility with Python libraries
- Easy Customization

Installation of Streamlit:

- Prerequisites: Ensure that you have Python installed on your system. Streamlit requires Python version 3.6 or later.Scalability
- Create a new file with (file.py) extension open the terminal and run this command to install streamlit “pip install streamlit”

- In (file.py) file import the streamlit librarie as “import streamlit as st” and using st write “st.title(“Hello world”)”
- Now using terminal run this command to see output “streamlit run file.py”

Advantages of Streamlit Over Other Web Development Frameworks:

1. Pythonic Workflow: Streamlit allows users to write web applications using familiar Python syntax and libraries, eliminating the need to learn new languages or frameworks. This simplifies the development process and reduces cognitive overhead.

2. Interactivity: Streamlit emphasizes interactivity and real-time updates, enabling users to build dynamic and responsive applications that engage users and facilitate exploration of data insights.

3. Focus on Data Science: Streamlit is specifically designed for data scientists and machine learning practitioners, providing tailored features and integrations for data visualization, model deployment, and experimentation.

4. Streamlined Development: Streamlit's high-level API abstracts away many of the complexities of web development, enabling users to focus on their data analysis and application logic rather than boilerplate code or infrastructure setup.

What is Python:

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and the indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard libraries which can be used for the following :

• GUI Applications (like Kivy, Tkinter, PyQt etc.)

• Web frameworks like Django (used by YouTube, Instagram, and Dropbox)

• Image processing (like Opencv, Pillow)

• Web scraping (like Scrapy, BeautifulSoup, and Selenium)

• Test frameworks

• Multimedia

Advantages of Python:

Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and indentation is mandatory. These further aids the readability of the code.

Disadvantages of Python:

- Speed Limitations
- Weak in Mobile Computing and Browsers
- Design Restrictions

CHAPTER-5

CODE

5. CODE

5.1 Tailored Recommendation:

Input: **get_user_input_tailored()**:

The module **get_user_input_tailored()** is used to take the input from the user to recommend the food items. The user input includes calories, fat content, saturated fat content, cholesterol content, sodium content, carbohydrate content, fiber content, sugar content, and protein content. Based on these input data further process and recommendation will be carried

```
def get_user_input_tailored():
    with st.form("recommendation_form"):
        st.header('Nutritional values:')
        Calories = st.slider('Calories', 0, 2000, 500)
        FatContent = st.slider('FatContent', 0, 100, 50)
        SaturatedFatContent = st.slider('SaturatedFatContent', 0, 13, 0)
        CholesterolContent = st.slider('CholesterolContent', 0, 300, 0)
        SodiumContent = st.slider('SodiumContent', 0, 2300, 400)
        CarbohydrateContent = st.slider('CarbohydrateContent', 0, 325, 100)
        FiberContent = st.slider('FiberContent', 0, 50, 10)
        SugarContent = st.slider('SugarContent', 0, 40, 10)
        ProteinContent = st.slider('ProteinContent', 0, 40, 10)
        nutritions_values_list=[Calories,FatContent,SaturatedFatContent,CholesterolContent,SodiumContent,CarbohydrateContent,FiberContent,SugarContent,ProteinContent]
        st.header('Recommendation options (OPTIONAL):')
        nb_recommendations = st.slider('Number of recommendations', 5, 20,step=5)
        generated = st.form_submit_button("Generate")
    if generated:
        with st.spinner('Generating recommendations...'):
            recommendation=Recommendation(nutritions_values_list,nb_recommendations)
            recommendations=recommendation.generate()
            st.session_state.recommendations=recommendations
        st.session_state.generated=True

    if st.session_state.generated:
        with st.container():
            display.display_recommendation(st.session_state.recommendations)
        with st.container():
            display.display_overview(st.session_state.recommendations)
    logging.info("Input in Tailored and forward success")
get_user_input_tailored()
```

Figure 5.1.1: Input for Tailored Recommendation

Prediction: **Generator.generate()**

The generate method in the Generator class constructs a request with user input. It sends this request to a backend server for food recommendation prediction. The server responds with the generated recommendation. Finally, the method returns this recommendation. Overall, it acts as a bridge between user input and food recommendations.

```
def generate(self,):
    request={
        'nutrition_input':self.nutrition_input,
        'ingredients':self.ingredients,
        'params':self.params
    }
    response=requests.post(url='http://backend:8080/predict/',data=json.dumps(request))
    return response
```

