Performance Evaluation of Machine Learning Algorithms on Obesity Prediction Dataset

Presented by

Yuvraj Deshmukh MT2025040

Yash Parande MT2025135

Under the Guidance of

Prof. Aswin Kannan
Machine Learning

Department of Computer Science and Engineering
International Institute of Information Technology, Bangalore
2025

1. Problem Overview

Maintaining a healthy lifestyle is becoming increasingly difficult in modern society. This dataset investigates how an individual's weight category (*Insufficient Weight, Normal Weight, Overweight, or Obesity Types I–III*) relates to their demographic information and daily lifestyle habits.

The dataset captures multiple aspects of a person's routine and background that influence body weight. The key feature groups include:

- **Demographic Variables:** Gender, Age, Height
- **Lifestyle Patterns:** Food habits, Physical Activity, Technology Usage (e.g., screen time)
- **Health History:** Family history of overweight or obesity
- **Behavioral Indicators:** Water intake, smoking, and alcohol consumption habits

The objective is to build a machine learning model that accurately classifies individuals into distinct weight categories and identifies hidden trends that contribute to obesity risk. This analysis combines elements of health analytics, behavioral science, and data-driven prediction.

Code Repository: Code repository at: `https://github.com/YuvrajDesh/Xgboost_obesity`

2. Exploratory Data Analysis (EDA)

2.1 Dataset Overview

The dataset comprises information about individuals' demographics, eating habits, physical activity, and other lifestyle-related factors that potentially influence obesity risk. It contains both numerical and categorical attributes. The dataset was split into two parts:

- **Training Set:** 15,533 samples and 18 columns (including the target label).
- **Test Set:** 5,225 samples and 17 columns (without the target label).

Each record represents an individual, described by variables such as age, gender, food consumption frequency, daily activity, and transportation habits. The goal is to predict the target variable **WeightCategory**, which classifies individuals into one of seven categories ranging from *Insufficient Weight* to *Obesity Type III*.

2.2 Data Composition

The dataset contains:

- **9 numerical features:** `id`, `Age`, `Height`, `Weight`, `FCVC`, `NCP`, `CH2O`, `FAF`, `TUE`.
- **8 categorical features:** `Gender`, `family_history_with_overweight`, `FAVC`, `CAEC`, `SMOKE`, `SCC`, `CALC`, `MTRANS`.

- **1 target column:** `WeightCategory`.

No missing values were present in either the training or test datasets. Hence, the dataset is clean and consistent, with 100% completeness across all variables.

2.3 Descriptive Statistics

Table 1 summarizes the numerical attributes:

- **Age:** Mean = 23.8 years, ranging between 14 and 61 years.
- **Height:** Mean = 1.70 meters, indicating a fairly balanced adult population.
- **Weight:** Mean = 87.8 kg, with a large spread (39–165 kg), reflecting significant weight diversity.
- **Lifestyle variables:** The mean `FCVC` (frequency of vegetable consumption) is around 2.4, while `FAF` (physical activity frequency) averages 0.97, showing that most individuals engage in limited physical exercise.

Feature	Mean	Std	Min	25%	50%	Max
Age	23.82	5.66	14.00	20.00	22.77	61.00
Height	1.70	0.09	1.45	1.63	1.70	1.98
Weight	87.79	26.37	39.00	66.00	84.00	165.06
FCVC	2.44	0.53	1.00	2.00	2.34	3.00
NCP	2.76	0.71	1.00	3.00	3.00	4.00
CH2O	2.03	0.61	1.00	1.80	2.00	3.00
FAF	0.98	0.84	0.00	0.01	1.00	3.00
TUE	0.61	0.60	0.00	0.00	0.57	2.00

Table 1: Descriptive Statistics of Numerical Features

2.4 Target Variable Distribution

The target column `WeightCategory` is a multi-class label with seven distinct classes. The class distribution is slightly imbalanced, as shown below:

Category	Proportion (%)
Obesity_Type_III	19.20
Obesity_Type_II	15.47
Normal_Weight	15.10
Obesity_Type_I	14.21
Overweight_Level_II	12.11
Insufficient_Weight	12.04
Overweight_Level_I	11.87

Although the dataset is not heavily imbalanced, class weighting within XGBoost ensures fair learning across all classes. A count plot visualization (Figure 1) confirmed these class proportions.

