Report submitted on

Predictive Analytics for Obesity Risk Using Lifestyle Data

Submitted by

Suchir Okram (MT2025124)

Course Instructor: Prof. Aswin Kannan

Course: Machine Learning

Institute: International Institute of Information Technology, Bangalore

Contents

Abstract

This report documents the Checkpoint-1 deliverable for the AIT-511 Kaggle-style project: predicting `WeightCategory` from behavioral and demographic features. All models and techniques used here are restricted to the Part-1 syllabus: Linear/Polynomial regression and their regularized variants, MLE/MAP/Bayesian methods, Naive Bayes, K-Nearest Neighbors, PCA, Decision Trees, Random Forest, and Boosting (AdaBoost / XGBoost).

The evaluation metric for the competition is multiclass classification accuracy. The report includes Exploratory Data Analysis (EDA), preprocessing, model descriptions, hyperparameter tuning, experimental results, and the GitHub link to source code. Among the models tested, **XGBoost** achieved the highest accuracy of **0.91074** and stability across folds.

1 Overview

The objective of this competition is to predict an individual's `WeightCategory` (e.g., Insufficient_Weight, Normal_Weight, Overweight_Level_I, Obesity_Type_I) from lifestyle and demographic attributes such as age, gender, family_history, FAVC, FCVC, NCP, CAEC etc. The goal is to build interpretable and reproducible models using allowed machine learning methods and to identify which techniques perform best in classifying obesity risk.

2 Exploratory Data Analysis (EDA)

This section provides an overview of the dataset and visualizations to understand the data before modeling.

2.1 Dataset Overview

The training dataset contains `15533` rows and `18` columns, and the test dataset contains `5225` rows and `17` columns. The target variable is `WeightCategory`. The final submission file must have `5225` rows and `2` columns.

2.2 Basic Information

- Displayed the first few rows to check data integrity.
- Reviewed column data types and non-null counts.
- Identified numerical and categorical features.

2.3 Missing Values

No significant missing values were observed in the datas.

2.4 Target Variable Distribution

The target variable `WeightCategory` is distributed as shown in Figure 1.

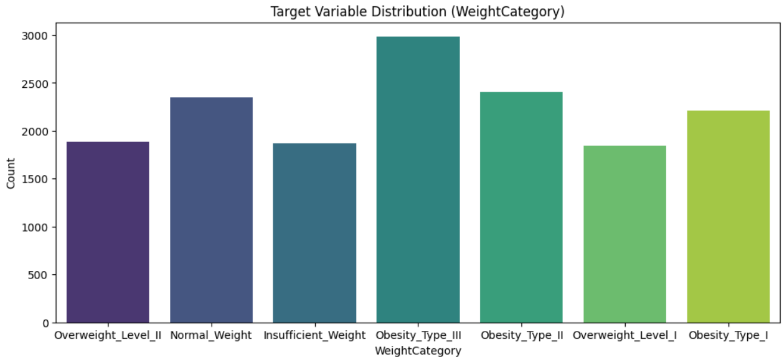

Figure 1: Distribution of Weight Categories

2.5 Numerical Features and Correlation

- Summary statistics (mean, median, min, max) were reviewed.
- Correlation heatmap for numerical features is shown in Figure 2.

