Wholesale customers Data

Abstract:

The data set refers to clients of a wholesale distributor. It includes the annual spending in monetary units (m.u.) on diverse product categories.

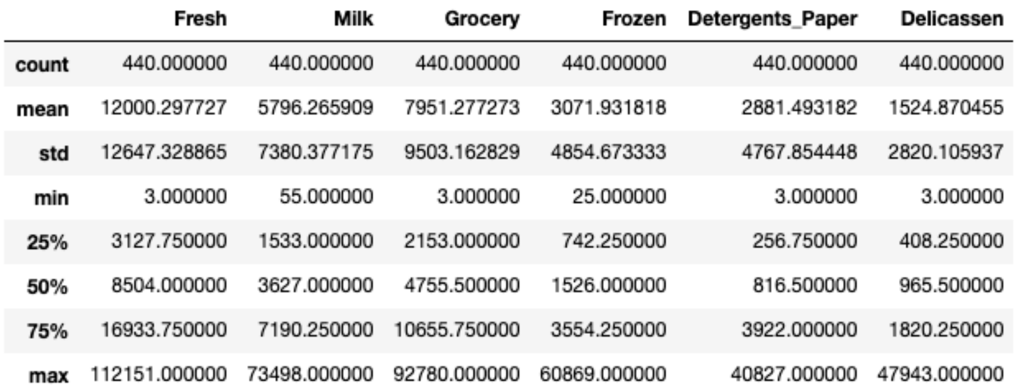
Data Set Information:

Provide all relevant information about your data set.

Attribute Information:

1) FRESH: annual spending (m.u.) on fresh products (Continuous);
2) MILK: annual spending (m.u.) on milk products (Continuous);
3) GROCERY: annual spending (m.u.)on grocery products (Continuous);
4) FROZEN: annual spending (m.u.)on frozen products (Continuous)
5) DETERGENTS_PAPER: annual spending (m.u.) on detergents and paper products (Continuous)
6) DELICATESSEN: annual spending (m.u.)on and delicatessen products (Continuous);
7) CHANNEL: customersâ€™ Channel - Horeca (Hotel/Restaurant/CafÃ©) or Retail channel (Nominal)
8) REGION: customersâ€™ Region â€“ Lisnon, Oporto or Other (Nominal)

Descriptive Statistics:



	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.000000
mean	12000.297727	5796.265909	7951.277273	3071.931818	2881.493182	1524.870455
std	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	2820.105937
min	3.000000	55.000000	3.000000	25.000000	3.000000	3.000000
25%	3127.750000	1533.000000	2153.000000	742.250000	256.750000	408.250000
50%	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	965.500000
75%	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	1820.250000
max	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	47943.000000

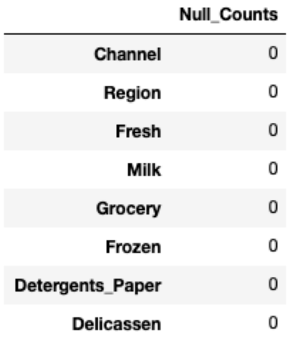
Region	Frequency
Lisbon	77
Oporto	47
Other Region	316
Total	440

CHANNEL	Frequency
Horeca	298
Retail	142
Total	440

Exploratory Data Analysis:

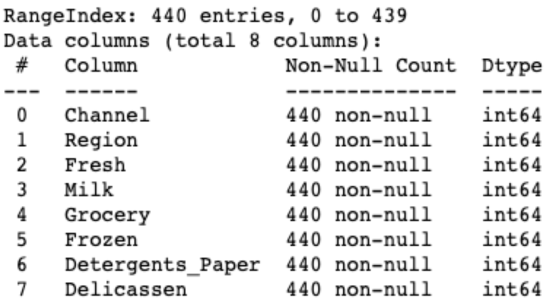
- The dataset contains 440 rows and 8 columns.
- There are no missing values in the dataset. So, there was no need to handle missing values
- The data type for all columns were numeric, as a result there was no need to do any data type casting.

