# Obesity Classification with and without Weight Features

Adelynn Shirts

## 1 Introduction

Understanding and predicting obesity levels is a critical aspect of public health. With obesity rates in the United States on the rise since the early 2000s, it has become essential to carefully monitor this health crisis. In this project, I used data from the UCI Obesity Level Estimation dataset to build models to classify participants as Obese or not based on their lifestyle and demographics.

My goal was to train and evaluate several models to identify the most effective approach to classifying obesity status. I used balanced accuracy as my primary evaluation metric, as the data is slightly imbalanced, but other key metrics, like the AUC and confusion matrices, were also calculated. I also explored feature importance across different models to understand which factors contribute most to obesity.

## 2 Data

The dataset is comprised of 2111 observations and contains both numerical and categorical variables regarding lifestyle and demographics. The original target variable, Nobeyesdad, contained 7 categories ranging from underweight to type 3 obesity. I collapsed the target into a binary response variable where: 0 = Not Obese (Insufficient Weight, Normal Weight, Overweight I, Overweight II) and 1 = Obese (Obesity I, Obesity II, Obesity III).

The final dataset included variables that cover gender, age, height, weight, and various eating and lifestyle habits. I also converted categorical variables to numeric encodings summarized in the table below. As some of the models require scaled inputs, the data was then standardized.

| **Variable** | **Original Values** | **Numeric Encoding** |
|---|---|---|
| Gender | Male, Female | Male = 0, Female = 1 |
| family_history_with_overweight | no, yes | no = 0, yes = 1 |
| FAVC | no, yes | no = 0, yes = 1 |
| SMOKE | no, yes | no = 0, yes = 1 |
| SCC | no, yes | no = 0, yes = 1 |
| CAEC | no, Sometimes, Frequently, Always | 0 = no, 1 = Sometimes, 2 = Frequently, 3 = Always |
| CALC | no, Sometimes, Frequently, Always | 0 = no, 1 = Sometimes, 2 = Frequently, 3 = Always |
| MTRANS | Public_Transportation, Automobile, Walking, Motorbike, Bike | 0 = Public_Transportation, 1 = Automobile, 2 = Walking, 3 = Motorbike, 4 = Bike |

Table 1: Encodings for categorical variables

## 3 Preliminary Data Analysis and Visualizations

After standardizing the data, I calculated some basic summary statistics to get a better idea of the distributions of the data. I further made histograms to model the numeric data and bar charts for the categorical features, as seen below.

