001

MODULE 4.2: PANDAS

DATA BOOTCAMP
UNIVERSITY OF KANSAS

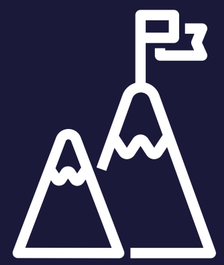
UNIVERISTY OF KANSAS

THIS WEEK: PYTHON

By the end of this week, you'll know how to:

- Read an external CSV file into a DataFrame.
- Determine data types of row values in a DataFrame.
- Format and retrieve data from columns of a DataFrame.
- Merge, filter, slice, and sort a DataFrame..
- Apply the groupby() function to a DataFrame..
- Use multiple methods to perform a function on a DataFrame.
- Perform mathematical calculations on columns of a DataFrame or Series.

THIS WEEK'S CHALLENGE



PyCity Schools Challenge

Use Python and the Pandas library to analyze school district data and showcase trends in school performance.

Using the skills learned throughout the week, help a mock school board with their investigation by adjusting specific data.

- **Deliverable 1**: Replace ninth-grade reading and math scores
- **Deliverable 2:** Repeat the school district analysis
- **Deliverable 3:** A written analysis (README.md)

MODULE 4.2: TODAY'S AGENDA

DATA BOOTCAMP
UNIVERSITY OF KANSAS

TODAY'S AGENDA

By completing today's activities, you'll learn the following skills:

Data Cleaning
- *Group Activity: Portland Crime*

Data Selection & Filtering
- *Activity: Good Movies*

Grouping Data
- *Pair Activity: Training Groupby*

Binning Data
- *Activity: Binning TED*

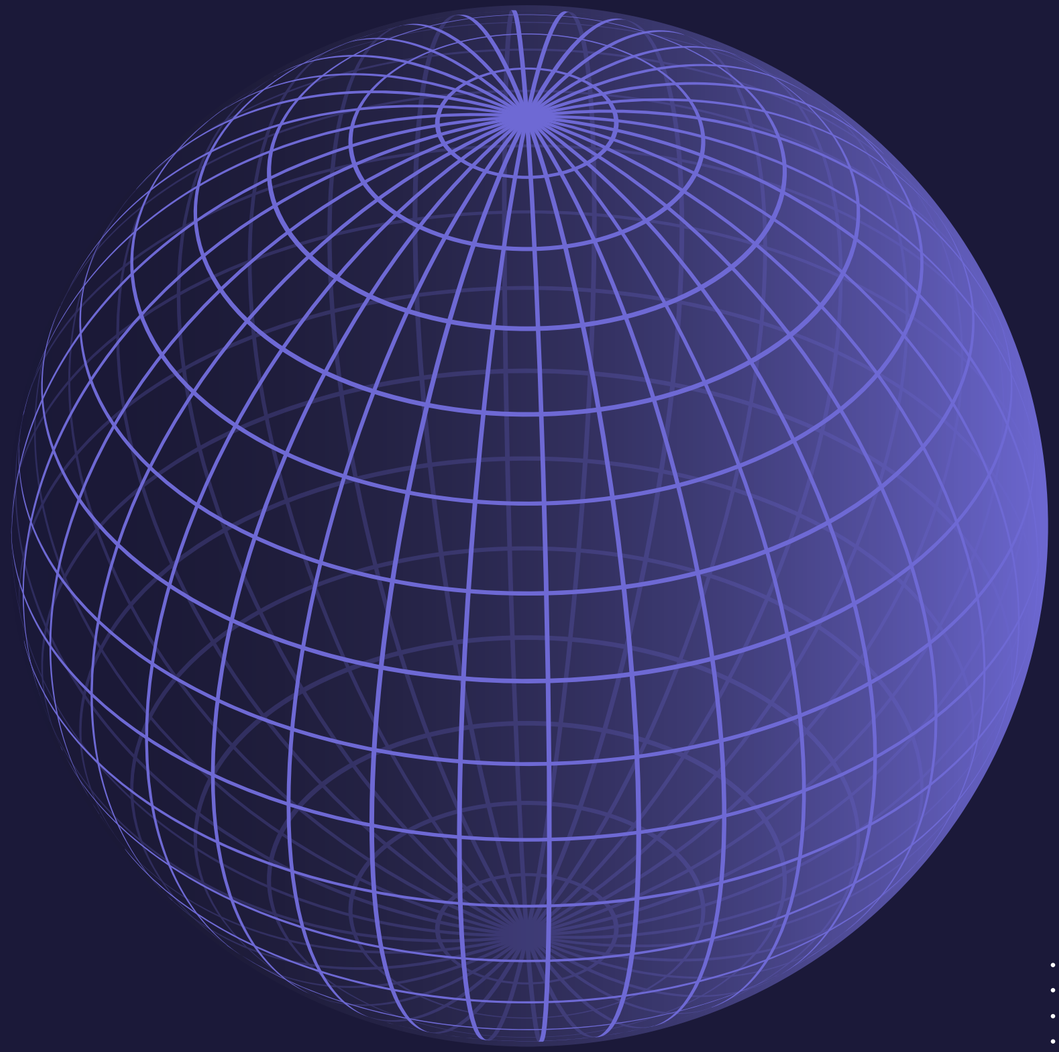
Make sure you've downloaded any relevant class files!

