Project 3

Ethan Lewis

4/22/2021

Abstract

In this report we were tasked with developing models to help individuals understand the relationships between Calories, as well as Food Categories, and several nutritional features for McDonald's food items based on a data set provided by the client. We were able to create these easy to comprehend models by training a regression tree and classification tree, respectively. In addition to visually representing the association between the various features and Calories, we were tasked with training two models aimed at predicting the number of Calories in a food item. These models successfully took the form of a bagged regression forest and random forest. In the report below we detail our findings, the steps of these model building processes, as well as discuss several relevant metrics for each model.

Data Cleaning

Prior to any analysis and model building, we need to clean up our data set. Looking at the nutritional features included, we notice several of them have been reported twice: both in terms of a standard unit and in terms of recommended daily percentage. Additionally there are a few nutritional features that have been reported in terms of only one of these measures.

For the features that have been "doubly" reported, we will remove their recommended daily percentage versions and move forward using their standard unit versions. Additionally, per our client's request, we will remove the "Calories From Fat" feature and, instead, only work with "Calories". Finally, we will also be removing the feature "Item" as it's presence would be extremely disruptive during the model building process and is ultimately unimportant.

Modeling Calories

Per our client's request, we will be constructing a model to explain the relationship between different nutritional variables and the amount of "Calories" in a food item. Ideally, this model will be easily explainable to the non-statistical eye and, as a result, we have elected to construct a tree since it is a much more visually friendly model when compared to a complex model equation. Specifically, we will be constructing a regression tree since our response variable, "Calories", is numeric.

Prior to training our regression tree, let's take a quick look at the distribution of "Calories" in the data set using Figures 1.1 and 1.2 below.

Figure 1.1

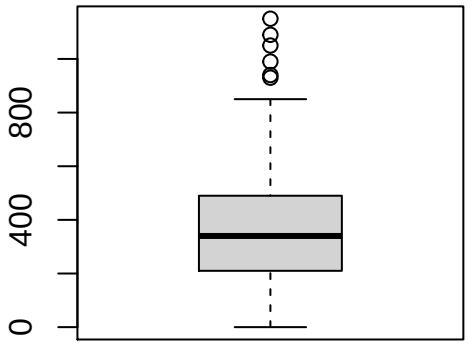
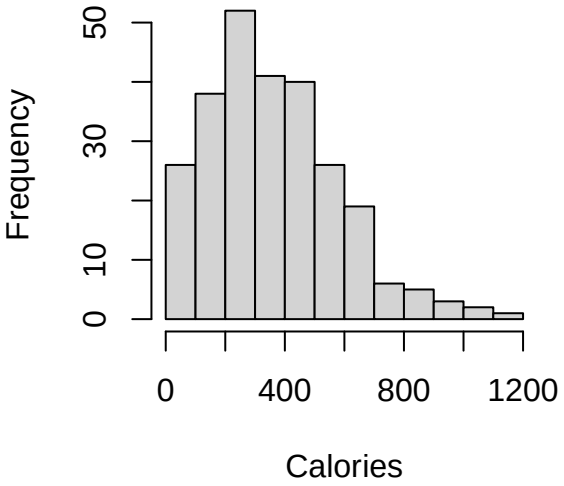


