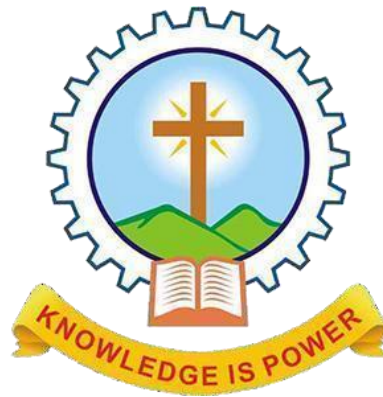


MAR ATHANASIUS COLLEGE OF ENGINEERING
(Affiliated to APJ Abdul Kalam Technological University, TVM)
KOTHAMANGALAM



Department of Computer Applications

Mini Project Report

OBESITY DETECTION USING
MACHINE LEARNING

Done by

Meryl John

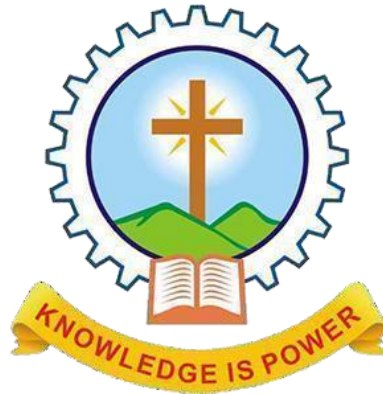
Reg No:MAC23MCA-2038

Under the guidance of
Prof. Nisha Markose

2023-2025

MAR ATHANASIUS COLLEGE OF ENGINEERING
(Affiliated to APJ Abdul Kalam Technological University, TVM)
KOTHAMANGALAM

CERTIFICATE



Obesity Detection Using Machine Learning

Certified that this is the bonafide record of project work done by

Meryl John
Reg No: MAC23MCA-2038

during the third semester, in partial fulfilment of requirements for award of the degree

Master of Computer Applications

of

APJ Abdul Kalam Technological University Thiruvananthapuram

Faculty Guide

Prof. Nisha Markose

Head of the Department

Prof. Biju Skaria

Project Coordinator

Prof. Sonia Abraham

Internal Examiners

CONTENTS

| | |
|---|----------|
| Acknowledgement | i |
| Abstract | ii |
| List of Tables | iii |
| List of Figures | iv |
| 1 Introduction | 1 |
| 2 Supporting Literature | 2 |
| 2.1 Literature Review | 2 |
| 2.1.1 Summary Table | 5 |
| 2.2 Findings and Proposals | 6 |
| 3 System Analysis | 7 |
| 3.1 Analysis of Dataset | 7 |
| 3.1.1 About the Dataset | 7 |
| 3.1.2 Explore the dataset | 9 |
| 3.2 Data Pre-processing | 11 |
| 3.2.1 Data Cleaning | 11 |
| 3.2.2 Analysis of Feature Variables | 15 |
| 3.2.3 Analysis of Class Variables..... | 17 |
| 3.3 Data Visualization | 17 |
| 3.4 Analysis of Algorithm | 24 |
| 3.5 Project Plan..... | 29 |
| 3.5.1 Project Pipeline | 29 |
| 3.6 Feasibility Analysis | 30 |
| 3.6.1 Technical Feasibility | 30 |
| 3.6.2 Economic Feasibility..... | 31 |
| 3.6.3 Operational Feasibility | 32 |
| 3.7 System Environment..... | 33 |
| 3.7.1 Software Environment | 33 |
| 3.7.2 Hardware Environment | 35 |

| | | |
|-----------|------------------------------------|-----------|
| 4 | System Design | 36 |
| 4.1 | Model Building..... | 36 |
| 4.1.1 | Model Planning..... | 36 |
| 4.1.2 | Training..... | 38 |
| 4.1.3 | Testing..... | 39 |
| 5 | Result and Conclusions | 41 |
| 6 | Model Deployment | 46 |
| 7 | Git History | 51 |
| 8 | Conclusion | 52 |
| 9 | Future Work | 53 |
| 10 | Appendix | 55 |
| 10.1 | Minimum Software Requirements..... | 55 |
| 10.2 | Minimum Hardware Requirements..... | 56 |
| 11 | References | 58 |

1. INTRODUCTION

Obesity, a complex and multifaceted health condition, poses significant challenges to public health worldwide. Characterized by excessive body fat accumulation, obesity increases the risk of various chronic diseases such as diabetes, cardiovascular conditions, and certain cancers. Traditional methods of detecting and predicting obesity often rely on body mass index (BMI) calculations and clinical assessments, which, while useful, can be limited by their static nature and reliance on physical examinations. Machine learning, a subset of artificial intelligence, offers promising advancements in the field of obesity detection and prediction. By leveraging vast amounts of data and sophisticated algorithms, machine learning models can identify patterns and correlations that might not be evident through conventional methods.

The “Obesity Detection Using Machine Learning” project is focused on detecting obesity. The performance of two machine learning algorithms such as Logistic Regression and Random Forest are compared to classify obesity into Underweight, Normal Weight, Overweight Level I, OverweightLevel II, Obesity Type I, Obesity Type II and Obesity Type III.

The dataset is taken from the Kaggle repository. The dataset contains 2111 sample observations and has 17 columns including 1 identifier, 1 class variable and 16 features. The dataset contains Numeric and Categorical values.

To address these challenges, an automated system using machine learning can assist medical professionals by providing more accurate predictions of obesity risk. This system leverages vast datasets containing clinical measurements, patient history, and lifestyle factors to train machine learning models. These models can identify patterns and correlations that might be missed by human experts.

2. SUPPORTING LITERATURE

2.1 Literature Review

Paper 1: Ferdowsy, Faria, Kazi Samsul Alam Rahi, Md Ismail Jabiullah, and Md Tarek Habib. "A machine learning approach for obesity risk prediction." *Current Research in Behavioral Sciences* 2 (2021): 100053.

This paper by Ferdowsy et al., titled "*A machine learning approach for obesity risk prediction*," explores the application of machine learning models in predicting obesity risk, leveraging behavioral and lifestyle data. Published in *Current Research in Behavioral Sciences*, this 2021 study recognizes obesity as a significant global health concern linked to various chronic conditions. It underscores the importance of predictive tools that can offer early identification of individuals at risk of obesity, ultimately supporting public health interventions.

The authors present a machine learning framework designed to identify potential obesity risk factors from lifestyle and health-related behaviors. Their model utilizes a dataset containing variables like dietary habits, physical activity levels, and demographic factors to predict the likelihood of an individual being at risk for obesity. Different machine learning algorithms are applied and evaluated to determine the most effective for accurate risk prediction.

The study findings indicate that specific machine learning models, such as decision trees and support vector machines, performed particularly well in predicting obesity risk based on the selected features. The authors discuss the strengths and limitations of each model, highlighting that while machine learning can offer significant predictive power, the quality and completeness of input data are crucial.

The paper concludes by emphasizing the potential impact of machine learning on public health, particularly in the realm of preventative health care for obesity. Ferdowsy et al. advocate for further research to refine predictive models and integrate them into healthcare systems to facilitate timely interventions and reduce obesity prevalence worldwide.

Paper 2: Rodríguez, Elias, Elen Rodríguez, Luiz Nascimento, Aneirson Francisco da Silva, and Fernando Augusto Silva Marins. "Machine learning Techniques to Predict Overweight or Obesity." *In IDDM*, pp. 190-204. 2021.

The paper by Rodríguez et al., titled "*Machine Learning Techniques to Predict Overweight or Obesity*," presented in the *IDDM Conference Proceedings (2021)*, explores various machine learning techniques to forecast the likelihood of overweight and obesity. Recognizing the global health burden posed by obesity and its associated comorbidities, the authors aim to harness machine learning to enable early detection and, consequently, more proactive health management strategies.

The authors describe their methodology, where they analyze data from diverse demographic and lifestyle-related variables to predict overweight and obesity outcomes. They implement and compare several machine learning algorithms, including decision trees, random forests, and support vector machines, focusing on which models offer the highest accuracy and robustness for predicting obesity-related outcomes. This study includes a feature selection process to identify the most influential factors impacting obesity prediction accuracy, such as dietary habits, physical activity, and socioeconomic status.

Findings reveal that machine learning models can successfully identify individuals at risk for overweight and obesity with a high degree of accuracy. Among the models, random forests and support vector machines emerged as particularly effective, highlighting their potential utility in clinical and public health contexts for risk assessment and patient counselling. The authors also discuss how the quality and diversity of data are crucial for refining the predictive capacity of these algorithms.

In conclusion, Rodríguez et al. emphasize the value of machine learning in obesity prevention and management. They advocate for continued research to improve model accuracy, integrate predictive tools into health care systems, and ultimately aid in combating the rising obesity epidemic through data-driven health interventions.

.

Paper 3: Yagin, F. H., Güllü, M., Gormez, Y., Castañeda-Babarro, A., Colak, C., Greco, G., & Cataldi, S. (2023). Estimation of obesity levels with a trained neural network approach optimized by the Bayesian technique. *Applied Sciences*, 13(6), 3875.

The paper by Yagin et al., titled "*Estimation of Obesity Levels with a Trained Neural Network Approach Optimized by the Bayesian Technique*," published in *Applied Sciences* (2023), explores the use of advanced neural network models to estimate obesity levels, focusing on optimizing prediction accuracy with Bayesian techniques. The study addresses the growing need for precise, data-driven models in predicting and categorizing obesity, a condition with significant health and economic implications globally.

In this work, the authors develop a neural network model trained on a dataset comprising demographic, lifestyle, and behavioral factors related to obesity. They employ Bayesian optimization to fine-tune the neural network's parameters, ensuring the model achieves high prediction accuracy and generalizability across different populations. Bayesian optimization plays a crucial role in this approach, as it systematically searches the parameter space to find optimal configurations, reducing the time and computational resources needed compared to traditional optimization methods.

The study results indicate that the Bayesian-optimized neural network achieves superior performance in predicting obesity levels, demonstrating high accuracy and robustness. The authors compare the performance of their approach with other machine learning models, finding that the neural network with Bayesian optimization consistently outperforms traditional models in terms of prediction precision and model stability. They attribute this success to the capability of Bayesian optimization to adapt the model parameters effectively to the complexity of the obesity dataset.

In conclusion, Yagin et al. highlight the potential of neural networks, especially when enhanced with Bayesian techniques, to improve obesity prediction and categorization. This research suggests that neural networks, with proper optimization, can be highly effective tools in health data analysis, paving the way for more accurate obesity detection systems that can support targeted health interventions and policy-making in public health.

2.1.1. Summary Table

| PAPER TITLE | ALGORITHM | DATASET | METHODOLOGY | PRECISION/ RMSE |
|---|---|---|---|---|
| Ferdowsy, Faria, Kazi Samsul Alam Rahi, Md Ismail Jabiullah, and Md Tarek Habib."A machine learning approach for obesity risk prediction." <i>Current Research in Behavioral Sciences</i> 2 (2021): 100053. | k-NN, SVM, LR, Naïve Bias, RF, Decision Tree, ADA Boosting, MLP, Gradient Boosting. | Collected 1100 data based on 28 factors and then labelled the class of each record of the data set by consulting with some nutritionists and student counsellors in educational institutions. | k-NN, Support Vector Machine, Logistic Regression, Naïve Bias, Random Forest, Decision Tree, ADA Boosting, MLP, Gradient Boosting are compared on the basis of Accuracy, Sensitivity, Specificity, Precision Recall and F1-score. | LR - 97.09% RF - 72.30% MLP - 66.02% |
| Rodríguez, Elias, Elen Rodríguez, Luiz Nascimento, Aneirson Francisco da Silva, and Fernando Augusto Silva Marins. "Machine learning Techniques to Predict Overweight or Obesity." In <i>IDDM</i> , pp. 190-204.2021. | Decision Tree, SVM, KNN, Gaussian Naive Bias, MLP, Random Forest, Gradient Boosting, Extreme Gradient Boosting. | The investigation included data for the estimation of obesity levels, including the eating habits and physical activity statuses of 498 participants between the ages of 14 and 61 from Barranquilla, Colombia; Lima, Peru; and the City of Mexico, Mexico. | Decision Tree, Support Vector Machines, K Nearest Neighbors, Gaussian Naive Bias, Multilayer Perceptron, Random Forest, Gradient Boosting, Extreme Gradient Boosting are compared on the basis of Accuracy, Precision, Recall and F1-score. | RF - 77.69% K-NN - 67.69% GB - 73.43% MLP - 63.77% |
| Yagin, F. H., Güllü, M., Gormez, Y., Castañeda-Babarro, A., Colak, C., Greco, G., ... & Cataldi, S. (2023). Estimation of obesity levels with a trained neural network approach optimized by the Bayesian technique. <i>Applied Sciences</i> , 13(6), 3875. | Neural Network Model and Bayesian Classification | The dataset was taken from the UCI Machine Learning Repository. It consists of 2111 records with 17 features related to eating habits and physical conditions collected from Mexico, Peru, and Colombia. | Chi-Square, F Classify, Mutual Information Classification are compared on the basis of Accuracy, SD Accuracy, F1-Score, Sensitivity, Specificity. | Original Model – 93.06% F-Classify – 90.32% Chi-Square – 89.04% Mutual Information Classification – 86.52% |

Table 2.1 Reference papers summary

2.2. Findings and Proposals

From the three literatures, we get to know that different approaches are used for the detection of obesity. The main theme of the first paper is the application of machine learning algorithms to predict obesity risk. The second paper deals with the application of machine learning techniques to predict overweight and obesity. Third paper aims at the prediction of obesity levels using a trained neural network approach optimized by Bayesian techniques.

Accurate and timely detection of obesity is crucial for effective treatment and management. The proposed system is the comparative study of two algorithms Logistic Regression and Random Forest. These algorithms are more accurate than the other algorithms with approximate accuracy of 97% and 78% respectively. The models will classify obesity under seven classes which are Underweight, Normal Weight, Overweight Level I, Overweight Level II, Obesity Type I, Obesity Type II and Obesity Type III. Random Forest is known for its robustness and handling of complex datasets and Logistic Regression for its simplicity and interpretability.

Dataset is collected from Kaggle. The dataset is then pre-processed to normalize data, and select relevant features. The dataset is split into training and testing dataset. Two different machine learning models such as Logistic Regression (LR) and Random Forest (RF), are trained using the prepared training dataset. The models are evaluated on a separate test dataset using metrics such as accuracy, precision, recall, F1-score etc.

An automated system based on these machine learning models can significantly benefit both medical professionals and patients. Incorrect diagnosis may lead to wrong medication and further complexities. So, an automated system can be very helpful to assist medical experts and even make automated disease predictions without any human mistakes. Patients can also diagnose their condition without the assistance of a medical expert.

3. SYSTEM ANALYSIS

3.1. Analysis of Dataset

3.1.1. About the Dataset

The Obesity Detection Dataset is a valuable resource in the field of health analytics and machine learning. It is designed to analyze and predict obesity levels based on various demographic, behavioral, and lifestyle factors. The dataset contains numeric and categorical values.

Dataset Link:

<https://www.kaggle.com/code/mabdullahabrar/multi-class-obesity-risk-prediction>

Size: The dataset contains 2,111 sample observations with 17 columns, including 1 identifier, 1 class variable, and 15 features.

Data Split: The dataset is organized into a single file containing a comprehensive set of features related to individual health metrics and lifestyle choices.

Sparsity: The dataset may contain some missing values due to the varied nature of health-related behaviors and demographics, although it is relatively dense compared to other health datasets.

| | Gender | Age | Height | Weight | Family History with Overweight | Frequent consumption of high caloric food | Frequency of consumption of vegetables | Number of main meals | Consumption of food between meals |
|---|--------|------|--------|--------|--------------------------------|---|--|----------------------|-----------------------------------|
| 0 | Female | 21.0 | 1.62 | 64.0 | yes | no | 2.0 | 3.0 | Sometimes |
| 1 | Female | 21.0 | 1.52 | 56.0 | yes | no | 3.0 | 3.0 | Sometimes |
| 2 | Male | 23.0 | 1.80 | 77.0 | yes | no | 2.0 | 3.0 | Sometimes |
| 3 | Male | 27.0 | 1.80 | 87.0 | no | no | 3.0 | 3.0 | Sometimes |
| 4 | Male | 22.0 | 1.78 | 89.8 | no | no | 2.0 | 1.0 | Sometimes |

| Smoke | Consumption of water daily | Calories consumption monitoring | Physical activity frequency | Time using technology devices | Consumption of alcohol | Transportation used | Obesity |
|-------|----------------------------|---------------------------------|-----------------------------|-------------------------------|------------------------|-----------------------|---------------------|
| no | 2.0 | no | 0.0 | 1.0 | no | Public Transportation | Normal Weight |
| yes | 3.0 | yes | 3.0 | 0.0 | Sometimes | Public Transportation | Normal Weight |
| no | 2.0 | no | 2.0 | 1.0 | Frequently | Public Transportation | Normal Weight |
| no | 2.0 | no | 2.0 | 0.0 | Frequently | Walking | Overweight Level I |
| no | 2.0 | no | 0.0 | 0.0 | Sometimes | Public Transportation | Overweight Level II |

Fig 3.1 Snapshot of Obesity.csv

This is the result produce by listing the dataset files using head() function.

3.1.2. Explore the Dataset

Dataset has 2111 rows and 17 columns.



Fig 3.2 Shape of the dataset

Dataset contains 8 numerical attributes and 9 categorical attributes including target class.

