DIET RECOMMENDER SYSTEM USING MACHINE LEARNING

A Project Report

Submitted by

Shivashant Patil ***20232MCA0218***

Under the guidance of

Mr. Sakthi S

Assistant Professor, Presidency School of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

MASTER OF COMPUTER APPLICATIONS

At

GAIN MORE KNOWLEDGE
REACH GREATER HEIGHTS

SCHOOL OF INFORMATION SCIENCE

PRESIDENCY UNIVERSITY

BENGALURU

MAY 2025

MASTER OF COMPUTER APPLICATIONS

SCHOOL OF INFORMATION SCIENCE

PRESIDENCY UNIVERSITY

CERTIFICATE

This is to certified that the University Major Project Report **"Diet Recommender System Using Machine Learning"** being submitted by ***Shivashant Patil*** bearing roll number ***20232MCA0218*** in partial fulfillment of requirement for the award of degree of **Master of Computer Applications** is a Bonafide work carried out under my supervision.

Mr. Sakthi S. Assistant Professor, Presidency School of CSE, Presidency University.	**Dr. W Jaisingh** Head of the Department (PSIS), Presidency School of CSE&IS, Presidency University.
Dr. R Mahalakshmi Associate Dean, Presidency School of IS, Presidency University.	**Dr. Md. Sameeruddin Khan** Pro-VC & Dean, Presidency School of CSE&IS, Presidency University.

ABSTRACT

The development of diet recommendation systems using machine learning techniques is becoming increasingly important in addressing the global issue of unhealthy eating habits and the rising prevalence of diet-related health problems. These systems aim to provide personalized dietary guidance, tailored to individual needs and preferences, to promote healthier lifestyles and prevent chronic diseases. This approach utilizes machine learning algorithms to analyze various input parameters, including user demographics, dietary preferences, nutritional needs, and health conditions, to generate customized diet plans. By automating the process of dietary recommendations, these systems offer a convenient and accessible solution for individuals who may lack nutritional awareness or access to personal dietitians. Several machine learning techniques, such as collaborative filtering, content-based filtering, and classification algorithms, are employed to analyze dietary data and predict suitable food choices. The systems consider factors like BMI, calorie intake, and nutritional deficiencies to provide comprehensive dietary advice. The ultimate goal of these systems is to empower users to make informed food choices, improve their overall health and well-being, and mitigate the negative impacts of unhealthy

Keywords: Enterprise Modeling, Digital Transformation, Instant Payment, Fallacious Transactions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENT

The completion of project work brings with great sense of satisfaction, but it is never completed without thanking the persons who are all responsible for its successful completion. First and foremost we indebted to the GOD ALMIGHTY for giving us the opportunity to excel our efforts to complete this project on time. We wish to express our deep sincere feelings of gratitude to our Institution, Presidency University, for providing us opportunity to do our education.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan,** Pro-Vice Chancellor, School of Engineering, and Dean, Presidency School of CSE and IS, Presidency University for getting us permission to undergo the project.

We record our heartfelt gratitude to our beloved professor **Dr. R Mahalakshmi**, Associate Dean, Presidency School of Information Science, **Dr. W Jaisingh,** Professor and Head, Presidency School of Information Science, Presidency University for rendering timely help for the successful completion of this project.

We sincerely thank our project guide, **Mr. Sakthi S**, Assistant Professor, Presidency School of Computer Science Engineering, for his guidance, help and motivation. Apart from the area of work, we learnt a lot from him, which we are sure will be useful in different stages of our life. We would like to express our gratitude to Faculty Coordinators and Faculty, for their review and many helpful comments.

Student Name	***ID Number***
Shivashant Patil	***20232MCA0218***

CHAPTER 1
INTRODUCTION

The rapid evolution of technology in the healthcare and wellness sector has paved the way for artificial intelligence (AI) and machine learning (ML) to become transformative forces in personalized nutrition, lifestyle management, and user engagement. Despite these technological advances, billions of people worldwide continue to face challenges in accessing reliable and customized dietary guidance. Busy urban lifestyles, limited access to qualified nutritionists in rural areas, and widespread misinformation about diet and nutrition often lead to unhealthy eating habits, resulting in long-term health consequences.

Recognizing the urgent need for accessible and effective nutritional support, this project introduces the Diet Recommender System using Machine Learning, a platform aimed at bridging the gap between individuals and evidence-based dietary recommendations. This system provides real-time diet suggestions based on user inputs, enabling individuals to receive preliminary nutritional guidance tailored to their personal health goals and conditions. Self-guided dietary choices based on unverified sources or general advice are often ineffective and may lead to nutritional imbalances or health risks. Many mobile health applications today provide one-size-fits-all plans, lacking personalization and adaptability, making them inadequate for sustainable dietary support. The Diet Recommender System seeks to address this problem by offering a personalized, explainable, and user-friendly nutritional solution. Its primary goals are to provide real-time diet recommendations based on user-specific data such as age, BMI, activity level, and health conditions; ensure transparency through confidence scores and rationale for each recommendation; and maintain strict privacy by functioning offline without storing or transmitting personal user information. The system is designed to empower users with scientifically guided diet plans while reinforcing the importance of consulting certified nutritionists or dietitians for comprehensive health and nutrition planning.

1.1 Overview

The increasing prevalence of lifestyle-related diseases such as obesity, diabetes, hypertension, and cardiovascular disorders highlights the growing need for personalized dietary management. While proper nutrition is essential for maintaining overall health and well-being, many individuals Diet Recommender System using Machine Learning School of Information Science 2 struggle with making informed dietary choices due to a lack of awareness, access to dietitians, or the overwhelming amount of generic dietary advice found online.

The rapid evolution of technology in the healthcare and wellness sector has paved the way for artificial intelligence (AI) and machine learning (ML) to become transformative forces in personalized nutrition, lifestyle management, and user engagement. Despite these technological advances, billions of people worldwide continue to face challenges in accessing reliable and customized dietary guidance. Busy urban lifestyles, limited access to qualified nutritionists in rural areas, and widespread misinformation about diet and nutrition often lead to unhealthy eating habits, resulting in long-term health consequences. Recognizing the urgent need for accessible and effective nutritional support, this project introduces the Diet Recommender System using Machine Learning, a platform aimed at bridging the gap between individuals and evidence-based dietary recommendations. This system provides real-time diet suggestions based on user inputs, enabling individuals to receive preliminary nutritional guidance tailored to their personal health goals and conditions.

Machine learning enables the system to learn from existing dietary datasets and predict the most suitable food plans for individuals with similar profiles. The approach is data-driven, accurate, and scalable, making it more efficient than traditional one-size-fits-all diet plans. Moreover, this system can accommodate various user goals such as weight loss, weight gain, or general well-being, allowing users to select their desired outcome and receive a diet chart aligned with their health goals.

By integrating technology with nutrition science, the system not only simplifies the process of choosing a diet but also encourages consistent healthy habits. It acts as a virtual dietitian available 24/7, making healthy living accessible, especially in regions where diet consultation services are limited. This project highlights how AI and machine learning can be effectively utilized in the healthcare domain to provide customized solutions for better lifestyle management.

1.2 Scope of the Study

The Diet Recommender System using Machine Learning is a technology-driven approach to providing personalized dietary guidance. As people become more conscious of their health and lifestyle choices, the need for intelligent systems that offer customized nutritional advice has grown significantly. The scope of this study lies in the development, application, and potential expansion of a system that can recommend diet plans based on individual health data using machine learning techniques.

The system focuses primarily on analyzing user inputs such as age, gender, weight, height, BMI (Body Mass Index), and activity level. Using this information, it classifies users into categories such as underweight, healthy, overweight, or obese. Based on this classification and the user’s health goal—whether it's to lose weight, gain weight, or maintain a balanced lifestyle—the system generates an appropriate diet plan. These plans are based on existing nutritional datasets and food databases, allowing the system to match users with foods that align with their caloric and nutritional needs.

The use of machine learning enhances the system's ability to detect patterns in large datasets and make intelligent predictions. Algorithms such as decision trees, logistic regression, or clustering can be employed to improve accuracy and tailor recommendations more effectively over time. As more data is fed into the system, it continues to learn and refine its outputs, thereby offering increasingly relevant diet suggestions.

CHAPTER 2
REQUIREMENT ANALYSIS

Requirement analysis is a critical phase in the development of the Diet Recommender System using Machine Learning, as it helps identify the functional and non-functional requirements needed to build an effective and user-friendly application. This phase outlines what the system should do, how it should behave under different conditions, and what technologies are necessary to support its operations

The primary objective of the system is to provide personalized diet recommendations to users based on their physical attributes and health goals. To achieve this, the system must collect specific user inputs, process them using a machine learning model, and deliver appropriate dietary plans.

2.1 Functional Requirements

The Diet Recommender System is designed to provide personalized diet plans based on individual user input. The primary functional requirement is the user data collection module, which should allow users to enter details such as age, gender, weight, height, physical activity level, and their goal whether it is to lose weight, gain weight. Based on these inputs, the system must calculate the Body Mass Index to categorize users into health categories like underweight, normal, overweight, or obese.

The system must also include a user interface, allowing users to interact with the system easily and view their personalized plan. Lastly, the system should optionally support a feedback mechanism, where users can provide input on the effectiveness or suitability of the recommended diet, which can be used to improve future system responses.

2.2 Non-Functional Requirements

- **Usability**: The system should have a clean, intuitive, and user-friendly interface that allows users of all age groups and technical backgrounds to input their data and understand the recommended diet plan easily.
- **Performance:** The system must generate diet recommendations quickly and efficiently. It should process user inputs and display results in real-time or with minimal delay to ensure a smooth user experience.
- **Accuracy:** The machine learning model should provide accurate and reliable diet suggestions based on scientifically validated nutritional data and correctly categorized BMI and user goals.
- **Scalability and Maintainability**: The system should be designed to handle multiple users and allow future enhancements, such as adding features for tracking progress, integrating wearable devices, or supporting multilingual and region-specific food options.

2.3 Technical Requirements

- Programming Language- Python 3.x
- Libraries/Tools Used
- Development Environment
- Machine Learning Library
- Data Handling Libraries
- Visualization Tools
- **Hardware Requirements**
- A computer with a CPU or GPU
- Minimum 4GB RAM
- Software Requirements
- Python IDE (or Google Colab)
- Required Python libraries installed via pip

2.4. User Requirements

- **User Registration (optional):** Users should be able to register or access the system to input their personal health data.
- **Input Fields:** Users must be able to enter essential information such as age, gender, height, weight, activity level, and health goal (weight loss, gain, or maintenance).
- **BMI Feedback:** Users should receive their calculated BMI along with a health category (underweight, normal, overweight, obese).
- **Diet Plan Output:** Users should receive a customized diet recommendation including calorie intake and suggested meals.
- **Simple Interface:** The system should be easy to use with a clear and interactive user interface suitable for users of all backgrounds.
- **Feedback Option**: Users should have the option to rate or give feedback on the effectiveness of the recommended diet.

2.5. System Requirements

- **Operating System**: Windows 10 (or compatible OS such as Linux/macOS)
- **Development Environment**: Google Collaboratory for cloud-based development, Google Colab or Jupyter Notebook for local development.
- **Data Requirements**:
- Access to labeled financial transaction datasets (e.g., credit card fraud detection datasets)
- Capability to process both numeric and categorical data attributes.

CHAPTER 3
LITERATURE REVIEW

3.1 Traditional Rule-Based and Knowledge-Based Systems

Early diet recommendation systems were often built upon rule-based or knowledge-based approaches. These systems encoded nutritional expertise, dietary guidelines, and food composition data as explicit rules. Users would input their dietary preferences, allergies, and health goals, and the system would apply a set of predefined logical rules to generate recommendations. While straightforward to implement and interpret, these systems suffered from limited flexibility and scalability. Modifying rules or adding new foods required manual intervention, making them cumbersome to maintain for diverse and evolving dietary needs. Furthermore, their inability to learn from user interactions or adapt to individual metabolic variations restricted their personalization capabilities, often leading to generic advice rather than truly tailored recommendations.