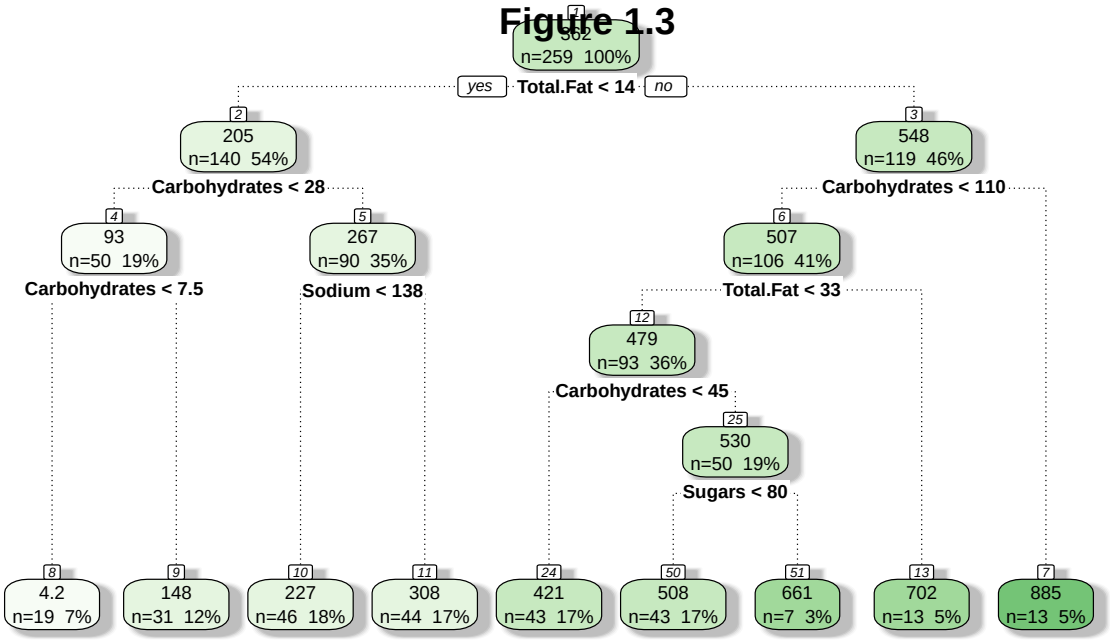
Figure 1.2



Both plots indicate our response variable is right skewed, meaning the majority of the data hovers around the sub-700 Calories level with a significantly smaller portion exceeding this mark. Additionally, Figure 1.1 indicates a few outliers that exceed 900 Calories: they will remain in the data set.

After exploring our response variable, we are ready to train our model and produce a visual of our regression tree. This process results in Figure 1.3 below.

