AIT511: Course Project 1 Report

Ishita Kar(MT2025052),Adithi P(MT2025011)

22nd October,2025

1 Introduction

Obesity is a growing global health concern influenced by lifestyle, physical, and behavioral factors. The aim of this study is to classify individuals into distinct obesity levels using supervised machine learning techniques. The dataset includes features such as age, gender, weight, height etc. The objective is to explore their relationship with obesity and build models capable of predicting obesity categories effectively.

2 Dataset Description

The data set consists of 17 attributes and 15533 rows. The attributes related to eating habits are: Frequent consumption of high-calorie food (FAVC), Frequency of vegetable consumption (FCVC), Number of main meals (NCP), Consumption of food between meals (CAEC), Consumption of water daily (CH20) and Consumption of alcohol (CALC). The attributes related with the physical condition are: Calories consumption monitoring (SCC), Physical activity frequency (FAF), Time using technology devices (TUE), Transportation used (MTRANS) variables obtained : Gender, Age, Height and Weight. The target consists of 6 classes Normal Weight, Overweight Level I, Overweight Level II,Obesity Type I, Insufficient Weight, Obesity Type II,Obesity Type III.

3 Data Preprocessing

3.1 Data Loading and Cleaning

The dataset was loaded using `pandas` and inspected for missing or inconsistent values. No missing data were present, ensuring a clean dataset for analysis.

3.2 Encoding Categorical Variables

Categorical Features like Gender, family history with overweight, FAVC, CAEC, SMOKE , SCC, CALC, MTRANS and WeightCategory(renamed as target) were encoded manually.

For example:

- **Gender:** 0 = Male, 1 = Female
- **MTRANS:** 0 = Public Transport, 1 = Automobile, 2 = Walking, 3 = Motorbike, 4 = Bike
- **CALC (Alcohol Consumption):** 0 = Sometimes, 1 = Frequently, 2 = No, 3 = Always

Manual encoding was chosen to maintain interpretability and ensure that each numeric value corresponds directly to a meaningful category. This approach also prevented unnecessary creation of multiple one-hot encoded columns, making the dataset compact and easier to interpret.

Additionally, several lifestyle-related features such as **NCP** (Number of main meals per day), **FAF** (Physical activity frequency), and **TUE** (Time using technology devices) contained decimal values. These were **rounded off to the nearest integer** (0, 1, 2, or 3) to categorize the responses into interpretable discrete levels. The integer variables representing discrete categories were converted to the `categorical` data type to ensure appropriate handling during analysis and model training.

3.3 Feature Scaling

Continuous features such as Height, Weight, and Age were standardized using `StandardScaler`:

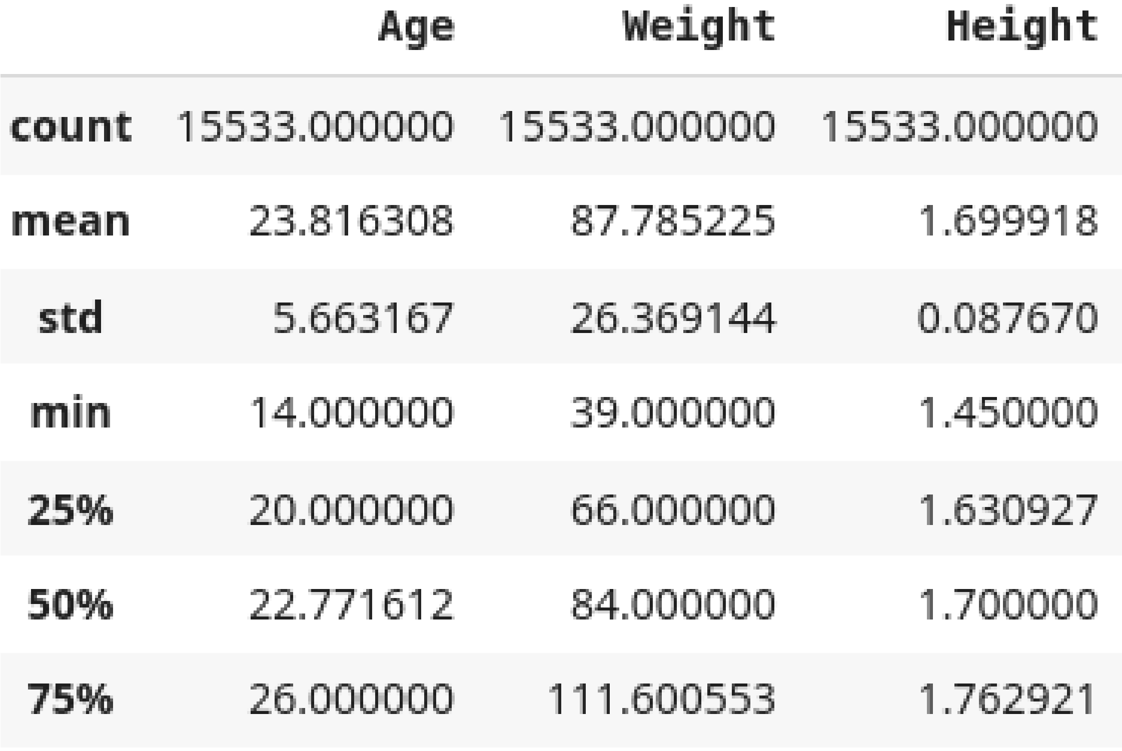
$$z = \frac{x - \mu}{\sigma}$$

4 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) was carried out to understand the statistical distribution of features and their relationships with obesity levels.

4.1 Statistical Summary

A statistical summary of the numerical features was obtained using the `describe()` function from `pandas`. This provided key descriptive statistics including count, mean, standard deviation, and quartile values for each numeric feature.



	Age	Weight	Height
count	15533.000000	15533.000000	15533.000000
mean	23.816308	87.785225	1.699918
std	5.663167	26.369144	0.087670
min	14.000000	39.000000	1.450000
25%	20.000000	66.000000	1.630927
50%	22.771612	84.000000	1.700000
75%	26.000000	111.600553	1.762921
max	61.000000	165.057269	1.975663

Figure 1: Summary statistics generated using the `describe()` function.

The descriptive statistics summarized in Figure 1 indicate that the dataset primarily represents a young adult population, with a mean age of approximately 23.8 years. The mean height is 1.70 meters with low variability, suggesting a consistent adult height range among participants. The mean weight of 87.8 kg, however, shows substantial variation (standard deviation of 26.37), reflecting the presence of individuals across different obesity levels, including overweight and severely obese categories. The 25th to 75th percentile range for weight (66–111.6 kg) further supports the dataset's diversity in body composition. Overall, the data distribution confirms that the dataset is well-suited for modeling obesity levels across a varied population.

4.2 Bivariate Analysis

Bivariate analysis was conducted to examine the relationships between pairs of variables and their influence on obesity levels.

4.2.1 Height and Weight

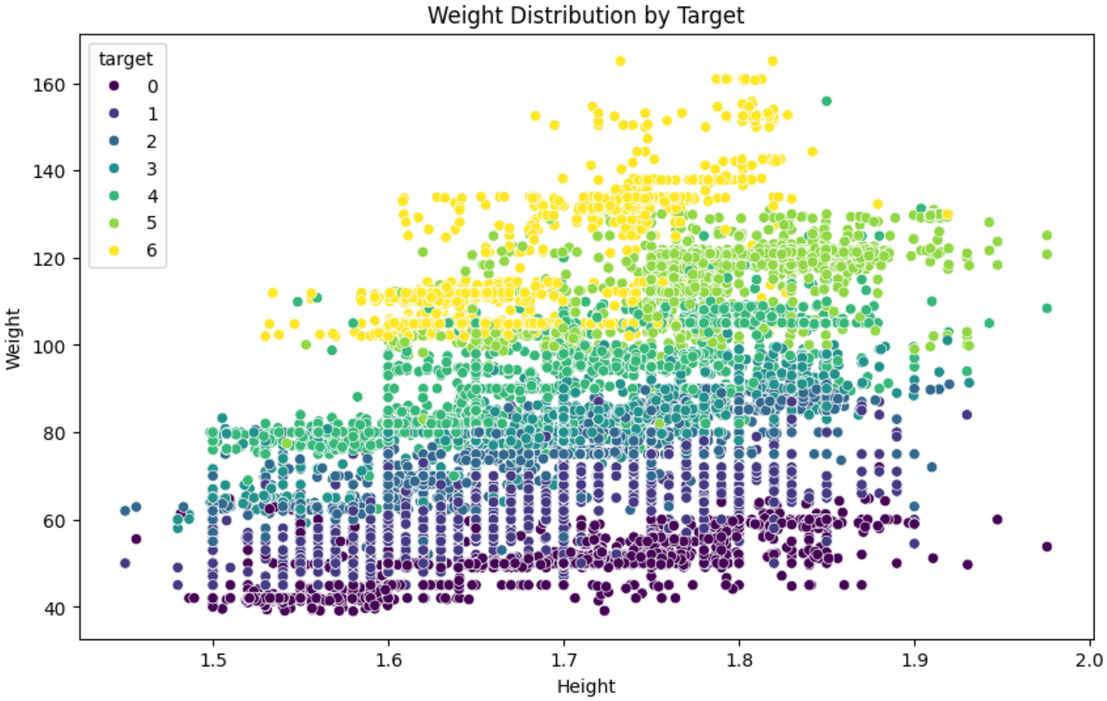


Figure 2: Height vs Weight Distribution by target.