Figure 5.1.2 : Generate method

Top 5 food items : **Display.display_recommendation()**

Display class defines a method display_recommendation to exhibit recipe recommendations using Streamlit. It iterates through recommendations, organizing them into columns with recipe details including images, nutrition values, ingredients, instructions, and cooking times. It handles cases where no recommendations are found, displaying a corresponding message. The method includes exception handling to log any encountered errors.

```
class Display:
    def __init__(self):
        self.nutrition_values=nutrition_values

    def display_recommendation(self,recommendations):
        try:
            st.subheader('Recommended recipes:')
            if recommendations!=None:
                rows=len(recommendations)//5
                for column,row in zip(st.columns(5),range(5)):
                    with column:
                        for recipe in recommendations[rows*row:rows*(row+1)]:
                            recipe_name=recipe['Name']
                            expander = st.expander(recipe_name)
                            recipe_link=recipe['image_link']
                            recipe_img=f'<div><center><img src={recipe_link} alt={recipe_name}></center></div>'
                            nutritions_df=pd.DataFrame({value:[recipe[value]] for value in nutrition_values})

                            expander.markdown(recipe_img,unsafe_allow_html=True)
                            expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Nutritional Values (g):</h5>', unsafe_allow_html=True)
                            expander.dataframe(nutritions_df)
                            expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Ingredients:</h5>', unsafe_allow_html=True)
                            for ingredient in recipe['RecipeIngredientParts']:
                                expander.markdown(f"""
                                            - {ingredient}
                                """)
                            expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Recipe Instructions:</h5>', unsafe_allow_html=True)
                            for instruction in recipe['RecipeInstructions']:
                                expander.markdown(f"""
                                            - {instruction}
                                """)
                            expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Cooking and Preparation Time:</h5>', unsafe_allow_html=True)
                            expander.markdown(f"""
                                    - Cook Time       : {recipe['CookTime']}min
                                    - Preparation Time: {recipe['PrepTime']}min
                                    - Total Time      : {recipe['TotalTime']}min
                                """)
            else:
                st.info('Couldn\'t find any recipes with the specified ingredients', icon="😩")
        except Exception as e:
            logging.info(f"Something went wrong: {e}")
```

Figure 5.1.3 : display_recommendation function

Nutritional value and Preparation guide: **Display.display_overview()**

display_overview() is a method present in class Display which is used to display the nutrition data with a visually appealing plot, along with preparation guide to cook the food items along with ingredients.

```
def display_overview(self,recommendations):
    try:
        if recommendations!=None:
            st.subheader('Overview:')
            col1,col2,col3=st.columns(3)
            with col2:
                selected_recipe_name=st.selectbox('Select a recipe',[recipe['Name'] for recipe in recommendations])
            st.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Nutritional Values:</h5>', unsafe_allow_html=True)
            for recipe in recommendations:
                if recipe['Name']==selected_recipe_name:
                    selected_recipe=recipe
            options = {
        "title": {"text": "Nutrition values", "subtext": " ", "left": "center"},
        "tooltip": {"trigger": "item"},
        "legend": {"orient": "vertical", "left": "left",},
        "series": [
            {
                "name": "Nutrition values",
                "type": "pie",
                "radius": "50%",
                "data": [{"value":selected_recipe[nutrition_value],"name":nutrition_value} for nutrition_value in self.nutrition_values],
                "emphasis": {
                    "itemStyle": {
                        "shadowBlur": 10,
                        "shadowOffsetX": 0,
                        "shadowColor": "rgba(0, 0, 0, 0.5)",
                    }
                },
            }
        ],
    }
            st_echarts(options=options, height="600px",)
            st.caption('You can select/deselect an item (nutrition value) from the legend.')
    except Exception as e:
        logging.info(f"Something went wrong: {e}")
```

Figure 5.1.4: display_overview function

5.2 Generalized Recommendation:

Input, BMI calculator, BMR calculator, Calorie intake, Prediction, Top 5 food items of each meal, Food combination, nutrition value and preparation guide**.**

Input: **get_user_input_generalized()**

get_user_input_generalized() is focused to take input from the user to recommend generalized diet, the input includes age, height, weight, gender, and activity level and weight

maintenance goals. Based on these input values, the further process towards recommendation will be carried.

```
def get_user_input_generalized():
    with st.form("recommendation_form"):
        st.write("Modify the values and click the Generate button to use")
        age = st.number_input('Age',min_value=14, max_value=120, step=1)
        height = st.number_input('Height(cm)',min_value=62, max_value=300, step=1)
        weight = st.number_input('Weight(kg)',min_value=40, max_value=300, step=1)
        gender = st.radio('Gender',('Male','Female'))
        activity = st.select_slider('Activity',options=['Little/no exercise', 'Light exercise', 'Moderate exercise (3-5 days/wk)', 'Very active (6-7 days/wk)',
        'Extra active (very active & physical job)'])
        option = st.selectbox('Choose your weight loss plan:',display.plans)
        st.session_state.weight_loss_option=option
        weight_loss=display.weights[display.plans.index(option)]
        number_of_meals=st.slider('Meals per day',min_value=3,max_value=5,step=1,value=3)
        if number_of_meals==3:
            meals_calories_perc={'breakfast':0.35,'lunch':0.40,'dinner':0.25}
        elif number_of_meals==4:
            meals_calories_perc={'breakfast':0.30,'morning snack':0.05,'lunch':0.40,'dinner':0.25}
        else:
            meals_calories_perc={'breakfast':0.30,'morning snack':0.05,'lunch':0.40,'afternoon snack':0.05,'dinner':0.20}
        generated = st.form_submit_button("Generate")

    if generated:
        if age < 13:
            st.warning("Sorry, this tool is only intended for users aged 13 and above.")
        else:
            st.session_state.generated=True
            person = Person(age,height,weight,gender,activity,meals_calories_perc,weight_loss)
            with st.container():
                display.display_bmi(person)
            with st.container():
                display.display_calories(person)
            with st.spinner('Generating recommendations...'):
                recommendations=person.generate_recommendations()
                st.session_state.recommendations=recommendations
                st.session_state.person=person

    if st.session_state.generated:
        with st.container():
            display.display_recommendation(st.session_state.person,st.session_state.recommendations)
            st.success('Recommendation Generated Successfully !', icon="✅")
        with st.container():
            display.display_meal_choices(st.session_state.person,st.session_state.recommendations)
    logging.info("Input in Generalized and forward success")
get_user_input_generalized()
```

