Using Data Mining to Predict the Obesity levels of Individuals

CMP7206 - Data Mining

Team Members

Shriya Sami - 19101611
Mitra Bitaraf Fazel - 20121126
Muqadas Sohail - 18103345

BIRMINGHAM CITY
University

MSc Big Data Analytics

December 2023

Contents

List of Tables

List of Figures

1 Introduction

This report discusses the issue of obesity within the healthcare domain and the data mining techniques that can be utilised to predict the obesity levels of individuals based on variables such as lifestyle and physical factors. This report will consist of seven main sections beginning with the discussion of the domain and the problem definition, followed by a literature review on the data mining techniques utilised within the domain. The subsequent sections include the dataset description and pre-processing techniques that have been applied. Lastly, the final sections include the experiments which detail the model development process followed by the analysis and conclusions section.

2 Domain Description

The healthcare domain aims at improving human health, medical services and research. Data mining is defined as the discovery of 'useful patterns and trends' from data, (Larose and Larose, 2014). Advances in the data mining field have led it to be used across many industries, including healthcare.

With data mining in healthcare, useful insights can be obtained from data. These insights allow healthcare professionals to recommend appropriate interventions, 'effective treatments and best practices' to patients, (Koh and Tan, 2011). Overall, this leads to improved health outcomes and timely resource allocation.

3 Problem Definition

Obesity is a growing problem worldwide with the potential of an increased risk of cardiovascular diseases and cancers, (De-La-Hoz-Correa et al., 2019; WHO, 2021). The WHO (2021) defines overweight and obesity as medical problems of 'abnormal or excessive fat accumulation.' As of May 2023, 63.8% and 25.9% of adults in England were reported to be overweight or obese, (Timpson, 2023). Although preventable, the prediction of obesity is complex as there are several possible contributing factors. This includes biological, physical, and environmental factors, (Wright and Aronne, 2012).

While data mining does not always identify causal relationships, it allows relationships to be identified between features (Seifert, 2004). This is particularly useful for the complex prediction of obesity to identify relationships between its possible contributing factors.

This report focuses on addressing the knowledge discovery problem of identifying the primary factors influencing an individual's classification as obese or overweight. The objective is to employ data mining techniques, analysing a spectrum of factors encompassing both physical and lifestyle factors. By classifying individuals' obesity levels through data mining, the aim is to pinpoint and understand the key contributors to obesity.

4 Literature Review

The K-Nearest Neighbours (KNN) classifier uses 'proximity' to make classification predictions, (IBM, 2023a). This proximity measure is defined by k: k is the number of neighbours. The k and the distance parameters of the model are its two key parameters that can be tuned for model improvement. This makes KNN straightforward to implement and tune. However, KNN is prone to the dimensionality curse which is when query points in high dimensional spaces become equal as the classifier is unable to discriminate data points, (Kouiroukidis and Evangelidis, 2011). This can lead to the model overfitting. KNN has been used in research by Ferdowsy et al. (2021) to predict risk of obesity. In this research, KNN achieved accuracy highs of 77.5% and, sensitivity and specificity of 100%. The sensitivity shows that the model was able to predict individuals with obesity correctly; this is highly useful for the domain.

The decision tree classifier follows a tree structure of nodes and branches. Decision trees follow a top-down approach to repeatedly decide and split the tree. Parameters of the decision tree can be specified to control how the tree is split and how much it can grow. To select the attribute to be split upon, the two popular splitting methods are information gain and Gini impurity, (IBM, 2023b). Decision trees are easy to implement and are flexible to various types of machine learning problems. However, decision trees are prone to overfitting which can be avoided with pruning. The decision tree has been used in research by (Dugan et al. (2015) and has obtained high values of sensitivity. Dugan et al. (2015) also found model performance to improve after 'noisy' attributes were removed, indicating the presence of high dimensionality.

The random forest classifier consists of multiple decision trees, and its prediction is based upon the 'most popular result,' (IBM, 2023c). To train a random forest classifier, the node size, number of trees and number of features must be specified; due to this specificity, random forest classifiers have less likelihood of overfitting, (IBM, 2023c). Random forests are more complex than the standard decision tree, but they allow important features to be determined by using Gini importance and permutation importance. As the random forest is built on decision trees, it was the second best performing model in research by (Dugan et al., 2015). Again, this was after feature selection to remove certain attributes.

A review of obesity prediction literature enhanced our understanding of the problem definition and the approach we should take. A more detailed review of some of the literature can be found at Appendix A. Based on the review of literature, a dataset and appropriate data mining techniques were chosen.

5 Dataset Description

The chosen dataset ('Estimation of Obesity Levels Based on Eating Habits and Physical Condition') consists of 17 attributes and 2111 records (see Table 1). This dataset was found on UCI Machine Learning Repository, where the data was originally collected through a web survey. Individuals aged between 14 and 61 from Mexico, Peru and Colombia completed this survey anonymously. The data was processed to remove missing values and the following seven categories were created based on the individuals' obesity status': InsufficientWeight, NormalWeight, OverweightLevelI, OverweightLevelII, ObesityTypeI, ObesityTypeII and ObesityTypeIII. However, a significant class imbalance was present. To overcome this, synthetic data was generated using SMOTE; the synthetic data equates to 77% of the complete dataset, (Palechor and Manotas, 2019).

Table 1: Description of the attributes in the dataset and their respective statistical data types.

Attribute	**Description**	**Statistical Data Type**
Gender	Gender of individuals in the dataset, either 'Male' or 'Female'.	Nominal
Age	Age of individuals in years.	Ratio
Height	Height of the individuals in the dataset in metres (m).	Ratio
Weight	Weight of individuals in kilograms (kg).	Ratio
family_history_with_overweight	Binary values ('Yes' or 'No') indicating whether an individual has a family history of being overweight.	Binary
FAVC	Binary variable ('Yes' or 'No') indicating whether the individual frequently consumes high caloric food.	Binary
FCVC	Measure of how often the individual consumes vegetables.	Ordinal
NCP	The number of main meals the individual consumes in a day.	Ordinal
CAEC	Consumption of food between meals as a categorical variable with values like 'Frequently,' 'Sometimes,' etc.	Ordinal
SMOKE	Whether the individual smokes or not as a binary value either 'Yes' or 'No'.	Binary
CH2O	Daily water intake of the individual as a numeric value.	Ordinal
SCC	Represents whether the individual calculates their daily caloric intake as a categorical value e.g., 'Sometimes', 'no', etc.	Binary
FAF	How often the individual partakes in physical activity.	Ordinal
TUE	Time using technology devices (TUE) - Represents the number of hours the individual spends using technology in a day.	Ordinal
CALC	Whether the individual frequently consumes alcohol.	Ordinal
MTRANS	Mode of transportation typically used by the individual and as categorical values.	Nominal
NObeyesdad	The obesity status of the individuals as categorical values.	Ordinal

From the attributes, the ‘NObeyesdad’ attribute was selected as the target variable as it allows a prediction of obesity status. This is a classification data mining task as the target attribute is categorical. A more detailed exploratory data analysis can be found at Appendix B.

6 Dataset Pre-processing

Pre-processing is a key stage that is done prior to model building given that raw data frequently exhibits inconsistencies, such as missing values or data being in incorrect/incomplete formats that are not necessarily understandable by machine learning models, (Kalra and Aggarwal, 2017).

This section will detail the pre-processing techniques applied to the obesity dataset, ensuring the data is formatted to adhere the model’s requirements and to avoid negatively impacting the performance of the models.

