

# **ENHANCED PREDICTIVE LEARNING APPROCHES FOR PERSONALIZED DIET RECOMMENDATION SYSTEM IN HEALTHCARE**

**A PROJECT REPORT**

*Submitted by*

<b>ABHISHEK VERMA</b>	<b>[Reg No: RA1611008010402]</b>
<b>MOHNISH RAVAL</b>	<b>[Reg No: RA1611008010453]</b>
<b>TANYA OJHA</b>	<b>[Reg No: RA1611008010525]</b>
<b>UDIT SAVLA</b>	<b>[Reg No: RA1611008010565]</b>

*Under the Guidance of*

**V. NALLARASAN**

(Assistant Professor, Department of Information Technology)

*In partial fulfillment of the Requirements for the Degree  
Of*

**BACHELOR OF TECHNOLOGY**



**DEPARTMENT OF INFORMATION TECHNOLOGY  
FACULTY OF ENGINEERING AND TECHNOLOGY  
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY  
KATTANKULATHUR – 603203**

**MAY 2020**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY  
KATTANKULATHUR-603203**

**BONAFIDE CERTIFICATE**

Certified that this project report titled **“ENHANCED PREDICTIVE LEARNING APPROCHES FOR PERSONALIZED DIET RECOMMENDATION SYSTEM IN HEALTHCARE”** is the bonafide work of **“ABHISHEK VERMA[RegNo: RA1611008010402], MOHNISH RAVAL[RegNo:RA1611008010453], TANYA OJHA [Reg No: RA1611008010525], UDIT SAVLA [RegNo:RA1611008010565]**, who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

**V. NALLARASAN**  
**GUIDE**  
Assistant Professor  
Dept. of Information Technology

**Dr. G. VADIVU**  
**HEAD OF THE DEPARTMENT**  
Dept. of Information Technology

Signature of Internal Examiner

Signature of External Examiner

## **ABSTRACT**

In this modern world various people suffer from different types of diseases and illnesses. It is generally very difficult to suggest a diet as quickly as possible. Most people have a dire need to Lose Weight, Gain Weight or stay Healthy. Time has also become a possible constraint. The project makes use of a dataset which contains various nutrients in the correct amount. In the wake of the situation, we have tried to develop a program that recommends diet to the people. The items recommended are limited to three categories: Weight Loss, Weight Gain and Healthy category. Our project uses Machine Learning Algorithms named K-Means Clustering for clustering the data and Random Forest Classification to classify according to the categories listed. To predict the food items the Diet Recommendation System uses user inputs from a Graphical User Interface including age, height, weight, vegetarian or non-vegetarian food and selecting the above three categories. The working prototype of the Diet Recommendation System lists a set of food items as per the user inputs. The module uses the weight and height to calculate the body Mass Index (BMI) of the user and based on the preference of the kind of diet he wants the Recommendation System predicts the list of food items. The various kinds of nutrients and their calories are taken into consideration along with other relevant details like Fats, Proteins, Iron, Calcium, Sodium, Potassium, Carbohydrates, Fiber, Vitamin D and Sugar. A diverse range of food items are considered like major Vegetarian and Non- Vegetarian foods. Carbohydras, Fats and Nutrients are the major contributors that are considered to make a food.

## **ACKNOWLEDGEMENT**

We as a team have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals. We would like to extend our sincere thanks to all of them. We are highly indebted to Mr. V. Nallarasan for his guidance and constant supervision as well as for providing necessary information regarding the project & also for his support in completing the project. I would like to express my special gratitude and thanks to the panel members Mrs. K. Nimala and Mrs. D. Saveetha for giving me such attention and time and guiding us throughout the tenure of project execution. Thanks to all my colleague in developing the project and to people who have willingly helped me out with their abilities.

**Author**

# TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	3
	LIST OF FIGURES	8
	LIST OF ABBREVIATIONS	9
1.	INTRODUCTION	
1.1	OVERVIEW	10
1.2	EVOLUTION OF HUMAN DIET	11
1.3	BACKGROUND AND MOTIVATION	11
1.4	TYPES OF DIET PLAN	12
1.5	PROBLEM STATEMENT	12
1.6	RESEARCH OBJECTIVES	13
1.7	ORGANIZATION OF THESIS	14
2.	LITERATURE STUDY	
2.1	AHP METHOD FOR DIET RECOMMENDATION	15
2.2	FUZZY APPROACH FOR DIET RECOMMENDATION	16
2.3	PERSONALIZED EXPERT RECOMMENDATION SYSTEM FOR OPTIMIZED NUTRITION	17
2.4	IoMT ASSISTED RECOMMENDATION SYSTEM	18
2.5	FOOD RECOMMENDATION SYSTEM CONSIDERING NUTRITION AND USER PREFERENCES	19
2.6	NUTRIGENOMICS BASED PERSONALIZED NUTRITION SOLUTION	20
2.7	TYPE-2 FUZZY LOGIC FOR DIABETIC PATIENTS DIET RECOMMENDATION	20
2.8	DIETOS: DIET ORGANISER SYSTEM	21
2.9	DASH: DIET RECOMMENDATION FOR HYPERTENSIVE PATIENTS	22
3.	PROPOSED METHODOLOGY	
3.1	FUNCTIONAL REQUIREMENTS	23
	3.1.1 GRAPHICAL USER INTERFACE	23

	3.1.2 WEB APPLICATION	23
	3.1.3 TREND ANALYSIS	23
	3.1.4 USER ACCOUNT AND PREFERENCE	23
	3.1.5 INTEREST PREDICTION	24
3.2	NON-FUNCTIONAL REQUIREMENTS	24
	3.2.1 PERFORMANCE REQUIREMENTS	24
	3.2.2 SAFETY REQUIREMENTS	24
	3.2.3 SECURITY REQUIREMENTS	24
	3.2.4 SOFTWARE REQUIREMENTS	25
	3.2.5 HARDWARE REQUIREMENTS	25
3.3	OBJECTIVE	25
3.4	DATASET	26
3.5	ARCHITECTURAL DIAGRAM	28
4.	IMPLEMENTATION	30
4.1	MODULES	30
4.2	DATA FLOW DIAGRAM	31
4.3	TOOLS	32
4.4	PROCEDURE	32
	4.4.1 GRAPHICAL USER INTERFACE	32
	4.4.2 DATASET	33
	4.4.3 MACHINE LEARNING	34
	4.4.3.1 K-MEANS CLUSTERING ALGORITHM	
	4.4.3.2 RANDOM FOREST CLASSIFICATION ALGORITHM	
5.	CODING AND TESTING	
5.1	GUI INTERFACE	40
5.2	SEGREGATION OF FOOD ITEMS INTO MEALS	41
5.3	AGE AND BMI CONDITION	42
5.4	K-MEANS CLUSTERING FOR ITEMS	43
5.5	PROCESSING OF AND AND BMI INDEX	44
5.6	RANDOM FOREST CLASSIFIER	44
5.7	WEIGHT GAIN, WEIGHT LOSS AND HEALTY	45
5.8	DISPLAY OF FOOD ITEMS	45
6.	CONCLUSION	46

<b>7.</b>	<b>FUTURE ENHANCEMENT</b>	<b>47</b>
<b>8.</b>	<b>REFERENCES</b>	<b>48</b>
	<b>APPENDIX</b>	<b>50</b>
	<b>PAPER PUBLICATION STATUS</b>	<b>58</b>
	<b>PLAGARISM REPORT</b>	<b>60</b>

## LIST OF FIGURES

1.	Balanced Diet Nutrition Proportion	9
2.	Evolution of food with proportion	11
3.	Diet Plan Chart	12
4.	A general Food Recommendation System	13
5.	Dataset snippet – 1	26
6.	Dataset snippet – 2	27
7.	Dataset snippet – 3	27
8.	Architectural Diagram	28
9.	GUI Interface	29
10.	Data Flow Diagram	31
11.	GUI Interface – 2	32
12.	Categorization of food items in various meals	33
13.	BMI index values	34
14.	K-Means Clustering	38
15.	Random Forest Classifier	39
16.	User-System Interface	40
17.	Segregating Items into Breakfast, Lunch and Dinner	41
18.	Segregated Items Analysis	41
19.	BMI Conditions	42
20.	Display BMI	42
21.	Depicting K-Means for Dinner Items	43
22.	K-Means Analysis	43
23.	Graph Plotting	43
24.	Combining Age and BMI	44
25.	Random Forest	44
26.	Prediction Analysis	45
27.	Weight Loss, Weight Gain, Healthy	45
28.	Suggest Food Items	45
29.	List of Recommended Foods	45



## **ABBREVIATIONS**

<b>AHP</b>	<b>Analytic Hierarchy Process</b>
<b>IoMT</b>	<b>Internet of Medical Things</b>
<b>DIETOS</b>	<b>Diet Organizer System</b>
<b>DASH</b>	<b>Diet Recommendation for Hypertensive Patients</b>
<b>GUI</b>	<b>Graphical User Interface</b>
<b>LSTM</b>	<b>Long Short Term Memory</b>
<b>DNA</b>	<b>Deoxyribonucleic Acid</b>

# CHAPTER 1

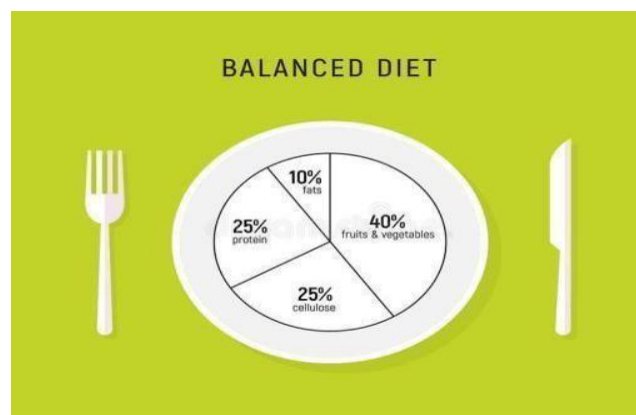
## INTRODUCTION

### 1.1 OVERVIEW

What should I eat ? When should I eat ? How often should I eat ? How much should I eat ? These are some of the questions which arise in the minds of the people. A healthy and balanced diet is a key element in everyone's life. A balanced diet is one which contains sufficient amounts of all the nutrients namely carbohydrates , fats , proteins , vitamins , minerals , sugar etc which are necessary for keeping a person in perfect health.

'Dosha' , a Sanskrit word which means the root of a body helps us to infer that if the Dosha of our body i.e the root causes is disturbed then it may result in a number of diseases. Thus, maintaining a healthy diet is essential for everyone.

A diet recommendation system using machine learning algorithms has been proposed in this project which will suggest food items to the user from a predefined dataset depending upon user preferences for healthy food or food for weight loss or food for weight gain.

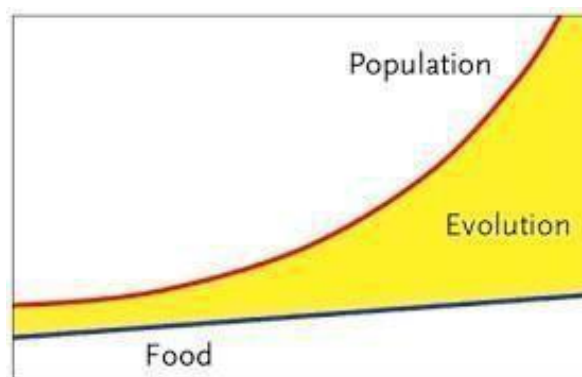


**Fig 1 : Balanced Diet Nutrient Proportion**

## 1.2 EVOLUTION OF HUMAN DIET

Through social development and changes in living space and biology, there have been various significant dietary evolution in human advancement, including meat eating, cooking, and those related with plant and animal hunting.

Right back from the Stone Age when it is said people were more prone to eating meat by hunting, with slow evolution to the Modern World where people have a vast variety of options for food. Diet evolves on several parameters like nutrition, availability, disease, technology. Examining the evolution of diet might help in finding the optimal solution for diet recommendation. The changing diet has also led to morphological changes in the human body. In this technological era, the diet is being suggested by machine learning models that are being developed after analyzing large sets of food data. With numerous options available for food, the diet of a person can be efficiently recommended based upon his requirements.