Find and fix Null Values



	Null_Counts
Channel	0
Region	0
Fresh	0
Milk	0
Grocery	0
Frozen	0
Detergents_Paper	0
Delicassen	0

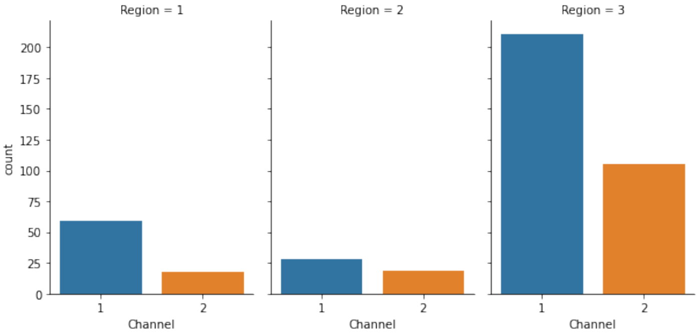
Check Data type of each column



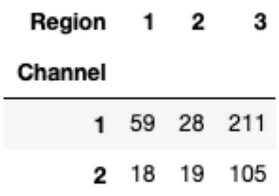
```
RangeIndex: 440 entries, 0 to 439
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Channel           440 non-null    int64
 1   Region            440 non-null    int64
 2   Fresh             440 non-null    int64
 3   Milk              440 non-null    int64
 4   Grocery           440 non-null    int64
 5   Frozen            440 non-null    int64
 6   Detergents_Paper  440 non-null    int64
 7   Delicassen        440 non-null    int64
```

- Created a count plot across categories Region and Channel. so, we can compare count across nested variables.

Count Plot by Region and Channel

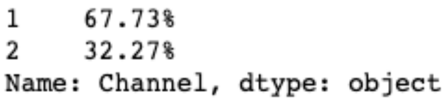


Tabular Representation of count plot.



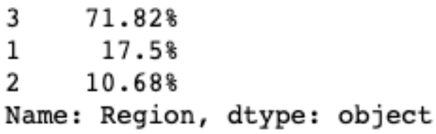
Region Channel	1	2	3
1	59	28	211
2	18	19	105

% Count of Unique Value – CHANNEL



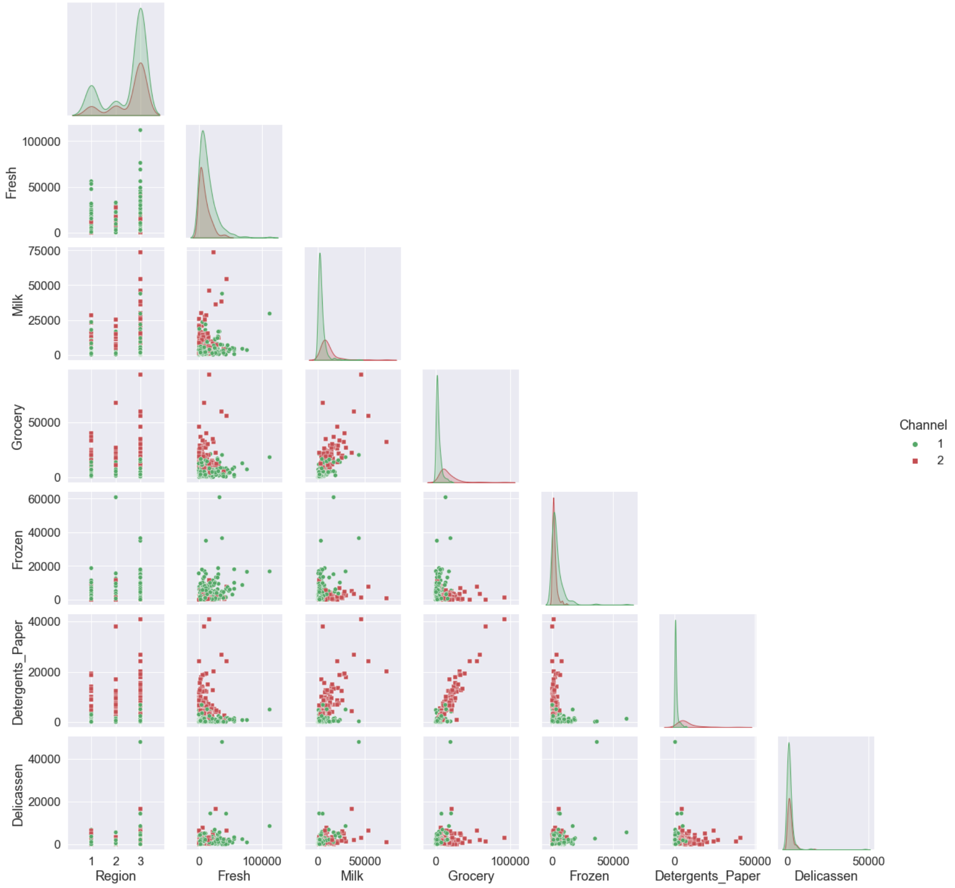
```
1     67.73%
2     32.27%
Name: Channel, dtype: object
```

