Stat108_FinalProject

Dylan Scoble and Aisha Lakshman

2/4/2022

Introduction

In the 2004 documentary Super Size Me, writer and director Morgan Spurlock took on a month-long challenge to only eat McDonalds food. Spurlock experienced a multitude of health issues, including weight gain, cholesterol spike, and negative impacts on his energy and mood, demonstrating the fast-food chains' instrumental role in America's obesity epidemic (Stossel 2006). Spurlock's film not only emphasized the consequences of caloric intake, but also brought light to the nutritional attributes of McDonalds menu items that caused adverse health effects. There are many factors that impact the quality and quantity of calories, such as levels of fat, protein, and carbohydrates, which is why many dieticians support the notion that "not all calories are created equal" (Tolar-Peterson, 2021). Spurlock's documentary and existing literature inspired an investigation of McDonalds menu items' caloric and nutritional records. Our research will address the following question: What nutritional attribute is the best predictor of calories for the McDonalds menu items? Since drink and food items have quite different caloric makeups, we are dividing our dataset between food menu items (Breakfast, Beef & Pork, Chicken & Fish, Salads, Snacks & Sides, Desserts) and drink menu items (Coffee and Tea, Smoothies and Shakes, Beverages). For food menu items, we hypothesize total carbohydrates (grams) is the most accurate predictor of calories. For drink menu items, we hypothesize that total sugar (grams) is the most accurate predictor of calories. We will analyze a 2018 dataset from Kaggle titled "Nutritional Facts for McDonald's Menu" to answer our research question. Our chosen dataset provides nutritional information for all of McDonald's menu items, including calories, saturated fat, and cholesterol levels. We will create a linear model for each nutritional attribute with calories as the response variable for each predictor. A linear model for regression analysis is useful in answering our question because it will allow us to confidently determine what nutritional attributes matter the most for calories and predict an item's calorie count based on its predictors.

References

Tolar-Peterson, Terezie. 2021. "Not all calories are created equal - a dietician explains the different ways the kinds of foods you eat matter to your body". The Conversation. Retrieved Febuary 8th, 2022. https://theconversation.com/not-all-calories-are-equal-a-dietitian-explains-the-different-ways-the-kinds-of-foods-you-eat-matter-to-your-body-156900

Stossel, John. 2006. " 'Super Size Me' Carries Weight With Critics". ABC News. Retrieved Febuary 8th, 2022. https://docs.google.com/document/d/1XB-22QylvnbasBKe7n_DfkkENcZWRvIR5X8vZgOK6LY/edit#

Our Data

```
data <- read.csv("data/menu 2.csv")
glimpse(data)
```

```
## Rows: 260
## Columns: 24
## $ Category                      <chr> "Breakfast", "Breakfast", "Breakfast", "~
## $ Item                          <chr> "Egg McMuffin", "Egg White Delight", "Sa~
## $ Serving.Size                  <chr> "4.8 oz (136 g)", "4.8 oz (135 g)", "3.9~
## $ Calories                      <int> 300, 250, 370, 450, 400, 430, 460, 520, ~
## $ Calories.from.Fat             <int> 120, 70, 200, 250, 210, 210, 230, 270, 1~
## $ Total.Fat                     <dbl> 13, 8, 23, 28, 23, 23, 26, 30, 20, 25, 2~
## $ Total.Fat....Daily.Value.     <int> 20, 12, 35, 43, 35, 36, 40, 47, 32, 38, ~
## $ Saturated.Fat                 <dbl> 5, 3, 8, 10, 8, 9, 13, 14, 11, 12, 12, 1~
## $ Saturated.Fat....Daily.Value. <int> 25, 15, 42, 52, 42, 46, 65, 68, 56, 59, ~
## $ Trans.Fat                     <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, ~
## $ Cholesterol                   <int> 260, 25, 45, 285, 50, 300, 250, 250, 35,~
## $ Cholesterol....Daily.Value.   <int> 87, 8, 15, 95, 16, 100, 83, 83, 11, 11, ~
## $ Sodium                        <int> 750, 770, 780, 860, 880, 960, 1300, 1410~
## $ Sodium....Daily.Value.        <int> 31, 32, 33, 36, 37, 40, 54, 59, 54, 59, ~
## $ Carbohydrates                 <int> 31, 30, 29, 30, 30, 31, 38, 43, 36, 42, ~
## $ Carbohydrates....Daily.Value. <int> 10, 10, 10, 10, 10, 10, 13, 14, 12, 14, ~
## $ Dietary.Fiber                 <int> 4, 4, 4, 4, 4, 4, 2, 3, 2, 3, 2, 3, 2, 3~
## $ Dietary.Fiber....Daily.Value. <int> 17, 17, 17, 17, 17, 18, 7, 12, 7, 12, 6,~
## $ Sugars                        <int> 3, 3, 2, 2, 2, 3, 3, 4, 3, 4, 2, 3, 2, 3~
## $ Protein                       <int> 17, 18, 14, 21, 21, 26, 19, 19, 20, 20, ~
## $ Vitamin.A....Daily.Value.     <int> 10, 6, 8, 15, 6, 15, 10, 15, 2, 6, 0, 4,~
## $ Vitamin.C....Daily.Value.     <int> 0, 0, 0, 0, 0, 2, 8, 8, 8, 8, 0, 0, 0, 0~
## $ Calcium....Daily.Value.       <int> 25, 25, 25, 30, 25, 30, 15, 20, 15, 15, ~
## $ Iron....Daily.Value.          <int> 15, 8, 10, 15, 10, 20, 15, 20, 10, 15, 1~
```

Exploratory Data Analysis

For this project, we understand that foods and beverages may have different predictors for their number of calories. Therefore, we will be splitting our dataset into two different dataframes: one for foods, and one for beverages.

We will also be removing all predictors that have “as % of Daily Value” attached at the end, since our purpose is not focused the daily values of the nutrients. These predictors add no value to our dataset or models.

