Project: Obesity Prediction Analysis – ML

Author: Miro Zilaji, 9.5.2025

Leitfaden für nachvollziehbare Schritte

 # **Guidelines for Reproducible Steps**

**1. Brief Presentation of the Problem Area / Overview of the Topic**

**1.1 Content**

****Core of the Investigation:****

The project focuses on **predicting obesity levels based on various health and lifestyle factors using machine learning techniques. The dataset includes features such as age, weight, height, dietary habits, physical activity, and family history, which are used to classify individuals into different obesity categories.**

****Main Objectives of the Work:****

- Analyze the relationship between different features and obesity levels.

- Compare different machine learning models (AdaBoost, Random Forest, Logistic Regression, SVM, Decision Trees) for obesity classification.

- Evaluate feature importance to identify key predictors of obesity.

- Provide insights into how obesity risk factors differ between genders.

**1.2 Justification of the Topic**

****Why is the Topic Relevant and Worth Investigating?****

- Obesity is a **global health crisis, linked to chronic diseases such as diabetes, cardiovascular disorders, and hypertension.**

- Early prediction and classification can help in **preventive healthcare and personalized treatment plans.**

- Machine learning provides a **data-driven approach to understanding obesity patterns beyond traditional statistical methods.**

****Personal Motivation:****

- Interest in **health informatics** and the application of AI in medical research.

- Desire to explore **feature importance** in obesity prediction to identify modifiable risk factors.

---

2. Reproducible Steps

2.1 State of Research / Literature Review

Has this problem been studied before?

- Previous studies have used **BMI as a primary indicator** of obesity.

- Machine learning models such as **Random Forest and Logistic Regression** have been applied in obesity prediction.

- Some research has explored **gender differences** in obesity risk factors.

Which aspects were examined, and which were not?

- Prior work focused on **binary classification** (obese vs. non-obese).

- Few studies compared **multiple obesity classes** (e.g., underweight, normal, overweight, obesity types I-III).

- Limited research on **feature importance differences between genders**.

Controversies and Methods Used So Far:

- Some studies debate whether **BMI alone is sufficient** for obesity classification.

- **Feature selection methods** vary—some rely on domain knowledge, while others use automated techniques.

2.2 Research Question

- **How accurately can machine learning models classify obesity levels?**

- **Which features are most predictive of obesity?**

- **Do obesity risk factors differ between males and females?**

2.3 State of Research

- Existing studies show **Random Forest and AdaBoost perform well** in obesity classification.

- **Feature importance analysis** often highlights **BMI, diet, and physical activity** as key predictors.

2.4 Knowledge Gap

- Most studies do not **compare multiple weak learners in AdaBoost** for obesity prediction.

- Few papers analyze **gender-specific feature importance**.

2.5 Methodology

Detailed and Reproducible Description of the Approach:

###**Liabraries used

```
# Import necessary libraries

import pandas as pd  # For data manipulation and analysis

import numpy as np  # For numerical operations

import matplotlib.pyplot as plt  # For data visualization

import seaborn as sns  # For enhanced data visualization

from sklearn.model_selection import train_test_split  # For splitting data into train/test sets

from sklearn.preprocessing import StandardScaler, LabelEncoder  # For data preprocessing

from sklearn.ensemble import AdaBoostClassifier, RandomForestClassifier  # Ensemble learning methods

from sklearn.tree import DecisionTreeClassifier  # Decision tree classifier

from sklearn.linear_model import LogisticRegression  # Logistic regression classifier

from sklearn.svm import SVC  # Support Vector Machine classifier

from sklearn.metrics import accuracy_score, classification_report  # Model evaluation metrics

import warnings  # For handling warnings

import logging  # For logging information
```

Data Preprocessing:

- **Outlier Removal:** Applied IQR method to clean `Age` and `NCP` (Number of Main Meals).

```
# Handle outliers in specified features

features_to_clean = ['Age', 'NCP']  # Features to clean

for feature in features_to_clean:

    Q1 = df[feature].quantile(0.25)  # Calculate first quartile

    Q3 = df[feature].quantile(0.75)  # Calculate third quartile

    IQR = Q3 - Q1  # Calculate interquartile range
```

```
    lower_bound = Q1 - 1.5 * IQR  # Calculate lower bound
    upper_bound = Q3 + 1.5 * IQR  # Calculate upper bound
    df = df[(df[feature] >= lower_bound) & (df[feature] <= upper_bound)]  # Filter outliers
```

BMI Calculation: Computed as `Weight / (Height²)`.

```
# --- BMI Calculation ---
logging.info("\n--- BMI Calculation ---")  # Log BMI calculation section
df['BMI'] = df['Weight'] / (df['Height']**2)  # Calculate BMI (weight/height^2)
logging.info(df[['Height', 'Weight', 'BMI']].head().to_string())  # Log first few rows of height, weight and BMI

# --- Add BMI to the dataset ---
logging.info("\n--- Dataset with BMI ---")  # Log dataset with BMI section
logging.info(df.head().to_string())  # Log first few rows of the dataset
```

- **Obesity Classification:** Categorized BMI into WHO-defined classes (Underweight, Normal, Overweight, Obesity I-III).

```
# Define function to categorize BMI according to WHO standards
def categorize_bmi(bmi):
    if bmi < 18.5:
        return 'Underweight'
    elif 18.5 <= bmi < 25:
        return 'Normal weight'
    elif 25 <= bmi < 30:
        return 'Overweight'
    elif 30 <= bmi < 35:
        return 'Obesity I'
    elif 35 <= bmi < 40:
        return 'Obesity II'
    else:
```

```
    return 'Obesity III'
```

