assignment07

October 29, 2023

[ ]:
```
import numpy as np
import pandas as pd

# Introduction

# In this section, please describe the dataset you are using.
# Include a link to the source of this data.
# You should also provide some explanation on why you choose this dataset.

# The dataset I'm using is from Kaggle. It is about the nutritional values of various cereals.
# I chose this dataset because I have to be conscious of what I eat because of an endocrinology related illness.
# I didn't realize there's so many brands of cereal so it'd be interesting to learn more about them
# Link to dataset: https://www.kaggle.com/datasets/crawford/80-cereals
```

[ ]:
```
# Data Exploration
# Import your dataset into your .ipynb, create dataframes, and explore your data.

#Include:

# * Summary statistics means, medians, quartiles,
# * Missing value information
# * Any other relevant information about the dataset.

# I'm naming my dataframe cereal_df and importing the data using pandas' read_csv() function
# I give the sep parameter in read_csv() a comma because some cereal brands uses ;
# ; can also be used to seperate columns which is what I don't want
cereal_df = pd.read_csv("cereal.csv", sep=",")
```

[ ]:
```
# Pandas' describe function gives a summary means, medians, quartiles, among other stats for numeric columns
cereal_df.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

[ ]:
```
# There are no missing values in all columns
# However from what is shown above with the describe function, there are␣
  ↪negative numbers which is impossible
print(cereal_df.isna().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
```

[ ]:
```
# Data Wrangling

# Create a subset of your original data and perform the following.

# 1. Modify multiple column names.
# I used pandas' rename function and using the columns parameter to rename 2 columns
# The columns parameter is given a dictionary to rename columns to make them more clear
# The cereal dataframe is a subset of cereal_df with renamed columns
cereal = cereal_df.rename(columns={"name": "Brand", "mfr": "Manufacturer",
                                   "carbo": "Carbohydrates", "potass": "Potassium"})
# A preview using the head function on the cereal dataframe shows the newly renamed columns
print(cereal.head())
```

```
                    Brand Manufacturer type  calories  protein  fat  \
0                 100% Bran            N    C        70        4    1
1          100% Natural Bran            Q    C       120        3    5
2                   All-Bran            K    C        70        4    1
3  All-Bran with Extra Fiber            K    C        50        4    0
4             Almond Delight            R    C       110        2    2

   sodium  fiber  Carbohydrates  sugars  Potassium  vitamins  shelf  weight  \
0     130   10.0            5.0       6        280        25      3     1.0
1      15    2.0            8.0       8        135         0      3     1.0
2     260    9.0            7.0       5        320        25      3     1.0
3     140   14.0            8.0       0        330        25      3     1.0
4     200    1.0           14.0       8         -1        25      3     1.0

   cups     rating
0  0.33  68.402973
1  1.00  33.983679
2  0.33  59.425505
3  0.50  93.704912
4  0.75  34.384843
```

[ ]:
```
# 2. Look at the structure of your data - are any variables improperly coded?
# Such as strings or characters? Convert to correct structure if needed.

# There doesn't seem to be any improperly coded variables.
# The numerics are either ints or floats and the strings are object types
```

```
print(cereal.dtypes)

# 3. Fix missing and invalid values in data.

# As shown earlier with the describe function, the carbohydrates, sugars, and␣
 ↪the potassium columns have a minimum value of 1 which is impossible to occur

# Gets the row indices for the values less than 0
carb_less_than_zero = cereal["Carbohydrates"] < 0
sugar_less_than_zero = cereal["sugars"] < 0
potass_less_than_zero = cereal["Potassium"] < 0

# Change those values to NaN because the exact value is unknown
# Use .loc to get the specific rows and the column
cereal.loc[carb_less_than_zero, ["Carbohydrates"]] = np.nan
cereal.loc[sugar_less_than_zero, ["sugars"]]
cereal.loc[potass_less_than_zero, ["Potassium"]] = np.nan

# The manufacturers of the cereals are only given a letter for the manufacturer␣
 ↪column
# I will rename them to use the full name so it's more clear
a = cereal["Manufacturer"] == "A"
g = cereal["Manufacturer"] == "G"
k = cereal["Manufacturer"] == "K"
n = cereal["Manufacturer"] == "N"
p = cereal["Manufacturer"] == "P"
q = cereal["Manufacturer"] == "Q"
r = cereal["Manufacturer"] == "R"

cereal.loc[a, ["Manufacturer"]] = "American Home Food Products"
cereal.loc[g, ["Manufacturer"]] = "General Mills"
cereal.loc[k, ["Manufacturer"]] = "Kellogs"
cereal.loc[n, ["Manufacturer"]] = "Nabisco"
cereal.loc[p, ["Manufacturer"]] = "Post"
cereal.loc[q, ["Manufacturer"]] = "Quaker Oats"
cereal.loc[r, ["Manufacturer"]] = "Ralston Purina"
```

```
Brand             object
Manufacturer      object
type              object
calories           int64
protein            int64
fat                int64
sodium             int64
fiber            float64
Carbohydrates    float64
sugars             int64
```

```
Potassium          int64
vitamins           int64
shelf              int64
weight           float64
cups             float64
rating           float64
dtype: object
```

[ ]:
```
# 4. Create new columns based on existing columns or calculations.