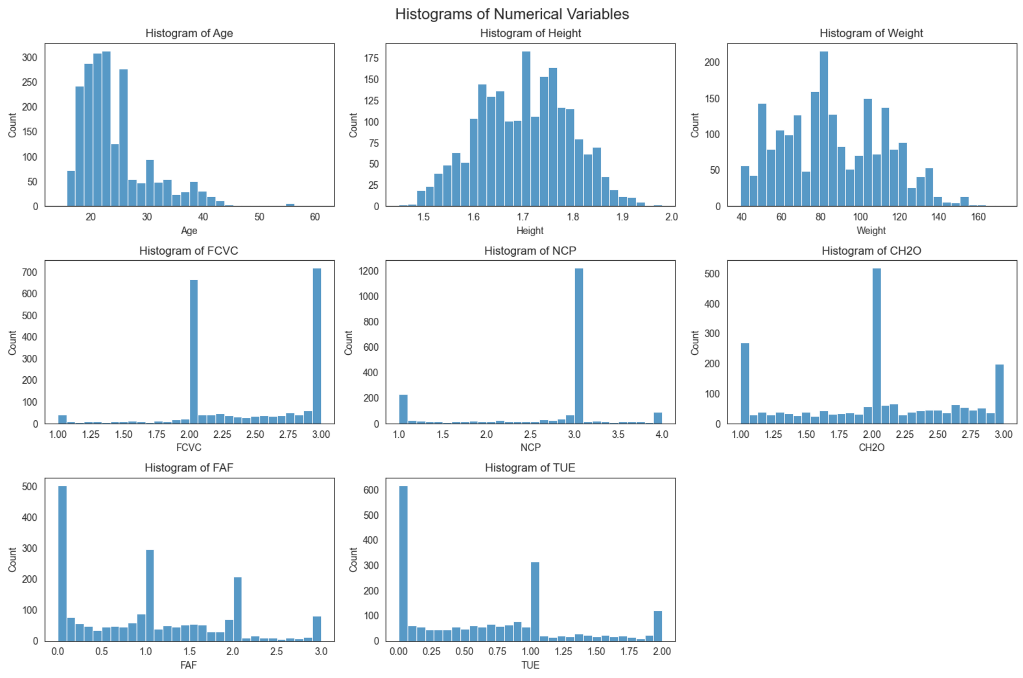


Figure 1: Histograms of numerical variables

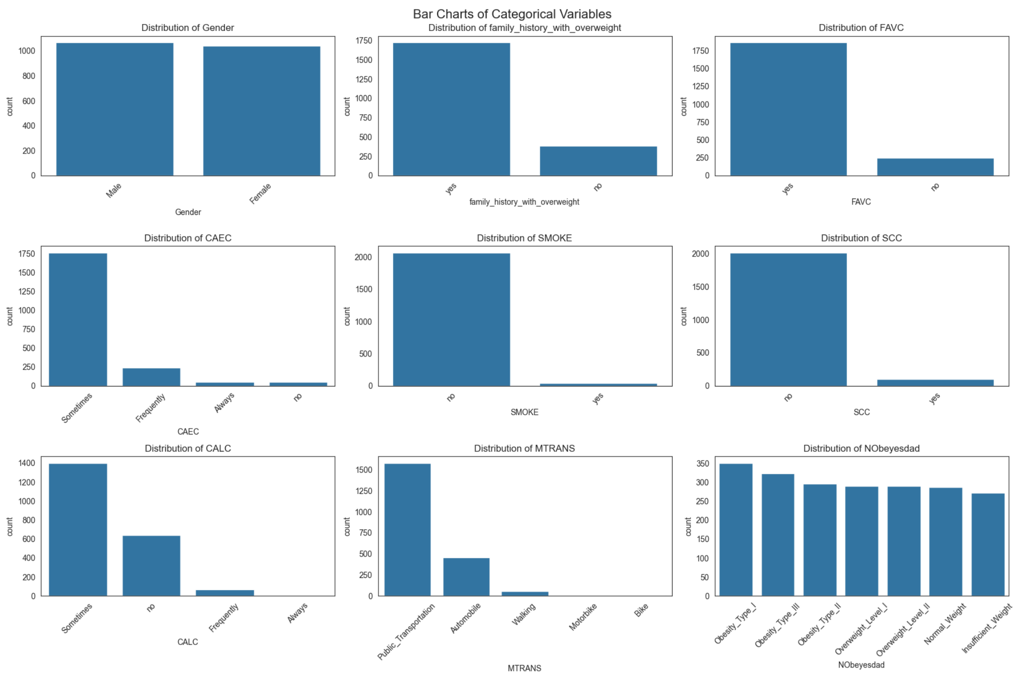


Figure 2: Bar charts of categorical variables

After collapsing the response variable, I replotted the response variable, as seen below. While there is only a slight class imbalance, I still account for it later when I build the models.

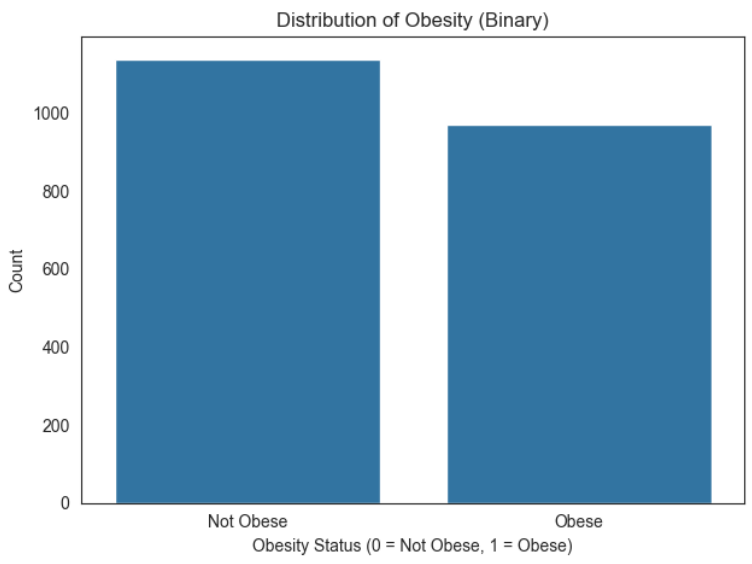
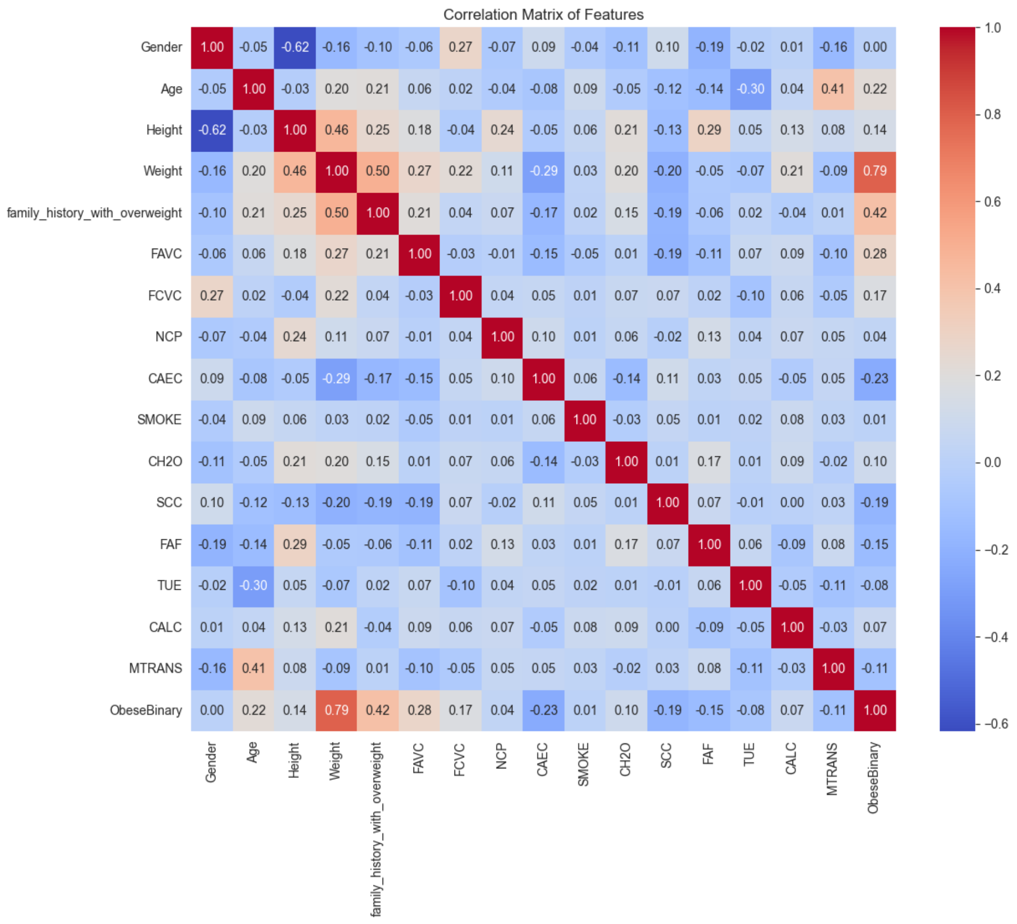


