Project 3

Ryan Fischbach

5/1/2021

Abstract

The ability to explain the relationship between a food and its nutritional information is incredibly important to our client, who works with clients as a nutritionist and helps them understand nutrition and make healthy food choices. An interpretable model to predict the number of calories in a food item and what characteristics explain different categories of food would help our client easily explain how to adopt healthier habits. In this report, we discuss the process of estimating the number of calories in a food item and classifying a food item using nutrition data from McDonald's. The steps in the process and comparison of results from different models are discussed.

Part 1: Data Cleaning

To transform this initial dataset into data that can be used to generate a model, the following steps were taken to clean the data.

The uncleaned dataset was loaded, containing 24 variables and 259 observations.

1. First, columns that encoded the same information as others (redundant columns were removed). An example of this is cholesterol and cholesterol daily value. Despite having different values, these are the same relative to each other but just on a different scale. Thus, we don't want to use this redundant information. During this step, Total Fat Daily Value, Saturated Fat Daily Value, Cholesterol Daily Value, Sodium Daily Value, Carbohydrates Daily Value, and Dietary Fiber Daily Value were removed.
2. Next, the Calories From Fat column was removed at the instruction of our client.
3. Lastly, the Item column was removed as it serves as just an identifier for our observations.

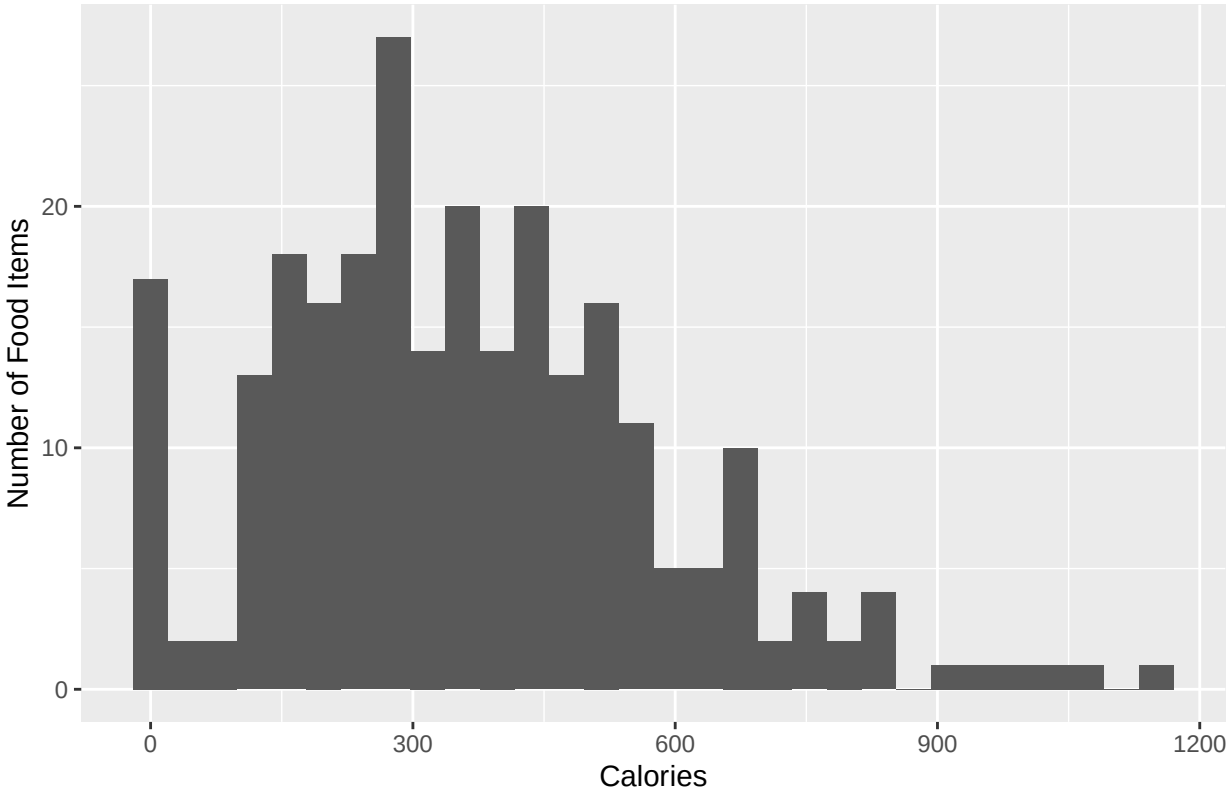
Additionally, the dataset was scanned for missing values to uncover if any additional work needed to be done to deal with them. After the previous data cleaning steps, there are no missing values, requiring no action on our part.