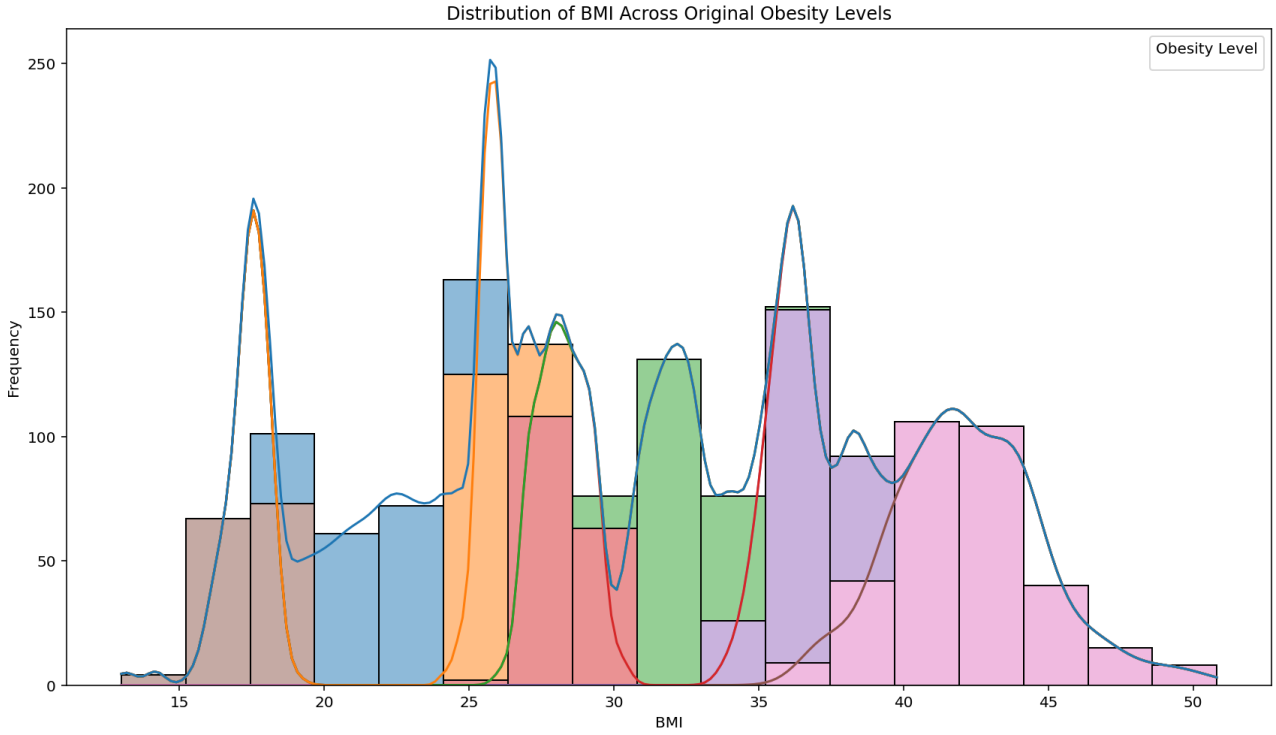
Exploratory Data Analysis (EDA):

