

Manipulating data tables I



Overview

Lab 1 discussion

Discussion of chapter 1 of Astrobball

Comparing players across time

Aging curves

Lab time

Lab 1: questions?

How did it go?

Did anyone try the lab using Pandas instead of the datascience package?

For lab 2, you can use Pandas instead of the datascience package

- Note: the lab was written based on using the datascience package so you'll have to look up corresponding Pandas methods, but if you're using Pandas I assume you are advanced enough to do this.

Discussion on chapter 1 of Astroball

Let's discuss the chapter for 7 minutes in breakout rooms and then have a larger conversation as a group.

- Discuss your quote and reaction to chapter 1.

Discussion on chapter 1 of Astrobball

Interesting quotes and reactions to share?



Sig Mejdal



Jeff Luhnow

Discussion on chapter 1 of Astrobball

More information on the 2009 draft:

- [Wikipedia](#)
- [Discussion of the Cardinals' picks](#)
- [Wikipedia information on baseball drafts in general](#)
- [Full draft results](#)



[Joe Kelly](#)

Round 3, 89th pick



[Matt Carpenter](#)

Round 13, 399th pick



[Trevor Rosenthal](#)

Round 21, 639th pick

Discussion on chapter 1 of Astrobball

Structure of the draft:

- 50 rounds, ~1,500 players drafted (30 teams * 50 rounds)

Pg. 30: A study in 2013 found that:

1. “if a club’s draft produces nine players who appear in the majors for even a single game, it’s in the 95th percentile.”
2. “The 95th percentile for number of everyday players (def 1,500 PA) is four.”

Review and continuation of Table functions

Operations we can apply to Tables

- Filter out rows
- Select columns
- Add columns to a table
- Apply functions to groups of data
- Create scatter plots

1. How can we get a subset of **rows** from a Table?

tb.**where**('col_name', value)

Predicate	Example	Result
are.equal_to	are.equal_to(50)	Find rows with values equal to 50
are.not_equal_to	are.not_equal_to(50)	Find rows with values not equal to 50
are.above	are.above(50)	Find rows with values above (and not equal to) 50
are.above_or_equal_to	are.above_or_equal_to(50)	Find rows with values above 50 or equal to 50
are.below	are.below(50)	Find rows with values below 50
are.between	are.between(2, 10)	Find rows with values above or equal to 2 and below 10

2. How can we get a subset of **columns** from a Table?

```
tb.select('col_1', 'col_2')    # returns a Table with 2 columns
```

```
my_table = tb['col_name']      # returns a ndarray with data from  
'col_name'
```

```
type(my_table)
```

3. How can we add a new **column** to a Table?

```
tb.with_column('col_name', ndarray)
```

4. How can we apply a function to groups in a Table?

```
tb.group('group_col', function_name)
```

Applies the function_name to all columns separately based on the value levels of 'group_col'

5. How can create a scatter plot

```
tb.scatter('x_col', 'y_col')
```

Questions?

Last class we asked: who is better?