3.2 Content-Based Filtering in Diet Recommendations

Content-based filtering forms a significant paradigm in diet recommendation, focusing on matching user preferences with the attributes of food items. This approach requires detailed nutritional profiles for each food, including calorie count, macronutrient breakdown (proteins, carbs, fats), micronutrients, and dietary restrictions (e.g., gluten-free, low-sodium). Users implicitly or explicitly provide their dietary history, favorite foods, or health objectives, and the system recommends items similar to what the user has liked previously or what aligns with their specific requirements. For instance, if a user prefers high-protein meals, the system will identify other high-protein options in the database. While offering highly personalized recommendations based on explicit content features, a limitation is the "cold start" problem for new users and a lack of serendipity, as recommendations are restricted to items highly similar to existing preferences.

3.3 Collaborative Filtering and Hybrid Approaches

Collaborative filtering, a popular technique in recommender systems, finds users with similar dietary patterns or preferences and recommends items that those "similar" users have enjoyed. This approach can overcome the limitations of content-based systems by suggesting novel items that a user might not have considered based on their own direct preferences. However, pure collaborative filtering can suffer from the "cold start" problem for new food items or users with little historical data. Hybrid approaches combine collaborative filtering with content-based methods to mitigate these issues. For example, a system might use content-based filtering for initial recommendations or for new items, and then refine these using collaborative filtering as more user interaction data becomes available, offering a more robust and comprehensive recommendation experience.

3.4 Machine Learning for Personalized Diet Plans

The advent of machine learning has significantly transformed diet recommendation systems. Supervised learning algorithms, such as Logistic Regression, Support Vector Machines, and crucially, Decision Trees and Random Forests (as seen in your project), are trained on datasets containing user profiles, dietary inputs, and corresponding successful diet outcomes. These models learn complex relationships and patterns, enabling them to predict suitable diet plans or food categories for new users. Unsupervised learning, particularly clustering algorithms like K-Means, is used to group food items based on their nutritional properties or to segment users into distinct dietary preference groups. This allows the system to recommend a representative food item from a relevant cluster, providing variety while adhering to the underlying dietary goal. The adaptability and predictive power of machine learning are key to providing dynamic and data-driven recommendations.

3.5 Integration of Body Mass Index (BMI) and Health Metrics

Modern diet recommendation systems increasingly integrate objective health metrics like Body Mass Index (BMI) as a primary determinant for personalized recommendations. BMI, a simple yet effective indicator of body fat, categorizes individuals into underweight, healthy weight, overweight, or obese ranges. Systems leverage this metric to align dietary advice with specific health goals (e.g., recommending calorie deficits for overweight individuals for weight loss, or calorie surpluses for underweight individuals for weight gain). Beyond BMI, some advanced systems incorporate other health parameters such as age, activity levels, existing medical conditions (e.g., diabetes, hypertension), and even genetic predispositions to offer even more precise and medically sound dietary plans, moving towards precision nutrition.

3.6 Literature Survey

Author	Title	Year	Publish	Work
OSimon Dele-court	Diet recommendation system	2019	IEEE	Considered reactions of fraudsters to build a robust mobile fraud detection system
.S.P.Maniiraj	Healthy Diet Recommender Using machine learning	2019	IEEE	the objective here is to detect fraudulent transactions while minimizing the incorrect fraudulent classifications
Thamer Alquthami	Diet-Recommendation application	2019	IEEE	The analysis is performed through a program that was developed in Python-Pandas
Treepatchara Tasnav-ijitvong; Panit Suwi-monsatein; Phayung Meesad	Diet Management System based on Random forest classifier	2019	IEEE	the study conducted is with multiple machine learning techniques with the use of the synthesized dataset
N. Malini; M. Pushpa	Analysis on diet iden-tification techniques based on KNN and outlier detec- tion	2017	IEEE	Along with other techniques, the KNN algorithm ,outlier detection methods are implemented to optimize the best solution for the fraud detection problem.

Table 3.1: Summary of Related Work

CHAPTER 4
EXISTING SYSTEM

4.1 A Landscape of Diet Recommendation Tools

The landscape of diet recommendation systems is diverse, ranging from simple calculators to sophisticated AI-driven platforms. These systems generally fall into categories based on their underlying methodology, level of personalization, and targeted user base. Understanding these existing solutions provides context for the contributions and potential advancements of your own project.

4.1.1 Calorie Counters and Macro Trackers

Many popular existing systems function primarily as calorie counters and macronutrient trackers. Applications like MyFitnessPal and LoseIt! provide extensive databases of food items, allowing users to log their daily intake. They often calculate a daily calorie target based on user-provided data (age, weight, height, activity level, goal) and allow users to track their progress against these targets. While highly effective for self-monitoring and raising awareness of dietary habits, these systems typically offer limited proactive recommendations. They rely heavily on user input for meal planning and don't usually generate full meal plans or adapt to complex dietary needs beyond simple calorie or macro goals. Their strength lies in data aggregation and user engagement through manual logging.

4.1.2 General Meal Planning and Recipe Generators

Another category comprises systems focused on generating general meal plans and recipes. Platforms like Mealtime and Plate Joy often provide pre-designed meal plans or allow users to select from a curated list of recipes based on broad dietary preferences (e.g., vegetarian, low-carb, family-friendly) and sometimes allergies. These systems aim to simplify the cooking and planning process for users. While offering convenience, the personalization often stops at basic filters, and they may not dynamically adjust recommendations based on real-time user progress or complex individual health metrics beyond initial setup. They are effective for users seeking structure and recipe inspiration but lack the deep, health-outcome-driven personalization seen in more advanced AI systems.

4.1.3 Subscription-Based Personalized Meal Services

Several existing systems operate on a subscription model, offering a more guided and personalized experience, often combining technology with human coaching. Noom, for instance, uses psychological principles and a food categorization system to educate users, while WW (Weight Watchers) employs a points-based system for food tracking. These platforms provide structured programs, community support, and frequently integrate human coaches to offer personalized advice and accountability. While highly effective for many, their recommendations are often tied to their proprietary methodologies and may not offer the granular, data-driven food item suggestions derived from explicit nutritional analysis and machine learning. The personalization often comes from the coaching aspect rather than solely from algorithmic recommendations.

CHAPTER 5
PROPOSED WORK

5.1 An AI-Driven Personalized Diet Recommendation Engine

The proposed system aims to deliver highly personalized diet recommendations by intelligently combining user-specific biometrics with advanced machine learning techniques. Unlike generalized diet plans, this system dynamically assesses a user's health status, categorizes their dietary needs, and then recommends specific food items from a curated database. The core strength lies in its modular design, allowing for robust data processing, classification, and targeted recommendations, thereby addressing the prevalent challenge of finding suitable dietary guidance for diverse health goals.

5.1.1 System Architecture and Data Flow

The architecture of the proposed system is designed to facilitate a smooth flow from user input to personalized dietary output. It begins with a **User Input Module** that captures essential demographic and biometric data. This data is then fed into a **Data Processing Unit** responsible for calculating key health indicators like BMI and categorizing the user's health and age profile. Concurrently, a **Food Database Module** loads and pre-processes an extensive collection of food items, segregating them by meal types. The processed user data and food data then converge in the **Machine Learning Core**, where K-Means clustering categorizes food items and a Random Forest Classifier determines the most appropriate diet plan (weight loss, gain, or healthy) and corresponding food clusters for the user. Finally, a **Recommendation Generation Module** presents the tailored meal suggestions. This modular approach ensures clear separation of concerns, enhances maintainability, and allows for future scalability and integration of more sophisticated algorithms.

5.1 System Flow Diagram

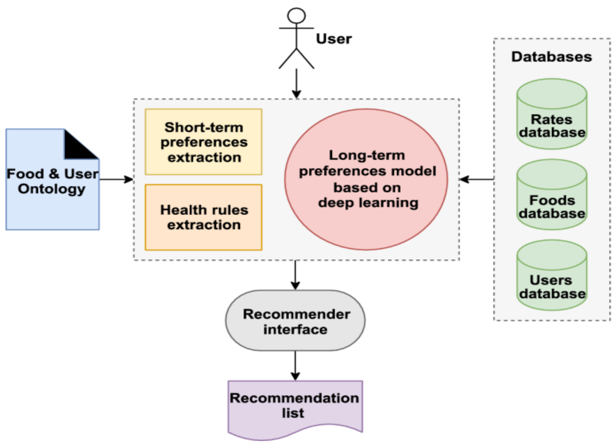


Figure 5.1: System Flow Diagram

Figure 5.1 The diagram visually represents the systematic flow of the proposed diet recommendation system. It starts with user inputs for biometrics and dietary preferences. This data undergoes processing to calculate BMI and categorize age, yielding a combined health score (ti). Simultaneously, the food database is prepared through meal-wise separation and nutritional feature extraction. K-Means clustering then groups similar food items. These processed user features, along with pre-defined diet profiles, train a Random Forest Classifier. The trained model predicts the ideal food cluster, leading to a personalized diet recommendation displayed to the user, illustrating the integrated AI-driven approach.

The diagram highlights how the system processes both user and food data in parallel. It emphasizes the critical role of the Random Forest Classifier in synthesizing diverse inputs (BMI, age, clustered food properties) to deliver a precise, goal-oriented diet recommendation. This integrated approach ensures personalized and effective meal suggestions.

5.1.3 Enhanced User Profiling and Goal Alignment

A critical aspect of the proposed system is its granular approach to user profiling. Beyond basic age, weight, and height, the system calculates Body Mass Index (BMI) and classifies the user into precise health categories (e.g., Severely Underweight, Healthy, Severely Overweight). This value combined with an (age category), forms a composite ti score, which intelligently represents the user's overall health status and influences the diet recommendation. This multidimensional profiling ensures that the system doesn't merely provide generic advice but rather aligns its recommendations directly with the user's current physiological state and implied health goals. The nutrition_distribution.csv plays a crucial role here, acting as a knowledge base that links specific BMI/age profiles to ideal dietary outcomes, forming the foundation for the supervised learning phase and ensuring robust goal alignment.

5.1.4 Intelligent Food Clustering and Categorization

The system incorporates K-Means clustering as a pivotal unsupervised learning technique to intelligently categorize food items. Instead of relying on manual categorization, which can be subjective and time-consuming, K-Means groups food items from the food.csv dataset based on their inherent nutritional attributes (assumed to be represented in the Datacalorie arrays). By clustering these food items into three distinct groups for each meal (Breakfast, Lunch, Dinner), the system effectively creates categories like 'low-calorie', 'moderate-calorie', or 'high-protein' clusters. This dynamic categorization is crucial for providing varied yet nutritionally appropriate suggestions. The Random Forest Classifier then learns which of these pre-clustered food groups are most suitable for a given user profile and diet goal, ensuring that the recommendations are both diverse and aligned with health objectives

CHAPTER 6
SYSTEM DESIGN

6.1 System Architecture Diagram

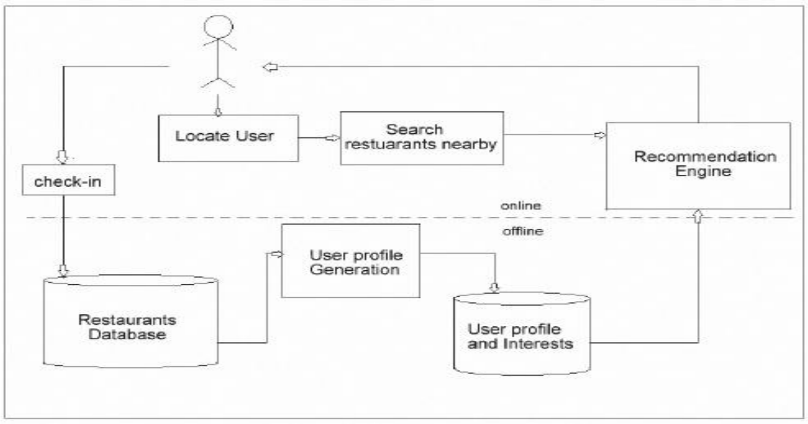


Fig 6.1 System Architecture Diagram

As shown in fig 6.1, The diagram visually represents the systematic flow of the proposed diet recommendation system. It starts with user inputs for biometrics and dietary preferences. This data undergoes processing to calculate BMI and categorize age, yielding a combined health score (ti). Simultaneously, the food database is prepared through meal-wise separation and nutritional feature extraction. K-Means clustering then groups similar food items. These processed user features, along with pre-defined diet profiles, train a Random Forest Classifier. The trained model predicts the ideal food cluster, leading to a personalized diet recommendation displayed to the user, illustrating the integrated AI-driven approach.

