# 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`

- **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**). (*IN A MARKDOWN CELL*)

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

(Please 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 algorithm** you think will work best here using the 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 model.
  - 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. (*IN A MARKDOWN CELL*)
  - make a new model, identical to the model 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.

In [88]:
```
# 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.neighbors import NearestNeighbors

from sklearn.cluster import AgglomerativeClustering

from sklearn.cluster import DBSCAN

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
```

In [57]:
```
data = pd.read_csv("https://raw.githubusercontent.com/cmparlettpelleriti/CPSC392Parlett
```

In [58]:
```
data
```

Out[58]:

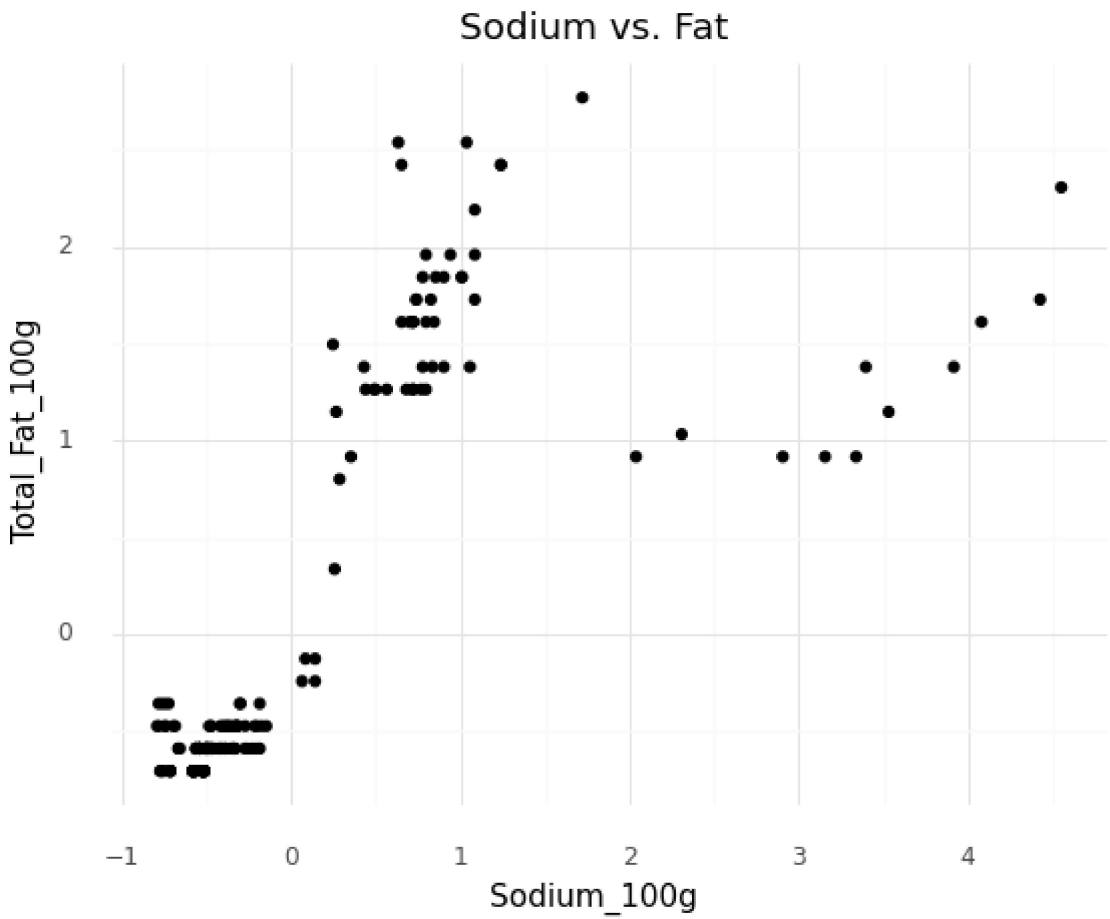
| | Restaurant_Item_Name | restaurant | Restaurant_ID | Item_Name | Item_Description | Food_Category | S |
|---|---|---|---|---|---|---|---|
| **0** | Krispy Kreme Apple Fritter | Krispy Kreme | 49 | Apple Fritter | Apple Fritter, Doughnuts | Baked Goods | |
| **1** | Krispy Kreme Chocolate Iced Cake Doughnut | Krispy Kreme | 49 | Chocolate Iced Cake Doughnut | Chocolate Iced Cake Doughnut, Doughnuts | Baked Goods | |