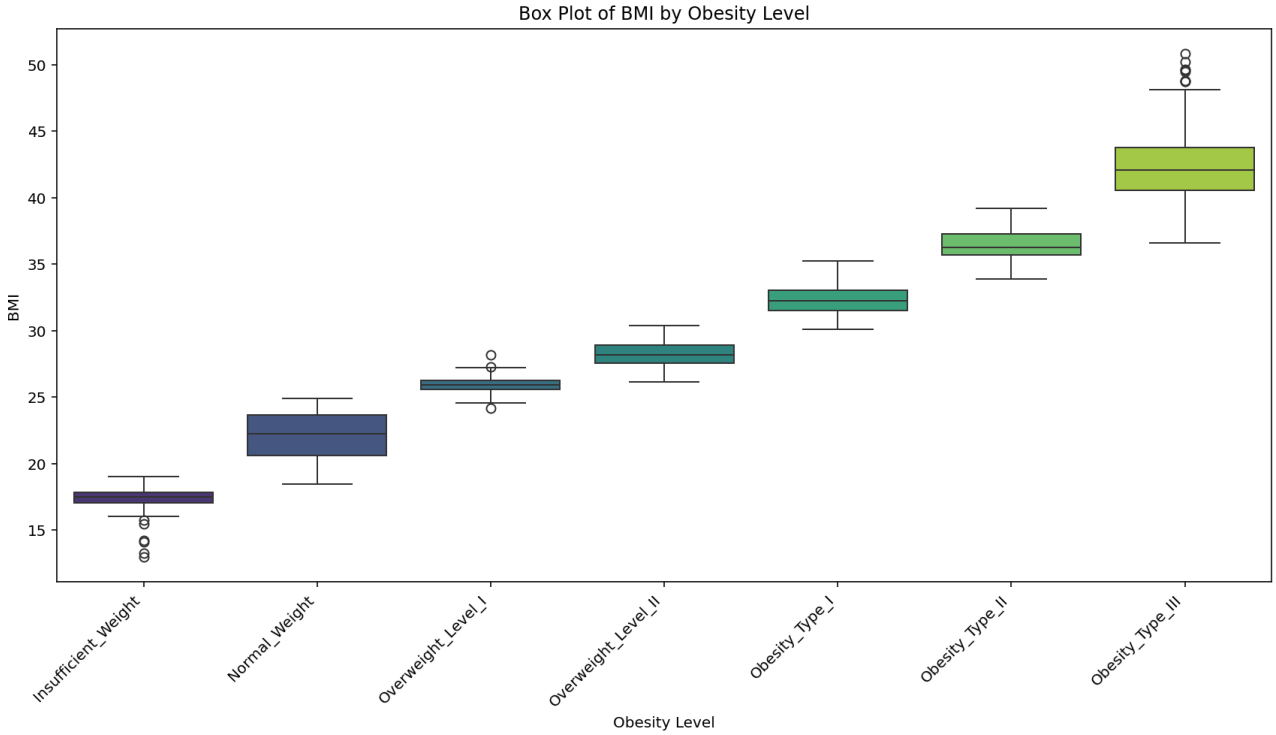
- **Visualizations:**

  - Histograms and boxplots of BMI distribution across obesity levels.

```
# --- Plot BMI distribution across original obesity classes ---
logging.info("\n--- Plotting BMI Distribution across Original Obesity Classes ---")  # Log plot section
plt.figure(figsize=(12, 7))  # Set figure size
sns.histplot(data=df, x='BMI', hue='NObeyesdad', kde=True, multiple="stack")  # Create stacked histogram with KDE
plt.title('Distribution of BMI Across Original Obesity Levels')  # Set title
plt.xlabel('BMI')  # Set x-axis label
plt.ylabel('Frequency')  # Set y-axis label
plt.legend(title='Obesity Level')  # Add legend with title
plt.tight_layout()  # Adjust layout
plt.show()  # Display plot

# --- Plot boxplot of BMI by original obesity level ---
logging.info("\n--- Plotting Boxplot of BMI by Original Obesity Level (Reordered) ---")  # Log boxplot section
original_order = ['Insufficient_Weight', 'Normal_Weight', 'Overweight_Level_I', 'Overweight_Level_II',
                  'Obesity_Type_I', 'Obesity_Type_II', 'Obesity_Type_III']  # Define order of categories
plt.figure(figsize=(12, 7))  # Set figure size
sns.boxplot(x='NObeyesdad', y='BMI', data=df, order=original_order, palette='viridis')  # Create boxplot with viridis palette
plt.title('Box Plot of BMI by Obesity Level')  # Set title
plt.xlabel('Obesity Level')  # Set x-axis label
plt.ylabel('BMI')  # Set y-axis label
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels
plt.tight_layout()  # Adjust layout
plt.show()  # Display plot
```





- Bar plots for categorical features (e.g., gender, family history).