% Count of Unique Value – REGION

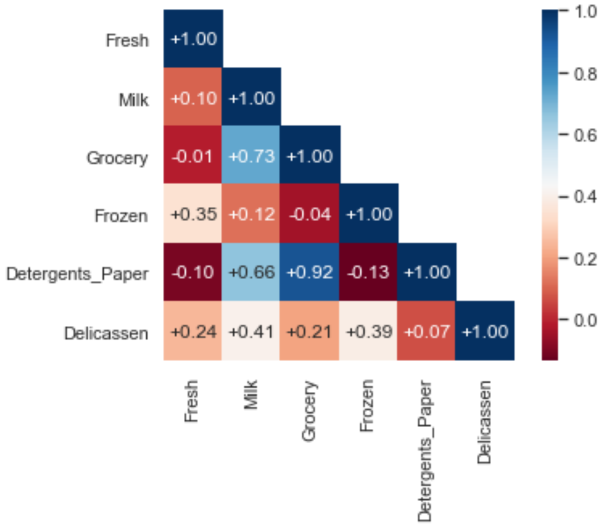


```
3     71.82%
1      17.5%
2     10.68%
Name: Region, dtype: object
```

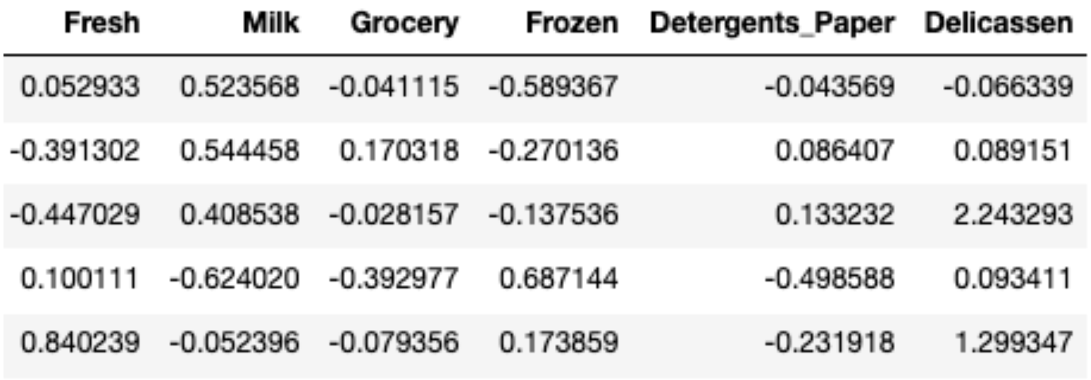
- Plot Pairwise relationship in the dataset by **Channel**. The Green color represents channel 1 while the RED color represents channel 2 in the below plot. Looking at the below chart we can see that there is some separation between two channels classes.



- **Computed the pairwise correlation** of all variables to better understand the direction of relationship. Looking at the below plot we can see the variables that are positively correlated vs the variables that are negatively correlated.

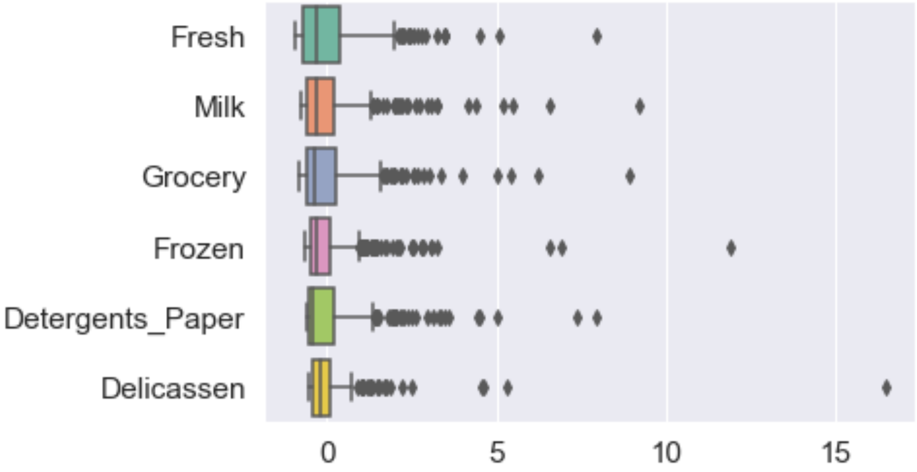


- **Standardize features** by removing the mean and scaling to unit variance. I applied the **StandardScaler** on all the features.



Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0.052933	0.523568	-0.041115	-0.589367	-0.043569	-0.066339
-0.391302	0.544458	0.170318	-0.270136	0.086407	0.089151
-0.447029	0.408538	-0.028157	-0.137536	0.133232	2.243293
0.100111	-0.624020	-0.392977	0.687144	-0.498588	0.093411
0.840239	-0.052396	-0.079356	0.173859	-0.231918	1.299347

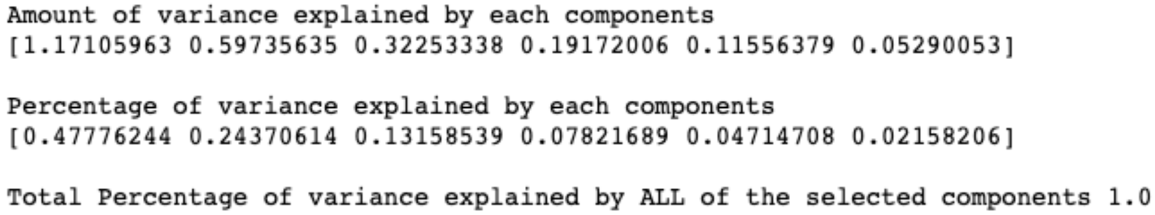
- **Handle Outlier:**
  - Generated box plot to identify Outliers. It is a method for graphically depicting groups of numerical data through their quartiles.



  - **Z score** helps to understand if a data value is greater or smaller than mean and how far away it is from the mean. If the z score of a data point is more than 3, it indicates that the data point is quite different from the other data points. Such a data point can be an outlier and it was removed.

- **Principal Component Analysis**: - We will use principal component analysis (PCA) to draw conclusions about the underlying structure of the wholesale customer data. Since using PCA on a dataset calculates the dimensions which best maximize variance, we will find which compound combinations of features best describe customers.
  Now that the data has been scaled to a more normal distribution and has had any necessary outliers removed, we can now apply PCA to the cleaned data to discover which dimensions about the data best maximize the variance of features

