A. Importing the libraries and Dataset

Importing the Libraries

```python
# importing the required libraries for calculation

import numpy as np
import pandas as pd

# importing the required libraries for visualisation

import matplotlib.pyplot as plt
import seaborn as sns

# importing the required libraries for prediction

from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
```

Importing the Dataset

```python
data = pd.read_csv ("Downloads\cereal.csv")
```

Copying the dataset for further Analysis

```python
brk_cr = data.copy()
brk_cr
```

```
                          name mfr type  calories  protein  fat  sodium
fiber  \
0                    100% Bran   N   C        70        4    1     130
10.0
1            100% Natural Bran   Q   C       120        3    5      15
2.0
2                      All-Bran   K   C        70        4    1     260
9.0
3   All-Bran with Extra Fiber   K   C        50        4    0     140
14.0
4               Almond Delight   R   C       110        2    2     200
1.0
..                         ...  ..  ...       ...      ...  ...     ...
...
72                     Triples   G   C       110        2    1     250
0.0
73                        Trix   G   C       110        1    1     140
0.0
74                  Wheat Chex   R   C       100        3    1     230
3.0
75                    Wheaties   G   C       100        3    1     200
3.0
```

```
76          Wheaties Honey Gold   G   C        110        2    1        200
1.0

      carbo  sugars  potass  vitamins  shelf  weight  cups      rating
0       5.0       6     280        25      3     1.0  0.33  68.402973
1       8.0       8     135         0      3     1.0  1.00  33.983679
2       7.0       5     320        25      3     1.0  0.33  59.425505
3       8.0       0     330        25      3     1.0  0.50  93.704912
4      14.0       8      -1        25      3     1.0  0.75  34.384843
..      ...     ...     ...       ...    ...     ...   ...        ...
72     21.0       3      60        25      3     1.0  0.75  39.106174
73     13.0      12      25        25      2     1.0  1.00  27.753301
74     17.0       3     115        25      1     1.0  0.67  49.787445
75     17.0       3     110        25      1     1.0  1.00  51.592193
76     16.0       8      60        25      1     1.0  0.75  36.187559

[77 rows x 16 columns]
```

Exploring the Data

```
# Finding the total records and features provided in the dataset

brk_cr.shape

(77, 16)

# Memory storage used for each column

brk_cr.memory_usage(index = False,deep = True)

name         5531
mfr          4466
type         4466
calories      616
protein       616
fat           616
sodium        616
fiber         616
carbo         616
sugars        616
potass        616
vitamins      616
shelf         616
weight        616
cups          616
rating        616
dtype: int64

# Getting information regarding all columns

brk_cr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 77 entries, 0 to 76
Data columns (total 16 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   name      77 non-null     object
 1   mfr       77 non-null     object
 2   type      77 non-null     object
 3   calories  77 non-null     int64
 4   protein   77 non-null     int64
 5   fat       77 non-null     int64
 6   sodium    77 non-null     int64
 7   fiber     77 non-null     float64
 8   carbo     77 non-null     float64
 9   sugars    77 non-null     int64
 10  potass    77 non-null     int64
 11  vitamins  77 non-null     int64
 12  shelf     77 non-null     int64
 13  weight    77 non-null     float64
 14  cups      77 non-null     float64
 15  rating    77 non-null     float64
dtypes: float64(5), int64(8), object(3)
memory usage: 9.8+ KB
```

Gettinng statistical information of all numeric features of dataset

```
brk_cr.describe()
```

	calories	protein	fat	sodium	fiber	carbo
count	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000
mean	106.883117	2.545455	1.012987	159.675325	2.151948	14.597403
std	19.484119	1.094790	1.006473	83.832295	2.383364	4.278956
min	50.000000	1.000000	0.000000	0.000000	0.000000	-1.000000
25%	100.000000	2.000000	0.000000	130.000000	1.000000	12.000000
50%	110.000000	3.000000	1.000000	180.000000	2.000000	14.000000
75%	110.000000	3.000000	2.000000	210.000000	3.000000	17.000000
max	160.000000	6.000000	5.000000	320.000000	14.000000	23.000000

	sugars	potass	vitamins	shelf	weight	cups
count	77.000000	77.000000	77.000000	77.000000	77.000000	

```
           77.000000
mean        6.922078      96.077922     28.246753    2.207792    1.029610
0.821039
std         4.444885      71.286813     22.342523    0.832524    0.150477
0.232716
min        -1.000000      -1.000000      0.000000    1.000000    0.500000
0.250000
25%         3.000000      40.000000     25.000000    1.000000    1.000000
0.670000
50%         7.000000      90.000000     25.000000    2.000000    1.000000
0.750000
75%        11.000000     120.000000     25.000000    3.000000    1.000000
1.000000
max        15.000000     330.000000    100.000000    3.000000    1.500000
1.500000

           rating
count   77.000000
mean    42.665705
std     14.047289
min     18.042851
25%     33.174094
50%     40.400208
75%     50.828392
max     93.704912
```

B. Data Cleaning and Data Wrangling

Checking for Null Value

```
brk_cr.isnull().sum()
```

```
name        0
mfr         0
type        0
calories    0
protein     0
fat         0
sodium      0
fiber       0
carbo       0
sugars      0
potass      0
vitamins    0
shelf       0
weight      0
cups        0
rating      0
dtype: int64
```

-- There is No missing values in the dataset

Changing the Values

```
#Changing the values in column 'mfr' into their expanded form Using
replace function

brk_cr.mfr = brk_cr.mfr.replace ({'A':'American Home Food
Products','G':'General Mills','K':'Kelloggs','N':'Nabisco',
                        'P':'Post','Q':'Quaker Oats','R':'Ralston
Purina'})

# Changing the values in column 'type' into their expanded form using
replace function

brk_cr.type = brk_cr.type.replace({'C':'Cold','H':'Hot'})

brk_cr.head(5)
```

```
                        name              mfr  type  calories  protein
fat  \
0                    100% Bran          Nabisco  Cold        70        4
1
1          100% Natural Bran     Quaker Oats  Cold       120        3
5
2                     All-Bran        Kelloggs  Cold        70        4
1
3  All-Bran with Extra Fiber        Kelloggs  Cold        50        4
0
4              Almond Delight  Ralston Purina  Cold       110        2
2

   sodium  fiber  carbo  sugars  potass  vitamins  shelf  weight  cups
\
0     130   10.0    5.0       6     280        25      3     1.0  0.33

1      15    2.0    8.0       8     135         0      3     1.0  1.00

2     260    9.0    7.0       5     320        25      3     1.0  0.33

3     140   14.0    8.0       0     330        25      3     1.0  0.50

4     200    1.0   14.0       8      -1        25      3     1.0  0.75

     rating
0  68.402973
1  33.983679
2  59.425505
3  93.704912
4  34.384843
```

Adding the new Columns

```python
# Defining function for calculating the percentage of fat in calorie

def perc(a,b):
    return round((a/b)*100,2)

#Creating new column and applying function

brk_cr['perc_fat_in_calorie'] = brk_cr.apply(lambda X: perc(X['fat'],
X['calories']), axis = 1)

# Defining function for creating sodium levels

def sod(a):
    if a>= 500:
        return 'High'
    elif a <= 140:
        return 'Low'
    elif a == 0:
        return 'No Sodium'
    else:
        return 'Good'

# Applying the def to new column

brk_cr['sodium_status'] = brk_cr['sodium'].apply(sod)

# Defining function for finding the levels of fiber source

def fib(a):
    if a > 4.9:
        return 'Excellent'
    elif a < 2.4:
        return 'Good'
    else:
        return 'Average'

# Applying the def to new column

brk_cr['fiber_source'] = brk_cr['fiber'].apply(fib)

# Defining function for creating pottasium content

def pott(a):
    if a > 500:
        return 'High'
    elif (a < 500) and (a > 300):
        return 'Safe'
    else:
        return 'Low'
```

```
# Applying the def to new column

brk_cr['Potass_status'] = brk_cr['potass'].apply(pott)
brk_cr
```

```
                          name              mfr   type  calories  protein
fat  \
0                     100% Bran          Nabisco  Cold        70        4
1
1             100% Natural Bran      Quaker Oats  Cold       120        3
5
2                       All-Bran         Kelloggs  Cold        70        4
1
3     All-Bran with Extra Fiber         Kelloggs  Cold        50        4
0
4                 Almond Delight  Ralston Purina  Cold       110        2
2
..                          ...              ...   ...       ...      ...
...
72                       Triples   General Mills  Cold       110        2
1
73                          Trix   General Mills  Cold       110        1
1
74                    Wheat Chex  Ralston Purina  Cold       100        3
1
75                      Wheaties   General Mills  Cold       100        3
1
76          Wheaties Honey Gold   General Mills  Cold       110        2
1

    sodium  fiber  carbo  sugars  potass  vitamins  shelf  weight
cups  \
0      130   10.0    5.0       6     280        25      3     1.0
0.33
1       15    2.0    8.0       8     135         0      3     1.0
1.00
2      260    9.0    7.0       5     320        25      3     1.0
0.33
3      140   14.0    8.0       0     330        25      3     1.0
0.50
4      200    1.0   14.0       8      -1        25      3     1.0
0.75
..     ...    ...    ...     ...     ...       ...    ...     ...    ..
.
72     250    0.0   21.0       3      60        25      3     1.0
0.75
73     140    0.0   13.0      12      25        25      2     1.0
1.00
74     230    3.0   17.0       3     115        25      1     1.0
0.67
```

```
75      200     3.0     17.0         3        110            25         1        1.0
1.00
76      200     1.0     16.0         8         60            25         1        1.0
0.75

       rating   perc_fat_in_calorie sodium_status fiber_source
Potass_status
0   68.402973                  1.43           Low    Excellent
Low
1   33.983679                  4.17           Low         Good
Low
2   59.425505                  1.43          Good    Excellent
Safe
3   93.704912                  0.00           Low    Excellent
Safe
4   34.384843                  1.82          Good         Good
Low
..        ...                   ...           ...          ...
...
72  39.106174                  0.91          Good         Good
Low
73  27.753301                  0.91           Low         Good
Low
74  49.787445                  1.00          Good      Average
Low
75  51.592193                  1.00          Good      Average
Low
76  36.187559                  0.91          Good         Good
Low

[77 rows x 20 columns]
```

C. Analysis and Visualisation

>Statistical Information of Data

```
brk_cr.describe().transpose()

                   count        mean         std         min
25%  \
calories            77.0  106.883117   19.484119   50.000000
100.000000
protein             77.0    2.545455    1.094790    1.000000
2.000000
fat                 77.0    1.012987    1.006473    0.000000
0.000000
sodium              77.0  159.675325   83.832295    0.000000
130.000000
fiber               77.0    2.151948    2.383364    0.000000
1.000000
```

carbo	77.0	14.597403	4.278956	-1.000000	12.000000
sugars	77.0	6.922078	4.444885	-1.000000	3.000000
potass	77.0	96.077922	71.286813	-1.000000	40.000000
vitamins	77.0	28.246753	22.342523	0.000000	25.000000
shelf	77.0	2.207792	0.832524	1.000000	1.000000
weight	77.0	1.029610	0.150477	0.500000	1.000000
cups	77.0	0.821039	0.232716	0.250000	0.670000
rating	77.0	42.665705	14.047289	18.042851	33.174094
perc_fat_in_calorie	77.0	0.890519	0.844767	0.000000	0.000000

	50%	75%	max
calories	110.000000	110.000000	160.000000
protein	3.000000	3.000000	6.000000
fat	1.000000	2.000000	5.000000
sodium	180.000000	210.000000	320.000000
fiber	2.000000	3.000000	14.000000
carbo	14.000000	17.000000	23.000000
sugars	7.000000	11.000000	15.000000
potass	90.000000	120.000000	330.000000
vitamins	25.000000	25.000000	100.000000
shelf	2.000000	3.000000	3.000000
weight	1.000000	1.000000	1.500000
cups	0.750000	1.000000	1.500000
rating	40.400208	50.828392	93.704912
perc_fat_in_calorie	0.910000	1.430000	4.170000

>Nutritional Breakdown of a few Cereals

```
brk_cr[brk_cr.name == '100% Natural Bran']
```

```
                name          mfr   type  calories  protein  fat
sodium  \
1  100% Natural Bran  Quaker Oats  Cold       120        3    5
15

   fiber  carbo  sugars  potass  vitamins  shelf  weight  cups
rating  \
1    2.0    8.0       8     135         0      3     1.0   1.0
33.983679
```

```
    perc_fat_in_calorie sodium_status fiber_source Potass_status
1                   4.17           Low         Good           Low
```

```
brk_cr[brk_cr.name == 'All-Bran with Extra Fiber']
```

```
                         name       mfr   type   calories   protein   fat
sodium  \
3   All-Bran with Extra Fiber   Kelloggs   Cold         50         4     0
140

    fiber   carbo   sugars   potass   vitamins   shelf   weight   cups
rating  \
3    14.0     8.0        0      330         25       3      1.0    0.5
93.704912

    perc_fat_in_calorie sodium_status fiber_source Potass_status
3                   0.0           Low    Excellent          Safe
```

```
brk_cr[brk_cr.name == 'Frosted Mini-Wheats']
```

```
                  name       mfr   type   calories   protein   fat
sodium  \
26   Frosted Mini-Wheats   Kelloggs   Cold        100         3     0
0

    fiber   carbo   sugars   potass   vitamins   shelf   weight   cups
rating  \
26    3.0    14.0        7      100         25       2      1.0    0.8
58.345141

    perc_fat_in_calorie sodium_status fiber_source Potass_status
26                  0.0           Low      Average           Low
```

>Numerical Analysis and Visualisation of Data

1. Number of Cereals in the market

```
brk_cr['name'].count()
```

```
77
```

--> There are 77 Cereals available in the market for analysis

2. Who are the manufactures producing cereals

```
brk_cr['mfr'].unique()
```

```
array(['Nabisco', 'Quaker Oats', 'Kelloggs', 'Ralston Purina',
       'General Mills', 'Post', 'American Home Food Products'],
      dtype=object)
```

--> A total of 7 manufactures are producing cereals. Some of them are widley known and advertised brands.