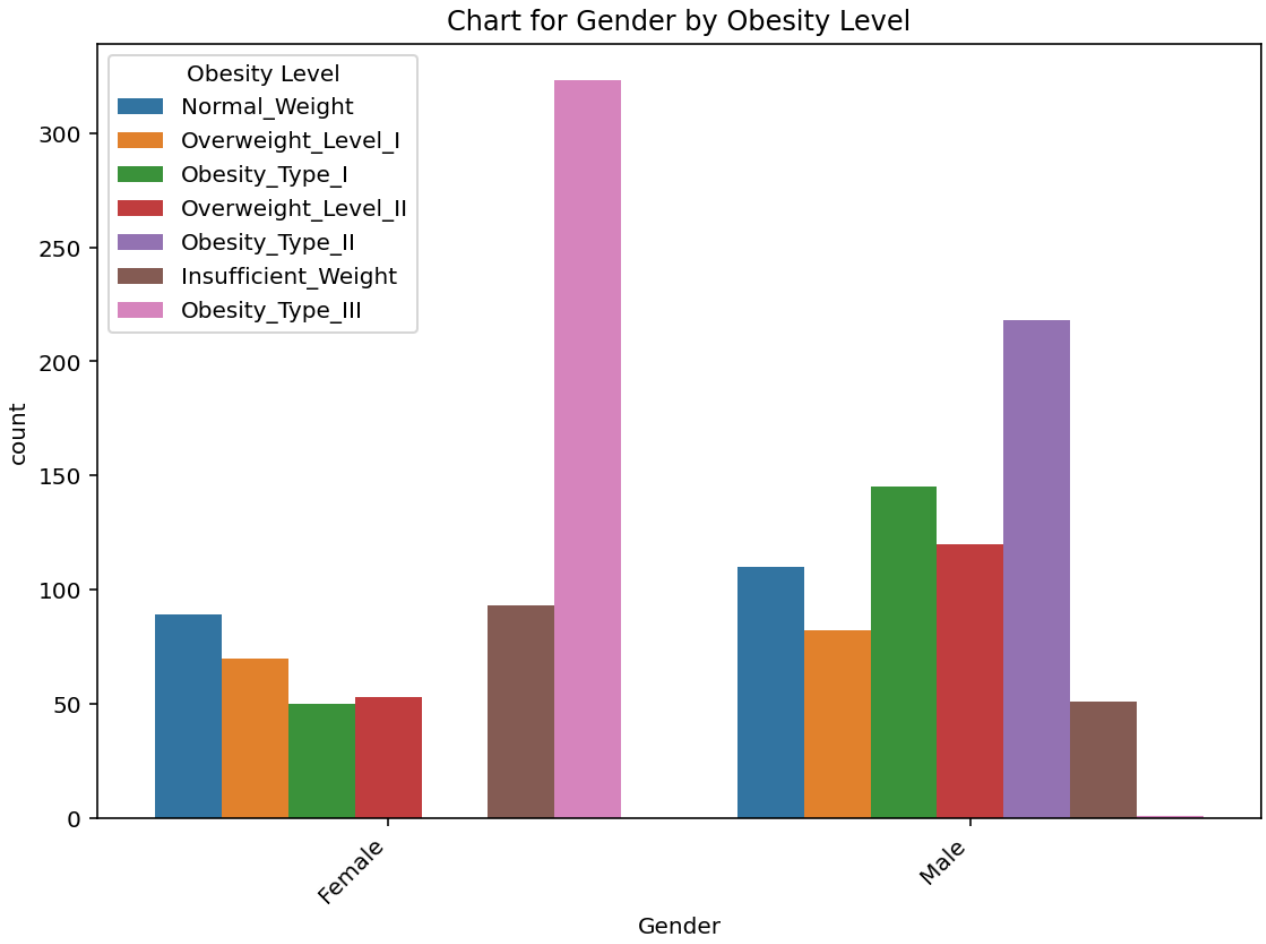


Chart for Gender by Obesity Level
Obesity Level
Normal_Weight
Overweight_Level_I
Obesity_Type_I
Overweight_Level_II
Obesity_Type_II
