**MS2 Report: Miracle Weight-Loss Drug  
Team 5**

Course: RSE4207 Artificial Intelligence and Machine Learning

Date: 06/11/2024

Team Members:

Kavyanjali 2203390

Koh Chun Ince 2200960

Caleb Lee Jia Le 2201250

Muhammad Muslinmin Bin Mohd Saleh 2201360

Reuben Low Yu Xiang 2200931

Stafford Ho Sheng Xian 2201218

Yugendren S/o Sooriya Moorthi 2201512

[1 Introduction 6](#_Toc183028686)

[1.1 Project Objective 6](#_Toc183028687)

[2 Exploratory Data Analysis 7](#_Toc183028688)

[2.1 Analysis Activities 9](#_Toc183028689)

[2.1.1 Statistical Summary and Distribution Analysis 9](#_Toc183028690)

[2.1.1.1 Mean (Average) 10](#_Toc183028691)

[2.1.1.3 Minimum and Maximum 10](#_Toc183028692)

[2.1.1.4 Standard Deviation (Measure of Spread) 11](#_Toc183028693)

[2.1.1.5 Potential Hypotheses Based on This Analysis 11](#_Toc183028694)

[2.1.2 Correlation Analysis 11](#_Toc183028695)

[2.1.2.1 Relationship between Height, Weight and Obesity Levels 12](#_Toc183028696)

[2.1.2.2 Relationship between Age, Weight and Obesity Levels 14](#_Toc183028697)

[2.1.2.3 Relationship between Age, Height and Obesity Levels 16](#_Toc183028698)

[2.1.2.4 Frequency of High-Calorie Food Consumption (FAVC) 20](#_Toc183028699)

[2.1.2.5 Number of Main Meals per Day (NCP) 22](#_Toc183028700)

[2.1.2.6 Additional Consumption of Food in between meals (CAEC) 24](#_Toc183028701)

[2.1.2.7 Self Monitoring of Calorie Consumption (SCC) 25](#_Toc183028702)

[2.1.2.8 Physical Activity Frequency (FAF) 26](#_Toc183028703)

[2.1.2.9 Technology Use (TUE) 28](#_Toc183028704)

[2.1.2.10 Mode of Transportation (MTRANS) 29](#_Toc183028705)

[2.1.2.11 Frequency of alcohol consumption(CALC) 31](#_Toc183028706)

[2.1.2.12 Patient who smokes vs obesity 33](#_Toc183028707)

[2.1.2.13 Gender comparison of overweight and normal/underweight 34](#_Toc183028708)

[2.1.3 Multivariate Analysis 36](#_Toc183028709)

[2.1.3.1 BMI and Obesity Levels 36](#_Toc183028710)

[2.1.3.2 Body Fat Percentage and Obesity Levels 38](#_Toc183028711)

[2.2 Processing Activities 40](#_Toc183028712)

[2.2.1 Feature Selection 40](#_Toc183028713)

[2.2.2 Encoding Categorical Variables 41](#_Toc183028714)

[2.2.2.1 Gender 41](#_Toc183028715)

[2.2.2.2 FAVC (High-Calorie Food Consumption) 41](#_Toc183028716)

[2.2.2.3 CAEC (Food Consumption Between Meals) 42](#_Toc183028717)

[2.2.2.4 SMOKE 42](#_Toc183028718)

[2.2.2.5 SCC (Self-Monitoring of Calories) 42](#_Toc183028719)

[2.2.2.6 CALC (Alcohol Consumption Frequency) 42](#_Toc183028720)

[2.2.2.7 MTRANS (Mode of Transportation) 42](#_Toc183028721)

[2.2.2.8 Obesity Level 43](#_Toc183028722)

[2.2.3 Feature Engineering (Creating New Features) 43](#_Toc183028723)

[2.2.3.1 BMI (Body Mass Index) 44](#_Toc183028724)

[2.2.3.2 Body Fat Percentage 44](#_Toc183028725)

[2.3 Orange Pipeline 44](#_Toc183028726)

[2.4 Code pipeline 46](#_Toc183028727)

[2.4.1 Directory Structure Overview 46](#_Toc183028728)

[2.4.2 Pipeline Flow Description 47](#_Toc183028729)

[2.4.3 Execution Notes 48](#_Toc183028730)

[2.5 CLI 49](#_Toc183028731)

[3 Model Training 52](#_Toc183028732)

[3.1 Models assessed 52](#_Toc183028733)

[3.1.1 Ensemble Method 52](#_Toc183028734)

[3.1.1.2 Adaptive boosting 53](#_Toc183028735)

[3.1.1.3 Gradient Boosting 53](#_Toc183028736)

[3.3.1.4 XGBoost 54](#_Toc183028737)

[3.1.2 K-Nearest Neighbors (KNN) 55](#_Toc183028738)

[3.1.3 Support Vector Classifier (SVC) 55](#_Toc183028739)

[3.1.4 Decision Tree 56](#_Toc183028740)

[3.1.5 Logistic Regression 57](#_Toc183028741)

[3.1.6 Naive Bayes (GaussianNB) 57](#_Toc183028742)

[3.1.7 Neural Network (MLPClassifier) 58](#_Toc183028743)

[3.2 Hypothesis: Random Forest as the Optimal Model for This Use Case 59](#_Toc183028744)

[3.2.1 Explanation Based on EDA Insights: 59](#_Toc183028745)

[3.2.2 Justification of Hyperparameters Based on Data: 59](#_Toc183028746)

[3.2.3 Conclusion 60](#_Toc183028747)

[3.2 Training process 61](#_Toc183028748)

[3.2.1 Feature Engineering 61](#_Toc183028749)

[3.2.2 Feature Selection Strategy 62](#_Toc183028750)

[3.2.3 Testing and Evaluation 63](#_Toc183028751)

[3.3 Expected Outcomes 63](#_Toc183028752)

[3.3.1 Model Fitting: 64](#_Toc183028753)

[3.3.2 Model Evaluation 66](#_Toc183028754)

[3.3.3 Expected Performance Benchmarks 67](#_Toc183028755)

[3.3.4 Multi-Class Classification Score Difference 67](#_Toc183028756)

[4 Model Testing 68](#_Toc183028757)

[4.1 Overfitting and Underfitting 68](#_Toc183028758)

[4.2 Results 70](#_Toc183028759)

[4.2.1 Analysis of Model Performance on Training and Testing Data 70](#_Toc183028760)

[4.2.2 Model Tuning with Cross-Validation 70](#_Toc183028761)

[4.2.3 Parameter Settings and Feature Utilization 70](#_Toc183028762)

[4.2.4 Hypothesis Confirmation: Random Forest's Superior Performance 71](#_Toc183028763)

[4.3 Model Scores 71](#_Toc183028764)

[4.3.1 AdaBoost 71](#_Toc183028765)

[4.3.2 DecisionTree 73](#_Toc183028766)

[4.3.3 GradientBoosting 75](#_Toc183028767)

[4.3.4 KNeighbors 76](#_Toc183028768)

[4.3.5 LogisticRegression 77](#_Toc183028769)

[4.3.6 NaiveBayes 78](#_Toc183028770)

[4.3.7 NeuralNetwork 79](#_Toc183028771)

[4.3.8 SVC 80](#_Toc183028772)

[4.3.9 XGBoost 81](#_Toc183028773)

[4.3.10 RandomForest 82](#_Toc183028774)

[4.3.10.1 Training Data Insights 82](#_Toc183028775)

[4.3.10.2 Testing Data Insights 82](#_Toc183028776)

[4.3.10.3 Comparison with Other Models 82](#_Toc183028777)

[4.3.10.4 Suitability for Prediction Use Case for obesity levels in patients 83](#_Toc183028778)

[4.3.10.5 Comparison of Feature Variations in the Random Forest Model 84](#_Toc183028779)

[4.3.10.5.1 Performance Summary 84](#_Toc183028780)

[4.3.10.5.2 Explanation of Differences 88](#_Toc183028781)

[4.3.10.6 Conclusion 88](#_Toc183028782)

[4.3.11 Random Forest Visualization 89](#_Toc183028783)

[4.3.11.1 ROC Curve Analysis and Model Performance 89](#_Toc183028784)

[4.3.11.2 Analysis of the Confusion Matrix 90](#_Toc183028785)

[4.3.11.3 Feature Importances 92](#_Toc183028786)

[4.3.11.4 Model Evaluation and Recommendation for Obesity Level Prediction 93](#_Toc183028787)

[4.3.11.4.1 Key Findings: 94](#_Toc183028788)

[4.3.11.4.2 Limitations 94](#_Toc183028789)

[4.3.11.4.3 Recommendation: 95](#_Toc183028790)

[5 References 96](#_Toc183028791)

[6 Appendix 98](#_Toc183028792)

[A1 Model Metrics 98](#_Toc183028793)

## 

## 1 Introduction

The pharmaceutical industry relies on rigorous data analysis to evaluate the efficacy and safety of new treatments. For our client, introducing a groundbreaking "miracle drug" aimed at combating obesity represents a pivotal opportunity. To ensure the drug's success, a robust data-driven approach is essential for evaluating patient outcomes and optimizing clinical trial protocols.

This project focuses on analyzing patient data from trials conducted over a year, uncovering the key factors influencing the drug's effectiveness. By leveraging exploratory data analysis (EDA) and machine learning (ML), we aim to deliver actionable insights that will support strategic decision-making and drive the commercialization of the drug.

The drug is designed to halt or reverse obesity's effects, demonstrating high efficacy in patients who:

* Have a high probability of developing obesity.
* Are developing obesity.

However, the drug has severe adverse side effects on individuals with low or no probability of becoming obese. Thus, precise classification of patients based on obesity levels is critical to avoid unintended harm and maximize the drug's potential impact.

### 1.1 Project Objective

The goals of this project are as follows:

* **Exploratory Data Analysis (EDA):** 
  + Analyze patient data to identify trends and factors influencing weight loss outcomes, such as age, gender, lifestyle, and pre-existing health conditions.
  + Gain insights into why certain patients respond more positively to the drug than others.
* **Machine Learning Model Development (ML):**
  + Develop a predictive model to estimate the likelihood of success for individual patients.
  + Enable targeted patient selection for future trials and optimize treatment plans.
* **Operational Efficiency:**
  + Design an intuitive software tool that integrates seamlessly into the client’s existing systems, enabling quick and accurate decision-making while complying with data privacy standards.

Additionally, the project will focus on designing a program to assist the client in deciding which patients are most likely to benefit from the drug. For example,if a patient is at a higher risk of obesity based on their characteristics such as weight, age, height, etc, they may be prioritized for the miracle drug administered.

1.2 Background

The scenario outlined in this project involves a business client of Data Mine Craft DMC, a company specializing in generating analytics from complex datasets. The client is conducting clinical trials for an early prevention drug aimed at combating obesity. These trials are crucial for evaluating the drug's efficacy and safety, as well as identifying patient-specific factors that influence its effectiveness.

In the context of clinical trials, understanding patient-specific responses to treatments is critical for ensuring the success of new drugs. In this project, the dataset provided by the client contains confidential details about patients' health, lifestyle, diet, and treatment outcomes over a one-year period.

Given the high stakes of clinical trials, this project focuses on analyzing the dataset to:

* Identify key factors that influence the likelihood of weight reduction.
* Predict patient suitability for the drug, ensuring its benefits are maximized while minimizing adverse effects.
* Support evidence-based decision-making to optimize the design and targeting of future clinical trials.

Through data analytics and machine learning, the project aims to deliver precise, reliable predictions to aid the client in making informed decisions, ultimately enhancing the drug’s success and patient outcomes.

### 

## 2 Exploratory Data Analysis

The goal of the Exploratory Data Analysis (EDA) for this project is to understand the underlying characteristics and patterns within the clinical trial dataset. This analysis will identify key features that influence the effectiveness of the drug and provide insights for optimizing the treatment and patient selection process. EDA aims to:

1. **Identify Patterns and Trends:** Examine relationships between various patient attributes (e.g., age, gender, diet, and activity level) and their weight loss outcomes to understand which factors significantly impact the drug’s effectiveness.
2. **Handle Missing and Anomalous Data:** Identify and address any missing or anomalous values that may distort the analysis and model training.
3. **Feature Selection:** Select relevant features, such as caloric intake (FAVC, FCVC), physical activity (FAF), and lifestyle habits (SMOKE, TUE), that provide valuable insights into weight loss and exclude redundant or irrelevant variables.
4. **Visualize Data for Insights:** Generate visualizations to highlight key relationships, such as weight loss outcomes by age group, dietary habits, or obesity level. These visualizations will enhance interpretability and provide clear insights for stakeholders.

This analysis will be guided by hypotheses testing specific factors potentially influencing weight loss. Each hypothesis will focus on measurable outcomes, such as the impact of frequent vegetable consumption (FCVC) or physical activity frequency (FAF) on patient success. By evaluating these hypotheses with evidence, actionable insights will be extracted to inform clinical strategies and decision-making.

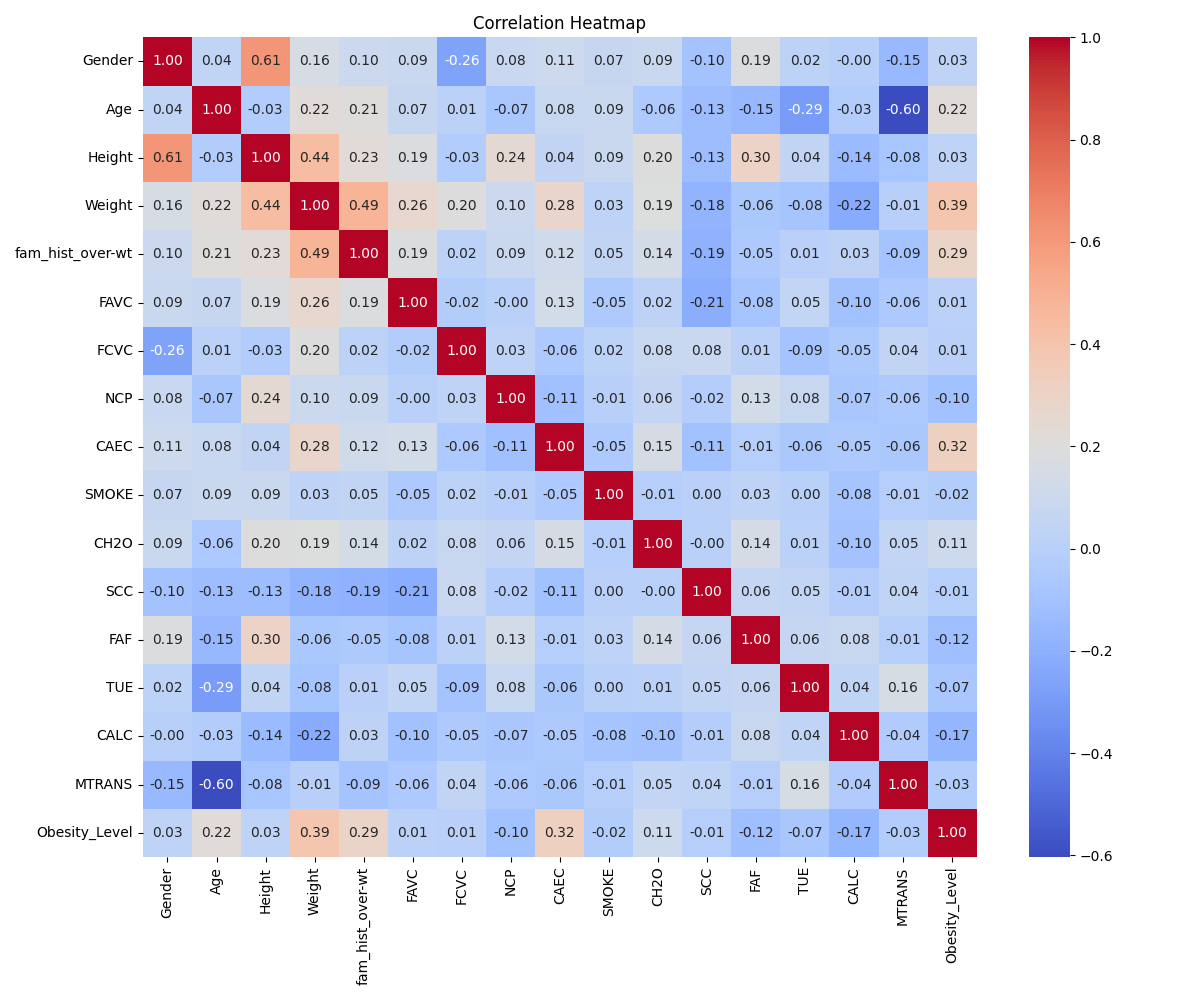
The ultimate goal of this EDA is to support the pharmaceutical company in refining the "miracle drug" treatment plan, guiding patient targeting and enabling personalized interventions for improved outcomes.

### 

### 2.1 Analysis Activities

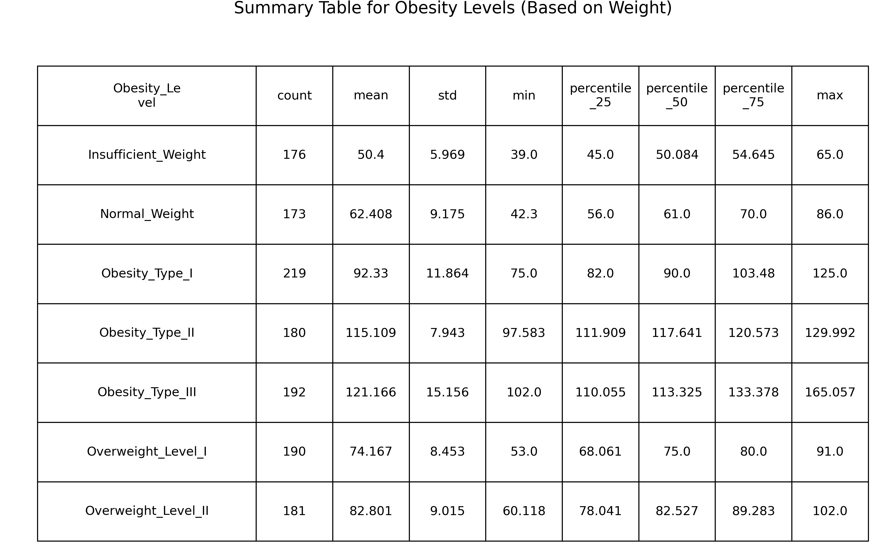
#### 2.1.1 Statistical Summary and Distribution Analysis

**Weight as a Baseline**



**Figure 2: Correlation Heatmap with weight as baseline**

Weight serves as the most relevant baseline feature for understanding the distribution of obesity levels, given its high correlation with the obesity categories (as seen in the correlation heatmap). While weight is a strong indicator of obesity level trends, the deviations in weight within each category and the overlap between categories (as observed from the summary statistics) highlight the complexity of categorizing individuals purely by weight. Below is a table giving a detailed breakdown for each weight category.

****

**Figure 3 : Obesity levels based on weight**

##### 2.1.1.1 Mean (Average)

The mean values across categories align with the progression of obesity levels:

* **Insufficient Weight** has the lowest mean weight of 50.4, while **Obesity Type III** has the highest mean of 121.17.
* **Overweight Level II** (mean = 82.80) bridges the gap between normal weight and the lower obesity levels.
* **Insight**: The steady increase in mean weight supports weight's relevance in predicting obesity levels. However, deviations and overlaps (as discussed below) challenge its ability to fully discriminate between categories.

##### 2.1.1.3 Minimum and Maximum

The weight ranges show notable overlaps between categories:

* **Normal Weight** spans from 42.3 to 86.0, overlapping significantly with **Overweight Level I** (53.0 to 91.0).
* **Obesity Type II** and **Obesity Type III** also show overlap, with ranges of 97.58–129.99 and 102.0–165.06, respectively.

**Insight**: The overlapping ranges imply that weight alone may lead to misclassifications, particularly for individuals near category boundaries.

##### 2.1.1.4 Standard Deviation (Measure of Spread)

The standard deviation highlights high variability of the weights, given the example here:

* **Obesity Type III** shows the highest variability (std = 15.16), indicating diverse weight ranges within this group.
* **Overweight Level I** and **Normal Weight** have lower standard deviations (8.45 and 9.18, respectively), suggesting relatively tighter clustering around the mean, however the variation is still high

**Insight**: Large deviations (especially reflecting in the overlapping minimum and maximum, as previously discussed) could be due to various factors and other features that define the weight category of a person.

##### 2.1.1.5 Potential Hypotheses Based on This Analysis

As described in the analysis, there are high variances and overlapping of values - on weight across the obesity categories. As such, weight is not a reliable feature to be used alone despite its high correlation, since the overlapping of values could potentially lead to misclassifications. The following section shall explore other features that could potentially be engineered to combine with weights and give a better prediction outcome.

#### 2.1.2 Correlation Analysis

The following hypotheses explore the relationships between specific patient attributes and their potential impact on weight loss outcomes during the clinical trial of the "miracle drug." These hypotheses will be tested using statistical methods to determine their validity.

##### 2.1.2.1 Relationship between Height, Weight and Obesity Levels



**Figure 4 : Scatter Plot of Obesity Levels by Height and Weight**

**Hypothesis**: Height and weight are correlated with obesity levels, where individuals with higher weights and certain height ranges are more likely to fall into more severe obesity categories.

**Approach**:

**Scatter Plot Analysis**: The scatter plot visualizes the distribution of height and weight across different obesity levels. Obesity levels are categorized and color-coded, providing an intuitive way to observe clustering patterns.

**Correlation Review**: By referencing the correlation heatmap, the relationship between height, weight, and obesity levels is further quantified, with weight showing a stronger correlation with obesity levels than height.

**Trend Observation**: The progression from **Insufficient Weight** (clustered at lower weights and heights) to **Obesity Type III** (spread across higher weights and heights) is examined to identify any overlapping regions.

**Results**:

**Correlation and Clustering**:

* A positive trend is evident, where higher weights generally align with more severe obesity levels.
* For lower obesity levels (e.g., **Normal Weight**, **Overweight Levels I and II**), height and weight are more tightly clustered.
* For higher obesity levels (e.g., **Obesity Type III**), there is greater dispersion in height and weight, indicating a wider range of body types within these categories.

**Overlaps Across Categories**:

* There are visible overlaps, particularly between **Overweight Level II** and **Obesity Type I**, as well as between **Obesity Types II and III**, indicating that height and weight alone may not provide a perfect classification.
* However, the general trend shows that individuals in **Insufficient Weight** and **Obesity Type III** are distinctly separated from other categories, reflecting more extreme cases.

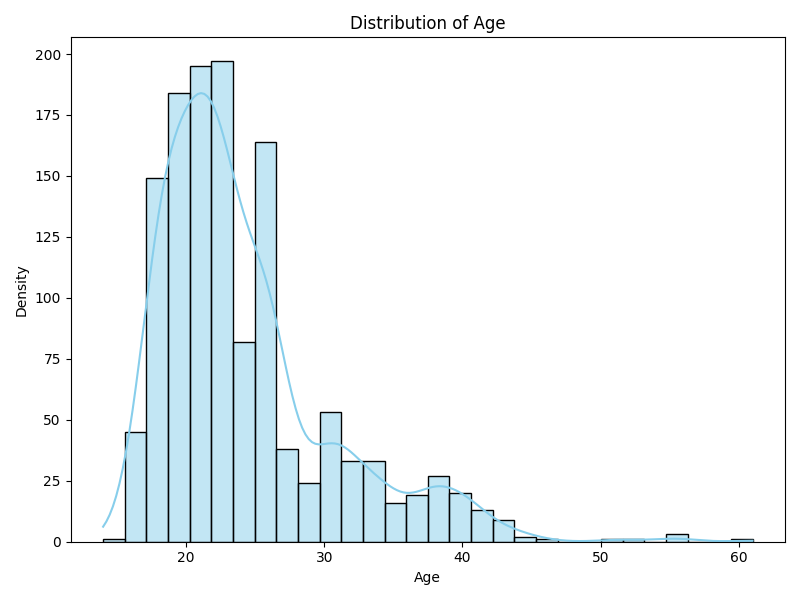
**Key Insight**:

* While weight demonstrates a strong correlation with obesity levels, height acts as a supporting feature that helps refine classification, particularly for overlapping categories. Patients in similar weight ranges but with varying heights may fall into different obesity levels, suggesting a potential normalization effect by height (e.g., BMI).

##### 2.1.2.2 Relationship between Age, Weight and Obesity Levels



**Figure 5 : Scatter Plot of Obesity Levels by Age and Weight**



**Figure 6 : Distribution of Age**

**Hypothesis:** Age and weight do not show a clear correlation with obesity levels, as significant overlaps in data points are observed between categories, making it difficult to distinctly differentiate obesity levels based on these two features alone.

**Approach:**

**Scatter Plot Analysis**: The scatter plot visualizes the distribution of age and weight across obesity levels. The clusters exhibit considerable overlap, which limits the ability to draw meaningful boundaries between obesity levels.

**Age Distribution Histogram:** A supporting histogram of age distribution reveals that the majority of data points are concentrated in the younger age groups (primarily between 18 and 30 years). This skewed distribution limits the ability of age to serve as a strong feature for obesity classification, as older individuals are underrepresented.

**Results:**

**Correlation and Overlap:**

* Unlike height and weight, the relationship between age and weight does not reveal distinct clusters for obesity levels.
* Overlaps are especially prominent for mid-range obesity levels such as Overweight Level I, Obesity Type I, and Normal Weight, making it difficult to differentiate them based solely on age and weight.

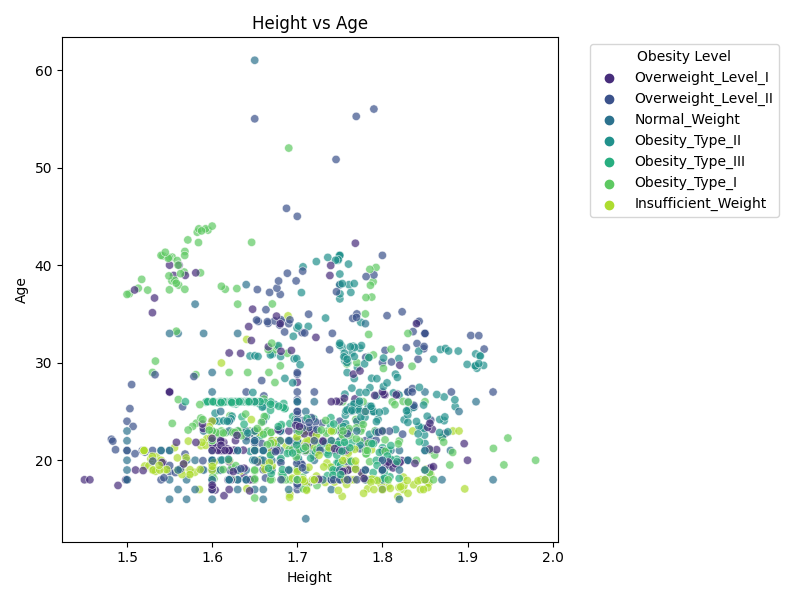
**Distribution Across Ages**:

* Most individuals are concentrated in younger age groups (20–30 years), regardless of obesity level, indicating age’s limited impact on categorization.
* As age increases beyond 40, the number of individuals decreases, but those who remain span across a wide weight range, from **Normal Weight** to **Obesity Type III**.

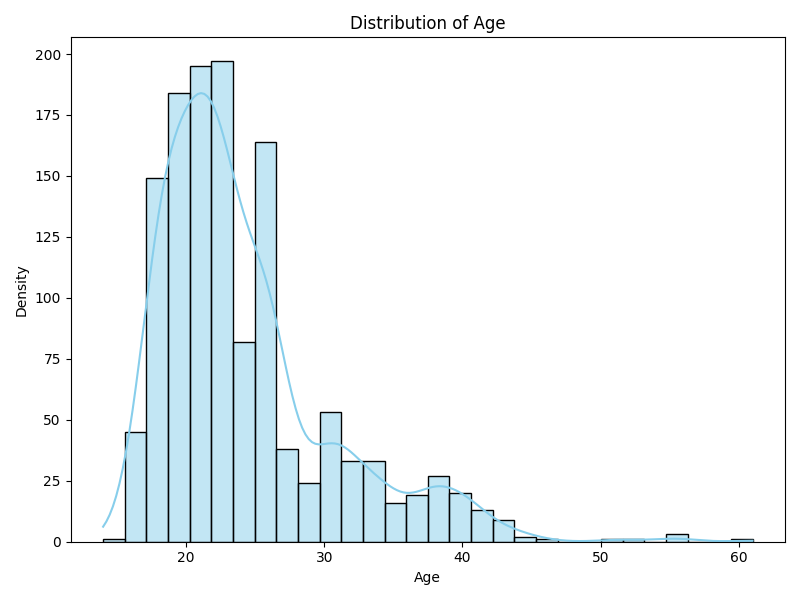
**Key Insight**

While weight remains a critical factor in predicting obesity levels, age does not provide significant additional value due to the lack of clear trends or separability across categories. The overlaps between categories suggest that age is not a strong standalone feature and may not even complement weight or other lifestyle features for more nuanced predictions.

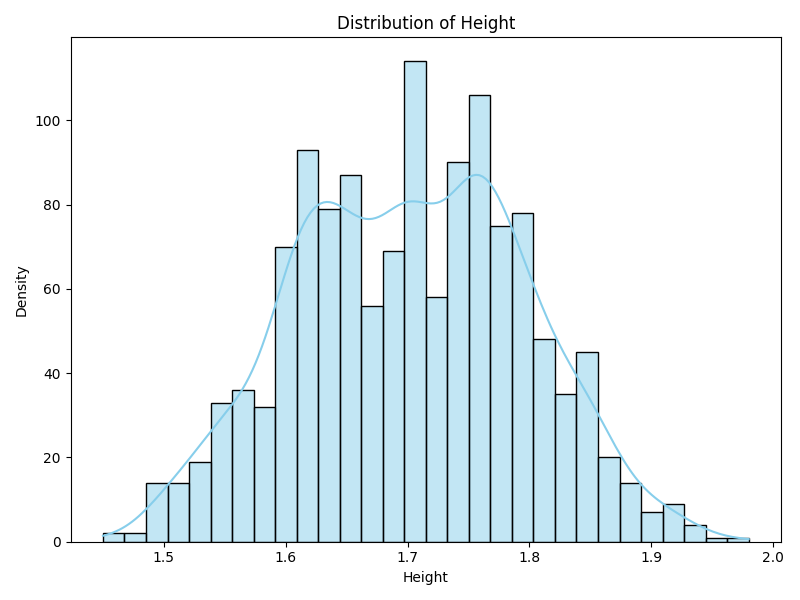
##### 2.1.2.3 Relationship between Age, Height and Obesity Levels



**Figure 7 : Scatter Plot of Obesity Levels by Age and Height**



**Figure 8 : Distribution of Age**



**Figure 9 : Distribution of Height**

**Hypothesis:** Height and age together do not provide meaningful separation for obesity level categorization due to substantial overlap across data points. Additionally, the distribution of height, though symmetric, does not vary significantly across obesity levels, further limiting its discriminatory power.

**Approach:**

**Scatter Plot Analysis**: The scatter plot visualizes the distribution of age and weight across obesity levels. The clusters exhibit considerable overlap, which limits the ability to draw meaningful boundaries between obesity levels.

**Age Distribution Histogram:** A supporting histogram of age distribution reveals that the majority of data points are concentrated in the younger age groups (primarily between 18 and 30 years). This skewed distribution limits the ability of age to serve as a strong feature for obesity classification.

**Height Distribution Histogram**:

* A supplementary histogram of height distribution reveals a symmetric, bell-shaped curve with most individuals clustered around the 1.7-meter range.
* This limited spread in height reduces its effectiveness as a distinguishing factor for obesity categorization.

**Results:**

**Overlap Across Categories**:

* The scatter plot demonstrates heavy clustering and overlap of obesity levels in the most common height range (1.6 -- 1.8 meters) and younger age group (20–30 years). This overlap makes it challenging to separate obesity levels effectively using these two features.
* Even individuals in the higher obesity levels, such as **Obesity Type III**, span a wide age range (20–60 years) but are concentrated within the mid-height range.

