INFOSYS SPRINGBOARD INTERNSHIP 5.0

DOMAIN : ARTIFICIAL INTELLIGENCE

(BATCH – 1)

REPORT ON

“ PREDICTING OBESITY LEVELS USING ML ”

SUBMITTED BY

(TEAM 1)

TEAM MEMBERS

ISHA NAIK
ADITHIYAN PV
KRITHIK NAVEEN A R
ABDUL LATEEF
ARCHANA T
NEHIL PATIL
AKSHAY NEHETE

ABSTRACT

Currently, safeguarding the community is vital in terms of finding solution to health related problems which can be achieved through medical research using the advent of technology. Obesity has become worldwide health concern as it is becoming a threat to the future. It is the most common health problems all over the world. Thousands of diseases as well as risks and death are associated to it. An early prediction of a disease will help both doctors and patients to act and minimize if not total eradication of the root cause or work on preventing the disease symptom from further deterioration. Going through patient's medical history is one of the methods of identifying a disease which most time consuming as processing manually and it comes with an error-prone analyses and expense. Therefore, there is need to scientifically develop a predicting model of the occurrence of the disease or its existence using an automated technique as it is becoming a need of the day. In this research work, we used machine learning techniques on a public clinical available dataset to predict obesity status using different machine learning algorithms. Varoius machine learning algorithms were applied:, Random Forest Classifier, XGBoost, Decision Tree Classifier, Support Vector Machine, Logistic Regression, K-Nearest Neighbor, Single Layer Perceptron, Multilayer Perceptron, ANN and and the model has shown promising results with as XGboost classifier achieves the highest accuracy of 97.35% as compared to other classifiers. Meanwhile, the Random Forest gave the relatively strong accuracy of 96.66 %.

Keywords: Obesity, Health prediction, Overweight, Machine learning models, Random Forest, XGBoost, Decision Tree Classifier, Support Vector Machine, Logistic Regression, K-Nearest Neighbor, Single Layer Perceptron, Multilayer Perceptron, ANN

TABLE OF CONTENTS

1. INTRODUCTION

More than 2.1 billion people worldwide are shuddering with overweightness or obesity, which represents approximately 30% of the global population. This condition poses a serious global health problem. By 2030, 41% of people will likely be overweight, and 20% will be obese if current trends continue. This growing epidemic presents significant challenges for healthcare systems globally. According to the World Health Organization (WHO), factors such as high caloric intake, sedentary lifestyles, and transportation habits are major contributors to obesity.

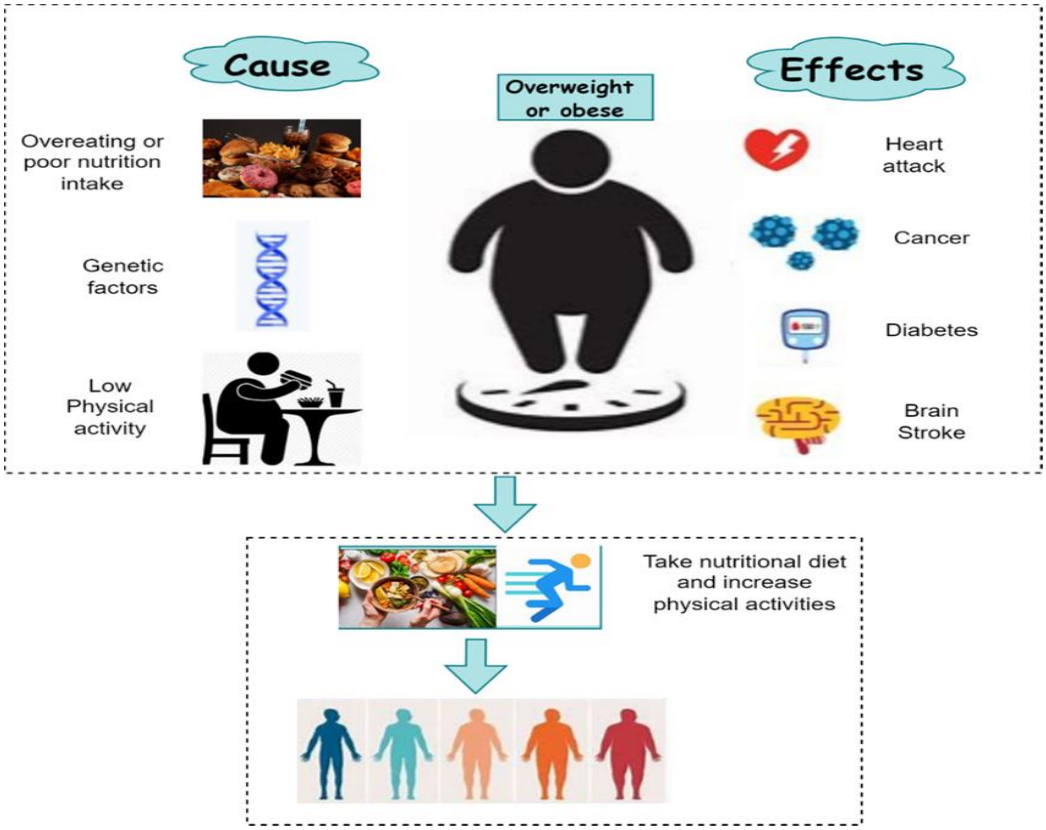


Fig 1: Causes and Effects of Obesity

Obesity is a worldwide public health issue influenced by factors such as dietary choices, physical activity, and lifestyle habits. People who show indications of weight increase or obesity run the risk of contracting life-threatening conditions, including type 2 diabetes, respiratory issues, heart disease, and stroke. Some intervention strategies, like regular exercise and a balanced diet, are essential to preserving a healthy lifestyle. Thus, it is crucial to identify obesity as soon as feasible.

Accurate prediction of obesity levels is vital for early intervention and the development of personalized health strategies. Traditional statistical approaches often struggle to capture the complex relationships

between the many factors contributing to obesity. Machine learning (ML), however, offers a promising alternative. By analyzing large datasets, ML can uncover valuable insights, enabling personalized treatments, predicting future obesity risks, and informing health policies. ML techniques are particularly well-suited for identifying patterns in this type of data.

This study aims to implement machine learning algorithms, particularly XGBoost along with various other models, on a dataset that includes attributes such as age, weight, height, physical activity, and dietary habits. The goal is to create robust models that can classify individuals into different obesity categories, thereby enhancing the effectiveness of health interventions.

2. OBJECTIVES

- Develop a machine learning model to accurately predict obesity levels based on individuals' lifestyle and physical attributes.
- Identify key features (e.g., eating habits, physical activity, gender, weight) that significantly influence obesity levels.
- Preprocess the dataset to handle missing values, normalize features, and prepare it for machine learning models.
- Train and compare the performance of various machine learning models, such as Decision Trees,KNN, Random Forest, XGBoost, SVM, and others to find the most effective one.
- Perform hyperparameter tuning on the models to optimize their performance and improve prediction accuracy.
- Evaluate the models using various metrics like accuracy, precision, recall, F1-score, and confusion matrix to measure their effectiveness.
- Visualize the confusion matrix to analyze the model's misclassifications and understand which obesity levels are commonly confused.
- Provide insights that can help healthcare professionals identify at-risk individuals based on their lifestyle patterns and make data-driven decisions.
- Document the entire process, including data preparation, model development, and results, to present findings in a structured and understandable manner.

3. LITERATURE SURVEY

Machine learning has been successfully applied in numerous healthcare domains, including obesity prediction. Previous studies have shown that factors such as caloric intake, physical activity, and family history are significant predictors of obesity, making early detection and intervention crucial for mitigating associated health risks.

Smith & Nguyen (2022) showed that Random Forest models can predict obesity with 85% accuracy, identifying diet and physical activity as key predictors for early detection in preventive healthcare.

García & Hernández (2021) achieved 87% accuracy using XGBoost, outperforming K-Nearest Neighbours (KNN) and Decision Trees, and highlighted hyperparameter tuning's importance for large, imbalanced datasets.

López-Martínez et al. (2020) employed decision trees and logistic regression to predict obesity levels with notable success, highlighting caloric intake and family history as key factors.

Similarly, Menéndez et al. (2018) used machine learning techniques such as random forests and support vector machines (SVM) on adolescent datasets to predict obesity levels, underscoring the influence of lifestyle factors.

Li & Kumar (2019) used ensemble methods like Gradient Boosting to achieve 83% accuracy, emphasizing these methods' robustness in noisy datasets.

Nascimento et al. (2019) demonstrated the superiority of XGBoost in handling large, complex datasets, as it iteratively corrects the errors of previous models through boosting.

More recent studies have leveraged advanced machine learning methods like XGBoost and ensemble models to improve predictive performance. This study builds on the strengths of previous research by applying XGBoost and Random Forest to a larger and more diverse dataset, with a focus on feature engineering, data exploration, and preprocessing.

4. SYSTEM ARCHITECHTURE

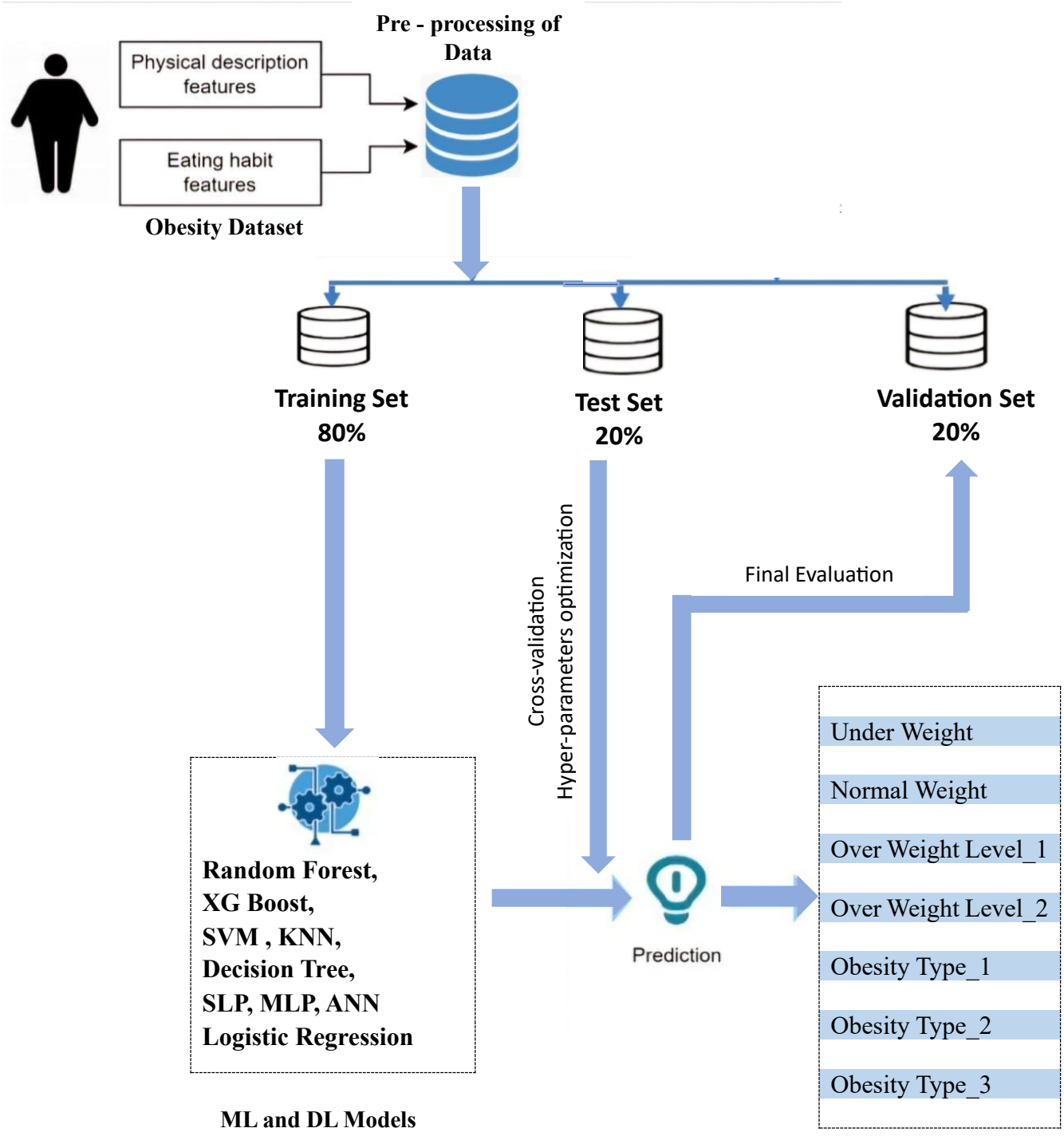


Fig: Overall System Architecture

5. METHODOLOGY

The goal of this project is to predict obesity levels based on personal, lifestyle, and health-related factors using various machine learning models, with a focus on achieving the highest possible accuracy. Below is the step-by-step methodology used to build, train, and evaluate the models.