3. How many products are produced by each Manufacturer

```
ag= brk_cr['mfr'].value_counts()

ag

Kelloggs                     23
General Mills                22
Post                          9
Quaker Oats                   8
Ralston Purina                8
Nabisco                       6
American Home Food Products   1
Name: mfr, dtype: int64
```

```
# pie chart of Manufactures producing Cereals (%)

plt.figure(figsize=(5,4))

plt.pie(ag, labels=['Kelloggs','General Mills','Post','Quaker
Oats','Ralston Purina','Nabisco','American Home Food Products'],
autopct = '%0.0f%%')
plt.title ("Share of Manufactures in Cereal Market %")

plt.savefig("1.png", bbox_inches='tight')
```
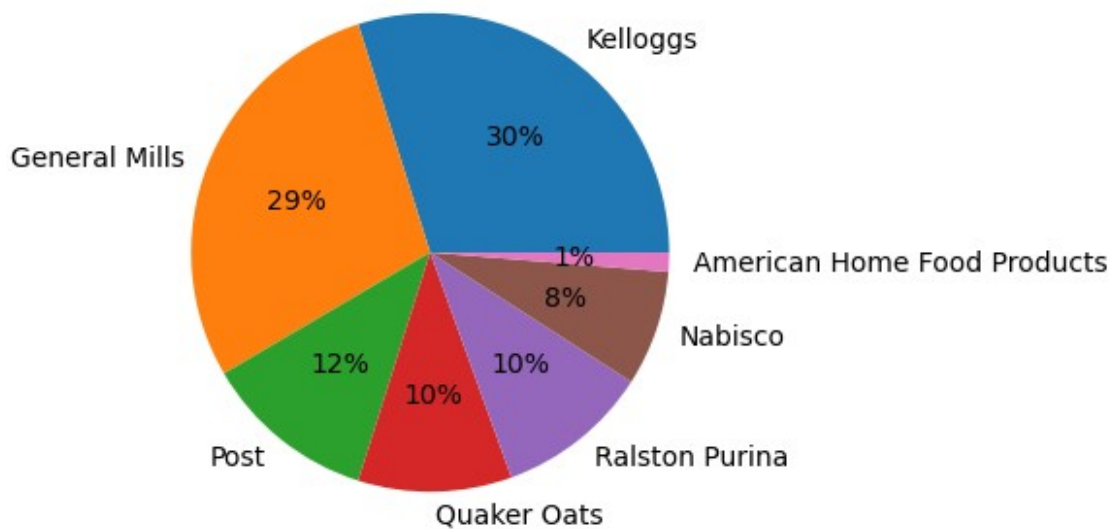
--> Out of the 7 brands, 'Kelloggs' produces the highest variety of cereals available in the market. 'General Mills' is the second in position with 22 varieties. The brand 'American Home Food Products' only presents with one cereal.

4. Number of serving types available

```
brk_cr["type"].value_counts()

Cold    74
Hot      3
Name: type, dtype: int64
```

```
# Countplot repressentation of serving types

plt.figure(figsize=(4,3))

x = sns.countplot (x = 'type', data = brk_cr )
x.set(xlabel = "Types", title = "Types of Servings in Cereals")

plt.show()

x.figure.savefig ("2.png", bbox_inches = 'tight')
```
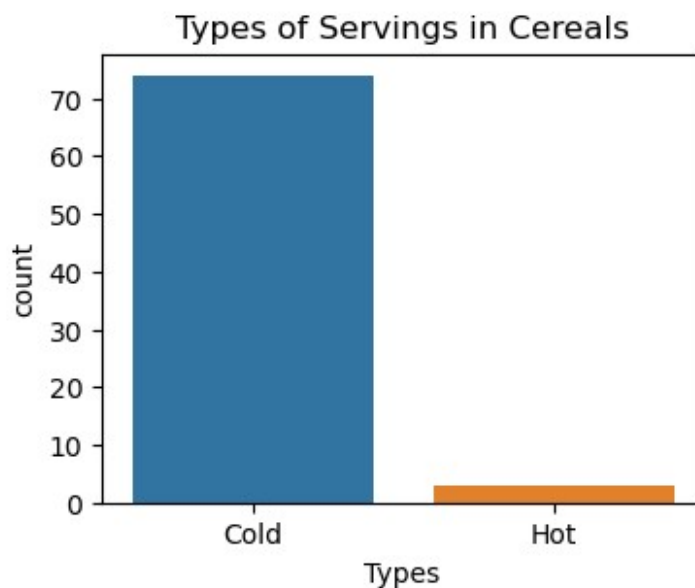

--> 2 types of cereals are in the market.

1. Cold served
2. Hot served

The most variety is in the cold type of cereals.

5. Cereals arrangement in shelfs

```
sh = brk_cr['shelf'].value_counts()

sh

3    36
2    21
1    20
Name: shelf, dtype: int64

plt.figure(figsize =(5,4))

sns.countplot(brk_cr, x='shelf', palette = 'flare')
plt.title ("Cereals Arrangement in shelf")
plt.xlabel ("Shelf Number")


plt.savefig('3.png', bbox_inches = 'tight')
```
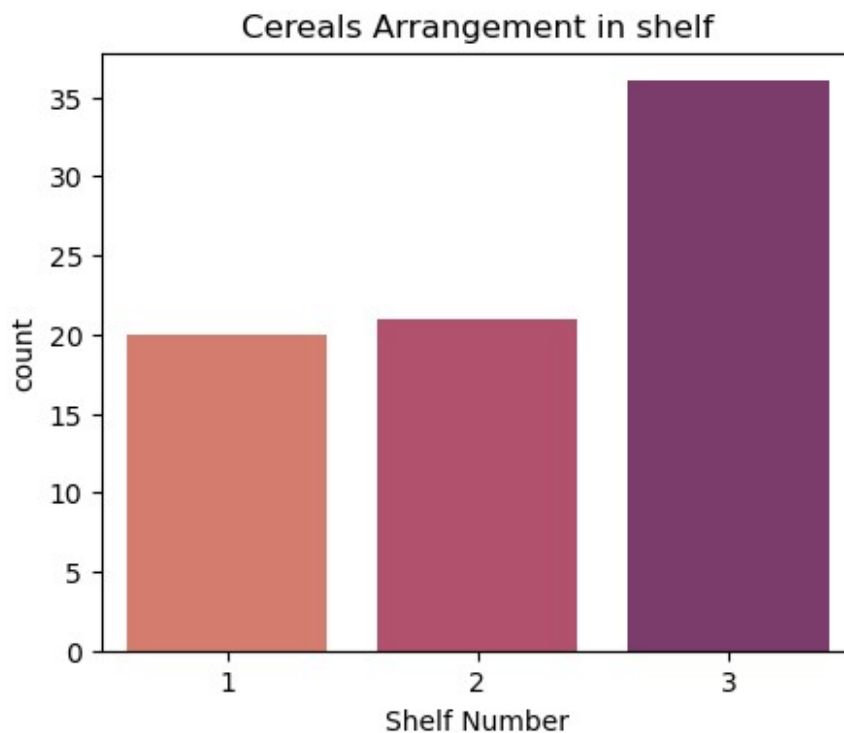

--> Most type of cereals are shown on the top shelf,ie, Shelf 3

6. Number of cups in one Serving

```
plt.figure(figsize =(5,4))

sns.countplot(brk_cr, x='cups', palette = 'cubehelix')
plt.title ("Number of cups in one Serving")
plt.xlabel ("Cups")
```

```
plt.savefig('4.png', bbox_inches = 'tight')
```


Number of cups in one Serving
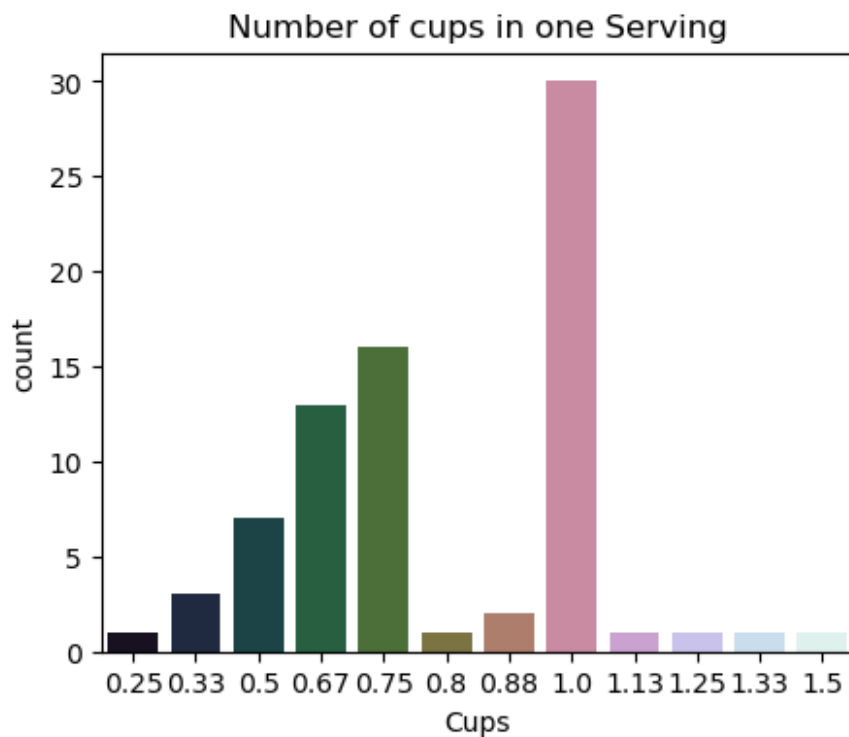
--> cereals are available in different packages. Packets and Containers are few of them. In this analysis, The quantity of 30 varieties of cereals are enough to be served in 1 cup each.

7. Calorie distribution in Cereals

```
plt.figure(figsize=(6,3))

plt.hist(brk_cr['calories'], bins=10, color='red', edgecolor='black')

plt.xlabel('Calories')
plt.title('Calorie Distribution in Cereals')


plt.savefig("5.png", bbox_inches = 'tight')
plt.show()
```

Calorie Distribution in Cereals
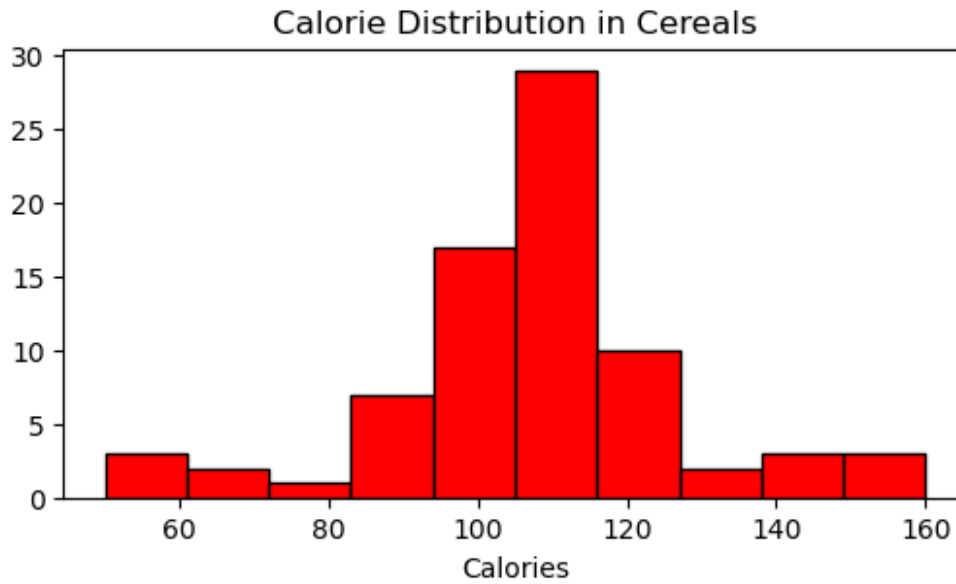
--> There is no skweness or it's Zero Skweness in the distribution of Calories in cereals

