Predicting Obesity

Data 2010 Project

Lucas Berzuk, Ethan Robson, Hugo Ng, Rodell Salonga

April 5, 2024

Introduction

Abstract

Obesity is a chronic disease that is defined by having an excessive amount of body fat. It increases the risk of other diseases and health problems including heart disease, high blood pressure, diabetes and certain cancers. In recent years, many first world countries are experiencing an epidemic of obesity, including Canada. In 2003, it was reported that Canadian adults had a 30% obesity rate, which was an increase from the 21% reported in 2003 [2]. In order to reverse this trend, it is necessary to be able to diagnose obesity early on for those at risk so that they can be provided with preventative therapies to decrease body fat percentage. To be able to diagnose early obesity we must first have a method of measuring levels of obesity.

There are many different tools for measuring obesity levels, some of which include body mass index, waist-to-hip ratio, skin fold thickness, and other more costly and intense body fat measuring procedures. The most common measure is body mass index (BMI) which is calculated using an individuals weight (in kilograms) and height (in metres), with the following formula: $BMI = \frac{weight}{height^2}$. Unfortunately, in many cases the height and weight of individuals are not available to be used to determine BMI for diagnosing obesity level.

In this paper, different statistical models were created that attempt to predict obesity levels based on variables other than height and weight. These models were trained and tested using data obtained from a study conducted at Coast University in Colombia [1]. The data includes 17 different attributes regarding an individual's lifestyle choices and patterns. This study will be explained in greater depth in the following "Data Analysis" section.

Various regression models were built using BMI as the target variable and the rest of the variables in the data set as the predictor variables to determine which ones are the most valuable in predicting obesity. In doing so, an individual's BMI is able to be determined, which helps predict the risk of obesity using information about their lifestyle. The regression methods used to create these models were Linear, Spline, Lasso, and Ridge regression. Root mean squared error (RMSE) was calculated for each model as a means of comparing which model had the greatest prediction ability. Unfortunately, the results from the regression models was not found to be adequate in accurately determining an individual's BMI, as the lowest RMSE value that could be obtained was *7.929959* using the *SMOKE* variable with Linear regression.

As an alternative to the regression models, classification models were built using the same data. These models perform a binary classification predicting whether or not an individual is obese. Based on the CDC's definition of obese, an individual with a BMI level greater than 30 was considered obese for the classification models [3]. Filtering variable selection was performed to determine the features that would be used in training the models. The classification methods used to create these models include logistic regression, nearest neighbors, and decisions trees. For each of these models, techniques were then applied in attempts in improve the models ability to predict obesity. The resulting models are then analyzed by classification metrics such as accuracy and the Receiver Operating Characteristic (ROC) curve to determine the best model. These classification models determined that the variable *family_history_with_overweight* has a significant role in predicting the obesity of an individual.

Data Analysis

The data used as the basis for this paper comes from a study performed in Mexico, Peru and Colombia. It has 17 attributes and 2111 data points. The 17 attributes comprise 4 different variable types: 7 numerical, 6 categorical, and 4 binary. This data consists of an individual's behavioural patterns such as eating habits and physical condition, as well as their level of obesity. It is important to note that up to 77% of this data has been synthetically generated because of a greatly unbalanced number of samples between the different obesity categories in the sample data. The balancing was performed using the SMOTE filter with the Weka tool. The purpose was to create data points within the obesity categories with lower counts. This lead to an overall data set with uniform counts for the different obesity levels. The data was collected through a voluntary online survey that had 16 questions with a variety of responses. It was accessible for 30 days for users to complete. The researchers of this study calculated Body Mass Index using the equation $BMI = \frac{weight}{height^2}$ to determine the BMI of every respondent. The BMI values were then placed into different obesity level categories based on the World Health Organization and Mexican Normativity.

In our regression and classification models we use the same training and testing data. To do this we split our data up so that 70% of the original data will be used as the training data and the other 30% will be used to test our models. We first check our data set for missing values and find that there are no missing entries. We then convert all categorical variables to factored variables with levels so that we can involve them in our models. BMI levels are not present in the given dataset so we directly use height and weight with the above formula to add BMI values for each individual to our dataset.

Dataset Variables

The variables from the study that were considered in our models were as follows:

Table 1: Dataset Variables

Variable	Variable Name	Variable Type	Variable Response
Smoke	SMOKE	Binary	Yes / No
Number of main meals	NCP	Categorical	Between 1 and 2 / 3 / More than 3
Gender	Gender	Binary	Male / Female
Time using electronics	TUE	Numerical	0-2 hours / 3-5 hours / More than 5 hours daily
Consumption of water	CH20	Numerical	Less than a liter / Between 1 and 2 liters / More than 2 liters daily
Transportation used	MTRANS	Categorical	Automobile / Motorbike / Bike / Public transportation / Walking
Calorie consumption monitoring	SCC	Binary	Yes / No
Freq of physical activity	FAF	Numerical	I do not have / 1 or 2 days / 2 or 4 days / 4 or 5 days a week
Freq of Consumption of alcohol	CALC	Categorical	I do not drink / Sometimes / Frequently / Always
Freq consumption of high caloric food	FAVC	Binary	Yes / No
Freq consumption of vegetables	FCVC	Categorical	Never / Sometimes / Always
Age	Age	Numerical	Numeric value for age
Consumption of food between meals	CAEC	Categorical	No / Sometimes / Frequently / Always
History of overweight	family_history_with_overweight	Binary	Yes / No