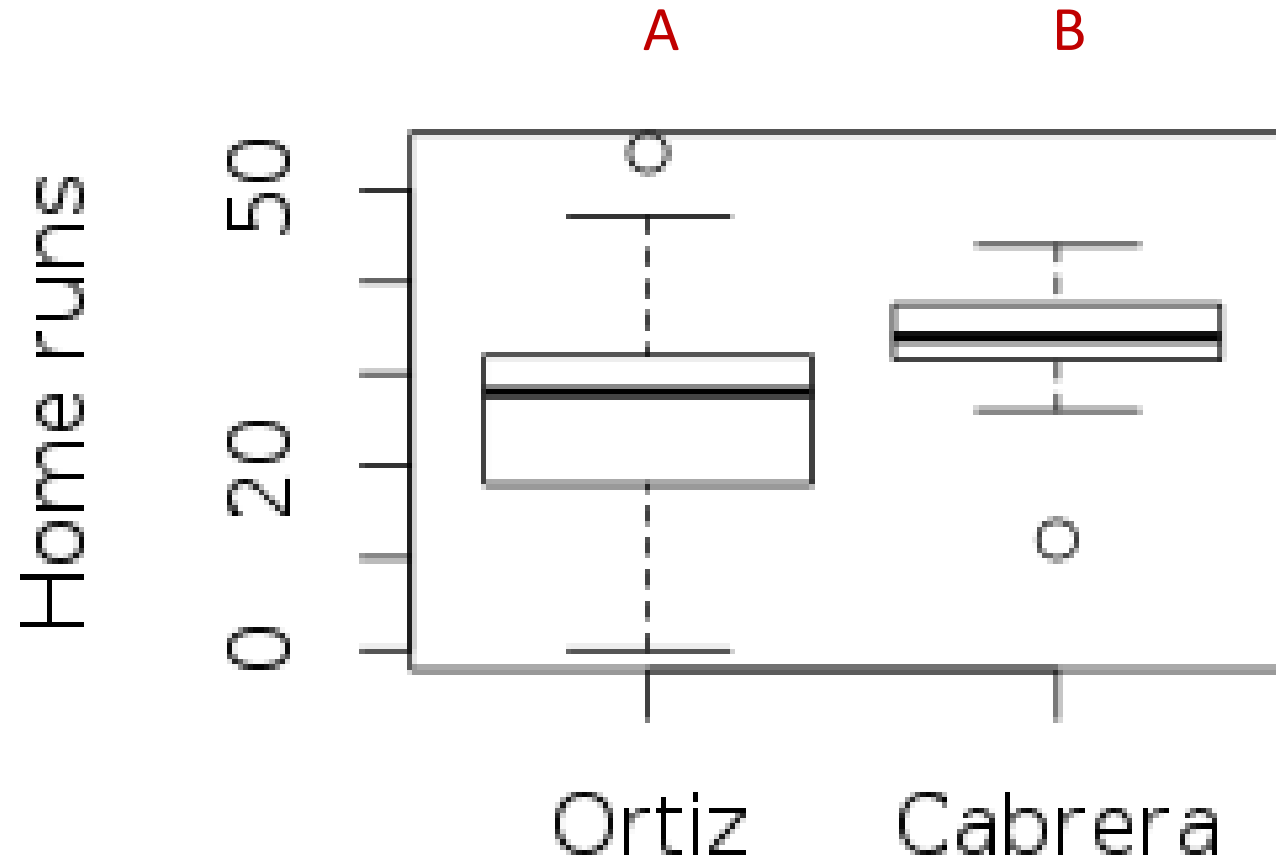


Miguel Cabrera:
HR in 2014 = 25



David Ortiz:
HR in 2014 = 35

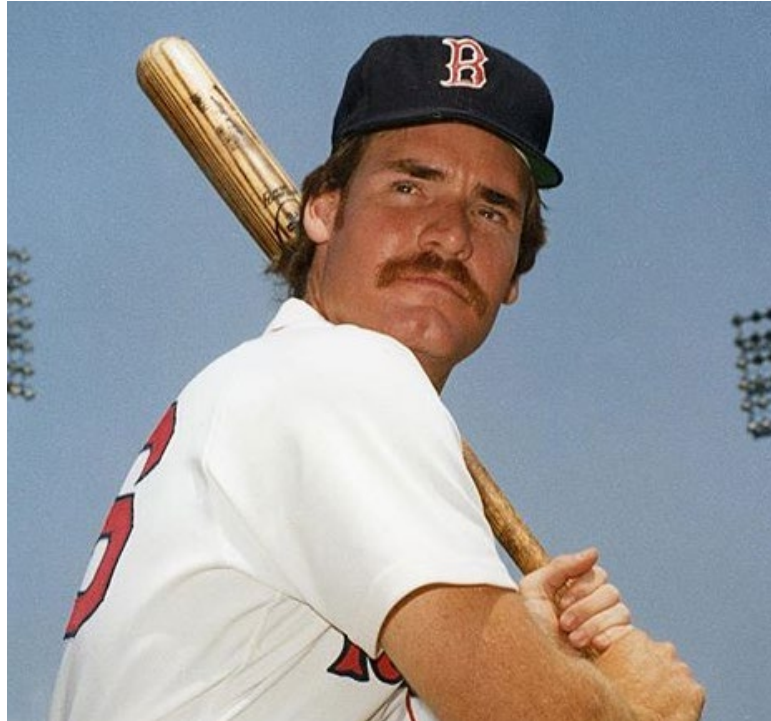
Comparing players with side-by-side box plots



How would you describe the differences between these two players in terms of HRs? **Who is better?**

