hw03

September 21, 2024

[23]:
```
# Initialize Otter
import otter
grader = otter.Notebook("hw03.ipynb")
```

1 Homework 3: Table Manipulation and Visualization

Please complete this notebook by filling in the cells provided. Before you begin, execute the previous cell to load the provided tests.

Helpful Resource: - Python Reference: Cheat sheet of helpful array & table methods used in our class!

Recommended Reading: * Visualization

For all problems that you must write explanations and sentences for, you **must** provide your answer in the designated space. **Moreover, throughout this homework and all future ones, please be sure to not re-assign variables throughout the notebook!** For example, if you use `max_temperature` in your answer to one question, do not reassign it later on. Otherwise, you will fail tests that you thought you were passing previously!

Note: This homework has hidden tests on it. That means even though tests may say 100% passed, doesn't mean your final grade will be 100%. We will be running more tests for correctness once everyone turns in the homework.

Directly sharing answers is not okay, but discussing problems with the course staff or with other students is encouraged.

You should start early so that you have time to get help if you're stuck.

[24]:
```
# Don't change this cell; just run it.

import numpy as np
from datascience import *
import warnings
warnings.simplefilter('ignore', FutureWarning)

# These lines do some fancy plotting magic.\n",
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plots
```

```
plots.style.use('fivethirtyeight')
```

1.1 1. Unemployment

The Great Recession of 2008-2009 was a period of economic decline observed globally, with scale and timing varying from country to country. In the United States, it resulted in a rapid rise in unemployment that affected industries and population groups to different extents.

The Federal Reserve Bank of St. Louis publishes data about jobs in the US. Below, we've loaded data on unemployment in the United States. There are many ways of defining unemployment, and our dataset includes two notions of the unemployment rate:

1. Among people who are able to work and are looking for a full-time job, the percentage who can't find a job. This is called the Non-Employment Index, or NEI.
2. Among people who are able to work and are looking for a full-time job, the percentage who can't find any job *or* are only working at a part-time job. The latter group is called "Part-Time for Economic Reasons", so the acronym for this index is NEI-PTER. (Economists are great at marketing.)

The source of the data is here.

Question 1. The data are in a CSV file called `unemployment.csv`. Load that file into a table called `unemployment`.

[25]:
```
unemployment = Table.read_table("unemployment.csv")
unemployment
```

[25]:
```
Date       | NEI     | NEI-PTER
1994-01-01 | 10.0974 | 11.172
1994-04-01 | 9.6239  | 10.7883
1994-07-01 | 9.3276  | 10.4831
1994-10-01 | 9.1071  | 10.2361
1995-01-01 | 8.9693  | 10.1832
1995-04-01 | 9.0314  | 10.1071
1995-07-01 | 8.9802  | 10.1084
1995-10-01 | 8.9932  | 10.1046
1996-01-01 | 9.0002  | 10.0531
1996-04-01 | 8.9038  | 9.9782
… (80 rows omitted)
```

[26]:
```
grader.check("q1_1")
```

[26]:
```
q1_1 results: All test cases passed!
```

Question 2. Sort the data in descending order by NEI, naming the sorted table `by_nei`. Create another table called `by_nei_pter` that's sorted in descending order by NEI-PTER instead.

[27]:
```
by_nei = unemployment.sort("NEI", descending = True)
by_nei_pter = unemployment.sort("NEI-PTER", descending = True)
```

[28]:
```
grader.check("q1_2")
```

[28]:
```
q1_2 results: All test cases passed!
```

[18]:
```
# Run this cell to check your by_nei table. You do not need to change the code.
by_nei.show(5)
```

```
<IPython.core.display.HTML object>
```

[29]:
```
# Run this cell to check your by_nei_pter table. You do not need to change the␣
↪code.
by_nei_pter.show(5)
```

```
<IPython.core.display.HTML object>
```

Question 3. Use `take` to make a table containing the data for the 11 quarters when NEI was greatest. Call that table `greatest_nei`.

`greatest_nei` should be sorted in descending order of `NEI`. Note that each row of `unemployment` represents a quarter.

[72]:
```
greatest_nei = by_nei.take(np.arange(11))
greatest_nei
```

[72]:
```
Date       | NEI     | NEI-PTER
2009-10-01 | 10.9698 | 12.8557
2010-01-01 | 10.9054 | 12.7311
2009-07-01 | 10.8089 | 12.7404
2009-04-01 | 10.7082 | 12.5497
2010-04-01 | 10.6597 | 12.5664
2010-10-01 | 10.5856 | 12.4329
2010-07-01 | 10.5521 | 12.3897
2011-01-01 | 10.5024 | 12.3017
2011-07-01 | 10.4856 | 12.2507
2011-04-01 | 10.4409 | 12.247
… (1 rows omitted)
```

[58]:
```
grader.check("q1_3")
```

[58]:
```
q1_3 results: All test cases passed!
```

Question 4. It's believed that many people became PTER (recall: "Part-Time for Economic Reasons") in the "Great Recession" of 2008-2009. NEI-PTER is the percentage of people who are unemployed (included in the NEI) plus the percentage of people who are PTER.

Compute an array containing the percentage of people who were PTER in each quarter. (The first element of the array should correspond to the first row of `unemployment`, and so on.)