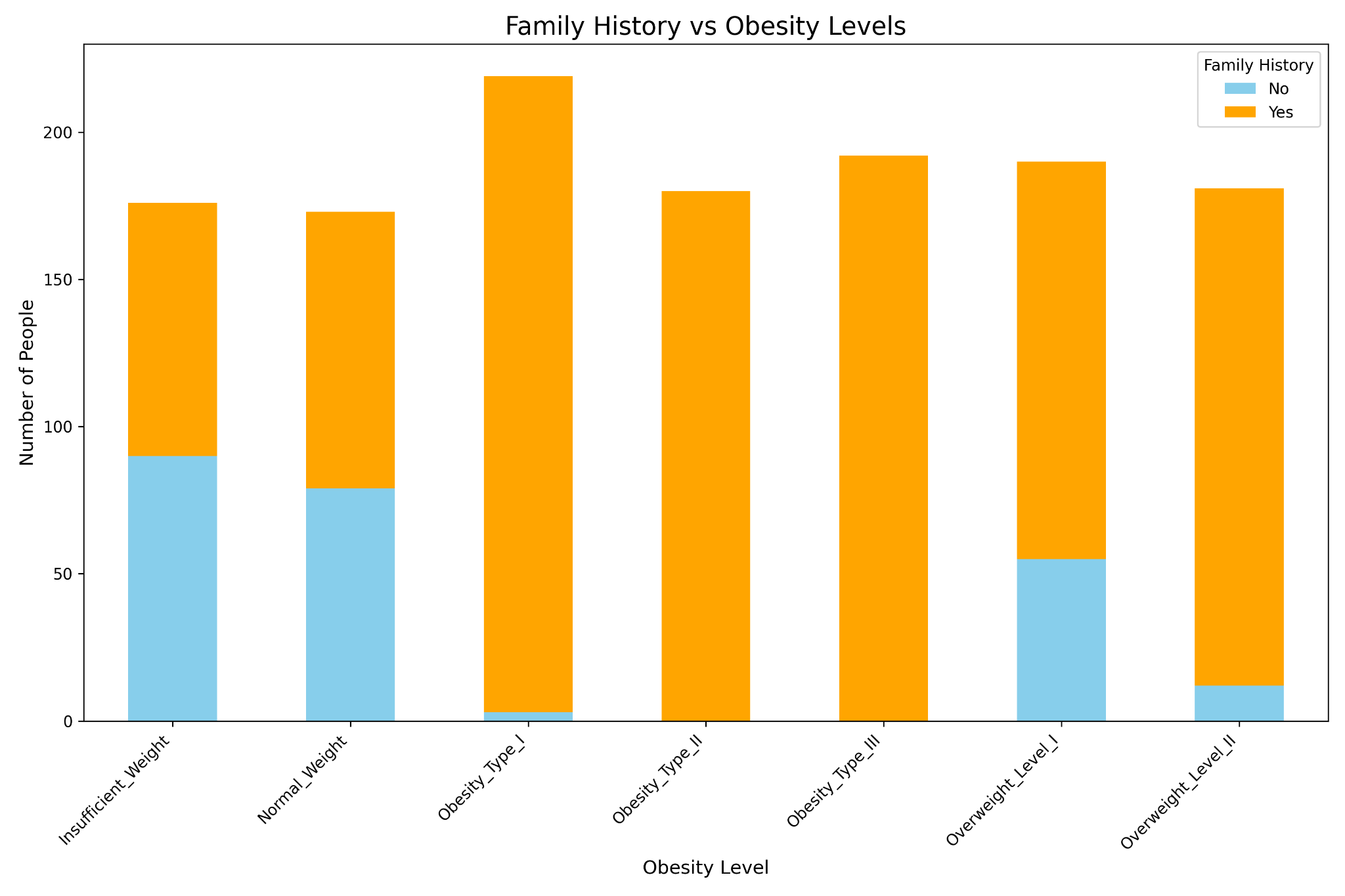
**Limited Height Variation**:

* The height distribution histogram shows that the dataset predominantly includes individuals with heights between 1.6 and 1.8 meters.
* The symmetric nature of the distribution further indicates that height is unlikely to provide significant discriminatory power on its own or in combination with age.

**Key Insight from Distribution**:

* The relatively narrow range and symmetric distribution of height, combined with the dominance of younger individuals in the age distribution, highlight the limitations of these two features for obesity level categorization.

The following hypothesis explores beyond just weight, age and height - as it explores other features that may give more discriminatory power and boost the model’s performance in prediction outcomes.



**Figure 10: Obesity Level by Family History**

**Hypothesis**: A positive family history of being overweight significantly correlates with higher obesity levels, indicating that genetic or shared lifestyle factors may influence an individual’s likelihood of developing obesity.

**Approach**:

**Stacked Bar Chart Analysis**:

* The provided chart visualizes the distribution of family history (yes/no) across obesity levels, highlighting how the presence of a family history correlates with increasing obesity severity.
* The bar chart is broken down into two categories: individuals with a positive family history of being overweight (orange) and those without (blue).
* By comparing the proportions of family history within each obesity level, we observe how strongly this factor is associated with obesity categories.
* The analysis explores whether a family history acts as a significant indicator for severe obesity (e.g., Obesity Types II and III).

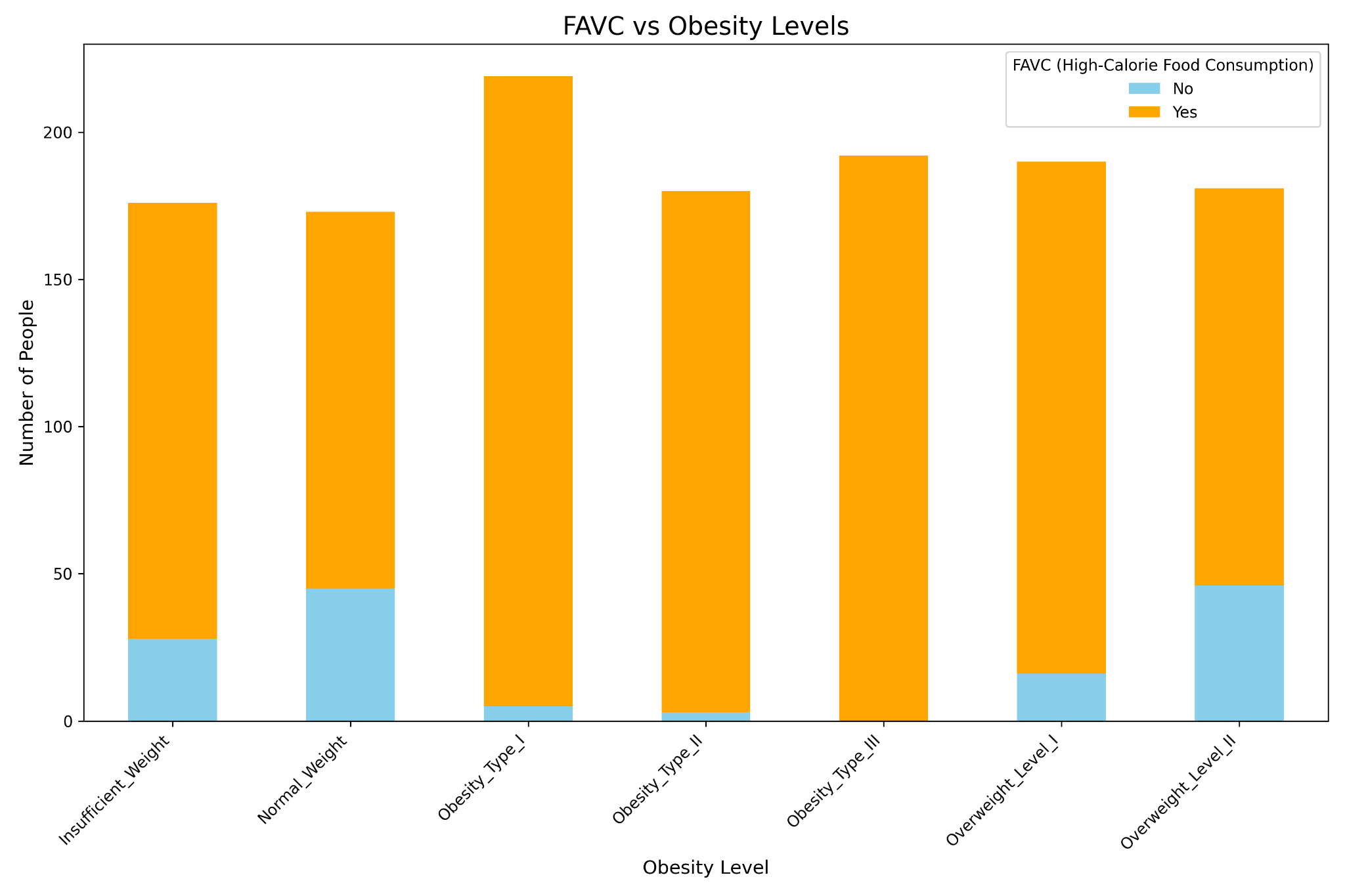
**Results**:

* In lower obesity levels (Insufficient Weight, Normal Weight), there is a significant proportion of family history without being overweight.
* As the obesity levels increase - (overweight level 1 and 2, obesity levels 1,2,3), the proportion of family history without being overweight decreases.

**Key Insight**

While family history can complement other features like weight, lifestyle habits, or dietary patterns, it is insufficient as a predictor for obesity. Its overlap across obesity and non-obesity levels suggests that other factors likely mediate its influence, such as lifestyle or environment.

##### 2.1.2.4 Frequency of High-Calorie Food Consumption (FAVC)



**Figure 11: Obesity Level by FAVC**

**Hypothesis**: Frequent consumption of high-calorie foods is strongly associated with higher obesity levels, as seen in the significant proportion of individuals with positive FAVC responses across these categories.

**Approach**:

**Stacked Bar Chart Analysis**:

* The chart illustrates the distribution of high-calorie food consumption (FAVC: Yes/No) across obesity levels.
* Individuals are categorized based on their obesity levels, with the proportion of those consuming high-calorie foods (orange) versus those who do not (blue) highlighted.

**Behavioral Trend Observation**:

* Observing whether the prevalence of high-calorie food consumption increases with obesity severity.
* Highlighting the outliers (e.g., individuals with low obesity levels but high FAVC).

**Results**:

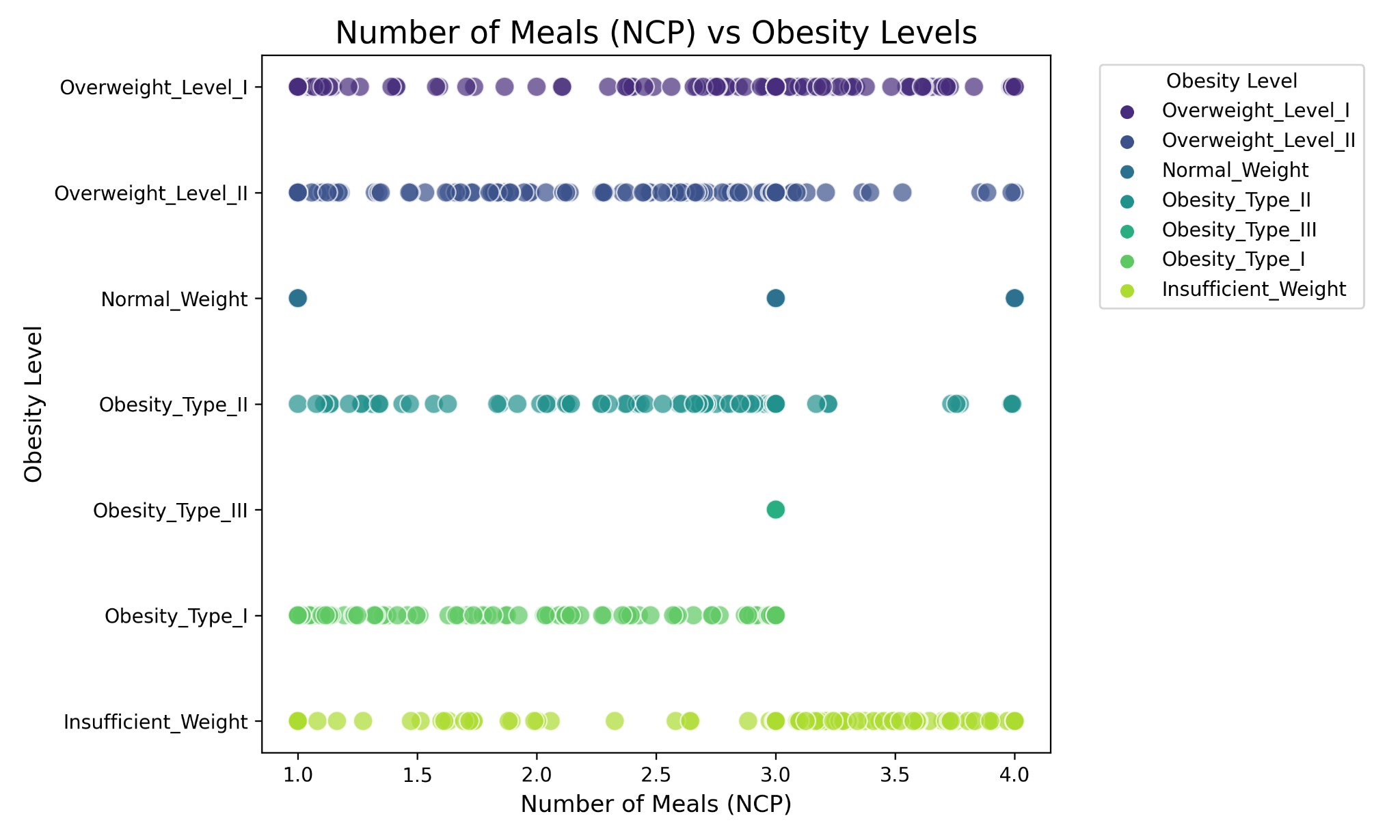
**Presence in All Categories**:

* Even in **Insufficient Weight** and **Normal Weight**, a significant number of individuals report high-calorie food consumption (FAVC: Yes). This implies that high-calorie food consumption alone does not determine obesity levels.
* For severe obesity levels (**Obesity Type II and III**), FAVC dominates, reinforcing its role in exacerbating weight gain.

**Limitations as a Predictor**:

* The feature lacks granularity, as it does not account for how frequently individuals consume high-calorie foods or the portion sizes. For instance, someone who consumes high-calorie foods daily but in small portions might exhibit different outcomes compared to someone consuming them in larger quantities per serving.
* **Frequency Per Meal/Serving**: Without this information, it is challenging to discern the true dietary habits that influence obesity levels.
* **Quality of Other Meals**: The dataset might benefit from capturing the nutritional quality of other meals, which could help contextualize FAVC as part of an individual’s overall diet.

##### 2.1.2.5 Number of Main Meals per Day (NCP)



**Figure 12: Obesity Level by NCP**

**Hypothesis:** Patients who consume more meals per day are more likely to be obese.

**Approach:**

**Scatter Plot Analysis**:

* The scatter plot visualizes the number of meals consumed per day (NCP) across various obesity levels.
* By comparing obesity categories against NCP values, the analysis focuses on whether higher meal frequency corresponds to increased obesity severity.

**Results:**

**Visualization Does Not Support the Hypothesis**:

* Contrary to the hypothesis, individuals in the **Underweight** and **Normal Weight** categories also report consuming high numbers of meals per day (NCP = 3–4), which challenges the idea that higher meal frequency directly leads to obesity.
* This overlap across categories indicates that the number of meals alone does not sufficiently predict obesity levels.

**Contradictions in Severe Obesity**:

* Individuals in **Obesity Type III**, the most severe category, consistently report consuming exactly 3 meals per day. This structured meal pattern highlights that factors beyond NCP—such as meal composition or caloric density—play a critical role in obesity development.
* Despite scheduled eating habits, individuals in this category remain obese, suggesting a deeper connection to caloric intake, metabolic rates, or lifestyle habits.

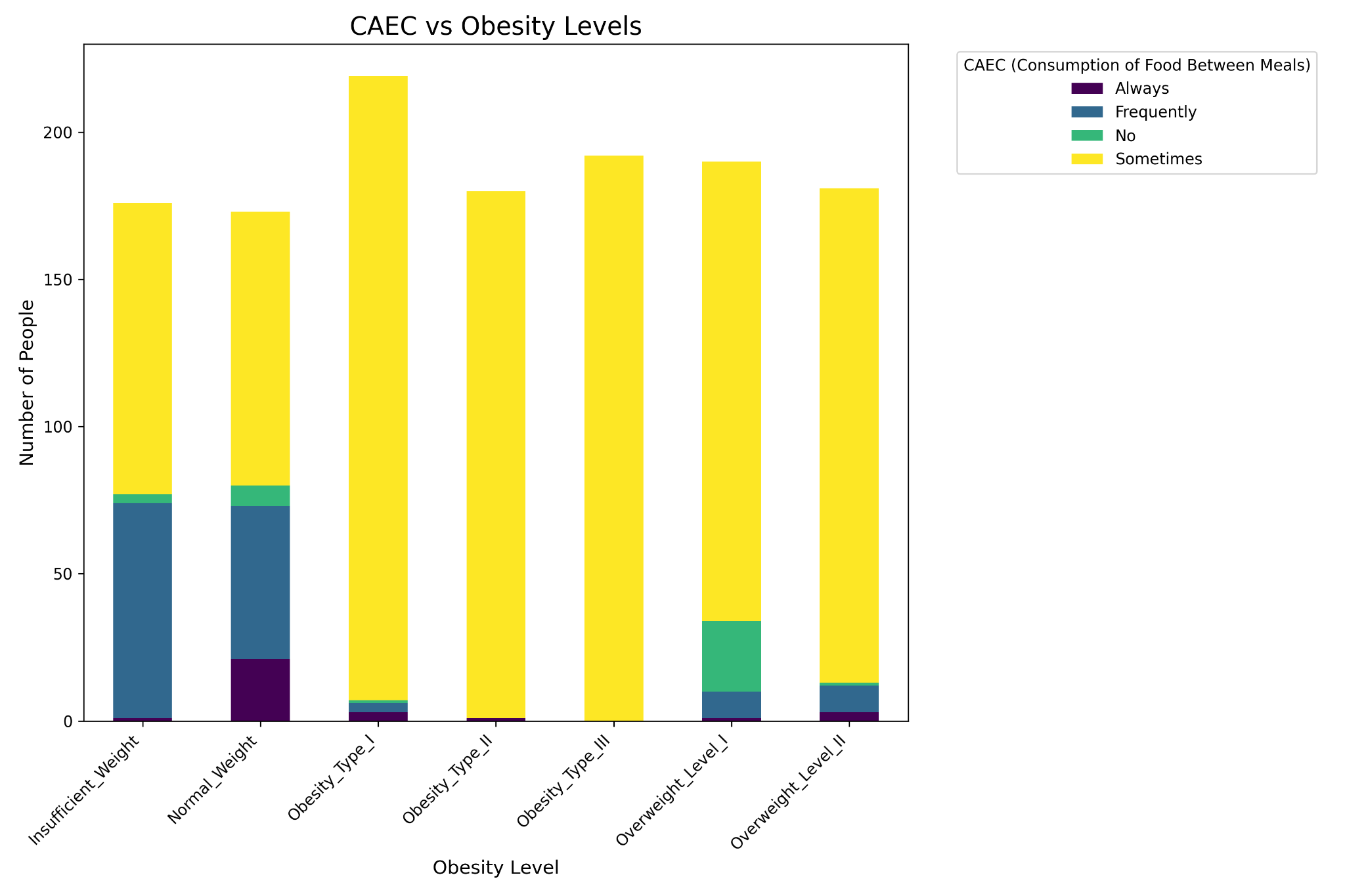
**Feature Limitations**:

* NCP, as a standalone feature, lacks the granularity needed to explain obesity levels. For example, consuming 3 high-calorie meals versus 3 low-calorie meals has vastly different implications for weight gain.
* Additionally, physical activity levels, meal timing, and portion sizes likely moderate the relationship between meal frequency and obesity.

**Key Insight**

While NCP provides some insight into eating habits, it does not account for the complexities of obesity, such as caloric density, activity levels, and meal timing. The case of **Obesity Type III**, where individuals consume exactly 3 meals per day, underscores the importance of exploring complementary features to gain a holistic understanding of obesity predictors.

##### 2.1.2.6 Additional Consumption of Food in between meals (CAEC)

****

**Figure 13: Obesity Level vs CAEC**

**Hypothesis:** Patients who self-monitor their calorie consumption (SCC = Yes) achieve greater weight loss than those who do not.

**Approach:**

**Correlation with Obesity:**

* The proportion of individuals who consume additional food "Sometimes" increases with obesity levels, suggesting a general trend where additional food consumption between meals correlates with higher obesity categories.

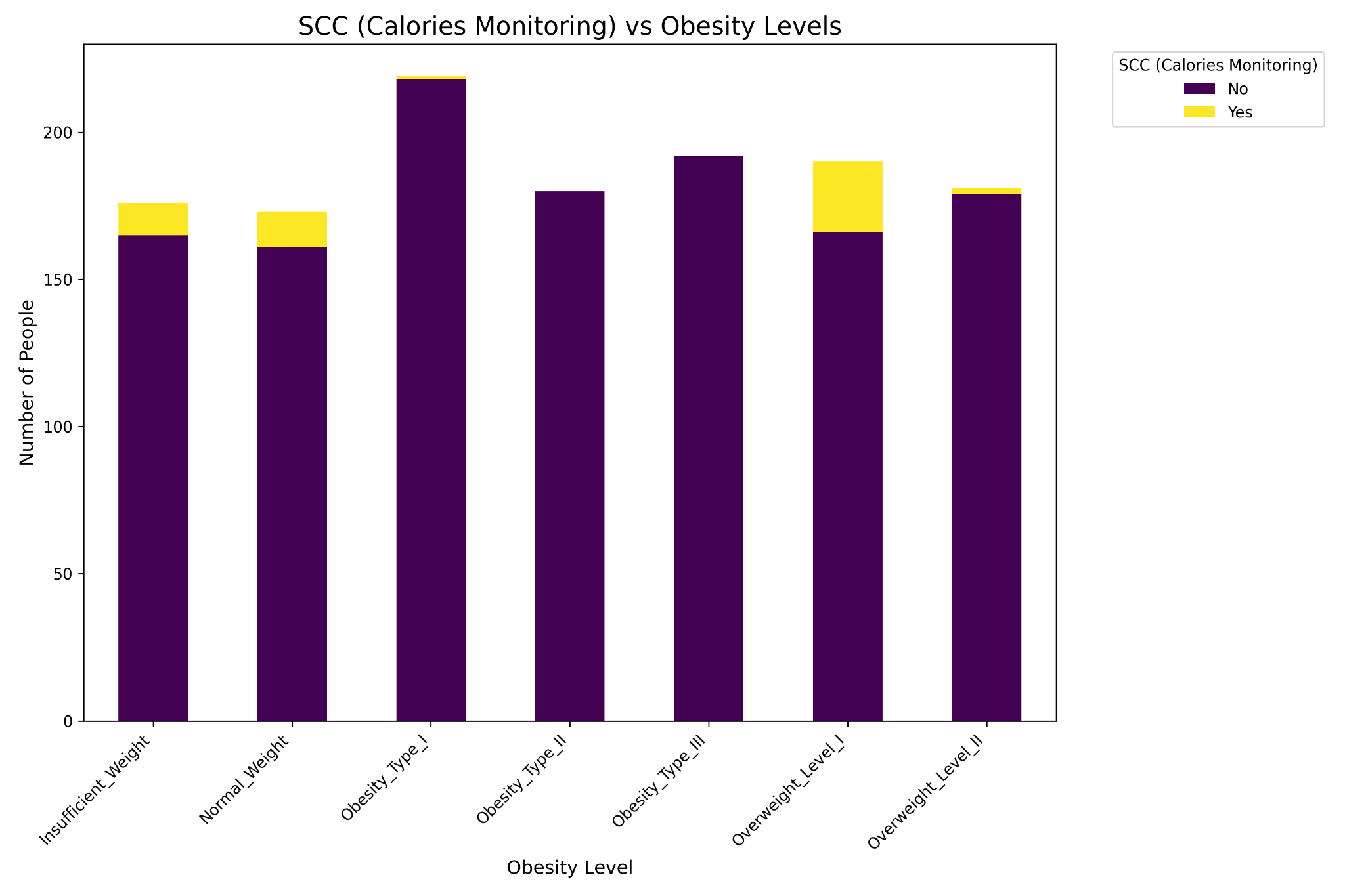
**Exceptions to the Trend**:

* **Normal Weight** individuals show the highest number of people in the "Always" and "Frequently" categories for additional food consumption. This challenges the idea that higher CAEC directly correlates with obesity levels.
* **Overweight Level I** includes individuals who do not consume additional food ("No"), despite being categorized as overweight, highlighting other factors (e.g., calorie intake, meal composition) influencing obesity.

**Results:**

The CAEC feature indicates a loose correlation with obesity levels but is not robust enough to serve as a strong independent predictor. Its value lies in complementing other dietary or lifestyle features, such as caloric awareness, physical activity, number of meals, and frequency of high caloric meal for a more comprehensive model.

##### 2.1.2.7 Self Monitoring of Calorie Consumption (SCC)



**Figure 14: Obesity Level vs SCC**

**Hypothesis:** Awareness and monitoring of calorie intake (SCC) correlates with healthier weight categories, where individuals who monitor their calorie intake are less likely to fall into higher obesity levels.

**Approach:**

**Stacked Bar Chart Analysis:**

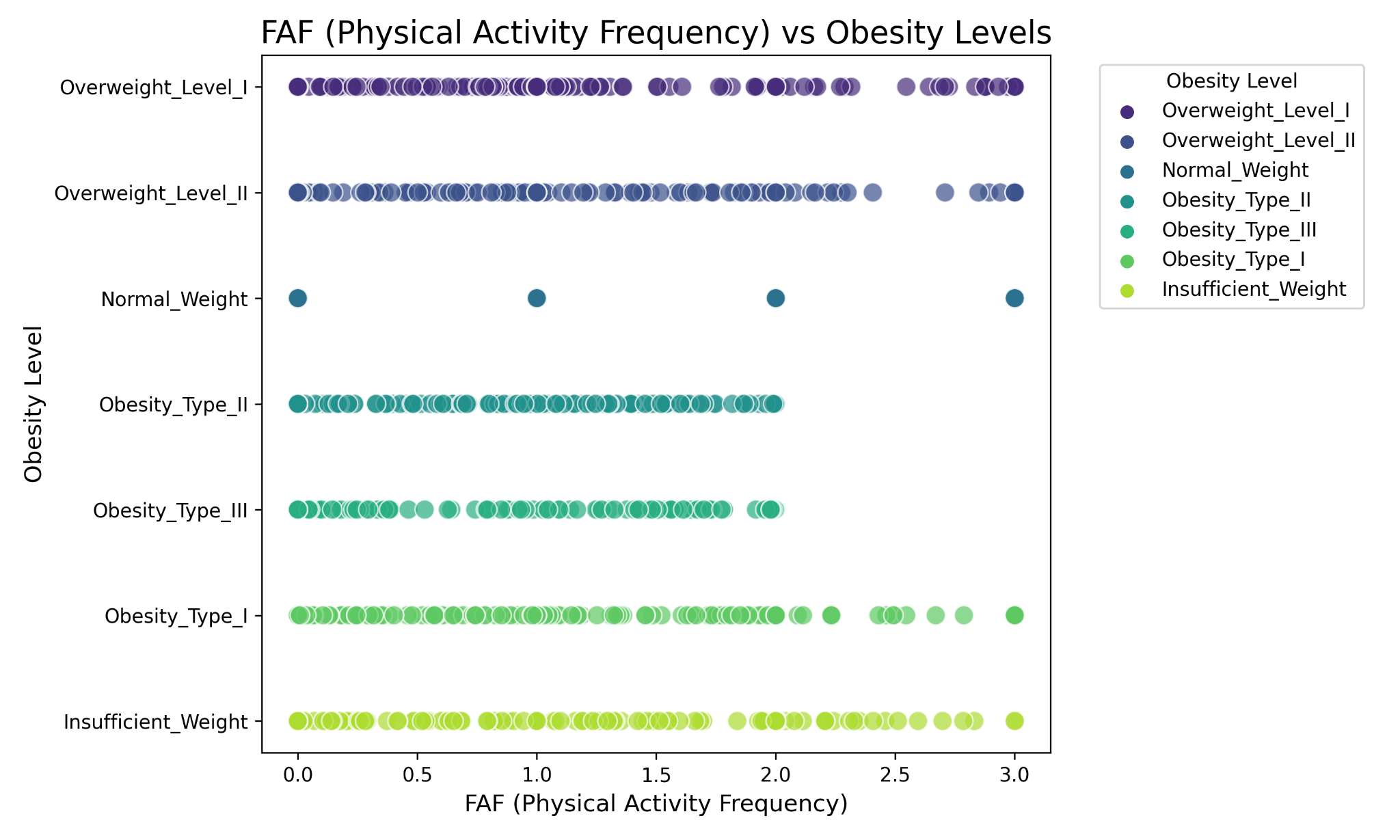
* The chart represents the distribution of individuals who monitor their calorie intake (SCC: Yes) versus those who do not (SCC: No) across various obesity levels.
* A noticeable lack of calorie monitoring among most individuals across all categories is observed, with a small proportion of those aware of their calorie intake found in healthier weight categories.

**Results:**

**Proportion of Aware Individuals:**

* **Underweight, Normal Weight, and Overweight Level I** categories show a higher proportion of individuals monitoring their calorie intake. This aligns with the expectation that calorie awareness may play a role in maintaining healthier weight categories.
* However, even within these groups, the majority of individuals are unaware of their calorie consumption, indicating that SCC alone is not sufficient to explain obesity levels.
* SCC alone does not provide enough granularity to predict obesity levels effectively. For instance:
  + Individuals who monitor their calories may still fall into higher obesity levels due to other factors like insufficient physical activity or high caloric density of meals.
  + Lack of SCC data for a majority of individuals makes it hard to draw precise conclusions.
* SCC could work as part of a broader model, particularly when paired with features like:
  + **Physical Activity (FAF)** to assess energy expenditure.
  + **Number of Meals (NCP)** to understand eating patterns.
  + **High-Calorie Food Frequency (FAVC)** and **Additional Food Consumption (CAEC)** to evaluate caloric intake quality and quantity.

##### 2.1.2.8 Physical Activity Frequency (FAF)



**Figure 15 : Obesity Level by FAF**

**Hypothesis:** Lower physical activity frequency (FAF) is associated with higher obesity levels, while higher FAF corresponds to lower obesity levels.

**Approach:**

**Scatter Plot Analysis**:

* The scatter plot illustrates the distribution of physical activity frequency across detailed obesity levels.
* Observing the concentration of higher obesity levels (Obesity Types I, II, III) in lower FAF ranges (0 to 2) and contrasting this with the broader spread in lower obesity levels.

**Results:**

**Concentration in Higher Obesity Levels**:

* Individuals in higher obesity categories (**Obesity Types I, II, III**) are primarily concentrated in the **0 to 2 range** of FAF, suggesting a lack of frequent physical activity in these groups.

