AIT511: Machine Learning

Group Project Report

Obesity Level Prediction using Machine Learning

Group Members:

Jenish Niteshbhai Vekariya (MT2025055)

Bhautik Pravinbhai Vekariya (MT2025029)

Department of Computer Science
IIITB

Contents

1 Introduction **3**
- 1.1 Background . . . 3
- 1.2 Problem Statement . . . 3
- 1.3 Dataset Description . . . 3

2 Data Overview and Exploratory Data Analysis **4**
- 2.1 Dataset Overview . . . 4
- 2.2 Data Quality Check . . . 4
- 2.3 Univariate Analysis . . . 5
  - 2.3.1 Target Variable Distribution . . . 5
  - 2.3.2 Numerical Features Distribution . . . 6
- 2.4 Bivariate Analysis . . . 7
  - 2.4.1 Categorical Features vs Target . . . 7
  - 2.4.2 Numerical Features vs Target . . . 8
- 2.5 Correlation Analysis . . . 12
- 2.6 Feature Importance Analysis . . . 12

3 Data Preprocessing and Feature Engineering **14**
- 3.1 Data Cleaning . . . 14
- 3.2 Encoding Categorical Variables . . . 14
  - 3.2.1 Label Encoding for Target Variable . . . 14
  - 3.2.2 One-Hot Encoding for Categorical Features . . . 14
- 3.3 Train-Test Separation Strategy . . . 15
- 3.4 Feature Scaling . . . 15
- 3.5 Dimensionality Reduction with PCA . . . 15

4 Methodology **16**
- 4.1 Model Selection . . . 16
  - 4.1.1 Approach 1: SVM Pipeline with Dimensionality Reduction . . . 16
  - 4.1.2 Approach 2: Random Forest with Optuna Optimization . . . 16
  - 4.1.3 Approach 3: XGBoost with Optuna . . . 17
- 4.2 Hyperparameter Tuning . . . 17
  - 4.2.1 Optuna Framework . . . 17
  - 4.2.2 Parameter Search Spaces . . . 17
- 4.3 Training Strategy . . . 18
  - 4.3.1 Cross-Validation Strategy . . . 18
  - 4.3.2 Ensemble Methods . . . 18
  - 4.3.3 Data Splitting . . . 18

4.3.4 Handling Class Imbalance 19

5 Experiments and Results **20**
5.1 Experimental Setup 20
5.1.1 Evaluation Metrics 20
5.2 Hyperparameter Tuning Results 20
5.2.1 Random Forest Optimization 20
5.2.2 XGBoost Optimization 20
5.2.3 Top Ensemble Models 21
5.3 Model Performance 22
5.3.1 Approach 1: SVM Pipeline Results 22
5.3.2 Approach 2: Random Forest Pipeline 22
5.3.3 Approach 3: XGBoost Pipeline 22
5.4 Comparative Analysis 23
5.4.1 Overall Performance Ranking 23
5.4.2 Key Observations 23
5.4.3 Feature Importance Analysis 23

6 Discussion **24**
6.1 Performance Interpretation 24
6.2 Feature Importance Analysis 24
6.3 Model Analysis 24
6.3.1 Impact of Dimensionality Reduction 24
6.3.2 Ensemble Strategy Effectiveness 25
6.3.3 Analyzing SVM to Remove Outliers 25
6.4 Impact of Synthetic Data 26
6.4.1 Data Quality Considerations 26
6.4.2 Modeling Implications 26
6.4.3 Limitations and Considerations 26
6.4.4 Recommendations for Synthetic Data Usage 26

7 GitHub Repository **27**

Chapter 1

Introduction

1.1 Background

It's getting more and harder to maintain a healthy lifestyle. This dataset examines the relationships between an individual's weight category and their demographic data, daily routines, eating patterns, and physical activity. Participants are given the challenge of creating models that can accurately classify individuals into groups such as inadequate weight, normal weight, overweight, or obesity levels.

The dataset includes characteristics such as age, gender, family history, food consumption habits, physical activity, technology use, and transportation modes. The combination of behavioral science, machine learning, and healthcare makes it a challenging and realistic problem.

1.2 Problem Statement

In order to forecast a person's *WeightCategory* based on lifestyle and physiological characteristics, this study tackles a multi-class classification problem. The target variable consists of multiple classes, including *Insufficient_Weight*, *Normal_Weight*, *Overweight_Level_I*, *Overweight_Level_II*, *Obesity_Type_I*, *Obesity_Type_II*, and *Obesity_Type_III*.

1.3 Dataset Description

A deep learning model uses the actual Obesity/CVD risk data to create a synthetic dataset for this project. While preserving the statistical characteristics and connections of the original data, this synthetic generation guarantees privacy. Numerous facets of people's lifestyles and physiological traits are captured by the dataset's numerical and category attributes.

Chapter 2

Data Overview and Exploratory Data Analysis

2.1 Dataset Overview

Both training and testing divisions of the dataset are included, and it has numerous variables that capture physiological, behavioral, and demographic traits. Each feature's distribution and properties are revealed via basic statistical analysis.