The first thing we are doing is filtering out Total Fat (% Daily Value), Saturated Fat (% Daily Value), Cholesterol (% Daily Value), Sodium (% Daily Value), Carbohydrataes (% Daily Value), Dietary Fiber (% Daily Value) from our nutritional attributes. These attributes don’t aid to answering our research question, so we are taking these predictor variables out of consideration.

```
data <- data %>%
  select(Category, Item, Serving.Size, Calories, Calories.from.Fat, Total.Fat,
         Saturated.Fat, Trans.Fat, Cholesterol, Sodium, Carbohydrates,
         Dietary.Fiber, Sugars, Protein, Vitamin.A....Daily.Value.,
         Vitamin.C....Daily.Value., Calcium....Daily.Value., Iron....Daily.Value.)

glimpse(data)
```

```
## Rows: 260
## Columns: 18
## $ Category                      <chr> "Breakfast", "Breakfast", "Breakfast", "Brea~
## $ Item                          <chr> "Egg McMuffin", "Egg White Delight", "Sausag~
## $ Serving.Size                  <chr> "4.8 oz (136 g)", "4.8 oz (135 g)", "3.9 oz ~
## $ Calories                      <int> 300, 250, 370, 450, 400, 430, 460, 520, 410,~
## $ Calories.from.Fat             <int> 120, 70, 200, 250, 210, 210, 230, 270, 180, ~
## $ Total.Fat                     <dbl> 13, 8, 23, 28, 23, 23, 26, 30, 20, 25, 27, 3~
## $ Saturated.Fat                 <dbl> 5, 3, 8, 10, 8, 9, 13, 14, 11, 12, 12, 13, 1~
## $ Trans.Fat                     <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0,~
## $ Cholesterol                   <int> 260, 25, 45, 285, 50, 300, 250, 250, 35, 35,~
## $ Sodium                        <int> 750, 770, 780, 860, 880, 960, 1300, 1410, 13~
## $ Carbohydrates                 <int> 31, 30, 29, 30, 30, 31, 38, 43, 36, 42, 34, ~
## $ Dietary.Fiber                 <int> 4, 4, 4, 4, 4, 4, 2, 3, 2, 3, 2, 3, 2, 3, 2,~
## $ Sugars                        <int> 3, 3, 2, 2, 2, 3, 3, 4, 3, 4, 2, 3, 2, 3, 3,~
## $ Protein                       <int> 17, 18, 14, 21, 21, 26, 19, 19, 20, 20, 11, ~
## $ Vitamin.A....Daily.Value.     <int> 10, 6, 8, 15, 6, 15, 10, 15, 2, 6, 0, 4, 6, ~
## $ Vitamin.C....Daily.Value.     <int> 0, 0, 0, 0, 0, 2, 8, 8, 8, 8, 0, 0, 0, 0, 0,~
## $ Calcium....Daily.Value.       <int> 25, 25, 25, 30, 25, 30, 15, 20, 15, 15, 6, 8~
## $ Iron....Daily.Value.          <int> 15, 8, 10, 15, 10, 20, 15, 20, 10, 15, 15, 1~
```

```
count(data, Category)
```

```
##             Category  n
## 1        Beef & Pork 15
## 2          Beverages 27
## 3          Breakfast 42
## 4     Chicken & Fish 27
## 5        Coffee & Tea 95
## 6           Desserts  7
## 7             Salads  6
## 8 Smoothies & Shakes 28
## 9      Snacks & Sides 13
```

Next, we are dividing our categorical variables between food items (Breakfast, Beef & Pork, Chicken & Fish, Salads, Snacks & Sides, Desserts) and drink items (Coffee and Tea, Smoothies and Shakes, Beverages).

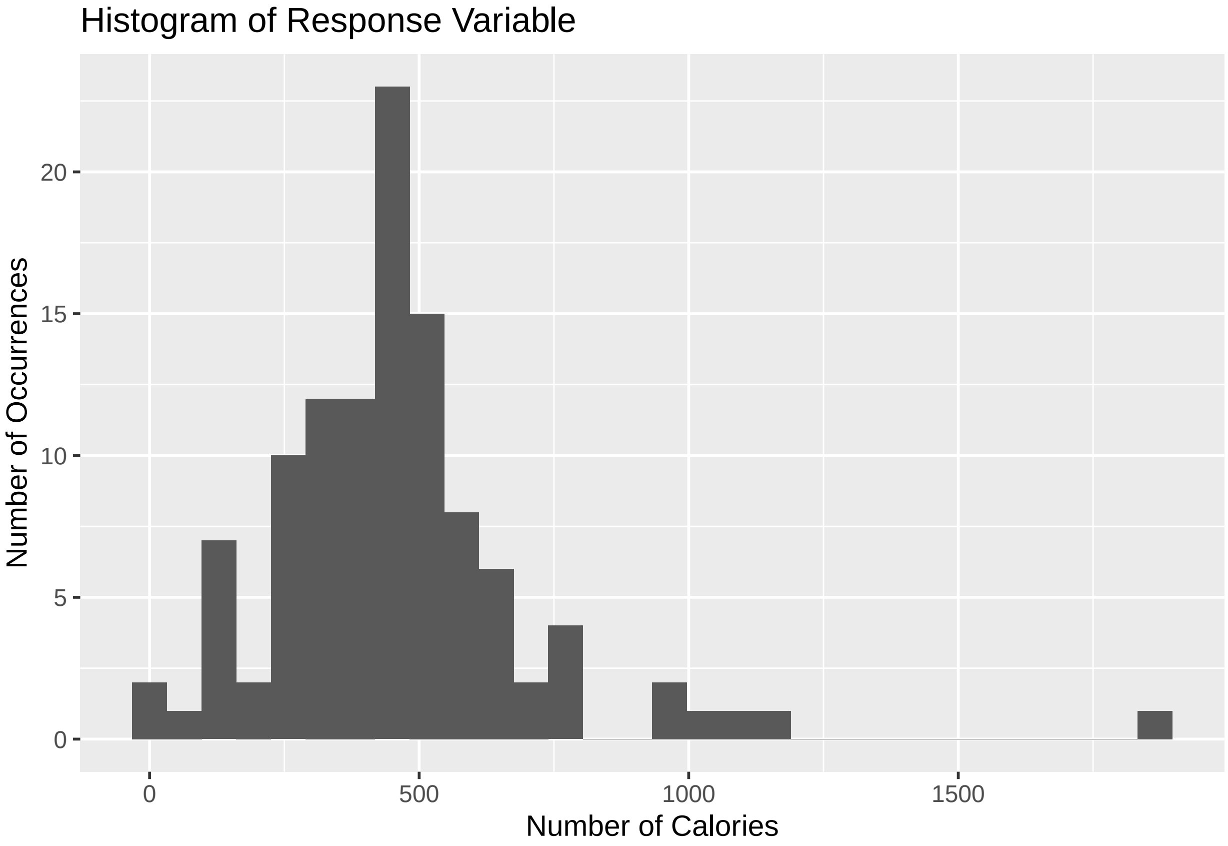
```
food_data <- data %>%
  filter(Category == "Beef & Pork" |
         Category == "Breakfast" |
         Category == "Chicken & Fish" |
         Category == "Desserts" |
         Category == "Salads" |
         Category == "Snacks & Sides")

bev_data <- data %>%
  filter(Category == "Beverages" |
         Category == "Coffee & Tea" |
         Category == "Smoothies & Shakes")
```

Response Variable

The next step is to create histograms for occurences of food items (food_data) and occurences of drink items (bev_data) against our response variable (calories)