Figure 5.2.1: Input of Generalized Recommendation

BMI calculator: **Person.calculate_bmi()**

calculate_bmi() method present in the class Person, which will be used to calculate Body Mass Index (BMI), to identify the weight category.

```
def calculate_bmi(self):
    bmi=round(self.weight/((self.height/100)**2),2)
    return bmi

def display_result(self):
    bmi=self.calculate_bmi()
    bmi_string=f'{bmi} kg/m²'
    if bmi<18.5:
        category='Underweight'
        color='Red'
    elif 18.5<=bmi<25:
        category='Normal'
        color='Green'
    elif 25<=bmi<30:
        category='Overweight'
        color='Yellow'
    else:
        category='Obesity'
        color='Red'
    return bmi_string,category,color
```

Figure 5.2.2: BMI Results

BMR calculator: **Person.calculate_bmr()**

calculate_bmr() method present in the class Person, which will be used to calculate the BMR (Basal Metabolic Rate), indicate the number of calories a person burns at rest.

```
def calculate_bmr(self):
    if self.gender=='Male':
        bmr=10*self.weight+6.25*self.height-5*self.age+5
    else:
        bmr=10*self.weight+6.25*self.height-5*self.age-161
    return bmr
```

Figure 5.2.3: BMR calculator

Calorie intake: **Person.calories_calculator()**

calories_calculator() method present in the class Person, designed to calculate the calorie intake of a person based on BMR, activity level and weght maintenance goals.

```
def calories_calculator(self):
    activites=['Little/no exercise', 'Light exercise', 'Moderate exercise (3-5 days/wk)', 'Very active (6-7 days/wk)', 'Extra active (very active & physical job)']
    weights=[1.2,1.375,1.55,1.725,1.9]
    weight = weights[activites.index(self.activity)]
    maintain_calories = self.calculate_bmr()*weight
    return maintain_calories
```

Figure 5.2.4: Person calorie calculator

Top 5 food items of each meal: **Display.display_recommendation()**

display_recommendation() is a method present in Display class to display the food items recommended.

```
def display_recommendation(self,person,recommendations):
    try:
        st.header('DIET RECOMMENDATOR')
        with st.spinner('Generating recommendations...'):
            meals=person.meals_calories_perc
            st.subheader('Recommended recipes:')
            for meal_name,column,recommendation in zip(meals,st.columns(len(meals)),recommendations):
                with column:
                    st.markdown(f'##### {meal_name.upper()}')
                    for recipe in recommendation:
                        recipe_name=recipe['Name']
                        expander = st.expander(recipe_name)
                        recipe_link=recipe['image_link']
                        recipe_img=f'<div><center><img src={recipe_link} alt={recipe_name}></center></div>'
                        nutritions_df=pd.DataFrame({value:[recipe[value]] for value in nutritions_values})

                        expander.markdown(recipe_img,unsafe_allow_html=True)
                        expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Nutritional Values (g):</h5>', unsafe_allow_html=True)
                        expander.dataframe(nutritions_df)
                        expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Ingredients:</h5>', unsafe_allow_html=True)
                        for ingredient in recipe['RecipeIngredientParts']:
                            expander.markdown(f"""
                                        - {ingredient}
                            """)
                        expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Recipe Instructions:</h5>', unsafe_allow_html=True)
                        for instruction in recipe['RecipeInstructions']:
                            expander.markdown(f"""
                                        - {instruction}
                            """)
                        expander.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Cooking and Preparation Time:</h5>', unsafe_allow_html=True)
                        expander.markdown(f"""
                                - Cook Time       : {recipe['CookTime']}min
                                - Preparation Time: {recipe['PrepTime']}min
                                - Total Time      : {recipe['TotalTime']}min
                            """)
    except Exception as e:
        logging.info(f"something went wrong {e}")
```

Figure 5.2.5: Recommendation display for Generalized Recommendation

Food combination, nutrition value and preparation guide**: Display.display_meal_choices()**

Display.display_meal_choices() :