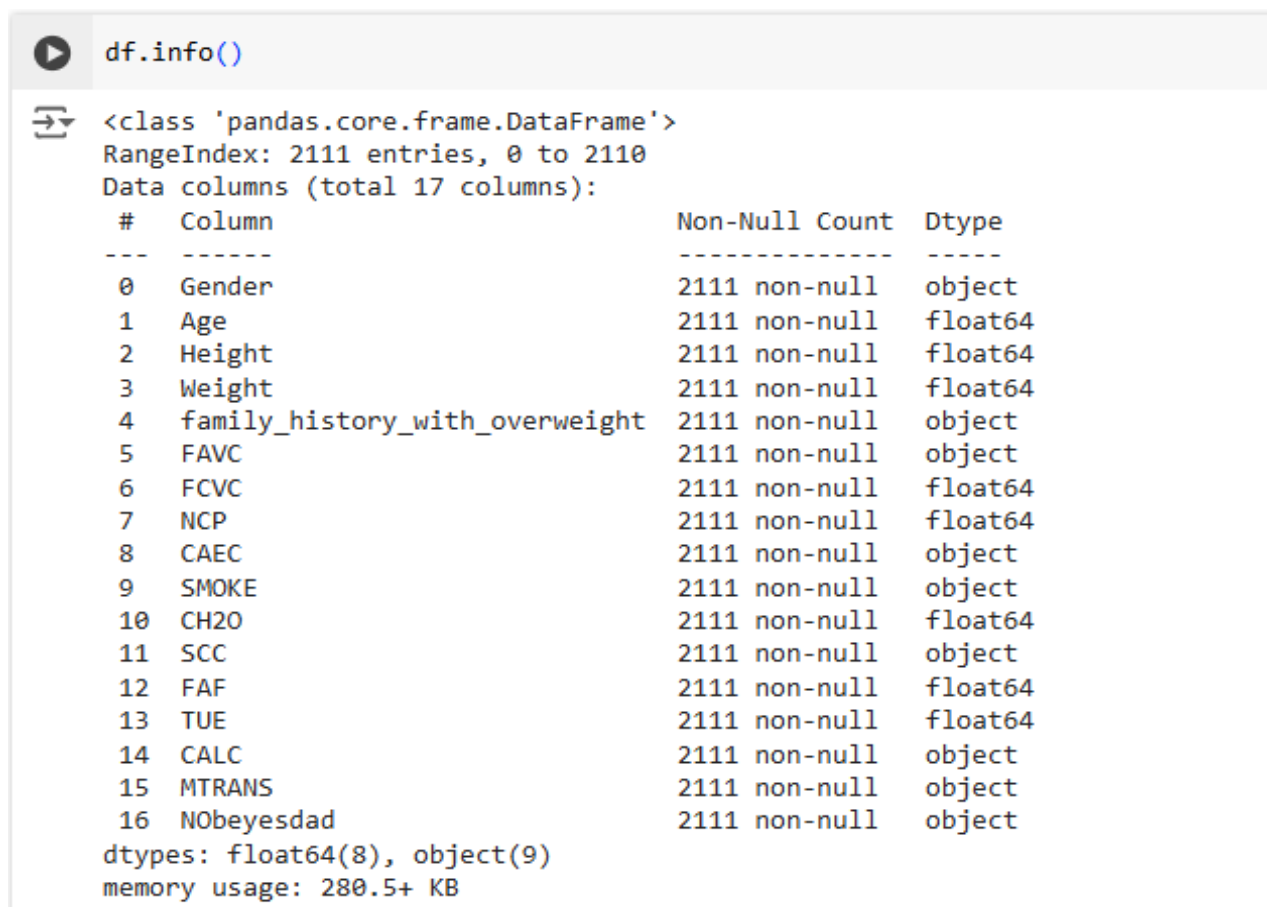


Fig 3.3 Datatypes of each attribute

```
df.describe()
```

| | Age | Height | Weight | FCVC | NCP | CH2O | FAF | TUE |
|--------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| count | 2111.000000 | 2111.000000 | 2111.000000 | 2111.000000 | 2111.000000 | 2111.000000 | 2111.000000 | 2111.000000 |
| mean | 24.312600 | 1.701677 | 86.586058 | 2.419043 | 2.685628 | 2.008011 | 1.010298 | 0.657866 |
| std | 6.345968 | 0.093305 | 26.191172 | 0.533927 | 0.778039 | 0.612953 | 0.850592 | 0.608927 |
| min | 14.000000 | 1.450000 | 39.000000 | 1.000000 | 1.000000 | 1.000000 | 0.000000 | 0.000000 |
| 25% | 19.947192 | 1.630000 | 65.473343 | 2.000000 | 2.658738 | 1.584812 | 0.124505 | 0.000000 |
| 50% | 22.777890 | 1.700499 | 83.000000 | 2.385502 | 3.000000 | 2.000000 | 1.000000 | 0.625350 |
| 75% | 26.000000 | 1.768464 | 107.430682 | 3.000000 | 3.000000 | 2.477420 | 1.666678 | 1.000000 |
| max | 61.000000 | 1.980000 | 173.000000 | 3.000000 | 4.000000 | 3.000000 | 3.000000 | 2.000000 |

Fig 3.4 Statistical Summary of numeric attributes

1. 'Age' : The average age is about 24 years, with a standard deviation of 6.35 years. Ages range from 14 to 61, showing a wide age distribution, mostly concentrated in younger adults.
2. 'Height (in meters)' : Heights vary between 1.45m and 1.98m, with an average height of 1.70m and a small standard deviation of 0.093, suggesting most values are close to the mean.
3. 'Weight (in kg)' : The mean weight is 86.6 kg, ranging widely from 39 to 173 kg, indicating significant variability in the dataset's weight distribution.
4. 'Food Consumption':
 - 'FCVC (Frequency of Consumption of Vegetables)' : With a mean of 2.42 and a maximum value of 3, most individuals consume vegetables regularly.
 - 'NCP (Number of Main Meals per Day)' : The average is 2.69, with values between 1 and 4, suggesting most people consume around 2-3 meals daily.
 - 'CH2O (Daily Water Consumption)' : The average is around 2 glasses per day, with a relatively low standard deviation, implying consistency in water intake across individuals.
5. 'Physical Activity and Time on Electronic Devices' :
 - 'FAF (Physical Activity Frequency)' : The mean value is 1.01, with a max of 3, indicating varying activity levels but generally lower frequencies.
 - 'TUE (Time Using Electronic Devices)' : The mean is around 0.66, showing a moderate use of electronic devices.

This data indicates that the population is generally young, with moderate height and a wide weight range. Daily habits such as vegetable consumption, meal frequency, and physical activity vary but are generally moderate.

3.2. Data Preprocessing

3.2.1. Data Cleaning

Data Cleaning is the data pre-processing method we choose. Data cleaning routines attempt to fill in missing values, smooth out noisy data and correct inconsistencies. The dataset taken is already pre-processed, so pre-processing techniques are not need for the dataset.

Rounding Numerical values and Mapping categorical values

```
# Converting Height to cm and rounding numerical values
df['Height'] = df['Height'] * 100
df['Height'] = df['Height'].round(1)
df['Weight'] = df['Weight'].round(1)
df['Age'] = df['Age'].round(1)
df.head()
```

```
[ ] # Round relevant columns
for x in ['Frequency of consumption of vegetables', 'Number of main meals',
         'Consumption of water daily', 'Physical activity frequency', 'Time using technology devices']:
    df[x] = df[x].apply(round)
df.head()
```

```
# Replacing numerical values with categorical mappings
mapping0 = {1: 'Never', 2: 'Sometimes', 3: 'Always'}
mapping1 = {1: '1', 2: '2', 3: '3', 4: '3+'}
mapping2 = {1: 'Less than a liter', 2: 'Between 1 and 2 L', 3: 'More than 2 L'}
mapping3 = {0: 'I do not have', 1: '1 or 2 days', 2: '2 or 4 days', 3: '4 or 5 days'}
mapping4 = {0: '0-2 hours', 1: '3-5 hours', 2: 'More than 5 hours'}
```

```
[ ] df['Frequency of consumption of vegetables'] = df['Frequency of consumption of vegetables'].replace(mapping0)
df['Number of main meals'] = df['Number of main meals'].replace(mapping1)
df['Consumption of water daily'] = df['Consumption of water daily'].replace(mapping2)
df['Physical activity frequency'] = df['Physical activity frequency'].replace(mapping3)
df['Time using technology devices'] = df['Time using technology devices'].replace(mapping4)
df.head()
```



| | Gender | Age | Height | Weight | Family History with Overweight | Frequent consumption of high caloric food | Frequency of consumption of vegetables | Number of main meals | Consumption of food between meals | Smoke |
|---|--------|------|--------|--------|--------------------------------|---|--|----------------------|-----------------------------------|-------|
| 0 | Female | 21.0 | 162.0 | 64.0 | yes | no | Sometimes | 3 | Sometimes | no |
| 1 | Female | 21.0 | 152.0 | 56.0 | yes | no | Always | 3 | Sometimes | yes |
| 2 | Male | 23.0 | 180.0 | 77.0 | yes | no | Sometimes | 3 | Sometimes | no |
| 3 | Male | 27.0 | 180.0 | 87.0 | no | no | Always | 3 | Sometimes | no |
| 4 | Male | 22.0 | 178.0 | 89.8 | no | no | Sometimes | 1 | Sometimes | no |

| Consumption of water daily | Calories consumption monitoring | Physical activity frequency | Time using technology devices | Consumption of alcohol | Transportation used | Obesity |
|----------------------------|---------------------------------|-----------------------------|-------------------------------|------------------------|-----------------------|---------------------|
| Between 1 and 2 L | no | I do not have | 3–5 hours | no | Public Transportation | Normal Weight |
| More than 2 L | yes | 4 or 5 days | 0–2 hours | Sometimes | Public Transportation | Normal Weight |
| Between 1 and 2 L | no | 2 or 4 days | 3–5 hours | Frequently | Public Transportation | Normal Weight |
| Between 1 and 2 L | no | 2 or 4 days | 0–2 hours | Frequently | Walking | Overweight Level I |
| Between 1 and 2 L | no | I do not have | 0–2 hours | Sometimes | Public Transportation | Overweight Level II |

Fig 3.5 Rounding and Mapping of attributes

Handling missing values

```

▶ print(df.isnull().sum())
Gender      0
Age         0
Height      0
Weight      0
family_history_with_overweight  0
FAVC        0
FCVC        0
NCP         0
CAEC        0
SMOKE       0
CH20        0
SCC         0
FAF         0
TUE         0
CALC        0
MTRANS      0
NObeyesdad  0
dtype: int64

```

Fig 3.6 Handling missing values

No null values were found.

Handling class imbalance

```

✓ 0s ▶ df['Obesity'].value_counts()

```

| Obesity | count |
|---------------------|-------|
| Obesity Type I | 351 |
| Obesity Type III | 324 |
| Obesity Type II | 297 |
| Overweight Level I | 290 |
| Overweight Level II | 290 |
| Normal Weight | 287 |
| Insufficient Weight | 272 |

dtype: int64

Fig 3.7 Count of Class distribution

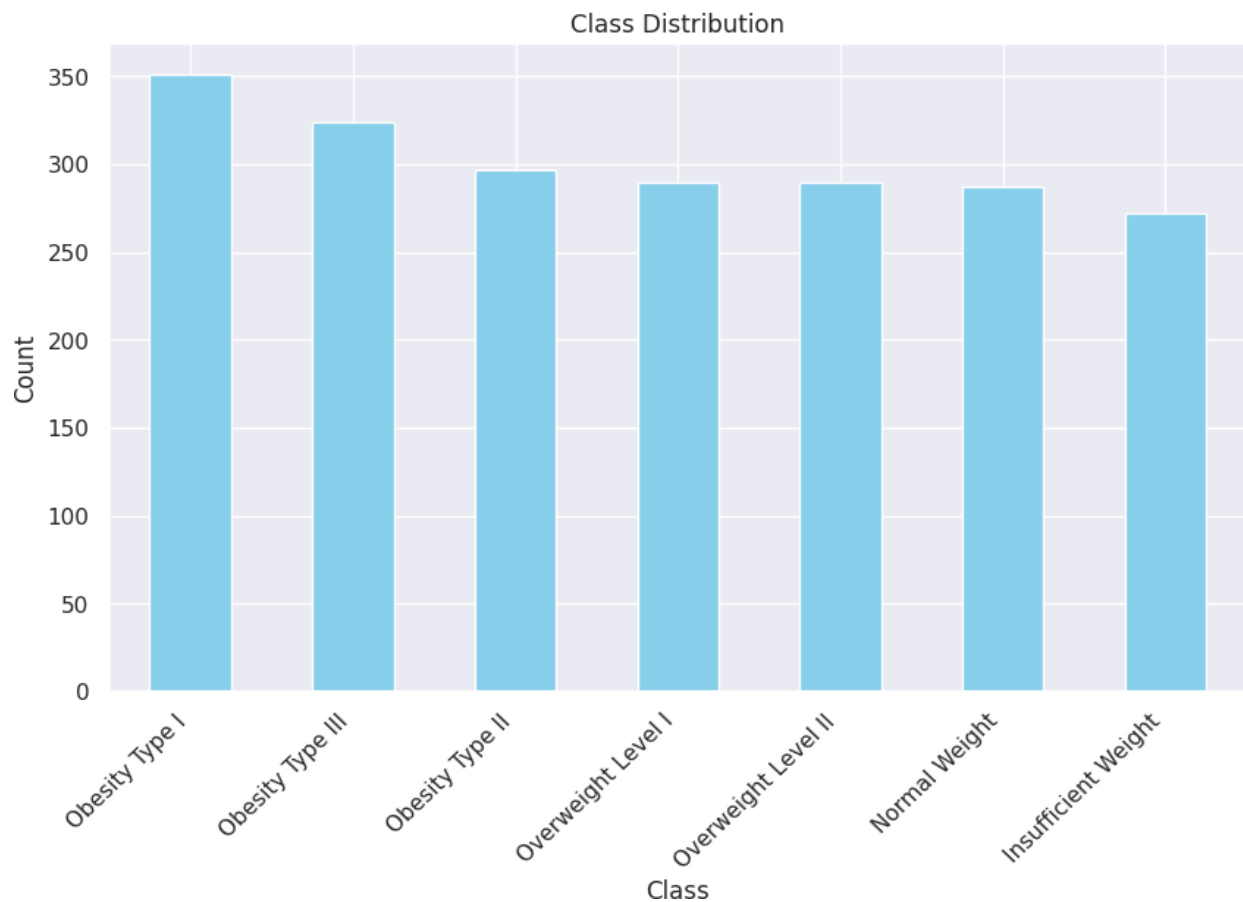


Fig 3.8 Class Distribution

The slight differences in class counts within the dataset suggest a relatively balanced distribution across the various obesity levels. This balance is advantageous for machine learning model training, as it reduces the risk of bias towards any single class. With such a distribution, models are likely to perform better across all categories, rather than favoring the more prevalent classes. This balanced dataset allows for a more accurate representation of each obesity category, enhancing the model's ability to detect and predict a range of obesity levels effectively.

3.2.2 Analysis of Feature Variables

Physical Features :

These features are essential to calculating BMI and understanding basic demographic characteristics of individuals, which are directly related to obesity classification.

1. 'Gender' : This binary feature is crucial as obesity trends can vary by gender due to hormonal differences, metabolism, and societal habits. Men and women may also have different fat distribution patterns, which can influence obesity-related health risks.
2. 'Age' : Age often correlates with metabolic rate, lifestyle changes, and general health. Obesity prevalence may increase with age, particularly in middle age and older adults, where metabolic slowdown and lifestyle changes can lead to weight gain.
3. 'Height' : Used in BMI calculations along with weight, height is a physical measure necessary to assess body mass in relation to height. In children and adolescents, height also helps track growth trends relative to weight.
4. 'Weight' : This is one of the core indicators of obesity. Bodyweight, when used with height, helps categorize individuals into underweight, normal, overweight, or obese classes. Weight alone also provides insight into muscle mass and general health.

Family and Lifestyle related factors :

These attributes consider lifestyle habits that impact obesity risk due to behavior patterns and environmental influences.

1. 'family_history_with_overweight' : Indicates if the individual has a genetic or familial predisposition to obesity, suggesting that genetic factors, shared family dietary habits, and lifestyle might impact the person's weight.
2. 'SMOKE (Smoking Status)' : Smoking has complex interactions with weight; while it can suppress appetite in some, it's also associated with metabolic changes and unhealthy eating habits that might impact weight.
3. 'FAF (Frequency of Physical Activity)' : Physical activity frequency indicates lifestyle fitness, a significant predictor of body weight. Regular physical activity can help regulate body weight, improve metabolic health, and reduce the risk of obesity-related diseases.
4. 'TUE (Time Using Technology Devices)' : High device usage time often correlates with sedentary behavior. Extended screen time might be linked to reduced physical activity, unhealthy snacking habits, and disrupted sleep patterns, all of which can contribute to weight gain.

Food related Attributes :

These attributes cover dietary habits, focusing on food types, frequency, and overall nutrition—all directly influencing obesity risk.

1. 'FAVC (Frequent High-Caloric Food Consumption)' : High-caloric foods (fast food, fried snacks, sugary drinks) can contribute significantly to weight gain, especially if consumed frequently. This feature assesses how often an individual may be at risk of excess calorie intake.
2. 'FCVC (Frequency of Vegetable Consumption): Higher vegetable intake usually correlates with better diet quality, as vegetables are nutrient-dense and low in calories. Frequent vegetable consumption is often associated with a lower risk of obesity.
3. 'NCP (Number of Main Meals per Day)' : Regular main meals are vital for steady energy levels and metabolism. Skipping meals or irregular meal patterns can lead to overeating later and impact body weight regulation.
4. 'CAEC (Consumption of Food Between Meals)' : Snacking between meals can lead to excess calorie intake, especially if the snacks are calorie-dense and nutrient-poor. This feature shows how often the individual may be at risk of extra calories that might not be balanced by physical activity.
5. 'SCC (Caloric Sweet Consumption)' : High intake of caloric sweets (candies, desserts) can lead to sugar spikes, increased calorie intake, and weight gain, making this a potential indicator of obesity risk.
6. 'CH2O (Daily Water Intake)' : Proper hydration supports metabolism and digestion and can help manage hunger. Low water intake might correlate with poor dietary habits and could indirectly influence weight.
7. 'CALC (Alcohol Consumption Frequency)' : Alcohol can be calorie-dense and may also lead to poor dietary choices when consumed frequently. Additionally, alcohol's metabolic effects can influence weight, especially with high consumption levels.

