YData: Introduction to Data Science

Lecture 10: Groups

Overview

Review and continuation of functions and applying functions to a column in a Table

- Prediction example

Applying functions to groups

If there is time: Grouping by multiple columns

Announcements

Homework 3 has been posted

- Due Sunday February 20th at 11pm

Practice 4 has been posted

- Not handed in, but highly recommend that you do it!

Review/continuation of functions and applying functions to columns in a Table

Functions

Functions in Python are created with a def statement

```
def spread(values):
	return  max(values) - min(values)

# what is the output from running this?
spread(make_array(3.14159, 2.718, 1, 9))
```

Functions

Functions can take multiple arguments which can have default values

```
def spread(values, digits = 2):
    return  np.round(max(values) - min(values), digits)

# what is the difference between running these to commands?
spread(make_array(30.14159, 2.718, 20.8, 15.1))
spread(make_array(30.14159, 2.718, 20.8, 15.1), 1)
```

Review: the apply method

The tb.apply() method creates an array by calling a function on every element in input column(s)

tb.apply(function_name, "numeric col")

- First argument: Function to apply
- Other arguments: The input column(s)

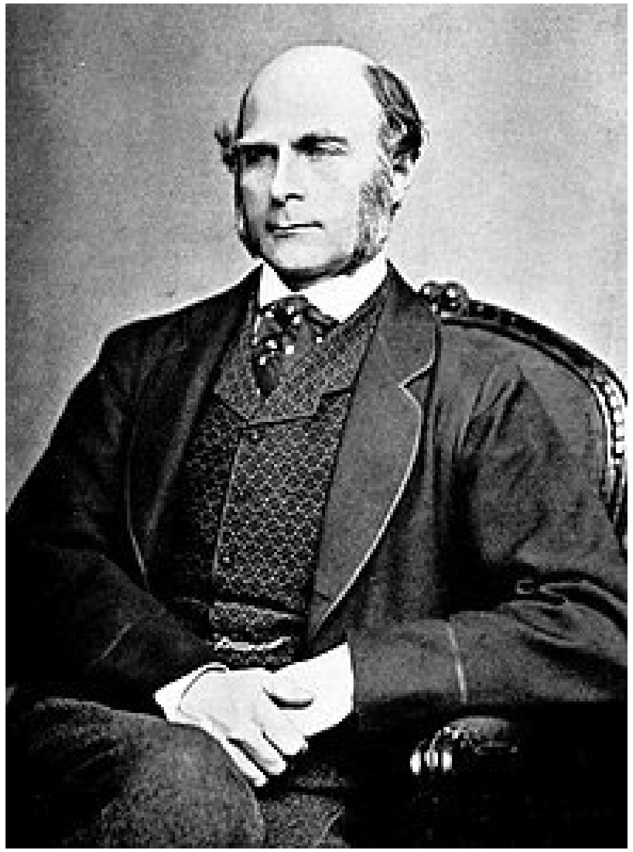
Let's explore this in Jupyter!

Example: Prediction

Francis Galton

- 1822 - 1911
  - Charles Darwin's half-cousin
- A pioneer in making predictions
- Particular (and troublesome) interest in heredity

One of his most famous results involved exploring the relationship between the heights of parents and their children



Let's explore this in Jupyter!

Applying functions to values in two columns

One argument function:

```
tb.apply(one_arg_function, "numeric col")
```

Two argument functions:

```
tb.apply(two_arg_function, "col label first arg", "col label second arg")
```

Grouping by one attribute

Grouping by one column

The tb.group() method aggregates all rows with the same value for a column into a single row in the resulting table.

tb.group("grouping col", agg_function)

- "grouping col": column to group data by
- agg_function: function on how data in each group should be combined

Examples of aggregating functions:

- len: number of items in each group (default if no second argument is specified)
- list: list of all values in each group
- sum: total of all grouped values

Let's explore this in Jupyter!

Cross-classification

Grouping by multiple columns

The tb.group() method can also aggregate values in rows that share the combination of values in multiple columns

```
tb.group(["grouping col1", "grouping col2"], agg_function)
```

- ["grouping col1", "grouping col2"]: list of columns to group by
- agg_function: function on how data in each group should be combined

Let's explore this in Jupyter!