This code segment is part of a project for recommending meal choices based on a person's nutritional requirements and preferences. The **display_meal_choices()** method presents options for breakfast, lunch, and dinner to the user using Streamlit. It dynamically adjusts the number of columns based on the number of meal options available. The user selects their preferred meals, and the method calculates the total nutritional values, focusing on calories. It computes the calorie deficit for weight loss based on the user's profile. This feature enhances user engagement by offering personalized meal suggestions tailored to their dietary needs and weight management goals.

```
def display_meal_choices(self,person,recommendations):
    try:
        st.subheader('Choose your meal composition:')
        if len(recommendations)==3:
            breakfast_column,launch_column,dinner_column=st.columns(3)
            with breakfast_column:
                breakfast_choice=st.selectbox(f'Choose your breakfast:',[recipe['Name'] for recipe in recommendations[0]])
            with launch_column:
                launch_choice=st.selectbox(f'Choose your launch:',[recipe['Name'] for recipe in recommendations[1]])
            with dinner_column:
                dinner_choice=st.selectbox(f'Choose your dinner:',[recipe['Name'] for recipe in recommendations[2]])
            choices=[breakfast_choice,launch_choice,dinner_choice]
        elif len(recommendations)==4:
            breakfast_column,morning_snack,launch_column,dinner_column=st.columns(4)
            with breakfast_column:
                breakfast_choice=st.selectbox(f'Choose your breakfast:',[recipe['Name'] for recipe in recommendations[0]])
            with morning_snack:
                morning_snack=st.selectbox(f'Choose your morning_snack:',[recipe['Name'] for recipe in recommendations[1]])
            with launch_column:
                launch_choice=st.selectbox(f'Choose your launch:',[recipe['Name'] for recipe in recommendations[2]])
            with dinner_column:
                dinner_choice=st.selectbox(f'Choose your dinner:',[recipe['Name'] for recipe in recommendations[3]])
            choices=[breakfast_choice,morning_snack,launch_choice,dinner_choice]
        else:
            breakfast_column,morning_snack,launch_column,afternoon_snack,dinner_column=st.columns(5)
            with breakfast_column:
                breakfast_choice=st.selectbox(f'Choose your breakfast:',[recipe['Name'] for recipe in recommendations[0]])
            with morning_snack:
                morning_snack=st.selectbox(f'Choose your morning_snack:',[recipe['Name'] for recipe in recommendations[1]])
            with launch_column:
                launch_choice=st.selectbox(f'Choose your launch:',[recipe['Name'] for recipe in recommendations[2]])
            with afternoon_snack:
                afternoon_snack=st.selectbox(f'Choose your afternoon:',[recipe['Name'] for recipe in recommendations[3]])
            with dinner_column:
                dinner_choice=st.selectbox(f'Choose your  dinner:',[recipe['Name'] for recipe in recommendations[4]])
            choices=[breakfast_choice,morning_snack,launch_choice,afternoon_snack,dinner_choice]


        total_nutrition_values={nutrition_value:0 for nutrition_value in nutritions_values}
        for choice,meals_ in zip(choices,recommendations):
            for meal in meals_:
                if meal['Name']==choice:
                    for nutrition_value in nutritions_values:
                        total_nutrition_values[nutrition_value]+=meal[nutrition_value]
```

Figure 5.2.6: Meal choice display

```
total_calories_chose=total_nutrition_values['Calories']
loss_calories_chose=round(person.calories_calculator()*person.weight_loss)

st.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Total Calories in Recipes vs {st.session_state.weight_loss_option} Calories:</h5>', unsafe_allow_html=True)
total_calories_graph_options = {
    "xAxis": {
        "type": "category",
        "data": ['Total Calories you chose', f"{st.session_state.weight_loss_option} Calories"],
    },
    "yAxis": {"type": "value"},
    "series": [
        {
            "data": [
                {"value":total_calories_chose, "itemStyle": {"color":["#33FF8D","#FF3333"][total_calories_chose>loss_calories_chose]}},
                {"value": loss_calories_chose, "itemStyle": {"color": "#3339FF"}},
            ],
            "type": "bar",
        }
    ],
}
st_echarts(options=total_calories_graph_options,height="400px",)
st.markdown(f'<h5 style="text-align: center;font-family:sans-serif;">Nutritional Values:</h5>', unsafe_allow_html=True)
nutritions_graph_options = {
    "tooltip": {"trigger": "item"},
    "legend": {"top": "5%", "left": "center"},
    "series": [
        {
            "name": "Nutritional Values",
            "type": "pie",
            "radius": ["40%", "70%"],
            "avoidLabelOverlap": False,
            "itemStyle": {
                "borderRadius": 10,
                "borderColor": "#fff",
                "borderWidth": 2,
            },
            "label": {"show": False, "position": "center"},
            "emphasis": {
                "label": {"show": True, "fontSize": "40", "fontWeight": "bold"}
            },
            "labelLine": {"show": False},
            "data": [{"value":round(total_nutrition_values[total_nutrition_value]),"name":total_nutrition_value} for total_nutrition_value in total_nutrition_values],
        }
    ],
}
st_echarts(options=nutritions_graph_options, height="500px",)
```

Figure 5.2.7 : Meal choice display

5.3 Backend:

This code snippet presents a recommendation system for recipes based on user input and ingredient preferences. It consists of several functions:

scaling: Utilizes StandardScaler to scale the input data.

nn_predictor: Constructs a nearest neighbors model (NearestNeighbors) using cosine distance.

build_pipeline: Constructs a data processing pipeline incorporating the scaler and nearest neighbors model.

extract_data: Filters the dataset based on user-provided ingredients.

extract_ingredient_filtered_data: Performs ingredient-based filtering on the dataset.