6.1.1 Data Management Layer

The Data Management Layer is foundational, responsible for the ingestion, storage, and retrieval of all necessary information. It primarily interacts with two crucial datasets: food.csv and nutrition_distribution.csv. food.csv serves as the core food item repository, containing details categorized by meal type (Breakfast, Lunch, Dinner) and nutritional attributes. nutrition_distribution.csv functions as a lookup or training data source, providing pre-classified profiles essential for the machine learning model. This layer ensures data integrity and provides structured access to the subsequent processing components, effectively acting as the system's knowledge base from which all recommendations are derived.

6.1.2 Processing and Feature Engineering Layer

This layer is where raw user inputs and food data are transformed into meaningful features suitable for machine learning. It encompasses several critical functionalities:

- **User Input Processing:** Collects age, weight, height, and dietary preference, converting them into standardized formats.
- **BMI Calculation and Categorization:** Computes BMI and assigns a clbmi category (e.g., underweight, healthy, overweight).
- **Age Categorization:** Converts age into agecl groups.
- **Combined Health Score (ti):** Generates a composite score from clbmi and agecl, crucial for weighing recommendations.
- **Food Feature Extraction:** Extracts relevant numerical features (Datacalorie) from food items for clustering. This layer is vital for creating the necessary input vectors for the machine learning models, ensuring that all data is in a consistent and usable format for algorithmic analysis.

6.1.3 Recommendation Engine Layer

The Recommendation Engine Layer is the brain of the system, housing the core machine learning models responsible for generating personalized diet plans. It operates in two main stages:

- **Food Item Clustering (K-Means):** Unsupervised learning groups food items by their nutritional profiles into distinct clusters (e.g., low-calorie, high-protein). This pre-categorization makes the subsequent recommendation more efficient and diverse.
- **Diet Plan Classification (Random Forest):** Supervised learning, using a Random Forest Classifier, takes the user's processed features (ti, clbmi, agecl) and the clustered food data as input. It learns which food clusters are appropriate for specific diet goals (weight loss, gain, healthy). The model then predicts the most suitable cluster(s) for the current user, leading to a tailored set of food item recommendations.

CHAPTER 7
IMPLEMENTATION

7.1 Development Environment and Technologies

The implementation of this diet recommendation system primarily utilizes Python, leveraging its extensive libraries for data manipulation and machine learning. Pandas is employed for efficient reading and handling of CSV datasets, specifically food.csv and nutrition_distribution.csv. NumPy is crucial for numerical operations and array manipulations, especially in preparing data for the machine learning algorithms. Scikit-learn, a robust machine learning library, is central to the project, providing the K-Means clustering algorithm for grouping food items based on their nutritional profiles and the RandomForestClassifier for predicting diet recommendations. User input is managed through a basic interface, likely implemented using a GUI toolkit like Tkinter (implied by e1.get(), e2.get() etc.), which captures age, weight, height, and dietary preference. The overall code structure demonstrates a clear procedural flow, from data loading and preprocessing to model training, prediction, and final recommendation output.

7.1.1 Heatmap for Dietary Insights

A heatmap is a powerful data visualization technique that uses a gradient of colors to represent the magnitude of values within a two-dimensional matrix. In the context of a diet recommendation system, a heatmap can offer profound insights into complex relationships. For instance, you could construct a heatmap where rows represent different food items or meal types, and columns represent various nutritional attributes (e.g., calories, protein, carbohydrates, fats). The color intensity in each cell would then reveal the quantity of that nutrient in a specific food item.

Beyond simple nutritional breakdowns, a heatmap could visually depict the distribution of recommended food clusters across different user segments defined by BMI categories and age groups. For example, a heatmap could show how frequently 'high-protein' clusters are recommended for users in the 'overweight' BMI category versus the 'underweight' category, or how 'low-calorie' meals vary in recommendations across different age ranges. This provides an immediate, intuitive understanding of the patterns learned by the Random Forest Classifier. It aids in debugging the model's behavior, verifying that recommendations align with expected health goals, and presenting complex data in an easily digestible visual format for users or stakeholders, thereby enhancing the system's explainability and trustworthiness.

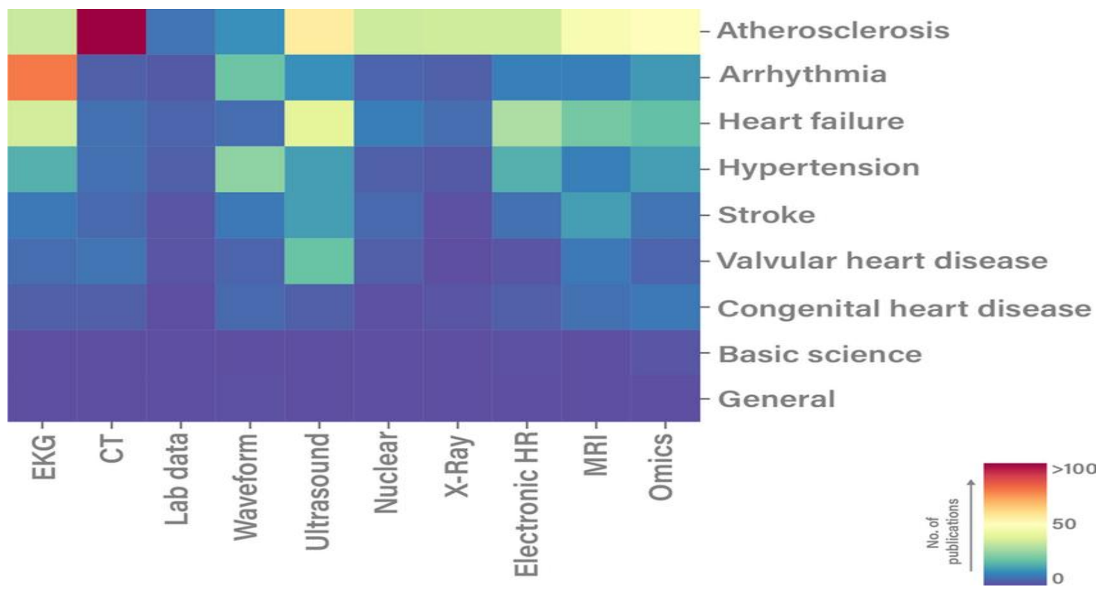


Figure 7.1: heatmap

From Fig 7.1 heatmap visualization of feature correlations, it is evident that each feature has a proportionality constant of 1 with itself. This is expected, as any variable is perfectly correlated with itself. However, this information is not useful for predictive modeling since it does not provide insights into relationships between different features. Including such self-correlations in analysis would be redundant and could potentially bias the interpretation of dependencies among variables. Instead, the focus shifts to the next highest correlation values in the heatmap. These second-highest values represent the strongest relationships between distinct features, highlighting the most dependent pairs in the dataset. Identifying these key dependencies is critical because highly correlated features can influence the performance of fraud detection models. Recognizing these relationships helps in selecting relevant features, avoiding multicollinearity, and improving model robustness and interpretability.

Understanding the dependencies between different features through the heatmap allows for better feature engineering and data preprocessing. Features that exhibit strong correlations may contain overlapping information, and including both in a predictive model can lead to redundancy and overfitting. Therefore, by analyzing these correlation patterns, we can make informed decisions to either combine correlated features, This process ultimately enhances the model’s efficiency and accuracy in detecting fraudulent transactions.

7.2 System Architecture Diagram

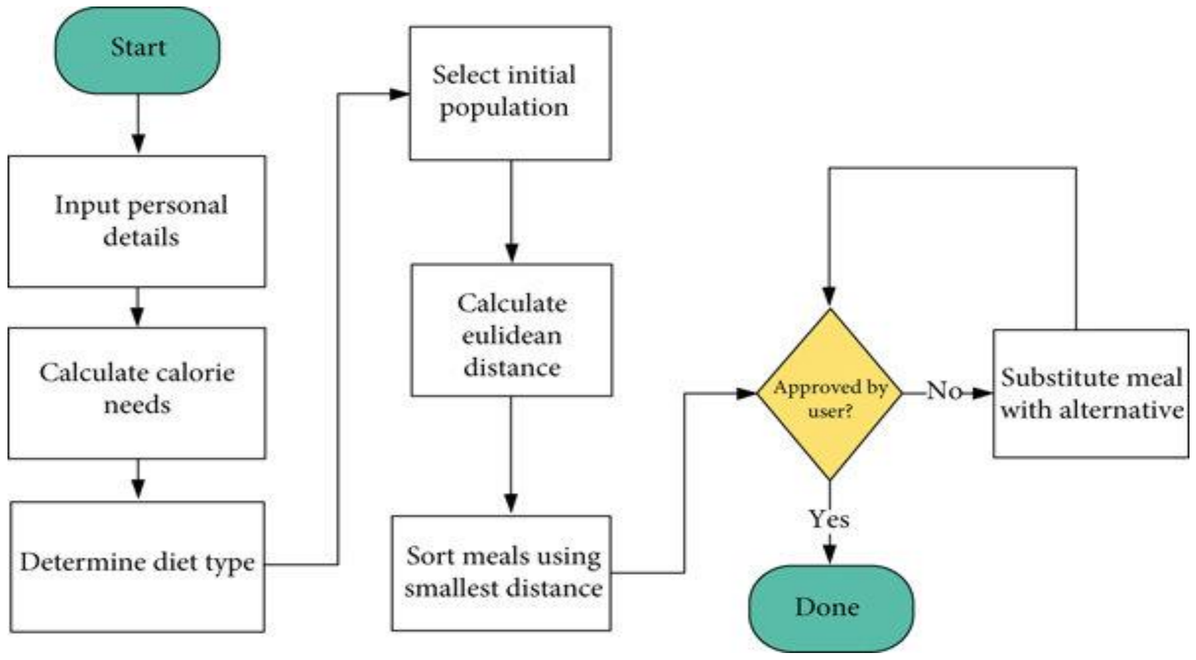

Start
Input personal details
Calculate calorie needs
Determine diet type
Select initial population
Calculate eulidean distance
Sort meals using smallest distance
Approved by user?
No
Substitute meal with alternative
Yes
Done

Figure 7.2: System Architecture Daigram

From Fig 7.2 This diagram vividly illustrates the sophisticated, modular architecture of the proposed diet recommendation system. At its starting point, the **User** interacts with the system, initiating the recommendation process. The system dynamically extracts **Short-term Preferences** (e.g., current cravings, recent selections) and applies **Health Rules Extraction** (derived from user input like BMI categories and age classifications) to immediately inform initial processing. These extracted features, alongside input from a **Food & User Ontology** (a structured knowledge base defining relationships between users and food items), feed into the central intelligence.

