AI511: Course Project 1

Subject: AI511 — Machine Learning

Topic: Obesity Disease Risk Prediction Using Machine Learning

Submitted By:

Vibhuti Jain [MT2025731]

Meenal Hirwani [MT2025071]

Institution: International Institute of Information Technology Bangalore

Submission Date: October 26, 2025

GitHub Link:

`https://github.com/vibhuvj27/AIT-511---ML_Project`

International Institute of Information Technology Bangalore

Abstract

This project report offers a thorough examination of how different machine learning algorithms can be used to predict a person's weight category. The notable class disparity, which frequently lowers the prediction accuracy for underrepresented categories, is one of the main issues this study addresses. One-hot encoding for categorical variables, feature standardization, and outlier identification using the Interquartile Range (IQR) method were among the preprocessing steps in the workflow. KNearestClassifier, Decision Tree, Random Forest, and XGBoost were among the various classification techniques that were assessed Among these, XGBoost demonstrated the highest performance and resilience to imbalanced data. Further improvements were achieved through hyperparameter optimization, while oversampling techniques such as RandomOverSampler did not yield any improvements

The optimized model achieved a validation accuracy of 91.32%, highlighting the critical role of data preprocessing, feature engineering in multi-class classification problems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This project aims to predict with high accuracy the weight category of a person, including Insufficient Weight, Normal Weight, Overweight Levels I and Overweight Levels II, and Obesity Types I–III, and formulate the task as a multi-class classification problem. One major difficulty in this study is the class imbalance, which is characterized by the fact that some weight categories have many fewer samples than others, causing predictive models to favor dominant classes while performing poorly on minority classes. Additionally, the numerical features height and weight may have outliers that will disrupt scaling and prevent models from converging appropriately.

The proposed methodology is organized into three main phases:

1. **Data Preprocessing:** Detection and removal of outliers using the Interquartile Range (IQR) method, application of one-hot encoding for categorical attributes, and standardization of numerical variables.

2. **Model Selection:** A comparative assessment of classifiers such as Decision Tree, Random Forest, and XGBoost to determine the most effective model.

3. **Hyperparameter Optimization:** Systematic tuning of the selected model's hyperparameters using techniques like random search and optuna to maximize predictive accuracy and generalization.

4. **Addressing Class Imbalance:** Check of oversampling strategy including Random Oversampling to enhance the model's ability to accurately predict minority categories.

Chapter 2

Exploratory Data Analysis (EDA)

The exploratory data analysis (EDA) uncovers several important patterns in the dataset: `WeightCategory`.

Train Data - Summary Statistics for Numerical Features:

	Age	Height	Weight	FCVC	NCP	CH2O	FAF	TUE
count	15533.000000	15533.000000	15533.000000	15533.000000	15533.000000	15533.000000	15533.000000	15533.000000
mean	23.816308	1.699918	87.785225	2.442917	2.760425	2.027626	0.976968	0.613813
std	5.663167	0.087670	26.369144	0.530895	0.706463	0.607733	0.836841	0.602223
min	14.000000	1.450000	39.000000	1.000000	1.000000	1.000000	0.000000	0.000000
25%	20.000000	1.630927	66.000000	2.000000	3.000000	1.796257	0.007050	0.000000
50%	22.771612	1.700000	84.000000	2.342220	3.000000	2.000000	1.000000	0.566353
75%	26.000000	1.762921	111.600553	3.000000	3.000000	2.531456	1.582675	1.000000
max	61.000000	1.975663	165.057269	3.000000	4.000000	3.000000	3.000000	2.000000

Figure 2.1: Summary Statistics Of train data

Test Data - Summary Statistics for Numerical Features:

	Age	Height	Weight	FCVC	NCP	CH2O	FAF	TUE
count	5225.000000	5225.000000	5225.000000	5225.000000	5225.00000	5225.000000	5225.000000	5225.000000
mean	23.917600	1.701216	88.192612	2.454802	2.76403	2.034746	0.995953	0.625505
std	5.761355	0.086240	26.410210	0.540020	0.70219	0.610672	0.842551	0.601761
min	14.000000	1.463167	39.101805	1.000000	1.00000	1.000000	0.000000	0.000000
25%	20.000000	1.632983	66.500000	2.000000	3.00000	1.785804	0.015860	0.000000
50%	22.875223	1.700000	85.000000	2.499388	3.00000	2.000000	1.000000	0.582840
75%	26.000000	1.761519	111.720238	3.000000	3.00000	2.559750	1.624981	1.000000
max	55.246250	1.947406	165.057269	3.000000	4.00000	3.000000	3.000000	2.000000

Figure 2.2: Summary Statistics of test data

Table 2.1: EDA Table: Class Distribution and Significance

Weight Category	**Distribution Status**	**Key Significance**
Normal Weight	Majority Class	If class imbalance is ignored, the model may develop a bias towards this group.
Overweight Level I & II	High Counts	Although substantial in number, careful classification remains important.
Obesity Type I & II	Medium Counts	These are relevant groups that still benefit from balancing methods.
Insufficient Weight	Lower Count	Represents a clear minority class, requiring attention in modeling.
Obesity Type III	Extreme Minority Class	Very low representation, necessitating oversampling techniques like Random oversampling.