The Height vs Weight plot shows a strong positive relationship, indicating that weight increases with height. However, the broad vertical spread for any given height suggests variability in body composition — a key factor in obesity. Thus, for the same height, heavier individuals are more likely to fall into higher obesity categories.

When categorized by obesity level Insufficient weight individuals lie in the lower band Normal weight individuals lie in the middle band while overweight and obese individuals cluster in the higher weight regions(for same height). Although distinct regions can be observed for different obesity levels, there is noticeable merging or overlap between classes(in particular between normal weight and overweight level I or between obesity level II and obesity level III).

4.2.2 Gender and Target

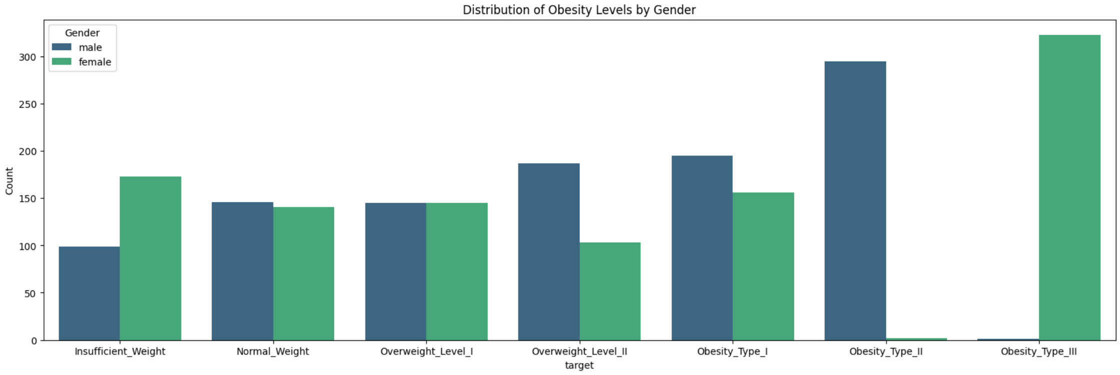


Figure 3: Distribution of Obesity Levels by Gender.

For the given data it can be seen that a large percentage of females belong to the category of Obesity Type III while a large percentage of male belong to the category of Obesity Type II.This indicates that severe obesity is more common among females, whereas males tend to exhibit moderate obesity.

4.2.3 MTRANS and Weight

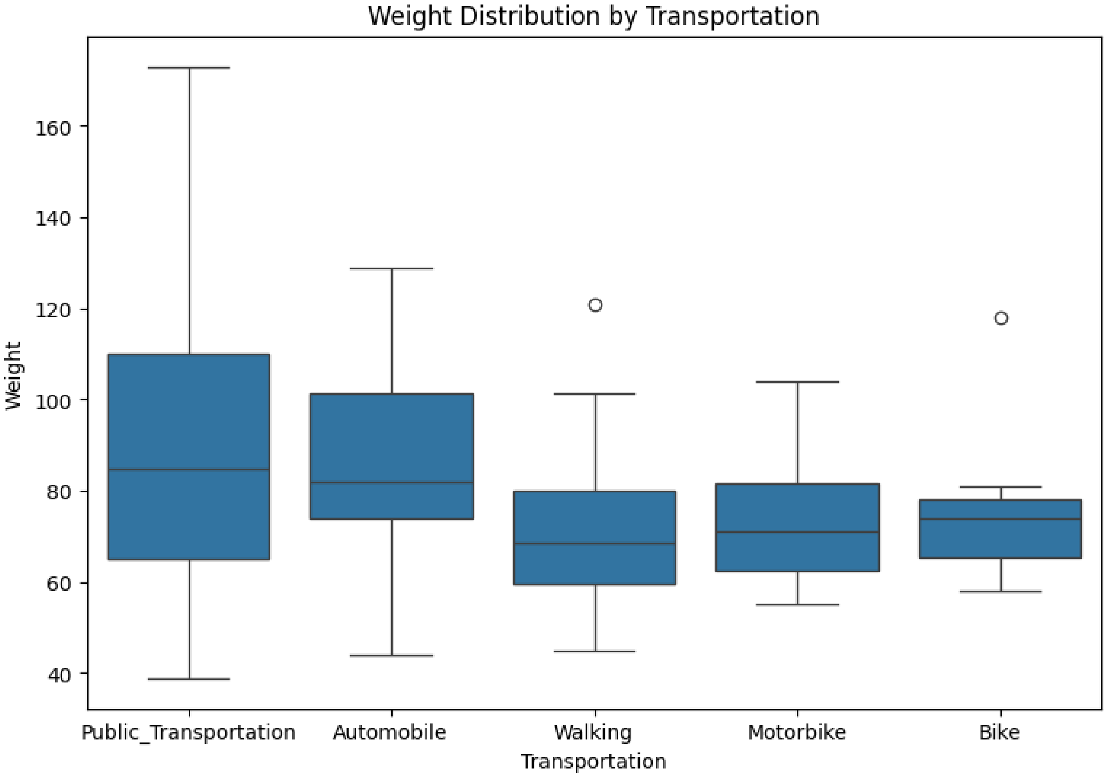


Figure 4: Distribution of Weight by Transport.

It can be seen that People using public transport, Automobile have higher median weights than people who are Walking, using Motorbike or Bike. It can be inferred that people using sedentary transportation modes (public transport, cars) tend to have higher average weights than people using bike or walking. There are a few outliers in Walking and Bike categories — individuals heavier than the typical range for those groups. These could represent exceptions (e.g., people with obesity who still walk or cycle).

4.2.4 MTRANS and target

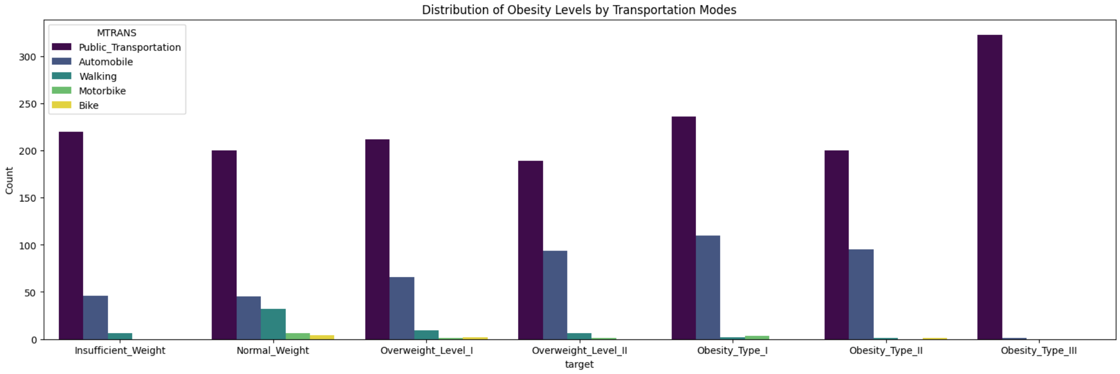


Figure 5: Distribution of Obesity Levels by Transportation modes.

A Large Percentage of people using Public transportation tend to be in the category of Obesity Type III while a significant percentage of people using Automobiles lie in the category of Overweight level II, Obesity Type I or Obesity type II. People walking or using Bike or Motorbike predominantly lie in the category of Normal weight. It indicates that the mode of transport used has an impact on weight of a person and in turn the obesity level.