Methods

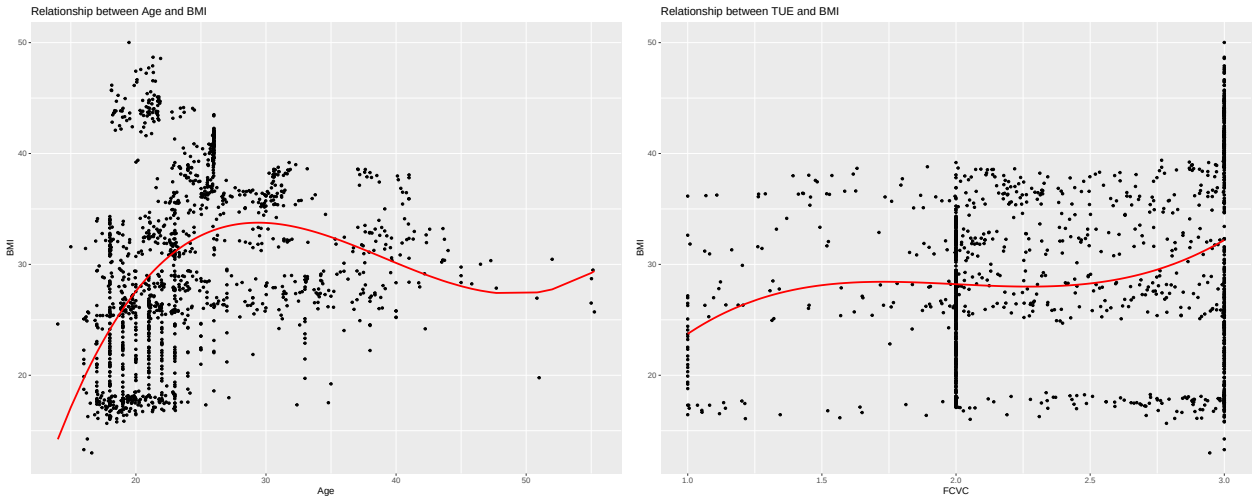
Regression

To ensure that the most effective regression method was used, many different methods were utilized and subsequently compared using their RMSE values. These methods were Linear, Spline, Lasso, and Ridge regression.

Before proceeding with creating the regression models, it was important to check the correlation between the variables and *BMI*. These investigations lead to discovering relatively high correlation between *BMI* and *vegetable consumption* (FCVC), as well as between *BMI* and *age*. A table of these findings can be found below along with graphs of the two aforementioned variables.

Table 2: Correlation

Variable	Age	Vegetable Consumption	Meals Quantity	Water Consumption	Physical Activity	Screen Time
Correlation	0.4006	0.2605	-0.0526	0.1593	-0.1686	-0.0752



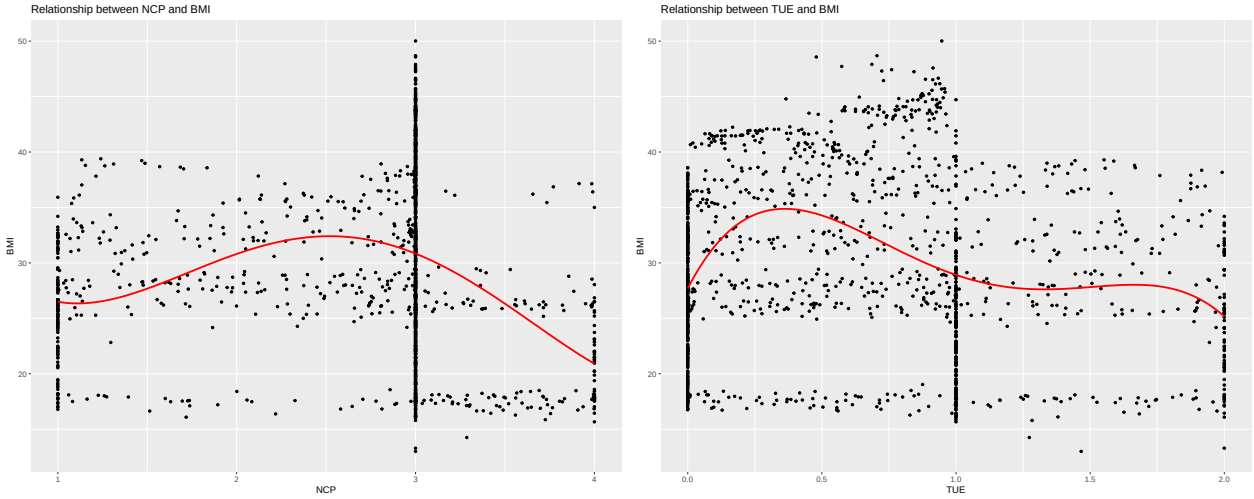
Next, evidence of multicollinearity was checked for. A correlation matrix was created and checked to see if there was any values close enough to 1 or -1 to be considered evidence of multicollinearity. The results showed that there were no values close enough to consider the dataset as having evidence of multicollinearity, as can be found below.

```
##                Age     Height     Weight       FCVC        NCP        CH2O
## Age    1.00000000 0.02080862 0.21793935 0.02225751 0.04393122 0.034840229
## Height 0.02080862 1.00000000 0.47117782 0.04608912 0.25248988 0.204148356
## Weight 0.21793935 0.47117782 1.00000000 0.19534515 0.11347087 0.190483184
## FCVC   0.02225751 0.04608912 0.19534515 1.00000000 0.05008597 0.052498525
## NCP    0.04393122 0.25248988 0.11347087 0.05008597 1.00000000 0.042227619
## CH2O   0.03484023 0.20414836 0.19048318 0.05249853 0.04222762 1.000000000
## FAF    0.16415271 0.30640582 0.05650405 0.01100181 0.11328858 0.161417486
## TUE    0.30123390 0.05600413 0.06577166 0.10525079 0.04894118 0.009913777
##                FAF         TUE
## Age    0.16415271 0.301233897
## Height 0.30640582 0.056004127
## Weight 0.05650405 0.065771664
## FCVC   0.01100181 0.105250787
## NCP    0.11328858 0.048941181
## CH2O   0.16141749 0.009913777
## FAF    1.00000000 0.068454890
## TUE    0.06845489 1.000000000
```

In order to determine which variables in the dataset are the most valuable in predicting *BMI*, linear regression models were created using each variable available, with the exception of height and weight, which were used directly for calculating BMI, as well as obesity level category, which was categorized directly from *BMI*. Prior to making the models, the categorical variables in the dataset were factorized to allow them to be used in the model. The following list presents the 14 variables that were considered alongside their respective linear regression model's RMSE values in order from least to greatest:

```
## Smoke = 7.929959
## Number of main meals = 7.932018
## Gender = 7.939414
## Time using electronics = 7.973657
## Consumption of water = 8.065398
## Transportation used = 8.07459
## Calorie consumption monitoring = 8.106003
## Freq of physical activity = 8.109581
## Consumption of alcohol = 8.130043
## Freq consumption of high caloric food = 8.134233
## Freq consumption of vegetables = 8.167971
## Age = 8.178174
## Consumption of food between meals = 8.807099
## History of overweight = 9.08678
```

To create a visual reference for how the regression models were performing, *number of main meals* (NCP) and *time using electronics* (TUE) were graphed because they were the numerical variables that lead to the lowest RMSE values (under 8). These graphs can be found below.