Insufficient_Weight
Obesity_Type_III
count
0
50
100
150
200
250
300
Female
Male
Gender

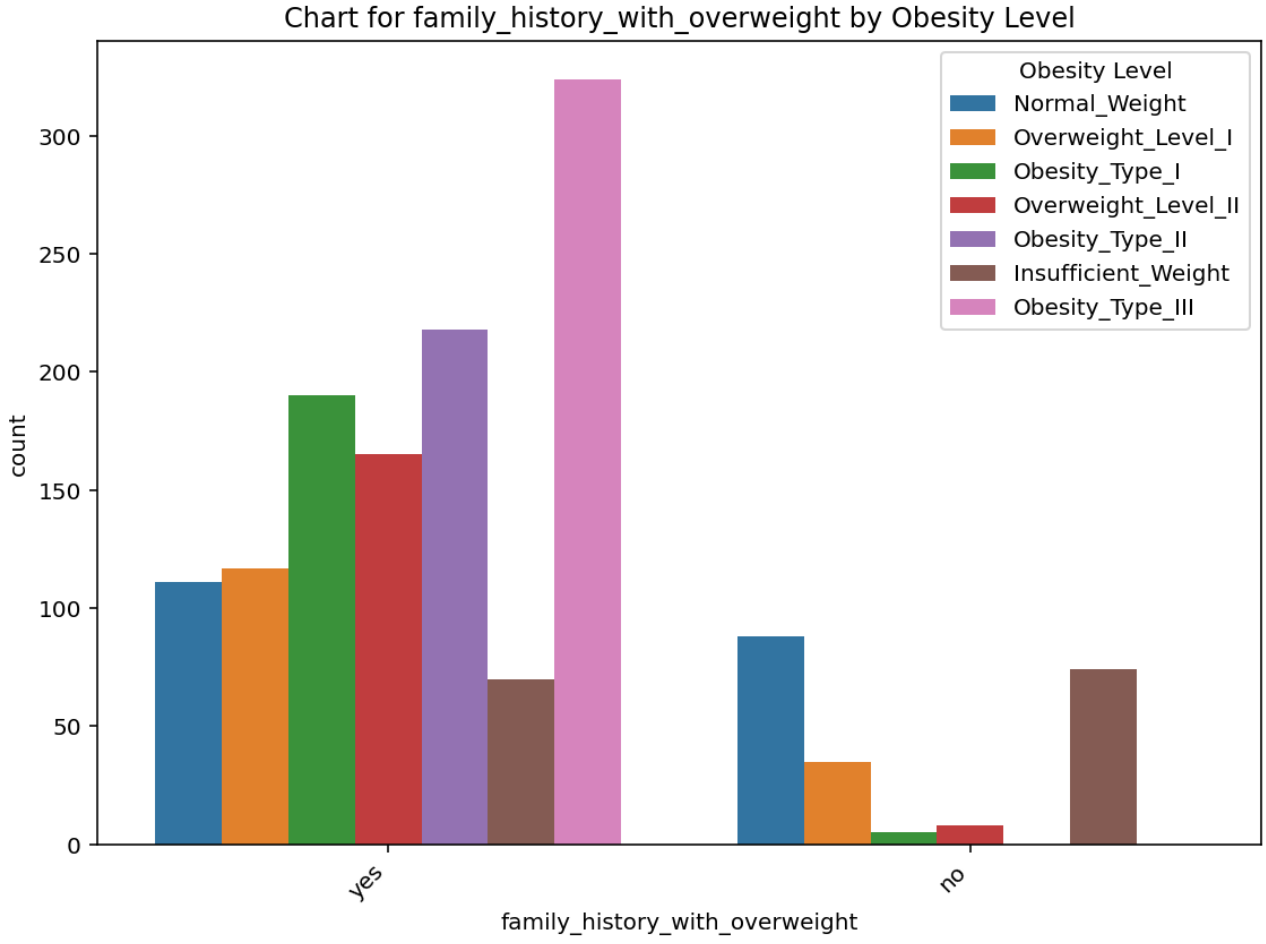
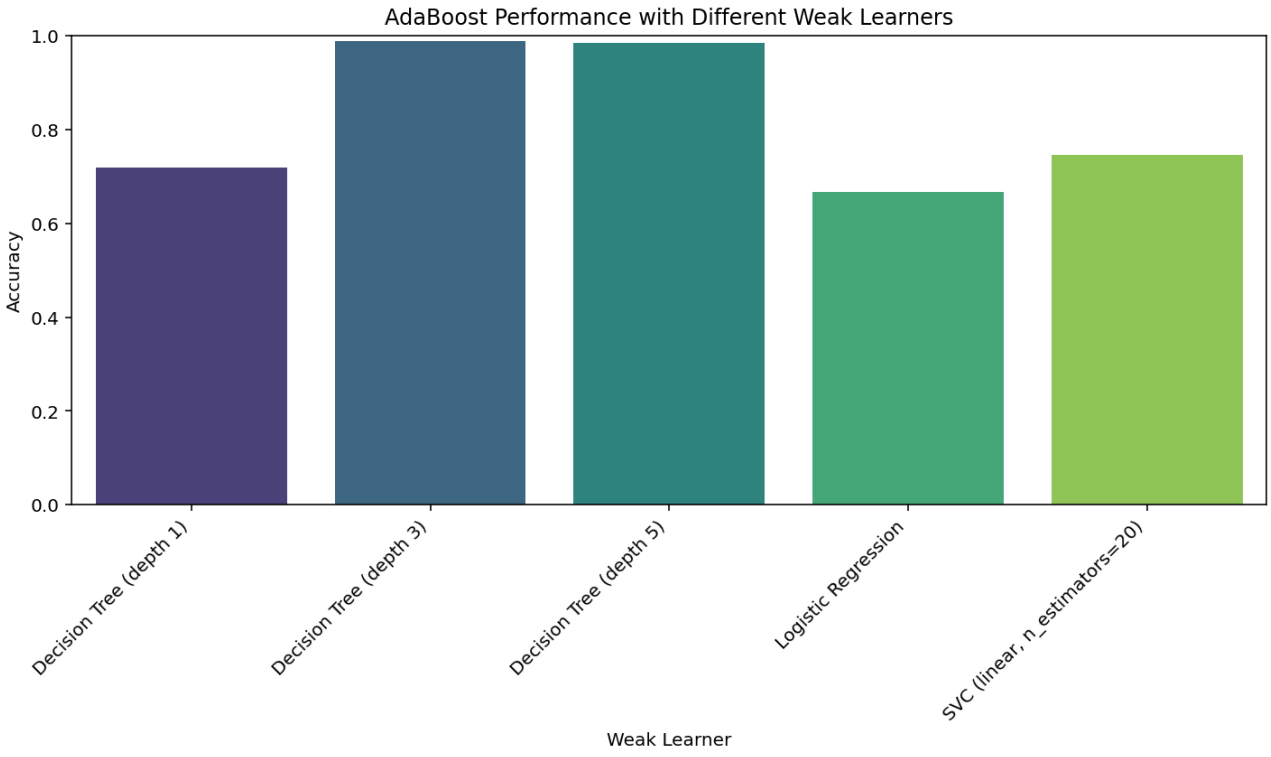