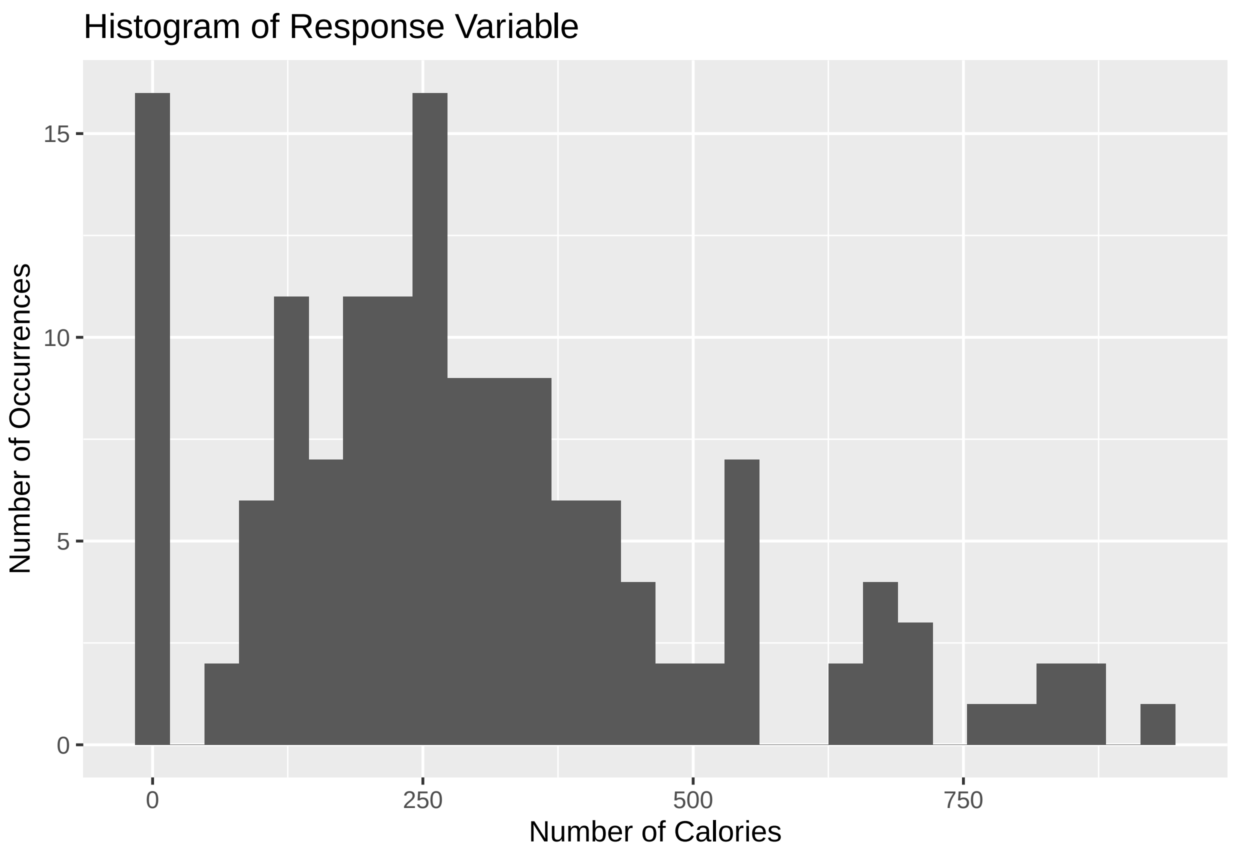
```
ggplot(data=food_data, aes(x=Calories)) +
  geom_histogram() +
  labs(title="Histogram of Response Variable",
       x="Number of Calories",
       y="Number of Occurrences")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



```
ggplot(data=bev_data, aes(x=Calories)) +
  geom_histogram() +
  labs(title="Histogram of Response Variable",
       x="Number of Calories",
       y="Number of Occurrences")
```

```
## ‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



Now we calculate the appropriate summary statistics for calories (mean, median, standard deviation, IQR) for food items and drink items.

```
food_data %>%
  summarise(mean = mean(Calories),
            median = median(Calories),
            std_dev = sd(Calories),
            iqr = IQR(Calories))
```

```
##       mean median  std_dev iqr
## 1 462.0909    445 249.3343 210
```

```
bev_data %>%
  summarise(mean = mean(Calories),
            median = median(Calories),
            std_dev = sd(Calories),
            iqr = IQR(Calories))
```

```
##       mean median  std_dev iqr
## 1 299.4667    270 208.8215 235
```

Regression

Model Selection with AIC

The code below creates a linear model for McDonalds food menu items and displays the model output.

```
food_model = lm(Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + Sodium + Carbohydrates
         Dietary.Fiber + Sugars + Protein + Vitamin.A....Daily.Value. +
         Vitamin.C....Daily.Value. + Calcium....Daily.Value. + Iron....Daily.Value., data = food_data)

tidy(food_model, conf.int = TRUE) %>%
  kable(format="markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-1.12936	1.19051	-0.94864	0.34519	-3.49250	1.23377
Total.Fat	8.90349	0.11010	80.86937	0.00000	8.68495	9.12203
Saturated.Fat	0.64499	0.27515	2.34419	0.02113	0.09883	1.19115
Trans.Fat	1.31130	1.52854	0.85788	0.39310	-1.72284	4.34544
Cholesterol	-0.00930	0.00546	-1.70272	0.09186	-0.02015	0.00154
Sodium	-0.00179	0.00297	-0.60505	0.54657	-0.00768	0.00409
Carbohydrates	4.12772	0.07185	57.45281	0.00000	3.98511	4.27033
Dietary.Fiber	-1.35003	0.51653	-2.61367	0.01040	-2.37533	-0.32473
Sugars	-0.06024	0.09780	-0.61593	0.53940	-0.25436	0.13389
Protein	3.97862	0.10811	36.80011	0.00000	3.76402	4.19323
Vitamin.A....Daily.Value.	0.01661	0.01587	1.04647	0.29797	-0.01489	0.04811
Vitamin.C....Daily.Value.	0.04194	0.01969	2.13042	0.03569	0.00286	0.08102
Calcium....Daily.Value.	-0.05409	0.08228	-0.65740	0.51250	-0.21743	0.10924
Iron....Daily.Value.	-0.06604	0.13469	-0.49029	0.62505	-0.33339	0.20132

The code below creates a linear model for McDonald's drink menu items and displays the model output.

```
bev_model = lm(Calories ~ Total.Fat + Sodium + Carbohydrates + Sugars + Protein + Vitamin.A....Daily.Val
         Vitamin.C....Daily.Value. + Calcium....Daily.Value. + Iron....Daily.Value., data = bev_data)

tidy(bev_model, conf.int = TRUE) %>%
  kable(format="markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-1.07350	0.94628	-1.13444	0.25855	-2.94435	0.79735
Total.Fat	9.04943	0.08198	110.38674	0.00000	8.88735	9.21151
Sodium	-0.05167	0.01732	-2.98320	0.00337	-0.08591	-0.01743
Carbohydrates	4.34711	0.11325	38.38434	0.00000	4.12321	4.57102
Sugars	-0.47748	0.11734	-4.06924	0.00008	-0.70946	-0.24549
Protein	3.79427	0.56608	6.70269	0.00000	2.67510	4.91345
Vitamin.A....Daily.Value.	0.15757	0.07876	2.00069	0.04736	0.00186	0.31328
Vitamin.C....Daily.Value.	0.04432	0.01808	2.45206	0.01543	0.00859	0.08006
Calcium....Daily.Value.	0.26133	0.15820	1.65187	0.10080	-0.05145	0.57411
Iron....Daily.Value.	0.76536	0.21183	3.61317	0.00042	0.34657	1.18416