In an attempt to improve RMSE values further, spline regression was used on the two best performing numerical variables mentioned above. However, the RMSE values were found to be worse than those found with linear regression. The RMSE values after spline regression were as follows:

```
## Number of main meals = 8.540398
## Time using electronics = 8.458963
```

Next, Lasso regression was attempted to again improve RMSE values. Lasso regression was performed using the four variables that had RMSE under 8. With a lamba value of 1.2, the RMSE was:

```
## RMSE = 7.968656
```

Relative to the previous RMSE values found, this value was good but not lower than any of the RMSE values found using Linear regression. As a further attempt using Lasso regression, all of the variables were used with a lamba value of 4. The RMSE with this approach was:

```
## RMSE = 7.968656
```

The RMSE produced was identical. As a final attempt at Lasso regression, *gender* and *smoke* were attempted with a lambda value of 0.27. The RMSE was:

```
## RMSE = 7.96817
```

This RMSE value is only slightly lower than the previous 2 Lasso regression attempts, so it was decided that Lasso regression would not be the best way to model the data.

The final regression method attempted was Ridge regression. Ridge regression gave similar RMSE values to Lasso, but always slightly higher regardless of the variables used in the model.

To determine the merit of RMSE values of approximately 8, the range of BMI values in the dataset were considered:

```
## Minimum = 12.99868
```

```
## Maximum = 50.01402
```

Given this range, RMSE values of 8 are not unusable, however they are not good enough to be confident in using these models unless improvements are made.

Results

After performing all of the various types of regression methods, the lowest RMSE values that were able to be obtained was always around 8. None of the methods attempted were able to narrow the model even further in order to get a more accurate prediction of BMI.

Linear regression gave RMSE values in the range of *7.929959* to *9.08678*. Spline regression produced worse results, with RMSE values of *8.540398* and *8.488051*. Lasso regression was the most consistent in producing RMSE values under 8, regardless of which variables were used to build the model. Across three Lasso regression tests, RMSE values of *7.96817*, *7.968656* , and *7.968656* were obtained. Finally, the models developed using ridge regression (whose analyses are not included in this paper for brevity) produced RMSE values of *8.034446* and *8.793966*.

A possible reason that the regression tests were not as successful as hoped is because of the synthetically generated data in the dataset. The generated data may have lead to less ideal, or possibly inaccurate inputs which if so could hinder the ability of the regression models from accurately predicting BMI. Although the data generated was useful for filling in noticeable gaps, it has most likely lead to possible skew and/or bias in the dataset.

After performing these various regression tests, the best method for obtaining the lowest RMSE value was found to be Linear regression. The variables that resulted in the lowest RMSE values when used with Linear regression to predict BMI were *SMOKE*, *NCP*, *Gender* and *TUE*, with *SMOKE* being the one that resulted in the lowest RMSE value of *7.929959*.

Classification

In addition to predicting BMI levels, determining whether or not an individual is obese would be a powerful tool in health care to diagnose individuals early and prevent further disease. The focus of this section will be on developing classification models to determine whether or not an individual is obese based on various variables of our dataset.

Pre-processing

The BMI values have been calculated for each individual and will be used to determine the true class values of if an individual is obese or not. Based on the CDC's definition of obese, an individual with a BMI level greater than 30 is considered obese [3]. These true class values are required in our dataset for the developing and testing of our models. Therefore, before proceeding further, the following variable was added to the dataset:

$$Y = \begin{cases} 0, & BMI \leq 30 \\ 1, & BMI > 30 \end{cases}$$

The variables *BMI*, *Height*, *Weight*, and *NObeyesdad* were then removed from the training data. This is essential to the purpose behind building models as the values of the target variable Y is directly calculated from BMI, which is a result of Height and Weight and produces the resulting NObeyesdad categories. Therefore, because the objective is to discover relationships between obesity and the other variables present in the dataset, these variables must not be considered so that an obesity prediction can be provided when height and weight are not available.

When developing a model, it is important to initially create the model assuming that the training data comprises of the population. Therefore, only the training data is analyzed for feature selection. For feature selection in the classification models, filtering methods were performed. Filtering methods involve looking at the each feature of the dataset individually without considering its relationship with other covariates. Although this may overlook relationships between covariates such as multicollinearity, it is an effective and simple method. For each feature, the appropriate statistical test was performed to determine its relationship with the target variable Y.

For the categorical variables, **Pearson's Chi-Squared Test of Independence** was utilized. The Chi-Squared test is designed to analyze the relationship between categorical variables. It is called a "goodness of fit" test because it measures how likely the observed distribution of values fits the distribution under the null hypothesis, which assumes the two variables are independent.

Table 3: P-values for Chi-Squared Test

Variable	Gender	FHWO	FAVC	CAEC	SMOKE	SCC	CALC	MTRANS
p-value	0.875	4.03e-56	6.13e-27	2.1e-45	0.215	5.35e-12	2.14e-09	1.56e-06

For the remaining variables, which are numerical, a **Two-sided T-test** was used. The Two-sided T-test is a statistic test used for evaluating whether the means of two populations differ or not. Here the two populations are $Y = 0$, Non-obese individuals, and $Y = 1$, Obese individuals. For each numerical feature it was tested whether their mean values within the two groups are significantly different.

Table 4: P-values for T-test

Variable	Age	FCVC	NCP	CH2O	FAF	TUE
p-value	6.65e-20	8.33e-11	0.0249	0.000231	6.54e-10	0.0011

From the Table 3, it can be observed that all but two categorical variables have a resulting p-value less than 0.01 from the Chi-Squared tests. Therefore the null hypotheses fails to be rejected for both *Gender* and *SMOKE* at a 1% significance level. There is insufficient evidence that obesity is dependent of *Gender* and also *SMOKE*. From the results in Table 4, it can be observed that only *NCP* has a p-value greater than 0.01. Therefore, for all numerical variables but *NCP*, the null hypothesis is rejected at a 1% significance level. In the *NCP* test, the null hypotheses fails to be rejected and it is concluded that we have insufficient evidence that the mean *NCP* differs between the obese and non-obese groups. From these results, the following three variables *Gender*, *SMOKE*, and *NCP* will not be included in the development of the next models.

Now that the features to build our model have been selected, they will be used to predict the target class, which is the binary variable Y indicating if an individual is obese. There are many different classification models, but in this report, the focus will be on logistic regression, nearest neighbors, and decision trees.