8. Sodium content in Cereals

```python
plt.figure(figsize=(6,3))

plt.hist (brk_cr['sodium'], bins= 10, color =
'green',edgecolor='black' )

plt.xlabel('Sodium')
plt.ylabel('mg')
plt.title('Sodium content in Cereal')

plt.savefig ("6.png", bbox_inches = 'tight')

plt.show()
```

Sodium content in Cereal
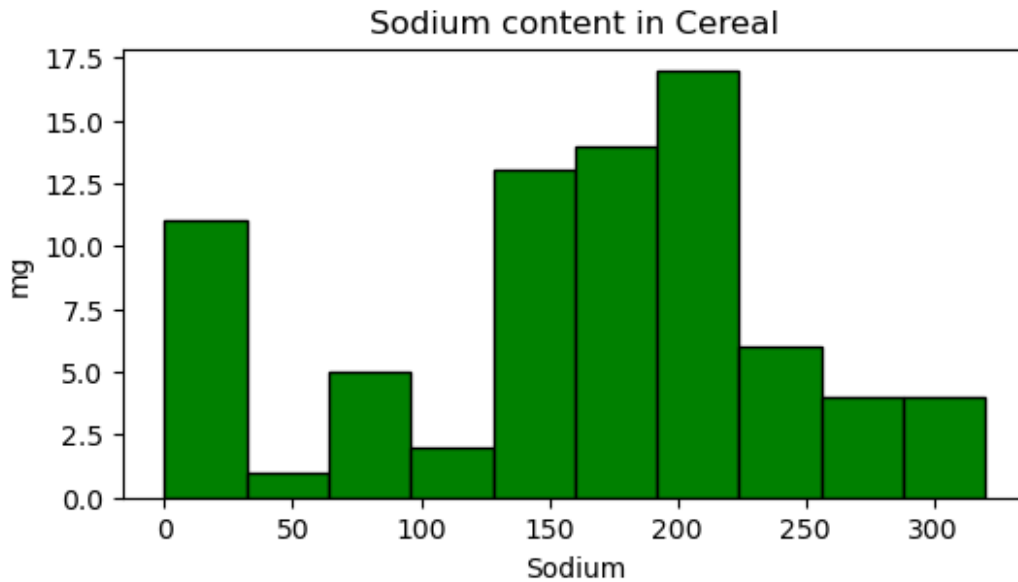
--> Right skweness

-->In the distribution of sodium in cereals, there is right skweness, which means the negative skweness.

9. Sugar in Cereals

```
plt.figure(figsize=(6,3))

plt.hist (brk_cr['sugars'], bins= 10, color = 'blue',edgecolor='black'
)

plt.xlabel('Sugar')
plt.ylabel('g')
plt.title('Suagr  in Cereal')

plt.savefig ("7.png", bbox_inches = 'tight')

plt.show()
```

Suagr in Cereal
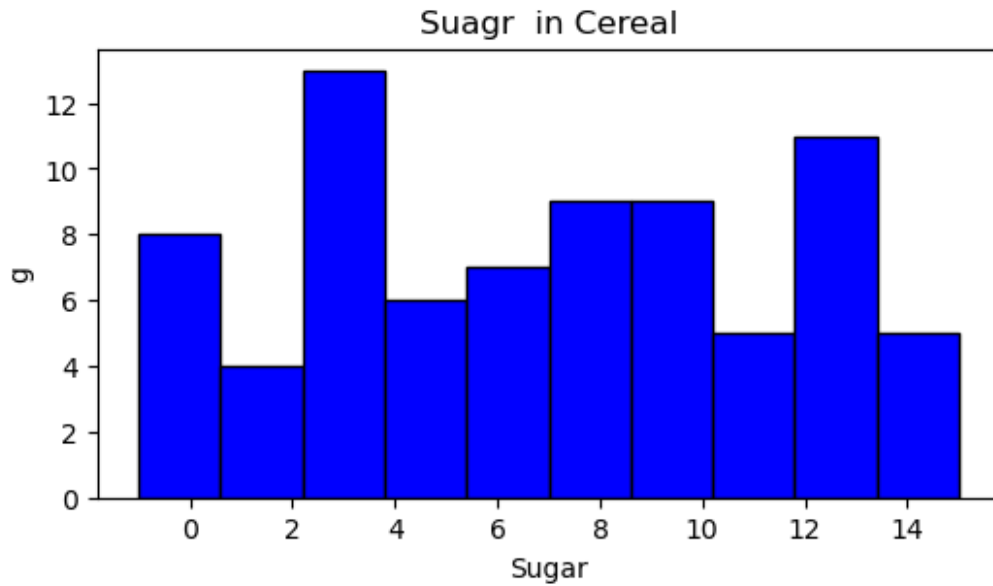
--> Zero skweness

10.Potassium content in cereals

```python
plt.figure(figsize=(6,3))

plt.hist (brk_cr['potass'], bins= 10, color =
'purple',edgecolor='black' )

plt.xlabel('Pottasium')
plt.ylabel('mg')
plt.title('Potassium content in Cereal')

plt.savefig ("8.png", bbox_inches = 'tight')

plt.show()
```

Potassium content in Cereal
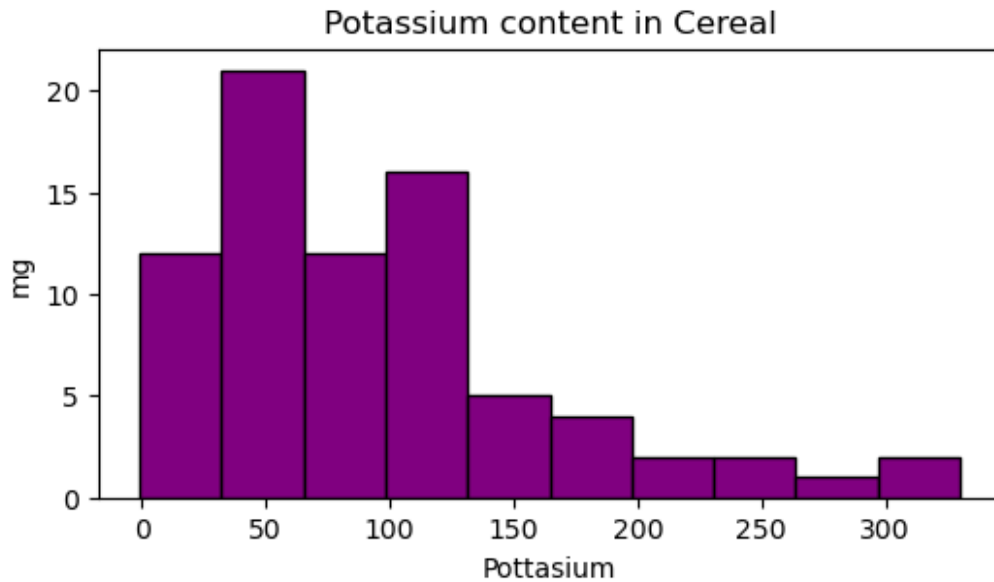
--> Left skweness

-->In the distribution of potassium in cereals, there is left skweness, which means the positive skweness.

11. Top 10 Cereals with Highest Rating

```python
# To find the Top 10 Cereals inrespect of their Ratings

tab = brk_cr.nlargest(10,'rating')

#Horizontal barplot of the finding

plt.figure (figsize=(7,4))
d= sns.barplot ( x='rating',
                 y="name",
                 data=tab,
                 orientation = 'horizontal')
d.set(xlabel = 'Customer Rating', ylabel = 'Cereal Name', title = 'Top
10 Cereals with Highest Customer Rating')

d.figure.savefig ('9.png', bbox_inches = 'tight')
```

Top 10 Cereals with Highest Customer Rating
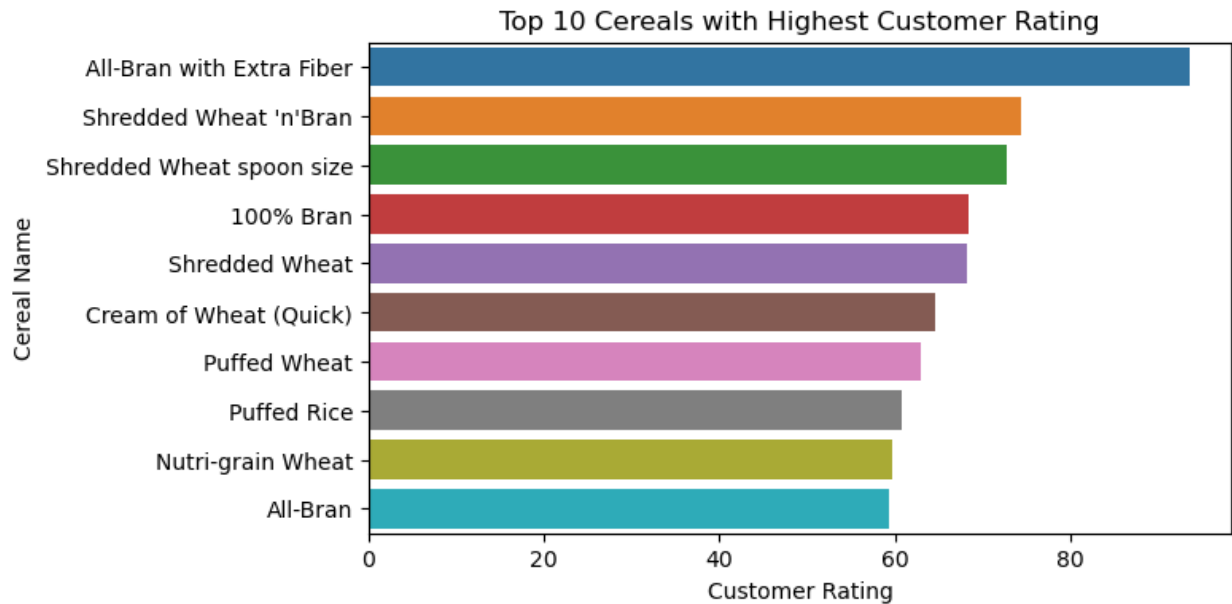
12. Top 10 Cereals with Lowest Rating

```python
# To find the Top 10 Cereals inrespect of their Ratings

tab = brk_cr.nsmallest(10,'rating')

#Horizontal barplot of the finding

plt.figure (figsize=(7,4))

e = sns.barplot ( x='rating',
                  y="name",
                  data=tab,
                  orientation = 'horizontal')
e.set(xlabel = 'Customer Rating', ylabel = 'Cereal Name', title = 'Top
10 Cereals with Lowest Customer Rating')

e.figure.savefig ('10.png', bbox_inches = 'tight')
```

Top 10 Cereals with Lowest Customer Rating
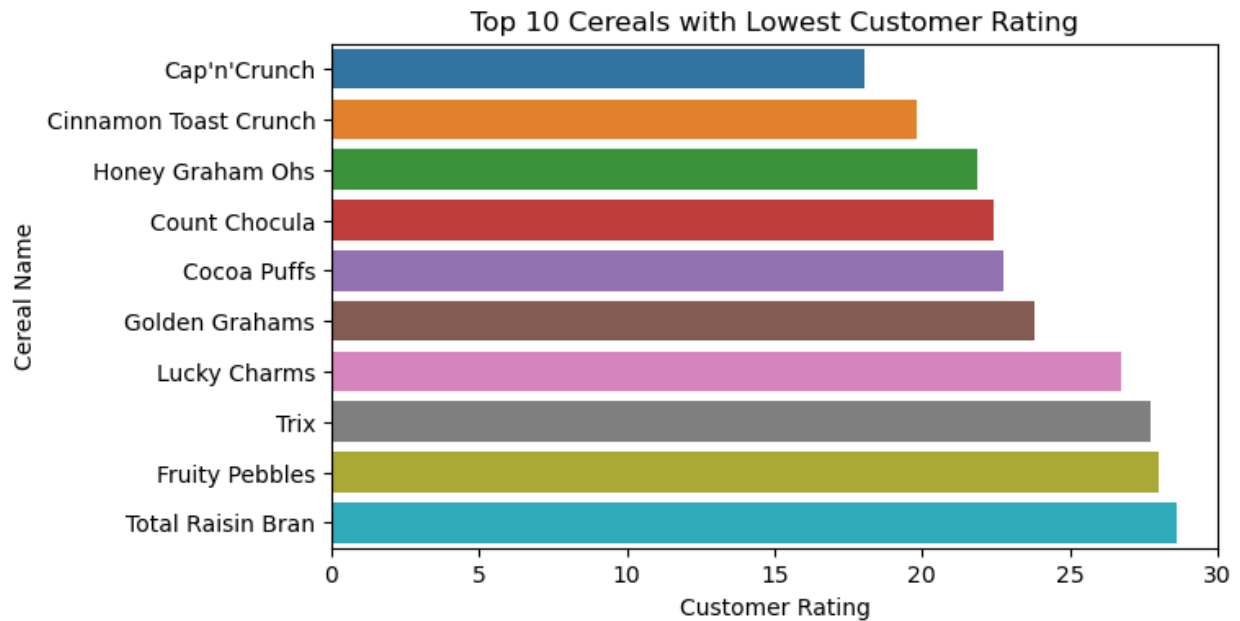
13. Top 10 Cereals with Highest protein

```python
tab = brk_cr.nlargest (10, 'protein')

plt.figure (figsize=(7,4))
f = sns.barplot (
                x = 'protein',
                y = 'name',
                data = tab,
                orientation = 'horizontal',
                palette = 'hls')
f.set(xlabel = 'Protein', ylabel = 'Cereal Name', title = 'Top 10
Cereals with Highest protein')

f.figure.savefig('11.png', bbox_inches = 'tight')
```