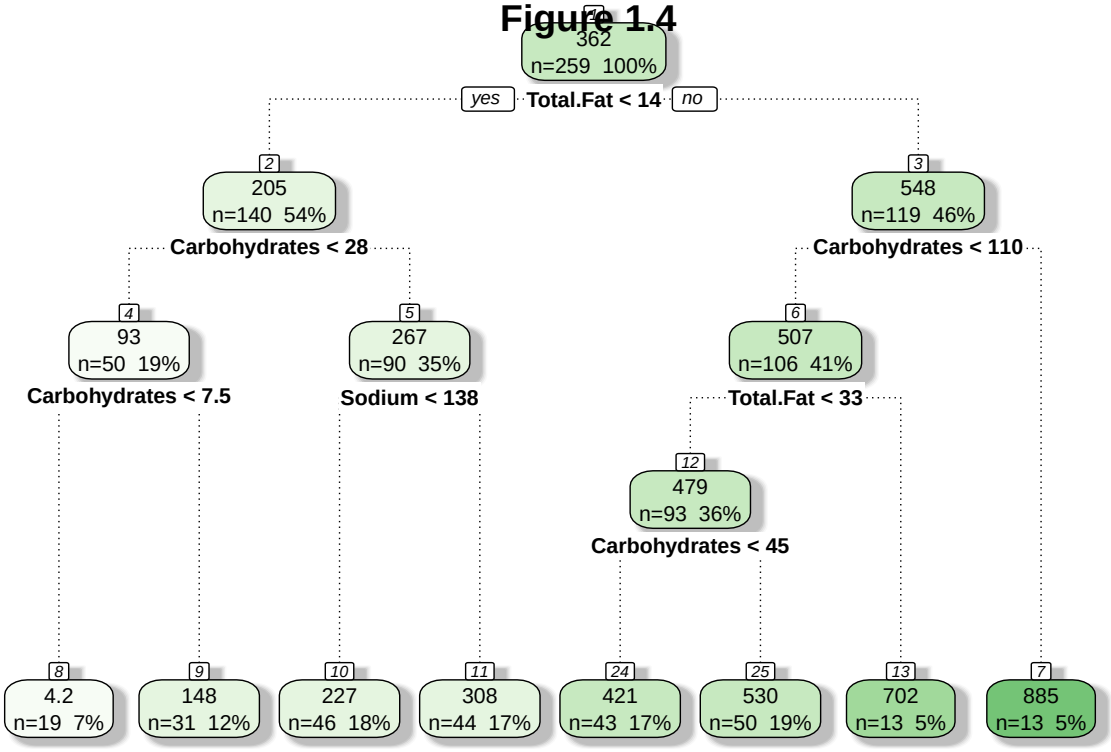
Figure 1.3



Although we have successfully produced a regression tree modeling "Calories" and it's various relationships, there are still some improvements to be made. We now need to prune our tree as a way to make the regression

tree a bit more interpretable, perform feature selection, and ultimately avoid overfitting. Doing so results in Figure 1.4 below. Additionally, we have calculated the pruned tree's test RMSE value to be 23.37007 using 10-fold cross validation. This is a large improvement on our training RMSE value of 68.79363 which was obtained in the usual way by performing the necessary functions to the squared differences between the true "Calories" values and our model's predicted "Calories" values.

Figure 1.4



It may be a bit useful to understand how Figure 1.4 was created, especially considering not all features seem to make an appearance in the tree (even before pruning) and the branches appear to split at fairly arbitrary values; obviously, this is not the case.