CHECK IN

UNIVERSITY OF KANSAS

007

PANDAS: DATA CLEANING



UNIVERISTY OF KANSAS

WHEN DEALING WITH MASSIVE DATASETS, IT IS ALMOST INEVITABLE THAT YOU HAVE

- INVALID COLUMNS
- DUPLICATE ROWS
- INCONSISTENT SPELLING
- MISSING VALUES

DATA CLEANING WITH PANDAS

`count()`	To look for missing values, we use the `count()` method on the DataFrame.
`dropna(how="any")`	To drop rows with null values, we use `dropna(how="any")`, then verify the counts.
`value_counts()`	To look for any misspelled offenses and to find if similar offenses can be combined, we use `value_counts()` on the `Offense Type` column.
`replace()`	We combine similar offenses using the `replace()` method on the column in question and pass a dictionary into it, with the keys being those values to replace and the value being a common offense in the column.

CLEANING DATA: DELETE COLUMN

To delete invalid or unnecessary, or junk columns

del <DataFrame>[<columns>]

```
In [4]: # Preview of the DataFrame
        # Note that FIELD8 is likely a meaningless column
        df.head()
```

Out[4]:

	LastName	FirstName	Employer	City	State	Zip	Amount	FIELD8
0	Aaron	Eugene	State Department	Dulles	VA	20189	500.0	NaN
1	Abadi	Barbara	Abadi & Co.	New York	NY	10021	200.0	NaN
2	Adamany	Anthony	Retired	Rockford	IL	61103	500.0	NaN
3	Adams	Lorraine	Self	New York	NY	10026	200.0	NaN
4	Adams	Marion	None	Exeter	NH	03833	100.0	NaN

```
In [5]: # Delete extraneous column
        del df['FIELD8']
        df.head()
```

Out[5]:

	LastName	FirstName	Employer	City	State	Zip	Amount
0	Aaron	Eugene	State Department	Dulles	VA	20189	500.0
1	Abadi	Barbara	Abadi & Co.	New York	NY	10021	200.0
2	Adamany	Anthony	Retired	Rockford	IL	61103	500.0
3	Adams	Lorraine	Self	New York	NY	10026	200.0
4	Adams	Marion	None	Exeter	NH	03833	100.0

CLEANING DATA: DROP ROWS WITH MISSING DATA

count()

<DataFrame>.dropna(how='any')

```
In [6]: # Identify incomplete rows
        df.count()

Out[6]: LastName      1776
        FirstName     1776
        Employer      1743
        City          1776
        State         1776
        Zip           1776
        Amount        1776
        dtype: int64

In [7]: # Drop all rows with missing information
        df = df.dropna(how='any')

In [8]: # Verify dropped rows
        df.count()

Out[8]: LastName      1743
        FirstName     1743
        Employer      1743
        City          1743
        State         1743
        Zip           1743
        Amount        1743
        dtype: int64
```