Note: Use the original `unemployment` table for this.

[83]:
```
pter = (unemployment.column("NEI-PTER") - unemployment.column("NEI"))
pter
```

[83]:
```
array([ 1.0746,  1.1644,  1.1555,  1.129 ,  1.2139,  1.0757,  1.1282,
        1.1114,  1.0529,  1.0744,  1.1004,  1.0747,  1.0705,  1.0455,
        1.008 ,  0.9734,  0.9753,  0.8931,  0.9451,  0.8367,  0.8208,
        0.8105,  0.8248,  0.7578,  0.7251,  0.7445,  0.7543,  0.7423,
        0.7399,  0.7687,  0.8418,  0.9923,  0.9181,  0.9629,  0.9703,
        0.9575,  1.0333,  1.0781,  1.0675,  1.0354,  1.0601,  1.01  ,
        1.0042,  1.0368,  0.9704,  0.923 ,  0.9759,  0.93  ,  0.889 ,
        0.821 ,  0.9409,  0.955 ,  0.898 ,  0.8948,  0.9523,  0.9579,
        1.0149,  1.0762,  1.2873,  1.4335,  1.7446,  1.8415,  1.9315,
        1.8859,  1.8257,  1.9067,  1.8376,  1.8473,  1.7993,  1.8061,
        1.7651,  1.7927,  1.7286,  1.6387,  1.6808,  1.6805,  1.6629,
        1.6253,  1.6477,  1.6298,  1.4796,  1.5131,  1.4866,  1.4345,
        1.3675,  1.3097,  1.2319,  1.1735,  1.1844,  1.1746])
```

[84]:
```
grader.check("q1_4")
```

[84]:
```
q1_4 results: All test cases passed!
```

Question 5. Add `pter` as a column to `unemployment` (name the column `PTER`) and sort the resulting table by that column in descending order. Call the resulting table `by_pter`.

Try to do this with a single line of code, if you can.

[102]:
```
by_pter = unemployment.with_columns("PTER", pter).sort("PTER", descending =␣
↪True)
by_pter
```

[102]:

Date	NEI	NEI-PTER	PTER
2009-07-01	10.8089	12.7404	1.9315
2010-04-01	10.6597	12.5664	1.9067
2009-10-01	10.9698	12.8557	1.8859
2010-10-01	10.5856	12.4329	1.8473
2009-04-01	10.7082	12.5497	1.8415
2010-07-01	10.5521	12.3897	1.8376
2010-01-01	10.9054	12.7311	1.8257
2011-04-01	10.4409	12.247	1.8061
2011-01-01	10.5024	12.3017	1.7993
2011-10-01	10.3287	12.1214	1.7927

… (80 rows omitted)

[103]:
```
grader.check("q1_5")
```

[103]:
```
q1_5 results: All test cases passed!
```

Question 6. Create a line plot of PTER over time. To do this, create a new table called

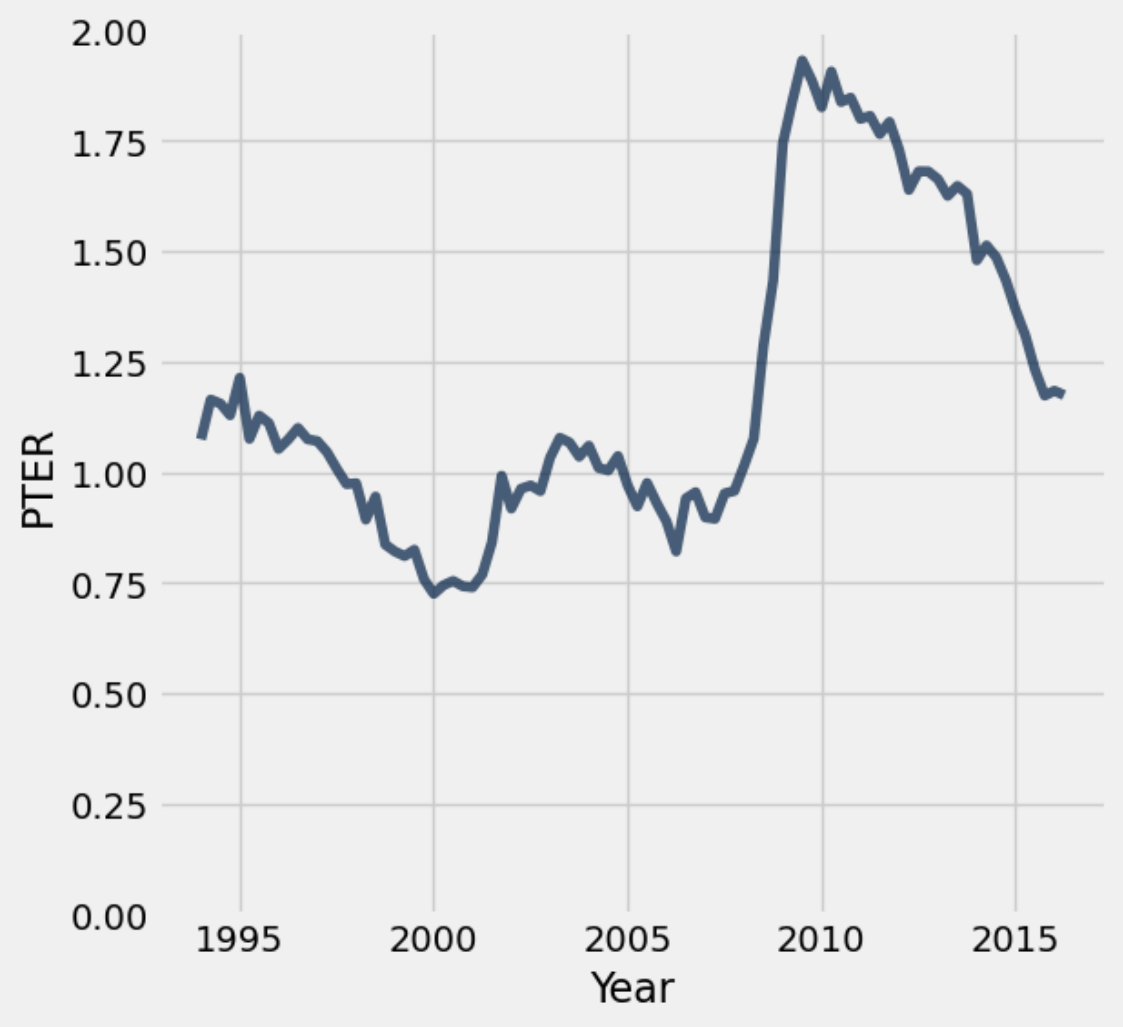
`pter_over_time` by making a copy of the `unemployment` table and adding two new columns: `Year` and `PTER` using the `year` array and the `pter` array, respectively. Then, generate a line plot using one of the table methods you've learned in class.