4.2.5 MTRANS and Gender

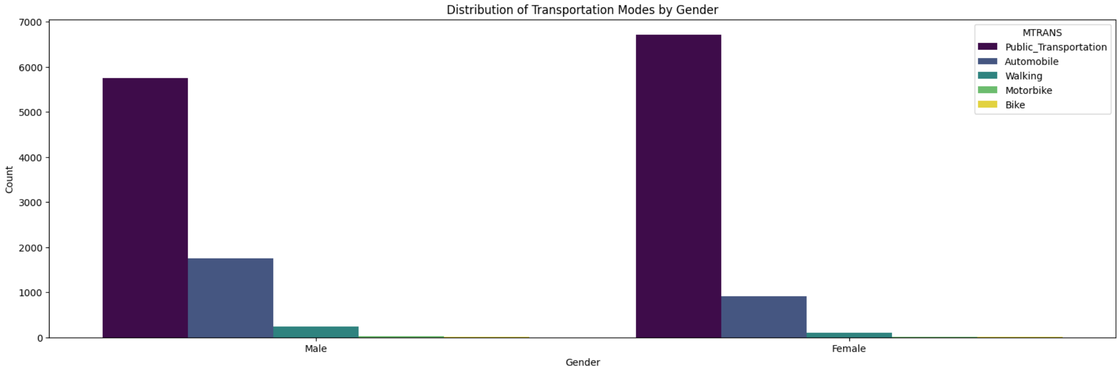


Figure 6: Distribution of Transportation modes by Gender.

As seen earlier in Figure 3 severe obesity is more common among females, whereas males tend to exhibit moderate obesity. Here it can be seen that the number of females availing public transport is slightly higher than the number of males availing public transportation. The number of males who prefer to walk is slightly higher than the number of females who prefer to walk, further justifying the high obesity among females.

4.2.6 SMOKE and target

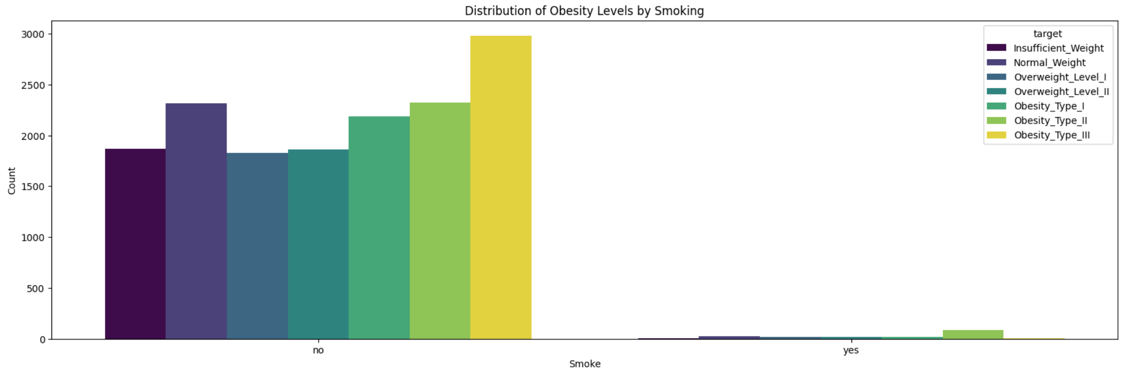


Figure 7: Distribution of Obesity Levels by Smoking.

From the Figure it can be seen that people who are non smoking tend to have more obesity issues than people who are smoking.

4.2.7 FAMILY HISTORY WITH OVERWEIGHT

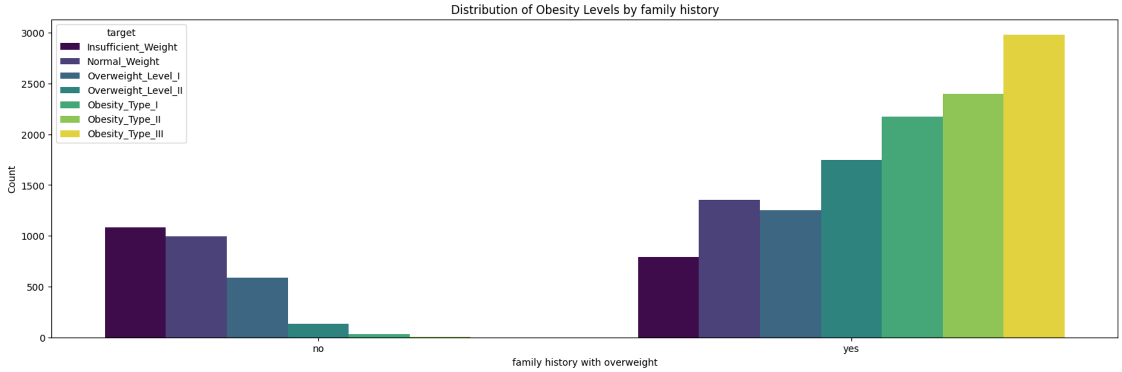


Figure 8: Distribution of Obesity Levels by family history.

It can be seen that a large number of people having family history of overweight tend to suffer from obesity or overweight more than people having no such family history. Indicates that this factor has an influence on the obesity level.

4.2.8 Age and Target

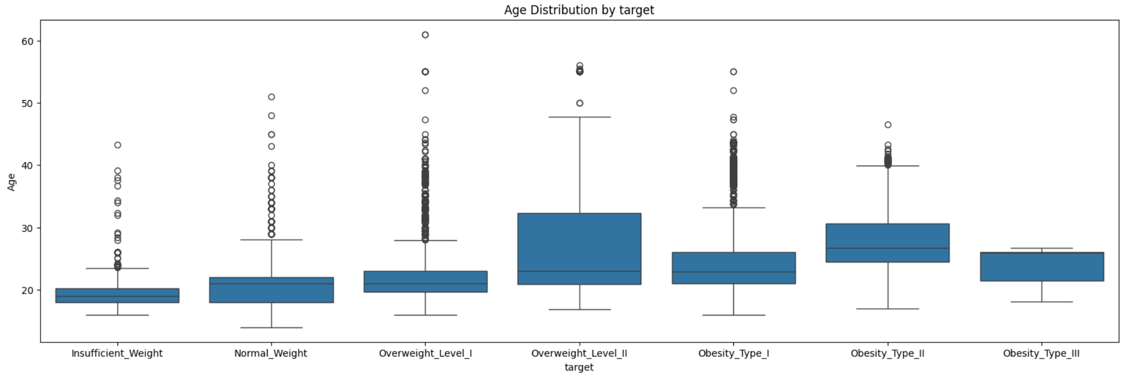


Figure 9: Distribution of age by target.

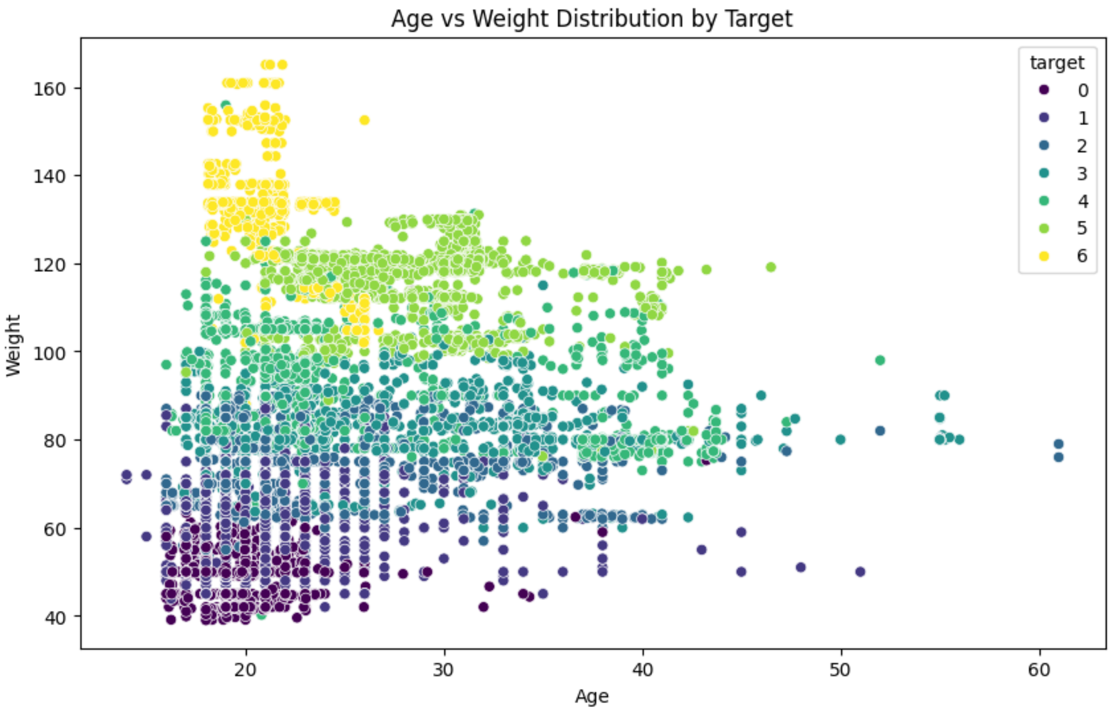


Figure 10: Age vs Weight Distribution by target.