**Fig 2: Evolution of food with population**

## 1.3 BACKGROUND AND MOTIVATION

Considering the busy life of people in the present world, it results in situations when people tend to skip proper meals or have food at irregular intervals. In such a scenario, it becomes essential for someone to provide them with a proper diet plan which needs to be strictly adhered to.

For the same, people have personal dieticians which provide them with a proper food chart regularly to keep them fit. However, it is not possible for humans to handle many cases and analyze all the elements which are needed to be considered for providing a healthy and balanced diet. Keeping this in mind, an idea struck among the researchers to develop certain systems which may implement machine learning algorithms to gather the user data such as

age, weight, height , diseases etc. and analyze them to provide a menu of food items to the user for the proper meals at the right time. Studying such systems and considering the importance of it, we carried on with diet recommendation.

#### 1.4 TYPES OF DIET PLAN

The need of diet has increased sharply such that it resulted in creation of various types of diet plans namely Keto Diet, Paleo Diet, Vegan Diet , Low Carbs Diet , Low Fat Diet etc. Each diet plans have certain benefits which can be used by people for weight gain or Weight loss or to remain fit.



**Fig 3 : Diet Plan Chart**

#### 1.5 PROBLEM STATEMENT

Making decisions about what to eat is a major problem in our everyday lives due to a wide variety of ingredients, culinary styles, ethnicities, cultures, and personal tastes. Choosing the right dish at the right time seems to be a very difficult task.

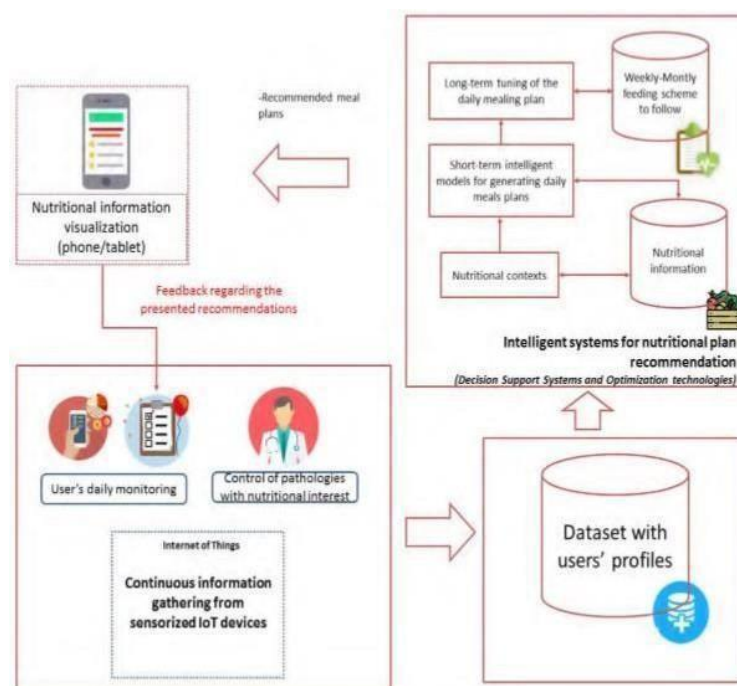
Today, many diseases that were previously thought as hereditary are now shown to be seen connected to biological dis-function related to nutrition. Although being healthy and eating better is something the vast majority of the population want, doing so usually requires great effort and organization.

Thus, this project proposes an integration of machine learning algorithms to recommend the right food at the right time and with the right nutrition, calories, fat etc. here, we are using three attributes of food nutrition details, a person's physical details and time.

## 1.6 RESEARCH OBJECTIVES

By studying and analyzing the research paper, an idea was grasped about the various machine learning algorithms which have been implemented in various ways to recommend diet to the user and which algorithms are more accurate and precise than the other.

It briefed us about the idea of various elements such as age, gender, weight, height etc. which are necessary for considering in a person's profile for recommendation. Lack of this elements may reduce the efficiency of the system.



**Fig 4 : A general Food Recommendation System**

A general food recommendation system consisting of information gathering layer, user profile dataset, intelligent systems layer and an end user interface.

The system shows that it user data is collected and analyzed , the machine learning algorithms applied will be used to predict the diet plan and to make the application to be useful to the user mobile or web applications can be prepared to increase human-system interaction.

## **1.7 ORGANIZATION OF THESIS**

The organization of the latter takes place in the following steps:

1. Writing Literature review on the existing system
2. Collection of dataset
3. Pre-processing
4. Data Cleaning
5. Training the data
6. Results

## CHAPTER 2

### LITERATURE STUDY

#### 2.1 DIET RECOMMENDATION TO PATIENTS USING AHP METHOD-2019:

**Author(s):** Sakshi Singh, Sanjay Kumar Dubey

The problem of malnutrition is of great concern in today's world. It can lead to various diseases which can be cured by a balanced diet in its initial stage. Marasmus - a disease which occurs due to malnutrition and which methods can be used to provide a diet plan has been discussed in this paper [1].

Marasmus is a disease which can occur due to lack of carbohydrates, fats, proteins, lipids, glycogen and other essential nutrients. When these nutrients are in deficient amount in our body, the stored nutrients are consumed to provide energy to the body and slowly the body starts consuming muscles and tissues energy as well. The end results of this include underweight, loss of fat mass and muscle mass, dehydration and some other symptoms. It was found that by giving a decent eating routine containing all the essential supplements, for example, proteins, starches and so on, marasmus can be dealt with. It utilized AHP which helps in deciding the decent eating regimen which has all the essential supplements in the right extents and finally to approve the outcomes it utilized the Fuzzy Logic to make comparisons in the result.

The D. Rao et al. referenced in their article about lack of supplements. The lack of healthy sustenance results in predominated development and delicate insusceptible framework [2]. The article portrays an exhaustive audit of an examination quickly referencing the different strategies for the developing nations in evacuating the lack of healthy sustenance. A. Thapar and M. Goyal approached with a fuzzy method to analyze whether a newborn child is experiencing malnutrition [3].

The AHP method demonstrated made use of determining the main factors and the diet choices as prescribed by the dietician. After the analysis, a pairwise comparison matrix is developed to determine the Consistency Index and Ratio. Calculating these values, gave an

idea about the ranking for various diets as which are more definite. The analysis result is confirmed by performing the fuzzy logic.

It made use of two techniques and gave an idea about how diet prediction can be provided but a more detailed and more efficient methodologies can be implemented to provide a more balanced diet.

## **2.2 DIET RECOMMENDATION USING FUZZY APPROACH-2019:**

**Author(s):** Arushi Singh, Nandini Kashyap, Rakesh Garg

This paper describes how the human population lacks focus on diet and ends up having a dynamic eating routine with no appropriate food items and nutrients. To facilitate with a proper diet, a fuzzy methodology is provided [4].

In today's busy schedule, people tend to skip proper meals at proper times. It is important to ensure that the nutrients are perceived by the body in the right proportion and regularly, lack of which the person ends up with various illnesses starting with temperature, joint pains etc and ending up with serious diseases. Thus, it is necessary for a person to have a balanced diet. People tend to consult dieticians for providing a diet chart to them. However, it is considered that dieticians may tend to forget some parameters for consideration. Hence, a fuzzy logic method was implemented which considers a number of human parameters essential for diet plan depending upon the age group, health issues and other factors.

Fuzzy logic proposed makes use of input, output and a function providing values between 0 and 1. This method was used because the diet recommendation data is highly uncertain and vast. The inputs used were age group and nutrition which was fed into the fuzzy ontology and the output provided the percentage of the nutrition which should be received by the person per day.

As indicated by the test results, the suggested diet is more precise than the eating routine taken from the dietician. The drawbacks found for providing a diet included lack of food items list which can be also suggested to the user instead of just the nutrition proportion. Providing a list of food items for different meals can prove to be more efficient for a healthy diet.



### **2.3 PERSONALIZED EXPERT RECOMMENDATION SYSTEM FOR OPTIMIZED NUTRITION-2018:**

**Author(s)** : Chih-Han Chen, Maria Karvela, Mohammadreza Sohbati, Thaksin Shinawatra, Christofer Toumazou

Studies have found strong correlation between genetic information of a person to the person's personalized diet. It is believed that all particles associated with human digestion are controlled straightforwardly or by implication of genes and in this manner the soundness of people can be enhanced through customized dietary guidance. To achieve what kind of system can be developed is discussed in this paper [5].

Nowadays, people conduct tests to gather detailed information about their genes so that they can plan their diet accordingly. Consequently, a system which would take into account/correlate the grocery product information and genotypic information of a person was proposed. However, the fundamental obstruction of associating hereditary data to a diet is the multifaceted nature of information and the versatility of the applied frameworks.

The focus was to perform the operations of categorization and analysis of data and eventually with a recommendation of list of products. The system architecture was so developed that it would accept new product information including the nutritional values. The block was then trained with the new values and receives updates from the state machine block and stores the trained data. During training, food products are classified into different categories. The block sets up a threshold for nutrition using genetic algorithms. Ultimately, the decision system would recommend a suggestion of products based on phenotype of personal data, the categories of products and nutritional values.

DNN model was applied to accomplish the data categorization and the capacity of scaling of the framework with obscure information is accomplished via word embedding. Genetic Algorithm was used for providing the recommendation. The genetic information of a person is collected from various databases to provide an efficient system.

The paper utilized methodologies which considered the nutritional values of the items but the ingredients of the food products were made ignorant which is also an important factor in making a decision for an effective diet plan.

## **2.4 IoMT ASSISTED PATIENT-DIET RECOMMENDATION SYSTEM-2020:**

**Author(s):** Celestine Iwendi, Suleman Khan, Joseph Henry Anajemba, Ali Kashif Bashir

The need to increase the efficiency of recommender systems to predict the food items to patients resulted in implementation of machine and deep learning algorithms which is discussed in this paper [6].

Healthy diet prescribed by dieticians or artificial system automated diet on cloud servers can help to improve the life of a person; free from stress and diseases. However, the efficiency of such a system can be improved by training and implementing machine and deep learning algorithms namely Naive Bayes , Logistic Regression , Recurrent Neural Network , Multilayer Perceptron , Long Short-Term Memory.

The dataset was collected of some patients with details about their diseases, age, gender, nutrient levels, weight and height. On applying the machine and deep learning algorithms for it, it was found that LSTM provided more accuracy than other algorithms. From the dataset the essential features were analyzed using random forest classifier.

Specialists have found that a healthy diet is a good alternative to cure patients with diseases [7-8]. Researchers found that people are prone more to medicines rather than a healthy diet, the nutritional intake is not sufficient for them and to solve this issue a cloud based system was proposed. In this paper key points involve the usage of machine and deep learning algorithms. Provide an idea about how the model works under the items and patient ailment specifications. Study the various mechanisms for deeper understanding of patients' diets.

The entire process was carried out in five phases: Dataset Collection, Analysis of Optimal Features, Training and Testing, Evaluation. For the purpose, dataset was divided into training (70%) and testing (30%) sets. As an end result, it was concluded that LSTM has higher accuracy among all the other methods.

## **2.5 Food Recommendation System Considering Nutrition and User Preferences - 2019**

**Author(s):** Raciél Yera Toledo, Ahmad A. Alzahrani, Luis Martinez

Health issues have always been a concern for the majority of the population. Researchers are implementing various methods for providing an efficient personalized diet suggestion. This paper deals with basic recommendations along with laying emphasis on nutrition and user preferences as well [9]. Research challenges involve inquiring about difficulties with the assortment of client data, the social occasion of healthful data from nourishments and plans, and the changing of eating practices.

For food recommendation, various factors such as personal attributes, lifestyle, needs and preferences, budget etc. are involved. Nutrition can vary from people to people, from healthy people to patients, vulnerable groups, allergic people and so on. Personal information can enhance the accuracy of menu generated frameworks. The focus was on two important points: building information model which can be obtained through questionnaires, flowcharts and processing nutritional information.

The methodology initiates with preparation of food profiles which have about 600 food items and about 20 nutrients. A template was also prepared for classifying the food depending upon breakfast, lunch and dinner. This acted as a decision table wherein the foods suggested are provided as an alternative depending upon the nutrients.

AHP Sort based pre filtering stage was used to classify important and unimportant food items. AHP Sort behaves as a multi-criteria decision analysis which helps in making decisions depending upon various alternatives. Furthermore, an optimization based approach for menu suggestion was implemented so that it focuses on user preferences also. It is dependent upon probabilistic research which recommends food on the basis as which food item is frequently consumed by people. A case study was carried on to discuss the efficiency of the system and improvements required.