This feature explores transportation choice, which can influence physical activity levels.

1. 'MTRANS (Mode of Transportation)' : Transportation choice often reflects activity level. For instance, people who walk or cycle may have higher physical activity levels, while those relying on cars or public transportation might engage in less daily activity, indirectly affecting body weight.

Each category provides essential insights: Physical Attributes help establish baseline measures, Family and Lifestyle Related Factors give context on environmental influences, Food-Related Attributes highlight dietary habits, and Transportation-Related Factors offer clues about daily physical activity. Together, they create a comprehensive framework that not only enhances our understanding of the multifaceted factors contributing to obesity but also enables us to predict obesity trends more accurately and improving overall health outcomes.

3.2.3. Analysis of Class Variables

In this project, there are no explicit class labels used. The primary objective is to analyze user demographics, obesity-related metrics, and lifestyle factors to develop a predictive model for obesity detection. The target variable, 'Obesity,' represents the classification of individuals based on their health status, with categories such as Obesity Type I, Obesity Type II, Obesity Type III, Overweight Levels I and II, Normal Weight, and Insufficient Weight. The project focuses on employing machine learning techniques, such as logistic regression and random forest, along with feature analysis, to uncover patterns and relationships within the data rather than traditional classification tasks with distinct class labels. This approach aims to provide insights that can inform effective interventions and public health strategies for managing obesity.

3.3. Data Visualization

Data visualization serves as a fundamental technique in data analysis, offering a graphical representation of information through charts, graphs, and maps. Its significance lies in transforming complex datasets into visually intuitive formats, allowing for the identification of trends, patterns, and outliers. This visual representation enhances data exploration, enabling easy comparison between data points and facilitating effective communication of insights. The diverse types of visualizations, such as charts for trends, tables for structured data, graphs for relationships, maps for geographic data, and dashboards for comprehensive displays, cater to various analytical needs. Utilizing tools like Tableau, Excel, or programming libraries like Matplotlib, data visualization aids in making data-driven decisions by providing accessible insights into massive amounts of information.

Age, Height and Weight

In terms of height, male and female are similarly distributed according to the box plot below. While males are generally taller than female, both male and female share a similar average. In weight, females have a much larger range of (as well as BMI) compared to male. This is further illustrated by the steeper line plot between weight and height of female than male.

```
# Boxplots for Gender vs Height and Weight
sns.set()
fig = plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
sns.boxplot(x='Gender', y='Height', data=df)
plt.subplot(1, 2, 2)
sns.boxplot(x='Gender', y='Weight', data=df)
```

```
<Axes: xlabel='Gender', ylabel='Weight'>
```

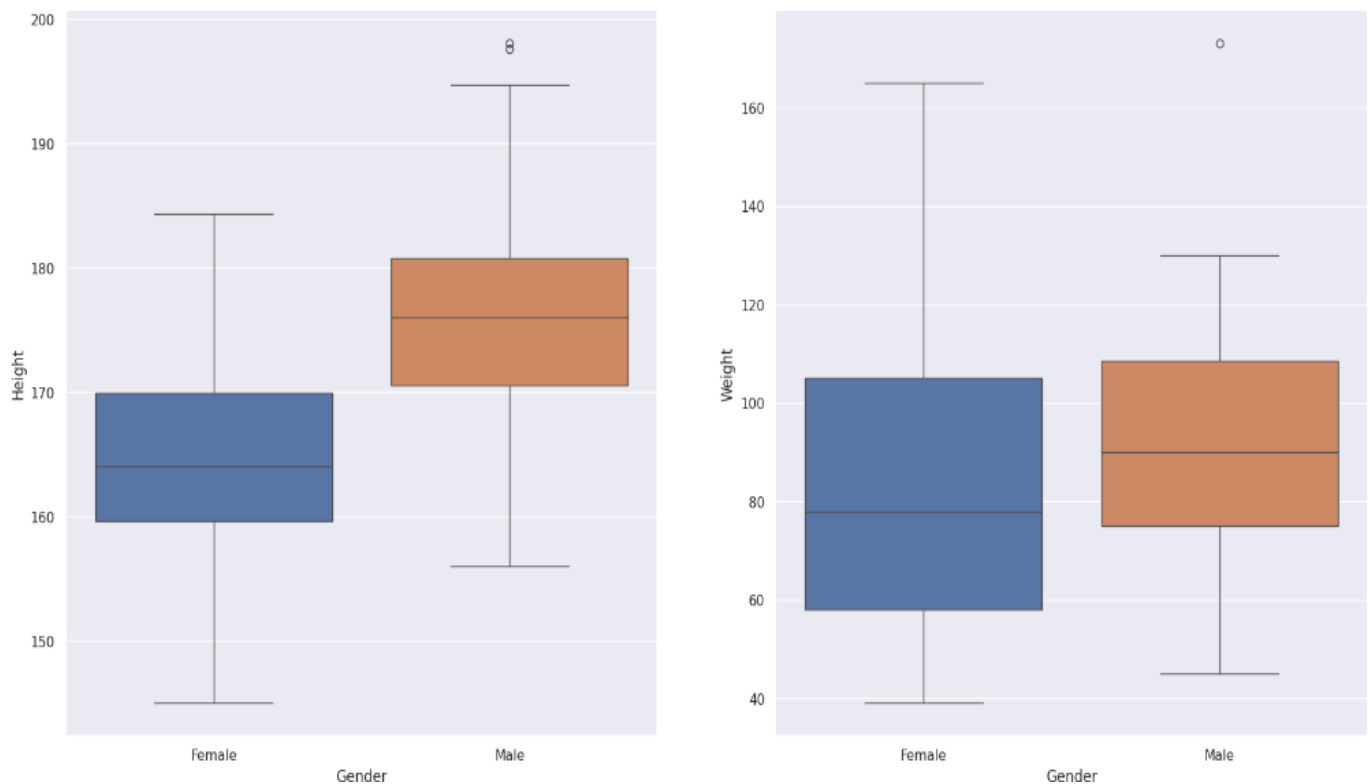


Fig 3.9 Box plot for Gender Vs Height and Weight

```
# Visualization: Height vs Weight
sns.set()
g = sns.jointplot(x="Height", y="Weight", data=df, kind="reg", truncate=False, xlim=(125, 200), ylim=(35, 180), color="m", height=10)
g.set_axis_labels("Height (cm)", "Weight (kg)")
```

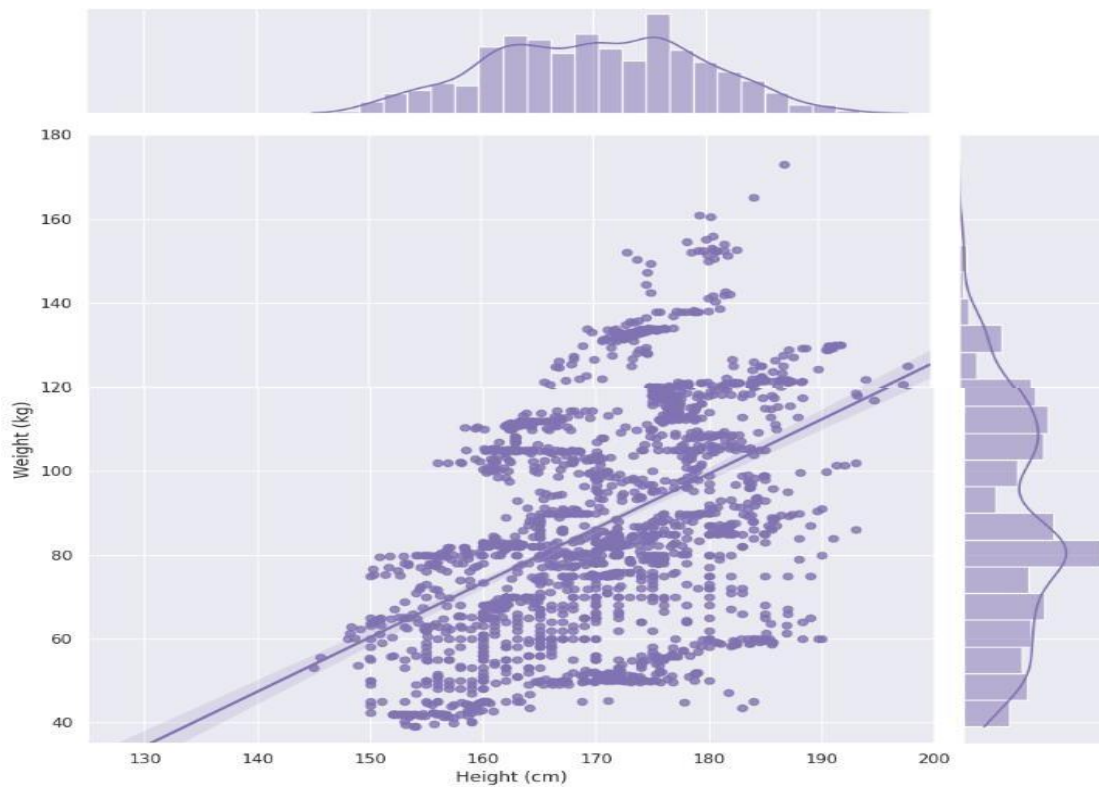
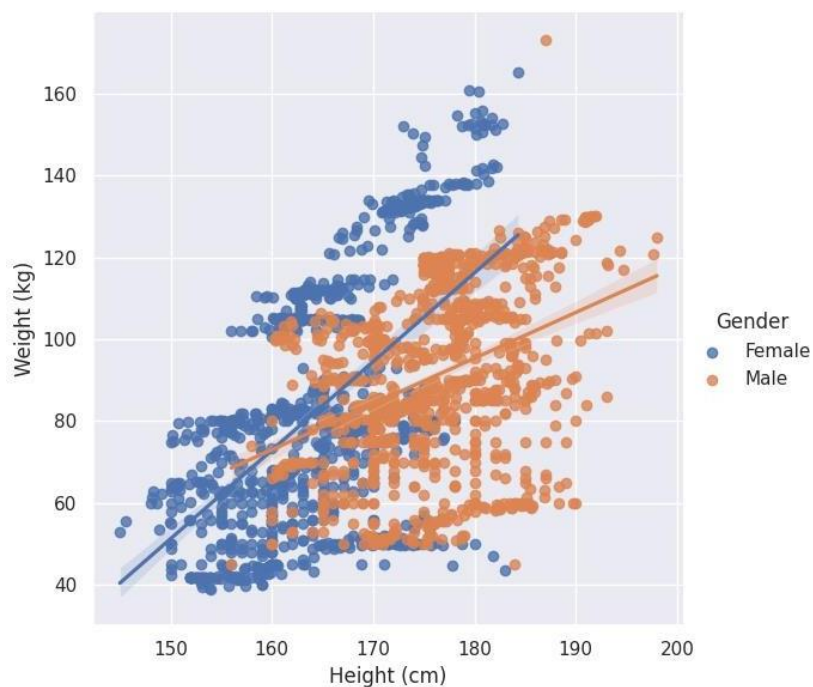


Fig 3.10 Joint plot for Height Vs Weight

Height Vs Weight by Gender

```
# Visualization: Height vs Weight by Gender  
g = sns.lmplot(x="Height", y="Weight", hue="Gender", height=10, data=df)  
g.set_axis_labels("Height (cm)", "Weight (kg)")
```



Obesity

```
# Pie Chart of Obesity
c = Counter(df['Obesity'])
fig = plt.figure(figsize=(8,8))
plt.pie([float(c[v]) for v in c], labels=[str(k) for k in c], autopct=None)
plt.title('Weight Category')
plt.tight_layout()
```

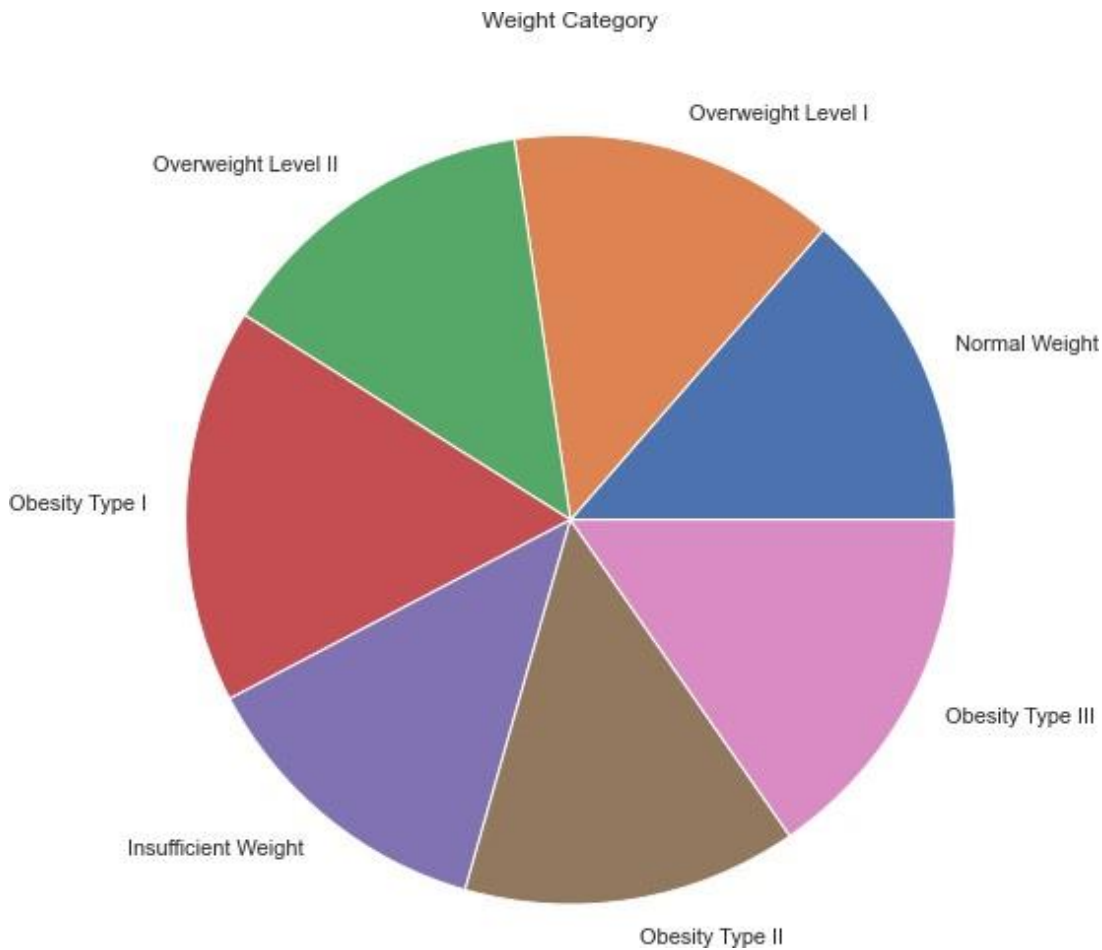


Fig 3.12 Pie Chart of Obesity

A bigger proportion of female with a higher BMI is reflected by the large slice of Obesity Type III in the pie chart below, while Obesity Type II is the most prevalent type of obesity in male. Interestingly, there is also a higher proportion of Insufficient Weight in female compared to male, this could be explained by a heavier societal pressure on women to go on diets.


```

fig = plt.figure(figsize=(20,8))
plt.subplot(1, 2, 1)
plt.pie([float(c_m[v]) for v in c_m], labels=[str(k) for k in c_m], autopct=None)
plt.title('Weight Category of Male')
plt.tight_layout()

plt.subplot(1, 2, 2)
plt.pie([float(c_f[v]) for v in c_f], labels=[str(k) for k in c_f], autopct=None)
plt.title('Weight Category of Female')
plt.tight_layout()

