Lesson 8

Data Exploration and Analysis II

EM384

Lesson 8

L8 **LESSON OBJECTIVE #1**
Understand the Python pandas DataFrame data structure.

L8 **LESSON OBJECTIVE #2**
Generate a DataFrame from a CSV file.

L8 **LESSON OBJECTIVE #3**
Generate summary statistics for data in Python

L8 **LESSON OBJECTIVE #4**
Filter a pandas DataFrame using [ ] and conditionals.

What is pandas?

“pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.”

https://pandas.pydata.org/docs/getting_started/overview.html

```python
#MR COOKIE'S COOKIES - Revisited

#Parameters
cookies_sold = 500
selling_price = 1.25
cookies_bought = 600
packaging_cost = 0.05
ingredient_cost = 0.30
fixed_cost = 250

#Model
revenue = cookies_sold * selling_price
cost = cookies_bought * (packaging_cost + ingredient_cost) + fixed_cost

profit = revenue - cost
print('The profit for Mr cookie\'s cookies is', profit,'dollars')
#note the \ tells python that the ' is in the string and not the end of the string
```

Add calculated columns to a DataFrame

- Let's look at the % of calories that come from fat, protein, and carbos in each cereal by adding three calculated columns to the df_cereal DataFrame.

- *Is this easier or harder than Excel?*

```
df_cereal["fat_calorie_prcnt"] = df_cereal.fat / (df_cereal.fat + df_cereal.protein + df_cereal.carbo)

df_cereal["protein_calorie_prcnt"] = df_cereal.protein / (df_cereal.fat + df_cereal.protein + df_cereal.carbo)

df_cereal["carbo_calorie_prcnt"] = df_cereal.carbo / (df_cereal.fat + df_cereal.protein + df_cereal.carbo)


df_cereal["total_check"] = df_cereal.fat_calorie_prcnt + df_cereal.protein_calorie_prcnt + df_cereal.carbo_calorie_prcnt
```

```
#Use groupby to create a new DataFrame showing the average of all metrics by manufacturer.
df_cereal_avgs_by_company = df_cereal.groupby("mfr").mean(numeric_only=True)
```

>>> Remember when we made this pivot table in the tutorial to calculate the arithmetic means for all of the numeric metrics, by Manufacturer?

>>> Re-run the same exact command to create the pivot table DataFrame. Is there anything different about the results?

Sort a DataFrame by one or more columns

- Using "% of calories from fat" as our proxy for how healthy a cereal is, **what are the ten least healthy cereals in the dataset?**

- Is there an easy way to get the answer with one line of code using pandas **.sort_values( )** and **.iloc[ ]** ? *Hint, the answer is yes.*

```
# sort the DataFrame based on fat_calorie_prcnt
# and create a new DataFrame with the 10 most unhealthy cereals.
df_unhealthy_cereals = df_cereal.sort_values(["fat_calorie_prcnt"], ascending = [False]).iloc[:10]

# sort the DataFrame based on manufacturer (A-Z) and rating (high-to-low)
# method 1- specify column names
df_rating_by_mfr = df_cereal[["name","mfr","type","shelf","rating"]].sort_values(["mfr","rating"], ascending = [True, False])
# method 2- use column index numbers (recall df_cereal.info())
df_rating_by_mfr = df_cereal.iloc[:,[0, 1, 2, 15, 16]].sort_values(["mfr","rating"], ascending = [True, False])
```

```
#create a new DataFrame that only includes the following subset of columns
#name, manufacturer, type, shelf, and rating
df_cereal_retail = df_cereal[["name","mfr","type","shelf","rating"]]
```

df_cereal_retail - DataFrame

Index	name	mfr	type	shelf	rating
0	100% Bran	N	C	3	68.403
1	100% Natural Bran	Q	C	3	33.9837
2	All-Bran	K	C	3	59.4255
3	All-Bran with Extra Fiber	K	C	3	93.7049
4	Almond Delight	R	C	3	34.3848
5	Apple Cinnamon Cheerios	G	C	1	29.5095
6	Apple Jacks	K	C	2	33.1741
7	Basic 4	G	C	3	37.0386
8	Bran Chex	R	C	1	49.1203
9	Bran Flakes	P	C	3	53.3138
10	Cap'n'Crunch	Q	C	2	18.0429
11	Cheerios	G	C	1	50.765