Let's compare two more players

1985



Wade Boggs: BA = .368

Career best seasons

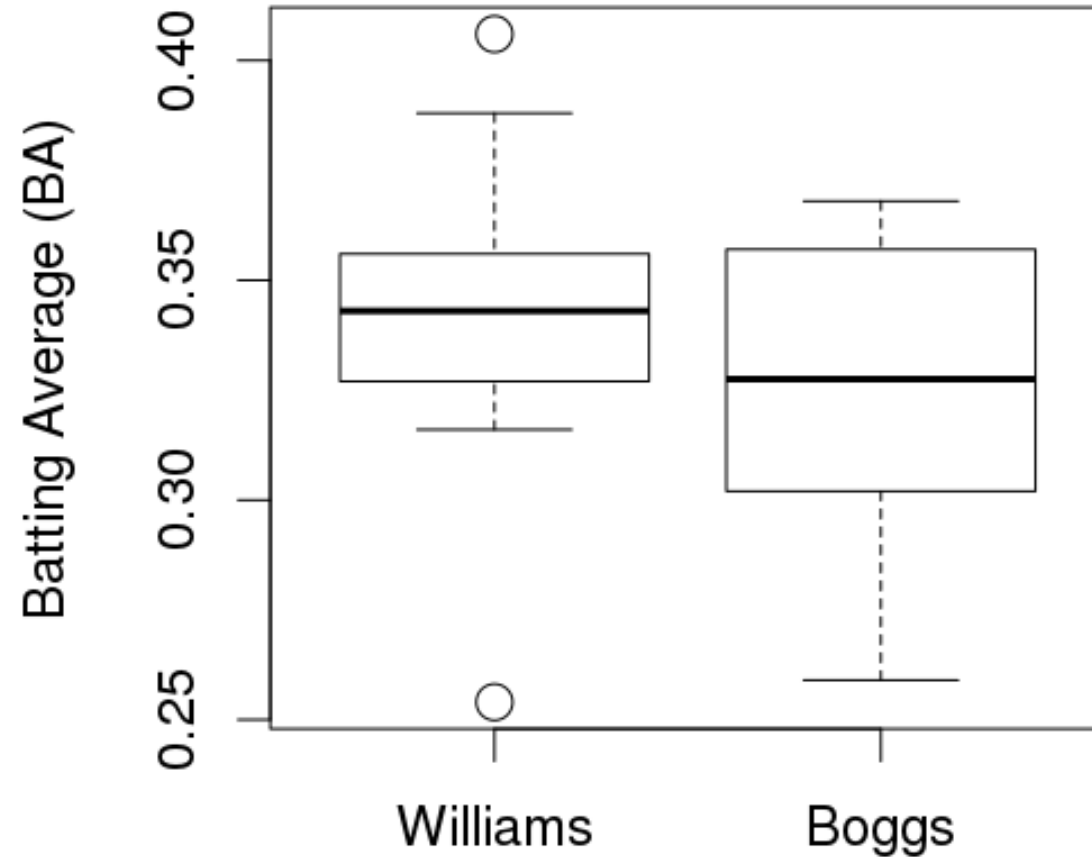
1941



Ted Williams: BA = .406

Who is better?

Who is best here?



Is Ted Williams better than Wade Boggs?

Ted Williams hit .406 in 1941
23 people hit over .400 before him in MLB
but no one in the MLB has since...



Examining how the game has changed

Let's figure out how to plot the MLB leader in batting average each year to see how the game has changed

- i.e., what is the highest BA every year since 1871

Examining how the game has changed

Let's start with the Lahman batting data

```
batting = Table.read_table('https://raw.githubusercontent.com/emeyers/SDS173/master/data/Batting.csv')
```

playerID	yearID	stint	teamID	lgID	G	AB	R	H	2B	3B	HR	RBI	SB	CS	BB	SO	IBB	HBP	SH	SF	GIDP
abercda01	1871	1	TRO	nan	1	4	0	0	0	0	0	0	0	0	0	0	nan	nan	nan	nan	0
addybo01	1871	1	RC1	nan	25	118	30	32	6	0	0	13	8	1	4	0	nan	nan	nan	nan	0
allisar01	1871	1	CL1	nan	29	137	28	40	4	5	0	19	3	1	2	5	nan	nan	nan	nan	1

... (89518 rows omitted)

What are the steps needed to find the league leader in batting average as a function of the year?

What are the steps needed to find the league leader in batting average as a function of the year?

Steps to find the league leader in batting average as a function of the year?