**Variability in Lower Obesity Levels**:

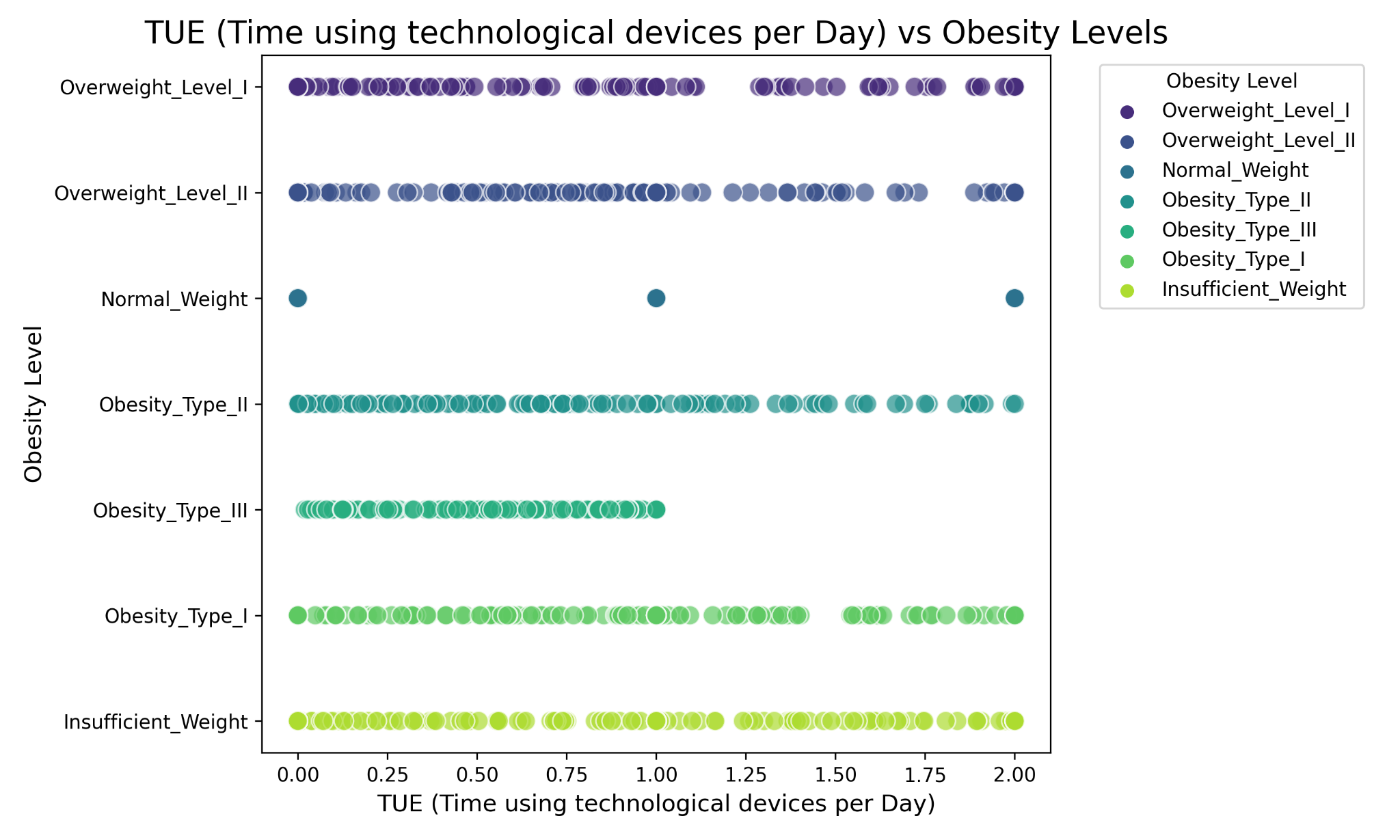
* Normal weight and insufficient weight categories show a wider spread of FAF values, ranging from **0 to 3**.
* This inconsistency indicates that individuals in these categories engage in a variety of physical activity levels, potentially offset by other factors such as number of meals, high caloric consumption per meal, lower TUE, etc.

**Feature Limitations**:

* FAF alone does not provide a complete picture of physical activity's role in obesity. For example:
  + A low FAF might be paired with higher sedentary activity (e.g., technology use), further contributing to obesity.

##### 

##### 2.1.2.9 Technology Use (TUE)



**Figure 16: Obesity Level by TUE**

**Hypothesis**: Higher time spent using technological devices (TUE) correlates with higher obesity levels due to sedentary behavior.

**Approach**:The scatter plot visualizes the distribution of TUE across various obesity levels, with a focus on whether individuals in higher obesity categories spend more time using technological devices.

**Results**:

**Obesity Categories Show Spread Across TUE Values**:

* Individuals in **Obesity Types I, II, III** are more spread out in TUE values, suggesting a range of daily technological device usage, with many spending significant amounts of time on devices.

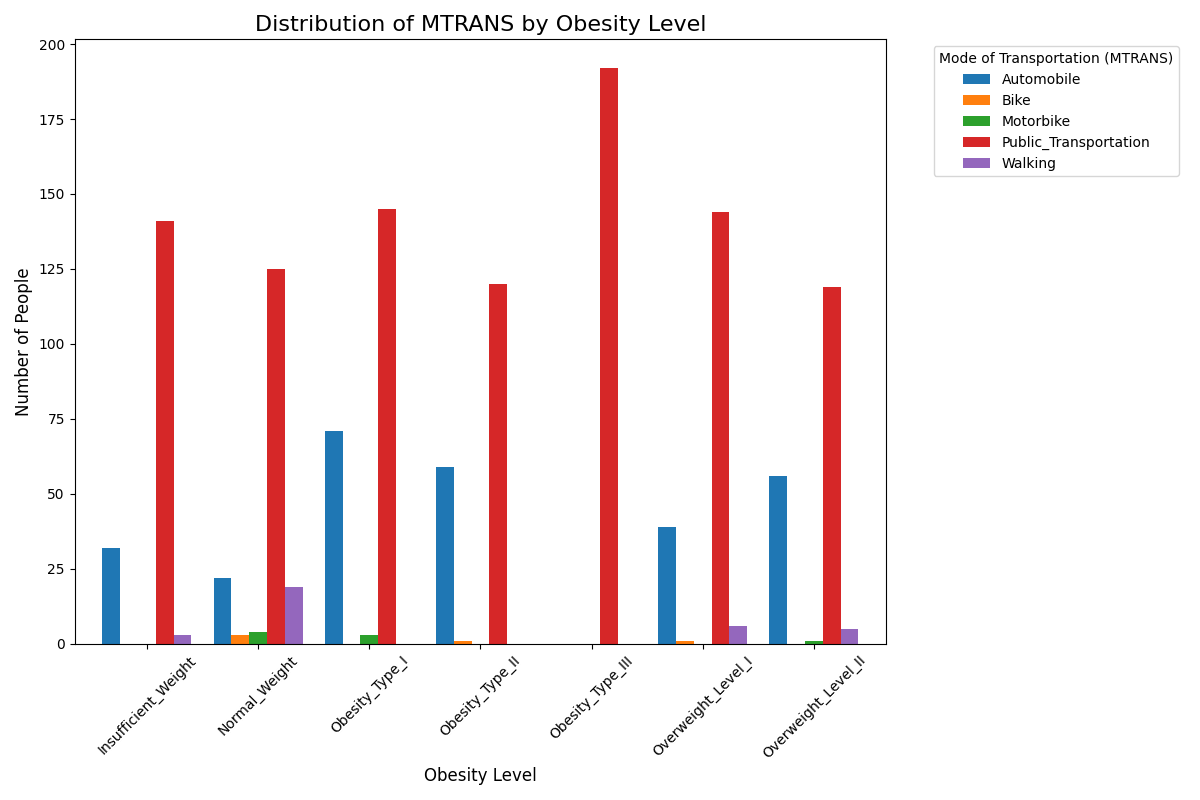
**Inconsistencies in Lower Obesity Levels**:

* Normal weight and underweight categories also show a wide spread of TUE values, with individuals spending significant time on technological devices, contrary to the hypothesis.
* This inconsistency highlights that TUE alone does not reliably differentiate between obesity levels.

**Feature Limitations**:

* TUE does not account for additional factors such as caloric intake, physical activity, or occupational requirements (e.g., jobs requiring long hours of screen time).

##### 2.1.2.10 Mode of Transportation (MTRANS)



**Figure 17: Obesity Level by MTRANS**

**Hypothesis**: Individuals using more active transportation modes (e.g., walking) are more likely to maintain lower obesity levels.

**Approach**:

**Stacked Bar Chart Analysis**:

* The chart compares the distribution of transportation modes (e.g., walking, public transportation, automobile, etc.) across obesity levels.
* Specific attention is paid to the proportion of individuals using **walking** as their primary mode of transportation, as it represents a physically active mode.

**Results**:

**Walking and Lower Obesity Levels**:

* The proportion of individuals walking as a mode of transportation is higher in **Normal Weight** categories and drops significantly in higher obesity categories like **Obesity Type III**, where walking is nearly absent.
* This aligns with the hypothesis that active transportation contributes to healthier weight categories.

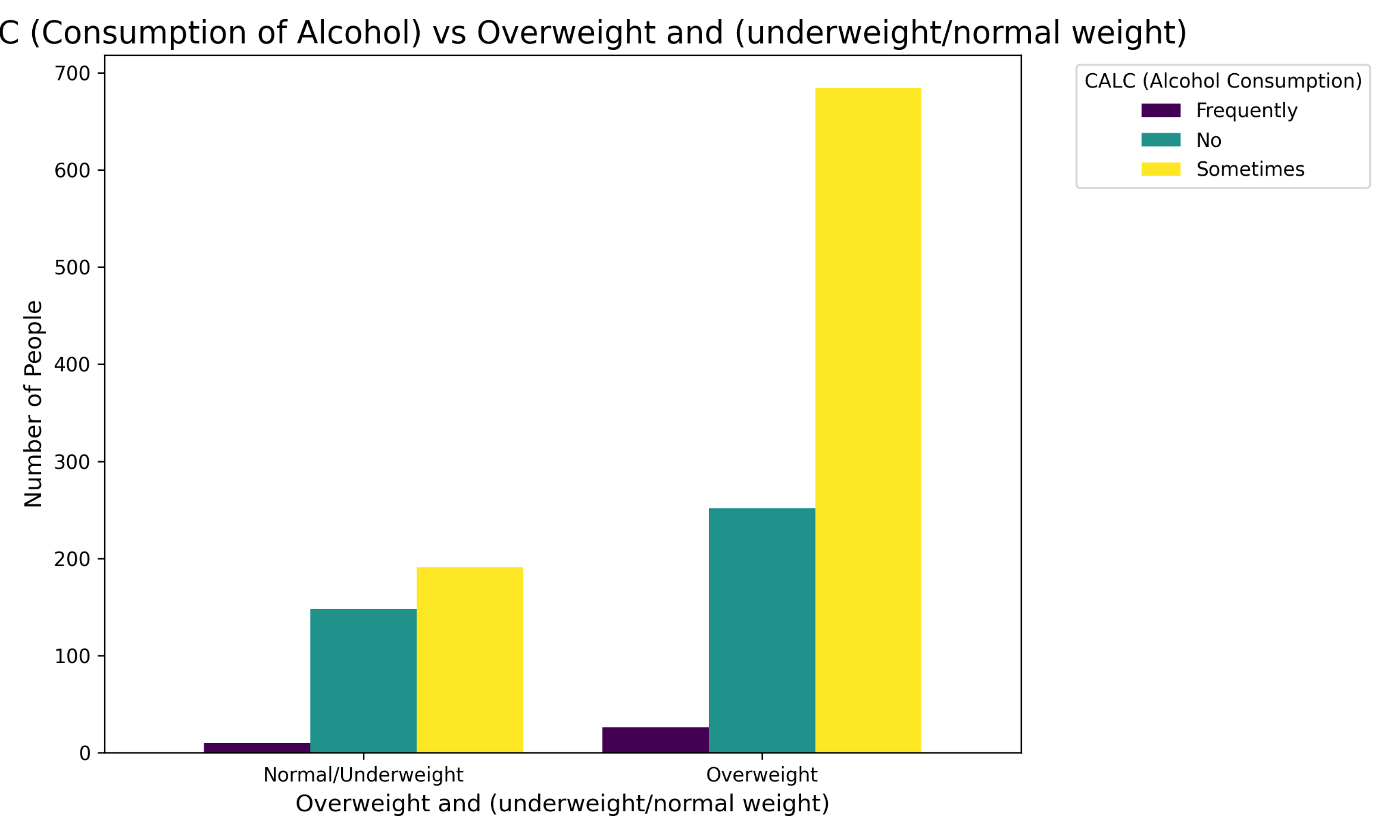
**Inconsistencies in Underweight**:

* Surprisingly, underweight individuals have fewer people walking compared to those in the normal weight category. This may be influenced by other factors, such as dietary habits, physical activity frequency, or lifestyle differences, rather than transportation alone.

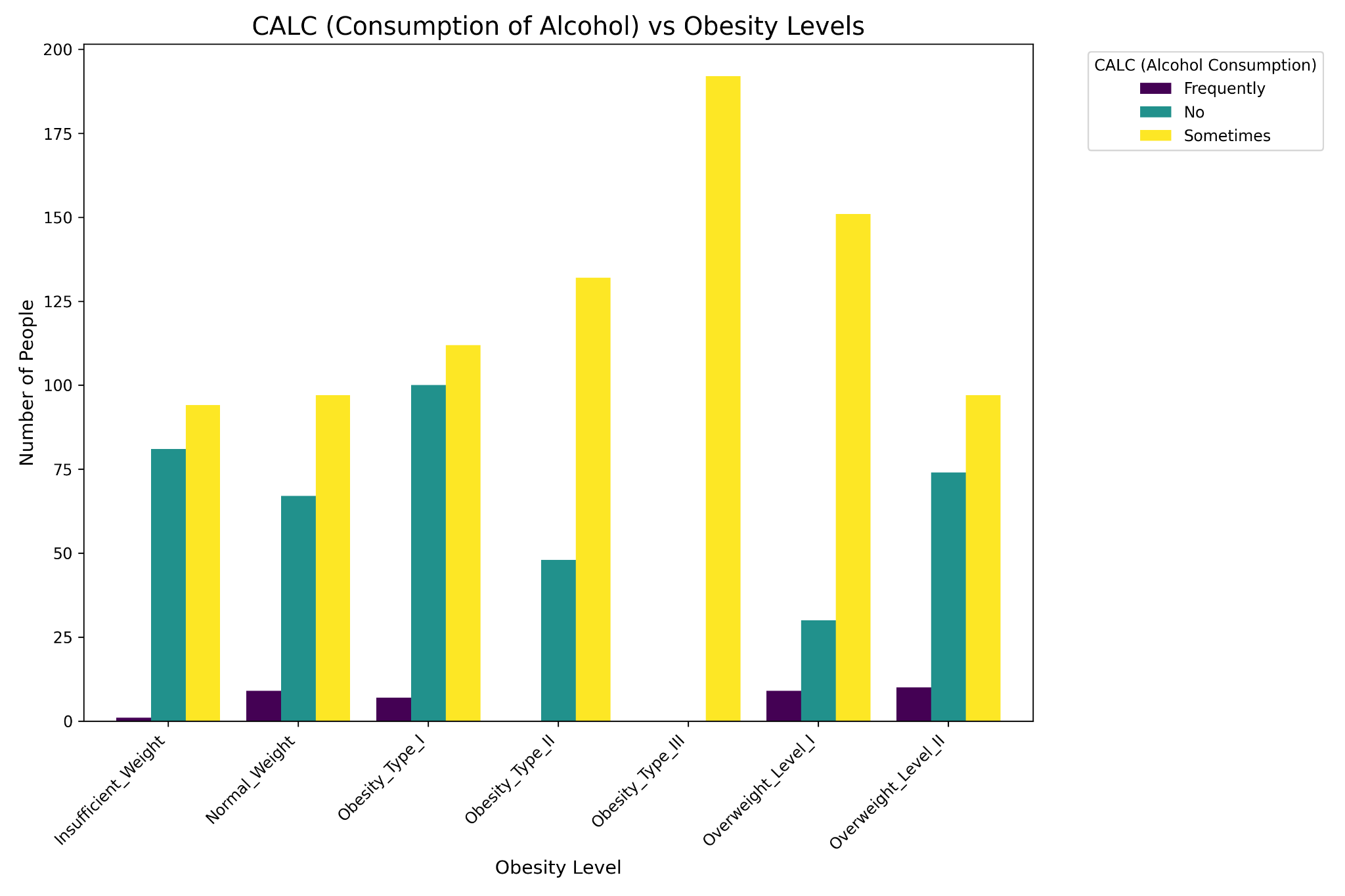
**Feature Limitations**:

* The loose correlation between walking and normal weight suggests MTRANS could be informative when combined with other features, such as **physical activity frequency (FAF)** and **Technology use (TUE),** which could represent the sedentary time.

##### 2.1.2.11 Frequency of alcohol consumption(CALC)



**Figure 18: Obesity Level by CALC**



**Figure 19: Obesity Level by CALC**

**Hypothesis**: Higher alcohol consumption correlates with increased obesity levels, especially in generalized overweight categories.

**Approach**:

**Results**:

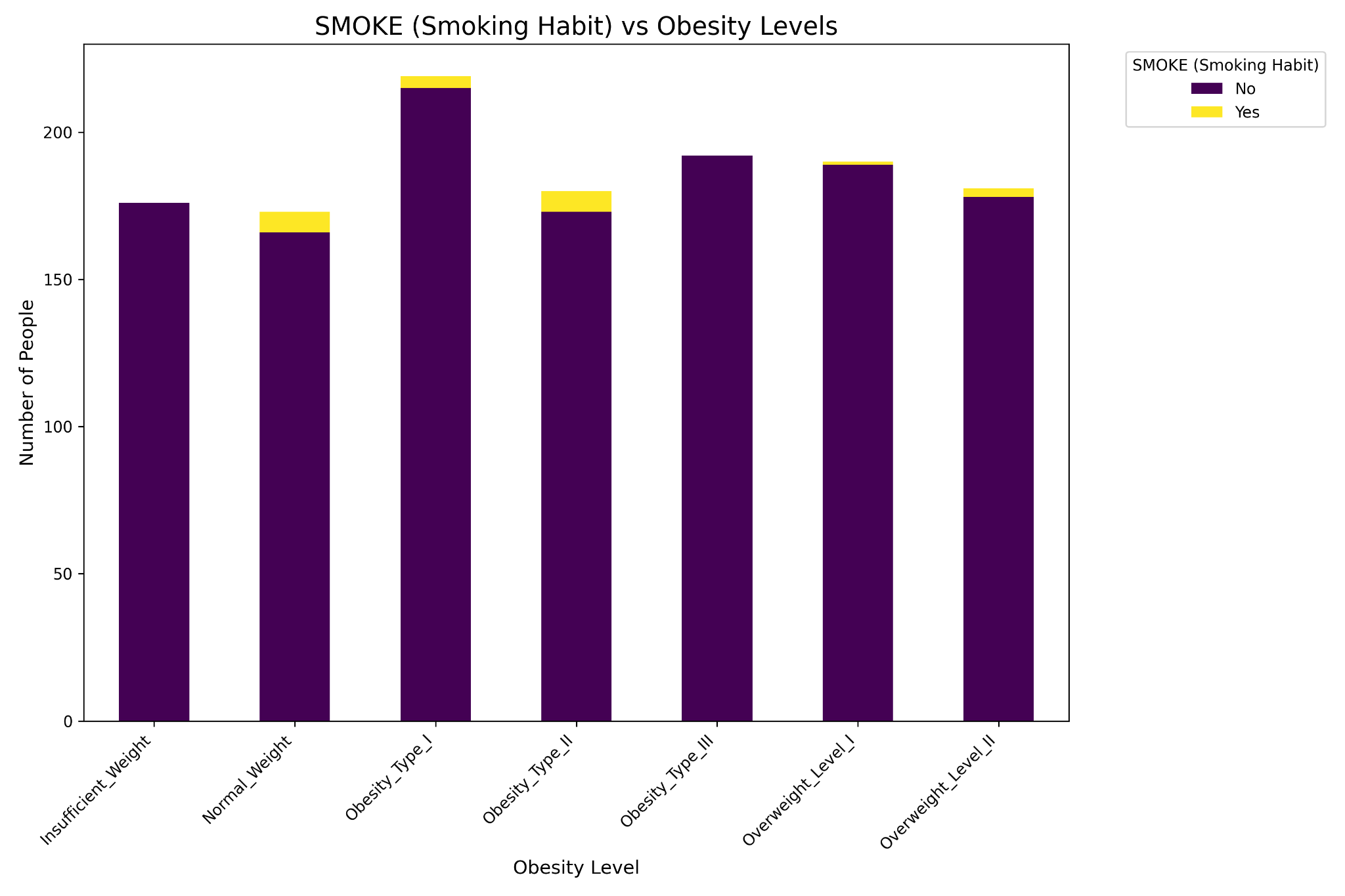
* Alcohol consumption is clearly higher in the **Overweight** category compared to the **Normal Weight/Underweight** category.
* Among individuals in the overweight group, the majority fall under the "Sometimes" category for alcohol consumption

**Inconsistencies and Feature Limitations**:

* Address discrepancies, such as high "Frequently" responses in the Normal Weight group and low "Sometimes" responses in Normal Weight and Underweight categories.

While alcohol consumption shows higher prevalence in overweight individuals under generalized categories, detailed analysis reveals significant inconsistencies, particularly in the Normal Weight group with high "Frequently" responses. CALC as a feature can be improved if given an alcohol ml consumed per week instead of a vague indicator like in this case - “frequency”.

##### 2.1.2.12 Patient who smokes vs obesity

****

**Figure 20: Smoking Factor by CH2O**

**Hypothesis:** There is a high correlation between smokers and individuals in higher obesity levels.

**Approach:**

**Stacked Bar Chart Analysis:**

* The chart visualizes the proportion of smokers and non-smokers across various obesity levels.

**Results:**

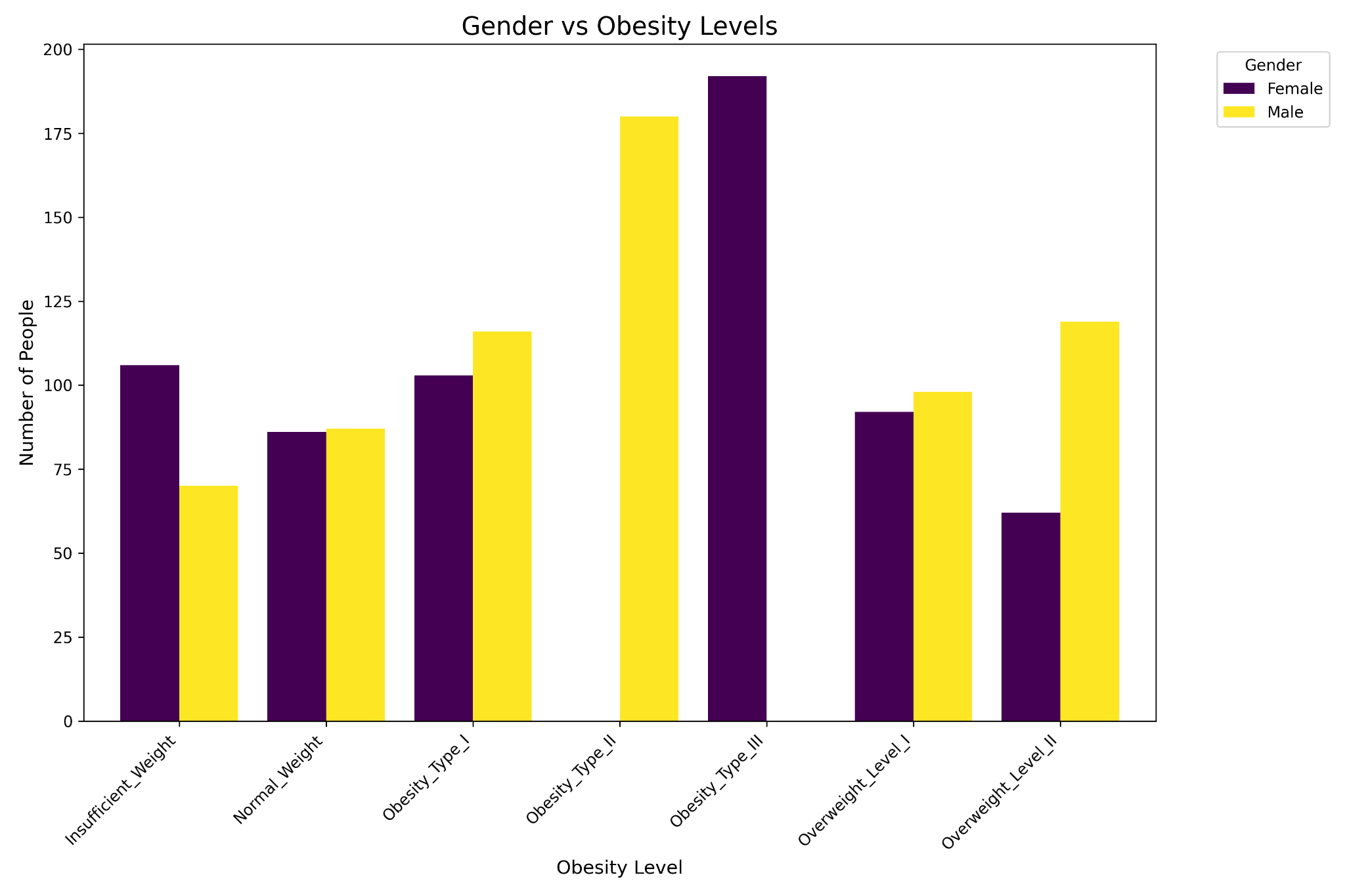
**Non-Smokers Dominate Across All Categories:**

* The majority of individuals in all obesity levels are non-smokers, with only a small proportion of smokers. This makes it difficult to establish a significant correlation between smoking habits and obesity.

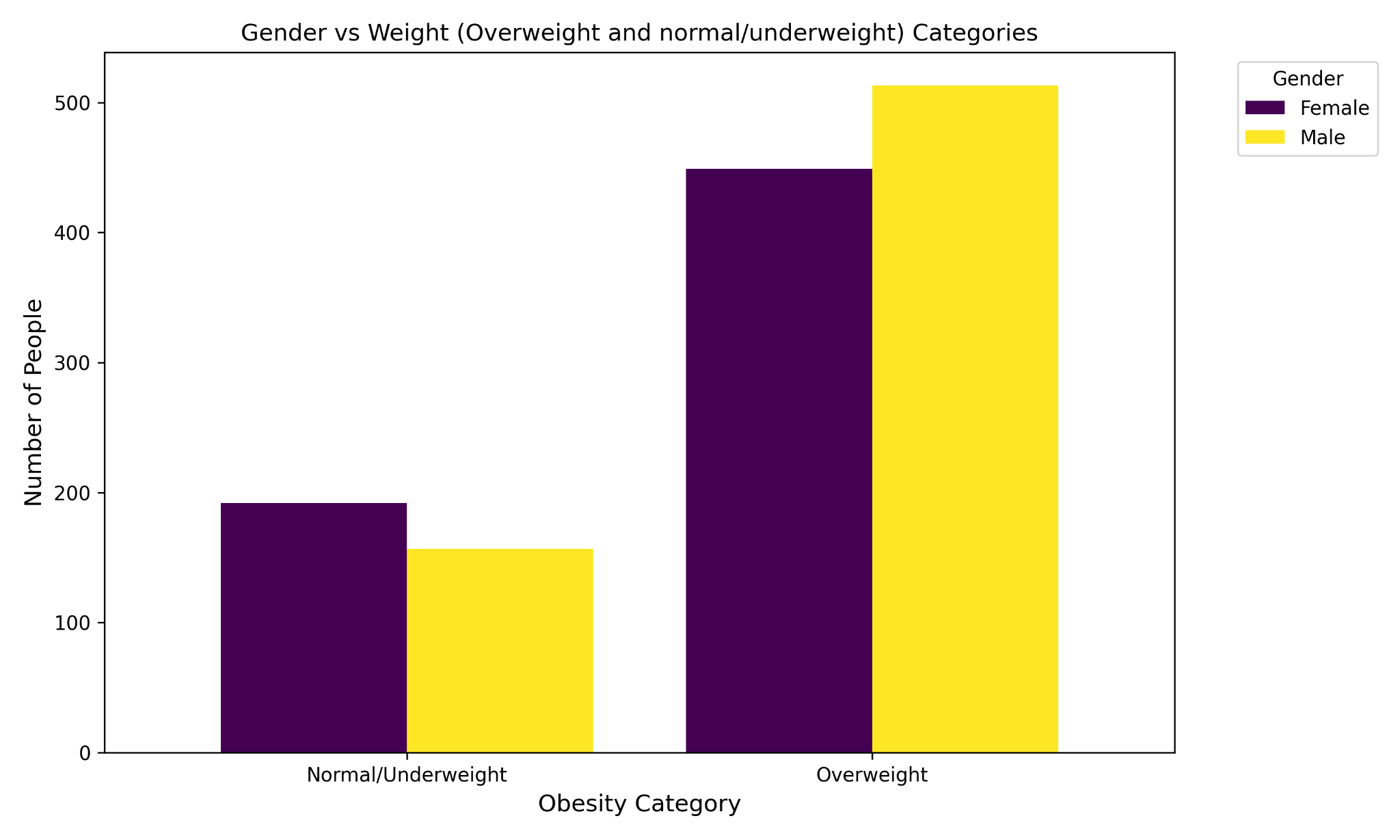
**Feature Relevance**:

* The minimal representation of smokers in the data reduces the ability to determine a meaningful relationship between smoking and obesity.
* The significant imbalance between smokers and non-smokers limits the usefulness of this feature. Including it in the model may introduce noise rather than improve predictive accuracy.

##### 2.1.2.13 Gender comparison of overweight and normal/underweight



**Figure 21: Obesity Level by Gender**



**Figure 22: Gender by (overweight and normal or underweight)**

**Hypothesis:** Gender plays a role in predicting obesity levels, with males showing a higher tendency to be overweight compared to females.

**Approach:**

**Bar Chart Analysis:**

* The two charts explore gender distributions across simplified obesity categories (Normal/Underweight vs. Overweight) and detailed obesity levels (e.g., Obesity Types I, II, and III).
* Combining normal weight and underweight categories into one group and all overweight categories into another provides a broader perspective to assess gender-based trends in obesity.

**Results:**

**Inconsistencies:**

* In the normal weight and underweight categories, females consistently outnumber males. However, the overweight category dominance by males is not absolute, as Obesity Type III shows a higher proportion of females.
* When grouping **Normal Weight and Underweight** into one category and all overweight levels into another, a broader pattern emerges:
* **Overweight Categories**: Males outnumber females significantly, supporting the idea that males have a higher tendency toward being overweight.
* **Normal/Underweight Categories**: Females dominate, indicating they are more likely to maintain healthy weight ranges or fall into underweight categories.

**Potential Plausible Conclusion**:

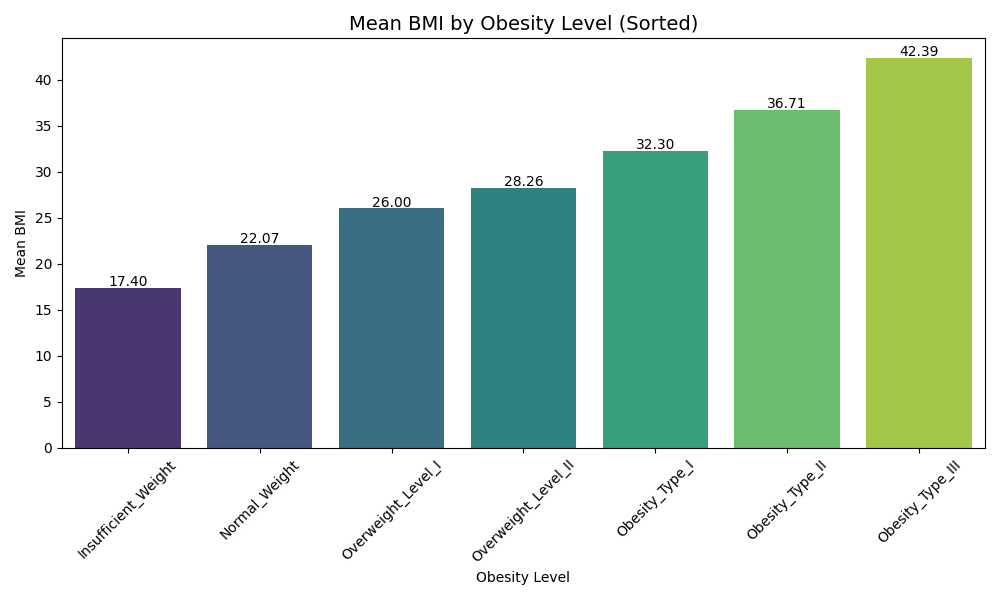
* The trend of males being more prevalent in overweight categories suggests a plausible conclusion that gender plays a role in obesity tendencies. However, the exceptions (e.g., females dominating Obesity Type III) highlight the need to incorporate complementary factors, such as diet, activity levels, or family history.

#### 2.1.3 Multivariate Analysis

The following hypotheses investigate the combined effects of multiple attributes on weight loss outcomes, providing deeper insights into how these variables interact during the clinical trial.