6.1 Rounding

Some features of the dataset were floating point values. When training machine learning models, floating point values are understood by the algorithm but require ‘resource intensive’ formatting to avoid any inaccuracies, (Atwell, 2022). Therefore, rounding was performed to standardise all values and remove leading digits.

6.2 Feature Encoding

Some features within the dataset were identified as categorical. Since most machine learning algorithms require input features to be numerical, the categorical variables were encoded via dummy, one-hot and label encoding which are explained further in the following subsections.

6.2.1 One-hot Encoding

One hot encoding is used for nominal data, and it uses N binary variables for N categories per variable, (Saxena, 2023). For each category, a new variable is created, mapping to 1 or 0, indicating presence or absence. This poses a possible challenge of multicollinearity between attributes. To overcome this, dummy encoding was used alongside.

6.2.2 Dummy Encoding

Dummy encoding is also used for nominal data, but it uses N-1 for N categories, (Saxena, 2023). Similar to one hot encoding, for each category, a new variable is created, mapping to 1 or 0. However, in dummy encoding, the redundant variables are then removed. Thus, after dummy encoding, there will always be one less column than categories. This reduces multicollinearity.

6.2.3 Label Encoding

Label encoding is used when the data possess an order and hence, assigns ordered labels to categories. A limitation of label encoding is that some models may assume an order between the categories. Therefore, it is critical that label encoding is only used for ordered data.

7 Experiments (three DM techniques)

The code for the models is provided in the R script file, called 'A2_Final_Code' and will be referenced within the following sections via line numbers.

7.1 Team Identification

Table 2 identifies the team members and the data mining technique they have implemented.

Table 2: Team members and their respective models

Student Name	**Student ID**	**DM technique**
Mitra Bitaraf Fazel	20121126	K – Nearest Neighbours
Shriya Sami	19101611	Decision Tree
Muqadas Sohail	18103345	Random Forest

7.2 K-Nearest Neighbour (KNN)

To build a KNN model for multi class classification task (R file; line 438 – 459), for the initial model three main steps has been taken. Firstly, the class library is included, which is necessary to access the KNN function, a key element for implementing KNN in R. Secondly, an important parameter in KNN, the number of neighbours (k), is set to 5 in this case, indicating that five neighbours in the feature space are considered for each prediction. Finally, the code focuses on training the KNN model using a specified training set (obesity_X_train and obesity_Y_train) and assessing its performance on a validation set (obesity_X_val). The knn function is employed for this task, requiring the data to be converted from data frames to matrices using the "as.matrix" function. The as.matrix function is used to convert the data frames obesity_X_train and obesity_X_val into matrices before training the KNN model. This conversion is necessary because the KNN function in the class library expects its input data to be in matrix format.

Evaluating a model is crucial to assess its performance and generalisation to new, unseen data. This process helps quantify the model's accuracy, reliability, and suitability for the intended task. It provides insights into the model's predictive capabilities, enabling informed decisions about its deployment or refinement. Evaluation metrics, such as accuracy, sensitivity, precision, etc. offer a quantitative basis for comparing models and selecting the most effective one for a given application. Ultimately, robust model evaluation is fundamental for building trustworthy and effective machine learning systems, (McAvaney et al. ,2001).

A validation set is considered for assessing the initial model with the accuracy of 89% on the unseen data. The validation set serves as an independent dataset not used during training, allowing for unbiased evaluation. The validation set helps identify issues like overfitting, where a model performs well on the training set but struggles with new data. It guides model selection, hyperparameter tuning, and ensures the chosen model generalises well to real-world scenarios, enhancing its reliability and predictive accuracy, (Xu & Goodacre, 2018).

The following subsequent sections explore the performances of multiple model iterations in contrast to the initial model.

7.2.1 Iteration 1 – Feature Selection

Feature importance is not inherently applicable to k-Nearest Neighbours (KNN) models (R File; 460 – 493), unlike models such as decision trees or random forests where features are explicitly ranked by importance. However, in the context of a group project, decision tree and random forest models were generated and utilised to assess feature importance. During the analysis, it was observed that the "MTRANSMotorbike" and "MTRANSBike" columns held the least importance among all features according to the Random Forest feature importance technique. As a result, in the second iteration of the project, these two columns were deliberately dropped from consideration, presumably due to their limited impact on the model's predictive performance or relevance to the target variable.

After dropping these columns, the model slightly improved. Even marginal enhancements in model performance can significantly contribute to mitigating challenges such as increased computational complexity, overfitting, heightened training time, and the risk of multicollinearity. Each of these factors plays a pivotal role in crafting a high-performance model. Incremental improvements can help streamline computational demands, enhance generalisation by addressing overfitting concerns, expedite training processes, and reduce the risk of multicollinearity-induced instability. Recognising the importance of these aspects underscores the significance of fine-tuning models to achieve optimal balance and effectiveness in addressing complex data science challenges.

7.2.2 Iteration 2 – Scaling

Scaling the data (R File; line 494 – 513) is crucial in KNN models due to their sensitivity to the magnitude of features. KNN relies on distance metrics, such as Euclidean distance, to identify nearest neighbours. When features have different scales, those with larger magnitudes can dominate the distance calculation, rendering others less influential. Scaling ensures that all features contribute equally to distance computations, preventing bias towards variables with larger scales. This results in a more accurate representation of proximity, enhances model stability, and better generalisation by preventing the influence from certain features, therefore improving the overall reliability and effectiveness of the KNN algorithm, (Ahsan et al., 2021).

The drop in validation set accuracy from 89% in the initial model to 75% after scaling may stem from the impact of feature scaling on the model's sensitivity to varying magnitudes. Scaling ensures uniformity in the contribution of each feature during model training. However, it depends on the nature of the algorithm. As an example, scaling typically does not have a significant impact on tree-based models such as decision trees, random forests, and gradient boosting machines. Tree-based models make decisions based on feature thresholds and do not rely on the scale of the features, (Ahsan et al., 2021).

7.2.3 Iteration 3 – Hyperparameter Tuning

Hyperparameter tuning in the k-Nearest Neighbors (KNN) model (R File; line 514 – 548) offers substantial benefits by optimising its performance. By systematically exploring a range of k values through techniques like grid search, the ideal hyperparameter configuration is identified. This process enhances the model's ability to capture underlying patterns in the data, preventing underfitting or overfitting. Fine-tuning the hyperparameters, particularly k, ensures a balance between model complexity and generalisation, improving predictive accuracy on unseen data. It also aids in mitigating the risk of sub-optimal choices, enabling the KNN algorithm to adapt more effectively to the intricacies of diverse datasets and enhancing its overall reliability in real-world applications, (Ghawi & Pfeffer, 2019).

To be able to tune the model, the caret package is used to perform a grid search for hyperparameter tuning in a k-Nearest Neighbours (KNN) model. It initialises a grid of k values ranging from 1 to 20. Using 5-fold cross-validation as the evaluation method, it sets up control parameters. The train function then iteratively fits KNN models with varying k values on the scaled training data (obesity_X_train_scaled) and corresponding labels (obesity_Y_train). The model's performance is assessed using the mean accuracy across folds, and the optimal k value is determined. The resulting tuned KNN model (knn_tuned) is produced for validation on new data to evaluate its generalisation performance.

7.2.4 Analysis of K-NN Results

Table 3 is illustrating the initial KNN's accuracy score and the three iterations on validation set.