Figure 3: Distribution of the target variable

I finally calculated and plotted a correlation matrix to observe the relationships between the variables in the dataset. While most variables showed insignificant correlations, the relationship between weight and the binary response variable, ObeseBinary, is, unsurprisingly, notable. For this reason, I ran two versions of this project with the same set of testing and training data, one with weight included in the model and one without, to observe the differences in performance and hyperparameter selection between the two variants.



| | Gender | Age | Height | Weight | family_history_with_overweight | FAVC | FCVC | NCP | CAEC | SMOKE | CH2O | SCC | FAF | TUE | CALC | MTRANS | ObeseBinary |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Gender | 1.00 | -0.05 | -0.62 | -0.16 | -0.10 | -0.06 | 0.27 | -0.07 | 0.09 | -0.04 | -0.11 | 0.10 | -0.19 | -0.02 | 0.01 | -0.16 | 0.00 |
| Age | -0.05 | 1.00 | -0.03 | 0.20 | 0.21 | 0.06 | 0.02 | -0.04 | -0.08 | 0.09 | -0.05 | -0.12 | -0.14 | -0.30 | 0.04 | 0.41 | 0.22 |
| Height | -0.62 | -0.03 | 1.00 | 0.46 | 0.25 | 0.18 | -0.04 | 0.24 | -0.05 | 0.06 | 0.21 | -0.13 | 0.29 | 0.05 | 0.13 | 0.08 | 0.14 |
| Weight | -0.16 | 0.20 | 0.46 | 1.00 | 0.50 | 0.27 | 0.22 | 0.11 | -0.29 | 0.03 | 0.20 | -0.20 | -0.05 | -0.07 | 0.21 | -0.09 | 0.79 |
| family_history_with_overweight | -0.10 | 0.21 | 0.25 | 0.50 | 1.00 | 0.21 | 0.04 | 0.07 | -0.17 | 0.02 | 0.15 | -0.19 | -0.06 | 0.02 | -0.04 | 0.01 | 0.42 |
| FAVC | -0.06 | 0.06 | 0.18 | 0.27 | 0.21 | 1.00 | -0.03 | -0.01 | -0.15 | -0.05 | 0.01 | -0.19 | -0.11 | 0.07 | 0.09 | -0.10 | 0.28 |
| FCVC | 0.27 | 0.02 | -0.04 | 0.22 | 0.04 | -0.03 | 1.00 | 0.04 | 0.05 | 0.01 | 0.07 | 0.07 | 0.02 | -0.10 | 0.06 | -0.05 | 0.17 |
| NCP | -0.07 | -0.04 | 0.24 | 0.11 | 0.07 | -0.01 | 0.04 | 1.00 | 0.10 | 0.01 | 0.06 | -0.02 | 0.13 | 0.04 | 0.07 | 0.05 | 0.04 |
| CAEC | 0.09 | -0.08 | -0.05 | -0.29 | -0.17 | -0.15 | 0.05 | 0.10 | 1.00 | 0.06 | -0.14 | 0.11 | 0.03 | 0.05 | -0.05 | 0.05 | -0.23 |
| SMOKE | -0.04 | 0.09 | 0.06 | 0.03 | 0.02 | -0.05 | 0.01 | 0.01 | 0.06 | 1.00 | -0.03 | 0.05 | 0.01 | 0.02 | 0.08 | 0.03 | 0.01 |
| CH2O | -0.11 | -0.05 | 0.21 | 0.20 | 0.15 | 0.01 | 0.07 | 0.06 | -0.14 | -0.03 | 1.00 | 0.01 | 0.17 | 0.01 | 0.09 | -0.02 | 0.10 |
| SCC | 0.10 | -0.12 | -0.13 | -0.20 | -0.19 | -0.19 | 0.07 | -0.02 | 0.11 | 0.05 | 0.01 | 1.00 | 0.07 | -0.01 | 0.00 | 0.03 | -0.19 |
| FAF | -0.19 | -0.14 | 0.29 | -0.05 | -0.06 | -0.11 | 0.02 | 0.13 | 0.03 | 0.01 | 0.17 | 0.07 | 1.00 | 0.06 | -0.09 | 0.08 | -0.15 |
| TUE | -0.02 | -0.30 | 0.05 | -0.07 | 0.02 | 0.07 | -0.10 | 0.04 | 0.05 | 0.02 | 0.01 | -0.01 | 0.06 | 1.00 | -0.05 | -0.11 | -0.08 |
| CALC | 0.01 | 0.04 | 0.13 | 0.21 | -0.04 | 0.09 | 0.06 | 0.07 | -0.05 | 0.08 | 0.09 | 0.00 | -0.09 | -0.05 | 1.00 | -0.03 | 0.07 |
| MTRANS | -0.16 | 0.41 | 0.08 | -0.09 | 0.01 | -0.10 | -0.05 | 0.05 | 0.05 | 0.03 | -0.02 | 0.03 | 0.08 | -0.11 | -0.03 | 1.00 | -0.11 |
| ObeseBinary | 0.00 | 0.22 | 0.14 | 0.79 | 0.42 | 0.28 | 0.17 | 0.04 | -0.23 | 0.01 | 0.10 | -0.19 | -0.15 | -0.08 | 0.07 | -0.11 | 1.00 |

Figure 4: Correlation matrix for the obesity dataset

## 4 Methods

After separating my data into a stratified test/train split, I employed 8 methods on this dataset. For each method, I fit a baseline model, tuned the hyperparameters and cross-validated using Optuna, and finally evaluated the model on the held-out testing data. I finished the project by using permutation feature importance to compare different methods to determine which variables were most significant in each model.

Each baseline model was fit using various packages from Python's sklearn library. Initially, no hyperparameters were specified; however, in models that learn by minimizing a loss function, such as Logistic Regression, SVM, Decision Trees, and Random Forests, I specified class weights to deal with the slight class imbalance observed in the target variable. The hyperparameters were tuned using Optuna, as mentioned above. Instead of grid search or random search, which can be either computationally demanding or unreliable, Optuna uses Bayesian optimization to efficiently explore the hyperparameter space. In this project, the objective was to maximize the balanced accuracy. Simultaneously, Optuna cross-validates each trial, making this method effective, computationally feasible, and statistically rigorous. After iterating through 50 trials, the best hyperparameters are then implemented into a final model, which evaluates the testing data, giving the final performance metrics and confusion matrices. Finally, I opted to use permutation feature importance to explore the importance of variables since it can be applied to all my models, making it ideal for comparison between methods.