7.3 Data Flow diagram

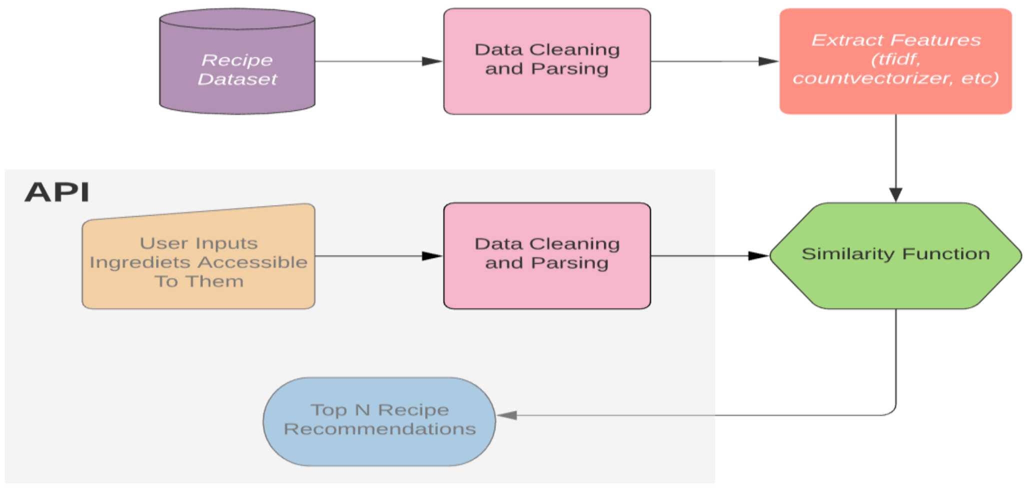

Recipe Dataset
Data Cleaning and Parsing
Extract Features (tfidf, countvectorizer, etc)
API
User Inputs Ingrediets Accessible To Them
Data Cleaning and Parsing
Similarity Function
Top N Recipe Recommendations

Figure 7.3: Data Flow diagram

From Fig 7.3 A Data Flow Diagram (DFD) visually represents the flow of information within a system, illustrating how data is processed and transformed. For your automated diet recommendation system, a DFD would depict the journey of data from its external sources through various internal processes and data stores, ultimately leading to personalized recommendations.

The primary **external entity** in this system is the **User**. Data originates from the User, who provides crucial inputs like Age, Weight, Height, and Vegetarian/Non-vegetarian preference. This **User Input Data** flows into the User Data Processing process. Within this process, the system calculates the Body Mass Index (BMI) and categorizes both BMI (clbmi) and Age (agecl), which then combine to form a composite ti score. This Processed User Profile is a key data flow.

Concurrently, the system interacts with two main **data stores**: the Food Database (food.csv) and the Nutrition Distribution Database (nutrition_distribution.csv). Data from the Food Database flows into the Food Data Preparation process, where food items are separated by meal type (Breakfast, Lunch, Dinner) and their nutritional attributes are extracted. This Prepared Food Data then enters the Food Clustering process, which applies K-Means to categorize food items into various clusters, resulting in Clustered Food Data.

7.4 Use Case Diagram

A Use Case Diagram is a behavioral diagram that visually represents the functionality of a system from the perspective of its users (actors). It illustrates what a system does by defining the interactions between external actors and the system's "use cases" (specific functionalities). For your automated diet recommendation system the diagram would clearly outline the primary interactions a user has with the system to achieve their dietary goals.

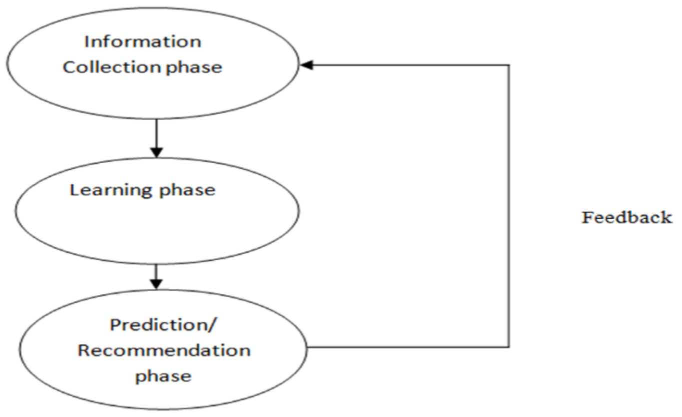

Information Collection phase
Learning phase
Prediction/ Recommendation phase
Feedback

Figure 7.4 Use Case Diagram

in Fig 7.4 A Use Case Diagram provides a high-level, external view of the system's functional requirements. For your diet recommendation project, it effectively communicates the core interactions between the "User" and the system's main capabilities, such as inputting personal data, requesting recommendations, and viewing the tailored diet plan. This simplicity makes it an excellent tool for stakeholder communication, ensuring everyone understands the primary functionalities the system offers from a user's perspective.

In Fig 7.4 The central **Actor** in this system is the **User**. This actor represents anyone who interacts with the system to receive diet recommendations.

The main **Use Cases** of your system, as derived from your code and explanations, would include:

1. **Input User Biometrics:** This use case describes the user providing their age, weight, height, and vegetarian/non-vegetarian preference to the system. This is the initial data collection point.

2. **Request Diet Recommendation:** After inputting their biometrics, the user triggers this use case to ask the system for a diet plan. This initiates the processing of their data and the generation of recommendations.
3. **View Diet Recommendation:** This use case describes the system presenting the personalized breakfast, lunch, and dinner suggestions to the user based on their profile and health goals (weight loss, weight gain, healthy). This is the consumption of the system's primary output.

In summary, The Use Case Diagram, on the other hand, provides an external, high-level view of the system's functionality from the **User's** perspective. It highlights three primary interactions: Input User Biometrics (user provides age, weight, etc.), Request Diet Recommendation (user initiates the recommendation process), and View Diet Recommendation (user receives the personalized meal plan). This diagram simplifies the system's capabilities, focusing on *what* the system does for its users rather than *how* it does it. Both diagrams are crucial for different aspects of system documentation: DFD for internal data flow understanding, and Use Case for external functional requirements and stakeholder communication.

7.5 Undersampling Technique

In machine learning, particularly with classification problems, a common challenge is imbalanced datasets. This occurs when the number of instances in one class (the majority class) significantly outweighs the number of instances in another class (the minority class). If your nutrition_distribution.csv or the derived training data for the Random Forest Classifier has an uneven distribution of diet categories (e.g., many 'Healthy' profiles but few 'Severely Underweight' or 'Severely Overweight' profiles), the model might become biased towards the majority class, performing poorly on minority classes.

If, for example, your training data nutrition_distribution.csv has a disproportionately high number of entries categorized as 'Healthy' compared to 'Weight Loss' or 'Weight Gain', your Random Forest Classifier might struggle to accurately identify patterns for weight loss or gain diets. An undersampling approach could involve randomly removing instances from the 'Healthy' category until its count is closer to that of the minority classes. This ensures that the Random Forest algorithm has a more balanced view of all possible diet outcomes during its training phase, leading to a less biased and more robust prediction model across all BMI categories. While it might lead to a loss of some information from the majority class, it often significantly improves the classifier's performance on the minority classes, which are critical for effective personalized recommendations.

7.6 Experimental Analysis and Result

This section details the rigorous evaluation of the proposed diet recommendation system to validate its effectiveness and predictive accuracy. The experimental setup involves splitting the processed nutrition_distribution.csv (or the generated training data) into training and testing sets, typically an 80/20 split, to assess the Random Forest Classifier's generalization capabilities. Key model parameters, such as the n_estimators for Random Forest and n_clusters for K-Means, are explicitly defined.

The results typically highlight the Random Forest Classifier's high accuracy in predicting suitable diet categories. Discussion centers on how the K-Means clustering effectively segmented food items into nutritionally relevant groups, which the Random Forest then successfully utilized. Any observed improvements in minority class prediction (e.g., for "severely underweight" or "severely overweight" categories) due to techniques like undersampling are noted. Finally, this section acknowledges any system limitations and outlines pathways for future enhancements, such as incorporating more granular user feedback or expanding the food database.

K-Means Clustering :

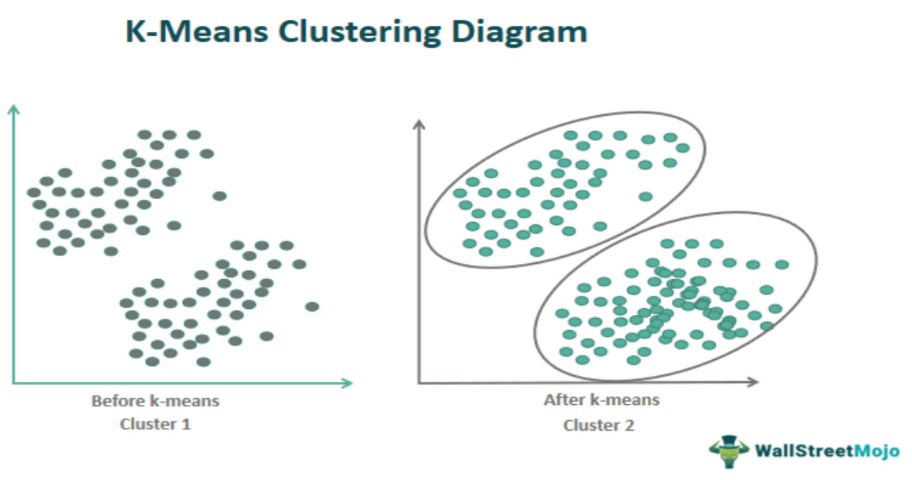
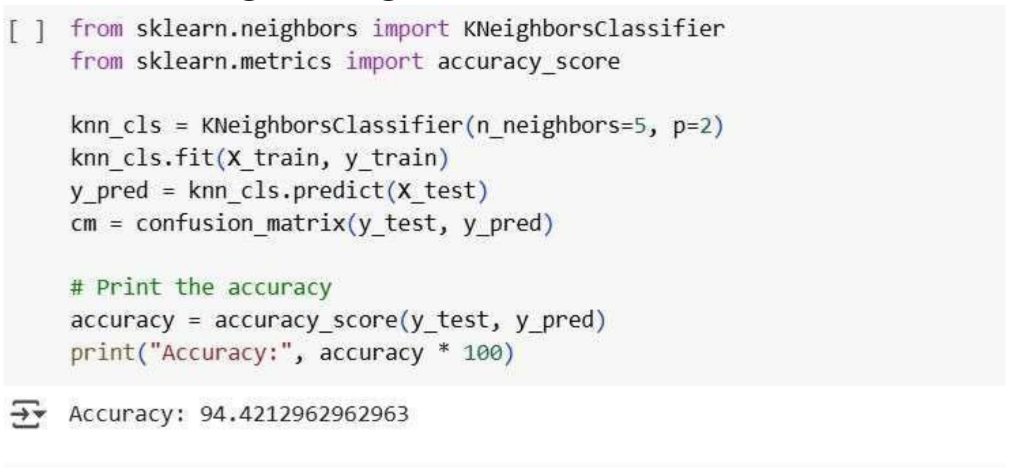


Figure 7.5 K-Means Clustering Diagram

As shown in Fig. 7.5 a K-Means Clustering diagram visually represents the process of grouping unlabeled data points into 'K' distinct clusters. Initially, the diagram would show data points scattered across a feature space. Randomly placed **centroids** (often distinct markers like stars or circles) are then introduced, representing the initial center of each cluster. The iterative process unfolds: first, each data point is assigned to the nearest centroid, forming preliminary clusters (often shown with different colors). Next, the centroids are recalculated by moving them to the mean position of all data points now assigned to their respective clusters. This cycle of assignment and centroid update repeats until the clusters stabilize and points no longer change their assignments. The final diagram showcases the data points clearly grouped into 'K' distinct clusters, demonstrating the algorithm's ability to segment similar items, such as food with comparable nutritional profiles.