Top 10 Cereals with Highest protein
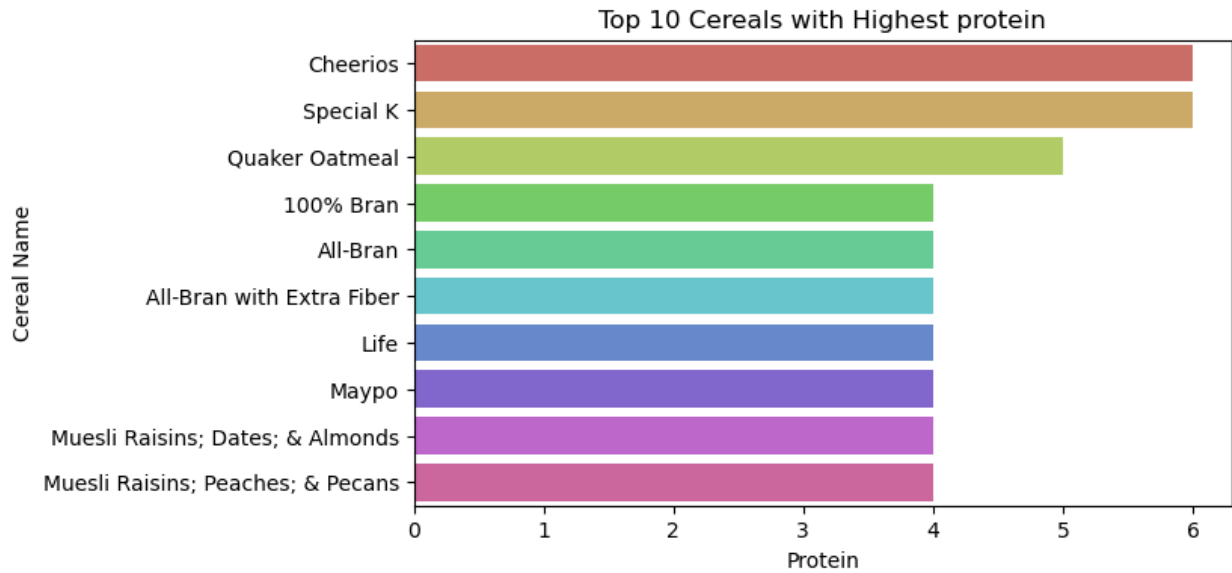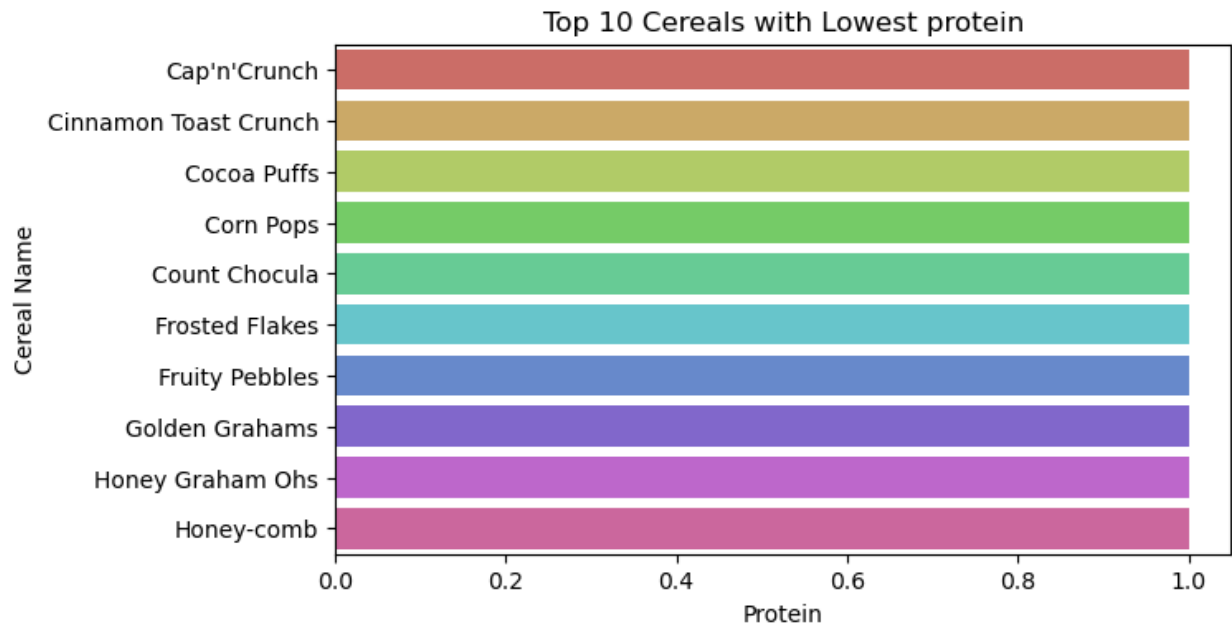
14. Top 10 Cereals with Lowest Protein

```
tab = brk_cr.nsmallest (10, 'protein')

plt.figure (figsize=(7,4))
f = sns.barplot (
                 x = 'protein',
                 y = 'name',
                 data = tab,
                 orientation = 'horizontal',
                 palette = 'hls')
f.set(xlabel = 'Protein', ylabel = 'Cereal Name', title = 'Top 10
Cereals with Lowest protein')

f.figure.savefig('12.png', bbox_inches = 'tight')
```
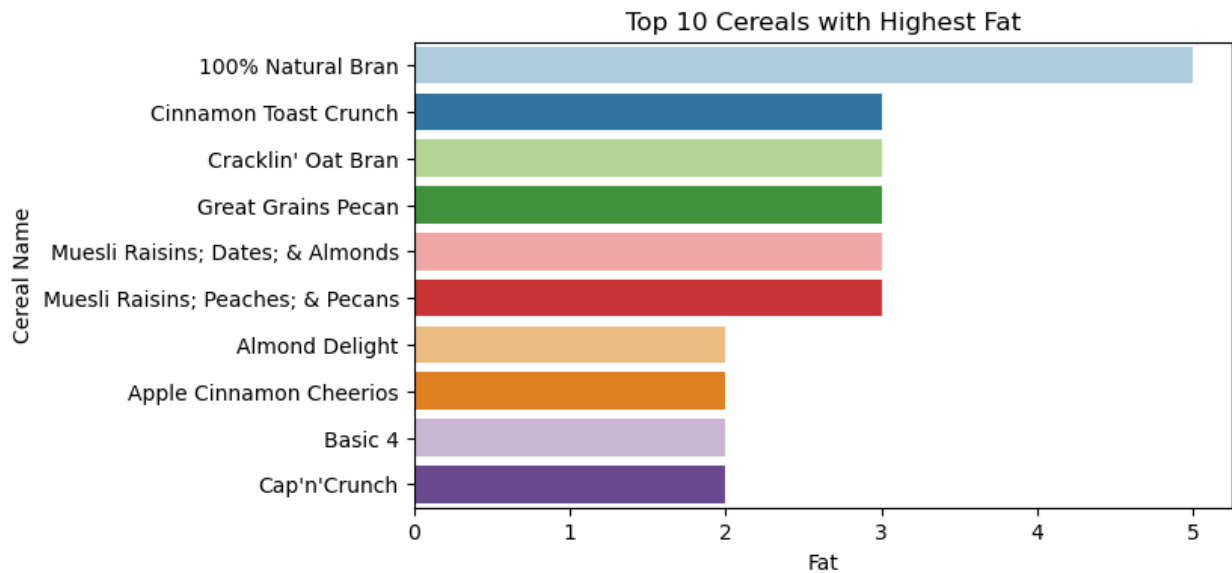
Top 10 Cereals with Lowest protein

15. Top 10 Cereals with Highest Fat

```
tab = brk_cr.nlargest (10,'fat')

plt.figure (figsize=(7,4))
a = sns.barplot (
                  x = 'fat',
                  y = 'name',
                  data = tab,
                  orientation = 'horizontal',
                  palette = 'Paired')

a.set(xlabel = 'Fat' , ylabel = 'Cereal Name', title = 'Top 10 Cereals
with Highest Fat')

a.figure.savefig ('13.png', bbox_inches = 'tight')
```
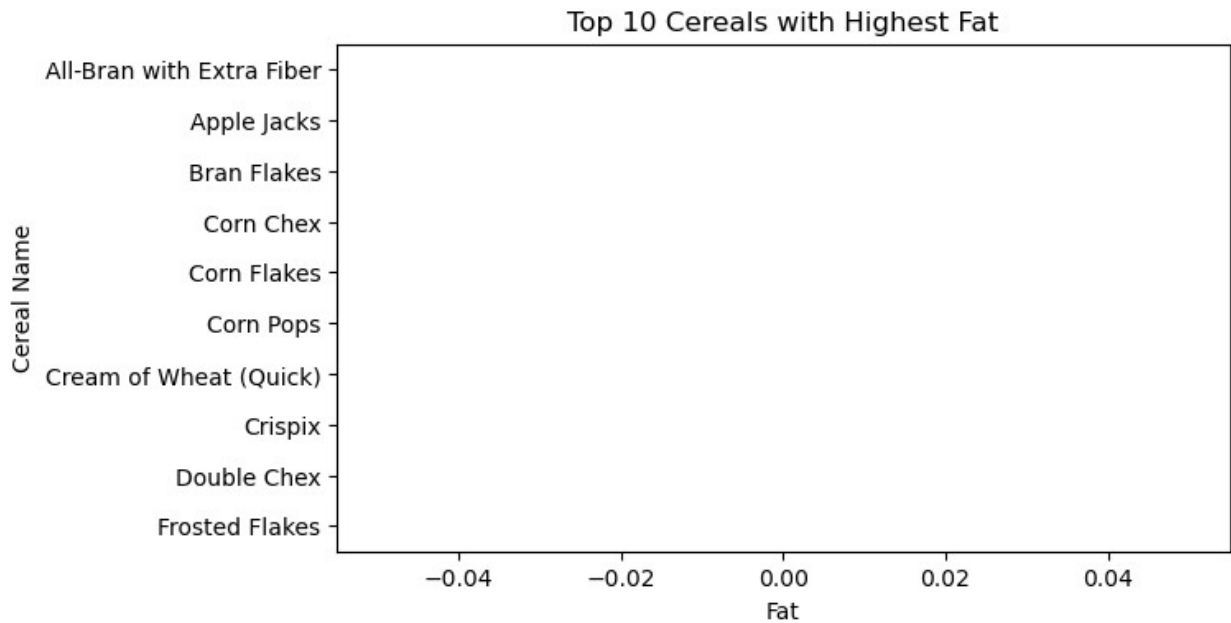
Top 10 Cereals with Highest Fat

16. Top 10 Cereals with Lowest Fat

```
tab = brk_cr.nsmallest (10,'fat')

plt.figure (figsize=(7,4))
b = sns.barplot (
                 x = 'fat',
                 y = 'name',
                 data = tab,
                 orientation = 'horizontal',
                 palette = 'Paired')

b.set(xlabel = 'Fat' , ylabel = 'Cereal Name', title = 'Top 10 Cereals
with Highest Fat')

b.figure.savefig ('14.png', bbox_inches = 'tight')
```
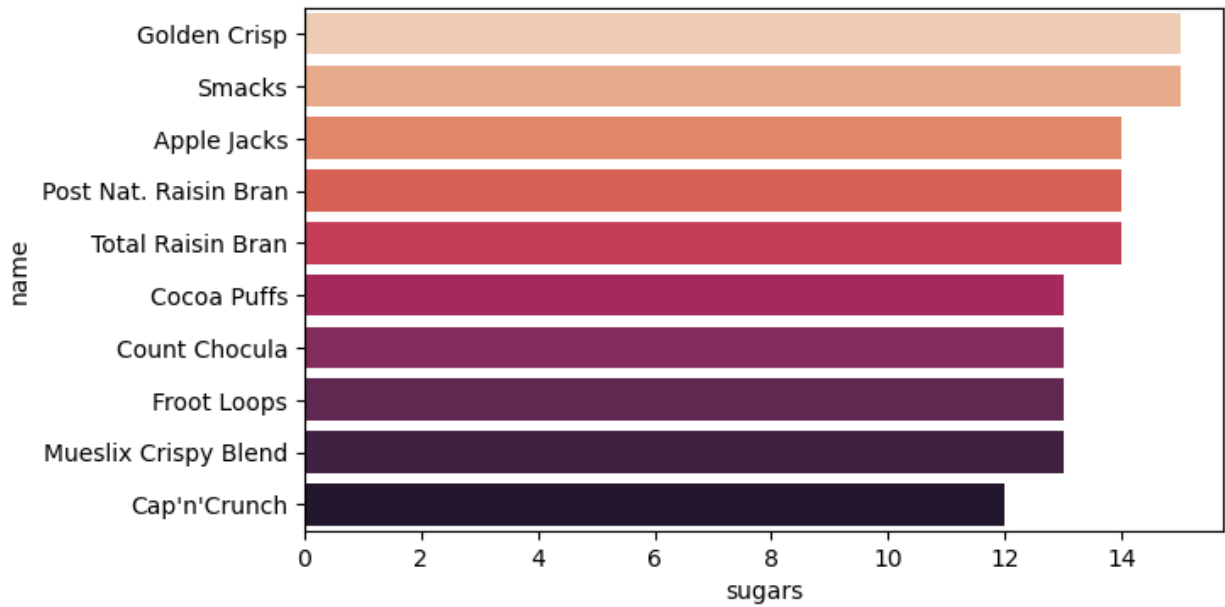
Top 10 Cereals with Highest Fat