The tuned hyperparameters and balanced accuracy for the tuned models and tables summarizing evaluation metrics for the trials with and without the weight variable are reported below. The confusion matrices for each variant of test data are also reported.

**k-NN:** The optimal value of n-neighbor was 4, and the cross-validated balanced accuracy for the tuned k-NN model was 0.9510 for the variant with weight. In the variant without weight, the optimal value of n-neighbor was, again, 4, and the cross-validated accuracy was 0.9169.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9504 | 0.9517 | 0.9265 | 0.9692 | 0.9474 | 0.9785 |
| Without Weight | 0.9220 | 0.9210 | 0.9219 | 0.9077 | 0.9147 | 0.9675 |

Table 2: Evaluation metrics for k-NN with and without the weight variable

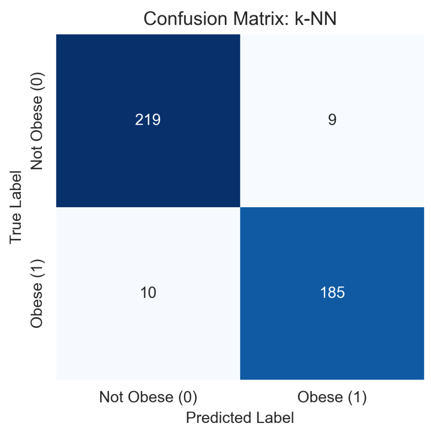


Figure 5: k-NN Confusion Matrix (with weight)

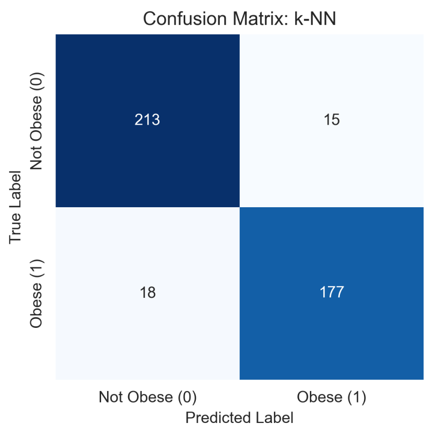


Figure 6: k-NN Confusion Matrix (without weight)

In this model, the evaluation metrics seem fairly consistent across both variants of the analysis. While the model without weight did perform worse, that was unsurprising. Overall, the model seemed to classify the presence and lack of obesity equally. The number of nearest neighbors, n-neighbor, was the same for both analyses at 4. Interestingly, the cross-validated balanced accuracy was lower than the testing balanced accuracy for both variants of the analysis.

**LDA:** Since LDA does not require hyperparameter tuning, it was evaluated directly using 5-fold cross-validation. The balanced accuracy achieved was 0.9698 in the variant with weight. In the variant without weight, the balanced accuracy was 0.7433.

| **Class Weights** | **Accuracy** | **Balanced Acc.** | **Precision** | **Recall** | **F1 Score** | **ROC AUC** |
|---|---|---|---|---|---|---|
| With Weight | 0.9764 | 0.9766 | 0.9695 | 0.9795 | 0.9745 | 0.9984 |
| Without Weight | 0.7589 | 0.7630 | 0.7067 | 0.8154 | 0.7571 | 0.8523 |

Table 3: Evaluation metrics for LDA with and without the weight variable

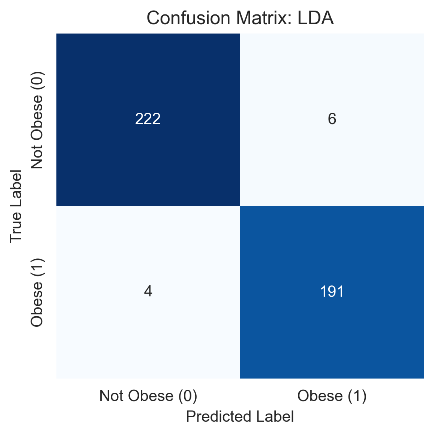


Figure 7: LDA Confusion Matrix (with weight)

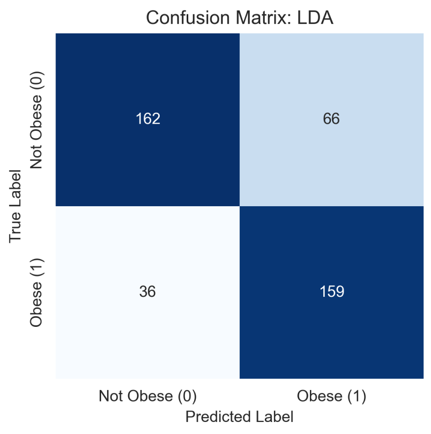


Figure 8: LDA Confusion Matrix (without weight)

LDA performed significantly worse when weight was removed from the analysis, indicating weight was critical for this model to classify accurately. Regardless, the confusion matrices report fairly even misclassifications for both categories.