# These newly created columns will have the values true of false depending on the amount of sugar
cereal["No Sugar"] = cereal["sugars"] == 0
cereal["No Fat"] = cereal["fat"] == 0
```

[ ]:
```
# 5. Drop column(s) from your dataset.

# I don't know what the shelf column does other than only be given the numbers 1-3 so I will drop that
cereal = cereal.drop("shelf", axis = 1)

#Shows the shelf column is gone
print(cereal.head())
```

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

   sodium  fiber  Carbohydrates  sugars  Potassium  vitamins  weight  cups  \
0     130   10.0            5.0       6      280.0        25     1.0  0.33
1      15    2.0            8.0       8      135.0         0     1.0  1.00
2     260    9.0            7.0       5      320.0        25     1.0  0.33
3     140   14.0            8.0       0      330.0        25     1.0  0.50
4     200    1.0           14.0       8        NaN        25     1.0  0.75

      rating  No Sugar  No Fat
0  68.402973     False   False
1  33.983679     False   False
2  59.425505     False   False
3  93.704912      True    True
4  34.384843     False   False
```

[ ]:
```
# 6. Drop a row(s) from your dataset.

# Shows the previous numbers of rows and columns
```

```
print(cereal.shape)

# I will remove the rows containing nan like those set to it earlier
cereal.dropna(axis=0, inplace=True)

# Show the change afterwards. 3 rows were removed
print(cereal.shape)
```

```
(77, 17)
(74, 17)
```

[ ]:
```
# 7. Sort your data based on multiple variables.

cereal = cereal.sort_values(by=['rating', 'sugars'], ascending=[False, True])

# A preview showing the dataframe being sorted by highest rating to lowest then␣
 ↪amount of sugar from low to high
print(cereal.head())
```

```
                        Brand Manufacturer type  calories  protein  fat  \
3   All-Bran with Extra Fiber      Kellogs    C        50        4    0
64     Shredded Wheat 'n'Bran      Nabisco    C        90        3    0
65  Shredded Wheat spoon size      Nabisco    C        90        3    0
0                    100% Bran      Nabisco    C        70        4    1
63             Shredded Wheat      Nabisco    C        80        2    0

    sodium  fiber  Carbohydrates  sugars  Potassium  vitamins  weight  cups  \
3      140   14.0            8.0       0      330.0        25    1.00  0.50
64       0    4.0           19.0       0      140.0         0    1.00  0.67
65       0    3.0           20.0       0      120.0         0    1.00  0.67
0      130   10.0            5.0       6      280.0        25    1.00  0.33
63       0    3.0           16.0       0       95.0         0    0.83  1.00

       rating  No Sugar  No Fat
3   93.704912      True    True
64  74.472949      True    True
65  72.801787      True    True
0   68.402973     False   False
63  68.235885      True    True
```

[ ]:
```
# 8. Filter your data based on some condition.

# Filters rows for cereal rating greater than 60
rating_greater_than_60 = cereal['rating'] > 60.0

# Print the filtered rows
print(cereal[rating_greater_than_60])
```

```
                        Brand Manufacturer type  calories  protein  fat  \
```

```
3   All-Bran with Extra Fiber      Kellogs    C        50        4    0
64     Shredded Wheat 'n'Bran      Nabisco    C        90        3    0
65  Shredded Wheat spoon size      Nabisco    C        90        3    0
0                    100% Bran      Nabisco    C        70        4    1
63              Shredded Wheat      Nabisco    C        80        2    0
55                Puffed Wheat  Quaker Oats    C        50        2    0
54                 Puffed Rice  Quaker Oats    C        50        1    0

    sodium  fiber  Carbohydrates  sugars  Potassium  vitamins  weight  cups  \
3      140   14.0            8.0       0      330.0        25    1.00  0.50
64       0    4.0           19.0       0      140.0         0    1.00  0.67
65       0    3.0           20.0       0      120.0         0    1.00  0.67
0      130   10.0            5.0       6      280.0        25    1.00  0.33
63       0    3.0           16.0       0       95.0         0    0.83  1.00
55       0    1.0           10.0       0       50.0         0    0.50  1.00
54       0    0.0           13.0       0       15.0         0    0.50  1.00

       rating  No Sugar  No Fat
3   93.704912      True    True
64  74.472949      True    True
65  72.801787      True    True
0   68.402973     False   False
63  68.235885      True    True
55  63.005645      True    True
54  60.756112      True    True
```