```

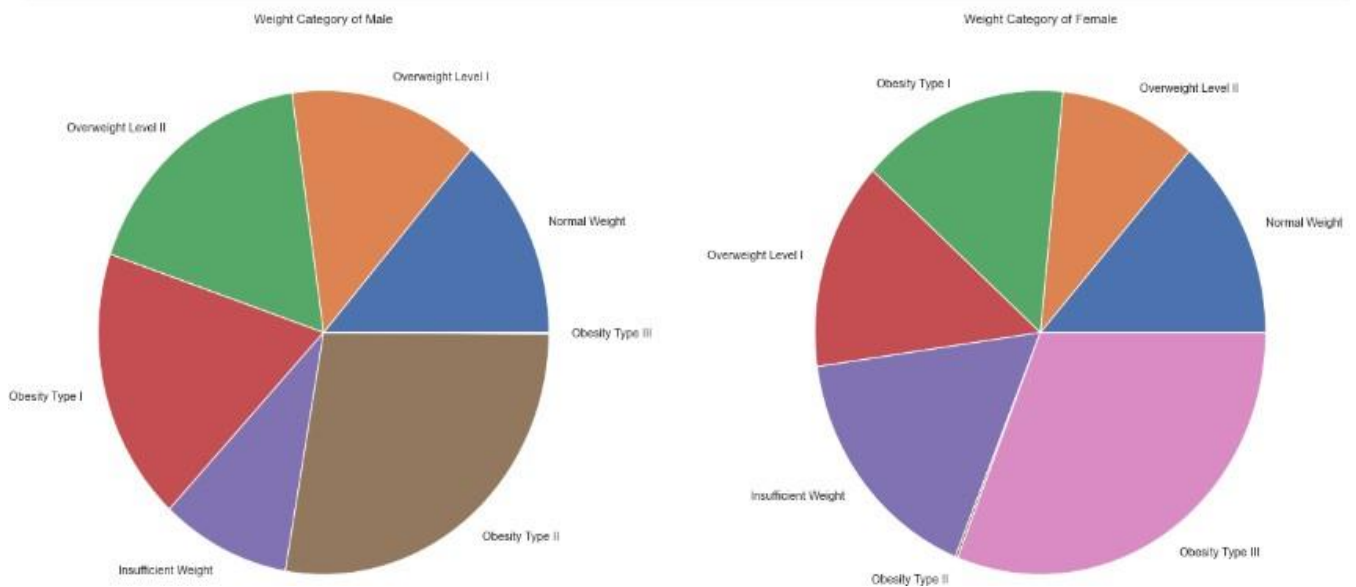


Fig 3.13 Pie Chart of Obesity for Male Vs Female

Eating and Exercise Habits

Family History with Overweight ['yes', 'no'] [1726, 385]
 Frequent consumption of high caloric food ['yes', 'no'] [1866, 245]
 Frequency of consumption of vegetables ['Sometimes', 'Always', 'Never'] [1013, 996, 102]
 Number of main meals ['3', '1', '2', '3+'] [1470, 316, 176, 149]
 Consumption of food between meals ['Sometimes', 'Frequently', 'Always', 'no'] [1765, 242, 53, 51]
 Smoke ['no', 'yes'] [2067, 44]
 Consumption of water daily ['Between 1 and 2 L', 'More than 2 L', 'Less than a liter'] [1110, 516, 485]
 Calories consumption monitoring ['no', 'yes'] [2015, 96]
 Physical activity frequency ['1 or 2 days', 'I do not have', '2 or 4 days', '4 or 5 days'] [776, 720, 496, 119]
 Time using technology devices ['0-2 hours', '3-5 hours', 'More than 5 hours'] [952, 915, 244]
 Consumption of alcohol ['Sometimes', 'no', 'Frequently', 'Always'] [1401, 639, 70, 1]
 Transportation used ['Public Transportation', 'Automobile', 'Walking', 'Motorbike', 'Bike'] [1580, 457, 56, 11, 7]

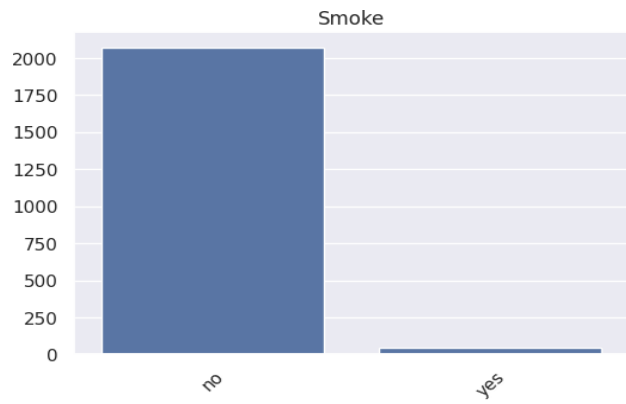


Fig 3.14 Smoke

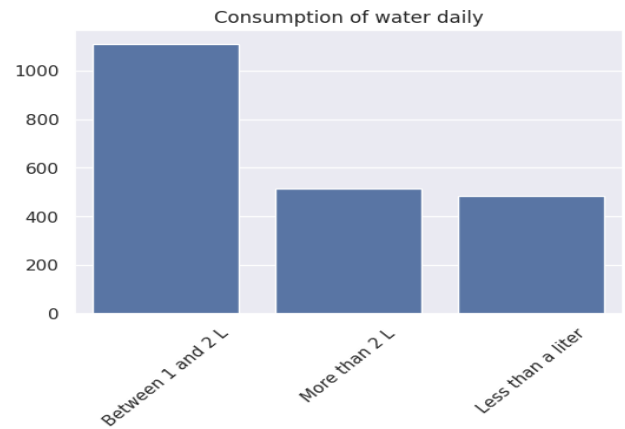


Fig 3.15 Consumption of water daily

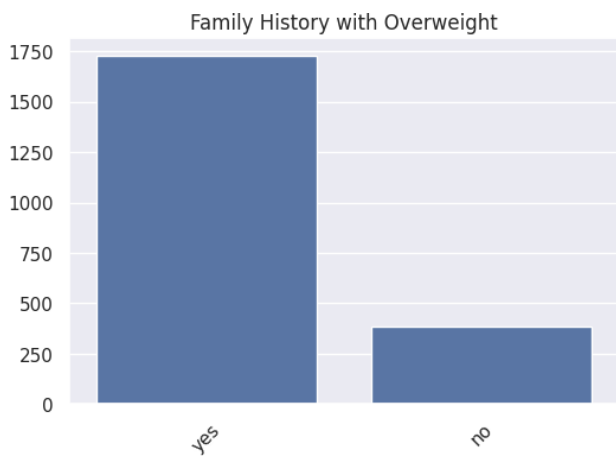


Fig 3.16 Family History with Overweight

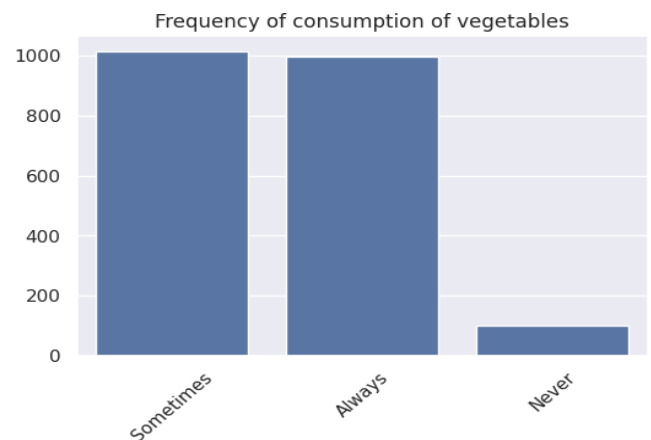


Fig 3.17 Frequency of consumption of vegetables

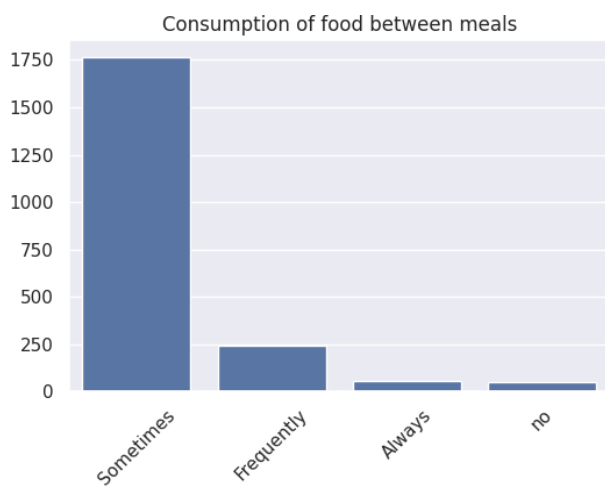


Fig 3.18 Consumption of food between meals

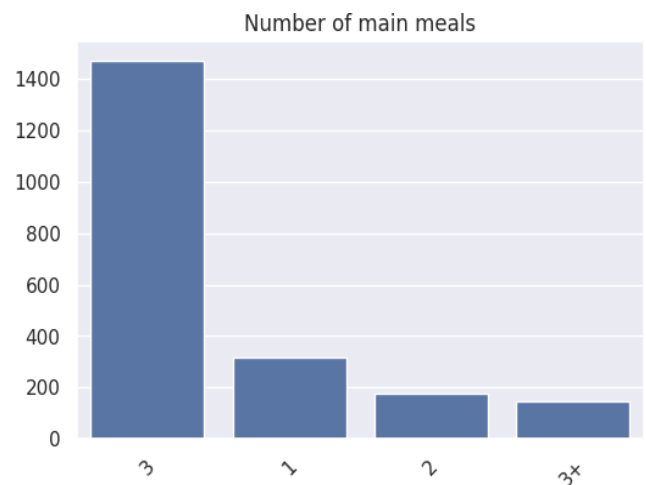


Fig 3.19 Number of main meals

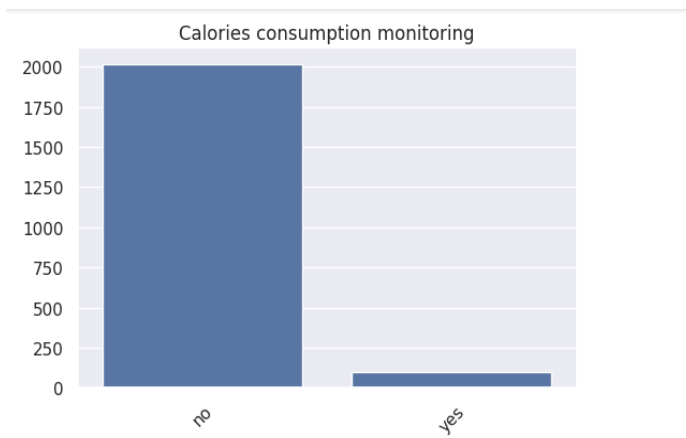


Fig 3.20 Calories consumption monitoring

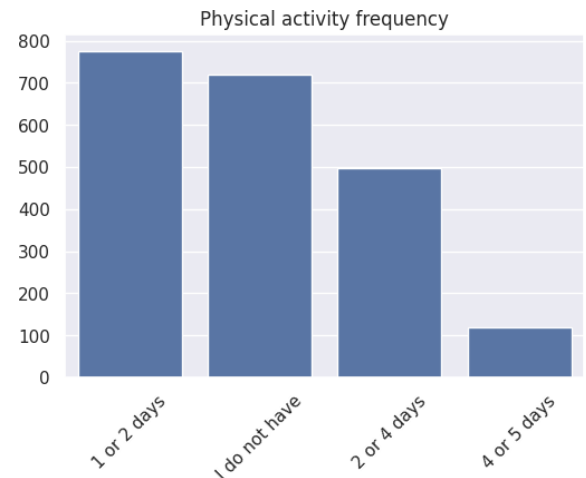


Fig 3.21 Physical activity frequency

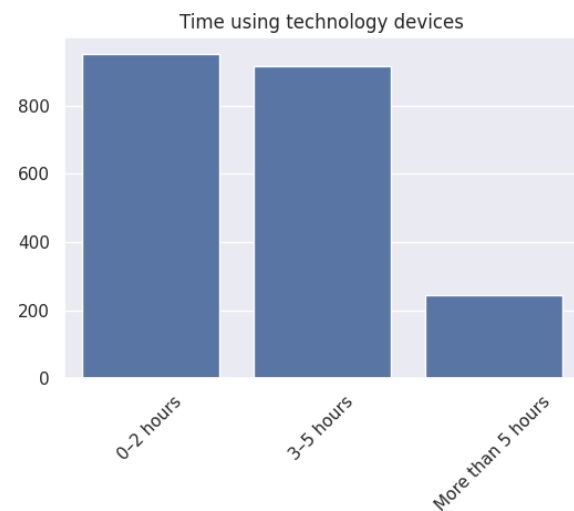


Fig 3.22 Time using technology devices

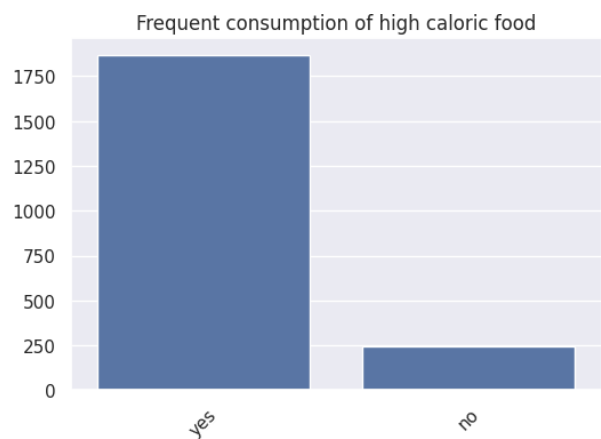


Fig 3.23 Frequent consumption of high caloric food

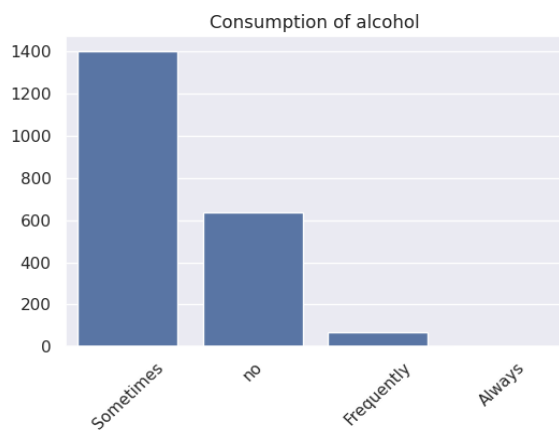


Fig 3.24 Consumption of alcohol

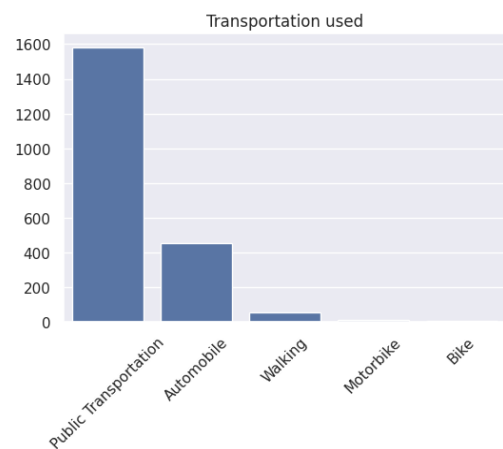


Fig 3.25 Transportation Used

3.4. Analysis of Algorithm

Algorithms used in ‘Obesity Detection Using Machine Learning’ are Logistic Regression and Random Forest.

Logistic Regression

Logistic regression comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression has several variants, including binary logistic regression, multinomial logistic regression, and ordinal logistic regression. Logistic regression is used for solving the classification problems.

Logistic Function (Sigmoid Function)

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.

Working

- Logistic regression works by creating a linear combination of input features, which involves multiplying each feature by a coefficient and summing the results.

Applies the multi-linear function to the input variables

$$z = (\sum_{i=1}^n w_i x_i) + b$$

x_i is the i th observation of X

$w_i = [w_1, w_2, w_3, \dots, w_m]$ is the weights or Coefficient

b is the bias term also known as intercept.

- This value is then passed through the logistic (sigmoid) function, which transforms it into a probability value between 0 and 1.
 Z will be input to the sigmoid function and find the probability between 0 and 1.

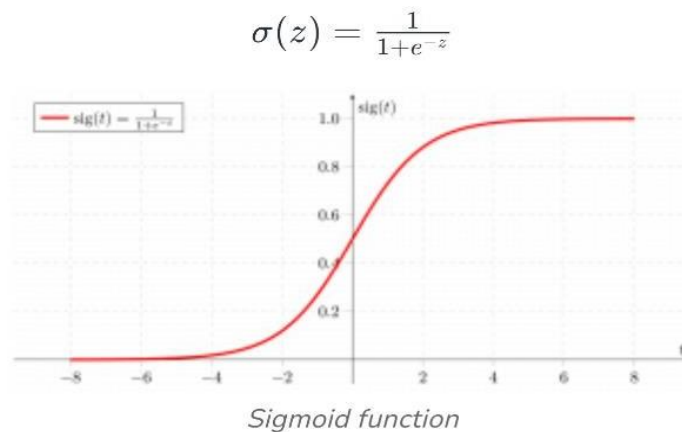


Fig 3.26 Sigmoid Function

- Sigmoid function converts the continuous variable data into the probability i.e. between 0 and 1.
- For binary classification, this probability is compared to a threshold (usually 0.5) to decide the class label.

Multinomial logistic regression, also known as softmax regression, is used for multi-class classification tasks, where there are more than two possible outcomes for the output variable. In this case, the softmax function is used in place of the sigmoid function. Softmax function for K classes will be:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

σ = softmax

\vec{z} = input vector

e^{z_i} = standard exponential function for input vector

K = number of classes in the multi-class classifier

e^{z_j} = standard exponential function for output vector

e^{z_j} = standard exponential function for output vector

- The calculated probabilities will be in the range of 0 to 1.
- The sum of all the probabilities is equals to 1.
- The class with highest probability will be the final output.

Pseudocode

1. Input:

- Training data with N samples and M features.
- Labels (0 or 1) for each sample.
- Learning rate (how big of a step to take when updating weights).
- Number of iterations (how many times to update weights).

2. Initialize:

- Set weights and bias to 0 (or small random numbers).

3. For a set number of iterations:

a. For each sample in the training data:

i. Calculate the weighted sum (z):

$$z = (\text{weight}_1 * \text{feature}_1) + (\text{weight}_2 * \text{feature}_2) + \dots + (\text{weight}_M * \text{feature}_M) + \text{bias}$$

ii. Apply the sigmoid function to get a

$$\text{probability} = 1 / (1 + \exp(-z))$$

iii. Calculate the error:

$$\text{error} = \text{probability} - \text{actual_label}$$

iv. Update each weight:

$$\text{weight}_j = \text{weight}_j - (\text{learning rate} * \text{error} * \text{feature}_j)$$

v. Update the bias:

$$\text{bias} = \text{bias} - (\text{learning rate} * \text{error})$$

4. After finishing all iterations, use the weights and bias to make predictions:

a. For a new sample:

i. Calculate the weighted sum (z) using the final weights and bias.

ii. Apply the sigmoid function to get a probability.

iii. If the probability is 0.5 or higher, output class 1. Otherwise, output class 0.