The process of creating a split in a tree looks something like this. At every potential splitting point, we consider every possible value (on a specified increment) of every possible feature and calculate what our model's RSS value (essentially a measure of how wrong our model is) would be if we split on this feature at this value. The feature and value that result in the smallest RSS value is the combination that is chosen to be the split. This process is repeated at every potential split point until either our RSS value stops decreasing or we hit a stopping rule.

As an example, we have calculated the RSS and splitting values of the feature "Total Fat" at the first potential split (the root) to be 5078255 and 14, respectively. Had we performed these calculations using a different feature, we would have obtained a larger RSS value and, obviously, a different splitting value.

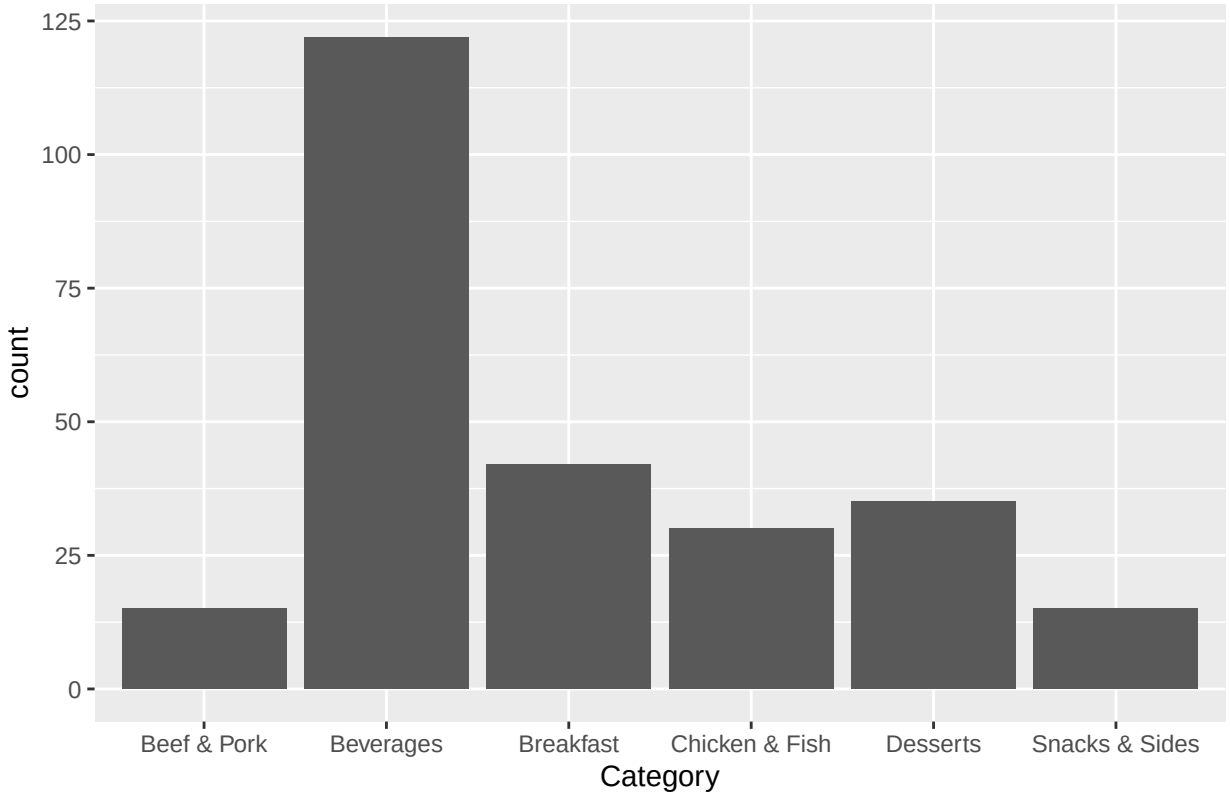
With a better understanding of how our tree was constructed, we can now attempt to decipher the story the tree is telling. Revisiting Figure 1.4, it quickly becomes apparent that a high amount of "Total Fat" and "Carbohydrates" is associated with a high number of "Calories" in a food item and vice versa. Additionally, our "Sodium" feature makes a brief appearance in the model and suggests a larger amount of "Sodium" is associated with a larger number of "Calories".