**QDA:** Since QDA does not require hyperparameter tuning, it was evaluated directly using 5-fold cross-validation. The balanced accuracy achieved was 0.8696 in the variant with weight. In the variant without weight the balanced accuracy was 0.7880.

| **Class Weights** | **Accuracy** | **Balanced Acc.** | **Precision** | **Recall** | **F1 Score** | **ROC AUC** |
|---|---|---|---|---|---|---|
| With Weight | 0.8983 | 0.9024 | 0.8455 | 0.9538 | 0.8964 | 0.9756 |
| Without Weight | 0.8038 | 0.8143 | 0.7171 | 0.9487 | 0.8168 | 0.9353 |

Table 4: Evaluation metrics for QDA with and without the weight variable

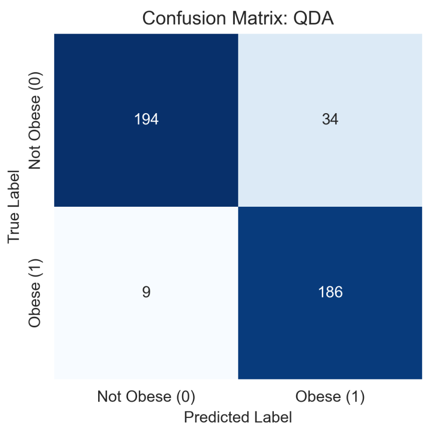


Figure 9: QDA Confusion Matrix (with weight)

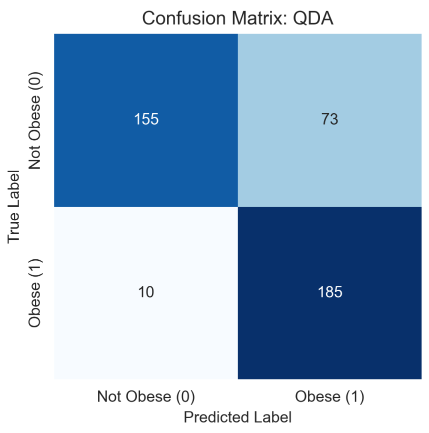


Figure 10: QDA Confusion Matrix (without weight)

QDA also shows a significant drop in balanced accuracy from the model with weight to the model without weight, much like LDA. However, like k-NN, the cross-validated balanced accuracies are higher than the testing balanced accuracies. QDA also seems to misclassify those who are not obese at higher rates than other models, as is confirmed by the confusion matrices and low precision metric.

**Logistic Regression:** The best parameters for Logistic Regression were penalty $= L_1$ and $C = 0.1277$. The cross-validated balanced accuracy was 0.9982 for the variant with weight. In the variant without weight, the best parameters were penalty = elasticnet and $c = 0.1250$. The optimal $L_1$ ratio was 0.3301 and the balanced accuracy was 0.7657.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9976 | 0.9978 | 0.9949 | 1.0000 | 0.9974 | 1.0000 |
| Without Weight | 0.7849 | 0.7908 | 0.7222 | 0.8667 | 0.7879 | 0.8614 |

Table 5: Evaluation metrics for Logistic Regression with and without the weight variable

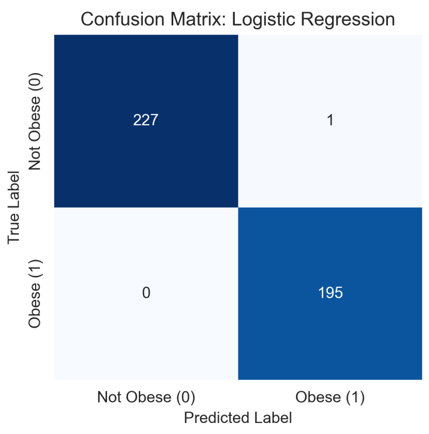


Figure 11: Logistic Regression Confusion Matrix (with weight)

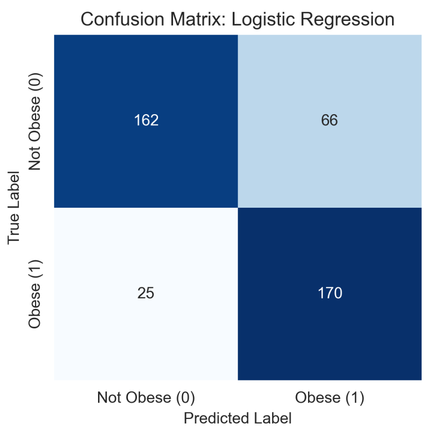


Figure 12: Logistic Regression Confusion Matrix (without weight)

For the logistic regression models, Optuna found different regularization methods were best when the analysis did and did not contain the weight variable. For the model with weight, it found $L_1$ regularization, or LASSO regression, was best for the data. Without weight, it found elastic net regularization, a combination of $L_1$ and $L_2$ penalties, was better. This model saw significant drops in balanced accuracy from one variant to the other. Surprisingly, the model with weight had perfect recall, and only misclassified one observation. In the model without weight, it seems to have lower precision, which is confirmed by looking at the misclassification distribution in the confusion matrix.

**Decision Tree:** The optimal hyperparameters for the Decision Tree model were max_depth = 7, min_samples_split = 3, and min_samples_leaf = 7 and the cross-validated balanced accuracy was 0.9911 for the variant with weight. In the variant without weight, the best hyperparameters were max_depth = 17, min_samples_split = 8, and min_samples_leaf = 8. The balanced accuracy was 0.8877.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9764 | 0.9773 | 0.9602 | 0.9897 | 0.9747 | 0.9900 |
| Without Weight | 0.9173 | 0.9184 | 0.8922 | 0.9333 | 0.9123 | 0.9474 |

Table 6: Evaluation metrics for Decision Tree with and without the weight variable

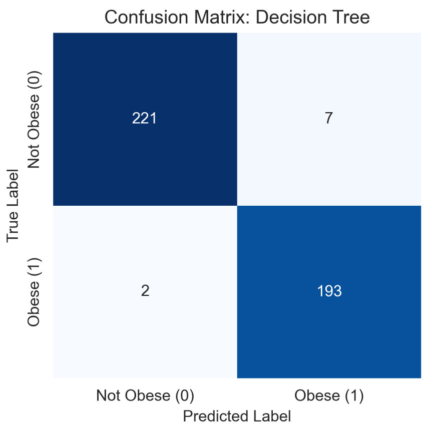


Figure 13: Decision Tree Confusion Matrix (with weight)

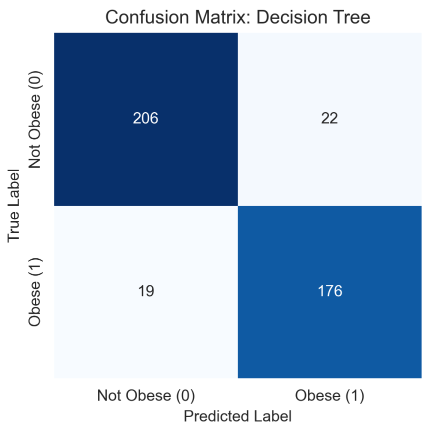


Figure 14: Decision Tree Confusion Matrix (without weight)

The decision tree model in both variants of the analysis were well balanced. It doesn't seem to misclassify one class much more than the other. While the balanced accuracy drops when weight is removed from the analysis, the balanced accuracies are higher than the cross-validated balanced accuracies. The maximum depth of the tree also varied widely with and without weight in the model. With weight, it had a maximum depth of 7, which jumped to 17 when weight was removed from the model.