apply_pipeline: Applies the constructed pipeline to the input data.

recommend: The main recommendation function orchestrating the entire process. It filters the dataset based on provided ingredients, scales the data, builds a nearest neighbors model, creates a pipeline, and applies it to the input data to recommend similar recipes.

```
def scaling(dataframe):
    scaler=StandardScaler()
    prep_data=scaler.fit_transform(dataframe.iloc[:,6:15].to_numpy())
    return prep_data,scaler

def nn_predictor(prep_data):
    neigh = NearestNeighbors(metric='cosine',algorithm='brute')
    neigh.fit(prep_data)
    return neigh

def build_pipeline(neigh,scaler,params):
    transformer = FunctionTransformer(neigh.kneighbors,kw_args=params)
    pipeline=Pipeline([('std_scaler',scaler),('NN',transformer)])
    return pipeline

def extract_data(dataframe,ingredients):
    extracted_data=dataframe.copy()
    extracted_data=extract_ingredient_filtered_data(extracted_data,ingredients)
    return extracted_data

def extract_ingredient_filtered_data(dataframe,ingredients):
    extracted_data=dataframe.copy()
    regex_string=''.join(map(lambda x:f'(?=.*{x})',ingredients))
    extracted_data=extracted_data[extracted_data['RecipeIngredientParts'].str.contains(regex_string,regex=True,flags=re.IGNORECASE)]
    return extracted_data

def apply_pipeline(pipeline,_input,extracted_data):
    _input=np.array(_input).reshape(1,-1)
    return extracted_data.iloc[pipeline.transform(_input)[0]]

def recommend(dataframe,_input,ingredients=[],params={'n_neighbors':5,'return_distance':False}):
        extracted_data=extract_data(dataframe,ingredients)
        if extracted_data.shape[0]>=params['n_neighbors']:
            prep_data,scaler=scaling(extracted_data)
            neigh=nn_predictor(prep_data)
            pipeline=build_pipeline(neigh,scaler,params)
            return apply_pipeline(pipeline,_input,extracted_data)
        else:
            return None
```

Figure 5.3.1 : Backend

CHAPTER-6

RESULTS AND ANALYSIS

6. RESULTS AND ANALYSIS

6.1 Evaluation Metrics:

Mean Absolute Error (MAE):

Mean Absolute Error (MAE) is a metric used to evaluate the performance of a regression model. It calculates the average of the absolute differences between the predicted values and the actual values. The absolute difference represents the magnitude of the error without considering its direction, making MAE a robust measure of model accuracy. A lower MAE indicates that the model's predictions are closer to the actual values, implying better performance shown

$$\mathbf{MAE} = \frac{1}{\mathrm{N}} \sum_{i=1}^{N} | \mathrm{Y} - \widehat{\mathrm{Y}} |$$

Mean Squared Error (MSE):

Mean Squared Error (MSE) is another widely used metric for assessing regression model performance. It computes the average of the squared differences between the predicted values and the actual values. Squaring the errors penalizes larger deviations more heavily, making MSE sensitive to outliers. Like MAE, a lower MSE signifies better model performance. However, because MSE squares the errors, its values tend to be larger than those of MAE.

$$\mathbf{MSE} = \frac{1}{\mathrm{N}} \sum_{i=1}^{N} (Y - \hat{Y})^2$$

Root Mean Squared Error (RMSE):

Root Mean Squared Error (RMSE) is derived from MSE by taking the square root of the average squared differences between the predicted values and the actual values. RMSE shares the same unit of measurement as the target variable, making it easier to interpret. It provides a measure of the average magnitude of errors in the same units as the target variable, allowing for a more intuitive understanding of the model's performance. RMSE is particularly useful for comparing models with different scales of target variables.

$$\mathbf{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y - \hat{Y})^2}$$

4. R-squared (R2) Score:

R-squared (R2) score is a statistical measure that represents the proportion of the variance in the target variable that is explained by the regression model. It ranges from 0 to 1, where 1 indicates a perfect fit and 0 indicates that the model does not explain any variance in the target variable. R2 score provides insights into how well the regression model captures the variability in the data. Higher R2 scores indicate that the model explains a larger proportion of the variance and thus, represents a better fit to the data.

$$\mathbf{R^2} = \frac{\sum_{i=1}^{N}(Y - \widehat{Y})^2}{\sum_{i=1}^{N}(Y - \bar{Y})^2}$$

