hw03-solution

February 28, 2020

1 Homework 3: Tables and Charts

Reading: Textbook chapters 5 and 6.

Please complete this notebook by filling in the cells provided. Before you begin, execute the following cell to load the provided tests. Each time you start your server, you will need to execute this cell again to load the tests.

[2]:

```
# Don't change this cell; just run it.

import numpy as np
from datascience import *

%matplotlib inline
import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')

from client.api.notebook import Notebook
ok = Notebook('hw03.ok')
_ = ok.auth(inline=True)
```

```
=====================================================================
Assignment: Homework 3: Tables and Charts
OK, version v1.14.20
=====================================================================

Successfully logged in as m.zareei@ieee.org
```

Important: The `ok` tests don't usually tell you that your answer is correct. More often, they help catch careless mistakes. It's up to you to ensure that your answer is correct. If you're not sure, ask someone (not for the answer, but for some guidance about your approach).

Once you're finished, select "Save and Checkpoint" in the File menu and then execute the `submit` cell below. Check your OkPy account to see if your backup has saved successfully. Don't worry about the submission part - OkPy automatically converts your most recent backup to your final submission. Just make sure that's the one you want to submit!

[ ]:

```
_ = ok.submit()
```

1.1 1. Unemployment

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`.

```
[3]: unemployment = Table().read_table("unemployment.csv") #SOLUTION
     unemployment
```

```
[3]: 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)
```

```
[4]: _ = ok.grade('q1_1')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

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

[5]:

```
by_nei = unemployment.sort("NEI", descending=True) #SOLUTION
by_nei_pter = unemployment.sort("NEI-PTER", descending=True) #SOLUTION
```

[6]:

```
_ = ok.grade('q1_2')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 3. Use `take` to make a table containing the data for the 10 quarters when NEI was greatest. Call that table `greatest_nei`.

[7]:

```
greatest_nei = by_nei.take(np.arange(10)) #SOLUTION
greatest_nei
```

[7]:

```
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
```

[8]:

```
_ = ok.grade('q1_3')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% 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 (and counted 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.

```
[9]: pter = unemployment.column("NEI-PTER") - unemployment.column("NEI") #SOLUTION
     pter
```

```
[9]: 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])
```

```
[10]: _ = ok.grade('q1_4')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

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

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

```
[11]: by_pter = unemployment.with_column("PTER", pter).sort("PTER", descending=True)␣
       ↪#SOLUTION
      by_pter
```

```
[11]: 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)
```

[12]:

```python
_ = ok.grade('q1_5')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 6. Does it seem true that the PTER rate was very high during the Great Recession, compared to other periods in the dataset? Justify your answer by referring to specific values in the table or by generating a chart.

Write your answer here, replacing this text.

1.2 2. Birth Rates

The following table gives census-based population estimates for each state on 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 & the District of Columbia.*

[13]:

```python
# Don't change this cell; just run it.
# From http://www2.census.gov/programs-surveys/popest/datasets/2010-2016/national/totals/nst-est2016-alldata.csv
# See http://www2.census.gov/programs-surveys/popest/datasets/2010-2015/national/totals/nst-est2015-alldata.pdf
#     for column descriptions. (As of Feb 2017, no descriptions were posted for 2010-2016.)
pop = Table.read_table('nst-est2016-alldata.csv').where('SUMLEV', 40).select([1, 4, 12, 13, 27, 34, 62, 69])
pop = pop.relabeled(2, '2015').relabeled(3, '2016')
pop = pop.relabeled(4, 'BIRTHS').relabeled(5, 'DEATHS')
pop = pop.relabeled(6, 'MIGRATION').relabeled(7, 'OTHER')
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 number of births in that period as a proportion of the population at the start of the period.

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

```
[14]: 0.012358536498646102
```

```
[15]: _ = ok.grade('q2_1')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 2. Assign `fastest_growth` to an array of the names of the five states with the fastest population growth rates in *descending order of growth rate*.