CLEANING DATA: REPLACE VALUES WITH STANDARD COLUMN NAME

value_counts() and replace()

```
In [12]: # Display an overview of the Employers column
         df['Employer'].value_counts()

Out[12]: None                249
         Self                241
         Retired             126
         Self Employed        39
         Self-Employed        34

In [13]: # Clean up Employer category. Replace 'Self Employed' and 'Self' with 'Self-Employed'
         df['Employer'] = df['Employer'].replace(
             {'Self Employed': 'Self-Employed', 'Self': 'Self-Employed'})

In [14]: # Verify clean-up.
         df['Employer'].value_counts()

Out[14]: Self-Employed       314
         None                249
         Retired             126
         Google                6
```

GROUP ACTIVITY:
PORTLAND CRIME

In this activity, we will take a crime dataset from Portland and do our best to clean it up so the DataFrame is consistent and has no rows with missing data.

Suggested Time:
15 minutes

UNIVERSITY OF KANSAS

INSTRUCTIONS: PORTLAND CRIME

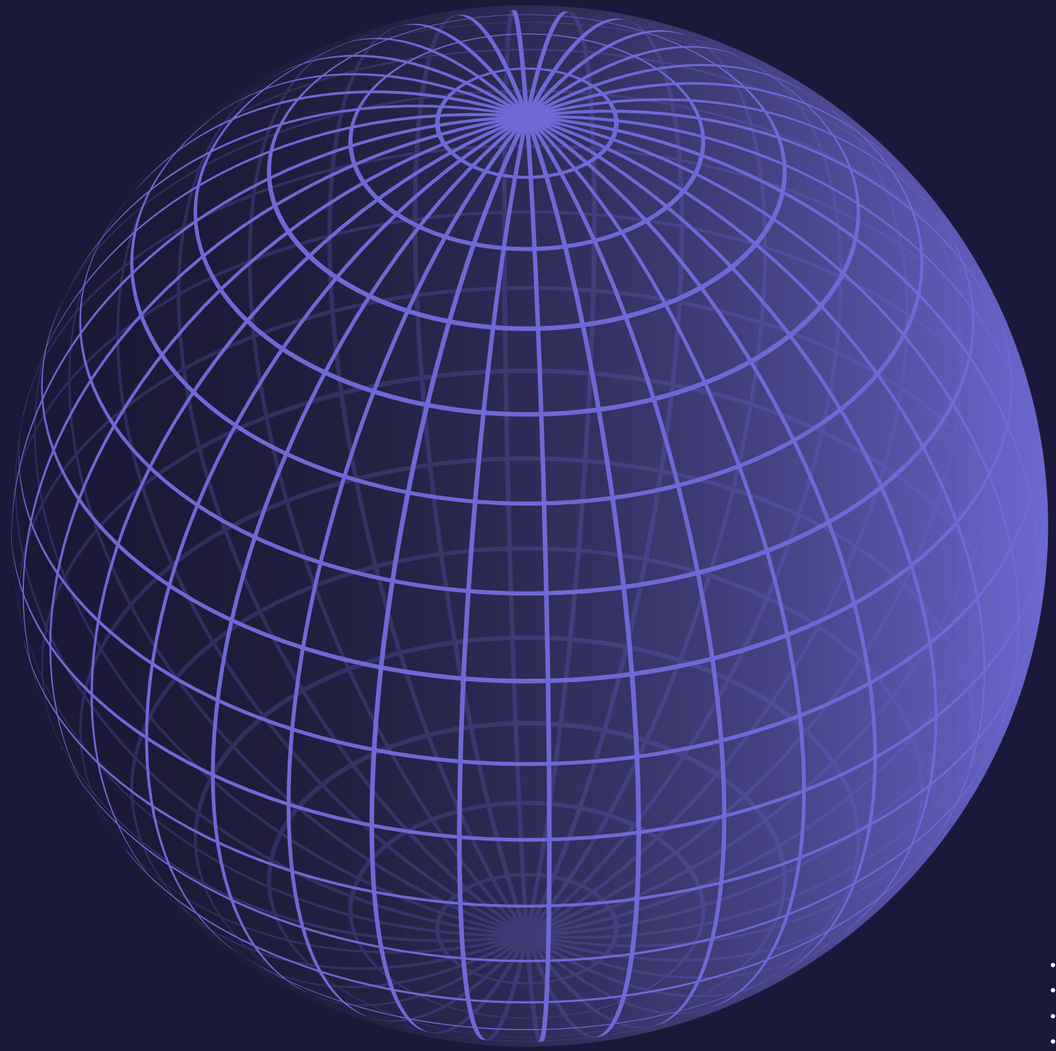
- Read in the csv using Pandas and print out the DataFrame that is returned.
- Get a count of rows within the DataFrame in order to determine if there are any null values.
- Drop the rows which contain null values.
- Search through the "Offense Type" column and "replace" any similar values with one consistent value.