17. Top 10 Cereals with Highest Sugar

```python
tab = brk_cr.nlargest (10,'sugars')

plt.figure (figsize=(7,4))
c = sns.barplot (
                    x = 'sugars',
                    y = 'name',
                    data = tab,
                    orientation = 'horizontal',
                    palette = 'rocket_r')

a.set(xlabel = 'Sugar' , ylabel = 'Cereal Name', title = 'Top 10
Cereals with Highest Sugar')

a.figure.savefig ('15.png', bbox_inches = 'tight')
```
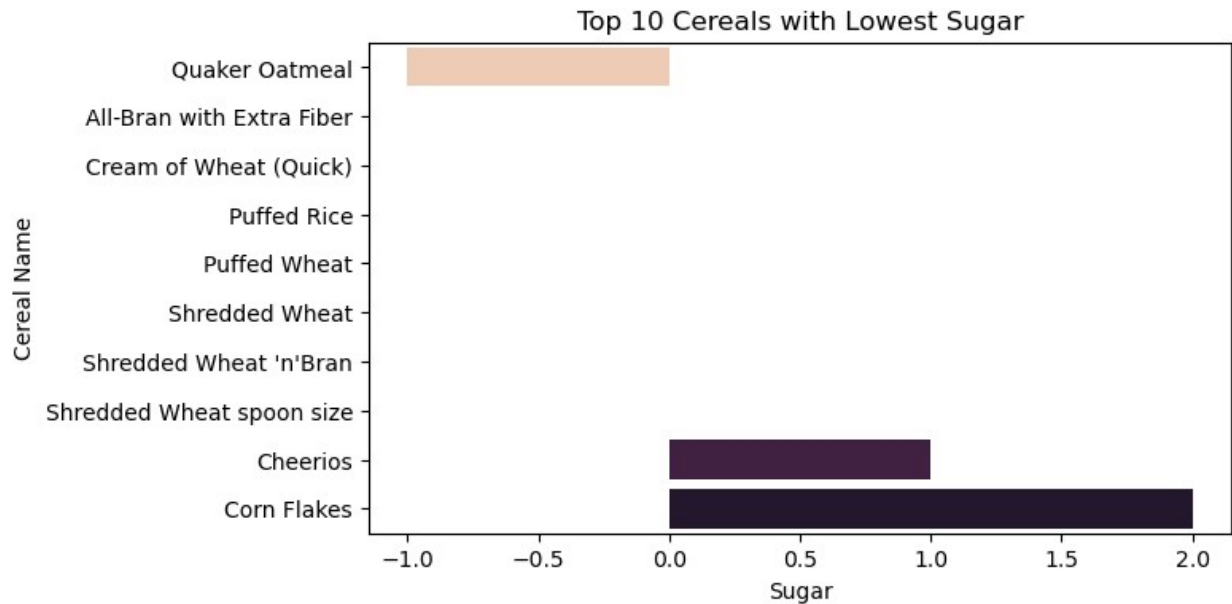
18. Top 10 Cereals with Lowest Sugar

```python
tab = brk_cr.nsmallest (10,'sugars')

plt.figure (figsize=(7,4))
s = sns.barplot (
                x = 'sugars',
                y = 'name',
                data = tab,
                orientation = 'horizontal',
                palette = 'rocket_r')

s.set(xlabel = 'Sugar' , ylabel = 'Cereal Name', title = 'Top 10
Cereals with Lowest Sugar')

s.figure.savefig ('16.png', bbox_inches = 'tight')
```
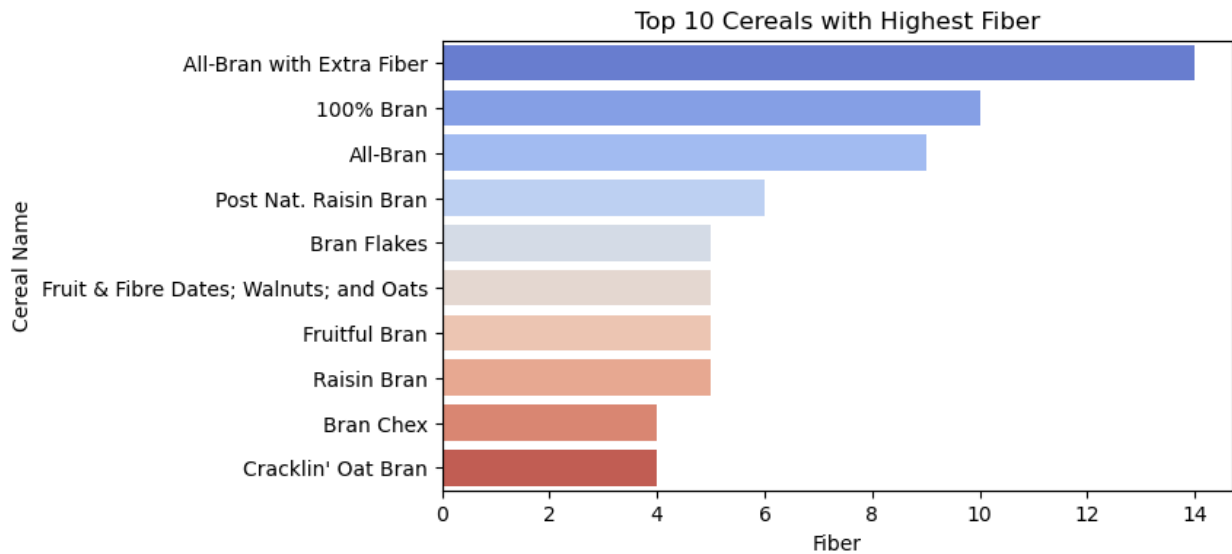
Top 10 Cereals with Lowest Sugar

19. Top 10 Cereals with Highest Fiber

```
tab = brk_cr.nlargest (10,'fiber')

plt.figure (figsize=(7,4))
n = sns.barplot (
                x = 'fiber',
                y = 'name',
                data = tab,
                orientation = 'horizontal',
                palette = 'coolwarm')

n.set(xlabel = 'Fiber' , ylabel = 'Cereal Name', title = 'Top 10
Cereals with Highest Fiber')

n.figure.savefig ('17.png', bbox_inches = 'tight')
```
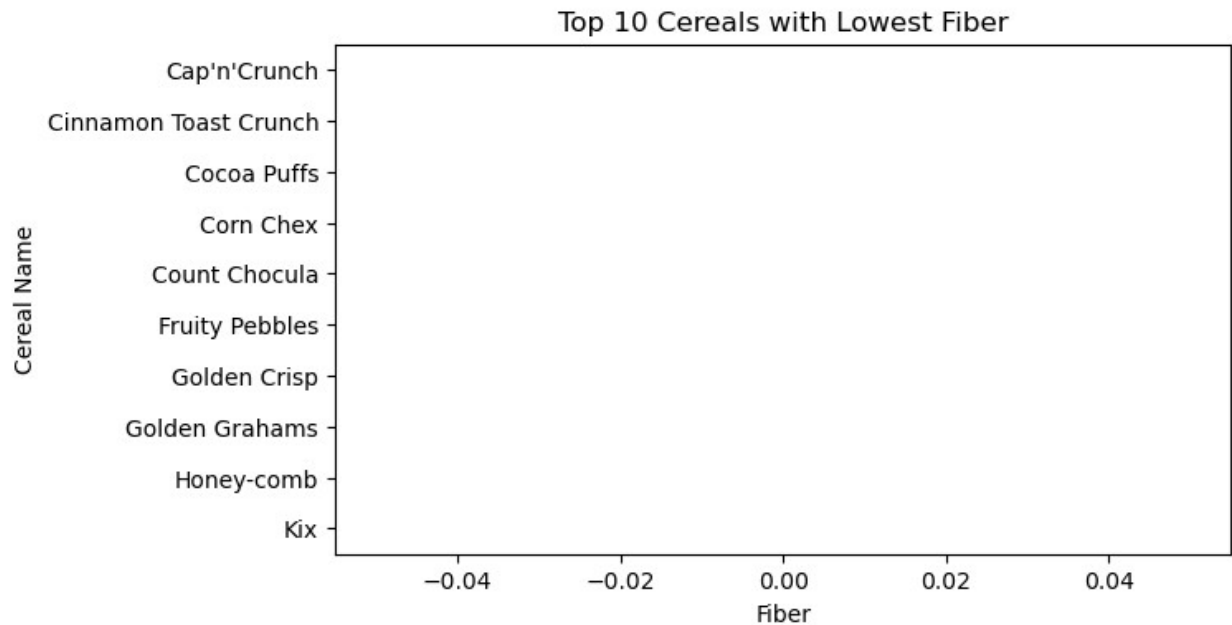
Top 10 Cereals with Highest Fiber

20. Top 10 Cereals with Lowest Rating

```python
tab = brk_cr.nsmallest (10,'fiber')

plt.figure (figsize=(7,4))
m = sns.barplot (
                x = 'fiber',
                y = 'name',
                data = tab,
                orientation = 'horizontal',
                palette = 'coolwarm')

m.set(xlabel = 'Fiber' , ylabel = 'Cereal Name', title = 'Top 10
Cereals with Lowest Fiber')

m.figure.savefig ('18.png', bbox_inches = 'tight')
```

Top 10 Cereals with Lowest Fiber

21.Which cereal has the highest Vitamins&Minerals prescribed by FDA

```
tab = brk_cr[brk_cr['vitamins'] == 100]



plt.figure (figsize=(7,4))
m = sns.barplot (
                x = 'vitamins',
                y = 'name',
                data = tab.head(10),
                orientation = 'horizontal',
                palette = 'Paired')

m.set(xlabel = 'vitamins' , ylabel = 'Cereal Name', title = 'Cereals
with Vitamins prescribed by FDA')

m.figure.savefig ('19.png', bbox_inches = 'tight')
```