After these efforts, the dataset has 17 features with 259 rows.

Part 2: Modeling Calories

Before modeling calories, we will first explore the response variable to better understand its distribution. A histogram will be used for this task.

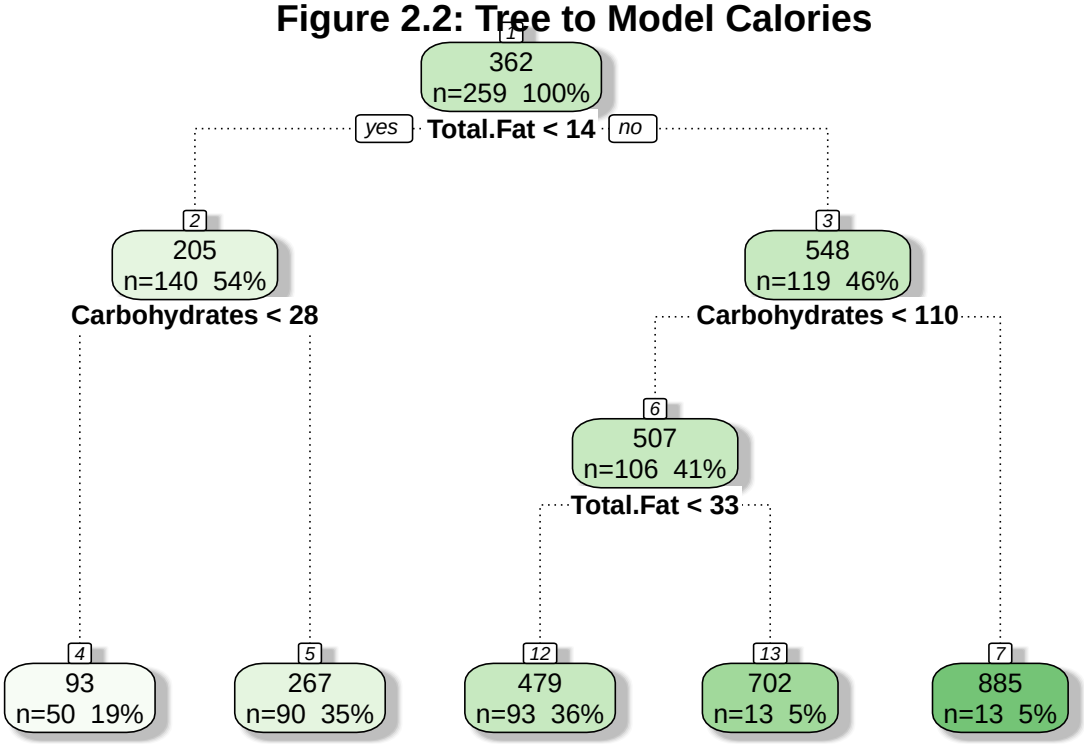
Figure 2.1: Distribution of Calories



Looking at the distribution of the response variable above (see Figure 2.1), it appears to be skewed-right, with its right tail being longer than its left tail. Additionally, it appears to be multimodal. The distribution has a median of 340 and of mean 362.43. The distribution has a large spread with a standard deviation of 221.49. Additionally, there are multiple outliers that exist on the right tail with calories greater than 910 (1.5*IQR + Q3).

Given this situation, I would recommend using a regression tree-based model to handle this request. With the combination of categorical and numeric data, trees allow these to be handled without having to use any sort of encoding or other practices. Additionally, a tree can perform feature selection, allowing for the most important factors to be used when attempting to model calories. This will allow for an interpretable model given the flowchart-like structure of trees, which can also be visualized.