Modeling All Categories

After modeling the relationship between different nutritional variables and the amount of "Calories" in a food item, our client would like us to explore how these nutritional variables differ across all various food categories on the McDonald's menu. To do this, we will be constructing a classification tree to model "Category" since it is a categorical variable.

Prior to training our classification tree, let's take a quick look at the distribution of "Category" in the data set using Figure 2.1 below.

Figure 2.1



The bar graph in Figure 2.1 helps us visualize the number of items in each "Category" within our data set. Immediately, we notice an overwhelming number of 122 items classified as 'Beverages', with 'Breakfast items lagging behind in second with 42 items, while the rest of the categories hover around the 15-30 items.

Having explored our response variable, we are ready to train our model and produce a visual of our classification tree. This process results in Figure 2.2 below.

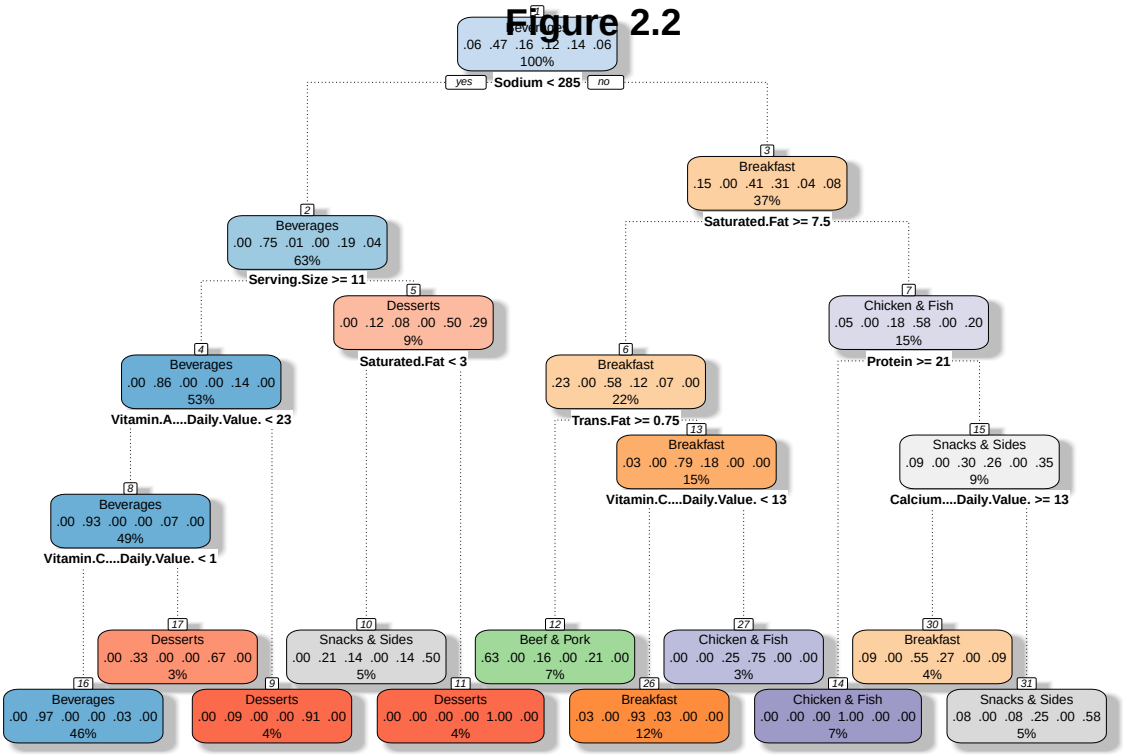


Figure 2.2

Although the typical next step in our process would be to prune the tree, as we did with our regression tree, pruning Figure 2.2 results in no changes thus making the process irrelevant. Instead, we have calculated our classification tree's test CER value to be 0.21622 (approximately 56/259 observations) by multiplying the tree's root node error (0.52896) by our reduction value (0.40876). Interestingly, this is a bit of a deterioration when compared to our training CER value of 0.135135 (13/259 observations) which was calculated by creating a confusion matrix of Actual vs. Predicted Categories and dividing the total number of incorrect predictions (13) by the sample size (259).

Similar to our regression tree, it may be a bit useful to understand how Figure 2.2 was created. While creating splits in both regression and classification trees involve the minimization of a specific metric, we will not be minimizing the RSS value with a classification tree. Instead, we will be choosing the splitting variable and value which minimizes the Gini Index of the tree (essentially a measure of prediction consistency).

The process of creating a split in a classification tree looks something like this. At every potential splitting point, we consider every possible value (on a specified increment) of every possible feature and calculate our model's Gini Index using a specified formula found below. The feature and value that result in the smallest Gini Index is the combination that is chosen to be the split. This process is repeated at every potential split point until we hit a stopping rule.

As an example, we can calculate the Gini Index of our first split, "Sodium < 285". Looking at the two leaves created by this split we can calculate their Gini Impurity scores to be 0.3997 (calculated by $1 - (0.75^2 + 0.01^2 + 0.19^2 + 0.04^2)$ and 0.7053 (calculated by $1 - (0.15^2 + 0.41^2 + 0.31^2 + 0.04^2 + 0.08^2)$), respectively. Finally, to calculate the Gini Index of the split we multiply each Gini Impurity score by the percentage of data present in each leaf and sum the products $((0.63)(0.3997) + (0.37)(0.7053))$ leaving us with a Gini Index of 0.512772. Had we performed these calculations using a different feature, we would have obtained a larger Gini Index value.

With a better understanding of how our tree was constructed, we can now attempt to decipher the story the tree is telling. Revisiting Figure 2.2, we immediately notice a distinct divide among our categories based

on the first splitting rule, Sodium. Looking at the tree we notice 'Beverages' and 'Desserts' typically have lower amounts of Sodium than 'Beef & Pork', 'Breakfast', and 'Chicken & Fish' while 'Snacks & Sides' seem to vary in Sodium levels. 'Beverages' seem to offer very little in terms of Vitamins A and C while 'Dessert' interestingly contains a decent amount of Vitamin A and a bit of Saturated Fat. As we would expect, between 'Beef & Pork' and 'Chicken & Fish' the former appears to be the fattier meat category in terms of Trans Fat. Additionally, 'Chicken & Fish' contain a decent amount of Vitamin C while 'Breakfast' has a notable amount of Calcium. This is just a brief summary of our findings; further relationships can be identified and explored by revisiting Figure 2.2.