Format Resize Background color Column min/max Save and Close Close

LESSON OBJECTIVE #4

L8 Filter a pandas DataFrame using [ ] and conditionals.

```
#Create a DataFrame of all cereals with a rating which is at least
#one standard deviation higher than the mean.
series_highly_rated = df_cereal.loc[df_cereal["rating"] >= mean_rating + stdv_rating, "name"]
```

series_highly_rated - Series

Index	name
0	100% Bran
2	All-Bran
3	All-Bran with Extra Fiber
20	Cream of Wheat (Quick)
26	Frosted Mini-Wheats
50	Nutri-grain Wheat
54	Puffed Rice
55	Puffed Wheat
63	Shredded Wheat
64	Shredded Wheat 'n'Bran
65	Shredded Wheat spoon size
68	Strawberry Fruit Wheats

Format Resize Background color Column min/max Save and Close Close

Create a scatter plot from a DataFrame

- The marketing people think there might be a positive relationship between people's preferences (ratings of cereals) and the amount of sugar content. **What are you seeing in the data?**

- Let's use the simple pandas plot functionality to create a scatter plot.

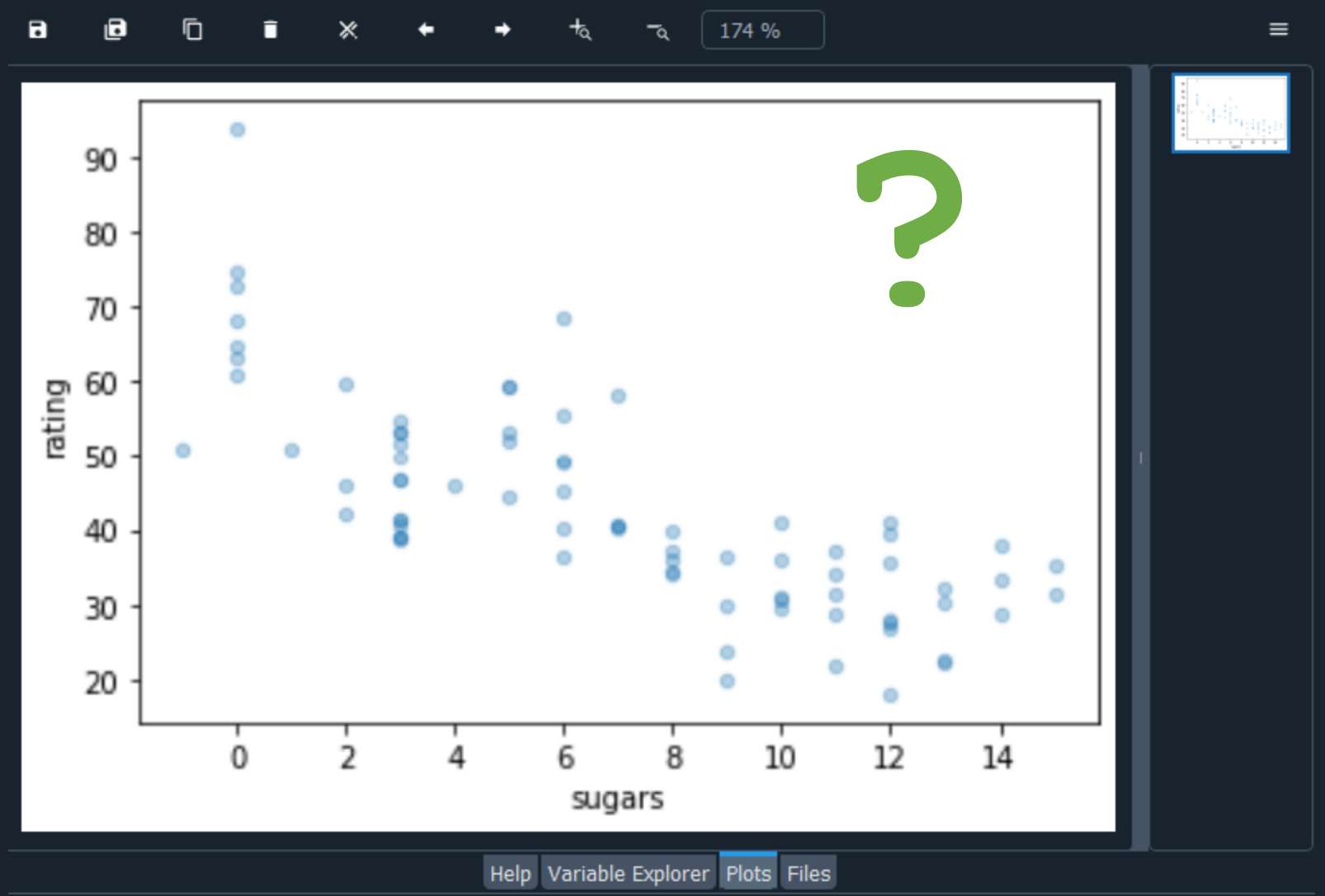
```
import pandas as pd
import matplotlib.pyplot as plt

df_cereal = pd.read_csv("cereal.csv")

df_cereal.plot.scatter(x="sugars", y="rating", alpha= 0.33)
```

