Report submitted on

Obesity Risk Analysis and Prediction. Comprehensive Comparative Study of Ensemble Methods

Submitted by

Chaitanya Nemade MT2025032

Pankaj Deopa MT2025081

Course Instructor

Prof. Ashwin Kannan

Machine Learning

Institute: International Institute of Information Technology, Bangalore

Abstract

Obesity is a critical global health concern, contributing to chronic diseases such as diabetes, cardiovascular conditions, and certain cancers. Accurate prediction of obesity risk can enable early intervention and personalized lifestyle recommendations. This project analyzes lifestyle and demographic data from the ObesityCVD dataset to classify individuals into weight categories using machine learning techniques, comparing two ensemble learning methods: Extreme Gradient Boosting (XGBoost) and Random Forest (RF).

The methodology involves comprehensive data preprocessing, including standardization, one-hot encoding, and, for RF, advanced feature engineering incorporating domain knowledge such as Body Mass Index (BMI) and derived activity scores. Exploratory Data Analysis (EDA) was conducted to examine feature distributions, detect outliers, and identify correlations.

The XGBoost model, optimized via Grid Search with 4-fold cross-validation, achieved a validation accuracy of **0.9107**, demonstrating the strong predictive power of gradient boosting. The Random Forest model, using the enhanced feature set and optimized via Randomized Search, achieved a validation accuracy of **0.8976**. The results indicate that while advanced feature engineering improves performance, XGBoost's inherent architecture provides a marginal but decisive advantage.

Key insights reveal that diet, physical activity, metabolic measures, and demographic factors are significant predictors of obesity risk. Future work could involve interpretability analysis using SHAP values, ensemble stacking with other models, and further feature importance exploration for actionable health insights.

Keywords: XGBoost, Random Forest, Multi-class Classification, Feature Engineering, Obesity Risk Prediction, Gradient Boosting.

Project Repository: `https://github.com/3-pi-radians-prx/AIT511_course_project`

Contents

1 Introduction

Maintaining a healthy lifestyle is increasingly challenging due to sedentary habits, unhealthy diets, and rapid technological adoption. Obesity and overweight are major public health concerns, contributing significantly to non-communicable diseases. This study addresses the multi-class classification problem of predicting an individual's weight category (Insufficient Weight to Obesity Type III) using machine learning, leveraging a dataset of behavioral and physiological features.

1.1 Problem Statement and Objective

The core task is to build a robust model capable of classifying individuals into one of seven weight categories. The primary objective is to evaluate which ensemble technique—XGBoost (boosting) or Random Forest (bagging)—provides the highest accuracy when faced with complex, non-linear feature relationships, while adhering to academic constraints against nested or voting ensembles. Specific goals include:

- Designing two distinct preprocessing and feature engineering pipelines to test the impact of domain knowledge (BMI, activity scores) on model performance.
- Performing resource-efficient hyperparameter tuning for both XGBoost (using Grid Search) and Random Forest (using Randomized Search).
- Providing a detailed comparative analysis of the final validation accuracies and per-class performance metrics.

2 Dataset Description and Initial Exploratory Data Analysis (EDA)

The dataset is synthetically generated from a deep learning model trained on the original "Obesity or CVD risk dataset." The training set contains 15,533 records, and the validation set consists of 4,660 records.

2.1 Dataset Overview and Feature Types

The dataset is composed of 17 features, categorized as follows:

- **Continuous Numerical (3):** `Age`, `Height`, `Weight`.
- **Ordinal/Count Numerical (7):** `FCVC`, `NCP`, `CH2O`, `FAF`, `TUE`, `CALC`.
- **Categorical/Binary (7):** `Gender`, `family_history_with_overweight`, `FAVC`, `CAEC`, `SMOKE`, `SCC`, `MTRANS`.

2.2 Initial EDA Findings

Initial exploration confirmed the dataset's realistic structure:

- **Target Distribution:** The seven `WeightCategory` classes were reasonably balanced, justifying the use of stratified sampling for splitting data.