This paper lacks using food history of consumers which can also be used for improving the recommendation, involving recipes of food items or group recommendations.

## **2.6    Personalized Nutrition Solution based on Nutrigenomics - 2019:**

**Author(s):** Jitao Yang

This paper brings in the concept of Nutrigenomics and how it is related to the diet of a person [10].

Nutrigenomics refers to the impact of the diet consumption on the DNA of the person. Genome (a set of DNAs) consists of protein genes. These genes are responsible for maintaining the health of a person. Any infected gene can have an impact on the health of the person. By analyzing the variation in genes, various algorithms can be implemented to provide the nutrition details to a person. The DNA sequence will be studied by generic algorithms. Also considering lifestyle and other attributes, the application will provide a nutrition report.

A mobile application is developed which will take in the genetic information of the person and depending upon these it will suggest a menu to a person. The food items suggested can be analyzed with fine details of weight or nutrient content and other factors. Also recipes for various foods are provided which can be utilized for an efficient diet.

The paper just brought in the relation between a person's diet and his/her genes but lacked the concept of implementing any algorithms which can be used for the same. Also, the algorithm used in the application developed could have been explained for a better understanding. It however did not consider various other factors such as age, height, weight etc. of a person which are equally important for a diet.

## **2.7    Type-2 Fuzzy Logic for Diabetic-Patients Diet Recommendation - 2018**

**Author(s):** Heba Abdelgader Mohammed, Hani Hagra

This paper discusses the Fuzzy Logic method for recommending diet to diabetic patients [11].

The number of diabetic patients are increasing at a faster rate in the world. It is becoming difficult for the doctors to take care of each diabetic patient and prevent them from attaining complications. Eventually, a system was proposed so that an efficient diet can be recommended to such people so that it becomes easy for the doctors to handle them.

Diabetes occurs when the glucose level in our blood cells increases to a high level. It is necessary to control the glucose level in the body for which one of the basic method is an efficient diet which can be suggested such that the sugar level in the body is maintained.

The proposed philosophy to build up this framework is to utilize Type 2 Fuzzy Logic Systems which have been attributed with giving a strategy to structure strong frameworks that can fight with the vulnerability, noise, uncertainties and imprecision ascribed to true situations and applications. A fuzzy system can handle both linguistic and numerical data. Type-2 fuzzy system uses a member function in which element values lies between 0 and 1.

Type -2 fuzzy system was proposed so that it can handle uncertainties such as uncertainties involved in data collection and analysis from diabetic patients, uncertainties of behavior or physical exercise of the patient, different kinds of opinions from different doctors or dieticians and people.

The paper lacked the comparison between other algorithms which could have been implemented as well. It was specific for diabetes patients and not in general.

## **2.8 DIETOS: Diet Organiser System - 2016**

**Author(s):** Agapito G., Calabrese B., Guzzi P. H., Cannataro M., Simeoni M., Car'e I., Lamprinoudi T., Fuiano G., Pujia A.

This paper deals with a web-based application called DIETOS which provides the prototype and the flow in which diet can be recommended to people [12].

The application makes usage of health profiles of the user by collection the information through random questionnaires. Also the questionnaire is so prepared that it is adaptive depending upon the user's answer to different questions. The health profile of the user is prepared. The database involves Calbrain foods and the system can also provide information about which foods can be allergic or harmful to the user depending upon his health status.

It consists of the following modules: User Profile, Reminder, History, Security, Foods Filter. The User Profile module is concerned with the collection of the health information of the person along with other attributes. Reminder helps the person updated with the food menu and that he/she needs to regularly update the profile to increase efficiency. History keeps the records of all the updates made. Security is necessary because the details related to health is

confidential to the person and hence the details collected are encrypted. Foods Filter basically filters out the food and then provides suggestions to the user depending upon the profile created by him.

The paper lacks the list of food items other than Calabrian foods. It aims to provide functions for adding food information by specialists themselves. Provide recipes of certain dishes. It overall aims to enhance the interface and functionalities so that more people use it and the work of doctors and dieticians become easy.

## **2.9 DASH Diet Recommendation For hypertensive patients - 2019:**

**Author(s):** Romeshwar Sookrah, Jaysree Deveen Dhowtal, Soulakshmee Devi Nagowah

Hypertension is quite prevalent among the citizens and it may end up with deaths as well. This paper discusses diet recommendations to such patients so that their situation may improve to a great extent [13].

The system used a framework implementing content-based filtering besides machine learning algorithms for diet suggestions to patients depending upon various points such as age, gender, sugar level, BP level, smoking, allergies. The end-product was displayed using a mobile application.

DASH refers to diet methods to limit hypertension among patients. It can be done so by reducing the level of sodium in the person's body. The food list consists mainly of fruits and vegetables with minimal sugar level. For hypertensive patients, salt level is around 1500 mg sodium and for non-hypertensive, it should be 2300 mg sodium.

The mobile application was developed for easy user interaction. The system would collect the user information and store it in a Firebase. Main elements considered for recommendation was age, food preference, allergies, smoking, alcohol, BP level. The system included estimation module: calculating alcohol, smoke and sodium level; food classifier model: implementing ML algorithm Multilayer Perceptron; content based filtering module which recommends the food from the classification made. The dataset was trained (85%) and tested (15%). The system needs to implement the concept for other non-hypertensive patients as well for better approach and usage.

## **CHAPTER 3**

### **PROPOSED METHODOLOGY**

#### **3.1 FUNCTIONAL REQUIREMENTS**

##### **3.1.1 GRAPHICAL USER INTERFACE**

The main graphical user interface will consist of various input fields taking into consideration the user preferences for food like vegetarian/ non-vegetarian food items at Lunch/ Dinner/ Breakfast along with other details like an individual's height, weight and age. The Diet Recommendation System will recommend the list of desired foods based on the fact that whether the user is expecting a weight gain/ weight loss/ healthy diet.

##### **3.1.2 WEB APPLICATION**

The Diet Recommendation System will be a web application. The application will have an interface as mentioned above. The interface will take the user inputs into account to make recommendation of the food items to the user based on the current requirements as stated by the user.

##### **3.1.3 TREND ANALYSIS**

The Diet Recommendation System will analyze the food items being predicted under each category like Weight Gain/ Weight Loss/ Healthy and will maintain a record of the healthy food items under the trending category of healthy food items.

##### **3.1.4 USER ACCOUNT AND PREFERENCE**

Diet Recommendation System will allow users to create an account and specify their interests. The user will have the option of choosing their interests once at the time of account creation and further change it anytime they desire. This makes the platform more versatile and allows it to cater to a wider range of audience.

### **3.1.5 INTEREST PREDICTION**

The final feature which Diet Recommendation System will implement will be tracking the user's interests and then classify them into groups (Weight Gain/ Weight Loss/ Healthy as per the Breakfast/ Lunch/ Dinner timings) and then display more food items related to the preferred group of the user.

## **3.2 NON-FUNCTIONAL REQUIREMENTS**

Non-functional requirements include performance requirements, security requirements and portability requirements which is equivalently important for the proper functioning of the system.

### **3.2.1 PERFORMANCE REQUIREMENTS**

To improve the performance of the software it needs to be executed with a high internet speed so that no issues are observed in the existing working system. Various other measures needs to be taken care of like the storage that of course requires a bigger workspace. As a result there is no performance issues observed and the module created is lightweight and can work on any platform.

### **3.2.2 SAFETY REQUIREMENTS**

In the time of an unexpected damage situation to the application a backup system needs to be established which can take care of software till the instances are revived.

### **3.2.3 SECURITY REQUIREMENTS**

Each and every user's data should be kept secret and will be only accessible through his own credentials in the system which any other user will not be able to access. The software needs to be hosted on cloud server to store all the user data. The rights of the user should be restricted so that whatever the commands that he must be running on the terminal or the interactive script that he is running should not cause any damage to the existing system or the working prototype of the module. This gives a high sense of rating to the system in terms of security measures.



### **3.2.4 SOFTWARE REQUIREMENTS**

Following software have been used for the Diet Recommendation System.

- Operating System: Windows or Linux for its best support and user friendliness.
- GUI is build using Python's Tkinter library.
- Machine Learning Algorithms like K-Means Clustering and Random Forest Classification are used

### **3.2.5 HARDWARE REQUIREMENTS**

Hardware supporting Windows or Linux based operating system.

- Minimum 4GB ram
- Atleast 5 GB of hard disk space
- Windows 7 or 10
- Mac OS X 10.11 or higher,64-bit
- CPU : 2.8GHZ or above
- Graphic card – minimum 512 MB
- GUI is build using Python's Tkinter library.
- Machine Learning Algorithms like K-Means Clustering and Random Forest Classification are used

## **3.3 OBJECTIVE**

Wide variety of ingredients, cultures and personal tastes makes decision about what to eat a great problem. Many diseases that were previously thought as hereditary are now seen to be connected to biological dysfunction related to nutrition. Being healthy and eating better is something the vast majority of the population wants and doing so usually requires great effort. The working prototype accomplishes a Personalized Diet Recommendation System with integration of Machine Learning Algorithms to recommend the right food at right time and with the right nutrition, calories, fat etc. The main objective of this project is to establish

working prototype of a Personalized Diet Recommendation System that can contribute to maintain and improve the eating habits and health of the people.

### 3.4 DATASET

The dataset used to implement this project is a collection of various food items along with the time during which the food needs to be consumed i.e. Dinner/ Breakfast/ Lunch. The dataset also contains information whether a specific food item is veg or non – veg. For every food item the information about the calorie intake, the amount of Fats, Proteins, Iron, Calcium, Sodium, Potassium, Carbohydrates, Fiber, Vitamin D and Sugar. The dataset contains 90 unique items with all the 16 above mentioned attributes which will be used to train the Machine Learning model and make predictions whether the food item or a list of food items is suitable for a user or not to maintain his healthy food habits. The value 0 or indicates if the particular food item successfully satisfies the column criteria or not. The data set is a collection of vital food items in an individual's life, each with all the following attributes like Food items, Breakfast, Lunch, Dinner, Veg, Nov-Veg, Calories, Fats, Proteins, Iron, Calcium, Sodium, Potassium, Carbohydrates, Fiber, Vitamin D and Sugars. The dataset is self-prepared and the values that are taken for each attribute is verified from various official food calorie websites which maintain the correct amount of the measure for each nutrient for a particular unit of measure. In this Diet Recommendation System the values that are taken are considered in units per 100 g. A common standard is maintained for each and every nutrient so that the food value which is being taken into consideration is normalized and when the application makes use of the dataset it considers it in general amount and then makes recommendations for the user.