Table 3: Tuning Process for the KNN models

Model	**Parameters**	**Accuracy (on validation set)**
Initial Model	Default parameters with all the features included	0.89
Iteration 1	Dropping the least important columns according to Feature importance using the LVQ method	0.87
Iteration 2	Scaling – Standardization	0.75
Iteration 3	Hyperparameter Tuning – Grid search on K value	0.80

7.3 Decision Tree

To build and fit a decision tree classifier, the rpart package is suitable for both classification and regression, and applicable to various predictive scenarios. The X and Y subsets of the training set were passed into the rpart function, (line 180). The method parameter has been specified as 'class' as the target variable, Y, is categorical.

The fancyRpartPlot wrapper was used to plot the decision tree classifier to provide an aesthetically enhanced visualisation of decision trees built with rpart. Line 187 displayed Figure 1.

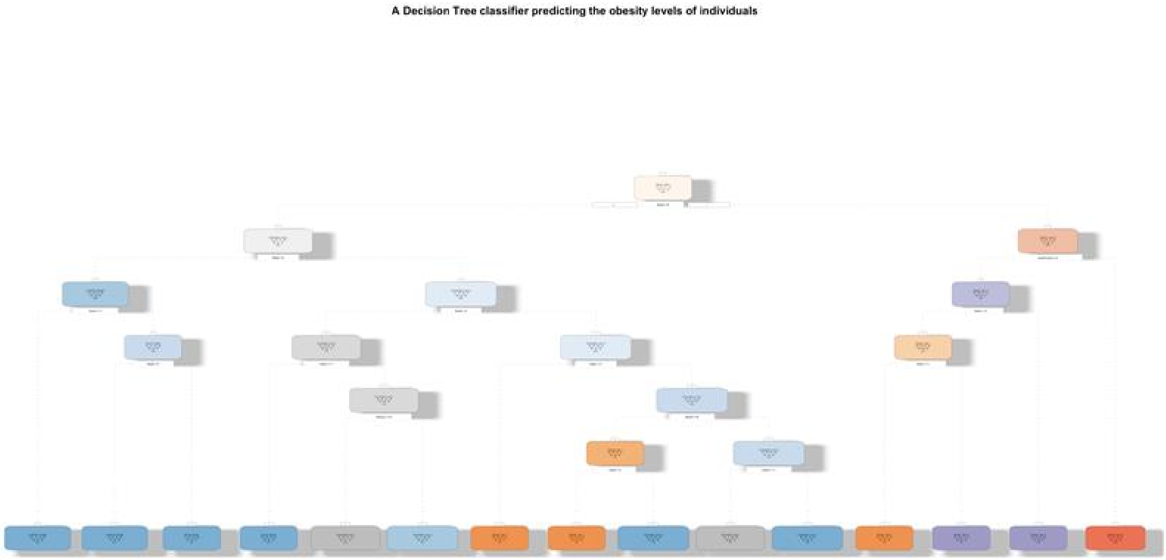

Figure 1: A Decision Tree classifier predicting the obesity levels of individuals

The decision tree classifier made a prediction using the predict function. The confusion matrix was returned: a tabular representation summarizing the performance of a classification model by detailing the counts of true positive, true negative, false positive, and false negative predictions, aiding in the evaluation of model accuracy and error. This showed that the model obtained 87% accuracy.

The decision tree classifier then made a prediction on the validation set which assesses model performance through an independent subset to identify potential overfitting, allows for robust model evaluation and an estimation of how the model would perform on testing data, (James et al., 2013). The confusion matrix for this prediction obtained 83% accuracy.

Accuracies of 87% and 83% show that the model performed better on the training set. This indicates that model is overfitting the training data. Overfitting occurs when a model learns the specific patterns and noise in the training data to such an extent that it performs poorly on new, unseen data, (Ying, 2019). To address this issue, an array of improvements has been done.

7.3.1 Iteration 1 – Feature Selection

The first improvement made was feature selection. By removing unnecessary or redundant variables, feature selection simplifies a model which then lowers the likelihood of overfitting and boosts generalisation to new data. This leads to more accurate machine learning models that are suitable for usage in a variety of real-world applications, (Brownlee, 2021).

To achieve feature selection, the cor function was used to assign the correlation values of the attributes in the validation set to a variable, correlationMatrix, (line 204). The cor function summarises the pairwise correlation coefficients between variables in a dataset and calculates and displays correlation matrices.

For this list of correlation values, the findCorrelation function was used to obtain the attribute names of those with the highest correlation, (line 210). The cutoff parameter was used to specify a 0.5 threshold for the correlation values.

The attributes with the highest correlation, meeting the 0.5 threshold, were: Height, MTRANSPublic_Transportation, Weight and Age. All these attributes were dropped from the training and validation sets. The modified training and validation sets were then used for the next model improvement.

7.3.2 Iteration 2 – Tree Splitting

With the reduced attributes, the next improvement that was done was tree splitting. Generally, decision trees follow a 'top-down, greedy approach' making the best split each time, (Dash, 2022). To modify this approach, a rpart decision tree can be optimally segmented to improve the accuracy of the model and adapt to nonlinear relationships.

To achieve tree splitting, the gini and information splits were considered. Again, the rpart function was used to split the tree, but the split parameter was specified. Lines 225 and 228 demonstrate the decision tree with the gini split being trained on and making a prediction for the training set. Similarly, the gini split decision tree was used to make a prediction on the validation set, (line 234). The confusion matrices for the gini split decision tree for the training and validation sets shows that the model obtained accuracies of 59% and 56% respectively.

Lines 240 and 244 demonstrate the decision tree with the information split being trained on and making a prediction for the training set. Likewise, the information split decision tree was used to make a prediction on the validation set, (line 250). The confusion matrices for the information split decision tree for the training and validation sets shows that the model obtained accuracies of 54% and 53% respectively.

This significant decrease in accuracies from the initial validation set highlights an issue in the feature selection process. In the feature selection process, all the highly correlated attributes were removed, which may have oversimplified the model. When considering the highly correlated attributes, in relation to the application domain, height, weight and age are key factors of obesity, (WHO, 2021). Although the MTRANSPublic_Transportation attribute could be linked with physical activity, it has a less direct relation to obesity. Therefore, the feature selection process was re-reviewed and only the MTRANSPublic_Transportation attribute has been dropped.

The gini split and information split decision trees were used again to make predictions on the validation sets. The confusion matrices showed the gini split and information split decision trees to obtain accuracies of 83% and 84%, respectively.

As the information split decision tree obtained a higher accuracy, it was used for the next model improvement.

7.3.3 Iteration 3 – Tree Pruning

With a more accurate model, the next improvement was tree pruning. Pruning helps prevent overfitting by strategically trimming branches of the tree, guaranteeing improved model simplicity and improved generalisation to new data. A pre-pruning approach was used for early stopping of the tree's growth so that it is unable to overfit, (Dash, 2022). To do this, the trees Complexity Parameter (CP) was obtained and plotted; the CP is used to control the complexity of the tree.

To determine the optimal CP value, cross-validation was conducted. By evaluating the model's performance on different subsets of the training data, the algorithm can find the complexity parameter that results in the best trade-off between model complexity and accuracy.

From the table of CP values, the minimum cross validated error was assigned to a variable, (line 262). The corresponding value for the minimum cross validated error was assigned to another variable called cp_optimal, (line 265). This cp_optimal variable is used when the rpart function is used again, (line 268). This rpart function is used to build the decision tree, but this time the control parameter has been specified as the cp_optimal variable. The xval parameter specifies the number of cross validation folds for the data to be divided into; in this case, 100 has been chosen.

The decision tree is then pruned and plotted (Figure 2).