Fig: Detailed Architecture of Building , Training and Evaluation of the ML Models

- **EXPLORATORY DATA ANALYSIS (EDA) - Data Understanding and Exploration**

1. Obesity Dataset

The obesity dataset contains 34,597 samples and 17 attributes such as:

Demographics: Age, Gender

Health Metrics: Height, Weight, Family history of obesity

Eating Habits: Number of meals, Frequency of vegetable consumption, Snacking habits

Lifestyle Patterns: Physical activity, Water consumption, Alcohol use, Smoking habits, and Mode of transportation.

Target Variable: The obesity level (labeled as NObeyesdad) is a categorical variable with seven categories:

1. Insufficient Weight
2. Normal Weight
3. Overweight (Level I and II)
4. Obesity (Type I, II, and III)

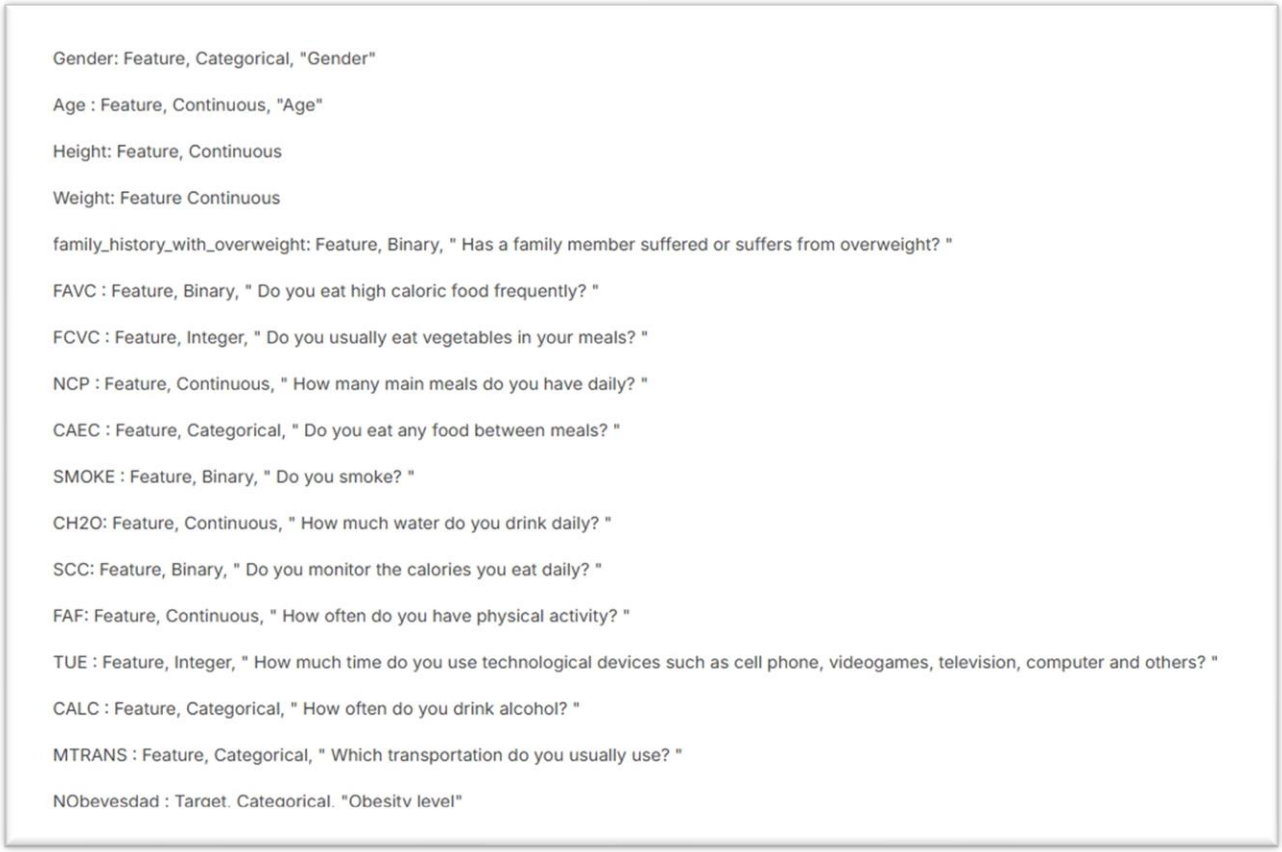


Fig : 17 Attributes and its Categories

Initial Analysis:

Plots such as histograms and count plots were created to study the distributions of numerical and categorical variables.

Through this exploration, class imbalance and the need for feature normalization were identified.

Univariate Analysis

Univariate analysis involves examining the distribution of each variable individually , which can help identify potential outliers, anomalies, or other insights.

- **Numerical Variables**: Used histogram to understand the distribution of continuous variables like age, weight , height, etc.

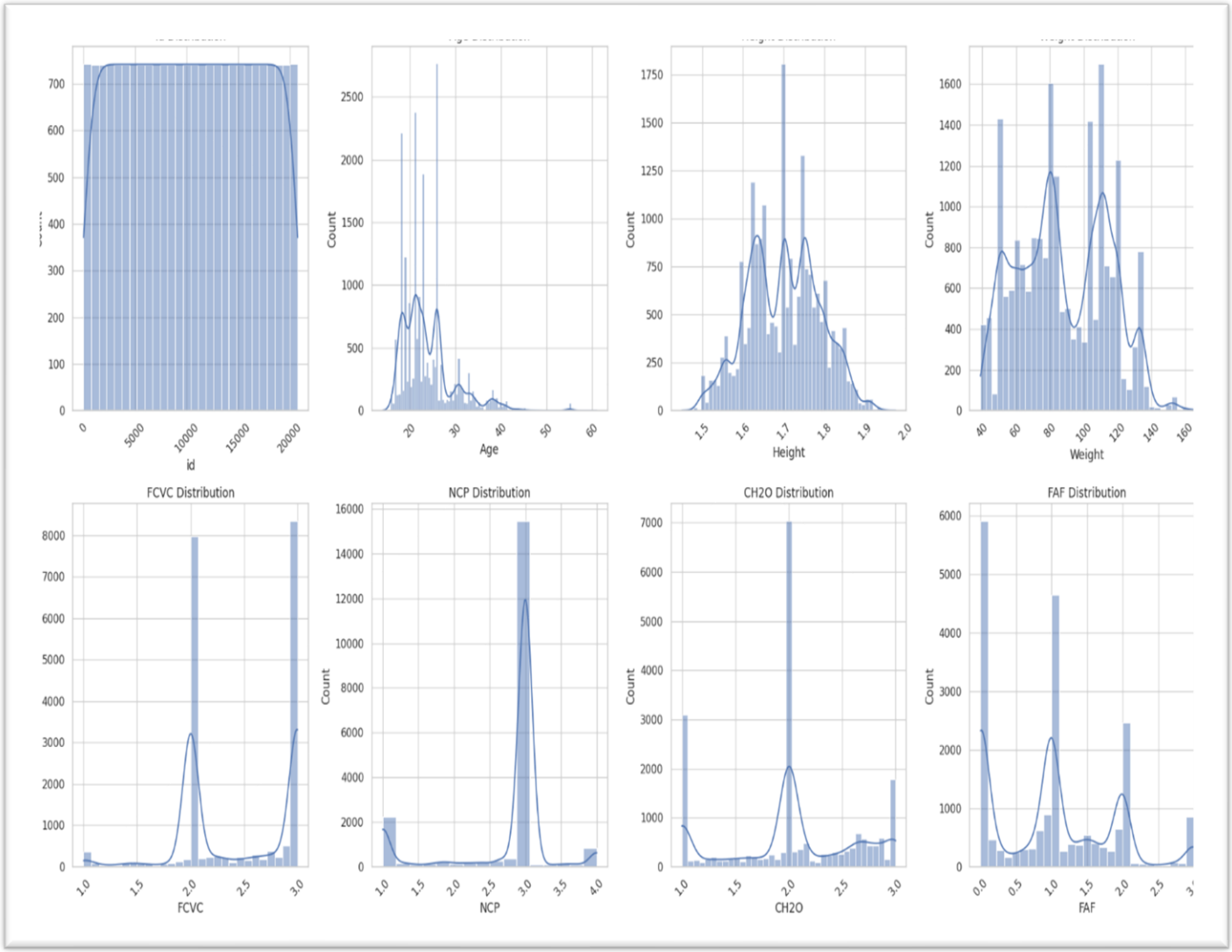


Fig: Numerical Data Visualization

- **Categorical Variables:** Count plots are used to visualize the frequency of categories in a categorical variable like gender distribution , obesity levels.

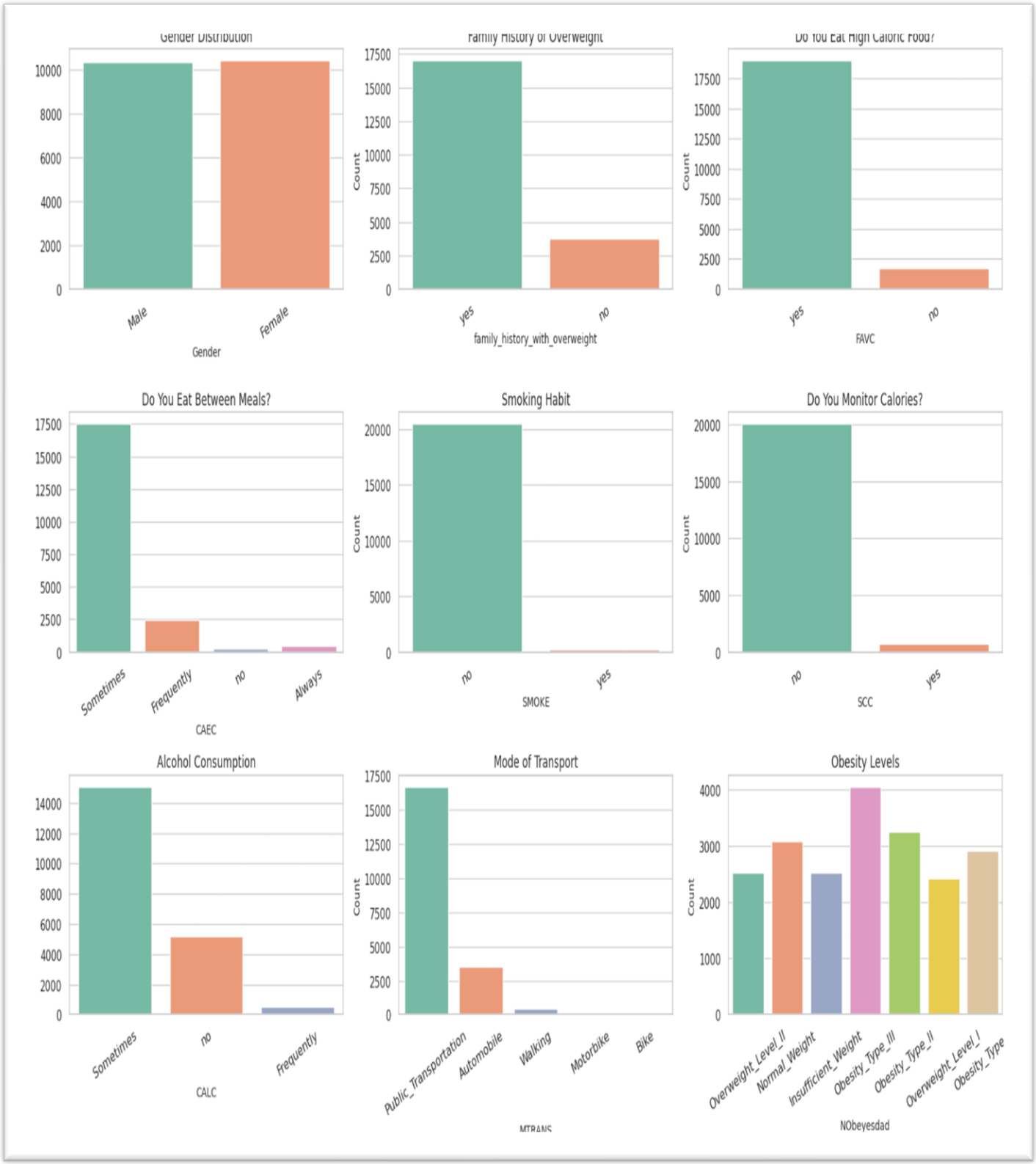


Fig: Countplots for Categorical Variables

Overall Insights:

1. **Patterns in Lifestyle Factors:** Features like calorie monitoring, caloric food consumption, alcohol intake, and mode of transport show behaviors that are likely associated with obesity.