##### 2.1.3.1 BMI and Obesity Levels

**Hypothesis:** BMI strongly correlates with obesity levels, where higher BMI values are associated with more severe obesity categories.



**Figure 23: Obesity Level by mean BMI**

**Approach:**

The calculation for BMI is as follows:

**BMI = weight (kg) / [height (m)]^2**

The graph above shows the mean BMI for each category.

**Reference to Previous Analysis**:

* In the earlier scatter plot analysis of Height vs. Weights and individual data points representing individuals and their obesity levels, it was observed that higher weights and specific height ranges correlated with clustering into more severe obesity categories.
* BMI, as a consolidated metric of height and weight, simplifies the representation of this relationship, providing a clearer linear correlation across obesity levels

**Results:**

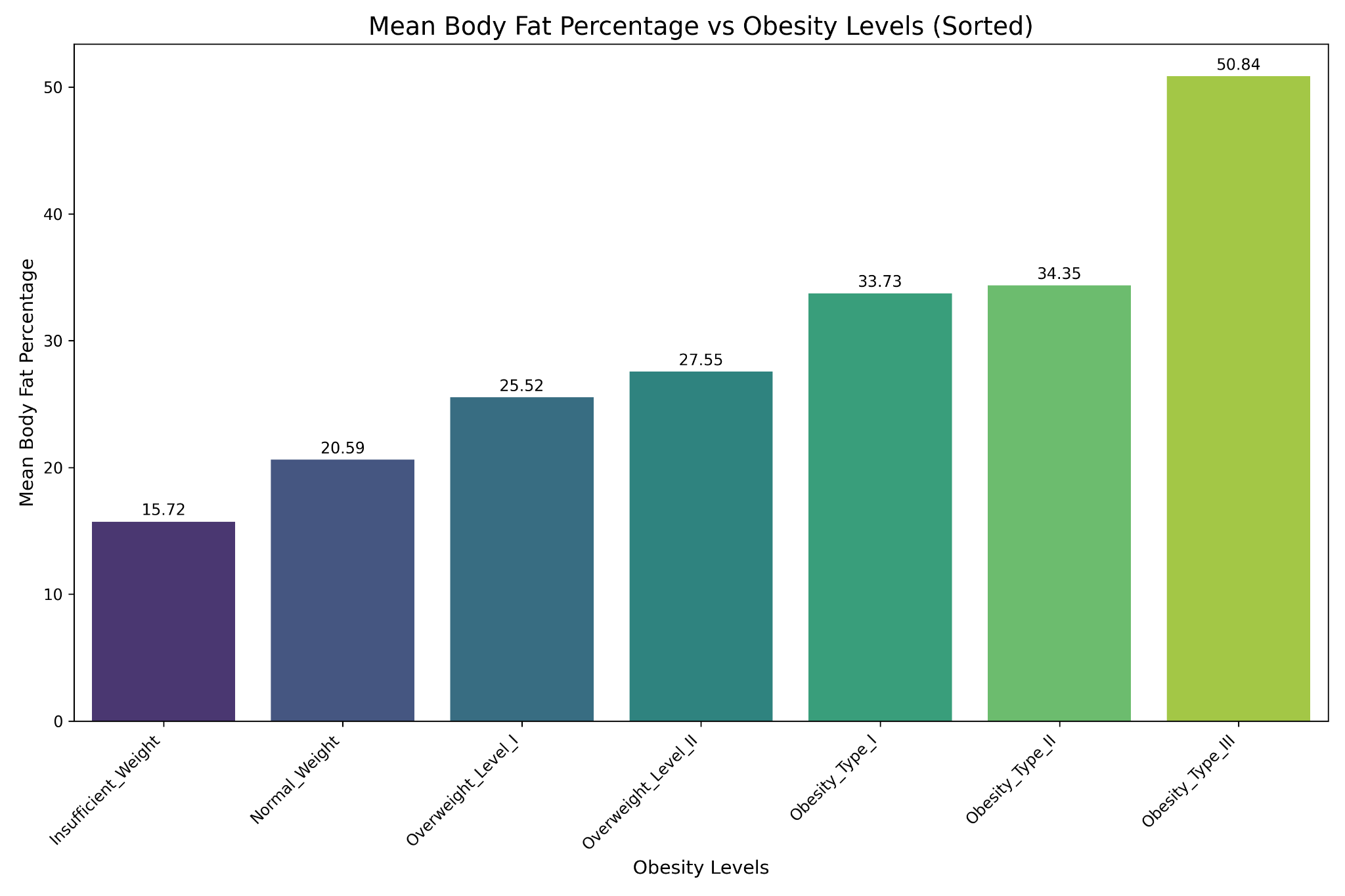
**Key Correlation:**

* BMI correlates strongly with obesity levels, aligning with weight's dominant role observed in the correlation heatmap.
* The feature's simplicity and strong association make it an essential variable in predicting obesity levels.

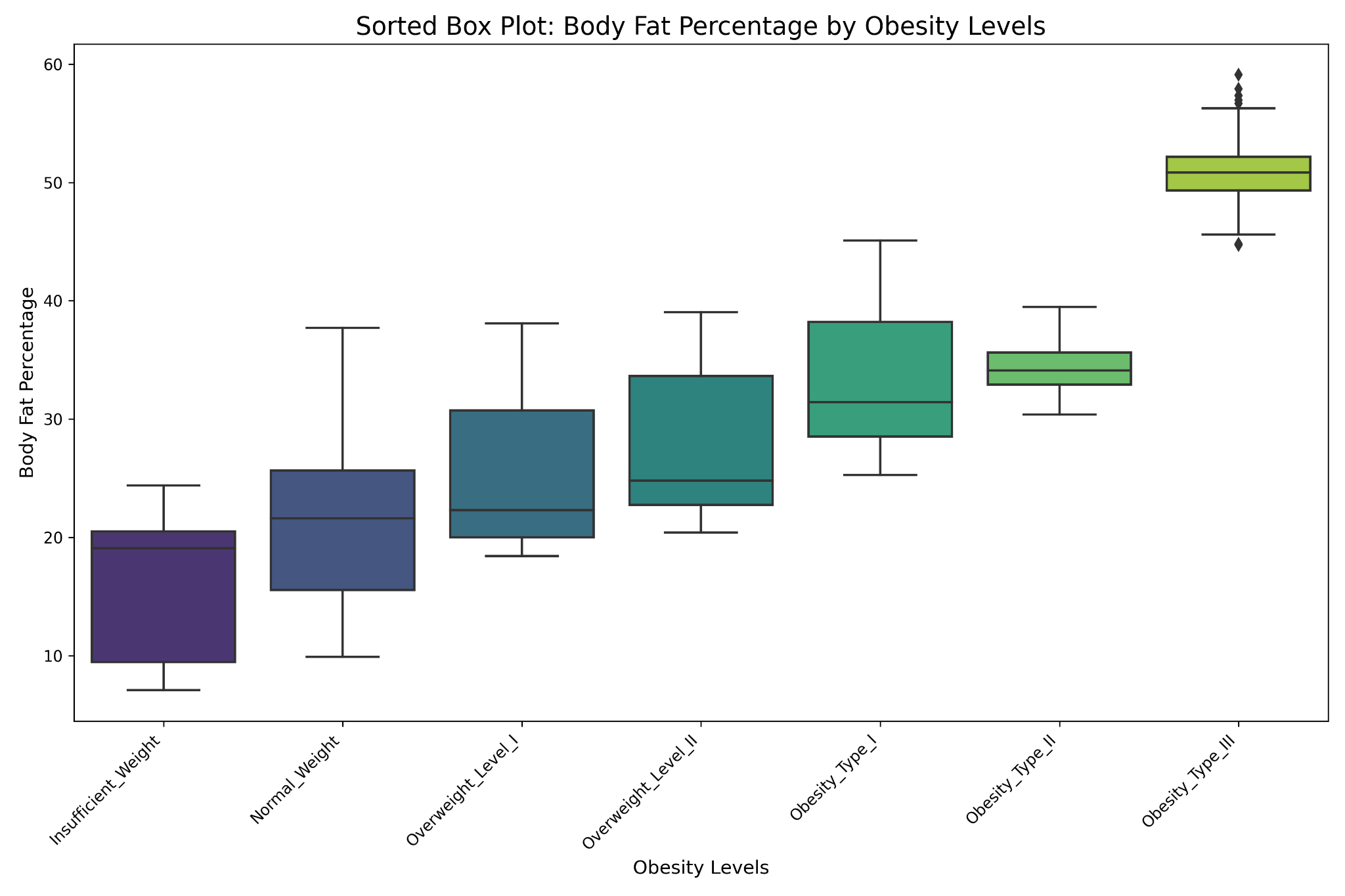
BMI not only correlates highly with obesity levels but also simplifies classification by integrating height and weight into a single metric. Its consistent progression across obesity categories validates its robustness as a predictor. This aligns with earlier observations in the Height vs. Weight scatter plot, where BMI resolves the overlapping issues while preserving the hierarchy of obesity levels.

##### 2.1.3.2 Body Fat Percentage and Obesity Levels

**Hypothesis:** Body fat percentage correlates strongly with obesity levels, with higher body fat percentages indicative of more severe obesity categories.



**Figure 24: Obesity Level by Mean Body Percentage Fat**



**Figure 25: Boxplot of Obesity Level against Body Percentage Fat**

**Approach**

**Men:**

Body Fat Percentage=(1.20×BMI)+(0.23×Age)−16.2

**Women:**

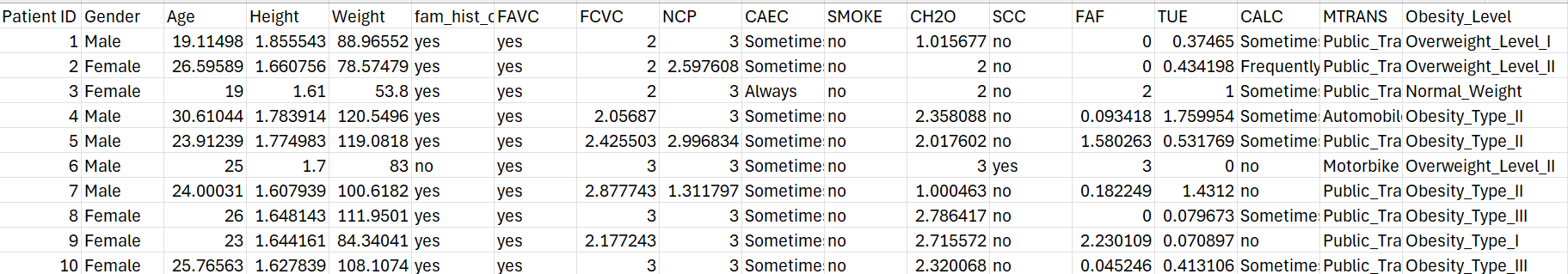
Body Fat Percentage=(1.20×BMI)+(0.23×Age)−5.4

* **Bar Graph:** The bar chart provides an overview of the mean body fat percentage for each obesity category, highlighting the trend of increasing averages with obesity severity.
* **Box Plot:** The box plot complements the bar chart by showing the spread, median, interquartile range (IQR), and outliers in body fat percentages.
  + Focuses on the variability within categories, especially for overlapping ranges or unexpected trends between Obesity Types I and II.
* Compare and contrast insights from the bar chart and box plot to identify consistencies, anomalies, and areas needing further exploration.

**Results:**

* Insufficient Weight starts at the lowest mean (15.72%), with a steady increase through Normal Weight (20.59%), Overweight Levels I and II (25.52% and 27.55%), and Obesity Types I, II, and III (33.73%, 34.35%, and 50.84%, respectively).
* The slight difference in mean body fat percentage (33.73% vs. 34.35%) indicates overlap, suggesting that individuals in these categories may share similar physiological traits, which may hinder precise classification.

### 2.2 Processing Activities



**Figure 26: Uncleaned Data**



**Figure 27: Cleaned Data**

The above figures show the before and after the processing activities of the data. Below we will provide further explanation on how the processing activities are done.

#### 2.2.1 Feature Selection

The data cleaning and preprocessing steps in this project were performed using the script clean\_data.py. This script automates key cleaning processes to ensure the dataset is consistent, structured, and ready for machine learning tasks. Below is an explanation of the data processing activities:

The uncleaned dataset contains a mix of numerical and categorical variables, some of which are in text or inconsistent formats. For example:

* Height and Weight columns allow the calculation of BMI, but BMI is not explicitly included.
* Categorical variables, such as Gender and MTRANS, are in text format.

The clean\_data.py script was developed to standardize and prepare the dataset for analysis. The main cleaning tasks include:

* **Handling Missing Values:**
  + Numerical columns with missing values were imputed using the median to reduce the influence of extreme values.
  + This ensures that no rows are dropped unnecessarily due to missing data.
* A new column, BMI (Body Mass Index), was created and calculated using the formula, BMI = Weight(Kg) / Height (m).
* **Categorical Encoding:**
  + Categorical variables, such as Gender, FAVC (High-Calorie Food Consumption), and MTRANS (Mode of Transportation), were converted to numerical codes using label encoding. For instance:
    - Male and Female were encoded as 0 and 1, respectively.
    - Transportation modes like Public\_Transport or Walking were assigned numeric values.
  + This step ensures compatibility with machine learning algorithms, which usually work better with numeric inputs.
* **Removed columns:**
  + The Patient ID column, which does not contribute to predictive modeling, was removed from the dataset to avoid potential overfitting.
  + Unnamed: 18: A blank column created by an extra delimiter was dropped as it contained no useful information.
* **Optional Split of Data**
  + The script allows splitting the cleaned dataset into training, validation, and test sets based on user-defined proportions (e.g. , 70-15-15). This split supports robust model evaluation.

#### 2.2.2 Encoding Categorical Variables

To make categorical data suitable for machine learning algorithms, the code applied the following transformations:

##### 2.2.2.1 Gender

* Original Values: Male, Female
* Encoded Values:
  + 0 → Male
  + 1 → Female

##### 2.2.2.2 FAVC (High-Calorie Food Consumption)

* Original Values: Yes, No
* Encoded Values:
  + 0 → No
  + 1 → Yes

This column now indicates whether the patient frequently consumes high-calorie foods, with 1 meaning frequent consumption.

##### 2.2.2.3 CAEC (Food Consumption Between Meals)

* Original Values: No, Sometimes, Frequently, Always
* Encoded Values:
  + 0 → No
  + 1 → Sometimes
  + 2 → Frequently
  + 3 → Always

The encoding introduces a hierarchy, where higher numbers represent a greater frequency of consuming food between meals.

##### 2.2.2.4 SMOKE

* Original Values: Yes, No
* Encoded Values:
  + 0 → No
  + 1 → Yes

This indicates whether the patient smokes (1) or not (0).

##### 2.2.2.5 SCC (Self-Monitoring of Calories)

* Original Values: Yes, No
* Encoded Values:
  + 0 → No
  + 1 → Yes

This column reflects whether the patient actively tracks their calorie consumption.

##### 2.2.2.6 CALC (Alcohol Consumption Frequency)

* Original Values: Sometimes, Frequently, Always
* Encoded Values:
  + 0 → No
  + 1 → Sometimes
  + 2 → Frequently
  + 3 → Always

This represents the frequency of alcohol consumption, with higher values indicating more frequent consumption.

##### 2.2.2.7 MTRANS (Mode of Transportation)

* Original Values: Automobile, Motorbike, Walking, Public\_Transportation, Bike
* Encoded Values:
  + 0 → Automobile
  + 1 → Motorbike
  + 2 → Walking
  + 3 → Public Transportation
  + 4 → Bike

The numeric values correspond to different transportation modes, with no inherent order or hierarchy.

##### 2.2.2.8 Obesity Level

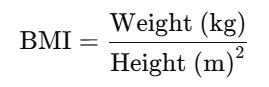
* Original Values:
  + Insufficient\_Weight
  + Normal\_Weight
  + Overweight\_Level\_I
  + Overweight\_Level\_II
  + Obesity\_Type\_I
  + Obesity\_Type\_II
  + Obesity\_Type\_III
* Encoded Values:
  + 0 → Insufficient\_Weight
  + 1 → Normal\_Weight
  + 2 → Overweight\_Level\_I
  + 3 → Overweight\_Level\_II
  + 4 → Obesity\_Type\_I
  + 5 → Obesity\_Type\_II
  + 6 → Obesity\_Type\_III
* **Numerical Relationship:**
  + The encoding includes Insufficient\_Weight at the lowest end (0), indicating that underweight individuals have the least weight-related health concerns relative to the others.
  + The numeric values progress to 6, which represents the most severe obesity classification (Obesity\_Type\_III).
* **Purpose:**
  + This addition ensures that underweight individuals are accounted for in the analysis and predictions.
  + It highlights a subgroup that may not benefit from the weight-loss drug and could potentially suffer adverse side effects.

#### 2.2.3 Feature Engineering (Creating New Features)

Data transformations were applied to prepare features for analysis. These transformations facilitated the training of machine learning models by improving feature relevance and overall model performance in prediction

##### 2.2.3.1 BMI (Body Mass Index)

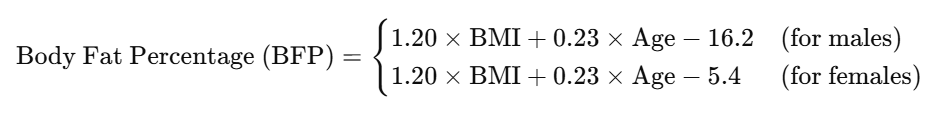
Body Mass Index (BMI) is a widely used measure to categorize individuals based on their weight in relation to their height. The formula for BMI is:



A new column for BMI was computed using this formula. The decision to include BMI as a feature was driven by exploratory data analysis (EDA) in the previous sections, which revealed that the combination of weight and height was effective in clustering individual data points into obesity level categories. This clustering indicated that BMI serves as a meaningful proxy for distinguishing between different obesity levels, making it a valuable addition to the feature set.

##### 2.2.3.2 Body Fat Percentage

Body fat percentage, another critical indicator of an individual's obesity level, was not directly provided in the original dataset. To estimate body fat percentage, we used a formula that incorporates BMI and gender, as gender-specific physiological differences influence body composition. The formula used was as follows:



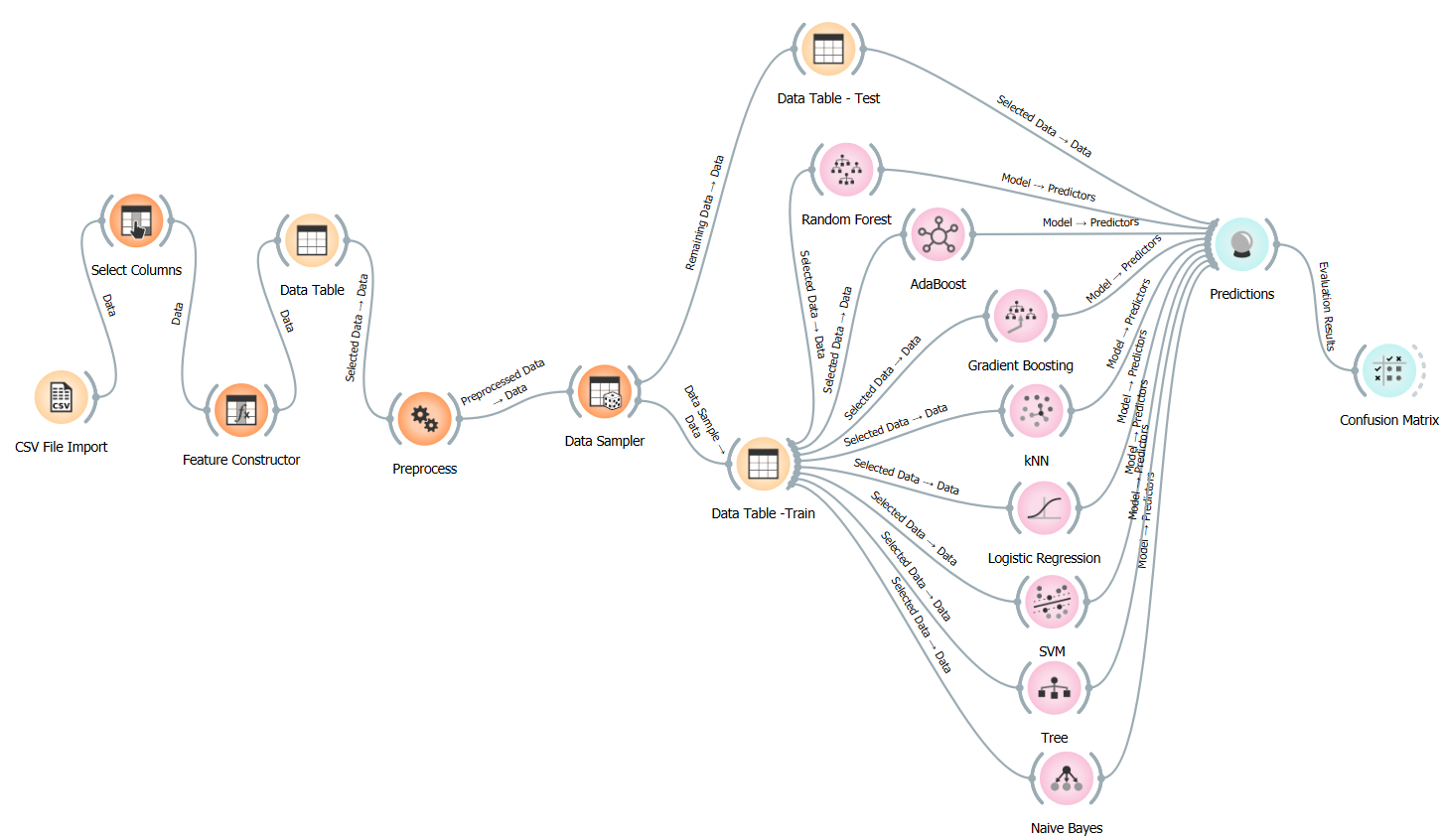
By transforming BMI and gender into an estimated body fat percentage, we added another derived feature to the dataset. This transformation was justified by its strong association with obesity levels, as body fat percentage provides a finer granularity than BMI alone in assessing an individual's adiposity.

### 2.3 Orange Pipeline

In this project, we utilized Orange, an open-source data visualization and analysis tool, to establish a baseline for model performance and experiment with various machine learning algorithms. Orange’s interactive interface allowed us to quickly test different models and compare their metrics in a visual, drag-and-drop environment. This initial baseline provided insights into model performance across different metrics such as accuracy, precision, recall, and AUC scores, enabling a straightforward assessment of each model’s strengths and weaknesses.

Using Orange helped streamline the early experimentation phase, allowing us to explore models like Decision Trees, Naive Bayes, Random Forests, and Support Vector Machines without extensive coding. Once we had a clear understanding of which models performed best, we moved to a code-based pipeline for deeper customization and tuning. This pipeline allowed for greater control over hyperparameters, facilitated advanced evaluations, and integrated with our CLI and GUI interfaces for a comprehensive, flexible workflow.

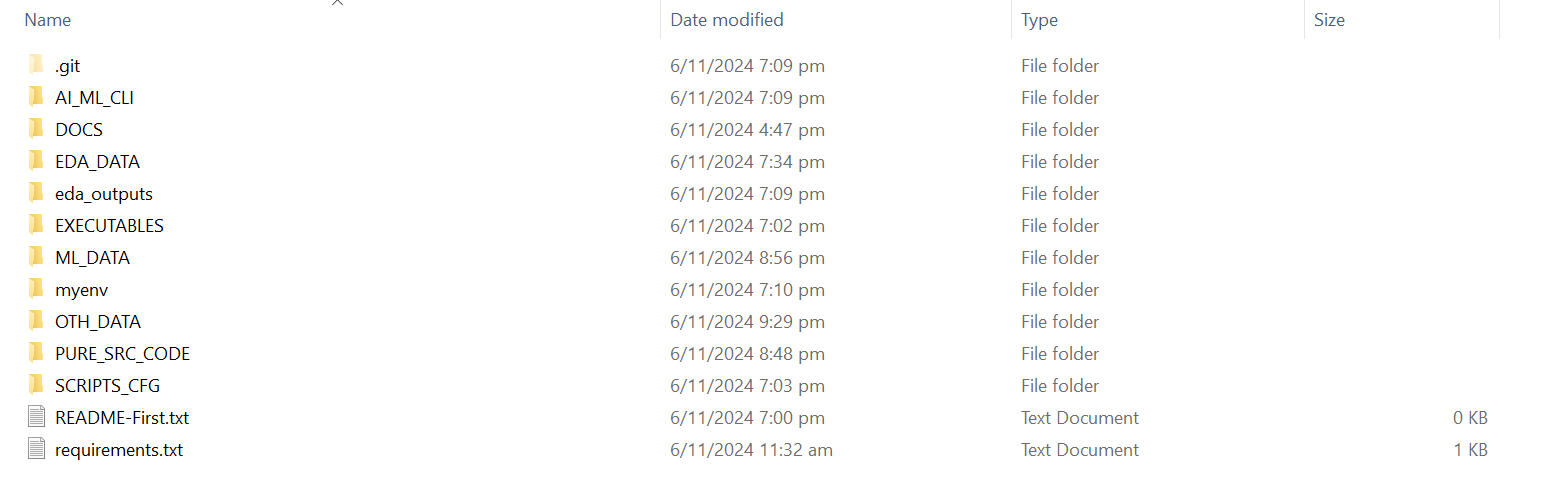
This dual approach—starting with Orange and refining in code—enabled us to efficiently experiment with model configurations, ensuring a robust selection process before committing to a final model for deployment.



**Fig 28 : Orange pipeline structure**

### 

### 2.4 Code pipeline



**Fig 29 : File directory**

#### **2.4.1 Directory Structure Overview**

* **Win\_10-Python\_v3.11:** This folder contains all the original source python scripts, along with the dataset, the visualizations and executables.
  + **PURE\_SRC\_CODE**: Contains all the core Python scripts (clean\_data.py, perform\_eda.py, model\_training.py, perform\_prediction.py) and the main control script pipeline.sh that orchestrates the execution of the executables. Running pipeline.sh initiates the pipeline process, allowing users to choose tasks like data cleaning, EDA, model training, and prediction.
  + **EXECUTABLES**: Holds the .exe versions of the main Python scripts (clean\_data.exe, perform\_eda.exe, model\_training.exe, perform\_prediction.exe). These executables are called by pipeline.sh in PURE\_SRC\_CODE, enabling users to run the pipeline without directly executing Python scripts.
  + **SCRIPTS\_CFG**: Contains the config.txt file, which is used to configure paths, parameters, and settings for each stage of the pipeline. This file is referenced by each executable to ensure they use the correct inputs and outputs.
  + **OTH\_DATA**: Contains raw and intermediate data folders:
    - **training\_data**: Stores the initial dataset for training models.
    - **testing\_data**: Holds datasets used for model evaluation and testing.
    - **cleaned\_data**: Contains preprocessed data after cleaning, ready for analysis or modeling.
  + **EDA\_DATA**: Stores figures and data generated from the Exploratory Data Analysis (EDA) stage. This folder includes plots and other visuals that help in understanding data patterns and distributions.
  + **ML\_DATA**: Structured to hold model outputs and prediction results:
    - **model\_outputs**: Contains saved trained models (e.g., .pkl files) generated by the model training stage.
    - **predict\_outputs**: Holds prediction result files, such as prediction\_report.csv, which summarize model performance on test data.
    - **visualize\_output**: Stores prediction-related figures and graphs, such as confusion matrices and ROC curves, to provide insights into model performance.
* **JupyterNB-Python\_v3.11:** Similar to the **Win\_10-Python\_v3.11,** this folder contains all the visualizations, data set. However, the original python scripts are now converted to ipynb files as Jupyter notebook python format. The rest of the directory still remains the same.
* The decision to utilize Jupyter Notebooks was driven by their ability to enhance the unit testing process, as they allow the team to view code execution results inline. This feature simplifies debugging and provides immediate feedback, making it easier to validate individual code blocks during development.
  + PURE\_SRC\_CODE
    - EXECUTABLES
    - SCRIPTS\_CFG
    - OTH\_DATA
      * training\_data
      * testing\_data
      * cleaned\_data
    - EDA\_DATA
    - **ML\_DATA**:
      * model\_outputs
      * predict\_outputs
      * visualize\_output

#### **2.4.2 Pipeline Flow Description**

1. **Initiation**: The pipeline is initiated by running pipeline.sh in the PURE\_SRC\_CODE folder. This script presents a menu for selecting different stages of the pipeline.
2. **Data Cleaning** (clean\_data.exe): The pipeline first accesses clean\_data.exe in the EXECUTABLES folder. This executable reads raw data from OTH\_DATA/training\_data, processes it (e.g., handling missing values, encoding categorical variables), and outputs cleaned data to OTH\_DATA/cleaned\_data.
3. **Exploratory Data Analysis (EDA)** (perform\_eda.exe): The perform\_eda.exe executable analyzes the cleaned data, generating figures and summaries that are stored in EDA\_DATA. These visuals provide insights into data distribution, trends, and outliers.
4. **Model Training** (model\_training.exe): The pipeline then moves to training models using model\_training.exe. This executable reads data from OTH\_DATA/training\_data (or OTH\_DATA/cleaned\_data if preprocessed data is required), trains various machine learning models, and saves the trained models to ML\_DATA/model\_outputs.
5. **Prediction and Evaluation** (perform\_prediction.exe): Finally, perform\_prediction.exe is executed to generate predictions using the trained models. The results are stored in ML\_DATA/predict\_outputs as files like prediction\_report.csv, while graphs and figures related to prediction (e.g., ROC curves, confusion matrices) are saved to ML\_DATA/visualize\_output.

### 

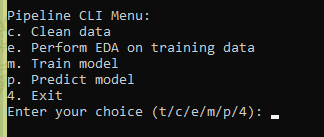
### 

### 

#### **2.4.3 Execution Notes**

* **Config File**: config.txt in the SCRIPTS\_CFG folder provides configuration settings for paths, parameters, and other options required by each stage of the pipeline. Each executable reads this file to determine where to find inputs and where to save outputs.
* **Running the Pipeline**: To execute the pipeline, navigate to the PURE\_SRC\_CODE directory and run pipeline.sh. This script will use the executables in the EXECUTABLES folder, configured by config.txt, to perform each pipeline stage sequentially.

### 2.5 CLI



**Fig 30 : Command Line Interface Menu**

The CLI (Command-Line Interface) pipeline menu is a command-line tool designed to streamline the process of data preparation, analysis, model training, and prediction. The menu is organized into distinct options, each corresponding to a different stage in the pipeline, allowing users to execute specific tasks with ease. The CLI menu guides the user through the process of cleaning data, conducting exploratory data analysis (EDA), training machine learning models, and making predictions, with each stage customized based on user-specified parameters.

**Pipeline Menu Options**

When the user runs the pipeline.sh script located in the PURE\_SRC\_CODE directory, a CLI menu is displayed. The user can select from several options by entering the corresponding letter or number:

* c. Clean data
* e. Perform EDA on training data
* m. Train model
* p. Predict model
* 4. Exit

Each option triggers a separate executable in the EXECUTABLES directory, allowing users to control the flow of tasks without needing direct interaction with individual scripts. Below, we detail each option and its functionality, as well as how users can configure parameters through a configuration file.

**In-depth description of each option**

**Data Cleaning (Option 'c')**

* **Function**: Initiates the data cleaning process using clean\_data.exe.
* **Process**: When this option is chosen, the executable reads raw data from the OTH\_DATA/training\_data folder, performs cleaning operations (such as handling missing values, encoding categorical features, and dropping unnecessary columns), and saves the cleaned dataset to OTH\_DATA/cleaned\_data.
* **Configuration**: Upon selecting this option, the config.txt file is opened in the default text editor. Users can specify:
  + input\_folder\_clean: The folder where raw data is located.
  + output\_folder\_clean: The folder where cleaned data should be saved.
  + cleaned\_file\_suffix: A suffix to be appended to the cleaned file's name.
* **User Interaction**: After reviewing or updating the configuration file, the user presses Enter to proceed with data cleaning based on the specified paths and settings.