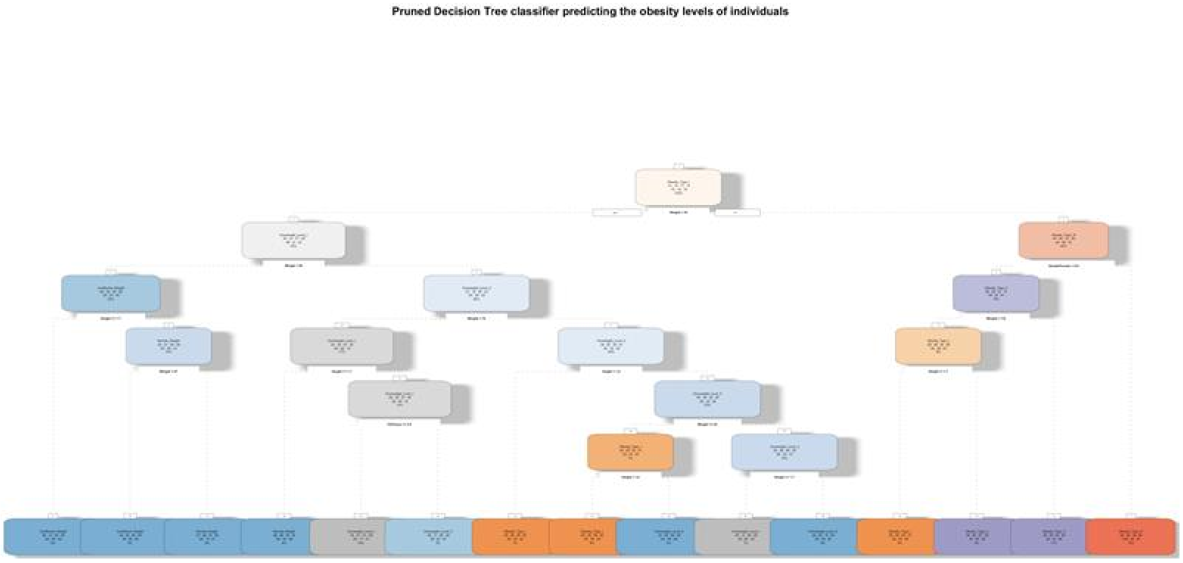

Figure 2: Pruned Decision Tree classifier predicting the obesity levels of individuals

The pruned decision tree is used to make a prediction on the training and validation sets. Confusion matrices were obtained which show that the decision tree achieved 87% and 83% accuracy for training and validation.

This pruned decision tree was then used to make the final prediction on the validation set.

7.3.4 Final Validation

For the final prediction on the validation set, the pruned decision tree was used. This returned an accuracy of 83%. This pruned decision tree will be used to test the model.

7.3.5 Model Testing

For the model testing, the pruned decision tree was used to make a prediction on the unseen, test set. This returned an accuracy of 83%.

7.3.6 Analysis of Decision Tree Results

Table 4 shows the accuracies obtained by the decision tree model through iterative model development. The model achieved the highest accuracy for the initial prediction on the training set; however, it is likely that the model was overfitting, (Brownlee, 2019). Through the iterations, the model did reduce in accuracy when the feature selection process oversimplified the model. After modifying the feature selection process, the model consistently achieved accuracies of 83% and 84%.

Table 4: Tuning Process for the Decision Tree models

Model	**Parameters**	**Accuracy (on validation set)**
Initial Model	Default parameters with all the features included	0.83
Iteration 1	Decision tree with Gini split	0.83
Iteration 2	Decision tree with information split	0.84
Iteration 3	Pruned decision tree	0.83

7.4 Random Forest

From the literature review, it was established that a random forest model would be one of the techniques employed for our problem since it was popular among researchers for both the healthcare domain and specifically for the prediction of obesity.

To build the initial random forest model, the package 'randomForest' was chosen and the random forest function was used to instantiate the model where the predictor variables (obesity_X_train) and target variable (obesity_Y_train) were specified (see line 316). The default parameters were used for initial model, where the number of trees, 'ntree' is 500 and the 'mtry' parameter is the square root of the total number of predictor variables. The model's hyperparameters were kept at the default in order to produce a baseline model that could be compared against the future tuned models.

The initial model was then evaluated on the training set (obesity_X_train) first using the predict function (line 320) with the predictions being assigned to the p1 variable. A confusion matrix was run using the confusionMatrix function, comparing the predicted values against expected values for the target variable, (see line 321). The decision to run the initial model on the training set first was done to better understand the model's performance and how it would change once unseen data was shown to it, this can help identify cases of the model underfitting or overfitting to the training data (Röhrich, 2020).

The model obtained an accuracy of 100%, on the training set, this is very high, however not entirely unexpected since the model has already seen this data but a result of 100% indicates that the model is overfitting which is often the case for tree-based models (Carvalho et al., 2018).

The initial model was then used to make predictions for the validation set (obesity_X_val), see line 324, to observe its performance on unseen data and to ensure it was able to generalise well and avoid the overfitting problem. Once again, a confusion matrix was computed against the actual and predicted values for the target variable. This time the initial model resulted in an accuracy of 95% on the validation set.

The accuracy scores indicate that the initial model is severely overfitting and is not generalising well to the unseen data therefore tuning the model is in order to combat the overfitting problem and ensure that the model can perform well on unseen data.

7.4.1 Iteration 1 – Feature Selection

To tune the model, the first technique that was applied was feature selection as it can improve the generalisation of the model via the removal of redundant features (Mozaffari, 2023).

To aide in the feature selection process a method called LVQ (Learning Vector Quantization) was used. Line 341 sets up a control structure for model training using 10-fold cross-validation that was repeated 3 times to improve the generalisation of the model (Brownlee, 2019).

The code in line 342 trains the model using the LVQ method with the target (NObeyesdad) and the predictor variables specified. The variable importance is then calculate using the varImp function (line 344) which calculates how much each feature contributes to the predictive performance of the model.

The variable importance is printed (line 346), and finally the feature importance is visualised (line 348) via a plot (see Figure 3). The goal is to identify which features are most important for predicting the target variable (Dubey, 2019).

Figure 3 shows the least important features for predicting the target variable are some of the transportation columns (MTRANSMotorbike, MTRANSBike) across every class. To improve the model performance, these columns are dropped (see line 350).