Key Insights from the EDA -

- **Weight Distribution:** The dataset exhibits various obesity levels with differing prevalence rates.
- **Gender Patterns:** Notable differences in weight distribution between males and females.
- **Age Impact:** A clear relationship between age and weight categories.
- **Family History:** Strong association between family history of overweight and current weight status.
- **Lifestyle Factors:**
  - Physical activity levels across weight categories.

  - Eating habits (high caloric food, vegetable consumption, snacking).
  - Transportation modes (active vs. passive).

- **Behavioral Factors:** Smoking habits, alcohol consumption, water intake.
- **Correlations:** Relationships between numerical features like height, weight, age, and behavioral factors.

The visualizations reveal specific patterns, including:

- Which weight categories are most prevalent.
- How lifestyle factors differ across weight groups.
- Which demographic and behavioral factors have the strongest associations with obesity levels.
- Potential risk factors for different weight categories.

This comprehensive EDA provides a solid foundation for understanding the factors influencing weight categories and can guide further analysis or predictive modeling.

2.1 Importance of Visualization

The visual representation highlights underrepresented categories, particularly *Obesity Type III* and *Insufficient Weight*, reinforcing the need for oversampling to balance the dataset.

2.2 Limitations of EDA

- Insights depend on raw data scale and outlier treatment.
- Analyst bias and feature selection may influence interpretation.
- EDA is descriptive and cannot guarantee predictive performance.
- Subtle overlaps near decision boundaries may remain undetected.

2.3 Observations from Data

The dataset exhibits strong class imbalances, outliers. Based on the comprehensive EDA of this obesity dataset, several key patterns emerge that highlight the multifactorial nature of weight management. The data reveals strong familial influences, with individuals having a family history of overweight showing significantly higher representation in obesity categories, suggesting genetic predisposition plays a crucial role. Age distribution patterns indicate that while weight issues affect all age groups, certain obesity types cluster in specific life stages, possibly reflecting cumulative lifestyle impacts. Clear behavioral patterns emerge across weight categories - individuals in higher weight groups consistently report lower physical activity frequency, higher consumption of high-calorie foods, and more frequent eating between meals. Transportation choices show striking correlations, with active methods like walking and biking associated with healthier weight categories, while sedentary transportation aligns with higher weight groups. Gender differences are pronounced, with distinct distribution patterns between males and females across various weight categories. Dietary habits further differentiate the groups, with lower vegetable consumption and higher caloric food intake

more prevalent in obesity categories. Interestingly, monitoring calorie consumption is substantially less common among individuals in higher weight categories, suggesting awareness and tracking behaviors may play a protective role. These observations collectively emphasize that weight management involves a complex interplay of genetic factors, daily activity choices, dietary patterns, transportation habits, and behavioral awareness rather than any single determining factor.