Chart for family_history_with_overweight by Obesity Level
Obesity Level
Normal_Weight
Overweight_Level_I
Obesity_Type_I
Overweight_Level_II
Obesity_Type_II
Insufficient_Weight
Obesity_Type_III
count
0
50
100
150
200
250
300
yes
no
family_history_with_overweight

Model Training & Evaluation:

- **Algorithms Tested:**

  - **AdaBoost** with different weak learners (Decision Trees, Logistic Regression, SVM).



  - **Random Forest** for feature importance analysis.

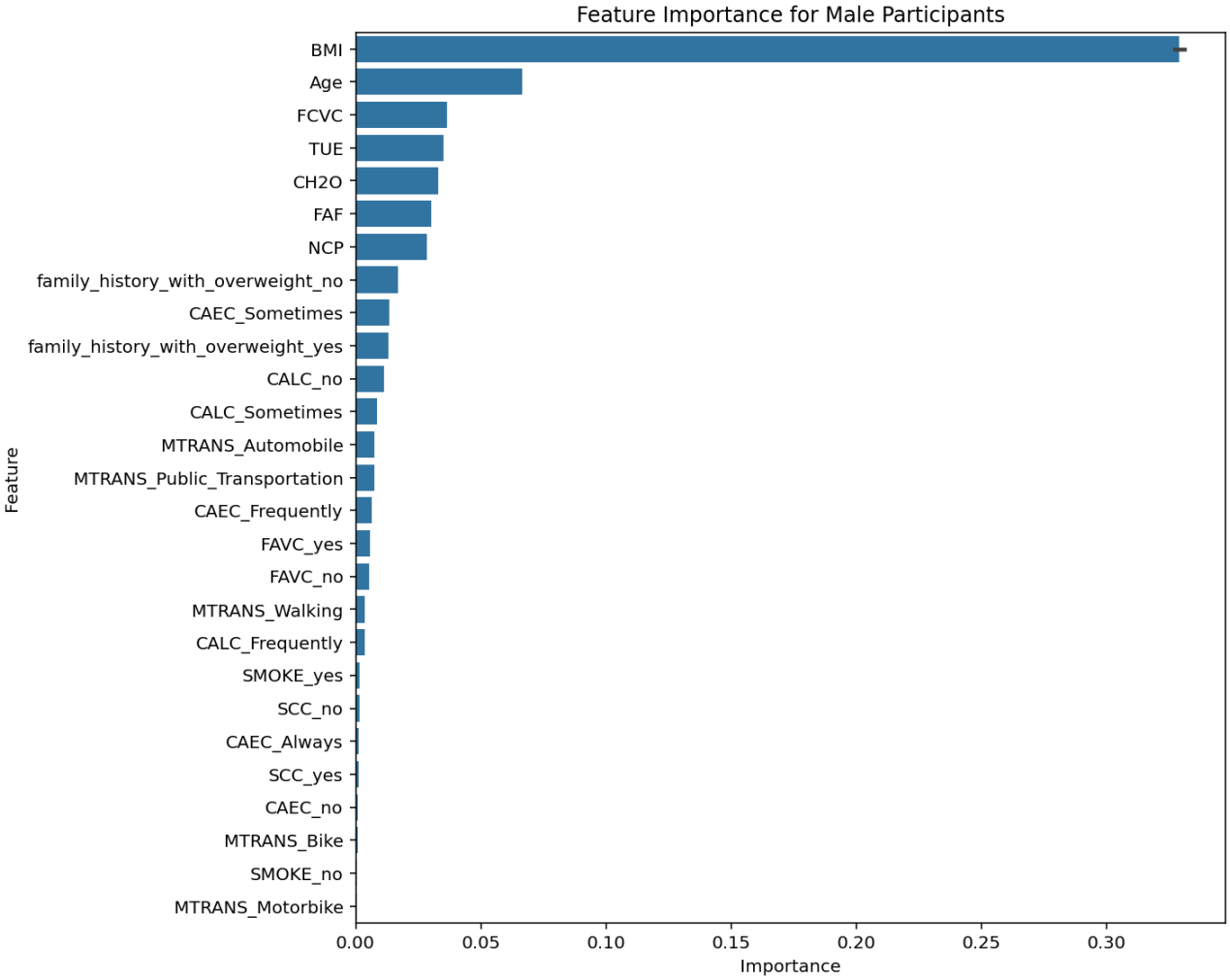
- **Evaluation Metrics:**

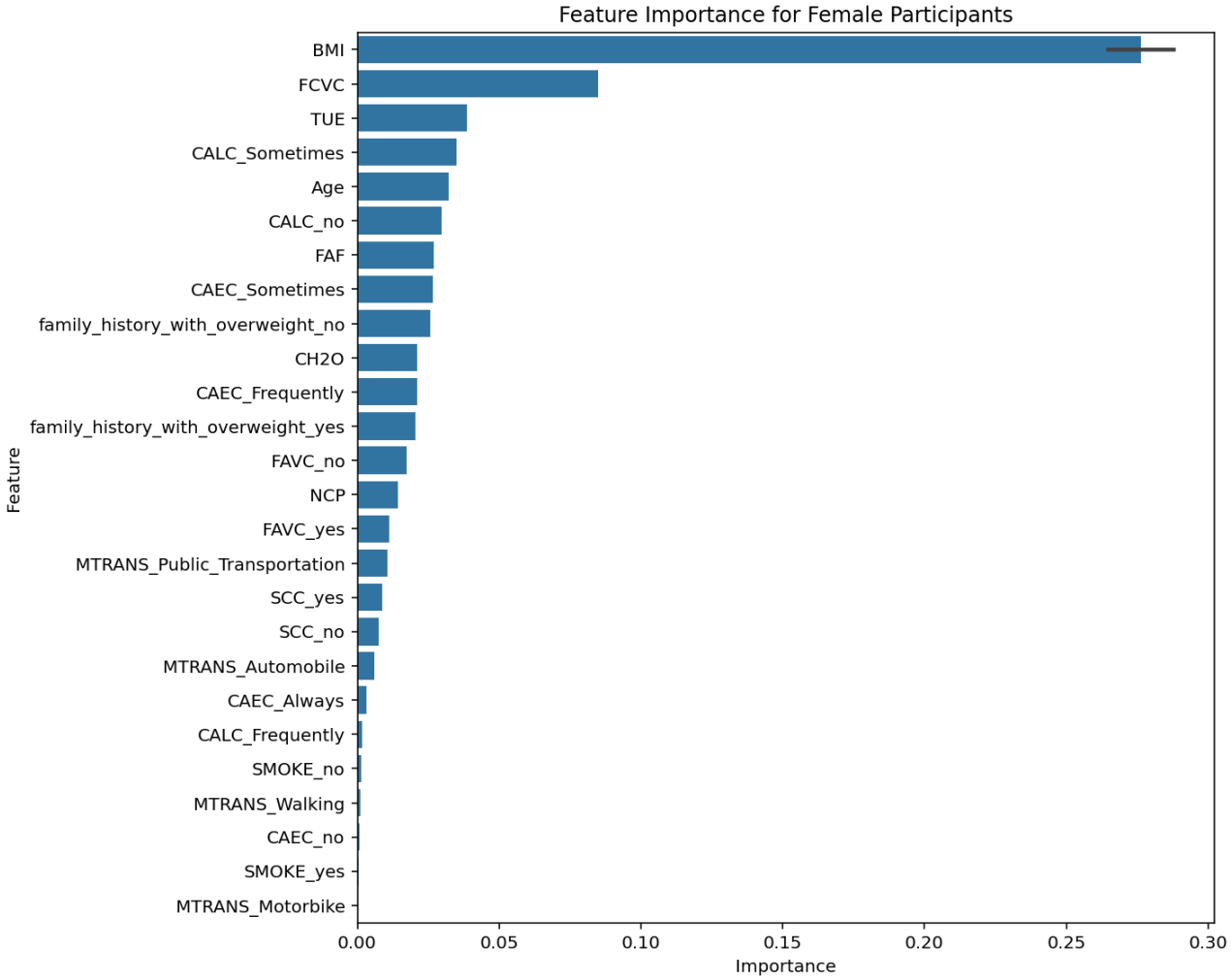
  - Accuracy, precision, recall, F1-score (via `classification_report`).

