Project 2

GENERAL INSTRUCTIONS:

this is NOT a group project

- **CLEARLY** mark where you are answering each question (all written questions must be answered in Markdown cells, NOT as comments in code cells)
- Show all code necessary for the analysis, but remove superfluous code

---

DONUTS

Using the dataset *krispykreme.csv*,

- **a)** make 3 scatterplots using ggplot to show:
  - `Sodium_100g` vs `Total_Fat_100g`
  - `Sodium_100g` vs. `Sugar_100g`
  - `Sugar_100g` vs `Total_Fat_100g`

  You will be graded on the effectiveness of the graphs as well as their content.

- **b)** Using the scatterplots from part **a** as well as the donuts dataset, **thouroughly discuss which clustering method** (KMeans, Gaussian Mixture Models (EM), Hierarchical Clustering, or DBSCAN) **you think would be best for this data and WHY**. Be sure to include discussions of assumptions each algorithm does/does not make, and what types of data they are good/bad for (**mention each of the 4 algorithms at least once**) and how they apply to this specific dataset. (*IN A MARKDOWN CELL*)
  - you should be making statements that both 1) discuss characteristics of the algorithm and 2) specifically discuss how that characteristic applies (or doesn't) to this dataset.

Please note that for this assignment, "It's easier to code" does not count as a valid reason. The reasons should be based on the algorithms/data.

(You must use "**" to make any mention of one of the algorithms bold in your discussion. For example "I think **DBSCAN** is the best algorithm ever!" will make the word "**DBSCAN**" bold in a Markdown cell).

- **c) Implement the TWO algorithms** you think will work BEST (1 algo) and WORST (1 algo) here using all 3 variables `Sodium_100g`, `Total_Fat_100g` and `Sugar_100g`, and describe **how you chose any hyperparameters** (such as distance, # of clusters, min_samples, eps, linkage...etc). Make sure to z-score your variables. (*IN A MARKDOWN CELL*)

- **d) Thouroughly discuss the performance** of your clustering models. For each algorithm (best and worst):
  - which metric did you use to asses your model? (*IN A MARKDOWN CELL*)
  - how did your model perform? (*IN A MARKDOWN CELL*)
  - remake the 3 graphs from part a, but color by cluster assignment. Describe what characterizes each cluster, and give an example of a label for that cluster (e.g. "these donuts are low fat, and low sugar so I would call these healthy donuts") (*IN A MARKDOWN CELL*)
- **e)** Choose ONE other of the `_100g` variables from the data set to **add to your clustering model** to improve it.
  - explain why you chose this variable. Either based on improvement in metrics, or outside knowledge you have about food/donuts (*IN A MARKDOWN CELL*)
  - make a new model, identical to the model you thought would be best in part c, but also including your new variable.
  - did this variable improve the fit of your clustering model? How can you tell? (*IN A MARKDOWN CELL*)