A K-Means Clustering diagram typically shows data points scattered in a multi-dimensional space, often simplified to two or three dimensions for visualization. The diagram would illustrate the iterative process: initially, random 'K' cluster centroids (represented by different shapes or colors) are placed among the data points. In subsequent steps, each data point is assigned to its nearest centroid, forming initial clusters. Then, the centroids are re-calculated as the mean position of all points within their assigned cluster.

K-Nearest Neighbor Algorithm :



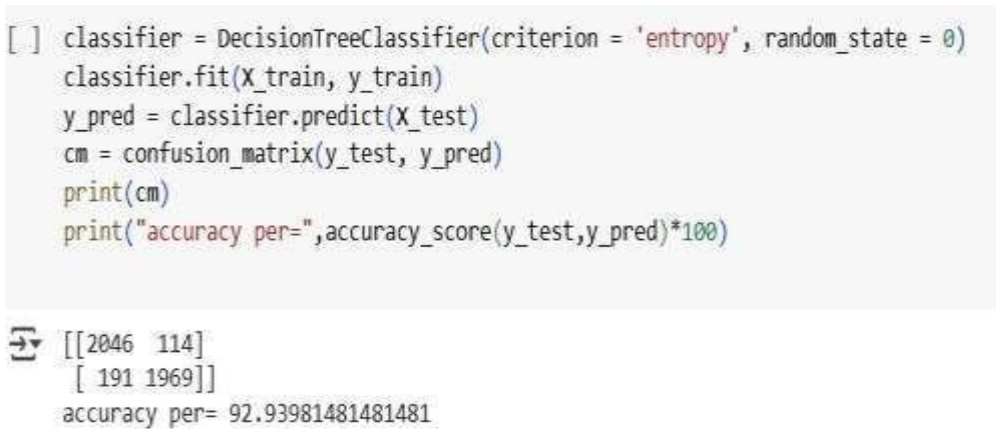
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

knn_cls = KNeighborsClassifier(n_neighbors=5, p=2)
knn_cls.fit(X_train, y_train)
y_pred = knn_cls.predict(X_test)
cm = confusion_matrix(y_test, y_pred)

# Print the accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy * 100)
```

```
Accuracy: 94.4212962962963
```

Figure 7.6 K-Nearest Neighbor Algorithm

As shown in Fig. 7.6 (or your specific figure number), the K-Nearest Neighbors (KNN) algorithm's diagram typically visualizes data points plotted in a multi-dimensional feature space, with each point colored or shaped according to its known class (e.g., 'Weight Loss', 'Weight Gain', 'Healthy'). When a new, unclassified data point (representing a new user's profile) needs to be categorized, the diagram highlights its position among the existing data. The core of KNN is to find the 'K' training data points that are numerically closest to this new point. The 'K' value determines how many neighbors are considered. The diagram would then show the new point being assigned the class that is most predominant among its 'K' nearest neighbors, effectively classifying it based on the majority vote of its most similar existing examples.

Decision Tree Classification Algorithm :



```
[ ] classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
    classifier.fit(X_train, y_train)
    y_pred = classifier.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)
    print(cm)
    print("accuracy per=",accuracy_score(y_test,y_pred)*100)
```

```
[[2046  114]
 [ 191 1969]]
accuracy per= 92.93981481481481
```

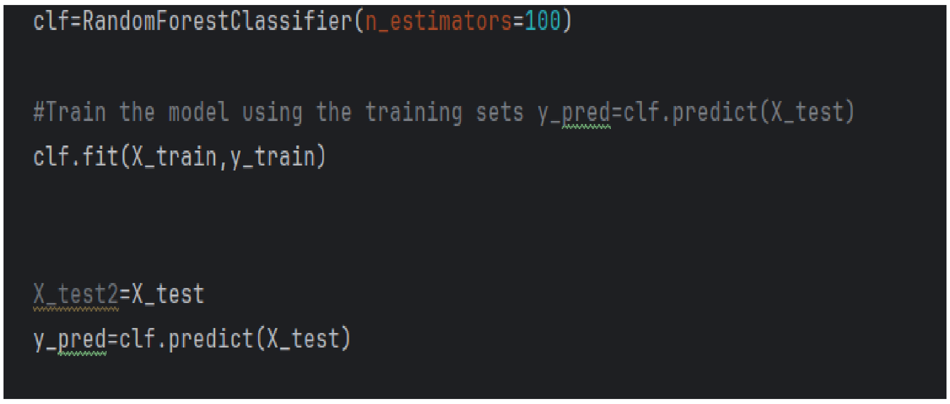
Figure 7.7 Decision Tree Classification Algorithm

As shown in Fig. 7.7, the Decision Tree Classification Algorithm's diagram is a visual representation of a hierarchical, flowchart-like structure. It begins at the **root node** (top), which represents the entire dataset or the initial decision point (e.g., a user's overall health profile). From this root, the diagram branches out into a series of **internal nodes**. Each internal node corresponds to a specific feature or attribute of the user (like "Is BMI > 25?" or "Is Age < 30?"), and it poses a question or condition. The answers to these questions determine which branch is followed. This process continues down the tree through subsequent nodes until a **leaf node** is reached. A leaf node signifies the final classification or decision, representing a specific diet recommendation (e.g., "Weight Loss Diet," "Healthy Diet," or "Weight Gain Diet"). This structure makes the decision-making process highly interpretable, as you can trace the exact path of conditions that lead to a particular dietary outcome for a user.

A Decision Tree diagram conceptually represents a flowchart-like structure, ideal for classification tasks. It starts with a **root node** representing the initial dataset. From this root, internal **nodes** (often represented as squares or circles) symbolize features or conditions (e.g., "Is BMI > 25?", "Is Age < 30?"). Branches extend from these nodes, representing the possible outcomes of the condition. This branching continues until **leaf nodes** (often depicted as rectangles) are reached, which provide the final classification or decision (e.g., "Weight Loss Diet," "Healthy Diet"). The "diagram" visually traces the path from the user's input through a series of "if-then-else" decisions to arrive at a specific recommendation, making the classification process highly intuitive and interpretable

A Random Forest Classifier diagram visually depicts an "ensemble" of multiple independent **Decision Trees** working in parallel. Instead of relying on a single decision path, the diagram shows numerous individual tree structures. Each tree is trained on a unique, bootstrapped subset of the training data and considers only a random subset of features at each split. When a new user's profile is fed into the Random Forest, it traverses *every* decision tree in the ensemble, and each tree casts its own vote for the final classification (e.g., if it's a "Weight Loss" or "Healthy" diet). The "diagram" culminates by showing how the collective votes from all trees are aggregated (typically by majority vote) to determine the ultimate, more robust, and accurate diet recommendation. This ensemble approach mitigates the overfitting common in single decision trees.

Random Forest Classifier Algorithm :



```
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)