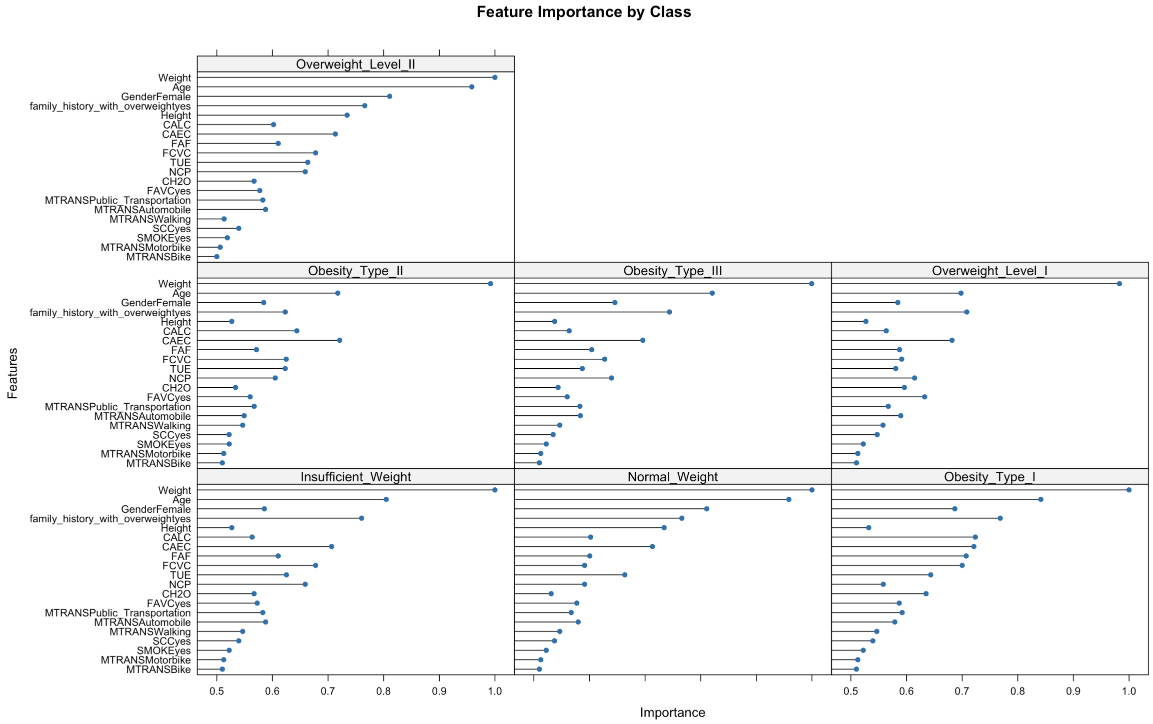

Figure 3: Importance of Features

The improved random forest model was built and trained on the modified training set (without the least important features) and was evaluated using both the modified training and validation set. This tuned model received an accuracy score of 95% on the validation set, indicating that further improvements could still be made.

7.4.2 Iteration 2 & 3 – Hyperparameter Tuning via GridSearch

The previous model was further tuned using the grid search technique to find the optimal values for the random forest hyperparameters (mtry and ntree).

The grid search technique was utilised as it can conduct an exhaustive search over a predefined set of hyperparameter values. It can help in identifying the combination of hyperparameters that result in the best performance on a chosen evaluation metric. This can lead to models that generalise well to new, unseen data.

The grid search was combined with cross-validation to obtain a more robust estimate of the model's performance. Cross-validation helps mitigate the risk of overfitting to a specific training-validation split and provides a more reliable evaluation of the model's generalisation ability (Brownlee, 2019).

The 'mtry' parameter refers to the number of variables that are randomly sampled as candidates at each split when growing a tree, it controls the size of the subsets of the predictor variables selected.

The 'ntree' parameter specifies the number of trees that the Random Forest algorithm will build. Each tree in the forest is constructed using a bootstrap sample of the data (sampling with replacement) and a random subset of features at each split. The idea behind using a forest of trees is to reduce overfitting and improve the generalisation performance of the model.

To perform the grid search to find the optimal value for 'mtry' the tuning parameters were set. The evaluation metric used to assess the model's performance was accuracy (see line 379). The train control object was created to specify the details of the cross-validation process. It used repeated cross-validation with 10 folds and 3 repeats this reduces the risk of overfitting. A grid of values for the 'mtry' parameter are specified from a range of 1 to 15 and the 'expand.grid' function generates all possible combinations. The grid search for the random forest model is performed using the tuning grid and control that were previously defined with the caret package (lines 380 to 384).

The results from the grid search are plotted and from Figure (4) we can see that the optimal value for the 'mtry' parameter is 11.

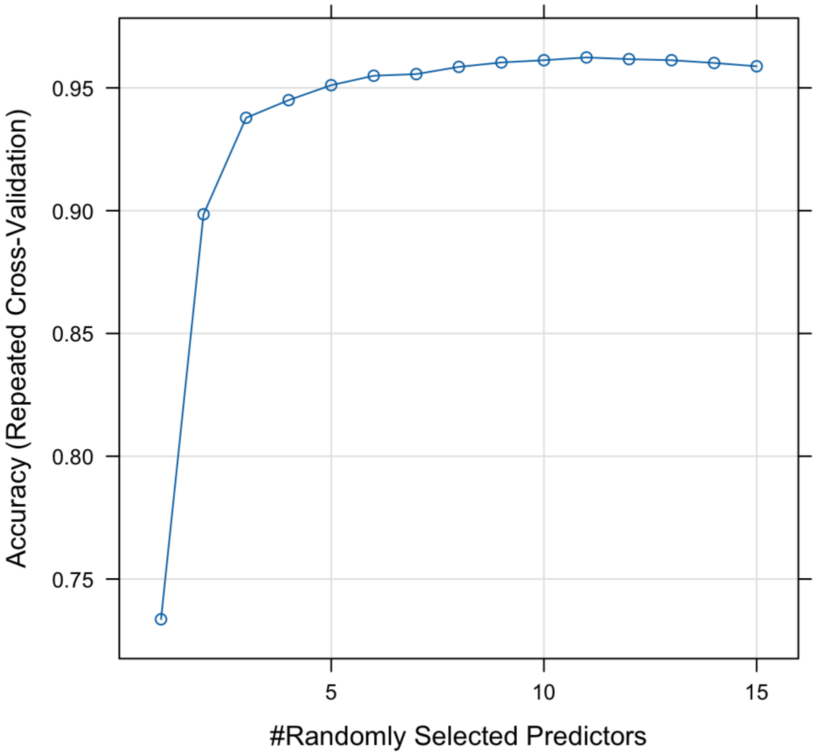

Figure 4: Grid search for 'mtry' hyperparameter

The results from the grid search for the optimal model are fitted onto the validation set for predictions where it obtained an accuracy of 97%. This is an increase from the previous iteration of the model showing that the hyperparameter tuning has indeed improved the performance of the model. This suggests that the model correctly predicted the class labels for 97% of the instances in the dataset.

To perform the grid search to find the optimal value for ‘ntree’ the control and metric used to assess the model’s performance were kept the same as the ‘mtry’ (see line 397).

This time, the tune grid specifies the number for the ‘mtry’ as 11 as this was the optimal value that was discovered from the previous model iteration (line 398).

A loop was then used to iterate over different values of the ’ntree’ parameter (1000, 1500, 2000, 2500) and within the loop the random forest model was trained on the training set (obesity_train) using the train function from the caret package. Each model was added to the modellist list, with the ’ntree’ value used as the key (lines 399 to 406).

The resamples function (line 409) was used to compare the performance of different models in the modellist based on resampling (cross-validation) results. A summary of the performance metrics for each model were then displayed and the optimal number of trees was identified as 2000 from its mean accuracy.

The model with the optimal number of trees (2000) was used to predict on the validation set and it obtained and accuracy of 97%. This did not change from the previous model and could suggest that the hyperparameter space should be increased across a larger range.

7.4.3 Analysis of Random Forest Results

Table 5 shows the iteration process for the random forest models and their respective accuracy scores on the validation set based on the changes that have been made.

For each iteration the accuracy value increases suggesting the models are improving in predicting the correct classes, however it stays the same for the third iteration indicating that further tuning may be required.

The model for iteration 3 was selected as the final model and was used evaluate the testing set (the truly unseen data) where it obtained an accuracy of 97%.

Table 5: Tuning Process for the Random Forest models

Model	**Changes made**	**Accuracy (on validation set)**
Initial Model	Default parameters with all the features included	0.95
Iteration 1	Feature selection – removed transportation columns as they had low importance	0.95
Iteration 2	Used grid search to find the best value for ‘mtry’ parameter. Best value for mtry = 11	0.97
Iteration 3	Used grid search to find the best value for ‘ntree’ parameter. Best value for ntree = 2000	0.97

8 Analysis of Results

To analyse the results, the following metrics have been chosen: accuracy, specificity and sensitivity.