CHECK IN

UNIVERSITY OF KANSAS

016

PANDAS: SELECTING & FILTERING DATA

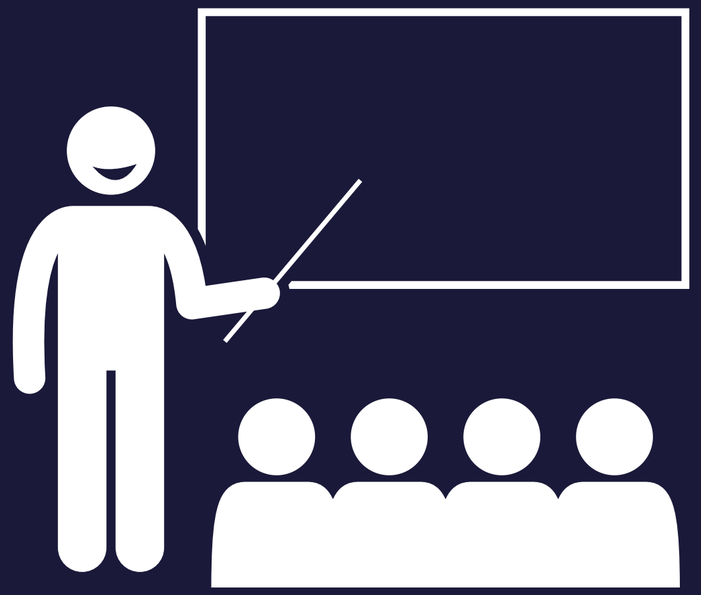


UNIVERISTY OF KANSAS

PANDAS: SELECTING DATA

Programmers can easily collect specific rows or columns of data from a DataFrame using...

- **loc() method**
- **iloc() methods**



INSTRUCTOR DEMONSTRATION

loc() and iloc() in Python Pandas

UNIVERSITY OF KANSAS

ACTIVITY:
GOOD MOVIES

In this activity, you will create an application that looks through IMDB data in order to find only the best movies out there.

