

Module 4

This Week: Pandas

This Week: Pandas

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

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

Module 4

Today's Agenda

Today's Agenda

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

01

Working with DataFrames

02

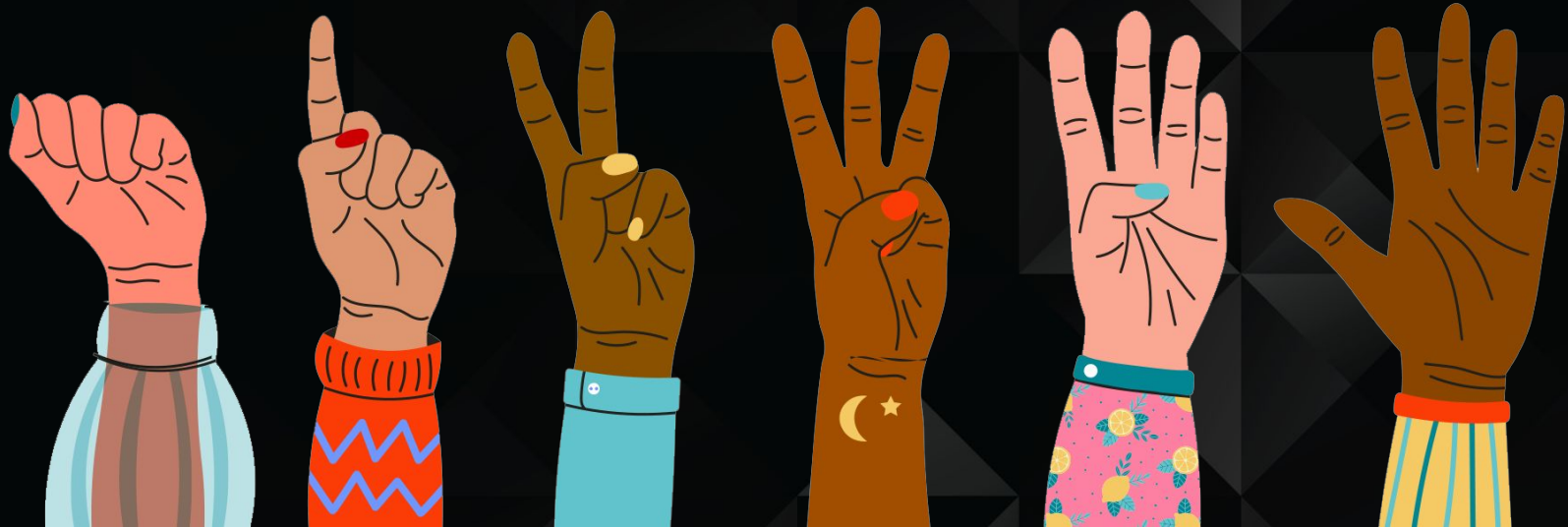
Segment and sort values into bins



**Make sure you've downloaded
any relevant class files!**

FIST TO FIVE:

How comfortable do you feel with this topic?





Cleaning Data



When dealing with massive datasets, it is almost inevitable that duplicate rows, inconsistent spelling, and missing values will crop up.

Cleaning Data

```
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

```
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

value_counts()