```
[17]: growth_rate = pop.column('2016')/pop.column('2015') # SOLUTION
      fastest_growth = pop.with_column('growth', growth_rate).sort('growth',␣
       ↪descending=True).take(np.arange(5)).column('NAME') # SOLUTION
      fastest_growth
```

```
[17]: array(['Utah', 'Nevada', 'Idaho', 'Florida', 'Washington'], dtype='<U20')
```

```
[18]: _ = ok.grade('q2_2')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 3. Assign `movers` to the number of states for which the absolute 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 at the start of the period. The `MIGRATION` column contains estimated annual net migration counts by state.

```
[19]: movers = pop.with_column('R', np.abs(pop.column(6)/pop.column(2))).where('R',␣
       ↪are.above(0.01)).num_rows # SOLUTION
      movers
```

```
[19]: 9
```

```
[20]: _ = ok.grade('q2_3')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

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

```
[23]: west_births = sum(pop.where('REGION', '4').column('BIRTHS')) # SOLUTION
      west_births
```

```
[23]: 979657
```

```
[24]: _ = ok.grade('q2_4')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 5. Assign `less_than_west_births` to the number of states that had a total population in 2016 that was smaller than the *number of babies born in region 4 (the Western US)* during this time interval.

```
[25]: less_than_west_births = pop.where('2016', are.below(west_births)).num_rows #␣
       ↪SOLUTION
      less_than_west_births
```

```
[25]: 7
```

```
[26]: _ = ok.grade('q2_5')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

----------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

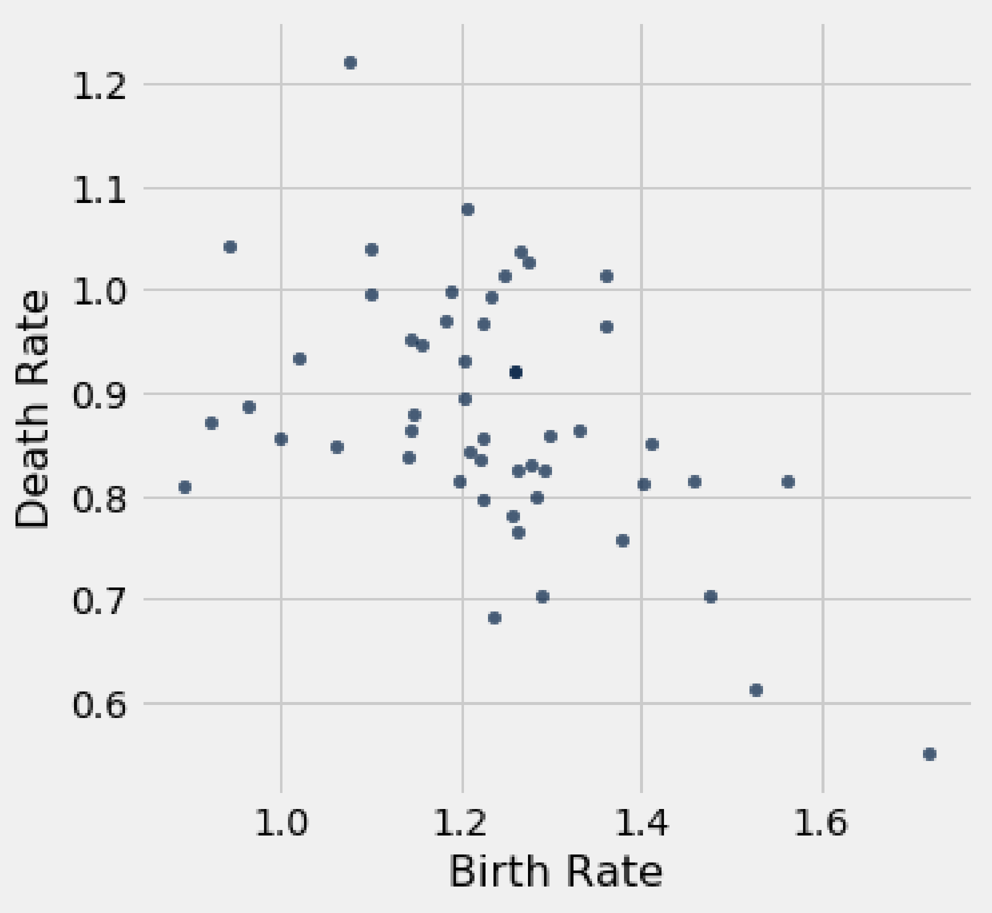
Question 6. Was there an association between birth rate and death rate during this time interval? Use the code cell below to support your conclusion with a chart. If an association exists, what might explain it?

Write your answer here, replacing this text.

SOLUTION: Yes, there is an association because the scatter plot below slopes down. Higher birth rates are associated with lower death rates. A possible explanation is that different states have different age distributions. A quick Internet search confirms that the proportion of seniors in each state varies quite a bit: http://www.worldatlas.com/articles/the-us-states-with-the-oldest-population.html (http://www.worldatlas.com/articles/the-us-states-with-the-oldest-population.html)

[27]:

```
# Generate a chart here to support your conclusion
pop.with_columns("Birth Rate",
100*pop.column('BIRTHS')/pop.column('2015'),"Death Rate",
100*pop.column('DEATHS')/pop.column('2015')).scatter(8, 9) # SOLUTION
```



1.3 3. Consumer Financial Protection Bureau Complaints

The Consumer Financial Protection Bureau has collected and published consumer complaints against financial companies since 2011. The data are available here (or at this direct link). For this exercise, to make your code run faster, we've selected only the data from May 2016.

Run the next cell to load the data. Each row represents one consumer's complaint.

[28]:
```
# Just run this cell.
complaints = Table.read_table("complaints.csv")
complaints
```

[28]:
```
company                                          | company_public_response
| company_response                     | complaint_id | complaint_what_happened
| consumer_consent_provided | consumer_disputed | date_received            |
date_sent_to_company    | issue                                     | product
| state | sub_issue                                 | sub_product
| submitted_via | tags             | timely | zip_code
```

```
TransUnion Intermediate Holdings, Inc. | Company has responded to the consumer
and the CFPB and c … | Closed with explanation         | 1920073      | (None)
| (None)                    | Yes               | 2016-05-11T15:39:07.000 |
2016-05-11T15:39:07.000 | Credit reporting company's investigation | Credit
reporting | VT    | Inadequate help over the phone        | (None)
| Phone         | (None)         | Yes    | 05035
TransUnion Intermediate Holdings, Inc. | Company has responded to the consumer
and the CFPB and c … | Closed with explanation         | 1914777      | (None)
| Consent not provided      | No                | 2016-05-08T00:53:47.000 |
2016-05-12T18:40:34.000 | Incorrect information on credit report   | Credit
reporting | MO    | Information is not mine               | (None)
| Web           | (None)         | Yes    | 63020
Bank of America                        | Company has responded to the consumer
and the CFPB and c … | Closed with explanation         | 1907306      | I
became aware of several charges on a Bank of America c … | Consent provided
| No                | 2016-05-03T16:49:33.000 | 2016-05-03T16:49:34.000 | Other
| Credit card      | VA    | (None)                                | (None)
| Web           | (None)         | Yes    | 239XX
Finance of America Reverse LLC         | Company believes it acted appropriately
as authorized by … | Closed with explanation         | 1919055      | I
applied for a reverse mortgage and everthing was going … | Consent provided
| No                | 2016-05-10T20:13:22.000 | 2016-05-10T20:13:23.000 |
Application, originator, mortgage broker | Mortgage         | TX    | (None)
| Reverse mortgage                      | Web           | Older American | Yes
| 774XX
Acceptance Solutions Group, INC        | Company believes it acted appropriately
as authorized by … | Closed with explanation         | 1908628      | Keeps
calling numbers that are not mine. And talking to  … | Consent provided
| No                | 2016-05-03T21:05:42.000 | 2016-05-06T13:42:45.000 |