Random Forest

Random Forest is a supervised learning technique used for both Classification and Regression problems. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. Instead of relying on one decision tree, the random forest takes the output from each tree and based on the majority votes of outputs, and it takes the final output. The greater number of trees in the forest lead to higher accuracy and prevents the problem of overfitting.

Working

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new datapoints to the category that wins the majority votes.

Pseudocode

1. Input:

- Training dataset with N samples and M features.
- Number of trees to create: T.
- Number of features to randomly select at each split: F.

2. Initialize an empty list called 'forest' to store the trees.

3. For each tree (from 1 to T):

- a. Create a random sample of the training data (some samples may repeat).
- b. Build a decision tree:
 - i. At each decision point in the tree:
 - Randomly pick F features.
 - Find the best feature and split point among the F features to split the node.

- Split the node into two child nodes based on the selected split point.
 - ii. Keep splitting until the tree is fully grown or meets some stopping condition (like a max depth or minimum samples).
 - c. Add this decision tree to the 'forest'.
4. To make a prediction for a new sample x:
- a. Let each tree in the forest make a prediction.
 - b. The final prediction is the one that most trees agree on (majority vote).

In the context of your obesity detection system, the Random Forest algorithm plays a pivotal role in classifying the obesity level.

3.5. Project Plan

3.5.1. Project Pipeline

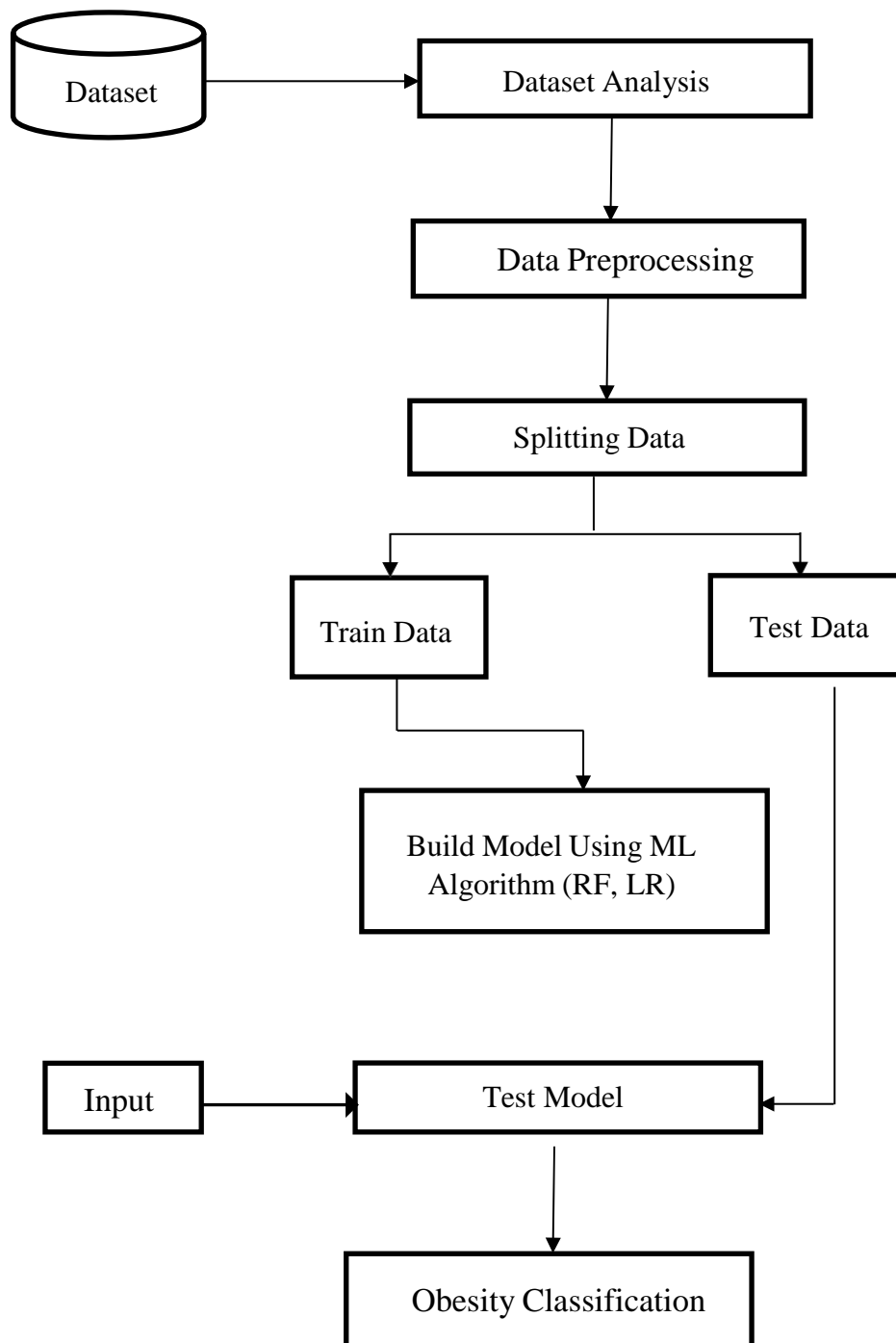


Fig 3.27 Project pipeline

3.6. Feasibility Analysis

A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing system or proposed system, opportunities and threats present in the natural environment, the resources required to carry through, and ultimately the prospects for success.

Evaluated the feasibility of the system in terms of the following categories:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

3.6.1. Technical Feasibility

Data Processing and Analysis

- Feasibility: The project involves processing and analyzing demographic, lifestyle, and health-related data using data science libraries like NumPy, Pandas, and scikit-learn. These tools are industry-standard, widely supported, and well-documented, ensuring technical feasibility for handling data preprocessing, feature selection, and model training.
- Expertise: Familiarity with these tools, demonstrated through their application in the project, reflects the technical competency needed to effectively manage and analyze the dataset for obesity prediction.

Machine Learning Algorithms

- Feasibility: The project employs Logistic Regression and Random Forest algorithms for obesity classification. These algorithms are well-suited for classification tasks and are commonly used in predictive health applications. Both are accessible through scikit-learn, ensuring the technical feasibility of implementing these models within the project.
- Expertise: The selection and implementation of these algorithms demonstrate a solid understanding of machine learning concepts, particularly for classification tasks. The project team's familiarity with Logistic Regression and Random Forest further enhances their capability to fine-tune these models for accurate obesity prediction.

Data Visualization:

- **Feasibility:** Matplotlib and Seaborn are used for data visualization, indicating the project's reliance on established and technically feasible visualization tools.
- **Expertise:** The team's ability to visualize data trends suggests proficiency in using these tools.

File Handling:

- **Feasibility:** Reading and writing data to CSV files using Pandas is a standard practice, ensuring technical feasibility.
- **Expertise:** File handling is a fundamental skill, and your team's successful utilization of it implies technical competence.

3.6.2. Economic Feasibility

Development Costs:

- **Feasibility:** Utilizing open-source machine learning libraries (e.g., Scikit-learn, TensorFlow, or Keras) significantly reduces software acquisition costs. Additionally, hardware costs for training models on appropriate computational resources should be considered, particularly if large datasets are involved.
- **Benefits:** The implementation of an effective obesity detection system can lead to improved health outcomes and increased awareness among users. Overall, these advantages can justify the initial development costs.

Maintenance Costs:

- **Feasibility:** Leveraging open-source tools generally results in lower ongoing maintenance costs compared to proprietary software. Monitoring the system's performance and user feedback will also incur costs over time.
- **Benefits:** The long-term advantages of maintaining a robust obesity detection system can be substantial. Regular updates and improvements can enhance user retention and satisfaction, which is vital for sustained engagement. Furthermore, maintaining an effective system may lead to higher trust and credibility among users and stakeholders, translating to increased usage and potential revenue streams through partnerships or subscription models.

3.6.3. Operational Feasibility

User Acceptance:

- Feasibility:

The project aims to improve user experience by providing personalized obesity risk assessments and recommendations based on user data. This focus on personalization is likely to enhance user acceptance and engagement with the system.

- Integration:

The system should seamlessly integrate with existing health and fitness applications or platforms (e.g., fitness trackers, health monitoring apps).

Scalability:

- Feasibility:

The project must consider scalability challenges, especially as user numbers and data volume grow. Algorithms like Random Forest and Logistic Regression can handle moderate amounts of data, but there may be limitations as usage increases.

- Mitigation:

Simplifying models or using ensemble methods that can efficiently manage larger.

Ease of Use:

- Feasibility:

The operational feasibility is enhanced by designing a user-friendly interface that allows users to easily input their data and understand their results.

- Training:

Assess the need for user training on how to use the system effectively. Providing clear documentation, tutorials, or in-app guidance will facilitate user understanding and promote a positive experience.

In conclusion, your *Obesity Detection using ML* project is operationally feasible, given its focus on user acceptance, scalability, ease of use, and a supportive software and hardware environment. By ensuring seamless integration with existing platforms, optimizing for growth, and maintaining a user-friendly design, the project can effectively meet operational requirements and contribute positively to user health outcomes.

3.7. System Environment

3.7.1. Software Environment

The software environment encompasses the tools, frameworks, and platforms utilized in the development and execution of the obesity detection system.

Programming Languages:

Python: The primary programming language employed for its versatility, extensive libraries (NumPy, Pandas, scikit-learn), and strong support in data science and machine learning.

Data Processing and Analysis:

NumPy and Pandas: Utilized for efficient data manipulation, handling, and analysis.

scikit-learn: Employed for implementing machine learning algorithms, particularly Random Forest and Logistic Regression for collaborative filtering.

Machine Learning Libraries:

scikit-learn: Used for implementing and training machine learning models, including the Random Forest algorithm and Logistic Regression algorithm.

SciPy: Complements NumPy and provides additional functionality for scientific computing.

Data Visualization:

Matplotlib and Seaborn: Chosen for creating insightful visualizations, aiding in the exploration and communication of data patterns.

File Handling:

Pandas: Applied for reading and writing data to CSV files, ensuring seamless data management.

Development Environment:

Google Colab

Colab is a free Jupyter notebook environment that runs entirely in the cloud. We can write and execute code in Python. Colab supports many machine learning libraries which can be easily loaded in the colab notebook.

Visual Studio Code

. Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tool a developer needs for a quick code-build-debug cycle and leaves more complex work flows to fuller featured IDEs, such as Visual Studio IDE

HTML and CSS

Hyper Text Markup Language is used for creating web pages. HTML describes the structure of the web page. Here, the user interface of my project is done using HTML. Cascading Style Sheet is used with HTML to style the web pages.

Flask

Flask is a web framework, it's a Python module that lets you develop web applications easily. It's has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features. It does have many cool features like URL routing, template engine. It is a WSGI web app framework.

GitHub

Git is an open-source version control system that was started by Linus Torvalds. Git is similar to other version control systems Subversion, CVS, and Mercurial to name a few. Version control systems keep these revisions straight, storing the modifications in a central repository. This allows developers to easily collaborate, as they can download a new version of the software, make changes, and upload the newest revision. Every developer can see these new changes, download them, and contribute. Git is the preferred version control system of most developers, since it has multiple advantages over the other systems available. It stores file changes more efficiently and ensures file integrity better.

The social networking aspect of GitHub is probably its most powerful feature, allowing projects to grow more than just about any of the other features offered. Project revisions can be discussed publicly, so a mass of experts can contribute knowledge and collaborate to advance a project forward.

3.7.2. Hardware Environment

The hardware environment refers to the physical infrastructure necessary to support the development and deployment of the classification system.

Computing Resources:

Processor:

2 GHz, A powerful multi-core processor is recommended to handle the computational demands efficiently. Consider a processor with at least quad- core architecture for optimal performance.

Storage:

Sufficient Disk Space: 512 GB SSD

Memory (RAM):

4GB RAM:

Sufficient RAM is crucial for efficiently handling large datasets and performing machine learning tasks.

Internet Connectivity:

High-Speed Internet:

A stable and high-speed internet connection is essential for accessing online resources, libraries, and datasets during development.

4. SYSTEM DESIGN

4.1. Model Building

4.1.1 Model Planning

The model planning phase for obesity detection focuses on developing a structured approach for detecting obesity levels using machine learning techniques. It involves selecting algorithms, defining the overall scope, and outlining strategies to accurately classify obesity based on demographic features, eating habits and physical condition features.

1. Objective:

Build an obesity classification system using machine learning techniques to predict and categorize various obesity levels. The goal is to accurately classify obesity such as underweight, normal weight, overweight level I, overweight level II, obesity type I, obesity type II and obesity type III using demographic and eating habit features like weight, height, family history with overweight and other relevant biomarkers.

2. Approach:

Use a comparative study of two machine learning algorithms: Random Forest (RF) and Logistic Regression (LR). These models will be evaluated for accuracy, precision, recall, and robustness in detecting obesity using a pre-processed dataset.

3. Data Preparation:

Import and preprocess the dataset that includes data with physical features and other relevant indicators. Split the dataset into training and test sets for model evaluation.

4. Exploratory Data Analysis (EDA):

Perform EDA to understand the distribution of obesity across the dataset, analyze feature distributions and relationships between physical indicators and different obesity levels to understand patterns in the data that helps in classification.

5. Model Building:

Implement the machine learning algorithms: Random Forest (RF) and Logistic Regression (LR). Train the models using the dataset features and evaluate their performance on the test set using evaluation metrics such as accuracy, confusion matrix and classification report to assess the models' predictive abilities.

6. Model Comparison:

Compare the performance of the two algorithms based on accuracy and speed. Choose the best-performing model based on its ability to accurately detect different obesity levels in unseen data.

Random Forest

```
[ ] # Build a Random Forest Classifier model with reduced complexity
rf_classifier = RandomForestClassifier(
    n_estimators=50,          # Reduce the number of trees
    max_depth=10,            # Limit the maximum depth of the trees
    max_features='sqrt',     # Limit the number of features used to split each node
    min_samples_split=4,     # Increase the minimum samples required to split a node
    min_samples_leaf=4,      # Increase the minimum samples required to be at a leaf node
    random_state=42
)
rf_classifier.fit(X_train, y_train)
```

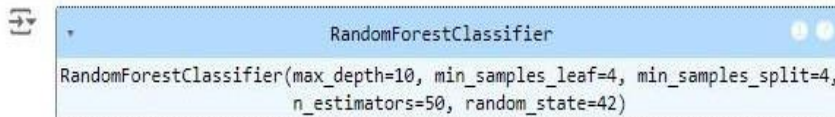


Fig 4.1 Fitted Random Forest Model

A Random Forest Classifier, which is an ensemble learning method based on multiple decision trees is used to train the model. The model consists of 50 decision trees. The maximum depth of each tree is limited to 10 levels. The number of features considered for splitting at each node is the square root of the total number of features.

Logistic Regression

```
[ ] # Logistic Regression Model
lr_classifier = LogisticRegression(max_iter=1000, random_state=42)
lr_classifier.fit(X_train, y_train)
```

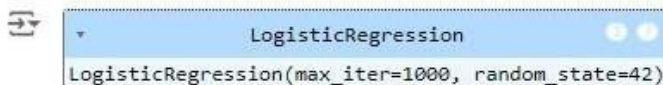


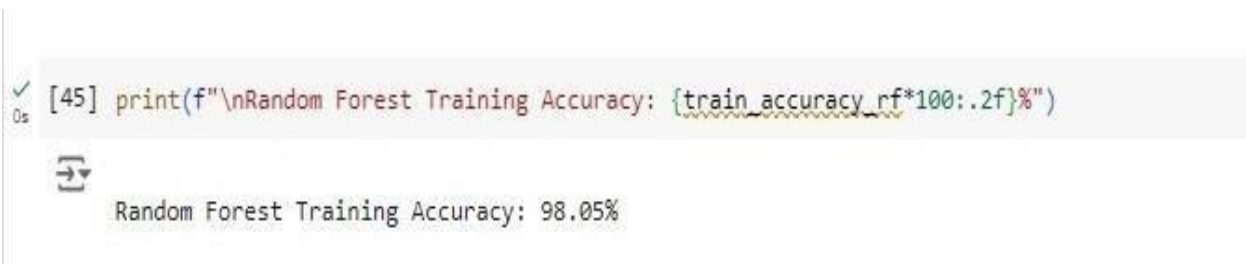
Fig 4.2 Fitted Logistic regression Model

Logistic Regression is used to train the model. The model was set to handle multiple classes using the multinomial logistic regression approach. The parameter controls the regularization strength and uses Softmax approaches to classify between multiple conditions.

4.1.2 Training

The training phase involves the construction of the Obesity detection System using the Random Forest and Logistic Regression for both user-based and item-based collaborative filtering.

Random Forest: This ensemble-based algorithm is trained on the entire feature matrix. It works by creating multiple decision trees and combining their results to improve prediction accuracy. During training, it learns patterns in the data to classify individuals into the correct obesity category. Hyperparameters like the number of trees, maximum depth, and minimum samples per leaf can be fine-tuned for better performance.