[ ]:
```
# 9. Convert all the string values to upper or lower cases in one column.

# Everything in the Brand column will have all caps
cereal["Brand"] = cereal["Brand"].str.upper()

# The Cereal brands are now all capitalized
print(cereal.head())
```

```
                        Brand Manufacturer type  calories  protein  fat  \
3   ALL-BRAN WITH EXTRA FIBER      Kellogs    C        50        4    0
64     SHREDDED WHEAT 'N'BRAN      Nabisco    C        90        3    0
65  SHREDDED WHEAT SPOON SIZE      Nabisco    C        90        3    0
0                    100% BRAN      Nabisco    C        70        4    1
63              SHREDDED WHEAT      Nabisco    C        80        2    0

    sodium  fiber  Carbohydrates  sugars  Potassium  vitamins  weight  cups  \
3      140   14.0            8.0       0      330.0        25    1.00  0.50
64       0    4.0           19.0       0      140.0         0    1.00  0.67
65       0    3.0           20.0       0      120.0         0    1.00  0.67
0      130   10.0            5.0       6      280.0        25    1.00  0.33
63       0    3.0           16.0       0       95.0         0    0.83  1.00
```

```
       rating  No Sugar  No Fat
3   93.704912      True    True
64  74.472949      True    True
65  72.801787      True    True
0   68.402973     False   False
63  68.235885      True    True
```

[ ]:
```
# 10. Check whether numeric values are present in a given column of your␣
 ↪dataframe.

# The rating column should be numeric, this shows that it is
# A float is numeric
print(cereal["rating"].dtype)
```

```
float64
```

[ ]:
```
# 11. Group your dataset by one column, and get the mean, min, and max values␣
 ↪by group.
#  * Groupby()
#  * agg() or .apply()

# Get only the numeric columns to use the mean, min, and max on
numeric_columns = cereal.select_dtypes(include=[int, float])
# Prints the mean, minimum, & max for each numeric column grouped by␣
 ↪manufacturer
print(numeric_columns.groupby(cereal["Manufacturer"]).agg(['mean', 'min',␣
 ↪'max']))
```

```
                            calories                protein                fat  \
                                mean  min  max        mean min max       mean
Manufacturer
American Home Food Products 100.000000  100  100  4.000000   4   4  1.000000
General Mills               111.363636  100  140  2.318182   1   6  1.363636
Kellogs                     108.695652   50  160  2.652174   1   6  0.608696
Nabisco                      84.000000   70   90  2.800000   2   4  0.200000
Post                        108.888889   90  120  2.444444   1   3  0.888889
Quaker Oats                  94.285714   50  120  2.285714   1   4  1.714286
Ralston Purina              115.714286   90  150  2.571429   1   4  1.142857

                                      sodium  … vitamins    weight        \
                            min max      mean  …      max      mean   min
Manufacturer                                  …
American Home Food Products   1   1  0.000000  …       25  1.000000  1.00
General Mills                 1   3 200.454545  …      100  1.049091  1.00
Kellogs                       0   3 174.782609  …      100  1.077826  1.00
Nabisco                       0   1  29.000000  …       25  0.966000  0.83
Post                          0   3 146.111111  …       25  1.064444  1.00
Quaker Oats                   0   5 105.714286  …       25  0.857143  0.50
```

```
Ralston Purina                0    3  197.857143  …       25  1.000000  1.00

                                          cups                    rating             \
                               max       mean   min   max       mean        min
Manufacturer
American Home Food Products  1.00   1.000000  1.00  1.00  54.850917  54.850917
General Mills                1.50   0.875000  0.50  1.50  34.485852  19.823573
Kellogs                      1.50   0.796087  0.33  1.00  44.038462  29.924285
Nabisco                      1.00   0.734000  0.33  1.00  68.655517  59.363993
Post                         1.33   0.714444  0.25  1.33  41.705744  28.025765
Quaker Oats                  1.00   0.845714  0.50  1.00  41.785647  18.042851
Ralston Purina               1.00   0.888571  0.67  1.13  42.565591  34.139765


                                    max
Manufacturer
American Home Food Products  54.850917
General Mills                51.592193
Kellogs                      93.704912
Nabisco                      74.472949
Post                         53.371007
Quaker Oats                  63.005645
Ralston Purina               49.787445

[7 rows x 36 columns]
```