2. **Potential Predictors:** Family history, eating between meals, and activity-related features like mode of transport may strongly influence obesity classification.

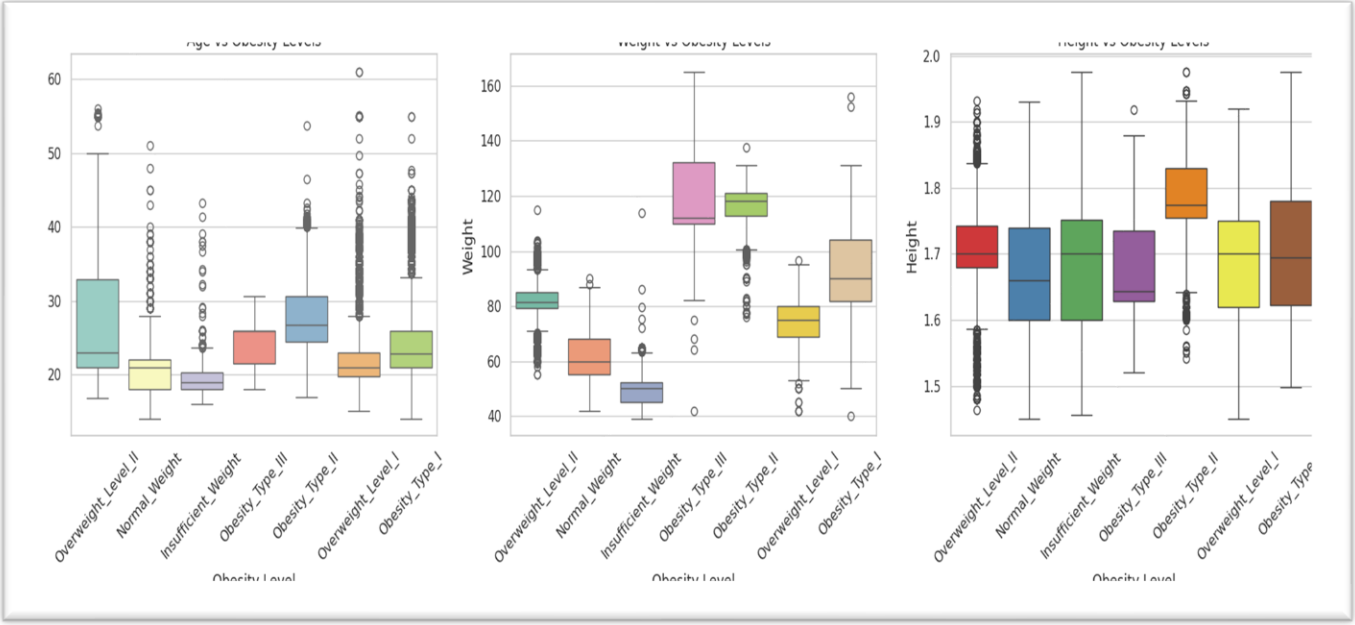
3. **Target Variable Distribution:** The distribution of obesity levels is important for model performance, as an imbalance might affect the classifier's ability to predict less represented classes accurately.

These insights can guide feature selection and model training by focusing on the most relevant categorical factors linked to obesity risk.

Bivariate Analysis

Bivariate analysis is a statistical method used to explore the relationship between two variables which helps to identify patterns, associations, or dependencies between the variables, which can aid in understanding how one variable may affect or relate to another.

Fig: Boxplots - Obesity Levels vs. Continuous Variables



Box plots are used to show the distribution of Obesity Levels across different continuous variables such as age , weight and height.

1. Age vs. Obesity Level:

The age distribution across obesity levels shows how age groups vary with obesity risk.

2. Weight vs. Obesity Level:

Weight distributions for each obesity level reflects a clear distinction between normal, overweight, and obese categories.

3. Height vs. Obesity Level:

Heights across obesity levels shows whether height has any notable influence on obesity classification.

General Insights:

Outliers: Observed extreme values (outliers) within these plots that helped in identifying unusual cases, which may impact model predictions.

These insights aid in understanding how key features relate to obesity and inform feature selection, as weight appears most directly correlated with obesity classification, followed by age.

Correlational Analysis

Correlational analysis is a statistical method used to measure the strength and direction of the relationship between two or more variables. It helps to determine if and how strongly pairs of variables are related to each other.A correlation heatmap is a visual tool used in correlation analysis to show the relationships between continuous variables in a dataset. It uses color gradients to represent the strength and direction of correlations, making it easier to identify which variables are positively or negatively correlated.

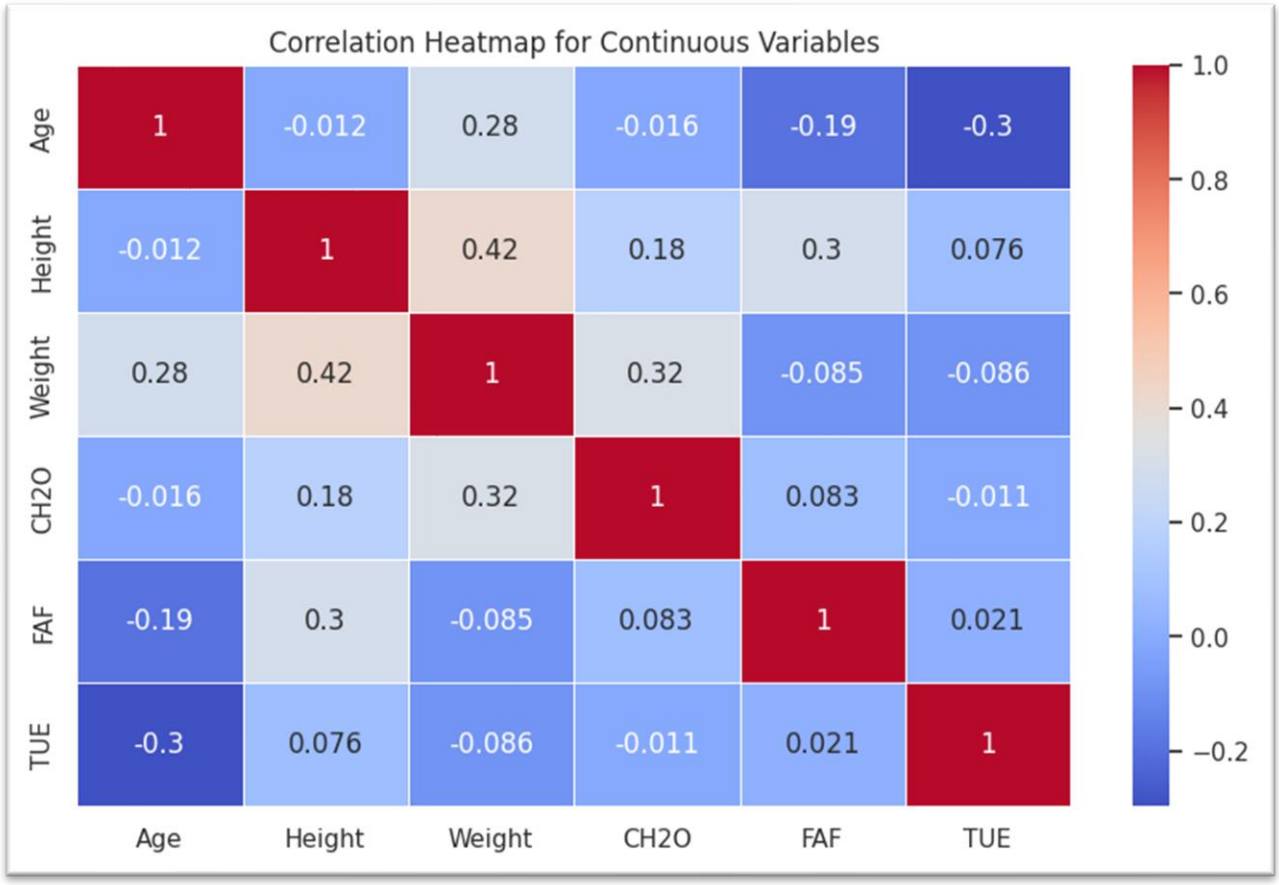


Fig: Correlation Heatmap for Continuous Variables

Heatmap is used to visualize the relationship between continuous variables like age, weight, height , water consumption(CH2O) , physical activity(FAF) , time spent on physical devices (TUE)

Observation from this heatmap:

Age:

Age doesn't show a strong correlation with most other features, indicating it may independently influence obesity levels without significant interdependence on these continuous variables.

Height and Weight:

There is a moderate to strong positive correlation between Height and Weight. This makes sense since taller individuals may generally weigh more.

The correlation of Weight with other variables (like FAF for physical activity frequency or CH2O for water consumption) explores how physical activity might affect body weight.

Water Consumption (CH2O):

CH2O (water intake) have weak to moderate correlations with certain variables like Weight or FAF. This correlation suggests links between hydration and obesity indicators or fitness routines.

Physical Activity Frequency (FAF):

FAF shows a weak or moderate negative correlation with Weight, suggesting that increased physical activity is associated with a decrease in weight, aligning with general fitness and obesity insights.

FAF also correlate with CH2O, as individuals engaging in more activity might also consume more water.

Technology Use (TUE):

TUE (time using technological devices) displays minimal correlation with other variables.This indicates that screen time independently impacts obesity without strong direct associations with physical measures like Height or Weight.

These observations highlights the potential predictors and indirect relationships in understanding factors that influence obesity.

2. Data Preprocessing

- **Encoding Categorical Variables:**
  - Binary variables (like Gender) were converted into 0/1 using one-hot encoding.
  - Multi-class variables (like MTRANS, CAEC) were label-encoded to convert categories into numbers.
  -

- **Handling Missing Data:**

Although this dataset didn't contain missing values, a check was performed to ensure data integrity.

- **Feature Scaling :**

Feature scaling is a preprocessing technique used to standardize or normalize data, bringing all features onto a similar scale. This is essential in machine learning, especially for models sensitive to feature magnitudes to ensure that each feature contributes proportionately, improving model performance and convergence.

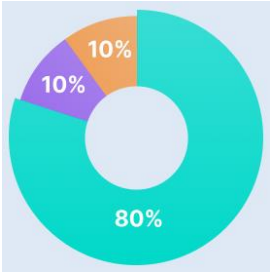
Normalizing Features:

Since some features (like Weight and Age) were highly skewed, applied Yeo-Johnson transformation to make the data more Gaussian-like.

➢ **Outlier Removal:**

Outliers were identified using the Interquartile Range (IQR) method, and rows with extreme values (such as CALC and NCP) were removed to prevent them from skewing the model's predictions.

3. Splitting the Dataset



Train-Validation-Test Split:

The dataset was divided into:

- 80% training data (used to build the model).
- 10% validation data (used to fine-tune hyperparameters and monitor performance during training).
- 10% test data (used for final evaluation).

This ensures the model is evaluated on unseen data, reducing the risk of overfitting.

4. Model Selection and Training

A range of models was selected for predicting obesity levels, leveraging their proficiency in managing large datasets with a combination of numerical and categorical features. Each model offers unique benefits, contributing to a comprehensive analysis of obesity prediction.

Model 1: Random Forest

A Random Forest is an ensemble of decision trees. It works by aggregating the predictions of multiple trees to improve accuracy and reduce overfitting.