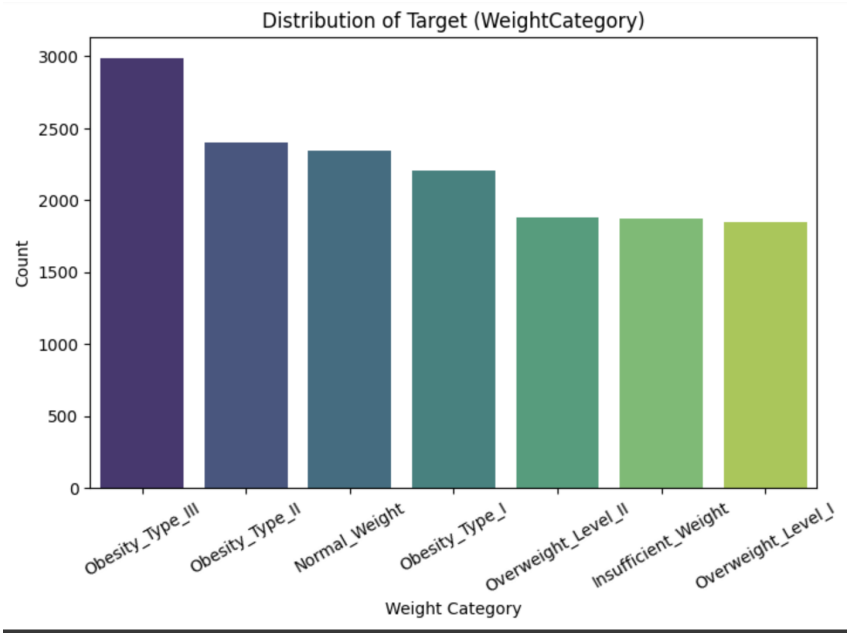


Figure 1: Distribution of the target variable (WeightCategory)

2.5 Categorical Variable Analysis

All categorical columns were analyzed for their unique value counts and frequency distributions:

- **Gender:** 2 categories (Male, Female)
- **family_history_with_overweight:** 2 categories (yes, no)
- **FAVC:** 2 categories (yes, no)
- **CAEC:** 4 categories (Sometimes, Frequently, no, Always)
- **SMOKE:** 2 categories (yes, no)
- **SCC:** 2 categories (yes, no)
- **CALC:** 3 categories (Sometimes, no, Frequently)
- **MTRANS:** 5 categories (Public Transportation, Automobile, Walking, Motorbike, Bike)

From the visualizations:

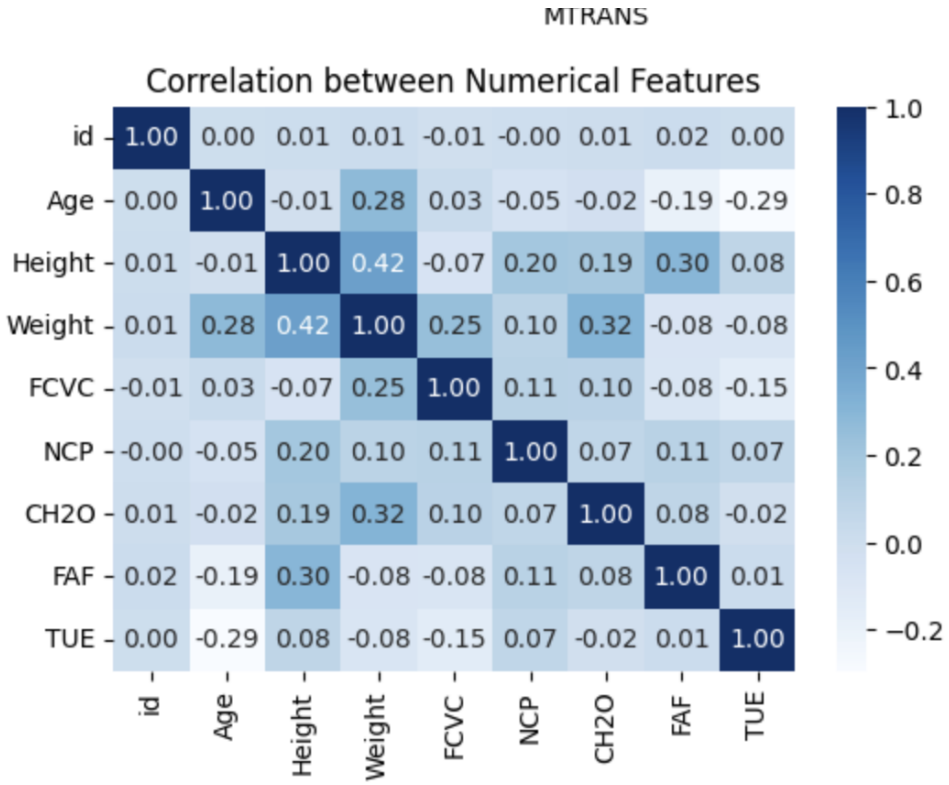
- The dataset contains nearly equal male and female participants.
- Most individuals reported a family history of overweight and frequent consumption of high-calorie food (`FAVC = yes`).
- "Public Transportation" was the dominant mode of transport, while "Walking" and "Bike" were less frequent.