From Figure 9 and 10 it can be seen that most datapoints are concentrated between ages 15 to 30 with fewer participants above 40. It can be seen that younger individuals tend to exhibit higher incidence of obesity compared to older participants.

4.2.9 CH2O(consumption of water daily) and target

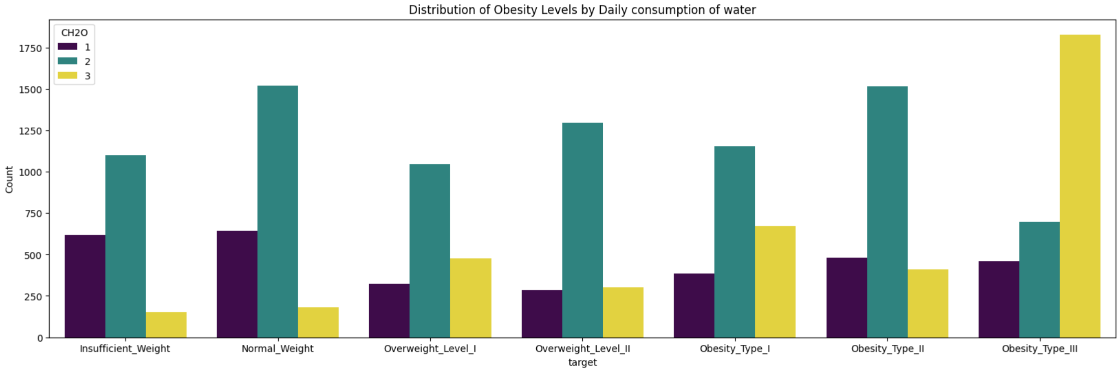


Figure 11: Distribution of obesity levels by daily consumption of water.

It can be seen that most people across all categories(Insufficient weight to Obese) report moderate water consumption(CH2O=2). Low water intake(CH2O=1) can be observed across all categories but at lower proportion. High water consumption(CH2O=3) appears more common among individuals in the category of Obesity Type III. Individual with low and high water intake appear across all categories.

4.2.10 CALC(Consumption of alcohol) and target

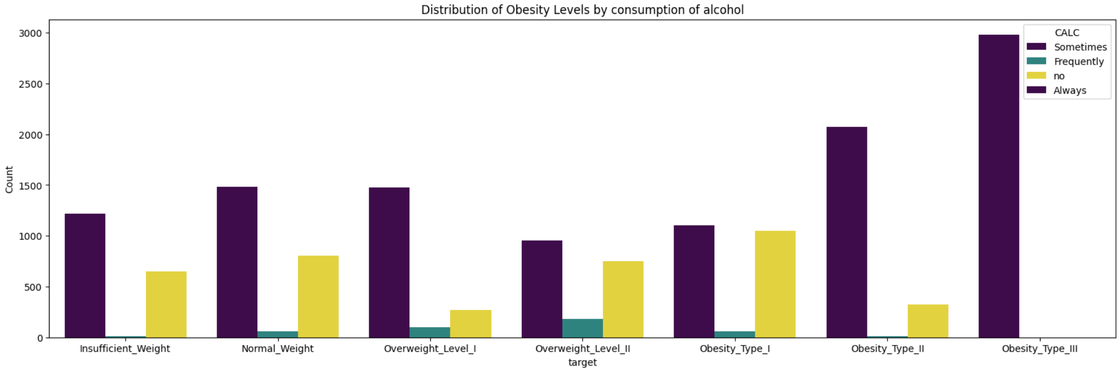


Figure 12: Distribution of obesity levels by consumption of alcohol.

From Figure 12 it can be seen that most of the individuals in Obesity type III consume alcohol sometimes. Very less people who do not consume alcohol fall in Obesity Type III. They mostly belong to the other categories. Number of people who frequently consume alcohol is very less across the categories.

4.2.11 FAVC(frequent consumption of high calorie food) and Target

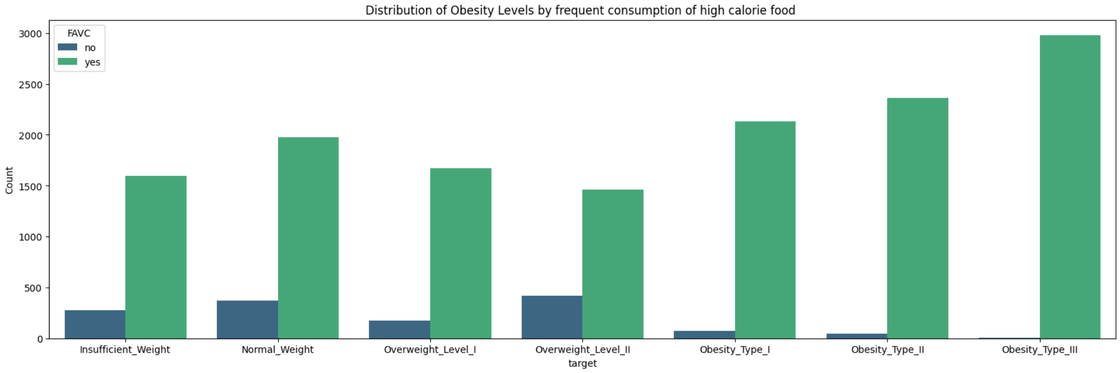


Figure 13: Distribution of obesity levels by frequent consumption of high calorific food.

It can be seen that a large proportion of people who consume high calorie food belong to the class Obesity Type III and Obesity Type I. However individuals who consume high calorie food frequently are also there in other categories like Insufficient Weight and Normal weight.

4.2.12 FCVC(Frequency of consumption of vegetables) and target

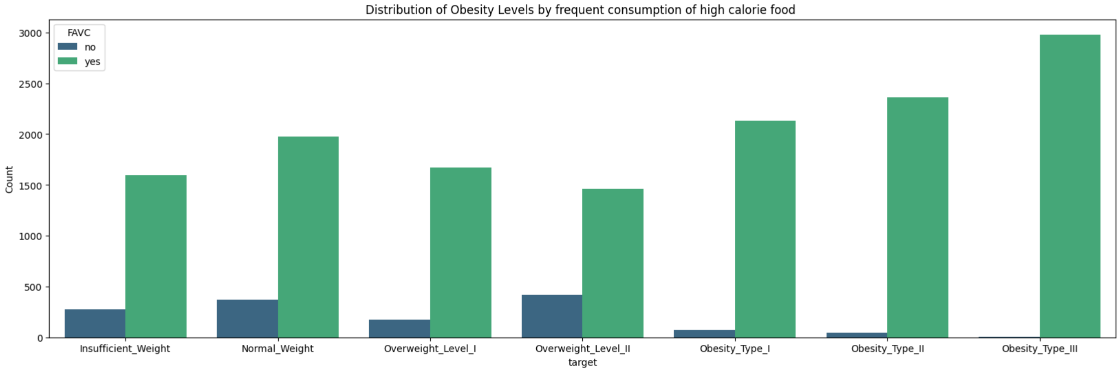


Figure 14: Distribution of obesity levels by frequency of consumption of vegetables.

It can be seen that a large percentage of individuals who consume vegetables more frequently lie in the category of Obesity Type III. Individuals who consume vegetables frequently are also present in other categories. For this dataset it can be seen that people who consume vegetables less frequently or moderately (FCVC=1, FCVC=2) do not suffer from obesity type III, however they are present in other categories.