Let's try this out in breakout rooms!

https://github.com/emeyers/SDS173/blob/main/labs/lab_02/class_03_material.ipynb

Steps:

- 1.
- 2.
- 3.

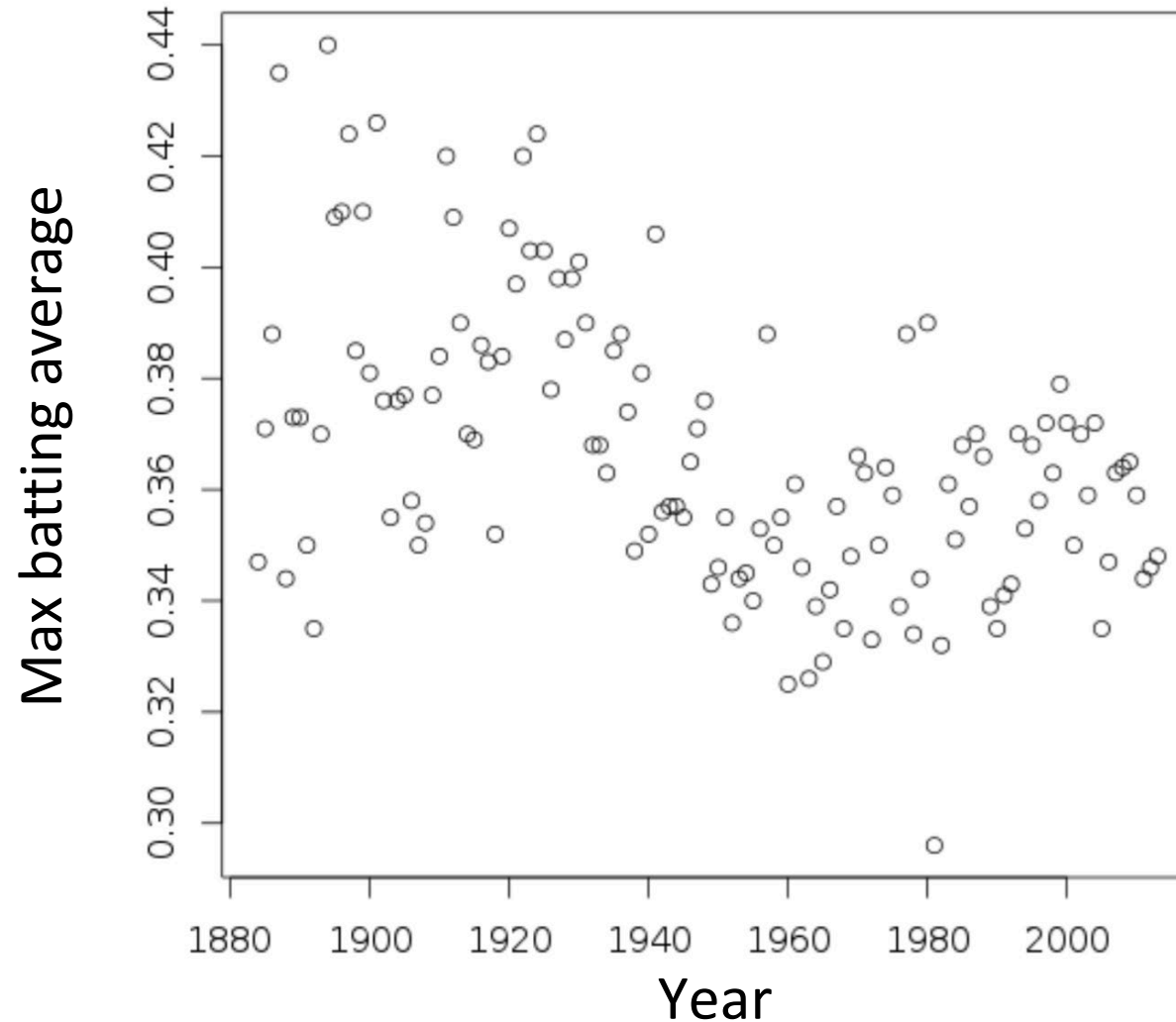
What are the steps needed to find the league leader in batting average as a function of the year?

What are the main steps we need to do to achieve this goal?