Food_items	Breakfast	Lunch	Dinner	Veg/Non-Veg	Calories	Fats (gm)	Proteins(g)	Iron(mg)	Calcium(mg)	Sodium(mg)	Potassium(mg)	Carbohydrates (g)	Fibre (gm)	Vitamin D (mcg)	Sugars (gm)
Asparagus Cooked	0	1	1	0	22	0.2	2.4	0.91	23	14	224	4.1	2	0	1.3
Avocados	1	0	0	0	160	15	2	0.55	12	7	485	8.5	6.7	0	0.7
Bananas	1	0	0	0	89	0.3	1.1	0.26	5	1	358	23	2.6	0	12
Bagels made in wheat	0	1	1	0	250	1.5	10	2.76	20	439	165	49	4.1	0	6.1
Berries	1	0	0	0	349	0.4	14	6.8	190	298	77	77	13	0	46
Broccoli	0	1	1	0	25	0.5	3.8	1.27	118	56	343	3.1	2.8	0	0.6
Brown Rice	0	1	1	0	362	2.7	7.5	1.8	33	4	268	76	3.4	0	0
Cauliflower	0	1	1	0	32	0.3	3	0.72	32	259	278	6.3	3.3	0	0
American cheese	1	0	0	0	331	24	20	0.84	497	966	363	8.3	0	0	0
Coffee	1	0	0	0	2	0	0.3	0.02	2	1	50	0.2	0	0	0
Corn	1	1	1	0	97	1.4	3.3	0.55	2	253	3.3	22	2.7	0	7.7
Dark chocolates	0	0	1	0	556	32	5.5	2.13	30	6	502	60	6.5	0	48
Grapes	1	0	0	0	93	2.1	5.6	2.63	363	9	272	17	11	0	6.3
Milk	1	0	1	0	97	6.9	3.8	0.12	169	52	178	5.2	0	0	0
Cashew Nuts	1	0	0	0	553	44	18	6.68	37	12	660	30	3.3	0	5.9
Onions	0	1	1	0	40	0.1	1.1	0.21	23	4	146	9.3	1.7	0	4.2
Orange	1	0	0	0	97	0.2	1.5	0.8	161	3	212	25	11	0	0
Pasta canned with tomat	0	1	1	0	71	0.7	2.2	0.91	13	381	192	14	0.9	0	4
Pears	1	0	0	0	57	0.1	0.4	0.18	9	1	116	15	3.1	0	9.8
Peas	0	1	1	0	81	0.4	5.4	1.47	25	5	244	14	5.7	0	5.7
Protein Powder	1	0	0	0	411	17	46	8.57	500	329	1129	19	7.1	200	5.7
Pumpkin	0	1	1	0	18	0.1	0.7	0.57	15	237	230	4.3	1.1	0	2.1
Tuna Salad	0	1	1	1	187	9.3	16	1	17	402	178	9.4	0	0	0
Tuna Fish	0	0	1	1	184	6.3	30	1.31	10	50	323	0	0	0	0
Peproni Pizza	0	0	1	0	298	14	12	2.14	146	692	199	30	1.8	0	3.2
Cheese Pizza	0	0	1	0	276	11	11	2.47	192	580	170	33	2.1	0	2.5
French Fries	0	1	1	0	289	14	3.5	0.91	17	357	545	37	3.9	0	0.3
Chicken Burzer	0	1	1	1	292	15	18	0.62	13	859	315	20	1.3	0	0

Fig 5: Dataset snippet -1

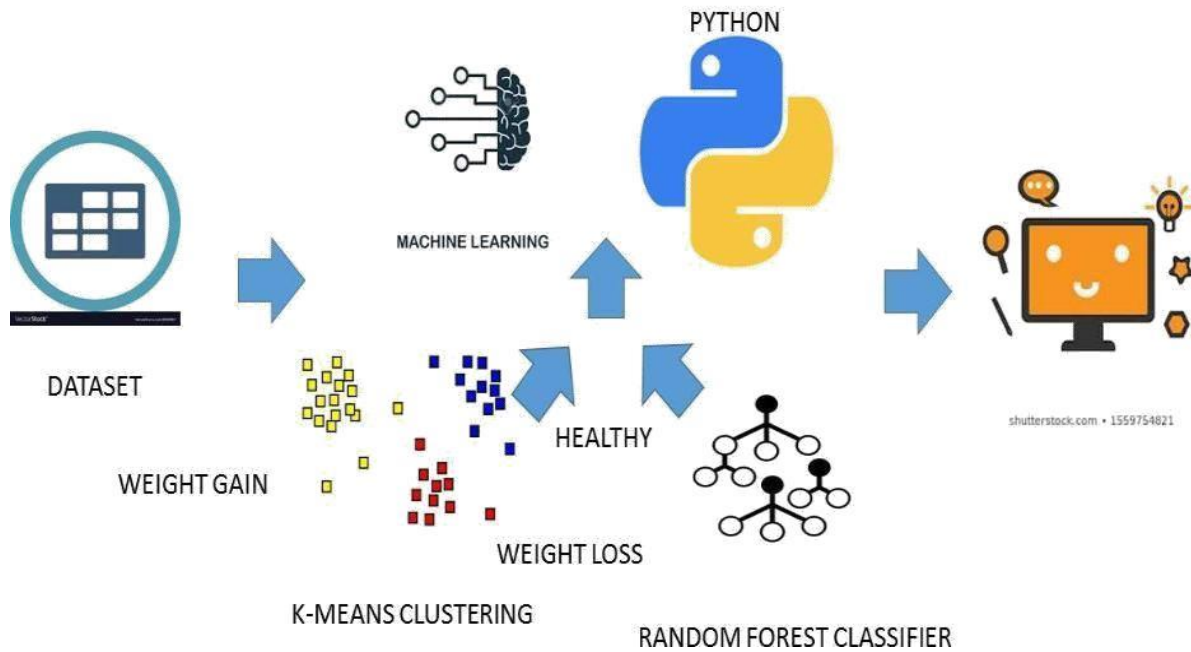
31	Chicken Sandwich	0	1	1	1	257	12	15	1.32	92	605	256	23	1.2	0	5
32	Sugar Doughnuts	0	1	1	0	426	23	5.2	1.06	60	402	102	51	1.5	0	32
33	Chocolate Doughnuts	0	1	1	0	452	25	4.9	4	24	326	201	51	1.9	0	27
34	Pop Corn - Caramel	1	0	1	0	381	1.4	2	0.8	18	286	110	90	2.5	0	65
35	Pop Corn	1	0	1	0	429	9.5	13	2.28	11	490	241	73	14	0	0.5
36	Dosa	1	1	1	0	168	3.7	4.5	8	0.7	94	76	29	0.9	0	0.1
37	Idli	1	1	1	0	156	1.7	5	17.2	4	207	63	30.2	2.1	0	0.74
38	Poha	1	0	0	0	130	1.5	2.6	3.16	1	201	117	26.9	1.1	0	0.5
39	Chappati	0	1	1	0	297	7.5	11	3.01	93	409	266	46	4.9	0	2.7
40	Tomato	1	1	1	0	16	0.2	1.2	0.47	5	42	212	3.2	0.9	0	2.63
41	Yogurt	1	1	1	0	60	4	3.1	0.08	183	70	234	7	0	1	7
42	Brownie	1	0	0	0	407	6.2	4.4	3.81	17	457	51	84	2.9	0	55
43	Noodles	0	1	1	0	108	0.2	1.8	0.14	4	19	4	24	1	0	0
44	Uttapam	1	1	1	0	188	7.2	4.4	24	6.4	522	91	26.4	2.2	0	0.24
45	Bhaji Pav	1	0	0	0	151	2.4	9	37.4	3.4	438	180	29.3	1	0	1.35
46	Dal Makhani	0	1	1	0	109	8.5	2.1	35.2	8.3	243	366	6.3	1.5	0	3.29
47	Almonds	1	0	0	0	579	50	21	3.71	269	1	733	22	13	0	4.4
48	Mushrooms	1	1	1	0	22	0.3	3.1	0.5	3	5	318	3.3	1	7	2
49	Egg Yolk cooked	1	0	0	1	196	15	14	1.89	62	207	152	0.8	0	88	0.4
50	Sweet Potatoes cooked	1	1	0	0	76	0.1	1.4	0.72	27	27	230	18	2.5	0	5.7
51	Boiled Potatoes	1	1	1	0	87	0.1	1.9	0.31	5	240	379	20	2	0	0.9
52	White Rice	0	1	1	0	360	0.6	6.6	4.36	9	1	86	79	1.4	0	0
53	Orange juice	1	0	0	0	45	0.2	0.7	0.2	11	1	200	10	0.2	0	8.4
54	Greek yogurt plain	1	1	1	0	73	1.9	10	0.04	115	34	141	3.9	0	0	3.6
55	Oat Bran Cooked	1	0	0	0	40	0.9	3.2	0.88	10	1	92	11	2.6	0	0
56	Green Tea	1	0	0	0	1	0	0.2	0.02	0	1	0.2	0	0	0	0
57	Chia seeds	1	0	0	0	486	31	17	7.72	631	16	407	42	34	0	0
58	Cottage cheese with vege	0	1	1	0	95	4.2	11	0.1	56	403	86	3	0.1	0	0.4
59	Salmon	0	1	1	1	127	4.4	21	0.38	7	75	366	0	0	435	0
60	Cereals-Corn Flakes	1	0	0	0	384	0.9	5.9	19.4	2	571	107	88	2.7	286	7.8
61	Beans	0	1	1	0	31	0.2	1.8	1.03	37	6	211	7	2.7	0	3.3
62	Lentils	0	1	1	0	101	0.5	8.8	3.1	14	246	284	21	0	0	0
63	Pasta with corn homema	0	1	1	0	126	0.7	2.6	0.25	1	0	31	28	4.8	0	0
64	Tea	1	0	0	0	1	0	0	0.08	2	1	9	0.2	0	0	0

Fig 6: Dataset snippet-2

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
57	Chia seeds	1	0	0	0	486	31	17	7.72	631	16	407	42	34	0	0
58	Cottage cheese with veg	0	1	1	0	95	4.2	11	0.1	56	403	86	3	0.1	0	0.4
59	Salmon	0	1	1	1	127	4.4	21	0.38	7	75	366	0	0	435	0
60	Cereals-Corn Flakes	1	0	0	0	384	0.9	5.9	19.4	2	571	107	88	2.7	286	7.8
61	Beans	0	1	1	0	31	0.2	1.8	1.03	37	6	211	7	2.7	0	3.3
62	Lentils	0	1	1	0	101	0.5	8.8	3.1	14	246	284	21	0	0	0
63	Pasta with corn homema	0	1	1	0	126	0.7	2.6	0.25	1	0	31	28	4.8	0	0
64	Tea	1	0	0	0	1	0	0	0.08	2	1	9	0.2	0	0	0
65	Apples	1	0	0	0	52	0.2	0.3	0.12	6	1	107	14	24	0	10
66	Strawberries	1	0	0	0	32	0.3	0.7	0.41	16	1	153	7.7	2	0	4.9
67	Quinoa	1	1	0	0	120	1.9	4.4	1.49	17	7	172	21	2.8	0	0.9
68	Goat meat	0	1	1	1	109	2.3	21	2.83	13	82	385	57	0	0	0
69	Rabbit meat	0	1	1	1	114	2.3	22	3.2	12	50	378	0	0	0	0
70	Chicken Strips	0	0	1	1	295	15	19	0.72	11	798	334	22	1.1	0	0.3
71	Steak Fries	0	1	1	1	255	13	3.3	0.69	19	43	551	31	3.5	0	0.9
72	Mexican Rice	0	1	1	0	195	4.9	3.6	1.18	39	677	131	34	1.1	0	1.4
73	Fried Shrimp	0	0	1	1	319	20	14	1.54	47	1400	128	21	1.5	0	0.8
74	Spaghetti and meatballs	0	0	1	1	170	8.5	7.8	1.29	45	351	206	16	1.5	0	2
75	Macroni n Cheese	0	1	1	0	194	12	6.5	57	145	375	127	16	1.1	0	2.8
76	Pork cooked	0	0	1	1	297	21	26	1.29	22	73	362	0	0	0	0
77	Bacon cooked	0	0	1	1	146	2.8	28	0.56	7	993	999	1.8	0	0	1.2
78	Nachos	1	1	1	0	350	22	4.3	0.75	63	313	362	35	3.2	0	2.2
79	Chicken Popcorn	0	1	1	1	351	22	18	1.42	32	1140	288	21	1	0	0
80	Turkey cooked	0	1	1	1	203	10	27	1.52	28	78	294	0	0	8	0
81	Oyster cooked	0	1	1	1	159	4	29	4.9	6	81	409	0	0	0	0
82	Beef sticks	1	0	0	1	550	50	22	3.4	68	1531	257	5.4	0	0	0
83	Banana Chips	1	0	0	0	519	34	2.3	1.25	18	6	536	58	7.7	0	35
84	Honey	1	0	1	0	304	0	0.3	0.42	6	4	52	82	0.2	0	82
85	Chocolate Icecream	0	0	1	0	216	11	3.8	0.93	109	76	249	28	1.2	0	25
86	Vanilla Ice cream	0	0	1	0	207	11	3.5	0.09	128	80	199	24	0.7	0	21
87	Strawberry Icecream	0	0	1	0	192	8.4	3.2	0.21	120	60	188	28	0.9	0	0
88	Marshmallows	0	0	1	0	318	0.2	1.8	0.23	3	80	5	81	0.1	0	58
89	Chocolate milk	1	0	0	0	535	30	7.7	2.35	189	79	372	59	3.4	0	52
90	Rice Pudding	0	0	1	0	376	0.1	2.7	1.79	14	366	5	91	0.7	0	0