➢ **Model Training**

- **Initialization**: Imported the necessary libraries, including RandomForestClassifier from sklearn.
- **Model Creation**: Instantiated the Random Forest Classifier model, selecting hyperparameters such as the number of trees (n_estimators), the maximum depth of trees (max_depth), and other relevant parameters.

- 100 trees (n_estimators=100) were trained, with each tree limited to a maximum depth of 20 to prevent over-complexity.
- **Model Fitting**: Trained the Random Forest model using the training dataset. The model learns to associate input features with the target variable (obesity status) by creating multiple decision trees through bootstrapping and aggregating their predictions.

➢ **Hyperparameter Tuning**

- **Objective:** Optimize RF performance by identifying the best hyperparameter combination.
- **Method: Bayesian Optimization** balances exploration and exploitation for efficient tuning.
- **Search Space:**
  - **n_estimators:** 100 to 300 (number of trees).
  - **max_depth:** 10 to 30 (tree depth).
  - **min_samples_split:** 2 to 10 (minimum samples to split nodes).
  - **min_samples_leaf:** 1 to 4 (minimum samples at leaf nodes).
  - **criterion:** Split quality metric (`gini` or `entropy`).
- **Tuning Tool:**
  - **BayesSearchCV:** Configures RF with a defined search space.
  - **n_iter=10:** Tests 10 hyperparameter combinations.
  - **cv=5:** Uses 5-fold cross-validation for robust evaluation.
  - **Scoring Metric:** Maximizes accuracy (`scoring='accuracy'`).
- **Best Parameters:** Retrieved using `best_params_`.
- **Final Model:** A new RF model (`best_rf`) is built using optimal hyperparameters, ensuring improved performance.

Why Random Forest ?

Random Forest is used for its ability to improve predictive accuracy and control overfitting by averaging the results of multiple decision trees, thus leveraging the strengths of ensemble learning.

Model 2: XGBoost

XGBoost (Extreme Gradient Boosting) is a boosting algorithm that builds models sequentially, where each new model corrects the errors made by the previous ones.

➢ **Model Training**

- **Initialization**: Imported the XGBClassifier from xgboost
- XGBoost was trained with 100 boosted trees and a maximum depth of 20 to control complexity.
- **Training the Model**: Trained the XGBoost model using the training dataset, leveraging its gradient boosting approach for accurate obesity prediction through the ensemble of multiple decision trees.

➢ **Hyperparameter Tuning**

- **Objective:** Perform an exhaustive search to find the best hyperparameters for XGBoost.
- **Search Space:**
    - **n_estimators:** Boosting rounds (100, 200).
    - **max_depth:** Tree depth (3, 5).
    - **learning_rate:** Boosting step size (0.1, 0.01).
    - **subsample:** Fraction of training samples per tree (0.8, 1.0).
    - **colsample_bytree:** Fraction of features sampled per tree (0.8, 1.0).
- **Cross-Validation:**
    - **cv=3:** Used for faster experimentation.
    - **cv=5:** Final robust evaluation.
- **Subset Training:** Initial tuning on a smaller subset (`X_train[:1000], y_train[:1000]`) for speed.
- **Parallel Processing:** Uses all CPU cores (`n_jobs=-1`) for efficiency.
- **Steps:**
    1. **Initial Fit:** Quick prototyping with subset data.
    2. **Final Fit:** Optimization on full training data.
    3. **Best Parameters:** Extracted using `grid_search.best_params_`.
    4. **Model Evaluation:** Optimal model (`best_xgb`) evaluated on the test set for predictions and metrics.

Why XGBoost?:

It is known for its high accuracy and ability to handle imbalanced datasets by focusing more on difficult-to-predict samples

Model 3: Decision Tree

Decision trees are a non-parametric supervised learning technique for classification and regression that represent decisions and their outcomes in a tree-like structure, allowing for easy interpretation and visualization.

➢ **Model Training**

- **Initialization**: Imported DecisionTreeClassifier from the scikit-learn library.
- A Decision Tree Classifier from sklearn is initialized with specific hyperparameters (e.g., max_depth=9 and random_state=45).
- The training data is fed into the model using the fit() function, which helps the model learn patterns and relationships.

➢ **Hyperparameter Tuning**

- **Objective:** Optimize the Decision Tree classifier for predicting obesity levels by finding the best hyperparameters.

- **Key Hyperparameters Tuned:**

  - **max_depth:** Maximum tree depth to control complexity.
    - Values: [3, 5, 7, 9, 11].
  - **min_samples_split:** Minimum samples required to split a node, avoiding overfitting.
    - Values: [2, 5, 10].
  - **min_samples_leaf:** Minimum samples required in a leaf node to ensure robust predictions.
    - Values: [1, 2, 4].
  - **criterion:** Split quality measure.
    - Options:
      - **gini**: Measures node homogeneity.
      - **entropy**: Measures information gain.

- **Process:**

  - **Cross-Validation (`cv=5`):** Ensures robust evaluation and prevents overfitting to specific subsets.
  - **Scoring (`accuracy`):** Maximizes the proportion of correct predictions.

- **Best Model:**

  - Extracted using `grid_search.best_estimator_`.
  - Used for predictions on the test set with `predict()`.

- **Result:** Enhanced performance of the Decision Tree classifier by systematically selecting optimal hyperparameters.

Why Decision trees?

Decision Trees are chosen for tasks like obesity prediction due to their unique advantages in interpretability, handling of complex data, and flexibility.

Model 4: SVM Model (Support Vector Machine)

SVM is a supervised machine learning algorithm used primarily for classification tasks. The core idea behind SVM is to find the optimal hyperplane that separates different classes in a high dimensional space.

➢ **Feature Scaling**

- SVMs are sensitive to the scale of input data, the features are standardized using StandardScaler from sklearn. This scales the features to have a mean of 0 and a standard deviation of 1.

➢ **Model Training**

- **Initialization:** Initialized the SVM model using SVC from scikit-learn .

- An SVM model with a linear kernel is chosen for training due to its interpretability, efficiency, and effectiveness in handling linearly separable data.

- The model is trained using the fit () method on the training dataset (features and corresponding labels). This process involves finding the optimal hyperplane that separates different classes in the dataset.

➢ **Hyperparameter Tuning**

- **Objective:** Optimize the SVM classifier for enhanced prediction accuracy using a preprocessing pipeline with StandardScaler.
- **Pipeline Components:**
  - **StandardScaler:** Ensures feature scaling for consistent preprocessing.
  - **SVC Model:** Used for classification.
- **Key Hyperparameters Tuned:**
  - **svm__C:** Regularization parameter controlling the trade-off between maximizing the margin and minimizing classification errors.
    - Values: [0.1, 1, 10, 100].
  - **svm__kernel:** Specifies the SVM kernel type.
    - Options: linear, rbf, poly.
  - **svm__gamma:** Kernel coefficient for rbf and poly kernels.
    - Options: scale, auto.
  - **svm__degree:** Degree of the polynomial kernel (poly).
    - Values: [2, 3, 4].
- **Process:**
  - **Cross-Validation (cv=5):** Ensures robust performance evaluation across different data splits.
  - **Scoring (accuracy):** Maximizes correct predictions during tuning.
  - **Parallel Processing (n_jobs=-1):** Speeds up computation.
- **Outcome:**
  - **Best Hyperparameters:** Retrieved using best_params_.
  - **Optimized Model:** best_estimator_ is used for final predictions and performance evaluation.

Why SVM?

SVM effectively handles high-dimensional data and can find the optimal hyperplane for class separation, making it suitable for complex obesity classification tasks.

Model 5: Logistic Regression

Logistic regression is a statistical method used for binary classification tasks, where the goal is to predict the probability that a given input belongs to a particular category.

➢ **Feature Scaling**

- Standardized the features using StandardScaler to ensure all variables are on the same scale, as Logistic Regression is sensitive to feature scaling. This helps improve convergence and accuracy.

➢ **Model Training**

- **Initialization:** Imported the LogisticRegression class from sklearn.linear_model and configured the model.
- **Regularization**: Chose an appropriate regularization strategy (L2 by default) to avoid overfitting, especially with many features. The C parameter can be tuned to control the strength of regularization.
- **Training the Model**: Fitted the Logistic Regression model on the training data to learn the relationship between features and obesity levels using fit() method.

➢ **Hyperparameter Tuning**

Objective

Optimize the Logistic Regression model for improved prediction of obesity levels using RandomizedSearchCV.

Key Hyperparameters Tuned

1. **C (Regularization Strength):**
    - Controls the trade-off between achieving a high-accuracy model and avoiding overfitting.
    - **Values Tested:** [0.001, 0.01, 0.1, 1, 10, 100, 1000].
2. **penalty (Regularization Type):**
    - Determines the type of regularization applied to the model.
        - **Options:**
            - 'l1': Lasso (L1) regularization.
            - 'l2': Ridge (L2) regularization.
            - 'elasticnet': Combination of L1 and L2 regularization.
            - 'none': No regularization.
3. **solver (Optimization Algorithm):**
    - Algorithm used for optimization, supporting different penalties.
        - **Options:**
            - 'liblinear': Supports L1 and L2.
            - 'saga': Supports L1, L2, and ElasticNet.

Search Process

- **RandomizedSearchCV Parameters:**
    - **n_iter=30:** Randomly evaluates 30 combinations of hyperparameters for efficient exploration.
    - **cv=5:** Uses 5-fold cross-validation to ensure robust performance evaluation.
    - **scoring='accuracy':** Accuracy is used as the performance metric.
- **Output:**
    - **Best Model:** Retrieved using random_search.best_estimator_.
    - **Best Hyperparameters:** Displayed with random_search.best_params_.

Advantages

- **Efficient Search:** RandomizedSearchCV balances computational efficiency and search space exploration.
- **Robust Evaluation:** Cross-validation minimizes overfitting and ensures generalization to unseen data.

Outcome

- **Optimal Model:** Best logistic regression model is used to predict test data.
- **Evaluation:** Achieves improved prediction accuracy with selected hyperparameters.

Why Logistic Regression?

Logistic regression is a straightforward and interpretable method for binary and multiclass classification, providing probabilities for obesity levels based on linear relationships between features.

Model 6: KNN Model (K-Nearest Neighbors)

KNN is a non-parametric and instance-based learning algorithm used for classification and regression tasks.

- **Feature Scaling**
    - StandardScaler was applied to normalize the features, ensuring all variables are on the same scale for the KNN model to function effectively.
- **Model Training:**
    - The K-Nearest Neighbors (KNN) algorithm was used for training, leveraging distance-based classification to predict obesity levels.
    - The model was trained on the preprocessed data using varying values of 'k' to optimize classification accuracy.

- **Initialization:** Imported the KNeighborsClassifier class from sklearn.neighbors to create the KNN model.
- **Selecting Hyperparameters**: Determined the value of k, the number of neighbors to consider.
- **Training the Model**: Fitted the KNN model to the training data, where the model learns the feature-label mapping based on the k nearest neighbors.

➢ **Key Hyperparameters Tuned**

1. **n_neighbors (Number of Neighbors):**
   - Specifies the number of nearest neighbors considered for classification.
   - **Range:** 1 to 30 (range(1, 31)).
2. **weights (Neighbor Influence):**
   - Determines the weighting of neighbors' votes.
     - 'uniform': Equal weight for all neighbors.
     - 'distance': Neighbors closer to the query point have higher influence.
     -
3. **metric (Distance Metric):**
   - Defines the method used to measure distance between data points.
     - 'euclidean': Straight-line distance.
     - 'manhattan': Sum of absolute differences.
     - 'minkowski': Generalized metric (includes Euclidean and Manhattan as special cases).

Search Process

- **cv=5:** Uses 5-fold cross-validation to ensure robustness and mitigate overfitting.
- **scoring='accuracy':** Evaluates the percentage of correct predictions.
- The best-performing hyperparameters are identified using grid_search.best_params_.

Advantages of Tuning

- **Improved Bias-Variance Tradeoff:** Balances overfitting and underfitting by optimizing the number of neighbors (n_neighbors).
- **Customizable Distance Influence:** Allows flexibility in weighting closer neighbors more heavily.
- **Robust Evaluation:** Cross-validation ensures the model generalizes well across data subsets.

Outcome

- **Best Model:** The tuned KNN model provides improved accuracy.
- **Optimal Parameters:** Displayed for reproducibility and further analysis.