Improper contact or sharing of info      | Debt collection  | OH    | Talked to
a third party about my debt | Payday loan                            | Web
| (None)         | Yes    | 430XX
Equifax                                | (None)
| Closed with explanation         | 1909176      | (None)
| (None)                    | No                | 2016-05-04T20:08:06.000 |
2016-05-09T15:11:00.000 | Incorrect information on credit report   | Credit
reporting | NC    | Information is not mine               | (None)
| Postal mail   | (None)         | Yes    | 28052
TransUnion Intermediate Holdings, Inc. | Company has responded to the consumer
and the CFPB and c … | Closed with explanation         | 1914477      | When I
enter my personal information to receive my credi … | Consent provided
| No                | 2016-05-06T23:09:50.000 | 2016-05-08T22:40:19.000 | Unable
to get credit report/credit score | Credit reporting | OH    | Problem getting
my free annual report | (None)                                 | Web           |
(None)         | Yes    | 450XX
Encore Capital Group                   | (None)
| Closed with non-monetary relief | 1919937      | (None)
| Consent not provided      | (None)            | 2016-05-11T18:58:25.000 |
2016-05-11T21:53:54.000 | Cont'd attempts collect debt not owed     | Debt
collection  | CT    | Debt is not mine                          | Credit card
| Web            | Older American | Yes    | 06801
Nationstar Mortgage                        | (None)
| Closed with explanation          | 1920517      | I am livid with Nation Star
for refusing to work with me … | Consent provided          | (None)
| 2016-05-11T20:38:09.000 | 2016-05-11T20:38:09.000 | Application, originator,
mortgage broker | Mortgage          | IL    | (None)
| Conventional adjustable mortgage (ARM) | Web            | (None)          | Yes
| 606XX
Convergent Resources, Inc.                 | (None)
| Closed with explanation          | 1920464      | (None)
| Consent not provided      | No                | 2016-05-11T12:16:31.000 |
2016-05-11T12:16:32.000 | Cont'd attempts collect debt not owed     | Debt
collection  | TX    | Debt is not mine                          | Other (i.e. phone,
health club, etc.)  | Web            | (None)          | Yes    | 78109
… (15021 rows omitted)
```

Question 1. Financial companies offer a variety of products. How many complaints were made against each kind of product? Make a table called `complaints_per_product` with one row per product category and 2 columns: ”product” (the name of the product) and ”number of complaints” (the number of complaints made against that kind of product).

[29]:

```
complaints_per_product = complaints.group("product").relabeled("count", "number␣
↪of complaints") #SOLUTION
complaints_per_product
```

[29]:

```
product                 | number of complaints
Bank account or service | 1687
Consumer Loan           | 775
Credit card             | 1566
Credit reporting        | 3820
Debt collection         | 3022
Money transfers         | 142
Mortgage                | 3468
Other financial service | 16
Payday loan             | 119
Prepaid card            | 110
… (1 rows omitted)
```

[30]:

```
_ = ok.grade('q3_1')
```

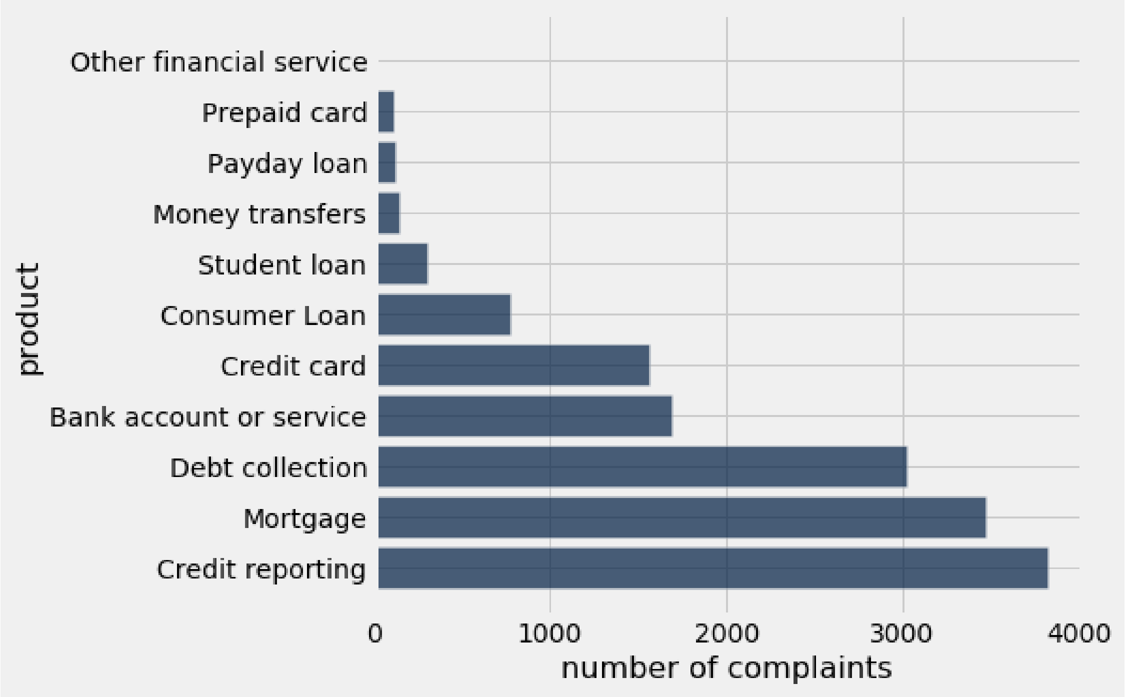
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
```

```
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 2. Make a bar chart showing how many complaints were made about each product category. Sort the bars from shortest to longest.

```
[31]: complaints_per_product.sort(1).barh('product') #SOLUTION
```



Question 3. Make a table of the number of complaints made against each *company*. Call it `complaints_per_company`. It should have one row per company and 2 columns: "company" (the name of the company) and "number of complaints" (the number of complaints made against that company).

```
[32]: complaints_per_company = complaints.group("company").relabeled("count", "number␣
      ↪of complaints") #SOLUTION
      complaints_per_company
```

```
[32]: company                     | number of complaints
      1st Preference Mortgage     | 2
      21st Mortgage Corporation   | 7
      2288984 Ontario Inc.        | 3
      360 Mortgage                | 1
      3rd Generation, Inc.        | 1
      4M Collections, LLC         | 1
```

```
A.R.M. Solutions, Inc.     | 2
AC Autopay, LLC            | 1
ACE Cash Express Inc.      | 21
ACS Education Services     | 8
… (1131 rows omitted)
```

[33]:

```
_ = ok.grade('q3_3')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Running tests