Cereals with Vitamins prescribed by FDA
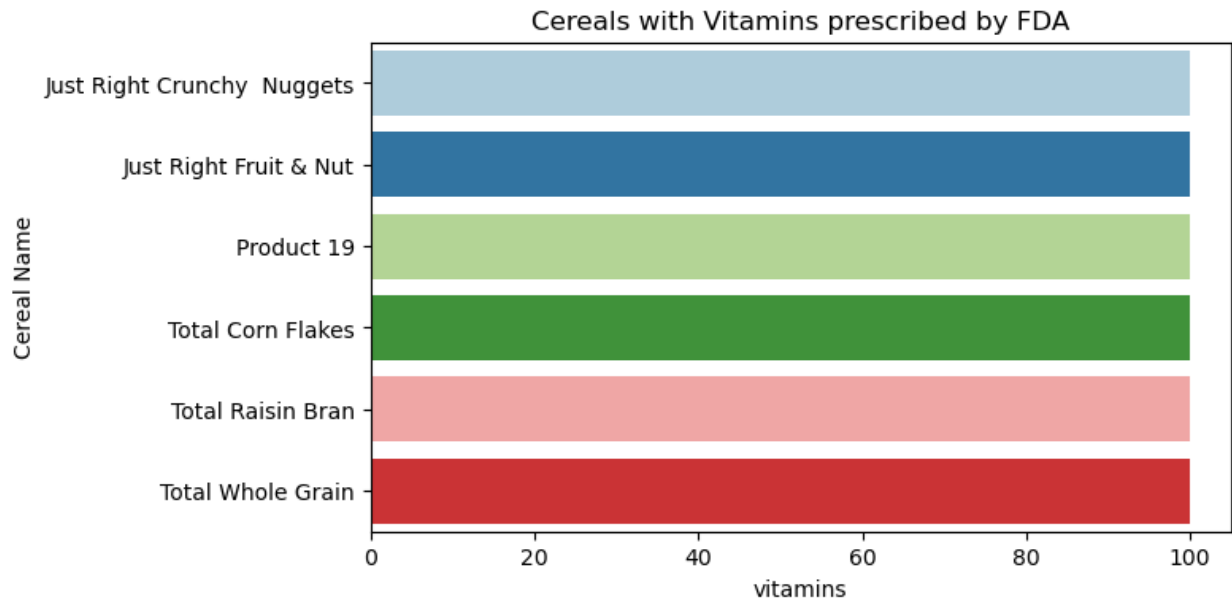
22. Number of cereals in each fiber Category

```
b = brk_cr.groupby (by = 'fiber_source')['fiber'].count()
b

fiber_source
Average      21
Excellent     8
Good         48
Name: fiber, dtype: int64
```
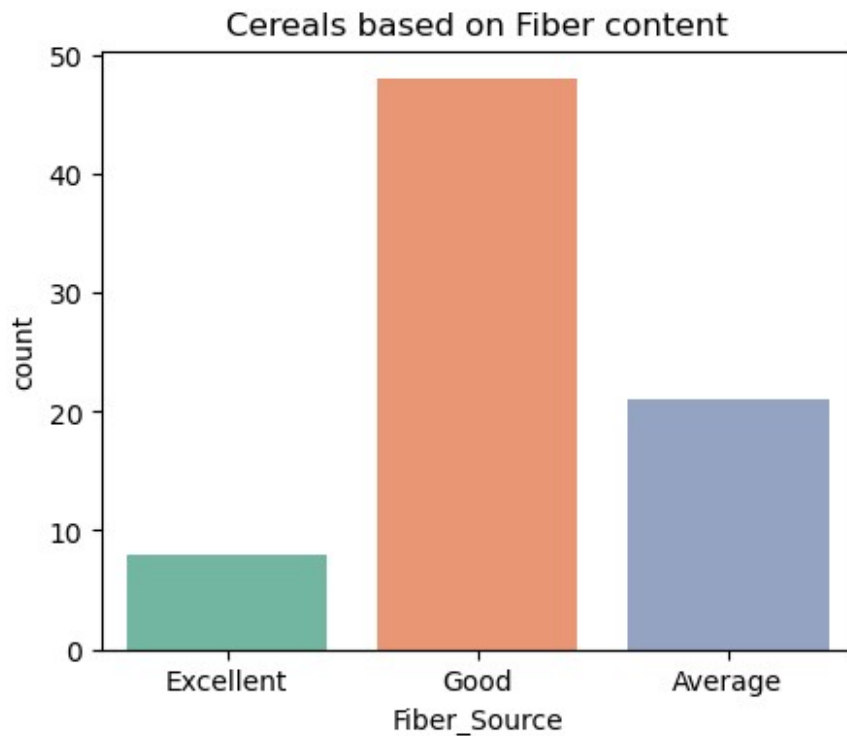
```
# Countplot respressentation of fiber content in cereals

plt.figure(figsize=(5,4))

q = sns.countplot (x = 'fiber_source', data = brk_cr, palette =
"Set2")
q.set(xlabel = "Fiber_Source",title = "Cereals based on Fiber
content")

q.figure.savefig("20.png", bbox_inches='tight')

plt.show()
```

Cereals based on Fiber content
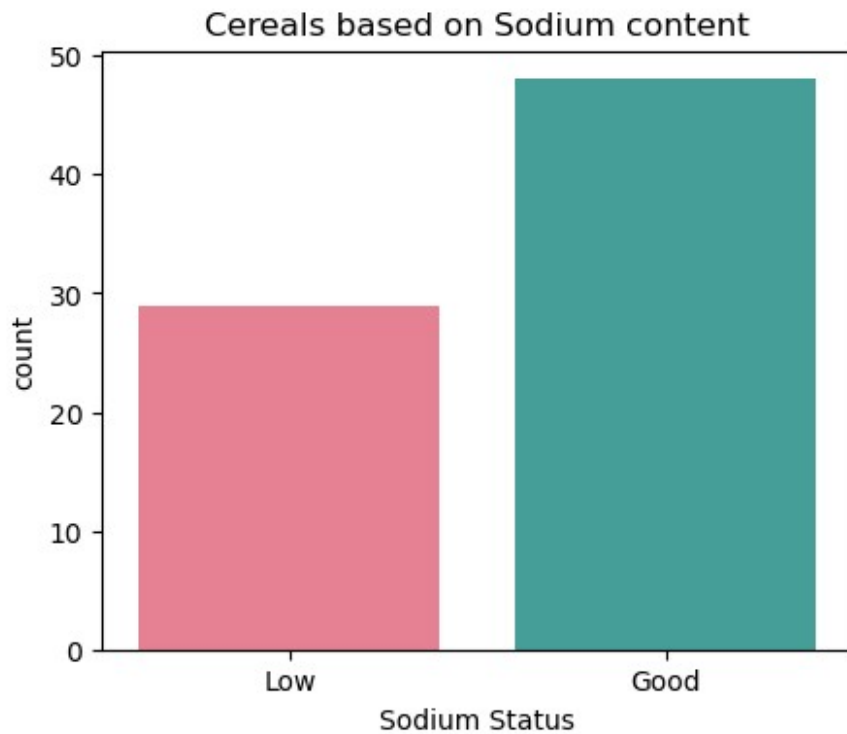
23. Number of cereals falling under different Sodium Category

```
sod = brk_cr['sodium_status'].value_counts()
sod

Good     48
Low      29
Name: sodium_status, dtype: int64

plt.figure(figsize =(5,4))
sns.countplot(brk_cr, x='sodium_status', palette = 'husl')
plt.title ("Cereals based on Sodium content")
plt.xlabel ("Sodium Status")

plt.savefig('21.png', bbox_inches = 'tight')
```

Cereals based on Sodium content

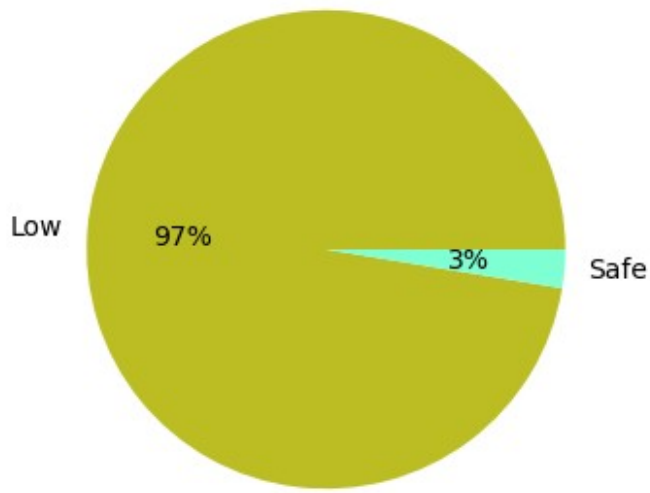24. How many products have a safe level of Potassium content

```python
a = brk_cr['Potass_status'].value_counts()


plt.figure(figsize=(5,4))

plt.pie(a, labels=['Low','Safe'], autopct = '%0.0f%%',
colors=['tab:olive','aquamarine'])
plt.title ("Potassium Level in Cereals %")

plt.savefig("22.png", bbox_inches='tight')
```

Potassium Level in Cereals %


```python
#How many products have a safe level of Potassium content
print(brk_cr[brk_cr['Potass_status']=='Safe']['name'].count())
#What are they?
brk_cr[brk_cr['Potass_status']=='Safe']
```

```
2
```

```
                        name       mfr  type  calories  protein  fat
sodium  \
2                    All-Bran  Kelloggs  Cold        70        4    1
260
3  All-Bran with Extra Fiber  Kelloggs  Cold        50        4    0
140

   fiber  carbo  sugars  potass  vitamins  shelf  weight  cups
rating  \
2    9.0    7.0       5     320        25      3     1.0  0.33
59.425505
3   14.0    8.0       0     330        25      3     1.0  0.50
93.704912

   perc_fat_in_calorie sodium_status fiber_source Potass_status
2                 1.43          Good    Excellent          Safe
3                 0.00           Low    Excellent          Safe
```

25. How many products are following the health guidelines interms of sodium, potassium and fiber content?

```
brk_cr[(brk_cr['Potass_status'] == 'Safe') & (brk_cr['sodium_status']
== 'Good') & (brk_cr['fiber_source'] == 'Excellent')]
```

```
        name        mfr   type  calories  protein  fat  sodium  fiber
carbo  \
2  All-Bran  Kelloggs  Cold        70        4    1     260    9.0
7.0

    sugars  potass  vitamins  shelf  weight  cups     rating  \
2        5     320        25      3     1.0  0.33  59.425505

    perc_fat_in_calorie sodium_status fiber_source Potass_status
2                  1.43          Good    Excellent          Safe
```

26. How many products are lacking in terms of sodium, potassium and fiber content?

```
count = brk_cr[(brk_cr['Potass_status'] == 'Less') &
(brk_cr['sodium_status'] == 'Low') & (brk_cr['fiber_source'] ==
'Average')]['name'].count()

print('A total of',count,"Cereals doesn't meet the standareds")

brk_cr[(brk_cr['Potass_status'] == 'Low') & (brk_cr['sodium_status']
== 'Low') & (brk_cr['fiber_source'] == 'Average')]['name']
```

```
A total of 0 Cereals doesn't meet the standareds

19                  Cracklin' Oat Bran
26               Frosted Mini-Wheats
32                 Grape Nuts Flakes
34                 Great Grains Pecan
44    Muesli Raisins; Dates; & Almonds
57                     Quaker Oatmeal
59                   Raisin Nut Bran
63                    Shredded Wheat
64             Shredded Wheat 'n'Bran
65          Shredded Wheat spoon size
68            Strawberry Fruit Wheats
Name: name, dtype: object
```

27. Is there any cereals with no sugar

```
brk_cr[brk_cr['sugars'] <= 0]
```

```
                          name        mfr   type  calories  protein
fat  \
3   All-Bran with Extra Fiber    Kelloggs  Cold        50        4
0
20        Cream of Wheat (Quick)     Nabisco   Hot       100        3
```

```
0
54                   Puffed Rice  Quaker Oats  Cold        50           1
0
55                  Puffed Wheat  Quaker Oats  Cold        50           2
0
57                Quaker Oatmeal  Quaker Oats  Hot        100           5
2
63                Shredded Wheat      Nabisco  Cold        80           2
0
64       Shredded Wheat 'n'Bran      Nabisco  Cold        90           3
0
65   Shredded Wheat spoon size      Nabisco  Cold        90           3
0

     sodium  fiber  carbo  sugars  potass  vitamins  shelf  weight
cups  \
3       140   14.0    8.0       0     330        25      3    1.00
0.50
20       80    1.0   21.0       0      -1         0      2    1.00
1.00
54        0    0.0   13.0       0      15         0      3    0.50
1.00
55        0    1.0   10.0       0      50         0      3    0.50
1.00
57        0    2.7   -1.0      -1     110         0      1    1.00
0.67
63        0    3.0   16.0       0      95         0      1    0.83
1.00
64        0    4.0   19.0       0     140         0      1    1.00
0.67
65        0    3.0   20.0       0     120         0      1    1.00
0.67

        rating   perc_fat_in_calorie sodium_status fiber_source
Potass_status
3    93.704912                   0.0           Low    Excellent
Safe
20   64.533816                   0.0           Low         Good
Low
54   60.756112                   0.0           Low         Good
Low
55   63.005645                   0.0           Low         Good
Low
57   50.828392                   2.0           Low      Average
Low
63   68.235885                   0.0           Low      Average
Low
64   74.472949                   0.0           Low      Average
Low
```

```
65  72.801787                                0.0          Low         Average
Low
```

28. Is there any cereals with no Sodium

```python
brk_cr[brk_cr['sodium'] <= 0]
```

```
                           name                            mfr   type
calories  \
26          Frosted Mini-Wheats                       Kelloggs   Cold
100
43                        Maypo   American Home Food Products    Hot
100
54                  Puffed Rice                     Quaker Oats  Cold
50
55                 Puffed Wheat                     Quaker Oats  Cold
50
57               Quaker Oatmeal                     Quaker Oats   Hot
100
60               Raisin Squares                       Kelloggs   Cold
90
63               Shredded Wheat                        Nabisco   Cold
80
64      Shredded Wheat 'n'Bran                        Nabisco   Cold
90
65   Shredded Wheat spoon size                        Nabisco   Cold
90

    protein  fat  sodium  fiber  carbo  sugars  potass  vitamins
shelf  \
26        3    0       0    3.0   14.0       7     100        25
2
43        4    1       0    0.0   16.0       3      95        25
2
54        1    0       0    0.0   13.0       0      15         0
3
55        2    0       0    1.0   10.0       0      50         0
3
57        5    2       0    2.7   -1.0      -1     110         0
1
60        2    0       0    2.0   15.0       6     110        25
3
63        2    0       0    3.0   16.0       0      95         0
1
64        3    0       0    4.0   19.0       0     140         0
1
65        3    0       0    3.0   20.0       0     120         0
1

    weight   cups     rating  perc_fat_in_calorie sodium_status
```

```
     fiber_source  \
26    1.00  0.80   58.345141                  0.0              Low
Average
43    1.00  1.00   54.850917                  1.0              Low
Good
54    0.50  1.00   60.756112                  0.0              Low
Good
55    0.50  1.00   63.005645                  0.0              Low
Good
57    1.00  0.67   50.828392                  2.0              Low
Average
60    1.00  0.50   55.333142                  0.0              Low
Good
63    0.83  1.00   68.235885                  0.0              Low
Average
64    1.00  0.67   74.472949                  0.0              Low
Average
65    1.00  0.67   72.801787                  0.0              Low
Average

     Potass_status
26            Low
43            Low
54            Low
55            Low
57            Low
60            Low
63            Low
64            Low
65            Low
```