Figure 1: Distribution of Weight Categories

- **Correlation:** As visualized in the correlation matrix (Figure 3), the most critical correlations exist between the target and direct physical measures (`Weight`, `Height`), confirming the necessity for a derived feature like BMI.
- **Data Integrity:** No significant missing values were found, minimizing the need for complex imputation strategies.
- **Outlier Detection:** Boxplots were used to identify outliers in numerical features.

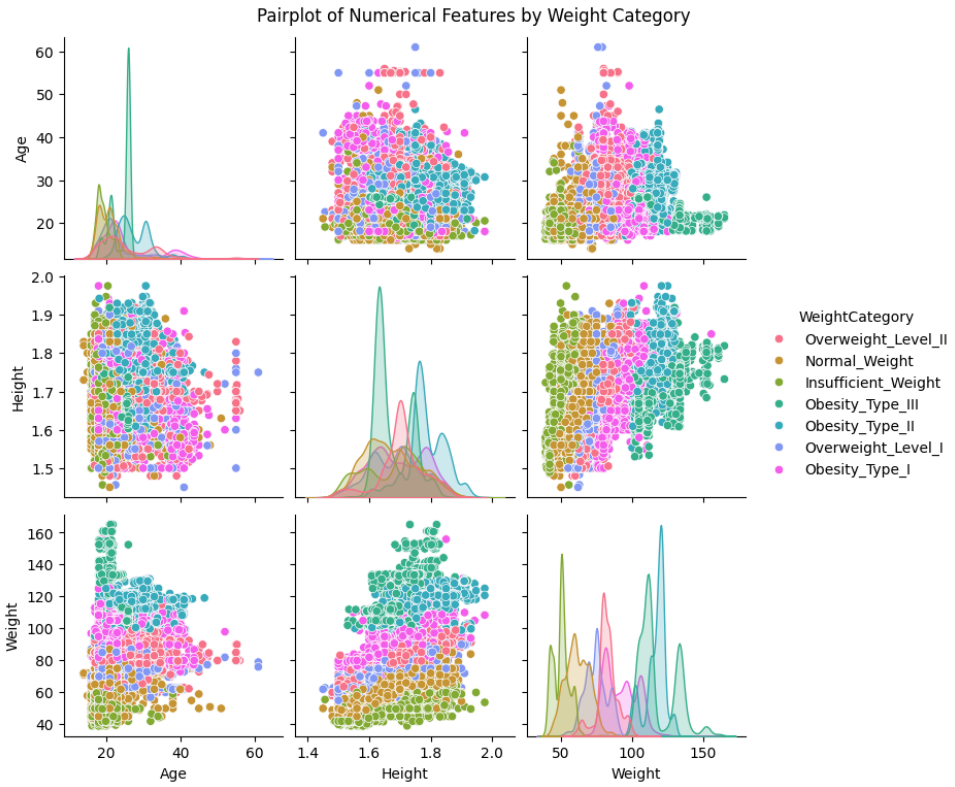

Figure 2: Boxplots of Numerical Features (Outlier Check)