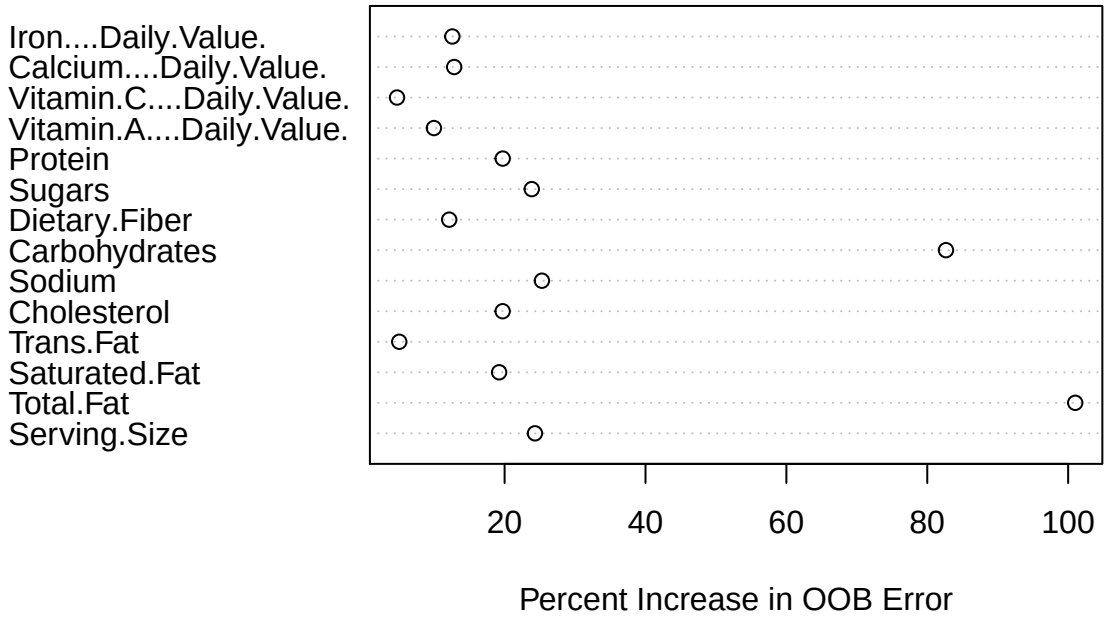
Predicting Calories

Per our client's request, we would like to construct a model aimed at predicting the number of "Calories" present in the various McDonald's food items. To achieve this, instead of relying on our regression forest from Figure 1.4 we will generate two forests: a bagged regression forest and a random forest, respectively.

Having trained our bagged regression forest using 1000 bootstrap samples (this essentially means our forest is comprised of 1000 trees each built on different variations of our data set), we know our Test MSE value to be 1000.499, and as a result, know our Test RMSE value to be 31.6307.

Additionally, let's compare our bagged regression forest and original regression tree in terms of features. To do this we can visualize the "importance" of each feature in our forest using Figure 3.1 below.

Figure 3.1



Prior to any interpretation, it is important to understand what we mean by "importance". As we mentioned, the process of creating a bagged regression forest will result in the calculation of a Test MSE value. In order to determine a feature's importance, we isolate all values of a specific feature in our data set, randomly shuffle them around, and assign them to new rows. Following this shuffle, we reconstruct our forest and obtain a new Test MSE value and compare it to our original Test MSE value in terms of percentage change. For reference: a positive importance value indicates the feature is important since the value suggests our Test MSE increased and predictive accuracy decreased, while a negative importance value results in the opposite

conclusion. Essentially, the larger the importance value, the more important the feature. *Note: the OOB Error metric of a bagged regression forest is Test MSE. This explains the discrepancy between our written summary and the y-axis label of Figure 3.1*