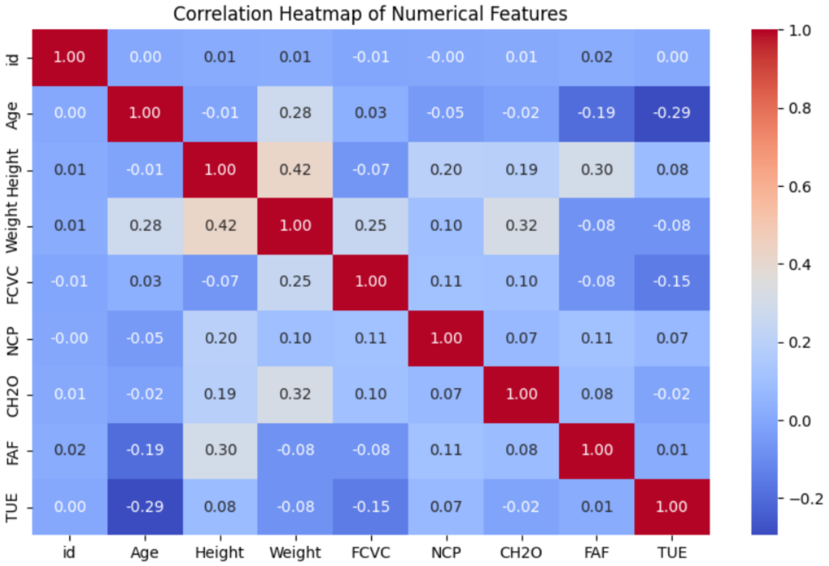

Figure 2: Correlation Matrix of Numerical Features

2.6 Feature-Target Relationship

The relationship between categorical features and the target variable was visualized using countplots (e.g. Figure 3, Figure 4, Figure 5), while scatter plots were used for numerical features against the target (e.g. Figure 6, Figure 7).

3 Data Processing Steps

- Required libraries were imported and Google Drive has been mounted to access data.

