```
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
%matplotlib inline
```

```
df = pd.read_csv('/content/ifood_df.csv')
```

```
df.head(50)
```

19	37040.0	0	0	41	86	2	73
20	2447.0	1	0	42	1	1	1725
21	58607.0	0	1	63	867	0	86
22	65324.0	0	1	0	384	0	102
23	40689.0	0	1	69	270	3	27
24	18589.0	0	0	89	6	4	25
25	53359.0	1	1	4	173	4	30
26	38360.0	1	0	26	36	2	42
27	84618.0	0	0	96	684	100	801
28	10979.0	0	0	34	8	4	10
29	38620.0	0	0	56	112	17	44
30	40548.0	0	1	31	110	0	5
31	46610.0	0	2	8	96	12	96
32	68657.0	0	0	4	482	34	471
33	49389.0	1	1	55	40	0	19
34	67353.0	0	1	37	702	17	151
35	23718.0	1	0	76	6	3	14
36	42429.0	0	1	99	55	0	6
37	48948.0	0	0	53	437	8	206
38	80011.0	0	1	3	421	76	536
39	20559.0	1	0	88	13	1	29
40	21994.0	0	1	4	9	0	6
41	7500.0	1	0	19	3	1	10
42	79941.0	0	0	72	123	164	266
43	7500.0	0	0	24	3	18	14
44	41728.0	1	0	92	13	6	15
45	72550.0	1	1	39	826	50	317
46	65486.0	0	1	29	245	19	125
47	79143.0	0	0	2	650	37	780
48	35790.0	1	0	54	12	6	20
49	82582.0	0	0	54	510	120	550

50 rows × 39 columns

```
df.head()
```

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts	MntFishProd
0	58138.0	0	0	58	635	88	546	
1	46344.0	1	1	38	11	1	6	
2	71613.0	0	0	26	426	49	127	
3	26646.0	1	0	26	11	4	20	
4	58293.0	1	0	94	173	43	118	

5 rows × 39 columns

```
def basic_info(df):
    print("This dataset has ", df.shape[1], " columns and ", df.shape[0], " rows.")
    print("This dataset has ", df[df.duplicated()].shape[0], " duplicated rows.")
    print(" ")
    print("Descriptive statistics of the numeric features in the dataset: ")
    print(" ")
    print(df.describe())
    print(" ")
    print("Information about this dataset: ")
    print(" ")
    print(df.info())
```