Logistic Regression

Logistic regression is a regression model that can be turned into a classification model. This is done through the logit function which takes probabilities between 0 and 1 and transforms them to any real number. The result of this is a model that when given values of covariates, it produces a probability that can be compared to a threshold to determine what binary class that observation is to be labelled. This is particularly useful for the desired model as the target variable Y is binary. Therefore, a logistic model could be used to predict the value of Y, that is, predict whether or not that individual is obese given the values of covariates. However, Logistic regression uses the regression equation and therefore requires assumptions about the data to be satisfied. These assumptions include : linearity of the logit, independence of errors, absence of multicollinearity, lack of overdispersion, and adequacy of the model. For simplicity these properties were assumed to be satisfied for the dataset.

The first classification model created was a logistic model using all features of our data set. It resulted in an accuracy of 0.755 when tested on the testing data. To improve the model, regularized regression through Lasso regression was explored. However, since the dataset does not have a large number of features compared to the 2111 total number of observations, and as noted in the regression section of this report there are no signs of multicollinearity, there was not an expectation to see a big improvement in the previous results. This is exactly what was found when creating a Lasso Logistic regression model. The accuracy of the model did not differ from our original Logistic model. Since Lasso can be used for variable selection we examine the coefficients of all the features created by this model. It was found that the category 'no' from the *CALC* variable and the category 'Bike' from the *MTRANS* variable are the two coefficients that have gone to zero. This suggests that these two values of these covariates do not have a significant impact on the expected value of the outcome variable Y. It was also observed that the category 'yes' from the variable *family_history_with_overweight* has the largest coefficient absolute value, indicating that it significantly impacts the expected value of indicating obesity.

Nearest Neighbors

While some classification models require assumptions, other models such as Nearest Neighbors do not make any assumptions about the structure of the data. Nearest Neighbors is a simple model that involves determining the class of an observation based on the class of the observations that surround it. Typically, the Euclidean distance is the measure used to calculate the distance between two observations base on values of their features. Since distance is calculated between the covariate values, it becomes clear that the covariates must be numeric. In the training dataset, there are 8 categorical variables that we will exclude them from this model. Therefore only the 5 numerical features were selected: Those variables were *AGE*, *FCVC*, *CH2O*, *FAF*, and *TUE*. However, there are techniques to convert categorical variables to numeric while keeping there relationship consistent and this should be explored in future analysis for improving these models. The class of a given observation is then determined by considering what the majority class is between the K-number of observations that are closest. K is a hyper-parameter of the model that determines how many nearest neighbors to consider when classifying a given observation.

Determining the value of K is specific to the data at hand and there is not a single value of K that will work best all of the time. Hence, multiple models were calculated and compared to determine the optimal value of K. To achieve this, multiple models were created using the following values of K, $K = \{1, 3, ..., 101\}$. It is important to note that only odd values for K were considered. This is because we have a binary classification, so if K were even then the majority class between its K nearest neighbors could result in a tie and directly choosing a class for that observation would not be possible. To compare the models numerically for each of the different K values, each one was evaluated by considering appropriate metrics such as Accuracy and Area Under the Curve (AUC).

Decision Tree

Nearest Neighbors is flexible as there are not assumptions, but it is not the most interpretable as it is difficult to tell which features are the most important in determining the resulting class. Like Nearest Neighbors, Decision trees are another classification model that are flexible and do not make assumptions about the data. The main advantage of decision trees is their ease of interpretation. Decision trees are known as universal approximators because they are easy to use when trying to create a model with perfect classification. However, it is important to consider and evaluate the bias-variance trade-off when considering the size and depth of the tree to avoid a perfect classification model that overfits the data.

All of the selected categorical and numerical variables were used in the development of the decision tree. From this tree it can be seen that the root node consists of the variable *family_history_with_overweight*. It was assumed that the `ctree()` function from the 'party' package selects the predicate at each node that maximizes the drop in entropy. This drop in entropy can be quantified by the *Information Gain* of a predicate. This means that the predicate for 'if an individual has history of overweight family' maximizes the information gain out of all possible predicates. Since this variable is binary (yes/no response), it means that this feature does the best job at segregating individuals so that the entropy of the child nodes is minimized based on the classification of obese and non-obese.

Because large trees can lead to overfitting and small trees can lead to underfitting, it is sometimes difficult to manually determine the best depth of the tree to minimize these risks while balancing the bias and variance. One strategy that addresses this issue is bagging. Bagging on decision trees involves creating bootstrap samples, that is sampling from the training data with replacement, and then building trees for each bootstrap sample. **Random Forests** are a classification model that utilizes bagging combined with random subspaces to improve the accuracy of decision trees. For each iteration, a random selection of the features is performed so that each tree is not highly correlated. Random forests then combine characteristics of the resulting trees to produce a final model.

In an attempt to improve the original decision tree's performance, a Random Forest model using our training dataset was created. Table 5 and Table 6 display the resulting confusion matrices when we applied these two models to our training data.

Table 5: Decision Tree Confusion Matrix

	0	1
0	273	58
1	48	255

Table 6: Random Forest Confusion Matrix

	0	1
0	296	35
1	42	261

Results

There are various numerical measures of classification models including accuracy, precision, and recall. When deciding on the most effective measure it is important to consider if our dataset is balanced. Since the target variable Y is divided into two classes of obese and non-obese, which have proportions 0.45 and 0.55, respectively, the binary class of obesity has a relatively balanced population. Therefore accuracy is an appropriate measure of the classification models that were developed. In the following analysis, the following four models that were developed will be compared: the original Logistic model, the nearest neighbors model

with the optimal K value, the basic decision tree, and the random forest model. Table 7 is a table containing the accuracy results from applying each model to the test data.

Table 7: Accuracy of Models

Logistic Regression	Nearest Neighbors	Decision tree	Random Forest
0.756	0.765	0.833	0.879

There are also other visual measures to evaluate the performance of classification models such as the Receiver Operating Characteristic (ROC) curve and the Precision-Recall curve. Since the data is balanced, a plot of the ROC curve for the various models was created to compare their effectiveness. The ROC curve plots the true positive rate against the false positive rate for values of t between 0 and 1, where t is the probability threshold that determines the outcome class for that individual. When looking at the resulting line for a model, the closer the line goes towards the top left corner of the unit square, the better the model. As a result of this, the models' performances can quantified by the Area Under the Curve (AUC) which directly calculates the area under the line within the unit square. An AUC value of 1 would be the result of a perfect classification model, while values from 0.8-0.9 represent an excellent model. An AUC value of a model that is at least greater than 0.5 is desired, otherwise it suggests that a better performance could be achieved by swapping the binary classes within the model. The following figure displays the ROC curve for all four of the models.