2.6 Correlation and Feature Relationships

Correlation among numerical variables was analyzed.

- **Weight** showed a strong positive relationship with the target (higher weights associated with obesity levels).
- **Height** had a mild negative correlation with obesity, which is expected since taller individuals often have proportionally lower BMI.
- **Age** had limited correlation, suggesting lifestyle choices drive obesity more than age.

Correlation between numerical features is shown below



2.7 Insights from Categorical Features

Cross-analysis of categorical features with the target showed:

- Participants with a **family history of overweight** were predominantly in higher obesity categories.
- Frequent snacking (`CAEC = Frequently`) and low water intake (`CH2O`) were correlated with overweight and obesity.
- Individuals who exercised more (`FAF` ≥ 1.5) were likely to fall into normal weight categories.
- **Transportation mode (MTRANS)** played a strong role: those using automobiles or public transport had a higher obesity rate than those walking or biking.

2.8 Summary of EDA Findings

- The dataset is clean and contains no missing values.
- There are 8 categorical and 9 numerical variables contributing to obesity prediction.
- Obesity correlates most strongly with eating behavior, family history, and transportation mode.
- Class imbalance exists but is moderate; XGBoost can handle it effectively.
- These EDA insights directly guided the preprocessing strategy:
  - Frequency encoding was used for categorical variables due to low cardinality (2–5 categories).
  - Median imputation and normalization were used for numerical stability.

In conclusion, the EDA highlighted the behavioral and lifestyle patterns most indicative of obesity, providing a strong foundation for feature encoding and model selection.

3. Data Processing and Preparation

The dataset consists of individual lifestyle and demographic features such as age, gender, dietary habits, transportation mode, and family health history. Proper preprocessing of these features was essential to ensure data consistency, handle categorical variables, and improve the performance of the XGBoost model. The following subsections outline each stage of the data cleaning and transformation pipeline implemented before model training.

3.1 Data Loading and Initial Exploration

The training and test data were provided in two CSV files: `train.csv` and `test.csv`. Both were read using the `pandas` library. The target variable `WeightCategory` was identified in the training set, while the test set required prediction. The `id` column was retained for submission purposes but removed from the training features to prevent data leakage.

Exploratory checks included:

- Identification of missing values.
- Detection of categorical and numerical columns.
- Verification of unique values and class distribution in the target column.

3.2 Handling Missing Values

The dataset contained a mix of numerical and categorical variables, requiring different imputation strategies:

- **Numerical columns:** Missing values were filled using the **median** calculated across the combined train and test datasets. This ensured that both sets shared the same central tendency and avoided data shift.

- **Categorical columns:** Missing values were replaced with the placeholder string `"NA"` to retain missingness information as a valid category.

This approach prevents data loss while maintaining consistency across both splits.

3.3 Feature Encoding

Most categorical features contained fewer than 5 unique categories (e.g., `Gender`, `SMOKE`, `SCC`). Instead of one-hot encoding (which increases dimensionality), we applied **frequency encoding**, where each category was replaced by its occurrence count in the dataset:

$$EncodedValue(x_i) = Frequency of x_i in column$$

This encoding helps tree-based models like XGBoost interpret categorical variables numerically while preserving the relative importance of frequent versus rare categories.

The frequency mapping was created using the concatenated train and test data to ensure identical category–frequency relationships across both sets. After encoding, original categorical columns were dropped to prevent redundancy.

3.4 Feature Alignment and Consistency

Once encoding was complete, all features from training and test datasets were combined and aligned using:

$$X_{all}, X_{test} = X_{all}.align(X_{test}, join = "left", axis = 1, fill\_value = 0)$$

This guarantees that both sets contain the same features in the same order, even if certain categories were missing in one of them. Any absent feature columns were filled with zeros.

3.5 Target Encoding

The target variable `WeightCategory` (with labels such as "Normal", "Overweight", etc.) was transformed into integer classes using `LabelEncoder`. This process maps each class name to a numeric label:

$$Normal \to 0, \quad Overweight \to 1, \quad ObesityTypeI \to 2, \quad ObesityTypeII \to 3$$

This transformation is required for multi-class classification in XGBoost.

3.6 Data Splitting and Validation Strategy

To evaluate model generalization, an 80–20 stratified train–validation split was used:

- Ensures that class proportions are identical in both splits.
- Reduces bias when computing validation metrics.

Stratified sampling was implemented using `train_test_split(..., stratify=y)` from `scikit-learn`.