| | Restaurant_Item_Name | restaurant | Restaurant_ID | Item_Name | Item_Description | Food_Category | S |
|---|---|---|---|---|---|---|---|
| **2** | Krispy Kreme Chocolate Iced Custard Filled Dou... | Krispy Kreme | 49 | Chocolate Iced Custard Filled Doughnut | Chocolate Iced Custard Filled Doughnut, Doughnuts | Baked Goods | |
| **3** | Krispy Kreme Chocolate Iced Glazed Doughnut | Krispy Kreme | 49 | Chocolate Iced Glazed Doughnut | Chocolate Iced Glazed Doughnut, Doughnuts | Baked Goods | |
| **4** | Krispy Kreme Chocolate Iced Glazed Cruller Dou... | Krispy Kreme | 49 | Chocolate Iced Glazed Cruller Doughnut | Chocolate Iced Glazed Cruller Doughnut, Doughnuts | Baked Goods | |
| **...** | ... | ... | ... | ... | ... | ... | |
| **200** | Krispy Kreme Caramel Latte w/ Skim Milk, 16 oz | Krispy Kreme | 49 | Caramel Latte w/ Skim Milk, 16 oz | Caramel Latte w/ Skim Milk, Whipped Cream & Ca... | Beverages | |
| **201** | Krispy Kreme Caramel Latte w/ Skim Milk, 20 oz | Krispy Kreme | 49 | Caramel Latte w/ Skim Milk, 20 oz | Caramel Latte w/ Skim Milk, Whipped Cream & Ca... | Beverages | |
| **202** | Krispy Kreme Jolly Rancher Blue Raspberry Chil... | Krispy Kreme | 49 | Jolly Rancher Blue Raspberry Chiller, 12 oz | Jolly Rancher Blue Raspberry Chiller, 12 oz, D... | Beverages | |
| **203** | Krispy Kreme Jolly Rancher Blue Raspberry Chil... | Krispy Kreme | 49 | Jolly Rancher Blue Raspberry Chiller, 16 oz | Jolly Rancher Blue Raspberry Chiller, 16 oz, D... | Beverages | |
| **204** | Krispy Kreme Jolly Rancher Blue Raspberry Chil... | Krispy Kreme | 49 | Jolly Rancher Blue Raspberry Chiller, 20 oz | Jolly Rancher Blue Raspberry Chiller, 20 oz, D... | Beverages | |

205 rows × 32 columns

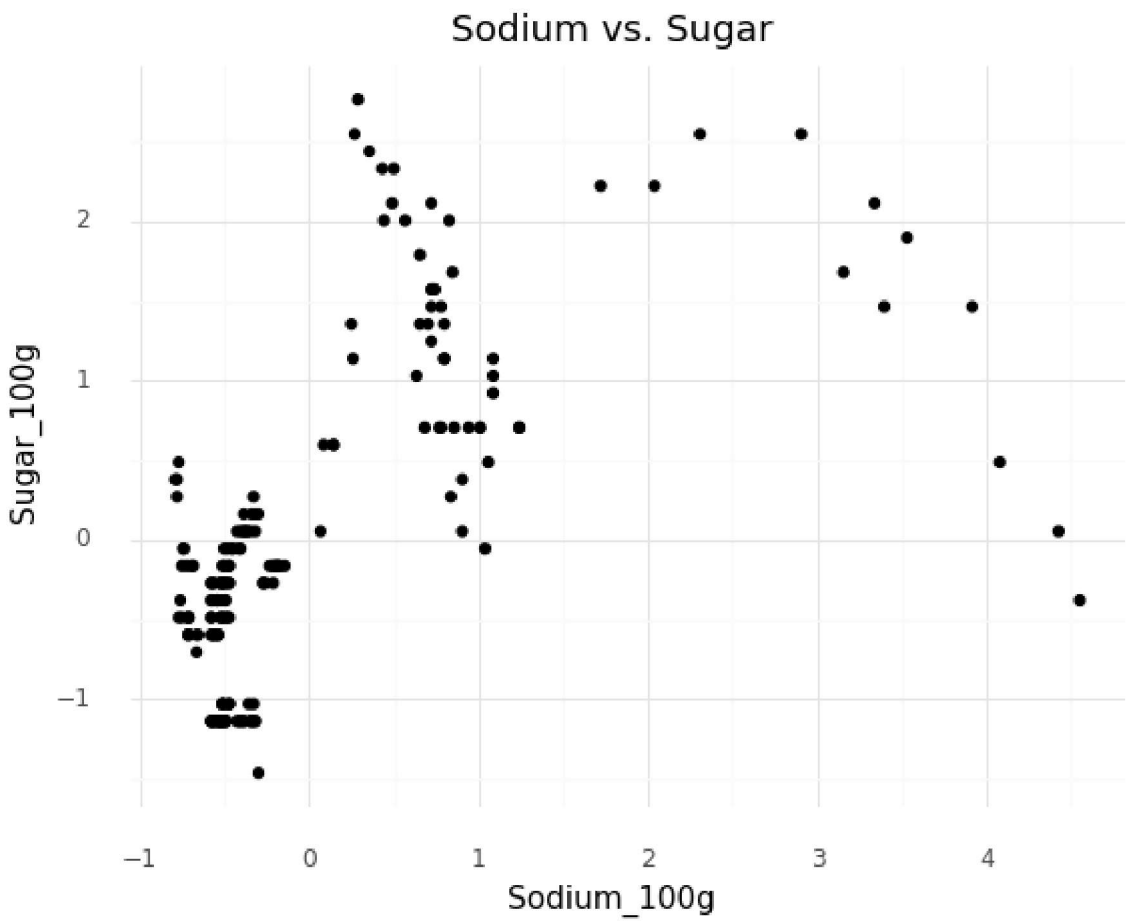
In [98]:

```
# a
#sodium vs fat
z = StandardScaler()
data[["Sodium_100g", "Total_Fat_100g"]] = z.fit_transform(data[["Sodium_100g", "Total_F

(ggplot(data, aes(x = "Sodium_100g", y = "Total_Fat_100g")) + geom_point() + theme_mini
```