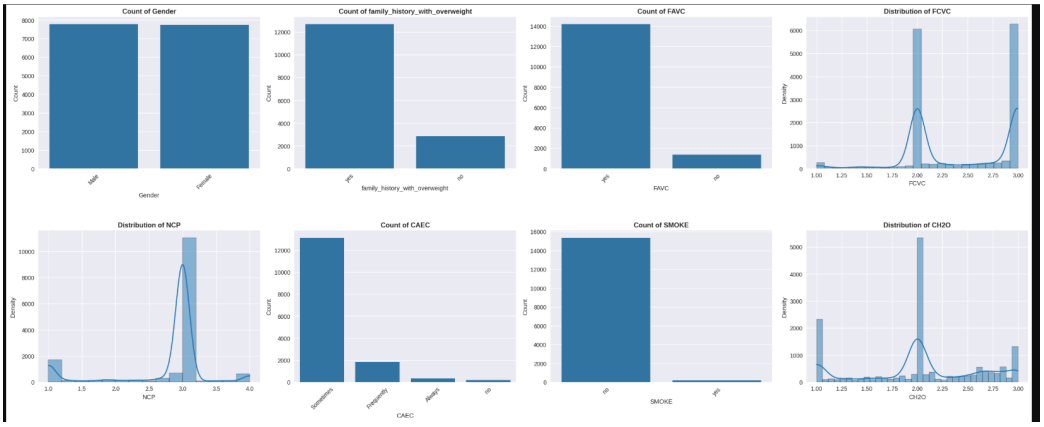

Figure 2.3: Observation Of train data

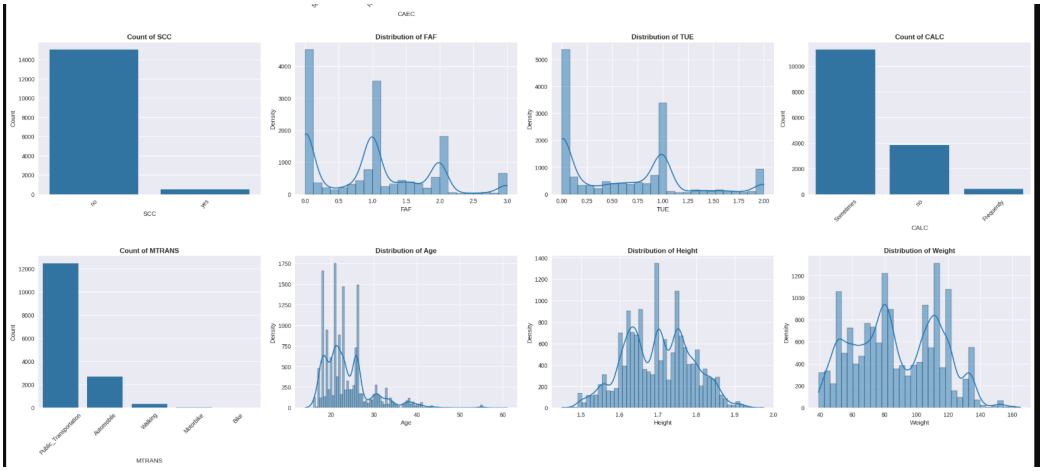

Figure 2.4: Continued Observation Of train data

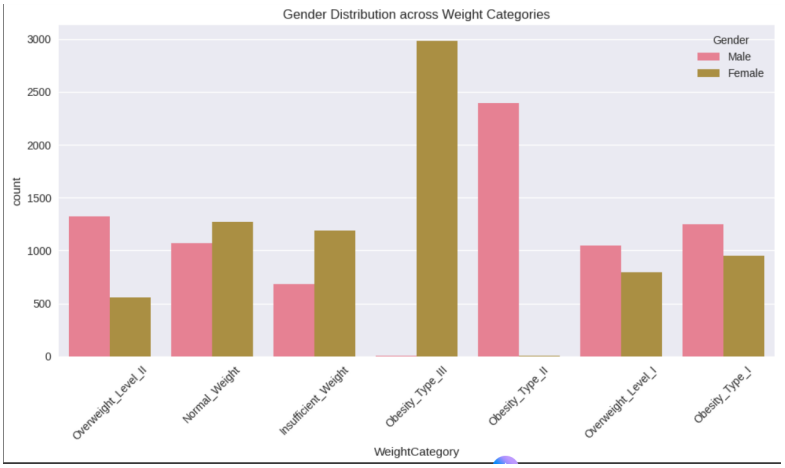

Figure 2.5: Gender Distribution across WeightCategory

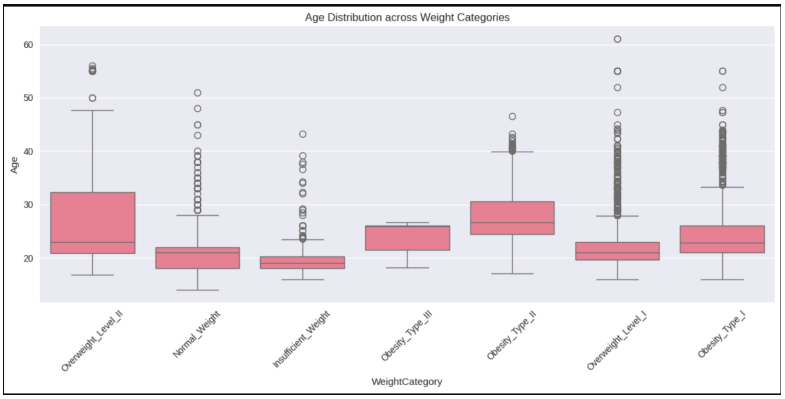

Figure 2.6: Age Distribution across WeightCategory

Figure 2.7: Height vs Weight by WeightCategory

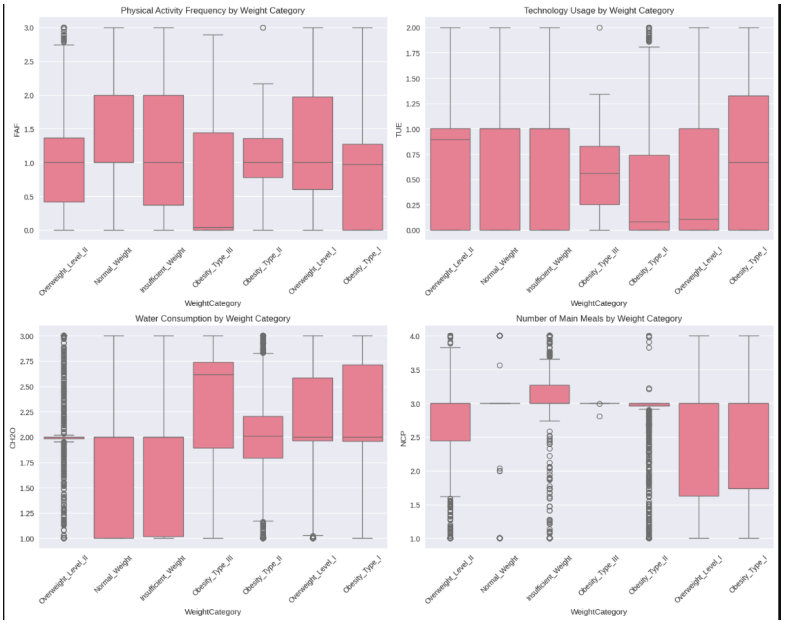