Fig 7: Dataset snippet-3

### 3.5 ARCHITECTURAL DIAGRAM

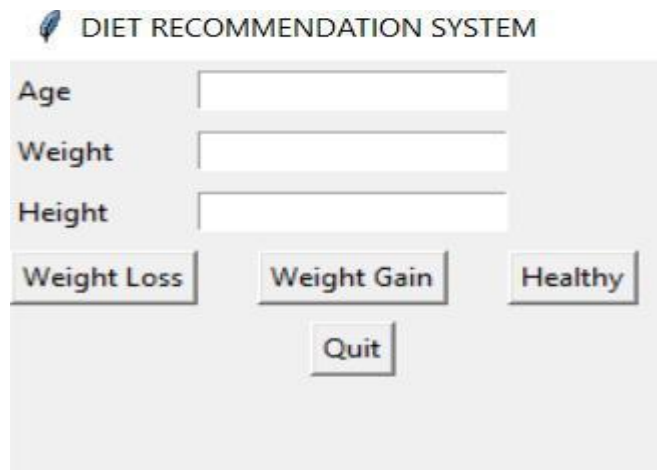


**Fig 8: Architectural Diagram**

The above figure shows the architectural diagram for the working prototype of the Diet Recommendation System. The code is written in Python which is used to implement various Clustering and Classification algorithms in Machine Learning for predicting a proper diet to the user. For training of the system, the initial process involves the segregation of food items depending upon the meal for which they are consumed i.e. Breakfast, Lunch and Dinner. The clustering of various nutrients depending upon which are essential for the weight-loss, weight-gain and healthy is performed. After the clustering is performed, using Random Forest classifier, the nearest food items are predicted which best suits for the appropriate diet. As part of user interface, the inputs needed from the user are Age, Height, Weight and for what the purpose the diet is required. Depending upon it, from the appropriate clustering, specific food items are classified and recommended to the user. The K – Means Clustering Algorithm takes into consideration various attributes and forms a cluster. The new attribute values being taken into account fall into one of the clusters and then the classification algorithms are used to predict the food item list based on the type of inputs given by the user. The input is taken from the Tkinter GUI which is designed using Python. The main graphical user interface will consist of various input fields taking into consideration the user preferences for food like vegetarian/ non– vegetarian food items at Lunch/ Dinner/ Breakfast along with other details like an individual's height, weight and age.

The Diet Recommendation System will recommend the list of desired foods based on the fact that whether the user is expecting a weight gain/ weight loss/ healthy diet.

The GUI for the Diet Recommendation System is depicted below with all the input fields as stated above:



The image shows a graphical user interface (GUI) for a "DIET RECOMMENDATION SYSTEM". At the top, there is a title bar with a small icon and the text "DIET RECOMMENDATION SYSTEM". Below the title bar, there are three input fields labeled "Age", "Weight", and "Height". Each label is to the left of its corresponding input field. Below these input fields, there are three buttons labeled "Weight Loss", "Weight Gain", and "Healthy". These buttons are arranged horizontally. Below these three buttons, there is a single button labeled "Quit" centered horizontally.

**Fig 9: GUI Interface**

# CHAPTER 4

## IMPLEMENTATION

### 4.1 MODULES

The implementation of the Diet Recommendation System is accomplished using various modules namely,

- **Dataset**
- **Machine Learning Algorithms**
  - **K – Means Clustering**

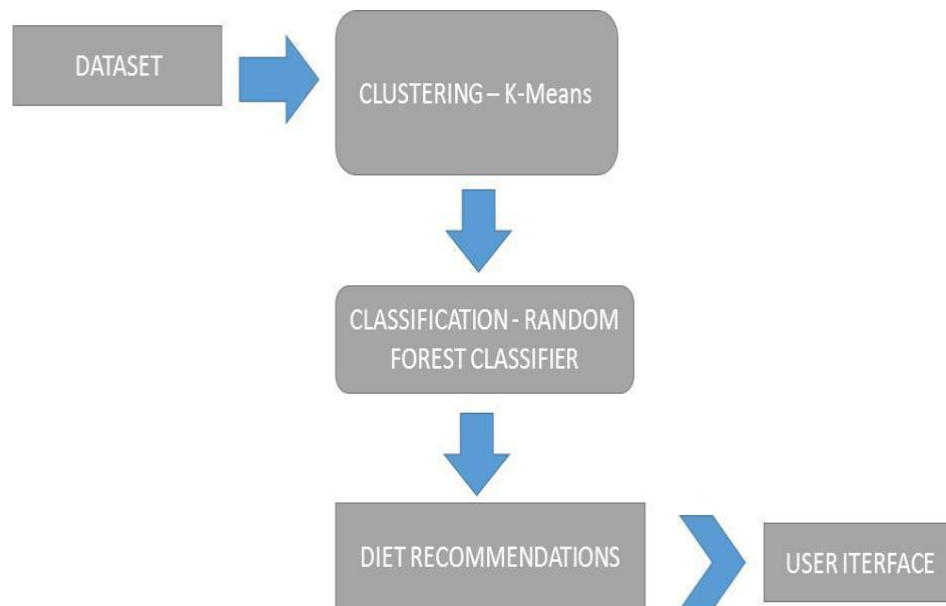
K-means algorithm is an iterative algorithm that tries to partition the dataset into  $K$  pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group. It tries to make the intra- cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

- **Random Forest Classification**

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

- **GUI (using Python's Tkinter Library)**

## 4.2 DATA FLOW DIAGRAM



**Fig 10: Data Flow Diagram**

The data flow in the above working prototype is as such, firstly, the dataset is taken into consideration which is utilized by the K – Means Clustering Algorithm to classify the entire data into clusters of Weight Loss, Weight Gain and Healthy and then training a Random Forest Classifier to make food items list suggestion to the user. After reading the dataset, the entire dataset is classified into Breakfast, Dinner and Lunch. The `to_numpy` function is used to convert the values to the numpy array which can be considered for further manipulation. All the details stored in each of the rows are retrieved in the specified category using the `i_loc` method. A Python's list is created which contains the food items and all the nutrients and then the Transpose of the value stored is stored in another variable. As stated above the process is repeated for the Lunch, Dinner and Breakfast itself. The Diet Recommendation System calculates the Body Mass Index taking into consideration the Height and Weight of the user. An age range is also displayed to state the comparison of the BMI for a specific range group. Based on the BMI calculated the category is specified that the human is underweight, healthy, overweight or severely overweight. Now the application of K – Means Clustering Algorithm comes into play. The number of clusters decided for the module is three namely Weight Gain, Weight Loss and Healthy. All the food items at the time of Breakfast, Lunch and Dinner falls

into any of these categories. We make use of the K- Means Clustering Algorithm after importing the functionalities present in this class which can be achieved using the statement from sklearn. Cluster import K-Means. The data is fitted to the model and then make predictions for the food items. Now we take the data of a specific cluster and divide it into the test and the train dataset which is further used to train the Random Forest Classification model to make recommendations of the food items to the user.

### 4.3 TOOLS

The tools being utilized are Python 3 for making use of the Machine Learning Algorithms and the execution of the Diet Recommendation takes place using a Spyder IDE which makes use of the Tkinter library.

### 4.4 PROCEDURE

#### 4.4.1 GRAPHICAL USER INTERFACE

The implementation of the Diet Recommendation System begins with the user input as accepted from the Graphical User Interface (GUI). The GUI is build using Python's library Tkinter. An object of this library class is created and used to build a GUI. To create a Tkinter based GUI we need to import all its function using the statement from tkinter import \*. As soon as we get all the functions of the tkinter library, we can use it to create and design our GUI. An object main\_win is created of the Tkinter class. The GUI contains four labels and text entry boxes each for Age, Veg/Non-Veg, Weight and Height and align it to a specific position on the screen. Four Buttons stating the Weight Loss, Weight Gain, Healthy and Quit options are made available on the GUI so that the required functions can be called and then the food can be recommended as per the user's inputs.



**Fig 11: GUI**



#### 4.4.2 DATASET

After reading the dataset, the entire dataset is classified into Breakfast, Dinner and Lunch. The `to_numpy` function is used to convert the values to the numpy array which can be considered for further manipulation. All the details stored in each of the rows are retrieved in the specified category using the `i_loc` method. A Python's list is created which contains the food items and all the nutrients and then the Transpose of the value stored is stored in another variable. As stated above the process is repeated for the Lunch, Dinner and Breakfast itself. The Diet Recommendation System calculates the Body Mass Index taking into consideration the Height and Weight of the user. An age range is also displayed to state the comparison of the BMI for a specific range group. Based on the BMI calculated the category is specified that the human is underweight, healthy, overweight or severely overweight.

```
Breakfastdata=data['Breakfast']
BreakfastdataNumpy=Breakfastdata.to_numpy()

Lunchdata=data['Lunch']
LunchdataNumpy=Lunchdata.to_numpy()

Dinnerdata=data['Dinner']
DinnerdataNumpy=Dinnerdata.to_numpy()

Food_itemsdata=data['Food_items']
breakfastfoodseparated=[]
Lunchfoodseparated=[]
Dinnerfoodseparated=[]

breakfastfoodseparatedID=[]
LunchfoodseparatedID=[]
DinnerfoodseparatedID=[]

for i in range(len(Breakfastdata)):
    if BreakfastdataNumpy[i]==1:
        breakfastfoodseparated.append(Food_itemsdata[i])
        breakfastfoodseparatedID.append(i)
    if LunchdataNumpy[i]==1:
        Lunchfoodseparated.append(Food_itemsdata[i])
        LunchfoodseparatedID.append(i)
    if DinnerdataNumpy[i]==1:
        Dinnerfoodseparated.append(Food_itemsdata[i])
        DinnerfoodseparatedID.append(i)
```

**Fig 12 : Categorisation of food items in various meals**

```

#conditions
print("Your body mass index is: ", bmi)
if ( bmi < 16):
    print("severely underweight")
    clbmi=4
elif ( bmi >= 16 and bmi < 18.5):
    print("underweight")
    clbmi=3
elif ( bmi >= 18.5 and bmi < 25):
    print("Healthy")
    clbmi=2
elif ( bmi >= 25 and bmi < 30):
    print("overweight")
    clbmi=1
elif ( bmi >=30):
    print("severely overweight")
    clbmi=0

```

**Fig 13 : BMI index values**

#### **4.4.2 MACHINE LEARNING**

In the Diet Recommendation System there are three main functions Weight Loss, Weight Gain and Healthy that will use the Machine Learning functionalities like K- Means Clustering and Random Forest Classification Algorithm to recommend the list of food to the user. The user defined functions makes use of various in build Python's function used in Machine Learning to provide the actual functionality. We are using various libraries like Pandas, Numpy and Sklearn in general. The Pandas library is used to import the data set so that the machine learning model can make use of it and the list of food items can be predicted to the user. The Numpy is used for values conversion to array so that it can provide the values which can be used to train the Machine Learning model. The Sklearn package has many of the Machine Learning Algorithms. The one which we are using are K- Means Clustering and Random Forest Classification Algorithms.

##### **4.4.2.1 K-MEANS CLUSTERING ALGORITHM**

###### **A. INTRODUCTION**

K-means clustering is a type of unsupervised algorithm, which is employed when you have unlabelled data (i.e., information without defined types or clusters). The objective of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to allot apiece data point to one of K groups constructed on

the features that are delivered. Data points are bundled based on feature resemblance.

The outcomes of the K-means clustering algorithm are:

1. The centroids of the K clusters, which can be used to label novel data
2. Labels for the preparation data (each data point is assigned to a single cluster)

Rather than describing groups before observing at the data, clustering permits you to find and analyse the groups that have made naturally.

Each centroid of a cluster is a assortment of feature values which define the resulting groups. Investigating the centroid feature weights can be used to qualitatively construe what kind of group each cluster signifies.

## **B. BUSINESS USES OF KMEANS**

The K-means clustering algorithm is used to search groups of data which have not been obviously labelled in the data. This can be used to authorize business expectations about what types of groups occur or to recognize unidentified groups in complex data sets. Once the algorithm has been run and the groups are definite, any new data can be easily assigned to the correct group.