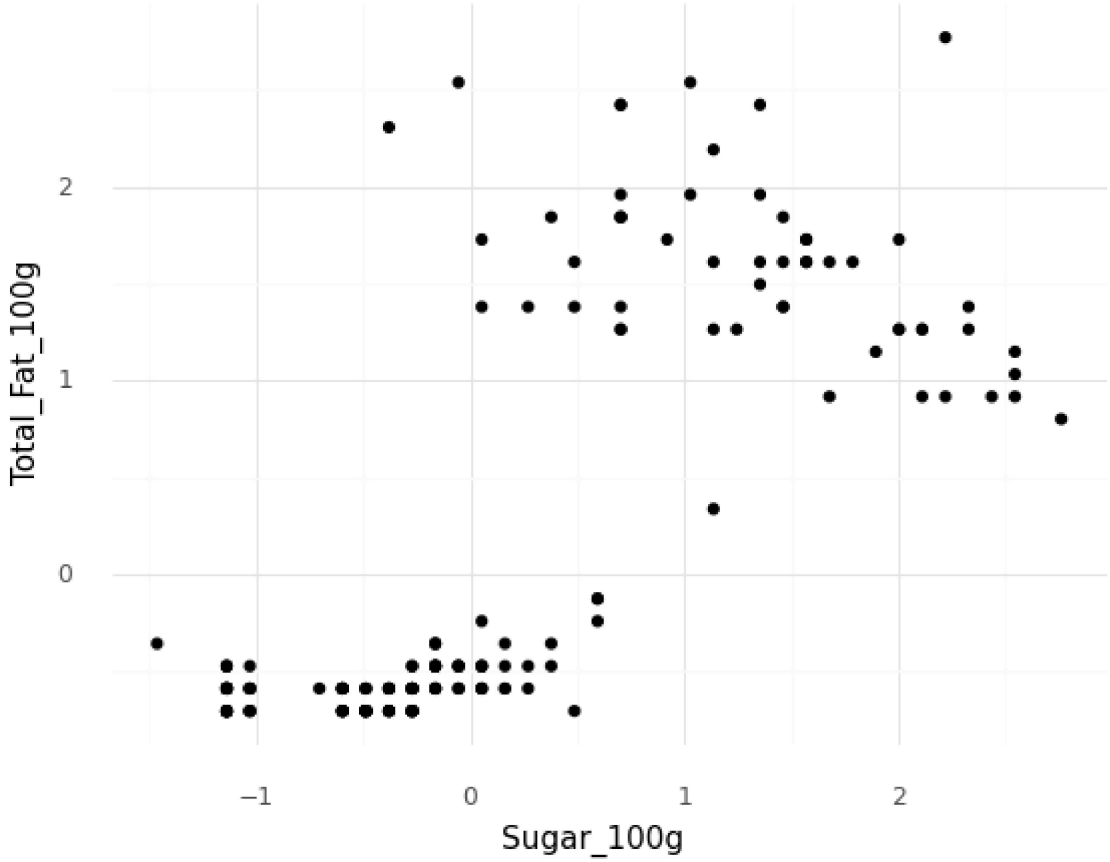

Out[98]: <ggplot: (127509936044)>

In [67]:
```
#sodium vs sugar
z = StandardScaler()
data[["Sodium_100g", "Sugar_100g"]] = z.fit_transform(data[["Sodium_100g", "Sugar_100g"

(ggplot(data, aes(x = "Sodium_100g", y = "Sugar_100g")) + geom_point() + theme_minimal(
```




Out[67]: <ggplot: (127511237958)>

In [99]:
```
#sugar vs fat
z = StandardScaler()
data[["Sugar_100g", "Total_Fat_100g"]] = z.fit_transform(data[["Sugar_100g", "Total_Fat

(ggplot(data, aes(x = "Sugar_100g", y = "Total_Fat_100g")) + geom_point() + theme_minim
```




Out[99]: (<ggplot: (127509954621)>, <plotnine.labels.ggtitle at 0x1db046355e0>)

b)

In [ ]:

```
I would choose Gaussian for this. First I will go over why I won't choose the other one
of data. K-Means assumes that there are spherical clusters. The three clusters here don
spherical and have unique shapes. K-Means also assumes that there are roughly the same
datapoints per cluster and that is just not the case because there looks to be a smalle
on the far right that is less dense for "sodium vs. sugar" and "sodium vs. fat".
DBSCAN could work here because DBSCAN can really work for any shaped dataset but it wor
well when there are points that might look like outliers. Those points that aren't with
the epsilan of the model will then be categorized as a noise point. However,
to me it doesn't quite look like there are a lot of points that could be considered noi
DBSCAN might also bring the clusters together if the epsilans are large enough and we d
Now it comes down to hierrarchical and gaussian. I think gaussian would be better becau
I can see 3 eliptical clusters for the first two but only two for "sugar vs. fat".
Hierrarchical is very flexible in the ways you can cluster your data
but you can't uncluster them. There was an example in one of our lecture notebooks that
linear and spread out dataset and made three clusters out of the. It seperated a cluste
assume are closer related. With gaussian I will be able to specify three and two cluste
for this dataset and know that I will get three instead of making multiple clusters out
of that right cluster seen in the first two graphs. Also overall, all the clusters look
to me and gaussian does that for its clustering method.
```

In [122…

```
#c
features1 = ["Sodium_100g", "Sugar_100g", "Total_Fat_100g"]
X1 = data[features]

z = StandardScaler()
```

```
X1[features] = z.fit_transform(X1)
EM = GaussianMixture(n_components = 3)
EM.fit(X1)

cluster = EM.predict(X1)

silhouette_score(X, cluster)
```

Out[122… 0.6486641909916648

In [ ]:

```
Since I only have 3 variables for the gaussian mixture I thought it would be best to ju
3 clusters hoping that the 3 variables might cluster separately. I recieved a silhouett
0.6486 which is okay. It's not great at clustering the data together but it did an alri
```