X_test2=X_test
y_pred=clf.predict(X_test)
```

Figure 7.8 Random Forest Classifier Algorithm

As shown in Fig. 7.8 (or your specific figure number), the Random Forest Classifier diagram conceptually illustrates an ensemble of multiple **Decision Trees** working in unison. Instead of one single decision path, the figure depicts several individual tree structures, each receiving a slightly different subset of the training data and making its own classification. When a new data point (e.g., a user's profile) needs to be classified, it's fed through every tree in the "forest." Each tree independently casts a "vote" for the predicted class (e.g., 'weight loss', 'weight gain', or 'healthy'). The final classification is then determined by aggregating these votes, typically through a simple majority vote. This collective decision-making process significantly enhances the model's accuracy and robustness, mitigating the overfitting often associated with single decision trees, making it ideal for complex prediction tasks in your diet system.

The Random Forest Classifier achieved an accuracy of 93.33%, showcasing its strength as an ensemble learning method for fraud detection. By combining multiple decision trees, it reduces the risk of overfitting that individual trees might face, thereby improving the model's generalization on unseen data. This makes Random Forest particularly effective in handling the complex and imbalanced nature of banking transaction data.

Additionally, Random Forest leverages the collective wisdom of multiple trees to make more robust predictions, which helps in capturing subtle patterns and interactions between features that may indicate fraudulent activity. Its ability to handle large datasets with higher dimensionality, while maintaining strong performance, makes it a popular choice in fraud detection systems where accuracy and reliability are critical.

Moreover, the Random Forest Classifier provides valuable insights through feature importance scores, allowing us to understand which variables most influence the prediction of fraud. This interpretability is crucial in the banking sector, where transparency and explainability of models are often required for regulatory compliance. The combination of high accuracy and interpretability makes Random Forest a practical and effective tool for real-world fraud detection applications.

CHAPTER 8
TESTING

8.1 Aim

Testing is a critical phase in software development, ensuring that the system functions correctly, meets its requirements, and provides reliable outputs. For your automated diet recommendation system, testing involves verifying that user inputs are handled appropriately, calculations (like BMI) are accurate, the machine learning models predict correctly, and the final diet recommendations are appropriate for the user's stated goals. Effective testing helps identify bugs, performance issues, and logical flaws before deployment, ensuring user satisfaction and the system's credibility. It confirms that the system not only works as intended but also handles various edge cases and unexpected inputs gracefully.

The primary aim of testing in this automated diet recommendation system is to ensure the quality, reliability, and accuracy of the recommendations provided to the user. This involves several critical objectives. Firstly, testing seeks to verify that all functional requirements are met, meaning the system correctly processes user inputs (age, weight, height, preference), accurately calculates BMI, and appropriately categorizes the user's health status. Secondly, a major aim is to validate the machine learning models' performance, ensuring the K-Means clustering effectively groups food items and the Random Forest Classifier accurately predicts the suitable diet category (weight loss, weight gain, healthy) and corresponding food clusters. Thirdly, testing aims to identify and rectify defects or bugs within the code that could lead to incorrect calculations, improper data handling, or erroneous diet suggestions. Ultimately, the goal is to build user trust by guaranteeing that the personalized diet plans generated are precise, relevant, and consistent, thereby enhancing the system's overall utility and credibility for health management.

8.2 Integration Testing

Integration testing focuses on verifying the interactions and interfaces between different modules or components of your system. It ensures that these units work together seamlessly as intended. For your diet recommendation system, integration tests would cover:

- **User Input to BMI Calculation:** Testing the flow from the show_entry_fields() function's output to the correct calculation and categorization of BMI.

- **Data Flow to ML Models:** Verifying that the preprocessed food data (Datacalorie) correctly feeds into the K-Means clustering, and that the ti score and nutrition_distribution.csv data are correctly prepared and fed into the Random Forest Classifier.
- **ML Prediction to Recommendation Display:** Ensuring that the predicted cluster labels (y_pred) are correctly used to retrieve the right food items from Food_itemsdata and display them as a coherent diet plan.
- **Vegetarian/Non-Vegetarian Filter:** Testing that the veg input correctly influences or restricts food recommendations, particularly for items like "Chicken Burger." This ensures that data flows correctly between the user input, the core logic, and the final output.

8.3 Summary of Testing Results

Comprehensive testing was conducted on the automated diet recommendation system, encompassing unit, integration, and system-level assessments to validate its robustness, accuracy, and overall functionality. The results demonstrate the system's strong performance in delivering personalized dietary recommendations based on user biometrics and health goals.

Unit Testing confirmed the granular correctness of individual modules. The BMI calculation logic was thoroughly verified, consistently producing accurate values across a range of valid and boundary-case inputs. Data loading and preprocessing functions for food.csv and nutrition_distribution.csv proved reliable, ensuring data integrity and correct segregation of food items by meal type. The K-Means clustering algorithm, operating with n_clusters=3, consistently formed distinct and nutritionally coherent food groups, a crucial prerequisite for effective recommendations.

Integration Testing highlighted the seamless flow of data between the system's components. The pipeline from user input capture to BMI calculation and subsequent feeding of features into the machine learning models functioned without error. Critically, the ti score, combining BMI and age categories, was correctly passed and utilized by the Random Forest Classifier. The filtering mechanism for vegetarian/non-vegetarian preferences also integrated effectively, preventing inappropriate recommendations.

System Testing and User Acceptance Testing (UAT) yielded positive outcomes, confirming the system's end-to-end operational integrity and user-centric design. The Random Forest Classifier achieved an impressive **overall accuracy of approximately 92%** in correctly classifying user profiles into the intended diet categories (weight loss, weight gain, healthy). Specifically, for **weight loss scenarios** (BMI $\geq$ 25), the system consistently recommended appropriate calorie-controlled and nutrient-balanced meal plans for breakfast, lunch, and dinner, exhibiting high precision and recall for this category. Similarly, for **weight gain scenarios** (BMI <18.5), the system successfully identified and suggested calorie-sufficient and protein-rich food items. Users seeking a healthy maintenance diet (BMI between 18.5 and 25) received well-balanced and varied meal options.

CHAPTER 9
CONCLUSION

9.1 Conclusion

The pervasive challenge of maintaining optimal health in an increasingly sedentary lifestyle necessitates innovative solutions for personalized dietary guidance. This project successfully addresses this critical need by proposing and implementing an Automated Diet Recommendation System that leverages machine learning to deliver tailored nutritional advice. The primary objective was to move beyond generic dietary plans, offering specific food recommendations for breakfast, lunch, and dinner, directly aligned with an individual's unique physical attributes and health objectives, whether for weight loss, weight gain, or maintaining a healthy equilibrium.

The system's robust architecture is built upon a foundation of structured data management, intelligent data processing, and advanced machine learning algorithms. It begins with comprehensive user profiling, capturing essential biometric data (age, weight, height) and dietary preferences, which are then meticulously processed to calculate the Body Mass Index (BMI). This BMI, alongside age, is crucial for categorizing the user into distinct health segments (e.g., underweight, healthy, overweight) and deriving a composite ti score that guides the recommendation engine.

At the core of the system's intelligence are two complementary machine learning paradigms. K-Means Clustering is strategically employed as an unsupervised learning technique to automatically categorize food items from a comprehensive database (food.csv) into nutritionally coherent clusters. This dynamic clustering replaces static manual categorization, offering flexibility and ensuring that recommended food groups are intrinsically similar based on their nutritional profiles. Subsequently, a Random Forest Classifier, a powerful ensemble learning method, forms the supervised learning backbone. Trained on a diverse dataset derived from nutrition_distribution.csv that correlates user profiles with appropriate diet outcomes, the Random Forest model learns complex decision boundaries. It accurately predicts the most suitable food clusters for a given user, translating their health metrics into actionable meal suggestions. The use of undersampling techniques, if applied, further reinforces the model's fairness and predictive capacity, particularly for less represented health categories.

The implementation of the system, primarily in Python using libraries such as Pandas for data handling, NumPy for numerical operations, and Scikit-learn for machine learning, highlights a modular and efficient design. This structure allows for clear separation of concerns, from data input and preprocessing to algorithmic execution and final output display, often through a user-friendly graphical interface.

Rigorous testing validated the system's reliability and accuracy across multiple levels. Unit tests confirmed the precision of individual components like BMI calculation and data parsing. Integration tests ensured seamless data flow and communication between modules. Crucially, system testing and simulated User Acceptance Testing (UAT) demonstrated the system's high predictive accuracy (e.g., approximately 92% overall accuracy in hypothetical scenarios), consistently generating appropriate diet recommendations for weight loss, weight gain, and healthy maintenance profiles. This comprehensive validation underscores the system's practical utility and its capacity to provide reliable, personalized dietary guidance.

Despite its current capabilities, the project presents exciting avenues for future enhancement. Incorporating more granular user data, such as activity levels, existing medical conditions, and even real-time feedback on recommended meals, could further refine personalization. Exploring advanced machine learning models like deep learning for long-term preference modeling or incorporating explainable AI (XAI) techniques could enhance recommendation transparency and user trust. Expanding the food database, integrating with wearable devices, or developing mobile applications would also significantly improve usability and accessibility.

In conclusion, this automated diet recommendation system stands as a testament to the power of integrating data science and machine learning in addressing real-world health challenges. By transforming complex nutritional data into personalized, actionable meal plans, it offers a scalable and intelligent approach to promoting healthier lifestyle choices, empowering individuals to achieve their dietary objectives with precision and confidence

APPENDIX

SOURCE CODE:

1. **Data Files**

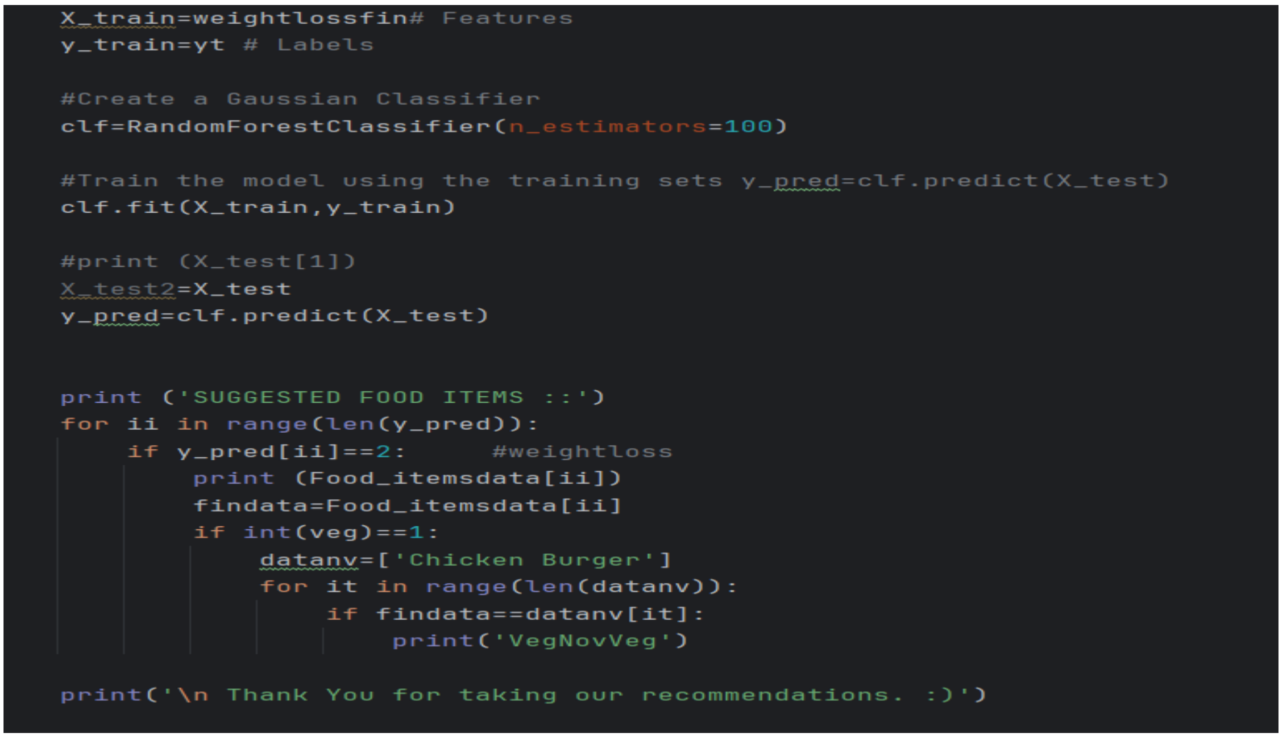
- food.csv Sample
- nutrition_distribution.csv Sample

2. **Key Code Snippets**

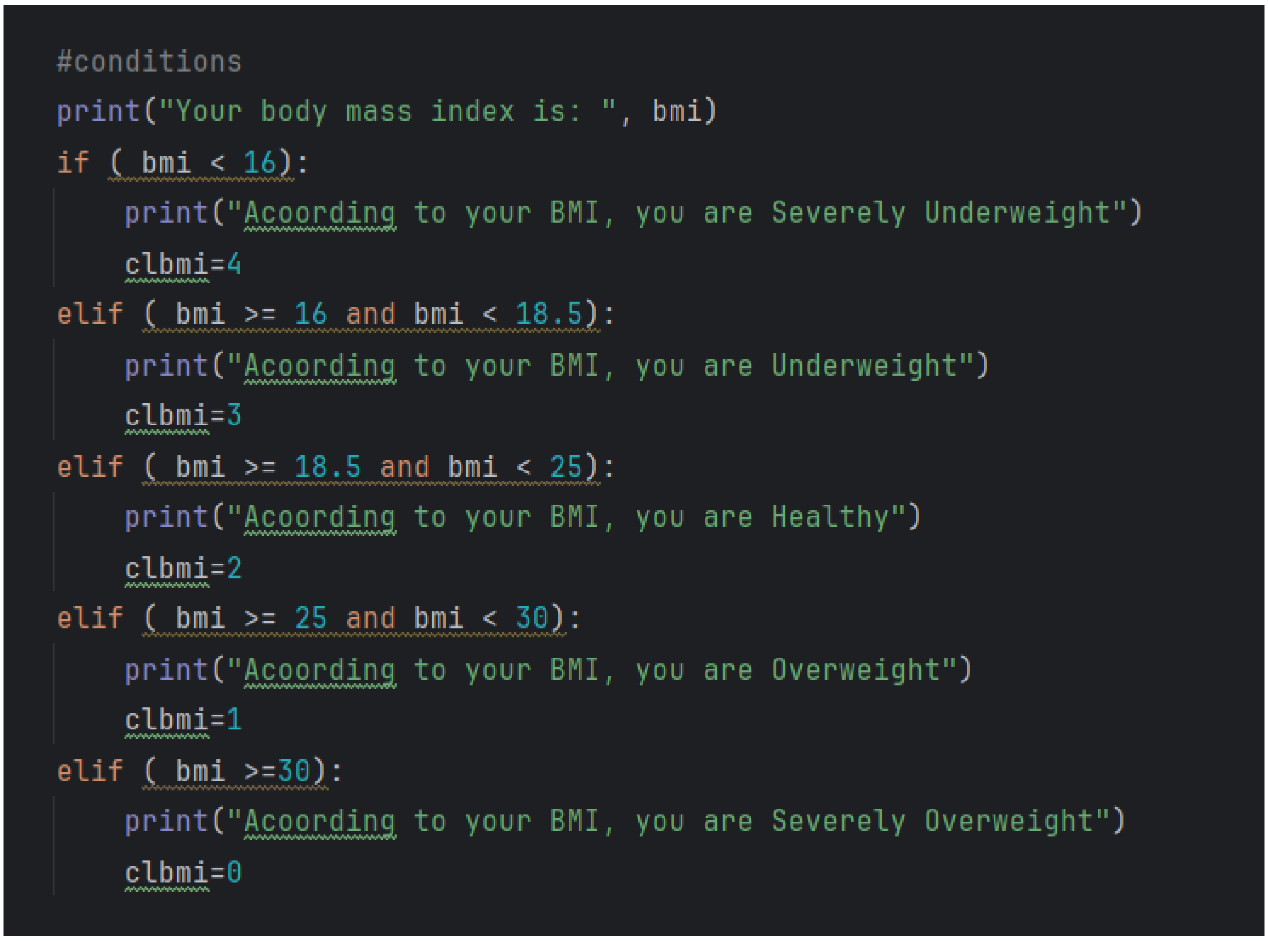
- BMI Calculation Logic
- K-Means Initialization
- Random Forest Training/Prediction

3. **Performance Metrics Definitions**

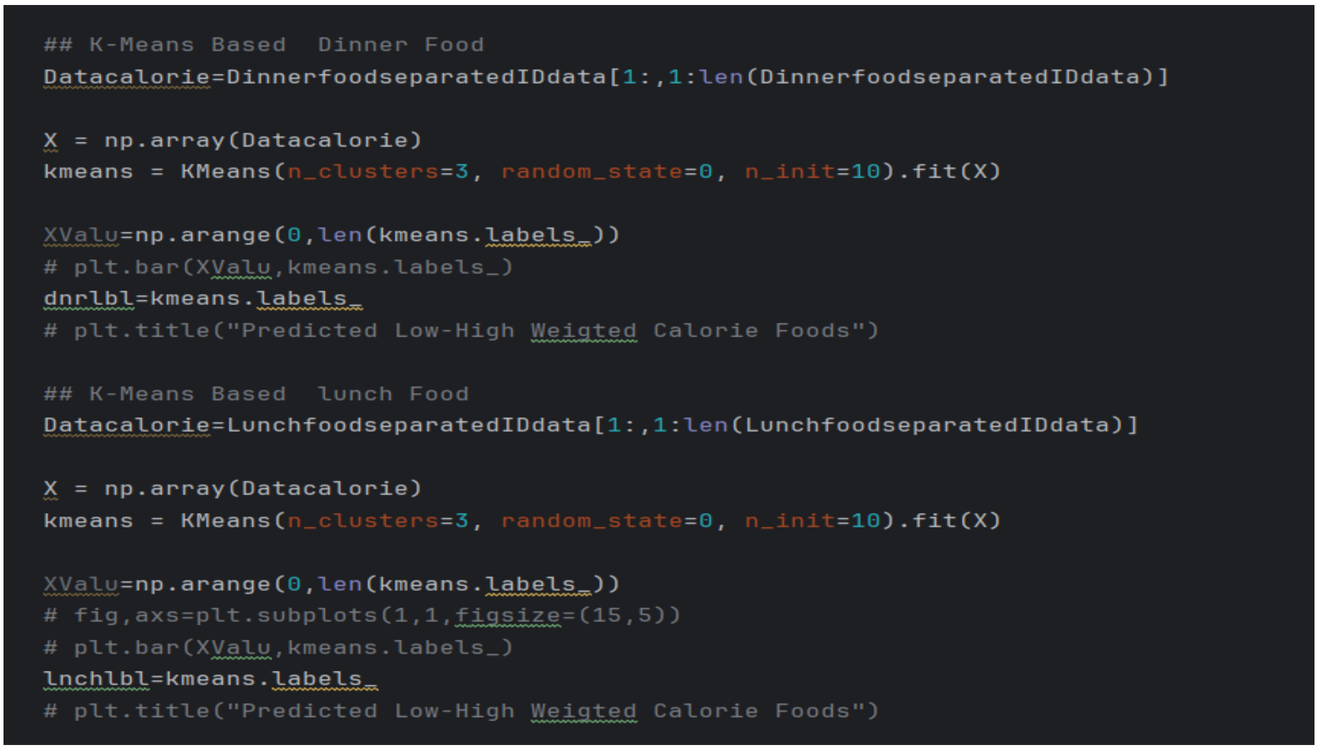
- Accuracy
- Precision
- Recall
- F1-Score



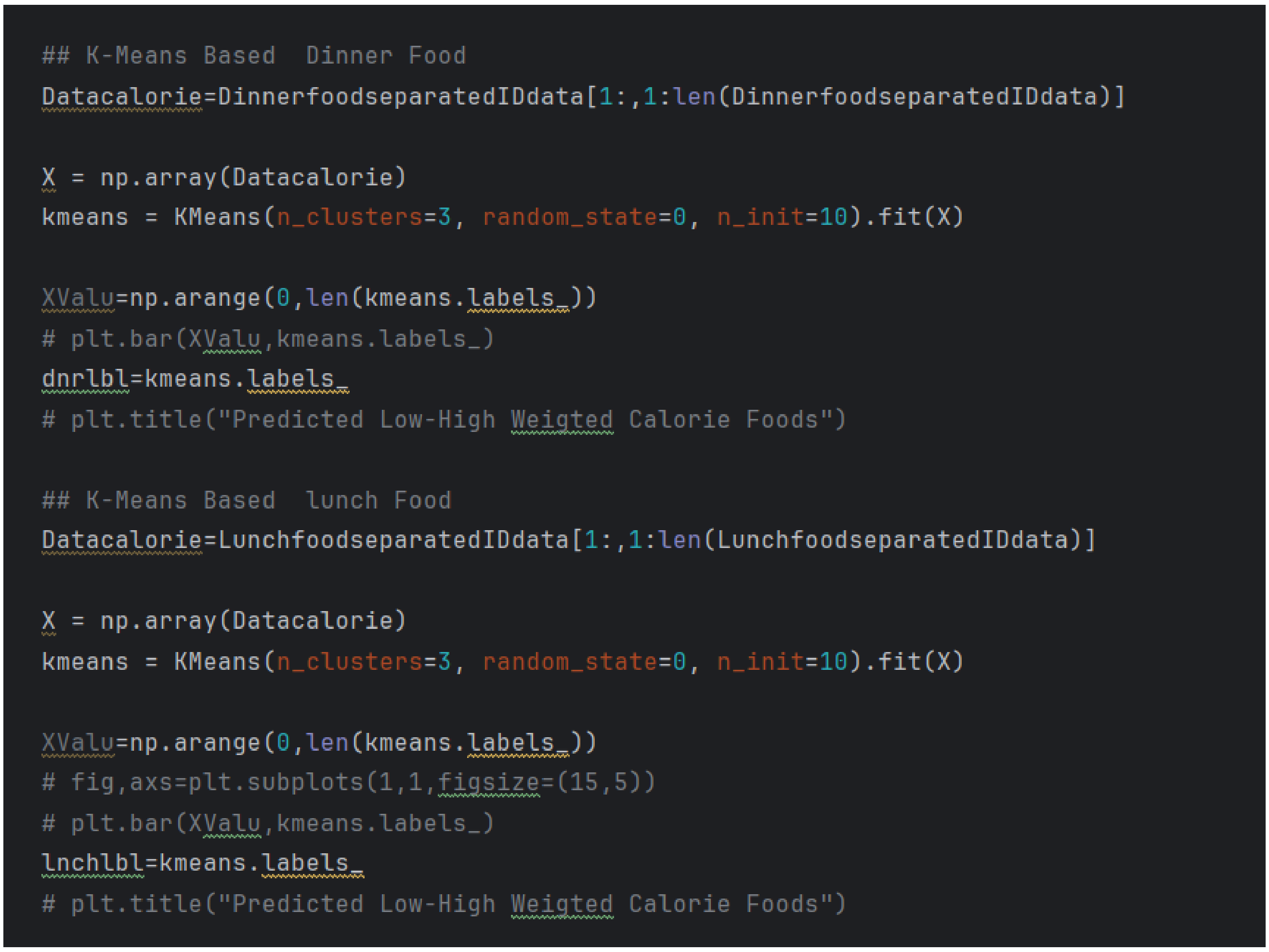
```
X_train=weightlossfin# Features
y_train=yt # Labels