- Behavioral segmentation:
- Section by purchase history
- Segment by activities on application, website, or display place
- Define facades based on interests
- Create profiles based on activity monitoring
- Inventory categorization
- Group images

- Detect activity types in motion sensors

### **C. STEPS FOR KMEANS CLUSTERING**

1. Step one: Initialize cluster centers
2. Assign observations to the closest cluster center
3. Revise cluster centers as mean of assigned observations
4. Repeating above 2 and 3 steps until getting accurate clusters

### **D. IMPLEMENTATION**

Now the application of K – Means Clustering Algorithm comes into play. The number of clusters decided for the module is three namely Weight Gain, Weight Loss and Healthy. All the food items at the time of Breakfast, Lunch and Dinner falls into any of these categories.

We make use of the K- Means Clustering Algorithm after importing the functionalities present in this class which can be achieved using the statement from sklearn.Cluster import K-Means. The data is fitted to the model and then make predictions for the food items. Now we take the data of a specific cluster and divide it into the test and the train dataset which is further used to train the Random Forest Classification model to make recommendations of the food items to the user. The same process is repeated for Weight Loss, Weight Gain and Healthy Categories as per the user requirements.

K- Means Clustering Algorithm allows to cluster the data and is a very convenient tool for discovering the groups in your dataset that might not have been predicted earlier. To identify the clusters just by observing the scatter plots as such is not an easy task. To help facilitate it we can use the K- Means Algorithm in Machine Learning. K stands for the number of clusters. In our case the number of clusters is 3. The way this algorithm works is by just selecting the centroids of each clusters, it can be any points in the dataset or any random points in the scatterplot. This step is followed by the assignment of several points to the nearest centroid. To make a cluster one of the approaches can be to divide the entire scatter plot into divisions such that if a line is drawn perpendicular to the line joining the two centroids (in case of 2 clusters) then the entire part is divided into two halves and each point on the line is equidistant

from the two centroids of the clusters. If a point is above the line, it is near to the first centroid

and thus is considered as a part of that cluster. On the other hand if it is in the lower half then it is near to the second centroid and thus forms the part of the second cluster. The Algorithm works on the simple principle of mathematics that states the concept of geometry. This centroid is the temporary centroid and is only for the initial stage of classification of the scattered points. Now comes into role the new and actual centroid that has to be placed in the formation of the cluster. This is calculated based on the center of mass of the points that are falling into the specific cluster. This average value helps in finding the actual centroid. Now after the actual centroid has been found then all the scattered data points needs to be reassigned based on the new changes. If any reassignment takes place then the above step of finding the centroid again has to be repeated and if this is not the case i.e. all the data points remain at the same point exactly then the centroid of the cluster is already decided and the process of finding the clusters is finalized. This can be referred as the intuition behind the K –Means Clustering.

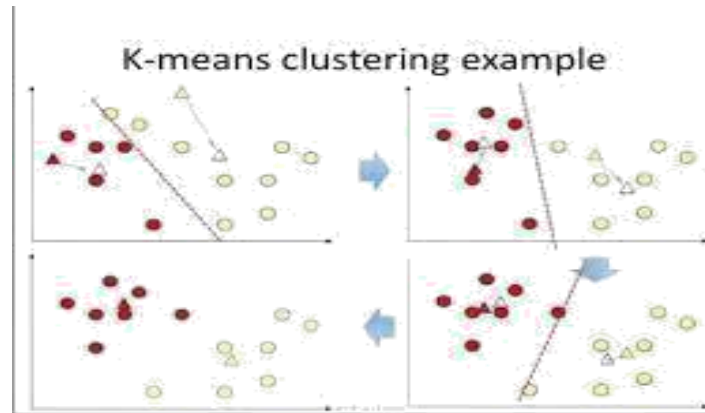
Another effective phenomenon is the Random Initialization Trap to form the clusters in the K – Means Algorithm. This approach considers a different approach to form the clusters. Select at random K points, the centroids (not necessarily from the existing dataset). Then the concept of equidistant lines comes into role and we divide the scatter plots into three parts based on the rule of simple geometry. Resulting into the formation of the final n clusters. Further the centroid is reallocated and the cluster is given a new form. Again the points are reassigned after the formation of clusters with new centroid. If the scatter points are at the same place with respect to the new centroid then no need to reform the entire cluster structure.

The K - Means++ Algorithm is a solution to the common problem when a situation arises about what is a True Cluster and what is False Cluster. The K-Means++ Algorithm solves the conditions above and results in the formation of true clusters.

In K – Means Clustering choosing the right number of clusters is an important decision. In a particular scenario whether 2 or n number of clusters is important can be decided as per the performance of the situation. The formula for deciding the number of clusters is given by:

$$WCSS = \text{Sum of all the points in Cluster 1 distance } (P_i, C_1)^2 + \text{Sum of all the points in Cluster 2 distance } (P_i, C_2)^2 + \text{Sum of all the points in Cluster 3 distance } (P_i, C_3)^2.$$

If we consider the initial situation of one cluster then we have only one centroid in a single cluster. On the other hand if we consider two clusters each having its own centroid then the formula for calculating the WCSS is more optimal.



**Fig 14: K means Clustering**

#### **4.4.2.1 RANDOM FOREST CLASSIFICATION ALGORITHM**

##### **A. INTRODUCTION**

Random Forest is a supervised machine learning algorithm which is widely used for classification problems. The working of this algorithm is based on construction of multiple decision trees at the time of training time and the output produced is the class with highest frequency or mean of the individual trees. Each decision tree constitutes a root node and each internal node of the tree represents an attribute and each leaf node defines class label.

##### **B. WORKING OF RANDOM FOREST**

The working consists of two steps:

###### **1. Creation of RF pseudo code:**

- I. Randomly selecting  $n$  features from total  $m$  features where  $n \ll m$
- II. Calculation of node “i” using the best split from  $n$  features
- III. Splitting the node into child nodes
- IV. Repeating the above steps until  $k$  number of nodes has been reached
- V. Building the forest by repeating step 1 to step 4 for  $n$  trees

2. RF prediction pseudo code:

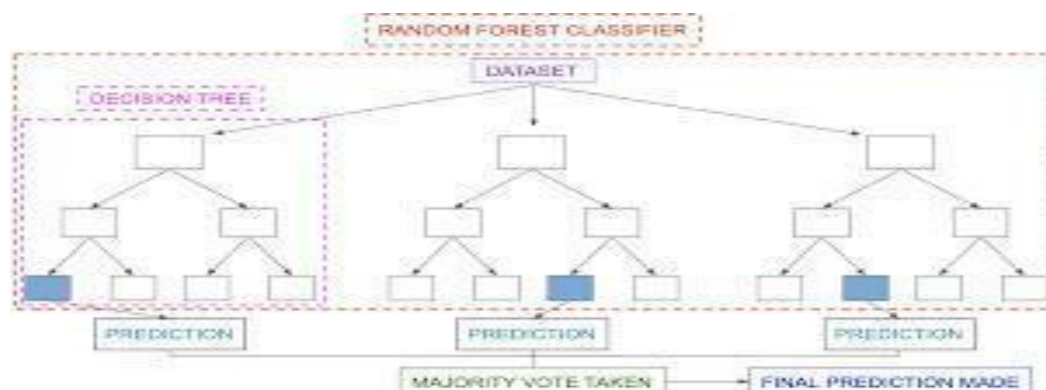
- I. Using the test features and rules of every created decision tree for predicting the result and storing the outcome or target
- II. Calculation of votes for every predicted target
- III. The final prediction is the most voted predicted target



### C. IMPLEMENTATION

The classification Algorithm that we will be using is Random Forest Algorithm. It is based on Decision Trees. Initially, we pick at random K data points in the dataset. Secondly, building the Decision Tree associated to these K data points is done. Choosing the number N tree of trees that has to be build is to be done. Now for every new data point, each one of the new N tree classify the category to which the data point must be assigned, and assign the new data points to the category that wins the majority vote.

The Forest Classifier will classify various food items depending upon the nutrient values into different categories and when the user prompts for a diet, the system would generate food items from different classifiers whichever suits more appropriate depending upon the user profiles.



**Fig 15: Random Forest Classifier**

# CHAPTER 5

## CODING AND TESTING

### 5.1 GUI INTERFACE

```
Label(main_win,text="Age").grid(row=0,column=0,sticky=W,pady=4)
Label(main_win,text="Weight").grid(row=1,column=0,sticky=W,pady=4)
Label(main_win,text="Height").grid(row=2,column=0,sticky=W,pady=4)

e1 = Entry(main_win)
e2 = Entry(main_win)
e3 = Entry(main_win)

e1.grid(row=0, column=1, sticky = W, pady = 2)
e2.grid(row=1, column=1, sticky = W, pady = 2)
e3.grid(row=2, column=1,sticky = W, pady = 2)

Button(main_win,text='Quit',command=main_win.quit).grid(row=4,column=1,sticky="",pady=4)
Button(main_win,text='Weight Loss',command=Weight_Loss).grid(row=3,column=0,sticky="",pady=4)
Button(main_win,text='Weight Gain',command=Weight_Gain).grid(row=3,column=1,sticky="",pady=4)
Button(main_win,text='Healthy',command=Healthy).grid(row=3,column=2,sticky="",pady=4)
main_win.geometry("400x200")
main_win.wm_title("DIET RECOMMENDATION SYSTEM")
#GUIEXECUTION
main_win.mainloop()
```



Fig 16 : User-System Interface



## 5.2 SEGREGATION OF FOOD ITEMS INTO DIFFERENT MEALS

```

Breakfastdata=data['Breakfast']
BreakfastdataNumpy=Breakfastdata.to_numpy()

Lunchdata=data['Lunch']
LunchdataNumpy=Lunchdata.to_numpy()

Dinnerdata=data['Dinner']
DinnerdataNumpy=Dinnerdata.to_numpy()

Food_itemsdata=data['Food_items']
breakfastfoodseparated=[]
Lunchfoodseparated=[]
Dinnerfoodseparated=[]

breakfastfoodseparatedID=[]
LunchfoodseparatedID=[]
DinnerfoodseparatedID=[]

for i in range(len(Breakfastdata)):
    if BreakfastdataNumpy[i]==1:
        breakfastfoodseparated.append(Food_itemsdata[i])
        breakfastfoodseparatedID.append(i)
    if LunchdataNumpy[i]==1:
        Lunchfoodseparated.append(Food_itemsdata[i])
        LunchfoodseparatedID.append(i)
    if DinnerdataNumpy[i]==1:
        Dinnerfoodseparated.append(Food_itemsdata[i])
        DinnerfoodseparatedID.append(i)

```

**Fig 17 : Segregating Items into Breakfast, Lunch and Dinner**

[0, 3, 5, 6, 7, 10, 15, 17, 19, 21, 22, 26, 27, 28, 29, 30, 31, 34, 35, 37, 38, 39, 41, 42, 44, 46, 48, 49, 50, 52, 56, 57, 59, 60, 61, 65, 66, 67, 69, 70, 73, 76, 77, 78, 79]							
	Food_items	Calories	Fats	...	Carbohydrates	Fibre	VitaminD
0	Asparagus Cooked	22	0.2	...	4.1	2	0
3	Bagels made in wheat	250	1.5	...	49	4.1	0
5	Broccoli	25	0.5	...	3.1	2.8	0
6	Brown Rice	362	2.7	...	76	3.4	0
7	Cauliflower	32	0.3	...	6.3	3.3	0
10	Corn	97	1.4	...	22	2.7	0
15	Onions	40	0.1	...	9.3	1.7	0
17	Pasta canned with tomato sauce	71	0.7	...	14	0.9	0
19	Peas	81	0.4	...	14	5.7	0
21	Pumpkin	18	0.1	...	4.3	1.1	0
22	Tuna Salad	187	9.3	...	9.4	0	0
26	French Fries	289	14	...	37	3.9	0
27	Chicken Burger	292	15	...	20	1.3	0
28	Cheese Burger	256	12	...	25	1.4	0