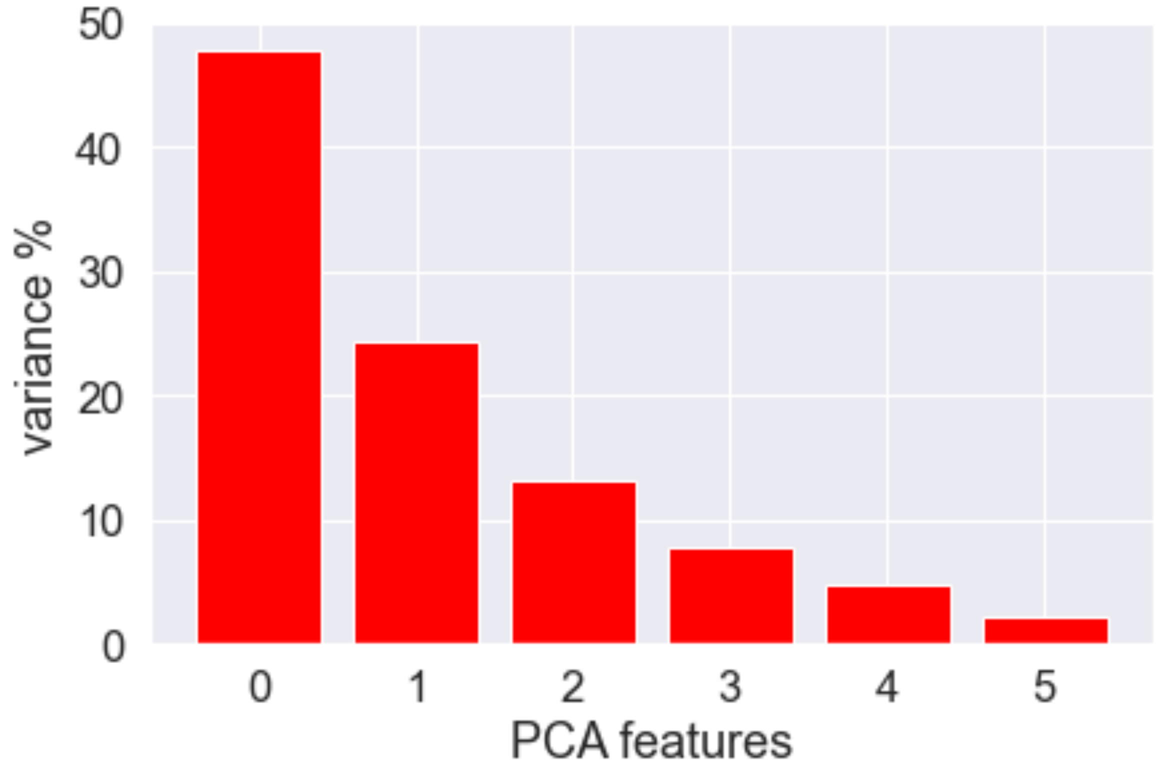
involved. In addition to finding these dimensions, PCA will also report the explained variance ratio of each dimension — how much variance within the data is explained by that dimension alone. Note that a component (dimension) from PCA can be considered a new "feature" of the space, however it is a composition of the original features present in the data.



```
Amount of variance explained by each components
[1.17105963 0.59735635 0.32253338 0.19172006 0.11556379 0.05290053]

Percentage of variance explained by each components
[0.47776244 0.24370614 0.13158539 0.07821689 0.04714708 0.02158206]

Total Percentage of variance explained by ALL of the selected components 1.0
```



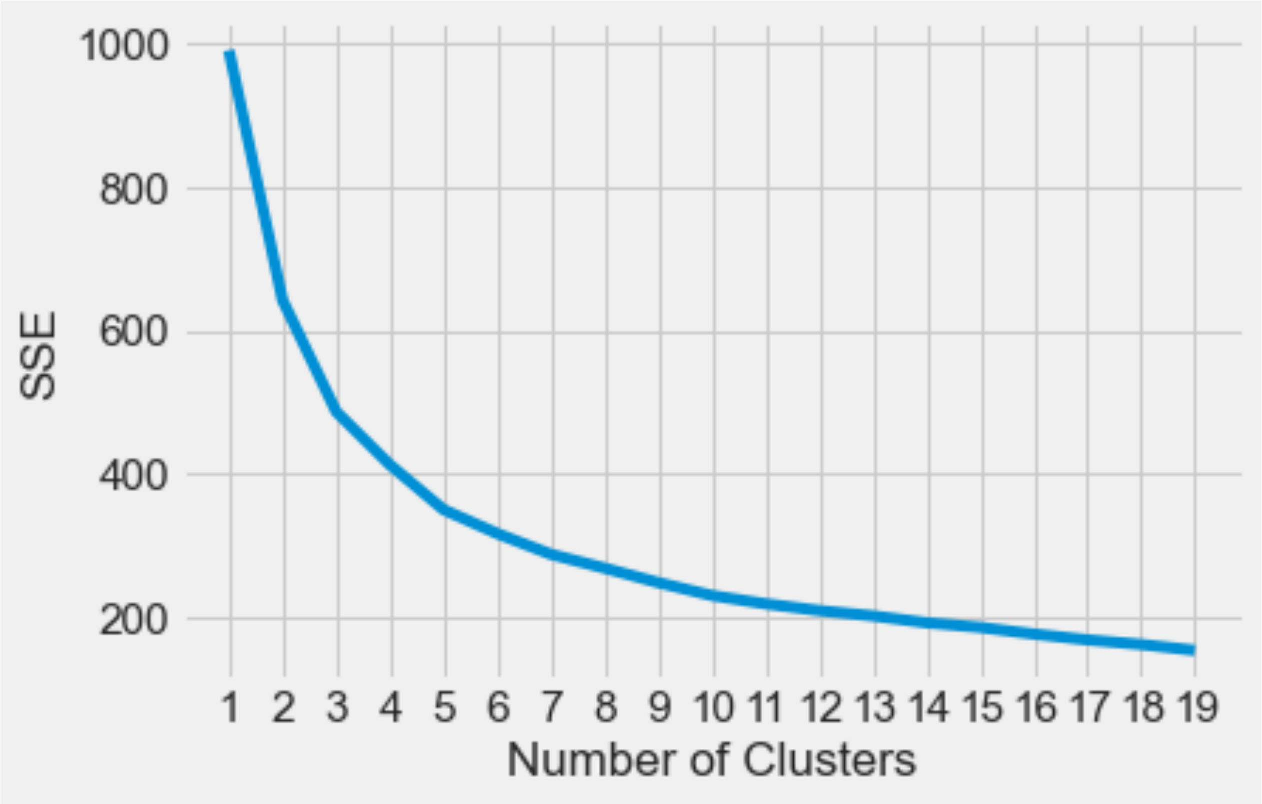
<u>Model Training:</u> Base line model with a **random** K Value