Model	accuracy
Decision Tree (depth 1)	0.72
Decision Tree (depth 3)	0.99
Decision Tree (depth 5)	0.98
Logistic Regression	0.67
SVC (linear, n_estimators=20)	0.75

- **Gender-Specific Analysis:**

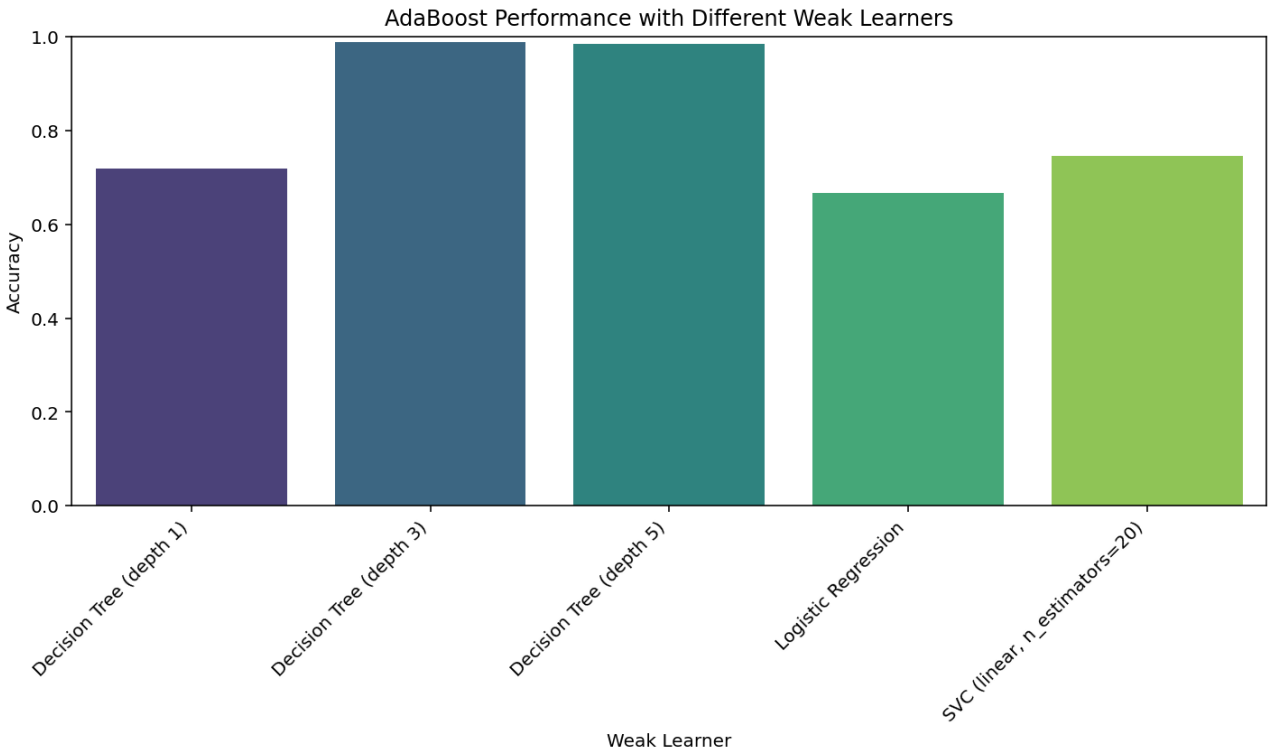
  - Separate feature importance for males and females.


Feature Importance for Male Participants
BMI
Age
FCVC
TUE
CH2O
FAF
NCP
family_history_with_overweight_no
CAEC_Sometimes
family_history_with_overweight_yes
CALC_no
CALC_Sometimes
MTRANS_Automobile
MTRANS_Public_Transportation
CAEC_Frequently
FAVC_yes
FAVC_no
MTRANS_Walking
CALC_Frequently
SMOKE_yes
SCC_no
CAEC_Always
SCC_yes
CAEC_no
MTRANS_Bike
SMOKE_no
MTRANS_Motorbike
Feature
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Importance



2.6 Results

- **Best Model:** AdaBoost with **Decision Tree (depth=3 and depth=5)** achieved the highest accuracy close to 100 %).