Why KNN?

KNN is a simple, intuitive algorithm that can effectively classify obesity levels based on similarity, requiring no assumptions about the data distribution.

Model 7: SLP Model (Single-Layer Perceptron)

SLP is the simplest form of a neural network, consisting of a single layer of output nodes connected directly to input features.

- **Feature Scaling:**
  - Neural networks, including Single Layer Perceptrons, perform better when the input data is standardized. The features are scaled using StandardScaler from Scikit-learn, which transforms the data to have a mean of 0 and a standard deviation of 1.
- **Model Training:**
  - **Initialization:** A Single Layer Perceptron (SLP) model is selected using MLPClassifier from Scikit-learn.
  - The model consists of one hidden layer with 10 neurons and uses the ReLU activation function to introduce non-linearity.
  - **Training the Model**: The SLP model is trained on the scaled training data (X_train and y_train) using the fit() method. The model iteratively adjusts its weights to minimize the loss and improve the classification performance.
- **Hyperparameter Grid**
  - **Hidden Layer Sizes:** Number of neurons in the single hidden layer: `(5,)`, `(10,)`, `(15,)`.
  - **Activation Functions:**
    - `relu`: Rectified Linear Unit.
    - `tanh`: Hyperbolic Tangent.
    - `logistic`: Sigmoid function.
  - **Alpha:** Regularization strength to prevent overfitting: `0.0001`, `0.001`, `0.01`.
  - **Learning Rate (learning_rate_init):** Initial learning rates for the optimizer: `0.001`, `0.01`, `0.1`.
  - **Maximum Iterations (max_iter):** Number of iterations for model training: `500`, `1000`.

Search Process

- **Model:** Single-Layer Perceptron Classifier (`MLPClassifier`).
- **Search Method:** GridSearchCV for exhaustive search across hyperparameter combinations.
- **Cross-Validation (cv):** 5-fold validation to ensure model robustness.
- **Metric (scoring):** Accuracy is used as the evaluation criterion.
- **Parallel Processing (n_jobs):** All available processors are utilized to accelerate the tuning process.

Results and Evaluation

- **Best Hyperparameters:** Extracted using `grid_search.best_params_`.
- **Optimal Model:** Retrieved as `grid_search.best_estimator_`.

Why SLP?

SLP is a basic neural network that can model linear decision boundaries, making it useful for initial experiments with obesity prediction using neural networks.

Model 8: MLP Model (Multi-Layer Perceptron)

MLP is a type of artificial neural network that consists of multiple layers of nodes, including one or more hidden layers, making it capable of modeling complex relationships in data.

- **Feature Scaling**:
  - Standardized features to ensure that each input feature contributes equally to the MLP model. Features are scaled using StandardScaler for zero-mean and unit-variance scaling.
- **Model Training**
  - **Input Layer**: The number of neurons in the input layer matches the number of features in the dataset (e.g., 17 for the obesity dataset).
  - **Hidden Layers**: MLP with two hidden layers are used; the first with 100 neurons and the second with 50 neurons.
  - The maximum number of iterations for training: max_iter=300
  - The optimization algorithm , solver='adam' is used for weight updates due to its adaptive learning rate capabilities.
  - A seed for random number generation , random_state=42 to ensure reproducibility.
  - **Activation Functions**: activation='relu': The activation function , ReLU (Rectified Linear Unit) is used for the hidden layers as it helps with faster convergence.
- **Hyperparameter Tuning**

Key Hyperparameters Tuned

1. **`hidden_layer_sizes` (Network Architecture):**
   - Defines the number of neurons and layers in the network.
   - **Tested Configurations:**
     - `(100, 50)`: Two layers with 100 and 50 neurons, respectively.
     - `(50, 25)`: Two layers with 50 and 25 neurons.
     - `(100,)`: Single layer with 100 neurons.
     - `(50, 50, 50)`: Three layers with 50 neurons each.
2. **`activation` (Activation Function):**
   - Determines the non-linearity applied to hidden layers.
     - **Options:**

        - `'relu'`: Commonly used for deep networks.
        - `'tanh'`: For smoother non-linearities.
        - `'logistic'`: Sigmoid function, suited for binary tasks.
3. **`alpha` (Regularization Strength):**
    - Prevents overfitting by adding L2 penalty.
    - **Range:** 0.0001 to 0.01 (uniform distribution).
4. **`learning_rate_init` (Initial Learning Rate):**
    - Influences step size during weight updates.
    - **Range:** 0.001 to 0.1 (uniform distribution).
5. **`max_iter` (Training Iterations):**
    - Limits the number of optimization steps.
    - **Range:** Random values between 200 and 500.

Search and Optimization Process

- **RandomizedSearchCV:**
    - Tests **30 combinations** of random hyperparameters (`n_iter=30`).
- **Cross-Validation:**
    - **`cv=5`:** Ensures robust model evaluation and reduces overfitting risk.
- **Performance Metric:**
    - **`scoring='accuracy'`:** Maximizes accuracy during tuning.
- **Efficiency:**
    - **`n_jobs=-1`:** Utilizes all CPU cores for faster computations.
- **Reproducibility:**
    - **`random_state=42`:** Fixed seed ensures consistent results.

Outcome

- **Best Parameters:** Identified using `random_search.best_params_`.
- **Optimal Model:** Trained using the best configuration, leading to improved performance.
- **Scalability:** Supports experimentation with deeper architectures and broader parameter ranges in future iterations.

Why MLP ?

MLP model is optimized for obesity prediction, balancing accuracy and interpretability while mitigating risks like overfitting.

Model 9: Artificial Neural Network (ANN) Model

- **Feature Scaling**: Normalized the features using StandardScaler to help scale all values between a consistent range (e.g., 0 to 1).

- **Model Training**

- Created an ANN model using TensorFlow Keras with three hidden layers, each using ReLU activation.

1. **Input Layer:** The first Dense layer has 128 neurons and uses ReLU (Rectified Linear Unit) as the activation function. The input_dim is set to the number of features in the training data (X_train.shape[1]).

2. **Hidden Layers:**
   - The second Dense layer has 64 neurons and uses ReLU activation.
   - A Dropout layer is added after the second Dense layer with a dropout rate of 0.3 to prevent overfitting by randomly setting 30% of the input units to 0 during training.
   - The third Dense layer has 32 neurons and also uses ReLU activation, followed by another Dropout layer with a rate of 0.3.

3. **Output Layer:** The final Dense layer has 7 neurons (corresponding to the 7 classes of obesity levels) and uses softmax activation for multi-class classification.

- **Compiling the Model:** Compiled the model with the Adam optimizer and sparse categorical cross-entropy loss., which is suitable for integer labels in multi-class classification.

- **Training the Model:** The model is trained on the training data (X_train, y_train) for 50 epochs with a batch size of 32.

➢ **<u>Hyper-parameter Tuning</u>**

Customization of ANN Model

1. **Custom Function:**
   - A function constructs the ANN with customizable parameters for hidden layers, dropout, and learning rate.
   - **Key Hyperparameters:**
     - units1, units2, units3: Number of neurons in each hidden layer.
     - dropout_rate: Dropout rates to prevent overfitting (values: 0.2–0.4).
     - learning_rate: Learning rate for the Adam optimizer (values: 0.001, 0.01, 0.0001).
2. **Keras Compatibility:**
   - Wrapped the create_ann_model function inside a KerasClassifier to integrate with RandomizedSearchCV.

Hyperparameter Grid

- **Model Parameters:**
  - model__units1, model__units2, model__units3: Possible neuron counts for each hidden layer.
  - model__dropout_rate: Dropout rates to test.
  - model__learning_rate: Learning rate values for optimization.
- **Training Parameters:**
  - batch_size: Sizes of mini-batches (16, 32, 64).
  - epochs: Training iterations (50 or 100).

Search Process

- **RandomizedSearchCV Settings:**
  - **n_iter=10:** Randomly samples 10 hyperparameter combinations.
  - **cv=3:** 3-fold cross-validation ensures robust evaluation.
  - **scoring='accuracy':** Accuracy is the optimization metric.
  - **n_jobs=-1:** Utilizes all available processors for faster tuning.

Outcome

- **Best Model:** Extracted using random_search.best_estimator_.
- **Optimal Configuration:** Ensures the ANN achieves maximum accuracy with efficient tuning.
- **Scalable Design:** Allows for adjustments in neuron counts, learning rates, and epochs for different datasets or objectives.

Why ANN?

ANN is capable of learning intricate patterns in large datasets through its deep architecture, making it highly effective for capturing the complexities involved in obesity prediction.

6. EXPERIMENTAL ANALYSIS AND RESULTS

Model Evaluation

After training both models, their performance was evaluated using the validation and test datasets.

Evaluation Metrics Used:

- **Accuracy Score:** Percentage of correctly predicted obesity levels.
- **Classification Report:** Provided metrics such as Precision, Recall, and F1-score for each obesity class.
- **Confusion Matrix:** Visualized the number of correct and incorrect predictions across all classes.

Why These Metrics?:

Since this is a multi-class classification problem, precision, recall, and the F1-score are more informative than accuracy alone.

The confusion matrix helps identify where the model struggles (e.g., confusing obesity Type II with Type III).

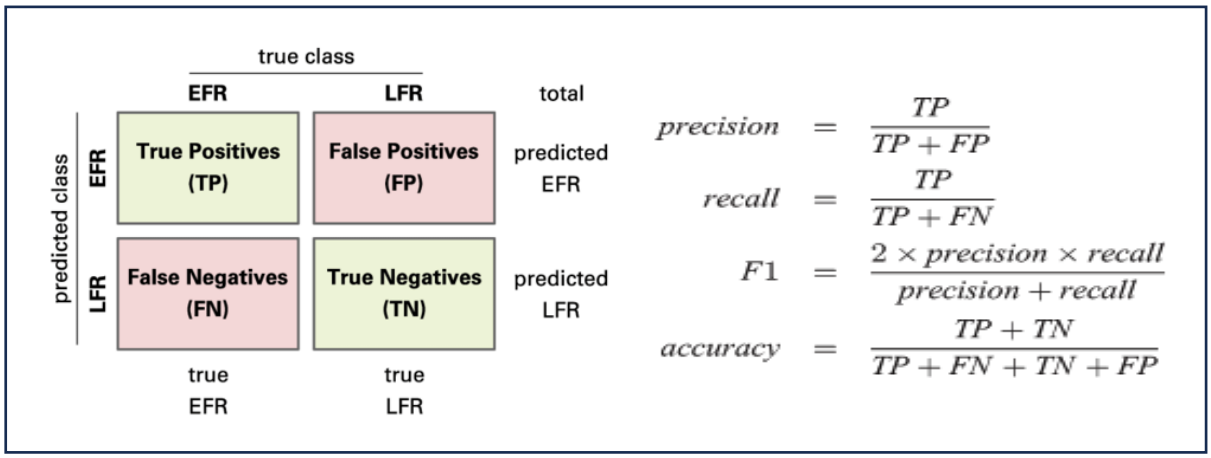


Figure : Confusion matrix and Evaluation metrix parameters

Sr no.	Approach	ALGORITHM USED	Accuracy (Before Tuning)	Accuracy (After Tuning)
1.	XGBOOST	GridSearchCV	97.35	97.49
2.	RANDOM FOREST	Bayesian Optimization	96.52	96.93
3.	DECISION TREE	GridSearchCV	95.82	96.52
4.	SLP	GridSearchCV	95.82	96.11
5.	SVM	GridSearchCV	95.96	96.10
6.	LOGISTIC REGRESSION	RandomizedSearchCV	95.13	95.54
7.	MLP	RandomizedSearchCV	94.29	94.57
8.	ANN	RandomizedSearchCV	93.18	94.29
9.	KNN	GridSearchCV	89.01	91.37

Fig: Model's Accuracy before and after Hyperparameter Tuning

➢ Classification Reports

1. XGBoost

Accuracy: 97.49 %

Classification Report:

- **High precision** for dominant classes like Obesity_Type_III and Normal_Weight, indicating the model accurately identifies these categories with minimal false positives.
- **High recall** for classes such as Obesity_Type_III and Normal_Weight, showing that most true cases of these categories were correctly predicted.
- **Moderate performance** for classes like Overweight_Level_I, with slightly lower precision and recall, indicating occasional misclassifications.
- **F1-scores** are consistent across the majority of classes, demonstrating the model's balanced performance in precision and recall.