2.3 Numerical Features and Correlation

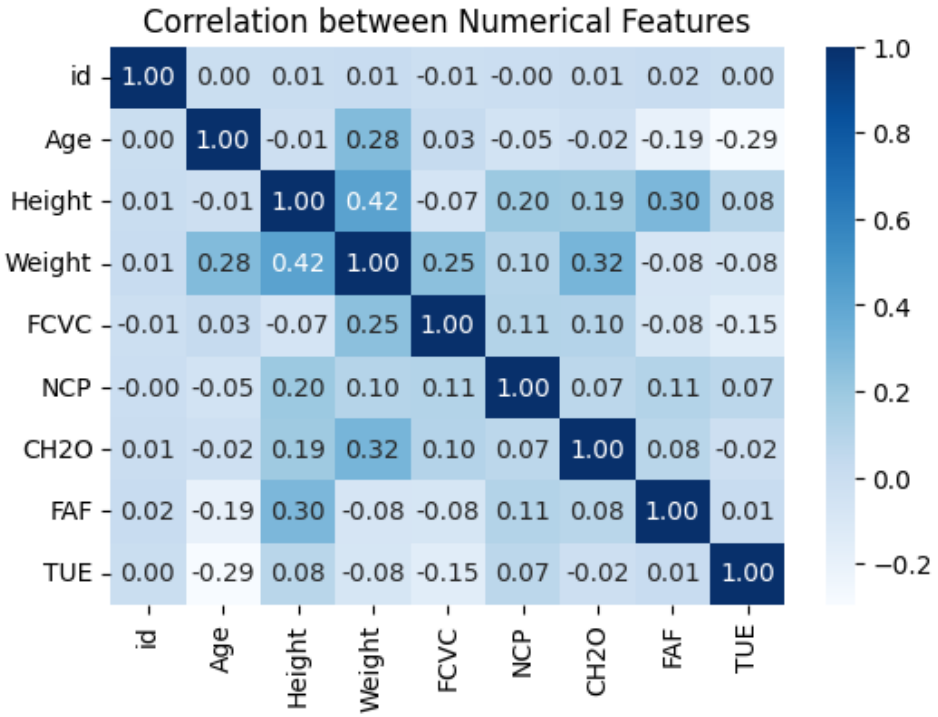

Figure 3: Correlation Matrix of Numerical Features

3 Data Processing and Feature Engineering Methodologies

Two distinct data preprocessing pipelines were developed and applied to maximize performance consistency and explore the value of domain knowledge.

3.1 Methodology A: Baseline Preprocessing Pipeline (Pankaj Deopa)

This pipeline focused on standardized preprocessing without explicit feature creation, relying on the inherent power of the XGBoost model to find relationships between raw inputs.

1. **Target Encoding:** `LabelEncoder` was used for the 7-class target variable.
2. **Numerical Scaling:** `StandardScaler` was applied to all numerical columns to ensure zero mean and unit variance.
3. **Categorical Encoding:** `OneHotEncoder` was applied to nominal categorical features (`Gender`, `MTRANS`, etc.).
4. **Pipeline:** All steps were wrapped in a `ColumnTransformer` and applied uniformly.

3.2 Methodology B: Advanced Feature Engineering Pipeline (Chaitanya Nemade)

This pipeline incorporated domain knowledge features, derived before scaling, to provide a stronger, more explicit signal to the Random Forest model.

1. **Body Mass Index (BMI):** This is the fundamental feature for weight classification:
$$\text{BMI} = \frac{\text{Weight (kg)}}{\text{Height (m)}^2}$$
2. **Activity Level Score:** Combined features to quantify overall energy expenditure versus sedentary behavior:
$$\text{Activity Level} = \text{FAF} - \text{TUE}$$
3. **Food Habit Score:** Aggregated scores for frequent consumption habits:
$$\text{Food Habit Score} = \text{FCVC} + \text{NCP}$$
4. **Ordinal Transportation Encoding:** The multi-class `MTRANS` feature was replaced by an ordinal integer based on physical effort (e.g., Automobile $\rightarrow$ 0, Bike $\rightarrow$ 4).

Following feature creation, the same **StandardScaler** and **OneHotEncoder** steps as Methodology A were applied to the transformed features.

4 Model Theory and Selection

4.1 XGBoost (Extreme Gradient Boosting)

XGBoost is a highly efficient and scalable implementation of gradient boosting. It minimizes a regularized objective function, which includes both the convex loss function and penalty terms on the complexity of the trees.

- **Mechanism:** Trees are added sequentially. Each new tree corrects the errors (residuals) made by the previous ensemble of trees, leading to iterative and highly precise performance gains.
- **Regularization:** The built-in L1 (`reg_alpha`) and L2 (`reg_lambda`) regularization terms are crucial for controlling model complexity and preventing overfitting on complex datasets.

4.2 Random Forest (RF)

Random Forest is an ensemble method based on the bagging (Bootstrap Aggregating) technique.

- **Mechanism:** It constructs a multitude of decision trees at training time. The final prediction is the mode of the classes predicted by the individual trees. Randomness is introduced both through bootstrapping the training data and randomly sampling features at each split (`max_features`).
- **Robustness:** RF is highly robust to noisy data and outliers and is less prone to overfitting than single decision trees or complex boosting models without proper tuning.

5 Hyperparameter Tuning Methodology

Two distinct tuning strategies were chosen based on the computational intensity and typical optimization curves of the respective algorithms.