The order of the columns matter for our correctness tests, so be sure `Year` comes before `PTER`.

Note: When constructing `pter_over_time`, do not just add the `year` column to the `by_pter` table. Please follow the directions in the question above.

[125]:
```
year = 1994 + np.arange(by_pter.num_rows)/4
pter_over_time = unemployment.with_columns("Year", year, "PTER", pter).
 ↪plot("Year", "PTER")


plots.ylim(0,2); # Do not change this line
```



[123]:
```
grader.check("q1_6")
```

[123]:
```
q1_6 results: All test cases passed!
```

Question 7. Were PTER rates high during the Great Recession (that is to say, were PTER rates particularly high in the years 2008 through 2011)? Assign `highPTER` to `True` if you think PTER rates were high in this period, or `False` if you think they weren't.

[130]:
```
highPTER = True
```

[131]:
```
grader.check("q1_7")
```

[131]:
```
q1_7 results: All test cases passed!
```

1.2 2. Birth Rates

The following table gives Census-based population estimates for each US state on both July 1, 2015 and July 1, 2016. The last four columns describe the components of the estimated change in population during this time interval. **For all questions below, assume that the word "states" refers to all 52 rows including Puerto Rico and the District of Columbia.**

The data was taken from here. (Note: If it doesn't download for you when you click the link, please copy and paste it into your address bar!) If you want to read more about the different column descriptions, click here.

The raw data is a bit messy—run the cell below to clean the table and make it easier to work with.

[132]:
```
# Don't change this cell; just run it.
pop = Table.read_table('nst-est2016-alldata.csv').where('SUMLEV', 40).
 ↪select([1, 4, 12, 13, 27, 34, 62, 69])
pop = pop.relabeled('POPESTIMATE2015', '2015').relabeled('POPESTIMATE2016',␣
 ↪'2016')
pop = pop.relabeled('BIRTHS2016', 'BIRTHS').relabeled('DEATHS2016', 'DEATHS')
pop = pop.relabeled('NETMIG2016', 'MIGRATION').relabeled('RESIDUAL2016',␣
 ↪'OTHER')
pop = pop.with_columns("REGION", np.array([int(region) if region != "X" else 0␣
 ↪for region in pop.column("REGION")]))
pop.set_format([2, 3, 4, 5, 6, 7], NumberFormatter(decimals=0)).show(5)
```

```
<IPython.core.display.HTML object>
```

Question 1. Assign `us_birth_rate` to the total US annual birth rate during this time interval. The annual birth rate for a year-long period is the total number of births in that period as a proportion of the total population size at the start of the time period.

Hint: Which year corresponds to the start of the time period?

[138]:
```
us_birth_rate = pop.column('BIRTHS').sum() / pop.column('2015').sum()
us_birth_rate
```

[138]:
```
0.012358536498646102
```

[139]:
```
grader.check("q2_1")
```

[139]:
```
q2_1 results: All test cases passed!
```

Question 2. Assign `movers` to the number of states for which the **absolute value** of the **annual rate of migration** was higher than 1%. The annual rate of migration for a year-long period is the net number of migrations (in and out) as a proportion of the population size at the start of the period. The `MIGRATION` column contains estimated annual net migration counts by state.

Hint: `migration_rates` should be a table and `movers` should be a number.

[184]:
```
migration_rates = pop.with_columns(('MIGRATION_RATES'), abs(pop.
 ↪column('MIGRATION') / pop.column('2015')))
movers = migration_rates.where("MIGRATION_RATES", (are.above(0.01))).num_rows␣
 ↪#u can also use len
movers
```

[184]:
```
9
```

[151]:
```
grader.check("q2_2")
```

[151]:
```
q2_2 results: All test cases passed!
```

Question 3. Assign `west_births` to the total number of births that occurred in region 4 (the Western US).