LESSON OBJECTIVE #3

Generate summary statistics for data in Python



Read these tutorials at *pandas.pydata.org*

- What kind of data does pandas handle?
- How do I read and write tabular data?
- How do I select a subset of a DataFrame?
- How to calculate summary statistics?
- How to create new columns derived from existing columns?
- How do I create plots in pandas?

More learning

- For anything you need to know about what pandas can do and how to use it, pandas.pydata.org maintains on online user guide:

https://pandas.pydata.org/docs/user_guide/index.html#user-guide

- *As far as documentation goes, this user guide (in addition to the other tutorials and technical content available on the website) set a very high standard for accessibility, completeness, and ease of use.*

- *This makes working with data in Python easy and enjoyable.*

Next Time…

- Homework Set 2 Due
- Homework Set 3 Assigned
- **Read Chapter 3.1 (pages 32-35, stop at Graphical Solution)**

Lesson 9: Introduction to Linear Programs!

- Understand the characteristics of linear programming as a subset of optimization.
- Formulate a linear program algebraically.
- Identify the three parts to a linear program: Objective Function, Decision Variables, and Constraints.

Slides from pandas tutorial video

EM384 pandas tutorial video.mp4

What is pandas?

“pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.”

https://pandas.pydata.org/docs/getting_started/overview.html

1. Import data from Desktop.

- Download data from Teams in Lesson 8 Folders.
- Confirm your working directory.
- Import the pandas package.

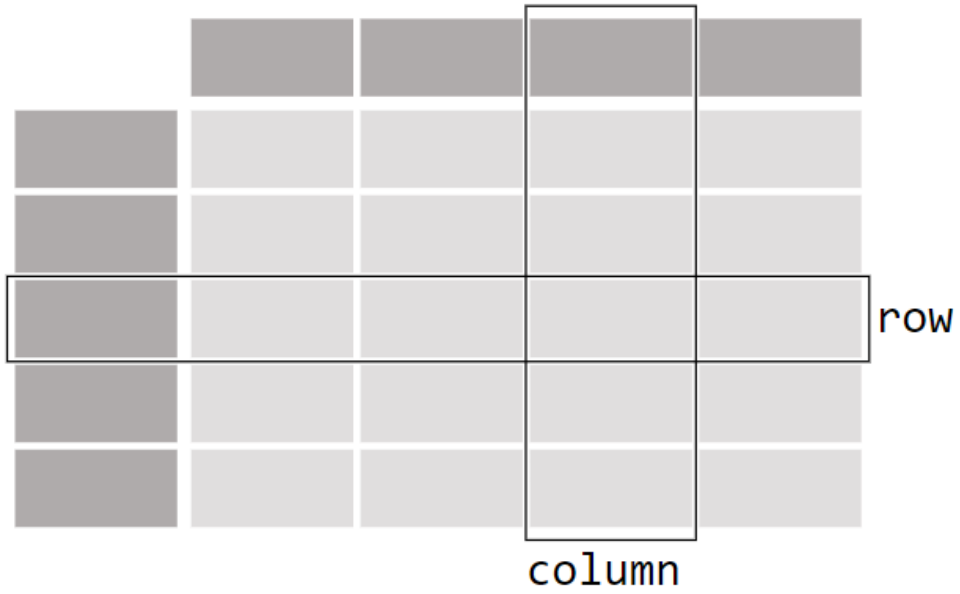
link to cereal.csv

```
import pandas as pd
```