Figure 2.8: Other Plots

Figure 2.9: Smoking Habits by WeightCategory

Figure 2.10: Weight vs Age by Gender and Family History

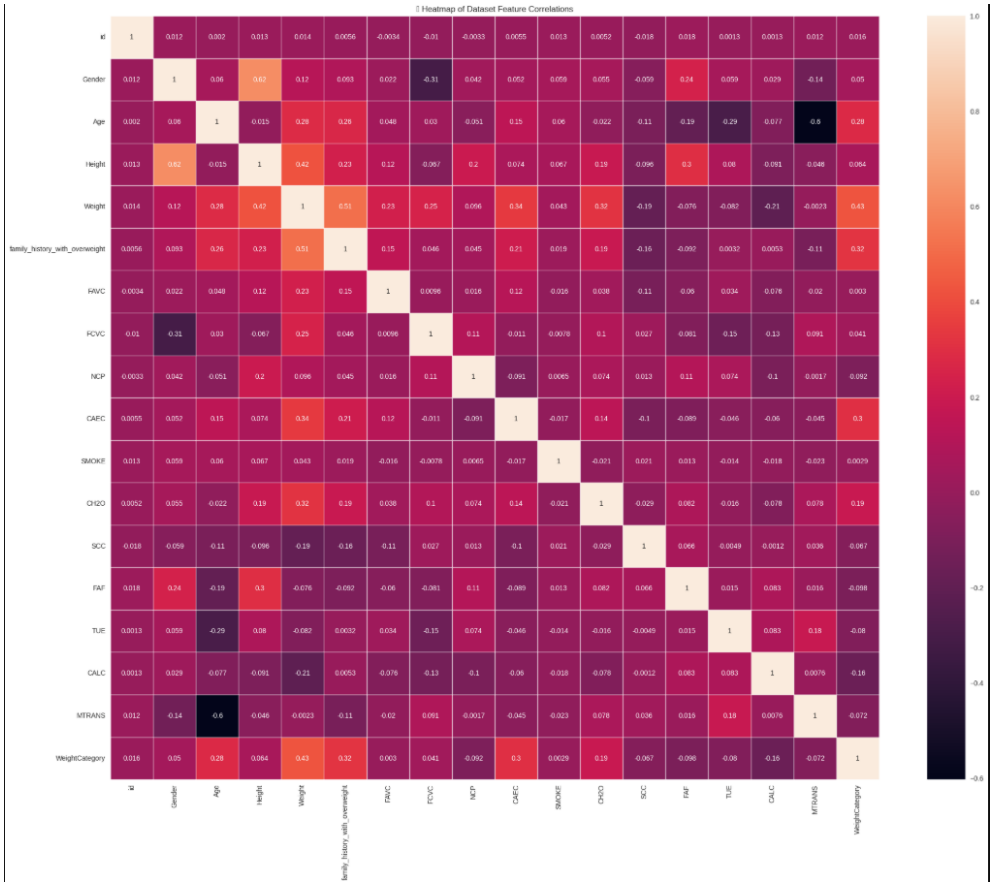

Figure 2.11: Correlation plot of train dataset

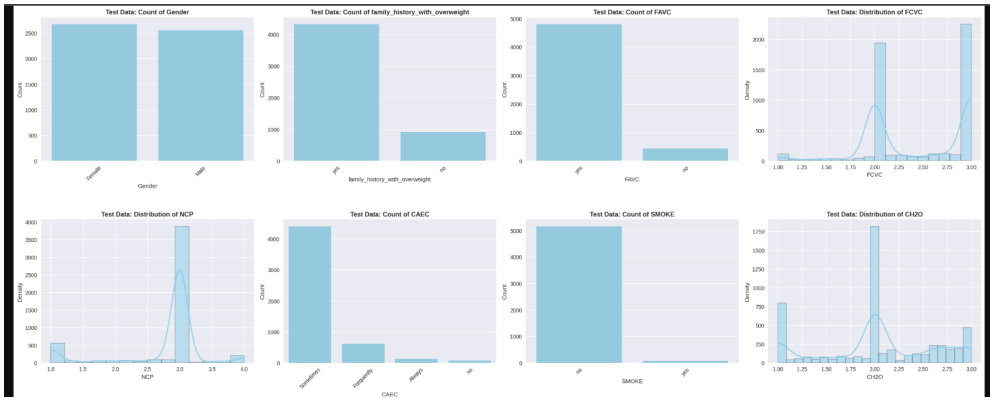

Figure 2.12: Observation of test data

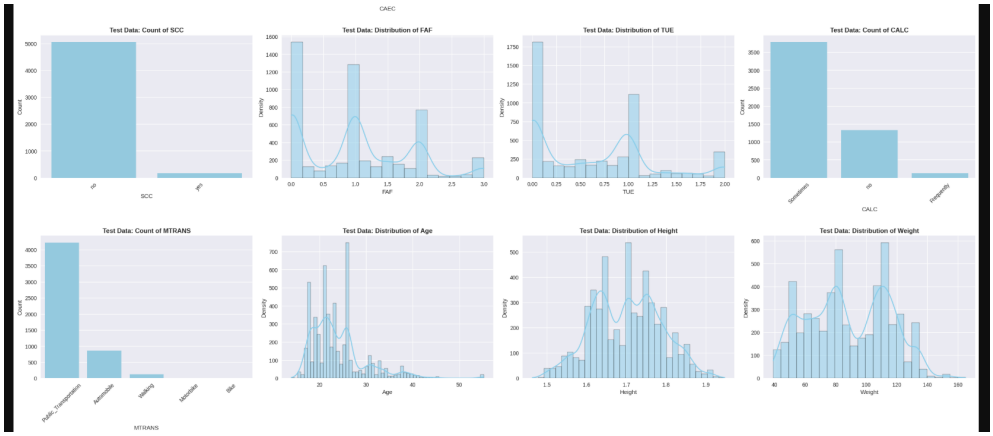

Figure 2.13: Continued Observation of test data

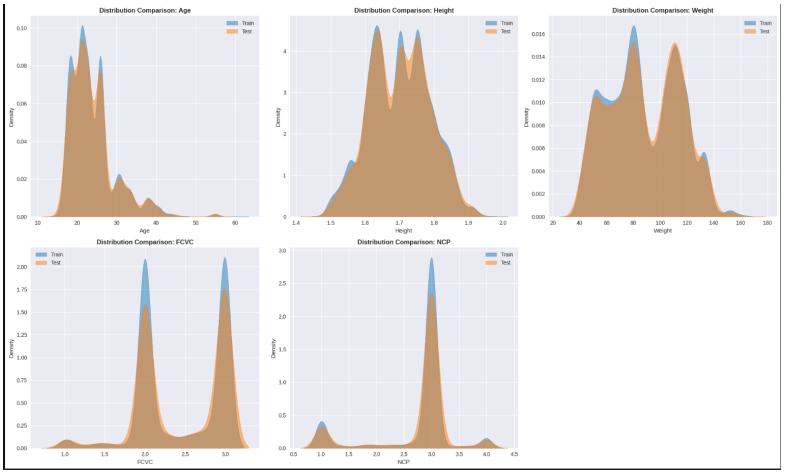