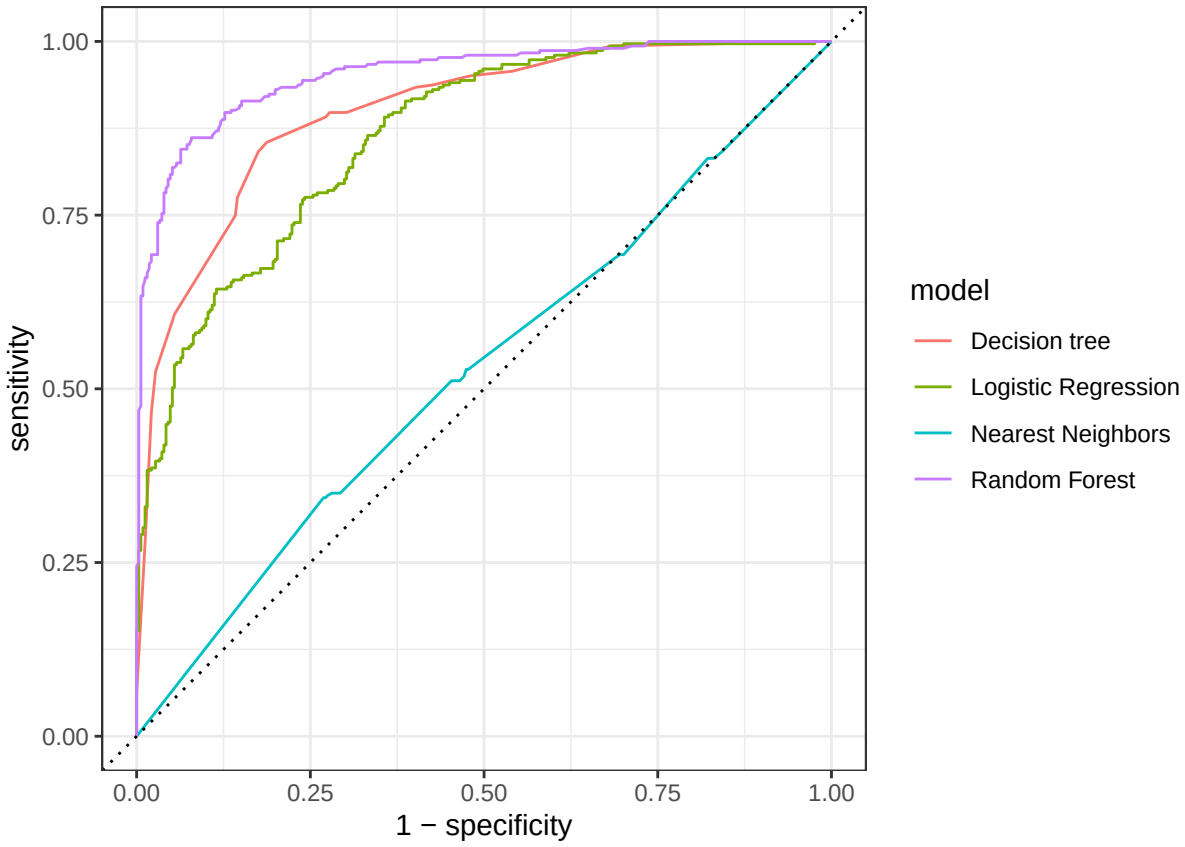


Table 8: Area Under the Curve for ROC

Model	Decision tree	Logistic Regression	Nearest Neighbors	Random Forest
AUC	0.899	0.863	0.527	0.952

From the plot, it can be observed that the nearest neighbor model had the worst performance with an AUC of *0.527*. This is insightful as the nearest neighbor classification was trained only using the numerical features, which suggests that the categorical variables play a strong role in predicting obesity. The logistic regression model involving all features had an AUC of *0.863*, performing much better than nearest neighbors but not nearly as strong as the decision tree and random forest models, which had AUC values of *0.899* and *0.952*, respectively. The weaker performance of the logistic model may be a result of certain assumptions that were not satisfied, such as the relationship between the features and the log-odds of the target variable being non-linear. The ROC curve and AUC values make it clear that the random forest classification model has the best capability of predicting obesity in individuals.

In the variable selection for the classification models, it was determined through statistical testing that obesity is likely not dependent on the variables *Gender*, *SMOKE*, and *NCP*. Furthermore, from developing these classification models, it was discovered that the variable *family_history_with_overweight* has a significant role in predicting the obesity of an individual, observed from the logistic regression coefficients and from the chosen root predicate of the decision tree.

Conclusion

In conclusion, this report explored possible relationships between obesity and different characteristics/behaviours of individuals. Using regression, a model to predict BMI levels directly was searched for. The regression methods used were linear regression, spline regression, lasso regression, and ridge regression. Out of the regression methods used, linear regression produced the best RMSE values, with the best one using the variable *SMOKE* with an RMSE value of *7.929959*. However, since the error in the models were not significantly reduced in a meaningful way, we conclude that the available variables may not be the best for predicting the BMI of individuals and that other variables should be explored.

After regression models were explored, classification models were then used to predict the binary outcome of whether or not an individual is obese. The three methods used in developing models were logistic regression, nearest neighbors, and decision trees. The model that had the best performance was the decision tree model, which produced excellent capabilities in predicting whether or not an individual was obese, with an Area Under the Curve (AUC) value of *0.899*. The decision tree model was able to be improved even further using the random forests technique, which improved the AUC value of the model to *0.952*.

Importantly, this model can be used as a tool to predict obesity in situations when weight and height are not available. In particular, in a situation where patient characteristics/behaviour data is available but not their weight and height, this model can give the ability to still diagnose someone with obesity. This diagnosis allows the early initiation of health protocols to prevent the obesity associated diseases from developing in individuals, which is very helpful in improving the health of affected individuals.