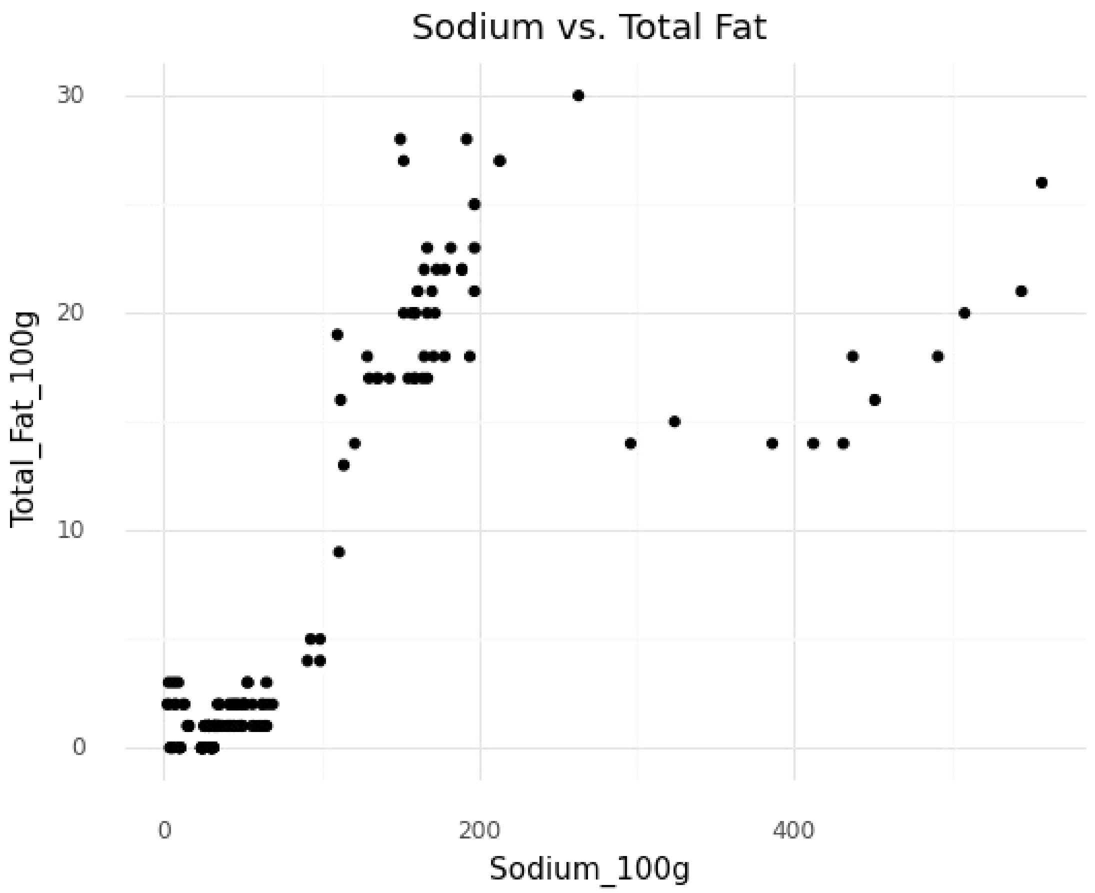
Note: The columns with `_100g` at the end represent the amount of that nutrient per 100 grams of the food. For example, `Total_Fat` tells you the total amount of fat in that food, whereas `Total_Fat_100g` tells you how much fat there is per 100 grams of that food.

```
# import necessary packages
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
from plotnine import *

from sklearn.preprocessing import StandardScaler
from sklearn.cluster import AgglomerativeClustering
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
from sklearn.metrics import silhouette_score

import scipy.cluster.hierarchy as sch
from matplotlib import pyplot as plt

%matplotlib inline

# load data set
donuts =
pd.read_csv("https://raw.githubusercontent.com/cmparlettpelleriti/
CPSC392ParlettPelleriti/master/Data/KrispyKreme.csv")

# look at variables
donuts.head()

                              Restaurant_Item_Name    restaurant  \
0                       Krispy Kreme Apple Fritter  Krispy Kreme
1          Krispy Kreme Chocolate Iced Cake Doughnut  Krispy Kreme
2  Krispy Kreme Chocolate Iced Custard Filled Dou...  Krispy Kreme
3        Krispy Kreme Chocolate Iced Glazed Doughnut  Krispy Kreme
4  Krispy Kreme Chocolate Iced Glazed Cruller Dou...  Krispy Kreme

   Restaurant_ID                              Item_Name  \
0             49                          Apple Fritter
1             49           Chocolate Iced Cake Doughnut
2             49  Chocolate Iced Custard Filled Doughnut
3             49         Chocolate Iced Glazed Doughnut
4             49  Chocolate Iced Glazed Cruller Doughnut

                                 Item_Description Food_Category  \
0                         Apple Fritter, Doughnuts   Baked Goods
1           Chocolate Iced Cake Doughnut, Doughnuts   Baked Goods
2  Chocolate Iced Custard Filled Doughnut, Doughnuts   Baked Goods
3          Chocolate Iced Glazed Doughnut, Doughnuts   Baked Goods
4  Chocolate Iced Glazed Cruller Doughnut, Doughnuts   Baked Goods

   Serving_Size  Serving_Size_text Serving_Size_Unit
Serving_Size_household  \
0           100                NaN                 g
NaN
1            71                NaN                 g
NaN
2            85                NaN                 g
NaN
3            63                NaN                 g
NaN
4            70                NaN                 g
NaN

   ...  Total_Fat_100g  Saturated_Fat_100g  Trans_Fat_100g
Cholesterol_100g  \
0  ...              19                   9               0
0
1  ...              18                   7               0
35
2  ...              18                   8               0
0
3  ...              17                   8               0
0
```

```
4  ...              14                    6               0
29

   Sodium_100g  Potassium_100g  Carbohydrates_100g  Protein_100g
Sugar_100g  \
0          110            45.0                  42             4
26
1          437            49.0                  52             4
27
2          165            59.0                  44             5
20
3          143            56.0                  52             5
32
4          386            29.0                  57             4
37

   Dietary_Fiber_100g
0                 1.0
1                 NaN
2                 1.0
3                 NaN
4                 NaN

[5 rows x 32 columns]
```

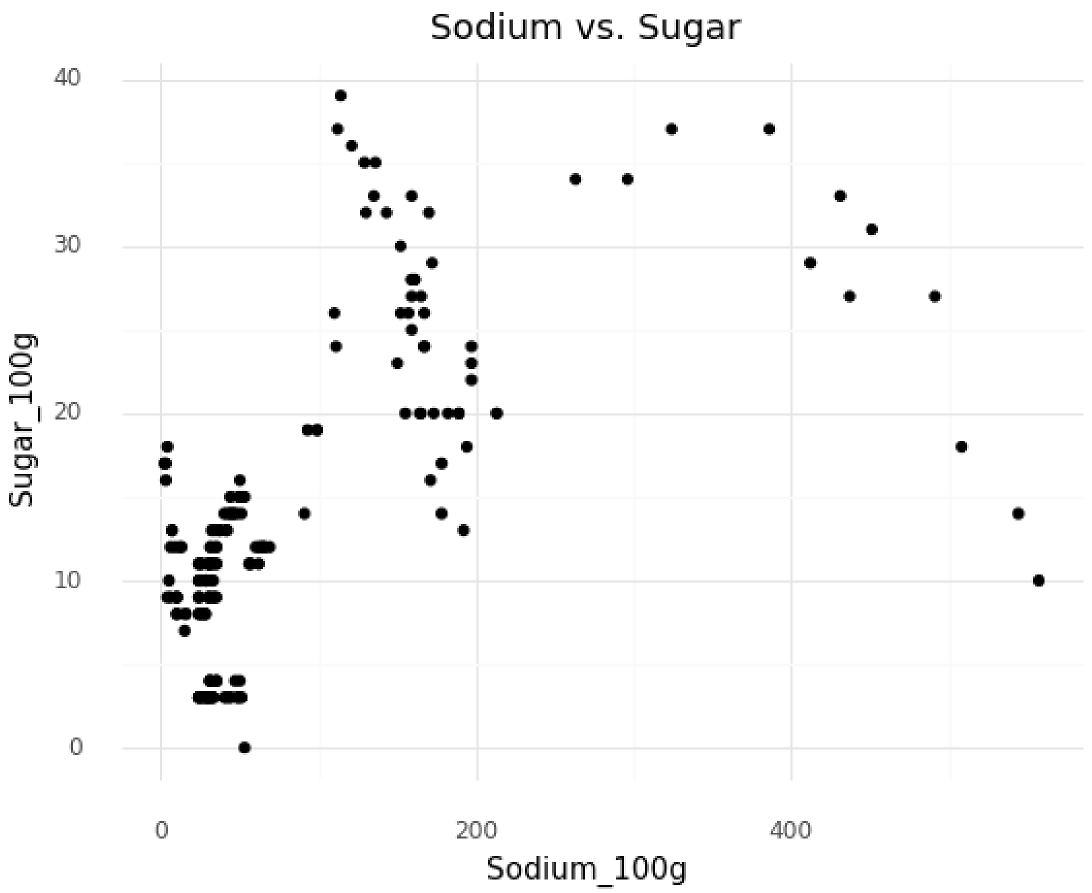
a) 3 scatterplots relating Sodium, Total Fat, and Sugar

```
# sodium vs total fat
(ggplot(donuts, aes("Sodium_100g","Total_Fat_100g")) +
 geom_point() + theme_minimal() +
 ggtitle("Sodium vs. Total Fat"))
```