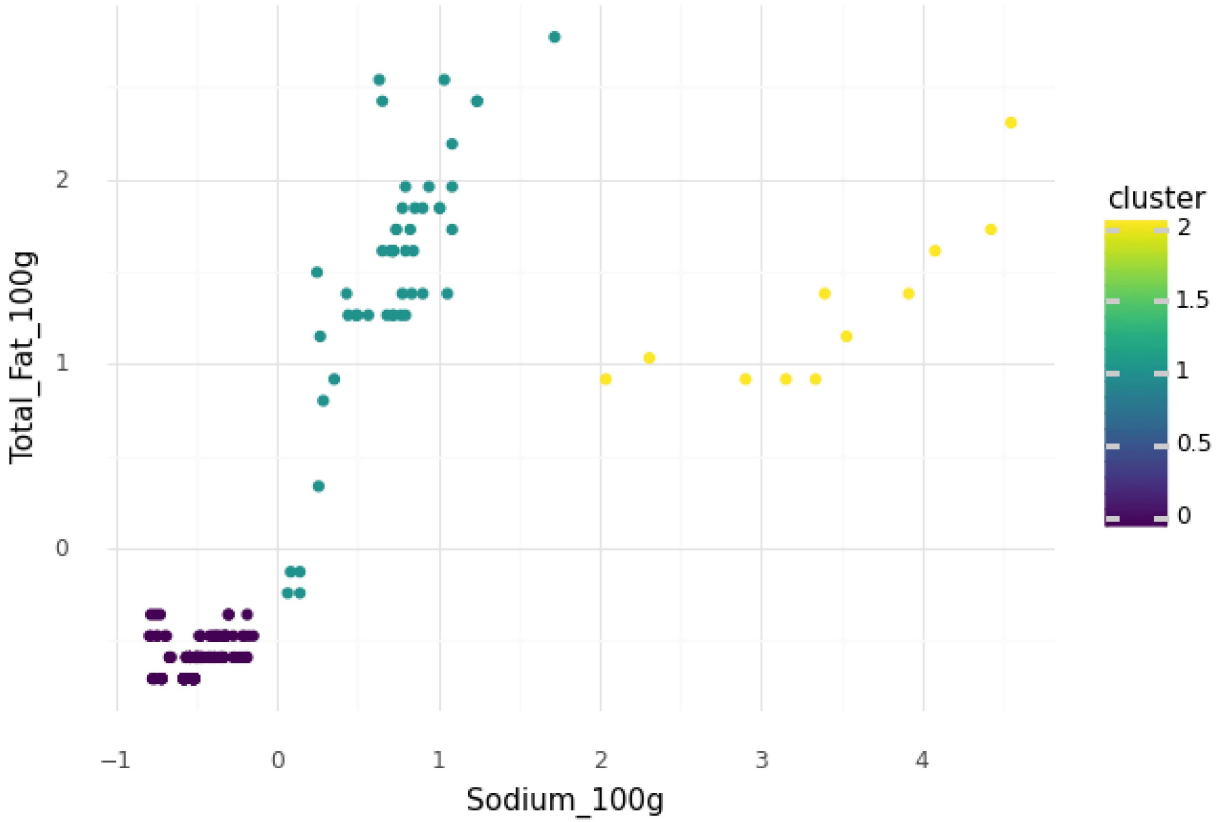
In [ ]:

```
# d
```

In [86]:

```
#Gaussian for Sodium vs. Fat
features = ["Sodium_100g", "Total_Fat_100g"]
X = data[features]

EM = GaussianMixture(n_components = 3)

EM.fit(X)

cluster = EM.predict(X)

print("SILHOUETTE: ", silhouette_score(X, cluster))

X["cluster"] = cluster

(ggplot(X, aes(x = "Sodium_100g", y = "Total_Fat_100g", color = "cluster")) + geom_poin
theme_minimal())
```