Figure 2.2: Tree to Model Calories



After training the tree-based model, the flowchart above was created. As you can see in Figure 2.2, the "splits" indicating a decision put the data into leaves (buckets) at the bottom to indicate a prediction. Using these thresholds, we can best classify a food's calories while balancing the interpretability of the tree. For example, starting from the top, for the first split we see if the total fat in the item is less than 14. If it is, the item moves to the left sub node and is assigned a prediction of 205 calories. If it is not, it moves to the right subnode with a prediction of 548 calories.

From this tree, we estimate that carbohydrates and fats are two important characteristics of lower and higher-calorie foods. To assign data into the final prediction "buckets", the fats and carbohydrates are looked at. For example, we can associate foods with fat greater than 14 and carbohydrates greater than 110 with the highest number of calories and foods with less than 14 grams of fat and 28 grams of carbs associated with the lowest calories. Thus, we can conclude that these are important factors in determining the number of calories in a food item. This makes sense as these are macronutrients that make up a food's total calories.

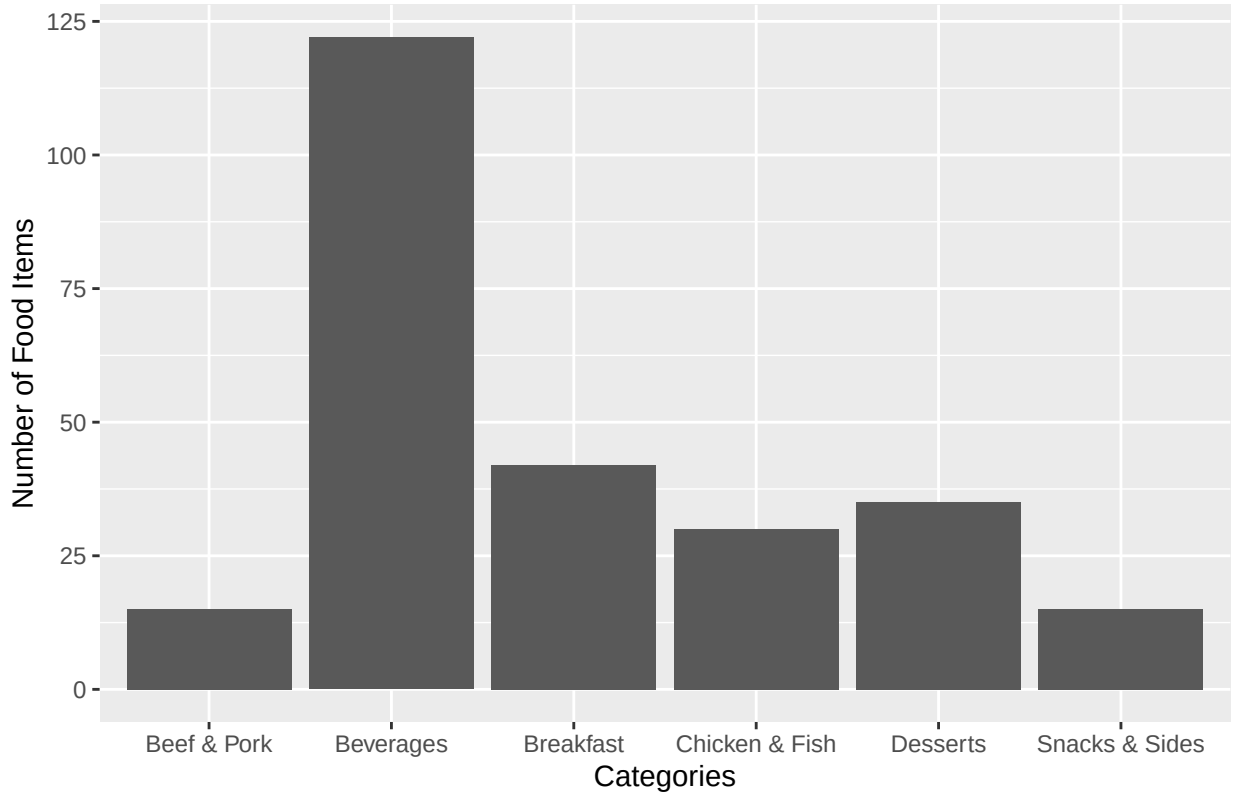
To validate this model, 10-fold cross-validation was chosen. To perform 10-fold cross-validation, we create 10 sets of the data, all roughly equal size. Each small dataset is a fold, derived from the original dataset. One fold is used as a test set to validate the model, with the rest being used to train the model. Each combination of training and test sets is used, yielding 10 combinations of cross-validation testing. This method was chosen because of its balance of computational complexity compared to a more rigorous method (like Leave One Out Cross-Validation) and the ability to eliminate randomness (unlike the Validation Approach). With 259 observations, this method was appropriate for this problem context.