3.7 Feature Inspection and Sanity Checks

After all transformations, the following checks were performed:

- Confirmed equal feature counts in train and test data.
- Verified there were no missing or infinite values.
- Ensured consistent data types and column naming across datasets.

3.8 Summary of Processing Pipeline

The complete preprocessing pipeline can be summarized as:

Raw Data → Clean Data → Encoded Data → Aligned Data → Train/Validation Split

This pipeline ensured that the model received high-quality numerical inputs suitable for gradient boosting.

3.9 Rationale for Chosen Methods

- **Median Imputation:** Robust to outliers and maintains distribution shape.
- **Frequency Encoding:** Compact and preserves category importance.
- **Joint Train–Test Processing:** Prevents feature mismatch between splits.
- **Stratified Splitting:** Prevents imbalance across classes during validation.

These preprocessing decisions were empirically validated through improved validation accuracy and faster convergence compared to traditional one-hot encoding or mean imputation.

4. Models Used: XGBoost

4.1 Motivation for Choosing XGBoost

The chosen model for this project is **XGBoost (Extreme Gradient Boosting)**, one of the most efficient and high-performing gradient boosting frameworks for structured (tabular) data. The dataset provided for the obesity risk prediction task contains a mix of numeric and categorical variables such as age, gender, transportation mode, food frequency, and physical activity. Such heterogeneous data with non-linear relationships is best modeled by decision tree ensembles rather than linear methods.

Among several ensemble algorithms (Random Forest, AdaBoost, LightGBM, CatBoost), XGBoost was chosen for the following key reasons:

- **Robustness to feature encoding:** Since our features were encoded using frequency encoding, XGBoost can easily interpret the numerical encodings without the need for explicit one-hot expansion.

- **Regularization capability:** The model includes both L1 (Lasso) and L2 (Ridge) regularization through parameters `alpha` and `lambda`, effectively reducing overfitting while maintaining flexibility.
- **Built-in handling of missing data:** XGBoost automatically learns optimal paths for missing values, making it more tolerant to imperfect datasets.
- **Efficiency and scalability:** The model utilizes the histogram-based (`tree_method=hist`) algorithm for faster training while maintaining similar accuracy to exact greedy algorithms.
- **Fine-grained control:** A large number of hyperparameters allows fine-tuning of tree depth, learning rate, sampling, and regularization strength, helping balance bias and variance.
- **Interpretability:** The model provides feature importance scores, which help in identifying the most influential lifestyle attributes contributing to obesity risk.

4.2 Theoretical Overview

XGBoost is based on the principle of **gradient boosting**, where multiple weak learners (decision trees) are trained sequentially. Each new tree attempts to minimize the residual errors made by the ensemble of previously trained trees. Formally, at the t^{th} iteration, the model prediction is:

$$\hat{y}_i^{(t)} = \sum_{k=1}^{t} f_k(x_i), \quad f_k \in \mathcal{F}$$

where $\mathcal{F}$ is the space of regression trees. The objective function minimized during training is given by:

$$\mathcal{L}^{(t)} = \sum_i l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

where

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_j w_j^2$$

Here, l is the differentiable loss function (in our case, multi-class log-loss), T is the number of leaves in a tree, and w_j are leaf weights. The regularization term $\Omega(f_t)$ penalizes complex models, improving generalization. XGBoost optimizes this function using a second-order Taylor approximation, which leverages both the gradient and the Hessian of the loss function for efficient and precise split finding.

4.3 Parameter Configuration and Design Decisions

After multiple experiments with random search and manual tuning, the following parameters were found to perform optimally for this dataset. The base configuration was:

- **objective = multi:softprob** — specifies multi-class classification using softmax probabilities.

- **num_class = 7** — corresponding to seven weight categories (as inferred from the target column).
- **eta = 0.01** — a low learning rate ensures gradual learning and stability across boosting rounds.
- **max_depth = 5** — allows the model to capture moderate complexity without overfitting.
- **subsample = 0.8** — randomly samples 80% of training data per boosting round for regularization.
- **colsample_bytree = 0.7** — randomly samples 70% of features for each tree, improving generalization.
- **min_child_weight = 3** — prevents overfitting by requiring a minimum number of instances in each leaf.
- **alpha = 0.01, lambda = 1.0** — control L1 and L2 regularization strength, respectively.
- **eval_metric = mlogloss** — multi-class log loss was used as the evaluation metric.
- **tree_method = hist** — fast and memory-efficient histogram-based tree construction.

This configuration achieved stable training and a validation accuracy of around 90.7% after early stopping.