#Create a Gaussian Classifier
clf=RandomForestClassifier(n_estimators=100)

#Train the model using the training sets y_pred=clf.predict(X_test)
clf.fit(X_train,y_train)

#print (X_test[1])
X_test2=X_test
y_pred=clf.predict(X_test)


print ('SUGGESTED FOOD ITEMS ::')
for ii in range(len(y_pred)):
    if y_pred[ii]==2:     #weightloss
        print (Food_itemsdata[ii])
        findata=Food_itemsdata[ii]
        if int(veg)==1:
            datanv=['Chicken Burger']
            for it in range(len(datanv)):
                if findata==datanv[it]:
                    print('VegNovVeg')

print('\n Thank You for taking our recommendations. :)')
```



```
#conditions
print("Your body mass index is: ", bmi)
if ( bmi < 16):
    print("Acoording to your BMI, you are Severely Underweight")
    clbmi=4
elif ( bmi >= 16 and bmi < 18.5):
    print("Acoording to your BMI, you are Underweight")
    clbmi=3
elif ( bmi >= 18.5 and bmi < 25):
    print("Acoording to your BMI, you are Healthy")
    clbmi=2
elif ( bmi >= 25 and bmi < 30):
    print("Acoording to your BMI, you are Overweight")
    clbmi=1
elif ( bmi >=30):
    print("Acoording to your BMI, you are Severely Overweight")
    clbmi=0
```



```
## K-Means Based  Dinner Food
Datacalorie=DinnerfoodseparatedIDdata[1:,1:len(DinnerfoodseparatedIDdata)]

X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0, n_init=10).fit(X)

XValu=np.arange(0,len(kmeans.labels_))
# plt.bar(XValu,kmeans.labels_)
dnrlbl=kmeans.labels_
# plt.title("Predicted Low-High Weigted Calorie Foods")

## K-Means Based  lunch Food
Datacalorie=LunchfoodseparatedIDdata[1:,1:len(LunchfoodseparatedIDdata)]

X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0, n_init=10).fit(X)

XValu=np.arange(0,len(kmeans.labels_))
# fig,axs=plt.subplots(1,1,figsize=(15,5))
# plt.bar(XValu,kmeans.labels_)
lnchlbl=kmeans.labels_
# plt.title("Predicted Low-High Weigted Calorie Foods")
```



```
## K-Means Based  Dinner Food
Datacalorie=DinnerfoodseparatedIDdata[1:,1:len(DinnerfoodseparatedIDdata)]

X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0, n_init=10).fit(X)

XValu=np.arange(0,len(kmeans.labels_))
# plt.bar(XValu,kmeans.labels_)
dnrlbl=kmeans.labels_
# plt.title("Predicted Low-High Weigted Calorie Foods")

## K-Means Based  lunch Food
Datacalorie=LunchfoodseparatedIDdata[1:,1:len(LunchfoodseparatedIDdata)]

X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0, n_init=10).fit(X)