Using this 10-fold cross-validation, the average training error for our predictions on the test fold was 103.17 calories. The average error on the data used to train the tree was 85.42 calories. Both were calculated by using the percentage reduction in the root node error multiplied by the root node error (see codebook for calculations).

Part 3: Modeling Category

The next goal of this analysis is to help the client understand how nutrition is across different categories of food. For this, we will model all categories in this dataset. This option was chosen because drinks and snacks can play an important role in nutrition as well, so being able to help model this relationship is important.

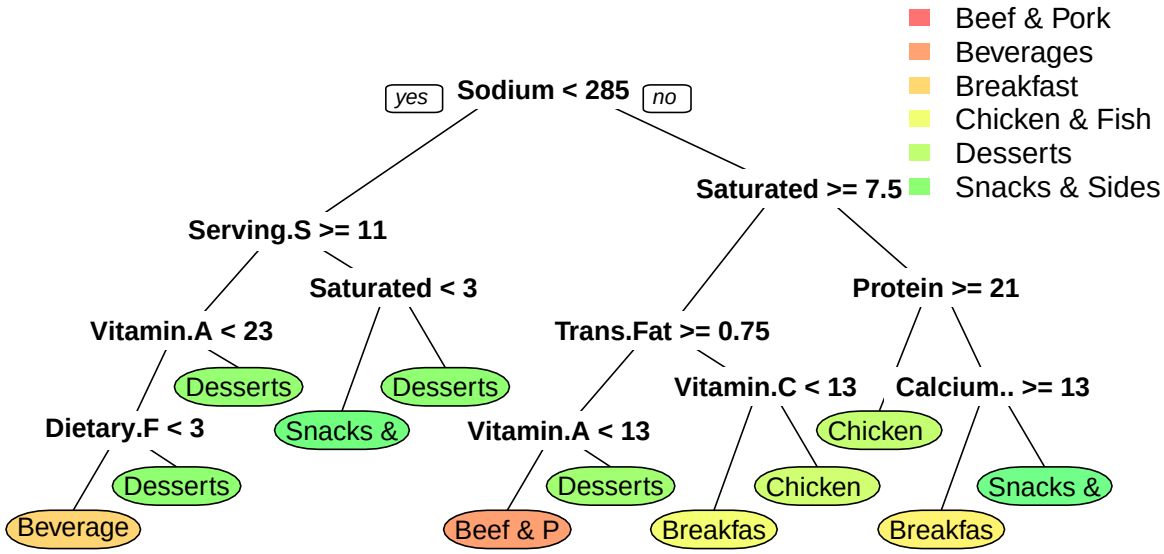
Figure 3.1: Distribution of Categories



To investigate the distribution of categories in the data, a barplot was created (see Figure 3.1). It is clear that beverage is the predominant class. The other classes are roughly equivalent (relatively), existing somewhere between 13 and 40 observations each.

Given this situation, I would recommend using a classification tree-based model to handle this request. With the combination of categorical and numeric data, trees allow these to be handled without having to use any sort of encoding or other practices. Additionally, a tree can perform feature selection, allowing for the most important factors to be used when attempting to model calories. This will allow for an interpretable model given the flowchart-like structure of trees, which can also be visualized.

Figure 3.2: Tree to Model Categories



After training the tree-based model, the flowchart above was created. As you can see in Figure 3.2, the "splits" indicating a decision to classify the data into leaves (buckets) at the bottom to indicate a prediction. Using these thresholds, we can best classify a food's category while balancing the interpretability of the tree. For example, starting from the top, for the first split we see if the sodium is less than 285. If it is, the item moves to the left sub node and is asked if the service size is greater than 11 oz. If it is not, it moves to the right part of the tree where it is asked if its saturated fat is greater than 7.5 grams. In this case, sodium was the first feature used to help the model by creating a cutoff and based on the item's nutrition, sending it down a part of the tree.