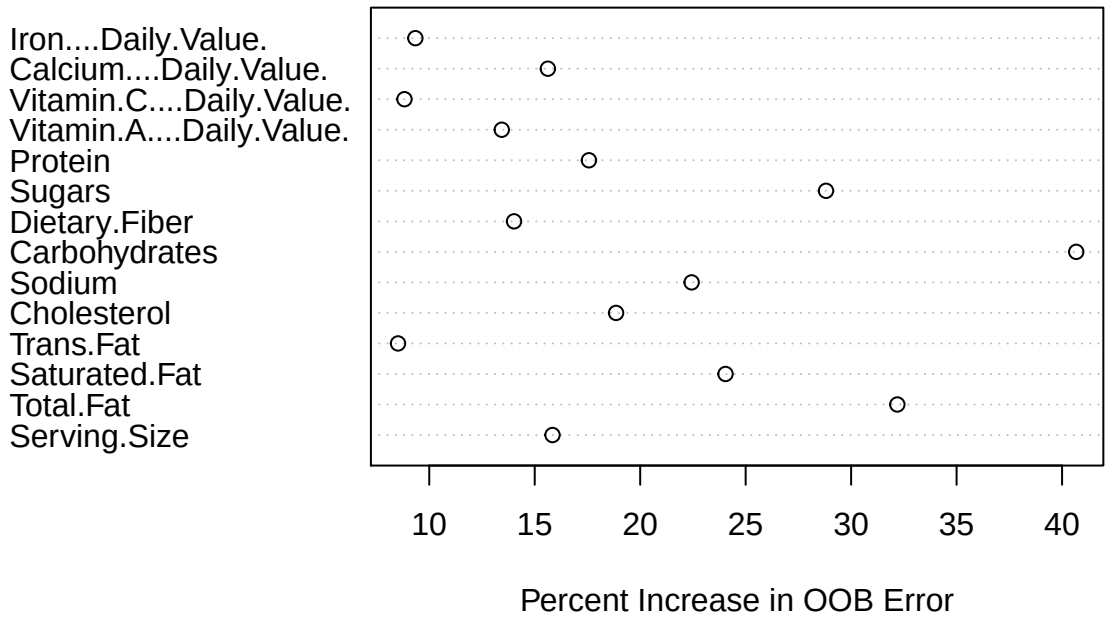
With the concept of importance in mind, we immediately notice "Total Fat" and "Carbohydrates" appear to be extremely important features. Lagging behind but rounding out the top 3 is "Sodium", although we notice there are several features with similar importance values. Now, let's compare our bagged regression forest's feature importance values to the feature importance values of our regression tree from Figure 1.4 which, in descending order, considers "Carbohydrates", "Total Fat", and "Sodium" to be the most important features.

Immediately, we notice a level of consistency among our values; mainly, all three features identified as important in our regression tree ("Total Fat", "Carbohydrates", and "Sodium") happen to be the top three most important features in our bagged regression forest (although "Sugars" is not far outside the top three). This makes sense considering these are the only features to appear in our regression tree as splitting features. The order of importance; however, is a bit intriguing as our regression tree identified "Carbohydrates" as the most important feature while the bagged regression forest chose "Total Fat". Overall, our bagged regression forest appears to match nicely with our original "Calories" regression tree.

After completing our bagged regression forest, we then trained a random forest using 1000 bootstrap samples and found our forest's Test MSE to be 1070.119 meaning it's Test RMSE is 32.71267.

Additionally, let's compare our random forest to our other models in terms of features using the same feature importance method we previously discussed. These results can be found below in Figure 3.2.

Figure 3.2



Immediately, we notice some key differences between our random forest's feature importance and our bagged regression forest's feature importance; primarily, "Total Fat" has been dethroned by "Carbohydrates" as the most important feature. In this role reversal, we also notice the importance values of both "Carbohydrates" and "Total Fat" have been drastically slashed: roughly a 50.92% and 68.06% decrease, respectively. Additionally, "Sugars" has taken "Sodium"'s third place spot and there are several features hovering around the 15%-20% increase range. Revisiting our regression tree's feature importance values, we notice the regression

tree and random forest share the same order of importance for their top two features: "Carbohydrates" followed by "Total Fat". As we previously mentioned; however, our random forest considers "Sugars" to be the third most important feature, yet the feature fails to make an appearance in our regression tree. Despite this, our random forest still seems to be a good match to our original "Calories" regression tree.