**Random Forest:** The best configuration for the Random Forest was n_estimators = 146, max_depth = 14, min_samples_split = 2, and min_samples_leaf = 1 and it achieved a balanced accuracy of 0.9928 in the variant with weight. For the variant without weight, the best hyperparameters were n_estimators = 159, max_depth = 16, min_samples_split = 3, and min_samples_leaf = 1. The balanced accuracy was 0.9432.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9929 | 0.9927 | 0.9948 | 0.9897 | 0.9923 | 0.9999 |
| Without Weight | 0.9669 | 0.9660 | 0.9738 | 0.9538 | 0.9637 | 0.9876 |

Table 7: Evaluation metrics for Random Forest with and without the weight variable

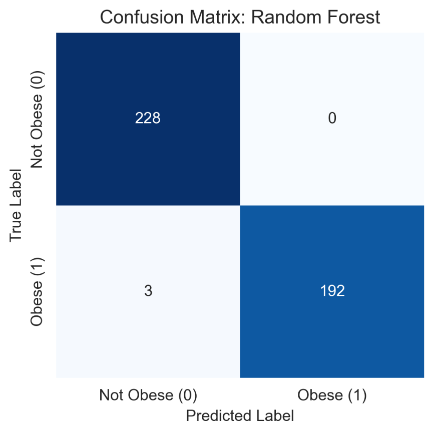


Figure 15: Random Forest Confusion Matrix (with weight)

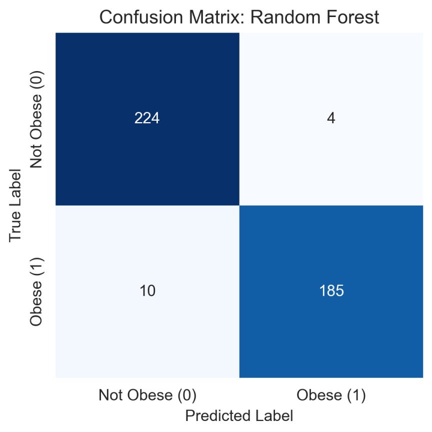


Figure 16: Random Forest Confusion Matrix (without weight)

The results of the random forest model were quite comparable between the analysis with and without weight. There are slight variations the the optimal model hyperparameters, and balanced accuracy did drop slightly when weight was removed, but it remained fairly consistent from one to the other.

**Gradient Boosting:** The tuned Gradient Boosting model used n_estimators = 64, learning_rate = 0.2058, max_depth = 4, and subsample = 0.6816 and the balanced accuracy from cross-validation was 0.9953 for the variant with weight. In the variant without weight, the tuned model used n_estimators = 125, learning_rate = 0.1198, max_depth = 10, and subsample = 0.7758. The balanced accuracy was 0.9513.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9976 | 0.9974 | 1.0000 | 0.9949 | 0.9974 | 1.0000 |
| Without Weight | 0.9764 | 0.9755 | 0.9843 | 0.9641 | 0.9741 | 0.9910 |

Table 8: Evaluation metrics for Gradient Boosting with and without the weight variable

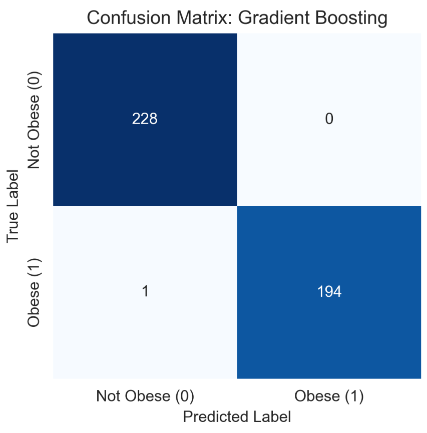


Figure 17: Gradient Boosting Confusion Matrix (with weight)

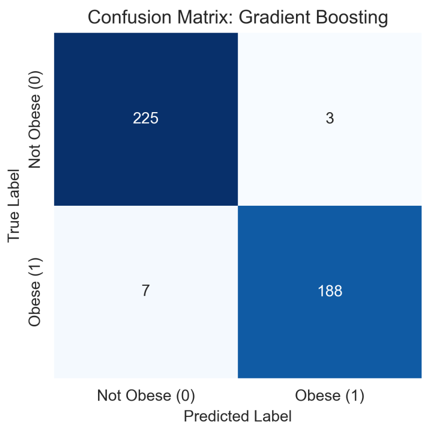


Figure 18: Gradient Boosting Confusion Matrix (without weight)

This model boasts high balanced accuracies, unsurprising for gradient boosting, which is known for strong accuracy rates. There was a negligible drop in balanced accuracy when weight was removed from the analysis. Like logistic regression, when weight was included, there was only 1 misclassification. The model hyperparameters do vary widely with and without weight, but that is to be expected, considering gradient boosting is known for its granularity.