To create this tree, several different parts of the item's nutritional information were used. Specifically, from top to bottom, Sodium, Service Size, Saturated Fat, Protein, Trans Fat, Vitamin A, Vitamin C, Calcium, and Dietary Fiber were all used to help classify the food item into a category. From this tree, we can assume that these are some of the important attributes that can help classify a food item into a category.

To validate this model, 10-fold cross-validation was chosen. To perform 10-fold cross-validation, we create 10 sets of the data, all roughly equal size. Each small dataset is a fold, derived from the original dataset. One fold is used as a test set to validate the model, with the rest being used to train the model. Each combination of training and test sets is used, yielding 10 combinations of cross-validation testing. This method was chosen because of its balance of computational complexity compared to a more rigorous method (like Leave One Out Cross-Validation) and the ability to eliminate randomness (unlike the Validation Approach). With 259 observations, this method was appropriate for this problem context.

Our tree's estimated classification error rate from 10-fold cross-validation is 18.9%. Additionally, using the tree on the data used to train it yields a training classification error rate of 10.81%. These were calculated using the root node error (assigning the predominant class label) and multiplying it by the reduction in error achieved through the tree (see codebook).

Part 4: Predicting Calories or Predicting Category

In the final part of our analysis, we will be creating a model that can predict calories. We will be comparing two methods: Random Forests and Bagged Forests.

A bagged forest is a technique that combines multiple regression trees (as seen earlier in the modeling calories portion of this analysis) to increase predictive accuracy. To do this, we create 100 bootstrap samples from our original training sample (sampling with replacement). This, in essence, creates a variety of new datasets to train the trees on. A tree is then fit to each of the 100 samples, yielding 100 trees. The average of all of these trees' predictions is used as our final prediction.

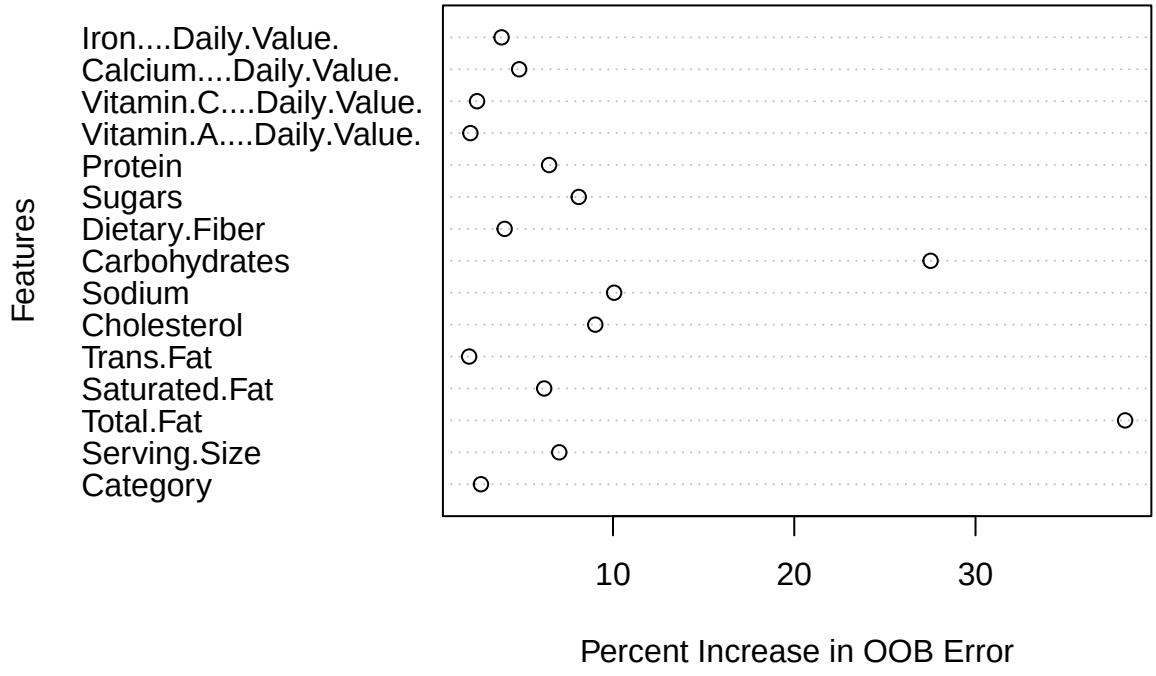
Bagged forests generally have higher predictive accuracy, but worse interpretability. This is because we are now using 100 trees for prediction, but also examining them becomes more difficult because we can no longer easily visualize the flowchart.