2.2 Data Quality Check

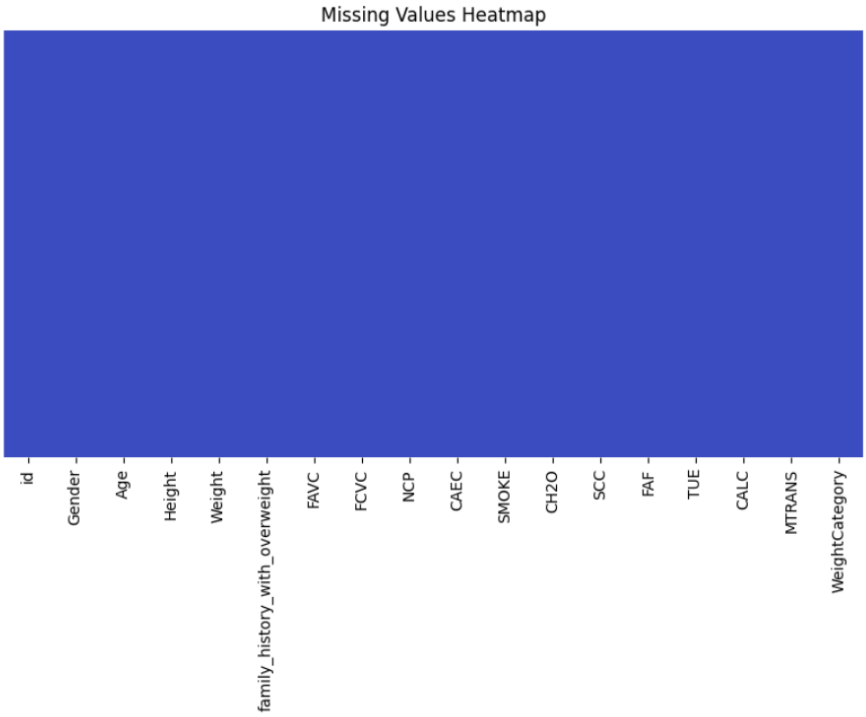

Figure 2.1: Missing Values Heatmap - Confirming no missing values in the dataset

The missing values heatmap verified the initial assessment's finding that there were no missing values in the dataset. Every feature was formatted correctly and used the right data types. Due to the dataset's combination of categorical and numerical variables, various preprocessing techniques are needed.

2.3 Univariate Analysis

2.3.1 Target Variable Distribution

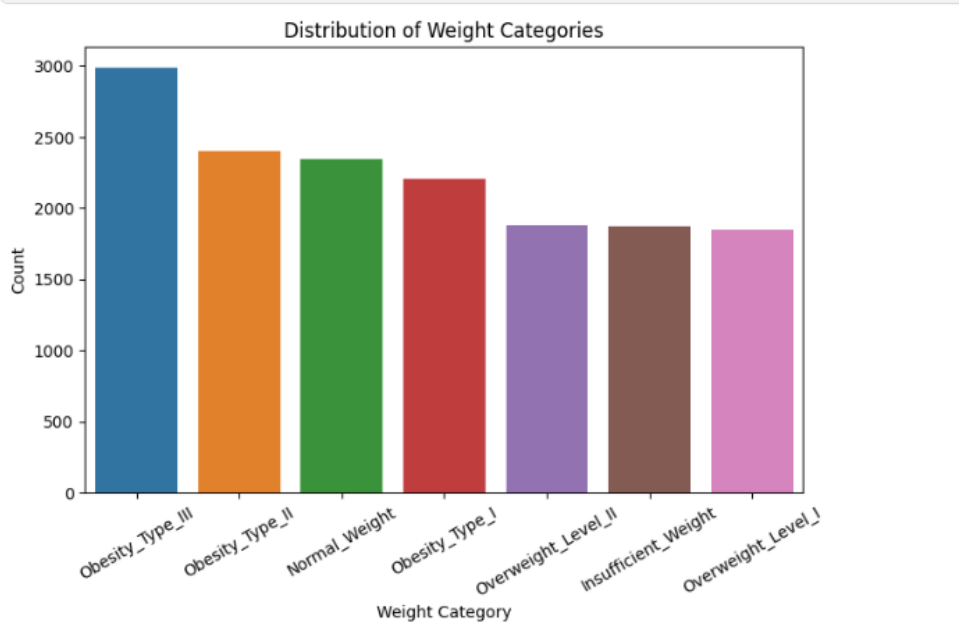

Figure 2.2: Distribution of Weight Categories - Shows heavy class imbalance with Obesity_Type_III and Obesity_Type_II as majority classes

The target variable distribution reveals significant class imbalance, which is crucial for modeling as classifiers might struggle to predict minority classes like Normal_Weight or Insufficient_Weight.

2.3.2 Numerical Features Distribution

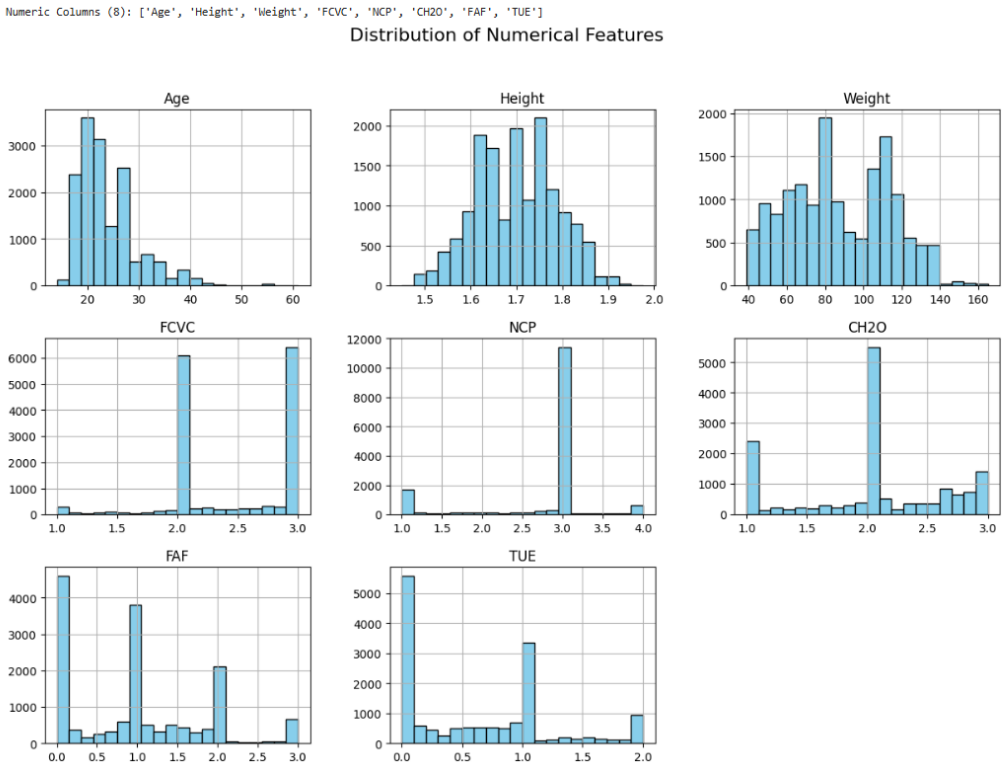

Figure 2.3: Distribution of Numerical Features - Showing bimodal/multi-modal distributions for Age, Height, Weight, NCP, and CH2O