>Statistical Analysis and Visualisation of Data

29. Average amount of sugar in one serving

```
brk_cr['sugars'].mean()
```

6.922077922077922

30. Average amount of sodium in one serving

```
brk_cr['sodium'].mean()
```

159.67532467532467

31. Average amount of calorie in one serving

```
brk_cr['calories'].mean()
```

106.88311688311688

32. Average amount of fiber in one serving

```
brk_cr['fiber'].mean()
```

```
2.1519480519480516
```

33. Average amount of fat in one serving

```
brk_cr['fat'].mean()
```

```
1.0129870129870129
```
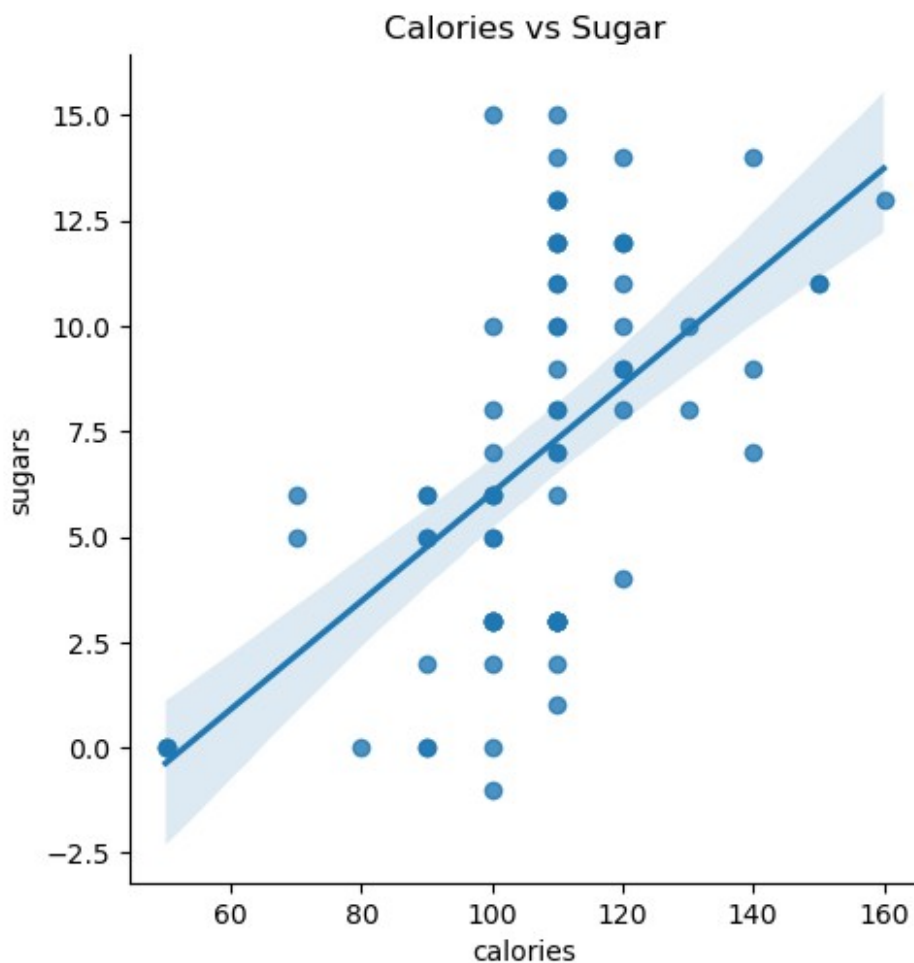
34. Correlation between Calories and Sugar

```
print('The corr : ',brk_cr['calories'].corr(brk_cr['sugars']))

sns.lmplot(x="calories", y="sugars", data=brk_cr)
plt.title("Calories vs Sugar")

plt.savefig("23.png", bbox_inches='tight')
```

```
The corr :  0.5623402898034883
```
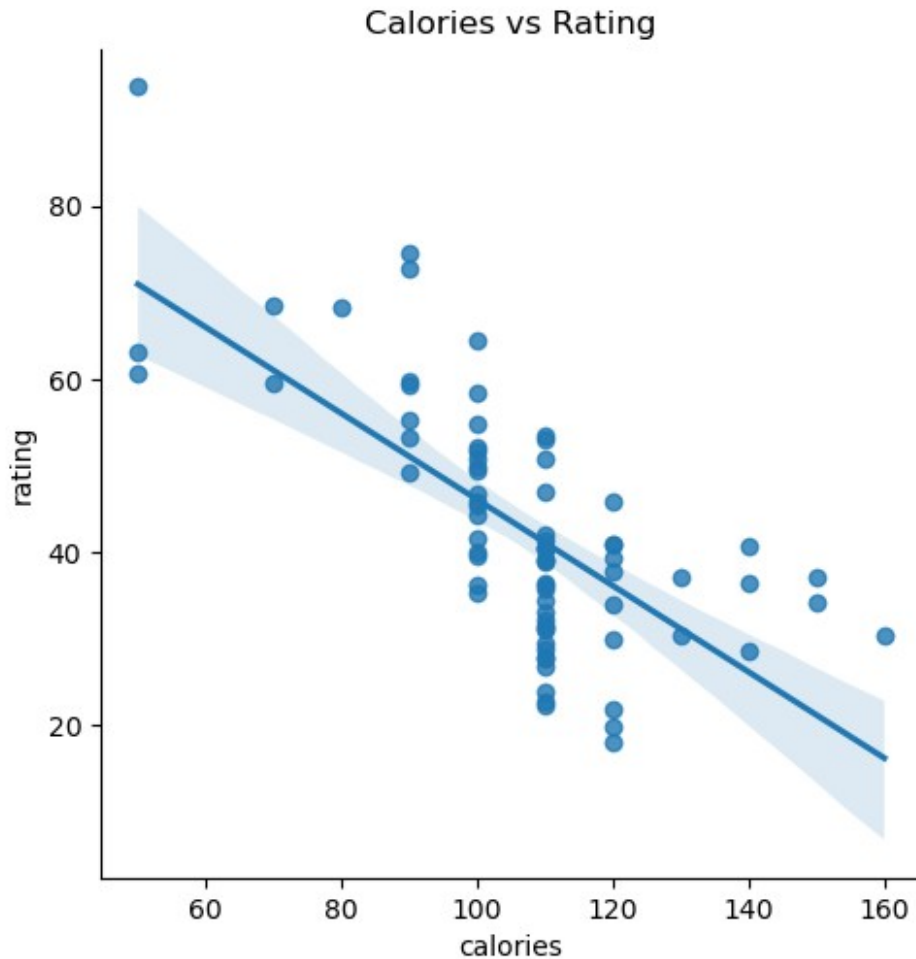
35. Correlation between Calories and Rating

```
print('The corr : ',brk_cr['calories'].corr(brk_cr['rating']))

sns.lmplot(x="calories", y="rating", data=brk_cr)
plt.title("Calories vs Rating")

plt.savefig("24.png", bbox_inches='tight')
The corr :  -0.6893760311652586
```
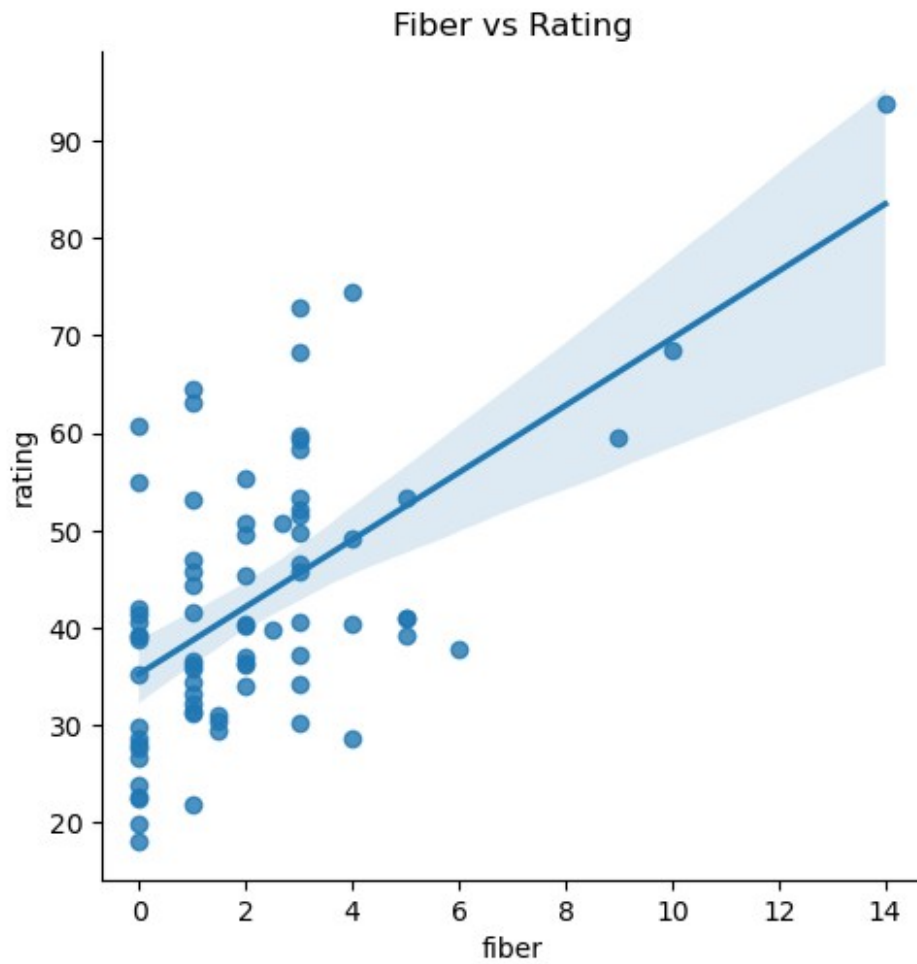

36. Correlation between Brand Fiber and Rating

```
print('The corr : ',brk_cr['fiber'].corr(brk_cr['rating']))

sns.lmplot(x="fiber", y="rating", data=brk_cr)
plt.title("Fiber vs Rating")

plt.savefig("25.png", bbox_inches='tight')
The corr :  0.5841604199515837
```
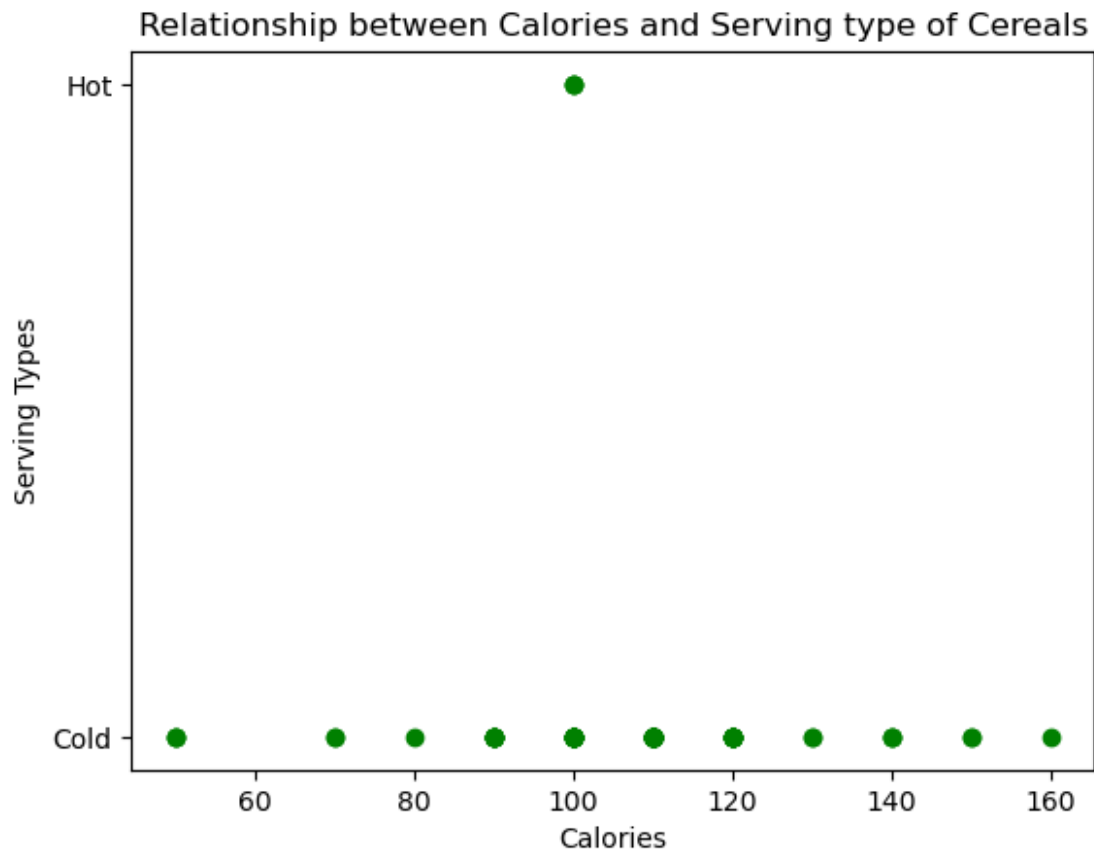
Fiber vs Rating