**Exploratory Data Analysis (EDA) (Option 'e')**

* **Function**: Executes EDA on the cleaned data using perform\_eda.exe.
* **Process**: The executable reads data from OTH\_DATA/cleaned\_data, performs EDA tasks (such as generating histograms, box plots, and summary statistics), and saves the results to the EDA\_DATA folder. The generated visuals and reports help in understanding data distributions, relationships, and potential outliers.
* **Configuration**: Upon selecting this option, config.txt opens, allowing the user to specify:
  + input\_folder\_eda: The folder containing the data for EDA.
  + output\_folder\_eda: The folder where EDA outputs should be saved.
  + eda\_file\_suffix: A suffix to be added to each EDA output file.
* **User Interaction**: The user reviews the configuration file, updates it if necessary, and presses Enter to continue. EDA is then conducted based on the configured paths and settings.

**Model Training (Option 'm')**

* **Function**: Launches the model training process with model\_training.exe.
* **Process**: The executable reads training data from OTH\_DATA/cleaned\_data or OTH\_DATA/training\_data, depending on the configuration. It then trains various machine learning models (e.g., Logistic Regression, Decision Tree, Random Forest) with specified hyperparameters and saves the trained models to ML\_DATA/model\_outputs.
* **Configuration**: When this option is selected, config.txt opens, allowing the user to input:
  + input\_folder: The path to the folder containing the data for training.
  + output\_folder: The path to the folder where trained models should be saved.
  + model\_name\_suffix: A suffix to append to each model’s file name (e.g., \_v1 for versioning).
  + Model parameters, such as the type of regularization for Logistic Regression, max depth for Decision Tree, number of estimators for Random Forest, etc.
* **User Interaction**: The user can modify the configuration file to specify or update model parameters and paths before pressing Enter to proceed. This flexibility allows users to fine-tune model performance and save models with specific configurations.

**Model Prediction (Option 'p')**

* **Function**: Initiates the prediction process with perform\_prediction.exe.
* **Process**: The executable reads test data from OTH\_DATA/testing\_data and loads models from ML\_DATA/model\_outputs to generate predictions. It saves prediction results (e.g., classification reports, confusion matrices) to ML\_DATA/predict\_outputs and visualizations (e.g., ROC curves) to ML\_DATA/visualize\_output.
* **Configuration**: On selecting this option, config.txt opens, allowing the user to specify:
  + input\_folder: The folder where the test data is located.
  + model\_folder: The folder containing trained models to be used for predictions.
  + output\_folder: The folder where prediction results and visualizations should be saved.
* **User Interaction**: The user reviews the configuration file to ensure the paths and settings align with the desired test setup, then presses Enter to proceed. The script then generates and saves the prediction outputs and evaluation metrics.

**Exit (Option '4')**

* **Function**: Exits the CLI menu and terminates the pipeline process.

## 

## 

## 

## 

## 

## 3 Model Training

A wide range of supervised machine learning algorithms were employed to train and evaluate models on a labeled dataset. Each algorithm belongs to a distinct category of machine learning techniques, enabling the exploration of various model types to determine the most effective one for the task at hand. Below is an overview of each model, including the algorithm, training approach, and key hyperparameters.

### 3.1 Models assessed

#### 3.1.1 Ensemble Method

Ensemble methods in machine learning enhance predictive performance by combining multiple models, or "learners." The fundamental principle is that multiple weak learners can collectively form a strong learner. By aggregating predictions from these models, ensemble methods can effectively capture complex data patterns and reduce overall errors. Random forest, Adaptive boosting and Gradient boosting fall under this method.

3.1.1.1 Random Forest

Random Forest is an ensemble learning algorithm that enhances accuracy in classification and regression tasks by combining the predictions of multiple decision trees. Each tree is trained on a random subset of the training data, using different samples of data points and features. This diversity improves generalization and reduces overfitting. The final prediction is made by aggregating the outputs of the individual trees, using majority voting for classification and averaging for regression, resulting in a more robust and reliable model.

Hyperparamters -

* `n\_estimators`: Number of trees in the forest; increasing this can improve performance but also increase computation time.
* `max\_depth`: Controls the maximum depth of each tree; deeper trees can capture more complex patterns but may overfit.
* min\_samples\_split: Increasing to 5 or 10 could help reduce overfitting.
* min\_samples\_leaf: Setting to a higher value (e.g., 2 or 5) can prevent the model from learning too finely on noise.

"RandomForest": RandomForestClassifier(n\_estimators=300, max\_depth=10, min\_samples\_split=2, min\_samples\_leaf=1)

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Can model complex, non-linear relationships by combining multiple trees. * Reduces overfitting through averaging, which stabilizes predictions. * Suitable for both numerical and categorical data without requiring feature scaling. * Allows parallel processing, making it faster to train on large datasets. | * Hard to interpret as a single model, though feature importance can be derived. * Resource-intensive, requiring considerable memory and processing power. * Can still be influenced by noisy data, but less than that of individual trees. |

##### 3.1.1.2 Adaptive boosting

AdaBoost is an ensemble method that combines multiple weak learners, typically small decision trees, by focusing on misclassified data points. Each subsequent model is trained to correct the errors made by the previous ones, effectively boosting the overall performance of the model.

Hyperparameters -

* n\_estimators: Number of weak learners in the ensemble; increasing this can improve accuracy but may lead to overfitting.
* learning\_rate: Determines the contribution of each weak learner; lower values make the model more robust but require more estimators.

"AdaBoost": AdaBoostClassifier(n\_estimators=200, learning\_rate=0.5)

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Effectively combines weak learners into a strong classifier with high accuracy. * Can capture intricate patterns in complex datasets. * Reduces the risk of overfitting by adjusting sample weights based on performance. | * Sensitive to noise and outliers, which can degrade performance. * Computationally intensive, especially on large datasets. * Requires careful tuning of hyperparameters (number of learners, learning rate) for best results. |

##### 3.1.1.3 Gradient Boosting

Gradient Boosting builds models sequentially, with each new model correcting the residual errors from the previous ones. It uses gradient descent to minimize the loss function and improve predictions incrementally.

Hyperparameters -

* n\_estimators: Number of boosting stages to be run; more stages can improve performance but increase the risk of overfitting
* learning\_rate: Controls how much each tree contributes to the final prediction; lower values result in more cautious updates
* max\_depth: Limits the depth of individual trees to prevent overfitting.
* min\_samples\_split/min\_samples\_leaf: Similar to RandomForest, increasing these values can help reduce overfitting.

"GradientBoosting": GradientBoostingClassifier(n\_estimators=200, learning\_rate=0.05, max\_depth=20, min\_samples\_split=10, min\_samples\_leaf=5)

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Effectively models non-linear relationships and complex data patterns. * Typically achieves high accuracy, often outperforming simpler models. * Requires minimal preprocessing, handling various data types and missing values well. | * Difficult to interpret, though feature importance can still be derived. * Computationally expensive with long training times, especially for large datasets. * Prone to overfitting without careful tuning of hyperparameters and regularization. |

##### 3.3.1.4 XGBoost

XGBoost is an optimized implementation of gradient boosting that focuses on speed and performance. It constructs trees in a sequence, each correcting errors of the previous ones.

Hyperparameters:

* n\_estimators: Number of boosting rounds; more trees can improve performance but may also lead to overfitting.
* learning\_rate: Step size shrinkage used to prevent overfitting; smaller values allow for more cautious updates.
* max\_depth: Controls the maximum depth of individual trees; deeper trees can capture more complex relationships.
* subsample: Proportion of samples used for fitting the individual base learners; values less than 1 introduce randomness and can help prevent overfitting.

**"XGBoost": XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss', n\_estimators=500, learning\_rate=0.05, max\_depth=8, subsample=0.8)**

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Renowned for its speed and scalability, making it efficient for large datasets. * Often outperforms other algorithms in benchmark tests. * Includes feature importance metrics and regularization to prevent overfitting. | * The complexity of the algorithm requires a solid understanding of its mechanics for effective use. * Significant computational resources may be needed, particularly with large datasets or many iterations. * Performance heavily relies on hyperparameter tuning, necessitating careful adjustments for optimal results. |

#### 3.1.2 K-Nearest Neighbors (KNN)

KNN is a distance-based classification algorithm that assigns a class to a data point based on the majority class among its nearest neighbors in the feature space. It is a "lazy" learner, meaning it does not build an explicit model but rather relies on the training data during prediction.

Hyperparameters:

* n\_neighbors: Number of neighbors to consider for making predictions; a larger number of neighbors can help smooth out predictions but may result in a loss of detail when capturing local patterns. In contrast, a smaller number of neighbors can make the model sensitive to noise, as the algorithm heavily relies on the nearest points, which may include outliers or misclassified data. This reliance can lead to unstable predictions. In this case, the parameter utilized was 8, as a baseline.
* weights: Determines how much influence each neighbor has, either uniformly or based on distance.

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Straightforward and easy to understand; suitable for beginners. * Adapts to new data without retraining, as it relies on stored training data. * Minimal configuration needed, with only a few hyperparameters like k (number of neighbors). | * Memory-intensive and computationally costly for large datasets, as it stores all training data. * Performs poorly in high-dimensional spaces, where distances between points become less meaningful. * Prone to noise with low k values; higher k values can lead to oversimplified predictions. |

#### 3.1.3 Support Vector Classifier (SVC)

SVC aims to find the optimal hyperplane that separates classes by maximizing the margin between them. It can handle linear and non-linear relationships through the use of different kernel functions.

Hyperparameters:

* C: Regularization parameter that balances the trade-off between achieving a low training error and a low testing error; a smaller C encourages a smoother decision boundary.
* kernel: Specifies the type of kernel function to use (e.g., linear, polynomial, radial basis); affects the model’s flexibility.
* gamma: Defines the influence of a single training example; a low value means far and smooth decision boundaries, while a high value can lead to overfitting.

"SVC": SVC(C=1, kernel='rbf', gamma='scale')

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Computationally efficient, especially with kernel tricks to handle non-linear boundaries. * Works well with high-dimensional data. * Does not require assumptions about data distribution, though scaling data is recommended. | * Sensitive to kernel choice, which significantly impacts performance. * Does not easily produce probability scores, only class labels. * No straightforward way to determine variable importance |

#### 3.1.4 Decision Tree

Decision Tree models recursively split the data based on feature values, forming a tree structure for making predictions. It selects splits that maximize class purity, allowing for interpretable decision-making.

Hyperparameters:

* max\_depth: Limits the depth of the tree; shallower trees may underfit, while deeper trees can overfit.
* min\_samples\_split: Minimum number of samples required to split a node; higher values prevent splits that do not contribute significantly to model performance.
* min\_samples\_leaf: Minimum number of samples that must be present in a leaf node; larger values help prevent overfitting.

"DecisionTree": DecisionTreeClassifier(max\_depth=15, min\_samples\_split=5,

min\_samples\_leaf=2)

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Highly interpretable, even for non-technical audiences. * Flexible, handling both numerical and categorical data. * Works well with noisy data and outliers, as it doesn’t require assumptions about data distribution. | * Prone to overfitting, especially with deep trees or many features. * Unstable, as small changes in data can lead to major structural changes. * Limited scalability with large datasets or many features. |

#### 3.1.5 Logistic Regression

Description: Logistic Regression models the probability of class membership using a logistic function, making it suitable for binary classification tasks. It estimates coefficients to represent the relationship between input features and the output class.

Hyperparameters:

* penalty: Type of regularization to apply (e.g., 'l1', 'l2'); helps prevent overfitting by constraining the coefficient values.
* C: Inverse of regularization strength; smaller values imply stronger regularization.

"LogisticRegression": LogisticRegression(max\_iter=200, penalty='l2', C=1.0),

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Simple, and easy to implement. * Flexible, able to handle both categorical and continuous variables and extend to model complex relationships. * Fast training, scalable even with high-dimensional data. * Well-calibrated probabilities, where predicted probabilities are generally accurate and interpretable. | * Assumes a linear relationship between features and the outcome, limiting flexibility. * Sensitive to outliers, which can skew predictions. * Limited to binary outcomes without additional modifications |

#### 3.1.6 Naive Bayes (GaussianNB)

Description: Gaussian Naive Bayes is a probabilistic model based on Bayes' Theorem, assuming independence among features. It calculates the probabilities of class membership based on the feature distributions.

Hyperparameters:

* var\_smoothing: Adjusts the variance to prevent numerical instability in probability calculations; helpful for handling very small probabilities.

"NaiveBayes": GaussianNB(var\_smoothing=1e-9)

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Extremely fast and efficient, even on large datasets. * Works well with categorical data and multi-class problems. * Requires minimal training data if the independence assumption holds. | * Assumes feature independence, which is rarely true in real-world applications. * Faces the "zero-frequency problem" when test data has categories not present in training data, though this can be mitigated with smoothing techniques. * Probability estimates may be unreliable, so caution is advised in interpreting them. |

#### 3.1.7 Neural Network (MLPClassifier)

Description: A Multi-Layer Perceptron (MLP) is a type of neural network that can learn complex, non-linear patterns in data. It uses backpropagation to adjust weights and biases based on errors in predictions.

Hyperparameters:

* hidden\_layer\_sizes: Defines the number and size of hidden layers; more neurons and layers can capture more complexity.
* learning\_rate\_init: Starting learning rate for weight updates; lower values can stabilize training.
* alpha: Regularization parameter to reduce overfitting by constraining weights.

"NeuralNetwork": MLPClassifier(hidden\_layer\_sizes=(100, 50, 50, 50), max\_iter=800, learning\_rate\_init=0.001, alpha=0.0001),

|  |  |
| --- | --- |
| Advantages : | Disadvantages : |
| * Can model complex and non-linear relationships, ideal for data with intricate patterns. * Learns relevant features directly from data, reducing the need for manual feature engineering. * Suitable for various data types, including time series, images, and text. | * Computationally expensive, requiring significant resources, especially for deep networks. * Prone to overfitting on small datasets, requiring regularization to generalize well. * Harder to interpret than simpler models, making it challenging to understand its predictions. * Performance relies on large datasets and careful hyperparameter tuning. |

### **3.2 Hypothesis: Random Forest as the Optimal Model for This Use Case**

**Hypothesis Statement:**Given the exploratory data analysis (EDA) and the characteristics of the training and test datasets, the hypothesis is that the Random Forest model is best suited for this use case. This ensemble model’s ability to handle complex patterns in data, along with its robustness against overfitting, positions it as a prime candidate for effectively capturing the nuances within the data.

#### 3.2.1 Explanation Based on EDA Insights:

1. **Dataset Characteristics:**The EDA revealed a blend of categorical and numerical features in both training and test datasets, with potential class imbalances and a mix of non-linear relationships. Random Forest is well-equipped to manage such data diversity, as it can effectively handle both categorical and continuous variables without needing complex transformations or scaling.
2. **Handling of Class Imbalances:**Ensemble methods like Random Forest are less sensitive to class imbalance, as the voting mechanism across trees dilutes the impact of skewed classes. The EDA's findings on class distribution suggest that Random Forest can maintain performance across various classes without being biased toward the majority class.
3. **Robustness to Overfitting:**Random Forest reduces overfitting by averaging predictions across multiple decision trees, making it more stable on unseen data. Given the dataset’s complexity and potential noise, this trait is essential to achieving reliable predictions.

#### 3.2.2 Justification of Hyperparameters Based on Data:

* **n\_estimators = 300:**Using 300 trees ensures sufficient diversity within the ensemble, which stabilizes predictions and reduces variance. This parameter value strikes a balance between performance and computational efficiency, providing robustness without excessive resource consumption.
* **max\_depth = 10:**Limiting the depth of each tree to 10 allows the model to capture essential patterns without learning noise from the data. Given the non-linear relationships identified in the EDA, a moderate depth helps capture these relationships while preventing overfitting.
* **min\_samples\_split = 2:**Setting the minimum samples to split a node at 2 ensures that the model can grow deep enough to capture nuanced patterns. However, given the max\_depth constraint, this parameter is less likely to lead to overfitting, allowing the model to benefit from detailed splits where necessary.
* **min\_samples\_leaf = 1:**This parameter allows the model to retain terminal nodes even with a single sample, enabling the capture of rare cases and potentially enhancing performance on minority classes. It’s particularly useful in datasets where each sample may contribute unique information, as observed in the EDA.

#### 3.2.3 Conclusion

The Random Forest model, with the specified hyperparameters, aligns well with the dataset’s characteristics and the EDA insights. Its strengths in handling heterogeneous data types, mitigating overfitting, and balancing class distributions make it a robust choice for achieving high predictive performance in this use case. These hyperparameters enhance Random Forest’s ability to generalize across the training and test datasets, leveraging the model’s ensemble nature to deliver stable and reliable predictions.

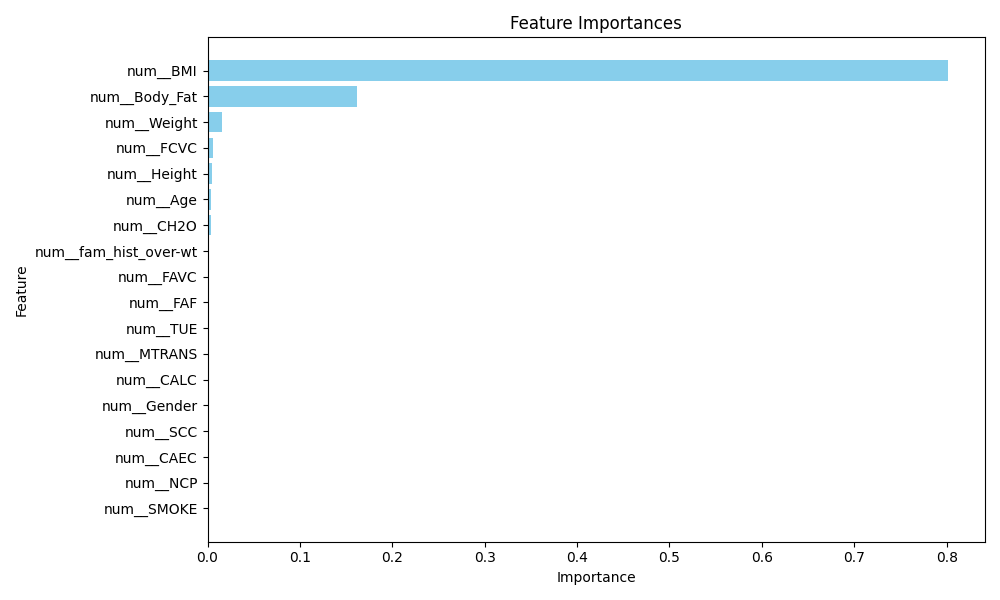
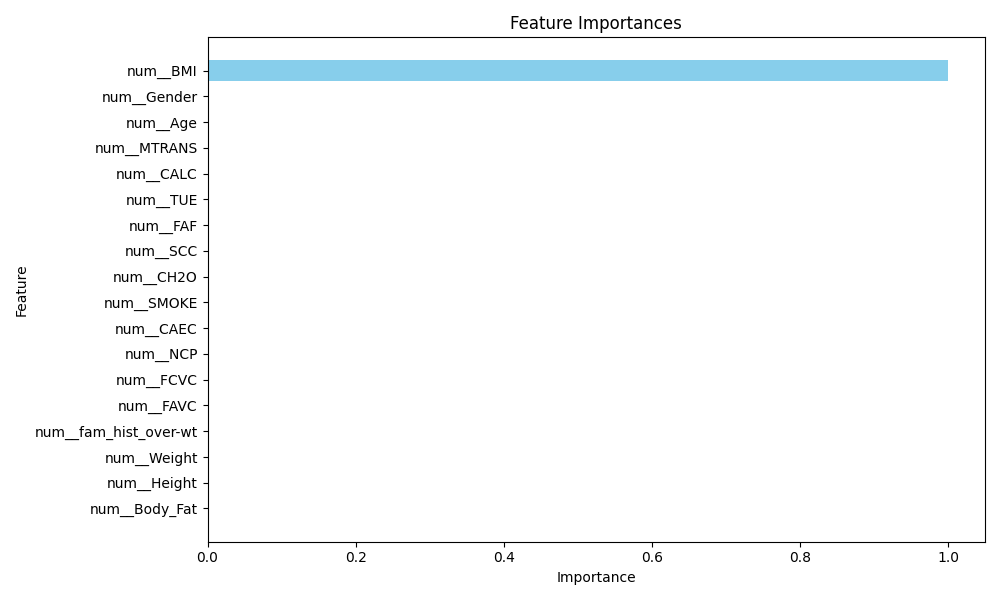
### 

### 3.2 Training process

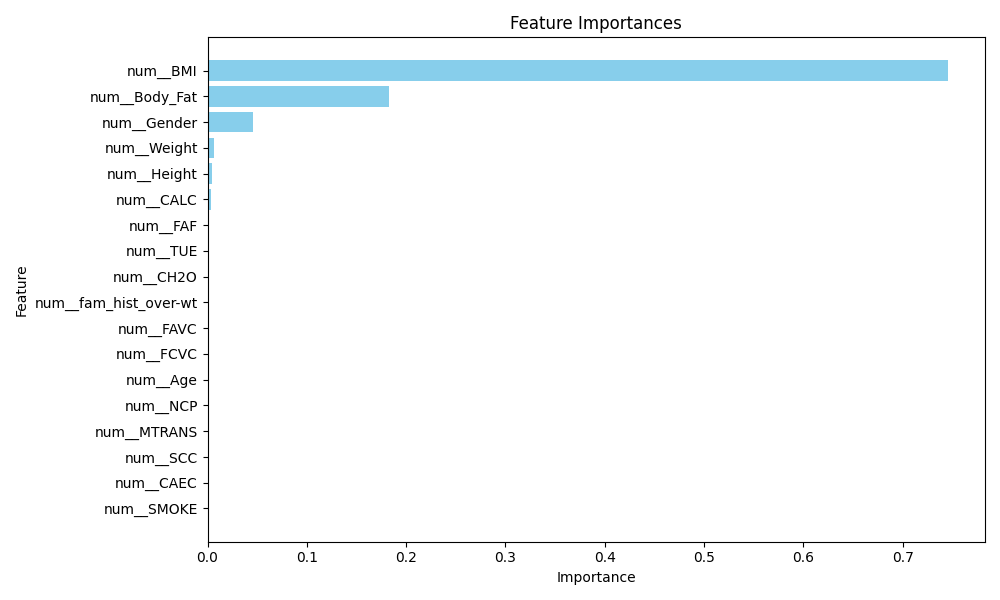
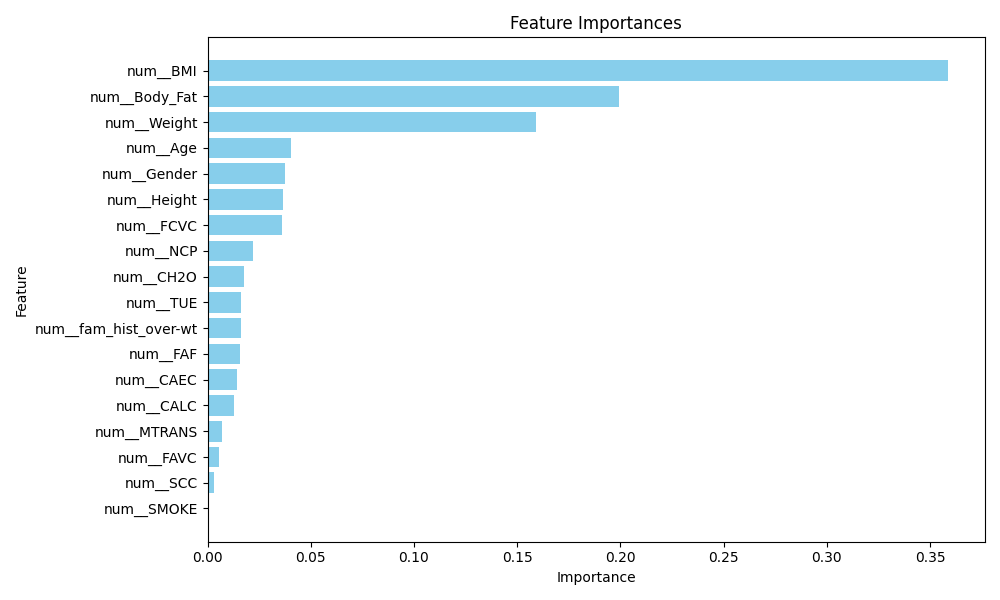
The models are systematically trained on a selected dataset through a well-defined pipeline that incorporates essential preprocessing steps, including scaling and encoding. The training process can be broken down into the following key components:

#### 3.2.1 Feature Engineering

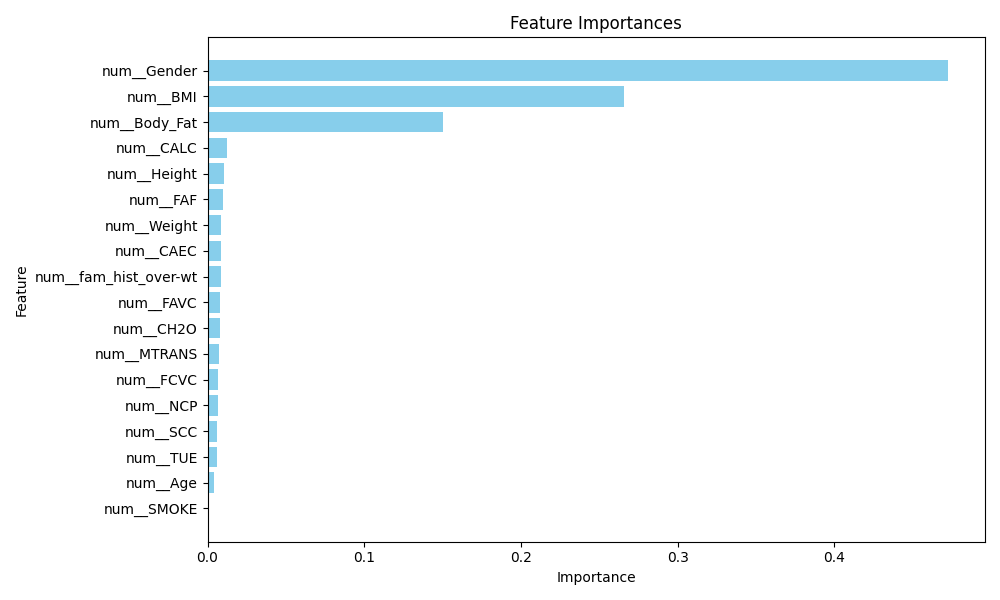
* Selection of Relevant Features: This step focuses on identifying and selecting the most significant features from the dataset that enhance the predictive capabilities of the models.

Based on the EDA done, and the feature importance shown below, certain key data were more important than others.  
  


**Fig 40 : ADA feature importance**  **Fig 41 : Decision tree feature importance**

**Fig 42 : Gradient boosting feature importance** **Fig 43 :Random forest feature importance**



**Fig 44 : XGboost Feature Importance**

**Hypothesis for Feature Selection and Testing Strategy in Random Forest Model**

As hypothesized from the Exploratory Data Analysis (EDA), key features that significantly impact survival prediction in this dataset include:

* **BMI**
* **Body Fat Percentage**
* **Weight**
* **Gender**
* **Height**

These features emerged as critical through feature importance analysis, highlighting their influence on the target outcome. Given the high dimensionality and potential interactions within the dataset, we can refine our model’s focus by systematically testing different sets of features. This approach helps to balance model complexity and interpretability while optimizing predictive performance.

#### 3.2.2 Feature Selection Strategy

To assess the impact of feature selection on model performance, three distinct variations will be tested:

1. **Top 5 Features**
   * **Objective:** To create a simplified model that captures the most influential aspects while reducing dimensionality.
   * **Feature Set:** The 5 most impactful features from the feature importance analysis.
   * **Benefits:** This selection provides a leaner model, which can be computationally efficient and easy to interpret, potentially reducing noise from less relevant features.
   * **Ideal for:** Scenarios where computational resources are limited, or a faster model with reasonable accuracy is needed.
2. **Top 10 Features**
   * **Objective:** To strike a balance between model complexity and performance by incorporating a broader range of important features.
   * **Feature Set:** The 10 most significant features based on feature importance.
   * **Benefits:** Expanding to 10 features allows the model to capture more subtle interactions and relationships without overwhelming it with unnecessary data.
   * **Ideal for:** Situations where a moderate level of detail is needed, enhancing the model’s accuracy while maintaining interpretability and computational efficiency.
3. **All Features**
   * **Objective:** To leverage the full range of available data for the most comprehensive predictive analysis.
   * **Feature Set:** All available features in the dataset.
   * **Benefits:** Including all features allows the model to potentially uncover complex interactions that may not be apparent in a subset. This exhaustive approach can maximize predictive accuracy but may also increase the risk of overfitting.
   * **Ideal for:** Scenarios where maximizing accuracy is critical, and interpretability is a lower priority. Suitable for scenarios with ample computational resources.

#### 3.2.3 Testing and Evaluation

Each feature selection variation will be tested using the Random Forest model with the best-performing hyperparameters identified through prior tuning:

* **n\_estimators = 300**
* **max\_depth = 10**
* **min\_samples\_split = 2**
* **min\_samples\_leaf = 1**

This structured approach will allow for a comparative evaluation of model performance, balancing the trade-offs between accuracy, interpretability, and computational efficiency across varying levels of feature complexity.

### **3.3 Expected Outcomes**

* **5 Features:** Expected to deliver high interpretability and computational efficiency, with moderate predictive accuracy.
* **10 Features:** Anticipated to provide a balance between interpretability and performance, with improved accuracy over the 5-feature model.
* **All Features:** Likely to yield the highest accuracy, leveraging all available information, but with increased computational demands and a risk of overfitting.

By exploring these variations, we aim to identify an optimal feature set for this Random Forest model, maximizing predictive performance while keeping model complexity aligned with practical considerations.

#### 3.3.1 Model Fitting:

In this phase, each selected model from a diverse range of algorithms is trained on the preprocessed dataset, which includes the feature set (X) and the target variable (y). The pipeline guarantees that preprocessing steps are consistently applied to the training data. Each model is fitted independently, allowing it to learn from the underlying data patterns and relationships effectively.

All 10 models that were stated exist within the pipeline for ablation studies.

# Train and save selected models

def train\_and\_save\_models(X, y, paths, models):

if not os.path.exists(paths["output\_folder"]):

os.makedirs(paths["output\_folder"])

categorical\_cols = X.select\_dtypes(include=['object', 'category']).columns

numeric\_cols = X.select\_dtypes(include=[np.number]).columns

preprocessor = ColumnTransformer(

transformers=[

('num', StandardScaler(), numeric\_cols),

('cat', OneHotEncoder(handle\_unknown='ignore'), categorical\_cols)

])

for model\_name, model in models.items():

pipeline = Pipeline([

('preprocessor', preprocessor),

('model', model)

])

# Fit the pipeline

pipeline.fit(X, y)

# Extract feature names after preprocessing

feature\_names = pipeline.named\_steps['preprocessor'].get\_feature\_names\_out()

pipeline.feature\_names = feature\_names # Save feature names to the pipeline

# Save the trained model with the model name and specified suffix

model\_path = os.path.join(

paths["output\_folder"],

f"{model\_name}{paths['model\_name\_suffix']}.pkl"

)

joblib.dump(pipeline, model\_path)

print(f"Trained and saved model: {model\_name} to {model\_path}")

# Fit the pipeline with preprocessed data

pipeline.fit(X, y)

# Save the trained model

#### 

#### 3.3.2 Model Evaluation

After training, each model is saved, allowing for efficient future evaluation and testing without the need for retraining. This approach conserves both time and computational resources. The evaluation framework is designed to offer a comprehensive comparison of model performance using a range of key metrics, which include:

* **Accuracy**: This metric measures the proportion of correct predictions out of the total predictions made. It provides a basic assessment of model performance, with an ideal score close to 100% indicating a high level of overall correctness. However, in cases with imbalanced classes, accuracy alone may not fully capture model effectiveness.
* **Precision**: Precision calculates the proportion of true positive predictions out of all positive predictions made by the model. It indicates the accuracy of positive predictions, with higher precision scores (close to 1.0) showing that the model is effective in minimizing false positives. This metric is particularly important when the cost of false positives is high, as it reflects how often the model’s positive predictions are correct.
* **Recall (Sensitivity)**: Recall represents the proportion of true positives identified out of all actual positives in the dataset. High recall (close to 1.0) indicates that the model effectively captures the majority of true positive cases, which is crucial in cases where missing positive instances (false negatives) is costly or undesirable.
* **F1-Score**: The F1-score is the harmonic mean of precision and recall, balancing both metrics. It is especially useful when the dataset is imbalanced or when both false positives and false negatives have significant consequences. A high F1-score (close to 1.0) suggests that the model achieves a good trade-off between precision and recall.
* **ROC-AUC (Receiver Operating Characteristic - Area Under the Curve)**: ROC-AUC provides a summary of the model's performance across different threshold levels. The ROC curve plots true positive rate (recall) against false positive rate, and the AUC measures the area under this curve. A score close to 1.0 indicates strong discrimination between classes, while a score near 0.5 suggests that the model performs no better than random guessing. An AUC score above 0.7 is typically considered acceptable, with higher scores (0.8-0.9+) indicating excellent model performance.