Now, we will select a model for food items using AIC. We are using the step function in R to conduct backward selection using AIC as the selection criterion, and storing the selected model as food_model_select_aic. Finally, we display the coefficients of the selected model.

```r
food_model_select_aic <- step(food_model, direction = "backward")
```

```
## Start:  AIC=325.07
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + 
##     Sodium + Carbohydrates + Dietary.Fiber + Sugars + Protein + 
##     Vitamin.A....Daily.Value. + Vitamin.C....Daily.Value. + Calcium....Daily.Value. + 
##     Iron....Daily.Value.
## 
##                             Df Sum of Sq    RSS    AIC
## - Iron....Daily.Value.       1         4   1642 323.34
## - Sodium                     1         6   1644 323.48
## - Sugars                     1         6   1644 323.50
## - Calcium....Daily.Value.    1         7   1645 323.56
## - Trans.Fat                  1        13   1650 323.91
## - Vitamin.A....Daily.Value.  1        19   1656 324.31
## <none>                                     1638 325.07
## - Cholesterol                1        49   1687 326.34
## - Vitamin.C....Daily.Value.  1        77   1715 328.15
## - Saturated.Fat              1        94   1731 329.19
## - Dietary.Fiber              1       117   1754 330.63
## - Protein                    1     23103  24741 621.73
## - Carbohydrates              1     56311  57949 715.35
## - Total.Fat                  1    111569 113206 789.01
## 
## Step:  AIC=323.34
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + 
##     Sodium + Carbohydrates + Dietary.Fiber + Sugars + Protein + 
##     Vitamin.A....Daily.Value. + Vitamin.C....Daily.Value. + Calcium....Daily.Value.
## 
##                             Df Sum of Sq    RSS    AIC
## - Sugars                     1         4   1646 321.59
## - Calcium....Daily.Value.    1         6   1648 321.74
## - Sodium                     1         6   1648 321.77
## - Trans.Fat                  1         9   1650 321.91
## - Vitamin.A....Daily.Value.  1        17   1659 322.51
## <none>                                     1642 323.34
## - Cholesterol                1        81   1723 326.64
## - Vitamin.C....Daily.Value.  1        89   1731 327.13
## - Saturated.Fat              1        96   1738 327.58
## - Dietary.Fiber              1       125   1767 329.40
## - Protein                    1     25403  27045 629.53
## - Carbohydrates              1     76026  77668 745.57
## - Total.Fat                  1    178420 180061 838.06
## 
## Step:  AIC=321.59
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + 
##     Sodium + Carbohydrates + Dietary.Fiber + Protein + Vitamin.A....Daily.Value. + 
##     Vitamin.C....Daily.Value. + Calcium....Daily.Value.
## 
##                             Df Sum of Sq    RSS    AIC
## - Sodium                     1         3   1649 319.79
## - Calcium....Daily.Value.    1        10   1655 320.24
## - Trans.Fat                  1        13   1659 320.47
## - Vitamin.A....Daily.Value.  1        14   1660 320.52
## <none>                                     1646 321.59
## - Cholesterol                1        77   1723 324.64
## - Vitamin.C....Daily.Value.  1        86   1731 325.18
## - Saturated.Fat              1        99   1745 326.04
## - Dietary.Fiber              1       145   1791 328.88
## - Protein                    1     27755  29400 636.71
## - Total.Fat                  1    194023 195668 845.21
## - Carbohydrates              1    224130 225775 860.95
##
## Step:  AIC=319.79
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol +
##     Carbohydrates + Dietary.Fiber + Protein + Vitamin.A....Daily.Value. +
##     Vitamin.C....Daily.Value. + Calcium....Daily.Value.
##
##                             Df Sum of Sq    RSS    AIC
## - Calcium....Daily.Value.    1         9   1657 318.38
## - Vitamin.A....Daily.Value.  1        15   1663 318.77
## - Trans.Fat                  1        27   1675 319.56
## <none>                                     1649 319.79
## - Cholesterol                1        75   1723 322.66
## - Vitamin.C....Daily.Value.  1        93   1741 323.82
## - Saturated.Fat              1       111   1759 324.95
## - Dietary.Fiber              1       144   1793 327.01
## - Protein                    1     55059  56707 706.97
## - Total.Fat                  1    195271 196920 843.91
## - Carbohydrates              1    235929 237577 864.55
##
## Step:  AIC=318.38
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol +
##     Carbohydrates + Dietary.Fiber + Protein + Vitamin.A....Daily.Value. +
##     Vitamin.C....Daily.Value.
##
##                             Df Sum of Sq    RSS    AIC
## - Vitamin.A....Daily.Value.  1        16   1673 317.43
## - Trans.Fat                  1        20   1678 317.72
## <none>                                     1657 318.38
## - Cholesterol                1        95   1753 322.53
## - Saturated.Fat              1       103   1760 322.99
## - Vitamin.C....Daily.Value.  1       112   1770 323.60
## - Dietary.Fiber              1       163   1820 326.69
## - Protein                    1     74501  76159 737.41
## - Carbohydrates              1    257339 258996 872.05
## - Total.Fat                  1    325761 327418 897.84
##
## Step:  AIC=317.43
## Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol +
##     Carbohydrates + Dietary.Fiber + Protein + Vitamin.C....Daily.Value.
##
##                             Df Sum of Sq    RSS    AIC
## - Trans.Fat                  1        17   1690 316.54
## <none>                                     1673 317.43
## - Cholesterol                1        90   1763 321.18
## - Saturated.Fat              1        97   1770 321.63
## - Vitamin.C....Daily.Value.  1       107   1781 322.27
## - Dietary.Fiber              1       162   1836 325.62
## - Protein                    1     80913  82586 744.32
## - Carbohydrates              1    301587 303260 887.41
## - Total.Fat                  1    334931 336605 898.88
##
## Step:  AIC=316.54
## Calories ~ Total.Fat + Saturated.Fat + Cholesterol + Carbohydrates + 
##     Dietary.Fiber + Protein + Vitamin.C....Daily.Value.
##
##                             Df Sum of Sq    RSS    AIC
## <none>                                     1690 316.54
## - Cholesterol                1       108   1798 321.34
## - Vitamin.C....Daily.Value.  1       124   1814 322.30
## - Saturated.Fat              1       164   1854 324.72
## - Dietary.Fiber              1       175   1866 325.39
## - Protein                    1    104647 106337 770.13
## - Carbohydrates              1    311255 312945 888.86
## - Total.Fat                  1    353515 355205 902.80
```

```
tidy(food_model_select_aic) %>%
  kable(format="markdown", digits=3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-1.396	1.066	-1.310	0.193
Total.Fat	8.957	0.061	146.055	0.000
Saturated.Fat	0.500	0.159	3.144	0.002
Cholesterol	-0.011	0.004	-2.550	0.012
Carbohydrates	4.057	0.030	137.048	0.000
Dietary.Fiber	-1.112	0.342	-3.252	0.002
Protein	3.934	0.050	79.465	0.000
Vitamin.C....Daily.Value.	0.049	0.018	2.730	0.007

```
coef(food_model_select_aic, 4)
```

```
##               (Intercept)                 Total.Fat             Saturated.Fat 
##               -1.39581459                8.95707901                0.49969868 
##               Cholesterol             Carbohydrates             Dietary.Fiber 
##               -0.01110469                4.05708994               -1.11187306 
##                   Protein Vitamin.C....Daily.Value. 
##                3.93430774                0.04881962
```

For food products, the predictors that give us the best model for predicting calorie count are: -Total.Fat -Saturated.Fat -Trans.Fat -Cholesterol -Sodium -Carbohydrates -Dietary.Fiber -Sugars -Protein -Vitamin.A....Daily.Value. -Vitamin.C....Daily.Value. -Calcium....Daily.Value. -Iron....Daily.Value.

Now, we will select a model for drink items using AIC. We are using the step function in R to conduct backward selection using AIC as the selection criterion, and storing the selected model as bev_model_select_aic. Finally, we display the coefficients of the selected model.