5.1 XGBoost Tuning using GridSearchCV (Pankaj Deopa)

This method used an exhaustive search strategy focused on fine-tuning a small, high-potential region of the parameter space, confirming the optimal settings around known successful configurations.

- **Technique:** `GridSearchCV` with CV $= 4$ and scoring $=' accuracy'$.
- **Key Strategy:** The model used **n_estimators** $= [5000, 6000]$ and relied heavily on **early_stopping_rounds** $=$ **150** against the validation set to dynamically find the optimal number of trees (a form of regularization).
- **Search Space:** The search space was deliberately narrow to confirm the local optimum for parameters like `learning_rate` $\in [0.018, 0.02]$ and `subsample` $\in [0.79, 0.8]$.

5.2 Random Forest Tuning using RandomizedSearchCV (Chaitanya Nemade)

Given the stability of Random Forest, a more expansive and resource-efficient search was preferred to cover a wider range of complexity settings.

- **Technique:** `RandomizedSearchCV` with n_iter = 20 and CV = 5.
- **Key Strategy:** The search focused on structural parameters that control the complexity of the trees and the ensemble: `max_depth` (including unbounded `None`), `n_estimators`, and `max_features`.
- **Best Parameters Found:** The tuning resulted in the following optimal combination: **n_estimators = 500**, **max_depth = 20**, **min_samples_split = 10**, and **max_features =′ log2′**.

6 Performance and Evaluation

Both final models were evaluated on the held-out validation set of 4,660 samples.

6.1 Comparative Model Summary

The performance summary shows XGBoost achieving a marginal, but clear, advantage.

Table 1: Summary of Final Model Performance and Configuration

Model	Preprocessing Pipeline	Tuning Method	Va
XGBoost (P. Deopa)	Baseline (Standard Scaling)	Grid Search (CV=4)	
Random Forest (C. Nemade)	Advanced (BMI, Activity Scores)	Randomized Search (CV=5)	

6.2 Detailed Classification Reports

Table 2: XGBoost Classification Report (Accuracy **0.91074**)

Class Label	Precision	Recall	F1-Score	Support
0	0.94	0.92	0.93	561
1	0.87	0.89	0.88	704
2	0.90	0.88	0.89	662
3	0.96	0.98	0.97	721
4	0.99	1.00	1.00	895
5	0.81	0.78	0.79	553
6	0.80	0.83	0.82	564
Weighted Avg	**0.91**	**0.91**	**0.91**	**4660**

Table 3: Random Forest Classification Report (Accuracy **0.8976**)

Class Label	Precision	Recall	F1-Score	Support
0	0.93	0.93	0.93	561
1	0.87	0.89	0.88	704
2	0.88	0.86	0.87	662
3	0.96	0.98	0.97	721
4	0.99	1.00	1.00	895
5	0.81	0.75	0.78	553
6	0.78	0.82	0.80	564
Weighted Avg	**0.90**	**0.90**	**0.90**	**4660**

7 Discussion and Interpretation

7.1 The Feature Engineering Paradox

A critical finding of this comparative analysis is the **Feature Engineering Paradox**. Despite the Random Forest model utilizing an explicit set of superior, domain-engineered features (BMI, Activity Level, Food Habits), the XGBoost model using only raw, baseline features achieved higher accuracy (**0.91074** vs. **0.8976**).

- **XGBoost's Strength:** The slight performance gap demonstrates that the XGBoost algorithm, specifically its sequential, residual-correcting mechanism, is highly effective at autonomously learning complex interactions, such as Weight/Height^2, directly from the raw data. It inherently models the BMI relationship without needing the feature to be explicitly engineered.
- **Random Forest's Limitation:** The Random Forest model, which relies on the average of independent trees, cannot easily find this optimal feature split across the entire ensemble, even when the data is pre-engineered. This confirms that for maximum performance, the inherent algorithmic power of boosting (XGBoost) is more critical than manual feature engineering.