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
```

Cleaning Data

`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.

Questions?





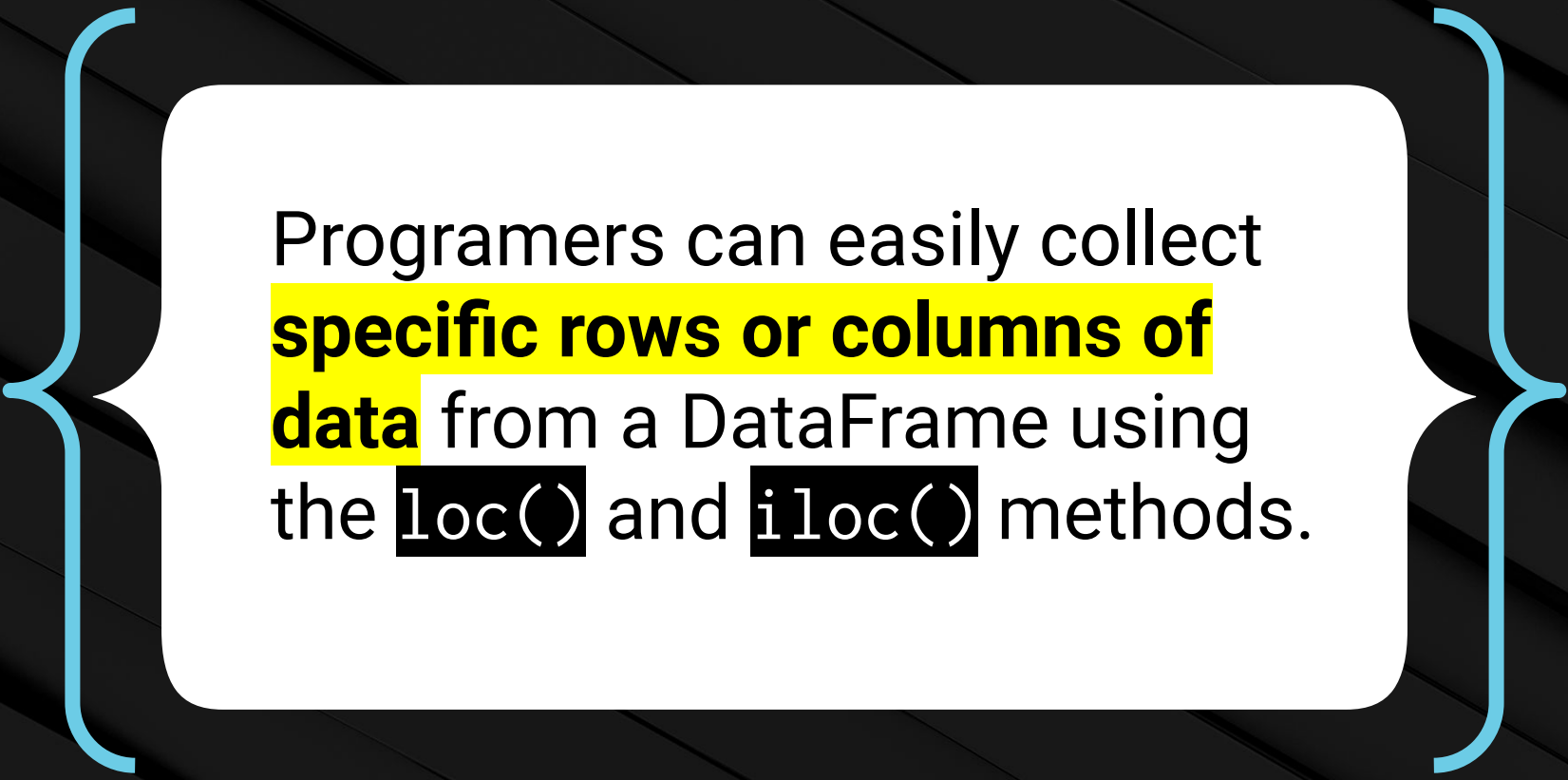
Activity: Portland Crime

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



Exploring Data With `loc` and `iloc`



Programmers can easily collect **specific rows or columns of data** from a DataFrame using the `loc()` and `iloc()` methods.



Instructor Demonstration

loc and lloc

Questions?





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





Let's Review

Pandas Grouping



The `.groupby()` function allows
you to **group Pandas objects
based on a common record.**

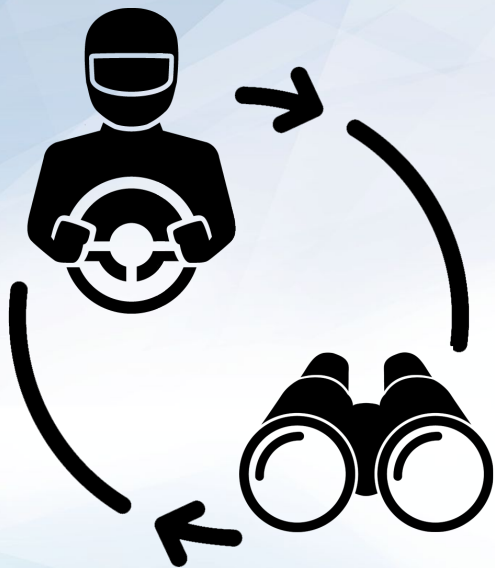


Instructor Demonstration

GroupBy

Questions?





Pair Programming 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



Pair Programming

There are 2 main roles in pair programming:

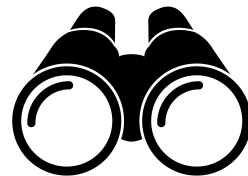
01 Driver

The first is the Driver, whose role is to focus on resolving the current task while talking through their thought process out loud.



02 Navigator

The second is the Navigator, which is equally as important. They will help catch bugs and typos, think about issues to address for efficiency, and use documentation to find resources to help the driver get past a hurdle.





Let's Review



**How would we sort the DataFrame
from greatest to fewest length of
membership in days?**

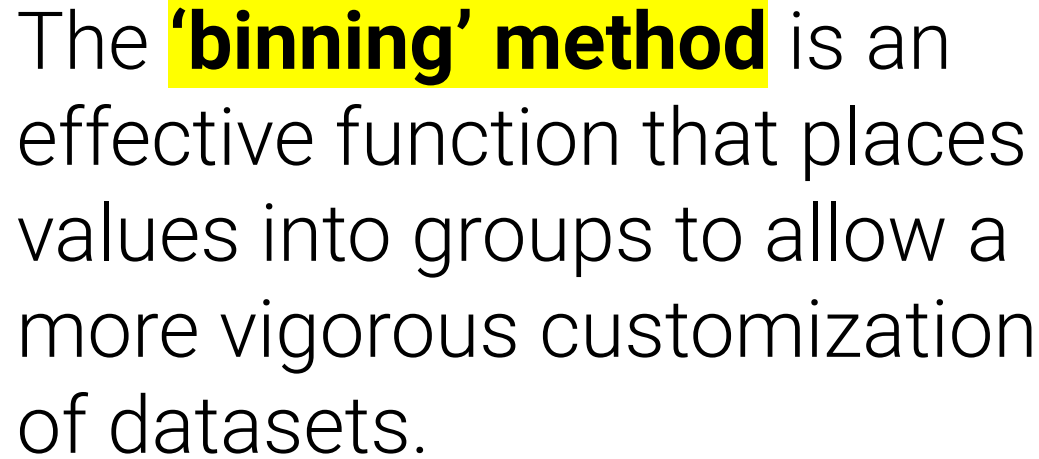
```
sort_values(by='Membership (Days)', ascending=False)  
on the trainers_means DataFrame.
```



Questions?



Binning Data



The **'binning' method** is an effective function that places values into groups to allow a more vigorous customization of datasets.

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

Questions?





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





Let's Review

Questions?