**SVM:** The best hyperparameters for SVM were $C = 9.4319$ and $\gamma = 0.0079$, and the balanced accuracy from cross-validation was 0.9921 for the variant with weight. In the variant without weight, the best hyperparameters were $C = 3.3828$ and $\gamma = 0.0963$. The balanced accuracy was 0.9172.

| Class Weights | Accuracy | Balanced Acc. | Precision | Recall | F1 Score | ROC AUC |
|---|---|---|---|---|---|---|
| With Weight | 0.9905 | 0.9909 | 0.9848 | 0.9949 | 0.9898 | 0.9998 |
| Without Weight | 0.9338 | 0.9334 | 0.9282 | 0.9282 | 0.9282 | 0.9763 |

Table 9: Evaluation metrics for SVM with and without the weight variable

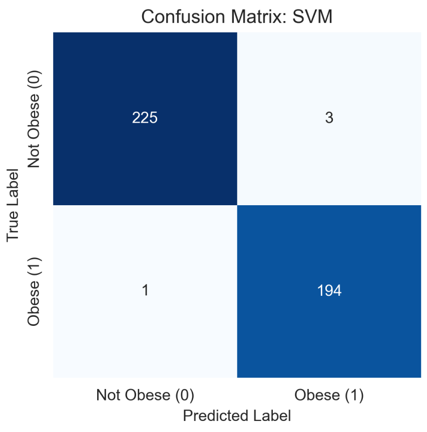


Figure 19: SVM Confusion Matrix (with weight)

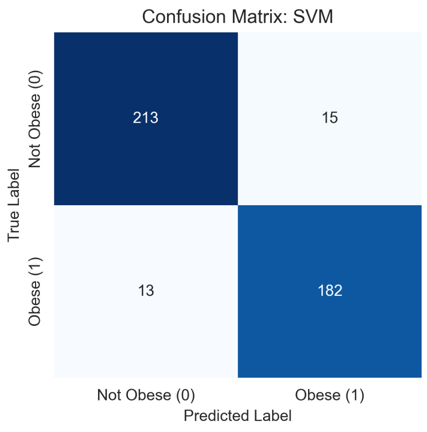


Figure 20: SVM Confusion Matrix (without weight)

Like several other models in this project, SVM seems to have higher balanced accuracies on the testing data compared to the cross-validated set. It seems to classify each category equally in both variants of the analysis. It has fairly high balanced accuracies for both versions of the analysis.

**Conclusions:** Predictably, when weight was removed, all models performed consistently worse. Weight seemed to be a very significant variable in all models. Interestingly enough, despite its removal for the second analysis, the models still performed very well. Balanced accuracies were still relatively high, with the exception of LDA, which was 0.7630, a stark drop from 0.9766 for the same metric when weight is included. In the analysis with weight, all models performed astoundingly high, with the lowest being QDA with a balanced accuracy of 0.9024. the best model was logistic regression with an $L_1$ penalty, which is LASSO regression. The analysis without weight, on the other hand, found the most success using Gradient Boosting, which boasts a balanced accuracy at 0.9755. The confusion matrices for models in each analysis are promising–they don't seem to favor one class over another.

## 5 Feature Importance

After calculating the performance metrics and confusion matrices for each method, I also did permutation feature importance. Below is a graph describing which features were most important for each model for both the variation with and without the weight variable.