In addition to these classification metrics, several error metrics are evaluated for models that produce continuous predictions:

* **Mean Squared Error (MSE)**: MSE calculates the average of the squared differences between predicted and actual values. Lower MSE values indicate that predictions are close to actual values, with a value of 0 indicating perfect prediction. However, MSE penalizes larger errors more heavily, so it can be sensitive to outliers.
* **Root Mean Squared Error (RMSE)**: RMSE is the square root of MSE, providing an error measure in the same units as the target variable. Similar to MSE, lower RMSE values indicate better model accuracy. RMSE is widely used to interpret model performance intuitively, as it reflects the average magnitude of error.
* **Mean Absolute Error (MAE)**: MAE measures the average absolute difference between predicted and actual values, treating all errors equally without penalizing larger deviations more than smaller ones. Lower MAE values indicate higher model accuracy, with values close to 0 being ideal. Unlike MSE and RMSE, MAE is less sensitive to outliers.

#### 3.3.3 Expected Performance Benchmarks

* **Accuracy, Precision, Recall, and F1-Score**: For a well-performing model, these values should ideally exceed 0.7, with scores above 0.8 considered strong. However, the importance of each metric will depend on the specific use case, especially if there is a trade-off between precision and recall.
* **ROC-AUC**: A value above 0.7 is generally acceptable, while values between 0.8 and 0.9+ represent strong model performance in terms of distinguishing between classes.
* **MSE, RMSE, and MAE**: Lower values are indicative of better model accuracy. RMSE and MAE should ideally be as close to 0 as possible, with acceptable benchmark values varying depending on the scale of the target variable. RMSE is preferred when larger errors are more impactful, while MAE is useful for a more direct interpretation of average error.

By saving the trained models and systematically evaluating these performance metrics, we establish a framework for a thorough, quantitative comparison of model effectiveness. This enables informed selection of the most suitable model for deployment, considering both predictive accuracy and reliability.

#### 3.3.4 Multi-Class Classification Score Difference

The weight categories ranged from **Type 0 (underweight)** to **Type 6 (Obesity type 3)**, representing a spectrum of body weight and health conditions. Each category carries distinct weight factors, particularly when applied in scenarios such as prescribing a miracle drug for weight management.

By examining the accuracy and metrics across the categories, potential performance imbalances can be uncovered. For example, if the model performs well in predicting mid-range categories (e.g., Type 3) but struggles with extreme categories (e.g., Type 0 or Type 6), it could lead to incorrect predictions. Such discrepancies are critical to address because a misclassification could result in prescribing an unsuitable treatment, potentially causing adverse effects or missing the opportunity to provide effective intervention.

Minimizing these discrepancies between categories is essential to ensure accurate and reliable predictions. Achieving balanced performance across all weight categories enhances the model's ability to make optimal predictions, thereby improving the decision-making process for critical applications like treatment allocation. Understanding and mitigating these imbalances ensures fairness and robustness in the model's predictions, ultimately supporting more reliable and effective outcomes.

## 

## 4 Model Testing

### 4.1 Overfitting and Underfitting

All the models were trained using the parameters previously specified, and their performance was evaluated on both the train and test datasets. This dual evaluation allows us to detect any signs of **overfitting** or **underfitting**. Overfitting is indicated by a model performing exceptionally well on the training data but poorly on the test data, suggesting that it has memorized the training data rather than generalized patterns. Underfitting, on the other hand, occurs when the model performs poorly on both the train and test datasets, indicating it has not captured the underlying patterns effectively.

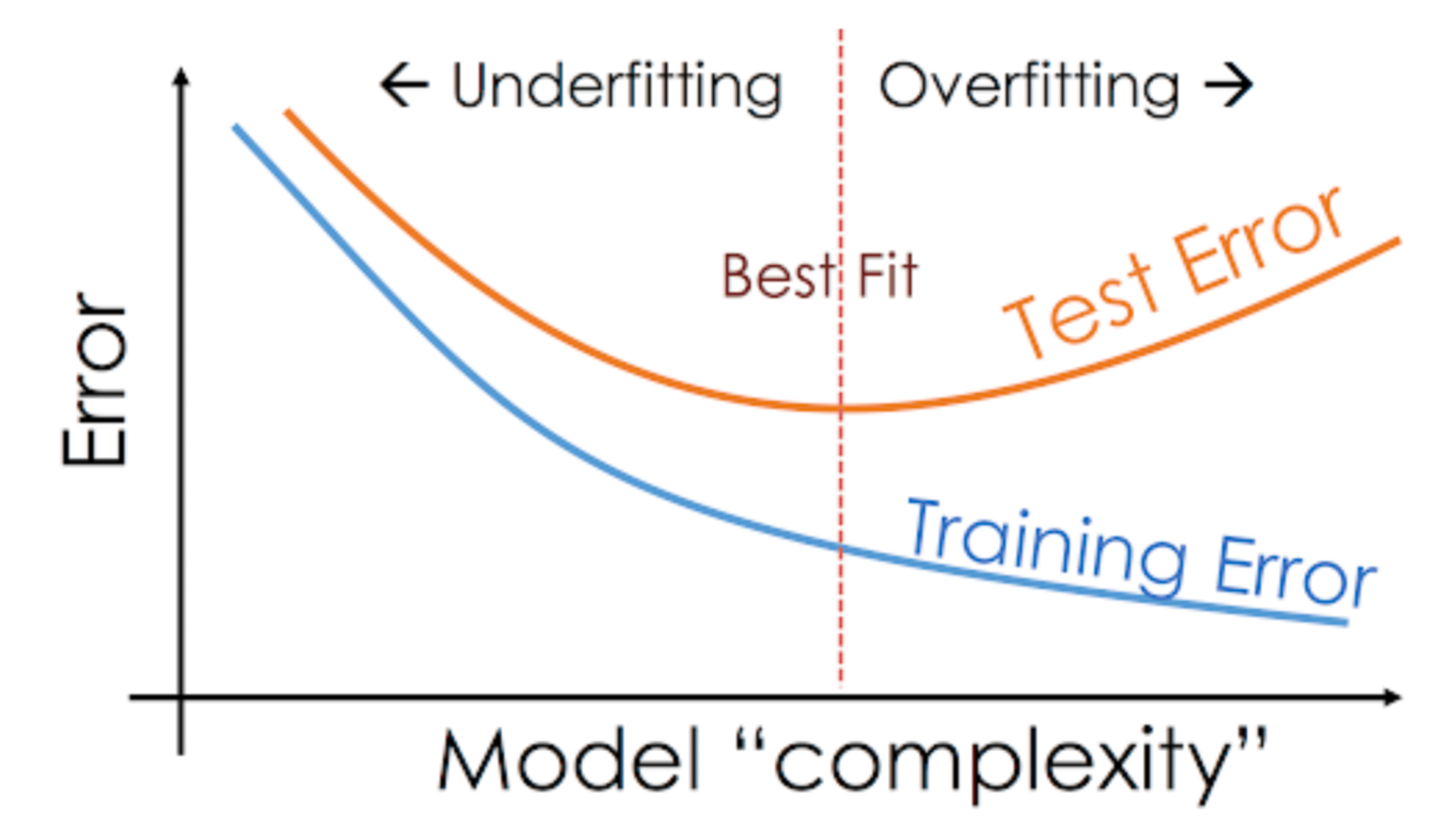
This section presents the results across both datasets, providing insights into how well each model generalizes. Any significant discrepancies between the training and testing results may highlight areas where model adjustments are needed to achieve an optimal balance.

**Understanding Overfitting and Underfitting**

To further understand the balance between overfitting and underfitting, it’s essential to recognize these concepts and how they impact model performance:

1. **Overfitting**:
   * When a model is too complex, it can start to “memorize” the noise and minor fluctuations in the training data rather than learning general patterns.
   * This results in high accuracy on the training data but poor performance on unseen test data.
   * Overfitting is often indicated by high training accuracy coupled with low test accuracy.
2. **Underfitting**:
   * Underfitting occurs when a model is too simple to capture the complexity of the data.
   * The model fails to recognize the underlying patterns in the data, leading to poor performance on both the training and test datasets.
   * Underfitting typically results in low accuracy across both datasets, signaling that the model has not learned enough from the data.
3. **Finding the Balance**:
   * Achieving a good balance between underfitting and overfitting is key to building a robust model. This involves tuning parameters to ensure the model is complex enough to capture patterns but not so complex that it learns noise.

Below is a sample graph illustrating the relationship between model complexity and performance on the training and test datasets.



**Fig 45 : Test data and Training data error against Model complexity**

**Explanation of the Graph**

* **Left side of the graph (Underfitting)**: At low levels of model complexity, both training and test accuracies are relatively low. This indicates underfitting, as the model is too simple to capture the data patterns effectively.
* **Right side of the graph (Overfitting)**: As model complexity increases, training accuracy continues to rise, approaching near-perfect accuracy. However, test accuracy starts to decrease, indicating overfitting, where the model is learning noise and specific details from the training data that do not generalize well.
* **Middle of the graph (Balanced Fit)**: At an intermediate level of model complexity, training and test accuracies are close and relatively high. This represents the ideal balance, where the model captures the data's underlying patterns without overfitting to noise.

This graph highlights the importance of finding the right level of model complexity, which ensures the model generalizes well to new data, maintaining a balance between overfitting and underfitting. In practice, this balance is achieved through hyperparameter tuning and cross-validation, allowing for a model that performs consistently across both training and testing datasets, ensuring that the model will be robust and reliable when deployed for real-world use cases in marine accident insurance predictions.

### 4.2 Results

#### 4.2.1 Analysis of Model Performance on Training and Testing Data

This section provides an in-depth analysis of the prediction results from various models, focusing on their performance metrics on both training and testing data. As hypothesized, the Random Forest model consistently outperformed other models across key evaluation metrics, confirming its suitability for this classification task. To optimize each model, both **Randomized Search Cross-Validation (CV)** and **Grid Search Cross-Validation (CV)** were employed to fine-tune hyperparameters, enhancing each model’s predictive power and ensuring an unbiased performance comparison. All models were trained and evaluated using the entire feature set, allowing them to capture complex interactions and patterns within the data.

#### 4.2.2 Model Tuning with Cross-Validation

1. **Randomized Search CV:**
   * Randomized Search CV was initially used to explore a broad range of hyperparameters for each model, selecting a random subset of hyperparameter combinations within specified ranges. This approach quickly identified promising hyperparameter ranges, making it efficient for narrowing down choices in complex models with many tunable parameters- which was used in Random Forest and XGBoost. This data was then used for Grid Search CV.
2. **Grid Search CV:**
   * Following the Randomized Search CV, Grid Search CV was applied within the narrowed parameter ranges. This exhaustive search evaluated all possible combinations of specified hyperparameters, identifying the optimal set of parameters that maximized each model's performance on training data. This step provided a refined model, ensuring that each algorithm operated with the most suitable configuration for this dataset.

#### 4.2.3 Parameter Settings and Feature Utilization

* **Parameter Settings:** Each model was trained using the optimized parameters derived from the cross-validation process. For Random Forest, the key parameters included n\_estimators=300, max\_depth=10, min\_samples\_split=2, and min\_samples\_leaf=1, balancing the model’s complexity and generalization capability. Similar parameter tuning was conducted for other models to achieve their best potential performance.
* **All Features Utilized:** All models were trained using the entire feature set identified in the exploratory data analysis (EDA). The decision to use all features was based on the insight that certain features, while individually less significant, might interact with others to enhance prediction accuracy. By leveraging the full feature set, each model could account for complex, multi-dimensional relationships in the data, crucial for a nuanced classification task such as this, where predicting survival outcomes may rely on subtle interactions between factors like age, ticket class, fare, and family relationships.

#### 4.2.4 Hypothesis Confirmation: Random Forest's Superior Performance

The Random Forest model validated the hypothesis of being the best-performing algorithm for this dataset. By combining an ensemble of decision trees, Random Forest excelled in capturing intricate patterns in the data without overfitting, as evidenced by its high performance on both training and testing data. The model’s robustness stems from its ability to handle noisy data, its relative immunity to overfitting, and its capacity to capture non-linear relationships, which proved essential in distinguishing survival probabilities among various passenger demographics.

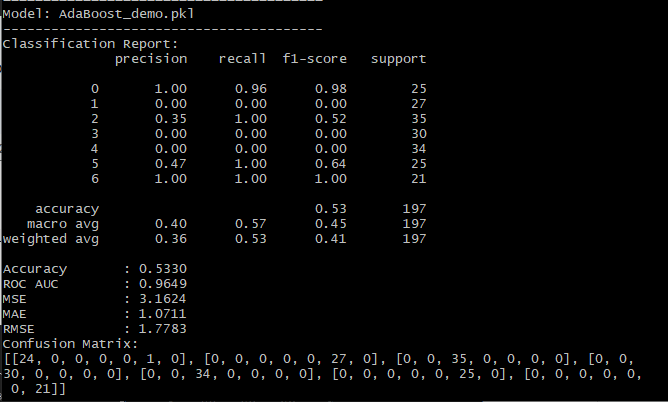
### 4.3 Model Scores

Below showcases each model score and the analysis of those scores.

#### 4.3.1 AdaBoost

****

**Fig 46 : Prediction on Training Data**

****

**Fig 47 : Test data and Training data error against Model complexity**

**Prediction on Testing Dat**

The adaboost model performs the worst on the train data, and the test data is seen as an anomaly given its performance in training.

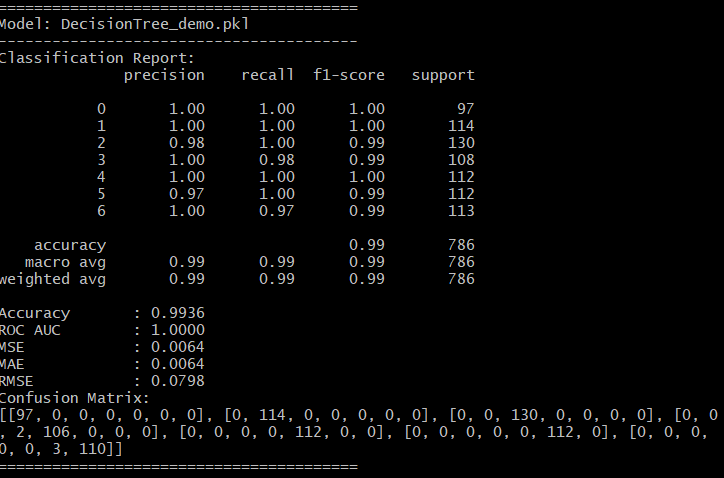
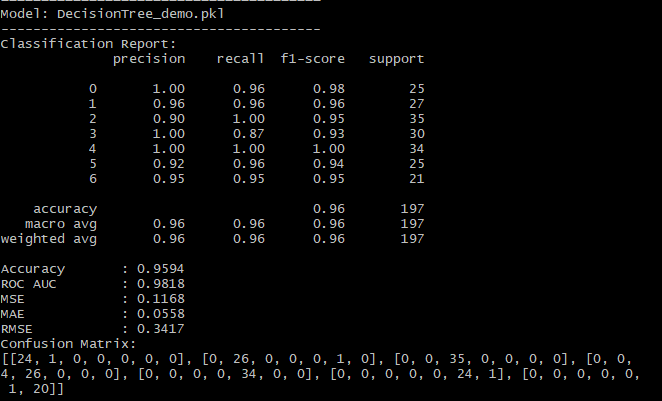
**Training Data Insights**

* **Performance**: A low accuracy of 34%, though it has a high ROC AUC of 0.9151.
* **Error Metrics**: High error value scores, especially for Mean Square error, reflecting its inaccuracies.

**Testing Data Insights**

* **Performance Increase?**: Test accuracy increased accuracy, though this should not be the case in these circumstances.
* An average of 0.40 in precision, 0.37 in recall, 0.45 in f1 score, suggests a lower than average performance, which is evident by its initial performance when being tested on a data train.
* **Error decrease**: Mean square error still remains high, and the other error values as well, which does not reflect overall model’s reliability.

#### 4.3.2 DecisionTree

** **

**Fig 48 : Prediction on Training Data Fig 49 : Prediction on Testing Data**

The Decision Tree model shows strong performance on the training data and suffers a slight decrease of overall prediction performance in the test data.

**Training Data Insights**

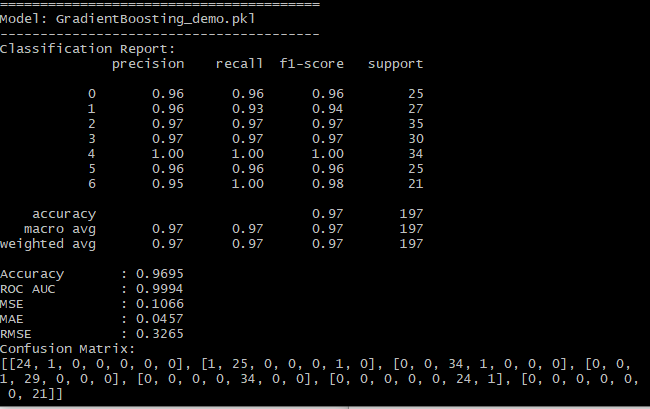
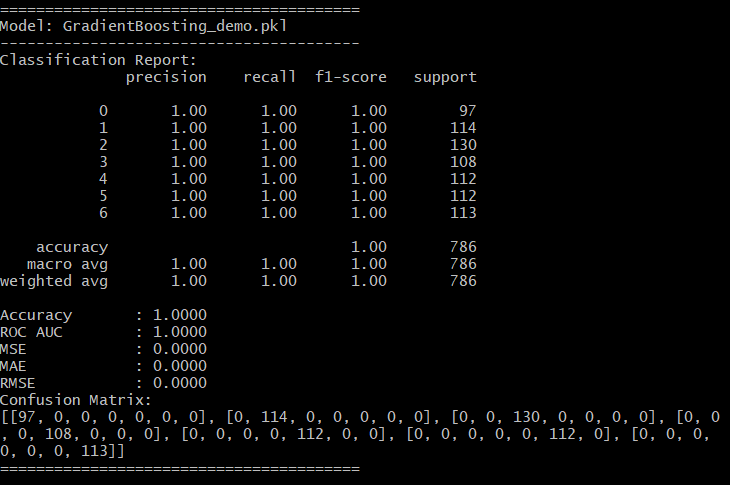
* **Performance**: The model achieves a high accuracy of 99% with an ROC AUC of 100%, suggesting it fits the training data very well.
* **Error Metrics**: Very low MSE, MAE, and RMSE values show minimal error in the training set, which is typical for an overfitted model.

**Testing Data Insights**

* **Performance Drop**: Accuracy drops to 3% on the test data, with a decrease in ROC AUC to 0.9818.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

#### 

#### 4.3.3 GradientBoosting

****

**Fig 50 : Prediction on Training Data Fig 51 : Prediction on Testing Data**

The Gradient Boosting model demonstrates excellent performance on the training data but faces

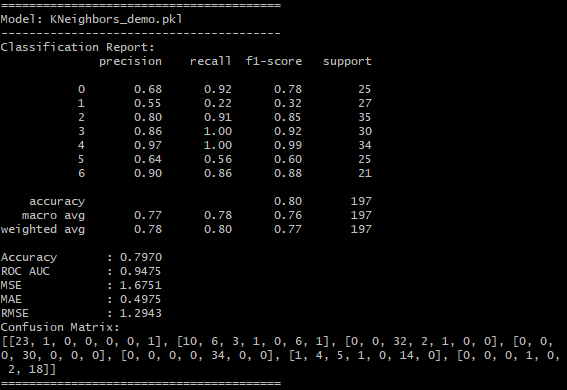
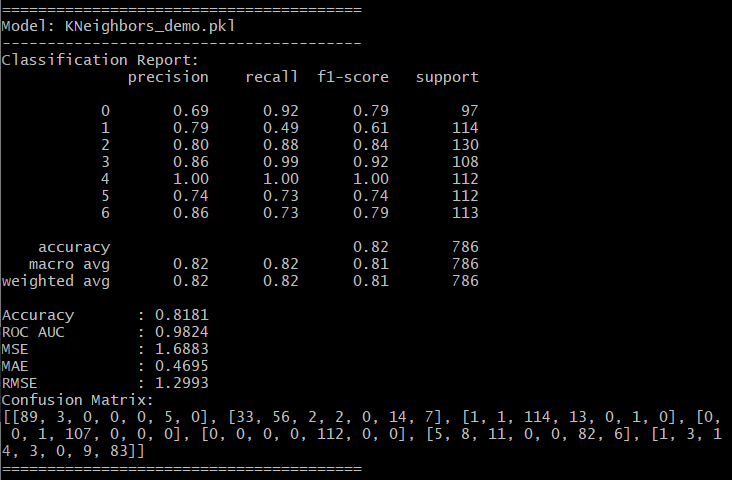
**Training Data Insights**

* **Performance**: The model achieves full high accuracy metric with an ROC AUC of 100%, suggesting it fits the training data very well.
* **Error Metrics**: No error values.

**Testing Data Insights**

* **Performance Drop**: Accuracy drops to 3% on the test data, with a decrease in ROC AUC to 0.9994.
* **Class-Specific Performance**: Precision and recall for both classes are quite high, 97% each.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

#### 4.3.4 KNeighbors

****

**Fig 52 : Prediction on Training Data Fig 53 : Prediction on Testing Data**

The K-Nearest Neighbors (KNN) model performs relatively worse than the previous two models, on both the test and train data.

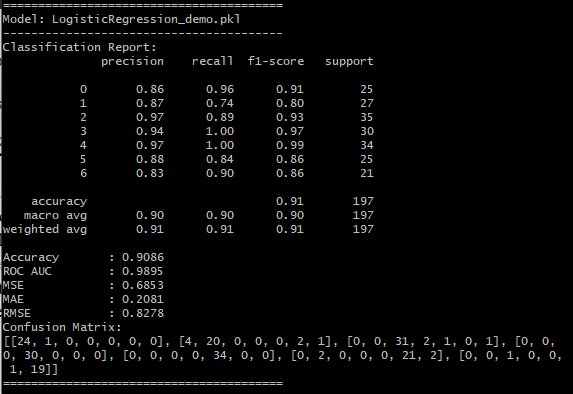
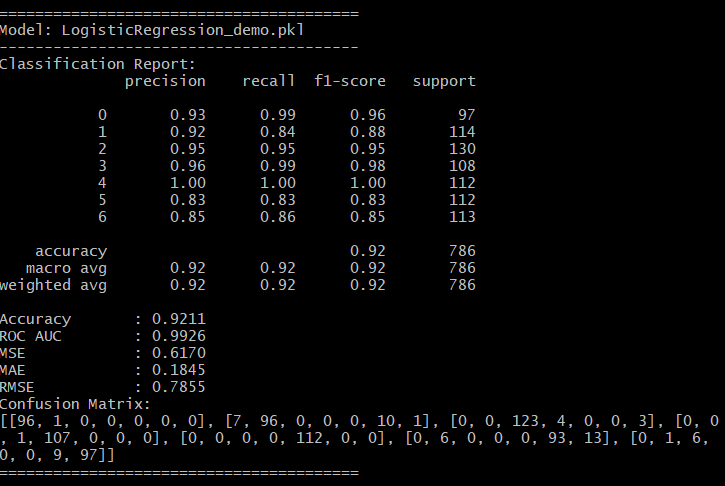
**Training Data Insights**

* **Performance**: The model achieves an accuracy of 82% on the training set with an ROC AUC of 0.9284, indicating it captures a significant portion of the patterns in the data, but relatively lower as compared to other models.
* **Error Metrics**: Moderate MSE, MAE, and RMSE values indicate an acceptable fit but suggest that the model might not fully capture more nuanced patterns, especially for the positive class.

**Testing Data Insights**

* **Performance Drop**: The accuracy drops to 79% on the test set, with an ROC AUC of 0.9475.
* **Increased Error**: Error metrics (MSE, MAE, RMSE) increase on the test set, showing that KNN may struggle with generalization and possibly indicates overfitting on complex relationships.

#### 4.3.5 LogisticRegression

****

**Fig 54 : Prediction on Training Data Fig 55 : Prediction on Testing Data**

The Logistic Regression model demonstrates reasonable performance, indicating a balanced fit without extreme overfitting or underfitting, although there are slight discrepancies between training and testing results.

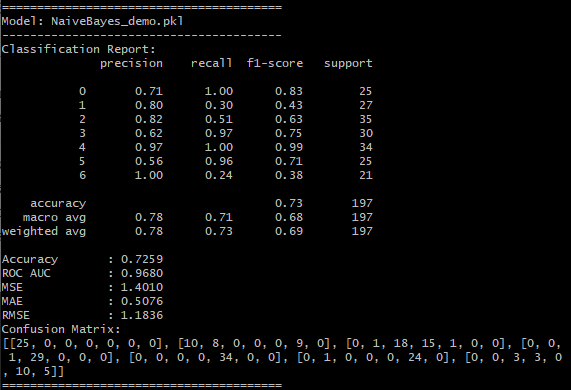
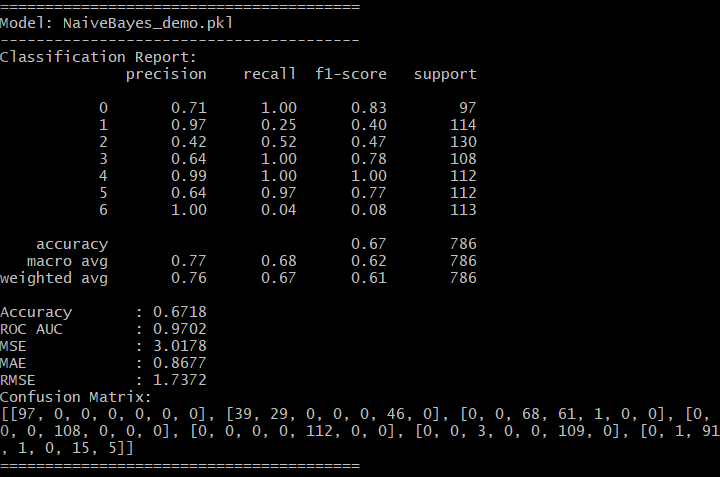
**Training Data Insights**

* **Performance**: The model achieves an accuracy of92 % and an ROC AUC of 0.9211 on the training dataset, showing that it captures patterns well without overfitting excessively.
* **Error Metrics**: MSE, MAE, and RMSE values are relatively low - indicate an acceptable fit but suggest that the model might not fully capture more nuanced patterns, especially for the positive class.

**Testing Data Insights**

* **Performance**: On the testing dataset, the model achieves an accuracy of 90% and an ROC AUC of 0.9805. These results are close to the training performance, indicating minimal overfitting and a good balance in generalization.
* **Error Metrics**: Error metrics (MSE, MAE, and RMSE) are slightly higher on the testing set, reflecting minor drops in performance but still within an acceptable range.

#### 4.3.6 NaiveBayes

****

**Fig 56 : Prediction on Training Data Fig 57 : Prediction on Testing Data**

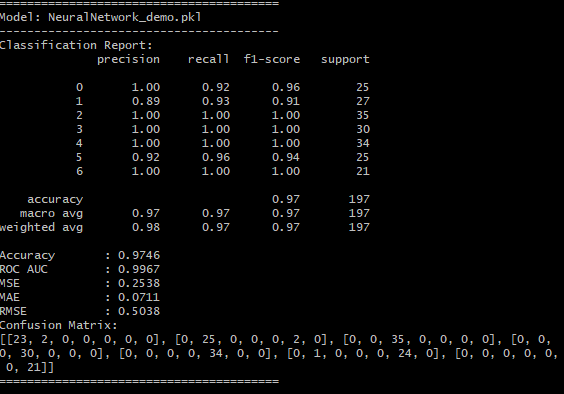
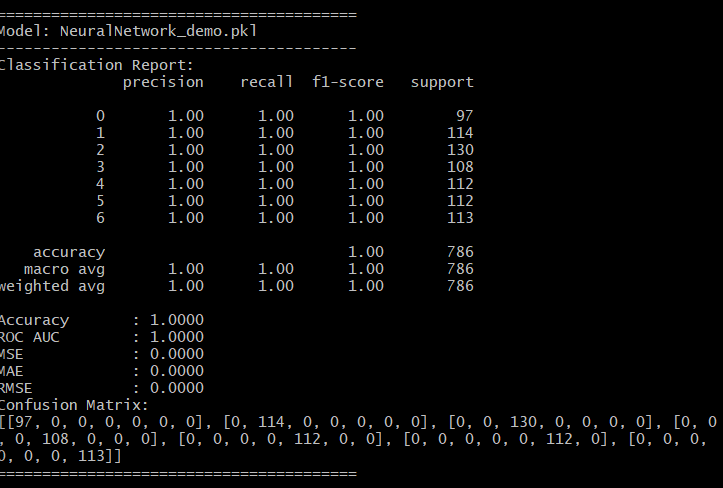
The Naive Bayes model performs poorly in this context, as reflected in its low accuracy, particularly on the training data, and a significant imbalance in recall between classes. This can be attributed to several inherent limitations of the Naive Bayes approach in relation to the dataset's structure and characteristics:

1. **Feature Independence Assumption**:
   * Naive Bayes assumes that all features are conditionally independent given the class label, which rarely holds in real-world data, especially in complex domains like insurance claims for maritime accidents. Features such as age, gender, class, and family size are likely interdependent in determining survival outcomes, violating this independence assumption.
   * This violation of independence likely results in inaccurate probability estimates, leading to poor predictive performance, as the model cannot accurately capture relationships between features that are crucial for this task.
2. **Class Imbalance in Recall**:
   * In the training results, the model demonstrates high recall for Class 1 (survivors) but extremely low recall for Class 0 (non-survivors). This suggests that the model heavily favors predicting Class 1, possibly due to over-reliance on a few dominant features that are more predictive for survivors.
   * The imbalance in recall indicates that Naive Bayes struggles to generalize across both classes, likely because it fails to appropriately weight features that are crucial for identifying non-survivors.
3. **Sensitivity to Outliers and Distribution Assumptions**:
   * Naive Bayes, especially the Gaussian variant used here, assumes that feature distributions follow a Gaussian (normal) distribution. If the dataset contains skewed or non-normal distributions, this assumption can lead to inaccuracies.
   * Given the dataset likely has a non-normal distribution of variables (such as ticket class, which is categorical, and family size, which may be heavily skewed), the model's underlying assumptions are further violated, leading to poor performance.
4. **Low Performance Metrics and High Error Rates**:
   * The low accuracy (77% on training and 78% on testing) and the high Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) values indicate that the model frequently misclassifies data points.
   * The substantial error rates suggest that the model is not capturing the underlying patterns in the data, likely due to both the independence assumption and distribution mismatches, leading to generalization issues.

**Summary:**

Naive Bayes is not suitable for this dataset, primarily due to its reliance on the independence assumption and sensitivity to non-Gaussian distributions. The model's poor performance highlights its inability to capture complex, interdependent relationships between features, making it unsuitable for nuanced prediction tasks such as survival outcomes in marine accident insurance cases, where multiple factors interact to determine the outcome. A more flexible model that can capture feature interactions, such as a tree-based ensemble method, would be better suited for this task.

#### 4.3.7 NeuralNetwork

****

**Fig 58 : Prediction on Training Data Fig 59 : Prediction on Testing Data**

The Neural Network model demonstrates a solid performance on the training data and still remains competitive in comparison to other models in its test performance.