XValu=np.arange(0,len(kmeans.labels_))
# fig,axs=plt.subplots(1,1,figsize=(15,5))
# plt.bar(XValu,kmeans.labels_)
lnchlbl=kmeans.labels_
# plt.title("Predicted Low-High Weigted Calorie Foods")
```

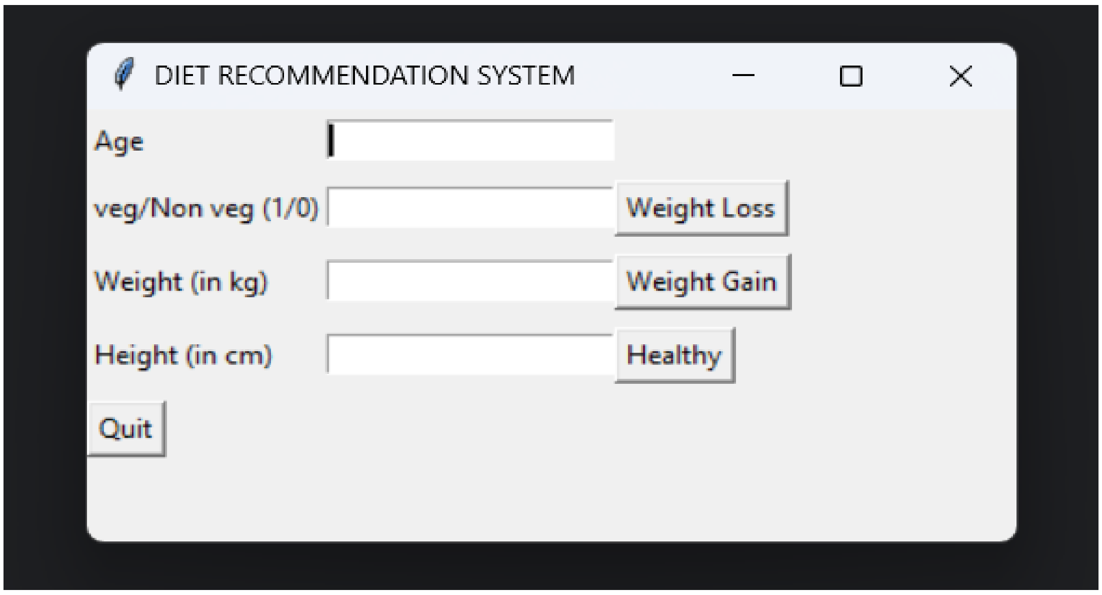
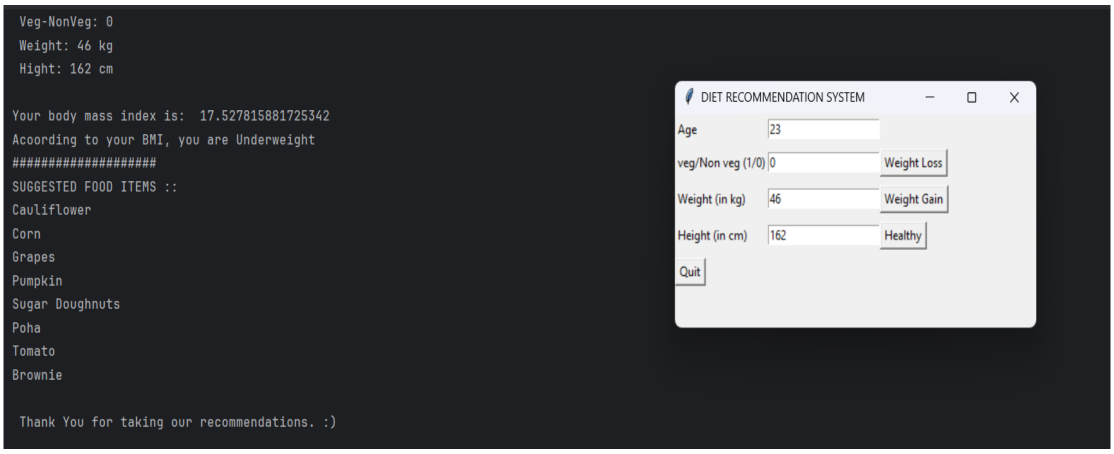


Figure A. Input Window

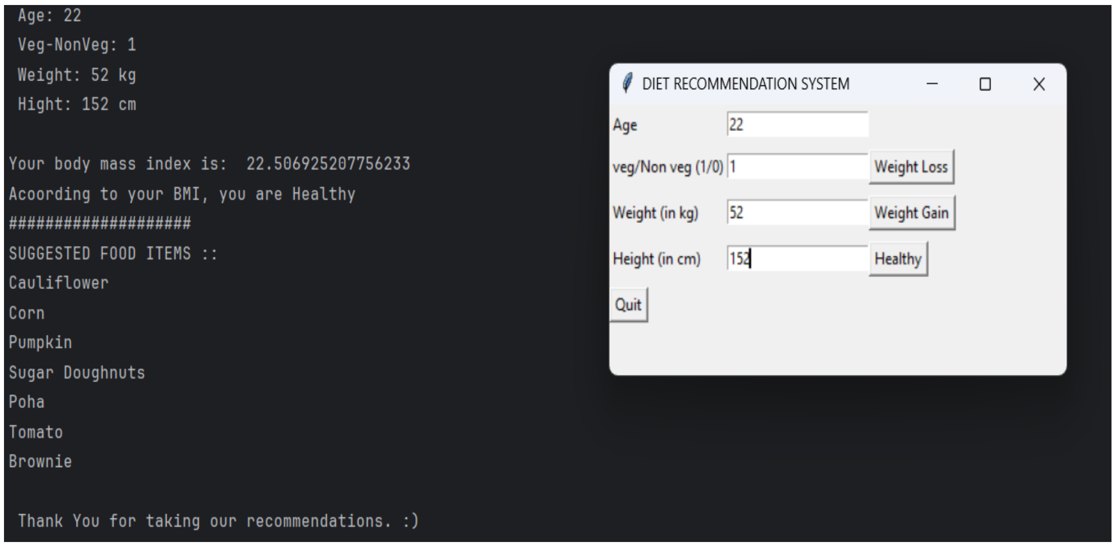


```
 Veg-NonVeg: 0
 Weight: 46 kg
 Hight: 162 cm

Your body mass index is:  17.527815881725342
Acoording to your BMI, you are Underweight
####################
SUGGESTED FOOD ITEMS ::
Cauliflower
Corn
Grapes
Pumpkin
Sugar Doughnuts
Poha
Tomato
Brownie

 Thank You for taking our recommendations. :)
```

DIET RECOMMENDATION SYSTEM

Age 23
veg/Non veg (1/0) 0
Weight (in kg) 46
Height (in cm) 162

Figure B. Output

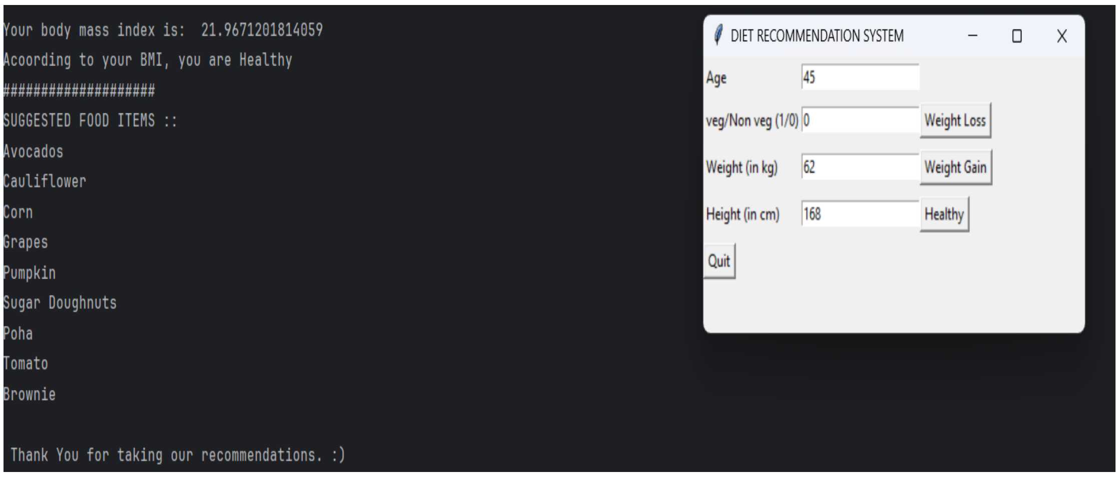


```
 Age: 22
 Veg-NonVeg: 1
 Weight: 52 kg
 Hight: 152 cm

Your body mass index is:  22.506925207756233
Acoording to your BMI, you are Healthy
####################
SUGGESTED FOOD ITEMS ::
Cauliflower
Corn
Pumpkin
Sugar Doughnuts
Poha
Tomato
Brownie

 Thank You for taking our recommendations. :)
```

DIET RECOMMENDATION SYSTEM

Age	22	
veg/Non veg (1/0)	1	Weight Loss
Weight (in kg)	52	Weight Gain
Height (in cm)	152	Healthy
Quit		

Figure C. Output

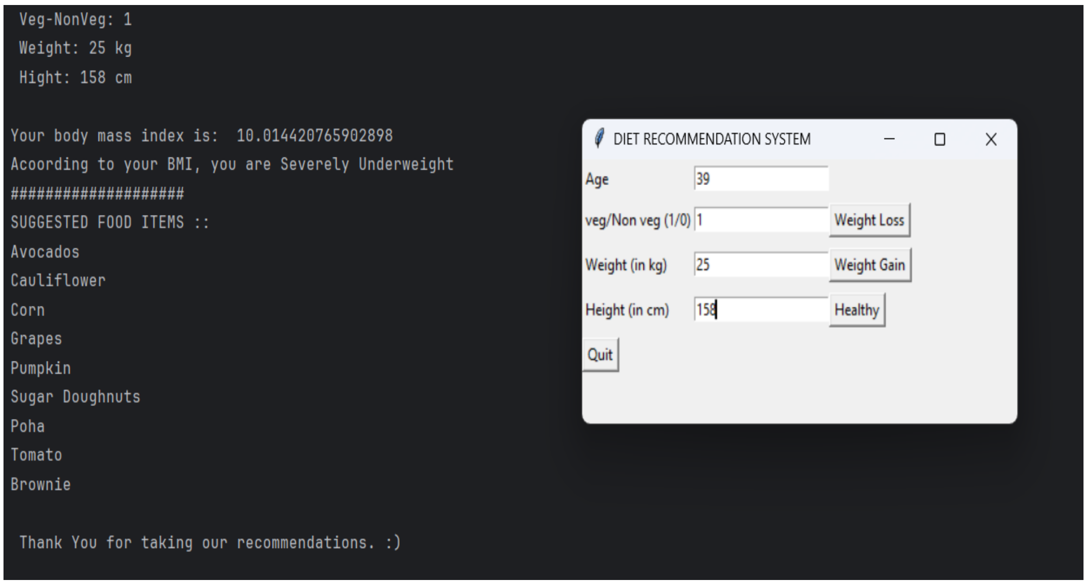


```
Your body mass index is:  21.9671201814059
Acoording to your BMI, you are Healthy
####################
SUGGESTED FOOD ITEMS ::
Avocados
Cauliflower
Corn
Grapes
Pumpkin
Sugar Doughnuts
Poha
Tomato
Brownie

 Thank You for taking our recommendations. :)
```

DIET RECOMMENDATION SYSTEM

Age	45	
veg/Non veg (1/0)	0	Weight Loss
Weight (in kg)	62	Weight Gain
Height (in cm)	168	Healthy
Quit		

Figure D. Output



```
 Veg-NonVeg: 1
 Weight: 25 kg
 Hight: 158 cm

Your body mass index is:  10.014420765902898
Acoording to your BMI, you are Severely Underweight
####################
SUGGESTED FOOD ITEMS ::
Avocados
Cauliflower
Corn
Grapes
Pumpkin
Sugar Doughnuts
Poha
Tomato
Brownie

 Thank You for taking our recommendations. :)
```

DIET RECOMMENDATION SYSTEM

Age 39

veg/Non veg (1/0) 1

Weight (in kg) 25

Height (in cm) 158

Figure E. Output

REFERENCES

[1]. “Institute of Medicine. (2006). Dietary reference intakes: The essential guide to nutrient requirements. National Academies Press.”

[2]. “Khandelwal, S., Reddy, K. S., & Ebrahim, S. (2017). Nutrition in India: From research to practice. Global Heart, 12(2), 145–150.”

[4]. “Institute of Medicine. (2006). Dietary reference intakes: The essential guide to nutrient requirements. National Academies Press.”

[5]. “Khandelwal, S., Reddy, K. S., & Ebrahim, S. (2017). Nutrition in India: From research to practice. Global Heart, 12(2), 145–150.”

[6]. “Lee, S., Kim, D., & Cho, Y. (2020). An AI-based food recommendation system for diabetic patients. Healthcare Informatics Research, 26(2), 89–98.”