4.2.13 FCVC and FAVC

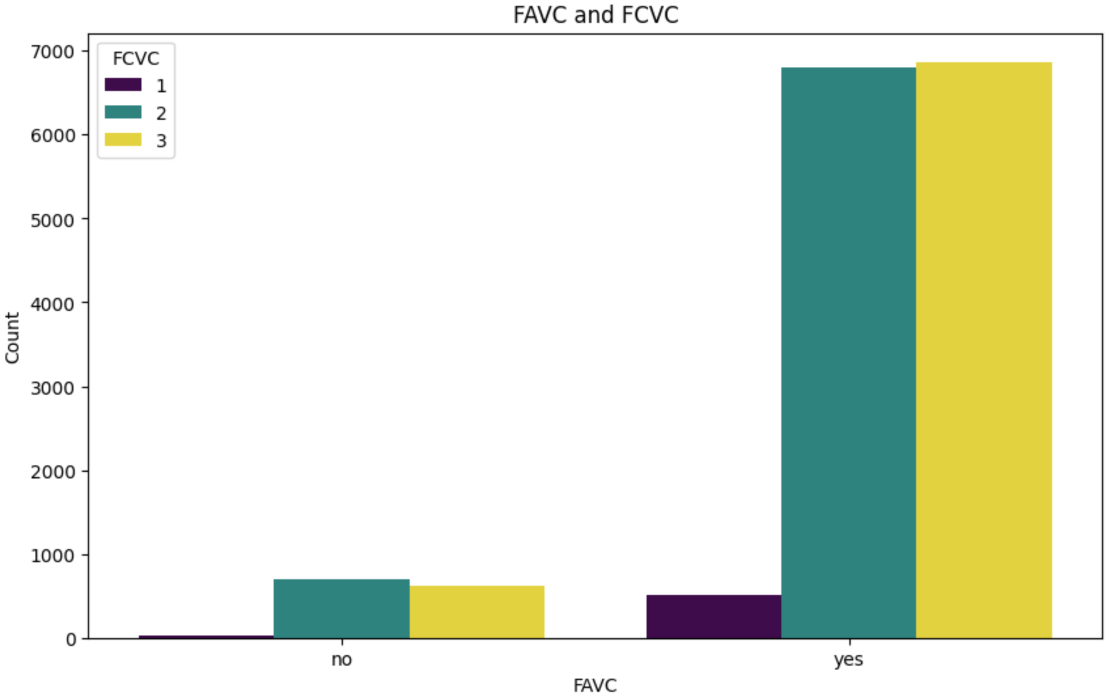


Figure 15: FAVC vs FCVC.

It can be seen that for a large percentage of people- individuals who consume high calorie food frequently also consume vegetables frequently(FCVC =2 or 3). Proportion of people who consume vegetables frequently and do not consume high calorie food is less.

4.2.14 FAVC vs FAF and MTRANS

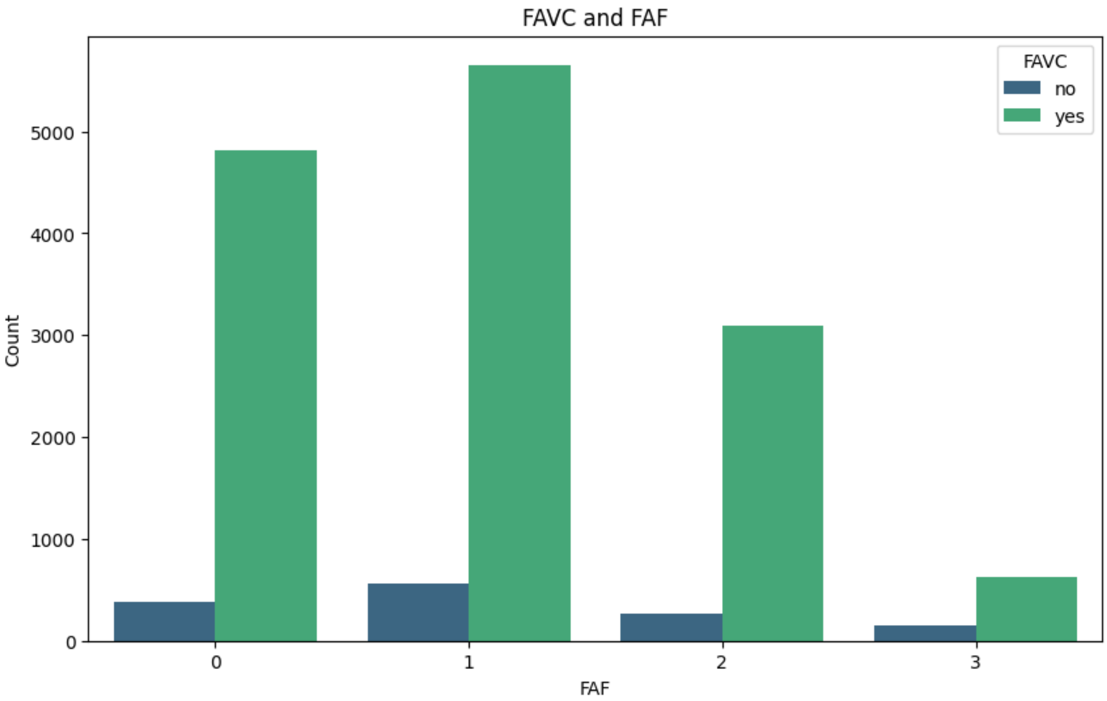


Figure 16: FAVC vs FAF.

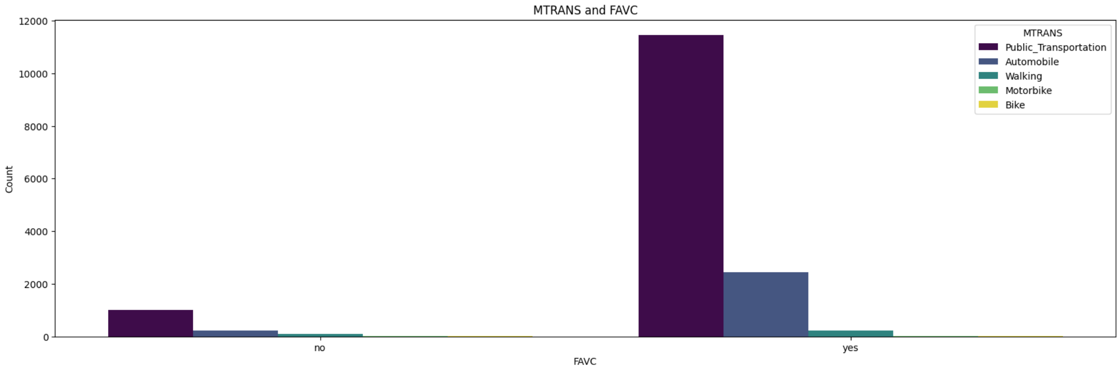


Figure 17: FAVC vs MTRANS.

From Figure 16 and 17 it can be seen that a large proportion of people who consume high calorie food frequently dont do Physical activity frequently(FAF=0 or 1) and use public transport or automobile. However there are individuals who consume high calorie food frequently but also do physical activity frequently (FAF=2 or 3) and Walk. This can justify the presence of individuals who consume high calorie food frequently but are also there in categories like Insufficient Weight and Normal weight.

4.3 Correlation Analysis

To further quantify the relationships between numerical features, Pearson correlation coefficients were computed and visualized using a heatmap.