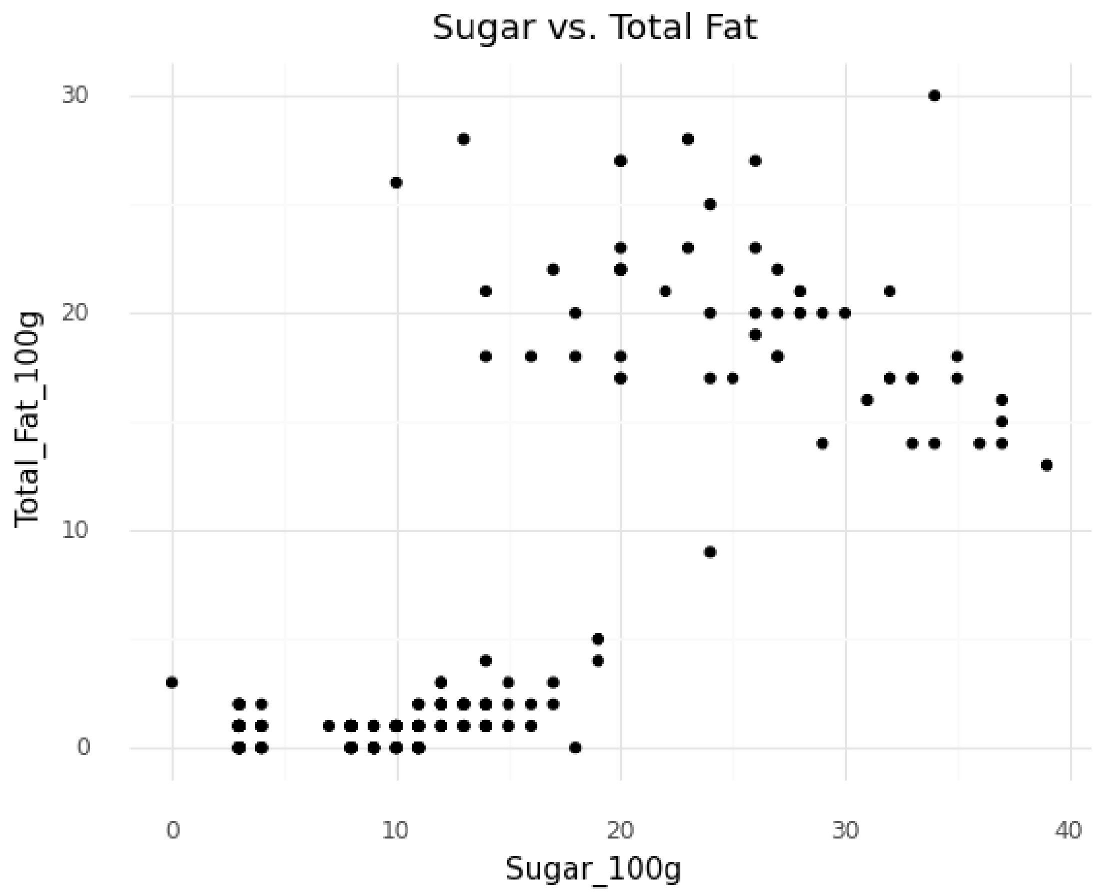
```
<ggplot: (8784218337273)>
```

```
# sodium vs sugar
(ggplot(donuts, aes("Sodium_100g","Sugar_100g")) +
 geom_point() + theme_minimal() +
 ggtitle("Sodium vs. Sugar"))
```



```
<ggplot: (8784217572393)>
```

```
# sugar vs total fat
(ggplot(donuts, aes("Sugar_100g","Total_Fat_100g")) +
 geom_point() + theme_minimal() +
 ggtitle("Sugar vs. Total Fat"))
```



```
<ggplot: (8784217528301)>
```

b) Choose the best clustering method

Using the scatterplots from above as well as the donuts dataset, I believe that out of the clustering method of **Hierarchical Clustering** would perform the best, compared to KMeans, Gaussian Mixture Models (EM), and DBSCAN.

KMeans - iterative algorithm that creates groups, or clusters based on variables of different values. The model will create k groups (we choose k random points to be cluster centers), and then for each point in the dataset it will assign it to the cluster whose center is closest. Using these assignments, it will recalculate the center and is an interative process of choosing centers, assigning points, until the clusters converge and there is little cluster membership and cluster center change. One of the big assumptions faced in this model is that the data will present itself in spherical clusters, which is important to consider when looking at the donut datasets and relationships between the variables we are evaluating. From a glace of the graph, none of the clusters appear to be very centered on a point with a spherical shape. Another key assumption of using KMeans is that the data set with have roughly the same amount of points in each cluster. On our graphs from a), there looks to be a lot of areas in the graphs that are much more concentrated (more points).