Confusion Matrix Insights:

- The majority of predictions for **Obesity_Type_III** and **Normal_Weight** were correct, with minimal misclassifications into other categories.
- **Some confusion** occurred between closely related classes, such as Overweight_Level_I and Overweight_Level_II, likely due to overlapping feature distributions.
- Smaller classes like Obesity_Type_I showed a few misclassifications, possibly due to limited sample sizes or feature ambiguity.

Key Insights:

- After **hyperparameter tuning**, the model achieved improved accuracy and reduced overfitting by optimizing parameters like:
- **n_estimators:** Increased to balance bias and variance.
- **max_depth:** Limited to prevent over-complex trees.
- **learning_rate:** Adjusted to refine the step size during training.
- **subsample and colsample_bytree:** Fine-tuned to enhance generalization.

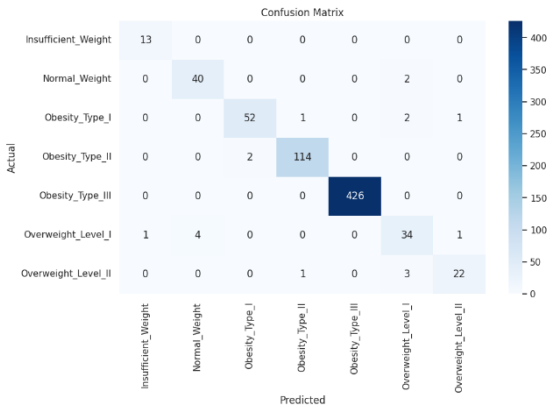
The XGBoost model exhibits high performance, particularly for the majority classes, with better differentiation across categories.

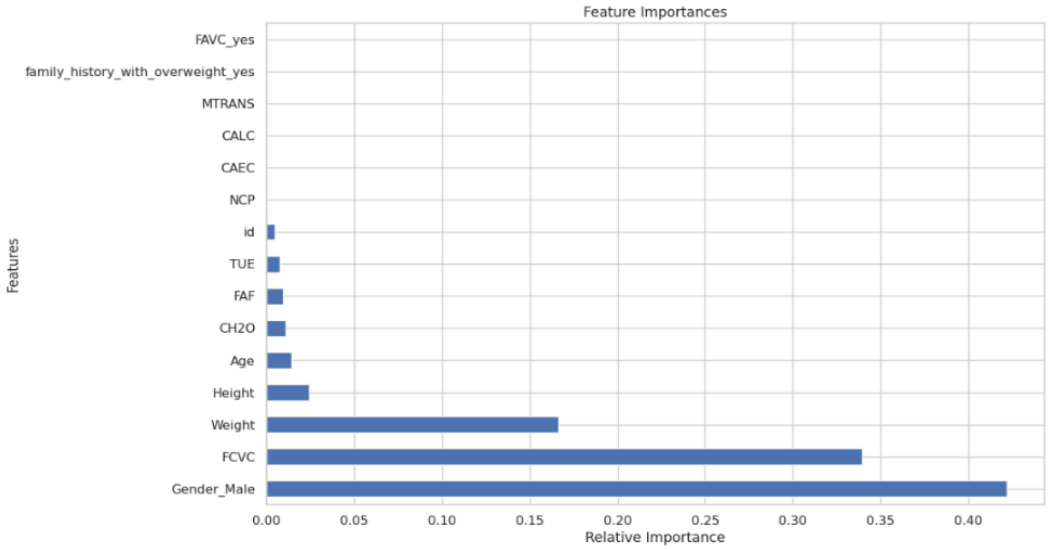
This tuned model is well-suited for robust obesity-level classification and demonstrates effective handling of both dominant and overlapping categories.

Classification Report:

	precision	recall	f1-score	support
0	0.93	1.00	0.96	13
1	0.91	0.95	0.93	42
2	0.96	0.93	0.95	56
3	0.98	0.98	0.98	116
4	1.00	1.00	1.00	426
5	0.83	0.85	0.84	40
6	0.92	0.85	0.88	26
accuracy			0.97	719
macro avg	0.93	0.94	0.93	719
weighted avg	0.98	0.97	0.97	719

Accuracy: 0.9749652294853964





2. Random Forest

Accuracy: 96.93 %

Classification Report:

- **High precision** for prominent classes like Obesity_Type_III and Normal_Weight, indicating accurate predictions with very few false positives.
- **High recall** for classes such as Obesity_Type_III, meaning most actual cases in these categories were identified correctly.
- **Moderate performance** for classes like Overweight_Level_I and Overweight_Level_II, showing slightly lower recall and precision due to feature overlaps between these categories.
- **Balanced F1-scores** across most classes, reflecting a consistent trade-off between precision and recall.

Confusion Matrix Insights:

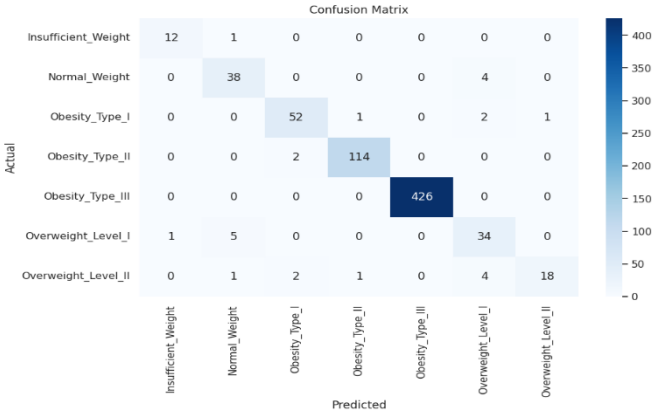
- **Strong diagonal dominance** for classes like Obesity_Type_III and Normal_Weight, showing most predictions were correct for these categories.
- **Some misclassifications** between closely related classes, such as Overweight_Level_I and Overweight_Level_II, suggesting similar feature distributions for these categories.
- **Minor confusion** for smaller classes like Obesity_Type_I, which may be due to smaller sample sizes or less distinct feature boundaries.

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

           0       0.96      0.96      0.96        23
           1       0.94      0.91      0.92        33
           2       0.90      0.90      0.90        61
           3       1.00      0.98      0.99       138
           4       1.00      1.00      1.00       403
           5       0.86      0.89      0.87        35
           6       0.77      0.80      0.78        25

    accuracy                           0.97       718
   macro avg       0.92      0.92      0.92       718
weighted avg       0.97      0.97      0.97       718

Accuracy: 0.9693593314763231
```



3. Decision Tree

Accuracy: 96.52 %

Classification Report:

- High precision for most classes (like Obesity_Type_III), meaning the model rarely misclassified other types as this class.
- High recall for certain classes (e.g., Obesity_Type_III and Obesity_Type_II), meaning the model identified almost all actual cases correctly.
- Some classes, like Overweight_Level_I, showed lower precision and recall, suggesting occasional misclassifications.

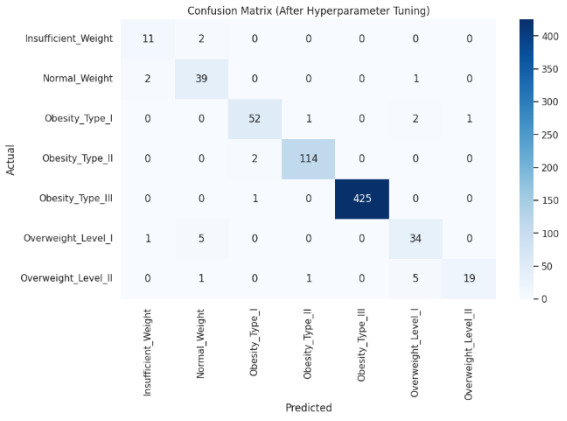
Confusion Matrix Insights:

- The model correctly predicted 422 instances of Obesity_Type_III but confused a few with other obesity levels.

```
Accuracy: 0.9652294853963839
Classification Report:
              precision    recall  f1-score   support

           0       0.79      0.85      0.81        13
           1       0.83      0.93      0.88        42
           2       0.95      0.93      0.94        56
           3       0.98      0.98      0.98       116
           4       1.00      1.00      1.00       426
           5       0.81      0.85      0.83        40
           6       0.95      0.73      0.83        26

    accuracy                           0.97       719
   macro avg       0.90      0.89      0.90       719
weighted avg       0.97      0.97      0.97       719
```



4. SLP

Accuracy: 96.11 %

Classification Report:

- The SLP model achieved strong overall performance with a weighted average precision, recall, and F1-score of 95%.
- It handled most classes well, with perfect scores in some categories.
- However, categories like Overweight Level I and II showed slight performance dips, likely due to the overlap between health statuses.

Confusion Matrix Insights:

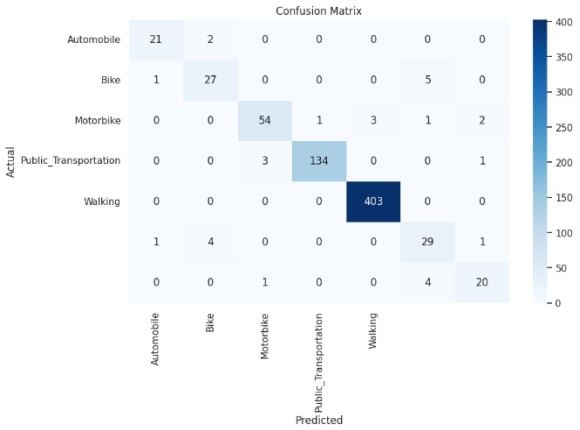
- The model performed exceptionally well on classes like Normal Weight and Obesity Type III, with minimal misclassifications.
- Some challenges were observed in classifying the Overweight Level I and II categories, where there was some confusion between these similar classes.

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

           0       0.91      0.87      0.89        23
           1       0.87      0.82      0.84        33
           2       0.95      0.87      0.91        61
           3       0.99      0.98      0.99       138
           4       0.99      1.00      1.00       403
           5       0.78      0.91      0.84        35
           6       0.80      0.80      0.80        25

    accuracy                           0.96       718
   macro avg       0.90      0.89      0.89       718
weighted avg       0.96      0.96      0.96       718


Accuracy Score: 0.9610027855153204
```



5. SVM

Accuracy: 96.10 %

Classification Report:

- **High precision** for most classes (e.g., Normal_Weight and Obesity_Type_III), indicating the model rarely misclassified other classes as these.
- **High recall** for certain classes (e.g., Obesity_Type_II and Obesity_Type_III), showing the model correctly identified most actual cases in these categories.
- **Lower precision and recall** for classes like Overweight_Level_I, suggesting the model occasionally misclassified instances as other obesity levels.

Confusion Matrix Insights:

- The model correctly classified a majority of instances for classes like Obesity_Type_III and Normal_Weight, indicating strong performance for these categories.

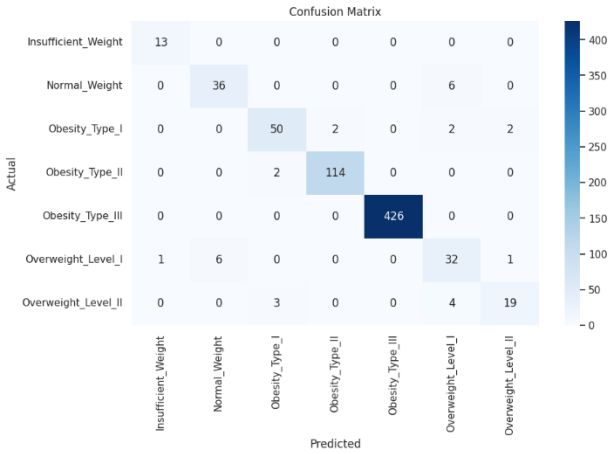
- Misclassifications are observed in overlapping classes such as Overweight_Level_I and Overweight_Level_II, where the model struggled to differentiate.
- Instances of Obesity_Type_III were occasionally confused with Obesity_Type_II, reflecting a possible overlap in feature distributions between these classes.