To determine performance, something called out-of-bag (OOB) rows are used. For each tree we generated, there are rows from the training data that were not used to grow that tree because of random sampling. These rows that were not used are called OOB and can be used with the tree to act as a "test" set and determines its performance. We then average predictions for each tree on that OOB row to yield a final prediction. After determining these predictions on all OOB rows, we can compute that average error (Root Mean Squared Error) on these OOB rows to determine performance.

Using this metric, we see that our prediction was off by 33.08 calories on average.

To answer the question of how important a feature is, we take a feature and scramble it (permute it). We then use this scrambled feature and run the bagged forest and determine its performance. This scrambled feature performance is compared to its performance when it's not scrambled and determine how the performance changed. If the feature is important, the error rate will change significantly, allowing us to determine the important explanatory variables.

Figure 4.1: % Increase in OOB Error in Bagged Fore



Looking at Figure 4.1, we can see that scrambling Total Fat and Carbohydrates resulted in the highest percentage increase in the OOB error. Thus, we can conclude that according to this bagged forest model, these features were the most important. This is in line with the regression tree that was created to model categories in part 2.

Moving on, a random forest is a technique that combines multiple regression trees (as seen earlier in the modeling calories portion of this analysis) to increase predictive accuracy. To do this, we create 100 bootstrap samples from our original training sample (sampling with replacement). This, in essence, creates a variety of new datasets to train the trees on. Up until this point, random forests are identical to bagged forests. However, at this point, we choose a random sub-sample of the features to explore a variety of trees and have more variety. Next, we consider those chosen features for the first split (decision in the flowchart). Next, we choose new features and repeat this process for the next split. This continues until we hit a stopping rule, yielding 100 trees. The average of all of these trees' predictions is used as our final prediction.

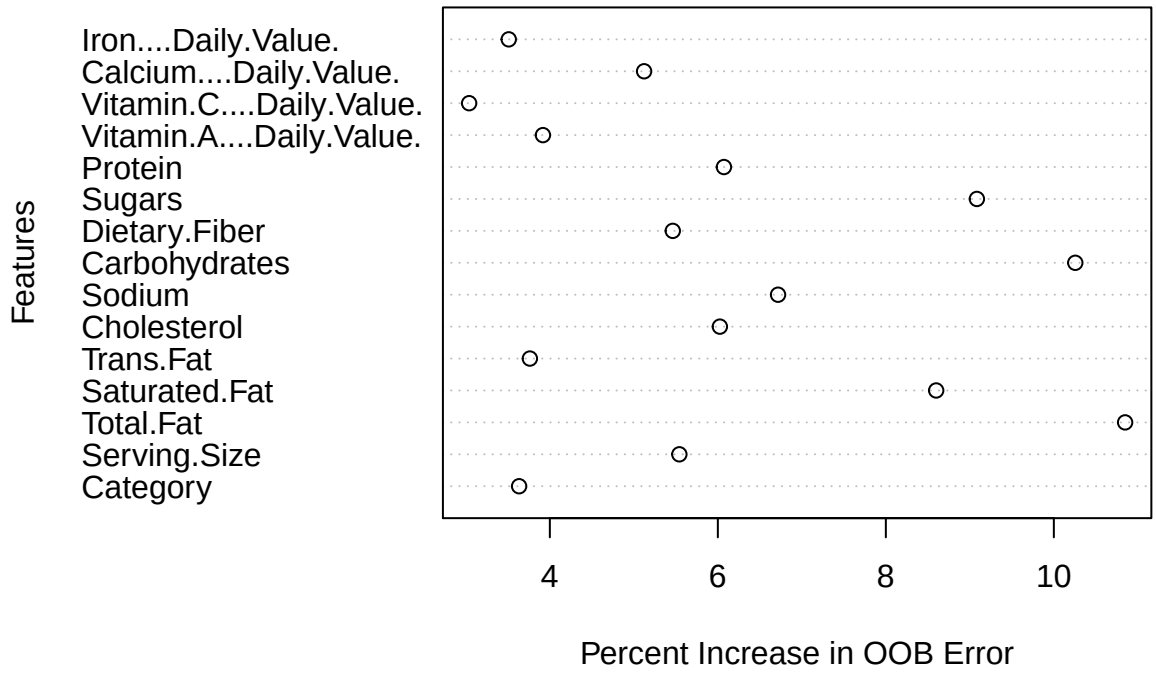
Random forests generally have higher predictive accuracy than single trees, but worse interpretability. This is because we are now using 100 trees for prediction, but also examining them becomes more difficult because we can no longer easily visualize the flowchart.