6.2 Test Results:

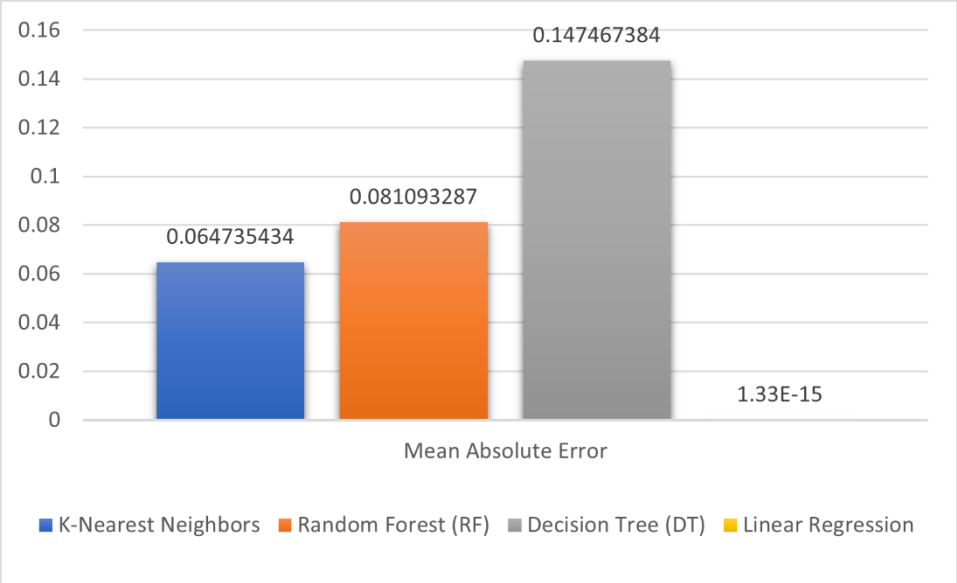


Figure 6.2.1: Mean Absolute Error Graph

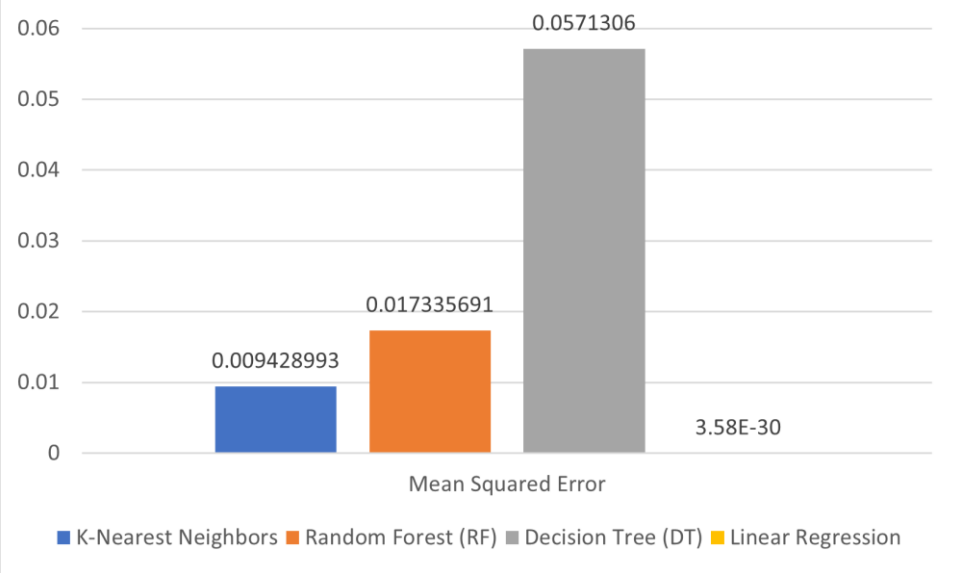


Figure 6.2.2: Mean Squared Error Graph

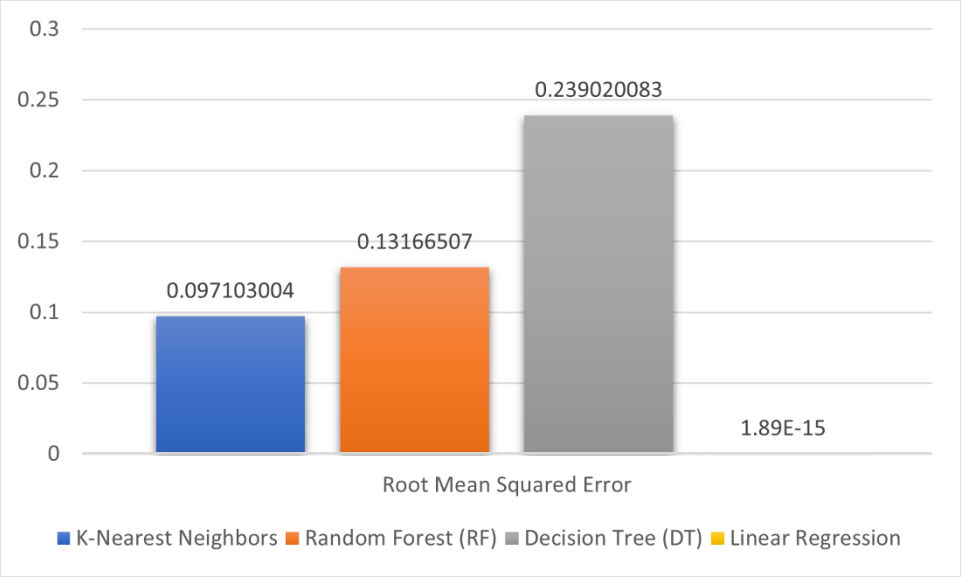


Figure 6.2.3: Root Mean Squared Error Graph

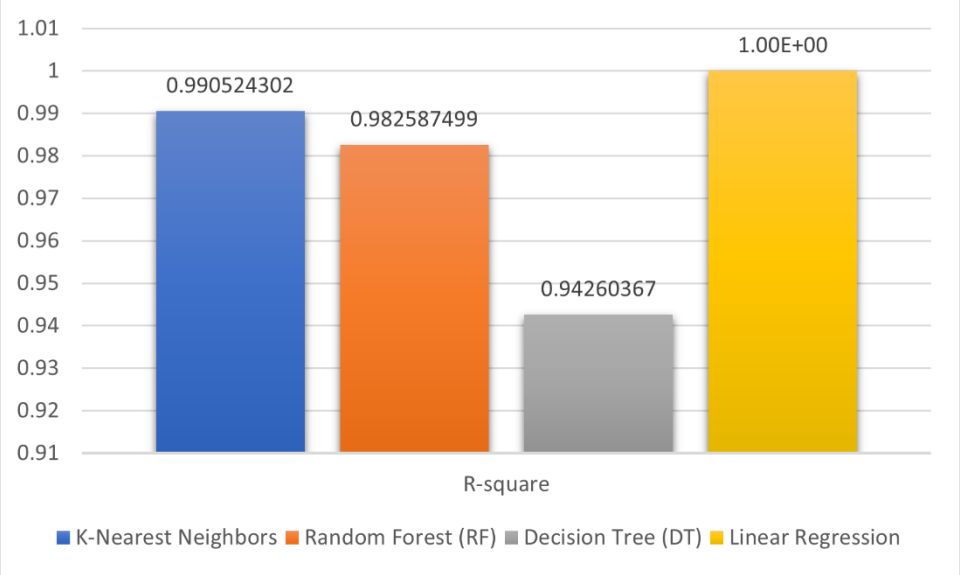


Figure 6.2.4: R-squared Error(R2) Graph

Based on the test results with different performance metrics including Mean Absolute Error (MAE) shown in Figure 6.2.1, Mean Squared Error (MSE) shown in Figure 6.2.2, Root Mean Squared Error (RMSE) shown in Figure 6.2.3 and R-squared (R2) shown in Figure 6.2.4 score for different machine learning algorithms including K-Nearest Neighbors (KNN), Random Forest (RF), Decision Tree (DT), and Linear Regression (LR). Choosing K-Nearest Neighbors (KNN) is more appropriate for the performance it has, though we have one more algorithm which is LR but it might lead to overfitting. The remining two algorithms RF and DT are performing lesser than KNN.

CHAPTER-7

USE CASES OF THE PROJECT

7. USE CASES OF THE PROJECT

Individual Users: Your personalized diet recommendation system offers tailored diet plans for health-conscious individuals based on their nutritional goals and preferences. It supports weight management and assists those with chronic illnesses like diabetes or heart disease by providing customized dietary recommendations.

Healthcare Providers: Healthcare professionals can use the system to guide patients on personalized diets, aiding them in managing various health conditions. The system integrates with healthcare platforms for patient monitoring, tracking adherence, and assessing progress over time.

Wellness and Fitness Coaches: The system allows coaches to create personalized nutrition plans for clients that align with their fitness goals. This helps in meal planning and ensures clients follow a healthy diet, supporting their overall wellness journey.

Workplace Wellness Programs: In corporate settings, the system encourages employees to adopt healthier eating habits and participate in group challenges based on personalized dietary recommendations. This leads to a healthier work environment and boosts employee engagement.