Numerical features exhibit various distribution patterns:

- Bimodal/Multi-modal distributions observed in Age, Height, Weight, NCP, and CH2O
- Discrete nature of FCVC, NCP, CH2O, FAF, and TUE despite float representation
- Moderate skewness in Age and Weight distributions

2.4 Bivariate Analysis

2.4.1 Categorical Features vs Target

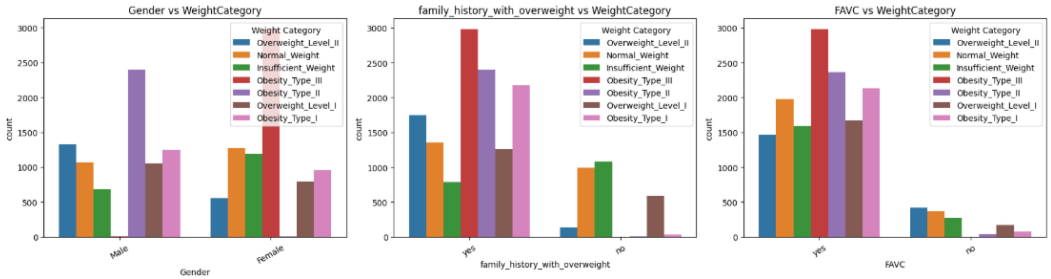

Figure 2.4: Categorical Features vs Target (Part 1) - Showing relationships for Gender, Family History, and FAVC

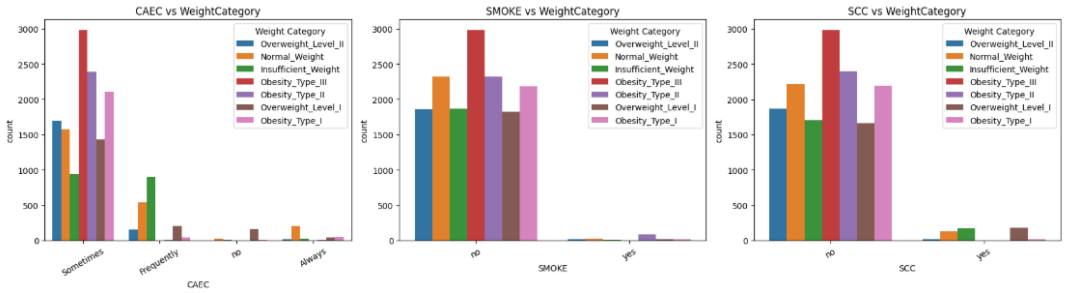

Figure 2.5: Categorical Features vs Target (Part 2) - Additional categorical relationships

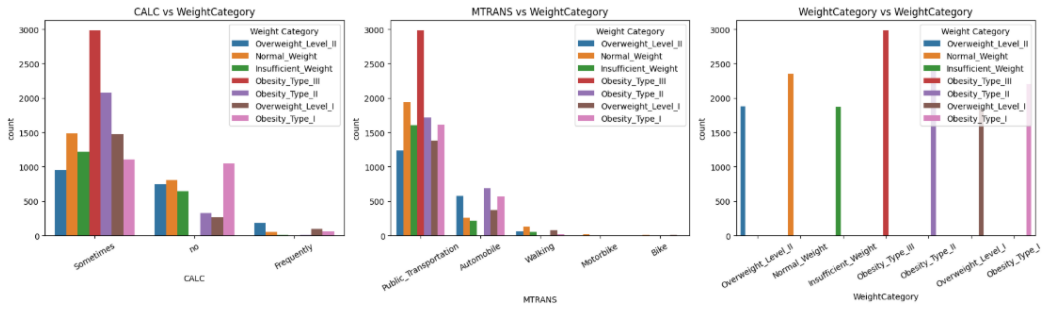

Figure 2.6: Categorical Features vs Target (Part 3) - Transportation mode (MTRANS) relationships

Key observations from categorical analysis:

- Clear gender differences in weight category distribution
- Strong family history influence on obesity categories
- Frequent high-calorie food consumption strongly associated with higher obesity classes
- Transportation mode shows significant correlation with weight categories