The sensitivity refers to true positive predictions, i.e. predicting an individual with obesity to have obesity. This is an important metric for this application domain as it ensures positive predictions are not missed, (New York State, 1999).

Specificity refers to the correct negative predictions from all those in the negative class, (Kundu, 2022a). Due to this being a multiclass classification, the average specificity across multiple classes will provide more meaningful insights.

This analysis is split into two sections: prediction of obesity and prediction of overweight classes. For the metrics of specificity and sensitivity the average was computed across the obese and overweight classes.

8.1 Prediction of Obesity type classes

To predict obesity, the following classes have been included: Obesity_Type_I, Obesity_Type_II and Obesity_Type_III.

Table 6 shows that the random forest model performed best in terms of accuracy, which means 97% of its predictions were correct regardless of the class. However, through the literature review it has been shown that the sensitivity is the most vital metric within the obesity prediction domain. In terms of sensitivity, the random forest model again performed the best, which means it was able to correctly identify individuals with obesity as having obesity.

Table 6: Evaluation metrics for the final models on the testing set for the Obesity classes.

Model	Accuracy	Specificity	Sensitivity
DT	0.83	0.993	0.91
RF	0.97	0.998	0.99
KNN	0.81	0.98	0.91

8.2 Prediction of Overweight classes

To predict overweight individuals, the following classes have been included: Overweight_Level_I and Overweight_Level_II.

Table 7 shows that the random forest model performed best in terms of accuracy, which means 97% of its predictions were correct regardless of the class. However, as sensitivity is the most vital metric within the domain, the random forest model performed the best again. This shows that the random forest model was able to correctly predict overweight individuals.

Table 7: Evaluation metrics for the final models on the testing set for the Overweight classes.

Model	Accuracy	Specificity	Sensitivity
DT	0.83	0.94	0.86
RF	0.97	1	0.98
KNN	0.81	0.96	0.73

9 Conclusions

To conclude, various data mining techniques have been exploited in this report for obesity prediction. Our exploration of three distinct data mining techniques k-Nearest Neighbours (KNN), Decision Trees (DT), and Random Forests (RF), across multiple iterations have provided valuable insights into their performance dynamics.

KNN, with its simplicity and flexibility, exhibited robustness but was sensitive to the choice of the number of neighbours. Decision tree showcased interpretability and the ability to capture non-linear relationships, while random forest, as an ensemble technique, demonstrated resilience against overfitting and enhanced predictive accuracy.

Through feature selection processes applied for each classifier, attributes with least or most importance were identified. Feature selection was a reoccurring technique used throughout the literature and was required for our dataset due to its high dimensionality. When using LVQ, the attributes with least importance were MTRANSBike and MTRANSMotorbike. Similarly, when retrieving correlation between attributes, MTRANSPublic_Transportation has the highest correlation. Therefore, these attributes were removed.

LVQ also obtained the most important attributes, which were: weight, age, gender and family_history_with_overweight. Hence, these attributes were always included. This aligns with domain research which states these factors to be significant factors of obesity.

When predicting obesity, the random forest classifier performed the best in terms of accuracy (97%) and sensitivity (99%). Similarly, the random forest classifier performed best when predicting overweight with an accuracy of 97% and sensitivity of 98%. The sensitivity indicates the model's ability to correctly predict the positive classes, i.e. correctly predicting an individual with obesity or overweight. Therefore, the random forest classifier would be most suited for obesity prediction, amongst these three models and, could be used in real-world applications.

10 References

Ahsan, M.M. et al. (2021) ‘Effect of data scaling methods on machine learning algorithms and model performance’, 9(3), p. 52.

Ali, J. et al. (2012) ‘Random forests and decision trees’, Journal of Computer Science Issues (IJCSI), p. 272.

Atwell, C. (2022) Floating-Point Formats in the World of Machine Learning.

Brownlee, J. (2019) Overfitting and Underfitting With Machine Learning Algorithms. Available at: https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/ (Accessed: 10 December 2023).

Brownlee, J. (2021) An Introduction to Feature Selection. Available at: https://machinelearningmastery.com/an-introduction-to-feature-selection/ (Accessed: 10 December 2023).

Carvalho, J., Santos, J.P.V., Torres, R.T., Santarém, F. and Fonseca, C., 2018. Tree-Based Methods: Concepts, Uses and Limitations under the Framework of Resource Selection Models. Journal of Environmental Informatics, 32(2).

Dash, S. (2022) Decision Trees Explained — Entropy, Information Gain, Gini Index, CCP Pruning. Available at: https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c. (Accessed: 11 December 2023).

De-La-Hoz-Correa, E. et al. (2019a) ‘Obesity Level Estimation Software based on Decision Trees’, Journal of Computer Science, 15(1), pp. 67–77. Available at: https://doi.org/10.3844/jcssp.2019.67.77.

De-La-Hoz-Correa, E. et al. (2019b) ‘Obesity Level Estimation Software based on Decision Trees’, Journal of Computer Science, 15(1), pp. 67–77. Available at: https://doi.org/10.3844/jcssp.2019.67.77.

Dubey, A. (2018) Feature Selection Using Random forest. Available at: https://towardsdatascience.com/feature-selection-using-random-forest-26d7b747597f (Accessed: 11 December 2023).

Dugan, T.M. et al. (2015) ‘Machine Learning Techniques for Prediction of Early Childhood Obesity’, Applied Clinical Informatics, 06(03), pp. 506–520. Available at: https://doi.org/10.4338/ACI-2015-03-RA-0036.

Ferdowsy, F. et al. (2021) ‘A machine learning approach for obesity risk prediction’, Current Research in Behavioral Sciences, 2, p. 100053. Available at: https://doi.org/10.1016/j.crbeha.2021.100053.

Ghawi, R. and Pfeffer, J. (2019) ‘Efficient hyperparameter tuning with grid search for text categorization using kNN approach with BM25 similarity’, Open Computer Science, 9(1), pp. 160–180.

IBM (2023a) K-Nearest Neighbors Algorithm. Available at: https://www.ibm.com/topics/knn. (Accessed: 11 December 2023).

IBM (2023b) ‘What is a Decision Tree?’ Available at: https://www.ibm.com/topics/decision-trees. (Accessed: 11 December 2023).

IBM (2023c) What is random forest? Available at: https://www.ibm.com/topics/random-forest. (Accessed: 11 December 2023).

James, G. et al. (2013) An Introduction to Statistical Learning: with Applications in R.

Kalra, V. and Aggarwal, R. (2017) ‘Importance of Text Data Preprocessing & Implementation in RapidMiner’, ICITKM, 17, pp. 71–75.

Koh, C.K. and Tan, G. (2011) ‘Data Mining Applications in Healthcare’, Journal of Healthcare Information Management , 19(2).

Kouiroukidis, N. and Evangelidis, G. (2011) ‘The Effects of Dimensionality Curse in High Dimensional kNN Search’, in 2011 15th Panhellenic Conference on Informatics. IEEE, pp. 41–45. Available at: https://doi.org/10.1109/PCI.2011.45.

Kundu, R. (2022a) Confusion Matrix: How To Use It & Interpret Results [Examples]. Available at: https://www.v7labs.com/blog/confusion-matrix-guide (Accessed: 11 December 2023).

Kundu, R. (2022b) Precision vs. Recall: Differences, Use Cases & Evaluation. Available at: https://www.v7labs.com/blog/precision-vs-recall-guide (Accessed: 8 December 2023).

Larose, D.T. and Larose, C.D. (2014) Discovering knowledge in data: an introduction to data mining. John Wiley & Sons.

McAvaney, B.J. et al. (2001) ‘Model evaluation’, Climate Change 2001: The scientific basis, pp. 471–523.