Food Service Industry and Educational Institutions: The system aids restaurants and catering services in menu planning to cater to customers' specific dietary needs and preferences. Educational institutions use the system for nutrition education and creating balanced cafeteria menus. Additionally, the system can be integrated into mobile and web apps for meal planning and grocery lists, enhancing user experiences and supporting a healthier lifestyle.

Mobile and Web Applications: The system can be integrated into existing health and fitness apps, providing users with personalized dietary recommendations as part of their overall wellness journey. It also supports meal planning apps, allowing users to conveniently create meals and grocery lists based on the system's guidance. This integration helps users maintain a healthy diet and manage their nutrition more effectively.

CHAPTER-8

CONCLUSION

8. CONCLUSION

In conclusion, this research highlights the effectiveness of K-Nearest Neighbours (KNN) in developing a personalized diet recommendation system. By utilizing data from Food.com and focusing on key nutritional metrics like BMI and user preferences, the system offers tailored dietary guidance aligned with individual health objectives and tastes. KNN's nuanced analysis ensures precise recommendations, supplemented by visualized nutrient values and preparation instructions, aiding users in making informed dietary decisions. Future enhancements could include additional machine learning techniques and broader datasets to further refine the system's accuracy and recommendations. Overall, this tailored diet recommendation system holds promise for advancing personalized nutrition, promoting better health outcomes for users globally.

8.1 LIMITATIONS

While the tailored diet recommendation system shows potential in providing personalized nutrition guidance, there are some limitations to the project that should be considered. The quality and diversity of the data used are crucial to the effectiveness of the KNN-based system. If the data is outdated, incomplete, or lacks variety in food and nutrient information, it may lead to less accurate recommendations.