Hint: Make sure you double check the type of the values in the `REGION` column and appropriately filter (i.e. the types must match!).

[161]:
```
west_births = pop.where('REGION', are.equal_to(4)).column("BIRTHS").sum()
west_births
#pop.wher or pop.column?
```

[161]:
```
979657
```

[162]:
```
grader.check("q2_3")
```

[162]:
```
q2_3 results: All test cases passed!
```

Question 4. In the next question, you will be creating a visualization to understand the relationship between birth and death rates. The annual death rate for a year-long period is the total number of deaths in that period as a proportion of the population size at the start of the time period.

What visualization is most appropriate to see if there is an association between birth and death rates during a given time interval?

1. Line Graph
2. Bar Chart
3. Scatter Plot

Assign `visualization` below to the number corresponding to the correct visualization.

[163]:
```
visualization = 3
```

[164]:
```
grader.check("q2_4")
```

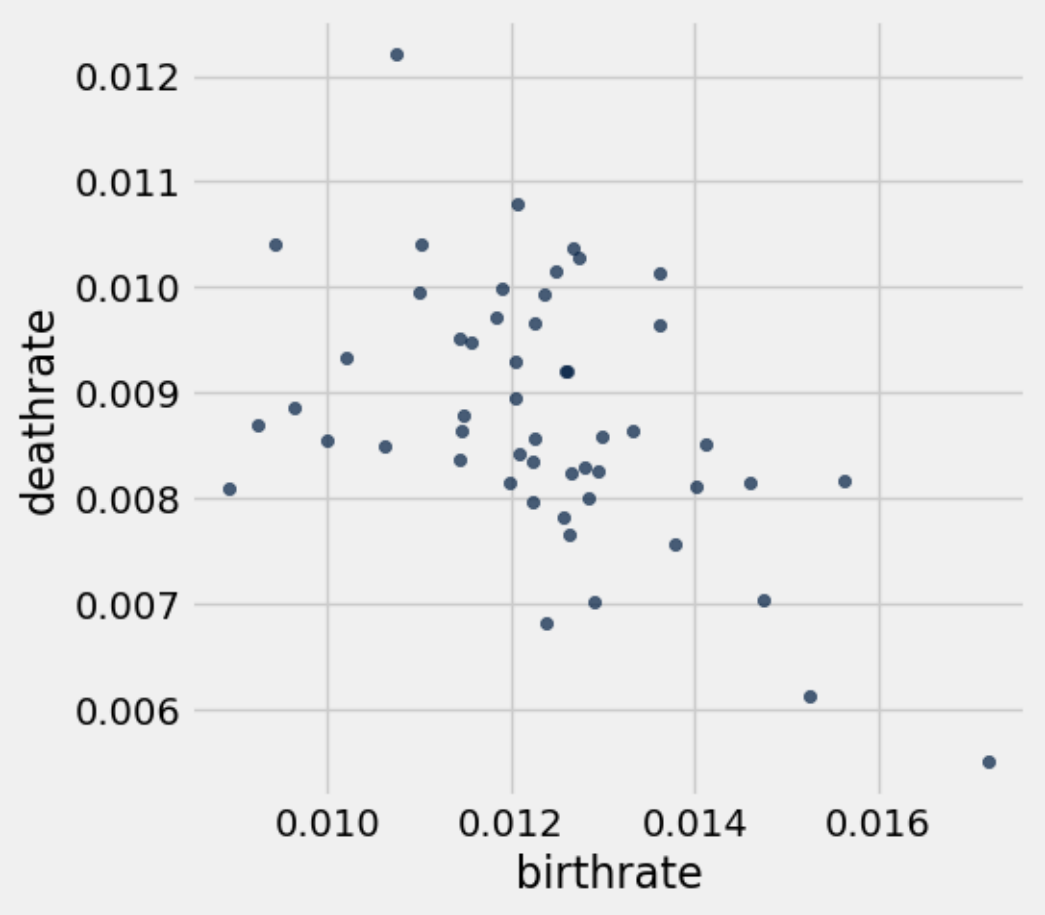
[164]:
```
q2_4 results: All test cases passed!
```

Question 5. In the code cell below, create a visualization that will help us determine if there is an association between birth rate and death rate during this time interval. It may be helpful to create an intermediate table here.

Things to consider:

- What type of chart will help us illustrate an association between 2 variables?
- How can you manipulate a certain table to help generate your chart?
- Check out the Recommended Reading for this homework!

[181]:
```
# In this cell, use birth_rates and death_rates to generate your visualization
birth_rates = pop.column('BIRTHS') / pop.column('2015')
death_rates = pop.column('DEATHS') / pop.column('2015')
pop.with_columns("birthrate", birth_rates, "deathrate", death_rates).
 ↪scatter("birthrate", "deathrate")
```



Question 6. True or False: There is an association between birth rate and death rate during this time interval.

Assign `assoc` to `True` or `False` in the cell below.