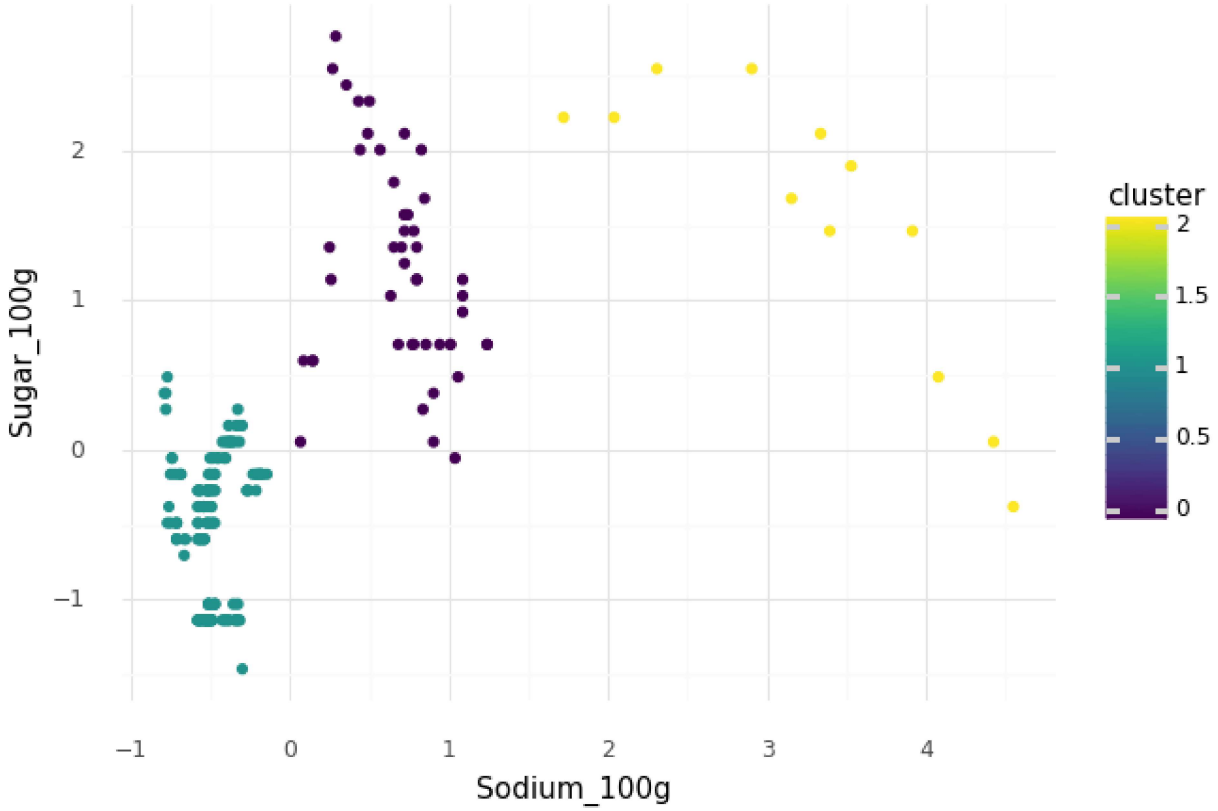
```
SILHOUETTE:  0.8127399609762124
```




Out[86]: <ggplot: (127510248683)>

In [84]:
```
#Gaussian Sodium vs. Sugar
features = ["Sodium_100g", "Sugar_100g"]
X = data[features]

X[features] = z.fit_transform(X)

EM = GaussianMixture(n_components = 3)

EM.fit(X)

cluster = EM.predict(X)

print("SILHOUETTE: ", silhouette_score(X, cluster))

X["cluster"] = cluster

(ggplot(X, aes(x = "Sodium_100g", y = "Sugar_100g", color = "cluster")) + geom_point()
```