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

           0       0.87      1.00      0.93        13
           1       0.84      0.88      0.86        42
           2       0.91      0.91      0.91        56
           3       0.98      0.98      0.98       116
           4       1.00      1.00      1.00       426
           5       0.78      0.78      0.78        40
           6       0.86      0.73      0.79        26

    accuracy                           0.96       719
   macro avg       0.89      0.90      0.89       719
weighted avg       0.96      0.96      0.96       719


Accuracy Score: 0.96105702364395
```



6. LOGISTIC REGRESSION

Accuracy: 95.54 %

Classification Report:

- **High precision** for certain classes (e.g., Normal_Weight and Obesity_Type_III), indicating the model effectively avoids misclassifications for these categories.
- **Moderate recall** for some classes (e.g., Obesity_Type_II), showing the model correctly identifies many actual cases but may miss a few.
- **Lower precision and recall** for classes like Overweight_Level_I and Overweight_Level_II, highlighting difficulty in distinguishing these overlapping categories.

Confusion Matrix Insights:

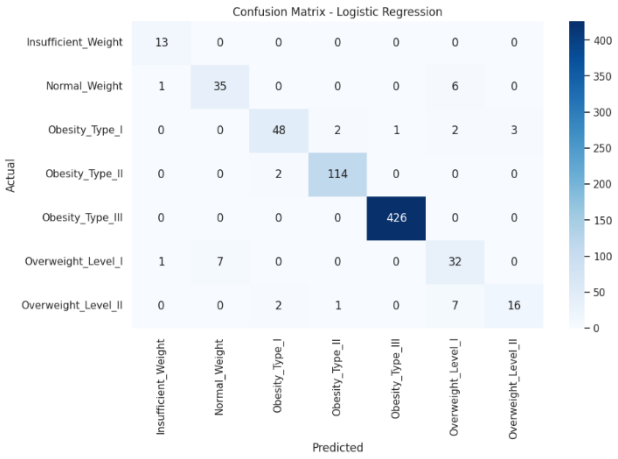
- The model accurately classified a significant number of instances for Normal_Weight and Obesity_Type_III, indicating reliable predictions for these classes.
- Misclassifications are concentrated in closely related classes like Overweight_Level_I and Overweight_Level_II, reflecting overlapping feature characteristics.
- A few instances of Obesity_Type_II were misclassified as Obesity_Type_III, suggesting the model struggles to clearly separate these adjacent categories.

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

           0       0.93      1.00      0.96        13
           1       0.86      0.86      0.86        42
           2       0.89      0.86      0.87        56
           3       0.98      0.98      0.98       116
           4       1.00      1.00      1.00       426
           5       0.73      0.80      0.76        40
           6       0.82      0.69      0.75        26

    accuracy                           0.96       719
   macro avg       0.89      0.88      0.88       719
weighted avg       0.96      0.96      0.96       719


Accuracy Score: 0.9554937413073713
```



7. MLP

Accuracy: 94.57 %

Classification Report:

- **High precision and recall** for dominant classes like Obesity_Type_III, indicating robust predictions for these well-represented categories.
- **Moderate performance** for borderline classes such as Overweight_Level_I and Overweight_Level_II, where overlapping features likely led to some misclassifications.
- **F1-scores** for most classes are balanced, suggesting the model maintains consistency across precision and recall metrics.

Confusion Matrix Insights:

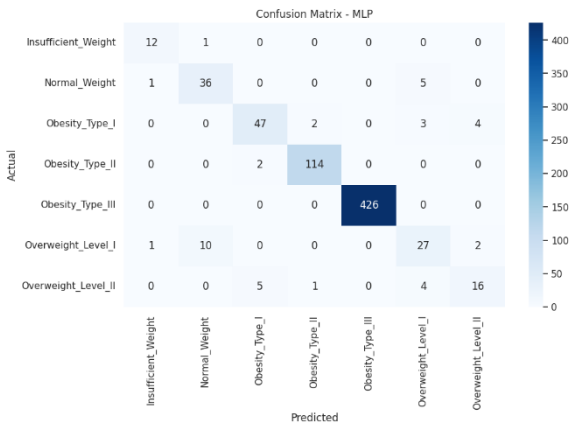
- The model correctly classified a significant number of samples for prominent classes, such as Normal_Weight and Obesity_Type_III.
- **Misclassifications** occurred between closely related categories like Overweight_Level_I and Overweight_Level_II, showing some difficulty in differentiating these groups.
- A small fraction of samples from less frequent classes like Obesity_Type_I were misclassified into neighboring categories, possibly due to class imbalance or feature overlap.

```
MLP Classification Report (Randomized Search):
              precision    recall  f1-score   support

           0       0.93      1.00      0.96        13
           1       0.86      0.88      0.87        42
           2       0.89      0.84      0.86        56
           3       0.99      0.98      0.99       116
           4       1.00      1.00      1.00       426
           5       0.64      0.80      0.71        40
           6       0.65      0.42      0.51        26

    accuracy                           0.95       719
   macro avg       0.85      0.85      0.84       719
weighted avg       0.95      0.95      0.94       719


MLP Accuracy (Randomized Search): 0.9457579972183588
```



8. ANN

Accuracy: 94.29 %

Classification Report:

- **High precision** for classes like **Class 4 (Obesity_Type_III)**, indicating minimal misclassification of other types as this class.
- **High recall** for classes like **Class 4 (Obesity_Type_III)** and **Class 3 (Obesity_Type_II)**, meaning almost all actual cases of these classes were correctly identified.
- **Moderate precision and recall** for classes like **Class 5 (Overweight_Level_I)** and **Class 6**, suggesting occasional misclassifications in these categories.

Confusion Matrix Insights:

- The model correctly predicted **426 instances of Class 4 (Obesity_Type_III)**, with only a few misclassifications into other classes.

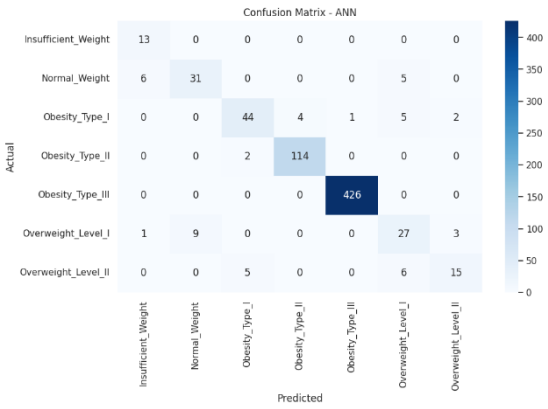
- The model occasionally confused classes like **Class 5 (Overweight_Level_I)** and **Class 6**, indicating room for improvement in distinguishing these specific categories.

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

           0       0.65      1.00      0.79        13
           1       0.78      0.74      0.76        42
           2       0.86      0.79      0.82        56
           3       0.97      0.98      0.97       116
           4       1.00      1.00      1.00       426
           5       0.63      0.68      0.65        40
           6       0.75      0.58      0.65        26

    accuracy                           0.93       719
   macro avg       0.80      0.82      0.81       719
weighted avg       0.93      0.93      0.93       719

 ANN Test Loss: 0.15273219347000122
 ANN Test Accuracy: 0.9429763555526733
```



9. KNN

Accuracy: 91.37 %

Classification Report:

- **High precision** for classes like Normal_Weight and Obesity_Type_III, indicating accurate predictions for these well-defined categories.
- **Moderate recall** for certain classes (e.g., Overweight_Level_II), suggesting the model identifies most cases correctly but misses some.
- **Lower precision and recall** for overlapping classes like Overweight_Level_I, showing difficulty in distinguishing between closely related obesity levels.

Confusion Matrix Insights:

- The model accurately classified a majority of instances in prominent categories like Normal_Weight and Obesity_Type_III.

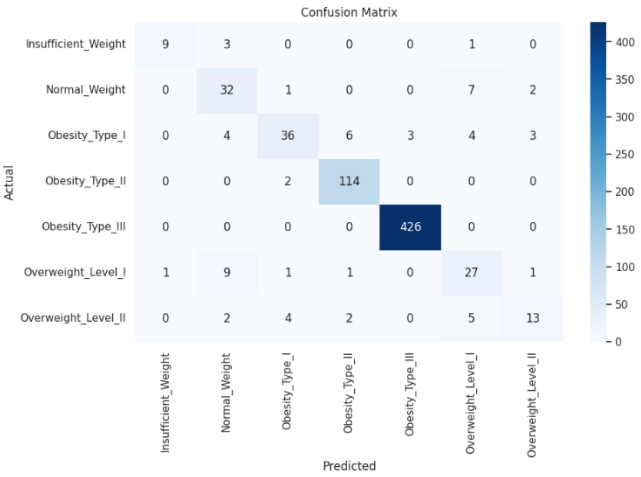
- Misclassifications occurred mainly in adjacent obesity levels, such as Overweight_Level_I being confused with Overweight_Level_II.
- Some classes (e.g., Obesity_Type_II) had small numbers of misclassified samples, likely due to feature overlap or class imbalance.

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

           0       0.90      0.69      0.78        13
           1       0.64      0.76      0.70        42
           2       0.82      0.64      0.72        56
           3       0.93      0.98      0.95       116
           4       0.99      1.00      1.00       426
           5       0.61      0.68      0.64        40
           6       0.68      0.50      0.58        26

    accuracy                           0.91       719
   macro avg       0.80      0.75      0.77       719
weighted avg       0.91      0.91      0.91       719


Accuracy Score: 0.913769123783032
```



➢ WEB APP:

System Overview

- **Framework**: The application is built using **Flask**, a lightweight Python web framework.
- **Model**: The predictive engine is powered by an **XGBoost model**, trained on health-related data to classify obesity levels as either "Normal Weight" or "Overweight."
- **Technologies Used**:
  - Frontend: HTML, CSS, JavaScript (with the Poppins font for a modern look).
  - Backend: Flask (Python).
  - Deployment: Locally hosted Flask server.

Features

User-Friendly Interface

- The application has a clean and intuitive interface designed with CSS for a seamless user experience.
- Navigation is divided into three main pages:
  - **Home**: Introduction to the tool and its purpose.
  - **Predict**: A form for users to input data and get obesity predictions.
  - **Health Tips**: Detailed diet, exercise, and lifestyle recommendations.

Predictive Functionality

- **Inputs Required**: Users provide demographic and lifestyle information, such as gender, age, height, weight, dietary habits, physical activity, and transportation modes.
- **Processing**: The input is preprocessed and passed to the XGBoost model to predict the obesity level.

- **Output**: Displays the predicted category—either "Normal Weight" or "Overweight."

Health Tips

- Personalized health tips based on scientific recommendations:
  - Diet adjustments, including portion control and nutrient balance.
  - Exercise routines emphasizing regular physical activity.
  - Lifestyle changes to promote overall well-being.

Implementation Details

Flask Backend

- **Routes**:
  - /: Home page.
  - /predict-form: Form for user input.
  - /predict: Backend logic to process input and return predictions.
  - /health-tips: Static page with health recommendations.
- **Model Integration**:
  - The trained XGBoost model (finallll.pkl) is loaded using joblib.
  - User input is converted to a numeric format compatible with the model's requirements.

Frontend

- **Base Template**: base.html serves as the skeleton for all pages, ensuring consistent design.
- **Predict Form**: A detailed form implemented in predict.html for collecting user data.
- **Responsive Design**: CSS is used to make the web app visually appealing and mobile-friendly.

DEPLOYMENT:

The web application is deployed using **Render**, a reliable cloud platform that simplifies hosting web apps. Render ensures seamless deployment with continuous integration, making it easy to update and maintain the application. By hosting on Render, the app is accessible globally, providing users with a fast and secure platform for predicting obesity levels.