[172]:
```
assoc = True
```

[173]:
```
grader.check("q2_6")
```

[173]:
```
q2_6 results: All test cases passed!
```

1.3 3. Uber

Below we load tables containing 200,000 weekday Uber rides in the Manila, Philippines, and Boston, Massachusetts metropolitan areas from the Uber Movement project. The `sourceid` and `dstid` columns contain codes corresponding to start and end locations of each ride. The `hod` column contains codes corresponding to the hour of the day the ride took place. The `ride time` column contains the length of the ride in minutes.

[174]:
```
boston = Table.read_table("boston.csv")
manila = Table.read_table("manila.csv")
print("Boston Table")
boston.show(4)
print("Manila Table")
manila.show(4)
```

```
Boston Table
```

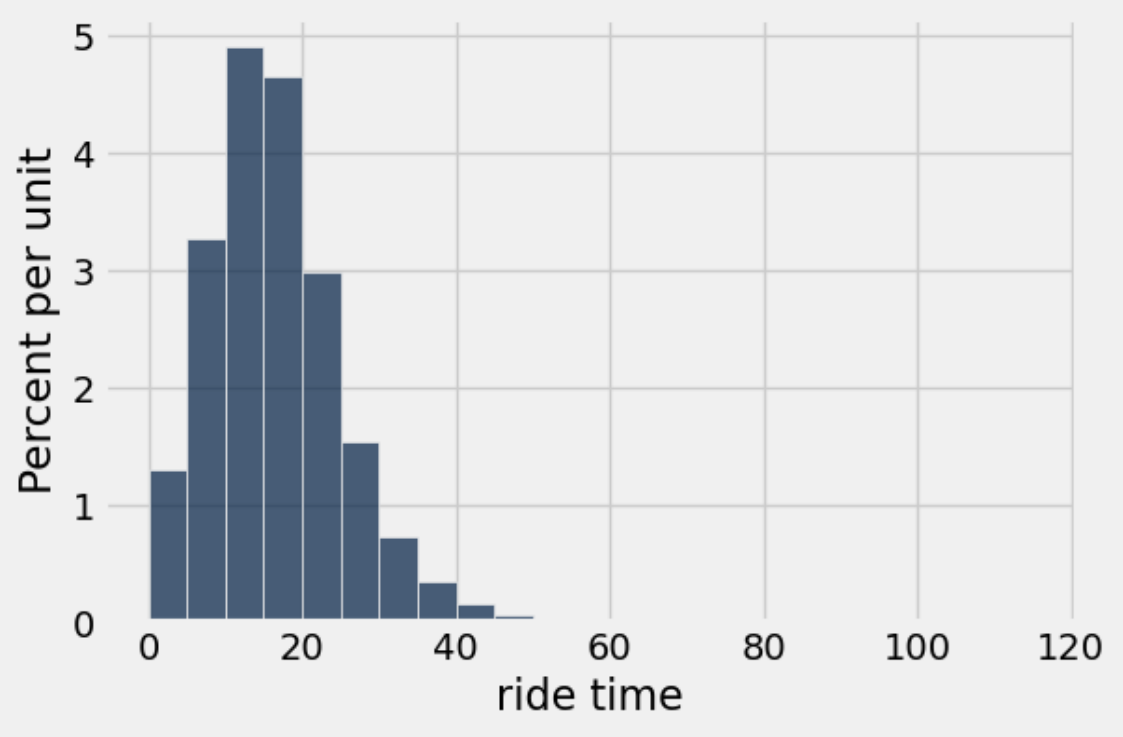
```
<IPython.core.display.HTML object>
```

```
Manila Table
```

```
<IPython.core.display.HTML object>
```

Question 1. Produce a histogram that visualizes the distributions of all ride times in Boston using the given bins in `equal_bins`.

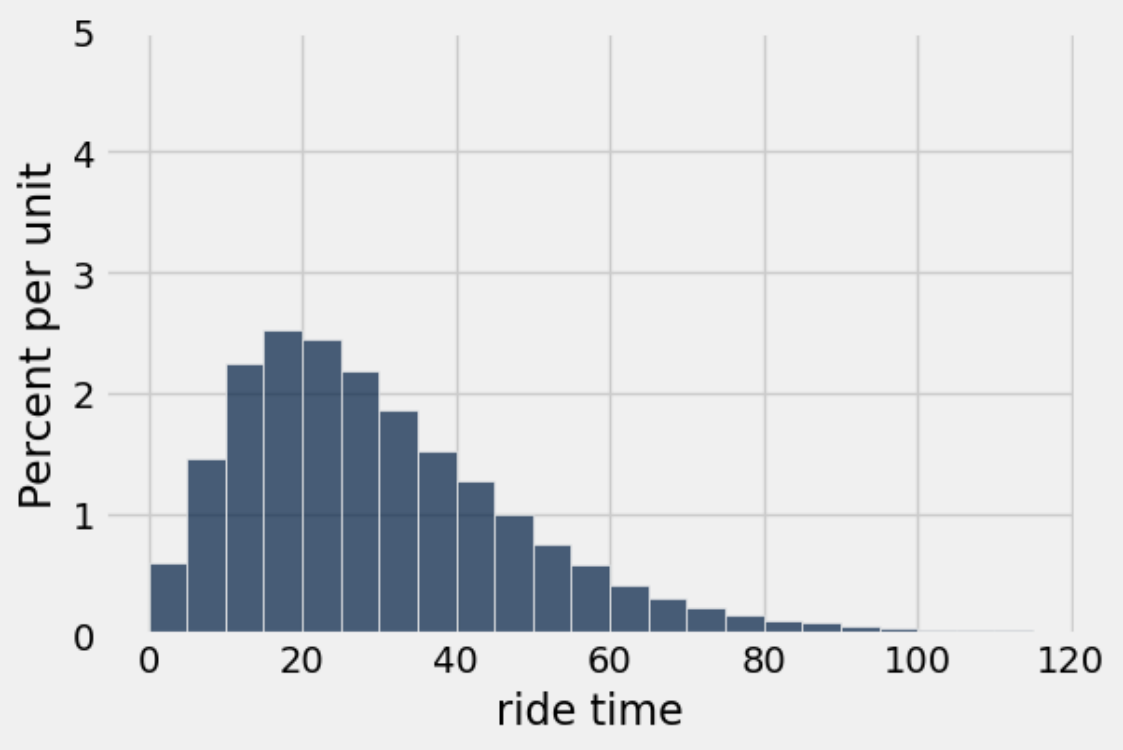
[188]:
```
boston.hist("ride time", bins = equal_bins)
```