With all of the above information in mind the question of "Which forest should we use for prediction?" arises. To answer this, we will compare the forest's Test RMSE values as this is an adequate way to asses predictive accuracy. As we previously mentioned, the bagged regression forest has a Test RMSE value of 31.6307 while the random forest has a value of 32.71267. While the difference between these values is minimal, it still exists, and, as a result, we recommend using the bagged regression forest for predicting "Calories".

Having chosen our predictive model, we want to explore the relationships some of our features have with "Calories" using Figures 3.3 and 3.4 below.

Figure 3.3

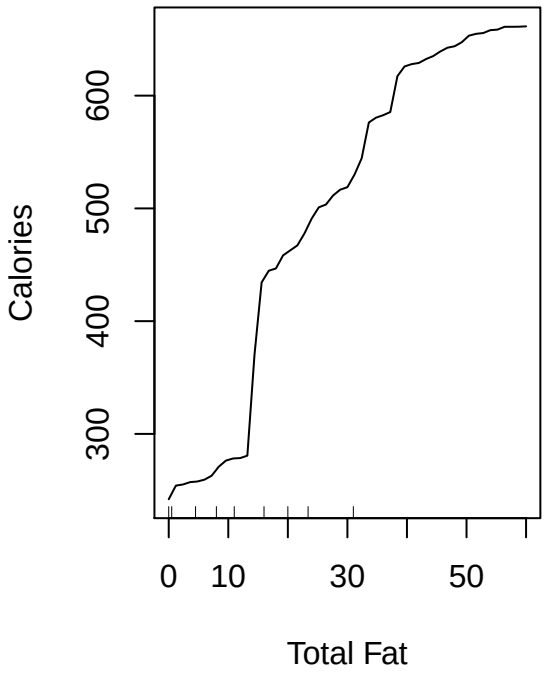
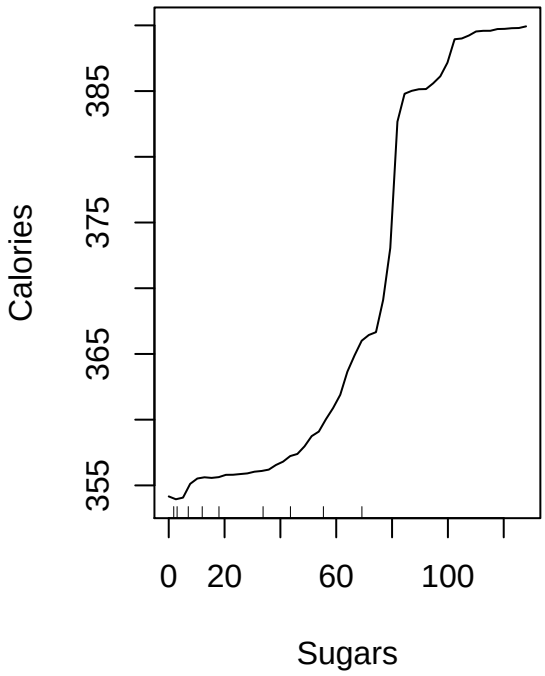


Figure 3.4



In FIgure 3.3 we are exploring the relationship between one of our most crucial features, "Total Fat" and "Calories". As we would have expected, the relationship is a noticeably positive one: as "Total Fat" in a food item increases, so too does "Calories". However, there is an interesting portion of this relationship: the relative flat line of about 250-275 Calories followed by a jump up to around 450 calories at about the 15g Total Fat mark. Further research into nutritional science may be needed to explain this phenomena.

Finally, in Figure 3.4we chose to explore the relationship between one of our more contentious features, "Sugars", and "Calories". Despite not making an appearance in our pruned regression tree in Figure 1.4 as a splitting feature, "Sugars" was a top 4 important feature at the minimum in both forests. With this in mind, we were expecting a positive relationship (when "Sugars" increases, "Calories" increase). We did not know; however, what the detailed shape of this relationship looked like and whether or not it may help us understand "Sugars"' importance a little better. Although further research into nutritional science may be needed for a deeper explanation, interestingly we detect a somewhat exponentially shaped relationship between "Sugars" and "Calories" until we hit a plateau of about 390 Calories around 80g of sugar.