Mitrani, A. (2019) ‘Evaluating Categorical Models II: Sensitivity and Specificity’. Available at: https://towardsdatascience.com/evaluating-categorical-models-ii-sensitivity-and-specificity-e181e573cff8 (Accessed: 7 December 2023).

Mozaffari, S. (2023) The Importance of Feature Selection and Feature Importance in Machine Learning. Available at: https://www.linkedin.com/pulse/importance-feature-selection-machine-learning-sadaf-mozaffari (Accessed: 11 December 2023).

New York State (1999) Disease Screening - Statistics Teaching Tools. Available at: https://www.health.ny.gov/diseases/chronic/discreen.htm. (Accessed: 11 December 2023).

Palechor, F.M. and Manotas, A. de la H. (2019) ‘Dataset for estimation of obesity levels based on eating habits and physical condition in individuals from Colombia, Peru and Mexico’, Data in Brief, 25, p. 104344. Available at: https://doi.org/10.1016/j.dib.2019.104344.

Reddy, G.C. (2021) Healthcare Domain Knowledge. Available at: https://www.gcreddy.com/2021/08/healthcare-domain-knowledge.html (Accessed: 5 December 2023)

Röhrich, G. (2020) Training, Validating and Testing — Successfully Comparing Model Performances. Available at: https://towardsdatascience.com/train-test-split-c3eed34f763b (Accessed: 11 December 2023).

Saxena, S. (2023) What are Categorical Data Encoding Methods — Binary Encoding.

Seifert, J.W. (2004) ‘Data mining: An overview’, National security issues, pp. 201–217.

Therneau, T., Atkinson, B. and Ripley, B. (2023) Package ‘rpart’. Available at: chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://cran.r-project.org/web/packages/rpart/rpart.pdf (Accessed: 10 December 2023).

Timpson, C. (2023) Obesity Profile: short statistical commentary May 2023 - GOV.UK. Available at: https://www.gov.uk/government/statistics/obesity-profile-update-may-2023/obesity-profile-short-statistical-commentary-may-2023 (Accessed: 4 December 2023).

WHO (2021) Obesity and overweight.
Wright, S.M. and Aronne, L.J. (2012) ‘Causes of obesity’, Abdominal Radiology, 37(5), pp. 730–732.

Available at: https://doi.org/10.1007/s00261-012-9862-x.

Xu, Y. and Goodacre, R. (2018) ‘On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning’, Journal of analysis and testing, 2(3), pp. 249–262.

Ying, X. (2019) ‘An Overview of Overfitting and its Solutions’, Journal of Physics: Conference Series, 1168, p. 022022. Available at: https://doi.org/10.1088/1742-6596/1168/2/022022.

Zhang, S. et al. (2009) ‘Comparing data mining methods with logistic regression in childhood obesity prediction’, Information Systems Frontiers, 11(4), pp. 449–460. Available at: https://doi.org/10.1007/s10796-009-9157-0.

11 Appendices

11.1 Appendix A – Analysis of Literature

Research by Koh and Tan (2011) takes a general approach of investigating data mining applications in healthcare. As detailed by Koh and Tan (2011) data mining can analyse large amounts of healthcare data from which data-led decisions can be taken. Such decisions can improve healthcare management, resource allocation and operating efficiencies: emphasising the usefulness of data mining within healthcare, (Koh and Tan, 2011).

In comparison, Zhang et al. (2009) specifically investigated whether young children can be at a risk of obesity. This was achieved by comparing logistic regression to six other data mining models (decision trees, association rules, neural networks, naïve bayes, Bayesian networks and support vector machines). The dataset used was collected by health visitors in England and comprised of 16,653 records and 53 attributes; data cleaning techniques were used to discard 'abnormal' records, (Zhang et al., 2009). When predicting overweight for 8 months old, SVMs provided a low accuracy yet high sensitivity; sensitivity refers to the model's ability of predicting true positives per category, (Mitrani, 2019). Similarly, when predicting obesity for 2-year-olds, the SVM model again achieved the highest sensitivity, followed by naïve bayes. A higher rate of sensitivity is most important for overweight prediction as it allows overweight patients to be identified, (Zhang et al., 2009).

Likewise, Dugan et al. (2015) used data mining techniques for prediction of childhood obesity. The dataset used by Dugan et al. (2015) was from a paediatric clinical system, and consisted of 7519 records and 168 attributes, after data pre-processing was complete. Dugan et al. (2015) used the following six models: random tree, random forest, naïve bayes, bayes net, decision tree (ID3 algorithm) and J48 (Java extension of ID3 algorithm). After several iterations of model tuning, the decision tree (ID3 algorithm) obtained the highest sensitivity, followed by the random tree. However, a key finding was that only two models improved between iterations after 'noisy' attributes were removed, (Dugan et al., 2015). This also allowed (Dugan et al., 2015) to identify the most influential attribute within their tree; this was a categorical attribute, 'OVERWEIGHT_BEFORE_24mo'.

De-La-Hoz-Correa et al. (2019) explored obesity prediction of 18 to 25 year-olds with the following models: decision trees, naïve bayes and logistic regression. The dataset used consisted of 712 records and 18 attributes. From the chosen models, decision tress obtained the highest precision and recall (i.e. true positive rate) values of 97% and 97.8%. Precision refers to the proportion of correct positive class predictions, whilst recall refers to the proportion of actual positive class samples identified by the model, (Kundu, 2022). When predicting obesity, this allows obese patients to be identified and appropriate interventions to be put in place, (De-La-Hoz-Correa et al., 2019).

Similar to De-La-Hoz-Correa et al. (2019), Ferdowsy et al. (2021) investigated adult obesity prediction, but with a larger sample size of 1100 records and 28 attributes. Data pre-processing techniques were implemented to remove noisy attributes and outliers. Ferdowsy et al. (2021) also used multiple models from which SVM and naïve bayes obtained the highest sensitivities of 100%. Naïve bayes also obtained a high precision and recall rate of 86%, which allows patients with obesity to be identified.

11.2 Appendix B – Exploratory Data Analysis

The boxplot in Figure 5 shows the distribution of age values through the dataset. Although the boxplot shows outliers, these values are not outliers as the age range of the dataset is 14 to 61 years.