2.4.2 Numerical Features vs Target

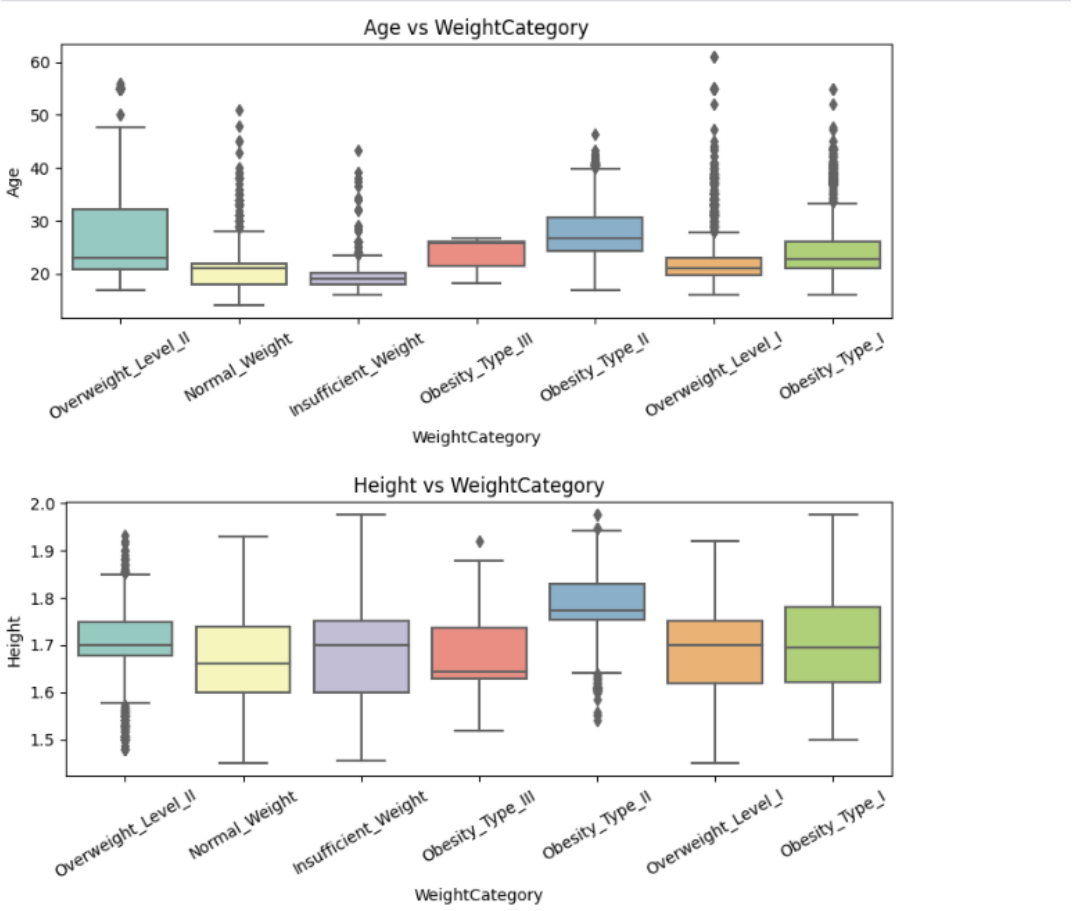

Figure 2.7: Box Plots (Part 1) - Weight and Height vs WeightCategory showing clear progressive increase in Weight across categories

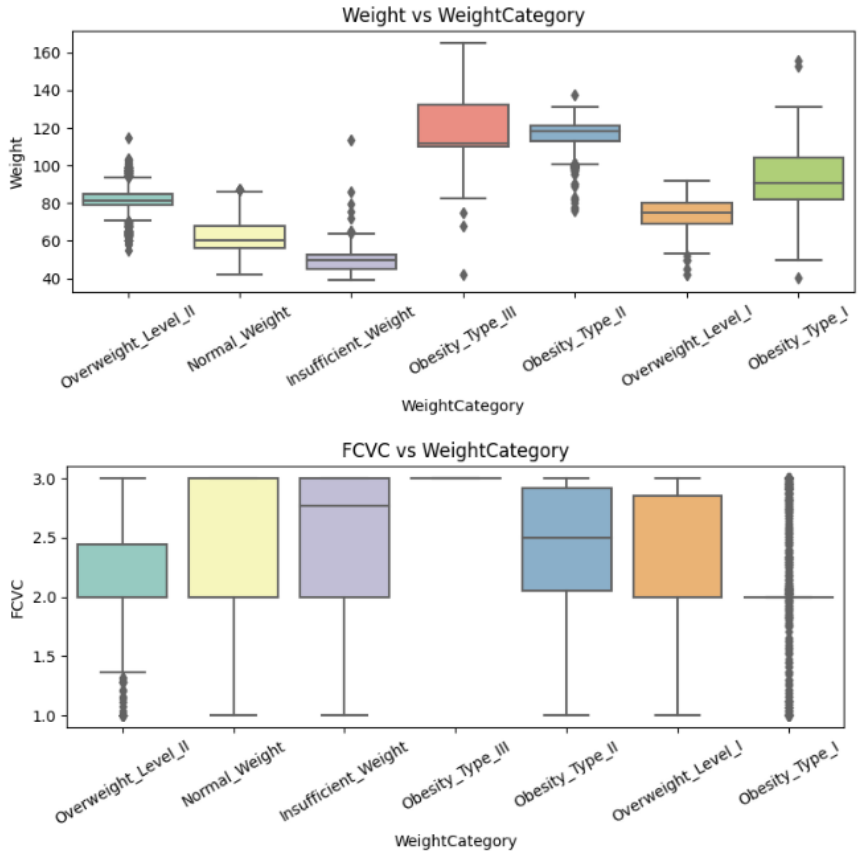

Figure 2.8: Box Plots (Part 2) - Age vs WeightCategory showing gradual age increase with obesity severity