Question 2. Now, produce a histogram that visualizes the distribution of all ride times in Manila using the given bins.

[189]:
```
equal_bins = np.arange(0, 120, 5)
manila.hist("ride time", bins = equal_bins)

# Don't delete the following line!
plots.ylim(0, 0.05);
```



Question 3. Let's take a closer look at the y-axis label. Assign `unit_meaning` to an integer (1, 2, 3) that corresponds to the "unit" in "Percent per unit".

1. minute
2. ride time
3. second

[190]:
```
unit_meaning = 2
unit_meaning
```

[190]:
```
2
```

[191]:
```
grader.check("q3_3")
```

[191]:
```
q3_3 results: All test cases passed!
```

Question 4. Assign `boston_under_10` and `manila_under_10` to the percentage of rides that are less than 10 minutes in their respective metropolitan areas. Use the height variables provided below in order to compute the percentages. Your solution should only use height variables, numbers, and mathematical operations. You should **not** access the tables `boston` and `manila` in any way.

[194]:
```
boston_under_5_height = 1.2
manila_under_5_height = 0.6
boston_5_to_under_10_height = 3.2
manila_5_to_under_10_height = 1.4

boston_under_10 = (boston_under_5_height + boston_5_to_under_10_height) * 5
#bins tell width
manila_under_10 = (manila_under_5_height + manila_5_to_under_10_height) * 5

boston_under_10, manila_under_10
```

[194]:
```
(22.0, 10.0)
```

[ ]:
```
grader.check("q3_4")
```

Question 5. Let's take a closer look at the distribution of ride times in Boston. Assign `boston_median_bin` to an integer (1, 2, 3, or 4) that corresponds to the bin that contains the median time.

1. 0-8 minutes
2. 8-14 minutes
3. 14-20 minutes
4. 20-40 minutes

Hint: The median of a sorted list has half of the list elements to its left, and half to its right.

[197]:
```
boston_median_bin = 3
boston_median_bin
```

[197]:
```
3
```

[198]:
```
grader.check("q3_5")
```

[198]:
```
q3_5 results: All test cases passed!
```

Question 6. Identify one difference between the histograms, in terms of the statistical properties. Can you comment on the average and/or skew of each histogram?

Hint: The best way to do this is to compare the two histograms (from 3.1 and 3.2) visually.

One difference between the histograms is the first histogram seems to have a peak at 15 and the second histogram seems to have a peak at 20. Another difference I can see between the histograms is the first one has a distribution for the ride time(y) that goes till 50, and the second one has a distribution for the ride time(y) that goes till 100. These are two right skewed histograms.

Question 7. Why is your solution in Question 6 the case? Based on one of the following two readings, why are the distributions for Boston and Manila different?