EM/Gaussian Mixture Models - similar to K-means, uses k random points of our choosing to be cluster centers, and then for each data point it calculates the probability of being in each cluster. Using these probabilities, it recalculates the means and variances and repeats this iterative process until the distributions converge. EM uses a soft (probabilistic) assignment and evaluates the probability each point has of being assigned to a specefic cluster. Because there is no hard assignment, the cluster centers/means and variances are calculated using EVERY data point weighted by the probability that the data point belongs to that cluster. An advantage compared to KMeans is that we don't assume Spherical variance within clusters, but instead clusters can be ellipses. However, the clusters we have in the graphs above are oddly shaped, and these groups of points don't fit into the ellipse shape.

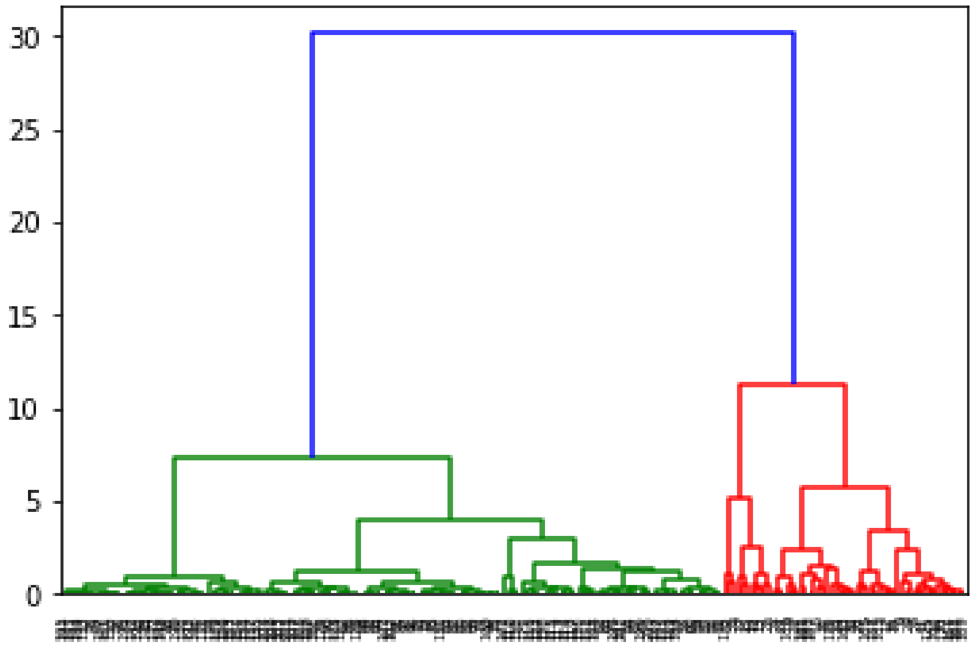
DBSCAN - stands for Density Based Spatial Clustering of Application with Noise, an iterative algorithm that chooses a random point and classifies it as a noise point or a border point based on # of neighbors. The main advantage of using DBSCAN is that it accounts for noise - data points that are outliers but in the graphs above noise does not seem to be a huge issue in the data set. Some other disadvantages if we used this model on the donut data set are that it is less effective in high dimensional data - which this data set has many variables. This algorithm is also bad at handling clusters that have different densities, which many of the clusters look to have different densities, some are more spread out around the center while other clusters are more concentrated.

Hierarchical Clustering - involves agglomerative clusters, or taking clusters and merging them together successfully. It builds a tree-like cluster and easily shows hierchical relationships graphically with a dendrogram. Using hierarchical clustering gives a lot of flexibility in choosing the number of clusters, and flexibility with lnkage criteria also. The donut dataset has many variables, a lot of them involved in subcategories and an overarching category: "Donuts". Hierchical clustering is the best to model a relationship involving variables that are of hierarchical nature. The downsides, which are less constrictatory than the algorithms previously mentioned for this specific situation, are a very slow runtime, and we cannot unmerge clusters once a dendrogram is produced. For these reasons, I believe **hierarchical clustering** is the best algorithm choice.

c) Implement the BEST and the WORST Algorithms

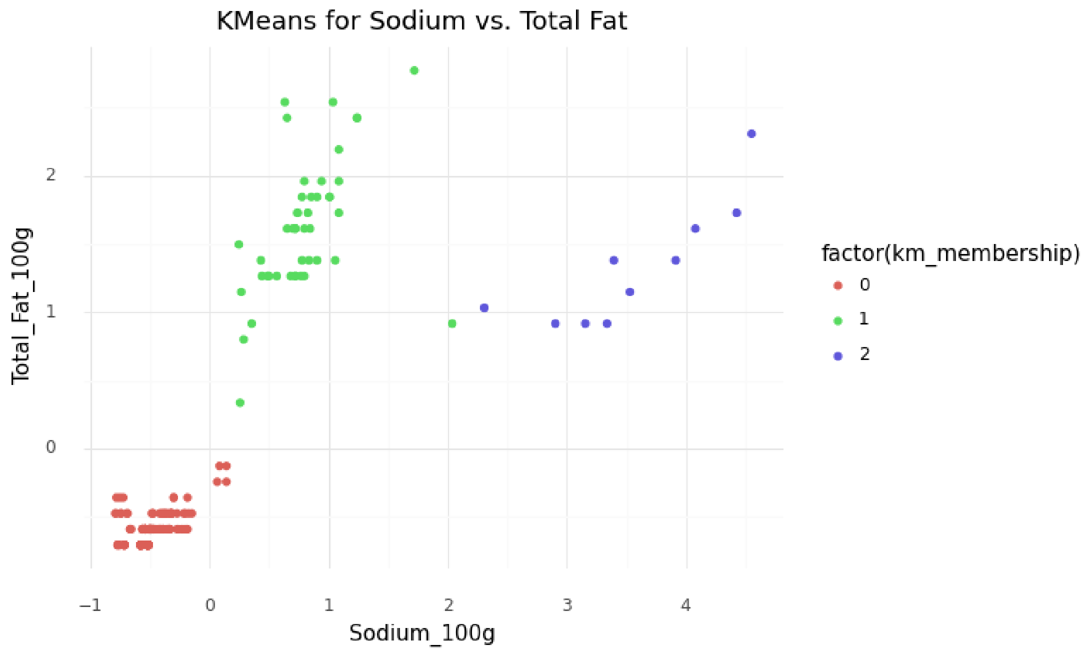
```
# get predictors (sodium, total fat, sugar)
predictors = ["Sodium_100g", "Total_Fat_100g", "Sugar_100g"]

# z-score variables
z = StandardScaler()
donuts[predictors] = z.fit_transform(donuts[predictors])

# Hierarchical Agglomerative Clustering
# create the model
hac = AgglomerativeClustering(n_clusters = 3,
                              affinity = "euclidean",
                              linkage = "ward")

# fit the model
```

```
hac.fit(donuts[predictors])

# represent results in a dendrogram
dendro = sch.dendrogram(sch.linkage(donuts[predictors],
method='ward'))
```