Suggested Time:
15 minutes

UNIVERSITY OF KANSAS

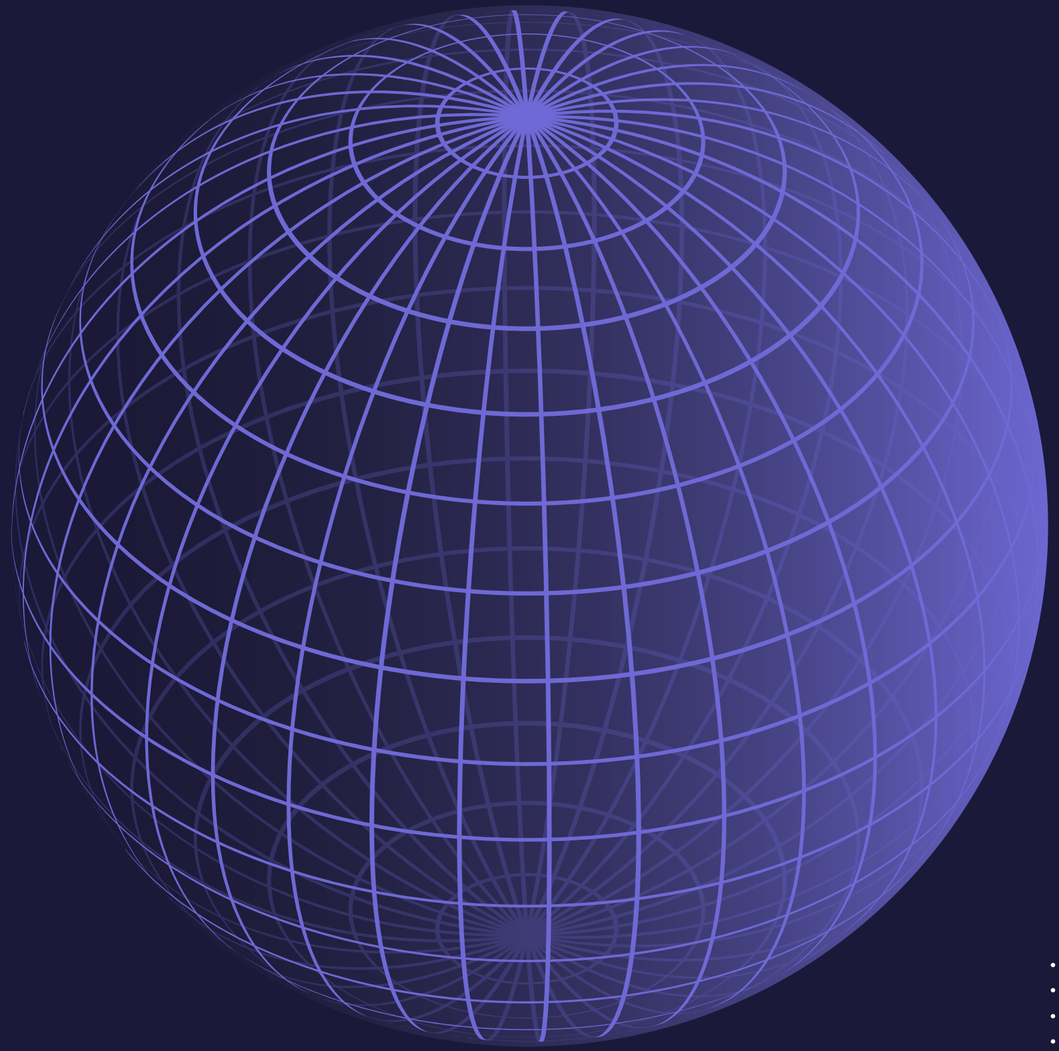
INSTRUCTIONS: GOOD MOVIES

- Use Pandas to load and display the CSV provided in Resources.
- List all the columns in the data set.
- We're only interested in IMDb data, so create a new table that takes the Film and all the columns relating to IMDB.
- Filter out only the good movies—i.e., any film with an IMDb score greater than or equal to 7 and remove the norm ratings.
- Find less popular movies that you may not have heard about - i.e., anything with under 20K votes

CHECK IN

UNIVERSITY OF KANSAS

PANDAS: GROUPING DATA

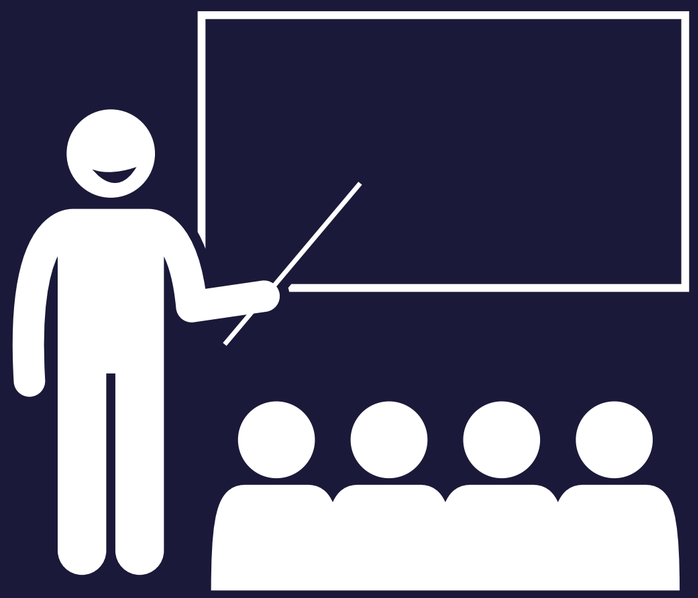


022

UNIVERISTY OF KANSAS

PANDAS: GROUPING DATA WITH .GROUPBY()

groupby() allows you to group Pandas objects based on a common record.