**Fig 18 : Segregated Items Analysis**

### 5.3 AGE AND BMI CONDITION:

```
age=int(e1.get())
weight=float(e2.get())
height=float(e3.get())
bmi = weight/(height**2)
agewiseinp=0

for lp in range (0,80,20):
    test_list=np.arange(lp,lp+20)
    for i in test_list:
        if(i == age):
            print('age is between',str(lp),str(lp+10))
            tr=round(lp/20)
            agecl=round(lp/20)

print("Your body mass index is: ", bmi)
if ( bmi < 16):
    print("severely underweight")
    clbmi=4
elif ( bmi >= 16 and bmi < 18.5):
    print("underweight")
    clbmi=3
elif ( bmi >= 18.5 and bmi < 25):
    print("Healthy")
    clbmi=2
elif ( bmi >= 25 and bmi < 30):
    print("overweight")
    clbmi=1
elif ( bmi >=30):
    print("severely overweight")
    clbmi=0
vall=DinnerfoodseparatedIDdata.describe()
valTog=val1.T
print (valTog.shape)
print (valTog)
DinnerfoodseparatedIDdata=DinnerfoodseparatedIDdata.to_numpy()
LunchfoodseparatedIDdata=LunchfoodseparatedIDdata.to_numpy()
breakfastfoodseparatedIDdata=breakfastfoodseparatedIDdata.to_numpy()
ti=(clbmi+agecl)/2
```

Fig 19 : BMI conditions

```
age is between 20 30
Your body mass index is: 36.111111111111114
severely overweight
```

Fig 20 : Display BMI

## 5.4 K-MEANS CLUSTERING FOR ITEMS

```
Datacalorie=DinnerfoodseparatedIDdata[1:,1:len(DinnerfoodseparatedIDdata)]
#print(Datacalorie)
X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0).fit(X)
print ('## Prediction Result ##')
print(kmeans.labels_)
print (kmeans.predict([Datacalorie[0]]))
XValu=np.arange(0,len(kmeans.labels_))
# fig,axs=plt.subplots(1,1,figsize=(15,5))
# plt.bar(XValu,kmeans.labels_)
dnrlbl=kmeans.labels_
```

Fig 21 : Depicting K-Means for Dinner Items

```
## Prediction Result ##
[1 2 2 2 2 2 2 2 1 2 2 1 2 1 1 1 0 1 1 1 1 1 2 2 1 2 2 2 1 2 2 2 2 1 2
 2 2 2 2 2 0 2 1 0 1 1 2 0 1 0 2 2 2 2 2 2 2 1]
[1]
## Prediction Result ##
[2 0 0 0 0 0 2 0 0 2 2 1 2 2 2 2 0 0 2 0 0 0 2 0 0 0 0 0 2 0 0 0 0 0 0
 0 2 2 2 1 0 0]
## Prediction Result ##
[0 0 1 0 0 0 0 2 0 0 2 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 2 0 0 0 0 0
 1 2 0 2]
```

Fig 22 : K-Means Analysis

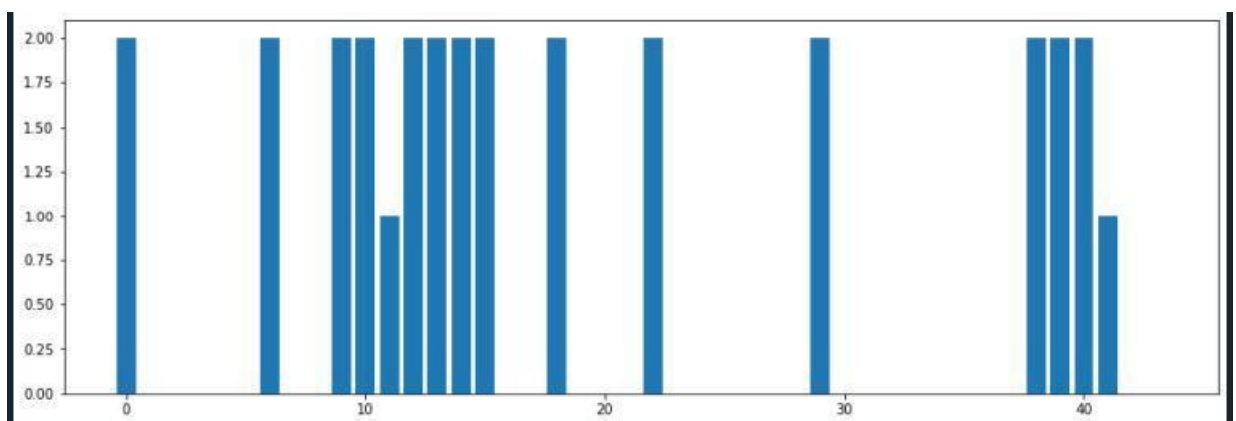


Fig 23 : Graph Plotting



## 5.5 PROCESSING AGE AND BMI TOGETHER USING K-MEANS

```
#age bmi data combining and processed data(kmeans)
dataTog=datafin.T
bmicls=[0,1,2,3,4]
agecls=[0,1,2,3,4]
weightlosscat = dataTog.iloc[[1,2,7,8]]
weightlosscat=weightlosscat.T
weightgaincat= dataTog.iloc[[0,1,2,3,4,7,9,10]]
weightgaincat=weightgaincat.T
healthycat = dataTog.iloc[[1,2,3,4,6,7,9]]
healthycat=healthycat.T
weightlosscatDdata=weightlosscat.to_numpy()
weightgaincatDdata=weightgaincat.to_numpy()
healthycatDdata=healthycat.to_numpy()
weightlosscat=weightlosscatDdata[1:,0:len(weightlosscatDdata)]
weightgaincat=weightgaincatDdata[1:,0:len(weightgaincatDdata)]
healthycat=healthycatDdata[1:,0:len(healthycatDdata)]

#print(weightgaincat)
#print (len(weightlosscat))
weightlossfin=np.zeros((len(weightlosscat)*5,6),dtype=np.float32)
weightgainfin=np.zeros((len(weightgaincat)*5,10),dtype=np.float32)
healthycatfin=np.zeros((len(healthycat)*5,9),dtype=np.float32)
```

Fig 24 : Combining age and BMI

## 5.6 RANDOM FOREST CLASSIFIER

```
#Random Forest
for jj in range(len(weightlosscat)):
    valloc=list(weightlosscat[jj])
    valloc.append(agecl)
    valloc.append(clbmi)
    X_test[jj]=np.array(valloc)*ti
# print (X_test)
# print (len(weightlosscat))
# print (len(X_test))

X_train=weightlossfin# Features
y_train=yt # Labels

# Split dataset into training set and test set
# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) #

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)
```

Fig 25 : Random Forest

```
[0.2  7.  4.25 6.5  0.5  0. ]
```

**Fig 26 : Prediction analysis**

## 5.6 WEIGHT LOSS, WEIGHT GAIN and HEALTHY OPTIONS



DIET RECOMMENDATION SYSTEM

Age

Weight

Height

**Fig 27 : Weightloss, Weightgain, Healthy**

## 5.7 DISPLAY OF FOOD ITEMS

```
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]
```

**Fig 28 : Suggest Food Items**

```
SUGGESTED FOOD ITEMS ::
Cauliflower
Corn
Grapes
Pumpkin
Sugar Doughnuts
Poha
Tomato
Brownie
```

**Fig 29 : List of Recommended Foods**

# **CHAPTER 6**

## **CONCLUSION**

A Diet Recommendation System is implemented with the working functionalities like:

- 5.7.1 Desired food list prediction.
- 5.7.2 Diet list based on weight category.
- 5.7.3 Body Mass Index (BMI) Calculation.

Health is vital for an individual and can be achieved with this working module. Thus making life healthy. Wide variety of ingredients, cultures and personal tastes makes decisions about what to eat a great problem. Many diseases that were previously thought as hereditary are now seen to be connected to biological dysfunction related to nutrition.

Being healthy and eating better is something the vast majority of the population wants and doing so usually requires great effort. The working prototype accomplishes a Personalized Diet Recommendation System with integration of Machine Learning Algorithms to recommend the right food at the right time and with the right nutrition.

## CHAPTER 7

### FUTURE ENHANCEMENT

- **Enhance the UI Layer** – The UI has to be given various better functionalities and must be made a device compatible version which when released in production can provide a better functionalities to its users. The various technologies that can be used are Bootstrap and further enhanced tech stack for Web Design like HTML 5, CSS 3 and Flask. Validation into the user's login and Sign-up for data privacy can be the next major step.
- **Diversifying the data being used in the working prototype** – A lot more enhancement in the dataset can be a major step in future so that a lot more details about the various food items can be taken into consideration to provide a better functionality to the user. More Nutrients can be included in the data being utilized along with other inputs in addition to what is used now from the GUI concerning the user details. A Data Layer needs to be used in the working module to keep a backup of all the information and user details with the food database.
- **Make it support on Cloud-Based Technology** – A cloud based technology version of the current supported module can help in better use of the packed module and dependencies being used. Cloud version of the module being released can provide better security, reliability, mobility and unlimited storage capacity.

## CHAPTER 8

### REFERENCES

- [1] Sakshi Singh , Sanjay Kumar Dubey, “Recommendation of Diet to a Patient using AHP and Fuzzy Approach”, International Conference on Cloud Computing, Data Science and Engineering , IEEE-2019.
- [2] D.Rao,C.Higgins,H.Margot,T.Lyle,S.Falls,E.Obeysekare,K.Mehta,”Micro nutrient Deficiencies in the Developing World:An evaluation of Delivery Methods”,Global Humanitarian Technology Conference, 2016.
- [3] A. Thapar and M. Goyal, "A fuzzy expert system for diagnosis of malnutrition in children," 2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Agra, India, 2016, pp. 1-6. doi: 10.1109/R10HTC.2016.7906819.\
- [4] Arushi Singh , Nandini Kashyap , Rakesh Garg, “Fuzzy based approach for diet prediction” , International Conference on Cloud Computing, Data Science and Engineering , IEEE-2019.
- [5] Chih-Han Chen, Maria Karvela, Mohammadreza Sohbati, Thaksin Shinawatra, Christofer Toumazou, “PERSON—Personalized Expert Recommendation System for Optimized Nutrition” , IEEE-2018.
- [6] Celestine Iwendi, Suleman Khan, Joseph Henry Anajemba, Ali Kashif Bashir, “Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model” , IEEE-2020.
- [7] S. Saini and S. K. Dubey, “Recommendation of diet to jaundice patient on the basis of nutrients using AHP and fuzzy AHP technique,” Int. J. Intell. Eng. Syst., vol. 10, no. 4, pp. 91–99, Jul. 2017.



- [8] L. Yang, C.-K. Hsieh, H. Yang, J. P. Pollak, N. Dell, S. Belongie, C. Cole, and D. Estrin, “Yum-Me: A personalized nutrient-based meal recommender system,” *ACM Trans. Inf. Syst.*, vol. 36, no. 1, pp. 1–31, Jul. 2017.
- [9] Raciél Yera Toledo, Ahmad A. Alzahrani, Luis Martinez, “Food Recommender System Considering Nutritional Information and User Preferences” , IEEE-2019.
- [10] Jitao Yang , “Personalized Nutrition Solution based on Nutrigenomics” , 2019 19th International Conference on Computational Science and Its Applications , IEEE-2019.
- [11] Heba Abdelgader Mohammed, Hani Hagra, “Towards Developing Type 2 Fuzzy Logic Diet Recommendation System for Diabetes” , IEEE-2018.
- [12] Agapito G., Calabrese B., Guzzi P. H., Cannataro M., Simeoni M., Car´e I., Lamprinoudi T., Fuiano G., Pujia A. , “DIETOS : a recommender system for adaptive diet monitoring and personalized food suggestion.” , IEEE-2016.
- [13] Romeshwar Sookrah, Jaysree Devesh Dhowtal, Soukshmee Devi Nagowah, “A DASH Diet Recommendation System for Hypertensive Patients using machine learning” , 2019 7th International Conference on Information and Communication Technology , IEEE-2019.