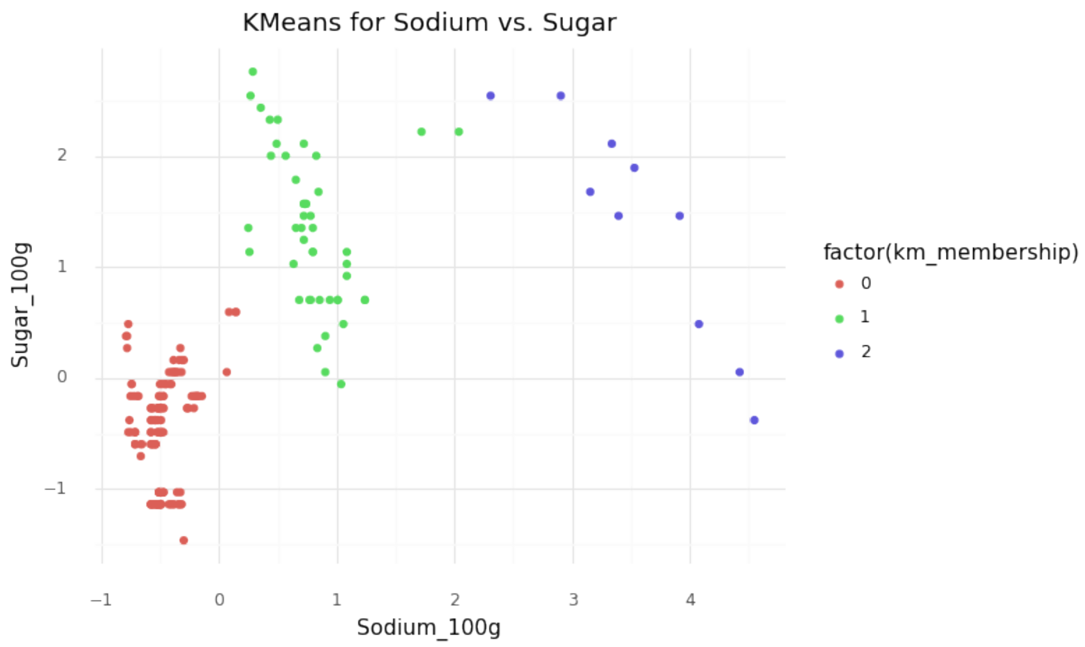
Hierarchical Agglomerative Clustering is the algorithm I chose that will perform the best for this data set using all 3 variables Sodium_100g, Total_Fat_100g and Sugar_100g. There are three hyperparameters: # of clusters, affinity, and linkage. I chose the number of clusters to be 3 for this model, because the scatterplots looked to be best representeted by three clusters. Affinity refers to the distance metric, and I chose Euclidean because it is used for continuous data. Lastly, the linkage criteria I chose was Ward's method, which involves the sum of squared errors and tells us which merge reduces error the most.

```
# KMeans
# create model
km = KMeans(n_clusters = 3)

# fit the model
km.fit(donuts[predictors])

# get assignments
km_membership = km.predict(donuts[predictors])

# add to data to plot
donuts["KMeans cluster"] = km_membership

# cluster membership results (sodium and total fat)
(ggplot(donuts, aes("Sodium_100g","Total_Fat_100g", color =
"factor(km_membership)")) +
 geom_point() + theme_minimal() +
 ggtitle("KMeans for Sodium vs. Total Fat"))
```