INSTRUCTOR DEMONSTRATION

Groupby

UNIVERSITY OF KANSAS

PAIR ACTIVITY:
TRAINING GROUPBY

In this exercise, you will work in pairs and use groupby() to get the average weight and length membership of the gym members for each trainer.

Suggested Time:
15 minutes

UNIVERSITY OF KANSAS

PAIR PROGRAMMING

There are 2 main roles in pair programming:

Driver

Role:
Focus on resolving the current task while talking through your thought process out loud.

Navigator

Role:
Equally as important. Help catch bugs and typos, think about issues to address for efficiency, and use documentation to find resources to help driver get over hurdles.

INSTRUCTIONS: TRAINING GROUPBY

Using the DataFrame provided, do the following:

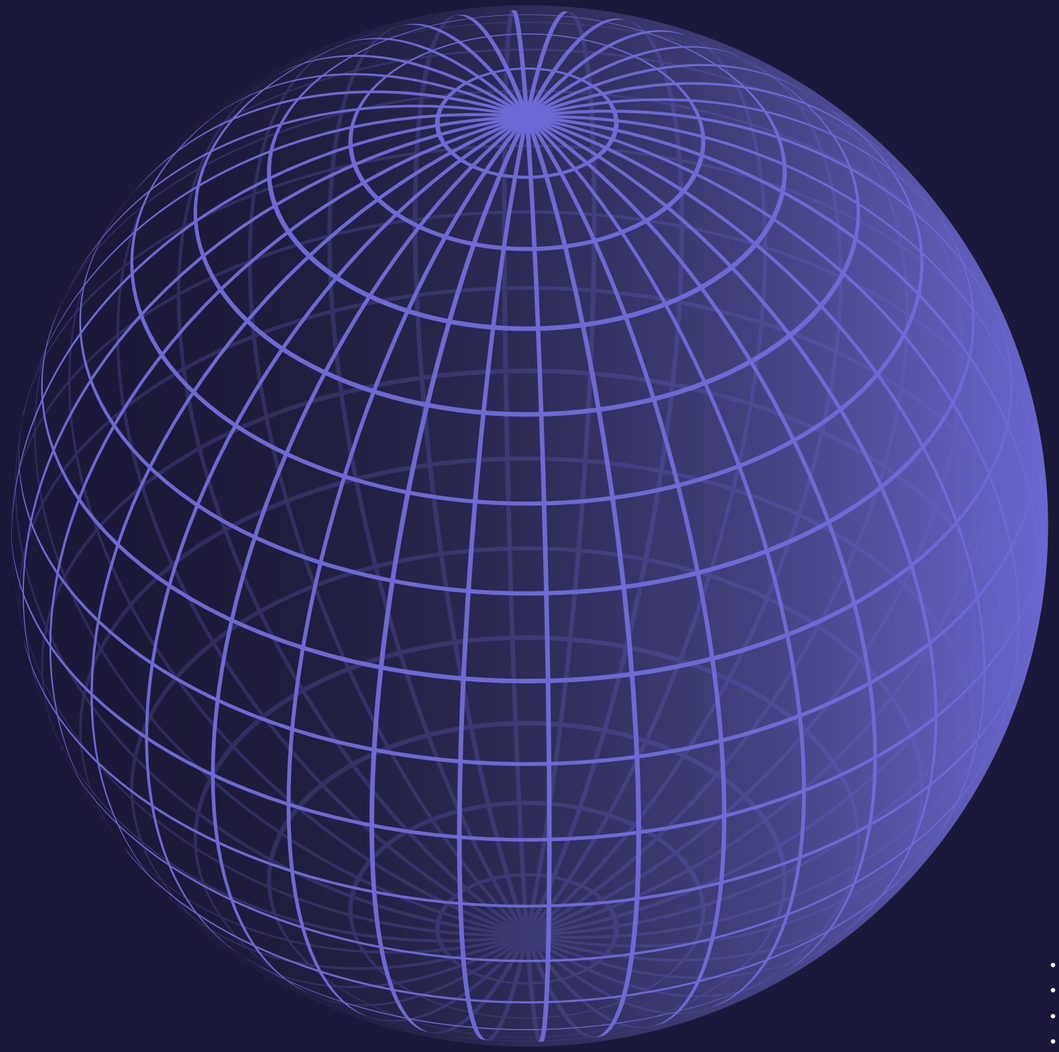
- **Convert the "Membership (Days)" column into weeks and then add this new series into the DataFrame**
- **Create a Dataframe that has only the "Trainer", "Weight", and membership in days and weeks.**
- **Using groupby get the average weight and length membership of the gym members for each trainer.**

CHECK IN

UNIVERSITY OF KANSAS

029

PANDAS: BINNING DATA



UNIVERISTY OF KANSAS

PANDAS: BINNING METHOD

- Function that places values into groups to allow a more vigorous customization of datasets.