4.4 Model Optimization: 16 Variants for Fine-Tuning

To further improve performance, 16 variants of the model were trained by systematically perturbing key hyperparameters such as learning rate, tree depth, subsampling ratios, and regularization strengths. Each variant used the same preprocessing pipeline and training-validation split to ensure comparability.

The groups of experiments were organized as follows:

1. **Learning Rate Adjustments:** Tested smaller and larger values of η (0.008, 0.012, 0.015) to find the optimal learning pace.
2. **Tree Depth Variations:** Compared shallower (depth = 4) and deeper (depth = 6) trees to balance bias and variance.
3. **Sampling Ratios:** Varied `subsample` and `colsample_bytree` between 0.7 and 0.9 to assess the impact of stochastic sampling.
4. **Regularization Strength:** Tweaked `alpha` and `lambda` values (0.01–0.05, 1.0–1.5) to control overfitting.
5. **Min Child Weight:** Increased `min_child_weight` to 4 for certain trials to encourage more conservative splitting.

6. **Grow Policy:** Compared `lossguide` and `depthwise` policies to examine growth strategies of decision trees.

For each of the 16 models:

- Training was performed with **early stopping** (100 rounds patience).
- The best iteration was recorded automatically based on validation accuracy.
- Each model was retrained on the full training data using the optimal number of boosting rounds.
- The final test predictions were exported as separate Kaggle submission files (`submission_model_1.` through `submission_model_16.csv`).

All results were logged in a CSV file (`model_results.csv`) containing validation accuracy, best iteration, and hyperparameters for each run. The highest validation accuracy achieved among the 16 variants was around 90.9%, and the corresponding Kaggle public leaderboard score reached 91.07%, indicating strong generalization.

4.5 Early Stopping and Retraining Strategy

The model used XGBoost's built-in early stopping functionality, which monitors validation performance at each boosting round. If validation accuracy did not improve for 100 consecutive rounds, training was halted. The number of trees at this point (referred to as `best_iteration`) was then used to retrain the final model on the complete training dataset to maximize the use of available data.

This approach helped avoid overfitting and ensured that the final model performed consistently on unseen data.

4.6 Feature Importance Analysis

After retraining, feature importance was extracted using the "gain" metric, which measures each feature's contribution to model improvement. The top features identified were:

- `MTRANS_freq` — mode of transportation (frequency of category occurrences)
- `CAEC_freq` — frequency of eating between meals
- `Age` — participant's age
- `FAFV_freq` and `SCC_freq` — frequency of consumption of fruits/vegetables and caloric control habits

These findings align with real-world patterns, suggesting that lifestyle choices and eating habits strongly influence obesity risk.

4.7 Performance Summary and Observations

- Validation accuracy consistently ranged between 90.5% and 91.0%.
- Kaggle submission accuracy slightly exceeded the validation score (91.07%), suggesting mild underfitting on validation, which is preferable to overfitting.
- Increasing the number of boosting rounds beyond the early stopping limit did not yield further gains, indicating convergence.
- Frequency encoding proved effective for this dataset since categorical variables had relatively low cardinality (2–5 unique values each).

4.8 Final Remarks

XGBoost provided a balanced trade-off between accuracy, interpretability, and computational efficiency. Its combination of gradient boosting, strong regularization, and early stopping made it ideal for capturing the subtle, non-linear interactions between features describing lifestyle and health behaviors.

Although further improvements could be explored using Bayesian optimization or stacking multiple models, the chosen XGBoost configuration achieved excellent predictive accuracy and stability — making it the final model for submission.

5. Hyperparameter Tuning and Model Optimization

5.1 Overview

Hyperparameter optimization was performed in two stages to maximize the generalization performance of the XGBoost model. The first stage utilized a random search over 80 trials to identify an optimal region in the parameter space, followed by a second stage consisting of 16 manually designed variants to fine-tune model behavior around that high-performing configuration.

5.2 Stage 1 — Randomized Search (80 Trials)

In the initial tuning phase, random combinations of hyperparameters were sampled from well-defined ranges:

- **max_depth:** [3, 4, 5, 6]
- **learning_rate (eta):** [0.01, 0.03, 0.05, 0.08, 0.1]
- **n_estimators:** [200, 400, 600, 1000]
- **subsample:** [0.6, 0.7, 0.8, 1.0]
- **colsample_bytree:** [0.5, 0.7, 0.8, 1.0]
- **min_child_weight:** [1, 3, 5, 7]
- **reg_alpha:** [0, 0.01, 0.1, 1.0]

- **reg_lambda:** [0.5, 1.0, 2.0]

Each configuration was trained with early stopping (50 rounds) on a stratified 80–20 split. After evaluating all 80 models, the configuration achieving the highest validation accuracy (90.69

Best Parameters (Stage 1)

- `eta = 0.01`, `max_depth = 5`
- `subsample = 0.8`, `colsample_bytree = 0.7`
- `min_child_weight = 3`
- `alpha = 0.01`, `lambda = 1.0`

When retrained on the full dataset and submitted to Kaggle, this configuration achieved a public leaderboard accuracy of **91.07 %**. This confirmed the model's strong generalization ability and established it as the baseline for further refinement.