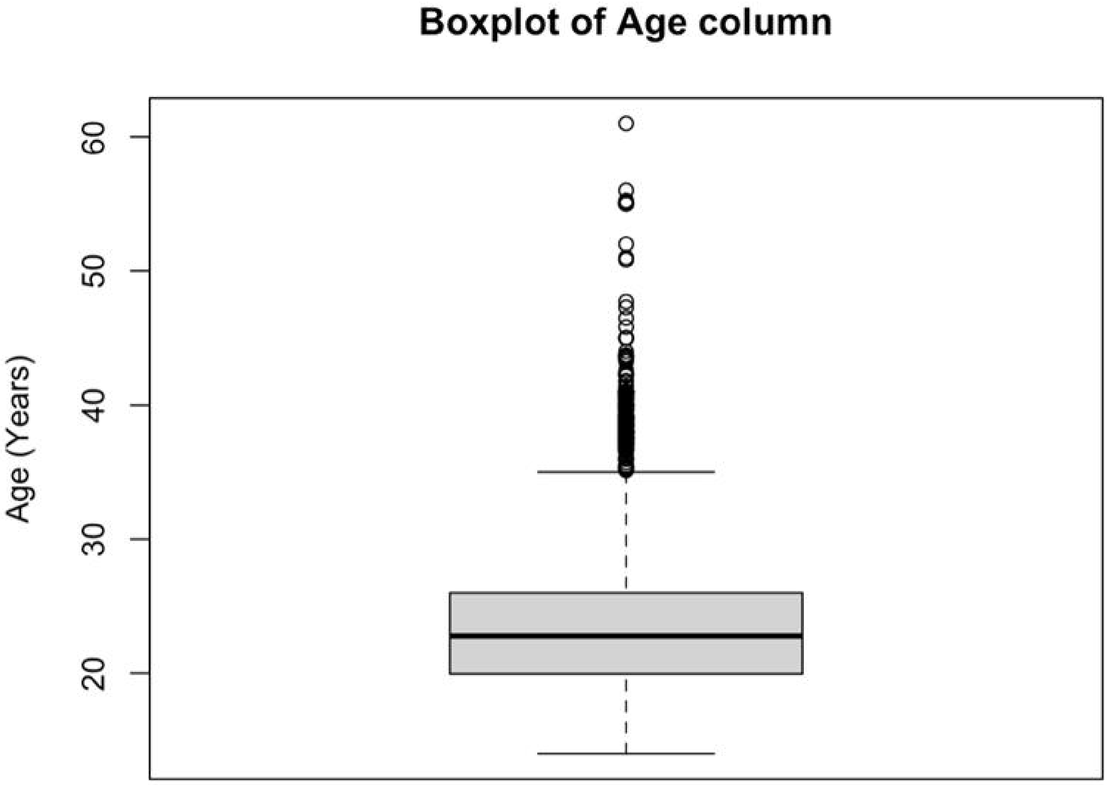

Figure 5: Boxplot for the Age column

The boxplot in Figure 6 shows the distribution of height values through the dataset. This shows an interquartile range of 1.61 to 1.75, indicating the spread of the middle half of the data.

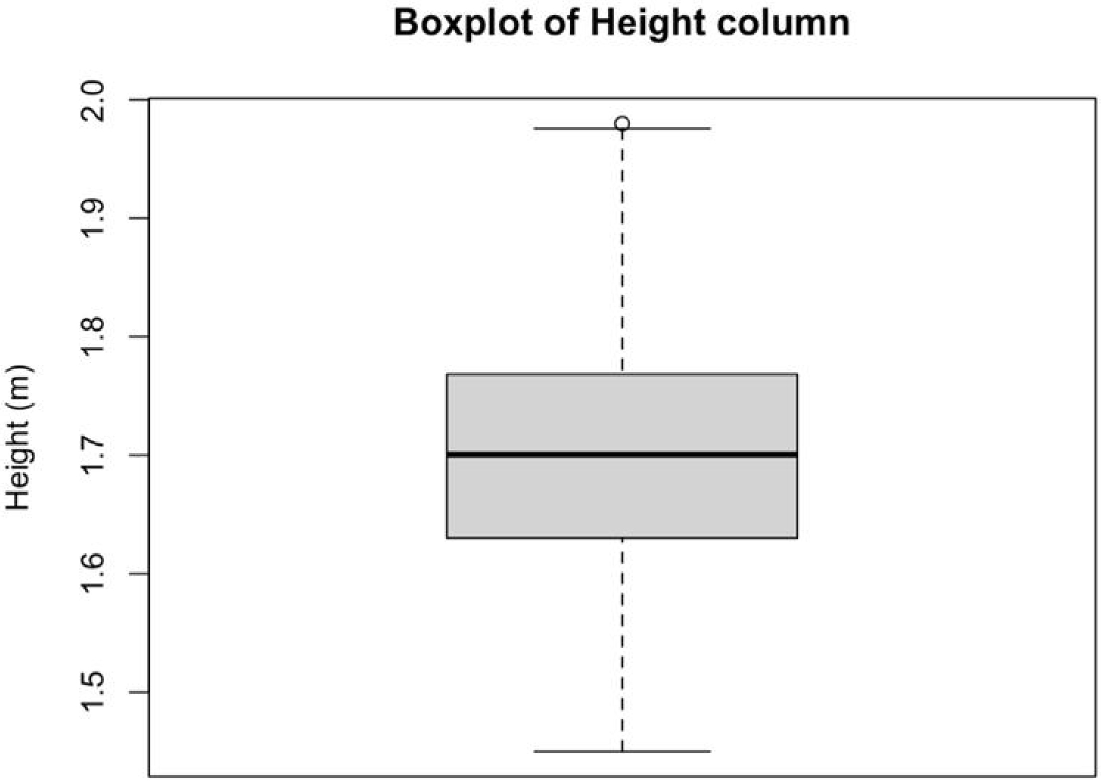

Figure 6: Boxplot for the Height column

The boxplot in Figure 7 shows the distribution of weight values through the dataset. The boxplot does show an outlier; this is not an unexpected value due to the problem domain.

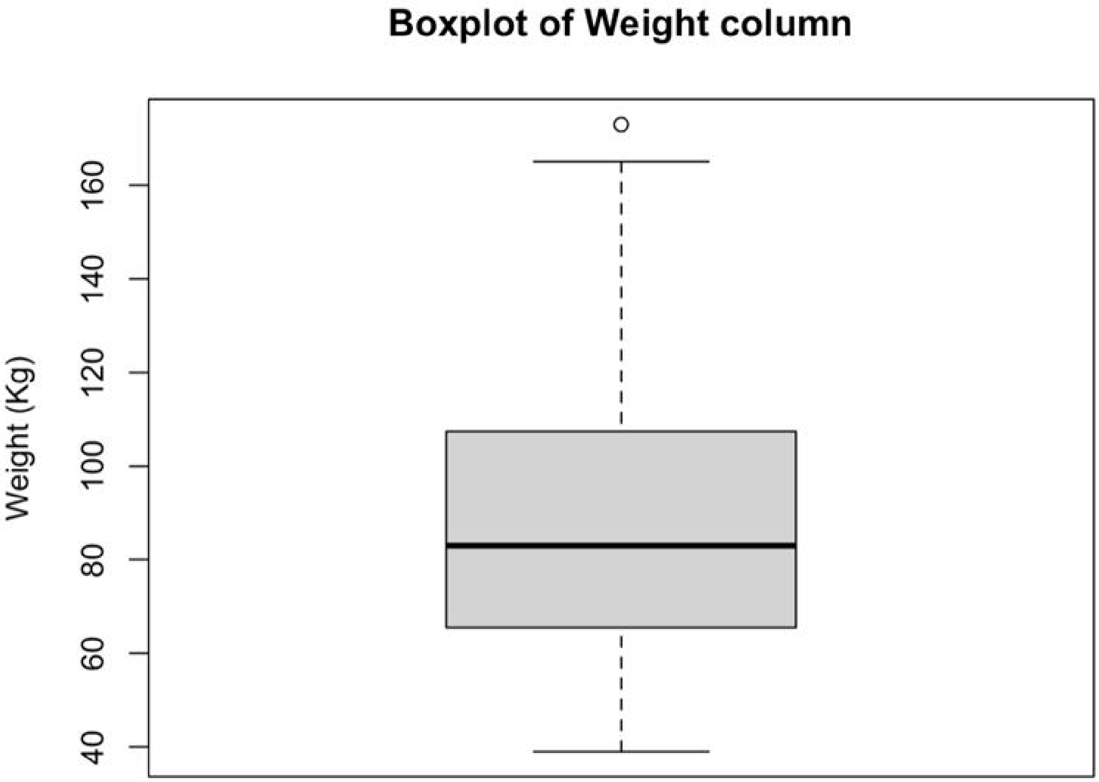

Figure 7: Boxplot for the Weight column