37.Relationship between Calories and Serving type of Cereals

```
plt.scatter(x = 'calories', y = 'type', data = brk_cr, c='green')

plt.xlabel('Calories')
plt.ylabel('Serving Types')
plt.title('Relationship between Calories and Serving type of Cereals')

plt.savefig("26.png", bbox_inches='tight')
```

Relationship between Calories and Serving type of Cereals

38. Correlation between Brand Name and Rating

```python
plt.scatter(x = 'rating', y = 'mfr', data = brk_cr, c='green')

plt.xlabel('Rating')
plt.ylabel('Brand Name')
plt.title('Brand power vs Rating')

plt.savefig("27.png", bbox_inches='tight')
```
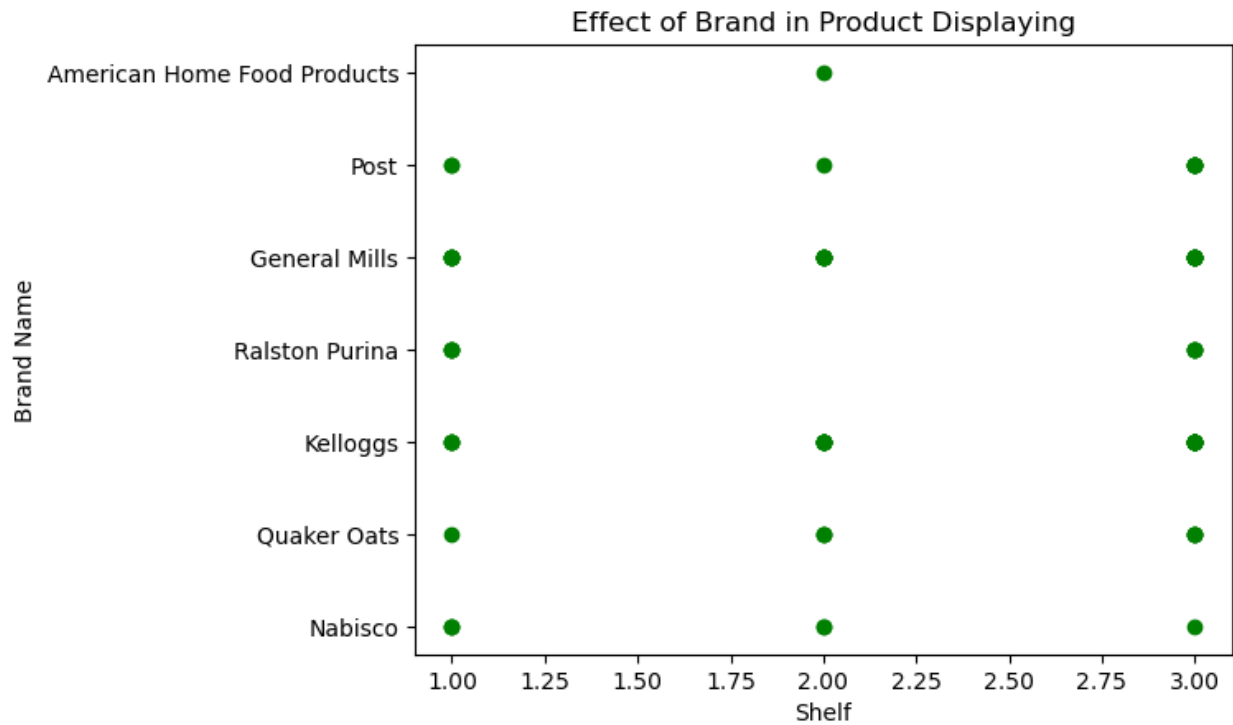
Brand power vs Rating
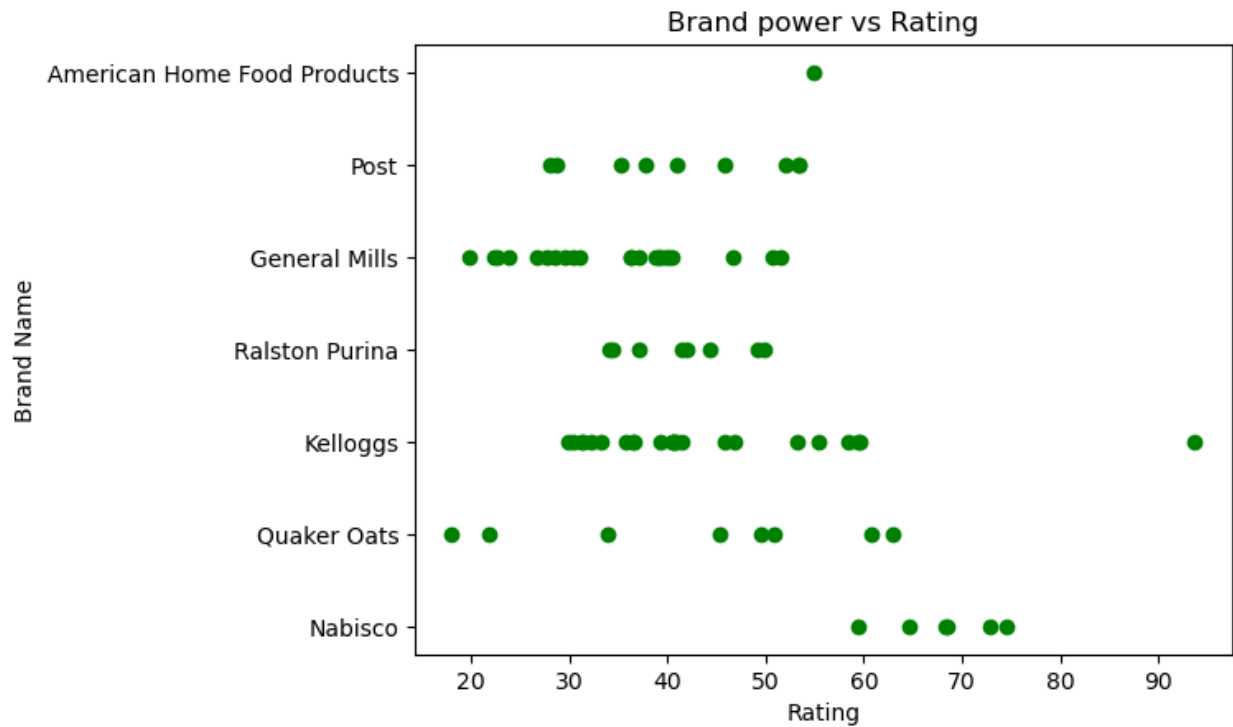
39. Correlation between Brand Name and Display

```
plt.scatter(x = 'shelf', y = 'mfr', data = brk_cr, c='green')

plt.xlabel('Shelf')
plt.ylabel('Brand Name')
plt.title('Effect of Brand in Product Displaying')

plt.savefig("28.png", bbox_inches='tight')
```
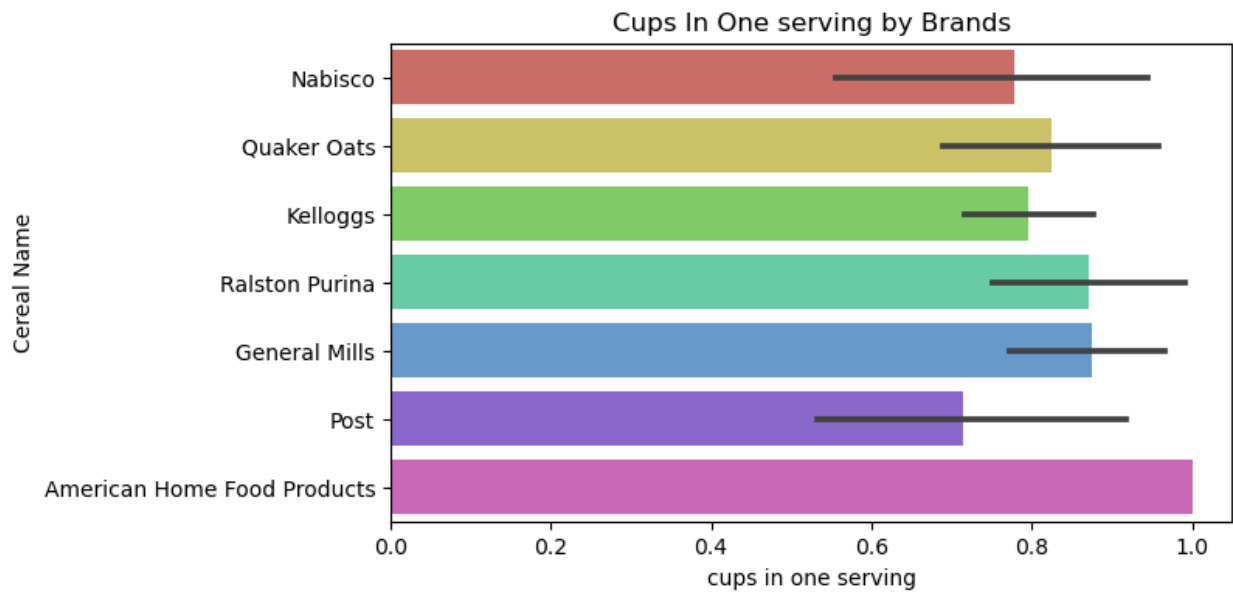
Effect of Brand in Product Displaying

40. Cups In One serving by Brands

```
plt.figure (figsize=(7,4))
f = sns.barplot (
                  x = 'cups',
                  y = 'mfr',
                  data = brk_cr,
                  palette = 'hls')
f.set(xlabel = 'cups in one serving', ylabel = 'Cereal Name', title =
'Cups In One serving by Brands')

f.figure.savefig('29.png', bbox_inches = 'tight')
```

Cups In One serving by Brands

41. Correlation matrix of Numerical features

```python
# subset containing the the numerical columns

brk_cr1 = brk_cr.select_dtypes(include = np.number)

# correlation metrix of Numerical features

brk_cr.corr
```

```
<bound method DataFrame.corr of                                        name
mfr   type   calories   protein   fat  \
0                         100% Bran            Nabisco  Cold        70         4
1
1            100% Natural Bran       Quaker Oats  Cold       120         3
5
2                          All-Bran          Kelloggs  Cold        70         4
1
3    All-Bran with Extra Fiber         Kelloggs  Cold        50         4
0
4              Almond Delight  Ralston Purina  Cold       110         2
2
..                              ...               ...   ...        ...       ...
...
72                         Triples   General Mills  Cold       110         2
1
73                            Trix   General Mills  Cold       110         1
1
74                      Wheat Chex  Ralston Purina  Cold       100         3
1
75                        Wheaties   General Mills  Cold       100         3
1
```

```
76          Wheaties Honey Gold   General Mills  Cold       110      2
1

    sodium  fiber  carbo  sugars  potass  vitamins  shelf  weight
cups  \
0      130   10.0    5.0      6     280       25      3      1.0
0.33
1       15    2.0    8.0      8     135        0      3      1.0
1.00
2      260    9.0    7.0      5     320       25      3      1.0
0.33
3      140   14.0    8.0      0     330       25      3      1.0
0.50
4      200    1.0   14.0      8      -1       25      3      1.0
0.75
..     ...    ...    ...    ...     ...      ...    ...      ...    ..
.
72     250    0.0   21.0      3      60       25      3      1.0
0.75
73     140    0.0   13.0     12      25       25      2      1.0
1.00
74     230    3.0   17.0      3     115       25      1      1.0
0.67
75     200    3.0   17.0      3     110       25      1      1.0
1.00
76     200    1.0   16.0      8      60       25      1      1.0
0.75

        rating   perc_fat_in_calorie sodium_status fiber_source
Potass_status
0    68.402973                  1.43          Low    Excellent
Low
1    33.983679                  4.17          Low         Good
Low
2    59.425505                  1.43         Good    Excellent
Safe
3    93.704912                  0.00          Low    Excellent
Safe
4    34.384843                  1.82         Good         Good
Low
..         ...                   ...          ...          ...
...
72   39.106174                  0.91         Good         Good
Low
73   27.753301                  0.91          Low         Good
Low
74   49.787445                  1.00         Good      Average
Low
75   51.592193                  1.00         Good      Average
```

```
Low
76  36.187559                    0.91          Good          Good
Low

[77 rows x 20 columns]>
```

Visualisation of correlation matrix using Heatmap

```python
plt.figure(figsize=(10,6))
a= sns.heatmap(brk_cr1.corr(), annot = True, fmt = '.1f', linewidth =
1).set_title("Correlation Matrix")

a.figure.savefig ("30.png", bbox_inches = "tight")
```