- 1. Reduce the table to only players with > 502 plate appearances
- 2. Add BA to the table
- 3. Find the maximum BA separately for each year

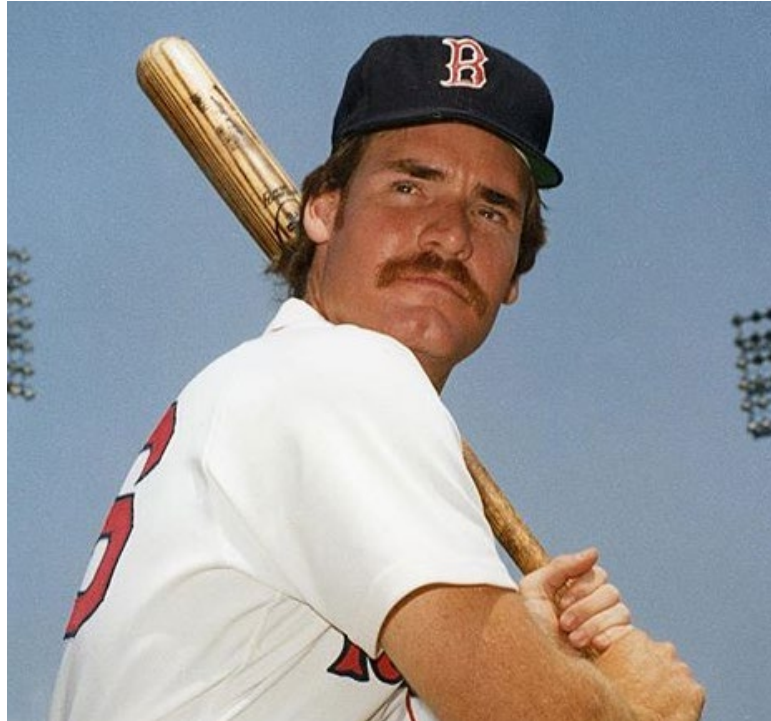
How can we do it?

Have the best players gotten worse at hitting over the past 140 years?



Let's compare two players

1985



Wade Boggs: BA = .368

Career best seasons

1941



Ted Williams: BA = .406

Who is better?