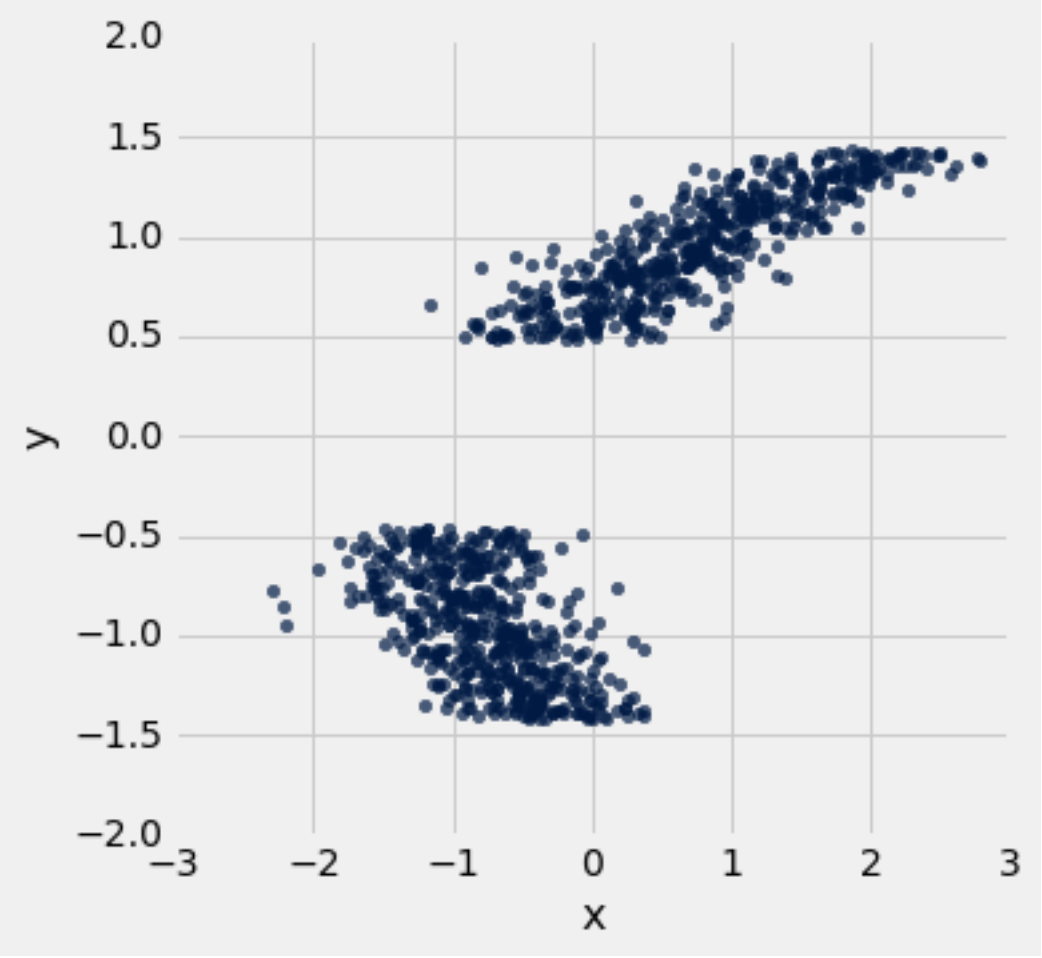
- Boston reading
- Manila reading

Hint: Try thinking about external factors of the two cities that may be causing the difference! There may be multiple different factors that come into play.

Type your answer here, replacing this text.

1.4 4. Histograms

Consider the following scatter plot:



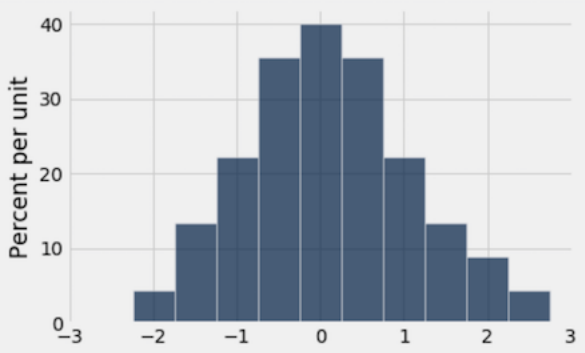
The axes of the plot represent values of two variables: x and y.

Suppose we have a table called `t` that has two columns in it:

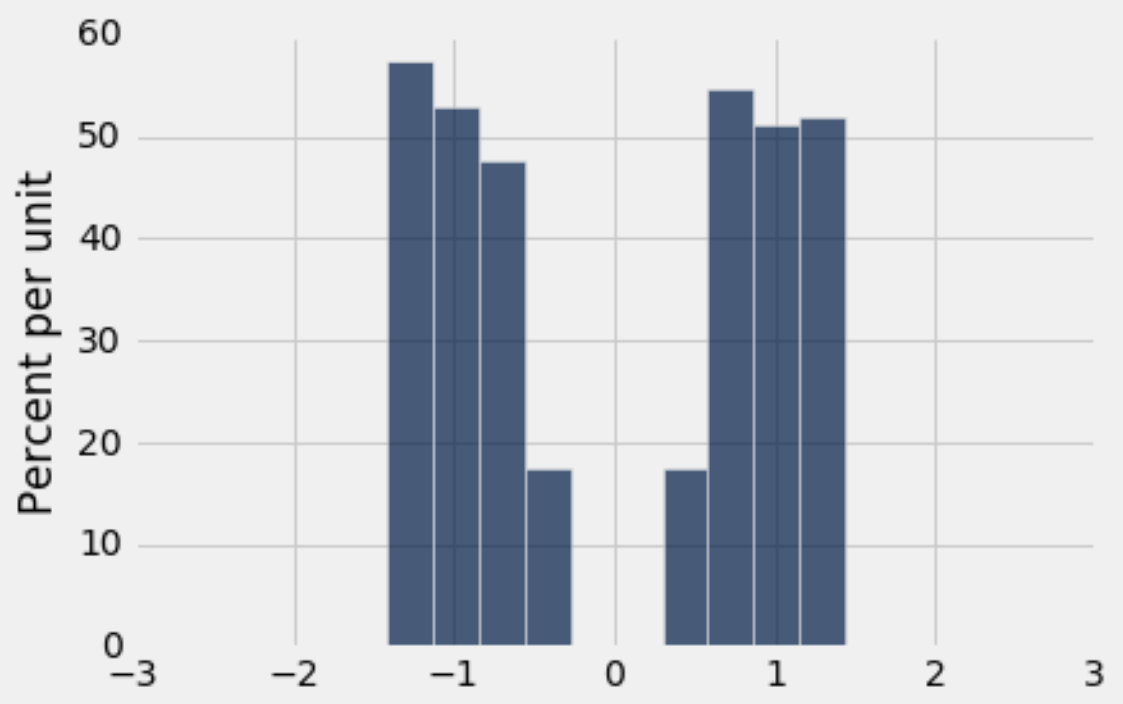
- `x`: a column containing the x-values of the points in the scatter plot
- `y`: a column containing the y-values of the points in the scatter plot

Below, you are given three histograms—one corresponds to column `x`, one corresponds to column `y`, and one does not correspond to either column.