Figure 2.14: Comparison between numeric features of train and test data

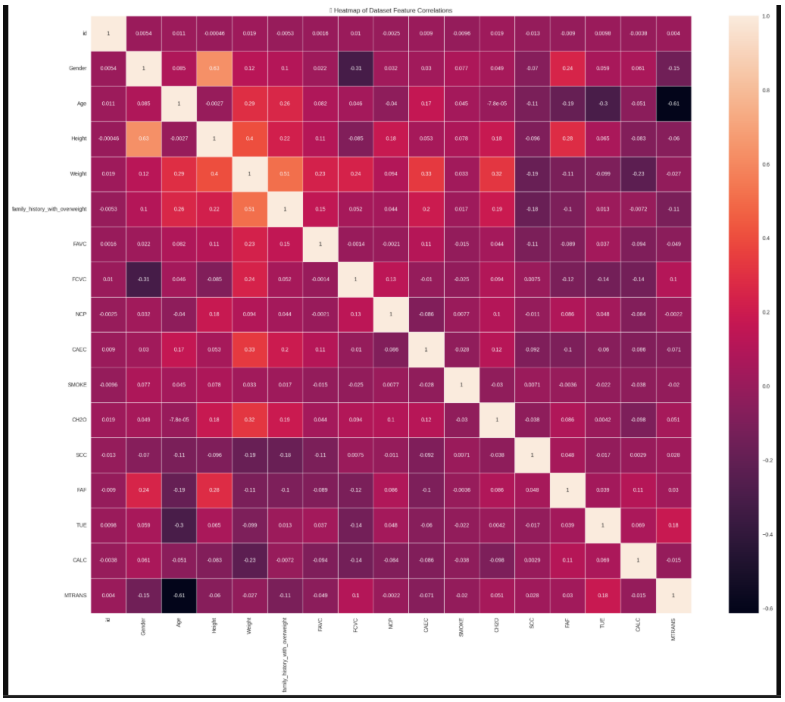

Figure 2.15: Correlation plot of test dataset

Chapter 3

Data Pre-Processing

This section describes the structured pre-processing workflow implemented on both the training and test datasets. The objective was to convert raw features into a numerical, standardized format compatible with XGBoost, while simultaneously addressing the multi-class imbalance problem.