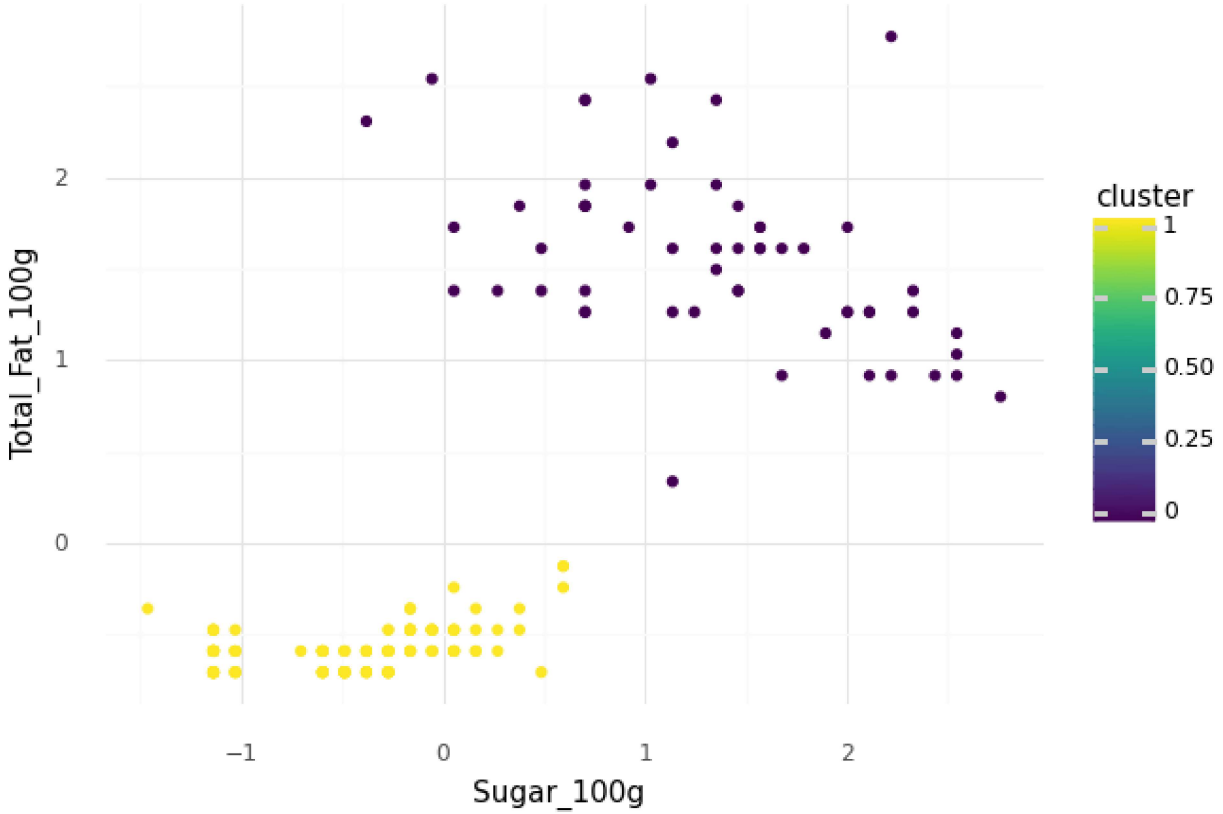
```
SILHOUETTE:  0.6703377282680327
```




Out[84]: <ggplot: (127510531813)>

In [96]:
```
#Gaussian for Sugar vs. Fat:
features = ["Total_Fat_100g", "Sugar_100g"]
X = data[features]

X[features] = z.fit_transform(X)

EM = GaussianMixture(n_components = 2)

EM.fit(X)

cluster = EM.predict(X)

print("SILHOUETTE: ", silhouette_score(X, cluster))

X["cluster"] = cluster

(ggplot(X, aes(x = "Sugar_100g", y = "Total_Fat_100g", color = "cluster")) + geom_point
```

```
SILHOUETTE:  0.7509980517389542
```




Out[96]: <ggplot: (127509871848)>

d)

In [ ]:
```
Gaussian Sodium vs. Fat:

I used silhouette score to test the accuracy of my model. I recieved a silhouette score
is pretty good. Based off of this I would say that Gaussian did a pretty good job of cr
data. The purple cluster are low sodium and fat, I would just call these ones "reduced
cluster is the largest out of the all the clusters and it is higher fat and low sodium,
"fatty" donuts. The yellow cluster has lower density than the other clusters and is med
would call these ones "savory" donuts.
```

In [ ]:
```
Gaussian Sugar vs. Sodium:

I used the silhouette score again for this Gaussian and I recieved a score around 0.670
but maybe if I tried hierrarchial or DBSCAN instead an Gaussian I could have gotten a h
okay job. The green cluster is low sugar and sodium and I would call these one's the "t
like these ones just wouldn't taste good. The purple cluster is medium to high sugar wi
I would call these donuts "sugar-rush" donuts. The last cluster is yellow and this one