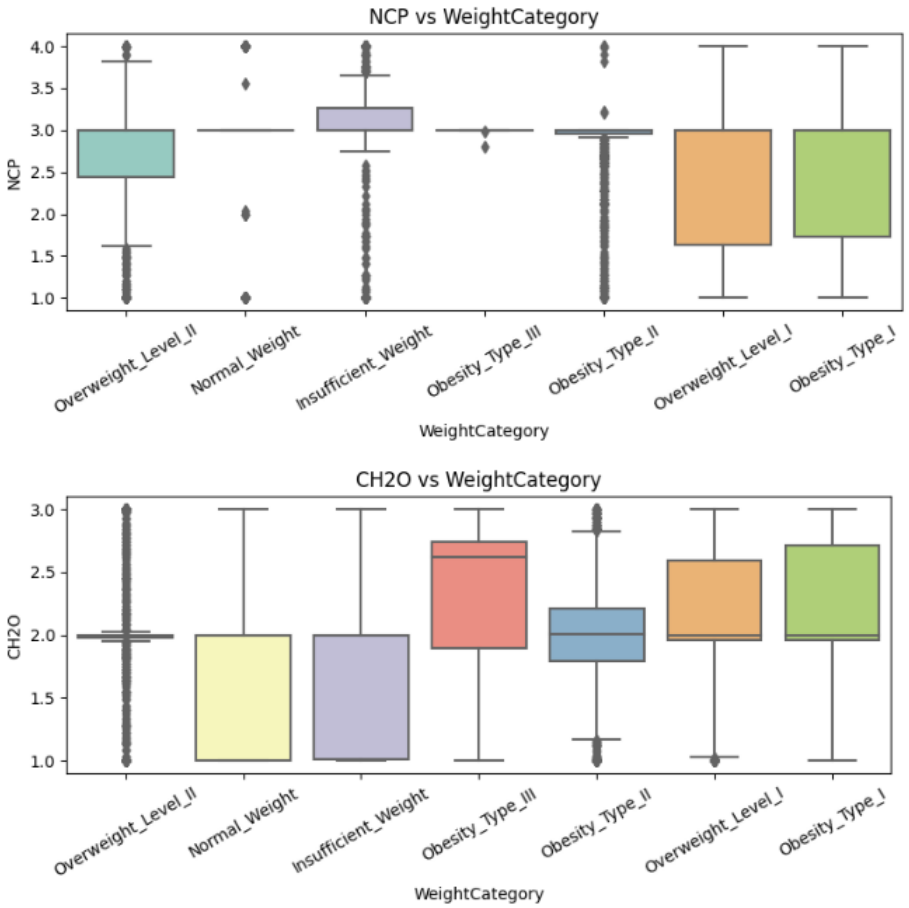

Figure 2.9: Box Plots (Part 3) - Additional numerical features vs target

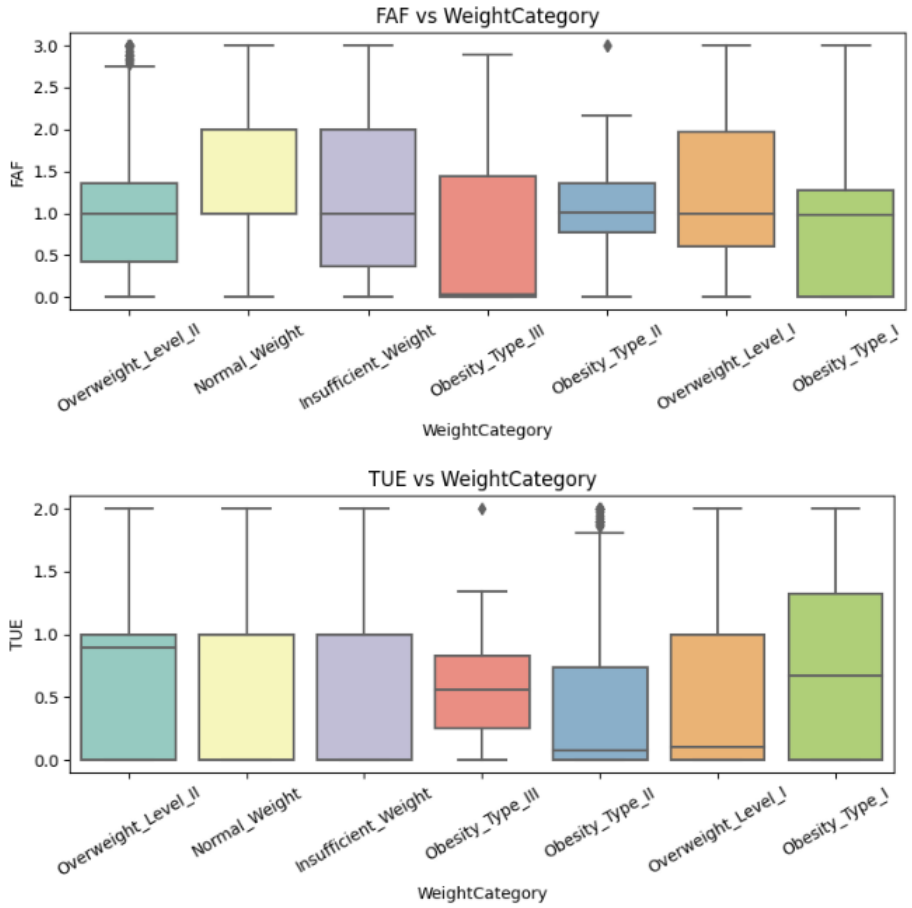

Figure 2.10: Box Plots (Part 4) - FAF (Physical activity) vs Target showing inverse relationship with weight categories

Key numerical relationships:

- Weight shows the clearest discriminative power across categories
- Height shows moderate discrimination with considerable overlap
- Age demonstrates gradual increase with obesity severity
- Physical activity frequency (FAF) decreases with increasing weight categories