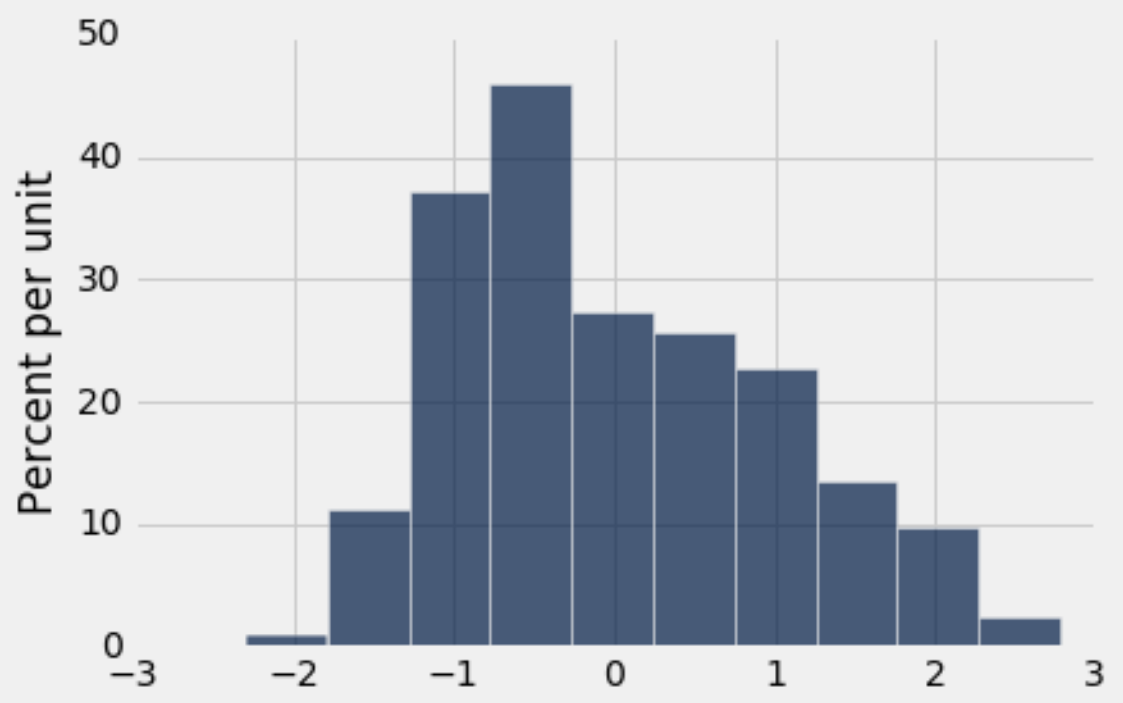
Histogram A:



Histogram B:



Histogram C:



Question 1. Suppose we run `t.hist('x')`. Which histogram does this code produce? Assign `histogram_column_x` to either 1, 2, or 3.

1. Histogram A
2. Histogram B
3. Histogram C

[ ]:
```
histogram_column_x = 3
```

[ ]:
```
grader.check("q4_1")
```

Question 2. State at least one reason why you chose the histogram from Question 1. **Make sure to clearly indicate which histogram you selected** (ex: "I chose histogram A because …").

I chose histogram C because when looking at the x position of the points in the scatter plot I saw that they had a more thicker distribution of points on the left compared to the right side of the graph. This matches with the skew of Histogram C where the bar is higher on the left side compared to the right side thus reflecting the scatter plot.

Question 3. Suppose we run `t.hist('y')`. Which histogram does this code produce? Assign `histogram_column_y` to either 1, 2, or 3.

1. Histogram A
2. Histogram B
3. Histogram C

[ ]:
```
histogram_column_y = 2
```

[ ]:
```
grader.check("q4_3")
```

Question 4. State at least one reason why you chose the histogram from Question 3. **Make sure to clearly indicate which histogram you selected** (ex: "I chose histogram A because …").

I chose histogram B because when looking at they y position of the points in the scatter plot I saw that the points had a equal distribution in terms of length on either side from -0.5 to -1.5 and 0.5 to 1.5 this leaving a space in the middle from 0. This reflects how the bars are in histogram B with the distribution being equal in terms of length from both sides as well as also leaving space in the middle from 0.

1.5 Congratulations! You're done with Homework 3!

Be sure to run all of the cells and the grader checks and verify that they all pass, then choose **Download as PDF via LaTeX** from the **File** menu, as well as **Download as Notebook** from the **File** menu, correctly name your files, and submit the two files on **canvas**.