```
bev_model_select_aic <- step(bev_model, direction = "backward")
```

```
## Start:  AIC=505.72
## Calories ~ Total.Fat + Sodium + Carbohydrates + Sugars + Protein + 
##     Vitamin.A....Daily.Value. + Vitamin.C....Daily.Value. + Calcium....Daily.Value. + 
##     Iron....Daily.Value.
## 
##                             Df Sum of Sq    RSS     AIC
## <none>                                     3823  505.72
## - Calcium....Daily.Value.    1        75   3898  506.62
## - Vitamin.A....Daily.Value.  1       109   3932  507.95
## - Vitamin.C....Daily.Value.  1       164   3987  510.03
## - Sodium                     1       243   4066  512.97
## - Iron....Daily.Value.       1       356   4180  517.10
## - Sugars                     1       452   4275  520.49
## - Protein                    1      1227   5050  545.47
## - Carbohydrates              1     40233  44056  870.39
## - Total.Fat                  1    332745 336568 1175.39
```

```
tidy(bev_model_select_aic) %>%
  kable(format="markdown", digits=3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-1.074	0.946	-1.134	0.259
Total.Fat	9.049	0.082	110.387	0.000
Sodium	-0.052	0.017	-2.983	0.003
Carbohydrates	4.347	0.113	38.384	0.000
Sugars	-0.477	0.117	-4.069	0.000
Protein	3.794	0.566	6.703	0.000
Vitamin.A....Daily.Value.	0.158	0.079	2.001	0.047
Vitamin.C....Daily.Value.	0.044	0.018	2.452	0.015
Calcium....Daily.Value.	0.261	0.158	1.652	0.101
Iron....Daily.Value.	0.765	0.212	3.613	0.000

```
coef(bev_model_select_aic, 4)
```

```
##               (Intercept)                 Total.Fat                    Sodium 
##               -1.07350124                9.04943065               -0.05166998 
##             Carbohydrates                    Sugars                   Protein 
##                4.34711487               -0.47747707                3.79427294 
## Vitamin.A....Daily.Value. Vitamin.C....Daily.Value.   Calcium....Daily.Value. 
##                0.15757301                0.04432252                0.26133357 
##      Iron....Daily.Value. 
##                0.76536478
```

For Beverages, the predictors that give us the best model for predicting calorie count are: -Total.Fat -Sodium -Carbohydrates -Sugars -Protein -Vitamin.A....Daily.Value. -Vitamin.C....Daily.Value. -Calcium....Daily.Value. -Iron....Daily.Value.

The code and its output below show us that the best predictors of food calories in order are Total Fat, Carbohydrates, and Protein.

```
food_models <- regsubsets(Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + Sodium + Car
         Dietary.Fiber + Sugars + Protein + Vitamin.A....Daily.Value. +
         Vitamin.C....Daily.Value. + Calcium....Daily.Value. + Iron....Daily.Value., data = food_data,
summary(food_models)
```

```
## Subset selection object
## Call: regsubsets.formula(Calories ~ Total.Fat + Saturated.Fat + Trans.Fat +
##     Cholesterol + Sodium + Carbohydrates + Dietary.Fiber + Sugars +
##     Protein + Vitamin.A....Daily.Value. + Vitamin.C....Daily.Value. +
##     Calcium....Daily.Value. + Iron....Daily.Value., data = food_data,
##     method = "backward")
## 13 Variables  (and intercept)
##                           Forced in Forced out
## Total.Fat                     FALSE      FALSE
## Saturated.Fat                 FALSE      FALSE
## Trans.Fat                     FALSE      FALSE
## Cholesterol                   FALSE      FALSE
## Sodium                        FALSE      FALSE
## Carbohydrates                 FALSE      FALSE
## Dietary.Fiber                 FALSE      FALSE
## Sugars                        FALSE      FALSE
## Protein                       FALSE      FALSE
## Vitamin.A....Daily.Value.     FALSE      FALSE
## Vitamin.C....Daily.Value.     FALSE      FALSE
## Calcium....Daily.Value.       FALSE      FALSE
## Iron....Daily.Value.          FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: backward
##          Total.Fat Saturated.Fat Trans.Fat Cholesterol Sodium Carbohydrates
## 1  ( 1 ) "*"       " "           " "       " "         " "    " "          
## 2  ( 1 ) "*"       " "           " "       " "         " "    "*"          
## 3  ( 1 ) "*"       " "           " "       " "         " "    "*"          
## 4  ( 1 ) "*"       " "           " "       " "         " "    "*"          
## 5  ( 1 ) "*"       " "           " "       " "         " "    "*"          
## 6  ( 1 ) "*"       "*"           " "       " "         " "    "*"          
## 7  ( 1 ) "*"       "*"           " "       "*"         " "    "*"          
## 8  ( 1 ) "*"       "*"           "*"       "*"         " "    "*"          
##          Dietary.Fiber Sugars Protein Vitamin.A....Daily.Value.
## 1  ( 1 ) " "           " "    " "     " "                      
## 2  ( 1 ) " "           " "    " "     " "                      
## 3  ( 1 ) " "           " "    "*"     " "                      
## 4  ( 1 ) "*"           " "    "*"     " "                      
## 5  ( 1 ) "*"           " "    "*"     " "                      
## 6  ( 1 ) "*"           " "    "*"     " "                      
## 7  ( 1 ) "*"           " "    "*"     " "                      
## 8  ( 1 ) "*"           " "    "*"     " "                      
##          Vitamin.C....Daily.Value. Calcium....Daily.Value. Iron....Daily.Value.
## 1  ( 1 ) " "                       " "                     " "                 
## 2  ( 1 ) " "                       " "                     " "                 
## 3  ( 1 ) " "                       " "                     " "                 
## 4  ( 1 ) " "                       " "                     " "                 
## 5  ( 1 ) "*"                       " "                     " "                 
## 6  ( 1 ) "*"                       " "                     " "                 
## 7  ( 1 ) "*"                      " "                     " "
## 8  ( 1 ) "*"                      " "                     " "
```