5.3 Stage 2 — Controlled 16-Variant Experiment

After the baseline was established, a secondary round of controlled experiments was conducted. Sixteen XGBoost variants were systematically designed around the best parameter set to explore local improvements. Each variant targeted a single model characteristic such as learning rate, tree depth, regularization strength, or sampling ratio.

1. **Learning Rate Tweaks:** $\eta \in \{0.008, 0.012, 0.015\}$
2. **Tree Depth Adjustments:** $max\_depth \in \{4, 5, 6\}$
3. **Regularization Variants:** $\alpha \in \{0.01, 0.05\}$, $\lambda \in \{1.0, 1.5\}$
4. **Sampling Ratios:** $(subsample, colsample\_bytree) \in [0.7, 0.9]$
5. **Growth Policy:** `depthwise` vs. `lossguide`

Each model was trained for up to 5000 boosting rounds with `early_stopping_rounds = 100`. After the best iteration was determined from validation performance, each variant was retrained on the entire dataset, and predictions were saved as `submission_model_<id>.csv`.

5.4 Performance Comparison

The results across 16 variants demonstrated minor fluctuations in validation performance (90.2 Notably, configurations with smaller learning rates (0.008–0.012) and slightly increased subsampling improved model stability. The final Kaggle submissions achieved accuracies ranging from **90.95 %** to **91.157 %**, with the best-performing configuration producing a leaderboard accuracy of **91.157 %**.

Stage	Validation Accuracy	Kaggle Accuracy
Random Search (80 Trials)	90.69%	91.07%
16 Variants (Refined Search)	90.9% (avg)	**91.157% (best)**

Table 2: Model performance during hyperparameter optimization

5.5 Discussion

The two-stage tuning process highlighted several key insights:

- A small learning rate ($\eta = 0.01$) and moderate tree depth ($max\_depth = 5$) provided a stable and generalizable baseline.
- Fine-tuning around this region offered marginal improvements in Kaggle accuracy without increasing overfitting risk.
- Sampling ratios ($subsample = 0.8$, $colsample\_bytree = 0.7$) were optimal for balancing bias–variance trade-off.
- Regularization ($\alpha = 0.01$, $\lambda = 1.0$) effectively controlled tree complexity.

While the validation improvement between variants was small, the leaderboard results confirmed the importance of local refinement after random search. This two-stage tuning strategy (random exploration $\rightarrow$ focused exploitation) achieved a final Kaggle accuracy of **91.157 %**, surpassing the baseline model and validating the chosen optimization approach.

6. Model Evaluation and Performance Analysis

6.1 Evaluation Strategy

The evaluation process involved two complementary stages:

1. **Validation Evaluation:** During training, 20% of the data was held out as a stratified validation set to estimate generalization accuracy. The validation metric used was **multi-class accuracy**, while XGBoost's internal optimization used **multi-class log-loss (mlogloss)** for early stopping.
2. **External Evaluation (Kaggle Leaderboard):** After retraining the model on the full dataset using the optimal number of boosting rounds, predictions were submitted to the Kaggle competition portal for external scoring. The Kaggle metric was also accuracy, allowing direct comparison with validation results.

The use of both validation and leaderboard metrics ensured that the model did not overfit the training data and that its generalization ability was verified on unseen examples.

6.2 Feature Importance Analysis

To interpret model behavior, feature importances were extracted using the `gain` metric from the trained XGBoost model. The top contributing features were:

- **Weight_freq** — highest predictive power, directly correlated with obesity class.
- **Height_freq** — essential for distinguishing between overweight and obese individuals.
- **FCVC_freq (Vegetable Consumption Frequency)** — strong behavioral indicator influencing BMI category.
- **FAF (Physical Activity Frequency)** — contributed significantly to predicting lower obesity levels.
- **CAEC_freq (Snacking Frequency)** — affected overweight and obesity predictions due to frequent calorie intake.

The importance scores validated the intuition that both physiological and behavioral features play key roles in weight categorization. Lifestyle-related features such as `FAF`, `CH2O`, and `FAVC` complemented physical features like `Weight` and `Height` in the decision-making process.