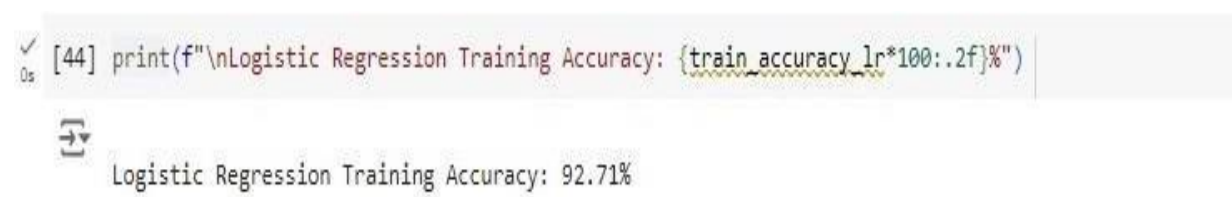
A screenshot of a Jupyter Notebook cell. The code cell contains the line: `[45] print(f"\nRandom Forest Training Accuracy: {train_accuracy_rf*100:.2f}%")`. Below the code, the output is displayed: `Random Forest Training Accuracy: 98.05%`.

```
[45] print(f"\nRandom Forest Training Accuracy: {train_accuracy_rf*100:.2f}%")  
  
Random Forest Training Accuracy: 98.05%
```

Fig 4.3 Training Accuracy of Random Forest Model

The model achieved an accuracy of 98.05% on the training data.

Logistic Regression: This algorithm models the relationship between features and the target variable, aiming to predict probabilities for each obesity category. It leverages a multinomial approach for this multiclass classification task. Training the logistic regression model involves optimizing the coefficients to maximize classification accuracy.

A screenshot of a Jupyter Notebook cell. The code cell contains the line: `[44] print(f"\nLogistic Regression Training Accuracy: {train_accuracy_lr*100:.2f}%")`. Below the code, the output is displayed: `Logistic Regression Training Accuracy: 92.71%`.

```
[44] print(f"\nLogistic Regression Training Accuracy: {train_accuracy_lr*100:.2f}%")  
  
Logistic Regression Training Accuracy: 92.71%
```

Fig 4.4 Training Accuracy of Logistic Regression Model

The model achieved an accuracy of 92.71% on the training data.

Testing

The testing phase evaluates the model's performance and its ability to generalize to unseen data. Model evaluation is conducted using the same dataset, assessing accuracy and other relevant metrics. Although the specific code for hyperparameter tuning and detailed error analysis is not explicitly provided, the iterative refinement process involves adjusting the model based on testing results to enhance its performance. The validation step ensures that the model aligns with the project's objectives, providing meaningful and accurate classification to the users.

Random Forest

```
[35] # Print accuracy of model using Random Forest
print(f"\nRandom Forest Training Accuracy: {train_accuracy_rf*100:.2f}%")
print(f"Random Forest Testing Accuracy: {test_accuracy_rf*100:.2f}%")
print("\nRandom Forest Classification Report:\n", class_report_rf)
```



```
Random Forest Training Accuracy: 98.05%
Random Forest Testing Accuracy: 92.91%
```

```
Random Forest Classification Report:
```

| | precision | recall | f1-score | support |
|---------------------|-----------|--------|----------|---------|
| Insufficient Weight | 0.95 | 0.98 | 0.96 | 56 |
| Normal Weight | 0.86 | 0.79 | 0.82 | 62 |
| Obesity Type I | 0.96 | 0.95 | 0.95 | 78 |
| Obesity Type II | 0.97 | 0.97 | 0.97 | 58 |
| Obesity Type III | 1.00 | 1.00 | 1.00 | 63 |
| Overweight Level I | 0.83 | 0.88 | 0.85 | 56 |
| Overweight Level II | 0.92 | 0.94 | 0.93 | 50 |
| accuracy | | | 0.93 | 423 |
| macro avg | 0.93 | 0.93 | 0.93 | 423 |
| weighted avg | 0.93 | 0.93 | 0.93 | 423 |

Fig 4.5 Testing Accuracy and Classification Report using Random Forest Model

Random forest obtained a testing accuracy of 92.91%.

Logistic Regression

```

✓ [36] # Print accuracy of model using Logistic Regression
0s print(f"\nLogistic Regression Training Accuracy: {train_accuracy_lr*100:.2f}%")
    print(f"Logistic Regression Testing Accuracy: {test_accuracy_lr*100:.2f}%")
    print("\nLogistic Regression Classification Report:\n", class_report_lr)

```

Logistic Regression Training Accuracy: 92.71%
 Logistic Regression Testing Accuracy: 91.96%

Logistic Regression Classification Report:

| | precision | recall | f1-score | support |
|---------------------|-----------|--------|----------|---------|
| Insufficient Weight | 0.87 | 0.95 | 0.91 | 56 |
| Normal Weight | 0.92 | 0.76 | 0.83 | 62 |
| Obesity Type I | 0.95 | 0.99 | 0.97 | 78 |
| Obesity Type II | 1.00 | 1.00 | 1.00 | 58 |
| Obesity Type III | 1.00 | 1.00 | 1.00 | 63 |
| Overweight Level I | 0.80 | 0.91 | 0.85 | 56 |
| Overweight Level II | 0.89 | 0.80 | 0.84 | 50 |
| accuracy | | | 0.92 | 423 |
| macro avg | 0.92 | 0.91 | 0.91 | 423 |
| weighted avg | 0.92 | 0.92 | 0.92 | 423 |

Fig 4.6 Testing Accuracy and Classification Report using Logistic Regression Model

Logistic Regression obtained an accuracy of 91.96%.

5. RESULTS AND DISCUSSION

```

# Load the saved models and label encoders
rf_classifier = joblib.load('random_forest_obesity_model.pkl')
lr_classifier = joblib.load('logistic_regression_obesity_model.pkl')
label_encoders = joblib.load('label_encoders.pkl')

# Load the uploaded dataset
df = pd.read_csv('/content/drive/MyDrive/sy_obesity_dataset.csv')

# Renaming columns for readability
df.columns = ['Gender', 'Age', 'Height', 'Weight', 'Family History with Overweight',
              'Frequent consumption of high caloric food', 'Frequency of consumption of vegetables',
              'Number of main meals', 'Consumption of food between meals', 'Smoke',
              'Consumption of water daily', 'Calories consumption monitoring', 'Physical activity frequency',
              'Time using technology devices', 'Consumption of alcohol', 'Transportation used', 'Obesity']

# Encode the features using saved label encoders
for column in df.columns:
    if df[column].dtype == 'object':
        df[column] = label_encoders[column].transform(df[column])

# Split the data into features and target variable
X = df.drop('Obesity', axis=1) # Features
y = df['Obesity'] # Target variable

# Decode the actual labels from encoded values using the saved label encoders
actual_obesity = label_encoders['Obesity'].inverse_transform(y)

# Predict using the Random Forest model
predicted_obesity_rf = rf_classifier.predict(X)
predicted_obesity_rf = label_encoders['Obesity'].inverse_transform(predicted_obesity_rf)

# Predict using the Logistic Regression model
predicted_obesity_lr = lr_classifier.predict(X)
predicted_obesity_lr = label_encoders['Obesity'].inverse_transform(predicted_obesity_lr)

# Display the comparison of actual and predicted values
comparison_df = pd.DataFrame({
    'Actual Output': actual_obesity,
    'Predicted RF Output': predicted_obesity_rf,
    'Predicted LR Output': predicted_obesity_lr
})

# Set pandas options to display all rows in the DataFrame
pd.set_option('display.max_rows', None)

# Print the entire comparison DataFrame with all actual and predicted outputs
print(comparison_df)

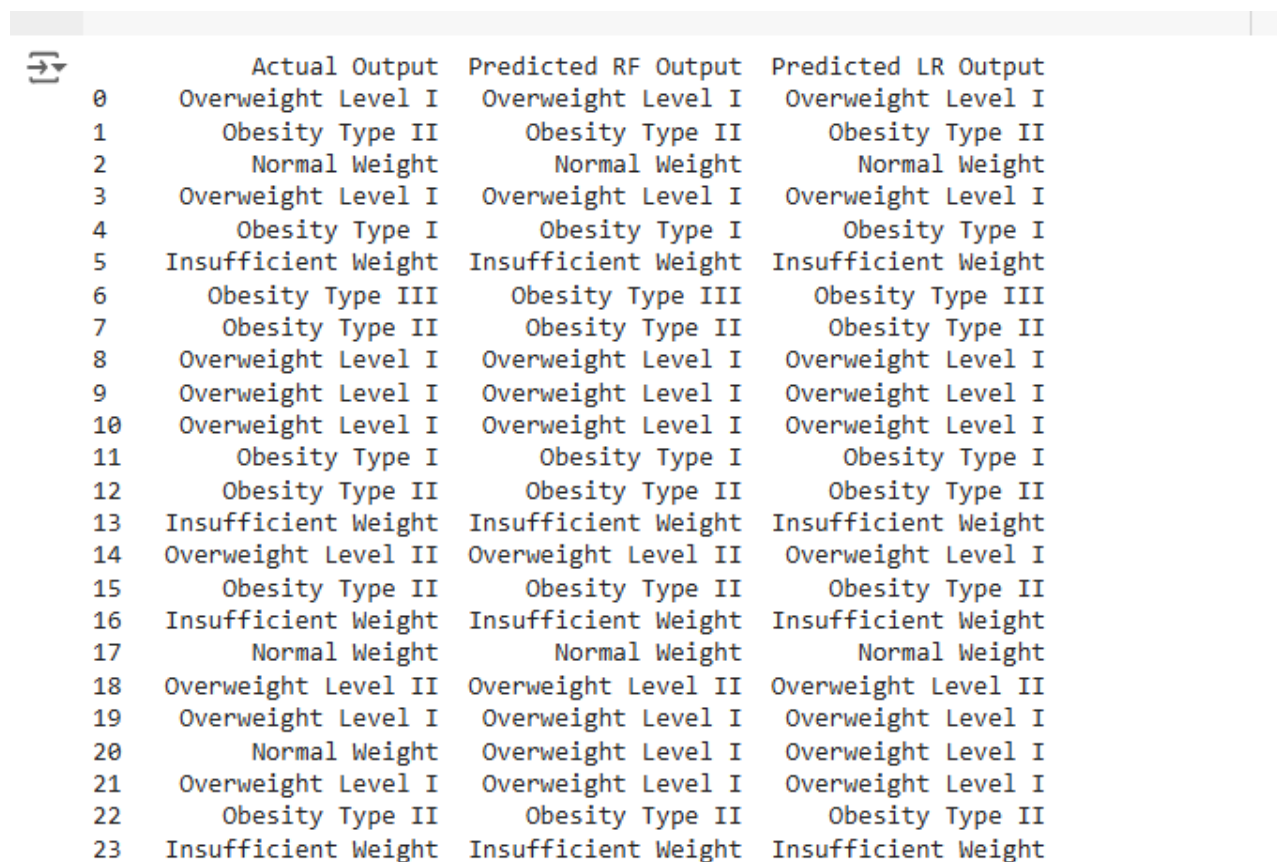
# Count the correct and incorrect predictions for Random Forest
correct_rf_count = (comparison_df['Actual Output'] == comparison_df['Predicted RF Output']).sum()
incorrect_rf_count = (comparison_df['Actual Output'] != comparison_df['Predicted RF Output']).sum()

# Count the correct and incorrect predictions for Logistic Regression
correct_lr_count = (comparison_df['Actual Output'] == comparison_df['Predicted LR Output']).sum()
incorrect_lr_count = (comparison_df['Actual Output'] != comparison_df['Predicted LR Output']).sum()

```

Fig 5.1 Code snippet of unseen data prediction

This code facilitates the loading and utilization of pre-trained Random Forest and Logistic Regression models, along with associated label encoders, for predicting obesity levels in a newly provided dataset. Initially, the dataset's column names are standardized for improved readability, and categorical features are encoded using the pre-saved label encoders to ensure consistency with the training data. The data is then partitioned into feature variables (X) and the target variable (y). To enable a comprehensive comparison, the actual obesity class labels are decoded, and predictions are generated using both models. The actual and predicted outputs are then compared, and the code further calculates and displays the count of correct and incorrect predictions for each model. This enables a detailed assessment of each model's performance in terms of prediction accuracy.



| | Actual Output | Predicted RF Output | Predicted LR Output |
|----|---------------------|---------------------|---------------------|
| 0 | Overweight Level I | Overweight Level I | Overweight Level I |
| 1 | Obesity Type II | Obesity Type II | Obesity Type II |
| 2 | Normal Weight | Normal Weight | Normal Weight |
| 3 | Overweight Level I | Overweight Level I | Overweight Level I |
| 4 | Obesity Type I | Obesity Type I | Obesity Type I |
| 5 | Insufficient Weight | Insufficient Weight | Insufficient Weight |
| 6 | Obesity Type III | Obesity Type III | Obesity Type III |
| 7 | Obesity Type II | Obesity Type II | Obesity Type II |
| 8 | Overweight Level I | Overweight Level I | Overweight Level I |
| 9 | Overweight Level I | Overweight Level I | Overweight Level I |
| 10 | Overweight Level I | Overweight Level I | Overweight Level I |
| 11 | Obesity Type I | Obesity Type I | Obesity Type I |
| 12 | Obesity Type II | Obesity Type II | Obesity Type II |
| 13 | Insufficient Weight | Insufficient Weight | Insufficient Weight |
| 14 | Overweight Level II | Overweight Level II | Overweight Level I |
| 15 | Obesity Type II | Obesity Type II | Obesity Type II |
| 16 | Insufficient Weight | Insufficient Weight | Insufficient Weight |
| 17 | Normal Weight | Normal Weight | Normal Weight |
| 18 | Overweight Level II | Overweight Level II | Overweight Level II |
| 19 | Overweight Level I | Overweight Level I | Overweight Level I |
| 20 | Normal Weight | Overweight Level I | Overweight Level I |
| 21 | Overweight Level I | Overweight Level I | Overweight Level I |
| 22 | Obesity Type II | Obesity Type II | Obesity Type II |
| 23 | Insufficient Weight | Insufficient Weight | Insufficient Weight |

fig 5.2 Result of unseen data prediction

```
# Print the counts
print("\nCounts of Predictions:")
print(f"Random Forest - Correct: {correct_rf_count}, Incorrect: {incorrect_rf_count}")
print(f"Logistic Regression - Correct: {correct_lr_count}, Incorrect: {incorrect_lr_count}")
```

Counts of Predictions:
Random Forest - Correct: 136, Incorrect: 4
Logistic Regression - Correct: 133, Incorrect: 7

fig 5.3 Count of predictions

Confusion Matrix

```
# Assuming you have the actual labels and predictions for Random Forest
# actual_obesity, predicted_obesity_rf

# Calculate confusion matrix for Random Forest
cm_rf = confusion_matrix(actual_obesity, predicted_obesity_rf, labels=label_encoders['Obesity'].classes_)

# Plot confusion matrix for Random Forest
plt.figure(figsize=(8, 6)) # Set the figure size
disp_rf = ConfusionMatrixDisplay(confusion_matrix=cm_rf, display_labels=label_encoders['Obesity'].classes_)
disp_rf.plot(cmap=plt.cm.Blues, ax=plt.gca()) # Plot on current axis

# Set titles and labels
plt.title('Confusion Matrix for Random Forest Model', fontsize=16) # Title
plt.xticks(rotation=45, fontsize=12) # Rotate x-tick labels
plt.yticks(fontsize=12) # Y-tick labels
plt.xlabel('Predicted Labels', fontsize=14) # X-axis label
plt.ylabel('True Labels', fontsize=14) # Y-axis label
plt.tight_layout() # Adjust layout
plt.show()
```

Fig 5.4 Code snippet of confusion matrix for Random Forest

The confusion matrix in Fig 5.5 shows the performance of a Random Forest model on predicting obesity categories. Each row represents the actual class (true labels), while each column represents the predicted class. The diagonal cells (from top left to bottom right) indicate correctly classified instances, while off-diagonal cells show misclassifications.