- **Data Loading and Target Assessment:** The training and testing datasets were imported, with test set IDs stored for final submission mapping. Initial examination confirmed notable class imbalances in the target variable.

- **Feature Separation:** The target column representing weight categories was separated from the features to facilitate model training.

- **Outlier Detection:** This dataset analysis employs a sophisticated two-stage approach to identify and manage unusual data points that could distort machine learning models. The method first detects statistical outliers in numerical features using inter-quartile range calculations, then applies intelligent decision rules based on feature importance and outlier prevalence. Features are evaluated through a dual-threshold

system that considers both their relationship to the target variable and the extent of outlier contamination. When outliers exceed 3% of observations, the system takes differentiated action: features demonstrating strong predictive value are preserved through median imputation, while those with weak target relationships are entirely removed to prevent noise introduction. This selective treatment balances data quality maintenance with feature retention, ensuring valuable predictive signals aren't lost during cleaning. The approach maintains consistency across training and evaluation datasets by applying identical transformations, which is critical for model reliability. The process is fully automated yet transparent, providing detailed documentation of all decisions made during the cleaning phase. This methodology effectively addresses the common challenge of outlier management in predictive modeling while preserving dataset integrity and model performance potential. Outlier mitigation was carried out using the Interquartile Range (IQR) technique, which establishes acceptable boundaries for numerical features and flags extreme values for adjustment. This process was confined to the training dataset to prevent information leakage. The steps are as follows:

- **Quartile Calculation:** Determine the 25th percentile (Q1) and 75th percentile (Q3) for each numerical feature.
- **IQR Computation:** Calculate the interquartile range:

$$\text{IQR} = Q3 - Q1$$

- **Boundary Definition:** Upper and lower bounds are determined

using a standard multiplier (typically 1.5):

$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR}$$

$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR}$$

– **Outlier Handling:** Data points outside these bounds are designated as outliers.If number of outliers are above 3% and with low correlation with target, these were dropped and if the number of outliers are above 3% and with high correlation with target, values were replaced with median values, reducing their influence on scaling while retaining the data point.

```
============================================================
OUTLIER DETECTION AND HANDLING (>3% THRESHOLD)
============================================================

Checking 8 numerical columns for outliers (threshold: 3.0%)...
  Age: 792 outliers (5.1%) REPLACED (corr: 0.276)
  Height: 4 outliers (0.0%) KEPT (corr: 0.064)
  Weight: No outliers (corr: 0.430)
  FCVC: No outliers (corr: 0.041)
  NCP: 4548 outliers (29.3%) DROPPED (corr: -0.092)
  CH2O: No outliers (corr: 0.188)
  FAF: No outliers (corr: -0.098)
  TUE: No outliers (corr: -0.080)

Dropped 1 columns with low correlation and >3.0% outliers: ['NCP']
Modified 1 columns with high correlation and >3.0% outliers: ['Age']
```

Figure 3.1: Outliers

- **Categorical Feature Encoding and Alignment:** To manage categorical variables and maintain consistent feature spaces between training and test sets, One-Hot Encoding was applied. The `drop_first=True` parameter was used to remove the first category of each feature, mitigating multicollinearity risks. After encoding, feature alignment en-

sured that both sets shared identical columns in the same order, preventing prediction errors. The `id` column was removed from both sets as it carries no predictive information.

- **Target Variable Processing:** `LabelEncoder` converts the nominal target variable to a numerical format, allowing the XGBClassifier to interpret multi-class labels numerically. The encoder was preserved for inverse-transforming predictions back into their original category names.

- **Feature Standardization:** Scaling parameters were computed solely on the training data. Validation and test sets were transformed using these precomputed parameters, a step crucial to avoid data leakage.

- **Data Splitting:** A dedicated split of the training data was performed for model validation and for comparing, dividing it into 80% temporary training and 20% validation sets.

Chapter 4

Model Selection

The final model was chosen through a two-phase procedure which involves comparative evaluation of multiple candidate algorithms. This stage of the project involved evaluating a range of high-performing classification algorithms to determine the most appropriate model for the characteristics of the dataset.