- Define the Kmeans function with initialization as K-means++
- Fit the KMeans Algorithm on processed data (Scaled + Outlier removed)
- Check for lowest SSE value
- Final locations of the Centroid
- The number of iterations to converge

Finding a Optimal K-Value:

There is no easy answer for choosing k value. One of the methods is known as **elbow method.** First of all compute the sum of squared error (SSE) for some value of K.SSE is defined as the sum of the squared distance between centroid and each member of the cluster. Then plot a K against SSE graph. We will observe that as K increases SSE decreases as distortion will be small. So, the idea of this algorithm is to choose the value of K at which the graph decreases abruptly. This sort of produces a “elbow effect” in the picture.

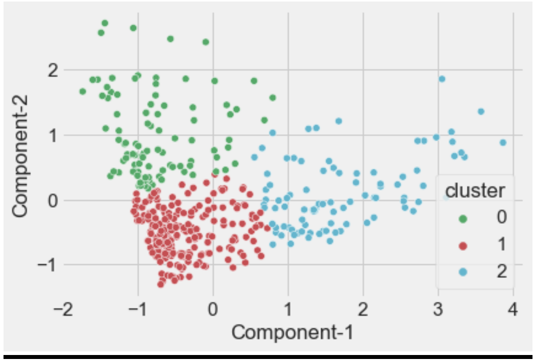
In the below chart we can see an elbow occurring around 3 so that’s a good number to choose.



Final Model:

- We will run the Kmeas algorithm again with cluster = 3 that we got using the **elbow method.**

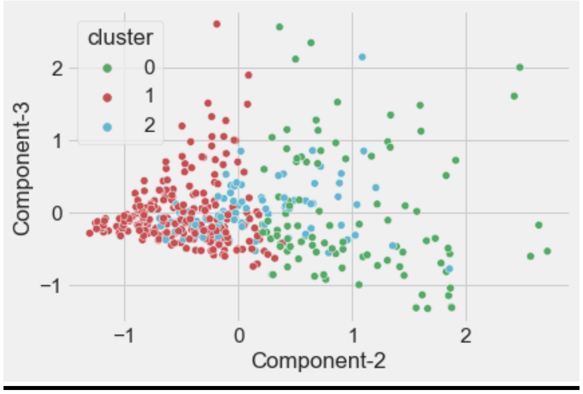
- Visualize the Labels/clusters generated by the KMeans model.

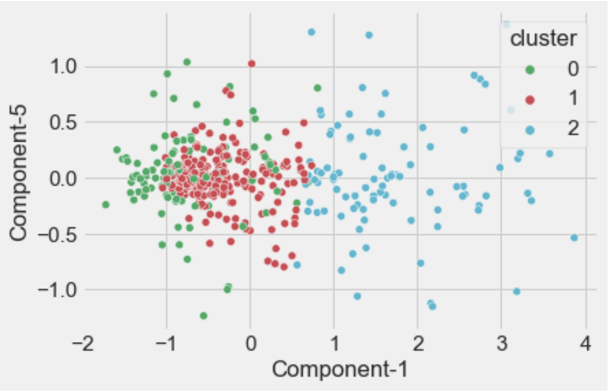


```
Cluster Values_count

1          222
0           100
2           92

Name: cluster, dtype: int64
```


cluster
Component-3
Component-2


Component-5
Component-1
cluster