- Convert numerical values into categorical values

Examples

- Grouping data on your users' ages into larger bins such as: 0-13 years old, 14-21 years old, 22-40 years old, and, 41+.

- You might create a new Price Range attribute such that:
  - If Price is less than 20 dollars, then Price Range is "Inexpensive".
  - If Price is greater than 20 dollars but less than 40 dollars, then Price Range is "Moderately Priced".
  - If Price is greater than 40 dollars but less than 60 dollars, then Price Range is "Expensive".
  - If Price is greater than 60 dollars, then Price Range is "Very Expensive".

BINNING DATA

Understand

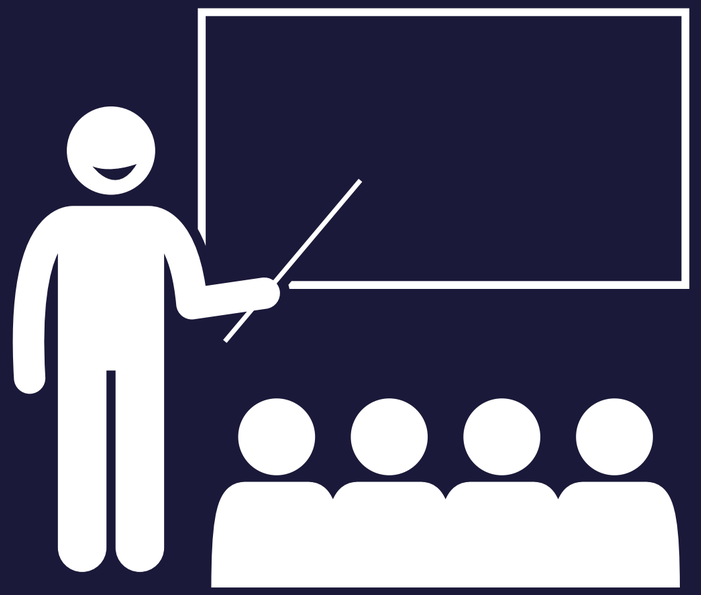
Not everyone is a numbers person, and sometimes there are so many values within a DataFrame that it becomes difficult to comprehend what exactly is going on.

Visualize

Grouping these values in bins can make it easier to visualize large datasets.

Function

Using the Pandas **pd.cut()** function will allow us to "bin" values into groups, which enables more vigorous customization of datasets.



INSTRUCTOR DEMONSTRATION

Binning Data

UNIVERSITY OF KANSAS

ACTIVITY:
BINNING TED

In this activity, you will create bins for TED Talks based on their viewership. After creating the bins, you'll group the DataFrame based on those bins, and then perform some analysis on them.

Suggested Time:
20 minutes

UNIVERSITY OF KANSAS

INSTRUCTIONS: BINNING TED

- Read in the CSV file provided and print it to the screen.
- Find the minimum "views" and maximum "views".
- Using the minimum and maximum "views" as a reference, create 10 bins in which to slice the data.
- Create a new column called "View Group" and fill it with the values collected through your slicing.
- Group the DataFrame based upon the values within "View Group".
- Find out how many rows fall into each group before finding the averages for "comments", "duration", and "languages".

CHECK IN

UNIVERSITY OF KANSAS

SUMMARY

- The dropna() and fillna() methods were covered in Lesson 4.5.2.
- The replace() method was covered in Lesson 4.5.5.
- The count() method was covered in Lesson 4.7.2.
- The value_counts() method was covered in Lesson 4.8.2.
- The groupby() function was covered in Lesson 4.8.4.
- The cut() function was covered in Lesson 4.11.2.