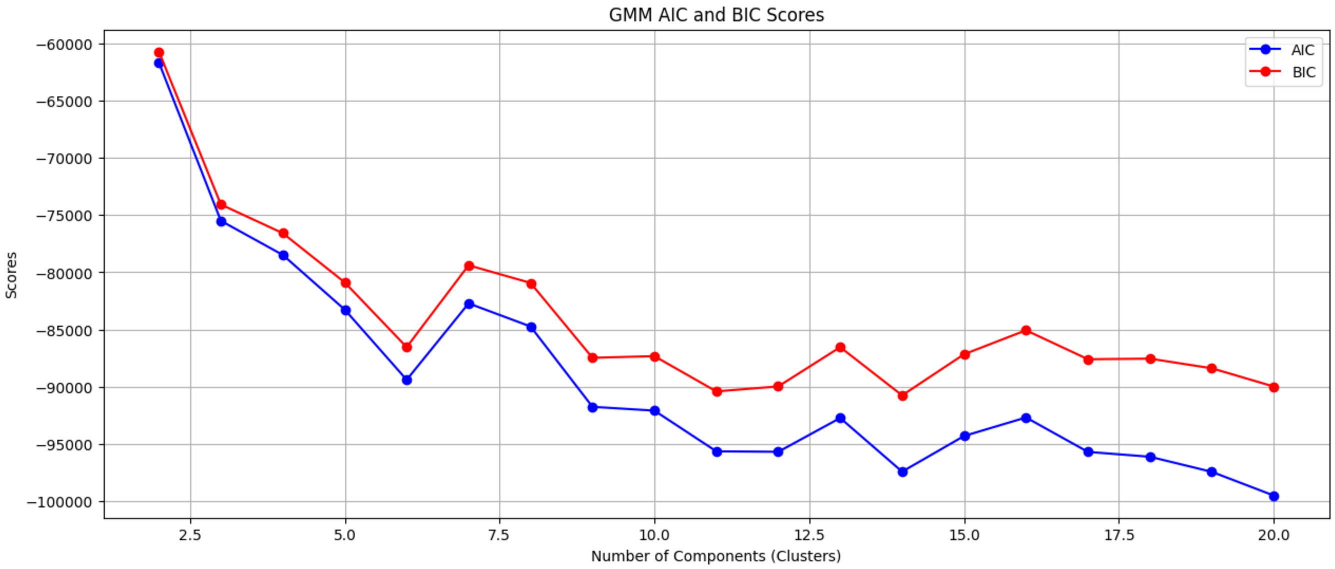
```
basic_info(df)
```

```
This dataset has  39  columns and  2205  rows.
This dataset has  184  duplicated rows.

Descriptive statistics of the numeric features in the dataset:

              Income      Kidhome     Teenhome      Recency     MntWines  \
count    2205.000000  2205.000000  2205.000000  2205.000000  2205.000000
mean    51622.094785     0.442177     0.506576    49.009070   306.164626
std     20713.063826     0.537132     0.544380    28.932111   337.493839
min      1730.000000     0.000000     0.000000     0.000000     0.000000
25%     35196.000000     0.000000     0.000000    24.000000    24.000000
50%     51287.000000     0.000000     0.000000    49.000000   178.000000
75%     68281.000000     1.000000     1.000000    74.000000   507.000000
max    113734.000000     2.000000     2.000000    99.000000  1493.000000

         MntFruits  MntMeatProducts  MntFishProducts  MntSweetProducts  \
count  2205.000000      2205.000000      2205.000000       2205.000000
mean     26.403175       165.312018        37.756463         27.128345
std      39.784484       217.784507        54.824635         41.130468
```

```
min      0.000000       0.000000       0.000000        0.000000
25%      2.000000      16.000000       3.000000        1.000000
50%      8.000000      68.000000      12.000000        8.000000
75%     33.000000     232.000000      50.000000       34.000000
max    199.000000    1725.000000     259.000000      262.000000

       MntGoldProds  ...  marital_Together  marital_Widow  education_2n Cycle  \
count   2205.000000  ...       2205.000000    2205.000000         2205.000000
mean      44.057143  ...          0.257596       0.034467            0.089796
std       51.736211  ...          0.437410       0.182467            0.285954
min        0.000000  ...          0.000000       0.000000            0.000000
25%        9.000000  ...          0.000000       0.000000            0.000000
50%       25.000000  ...          0.000000       0.000000            0.000000
75%       56.000000  ...          1.000000       0.000000            0.000000
max      321.000000  ...          1.000000       1.000000            1.000000

       education_Basic  education_Graduation  education_Master  education_PhD  \
count      2205.000000           2205.000000       2205.000000    2205.000000
mean          0.024490              0.504762          0.165079       0.215873
std           0.154599              0.500091          0.371336       0.411520
min           0.000000              0.000000          0.000000       0.000000
25%           0.000000              0.000000          0.000000       0.000000
50%           0.000000              1.000000          0.000000       0.000000
75%           0.000000              1.000000          0.000000       0.000000
max           1.000000              1.000000          1.000000       1.000000

          MntTotal  MntRegularProds  AcceptedCmpOverall
count  2205.000000      2205.000000          2205.00000
mean    562.764626       518.707483             0.29932
std     575.936911       553.847248             0.68044
min       4.000000      -283.000000             0.00000
25%      56.000000        42.000000             0.00000
50%     343.000000       288.000000             0.00000
75%     964.000000       884.000000             0.00000
max    2491.000000      2458.000000             4.00000

[8 rows x 39 columns]

Information about this dataset:
```

```python
def remove_outliers(df, column_names):
    for column in column_names:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        df = df[(df[column] >= lower_bound) & (df[column] <= upper_bound)]
    return df


columns = ['Income', 'Kidhome', 'Teenhome', 'Recency', 'MntFruits', 'MntMeatProducts',
           'NumDealsPurchases', 'AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
           'AcceptedCmp4', 'AcceptedCmp5']
```

```
df_no_outliers = remove_outliers(df, columns)



# Standardizing features
scaler = StandardScaler()
scaled_features_no_outliers = scaler.fit_transform(df_no_outliers[columns])


from sklearn.mixture import GaussianMixture
# Rerun KMeans clustering and silhouette analysis
n_components_range = range(2, 21)
aic_scores = []
bic_scores = []

for n in n_components_range:
    gmm = GaussianMixture(n_components=n, random_state=42)
    gmm.fit(scaled_features_no_outliers)
    aic_scores.append(gmm.aic(scaled_features_no_outliers))
    bic_scores.append(gmm.bic(scaled_features_no_outliers))


plt.figure(figsize=(15, 6))

plt.plot(n_components_range, aic_scores, marker='o', linestyle='-', color='b', label='
plt.plot(n_components_range, bic_scores, marker='o', linestyle='-', color='r', label='

plt.title('GMM AIC and BIC Scores')
plt.xlabel('Number of Components (Clusters)')
plt.ylabel('Scores')
plt.grid(True)
plt.legend()
plt.show()
```


GMM AIC and BIC Scores
AIC
BIC
Scores
Number of Components (Clusters)