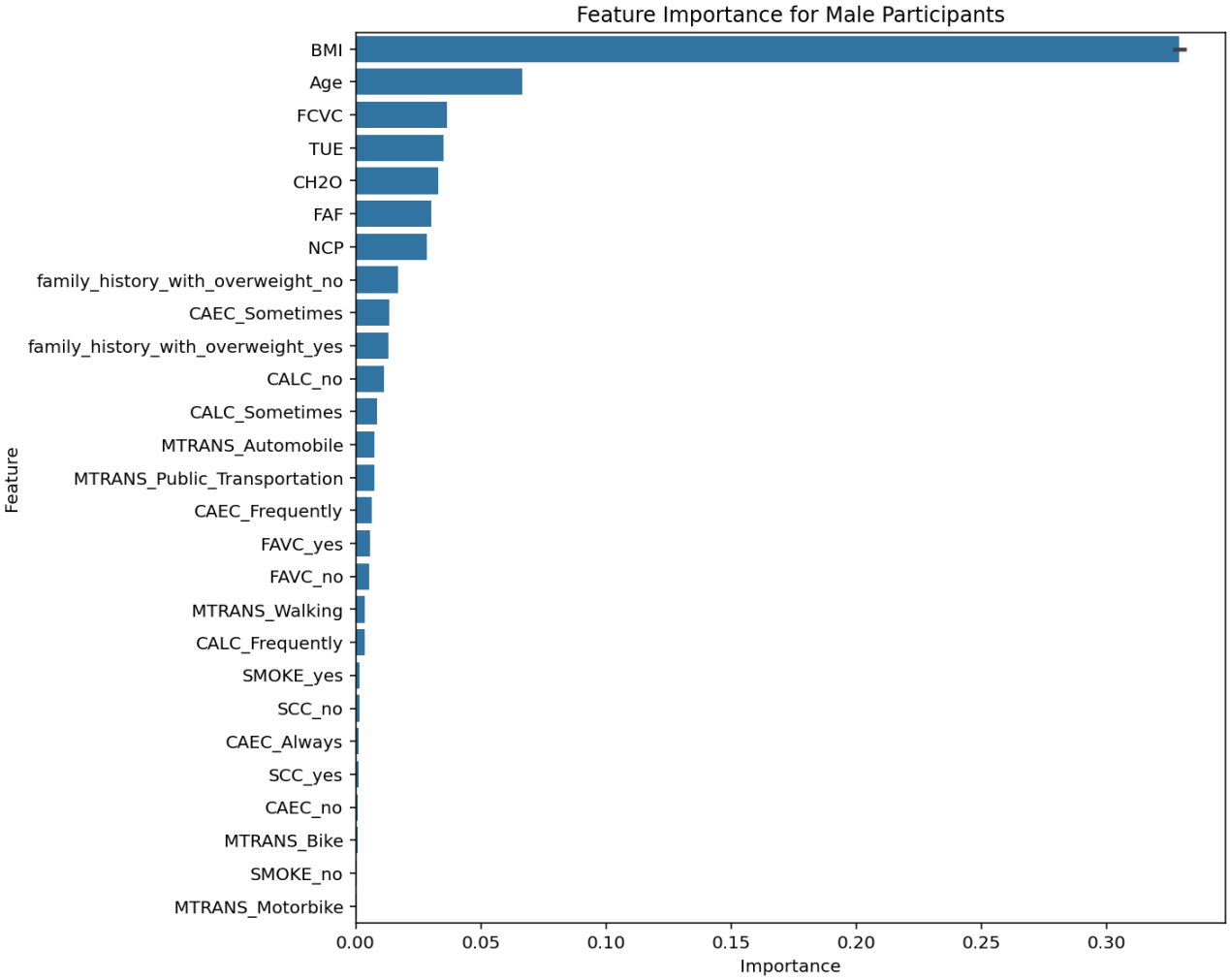


- **Key Findings:**

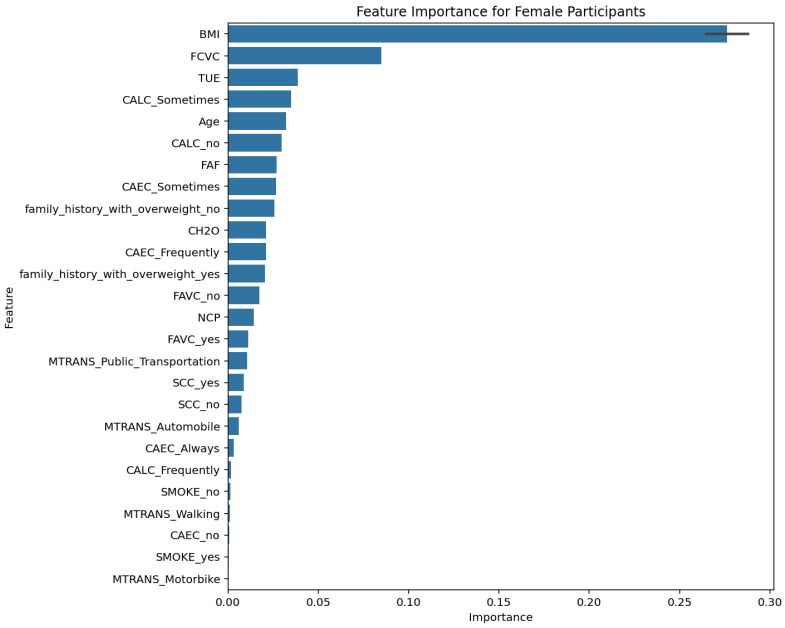
  - **BMI is the strongest predictor** of obesity level.

  - **Gender differences:**

    - For males: **physical activity** was more influential.



    - For females: **dietary habits** played a bigger role.



2.7 Outlook

- **Future Work:**

  - Incorporate **more granular dietary data** (e.g., sugar intake, processed food consumption).

  - Test **deep learning models** (e.g., Neural Networks) for improved classification.

  - Deploy the model as a **web-based obesity risk assessment tool**.

Conclusion

This project demonstrates that **machine learning can effectively classify obesity levels**, with AdaBoost emerging as a strong candidate. The analysis highlights **BMI as a critical feature** but also reveals **gender-based differences in risk factors**. Future improvements could involve **larger datasets** and **real-time health monitoring integration**.

Code and full results available in the repository. 🚀