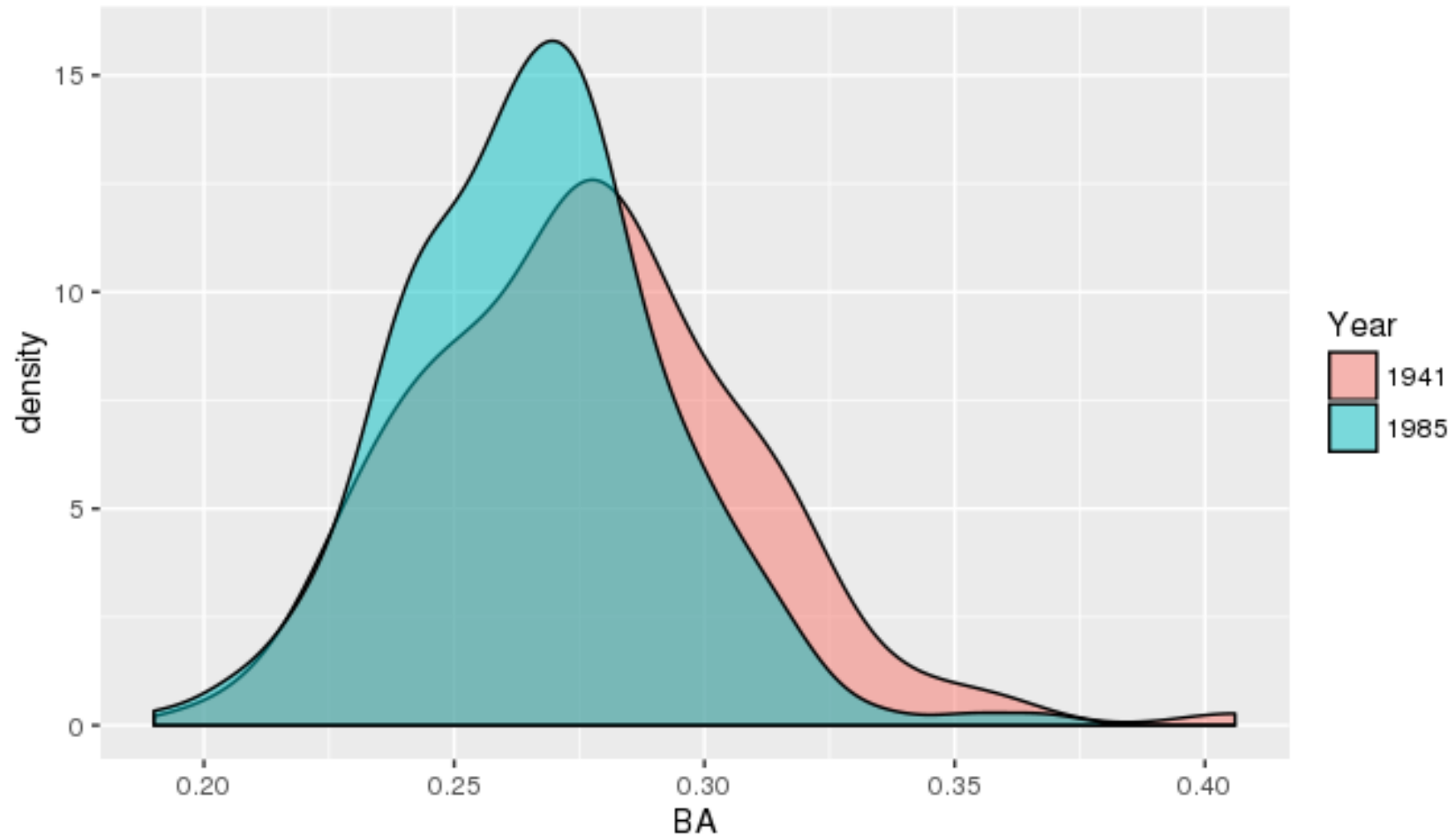
Comparing players across time periods

Problem: baseball has changed from 1871 to now

We can't simply compare statistics to judge how great a baseball player is when comparing across decades

Useful to judge the 'greatness' of players relative to their peers

Density of batting average 1941 vs. 1985



Do the batting averages look similar in these years?

z-scores

$$z_i = \frac{x_i - \bar{x}}{s}$$

The z-scores tells how many standard deviations a value x is from the mean (\bar{x}), in a way that is independent of the units of measurement.

z-scores for comparing players across eras

$$z_i = \frac{x_i - \bar{x}}{s}$$

When comparing players across eras, we will use the mean (\bar{x}), and standard deviation (s) from each era separately.

This will give a measure of player performance relative to their peers in the same era

Comparing Ted and Wade to their peers

$$z_i = \frac{x_i - \bar{x}}{s}$$

In 1941:

- Mean batting average was: .276
- Standard deviation in batting average was: .033
- Ted William's batting average was: .406

In 1985:

- Mean batting average was: .266
- Standard deviation in batting average was: .027
- Wade Bogg's batting average was: .368

Calculate z-scores for Ted and Wade's batting averages.

Who was better relative to their peers?

Comparing Ted and Wade to their peers

$$z_i = \frac{x_i - \bar{x}}{s}$$

Wade's batting average z-score: 3.78

Ted's batting average z-score: 3.94

Who is the better hitter relative to their peers?

Comparing Ted and Wade's full career

Use Python to get z-scores from Ted and Wade for all years in their careers

Create side-by-side boxplots comparing their z-score batting averages




Aging curves

Player's abilities change as they age

- Usually get better in the beginning of their careers
- Get worse toward the end of their careers

Can we create an average aging curve (say for BA) across players for the Lahman data?

Notice Age is not in that table



playerID	yearID	stint	teamID	lgID	G	AB	R	H	2B	3B	HR	RBI	SB	CS	BB	SO	IBB	HBP	SH	SF	GIDP
abercda01	1871	1	TRO	nan	1	4	0	0	0	0	0	0	0	0	0	0	nan	nan	nan	nan	0
addybo01	1871	1	RC1	nan	25	118	30	32	6	0	0	13	8	1	4	0	nan	nan	nan	nan	0
allisar01	1871	1	CL1	nan	29	137	28	40	4	5	0	19	3	1	2	5	nan	nan	nan	nan	1

... (89518 rows omitted)

Calculating average aging curve for BA

What are the main steps we need to do to achieve this goal?

- 1. Need to add Age in the table
 - How can we get Age?
 - Player's birth year is in the People.csv table
 - $\text{Age} = \text{Current year} - \text{birth year}$
- 2. Find the average BA separately for each age

How can we do it?

Calculating average aging curve for BA

What are the main steps we need to do to achieve this goal?

- 1. Need to add Age in the table
 - How can we get Age?
 - Player's birth year is in the People.csv table
 - $\text{Age} = \text{Current year} - \text{birth year}$
- 2. Find the average BA separately for each age

How can we do it?