References

1. Estimation of Obesity Levels Based On Eating Habits and Physical Condition
2. An overview of weight and height measurements on World Obesity Day
3. Defining Adult Overweight & Obesity

University of Manitoba DATA-2010 slides

Appendix

```
knitr::opts_chunk$set(echo = FALSE, warning = FALSE, message=FALSE, tidy.opts=list(width.cutoff=90), ti
library(kableExtra)
v = matrix(ncol = 4, nrow = 14)
v[1,] = c("Smoke", "SMOKE", "Binary", "Yes / No")
v[2,] = c("Number of main meals", "NCP", "Categorical", "Between 1 and 2 / 3 / More than 3")
v[3,] = c("Gender", "Gender", "Binary", "Male / Female")
v[4,] = c("Time using electronics", "TUE", "Numerical", "0-2 hours / 3-5 hours / More than 5 hours daily
v[5,] = c("Consumption of water","CH2O", "Numerical", "Less than a liter / Between 1 and 2 liters / More
v[6,] = c("Transportation used", "MTRANS", "Categorical", "Automobile / Motorbike / Bike / Public transp
v[7,] = c("Calorie consumption monitoring", "SCC", "Binary", "Yes / No")
v[8,] = c("Freq of physical activity", "FAF", "Numerical", "I do not have / 1 or 2 days / 2 or 4 days /
v[9,] = c("Freq of Consumption of alcohol", "CALC", "Categorical", "I do not drink / Sometimes / Frequen
v[10,] = c("Freq consumption of high caloric food", "FAVC", "Binary", "Yes / No")
v[11,] = c("Freq consumption of vegetables", "FCVC", "Categorical", "Never / Sometimes / Always")
v[12,] = c("Age", "Age", "Numerical", "Numeric value for age")
v[13,] = c("Consumption of food between meals", "CAEC", "Categorical", "No / Sometimes / Frequently / Al
v[14,] = c("History of overweight", "family_history_with_overweight", "Binary", "Yes / No")

kbl(v, col.names = c("Variable", "Variable Name", "Variable Type", "Variable Response"), align = "c",
    caption = "\\label{tab1}Dataset Variables", booktabs = T) %>%
  kable_styling(latex_options = c("striped", "scale_down", "HOLD_position"))
# Setup Code Chunk
# Add any libraries etc used here.
library(ggplot2)
library(tidyverse)
library(splines)
library(glmnet)

# Data importing
dataset = read.csv("ObesityDataSet_raw_and_data_sinthetic.csv")

# Is zero indicating no missing values
# sum(is.na(dataset))

# factor all categorical variables
dataset[sapply(dataset, is.character)] = lapply(dataset[sapply(dataset, is.character)], as.factor)

set.seed(1)
row.number = sample(1:nrow(dataset), 0.7 * nrow(dataset))
trainData = dataset[row.number, ]
testData = dataset[-row.number, ]

# BMI to use
BMI = trainData$Weight / (trainData$Height ^2)
library(kableExtra)
ObesityDataset = dataset %>% mutate(BMI = Weight / (Height^2))
table = ObesityDataset %>% summarize("Age" = round(cor(ObesityDataset$BMI,
                                                       ObesityDataset$Age, method = "spearman"), 4),
                                     "Vegetable Consumption" = round(cor(ObesityDataset$BMI,
                                                                         ObesityDataset$FCVC, method = "spearman"),
                                "Meals Quantity" = round(cor(ObesityDataset$BMI,
```

```
                                              ObesityDataset$NCP, method = "spearman"), 4),
                        "Water Consumption" = round(cor(ObesityDataset$BMI, ObesityDataset$CH2O,method =
                        "Physical Activity" = round(cor(ObesityDataset$BMI, ObesityDataset$FAF,
                                                  method = "spearman"),4),
                        "Screen Time" = round(cor(ObesityDataset$BMI, ObesityDataset$TUE,
                                                  method = "spearman"), 4))

Variable = matrix(c("Correlation"), nrow = 1)

kable(cbind(Variable, table), align = "c", longtable = TRUE, caption = "\\label{tab2}Correlation")
# Lets graph the regression models

# Age
m = lm(BMI ~ poly(Age, 4), data = trainData)
trainData |>
  mutate(fitted = fitted(m)) |>
  ggplot(aes(x = Age)) + geom_point(aes(y = BMI), size = 1) +
  geom_line(aes(y = fitted),
            colour = "red", linewidth = 1) +
  ggtitle("Relationship between Age and BMI")

# FCVC
m = lm(BMI ~ poly(FCVC, 4), data = trainData)
trainData |>
  mutate(fitted = fitted(m)) |>
  ggplot(aes(x = FCVC)) + geom_point(aes(y = BMI), size = 1) +
  geom_line(aes(y = fitted),
            colour = "red", linewidth = 1) +
  ggtitle("Relationship between TUE and BMI")
# Checking for multicollinearity
double_vars = sapply(dataset, function(x) is.numeric(x) && !all(x %% 1 == 0))
double_data = trainData[, double_vars]
corMatrix = cor(double_data)
abs(corMatrix)
# Performing regression on every variable with BMI

# Gender
m = lm(BMI ~ Gender, data = trainData)
p = predict(m, newdata = testData)
rmse1 = sqrt(mean((BMI - p)^2))

# Age
m = lm(BMI ~ Age, data = trainData)
p = predict(m, newdata = testData)
rmse2 = sqrt(mean((BMI - p)^2))

# family_history_with_overweight
m = lm(BMI ~ family_history_with_overweight, data = trainData)
p = predict(m, newdata = testData)
rmse3 = sqrt(mean((BMI - p)^2))

# FAVC - Frequent consumption of high caloric food
m = lm(BMI ~ FAVC, data = trainData)
```

```r
p = predict(m, newdata = testData)
rmse4 = sqrt(mean((BMI - p)^2))

# FCVC - Frequent consumption of vegetables
m = lm(BMI ~ FCVC, data = trainData)
p = predict(m, newdata = testData)
rmse5 = sqrt(mean((BMI - p)^2))

# NCP - Number of main meals
m = lm(BMI ~ NCP, data = trainData)
p = predict(m, newdata = testData)
rmse6 = sqrt(mean((BMI - p)^2))

# CAEC - Consumption of food between meals
m = lm(BMI ~ CAEC, data = trainData)
p = predict(m, newdata = testData)
rmse7 = sqrt(mean((BMI - p)^2))

# SMOKE
m = lm(BMI ~ SMOKE, data = trainData)
p = predict(m, newdata = testData)
rmse8 = sqrt(mean((BMI - p)^2))

# CH2O - Consumption of water
m = lm(BMI ~ CH2O, data = trainData)
p = predict(m, newdata = testData)