2. pandas DataFrame data structure

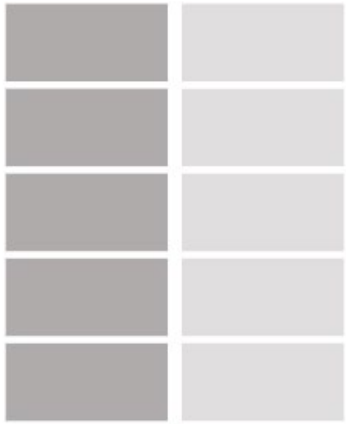
pandas data table representation

DataFrame



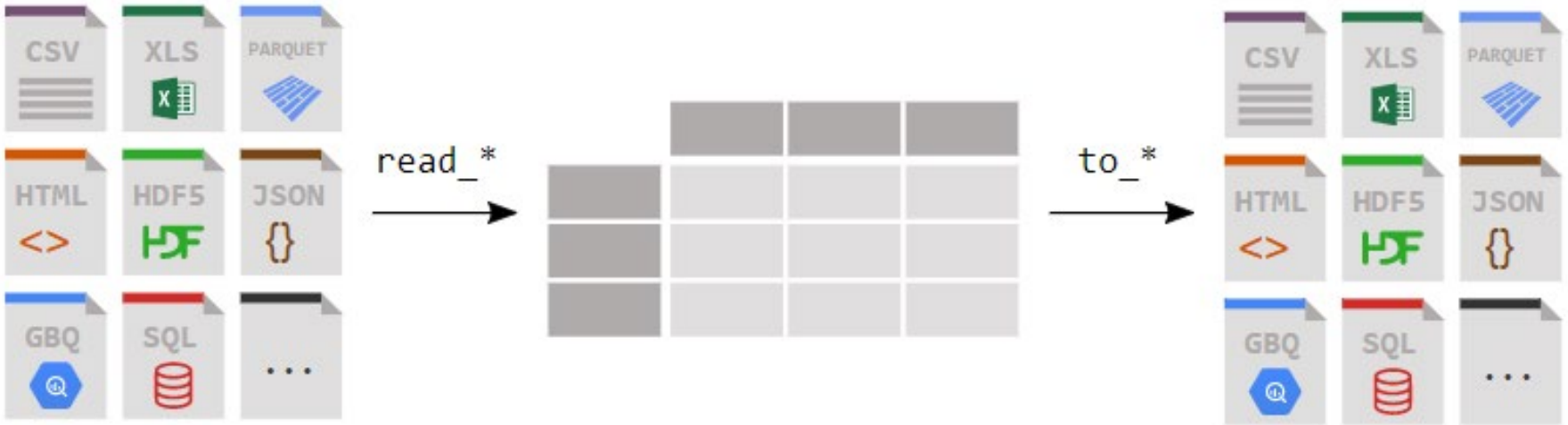
Each column in a `DataFrame` is a `Series`

Series



https://pandas.pydata.org/docs/getting_started/intro_tutorials/01_table_oriented.html#pandas-data-table-representation

3. Create a DataFrame from a csv file



https://pandas.pydata.org/docs/getting_started/intro_tutorials/02_read_write.html

```
#import data from local machine into a DataFrame.
df_cereal = pd.read_csv("cereal.csv")
```

df_cereal - DataFrame

Index	name	mfr	type	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
20	Cream of Wheat (Quick)	N	H	100	3	0	80	1	21	0	-1	0	2	1	1	64.5338
21	Crispix	K	C	110	2	0	220	1	21	3	30	25	3	1	1	46.8956
22	Crispy Wheat & Raisins	G	C	100	2	1	140	2	11	10	120	25	3	1	0.75	36.1762
23	Double Chex	R	C	100	2	0	190	1	18	5	80	25	3	1	0.75	44.3309
24	Froot Loops	K	C	110	2	1	125	1	11	13	30	25	2	1	1	32.2076
25	Frosted Flakes	K	C	110	1	0	200	1	14	11	25	25	1	1	0.75	31.436
26	Frosted Mini-Wheats	K	C	100	3	0	0	3	14	7	100	25	2	1	0.8	58.3451
27	Fruit & Fibre Dates; Walnuts; and Oats	P	C	120	3	2	160	5	12	10	200	25	3	1.25	0.67	40.917
28	Fruitful Bran	K	C	120	3	0	240	5	14	12	190	25	3	1.33	0.67	41.0155
29	Fruity Pebbles	P	C	110	1	1	135	0	13	12	25	25	2	1	0.75	28.0258
30	Golden Crisp	P	C	100	2	0	45	0	11	15	40	25	1	1	0.88	35.2524
31	Golden Grahams	G	C	110	1	1	280	0	15	9	45	25	2	1	0.75	23.804

Format Resize ☑ Background color ☑ Column min/max Save and Close Close

4. We have a DataFrame.

- Congratulations, you've just created a fresh new pandas DataFrame!
- Now you can get to work with your data in Python using pandas.
- By learning this package and practicing with it, you'll be able to do nearly anything you need to do with data of all shapes and sizes.