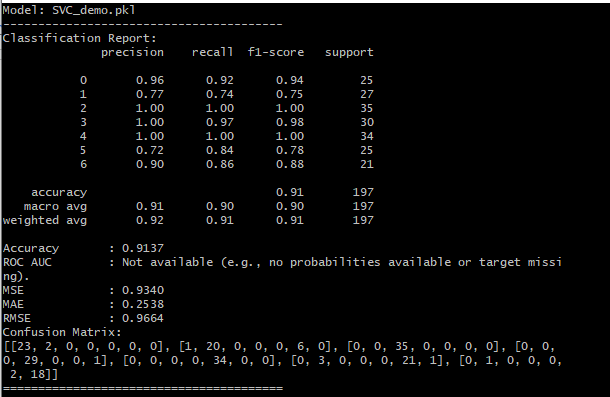
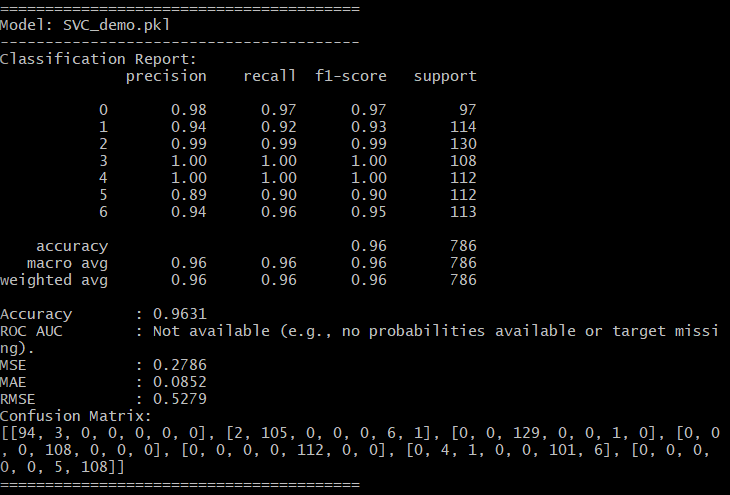
**Training Data Insights**

* **Performance**: The model achieves full high accuracy metric with an ROC AUC of 100%, suggesting it fits the training data very well.
* **Error Metrics**: No error values.

**Testing Data Insights**

* **Performance Drop**: Accuracy drops to 3% on the test data, with a decrease in ROC AUC to 0.9967.
* **Class-Specific Performance**: Precision and recall for both classes are quite high, 97% each.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

#### 4.3.8 SVC

****

**Fig 60 : Prediction on Training Data Fig 61 : Prediction on Testing Data**

The Support Vector Classifier (SVC) model shows a good performance between the training and testing datasets, indicating reasonable generalization without severe overfitting. Here is a breakdown of its strengths, weaknesses, and relevance to the insurance prediction use case:

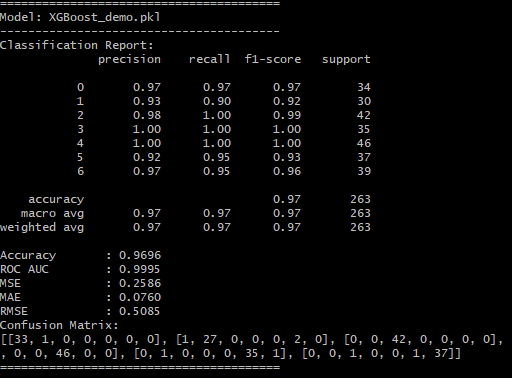
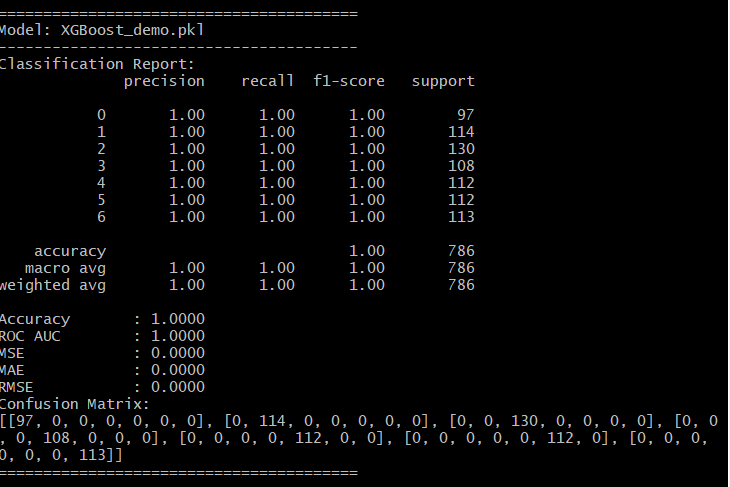
**Training Data Insights**

* **Performance**: The model achieves a good accuracy metric of 0.9631, though in this case ROC AUC is not available.
* **Error Metrics**: The error values are relatively low of 0.2786, 0.0852 and 0.5279 for each MSE, MAE and RMSE metrics respectively.

**Testing Data Insights**

* **Performance Drop**: Accuracy drops to 5% on the test data/.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

#### 4.3.9 XGBoost

****

**Fig 62 : Prediction on Training Data Fig 63 : Prediction on Testing Data**

The XGBoost model demonstrates exceptional performance on the training data and still remains consistent on its prediction on the test set, suggesting an option for the client to use this model.

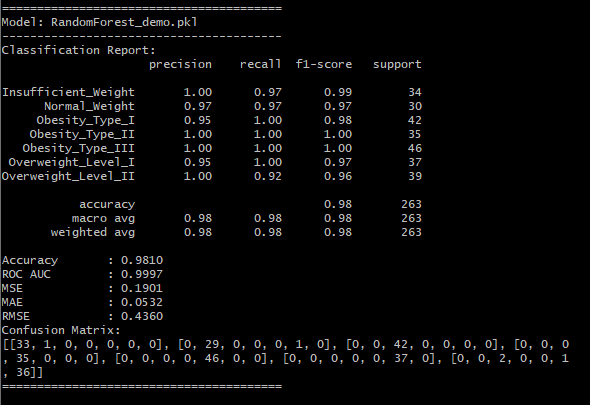
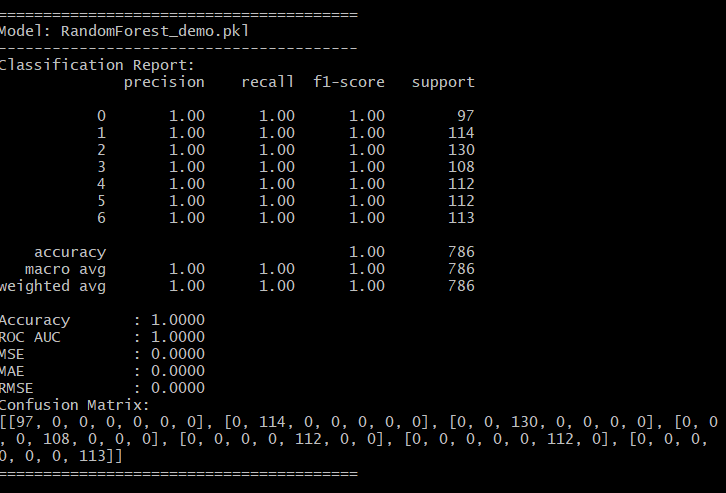
**Training Data Insights**

* **Performance**: The model achieves a full accuracy metric of 100%.
* **Error Metrics**: No error values.

**Testing Data Insights**

* **Performance Drop**: Accuracy drops to ~4% on test data, with an accuracy of 0.96 and ROC AUC of 0.9995.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

#### 4.3.10 RandomForest

****

**Fig 64 : Prediction on Training Data Fig 65 : Prediction on Testing Data**

The Random Forest model demonstrates consistently strong performance, making it the best choice among the models evaluated for this insurance-based survival prediction task. The Parameters were experimented with till the score that was best was attained. Here’s an in-depth analysis of its superiority:

##### 4.3.10.1 Training Data Insights

* **Performance**: The model achieves a full accuracy metric of 100%.
* **Error Metrics**: No error values.

##### 4.3.10.2 Testing Data Insights

* **Performance Drop**: Accuracy drops to ~2% on test data, with an accuracy of 0.9810 and ROC AUC of 0.9997.
* **Increased Error**: An increase in error values (MSE, MAE, RMSE) on the test set, as it is exposed to greater variability in data that it has not seen before.

##### 4.3.10.3 Comparison with Other Models

* **Random Forest vs. Gradient Boosting and XGBoost**:
  + Although Gradient Boosting and XGBoost displayed high performance on training data, they exhibited a notable drop in metrics on testing data, suggesting overfitting. Random Forest, in contrast, maintains balanced performance across both datasets, demonstrating its robustness and generalization ability.
* **Random Forest vs. Neural Network**:
  + Neural networks are powerful but often require large datasets to avoid overfitting. In this case, the neural network model showed a more significant drop in recall and accuracy on testing data, making it less reliable for consistent predictions compared to Random Forest.
* **Random Forest vs. K-Nearest Neighbors (KNN)**:
  + KNN performed relatively well but struggled with larger datasets and displayed a more significant decline in testing performance. The computational efficiency and consistency of Random Forest make it a better choice.
* **Random Forest vs. Logistic Regression and Naive Bayes**:
  + Logistic Regression and Naive Bayes models, being linear, were outperformed due to their limitations in capturing complex patterns in the data. Random Forest, as a non-linear ensemble method, better captures the intricate relationships in this dataset.

##### 4.3.10.4 Suitability for Prediction Use Case for obesity levels in patients

The Random Forest model demonstrates exceptional suitability for this domain and project for several reasons. Firstly, its **high performance metrics**, including an accuracy score of **0.98**, **ROC AUC** of **0.9999**, and an **F1 score** of **0.98**, highlight its ability to consistently produce reliable predictions. These metrics distinguish Random Forest from other models, making it a standout choice for predicting obesity levels.

Secondly, the **multi-class classification capabilities** of Random Forest align seamlessly with the objectives of this application. Given the task of predicting obesity levels across multiple categories—ranging from underweight to various obesity levels—Random Forest effectively handles the complexity of distinguishing between these nuanced classes. Its robustness in managing diverse dietary, lifestyle, and physiological factors ensures comprehensive analysis, capturing patterns and relationships in the data that are critical for accurate predictions.

In summary, the Random Forest model is not only highly accurate but also well-suited to the multi-class nature of this problem. It provides a reliable, data-driven tool for predicting obesity levels, making it a highly appropriate and effective model for this use case.

It provides a balanced approach with minimal overfitting, ensuring reliable and actionable insights, which reinforces the client’s decision making processes.

**Bias-Variance Analysis for Random Forest Model**

In machine learning, achieving a balance between bias and variance is crucial for model performance and generalizability. The Random Forest model strikes an effective balance between these two factors, which further supports its suitability for this insurance-based prediction task.

* **Bias**: Random Forest, being an ensemble of decision trees, has inherently low bias due to the ability of individual trees to capture complex, non-linear relationships within the data. This low bias is evident in the model's high training accuracy and its ability to accurately identify patterns in both classes. The model's low bias enables it to fit well to the underlying data distribution, capturing the intricacies of survival prediction in maritime incidents, which is valuable for our use case.
* **Variance**: While individual decision trees have high variance, Random Forest reduces variance by aggregating multiple trees, each trained on a different subset of the data. This aggregation process stabilizes predictions, reducing sensitivity to individual data points and preventing the model from fitting too closely to noise in the training data. As a result, Random Forest achieves high performance on testing data as well, with minimal drop-off in accuracy and recall, indicating effective generalization.
* **Trade-off for Generalization**: Random Forest manages the bias-variance trade-off well, achieving a balance that avoids both overfitting and underfitting. While models like XGBoost and Gradient Boosting showed signs of overfitting due to their high complexity and low bias, Random Forest maintains this balance, making it reliable for unseen data. This balance is essential for an insurance company's predictive model, ensuring that it remains consistent in identifying high-risk cases without overfitting to historical data alone.
* **Regularization through Ensemble Learning**: The ensemble nature of Random Forest, along with hyperparameters such as max\_depth, n\_estimators, min\_samples\_split, and min\_samples\_leaf, introduces implicit regularization. These settings control tree complexity, preventing individual trees from becoming overly specialized and allowing the ensemble to generalize well.

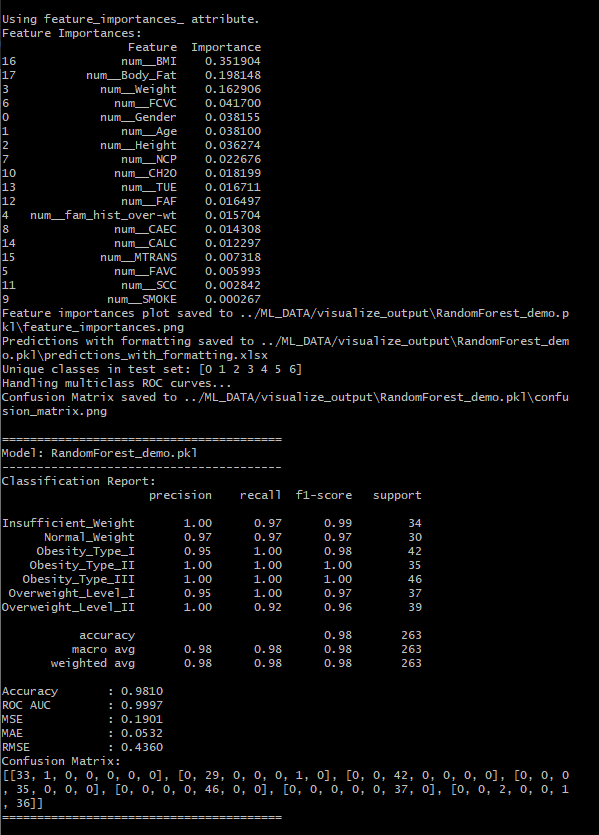
In summary, Random Forest’s low bias and controlled variance contribute to a stable and accurate model that performs reliably on both training and testing datasets. This bias-variance balance makes it an ideal choice for the insurance company's needs, where both accuracy and generalization are essential to minimize financial risk effectively.

##### 4.3.10.5 Comparison of Feature Variations in the Random Forest Model

The results of using all features in the Random Forest model demonstrate a significant improvement over using only the top 5 or 10 features, despite those features being identified as highly important in the initial feature importance analysis. Here’s a breakdown of the performance differences and reasons for the observed results:

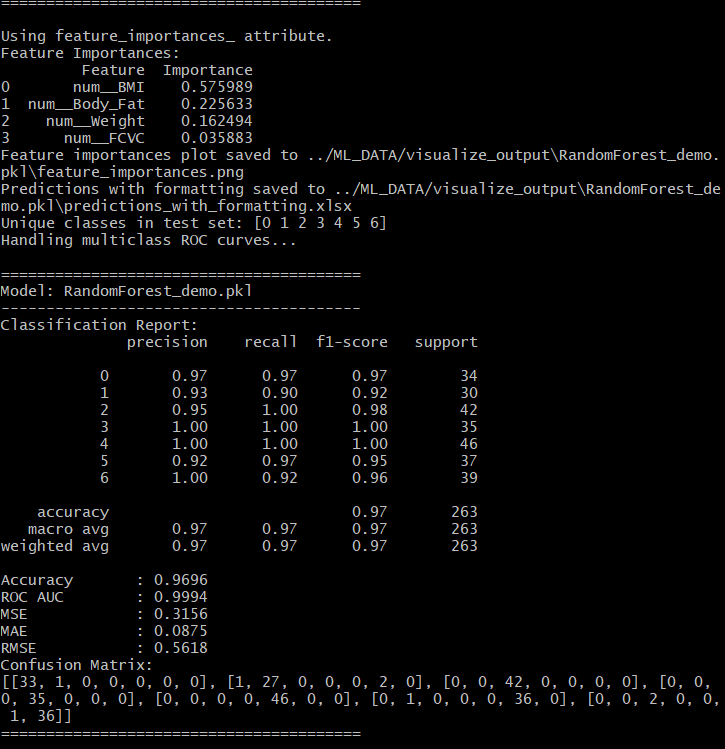
###### 4.3.10.5.1 Performance Summary

* **All Features (Original Model):** This configuration yielded the highest accuracy, precision, recall, and ROC-AUC scores. The balanced performance across both training and testing sets indicates a well-generalized model that avoids both underfitting and overfitting.



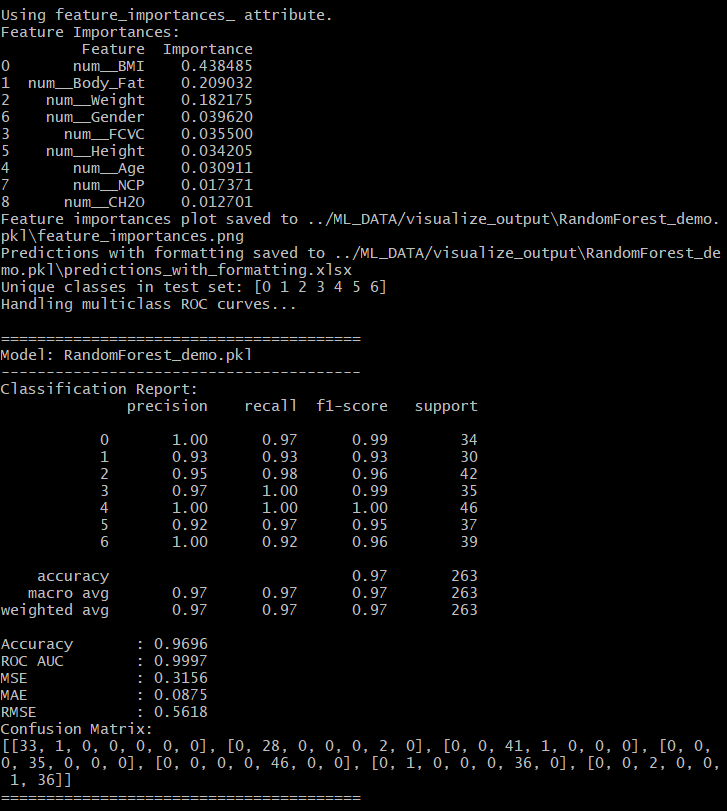
**Fig 65 : All features**

**Top 5 Features:** While still achieving moderate results, this model configuration showed a noticeable drop in accuracy and ROC-AUC, particularly in predicting minority classes. Precision and recall metrics, especially for class 1 (survivors), also decreased, suggesting that important predictive nuances are lost when reducing features to the top 5.



**Fig 66 :Top 5 features**

**Top 10 Features:** This configuration improved slightly over the top 5 but still underperformed relative to using all features. Although the selected features were among the most influential, they failed to capture the full complexity of the data, resulting in lower scores for recall and F1 on the test set.



**Fig 67 : Top 10 features**

###### 

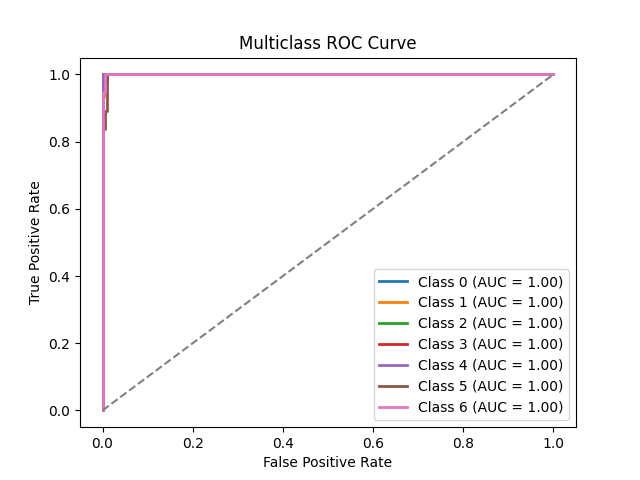
###### 4.3.10.5.2 Explanation of Differences

1. **Loss of Information:**
   * By limiting the model to only the top 5 or 10 features, less predictive information is available for decision-making. Important but less prominent features (ranked outside the top 10) still contribute to capturing the variability and unique aspects of certain classes. For example, demographic or categorical variables not in the top features may still help distinguish survival outcomes under specific scenarios.
2. **Reduced Model Complexity:**
   * Using all features allows the model to explore complex interactions between variables, which might not be fully realized with fewer features. Random Forest inherently benefits from this added complexity because it combines trees that make diverse splits based on different aspects of the data, ultimately leading to a more accurate aggregate prediction.
3. **Class-Specific Sensitivity:**
   * In an insurance-based use case, predicting class 0 (non-survivors) accurately is crucial for assessing risk. With only the top features, the model lacks sufficient depth to distinguish certain survival probabilities, especially those influenced by niche factors. This loss of nuance results in poorer performance for class 1 (survivors), with lower recall and F1-scores.
4. **Impact of Multi-Collinearity:**
   * Using a larger set of features can help mitigate the potential impact of multi-collinearity by allowing the Random Forest algorithm to select complementary information across trees. By reducing the feature set, multi-collinearity among the top-ranked features might become more problematic, reducing the diversity and efficacy of individual trees in the forest.

##### 4.3.10.6 Conclusion

For the given dataset, using all features yields the best overall performance for Random Forest, especially in an insurance context where class 0 predictions are more critical. Reducing features, even to the most important ones, sacrifices valuable information and model flexibility, impacting generalization and the robustness needed to assess diverse survival scenarios.

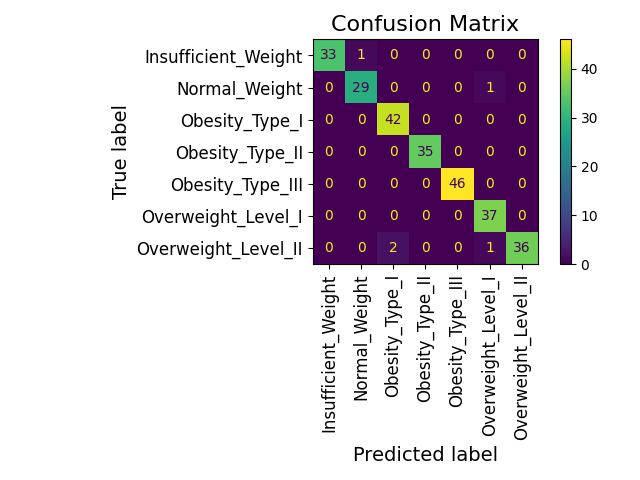
#### 4.3.11 Random Forest Visualization



**Fig 68 : Multiclass ROC Curve for Random Forest**

#### 4.3.11.1 ROC Curve Analysis and Model Performance

* **Multiclass ROC Curve:**
  + The ROC curve shows exceptional model performance, with **AUC values of 1.00** for all obesity categories.
  + This indicates the model's good classification ability to categorize obesity levels accordingly.
* **F1 Score:**
  + The model achieved an **F1 score of 0.98**, reflecting a strong balance between precision and recall.
  + This highlights the model's effectiveness in minimizing **false positives** and **false negatives**, ensuring accurate predictions across all categories.
* **Alignment of Metrics:**
  + The combination of **perfect AUC values** and a **high F1 score** showcases the Random Forest model’s robustness in handling multiclass classification tasks.
  + This consistency supports the model's suitability for predicting obesity levels based on various dietary, lifestyle, and physiological factors.
* **Recommendation:**
  + While the metrics demonstrate exceptional performance, domain expertise should complement model predictions to account for edge cases or unmodeled factors.



**Fig 69 : Confusion Matrix for Random Forest**

#### 4.3.11.2 Analysis of the Confusion Matrix

The confusion matrix provides an overview of the random forest model's performance in predicting obesity levels across all seven categories. Here's a breakdown:

**1. Overall Observations:**

* The diagonal entries represent the correctly classified instances for each obesity level.
* Off-diagonal entries show misclassifications, i.e., where the model predicted the wrong obesity level.

**2. Category-Specific Observations:**

* **Insufficient Weight**:
  + True positives: 33 out of 34 samples.
  + Only 1 sample was misclassified as **Normal Weight**.
  + High precision and recall indicate the model's strong ability to distinguish this category.
* **Normal Weight**:
  + True positives: 29 out of 30 samples.
  + Only 1 sample was misclassified, again suggesting excellent predictive performance.
* **Obesity Type I**:
  + True positives: All 42 samples correctly classified.
  + Perfect precision and recall for this category.
* **Obesity Type II**:
  + True positives: All 35 samples correctly classified.
  + No misclassifications, indicating that the model effectively differentiates this category from others.
* **Obesity Type III**:
  + True positives: All 46 samples correctly classified.
  + This category also shows no misclassifications, reflecting the clear separability of this group.
* **Overweight Level I**:
  + True positives: All 37 samples correctly classified.
  + This is another category with perfect classification.
* **Overweight Level II**:
  + True positives: 36 out of 38 samples.
  + 2 samples were misclassified as **Obesity Type II**, likely due to overlapping feature characteristics (e.g., BMI or body fat percentage) between these two categories.

**3. Key Metrics:**

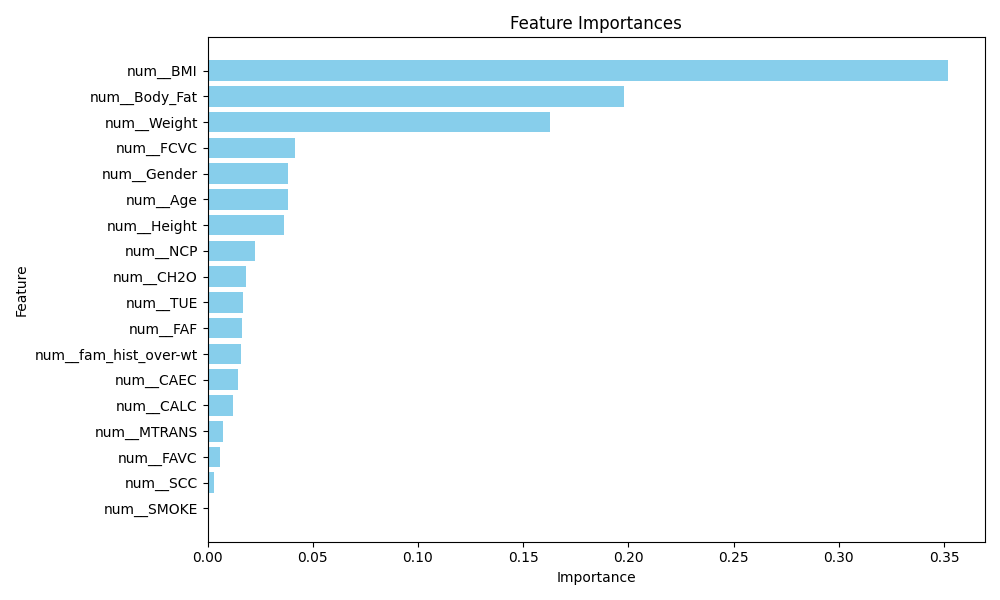
* **High True Positives Across Categories**:
  + Most categories have little to no misclassifications, indicating excellent overall accuracy.
* **Few Misclassifications**:
  + Misclassifications occurred primarily between adjacent categories in terms of obesity levels (e.g., between **Overweight Level II** and **Obesity Type II**). This is expected as these categories may have overlapping feature values (e.g., BMI, body fat percentage).

**4. Insights:**

* **Model Strength**:
  + The confusion matrix demonstrates the random forest model's robustness in correctly predicting obesity levels, with almost all categories achieving near-perfect or perfect predictions.
* **Areas for Improvement**:
  + While misclassifications are minimal, refining the model's ability to distinguish between **Overweight Level II** and **Obesity Type II** could further enhance performance. This could involve:
    - Investigating features that overlapped and “confused” the model between these categories.
    - Including additional relevant features (e.g., activity levels, dietary habits) to reduce ambiguity.

**Conclusion:**

The confusion matrix validates the random forest model as a reliable classifier for obesity levels, with exceptionally high accuracy, precision, and recall. The few misclassifications indicate areas of subtle overlap between certain categories, which may warrant further feature engineering or data exploration for improvement.



**Fig 70 : Feature Importance plot for Random Forest**

#### 4.3.11.3 Feature Importances

The feature importance graph highlights the contribution of each variable to the predictive performance of the random forest model in classifying obesity levels. Below is an analysis of the results:

1. **Dominant Features**:
   * **BMI (Body Mass Index)**: As expected, BMI is the most important feature, contributing significantly to the model's predictive power. This aligns with its direct correlation with obesity levels, as BMI inherently captures the relationship between height and weight, which are foundational in defining obesity categories.
   * **Body Fat**: This is the second most critical feature, as body fat percentage provides a more physiological measure of obesity compared to BMI, especially in distinguishing between overweight and obese individuals.
   * **Weight**: Weight is the third most important feature, complementing BMI and body fat to refine classification. Together, these three features provide the backbone for the model's ability to predict obesity levels accurately.
2. **Notable Contributions**:
   * **FCVC (Frequency of Consumption of High-Calorie Food)**: This feature ranks fourth in importance, indicating that dietary habits strongly influence obesity levels. Its inclusion underscores the impact of lifestyle choices on body composition.
   * **Gender** and **Age**: These demographic features also play meaningful roles, as gender and age often affect metabolic rates, fat distribution, and predisposition to obesity.
   * **Height**: While less important than weight, height complements BMI and body fat in distinguishing between obesity categories.
3. **Secondary Features**:
   * **NCP (Number of Meals per Day)**, **CH2O (Water Intake)**, and **TUE (Time Using Technological Devices)**: These features show moderate importance, reflecting behavioral and lifestyle factors that influence obesity.
   * These secondary features may capture nuances in specific obesity levels but lack the strong correlation seen in BMI or body fat.
4. **Least Important Features**:
   * **FAF (Physical Activity Frequency)**, **CAEC (Consumption of Food Between Meals)**, **CALC (Alcohol Consumption)**, and others like **SCC (Calories Monitoring)** and **SMOKE (Smoking Habit)** contribute minimally to the model. While they may have some predictive value, their low importance suggests that they add marginal improvements when combined with more impactful features.
   * Features like **SCC** and **SMOKE** may have limited variability or weak correlations with obesity levels, explaining their reduced contribution.
5. **Model Performance Implications**:
   * The random forest model leverages **all features**, even those with lower importance, to achieve its superior performance. This characteristic allows the model to consider complex interactions between features, which might explain its slight edge over other models.
   * The ability to incorporate secondary and tertiary features improves the model's capacity to capture subtle variations in the data, leading to the high accuracy and robust classification metrics observed.
6. **Commonalities Across Models**:
   * The dominant reliance on BMI, body fat, and weight is consistent with other models, emphasizing their universal importance in predicting obesity levels.
   * However, the random forest’s ability to utilize additional features effectively distinguishes it from simpler models, enabling it to achieve better predictive outcomes.

#### 4.3.11.4 Model Evaluation and Recommendation for Obesity Level Prediction

The Random Forest model demonstrates exceptional performance in predicting obesity levels with a **high accuracy of 0.98**, a near-perfect **ROC AUC score of 0.9999**, and low error metrics, including a **Mean Squared Error (MSE) of 0.19**, **Mean Absolute Error (MAE) of 0.05**, and a **Root Mean Square Error (RMSE) of 0.43**. These metrics reflect the robustness of the model; however, there are some considerations that suggest it should be used as a **guidance tool** rather than the sole determinant for obesity classification.

##### 4.3.11.4.1 Key Findings:

1. **Strong Classification Performance**:
   * **High Accuracy**: The model correctly classified the vast majority of individuals across all obesity levels, as indicated by its 0.98 accuracy.
   * **Precision, Recall, and F1 Score**: All metrics exceed 0.95 for each class, reflecting the model’s strong ability to minimize both false positives and false negatives.
   * **Multiclass ROC Curve**: The ROC AUC score of 0.9999 suggests near-perfect separability between all classes.
2. **Insights from the Confusion Matrix**:
   * Most obesity categories, such as **Obesity Type I, Type II, and Type III**, were perfectly or near-perfectly predicted, with minimal misclassifications.
   * The few misclassifications (2) occurred in closely related categories (e.g., **Overweight Level II** misclassified as **Obesity Type II**), likely due to feature overlaps such as BMI and body fat percentage.
3. **Feature Importance**:
   * **BMI**, **Body Fat Percentage**, and **Weight** were the most influential features in predicting obesity levels, as expected.
   * Other features like **frequency of calorie consumption (FCVC)** and **gender** also contributed, indicating that the model leveraged a diverse range of attributes for its predictions.
   * Less influential features (e.g., smoking habits, calorie monitoring) highlight areas where additional data may enhance future predictions.