rmse9 = sqrt(mean((BMI - p)^2))

# SCC - Calories consumption monitoring
m = lm(BMI ~ SCC, data = trainData)
p = predict(m, newdata = testData)
rmse10 = sqrt(mean((BMI - p)^2))

# FAF - Frequency of physical activity
m = lm(BMI ~ FAF, data = trainData)
p = predict(m, newdata = testData)
rmse11 = sqrt(mean((BMI - p)^2))

# TUE - Time using electronics
m = lm(BMI ~ TUE, data = trainData)
p = predict(m, newdata = testData)
rmse12 = sqrt(mean((BMI - p)^2))

# CALC - Consumption of alcohol
m = lm(BMI ~ CALC, data = trainData)
p = predict(m, newdata = testData)
rmse13 = sqrt(mean((BMI - p)^2))

# MTRANS - Transportation used
m = lm(BMI ~ MTRANS, data = trainData)
p = predict(m, newdata = testData)
rmse14 = sqrt(mean((BMI - p)^2))
```

```r
cat("Smoke =", rmse8, "\nNumber of main meals =", rmse6, "\nGender =", rmse1, "\nTime using electronics
# Lets graph the regression models

# NCP
m = lm(BMI ~ poly(NCP, 4), data = trainData)
trainData |>
  mutate(fitted = fitted(m)) |>
  ggplot(aes(x = NCP)) + geom_point(aes(y = BMI), size = 1) +
  geom_line(aes(y = fitted),
            colour = "red", linewidth = 1) +
  ggtitle("Relationship between NCP and BMI")

# TUE
m = lm(BMI ~ poly(TUE, 4), data = trainData)
trainData |>
  mutate(fitted = fitted(m)) |>
  ggplot(aes(x = TUE)) + geom_point(aes(y = BMI), size = 1) +
  geom_line(aes(y = fitted),
            colour = "red", linewidth = 1) +
  ggtitle("Relationship between TUE and BMI")
# Spline regression on the numerical variables to see if its a better approach

# NCP
m = lm(BMI ~ ns(NCP, df = 5), data = trainData)
p = predict(m, newdata = testData)
rmse1 = sqrt(mean((BMI - p)^2))

# TUE
m = lm(BMI ~ ns(TUE, df = 5), data = trainData)
p = predict(m, newdata = testData)
rmse2 = sqrt(mean((BMI - p)^2))

cat("Number of main meals =", rmse1, "\nTime using electronics =", rmse2)
# Lasso regression with all the lower error variables

# Create model
fit = lm(BMI ~ Gender + NCP + SMOKE + TUE, data = trainData)
X = model.matrix(fit)
y = BMI

# Remove intercept
X = X[, -1]
fit_lasso = glmnet(X, y)
X_test = model.matrix(~Gender + NCP + SMOKE + TUE, data = testData)
X_test = as.matrix(X_test)
# Remove intercept
X_test = X_test[, -1]
y_test = BMI[1:634]
predLasso = predict(fit_lasso, newx = X_test, s = 1.2)
rmseLasso = sqrt(mean((y_test - predLasso)^2))
cat("RMSE =", rmseLasso)
# Lasso regression all variables (no weight and height)
```

```r
# Create model
fit = lm(BMI ~ Gender + Age + family_history_with_overweight + FAVC + FCVC + NCP + CAEC + SMOKE + CH2O +
X = model.matrix(fit)
y = BMI

# Remove intercept
X = X[, -1]
fit_lasso = glmnet(X, y)
X_test = model.matrix(~Gender + Age + family_history_with_overweight + FAVC + FCVC + NCP + CAEC + SMOKE
X_test = as.matrix(X_test)
# Remove intercept
X_test = X_test[, -1]
y_test = BMI[1:634]
predLasso = predict(fit_lasso, newx = X_test, s = 4)
rmseLasso = sqrt(mean((y_test - predLasso)^2))
cat("RMSE =", rmseLasso)
# Lasso regression all variables (no weight and height)

# Create model
fit = lm(BMI ~ Gender + SMOKE, data = trainData)
X = model.matrix(fit)
y = BMI

# Remove intercept
X = X[, -1]
fit_lasso = glmnet(X, y)
X_test = model.matrix(~Gender + SMOKE, data = testData)
X_test = as.matrix(X_test)
# Remove intercept
X_test = X_test[, -1]
y_test = BMI[1:634]
predLasso = predict(fit_lasso, newx = X_test, s = 0.27)
rmseLasso = sqrt(mean((y_test - predLasso)^2))
cat("RMSE =", rmseLasso)
cat("Minimum =", min(BMI))
cat("Maximum =", max(BMI))
library(infer)
library(class)
library(Metrics)
library(yardstick)
library(randomForest)
library(party)
library(glmnet)

# Add class variabke Y
train = trainData %>% mutate(BMI = Weight / (Height^2),
                        Y = as.factor(as.numeric(BMI > 30)))
test = testData %>% mutate(BMI = Weight / (Height^2),
                        Y = as.factor(as.numeric(BMI > 30)))

# Remove unwanted variables
test = test %>% dplyr::select(-BMI, -Weight, -Height, -NObeyesdad)
train = train %>% dplyr::select(-BMI, -Weight, -Height, -NObeyesdad)
```

```r
# Check data balanced
prop_table = train %>%
  group_by(Y) %>%
  summarize(proportion = n()/(nrow(train)))
# population proportions for in text later
prop_y0 = prop_table[1,2]
prop_y1 = prop_table[2,2]


# Binary variable indicating whether an individual is obese or not obese
target = train$Y
# The actual classes of the individuals to compare our models results with
actual = test$Y
# chi-square test for categorical variables
categorical_table = tibble(Gender = chisq_test(train, Y ~ Gender)$p_value,
               FHWO = chisq_test(train, Y ~ family_history_with_overweight)$p_value,
               FAVC = chisq_test(train, Y ~ FAVC)$p_value,
               CAEC = chisq_test(train, Y ~ CAEC)$p_value,
               SMOKE = chisq_test(train, Y ~ SMOKE)$p_value,
               SCC = chisq_test(train, Y ~ SCC)$p_value,
               CALC = chisq_test(train, Y ~ CALC)$p_value,
               MTRANS = chisq_test(train, Y ~ MTRANS)$p_value)

# convert very small decimals to strings so are in table output
categorical_table = categorical_table %>% mutate_if(is.numeric, funs(as.character(signif(., 3))))

Variable = matrix(c("p-value"), nrow = 1)

kable(cbind(Variable, categorical_table),
      caption = "\\label{tab3}P-values for Chi-Squared Test",
      align = "c", longtable = TRUE)


# Numerical variables p_values
numerical_table = tibble(Age = t_test(train, Age ~ Y)$p_value,
               FCVC = t_test(train, FCVC ~ Y)$p_value,
               NCP = t_test(train, NCP ~ Y)$p_value,
               CH2O = t_test(train, CH2O ~ Y)$p_value,
               FAF = t_test(train, FAF ~ Y)$p_value,
               TUE = t_test(train, TUE ~ Y)$p_value)

numerical_table = numerical_table %>% mutate_if(is.numeric, funs(as.character(signif(., 3))))