4.1 XGBoost (Extreme Gradient Boosting)

XGBoost is recognized for its exceptional predictive performance, robustness to class imbalance, and built-in regularization mechanisms. Preliminary testing indicated that it achieved superior accuracy and learning capability compared to alternative tree-based models.

Additive Prediction Mechanism

For a given data point (x_i), the final predicted weight score category ($\hat{y}_i$) is obtained by summing the outputs of all T sequentially constructed decision

trees:

$$\hat{y}_i = \sum_{k=1}^{T} f_k(x_i)$$

Here, f_k represents the k^{th} tree in the sequence.

Incremental Learning

At iteration (t), the newly trained tree f_t refines the previous prediction $\hat{y}_i^{(t-1)}$:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \epsilon \cdot f_t(x_i)$$

where ϵ is the learning rate, set to 0.05 for this project.

Objective Function

The total objective function $L^{(t)}$ at iteration (t) combines a loss term and a regularization term:

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \Omega(f_t)$$

The loss term $l(y_i, \hat{y}_i^{(t)})$ measures the deviation between the true label y_i and the predicted value $\hat{y}_i^{(t)}$. The regularization component $\Omega(f_t)$ penalizes tree complexity to prevent overfitting:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 + \alpha \sum_{j=1}^{T} |w_j|$$

Here, γ penalizes the number of leaves, λ applies L2 regularization, and α applies L1 regularization. This combination ensures that the model generalizes well, even with imbalanced classes and the high-dimensional feature space resulting from one-hot encoding.

Key Advantages of XGBoost:

- Built-in handling of missing values and sparse features.
- Regularization prevents overfitting and improves generalization.
- Parallel computation and pruning reduce computational cost.
- Works effectively for both regression and classification tasks.

4.2 Random Forest

Random Forest is an ensemble approach that reduces variance and typically provides strong generalization. While competitive, its preliminary accuracy was slightly lower than that of XGBoost.

For multi-class predictions, Random Forest relies on majority voting:

$$\hat{Y}(x_i) = \arg\max_{c \in C} \sum_{k=1}^{T} I(h_k(x_i) = c)$$

Here, $\hat{Y}(x_i)$ denotes the predicted class for instance x_i, T is the total number of trees, and I is an indicator function that counts votes for each class.

Advantages:

- Handles non-linear data effectively.
- Resistant to overfitting due to averaging across multiple trees.
- Provides feature importance for interpretability.

Limitations:

- Less efficient for large-scale or high-dimensional data.
- Requires more memory and computation compared to simpler models.

4.3 Decision Tree

Decision Trees are interpretable base learners but often exhibit high variance and are therefore prone to overfitting on complex data.

Splits are determined based on **Gini Impurity**:

$$IG(m) = 1 - \sum_{j=1}^{J} p_j^2$$

where J is the total number of classes and p_j is the proportion of class j in node m.

The Gini Gain for a split from a parent node (P) into left (L) and right (R) children is:

$$G = IG(P) - \frac{N_L}{N_P} IG(L) - \frac{N_R}{N_P} IG(R)$$

where N_L, N_R, and N_P are the sample counts of left, right, and parent nodes, respectively.

Advantages:

- High interpretability and simplicity.
- Handles both numerical and categorical data.

Limitations:

- Sensitive to small data variations.
- High variance and prone to overfitting.

4.4 K-Nearest Neighbors (KNN) Algorithm

KNeighbors is a supervised learning algorithm available in `sklearn.neighbors`, used for both classification and regression tasks. It is based on the idea that similar data points exist close to each other in feature space. When predicting, the algorithm:

1. Calculates the distance (commonly Euclidean) between the new data point and all training samples.
2. Selects the K nearest neighbors (smallest distances).
3. For classification: predicts the majority class among those neighbors.
4. For regression: takes the average value of the neighbors.

It's a non-parametric, instance-based (lazy) learning algorithm — meaning it makes no assumption about data distribution and does not learn a model during training. Instead, it "learns" at prediction time.

Advantages

- **Simple and Intuitive** – Easy to understand and implement; based on the natural concept of similarity.
- **No Training Phase** – It's a lazy learner, so it doesn't require model training; suitable for small datasets.
- **Non-Parametric** – Makes no assumptions about data distribution; works well with complex, non-linear boundaries.
- **Adaptable to Multi-Class Problems** – Can easily handle multiple classes in classification tasks.

- **Versatile Distance Metrics** – Can use Euclidean, Manhattan, Minkowski, cosine, etc., depending on the problem.
- **Can Capture Local Patterns** – Performs well when local data relationships are important.

Disadvantages

- **Computationally Expensive** – Requires distance computation to all points during prediction → slow for large datasets.
- **Memory Intensive** – Needs to store all training data, increasing space complexity.
- **Sensitive to Irrelevant/Scaled Features** – Feature scaling (like normalization or standardization) is essential.