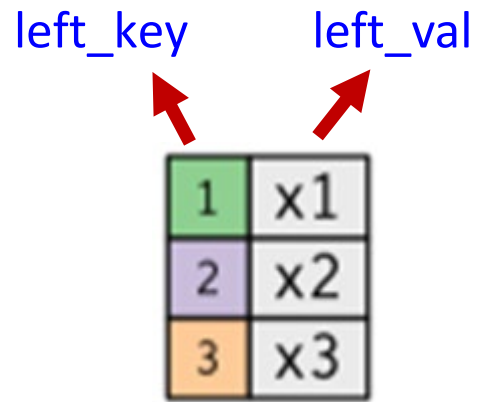
- We can combine Batting and People table using a join
 - `tb1.join('col_tb1', tb2, 'col_tb2')`
 - `batting.join('playerID', players, 'playerID')`

Joining tables

Left and right tables

Suppose we have two Tables x and y

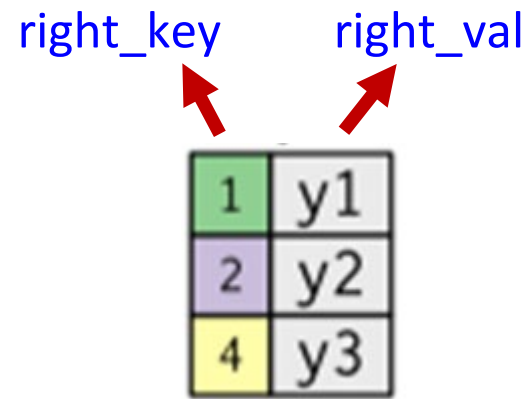
- x has two variables called left_key, and left_val
- y has two variables called right_key and right_val



The diagram shows a table with two columns. The first column is labeled 'left_key' in blue text with a red arrow pointing to it. The second column is labeled 'left_val' in blue text with a red arrow pointing to it. The table has three rows: the first row has a green cell with '1' and a grey cell with 'x1'; the second row has a purple cell with '2' and a grey cell with 'x2'; the third row has an orange cell with '3' and a grey cell with 'x3'.

1	x1
2	x2
3	x3

Left table (x)



The diagram shows a table with two columns. The first column is labeled 'right_key' in blue text with a red arrow pointing to it. The second column is labeled 'right_val' in blue text with a red arrow pointing to it. The table has three rows: the first row has a green cell with '1' and a grey cell with 'y1'; the second row has a purple cell with '2' and a grey cell with 'y2'; the third row has a yellow cell with '4' and a grey cell with 'y3'.

1	y1
2	y2
4	y3

Right table (y)

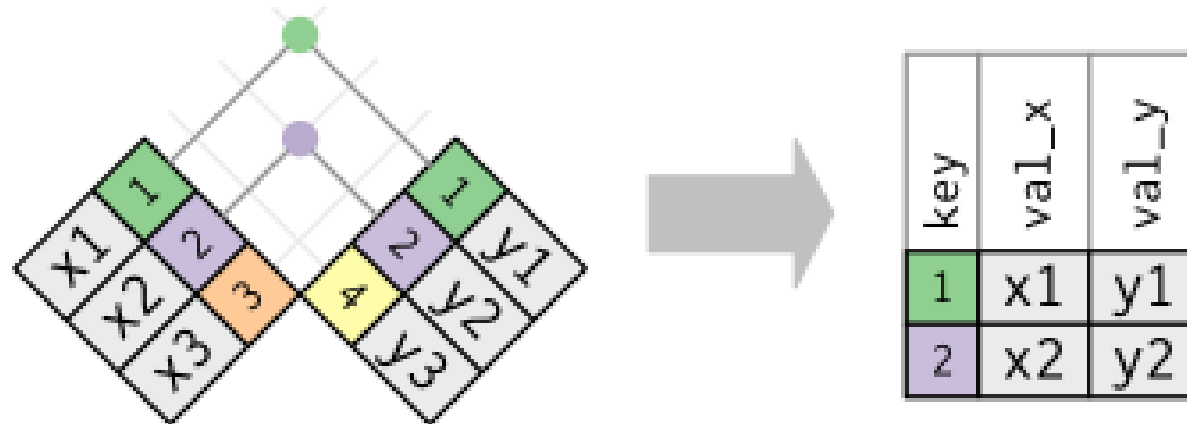
```
x = Table().with_column('left_key', make_array(1, 2, 3)).with_column('left_value', make_array("x1", "x2", "x3"))
```

```
y = Table().with_column('right_key', make_array(1, 2, 4)).with_column('right_value', make_array("y1", "y2", "y3"))
```