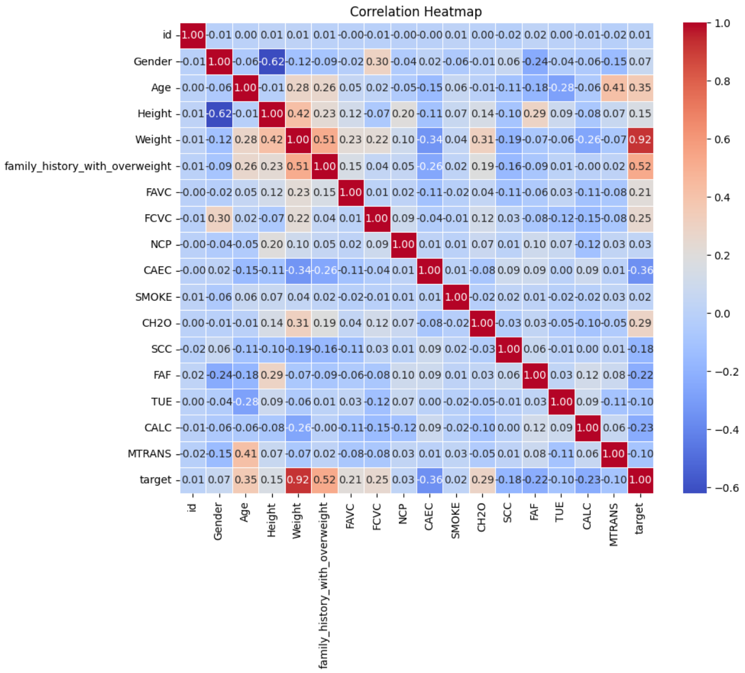
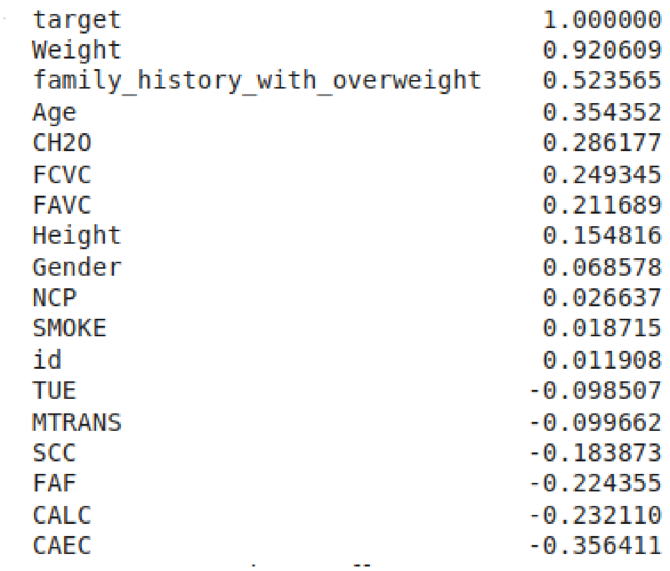


Figure 18: Correlation heatmap showing relationships among features.



```
target                               1.000000
Weight                               0.920609
family_history_with_overweight       0.523565
Age                                  0.354352
CH2O                                 0.286177
FCVC                                 0.249345
FAVC                                 0.211689
Height                               0.154816
Gender                               0.068578
NCP                                  0.026637
SMOKE                                0.018715
id                                   0.011908
TUE                                 -0.098507
MTRANS                              -0.099662
SCC                                 -0.183873
FAF                                 -0.224355
CALC                                -0.232110
CAEC                                -0.356411
```

Figure 19: Correlation values of each feature with the target variable.

Weight exhibited the highest positive correlation ($r = 0.92$), making it the most dominant predictor. **Family history with overweight** ($r = 0.52$) and **Age** ($r = 0.35$) also showed moderate positive correlations, indicating that both genetic predisposition and age contribute meaningfully to obesity risk. Behavioral factors such as **CH2O** (daily water consumption), **FCVC** (vegetable consumption), and **FAVC** (high-calorie food intake) displayed weak positive correlations.

On the other hand, **FAF** (physical activity frequency), **SCC** (calorie monitoring), and **MTRANS** (mode of transportation) exhibited weak to moderate negative correlations, suggesting that increased physical activity and conscious calorie tracking are associated with lower obesity levels. **CALC** (alcohol consumption) and **CAEC** (eating between meals) also showed negative correlations, indicating potential inverse behavioral associations. Variables such as **Gender**, **Smoking**, and **NCP** (number of main meals) had negligible correlations, implying limited predictive significance.

Overall, the correlation analysis confirmed that physical and genetic factors (Weight, Family History, Age) are the strongest determinants of obesity, while lifestyle-related features contribute secondary but complementary effects.

4.4 Outlier Analysis

Outlier detection was conducted to identify potential extreme or anomalous values in the continuous variables — **Age**, **Weight**, and **Height**. The Interquartile Range (IQR) method was employed on the *training data only* to prevent data leakage from the test set. The IQR was computed as follows:

$$\text{IQR} = Q_3 - Q_1, \quad \text{Lower Bound} = Q_1 - 1.5 \times \text{IQR}, \quad \text{Upper Bound} = Q_3 + 1.5 \times \text{IQR}$$

Samples lying outside these bounds were flagged as potential outliers.

Table 1: Outlier detection results on the training data

Feature	**No. of Outliers**	**% of Training Data**	**Comment**
Age	760	5.4%	Many older participants flagged as “outliers”
Weight	0	0%	Normal distribution
Height	4	< 0.5%	Minor anomalies

Further investigation revealed that most of the samples flagged as outliers in **Age** corresponded to valid obese participants, primarily from obesity-related target classes (3–5). This indicated that the statistical method was incorrectly identifying biologically valid samples as outliers due to the naturally skewed distribution of age in the dataset.

So no outliers were removed. The analysis was retained as part of data exploration, but all samples were preserved to maintain:

- Class balance across obesity levels,
- The natural demographic variability among participants, and
- The true representativeness of the dataset.

Consequently, all subsequent preprocessing steps, including feature scaling and model training, were performed on the complete dataset.

5 Initial Data Preparation:

There are 3 basic steps performed before the data is ready to be trained by out ML models.

1. **Combining Original and given dataset**: The given dataset was combined with the original obesity dataset containing 2111 samples thus increasing the no of rows to 17644.

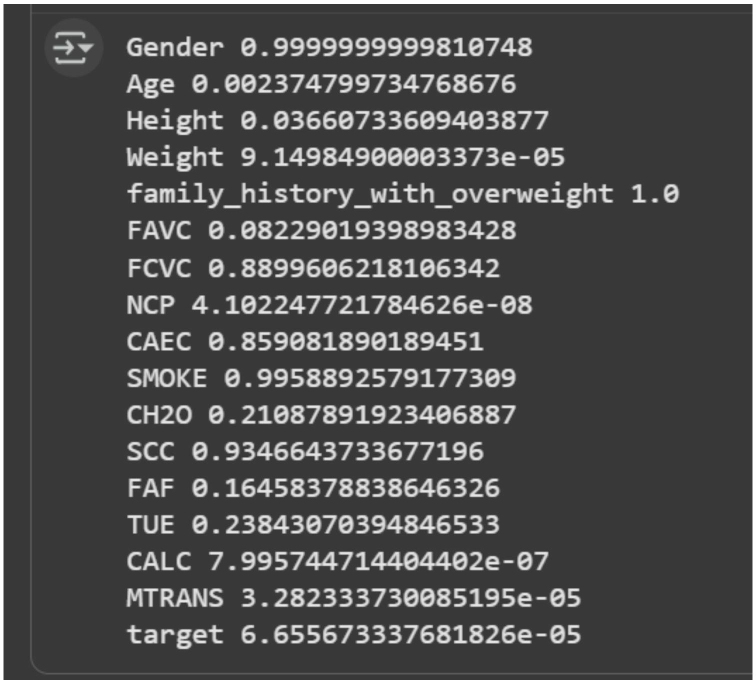
2. **Scaling**: As mentioned above Standard Scaler is used for scaling the continuous columns of the data. Standard scaler is not too sensitive to outliers and hence gives better scaling of the data according to the model as required.

3.**Train Test Split**: train_test_split is done the combined dataset, which will split the data to 80% train set and 20% test set. This will allow us to test the model for its accuracy and other requirements.

5.1 Kolmogorov–Smirnov (KS) Test

The Kolmogorov–Smirnov (KS) test is a non-parametric test used to determine whether two independent samples are drawn from the same distribution. In this study, the KS test was applied to compare the distributions of corresponding features between the two datasets. The test compares the cumulative distribution functions (CDFs) of the two samples, and the resulting p-value indicates whether the difference between them is statistically significant. A small p-value (typically less than 0.05) suggests that the two samples are likely drawn from different distributions.

It was observed that most of the variables in the dataset are **nominal** in nature, such as *GENDER*, *FAMILY*, *SMOKE*, *MTRANS*, *FAVC*, and *SCC*. These variables represent categorical values without any inherent ordering (for example, *GENDER* specifies *Male* or *Female*). The remaining variables are either **ordinal** or **continuous**, as described in the earlier sections.



```
Gender 0.9999999999810748
Age 0.002374799734768676
Height 0.03660733609403877
Weight 9.14984900003373e-05
family_history_with_overweight 1.0
FAVC 0.0822901939898342８
FCVC 0.8899606218106342
NCP 4.102247721784626e-08
CAEC 0.859081890189451
SMOKE 0.9958892579177309
CH2O 0.2108789192340688７
SCC 0.9346643733677196
FAF 0.16458378838646326
TUE 0.23843070394846533
CALC 7.995744714404402e-07
MTRANS 3.282333730085195e-05
target 6.655673337681826e-05
```

Figure 20: KS test result.