5. What data are we working with?

- Display the first ten records in the console.
- Display the columns and their data types.
- View high-level information about the DataFrame’s contents.

```
In [4]: df_cereal.head(10)
Out[4]:
                        name mfr type  calories  ...  shelf  weight  cups     rating
0                  100% Bran   N    C        70  ...      3    1.00  0.33  68.402973
1          100% Natural Bran   Q    C       120  ...      3    1.00  1.00  33.983679
2                   All-Bran   K    C        70  ...      3    1.00  0.33  59.425505
3  All-Bran with Extra Fiber   K    C        50  ...      3    1.00  0.50  93.704912
4             Almond Delight   R    C       110  ...      3    1.00  0.75  34.384843
5    Apple Cinnamon Cheerios   G    C       110  ...      1    1.00  0.75  29.509541
6                Apple Jacks   K    C       110  ...      2    1.00  1.00  33.174094
7                    Basic 4   G    C       130  ...      3    1.33  0.75  37.038562
8                  Bran Chex   R    C        90  ...      1    1.00  0.67  49.120253
9                  Bran Flakes   P    C        90  ...      3    1.00  0.67  53.313813

[10 rows x 16 columns]
```

```
In [5]: df_cereal.dtypes
Out[5]:
name          object
mfr           object
type          object
calories       int64
protein        int64
fat            int64
sodium         int64
fiber        float64
carbo        float64
sugars         int64
potass         int64
vitamins       int64
shelf          int64
weight       float64
cups         float64
rating       float64
dtype: object
```

```
In [6]: df_cereal.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
```

6. Let's generate some summary statistics

- We can quickly generate summary statistics about each of the cereals represented in the DataFrame using the .describe() function.
  - If you are unsure about the quality of your data, summary statistics are a good place to start!

- We can also aggregate the data and calculate summary statistics for all the cereals produced by the same manufacturer.

```
#use the describe function to provide statistics on all the nutritional content measures.
#save the results into a new DataFrame
df_cereal_summary = df_cereal[["calories", "protein", "fat","sodium","fiber",
"carbo","sugars","potass","vitamins"]].describe()
```

df_cereal_summary - DataFrame

Index	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins
count	77	77	77	77	77	77	77	77	77
mean	106.883	2.54545	1.01299	159.675	2.15195	14.5974	6.92208	96.0779	28.2468
std	19.4841	1.09479	1.00647	83.8323	2.38336	4.27896	4.44489	71.2868	22.3425
min	50	1	0	0	0	-1	-1	-1	0
25%	100	2	0	130	1	12	3	40	25
50%	110	3	1	180	2	14	7	90	25
75%	110	3	2	210	3	17	11	120	25
max	160	6	5	320	14	23	15	330	100

Format | Resize | ☑ Background color | ☑ Column min/max | Save and Close | Close

```
#Use groupby to create a new DataFrame showing the average of all metrics by manufacturer.
df_cereal_avgs_by_company = df_cereal.groupby("mfr").mean(numeric_only=True)
```

df_cereal_avgs_by_company - DataFrame

mfr	calories	protein	fat	sodium	fiber	carbo	sugars	potass	vitamins	shelf	weight	cups	rating
A	100	4	1	0	0	16	3	95	25	2	1	1	54.8509
G	111.364	2.31818	1.36364	200.455	1.27273	14.7273	7.95455	85.2273	35.2273	2.13636	1.04909	0.875	34.4859
K	108.696	2.65217	0.608696	174.783	2.73913	15.1304	7.56522	103.043	34.7826	2.34783	1.07783	0.796087	44.0385
N	86.6667	2.83333	0.166667	37.5	4	16	1.83333	120.667	8.33333	1.66667	0.971667	0.778333	67.9686
P	108.889	2.44444	0.888889	146.111	2.77778	13.2222	8.77778	113.889	25	2.44444	1.06444	0.714444	41.7057
Q	95	2.625	1.75	92.5	1.3375	10	5.25	74.375	12.5	2.375	0.875	0.82375	42.916
R	115	2.5	1.25	198.125	1.875	17.625	6.125	89.25	25	2	1	0.87125	41.543

Format Resize ☑ Background color ☑ Column min/max Save and Close Close

```
#Define variables for the mean and standard deviation of the cereal ratings
mean_rating = df_cereal["rating"].mean()
stdv_rating = df_cereal["rating"].std()
```

7. Filter a pandas DataFrame

- Filter the cereal DataFrame to only include the names of the cereals produced by the manufacturer of your favorite cereal.