To determine performance, something called out-of-bag (OOB) rows are used. For each tree we generated, there are rows from the training data that were not used to grow that tree because of random sampling. These rows that were not used are called OOB and can be used with the tree to act as a "test" set and determine its performance. We then average predictions for each tree on that OOB row to yield a final prediction. After determining these predictions on all OOB rows, we can compute that average error (Root Mean Squared Error) on these OOB rows to determine performance.

Using this metric, we see that our prediction was off by 34.30 calories on average.

To answer the question of how important a feature is, we take a feature and scramble it (permute it). We then use this scrambled feature and run the bagged forest and determine its performance. This scrambled feature performance is compared to its performance when it's not scrambled and determines how the performance changed. If the feature is important, the error rate will change significantly, allowing us to determine the important explanatory variables.

Figure 4.2: % Increase in OOB Error in Random Fore

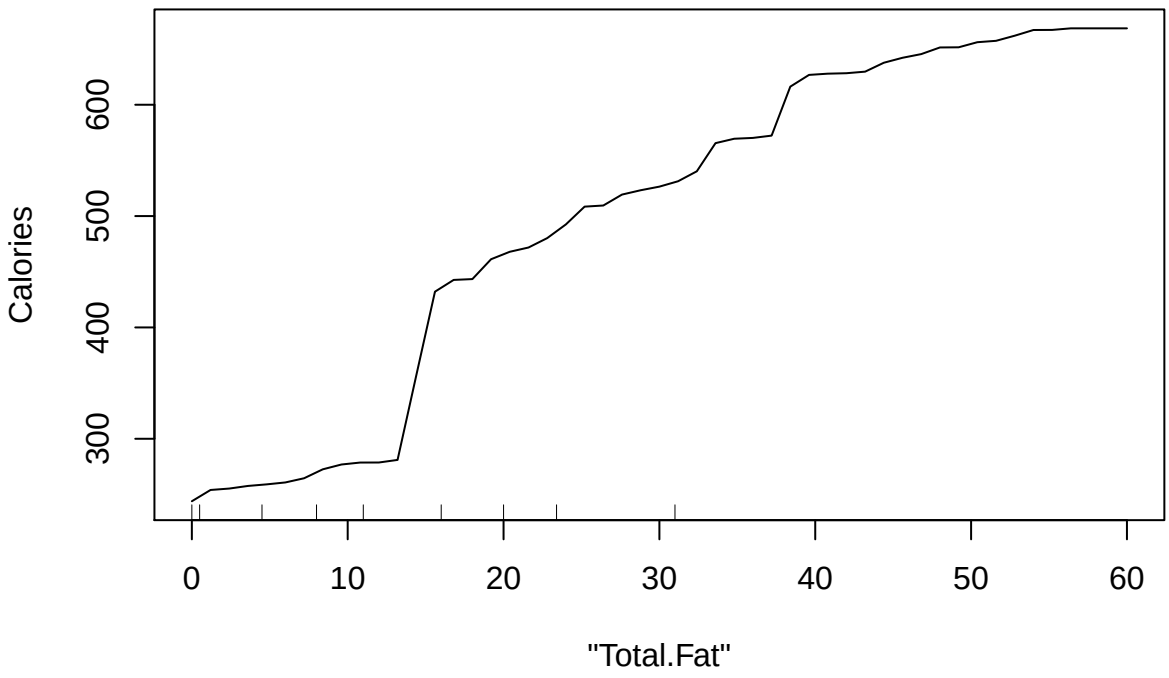


Looking at Figure 4.2, we can see that scrambling Total Fat and Carbohydrates resulted in the highest percentage increase in the OOB error. Thus, we can conclude that according to this bagged forest model, these features were the most important. This is in line with the regression tree that was created to model categories in part 2. However, the percent increase in OOB error was significantly less than with the bagged forest model in Figure 4.1.