From the KS test results, it can be seen that the p-values for nominal features indicate that their distributions differ with respect to the target variable. This variation is **expected and desirable**, as it reflects meaningful class distinctions that contribute to the model's predictive ability. Conversely, for most of the ordinal and continuous features, the p-values are greater than 0.05, suggesting that their distributions are statistically similar across the two datasets. Therefore, these datasets can be safely **combined** without introducing distributional bias.

5.2 Principal Component Analysis (PCA) on Original vs Given Data

To further validate the similarity between the original and given datasets, a two-component Principal Component Analysis (PCA) was performed. The transformed data for both datasets were plotted on the first two principal components to visually examine their distribution in the reduced feature space.

As shown in Figure 21, there is a significant degree of overlap between the two datasets. This indicates that the major sources of variance in both datasets are captured in a similar manner. The overlapping regions in the PCA plot confirm that both datasets share a similar underlying structure.

Hence, both the Kolmogorov–Smirnov (KS) test results and the PCA visualization consistently suggest that the original and given datasets exhibit similar distributional characteristics and can be **combined safely** without introducing bias or distortion in the data.

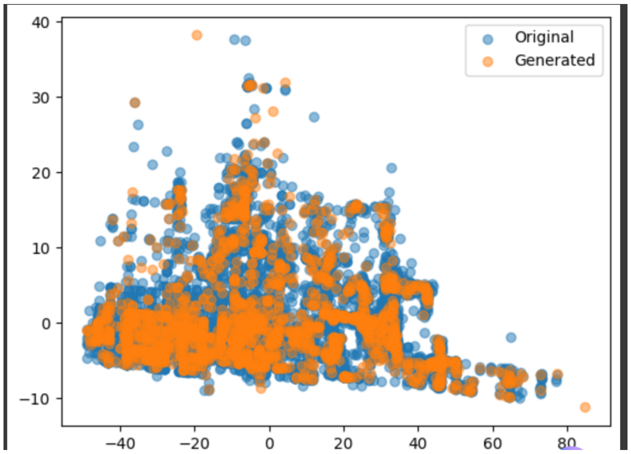


Figure 21: PCA on Original and Given Data

6 Metrics Used in the following Project:

1. **Accuracy**: It is the measure of actual number of proper predictions among the total predictions as given out by the model . The formula is given by :

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

OR

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Explanation of Terms:
TP: True Positives
TN: True Negatives
FP: False Positives
FN: False Negatives

2. **Precision**: Precision is the measure of Actual number of true positives among the total predictions given as positive by the model. In multiclass modelling it calculated for each class.

$$\text{Precision} = \frac{TP}{TP + FP}$$

3. **Recall**: it is the measure of actual true positives among the total number of positive predictions given in the dataset. In Multiclass data, recall is calculated with respect to every class.

$$\text{Recall} = \frac{TP}{TP + FN}$$

4. **F1Score**: The F1 Score is a metric that combines Precision and Recall into a single number, providing a measure of a model's accuracy that is more robust than simple accuracy, especially on datasets with uneven class distributions (imbalance)

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

OR

$$\text{F1 Score} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

5.**Macro Average and Micro average**: The Macro Average calculates the metric independently for each class and then takes the unweighted mean of the results. The Micro Average aggregates the contributions of all individual true positives (TPs), false positives (FPs), and false negatives (FNs) across all classes to compute the final metric.

Macro F1 is given by:

$$\text{Macro-F1} = \frac{1}{N}\sum_{i=1}^{N} \text{F1}_i$$

Micro F1 is given by:

$$\text{Micro-F1} = \frac{2 \cdot \sum_{i=1}^{N} TP_i}{2 \cdot \sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i + \sum_{i=1}^{N} FN_i}$$

Where N is the number of classes, and TP_i, FP_i, and FN_i are the counts for the i-th class.

7 Hyperparameter tuning

: The search for the best parameters is performed with the help of Scikit Learn's package called RandomizedSearchCV. RandomizedSearchCV uses a very different approach from GridSearchCV. It samples fixed number of values from specified search space and performs search accordingly, unlike grid search which searches for all possible combinations specified by the user as done by GridSearchCV.

RandomizedSearchCV needs a dictionary, usually referred as Parameter grid in general where the required parameter search space is specified by the user. We specify the number of iterations, the cross-validation folds and the scoring feature we need to look for like :

- F1macro(`F1-macro`)
- Accuracy (`accuracy`)
- Receiver Operating Characteristic Area Under the Curve (One-vs-Rest)(`ROC-AUC-ovr`)

RandomizedSearchCV uses the search space specified by the user and gives out the best possible output parameters and the best estimator (the model) which can be used for further processing and prediction.

8 Comparative analysis of various models:

In this project, we performed a comparative analysis of 3 basic algorithms: KNN, Random Forest, and XGBoost.

8.1 KNN

KNN or the K Nearest Neighbors is an algorithm that is non-probabilistic and also does not require training. K Nearest Neighbors uses the concept of Euclidean Distance.

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

K-Nearest Neighbors (KNN) Algorithm and Implementation

The algorithm takes a major parameter, K, and calculates the distance of a given data point to all training points. It then selects the K training points with the minimum distance. Since the data involves **multiclass prediction**, the K-Nearest Neighbors (KNN) algorithm selects the most voted class among the K neighbors as its final output.

For our specific dataset, we have chosen the Ball Tree algorithm implementation. This method utilizes a value called δ (radius) to efficiently select the best neighbors that are near the given data point, specifically those whose **Manhattan distance** is less than or equal to the radius δ.

The formula for Manhattan-distance (or L_1 norm) is given by:

$$d_1(\mathbf{p}, \mathbf{q}) = \sum_{i=1}^{n} |p_i - q_i|$$

where $\mathbf{p}$ and $\mathbf{q}$ are two data points in an n-dimensional space, and p_i and q_i are their i-th coordinates.

Limitations of KNN

A major drawback of the standard KNN algorithm is its susceptibility to several issues, including:

- **Sensitivity to Outliers**: Outliers can disproportionately influence the final prediction.
- **Sensitivity to Irrelevant Features**: Features that do not contribute to classification can skew the distance calculation.
- **Computational Expense**: The need to calculate the distance to all training points makes it expensive in terms of computation, especially for large datasets.

Model Performance and Parameters

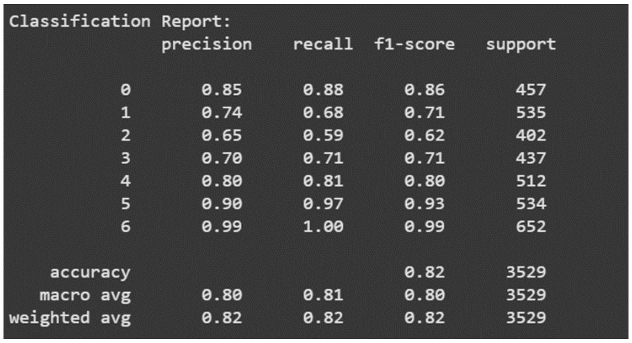
For our dataset, the final model was optimized using the following set of parameters:

Best Parameters:

```
{'weights': 'distance',
 'p': 2,
 'n_neighbors': 14,
 'metric': 'manhattan',
 'algorithm': 'ball_tree'}
```

The **accuracy** achieved by this optimized model is approximately 82.0345%, and the complete

Classification report



```
Classification Report:
              precision    recall  f1-score   support

           0       0.85      0.88      0.86       457
           1       0.74      0.68      0.71       535
           2       0.65      0.59      0.62       402
           3       0.70      0.71      0.71       437
           4       0.80      0.81      0.80       512
           5       0.90      0.97      0.93       534
           6       0.99      1.00      0.99       652

    accuracy                           0.82      3529
   macro avg       0.80      0.81      0.80      3529
weighted avg       0.82      0.82      0.82      3529
```

Figure 22: KNN Classification Report

The f1-score and f1-macro show that the model cannot differentiate the target classes very well and can be observed as under-fitted to classes 1, 2, and 3.

8.2 Random Forest Algorithm

Random Forest is an ensemble algorithm based on decision trees that uses the concept of **bagging**. Bagging combines two terms: **Bootstrapping** and **Aggregation**.