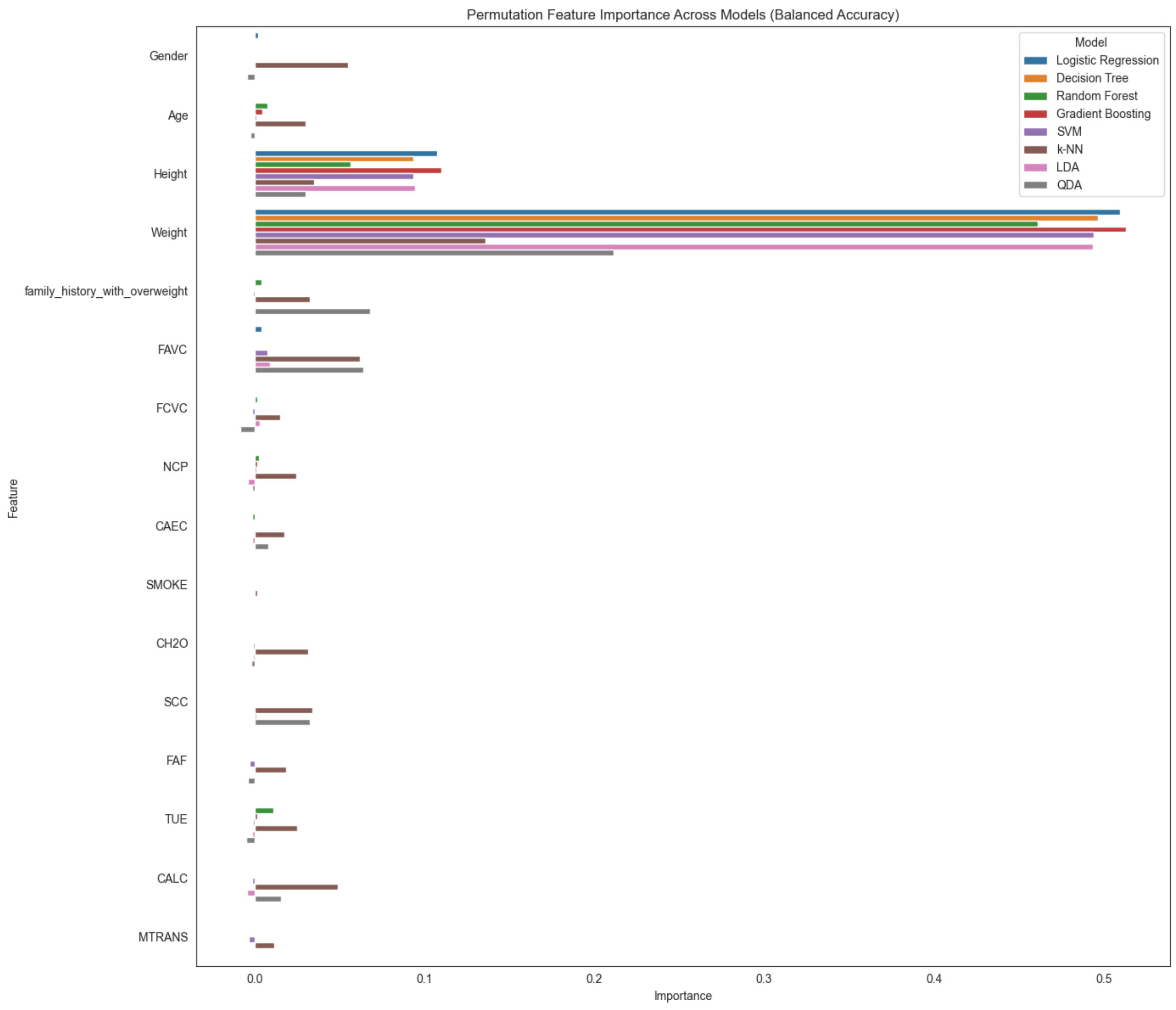


Figure 21: Permutation feature importance comparison with weight variable

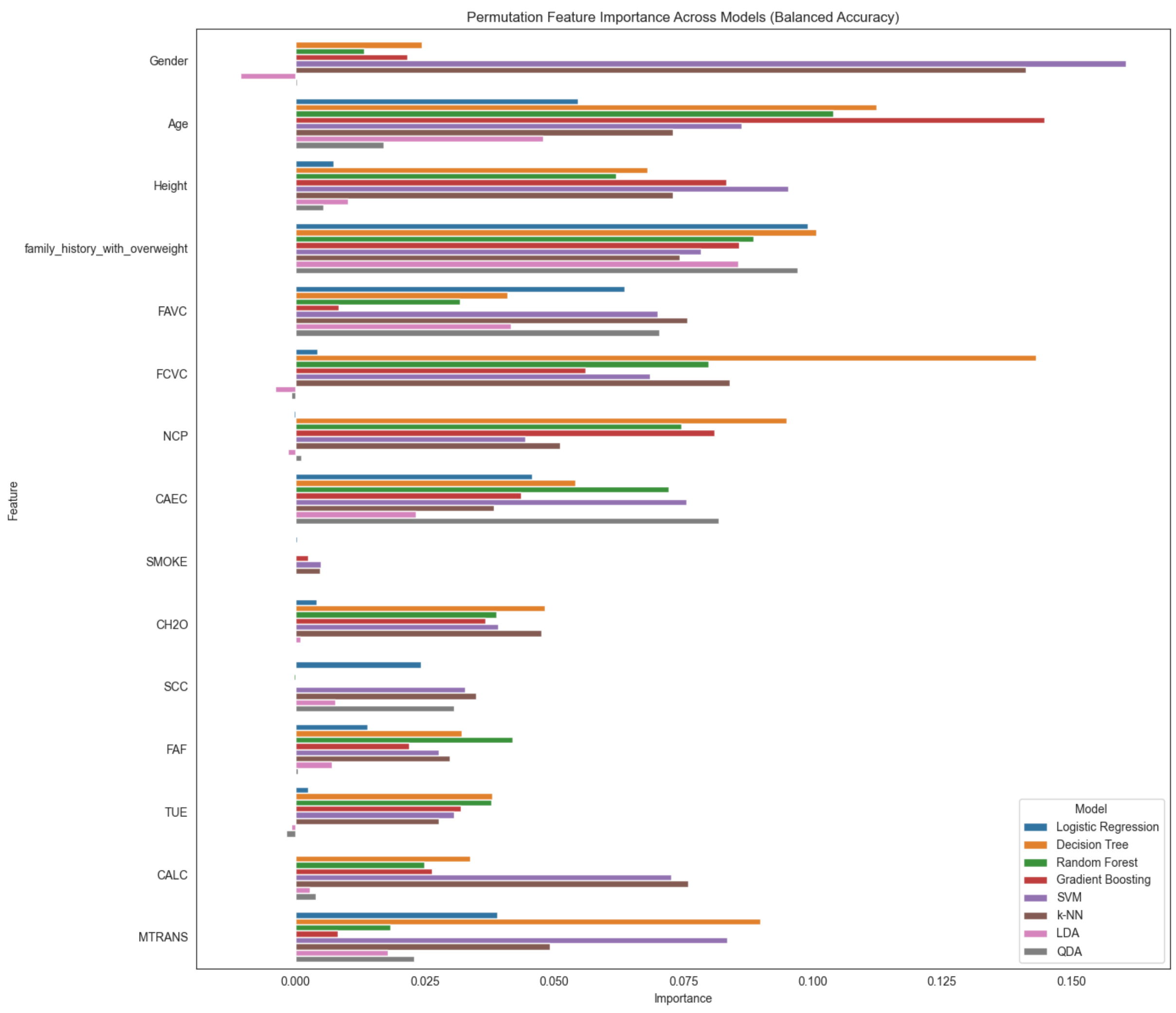


Figure 22: Permutation feature importance comparison without weight variable

Unsurprisingly, in the variant with weight included, it is the variable of most importance in all models, though height also seemed to be significant. Of all the models, it seems k-NN seems to make the most use of all the variables to make a decision, followed by QDA. When the weight variable is removed, all features are used in a greater capacity. It is challenging to determine which feature is most important overall, as each model operates differently; however, age, height, and family history, FCVC (usually eats vegetables with meals), and CAEC (eating food between meals), seem like good candidates. Smoking seems to be the least relevant feature in all models.

## 6 Final Thoughts

While obesity remains an existing issue facing the global community, the data is clear–lifestyle factors have a strong effect on weight. After removing weight from the model, many of the most influential factors in determining obesity had to do with lifestyle and dietary choices, as seen in Figure 22. I was surprised with how high the balance accuracies were in this analysis, even with the weight variable removed. While I suspect there would be a distinction between the categories, I was not expecting it to be so clear across all model types. After considering the evaluation metrics, gradient boosting stands out as a clear winner. With its strong, accurate classification power and adaptability between the 2 variants of the analysis, it is the most effective approach for classifying obesity status.