From these two models, we can see that the OOB error for the bagged forest was 33.08 calories and 34.3 calories for the random forest. Despite their improved performance over the individual trees, I would not recommend using them for your clients. The gain in predictive accuracy is not worth the loss in interpretability. I would recommend using a single, pruned regression tree for this application as you don't need exact numbers to explain to a client the important factors in a food's nutritional profile. However, I would recommend picking the bagged forest because of its slightly better performance using OOB test RMSE as a metric.

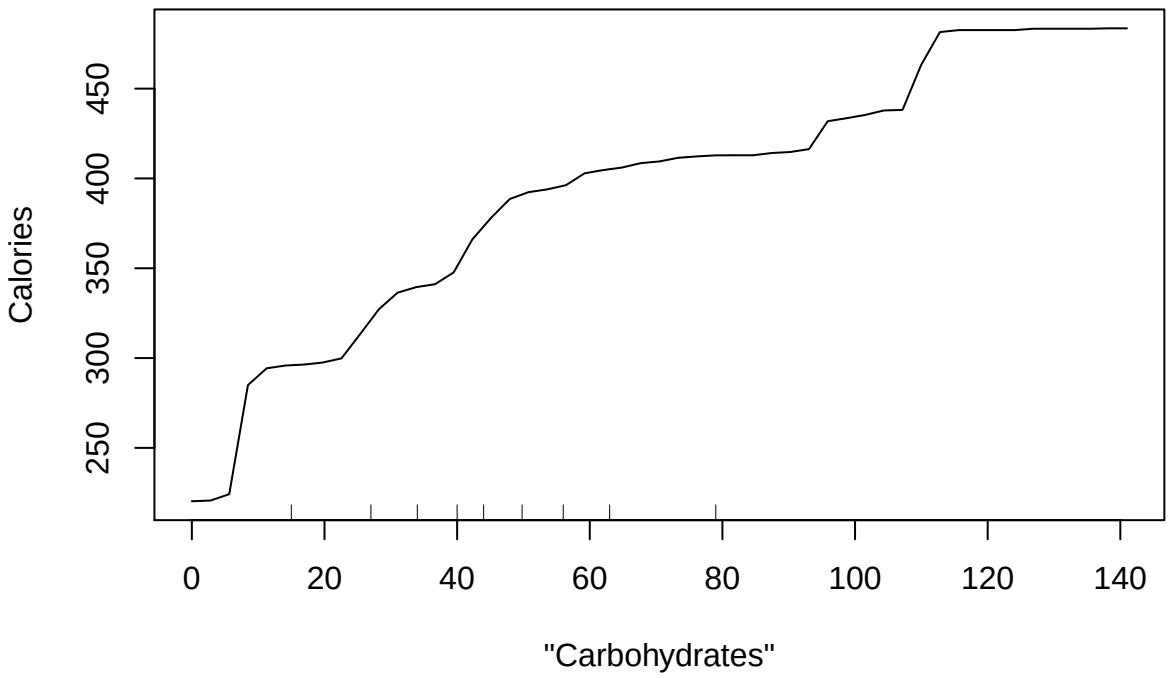
Now, we will look at partial dependence plots to allow us to see the impact of two important variables on the predictions made by our bagged forest.

Figure 4.3: Partial Dependence on Total.Fat



First, looking at fat (the most "important" feature to model calories in our models), we see a roughly linear positive relationship. As total fat increases, the number of calories increases as well. This makes sense because fat is a macronutrient directly related to calories.

Figure 4.3: Partial Dependence on Carbohydrates



Second, looking at carbohydrates (the second most "important" feature to model calories in our models), we see a roughly linear positive relationship. As carbohydrates increases, the number of calories increases as well. This makes sense because carbohydrates are a macronutrient directly related to calories.