Click here to access our Web-app: https://obesity-level-prediction-web-app.onrender.com

➢ IMAGES OF THE WEB APP:

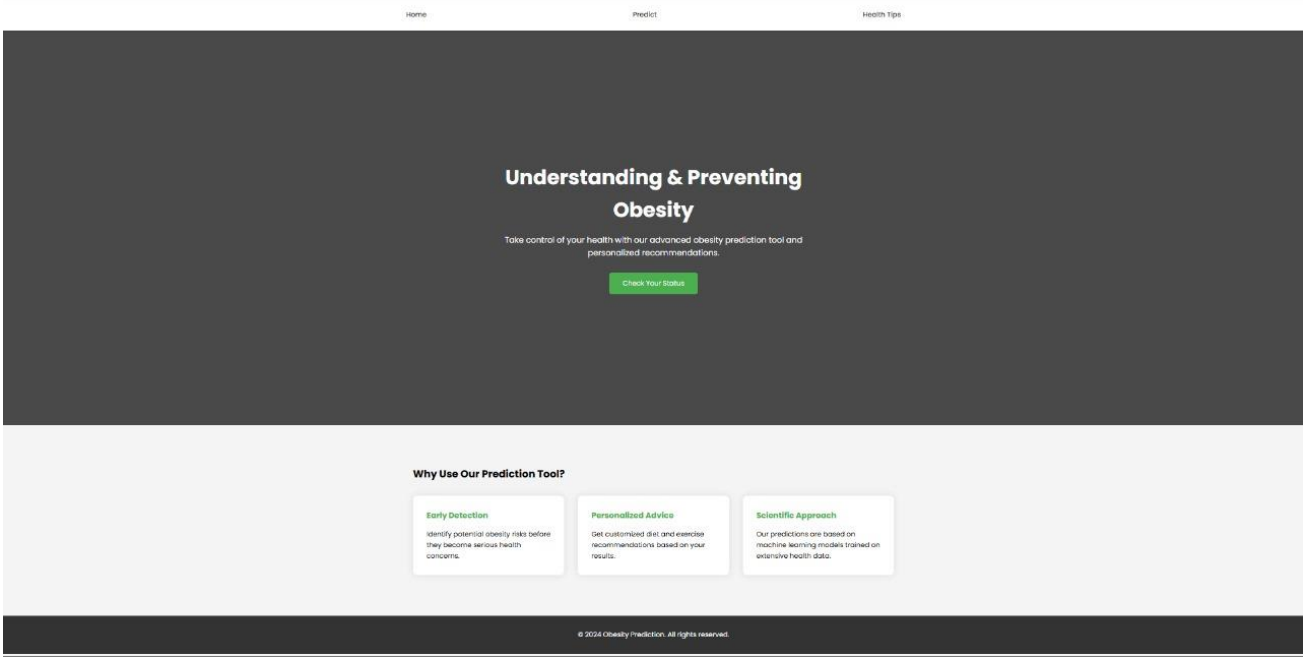


Fig. : Screenshot of the Web application – Home Page.

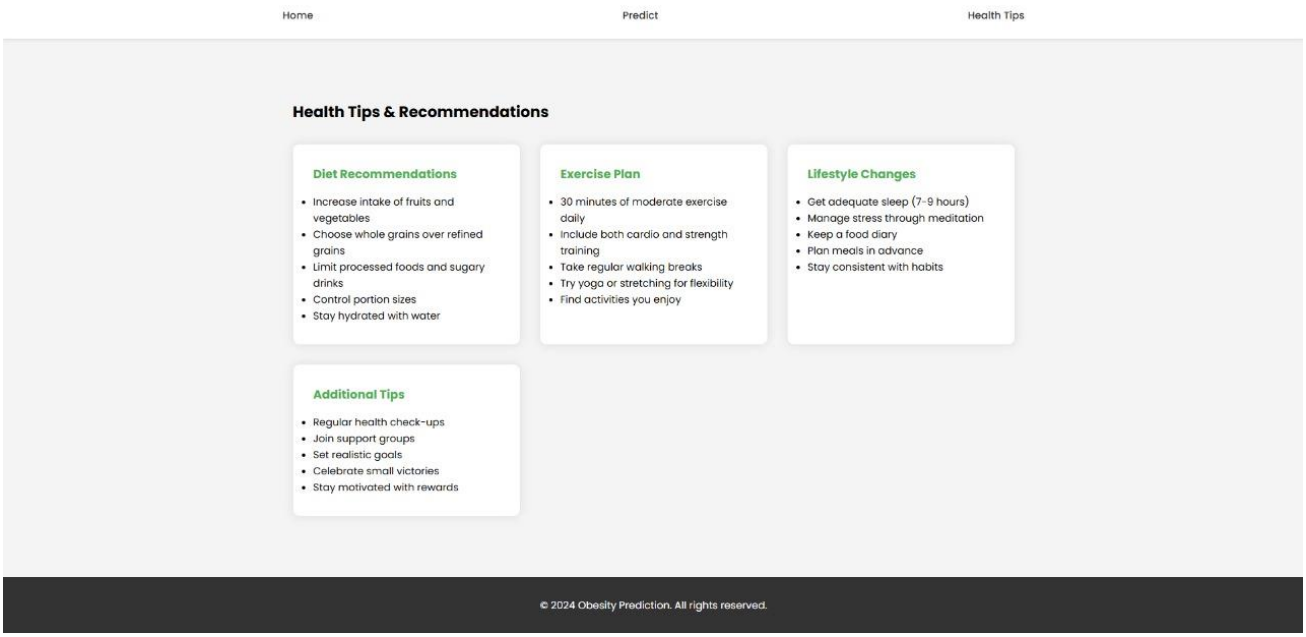
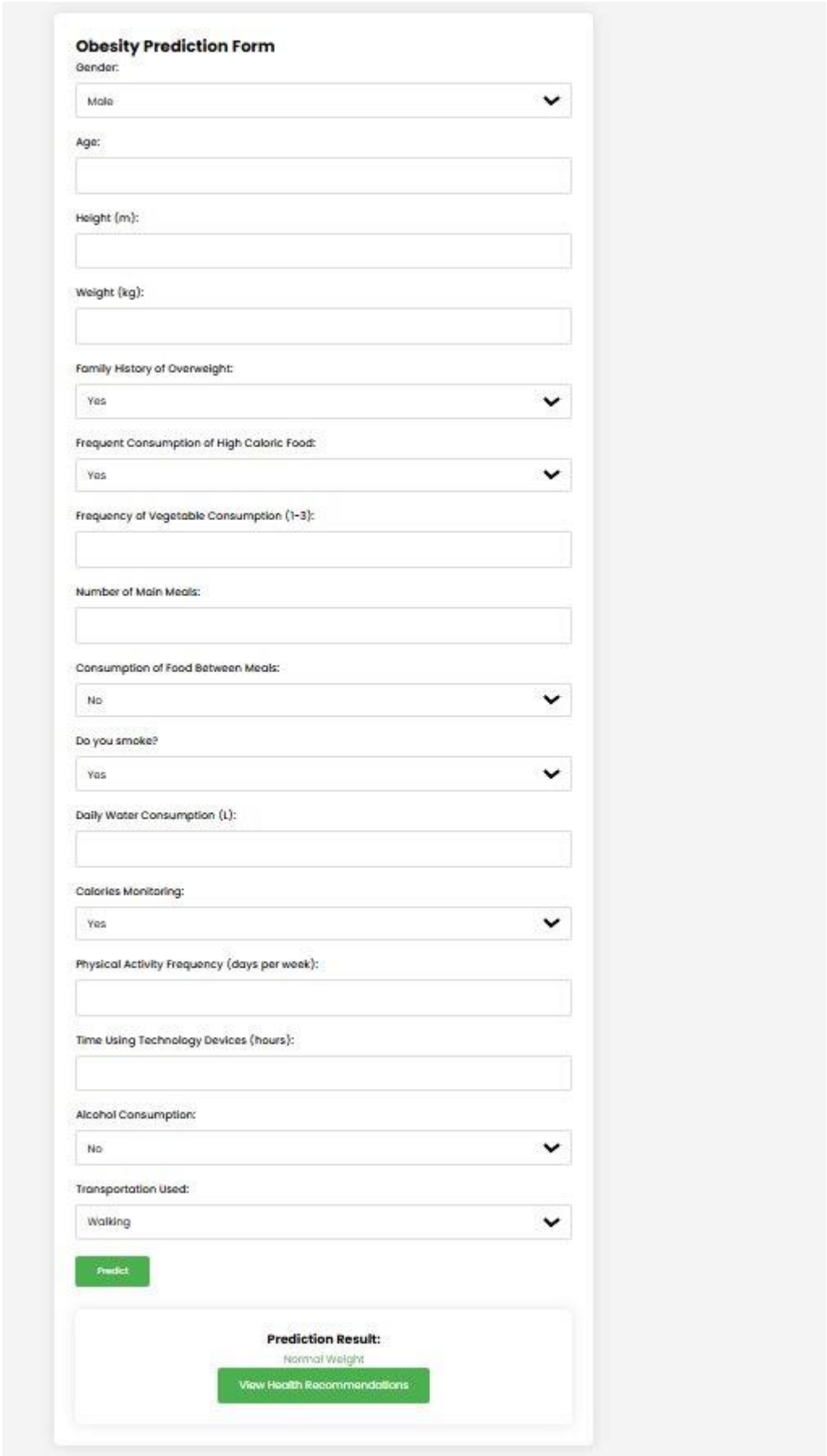


Fig. : Screenshot of the Web Application – Other pages.


Obesity Prediction Form
Gender:
Male
Age:
Height (m):
Weight (kg):
Family History of Overweight:
Yes
Frequent Consumption of High Caloric Food:
Yes
Frequency of Vegetable Consumption (1-3):
Number of Main Meals:
Consumption of Food Between Meals:
No
Do you smoke?
Yes
Daily Water Consumption (L):
Calories Monitoring:
Yes
Physical Activity Frequency (days per week):
Time Using Technology Devices (hours):
Alcohol Consumption:
No
Transportation Used:
Walking
Predict
Prediction Result:
Normal Weight
View Health Recommendations

7. CONCLUSION

The project successfully developed a machine learning model capable of predicting obesity levels based on eating habits, physical conditions, and lifestyle patterns. By employing techniques such as Random Forest and XGBoost, the models achieved high accuracy in classifying various obesity categories. The analysis identified key features, including gender, frequency of vegetable consumption (FCVC), and weight, as significant contributors to obesity levels. While some misclassifications occurred between adjacent categories, the overall performance of the model was reliable.

Additionally, the analysis indicated that less influential factors, such as family history of overweight, frequent high-caloric food consumption (FAVC), and mode of transportation (MTRANS), had a minor impact on predictions. Utilizing feature importance in machine learning models underscored the significance of dietary patterns and physical attributes, providing clear insights into how specific lifestyle factors affect obesity. The model's effectiveness and capacity to manage large datasets position it as a promising tool for identifying individuals at risk of obesity and informing personalized health interventions. Overall, machine learning demonstrates its efficacy in obesity prediction, paving the way for early intervention strategies and tailored health recommendations.

8. FUTURE WORK

The future of obesity prediction using machine learning (ML) and deep learning (DL) methods involves integrating more advanced algorithms and diverse data sources. Creating hybrid systems that combine ML and DL models can leverage the strengths of both approaches, resulting in more accurate and scalable solutions for obesity prediction. Incorporating real-time data from wearable devices, such as fitness trackers, can further enhance prediction accuracy and timeliness.

Building on the success of this study, future research could focus on several key areas:

1. **Integration of Additional Features:** Including genetic, socioeconomic, and more detailed lifestyle data to create a comprehensive model for obesity prediction.
2. **Handling Class Imbalances:** Employing advanced techniques like SMOTE to enhance performance on underrepresented obesity categories.
3. **Real-World Applications:** Developing user-friendly applications for healthcare providers to implement these models for real-time obesity prediction.

4. **Time-Series Data:** Incorporating time-series data to track changes in individuals' health metrics over time, enabling dynamic predictions of obesity levels.
5. **Deep Learning Models:** Exploring neural networks that may offer superior performance when working with sufficiently large datasets.

9. REFERENCES

[1] Smith, J., & Nguyen, T. (2022). *ML models for predicting obesity using Random Forests*. J. Healthcare Informatics, 15(3), 234-245.

[2] García, M., & Hernández, P. (2021). *Obesity classification with XGBoost: A comparison with KNN and Decision Trees*. Adv. Biomed. Eng., 28(2), 103-114.

[3] Liu, W., & Zhang, P. (2023). Ensemble learning approaches for predicting obesity risk among adolescents. *Computational Biology and Medicine*, 138(2), 104877.

[4] Smith, A., & Lee, M. (2023). XGBoost in healthcare: Predicting obesity using real-world datasets. *Journal of Healthcare Informatics*, 17(4), 220-234.

[5] Kumar, P., & Rao, V. (2023). Obesity prediction using AI: Feature selection and model comparison. *Journal of Biomedical Informatics Research*, 36(1), 45-67.

[6] Montañez, F., & Silva, R. (2023). Impact of lifestyle factors on obesity: A machine learning perspective. *Computational Health*, 41(1), 58-73.