high sugar to just below average sugar and high sodium. I would call this cluster the "
```

In [ ]:
```
Gaussian Sugar vs. Fat:

I used sihouette score for Sugar vs. Fat to test the accuarcy of the clustering and thi
of around 0.75099. I think that the gaussian did a pretty fair job of clustering this s
that could've been considered noise points if we used DBSCAN but overall I think that t
of creating the clusters. The purple cluster is low fat and average to low sugar, I wou
because it doesn't have two things that make food taste good. The yellow cluster is ave
```

```
cluster is less dense than the purple cluster but if you try to make 3 clusters out of
around 0.09 so 2 clusters seem about right.
```

e)

In [139…

```
# e
features1 = ["Sodium_100g", "Sugar_100g", "Total_Fat_100g", "Carbohydrates_100g"]
X1 = data[features1]

z = StandardScaler()

X1[features1] = z.fit_transform(X1)
EM = GaussianMixture(n_components = 3)
EM.fit(X1)

cluster = EM.predict(X1)

silhouette_score(X, cluster)
```

Out[139…

```
0.6176778819701901
```

In [ ]:

```
I chose chose carbohydrates as the new variables because I thought this could be a good
the products since not all of them are donuts. Carbohydrates variable could separate ou
lower carb options like the drinks. I thought that this could create more separation be
for Gaussian to make clusters but it didn't do anything. I used silhouette score again
clusters got any better and the silhouette score actually lowered. Not by much since it
of the data got worse when I added another variables. Maybe the gaussian needs more or
when adding a new variable.
```