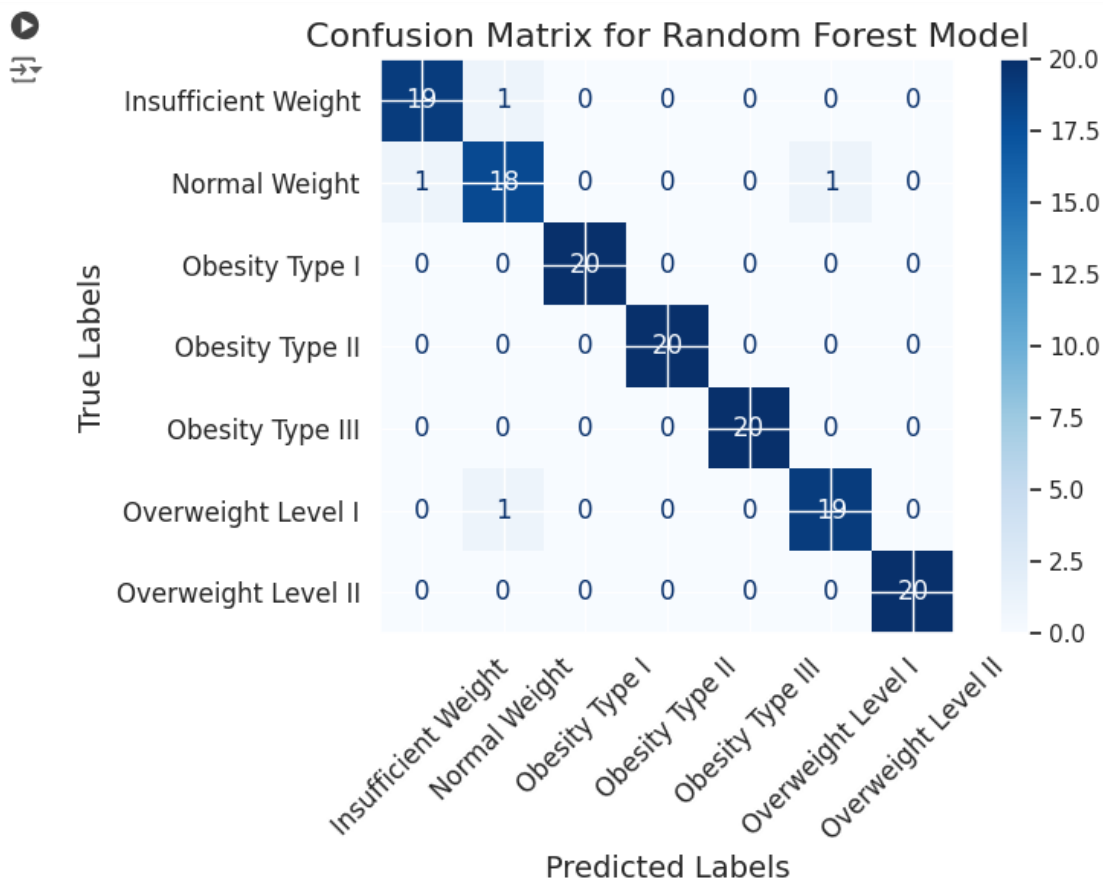


Fig 5.5 Confusion Matrix for Random Forest

```
# Assuming you have the actual labels and predictions for Logistic Regression
# actual_obesity, predicted_obesity_lr

# Calculate confusion matrix for Logistic Regression
cm_lr = confusion_matrix(actual_obesity, predicted_obesity_lr, labels=label_encoders['Obesity'].classes_)

# Plot confusion matrix for Logistic Regression
plt.figure(figsize=(8,6)) # Set the figure size
disp_lr = ConfusionMatrixDisplay(confusion_matrix=cm_lr, display_labels=label_encoders['Obesity'].classes_)
disp_lr.plot(cmap=plt.cm.Blues, ax=plt.gca()) # Plot on current axis

# Set titles and labels
plt.title('Confusion Matrix for Logistic Regression Model', fontsize=16) # Title
plt.xticks(rotation=45, fontsize=12) # Rotate x-tick labels
plt.yticks(fontsize=12) # Y-tick labels
plt.xlabel('Predicted Labels', fontsize=14) # X-axis label
plt.ylabel('True Labels', fontsize=14) # Y-axis label
plt.tight_layout() # Adjust layout
plt.show()
```

Fig 5.6 Code snippet of confusion matrix for Logistic Regression

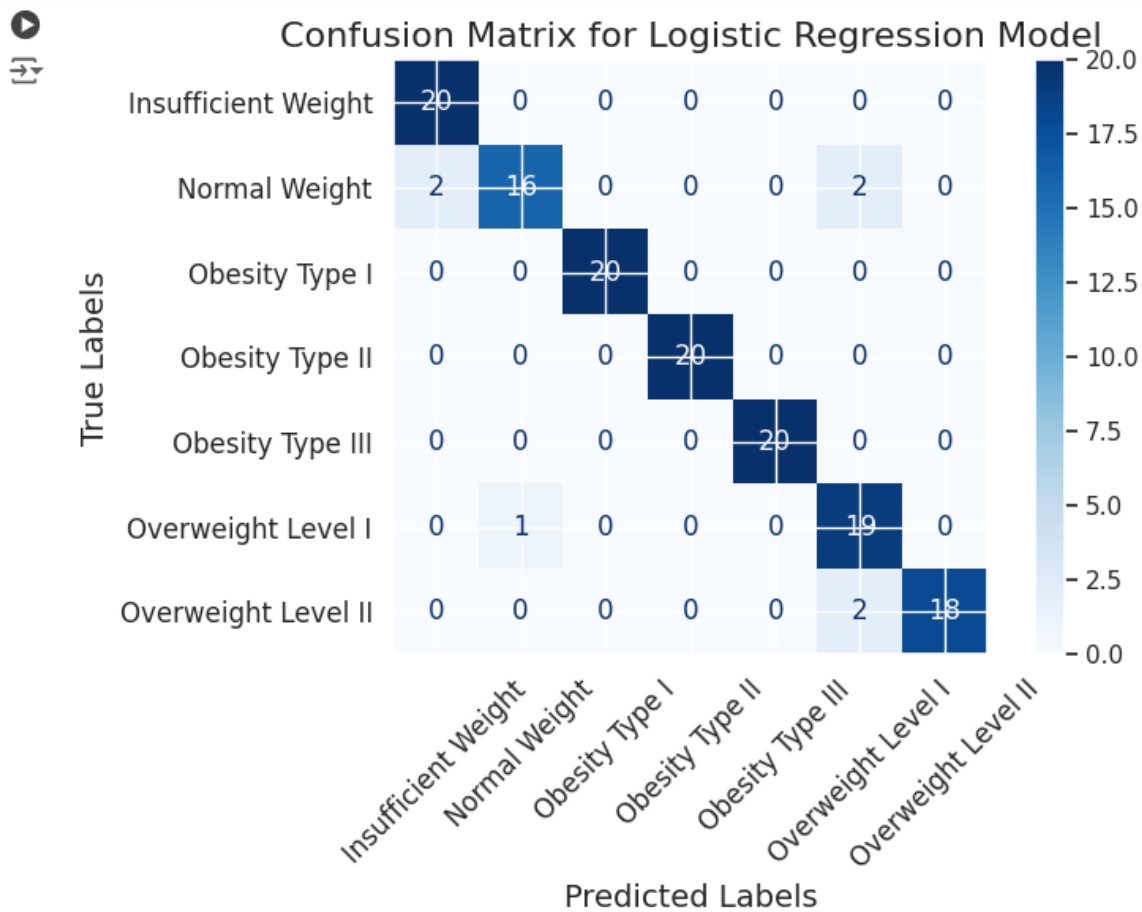


Fig 5.7 Confusion Matrix for Logistic Regression

This confusion matrix in Fig 5.7 shows the performance of a Logistic Regression model on predicting obesity categories. Each row represents the actual class (true labels), while each column represents the predicted class. The diagonal cells (from top left to bottom right) indicate correctly classified instances, while off-diagonal cells show misclassifications. This matrix suggests that the model is generally accurate, with only a few misclassifications, especially between closely related categories. The color intensity represents the count, with darker colors indicating higher values, and lighter colors indicating lower values or no misclassification.

6. MODEL DEPLOYMENT

The principal objective of model deployment is to seamlessly integrate the classification models developed during the project into a production environment. This transition empowers end-users to access and leverage the system for making informed decisions. Model deployment is a critical phase that bridges the gap between theoretical model development and practical, real-world application. By deploying the classification models, the project aims to provide a tangible and accessible solution that fulfills the needs and preferences of the target audience.

To facilitate the deployment process, the project utilizes [Specify the frameworks and libraries employed, e.g., Flask, scikit-learn, etc.]. These tools contribute to the efficiency and effectiveness of model integration into the production environment.

User Interface (UI):

The classification system is integrated into the user interface, creating a cohesive and user-friendly experience. Users can interact with the system through [Specify any UI elements, e.g., a web interface].

The following figures show the user interface of this application, which is designed to be simple and intuitive. The application opens with a clean, welcoming home page titled *Obesity Detection System*. A 'Get Started' button leads users to an input form that gathers essential data, including physical attributes, dietary habits, exercise frequency, time of technology devices using and types of transportation used. Upon submitting the form, the application classifies the user into one of several weight categories: Insufficient Weight, Normal Weight, Overweight Level 1, Overweight Level 2, Obesity Type 1, Obesity Type 2, Obesity Type or 3. A back button is provided to allow users to revisit the home page. This user-friendly design makes it easy to navigate and gain insights into personal health classifications.

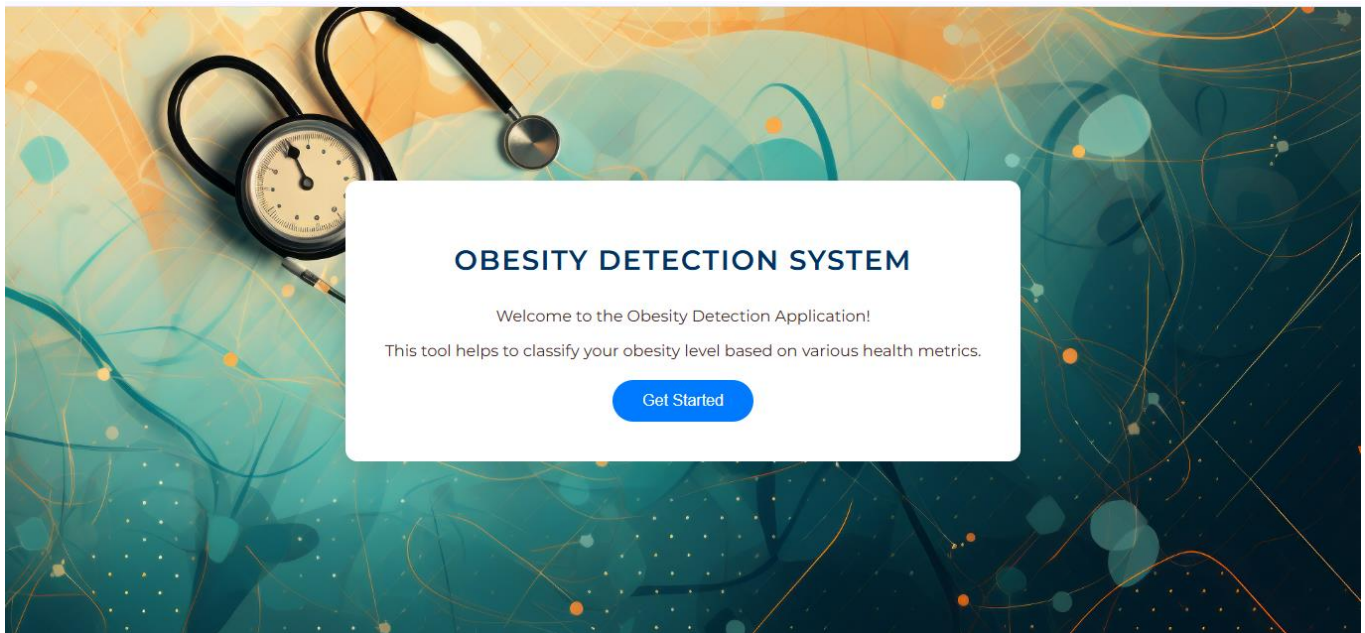
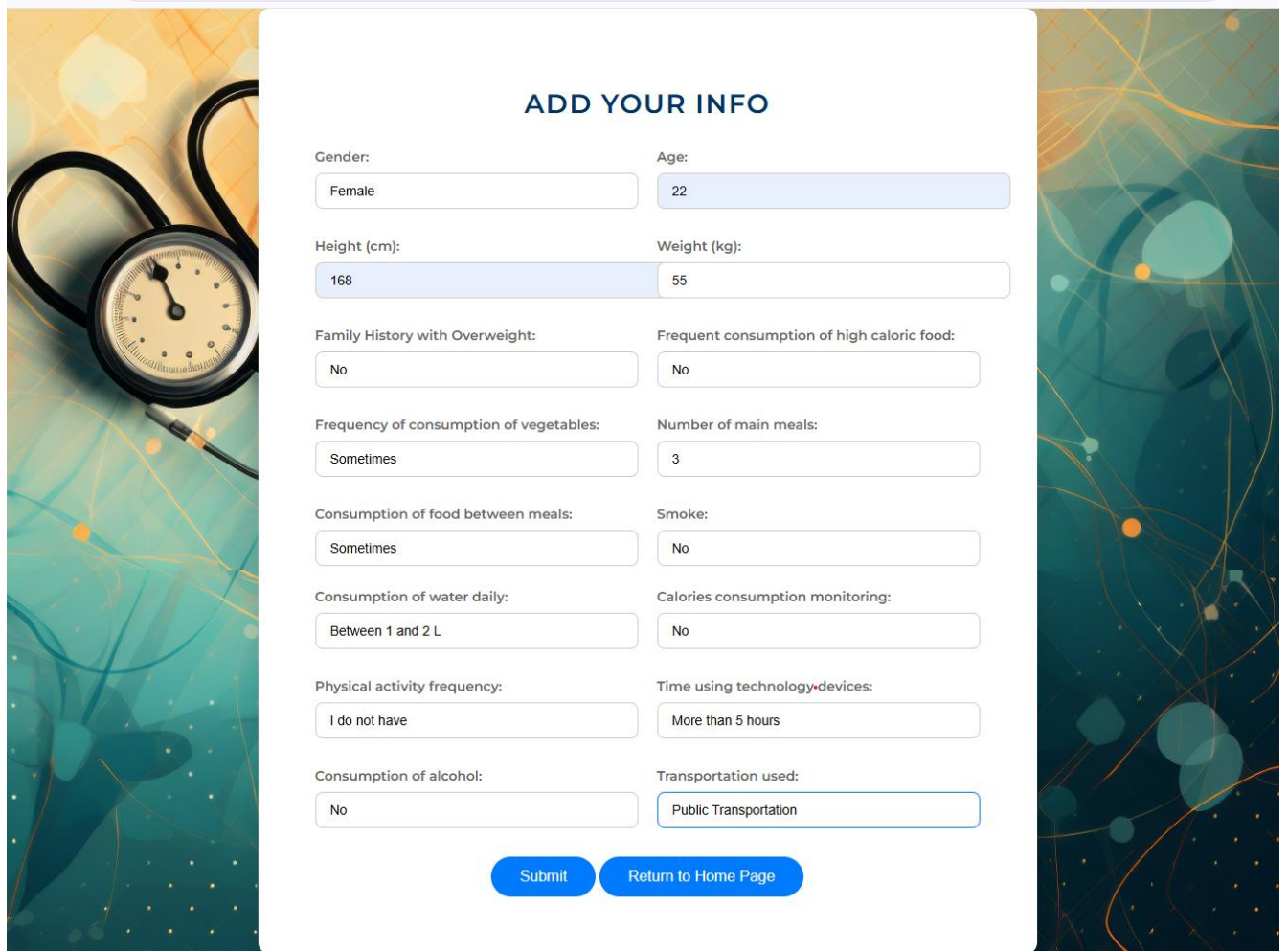


Fig 6.1 Homepage of Obesity Detection System

 The image shows the input page of the Obesity Detection System. It features a dark blue background with abstract orange and yellow patterns. A stethoscope is visible in the top left corner. A white rectangular box in the center contains the title "ADD YOUR INFO" in bold blue letters. Below the title, there are 16 input fields arranged in two columns. Each field has a label and a "Select Option" or "Enter your..." prompt. The fields are: Gender, Age, Height (cm), Weight (kg), Family History with Overweight, Frequent consumption of high caloric food, Frequency of consumption of vegetables, Number of main meals, Consumption of food between meals, Smoke, Consumption of water daily, Calories consumption monitoring, Physical activity frequency, Time using technology devices, Consumption of alcohol, and Transportation used. At the bottom of the box are two blue buttons labeled "Submit" and "Return to Home Page".

Fig 6.2 Input page of Obesity Detection System



ADD YOUR INFO

| | |
|---|--|
| Gender: <input type="text" value="Female"/> | Age: <input type="text" value="22"/> |
| Height (cm): <input type="text" value="168"/> | Weight (kg): <input type="text" value="55"/> |
| Family History with Overweight: <input type="text" value="No"/> | Frequent consumption of high caloric food: <input type="text" value="No"/> |
| Frequency of consumption of vegetables: <input type="text" value="Sometimes"/> | Number of main meals: <input type="text" value="3"/> |
| Consumption of food between meals: <input type="text" value="Sometimes"/> | Smoke: <input type="text" value="No"/> |
| Consumption of water daily: <input type="text" value="Between 1 and 2 L"/> | Calories consumption monitoring: <input type="text" value="No"/> |
| Physical activity frequency: <input type="text" value="I do not have"/> | Time using technology devices: <input type="text" value="More than 5 hours"/> |
| Consumption of alcohol: <input type="text" value="No"/> | Transportation used: <input type="text" value="Public Transportation"/> |