The code and its output below show us that the best predictors of beverage calories in order are Carbohydrates, Total Fat, and Protein

```
bev_models <- regsubsets(Calories ~ Total.Fat + Saturated.Fat + Trans.Fat + Cholesterol + Sodium + Carbo
         Dietary.Fiber + Sugars + Protein + Vitamin.A....Daily.Value. +
         Vitamin.C....Daily.Value. + Calcium....Daily.Value. + Iron....Daily.Value., data = bev_data, me
summary(bev_models)
```

```
## Subset selection object
## Call: regsubsets.formula(Calories ~ Total.Fat + Saturated.Fat + Trans.Fat +
##     Cholesterol + Sodium + Carbohydrates + Dietary.Fiber + Sugars +
##     Protein + Vitamin.A....Daily.Value. + Vitamin.C....Daily.Value. +
##     Calcium....Daily.Value. + Iron....Daily.Value., data = bev_data,
##     method = "backward")
## 13 Variables  (and intercept)
##                           Forced in Forced out
## Total.Fat                     FALSE      FALSE
## Saturated.Fat                 FALSE      FALSE
## Trans.Fat                     FALSE      FALSE
## Cholesterol                   FALSE      FALSE
## Sodium                        FALSE      FALSE
## Carbohydrates                 FALSE      FALSE
## Dietary.Fiber                 FALSE      FALSE
## Sugars                        FALSE      FALSE
## Protein                       FALSE      FALSE
## Vitamin.A....Daily.Value.     FALSE      FALSE
## Vitamin.C....Daily.Value.     FALSE      FALSE
## Calcium....Daily.Value.       FALSE      FALSE
## Iron....Daily.Value.          FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: backward
##          Total.Fat Saturated.Fat Trans.Fat Cholesterol Sodium Carbohydrates
## 1  ( 1 ) " "       " "           " "       " "         " "    "*"
## 2  ( 1 ) "*"       " "           " "       " "         " "    "*"
## 3  ( 1 ) "*"       " "           " "       " "         " "    "*"
## 4  ( 1 ) "*"       "*"           " "       " "         " "    "*"
## 5  ( 1 ) "*"       "*"           " "       " "         " "    "*"
## 6  ( 1 ) "*"       "*"           " "       " "         " "    "*"
## 7  ( 1 ) "*"       "*"           " "       " "         " "    "*"
## 8  ( 1 ) "*"       "*"           " "       " "         " "    "*"
##          Dietary.Fiber Sugars Protein Vitamin.A....Daily.Value.
## 1  ( 1 ) " "           " "    " "     " "
## 2  ( 1 ) " "           " "    " "     " "
## 3  ( 1 ) " "           " "    "*"     " "
## 4  ( 1 ) " "           " "    "*"     " "
## 5  ( 1 ) " "           " "    "*"     " "
## 6  ( 1 ) "*"           " "    "*"     " "
## 7  ( 1 ) "*"           "*"    "*"     " "
## 8  ( 1 ) "*"           "*"    "*"     " "
##          Vitamin.C....Daily.Value. Calcium....Daily.Value. Iron....Daily.Value.
## 1  ( 1 ) " "                        " "                       " "
## 2  ( 1 ) " "                        " "                       " "
## 3  ( 1 ) " "                        " "                       " "
## 4  ( 1 ) " "                        " "                       " "
## 5  ( 1 ) "*"                        " "                       " "
## 6  ( 1 ) "*"                        " "                       " "
## 7  ( 1 ) "*"                        " "                       " "
## 8  ( 1 ) "*"                        "*"                       " "
```

Because both foods and beverages have the same variables marked as their best predictors, we can conclude that splitting our dataset is unnecesary. Working with the full dataset at once will allow us to make better predictions, so we will recombine our data.

Model Selection with K-fold Cross Validoation

We will be comparing the predictors mentioned above regarding their relationship to Calories. For food and beverages, we will create four models each: one with each variable as the singular predictor. We will compare these models using K-fold Cross Validation, where k = 5.

```
set.seed(5747108)
folded_data <- crossv_kfold(data, 5)
```

Testing the Total Fat predictor:

```
models_fat <- map(folded_data$train,
              ~ lm(Calories ~ Total.Fat, data = .))
models_fat
```

```
## $`1`
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = .)
##
## Coefficients:
## (Intercept)    Total.Fat
##      148.30        15.38
##
##
## $`2`
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = .)
##
## Coefficients:
## (Intercept)    Total.Fat
##      150.84        15.63
##
##
## $`3`
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = .)
```

```
##
## Coefficients:
## (Intercept)    Total.Fat
##       148.4         15.5
##
##
## $`4`
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = .)
##
## Coefficients:
## (Intercept)    Total.Fat
##      154.00        14.92
##
##
## $`5`
##
## Call:
## lm(formula = Calories ~ Total.Fat, data = .)
##
## Coefficients:
## (Intercept)    Total.Fat
##      155.61        15.11
```

```r
train_mse_fat <- map2_dbl(models_fat, folded_data$train, mse)
test_mse_fat <- map2_dbl(models_fat, folded_data$test, mse)

fat <- tibble(
  1:5,
  train_mse_fat,
  test_mse_fat
)

fat %>%
  summarise(mean_train_mse = mean(train_mse_fat),
           mean_test_mse = mean(test_mse_fat))
```

```
## # A tibble: 1 x 2
##   mean_train_mse mean_test_mse
##            <dbl>         <dbl>
## 1         10449.        10645.
```

Testing the Carbohydrates predictor:

```r
models_carb <- map(folded_data$train,
               ~ lm(Calories ~ Carbohydrates, data = .))
models_carb
```

```
## $`1`
##
## Call:
## lm(formula = Calories ~ Carbohydrates, data = .)
```

```
##
## Coefficients:
##   (Intercept)  Carbohydrates
##        44.120          6.929
##
##
## $`2`
##
## Call:
## lm(formula = Calories ~ Carbohydrates, data = .)
##
## Coefficients:
##   (Intercept)  Carbohydrates
##        53.117          6.638
##
##
## $`3`
##
## Call:
## lm(formula = Calories ~ Carbohydrates, data = .)
##
## Coefficients:
##   (Intercept)  Carbohydrates
##        63.401          6.376
##
##
## $`4`
##
## Call:
## lm(formula = Calories ~ Carbohydrates, data = .)
##
## Coefficients:
##   (Intercept)  Carbohydrates
##         55.06           6.63
##
##
## $`5`
##
## Call:
## lm(formula = Calories ~ Carbohydrates, data = .)
##
## Coefficients:
##   (Intercept)  Carbohydrates
##        52.474          6.656
```

```
train_mse_carb <- map2_dbl(models_carb, folded_data$train, mse)
test_mse_carb <- map2_dbl(models_carb, folded_data$test, mse)

carb <- tibble(
  test_fold = 1:5,
  train_mse_carb,
  test_mse_carb
)
```

```
carb %>%
  summarise(mean_train_mse = mean(train_mse_carb),
            mean_test_mse = mean(test_mse_carb))
```

```
## # A tibble: 1 x 2
##   mean_train_mse mean_test_mse
##            <dbl>         <dbl>
## 1         22353.        22643.
```