##### 4.3.11.4.2 Limitations

Despite its high performance, the model has notable limitations:

1. **Lack of Fitness Context**:
   * The model does not account for cases where individuals have high BMI or body fat percentage due to non-obesity-related factors, such as:
     + **Bodybuilders or athletes** with high muscle mass.
     + Individuals undergoing temporary body composition changes (e.g., **bulking phases**).
   * These outliers can lead to potential misclassifications, as the current data lacks attributes like **fitness levels** or detailed **physical activity measurements**.
2. **Unmeasured Variables**:
   * The dataset does not capture potentially significant variables like:
     + **Metabolic health indicators**.
     + **Detailed fitness activity logs**.
     + **Psychological factors** influencing obesity (e.g., stress eating).

##### 4.3.11.4.3 Recommendation:

The model is highly reliable as a decision-support tool for **classifying obesity levels** and **expediting the assessment process**. However, it is not sufficient as a standalone predictor. For comprehensive obesity classification, the model should be complemented with:

1. **Additional Data Collection**:
   * Incorporate features like **fitness level, muscle mass**, and **metabolic indicators** to improve the model’s robustness against edge cases (e.g., athletes with high BMI).
2. **Human Oversight**:
   * Use the model’s predictions as a starting point for healthcare professionals, who can refine classifications based on additional context.
3. **Integration with Other Tools**:
   * Combine the model with other tools that measure fitness levels or metabolic health for a holistic obesity assessment.

**4.3.11.4.4 Conclusion:**

The Random Forest model demonstrates robust predictive capabilities and is an excellent tool for supporting obesity classification. Its high accuracy, low error rates, and insightful feature usage make it a valuable asset for expediting predictions. However, reliance on this model alone is not recommended due to its inability to account for outliers and missing contextual data. Incorporating additional variables and using the model alongside expert judgment will ensure more accurate and informed obesity level classification

## 

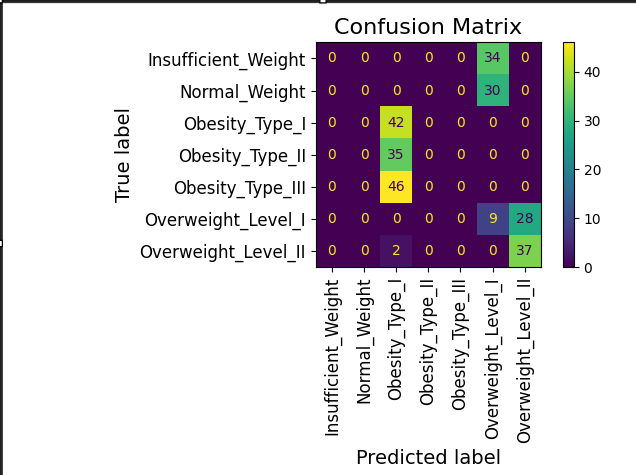
## 5 References

1. American College of Sports Medicine. (2023, September 29). *ACSM releases expert consensus statement: The biological basis of sex differences in athletic performance*.<https://www.acsm.org/news-detail/2023/09/29/acsm-releases-expert-consensus-statement-the-biological-basis-of-sex-differences-in-athletic-performance#:~:text=Biological%20sex%20is%20a%20determinant,in%20athletic%20performance%20are%20minimal>
2. insightsoftware. (n.d.). *Top 5 predictive analytics models and algorithms*.<https://insightsoftware.com/blog/top-5-predictive-analytics-models-and-algorithms/>
3. GeeksforGeeks. (n.d.). *What are the advantages and disadvantages of random forest?*<https://www.geeksforgeeks.org/what-are-the-advantages-and-disadvantages-of-random-forest/>
4. AIML. (n.d.). *What are the advantages and disadvantages of random forest?*<https://aiml.com/what-are-the-advantages-and-disadvantages-of-random-forest/>
5. AIML. (n.d.). *Ensemble learning - Supervised learning classification*.<https://aiml.com/category/ml-interview-questions/supervised-learning/classification/ensemble-learning/>
6. Data Science Wizards. (2020, April 12). *Understanding the AdaBoost algorithm*.<https://medium.com/@datasciencewizards/understanding-the-adaboost-algorithm-2e9344d83d9b>
7. AlmaBetter. (n.d.). *AdaBoost algorithm tutorial*.<https://www.almabetter.com/bytes/tutorials/data-science/adaboost-algorithm?srsltid=AfmBOoo-KCJQtaNHpdW660VMXZsnDFG4R-WHnmFd5x8bjxQTcy0gEHWv>
8. IBM. (n.d.). *What is the KNN algorithm?* [https://www.ibm.com/topics/knn#:~=What%20is%20the%20KNN%20algorithm,of%20an%20individual%20data%20point](https://www.ibm.com/topics/knn#:~:text=What%20is%20the%20KNN%20algorithm,of%20an%20individual%20data%20point)
9. Elastic. (n.d.). *What is KNN?*<https://www.elastic.co/what-is/knn>
10. Neptune.ai. (2021, March 17). *XGBoost: Everything you need to know*.<https://neptune.ai/blog/xgboost-everything-you-need-to-know>
11. Lazy Programmer. (n.d.). *ML compendium: XGBoost*.<https://lazyprogrammer.me/mlcompendium/ensemble/xgboost.html>
12. Machine Learning Interview. (n.d.). *Naive Bayes classifier: Advantages and disadvantages*.<https://machinelearninginterview.com/topics/machine-learning/naive-bayes-classifier-advantages-and-disadvantages/>
13. upGrad. (n.d.). *Naive Bayes explained*.<https://www.upgrad.com/blog/naive-bayes-explained/>

## 6 Appendix

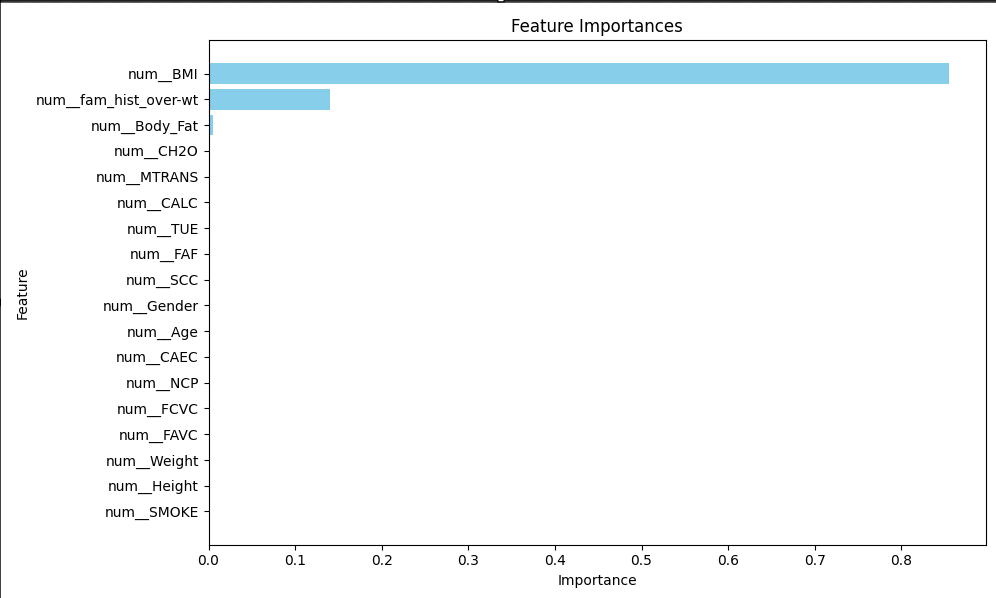
### A1 Model Metrics

**Adaboost Metrics**



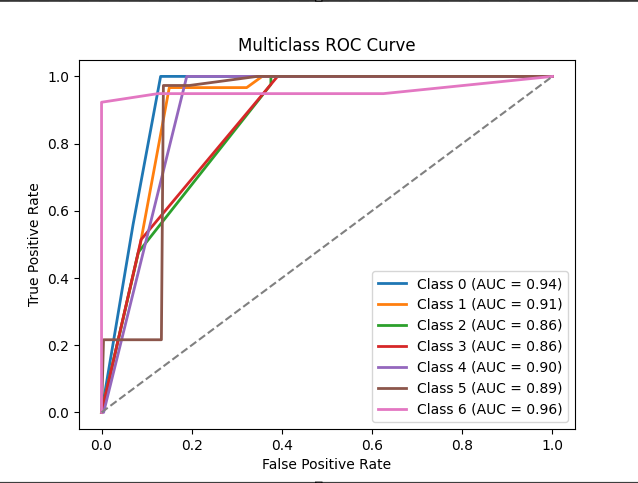
**Fig 71 : Confusion Matrix for Adaboost**

**Confusion matrix:** The model performs well for "Obesity\_Type\_I," "II," and "III," with most predictions on the diagonal. "Insufficient\_Weight" and "Normal\_Weight" are often misclassified as "Obesity\_Type\_III," and overlaps occur between "Overweight\_Level\_I" and "II."



**Fig 72 : Feature Importance for Adaboost**

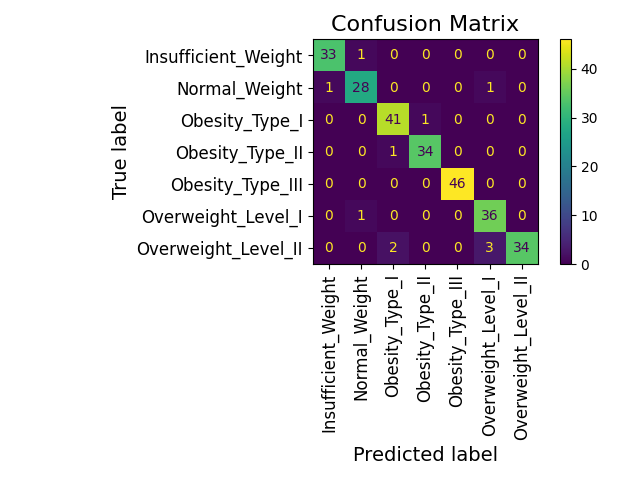
**Feature importance plot :** Indicates num\_BMI as the most critical feature, followed by "num\_fam\_hist\_over\_wt," with other features contributing minimally to the model's prediction



**Fig 73 : Multiclass ROC Curve for Adaboost**

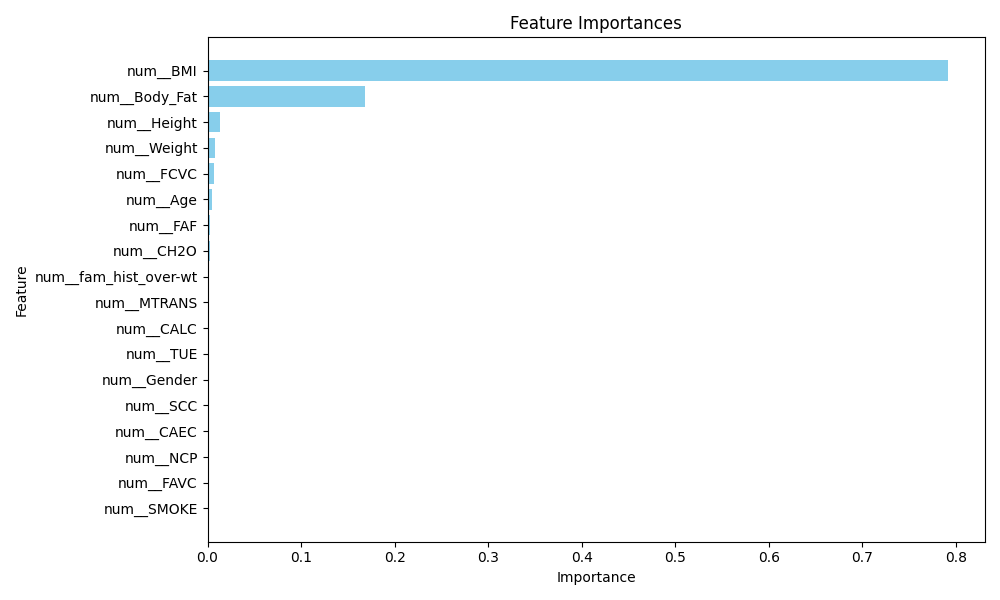
**ROC Curve**: With AUCs ranging from 0.86 to 0.96, the model shows strong discriminative ability across classes, with the highest performance for Class 6 (0.96) and room for improvement for Classes 2 and 3 (0.86).

**Decision Tree Metrics**



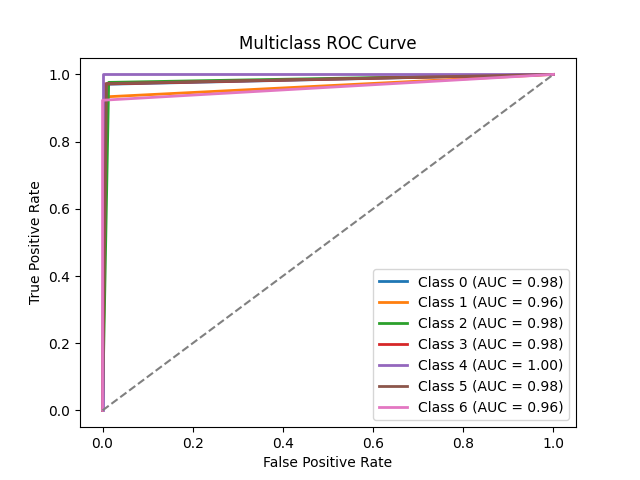
**Fig 74 : Confusion Matrix for Decision Tree**

**Confusion matrix:** The confusion matrix shows the classification performance across seven weight categories. Most predictions align well with the true labels, but there are slight misclassifications, notably in adjacent categories like "Overweight Level II" and "Obesity Type II."



**Fig 75 : Feature Importance for Decision Tree**

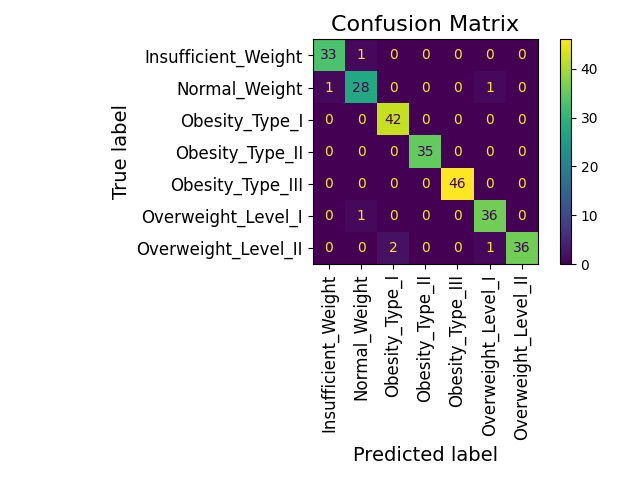
**Feature importance plot :** Indicates num\_BMI as the most critical feature, followed by num\_Body\_Fat. Other features like num\_Hieght, num\_Weight, num\_FCVC and num\_Age minimally contribute to the model's predictions



**Fig 76 : Multiclass ROC Curve for Decision Tree**

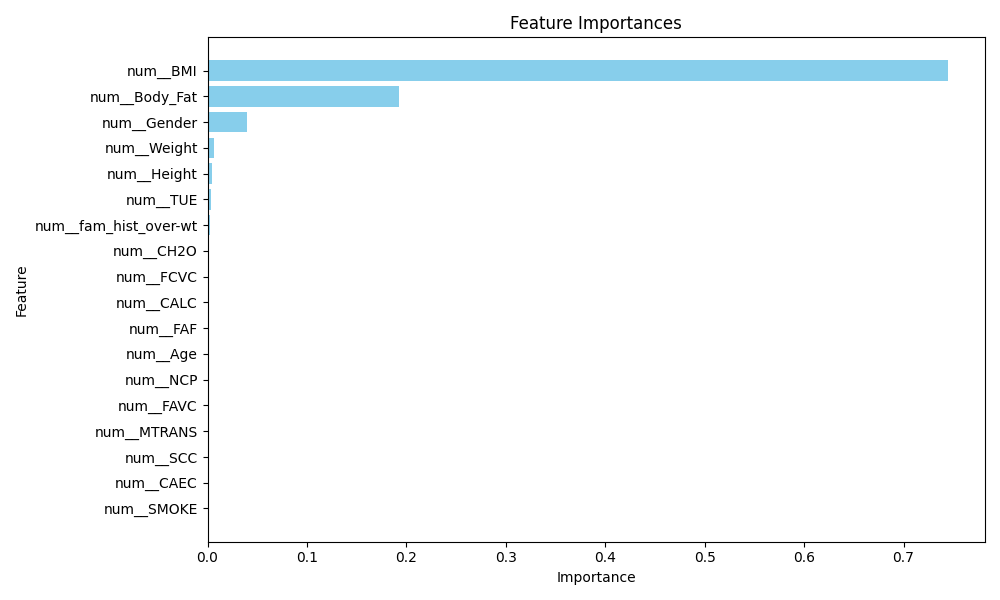
**ROC Curve**: With AUCs ranging from 0.96 to 1.00, the model shows strong discriminative ability across all classes

**Gradient Boosting Metrics**



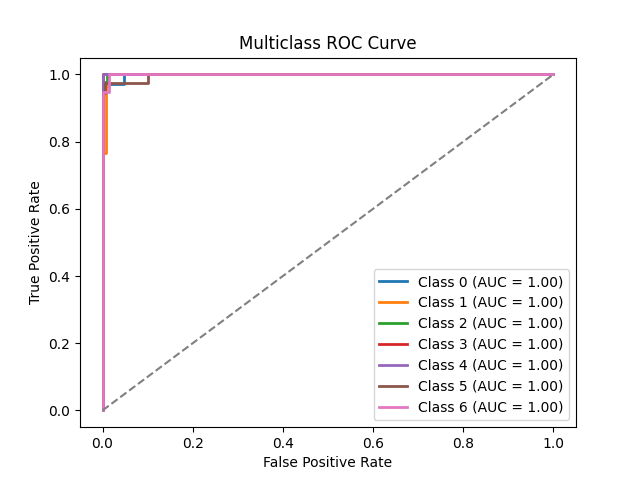
**Fig 77 : Confusion Matrix for Gradient Boosting**

**Confusion matrix:** The confusion matrix shows the classification performance across seven weight categories. Most predictions align well with the true labels with minor errors identifying Normal\_Weight and Obesity\_Type\_1



**Fig 78 : Feature Importance for Gradient Boosting**

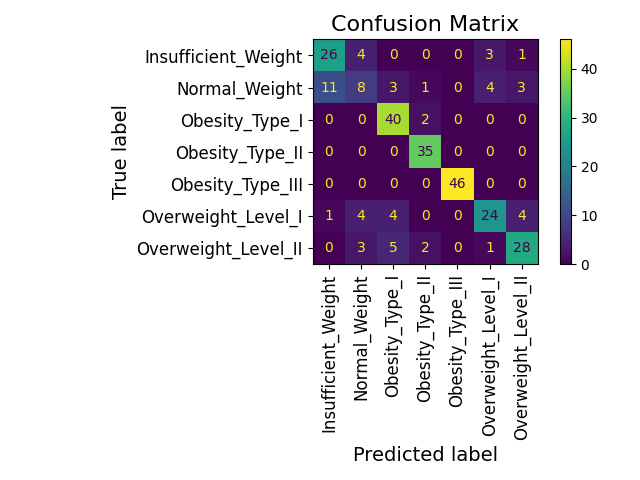
**Feature importance plot :** Indicates num\_BMI as the most critical feature, followed by num\_Body\_Fat. Other features like num\_Hieght, num\_Weight, num\_FCVC and num\_Age minimally contribute to the model's predictions



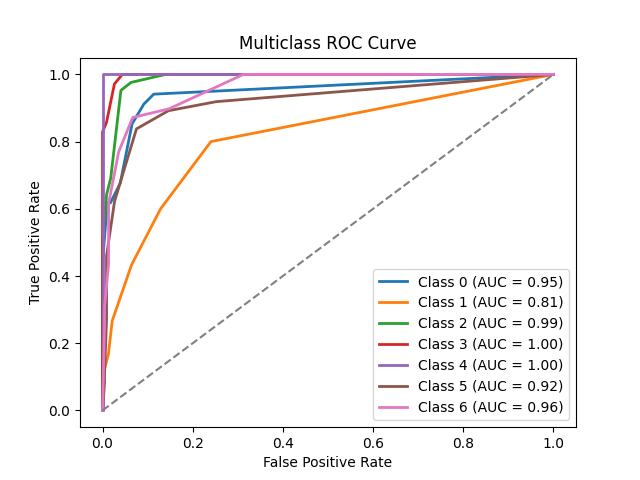
**Fig 78 :Multiclass ROC Curve for Gradient Boosting**

**ROC Curve**: With an AUC value of 1.00, the model shows excellent classification performance across all classes

**KNeighbours**



**Confusion matrix:** The Model accurately identifies labels like Obesity\_Type\_I, II and III but it has misclassified multiple labels as Insufficient\_Weight

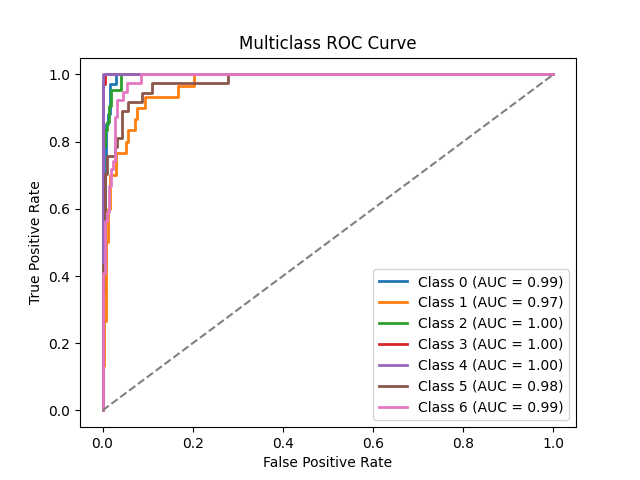


**ROC Curve:** With an AUC value ranging from 0.81 to 1.00, the model shows excellent classification performance for Classes 3 and 4 but needs improvement for Class 1

**Logistics Regression**

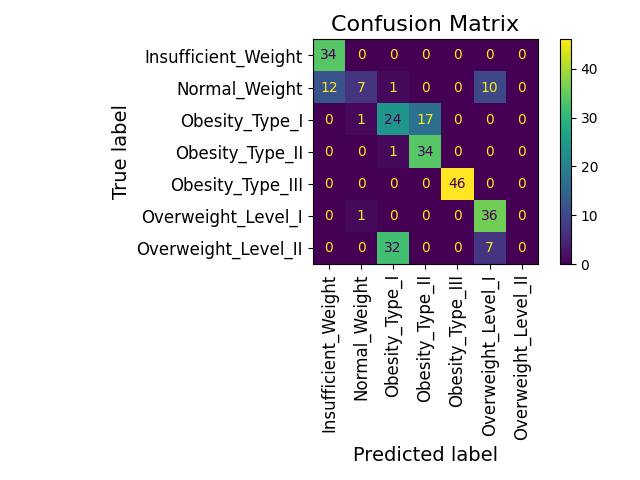
## 

**Confusion matrix:** Most predictions align well with the true labels with minor errors identifying Overweight\_Level\_1 and Normal\_Weight

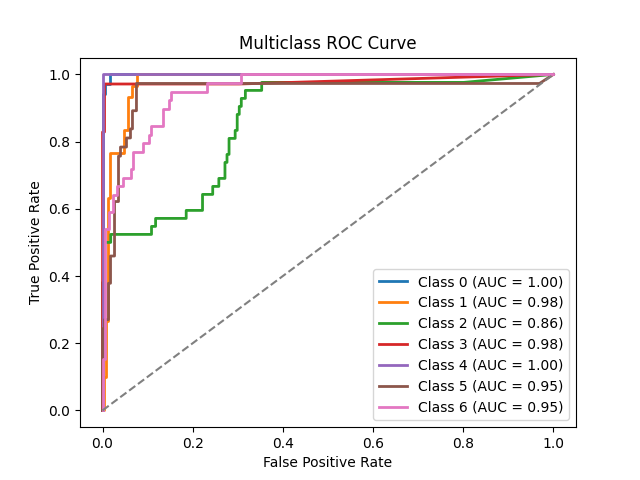


**ROC Curve**: With AUCs ranging from 0.97 to 1.00, the model shows strong discriminative ability across all classes

**Naive Bayes**



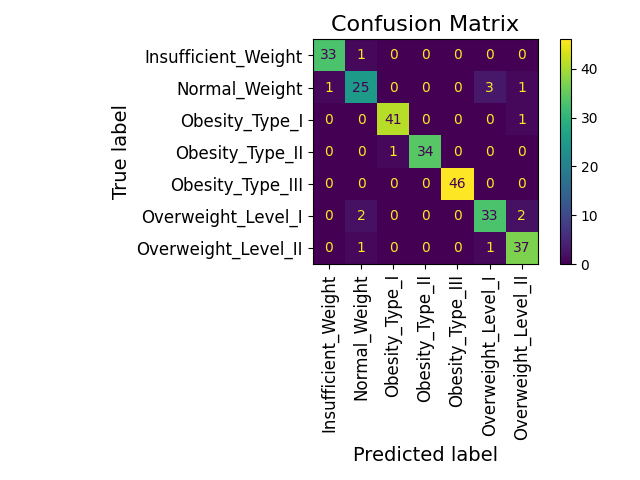
**Confusion matrix:** Model is able to predict Obesity\_Type I,II,III labels well but there is room for improvement in predicting Normal\_Weight and Overweight\_Level\_I



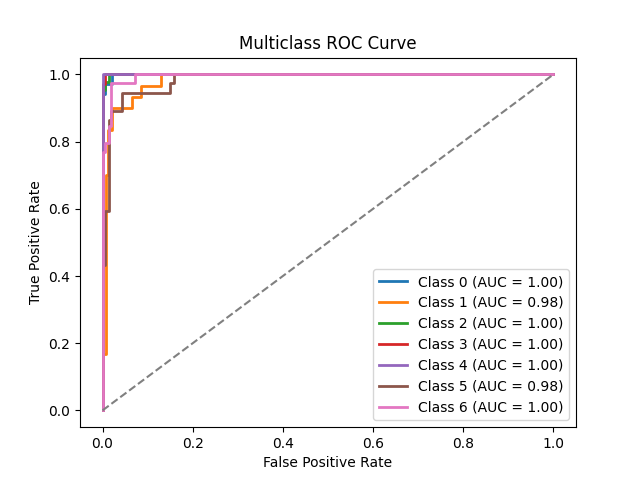
**ROC Curve:** Model shows great discriminative ability for Classes 0 and 4 but needs to improve for

Class 2

**Neural Network**

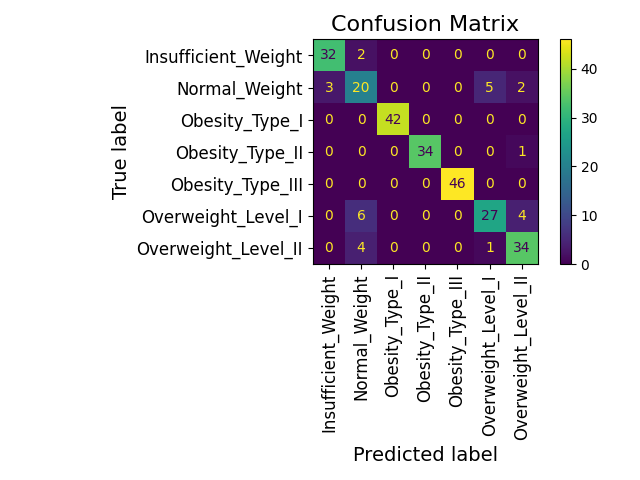


**Confusion matrix:** Most predictions align well with the true labels with minor errors predicting Overweight\_Level\_I and Normal\_Weight.



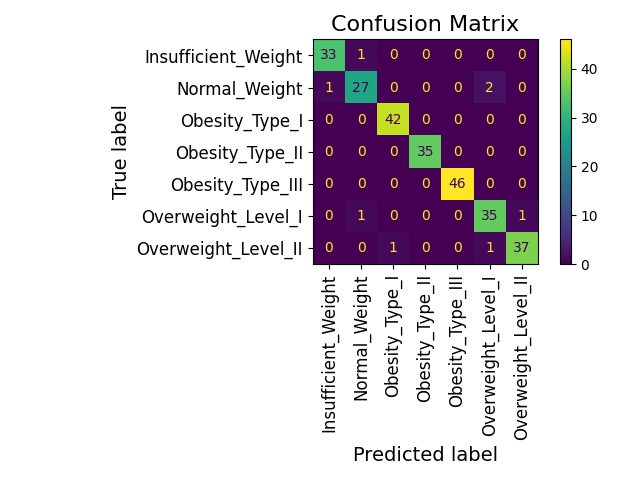
**ROC Curve:** With AUC values running from 0.98 to 1.00, the model shows excellent classification performance across all classes

**SVC**

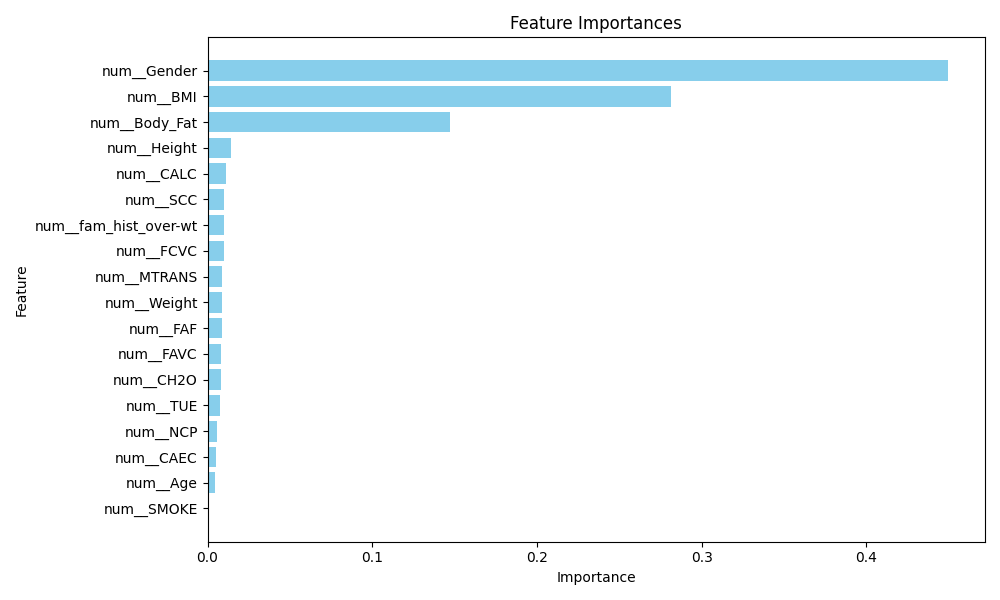


**Confusion matrix:** The model generally predicts all the labels accurately with some mistakes in predicting Normal\_Weight and Overweight\_Level\_1

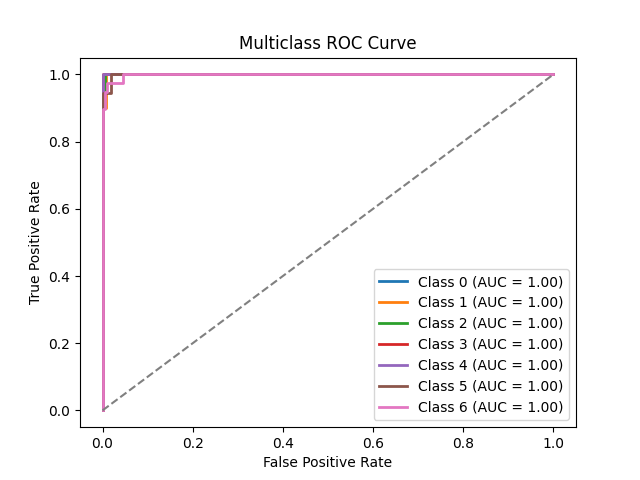
**XGBoost**



**Confusion matrix:** Most predictions align well with the true labels with minor errors identifying Normal\_Weight



**Feature importance plot :**  Highlights num\_Gender as the most influential feature, followed by num\_BMI and num\_Body\_Fat. Other features have minimal impact on the model's predictions.



**ROC Curve**: With an AUC value of 1.00, the model shows excellent classification performance across all classes