Choice of K Affects Performance –

1. Small K → noisy overfitting
2. Large K → biased underfitting

Curse of Dimensionality – As the number of features increases, distance metrics become less meaningful. Imbalanced Data Issue – If one class dominates, KNN may get biased toward that class.

Between all four models, XGBoost consistently produced the highest accuracy and was therefore selected as the final model for this project.

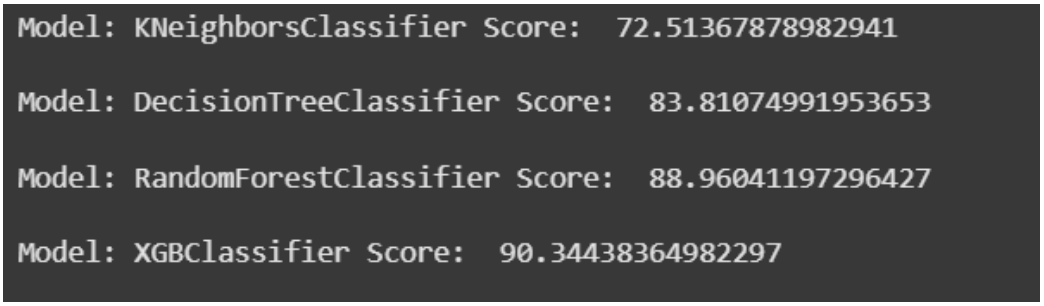

Figure 4.1: Comparison

Chapter 5

Hyperparameter Tuning and Data Strategy Optimization

5.1 Hyperparameter Tuning

Our machine learning workflow implements a systematic hyperparameter optimization strategy for an XGBoost classification model using randomized search with cross-validation to get approx hyperparameters, then optuna framework for more detailed correct hyperparameters. The approach explores a comprehensive parameter space encompassing nine key model configuration aspects given in image below -

```python
param_search = {
    'colsample_bytree': np.linspace(0.1, 1.0, 5),
    'gamma': np.linspace(0.0, 1.0, 5),
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'min_child_weight': [3, 5, 7],
    'n_estimators': [200, 300, 350],
    'reg_alpha': np.linspace(0.5, 1.0, 6),
    'reg_lambda': np.linspace(2.0, 3.0, 6),
    'subsample': np.linspace(0.5, 1.0, 6)
}
```

Figure 5.1: RandomSearch Parameters

```python
    params = {
        'colsample_bytree': trial.suggest_float('colsample_bytree', 0.55, 0.775),
        'gamma': trial.suggest_float('gamma', 0.25, 0.75),
        'learning_rate': trial.suggest_float('learning_rate', 0.1, 0.2),
        'max_depth': trial.suggest_int('max_depth', 3, 7),
        'min_child_weight': trial.suggest_int('min_child_weight', 3, 7),
        'n_estimators': trial.suggest_int('n_estimators', 300, 350),
        'reg_alpha': trial.suggest_float('reg_alpha', 0.6, 0.9),
        'reg_lambda': trial.suggest_float('reg_lambda', 2.8, 3.0),
        'subsample': trial.suggest_float('subsample', 0.5, 1.0)
    }
```

Figure 5.2: Application on Optuna Framework

5.2 Hyperparameter Configuration

Table 5.1: Best XGBoost Hyperparameters

Parameter	**Value**	**Rationale**
subsample	0.8	The subsample ratio of the training instances.
reg_lambda	2.95	L2 regularization for stability.
reg_alpha	0.7	L1 regularization for weight control.
n_estimators	345	Provides sufficient boosting iterations.
min_child_weight	5	Minimum sum of instance weight.
max_depth	3	Controls complexity to reduce overfitting.
learning_rate	0.175	Ensures stable convergence.
gamma	0.45	Minimum loss reduction for a split.
colsample_bytree	0.6	The fraction of columns to be subsampled.

5.3 Oversampling Technique Comparative Analysis

- **No Oversampling:** No Duplication of minority class samples
- **Random Oversampling (ROS):** Duplicates minority class samples; risk of overfitting.

5.4 Discussion of Performance

Table 5.2: Oversampling Strategies and Accuracy

Strategy	**Description**	**Accuracy**
Baseline	No oversampling applied	90.41
Random Oversampling	Duplicates minority class samples	90.15

Chapter 6

Conclusion

After comprehensive experimentation, the combination of a finely tuned XGBoost model with no oversampling method emerged as the best-performing configuration, achieving a validation accuracy of **91.322%**.

This approach effectively addressed data imbalance and captured complex, non-linear relationships among lifestyle, dietary, and physiological attributes. The model's robustness demonstrates that optimal performance arises from both algorithm selection and holistic data strategy design.

Bibliography

[1] Kaggle Dataset on Obesity Risk Prediction, `https://www.kaggle.com/`.

[2] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," KDD, 2016.

[3] He, H., and Garcia, E. (2009). "Learning from Imbalanced Data." IEEE Transactions on Knowledge and Data Engineering.