2.5 Correlation Analysis

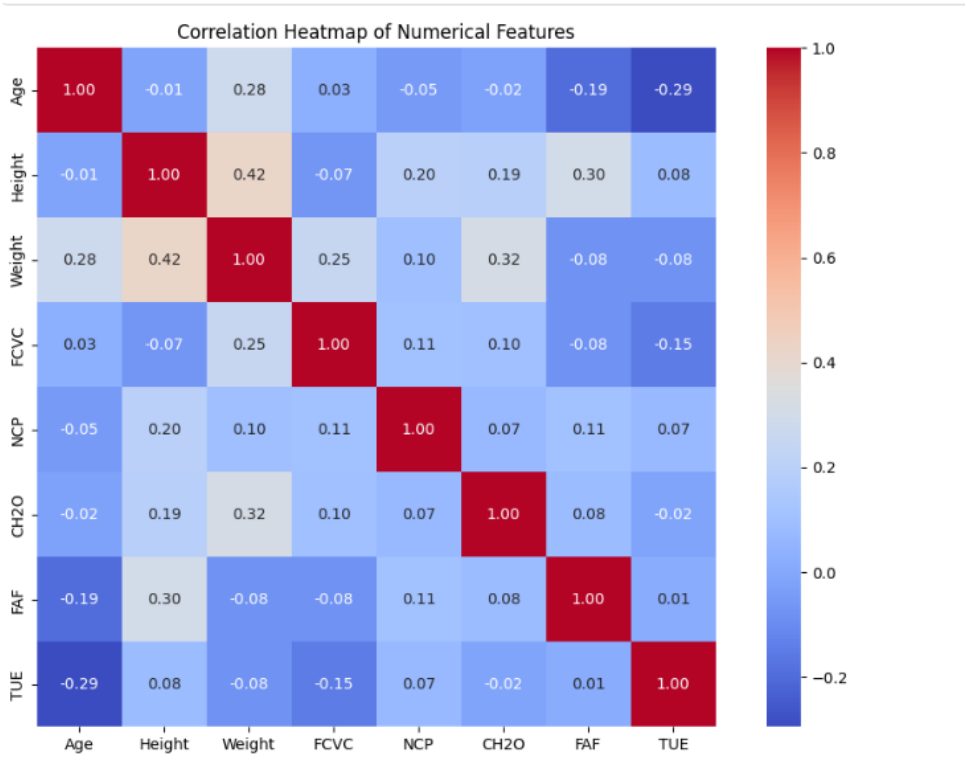

Figure 2.11: Correlation Heatmap of Numerical Features - Showing strongest correlation between Weight and Height

Correlation analysis reveals:

- Strongest correlation between Weight and Height
- Moderate correlations between dietary features (FCVC, NCP, CH2O)
- Low multicollinearity among most independent features
- Weak correlations suggest features provide complementary information

2.6 Feature Importance Analysis

Feature importance analysis using Random Forest's mean decrease in impurity technique revealed the following importance scores:

- Weight: 39.56%
- Age: 13.94%
- FCVC: 6.28%
- Gender_Male: 4.67%

- Gender_Female: 4.50%
- FAF: 4.29%
- NCP: 3.86%
- CH2O: 3.28%
- TUE: 3.28%
- CALC_Sometimes: 1.89%
- family_history_with_overweight_no: 1.63%
- family_history_with_overweight_yes: 1.56%
- Height: 0%

Chapter 3

Data Preprocessing and Feature Engineering

3.1 Data Cleaning

Categorical variable values had to be standardized as part of the data cleansing procedure. To maintain data integrity and ensure correct encoding, gender values were specifically changed from a mixed *'male'/'female'* format to a consistent *'Male'/'Female'* format.

3.2 Encoding Categorical Variables

3.2.1 Label Encoding for Target Variable

To transform the multi-class labels into a numerical format appropriate for machine learning methods, the target variable (WeightCategory) was encoded using Label Encoding:

- Insufficient_Weight $\rightarrow$ 0
- Normal_Weight $\rightarrow$ 1
- Overweight_Level_I $\rightarrow$ 2
- Overweight_Level_II $\rightarrow$ 3
- Obesity_Type_I $\rightarrow$ 4
- Obesity_Type_II $\rightarrow$ 5
- Obesity_Type_III $\rightarrow$ 6

3.2.2 One-Hot Encoding for Categorical Features

To avoid ordinal associations in nominal data, One-Hot Encoding was used to construct binary columns for each category, including gender, family history, FAVC, CAEC, SMOKE, SCC, CALC, and MTRANS.

3.3 Train-Test Separation Strategy

A mixed preprocessing approach was used to guarantee consistent encoding throughout training and testing data:

1. The testing and training sets were momentarily merged.
2. Every encoding transformation was used consistently.
3. Data was once again divided into sets for testing and training.
4. This method guarantees encoding consistency and stops data leaks.

3.4 Feature Scaling

To normalize the feature distributions, *StandardScaler* was used to standardize numerical features:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma} \tag{3.1}$$

where each feature's mean is denoted by μ and its standard deviation by σ. This transformation ensures that every numerical feature contributes equally to distance-based algorithms.

3.5 Dimensionality Reduction with PCA

The use of Principal Component Analysis (PCA) was used to investigate potential dimensionality reduction. The goal of the investigation was to determine the ideal number of main components to reduce feature dimensionality and preserve maximum variance. To find the right number of components, the cumulative explained variance ratio was examined.

Chapter 4

Methodology

4.1 Model Selection

To tackle the multi-class obesity classification problem, three unique experimental procedures were developed and put into practice, each examining various preprocessing techniques and model designs.

4.1.1 Approach 1: SVM Pipeline with Dimensionality Reduction

This method focused on getting the data ready and reducing the number of features:

- **Outlier Detection**: Took out about 10% of unusual data points using a One-Class SVM.
- **Feature Selection**: Used Random Forest to rank features and kept only the most important ones that made up 95% of the total importance.
- **Feature Scaling**: Scaled all numerical features using *StandardScaler*.
- **Dimensionality Reduction**: Applied PCA to keep 95% of the data's information, giving 11 main components.
- **Models**: Tested Random Forest and XGBoost classifiers with and without PCA to check how reducing features affected performance.

4.1.2 Approach 2: Random Forest with Optuna Optimization

This method improves the model and uses multiple models together:

- **Preprocessing**: Converted the target into numbers and split categorical features into separate columns so the model can use them
- **Hyperparameter Tuning**: Used Optuna with Bayesian optimization and five-fold cross-validation to find the best model settings
- **Model**: Trained a Random Forest classifier using these best settings
- **Ensemble Strategy**: Combined the top 5 trained models using majority voting to make more accurate predictions

4.1.3 Approach 3: XGBoost with Optuna

This method combined sophisticated optimization with gradient boosting:

- **Data Preparation**: Changed categories into numbers so the model can understand them.
- **Finding the Best Settings**: Used Optuna to try different settings and picked the ones that worked best, checking the results in three different groups of data.
- **Model Training**: Trained an XGBoost model with the best settings.
- **Ensemble Strategy**: The top five models were selected, and their probabilities were averaged.

4.2 Hyperparameter Tuning

4.2.1 Optuna Framework

Using Optuna, Bayesian optimization was used to effectively search the hyperparameter space:

- **Objective Function**: Increasing the accuracy of cross-validation
- **Search Strategy**: Parzen Estimator sampler with tree structure (TPE)
- **Evaluation**: To guarantee reliable performance estimation, use stratified K-Fold cross-validation.