- Create a series that only includes the names of cereals which have a rating that is at least one standard deviation above the mean.

https://www.innit.com/public/products/images/00835882005016-0Q1t9SyRz3tUJw-0_s500.jpg

```
#Filter the cereal DataFrame to show the names of all the cereals that are
#produced by the manufacturer of your favorite cereal, which is specified as a variable.
my_fav_brand = 'A'

series_my_fav_cereal    =    df_cereal["mfr"] == my_fav_brand

df_my_fav_cereal = df_cereal[df_cereal["mfr"] == my_fav_brand]
```

>>> What is the difference between the outputs?

LESSON OBJECTIVE #4

L8 Filter a pandas DataFrame using [ ] and conditionals.

```
#Create a DataFrame of all cereals with a rating which is at least
#one standard deviation higher than the mean.
series_highly_rated = df_cereal.loc[df_cereal["rating"] >= mean_rating + stdv_rating, "name"]
```

series_highly_rated - Series

Index	name
0	100% Bran
2	All-Bran
3	All-Bran with Extra Fiber
20	Cream of Wheat (Quick)
26	Frosted Mini-Wheats
50	Nutri-grain Wheat
54	Puffed Rice
55	Puffed Wheat
63	Shredded Wheat
64	Shredded Wheat 'n'Bran
65	Shredded Wheat spoon size
68	Strawberry Fruit Wheats

Format Resize Background color Column min/max Save and Close Close

```
#create a new DataFrame that only includes the following subset of columns
#name, manufacturer, type, shelf, and rating
df_cereal_retail = df_cereal[["name","mfr","type","shelf","rating"]]
```

df_cereal_retail - DataFrame

Index	name	mfr	type	shelf	rating
0	100% Bran	N	C	3	68.403
1	100% Natural Bran	Q	C	3	33.9837
2	All-Bran	K	C	3	59.4255
3	All-Bran with Extra Fiber	K	C	3	93.7049
4	Almond Delight	R	C	3	34.3848
5	Apple Cinnamon Cheerios	G	C	1	29.5095
6	Apple Jacks	K	C	2	33.1741
7	Basic 4	G	C	3	37.0386
8	Bran Chex	R	C	1	49.1203
9	Bran Flakes	P	C	3	53.3138
10	Cap'n'Crunch	Q	C	2	18.0429
11	Cheerios	G	C	1	50.765

Format Resize Background color Column min/max Save and Close Close

```
df_cereal_retail.to_csv("cereal_retail_data.csv")
```

	A	B	C	D	E	F
1		name	mfr	type	shelf	rating
2	0	100% Bran	N	C	3	68.40297
3	1	100% Natural Bran	Q	C	3	33.98368
4	2	All-Bran	K	C	3	59.42551
5	3	All-Bran with Extra Fiber	K	C	3	93.70491
6	4	Almond Delight	R	C	3	34.38484
7	5	Apple Cinnamon Cheerios	G	C	1	29.50954
8	6	Apple Jacks	K	C	2	33.17409
9	7	Basic 4	G	C	3	37.03856
10	8	Bran Chex	R	C	1	49.12025
11	9	Bran Flakes	P	C	3	53.31381
12	10	Cap'n'Crunch	Q	C	2	18.04285
13	11	Cheerios	G	C	1	50.765
14	12	Cinnamon Toast Crunch	G	C	2	19.82357
15	13	Clusters	G	C	3	40.40021
16	14	Cocoa Puffs	G	C	2	22.73645
17	15	Corn Chex	R	C	1	41.44502
18	16	Corn Flakes	K	C	1	45.86332
19	17	Corn Pops	K	C	2	35.78279
20	18	Count Chocula	G	C	2	22.39651
21	19	Cracklin' Oat Bran	K	C	3	40.44877
22	20	Cream of Wheat (Quick)	N	H	2	64.53382
23	21	Crispix	K	C	3	46.89564
24	22	Crispy Wheat & Raisins	G	C	3	36.1762
25	23	Double Chex	R	C	3	44.33086
26	24	Froot Loops	K	C	2	32.20758
27	25	Frosted Flakes	K	C	1	31.43597
28	26	Frosted Mini-Wheats	K	C	2	58.34514
29	27	Fruit & Fibre Dates; Walnuts; and Oat	P	C	3	40.91705

cereal_retail_data

8. Save your script.

- As “EM384_pandas_cereal_tutorial.py”
- We’ll come back to it later.