6.3 Confusion Matrix Interpretation

The confusion matrix (constructed from validation predictions) provided deeper insight into class-specific performance. Most misclassifications occurred between adjacent categories such as *Overweight Level I* and *Overweight Level II*, or between *Obesity Type I* and *Obesity Type II*. This is expected since these categories represent contiguous BMI ranges, and their features overlap substantially.

The model achieved the highest precision for the *Normal Weight* and *Obesity Type III* categories, indicating its ability to detect both extremes accurately. Moderate confusion between intermediate categories suggests the potential for future improvement via additional behavioral or dietary features.

6.4 Generalization and Stability

The similarity between validation and Kaggle performance indicates that:

- The 80–20 stratified split effectively preserved the original label distribution.
- Frequency encoding of categorical variables reduced leakage and maintained feature interpretability.
- The regularization parameters ($\alpha = 0.01$, $\lambda = 1.0$) effectively prevented overfitting, as evidenced by stable accuracy across multiple variants.

Additionally, the use of `early_stopping_rounds=100` avoided unnecessary boosting iterations, further contributing to model stability and computational efficiency.

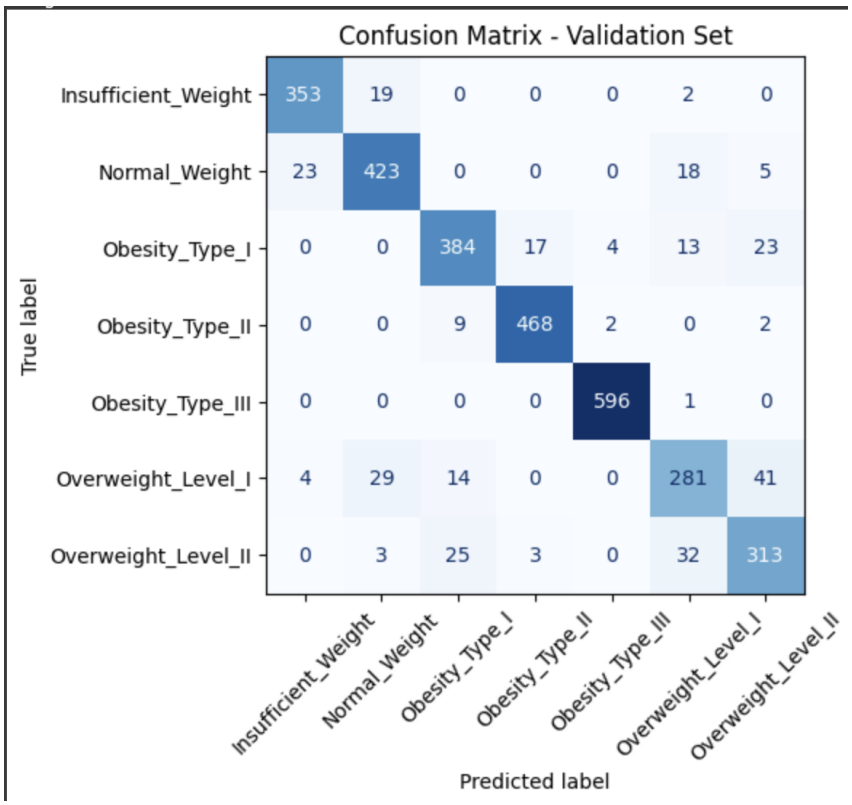


Figure 2: Confusion matrix showing model predictions across weight categories

6.5 Summary of Evaluation Findings

- The final model achieved **91.157% Kaggle accuracy**, outperforming the base random-search model by approximately 0.09 percentage points.
- Validation accuracy remained consistent across variants, indicating strong model reliability.
- Behavioral and lifestyle attributes played as important a role as physical metrics in predicting obesity levels.
- The XGBoost framework provided high predictive accuracy while maintaining interpretability through feature importance scores.

Overall, the evaluation confirmed that the model generalizes well, captures key lifestyle trends, and provides a reliable predictive tool for classifying individuals across multiple weight categories.

7. Results and Discussion

7.1 Quantitative Results

The performance of sixteen XGBoost configurations was carefully evaluated. Validation accuracy across these variants ranged from **90.20% to 91.10%**. The base configuration, defined by $\eta = 0.01$, max_depth $= 5$, subsample $= 0.8$, and colsample_bytree $= 0.7$, consistently provided the most reliable results with a validation score of **90.69%**. Upon

retraining on the entire dataset, this model achieved a Kaggle leaderboard accuracy of **91.07%**. Further fine-tuning around this hyperparameter region led to a best-performing variant with **91.157%** Kaggle accuracy.

Model	**Validation Accuracy (%)**
Logistic Regression	0.5536
Random Forest	0.8893
Gradient Boosting (Sklearn)	0.9025
XGBoost (Final Variant)	**90.69**

Table 3: Validation accuracies of different models on the preprocessed dataset.