4.2.2 Parameter Search Spaces

Random Forest

n estimators	100-500
max depth	10-30
min samples split	2-20
min samples leaf	1-20

XGBoost

n_estimators	200-900
max_depth	4-10
learning_rate	0.0015-0.08
subsample	0.55-0.95
colsample_bytree	0.55-0.95
min_child_weight	1-10
gamma	0.0-1.0
reg_lambda	0.0-5.0
reg_alpha	0.0-5.0

4.3 Training Strategy

4.3.1 Cross-Validation Strategy

To handle class imbalance, stratified K-Fold cross-validation was used.

- **Random Forest**: 5-fold stratification
- **XGBoost**: 3-fold stratification with early stopping
- **Stratification**: Preserved class distribution across folds

4.3.2 Ensemble Methods

Majority Voting (Random Forest)

$$\text{Final Prediction} = \text{mode}(\text{model}_1(x), \text{model}_2(x), \ldots, \text{model}_5(x)) \tag{4.1}$$

Probability Averaging (XGBoost)

$$\text{Final Probability} = \frac{1}{5}\sum_{i=1}^{5} P_i(x) \tag{4.2}$$

4.3.3 Data Splitting

- **Training**: For train the model, use 70% of data.
- **Validation**: Use 15% of the data to adjust the model.
- **Testing**: Use 15% of the data to test how well the model works.
- **Stratification**: Make sure each data split has the same ratio of target classes.

4.3.4 Handling Class Imbalance

- **Stratified Sampling**: Kept the class proportions the same in every data split.
- **Class Weights**: Used inverse frequency weights during XGBoost training.
- **Evaluation Metrics**: Measured accuracy with cross-validation to check for bias.

Chapter 5

Experiments and Results

5.1 Experimental Setup

Python 3.8 with scikit-learn 1.2, XGBoost 1.7, and Optuna 3.2 were used for all studies. For repeatability, train-test splits with random state 42 were used to assess each strategy.

5.1.1 Evaluation Metrics

- **Primary Metric**: Multi-class classification accuracy
- **Cross-Validation**: K-Fold stratified with constant random states
- **Statistical Significance**: Several trials using various seeds

5.2 Hyperparameter Tuning Results

5.2.1 Random Forest Optimization

Optuna used five experiments to determine the ideal parameters:

- **Best CV Score**: 0.90234
- **Optimal Parameters**: n_estimators=180, max_depth=18, min_samples_split=8, min_samples_leaf=20

5.2.2 XGBoost Optimization

Comprehensive search over 150 trials yielded:

- **Best CV Score**: 0.90780
- **Optimal Parameters**: n_estimators=864, max_depth=7, learning_rate=0.0161, subsample=0.8795, colsample_bytree=0.5662, gamma=0.4033, reg_lambda=1.8098, reg_alpha=1.6319

5.2.3 Top Ensemble Models

Random Forest Ensemble Members

Top 5 Optuna Trials used in ensemble:

Model 1:

- CV Score: 0.893067
- Parameters: n_estimators=507, max_depth=16, min_samples_split=8, min_samples_leaf=4, max_features=None

Model 2:

- CV Score: 0.892809
- Parameters: n_estimators=454, max_depth=16, min_samples_split=6, min_samples_leaf=5, max_features=None

Model 3:

- CV Score: 0.886242
- Parameters: n_estimators=329, max_depth=13, min_samples_split=5, min_samples_leaf=3, max_features=sqrt

Model 4:

- CV Score: 0.886178
- Parameters: n_estimators=469, max_depth=18, min_samples_split=8, min_samples_leaf=4, max_features=sqrt

Model 5:

- CV Score: 0.885341
- Parameters: n_estimators=351, max_depth=20, min_samples_split=10, min_samples_leaf=5, max_features=sqrt

XGBoost Ensemble Members

Top 5 Optuna Trials used in ensemble:

Model 1:

- CV Score: 0.907809
- Parameters: n_estimators=864, max_depth=7, learning_rate=0.016198, subsample=0.879524, colsample_bytree=0.566258, min_child_weight=4, gamma=0.403388, reg_lambda=1.809858, reg_alpha=1.631961

Model 2:

- CV Score: 0.907552

- Parameters: n_estimators=790, max_depth=7, learning_rate=0.016689, subsample=0.841185, colsample_bytree=0.550413, min_child_weight=4, gamma=0.478953, reg_lambda=1.888155, reg_alpha=1.369602

Model 3:

- CV Score: 0.907487
- Parameters: n_estimators=887, max_depth=7, learning_rate=0.020252, subsample=0.858392, colsample_bytree=0.559423, min_child_weight=4, gamma=0.500167, reg_lambda=1.570832, reg_alpha=1.714748

Model 4:

- CV Score: 0.907487
- Parameters: n_estimators=326, max_depth=9, learning_rate=0.026616, subsample=0.669123, colsample_bytree=0.641182, min_child_weight=6, gamma=0.530148, reg_lambda=1.305350, reg_alpha=0.298889

Model 5:

- CV Score: 0.907423
- Parameters: n_estimators=400, max_depth=7, learning_rate=0.028629, subsample=0.630873, colsample_bytree=0.596411, min_child_weight=5, gamma=0.506369, reg_lambda=2.125629, reg_alpha=0.620831

5.3 Model Performance

5.3.1 Approach 1: SVM Pipeline Results

Configuration	Random Forest	XGBoost
With PCA (11 components)	65.13%	75.30%
Without PCA (original features)	84.57%	86.70%

Table 5.1: Performance comparison with and without PCA

5.3.2 Approach 2: Random Forest Pipeline

- **Test Accuracy**: 90.385%
- **Ensemble Holdout Accuracy**: 90.30%

5.3.3 Approach 3: XGBoost Pipeline

- **Test Accuracy**: 91.212%
- **Ensemble Holdout Accuracy**: 90.25%

5.4 Comparative Analysis

5.4.1 Overall Performance Ranking

1. **XGBoost Pipeline**: 91.212% (Best performing)
2. **Random Forest Pipeline**: 90.385% (Close second)
3. **SVM Pipeline (without PCA)**: 86.70% (XGBoost), 84.57% (Random Forest)
4. **SVM Pipeline (with PCA)**: 75.30% (XGBoost), 65.13% (Random Forest)

5.4.2 Key Observations

- **Optuna Optimization**: Significant improvement over the default parameters.
- **Ensemble Benefits**: Predictions from the top five model ensembles increased accuracy and improved model generalization.
- **PCA Impact**: In both approaches, dimensionality reduction reduced models' performance.
- **Algorithm Comparison**: Under optimal settings, XGBoost performed marginally better than Random Forest.