[ ]:
```python
# 12. Group your dataset by two columns and then sort the aggregated results␣
 ↪within the groups.

# Function to only get numeric columns then calculate the min, max and mean␣
 ↪from those columns
def agg_numeric_columns(group):
    numeric_columns = group.select_dtypes(include=['int', 'float'])
    return numeric_columns.agg(['min', 'max', 'mean'])

# Sort first by manufacturer then the Brands that use them as a manufacturer
res = cereal.groupby(['Manufacturer', 'Brand']).apply(agg_numeric_columns)
print(res)
```

```
                                                      calories  protein  \
Manufacturer                Brand
American Home Food Products MAYPO                 min     100.0      4.0
                                                  max     100.0      4.0
                                                  mean    100.0      4.0
General Mills               APPLE CINNAMON CHEERIOS min     110.0      2.0
                                                  max     110.0      2.0
…                                                         …        …
Ralston Purina              RICE CHEX             max     110.0      1.0
```

```
                                          mean     110.0      1.0
                          WHEAT CHEX      min      100.0      3.0
                                          max      100.0      3.0
                                          mean     100.0      3.0

                                                fat  sodium  fiber  \
Manufacturer                Brand
American Home Food Products MAYPO           min   1.0     0.0    0.0
                                            max   1.0     0.0    0.0
                                            mean  1.0     0.0    0.0
General Mills               APPLE CINNAMON CHEERIOS min   2.0   180.0    1.5
                                            max   2.0   180.0    1.5
…                                              …      …    …
Ralston Purina              RICE CHEX       max   0.0   240.0    0.0
                                            mean  0.0   240.0    0.0
                            WHEAT CHEX      min   1.0   230.0    3.0
                                            max   1.0   230.0    3.0
                                            mean  1.0   230.0    3.0

                                                Carbohydrates  \
Manufacturer                Brand
American Home Food Products MAYPO           min            16.0
                                            max            16.0
                                            mean           16.0
General Mills               APPLE CINNAMON CHEERIOS min            10.5
                                            max            10.5
…                                                             …
Ralston Purina              RICE CHEX       max            23.0
                                            mean           23.0
                            WHEAT CHEX      min            17.0
                                            max            17.0
                                            mean           17.0

                                                sugars  Potassium  \
Manufacturer                Brand
American Home Food Products MAYPO           min      3.0       95.0
                                            max      3.0       95.0
                                            mean     3.0       95.0
General Mills               APPLE CINNAMON CHEERIOS min     10.0       70.0
                                            max     10.0       70.0
…                                                     …         …
Ralston Purina              RICE CHEX       max      2.0       30.0
                                            mean     2.0       30.0
                            WHEAT CHEX      min      3.0      115.0
                                            max      3.0      115.0
                                            mean     3.0      115.0

                                                vitamins  weight  \
```

```
Manufacturer                Brand
American Home Food Products MAYPO                    min       25.0     1.0
                                                     max       25.0     1.0
                                                     mean      25.0     1.0
General Mills               APPLE CINNAMON CHEERIOS  min       25.0     1.0
                                                     max       25.0     1.0
…                                                                …      …
Ralston Purina              RICE CHEX                max       25.0     1.0
                                                     mean      25.0     1.0
                            WHEAT CHEX               min       25.0     1.0
                                                     max       25.0     1.0
                                                     mean      25.0     1.0

                                                           cups     rating
Manufacturer                Brand
American Home Food Products MAYPO                    min   1.00  54.850917
                                                     max   1.00  54.850917
                                                     mean  1.00  54.850917
General Mills               APPLE CINNAMON CHEERIOS  min   0.75  29.509541
                                                     max   0.75  29.509541
…                                                          …        …
Ralston Purina              RICE CHEX                max   1.13  41.998933
                                                     mean  1.13  41.998933
                            WHEAT CHEX               min   0.67  49.787445
                                                     max   0.67  49.787445
                                                     mean  0.67  49.787445

[222 rows x 12 columns]
```

[ ]:
```
# Conclusions

# After exploring your dataset, provide a short summary of what you noticed␣
 ↪from this dataset.
# What would you explore further with more time?

# I noticed Nabisco is the manufacturer who makes the healthiest cereals.
# The top 5 cereals are also the cold type rather than the hot type.
# The healthiest cereal is ALL-BRAN WITH EXTRA FIBER by Kellog.
# It's not surprising that that the top cereals have no sugar and fats, however␣
 ↪the fourth ranked cereal does have them.
# What I would explore further with more time is looking up more cereals to add␣
 ↪to this data.
```