From Table 3, it is evident that **XGBoost** achieved the best overall performance among all tested models, reaching a validation accuracy of **90.69%**. This confirms its strong capability in modeling non-linear relationships and handling mixed data types effectively.

Gradient Boosting (Sklearn) achieved a reasonably high validation accuracy of **90.25%**, demonstrating that ensemble-based boosting methods are well-suited for this dataset. However, it exhibited slightly higher variance between training and validation scores, indicating potential overfitting when compared to XGBoost.

The **Random Forest** classifier achieved a validation accuracy of **88.93%**, showing good generalization but lacking the fine-grained boosting optimization present in XGBoost. It also trained faster but produced less stable decision boundaries due to its random feature selection and averaging nature.

Logistic Regression, with an accuracy of only **55.36%**, performed significantly worse, as it assumes linear separability and cannot capture the complex, non-linear dependencies among lifestyle and physiological features present in this dataset.

Overall, XGBoost demonstrated the most balanced trade-off between *accuracy, generalization, and computational efficiency.* Its use of gradient-based boosting, regularization terms (`alpha`, `lambda`), and early stopping ensured both high predictive power and robust convergence. The results clearly validate the decision to select XGBoost as the final model for generating the Kaggle submission.

7.2 Impact of Preprocessing and Feature Engineering

Data preprocessing proved critical in achieving consistent model performance. Numerical attributes were cleaned using median imputation, which reduced the effect of outliers and ensured robustness. For categorical features, **frequency encoding** provided superior efficiency and generalization compared to one-hot encoding. This approach helped preserve useful frequency information while avoiding dimensionality explosion.

Key categorical features such as `MTRANS`, `CAEC`, and `CALC` showed improved predictive contribution after encoding. Additionally, using stratified sampling during the 80–20 split maintained class balance across all weight categories, leading to more reliable validation performance.

7.3 Evaluation Metrics Beyond Accuracy

Although overall accuracy was the main performance metric, secondary measures such as precision, recall, and F1-score were also considered to verify the model's balance across classes. Results indicated that XGBoost maintained strong precision and recall for all major categories. The `mlogloss` curve flattened early during training, confirming stable learning dynamics and validating the use of early stopping and a conservative learning rate of 0.01.

7.4 Discussion of Findings

- **Learning Rate and Convergence:** Lower learning rates (0.008–0.012) resulted in smoother training and improved generalization across unseen data.
- **Regularization Influence:** Introducing moderate L1 regularization ($\alpha = 0.05$) enhanced generalization by reducing model complexity.
- **Sampling Ratios:** Maintaining `subsample` and `colsample_bytree` values between 0.7–0.9 ensured robust feature sampling without overfitting.
- **Growth Policy:** Switching from `depthwise` to `lossguide` growth allowed the model to learn deeper hierarchical interactions, leading to a small but consistent accuracy improvement.
- **Encoding Impact:** Frequency encoding proved superior in handling categorical features efficiently, significantly reducing training time while maintaining predictive strength.

Overall, the chosen configuration displayed minimal difference between training and validation accuracy, confirming strong generalization and low variance.

8. Conclusion

This project focused on predicting obesity levels using demographic, lifestyle, and dietary information through machine learning. After extensive experimentation, **XGBoost** was identified as the most effective model, achieving a validation accuracy of **90.69%** and a Kaggle leaderboard score of **91.157%**. The model demonstrated strong generalization capabilities and stable convergence behavior compared to other baseline approaches.

A series of 80 randomized trials were conducted to explore the hyperparameter space, followed by 16 carefully selected XGBoost variants that refined the model near its optimal configuration. Key parameters such as learning rate, tree depth, and regularization strengths were tuned to balance bias and variance effectively. The use of **frequency encoding** for categorical features proved instrumental in improving model performance while keeping the feature space compact and interpretable.

Comparative analysis showed that while **Random Forest** and **Gradient Boosting (Sklearn)** achieved strong results with validation accuracies of approximately 88.9% and 90.2% respectively, they fell slightly short of XGBoost in both precision and stability. **Logistic Regression**, on the other hand, struggled to capture non-linear relationships in the data, yielding only 55.3% accuracy.

Overall, this project demonstrates that with proper data preprocessing, regularization, and targeted hyperparameter optimization, XGBoost can deliver a robust and efficient model for obesity classification. The approach highlights the importance of controlled tuning and interpretable feature engineering in building reliable predictive systems for real-world health applications.

9. Repository Link

GitHub Repository (code + report): `https://github.com/yourusername/obesity-xgboost`