kable(cbind(Variable, numerical_table), caption = "\\label{tab4}P-values for T-test",
      align = "c", longtable = TRUE)


# remove independent variables and with no difference with mean values for two classes of Y
train = train %>% dplyr::select(-Gender, -SMOKE, -NCP)
test = test %>% dplyr::select(-Gender, -SMOKE, -NCP)
```

```r
# normal logistic regression model
log_model = glm(Y ~ ., data = train, family = "binomial")

# from all coefficients MTRANS variable with Bike category has the largest absolute value
# coefficients(log_model)

# calculate accuracy and get probabilities
prob_l = predict(log_model, newdata = test, type = "response")
pred_class_l = as.numeric(prob_l > 0.5)
accuracy_l = mean(pred_class_l == actual)


# Perform Lasso Logistic regression
X = model.matrix(Y ~ ., data = train)
y = as.numeric(train$Y)
X = X[,-1]

test_noY = test %>% dplyr::select(-Y)
X_test = model.matrix(~ ., data = test_noY)
X_test = as.matrix(X_test)
X_test = X_test[,-1]

# function that does k-fold cross-validation to determine best value for lambda
min_lambda = cv.glmnet(X, y, alpha = 1, family = "binomial")$lambda.min

lasso_model = glmnet(X, y, alpha = 1, family = "binomial", lambda = min_lambda)

# checked coefficients of the model at the set min lambda value
# coef(lasso_model)

# Checked accuracy of lasso which did not improve fomr our original model above
# probabilities_l = predict(lasso_model, newx = X_test, type = "response")
# pred_lasso = ifelse(probabilities_l > 0.5, "1", "0")
# accuracy_lasso = mean(pred_lasso == actual)




# only use numeric values for knn so calculates distance properly
cols = sapply(train, is.numeric)

# keep Y non-numeric target variable
cols[length(cols)] = TRUE

# alter dataset in preparation for knn() function
train_k = train[cols]
test_k = test[cols]


# Find the best odd K value
max_Acc = -Inf
best_k = 0
```

```
accuracys = numeric(50)
roc = numeric(50)

max_a = -Inf
a_k = 0

# Within each loop we calculate accuracy and AUC for the roc-curve to determine the optimal
# value of K
for(i in seq(from = 0, to = 49, by = 1)){

  pred_class2 = knn(train = train_k[,-length(train_k)], test = test_k[,-length(test_k)],
                    cl = train_k[,length(train_k)], k = (2*i + 1))
  knn_prob = knn(train = train_k[,-length(train_k)], test = test_k[,-length(test_k)],
                cl = train_k[,length(train_k)], k = (2*i + 1), prob = TRUE)

  accuracys[i+1] = mean(pred_class2 == actual)

  knn_tab =  tibble(truth = actual,
                  estimate = attr(knn_prob,'prob'))

  roc[i+1] = roc_auc(data = knn_tab, truth = truth, estimate,
                     event_level = "second")$.estimate

  if(roc[i+1] > max_a){
    max_a = roc[i+1]
    a_k  = (2*i + 1)
  }

  if(accuracys[i+1] > max_Acc){
    max_Acc = accuracys[i+1]
    best_k  = (2*i + 1)
  }

}
# get probabilities from model using optimal K value
knn_prob = knn(train = train_k[,-length(train_k)], test = test_k[,-length(test_k)],
                cl = train_k[,length(train_k)], k = a_k, prob = TRUE)

# get predictions and calculate accuracy
pred_class = knn(train = train_k[,-length(train_k)], test = test_k[,-length(test_k)],
                    cl = train_k[,length(train_k)], k = a_k)
accuracy_knn = mean(pred_class == actual)

# normal decision tree
tree = ctree(Y ~ ., data = train)

# printed it to observe its structure
# tree

tree_prob = predict(tree, newdata = test, type = "prob")
probs = unlist(lapply(tree_prob, FUN = function(x){x[2]}))

class_pred = predict(tree, newdata = test)
```

```r
# confusion matrix
table_t = table(actual, class_pred)
accuracy_t = sum(diag(table_t))/sum(table_t)


# Improve model with Random Forest function
rf = randomForest(Y ~ ., data = train)

prob_rf = predict(rf, newdata = test, type = "prob")
rf_probs = prob_rf[,2]

pred_rf = predict(rf, newdata = test)
accuracy_rf = mean(actual == pred_rf)
# confusion matrix
table_rf = table(actual, pred_rf)
kable(table_t, align = "clc", caption = "\\label{tab5}Decision Tree Confusion Matrix") %>%
  kable_styling(full_width = F)

kable(table_rf, align = "clc", caption = "\\label{tab6}Random Forest Confusion Matrix") %>%
  kable_styling(full_width = F)

# c("Decision Tree", "Decision Tree", "Random Forest", "Random Forest")

kable(matrix(c(accuracy_l, accuracy_knn, accuracy_t, accuracy_rf), ncol = 4),
      caption = "\\label{tab7}Accuracy of Models",
      align = "c", col.names = c("Logistic Regression", "Nearest Neighbors",
                    "Decision tree", "Random Forest"),
      digits = 3)


# create individual tables to then join into big one to plot all model results together
log_tab = tibble(truth = actual,
                 estimate = prob_l,
                 model = "Logistic Regression")
knn_tab =  tibble(truth = actual,
                 estimate = attr(knn_prob,'prob'),
                 model = "Nearest Neighbors")
tree_tab <- tibble(truth = actual,
                   estimate = probs,
                   model = "Decision tree")
tree_tab2 <- tibble(truth = actual,
                   estimate = rf_probs,
                   model = "Random Forest")

data_pred <- bind_rows(knn_tab, tree_tab, tree_tab2, log_tab)


# *** Since the data is balanced  ROC
data_pred %>%
  group_by(model) %>%
  roc_curve(truth, estimate, event_level = "second") %>%
  autoplot()
```

```r
table_auc = data_pred %>%
  group_by(model) %>%
  roc_auc(truth = truth, estimate, event_level = "second")

# DATA CLEANING to get table in correct format to be displayed
table_auc = table_auc %>% dplyr::select(-.metric, -.estimator) %>%
  pivot_wider(names_from = model,
              values_from  = .estimate)
Model = matrix(c("AUC"), nrow = 1)

kable(cbind(Model, table_auc), caption = "\\label{tab8}Area Under the Curve for ROC",
      align = "c", longtable = TRUE,
      digits = 3)
```