Inner joins

Inner joins only keep rows in which there are matches between the keys in both tables

- The `join()` method in the datascience package does inner joins

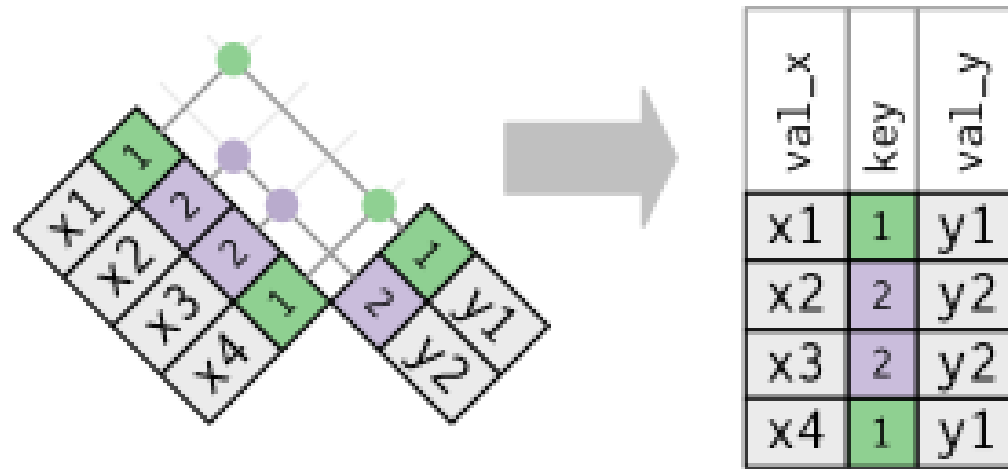


```
x.join("left_key" y, "right_key")
```

Duplicate keys

Duplicate keys are useful if there is a one-to-many relationship

- duplicates are usually in the left table

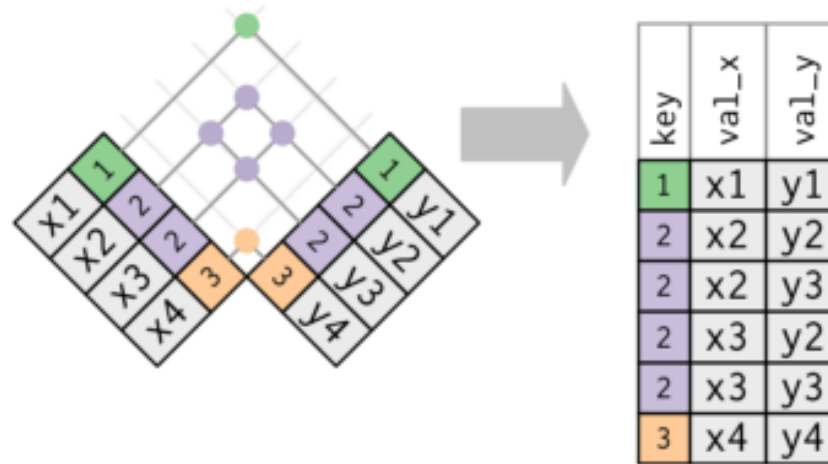


For example, joining the batting table with the player table

Duplicate keys

If both tables have duplicate keys you get all possible combinations (Cartesian product)

- This is usually an error!



Again, always check the output after you join a table because even if there is not a syntax error you might not get the table you are expecting!

Duplicate keys

To deal with duplicate keys in both tables, we can join the tables using multiple keys in order to make sure that each row is uniquely specified

We can do this using the syntax:

```
table1.join(['a', 'b'], table2, ['a', 'd'])
```

Calculating average aging curve for BA

What are the main steps we need to do to achieve this goal?

- 1. Need to add Age in the table
 - How can we get Age?
 - Player's birth year is in the People.csv table
 - $\text{Age} = \text{Current year} - \text{birth year}$
- 2. Find the average BA separately for each age

How can we do it?

- Once the tables are joined we can pull out the yearID and the birthyear columns as arrays
- Subtract these ndarrays and add them back to the table
- Calculate mean BA grouped by age

Let's try it in Python