Testing the Protein predictor:

```
models_protein <- map(folded_data$train,
              ~ lm(Calories ~ Protein, data = .))
models_protein
```

```
## $`1`
##
## Call:
## lm(formula = Calories ~ Protein, data = .)
##
## Coefficients:
## (Intercept)      Protein
##      141.92        16.76
##
##
## $`2`
##
## Call:
## lm(formula = Calories ~ Protein, data = .)
##
## Coefficients:
## (Intercept)      Protein
##       153.2         16.5
##
##
## $`3`
##
## Call:
## lm(formula = Calories ~ Protein, data = .)
##
## Coefficients:
## (Intercept)      Protein
##      158.65        15.74
##
##
## $`4`
##
## Call:
## lm(formula = Calories ~ Protein, data = .)
##
## Coefficients:
## (Intercept)      Protein
##      136.08        16.63
##
##
## $‘5‘
##
## Call:
## lm(formula = Calories ~ Protein, data = .)
##
## Coefficients:
## (Intercept)      Protein
##      147.98        17.09
```

```
train_mse_protein <- map2_dbl(models_protein, folded_data$train, mse)
test_mse_protein <- map2_dbl(models_protein, folded_data$test, mse)

protein <- tibble(
  1:5,
  train_mse_protein,
  test_mse_protein
)

protein %>%
  summarise(mean_train_mse = mean(train_mse_protein),
           mean_test_mse = mean(test_mse_protein))
```

```
## # A tibble: 1 x 2
##   mean_train_mse mean_test_mse
##            <dbl>         <dbl>
## 1         21748.        22399.
```

From our K-fold cross validation test, we can conclude that Total Fat is the best singular predictor of Calories, which goes against our hypothesis. We see that Protein and Carbohydrates have similar effectiveness in predicting calories, but total fat content is about twice as effective.

Final Model

```
final_model <- lm(Calories ~ Total.Fat, data=data)

tidy(final_model) %>%
  kable(format="markdown", digits = 5)
```

term	estimate	std.error	statistic	p.value
(Intercept)	151.58819	9.00436	16.83498	0
Total.Fat	15.29652	0.44927	34.04760	0

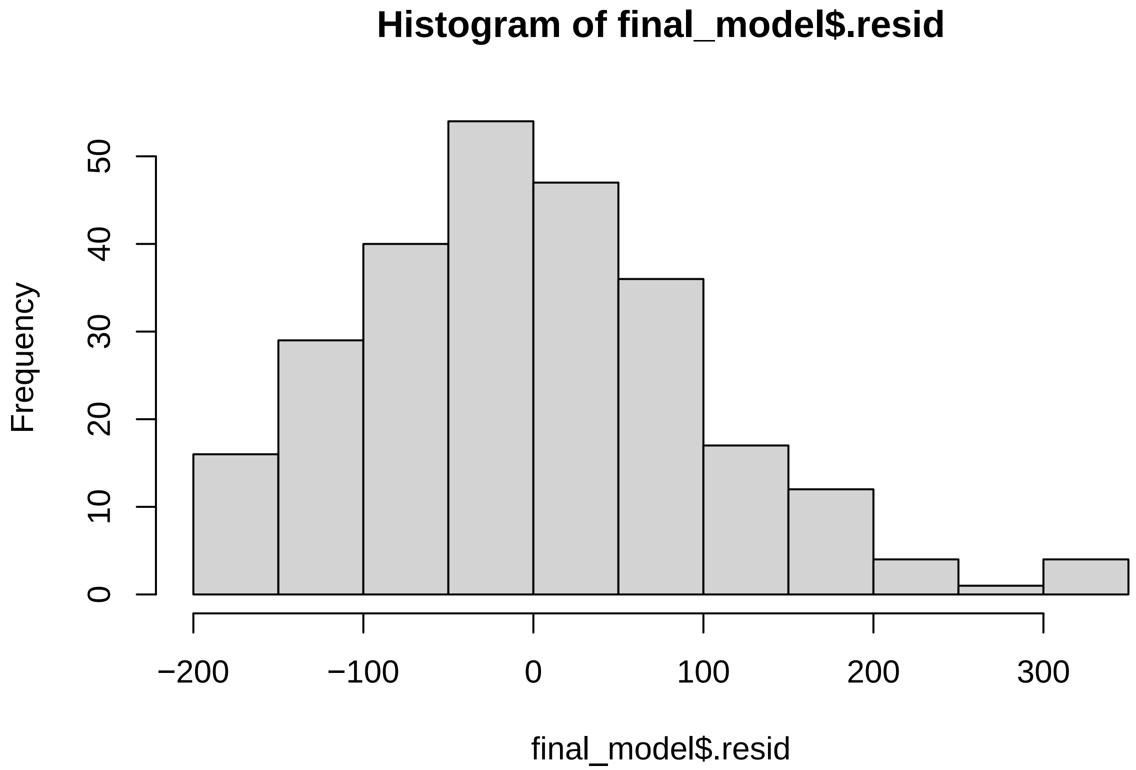
Checking the Linearity Assumption

```
final_model <- augment(final_model, type.predict = "response",type.residuals = "deviance")
glimpse(final_model)
```

```
## Rows: 260
## Columns: 8
## $ Calories   <int> 300, 250, 370, 450, 400, 430, 460, 520, 410, 470, 430, 480,~
## $ Total.Fat  <dbl> 13, 8, 23, 28, 23, 23, 26, 30, 20, 25, 27, 31, 33, 37, 27, ~
## $ .fitted    <dbl> 350.4429, 273.9603, 503.4081, 579.8907, 503.4081, 503.4081,~
## $ .resid     <dbl> -50.442906, -23.960322, -133.408072, -129.890655, -103.4080~
## $ .hat       <dbl> 0.003872137, 0.004573393, 0.005339402, 0.007507923, 0.00533~
## $ .sigma     <dbl> 102.8649, 102.9023, 102.5743, 102.5913, 102.7097, 102.8107,~
## $ .cooksd    <dbl> 4.705822e-04, 1.255799e-04, 4.552204e-03, 6.094474e-03, 2.7~
## $ .std.resid <dbl> -0.49205642, -0.23380855, -1.30231788, -1.26936563, -1.0094~
```