5.4.3 Feature Importance Analysis

The importance of features using the Random Forest model is shown below :

- **Weight**: 39.6%
- **Age**: 13.9%
- **Gender features**: 9.2%
- **FCVC**: 6.3%
- **FAF**: 4.3%
- **NCP**: 3.9%
- **CH2O**: 3.3%
- **Height**: 0%

Chapter 6

Discussion

6.1 Performance Interpretation

Using Random Forest pipeline came in second place with an accuracy of **90.385%**. This illustrates how tree-based ensemble approaches can be used to address this issue. Both enhanced ensemble methods significantly outperformed the basic SVM pipeline, with an accuracy of **86.70%** without PCA. This emphasizes how important it is to use ensemble methods and modify hyperparameters.

The high accuracy rates of all the methods attest to how effective the feature engineering and selection processes are. This demonstrates how the selected traits capture the connections between lifestyle factors and weight categories.

6.2 Feature Importance Analysis

Chapter 3's feature-importance results (Random Forest — mean decrease in impurity) revealed several clear, intuitive patterns. **Weight** was the strongest predictor — it directly shows how we define the weight categories. Feature **Age** was also important.

Everyday habits mattered: **NCP** (meal patterns), **FAF** (frequency of physical activity), and **FCVC** (frequency of vegetable consumption) all influenced the model's predictions. By contrast, **Height** contributed very little compared with weight.

Overall, the results suggest that current lifestyle choices in this dataset are more closely associated with weight status than inherited factors — family history features showed lower importance than several behavioral variables. It is worth remembering that feature importance indicates association, not causation; nevertheless, these findings point to diet and activity as practical areas for further study or intervention.

6.3 Model Analysis

6.3.1 Impact of Dimensionality Reduction

We applied PCA and saw a noticeable decline in performance, so it's important to understand why.

Performance drop (when using PCA)

- **Model performance**: After PCA, accuracy for both tree-based algorithms fell sharply from about 86% down to roughly 65–75%.
- **Loss of information**: PCA's linear projections appear to have discarded non-linear patterns that the tree models were relying on.

Why this makes sense (theoretical explanation)

- **Non-linear relationships**: Predicting obesity depends on complex, non-linear interactions among variables; with pca there is chances that model can't capture those relationships effectively by new features.
- **Dimensionality isn't the problem here**: With only about 30 features after encoding, the "curse of dimensionality" is not the main issue , preserving meaningful structure is more important than reduce features.

6.3.2 Ensemble Strategy Effectiveness

The impact of ensemble techniques was clearly reflected in the model performance, highlighting their role in achieving more stable and accurate results.

Diversity in Ensemble Models

- **Variation in Parameters**: The top-performing models used a variety of hyperparameter settings still provided strong results.
- **Complementary Strengths**: Different parameter combinations allowed the models to capture unique patterns and nuances within the data.
- **Improved Stability**: Compared with any single model, the ensemble provided better generalization and reduced variance, giving more reliable predictions.

Advantages of Optuna

- **Efficient Search**: Large hyperparameter spaces were handled efficiently using Bayesian optimization.
- **Increase Performance**: This systematic hyperparameter tuning led to improvement over the models' default hyperparameter configurations.

6.3.3 Analyzing SVM to Remove Outliers

Results from the One-Class SVM outlier reduction were not consistently reliable.

Benefits

- **Data Cleaning**: The method removed potentially noisy data, affecting about 10% of the dataset.
- **Training Stability**: Reducing noise helped the model train more smoothly and reliably.

Limitations

- **Information Loss**: Some real minority-class samples might have been removed by mistake.
- **Impact on Class Balance**: Removing outliers could have increased existing class imbalances.
- **Observations for Synthetic Data**: Outlier detection may be less accurate or not suitable for artificially generated datasets.

6.4 Impact of Synthetic Data

Since the data are generated synthetically, some components may be affected:

6.4.1 Data Quality Considerations

- **Privacy Protection**: We can analyze the data carefully while keeping sensitive health information secure.
- **Statistical Properties**: Keep the key patterns from the original data so the dataset still represents the real-world situation for modeling.

6.4.2 Modeling Implications

- **Characteristics of Outliers**: Unusual data points can reduce the performance of algorithm and affect their reliability.
- **Generalization**: When appropriate generation techniques are applied, models trained on synthetic data can generalize effectively to real-world datasets.

6.4.3 Limitations and Considerations

- **Unusual Trends**: Patterns from classes with low priority may affect model learning.
- **Pre-processing Sensitivity**: Pre-processing techniques might affect synthetic data differently from real data and potentially change the model performance.

6.4.4 Recommendations for Synthetic Data Usage

- **Feature Engineering**: We can choose features based on domain knowledge instead of machine learning model predictions.
- **Model Selection**: Select models that perform well on unexpected patterns that can effectively handle synthetic data generation.
- **Validation**: We can include validation against real-world data through techniques like cross-validation.

hyperref

Chapter 7

GitHub Repository

The complete source code, datasets, and resources for this project are available in our GitHub repository:

`github.com/jenish11052004/obesity-risk-project`

This repository contains all scripts, documentation, and materials necessary to reproduce our analysis and results.