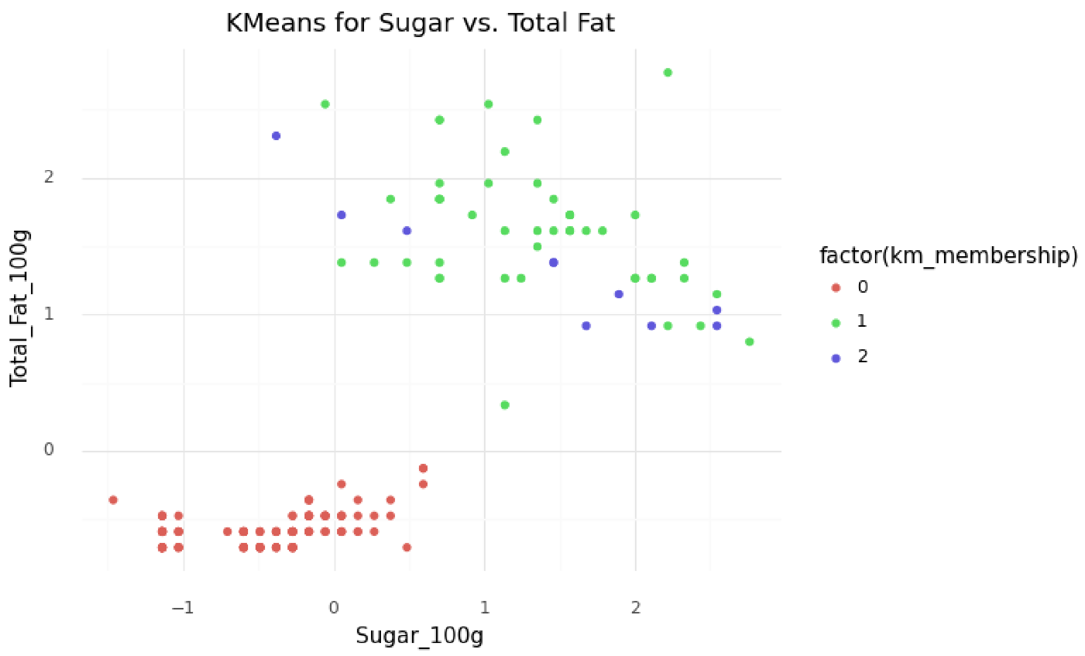
```
<ggplot: (8784214433333)>
```

```
# cluster membership results (sodium and sugar)
(ggplot(donuts, aes("Sodium_100g","Sugar_100g", color =
"factor(km_membership)")) +
 geom_point() + theme_minimal() +
 ggtitle("KMeans for Sodium vs. Sugar"))
```



```
<ggplot: (8784213886709)>
```

```
# cluster membership results (sugar and total fat)
(ggplot(donuts, aes("Sugar_100g", "Total_Fat_100g", color =
"factor(km_membership)")) +
 geom_point() + theme_minimal() +
 ggtitle("KMeans for Sugar vs. Total Fat"))
```



```
<ggplot: (8784214420045)>
```

KMeans is the algorithm I chose that will perform the worst on this data set, because an assumption in this model is that the data will present itself in spherical clusters. Looking at the donut datasets and relationships between the variables we are evaluating, none of the clusters appear to be very centered on a point with a spherical shape. I think this will perform poorly. For KMeans there is one hyperparameter: # of clusters (or *k* random points to become cluster centers). Again, I chose number of clusters to be 3 because of the group trends from the previous scatterplots.

d) Discuss Model Performance

```
# HAC model performance
# silhouette score
silhouette_score(donuts[predictors], hac.labels_)
```

```
0.7386187851455278
```

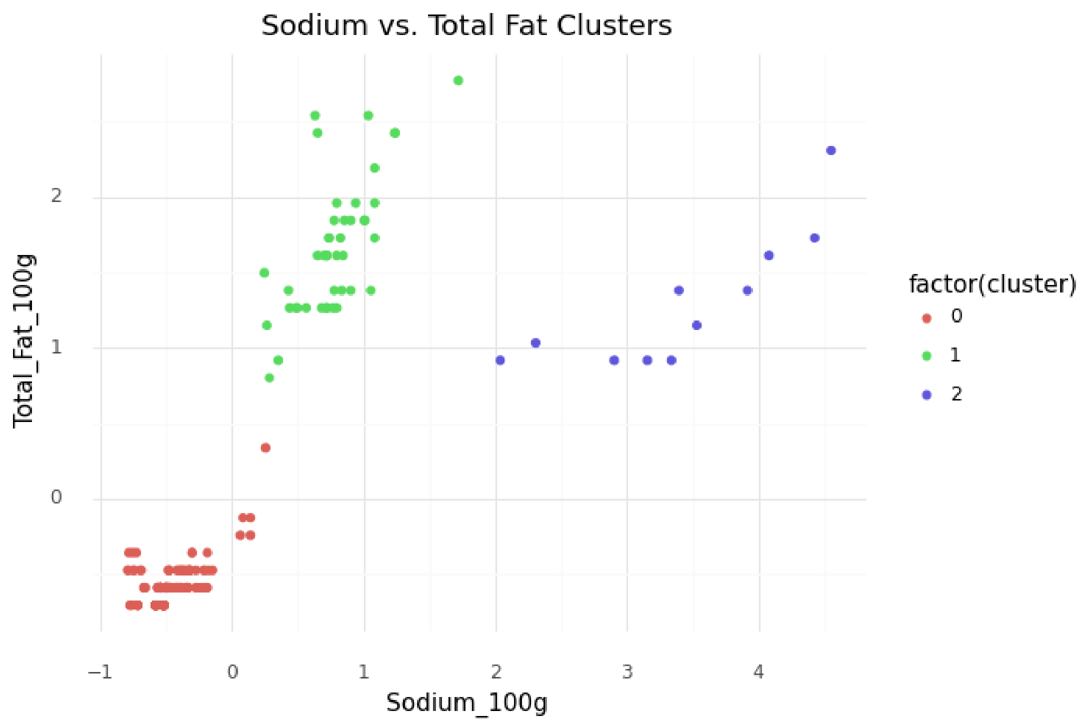
Hierarchical Agglomerative Clustering was the model I chose to perform the BEST. The metrics I used to assess my model is the silhouette score, and a simple dendrogram visual. The model has a silhouette score of 0.7386, which means it performed very well. Silhouette score is rated on a scale from -1 to 1, 1 meaning the model performed extremely well. We can also see from the dendrogram (in part c) that the top is much more spread out vertically, while the bottom is more dense. This means there is good seperation and cohesion for the clusters we evaluated, and the model performed well.

```
# KMeans model performance
# silhouette score
silhouette_score(donuts[predictors], km_membership)
```

```
0.7422190230687878
```