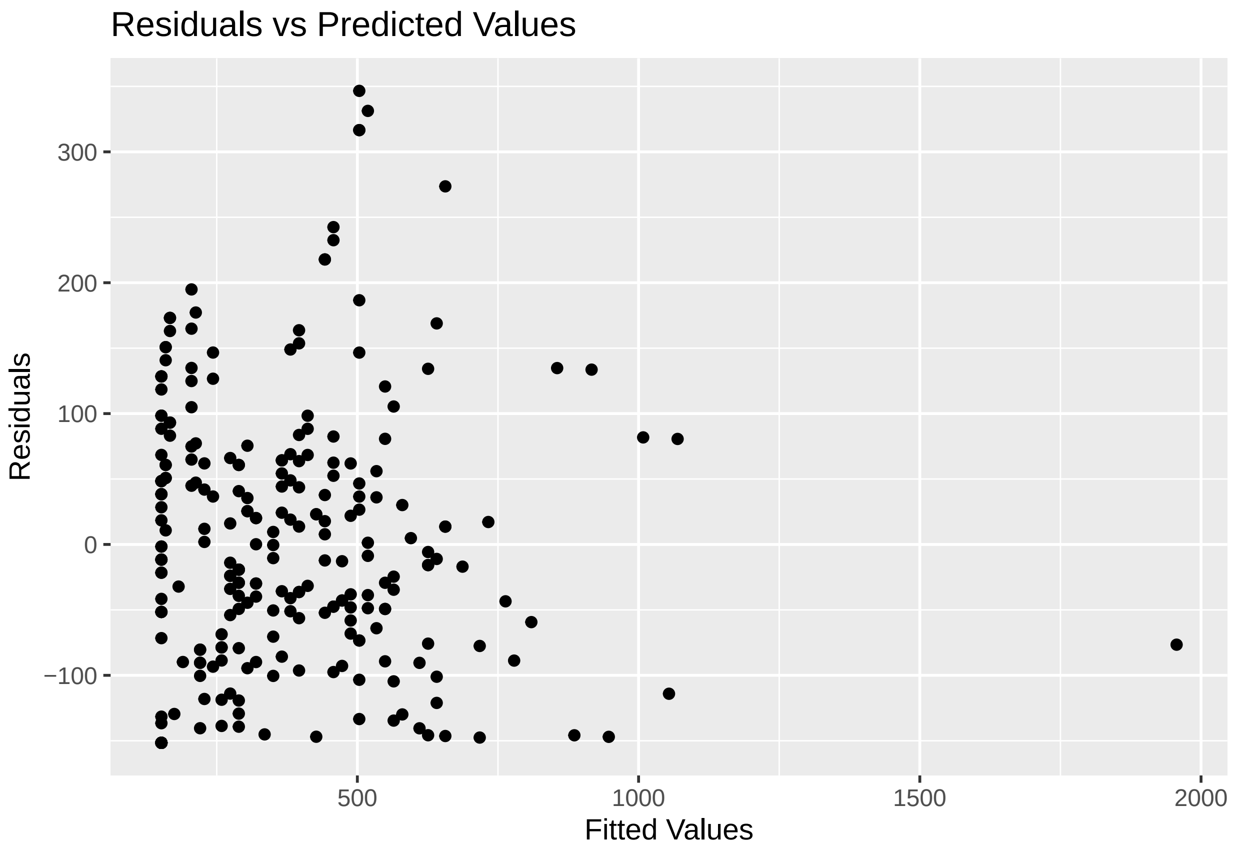
Based on the plot below, the normality assumption is satisfied because our model's residuals follow a normal distribution with mean zero.

```
hist(final_model$.resid)
```



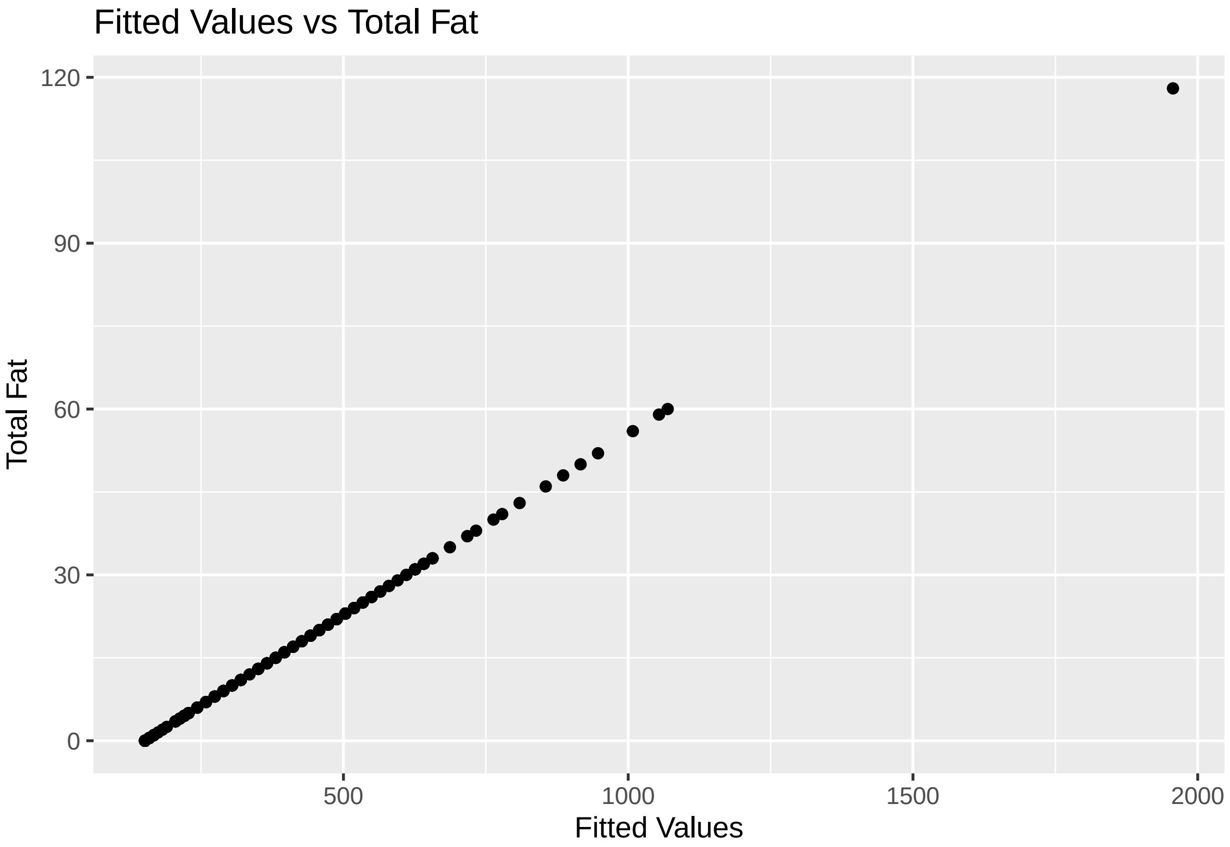
Based on the plot below, the constant variance assumption is satisfied because there is no fan pattern when comparing our fitted values to our residuals.

```
ggplot(data = final_model, aes(x=.fitted, y=.resid)) +
  geom_point() +
  labs(title="Residuals vs Predicted Values",
       x="Fitted Values",
       y="Residuals")
```



Based on the plot below, the linearity assumption is satisfied because there is a very clear linear relationship between our fitted values and our predictor.

```
ggplot(data = final_model, aes(x=.fitted, y=Total.Fat)) +
  geom_point() +
  labs(title="Fitted Values vs Total Fat",
       x="Fitted Values",
       y="Total Fat")
```


Fitted Values vs Total Fat
120
90
60
30
0
Total Fat
500
1000
1500
2000
Fitted Values