The system takes into account important metrics such as BMI, BMR, and user preferences, but it may not consider other critical factors such as medical conditions, allergies, or genetic predispositions. This could restrict the system's ability to fully personalize recommendations to individual health needs. Additionally, the system's scalability and performance could face challenges as the user base and data volume grow, potentially impacting response time and efficiency. User input accuracy is another limitation, as the system relies on the information provided by users for optimal functioning. Inaccuracies or omissions in user-reported data can lead to suboptimal recommendations. Moreover, the system may not be fully aware of broader lifestyle factors such as exercise habits, sleep patterns, or stress levels, all of which play a significant role in diet and nutrition. Finally, encouraging user adoption and engagement with the system may pose challenges. Factors such as ease of use, trust in the recommendations, and user interface design can influence whether users actively use the system and follow its guidance. Ethical and privacy concerns around the use of personal health data must also be addressed to ensure responsible and ethical implementation.

CHAPTER-9

FUTURE WORK

9. FUTURE WORK

Building upon the results and findings of this paper, future work can focus on several key areas to enhance the tailored diet recommendation system. One potential direction is improving the quality and diversity of the data used for training the KNN algorithm. This includes incorporating more up-to-date and comprehensive food databases with diverse nutrient information, as well as data from different populations to ensure the system can provide relevant recommendations for a wider range of users.

Another area of future work involves integrating additional health and lifestyle factors into the system's recommendations. By including information such as medical conditions, allergies, genetic factors, sleep patterns, exercise habits, and stress levels, the system can offer more holistic and personalized guidance that aligns with individual health goals and circumstances.

Additionally, research could explore methods to enhance user engagement and trust in the system. This might involve designing user-friendly interfaces and providing clear explanations for each recommendation. By helping users understand the reasoning behind their personalized guidance, they may be more inclined to follow the suggestions and maintain long-term use of the system.

Lastly, the ethical and privacy aspects of the system can be further studied to ensure responsible and secure data handling. Developing and implementing robust privacy-preserving techniques, as well as obtaining user consent for data usage, will be essential for the system's success and user trust. Investigating these areas will contribute to the development of a more advanced, reliable, and user-centric personalized diet recommendation system.

CHAPTER-10

REFERENCES

10. REFERENCES

[1] Diet, nutrition and the Prevention of Chronic Diseases: Report of a joint WHO/FAO expert consultation, Geneva, 28 January - 1 February 2002 (no date) World Health Organization. Available at: https://www.who.int/publications-detail redirect/924120916X (Accessed: 09 April 2024).

[2] C. Iwendi, S. Khan, J. H. Anajemba, A. K. Bashir and F. Noor, "Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model," in IEEE Access, vol. 8, pp. 28462-28474, 2020, doi: 10.1109/ACCESS.2020.2968537.

[3] Shah, M., Degadwala, S. and Vyas, D. (2022) 'Diet recommendation system based on different machine Computer Science, Engineering and Information Technology, pp. 01–10. doi:10.32628/cseit228249.

[4] Lambay, M.A. and Mohideen, S.P. (2022) 'A hybrid approach-based diet recommendation system using ML and Big Data Analytics', Journal of Mobile Multimedia [Preprint]. doi:10.13052/jmm1550-4646.1864.

[5] T. Islam, A. R. Joyita, M. G. R. Alam, M. Mehedi Hassan, M. R. Hassan and R. Gravina, "Human-Behavior Based Personalized Meal Recommendation and Menu Planning Social System," in IEEE Transactions on Computational Social Systems, vol. 10, no. 4, pp. 2099 2110, Aug. 2023, doi: 10.1109/TCSS.2022.3213506.B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.

[6] Chen, C.-H. et al. (2018) 'Person—personalized expert recommendation system for optimized nutrition', IEEE Transactions on Biomedical Circuits and Systems, 12(1), pp. 151–160. doi:10.1109/tbcas.2017.2760504.

[7] R. Yera Toledo, A. A. Alzahrani and L. Martínez, "A Food Recommender System Considering Nutritional Information and User Preferences," in IEEE Access, vol. 7, pp. 96695-96711, 10.1109/ACCESS.2019.2929413. 2019, doi:

[8] Sarker, I.H. (2021b) Machine learning: Algorithms, real world applications and research directions - SN computer science, doi: 10.1007/s42979-021-00592-x

[9] Cheng, D. et al. (2021) 'Why dataset properties bound the scalability of Parallel Machine Learning Training Algorithms', IEEE Transactions on Parallel and Distributed Systems, 32(7), doi:10.1109/tpds.2020.3048836. pp. 1702–1712.

[10] Harris, J.A. and Benedict, F.G. (1918) 'A biometric study of human basal metabolism', Proceedings of the National Academy of Sciences, 4(12), pp. 370–373. doi:10.1073/pnas.4.12.370.

[11] S. Karimulla Basha, T. N Shankar, "Fuzzy logic-based forwarder selection for efficient data dissemination in VANETs," Wireless Networks, The Journal of Mobile Communication, Computation and Information, vol.27,3, pp.2193-2216, Feb 2021,

[12] S. Karimulla Basha, T. N. Shankar, "Fuzzy Logic Based Multi-Hop Broadcasting in High-Mobility VANETs," International Journal of Computer Science and Network Security, vol. 21, no. 3, pp. 165-171, March 2021.