KMeans was the model I chose to perform the WORST. The metrics I used to assess my model are finding the silhouette score, and graphing the results of cluster membership. The model has a silhouette score of 0.7422, which means it actually performed slightly better than HAC. To get a visual of the clusters that KMeans evaluated, I created a new column in the data set to hold the predictions and used ggplot to see cluster membership. We can see from these graphs (in part c) that there are three distinct clusters when relating sodium vs. total fat, and sodium vs. sugar. However, the cluster membership for sugar vs. total fat is not as clean and distinct, it did not perform well relating these specific predictors.

```
# remake graphs from part a, color by cluster assignment
membership = hac.labels_
donuts["cluster"] = membership

# graph sodium vs total fat
(ggplot(donuts, aes(x = "Sodium_100g", y = "Total_Fat_100g", color =
"factor(cluster)")) +
 geom_point() + theme_minimal() +
 ggtitle("Sodium vs. Total Fat Clusters"))
```



```
<ggplot: (8784213797173)>
```

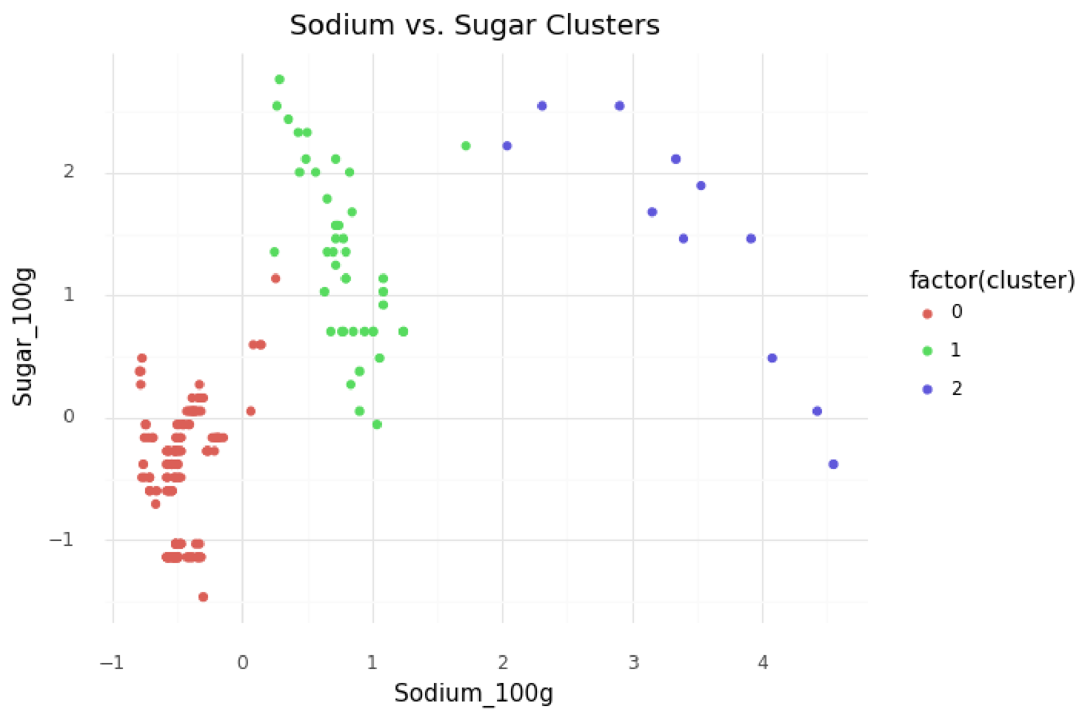
Sodium vs. Total Fat Clusters:

Cluster 0 - in red, represents the group of donuts that are low in fat and sodium. This group could be classified as "*healthy donuts*".

Cluster 1 - in green, represents the group of donuts that are medium to high in fat and have a medium amount of sodium. This group could be classified as "*classic donuts*".

Cluster 2 - in blue, represents the group of donuts that are medium to high in fat, and medium to high in sodium. This group could be classified as "*gourmet donuts*".

```
# graph sodium vs sugar
(ggplot(donuts, aes(x = "Sodium_100g", y = "Sugar_100g", color =
"factor(cluster)")) +
 geom_point() + theme_minimal() +
 ggtitle("Sodium vs. Sugar Clusters"))
```



```
<ggplot: (8784214434501)>
```

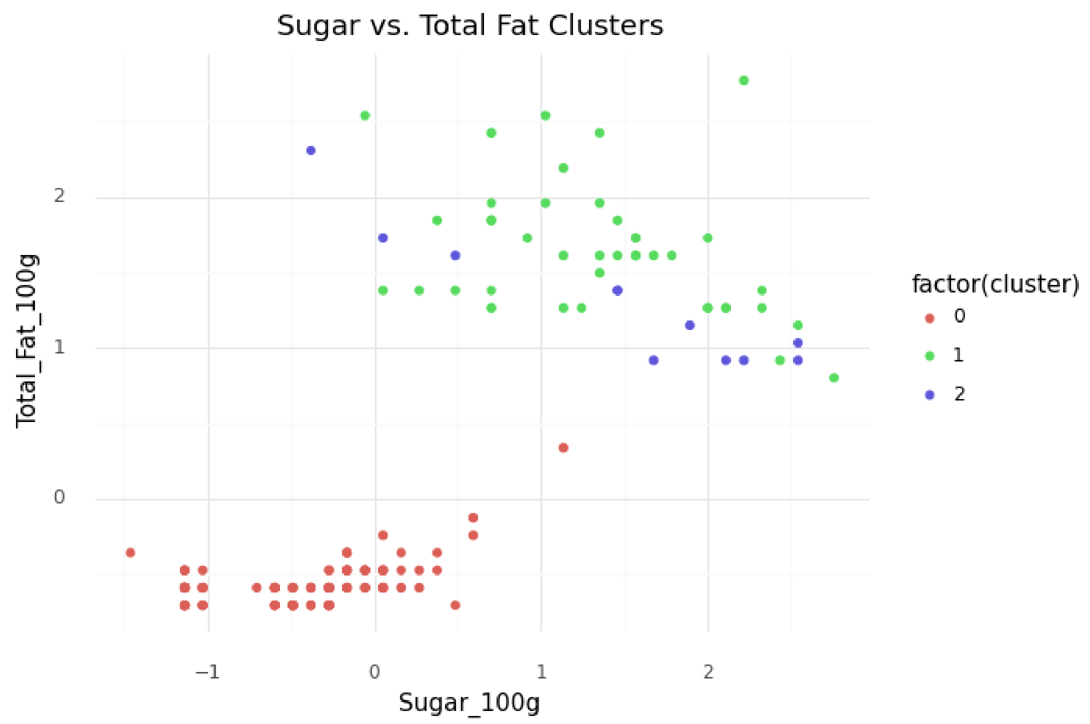
Sodium vs. Sugar Clusters:

Cluster 0 - in red, represents the group of donuts that are low in sugar and sodium. This group could be classified as "*simple & sweet donuts*".

Cluster 1 - in green, represents the group of donuts that are medium to high in sugar and have a medium amount of sodium. This group could be classified as "*a little salty but mostly sweet donuts*".

Cluster 2 - in blue, represents the group of donuts that are medium to high in sugar, and medium to high in sodium. This group could be classified as "*salty & sweet donuts*".

```
# graph sugar vs total fat
(ggplot(donuts, aes(x = "Sugar_100g", y = "Total_Fat_100g", color =
"factor(cluster)")) +
 geom_point() + theme_minimal() +
 ggtitle("Sugar vs. Total Fat Clusters"))
```



```
<ggplot: (8784213762301)>
```

Sugar vs. Total Fat Clusters:

Cluster 0 - in red, represents the group of donuts that are low in fat and medium to high in sugar. This group could be classified as "*healthy but sweet donuts*".

Cluster 1 - in green, represents the group of donuts that range from medium to high in fat and have a medium to high amount of sugar. This group could be classified as "*original donuts*".

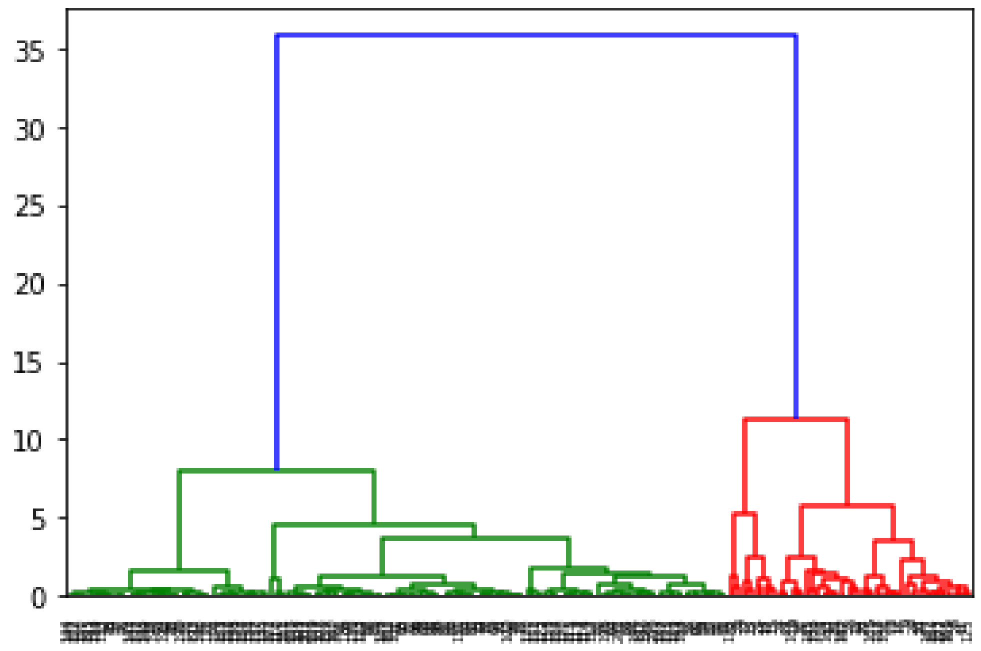
Cluster 2 - in blue, represents the sparse group of donuts that are medium to high in fat, and medium to high in sugar. This group has less fat and sugar than cluster 1 and could be classified as "*average donuts*".

e) Adding to HAC Clustering Model

I chose to add the variable **Carbohydrates_100g** to my clustering model to improve it because this is one of the main nutritional factors of donuts. They are loaded with carbs, and I think this is a necessary addition to evaluate donuts and classify them into groups.

```
# get predictors (sodium, total fat, sugar)
predictors2 = ["Sodium_100g", "Total_Fat_100g", "Sugar_100g",
               "Carbohydrates_100g"]

# z-score variables
```

```
z = StandardScaler()
donuts[predictors2] = z.fit_transform(donuts[predictors2])

# Hierarchical Agglomerative Clustering
# create the model
hac2 = AgglomerativeClustering(n_clusters = 3,
                               affinity = "euclidean",
                               linkage = "ward")

# fit the model
hac2.fit(donuts[predictors2])

# represent results in a dendrogram
dendro = sch.dendrogram(sch.linkage(donuts[predictors2],
method='ward'))
```



```
# model performance with silhouette score
silhouette_score(donuts[predictors2], hac2.labels_)

0.7461283109476201
```

The variable I added (Carbohydrates_100g) **improved the fit of my clustering model**. From the appearance of the dedrogram for this model, there is even a larger vertical distance at the top and more density at the bottom. Additionally, the silhouette score increased to 0.7461, when it was previously 0.7386.