Fig 6.3 User Input of Obesity Detection System

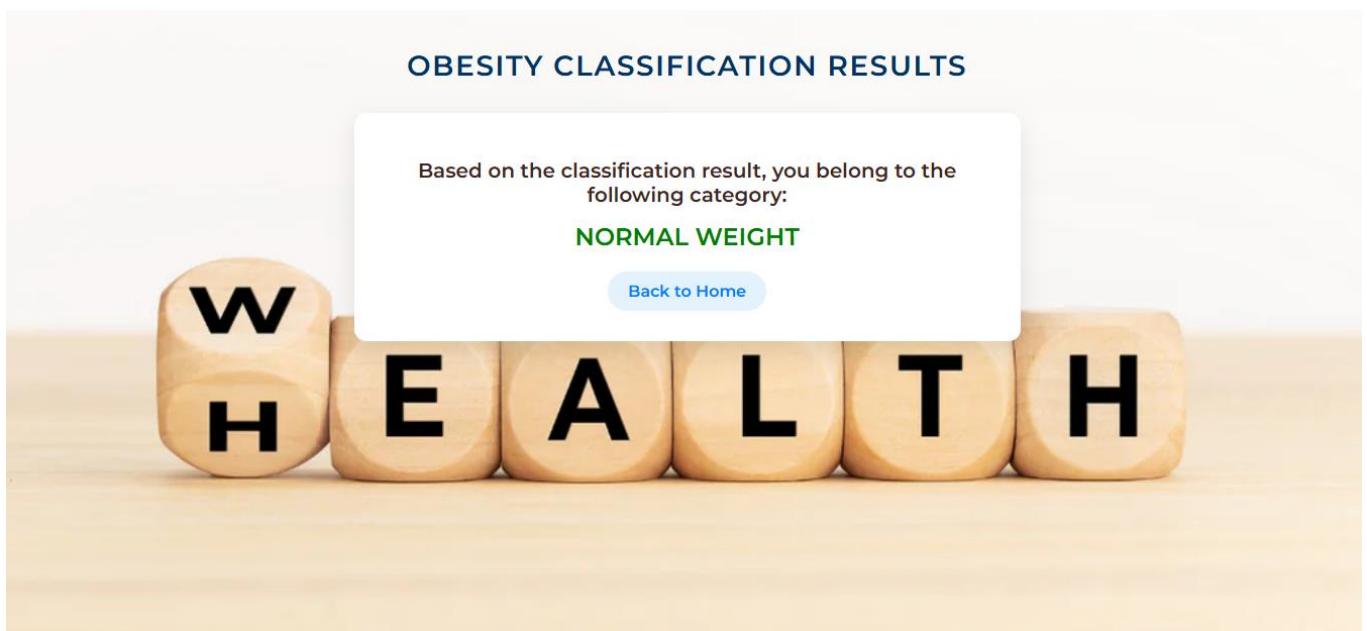
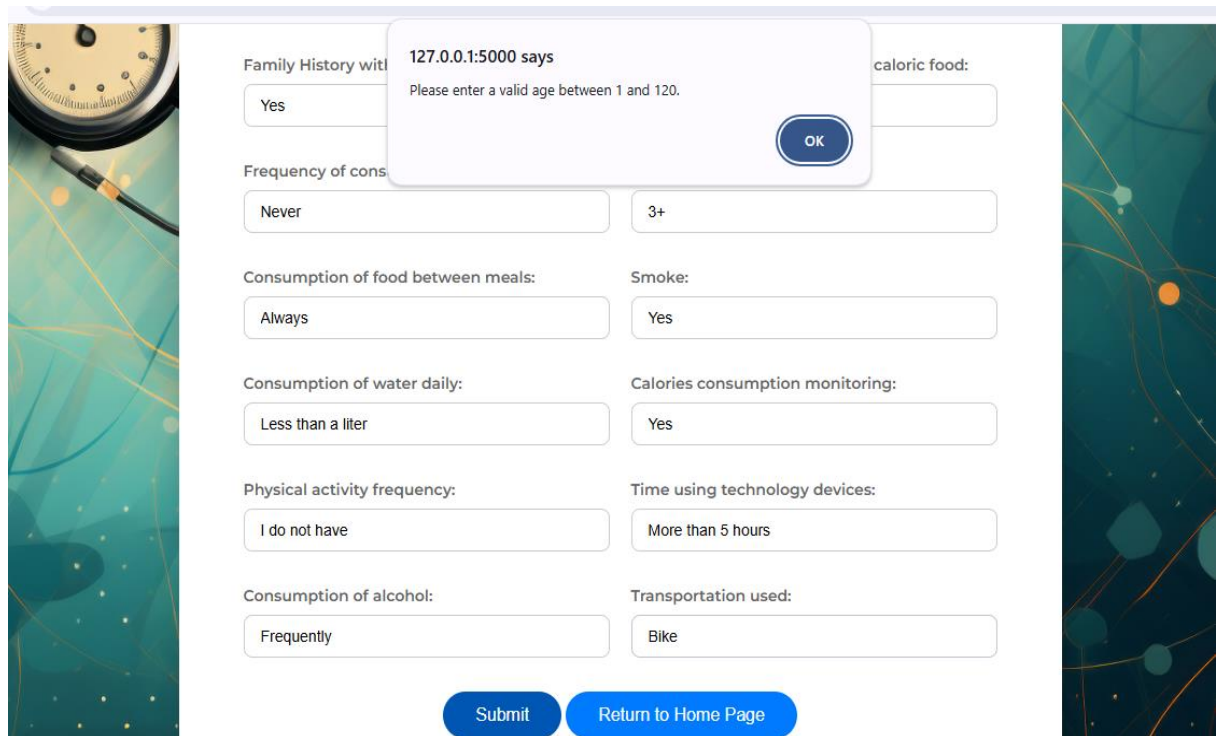


Fig 6.4 Result of Obesity Detection System

Validation is done for age, height and weight

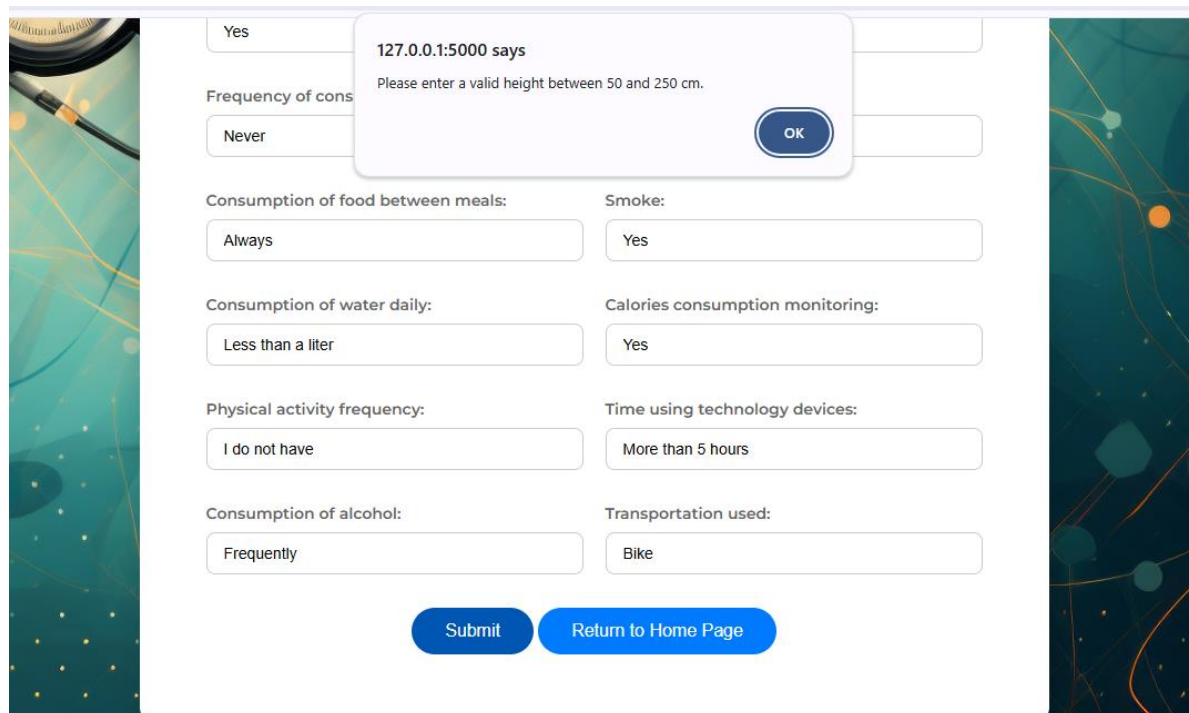


The screenshot shows a web form for obesity detection. A modal dialog box is displayed in the center, indicating a validation error for the age field. The dialog text reads: "127.0.0.1:5000 says Please enter a valid age between 1 and 120." with an "OK" button. The form fields and their current values are as follows:

| Field | Value |
|------------------------------------|-------------------|
| Family History with | Yes |
| Frequency of consumption | Never |
| Consumption of food between meals: | Always |
| Smoke: | Yes |
| Consumption of water daily: | Less than a liter |
| Calories consumption monitoring: | Yes |
| Physical activity frequency: | I do not have |
| Time using technology devices: | More than 5 hours |
| Consumption of alcohol: | Frequently |
| Transportation used: | Bike |

At the bottom of the form, there are two buttons: "Submit" and "Return to Home Page".

Fig 6.5 Validation for age



The screenshot shows the same web form as in Fig 6.5, but with a validation error for the height field. The modal dialog box text reads: "127.0.0.1:5000 says Please enter a valid height between 50 and 250 cm." with an "OK" button. The form fields and their current values are as follows:

| Field | Value |
|------------------------------------|-------------------|
| Family History with | Yes |
| Frequency of consumption | Never |
| Consumption of food between meals: | Always |
| Smoke: | Yes |
| Consumption of water daily: | Less than a liter |
| Calories consumption monitoring: | Yes |
| Physical activity frequency: | I do not have |
| Time using technology devices: | More than 5 hours |
| Consumption of alcohol: | Frequently |
| Transportation used: | Bike |

At the bottom of the form, there are two buttons: "Submit" and "Return to Home Page".

Fig 6.6 Validation for height

The screenshot shows a web application interface for obesity detection. A modal dialog box is displayed in the center, indicating a validation error for the weight input. The dialog text reads: "127.0.0.1:5000 says Please enter a valid weight between 10 and 300 kg." with an "OK" button. The background form contains several input fields for user data:

- Frequency of consumption: Yes, Never
- Consumption of food between meals: Always
- Smoke: Yes
- Consumption of water daily: Less than a liter
- Calories consumption monitoring: Yes
- Physical activity frequency: I do not have
- Time using technology devices: More than 5 hours
- Consumption of alcohol: Frequently
- Transportation used: Bike

At the bottom of the form, there are two buttons: "Submit" and "Return to Home Page". The interface is decorated with a teal and orange abstract pattern on the sides.

Fig 6.7 Validation for weight

7. GIT HISTORY

Git Repository Obesity Detection System contains all colab files, py files, html files and three related research papers. It is maintained for systematic way of project presentation and mainly for future reference. The colab files are created separately for three sprint releases during the project.

MINI-PROJECT Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch Tags Go to file Add file Code

| File | Commit | Time Ago | Commits |
|-------------------------------|---------|---------------|---------|
| DATASET | 9df2a16 | 2 minutes ago | 3 |
| JOURNAL | | 3 months ago | |
| Sprint 1 | | 5 minutes ago | |
| Sprint 2 | | 2 minutes ago | |
| Sprint 3 | | 2 minutes ago | |
| OBEsITY_DETECTION_SUMMARY.pdf | | 3 months ago | |

About

The Obesity Detection Using ML project aims to develop a machine learning model to classify individuals based on their obesity levels using various health and lifestyle features. The project utilizes a dataset containing demographic, behavioral factors, eating and health related factors.

Activity 0 stars 1 watching 0 forks

Fig 7.1 Git History of all Sprints

MINI-PROJECT / Sprint 3

Add file ...

| Name | Last commit message | Last commit date |
|---------------------------------|----------------------|------------------|
| .. | | |
| images | Add files via upload | 3 minutes ago |
| templates | Add files via upload | 3 minutes ago |
| app.py | Add files via upload | 3 minutes ago |
| label_encoders.pkl | Add files via upload | 3 minutes ago |
| random_forest_obesity_model.pkl | Add files via upload | 3 minutes ago |

About

The Obesity Detection Using ML project aims to develop a machine learning model to classify individuals based on their obesity levels using various health and lifestyle features. The project utilizes a dataset containing demographic, behavioral factors, eating and health related factors.

Activity 0 stars 1 watching 0 forks

Fig 7.2 Git History of sprint 3

8. CONCLUSION

In conclusion, the Obesity Detection using ML project has successfully achieved its primary objective of improving health insights and enabling proactive interventions through the development and deployment of accurate prediction models. Leveraging Logistic Regression and Random Forest algorithms, the system excels in accurately classifying obesity levels based on various demographic and lifestyle factors, thus providing valuable insights into individuals' health status.

The project's technical contributions, including algorithmic precision and adaptability, ensure its effectiveness in handling diverse data profiles and evolving health trends. The user-centric approach integrates a user-friendly interface, facilitating ease of interaction and interpretation for healthcare providers and individuals alike. The successful deployment of production-ready models, API endpoints, and frontend integration underscores the project's transition from theoretical analysis to practical, real-world application. Enhanced security measures, such as data protection and privacy controls, along with real-time monitoring, contribute to the system's reliability and compliance. Lessons learned from this project pave the way for future improvements, with plans for scalability and feature updates to support continuous advancement.

The Obesity Detection system stands as a practical application of data science and machine learning, poised to contribute to better-informed public health strategies and more personalized wellness interventions, meeting the dynamic needs of health monitoring and obesity management.

9. FUTURE WORK

- **Enhanced Feature Engineering:**

Further exploration of additional features such as genetic factors, metabolic rates, and psychological attributes could improve the model's predictive accuracy and relevance.
- **Integration of Advanced Algorithms:**

Implementing and evaluating more advanced machine learning and deep learning algorithms, such as Gradient Boosting Machines (GBMs) and Convolutional Neural Networks (CNNs), may increase prediction accuracy and uncover complex feature interactions.
- **Real-Time Monitoring and Prediction:**

Developing a real-time monitoring system that collects and processes data from wearable devices, health apps, and lifestyle logs can offer timely obesity predictions and allow for personalized intervention.
- **Explainable AI (XAI) Implementation:**

Applying explainable AI techniques, like SHAP (SHapley Additive exPlanations), could provide valuable insights into which factors contribute most to obesity, making the model's predictions more interpretable and actionable for healthcare professionals.
- **Personalized Intervention Plans:**

Leveraging machine learning to design customized intervention plans based on individual risk factors, such as dietary habits, genetic predispositions, and lifestyle patterns. Integrating reinforcement learning could enable the model to adapt and optimize classifications based on user progress.
- **Predictive Health Analytics for Comorbid Conditions:**

Extending the model to predict the likelihood of developing obesity-related comorbidities like Type 2 diabetes, hypertension, and cardiovascular diseases, providing healthcare providers with a more comprehensive risk profile and potential preemptive interventions.
- **Nutritional Recommendation Systems:**

Developing a dynamic, AI-driven nutrition recommendation system that adjusts dietary suggestions based on the user's current health metrics, obesity risk level, and other lifestyle factors, possibly integrating with grocery delivery apps for convenience.

- **Integrative Biomarker Analysis:**
Incorporating biomarkers from blood tests (e.g., glucose levels, lipid profiles) to enhance the prediction model's accuracy and reliability, potentially providing real-time feedback on how lifestyle changes affect obesity risk and health.
- **Collaborative Filtering for Community-Based Recommendations:**
Using collaborative filtering techniques to create community-driven health and wellness recommendations, identifying users with similar health profiles and suggesting beneficial practices observed in successful cases.
- **Augmented Reality (AR) Health Coaching:**
Implementing an AR-based health coaching experience that provides real-time dietary and exercise advice in users' daily environments, encouraging healthier habits and visualizing potential outcomes through AR feedback.
- **Longitudinal Health Monitoring and Prediction:**
Implementing a system for continuous health monitoring that combines obesity detection with predictive analytics to forecast long-term health outcomes, enabling users and healthcare providers to take preventive actions for sustained health improvements.
- **Ethical AI for Health Disparity Analysis:**
Employing machine learning to analyze how socioeconomic, environmental, and cultural factors impact obesity trends, identifying health disparities and informing policies that promote equitable healthcare access and intervention resources across diverse populations.

In summary, the *Obesity Detection Using Machine Learning* project utilizes predictive models, specifically Random Forest and Logistic Regression, to classify individuals into obesity categories based on lifestyle, demographic, and dietary data. This approach helps identify obesity risk factors through data-driven insights and holds potential for future enhancements, including real-time monitoring, personalized intervention plans, and integration with wearable devices. Ultimately, the project not only aids individuals in managing obesity risks but also supports public health by identifying trends and enabling targeted interventions, making it a valuable tool in both personalized and community-level healthcare.

10. APPENDIX

Minimum Software Requirements

To deploy and run the Obesity Detection System, the following minimum software requirements must be met:

- Python:

Version 3.6 or above is required to execute the project code and its dependencies.

- NumPy:

Essential for numerical operations and array manipulations in Python.

- Pandas:

Necessary for data manipulation, cleaning, and analysis.

- Scikit-learn:

The scikit-learn library is used for machine learning functionalities, including the implementation of the Random Forest and Logistic Regression algorithm.

- Matplotlib and Seaborn:

Required for data visualization and generating plots and charts.

- SciPy:

Used for scientific and technical computing, particularly for sparse matrix operations.

- TQDM:

Essential for displaying progress bars during time-consuming operations.

- Pickel:

Used for serialization and deserialization of Python objects, particularly for saving and loading machine learning models.

- Colab Notebook:

If using Colab notebooks for development and testing, having Colab installed is recommended.

- Other Dependencies:

Additional libraries and modules specified in the project code, including Random Forest and Logistic Regression algorithms, Scipy's sparse matrix (`csr_matrix`), and other utility modules.

Minimum Hardware Requirements

The Obesity Detection System is relatively lightweight, but it does involve some computational processes. The minimum hardware requirements to ensure smooth execution are as follows:

Processor (CPU):

A multi-core processor (e.g., Intel Core i3 or equivalent) is recommended to handle the computation involved in the classification algorithms efficiently.

RAM:

A minimum of 4 GB RAM is recommended to manage the dataset and perform machine learning operations smoothly.

Storage:

Adequate storage space for storing datasets, code, and any additional resources. While the system itself doesn't require extensive storage, having sufficient space for datasets and potential model persistence is essential.

Internet Connection:

An internet connection is necessary for downloading datasets, libraries, and dependencies during the setup phase.

11. REFERENCES

- Ferdowsy, Faria, Kazi Samsul Alam Rahi, Md Ismail Jabiullah, and Md Tarek Habib."A machine learning approach for obesity risk prediction." *Current Research in Behavioral Sciences* 2 (2021): 100053.
- Rodríguez, Elias, Elen Rodríguez, Luiz Nascimento, Aneirson Francisco da Silva, and Fernando Augusto Silva Marins. "Machine learning Techniques to Predict Overweight or Obesity." In IDDM, pp. 190-204. 2021.
- Yagin, F. H., Güllü, M., Gormez, Y., Castañeda-Babarro, A., Colak, C., Greco, G., ... & Cataldi, S. (2023). Estimation of obesity levels with a trained neural network approach optimized by the Bayesian technique. *Applied Sciences*, 13(6), 3875
- *Python for Data Analysis* by Wes McKinney
- *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* by Aurélien Géron