# APPENDIX

## **#Importing Python Pacakages**

```
from tkinter import *
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

## **#Tkinter Window**

```
main_win = Tk()
```

## **#Reading Dataset**

```
data=pd.read_csv('G:\Major project\input.csv')
data.head(5)
```

## **#Function for Weight Loss Option**

```
def Weight_Loss():
    print(" Age: %s\n Weight%s\n Height%s\n" % (e1.get(), e2.get(),e3.get()))
```

## **#Segregating Breakfast , Lunch and Dinner**

```
Items Breakfastdata=data['Breakfast']
BreakfastdataNumpy=Breakfastdata.to_numpy()
```

```
Lunchdata=data['Lunch']
LunchdataNumpy=Lunchdata.to_numpy()
```

```
Dinnerdata=data['Dinner']
DinnerdataNumpy=Dinnerdata.to_numpy()
```

```
Food_itemsdata=data['Food_items']
```

```

breakfastfoodseparatedID=[]
LunchfoodseparatedID=[]
DinnerfoodseparatedID=[]

for i in range(len(Breakfastdata)):
    if BreakfastdataNumpy[i]==1:
        breakfastfoodseparated.append(Food_itemsdata[i])
        breakfastfoodseparatedID.append(i)
    if LunchdataNumpy[i]==1:
        Lunchfoodseparated.append(Food_itemsdata[i])
        LunchfoodseparatedID.append(i)
    if DinnerdataNumpy[i]==1:
        Dinnerfoodseparated.append(Food_itemsdata[i])
        DinnerfoodseparatedID.append(i)

```

#### **# Retrieving rows by loc method |**

```

LunchfoodseparatedIDdata = data.iloc[LunchfoodseparatedID]
print(LunchfoodseparatedID)
LunchfoodseparatedIDdata=LunchfoodseparatedIDdata.T
val=list(np.arange(5,15))
Valapnd=[0]+val
LunchfoodseparatedIDdata=LunchfoodseparatedIDdata.iloc[Valapnd]
LunchfoodseparatedIDdata=LunchfoodseparatedIDdata.T
print (LunchfoodseparatedIDdata)
print (LunchfoodseparatedIDdata.describe())

breakfastfoodseparatedIDdata = data.iloc[breakfastfoodseparatedID]
breakfastfoodseparatedIDdata=breakfastfoodseparatedIDdata.T
val=list(np.arange(5,15))
Valapnd=[0]+val
breakfastfoodseparatedIDdata=breakfastfoodseparatedIDdata.iloc[Valapnd]

```

```
breakfastfoodseparatedIDdata=breakfastfoodseparatedIDdata.T
print (breakfastfoodseparatedIDdata)
print (breakfastfoodseparatedIDdata.describe())
```

```
DinnerfoodseparatedIDdata = data.iloc[DinnerfoodseparatedID]
DinnerfoodseparatedIDdata=DinnerfoodseparatedIDdata.T
val=list(np.arange(5,15))
Valapnd=[0]+val
DinnerfoodseparatedIDdata=DinnerfoodseparatedIDdata.iloc[Valapnd]
DinnerfoodseparatedIDdata=DinnerfoodseparatedIDdata.T
print (DinnerfoodseparatedIDdata)
print (DinnerfoodseparatedIDdata.describe())
```

```
age=int(e1.get())
weight=float(e2.get())
height=float(e3.get())
bmi = weight/(height**2)
agewiseinp=0
for lp in range (0,80,20):
    test_list=np.arange(lp,lp+20)
    for i in test_list:
        if(i == age):
            print('age is between',str(lp),str(lp+10))
            tr=round(lp/20)
            agecl=round(lp/20)
```

### **#BMI conditions**

```
print("Your body mass index is: ", bmi)
if ( bmi < 16):
    print("severely underweight")
    clbmi=4
```

```

elif ( bmi >= 16 and bmi < 18.5):
    print("underweight")
    clbmi=3
elif ( bmi >= 18.5 and bmi < 25):
    print("Healthy")
    clbmi=2
elif ( bmi >= 25 and bmi < 30):
    print("overweight")
    clbmi=1
elif ( bmi >=30):
    print("severely overweight")
    clbmi=0

val1=DinnerfoodseparatedIDdata.describe()
valTog=val1.T
print (valTog.shape)
print (valTog)

DinnerfoodseparatedIDdata=DinnerfoodseparatedIDdata.to_numpy()
LunchfoodseparatedIDdata=LunchfoodseparatedIDdata.to_numpy()
breakfastfoodseparatedIDdata=breakfastfoodseparatedIDdata.to_numpy()
ti=(clbmi+agecl)/2

```

### **# K-Means Applied For Dinner Food Items**

```

Datacalorie=DinnerfoodseparatedIDdata[1:,1:len(DinnerfoodseparatedIDdata)]
X = np.array(Datacalorie)
kmeans = KMeans(n_clusters=3, random_state=0).fit(X)
print ('## Prediction Result ##')
print(kmeans.labels_)
print (kmeans.predict([Datacalorie[0]]))
XValu=np.arange(0,len(kmeans.labels_))
# fig,axs=plt.subplots(1,1,figsize=(15,5))
# plt.bar(XValu,kmeans.labels_)

```

```
dnrlbl=kmeans.labels_
```

### **# K-Means Applied For Lunch Food Items**

```
Datacalorie=LunchfoodseparatedIDdata[1:,1:len(LunchfoodseparatedIDdata)]
```

```
#print(Datacalorie)
```

```
X = np.array(Datacalorie)
```

```
kmeans = KMeans(n_clusters=3, random_state=0).fit(X)
```

```
print ('## Prediction Result ##')
```

```
print(kmeans.labels_)
```

```
XValu=np.arange(0,len(kmeans.labels_))
```

```
#fig,axs=plt.subplots(1,1,figsize=(15,5))
```

```
#plt.bar(XValu,kmeans.labels_)
```

```
Inchlbl=kmeans.labels_
```

### **# K-Means Applied For Breakfast Food Items**

```
Datacalorie=breakfastfoodseparatedIDdata[1:,1:len(breakfastfoodseparatedIDdata)]
```

```
#print(Datacalorie)
```

```
X = np.array(Datacalorie)
```

```
kmeans = KMeans(n_clusters=3, random_state=0).fit(X)
```

```
print ('## Prediction Result ##')
```

```
print(kmeans.labels_)
```

```
XValu=np.arange(0,len(kmeans.labels_))
```

```
fig,axs=plt.subplots(1,1,figsize=(15,5))
```

```
plt.bar(XValu,kmeans.labels_)
```

```
brklbl=kmeans.labels_
```

```
datafin=pd.read_csv('G:\Major project\inputfin.csv')
```

```
datafin.head(5)
```

```
## train set
```

```
#arrayfin=[agecl,clbmi,]
```

```
#Age & BMI Data combining and processed (kmeans)
```

```

dataTog=datafin.T
bmicls=[0,1,2,3,4]
agecls=[0,1,2,3,4]
weightlosscat = dataTog.iloc[[1,2,7,8]]
weightlosscat=weightlosscat.T
weightgaincat= dataTog.iloc[[0,1,2,3,4,7,9,10]]
weightgaincat=weightgaincat.T
healthycat = dataTog.iloc[[1,2,3,4,6,7,9]]
healthycat=healthycat.T
weightlosscatDdata=weightlosscat.to_numpy()
weightgaincatDdata=weightgaincat.to_numpy()
healthycatDdata=healthycat.to_numpy()
weightlosscat=weightlosscatDdata[1:,0:len(weightlosscatDdata)]
weightgaincat=weightgaincatDdata[1:,0:len(weightgaincatDdata)]
healthycat=healthycatDdata[1:,0:len(healthycatDdata)]

#print(weightgaincat)
#print (len(weightlosscat))
weightlossfin=np.zeros((len(weightlosscat)*5,6),dtype=np.float32)
weightgainfin=np.zeros((len(weightgaincat)*5,10),dtype=np.float32)
healthycatfin=np.zeros((len(healthycat)*5,9),dtype=np.float32)
t=0
r=0
s=0
yt=[]
yr=[]
ys=[]
for zz in range(5):
    for jj in range(len(weightlosscat)):
        valloc=list(weightlosscat[jj])
        valloc.append(bmicls[zz])

```

```

        valloc.append(agecls[zz])
        weightlossfin[t]=np.array(valloc)
        yt.append(brklbl[jj])
        t+=1
for jj in range(len(weightgaincat)):
    valloc=list(weightgaincat[jj])
    valloc.append(bmicls[zz])
    valloc.append(agecls[zz])
    weightgainfin[r]=np.array(valloc)
    yr.append(lnchlbl[jj])
    r+=1
for jj in range(len(healthycat)):
    valloc=list(healthycat[jj])
    valloc.append(bmicls[zz])
    valloc.append(agecls[zz])
    healthycatfin[s]=np.array(valloc)
    ys.append(dnrblbl[jj])
    s+=1

X_test=np.zeros((len(weightlosscat),6),dtype=np.float32)
print('*****')

```

### **#Random Forest Classifier**

```

for jj in range(len(weightlosscat)):
    valloc=list(weightlosscat[jj])
    valloc.append(agecl)
    valloc.append(clbmi)
    X_test[jj]=np.array(valloc)*ti
# print (X_test)
# print (len(weightlosscat))
# print (len(X_test))

```



```
X_train=weightlossfin# Features
```

```
y_train=yt # Labels
```

```
# Split dataset into training set and test set
```

```
# X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) #
```

```
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)
```

```
#Recommended Food Items
```

```
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]
```

```
#GUI Interface window
```

```
Label(main_win,text="Age").grid(row=0,column=0,sticky=W,pady=4)
```

```
Label(main_win,text="Weight").grid(row=1,column=0,sticky=W,pady=4)
```

```
Label(main_win,text="Height").grid(row=2,column=0,sticky=W,pady=4)
```

```
e1 = Entry(main_win)
```

```

e2 = Entry(main_win)
e3 = Entry(main_win)

e1.grid(row=0, column=1, sticky = W, pady = 2)
e2.grid(row=1, column=1, sticky = W, pady = 2)
e3.grid(row=2, column=1,sticky = W, pady = 2)

Button(main_win,text='Quit',command=main_win.quit).grid(row=4,column=1,sticky="",pady=4)
Button(main_win,text='Weight
Loss',command=Weight_Loss).grid(row=3,column=0,sticky="",pady=4)
Button(main_win,text='Weight
Gain',command=Weight_Gain).grid(row=3,column=1,sticky="",pady=4)
Button(main_win,text='Healthy',command=Healthy).grid(row=3,column=2,sticky="",pady=4)
main_win.geometry("400x200")
main_win.wm_title("DIET RECOMMENDATION SYSTEM")
main_win.mainloop()

```

# PAPER PUBLICATION STATUS

Publication process not yet started

# PLAGIARISM REPORT

2884079.pdf

## ORIGINALITY REPORT

6%

SIMILARITY INDEX

5%

INTERNET SOURCES

4%

PUBLICATIONS

5%

STUDENT PAPERS

## PRIMARY SOURCES

1

Submitted to American University of the Middle East

Student Paper

1%

2

Submitted to Amity University

Student Paper

1%

3

morioh.com

Internet Source

1%

4

Submitted to University of Queensland

Student Paper

1%

Submitted to Vels University

Student Paper

<1%

Rutuja Rewane, P. M. Chouragade. "Food Nutritional Detection, Visualization and Recommendation for Health Monitoring using Image Processing", 2019

3rd International Conference on Trends in Electronics and Informatics (ICOEI),

2019

Publication

<1%

7

Celestine Iwendi, Suleman Khan, Joseph Henry

Anajemba, Ali Kashif Bashir, Fazal Noor. "Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model", IEEE Access, 2020  
Publication

<1%

8

Chih-Han Chen, Maria Karvela, Mohammadreza Sohbaty, Thaksin Shinawatra, Christofer Toumazou. "PERSON—Personalized Expert Recommendation System for Optimized Nutrition", IEEE Transactions on Biomedical Circuits and Systems, 2018  
Publication

<1%

9

[www.kdnuggets.com](http://www.kdnuggets.com)  
Internet Source

10

"Inventive Communication and Computational Technologies", Springer Science and Business Media LLC, 2020  
Publication

<1%

Farmaki, Cristina, Kostas Mavrigiannakis, Kostas Marias, Michalis Zervakis, and Vangelis Sakkalis. "Assessment of automated brain structures segmentation based on the mean- shift algorithm: Application in brain tumor", Proceedings of the 10th IEEE International Conference on Information Technology and Applications in Biomedicine, 2010.  
Publication

<1%

<1%

<1%

12

Submitted to University of Hong Kong  
Student Paper

<1%

13

Submitted to Limerick Institute of Technology  
Student Paper

14

e-space.mmu.ac.uk  
Internet Source

<1%

15

Submitted to Trident University International  
Student Paper

<1%

Exclude quotes      On

Exclude matches < 10 words

Exclude bibliography      On