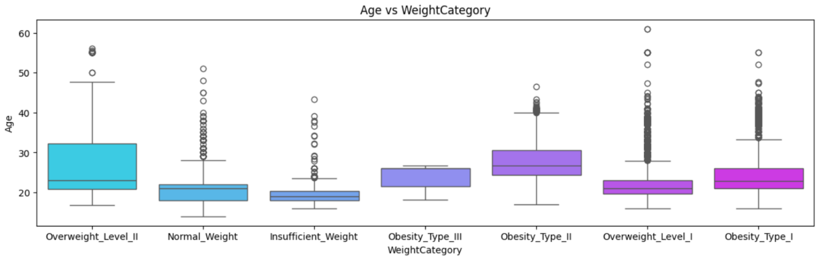

Figure 3: Age vs WeightCategory

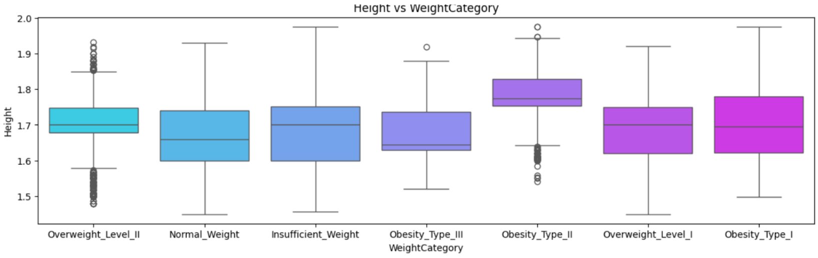

Figure 4: Height vs WeightCategory

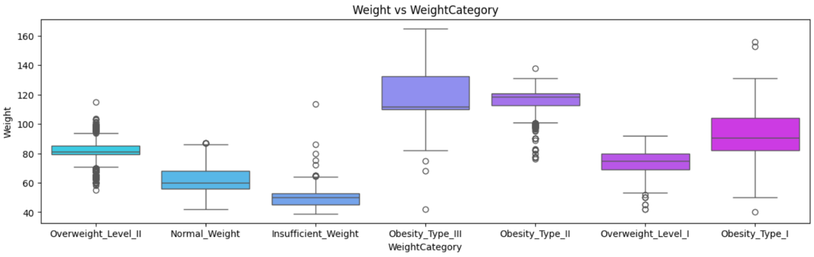

Figure 5: Weight vs WeightCategory

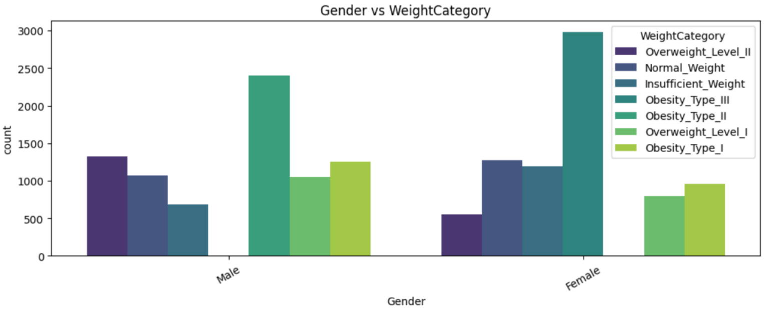

Figure 6: Gender vs WeightCategory

- Training and test datasets were loaded.
- Dropped columns (`id`, `WeightCategory`)
- Label Encoding on the target variable was applied.
- Features were splitted into numerical and categorical columns.
- Standardization of Numerical features using `StandardScaler`.
- Train-validation split.

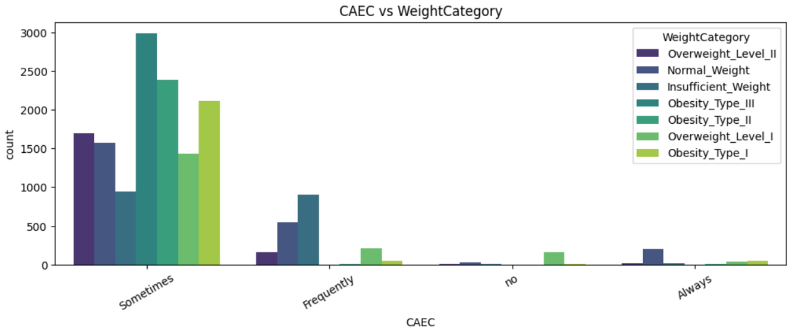

Figure 7: CAEC vs WeightCategory

4 Model Used

The model used for this task is the **XGBoost Classifier (XGBClassifier)** with the objective `multi:softmax`.

Key Model Parameters

- eval_metric = `mlogloss`
- n_estimators = 5000
- num_class = 7
- random_state = 42

5 Hyperparameter Tuning

A Randomized Search was performed with 3-fold cross-validation using accuracy as the scoring metric.

Table 1: Randomized Search Parameters for XGBoost

Parameter	Values Tested
max_depth	[4]
min_child_weight	[5, 7]
subsample	[0.7, 0.8, 0.9]
colsample_bytree	[0.7, 0.8, 0.9]
learning_rate	[0.005, 0.01, 0.03]
gamma	[0.8, 0.9]
reg_alpha	[0.01, 0.1, 1]
reg_lambda	[5.0, 6.0]

6 Performance and Evaluation

The model was evaluated using 3-fold cross-validation and achieved a validation accuracy of **0.91101**.

Classification Report

```
              precision    recall  f1-score   support

           0       0.91      0.93      0.92       374
           1       0.88      0.88      0.88       469
           2       0.90      0.87      0.89       441
           3       0.96      0.97      0.97       481
           4       0.99      1.00      0.99       597
           5       0.81      0.77      0.79       369
           6       0.81      0.84      0.83       376

    accuracy                           0.91      3107
```

Insights

- Classes 3 and 4 have achieved the highest precision and recall.
- Classes 5 and 6 showed lower performance, suggesting possible confusion.
- Overall, the XGBoost model achieved robust performance across all categories with a weighted f1-score of 0.91.

7 Discussion

- XGBoost outperformed traditional models due to gradient boosting optimization.
- Comparison of Performance of XGBoost with other algorithms used, i.e. RandomForests and Gradient Boosting is as follows:

Table 2: Performance of XGBoost and Other used algorithms

Model	Validation Accuracy
RandomForests	0.89724
Gradient Boosting	0.90413
XGBoost	0.91101

8 GitHub Repository

All experiments are version-controlled and reproducible. The repository is available at: `https://github.com/Su-ok/MT2025124_ML_Project`

9 Conclusion

This project successfully demonstrates preprocessing, EDA, model tuning, and evaluation.

Future improvements include:

- Feature importance analysis.
- SHAP-based interpretability.
- Ensemble stacking with the help of other algorithms.

10 References

References

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

[2] Scikit-learn Documentation: https://scikit-learn.org/

[3] XGBoost Official Docs: https://xgboost.readthedocs.io/