Bagging Components

- **Bootstrapping**: This is the technique of collecting M (total number of data points in the dataset) samples from the given dataset. These samples are drawn with replacement, meaning they could contain repeated data points.
- **Aggregation**: This involves considering the outputs of various estimators (individual decision trees) as specified by the user while training a Random Forest model and combining them (e.g., by voting or averaging) to produce the final output.

The process of Bagging ensures the effect of new data and outliers is nullified due to its very nature of Bootstrapping. In this process, new data or outliers get distributed among samples drawn from the dataset, and there exist multiple estimators (models) that train on different sample sets, making these outliers less likely to affect the overall prediction.

Random Forest Parameters

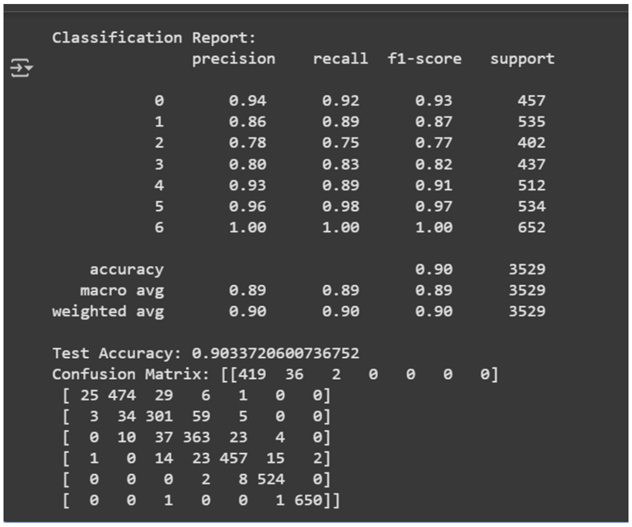
The key parameters used for the Random Forest algorithm are:

1. **N_estimators**: This is the number of estimators (decision trees) required in the ensemble.
2. **Max_depth**: It is the maximum depth of the constructed tree.
3. **Min_samples_split**: The minimum number of samples required at a node to be split.
4. **Min_samples_leaf**: It is the minimum number of samples required to be present in a leaf, which is resulted from a split.
5. **Max_features**: It is the maximum number of features required to be considered while building an estimator in the Random Forest.
6. **Bootstrap**: The process of bootstrapping, which can be set to true or false.

The parameters `max_depth`, `min_sample_leaf`, and `min_sample_split` are primarily used for regularization.

Model Performance

The test **Accuracy** achieved is approximately **90.3372**%.



```
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.92      0.93       457
           1       0.86      0.89      0.87       535
           2       0.78      0.75      0.77       402
           3       0.80      0.83      0.82       437
           4       0.93      0.89      0.91       512
           5       0.96      0.98      0.97       534
           6       1.00      1.00      1.00       652

    accuracy                           0.90      3529
   macro avg       0.89      0.89      0.89      3529
weighted avg       0.90      0.90      0.90      3529

Test Accuracy: 0.9033720600736752
Confusion Matrix: [[419  36   2   0   0   0   0]
 [ 25 474  29   6   1   0   0]
 [  3  34 301  59   5   0   0]
 [  0  10  37 363  23   4   0]
 [  1   0  14  23 457  15   2]
 [  0   0   0   2   8 524   0]
 [  0   0   1   0   0   1 650]]
```

Figure 23: Random Forest Classification Report

The classes 1, 2, and 3 seem to still have a low `f1-score` compared to other classes, but show improvement with respect to the KNN model. Hence, it can be concluded that the Random Forest model might still be affected slightly by underfitting, but not as badly as the KNN model.

8.3 XGBoost

XGBoost Algorithm

eXtreme Gradient Boosting referred to as XGBoost, is a Gradient Boosting Algorithm , along with Regularization. It uses the concept of Gradient Boosting, which is a additive modeling technique, It follows the following steps to build an ensemble:

1. A preliminary model provides an initial prediction.

2. In each subsequent iteration, the algorithm computes pseudo-residuals, which are the gradients (first derivatives) of the objective loss function.

3. A new decision tree is then trained specifically on these pseudo-residuals, aiming to minimize the loss and correct the existing error.

XGBoost enhances this standard technique by leveraging the second-order Taylor expansion of the loss function. This allows the algorithm to utilize not only the gradient but also the Hessian (second derivative) to analytically determine the precise, optimal leaf weights for the newly added tree.

*Basic Formula of XGBoost:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, F_{t-1}(\mathbf{x}_i) + f_t(\mathbf{x}_i)\right) + \Omega(f_t)$$

Where :
$\mathcal{L}^{(t)}$:The overall objective function being optimized at step
$t.F_{t-1}(\mathbf{x}_i)$:The prediction from the previous stage (the sum of all trees built before
$t).f_t(\mathbf{x}_i)$: The new decision tree being added at the current stage t.

Hyperparameter used in XGBoost:

1. **n_estimators**: The total number of estimators you need in your XGBoost model

2. **max_depth**: Maximum depth of the trees built

3. **learning_rate**: The learning rate in order to allow the fractional of leaf weights to be considered in subsequent stages

4. **subsample**: random fraction of data taken from the training set for training.

5. **colsample_bytree**: controls the fraction of features (columns) to be randomly sampled when building each decision tree

6. **gamma**: Crucial regularization hyperparameter for min_split_loss

7. **min_child_weight**: it is a regularization hyperparameter that controls the minimum number data points required in a leaf node for a split to be considered valid.

8. **reg_alpha**: L1 regularization parameter

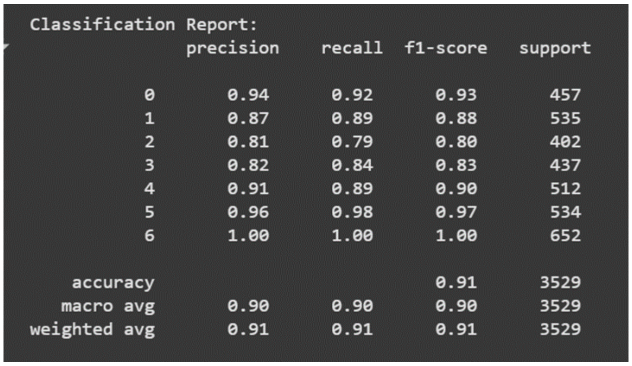
9. **reg_lambda**: L2 regularization parameter

Model Performance

*Best Parameters Used

The optimal hyperparameters found for the XGBoost model are:

```
{'subsample': 0.8,
 'reg_lambda': 2,
 'reg_alpha': 0.01,
 'n_estimators': 300,
 'min_child_weight': 3,
 'max_depth': 9,
 'learning_rate': 0.03,
 'gamma': 0.2,
 'colsample_bytree': 0.6}
```



```
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.92      0.93       457
           1       0.87      0.89      0.88       535
           2       0.81      0.79      0.80       402
           3       0.82      0.84      0.83       437
           4       0.91      0.89      0.90       512
           5       0.96      0.98      0.97       534
           6       1.00      1.00      1.00       652

    accuracy                           0.91      3529
   macro avg       0.90      0.90      0.90      3529
weighted avg       0.91      0.91      0.91      3529
```

Figure 24: XGBoost Performance Classification Report

Test Accuracy was 91.0456% which is much better than Random Forest.

The following classification report shows better class differentiation and f1macro of 0.90 and f1micro as 0.91 with respect to target, hence XGBoost performed better than Random Forest for the following dataset.

9 Conclusion

By the above Comparative analysis, XGBoost is chosen as the best suitable model for the given data , enchanced with regularization which helps to build a good model which has no overfitting.

10 Github link to notebook

The code to generate predictions:

https://github.com/karishita/MT2025011_MT2025052_Classification/blob/main/Generate_Predictions.ipynb

Notebook containing code for preprocessing and EDA:

Github: https://github.com/karishita/MT2025011_MT2025052_Classification/blob/main/Preprocessing_and_EDA.ipynb

Colab: https://colab.research.google.com/drive/1xLuFnZ2HvDC2_U1rL526x6PhNgrGpxkl?usp=sharing

Notebook containing code for model training

Github: https://github.com/karishita/MT2025011_MT2025052_Classification/blob/main/Model_Training.ipynb

Colab: https://colab.research.google.com/drive/1YeEyDQeXwx6WmhqYwhNWvUaoCCI_UPWa?usp=sharing