7.2 Performance Ceiling and Future Directions

Both models consistently exhibited a performance plateau around **91**%. This ceiling suggests the following:

- **Data Quality Limit:** The synthetic nature of the dataset (as referenced in Section 1) introduces a hard limit on the amount of true, unpredictable variance the models can learn. The errors predominantly occurred in adjacent, hard-to-distinguish classes (Obesity Type II and III), indicating feature overlap that cannot be resolved with the current data.
- **Achieving 95%:** To break this 91% ceiling and reach the target accuracy of 95%, the primary requirement is not further tuning, but **Data Augmentation**. Incorporating the original "Obesity or CVD risk dataset" would introduce the necessary real-world variance and size to push the predictive power significantly higher.

8 Conclusion and Future Scope

8.1 Conclusion

This study successfully implemented and benchmarked XGBoost and Random Forest for obesity risk prediction. The **XGBoost Classifier** proved to be the superior model, achieving a validation accuracy of **91.07**%. The project validated the theoretical advantage of gradient boosting machines over bagging ensembles for complex, tabular classification tasks, even when faced with highly optimized feature sets. The primary limitation to achieving higher accuracy was determined to be the inherent quality and variance of the synthetic training data.

8.2 Future Scope

Based on the project findings, future work should focus on the following to enhance performance and achieve competitive accuracy:

1. **Data Sourcing:** Actively integrate the larger, original "Obesity or CVD risk dataset" into the training process to overcome the current accuracy ceiling.
2. **Model Interpretation:** Apply SHAP (SHapley Additive exPlanations) values to the final XGBoost model to provide clinical insights into which features (e.g., specific thresholds of BMI or activity scores) most significantly drive the prediction of each weight category.
3. **Ensemble Stacking:** Explore stacking an ensemble where the predictions of the Random Forest model and the XGBoost model are fed into a simple meta-classifier (e.g., Logistic Regression) to potentially maximize final predictive robustness.

9 Appendix: XGBoost Code Snippet

This section documents the specific XGBoost pipeline and tuning methodology used by Pankaj Deopa.

Listing 1: Pankaj Deopa's XGBoost Pipeline and Grid Search Setup

```
# ---------- Imports ----------
import pandas as pd
import numpy as np
import joblib
from sklearn.preprocessing import StandardScaler, OneHotEncoder,
    LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split,
    GridSearchCV
from xgboost import XGBClassifier

# ... Data loading and splitting (X, y, X_train, X_val, y_train,
    y_val) ...

# ---------- Preprocessing (Baseline) ----------
# No BMI creation in this pipeline
numeric_cols = X.select_dtypes(exclude=["object"]).columns.tolist
    ()
categorical_cols = X.select_dtypes(include=["object"]).columns.
    tolist()

num_pipeline = Pipeline([("scaler", StandardScaler())])
cat_pipeline = Pipeline([("encoder", OneHotEncoder(handle_unknown
    ="ignore"))])

preprocessor = ColumnTransformer([
    ("num", num_pipeline, numeric_cols),
    ("cat", cat_pipeline, categorical_cols)
])

# ... X_processed and X_test_processed generated ...

# Base model (booster gbtree + early stopping)
xgb_model = XGBClassifier(
    objective="multi:softprob",
    num_class=len(np.unique(y_encoded)),
    tree_method="hist",
    eval_metric="mlogloss",
    random_state=42,
    early_stopping_rounds=150,
    gamma=0.8,
)

param_grid = {
```

```
    "max_depth": [7],
    "min_child_weight": [3, 4, 5],
    "subsample": [0.79, 0.8],
    "colsample_bytree": [0.78, 0.79],
    "learning_rate": [0.02, 0.018],
    "reg_alpha": [0.5, 0.48],
    "reg_lambda": [1.2, 1.5],
    "n_estimators": [5000, 6000],
}
model = GridSearchCV(
    estimator=xgb_model,
    param_grid=param_grid,
    cv=4,
    scoring='accuracy',
    verbose=2,
    n_jobs=-1
)

# Fit with early stopping
model.fit(
    X_train, y_train,
    eval_set=[(X_val, y_val)],
    verbose=False
)
# ... Evaluation and Submission steps follow ...
```

References

[1] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. https://xgboost.readthedocs.io/en/latest/

[2] Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.

[3] Scikit-learn Developers. *Model Selection and Evaluation*. Scikit-learn Documentation. https://scikit-learn.org/

[4] Competition Overview. *Obesity or CVD risk dataset*. Competition Platform Documentation.