---------------------------------------------------------------------
Test summary
    Passed: 1
    Failed: 0
[ooooooooook] 100.0% passed
```

Question 4. It wouldn't be a good idea to make a bar chart of that data. (Don't try it!) Why not?

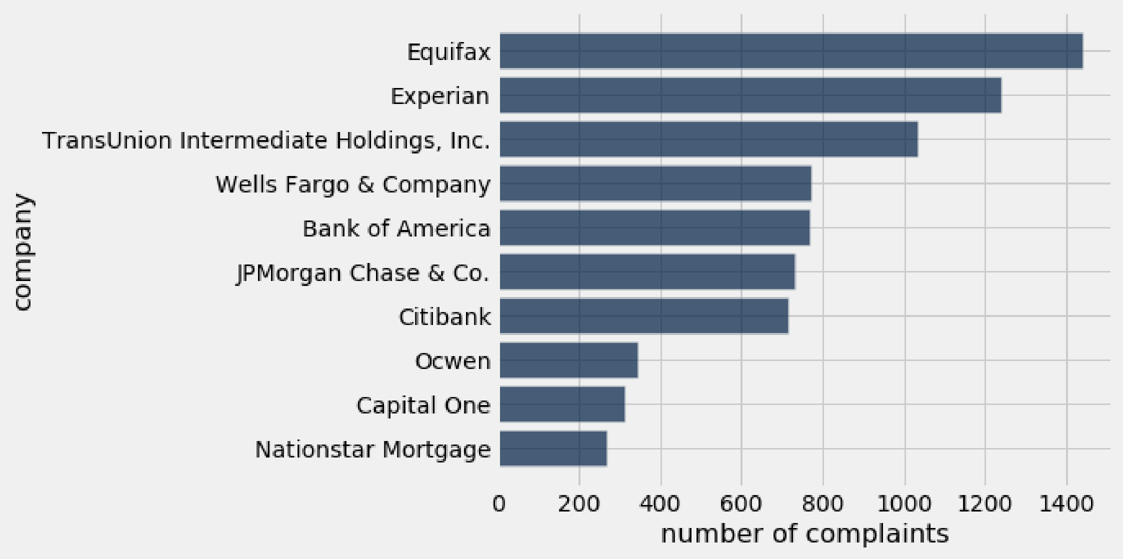
Write your answer here, replacing this text.

SOLUTION: There are thousands of companies, most with only a few complaints. A bar chart with 1 bar per company would be hard to read, and it even takes a long time for Python to generate it.

Question 5. Make a bar chart of just the 10 companies with the most complaints.

[34]:

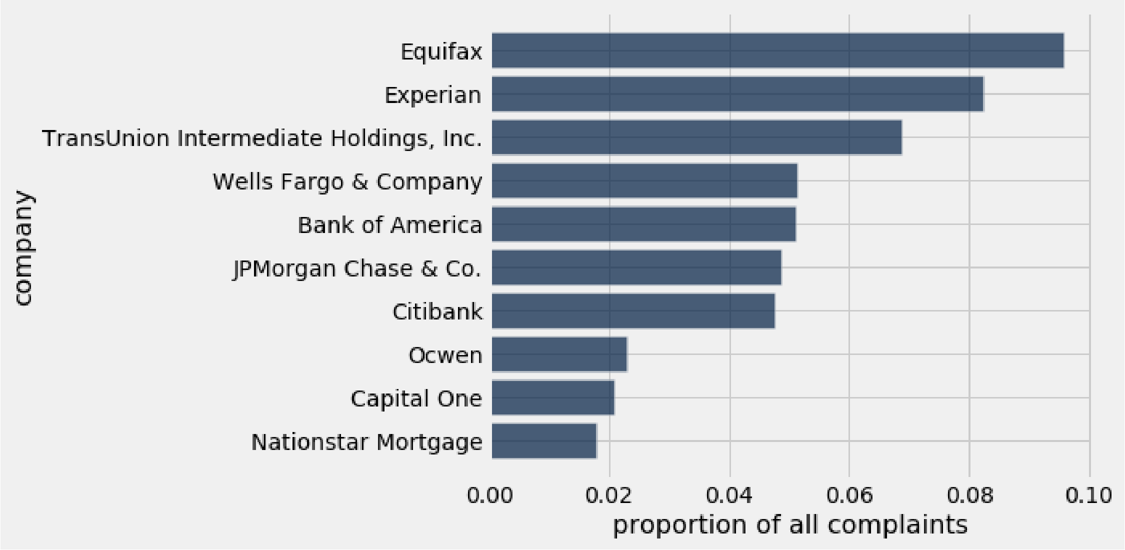
```
complaints_per_company.sort("number of complaints", descending=True).take(np.
 ↪arange(10)).barh("company") #SOLUTION
```



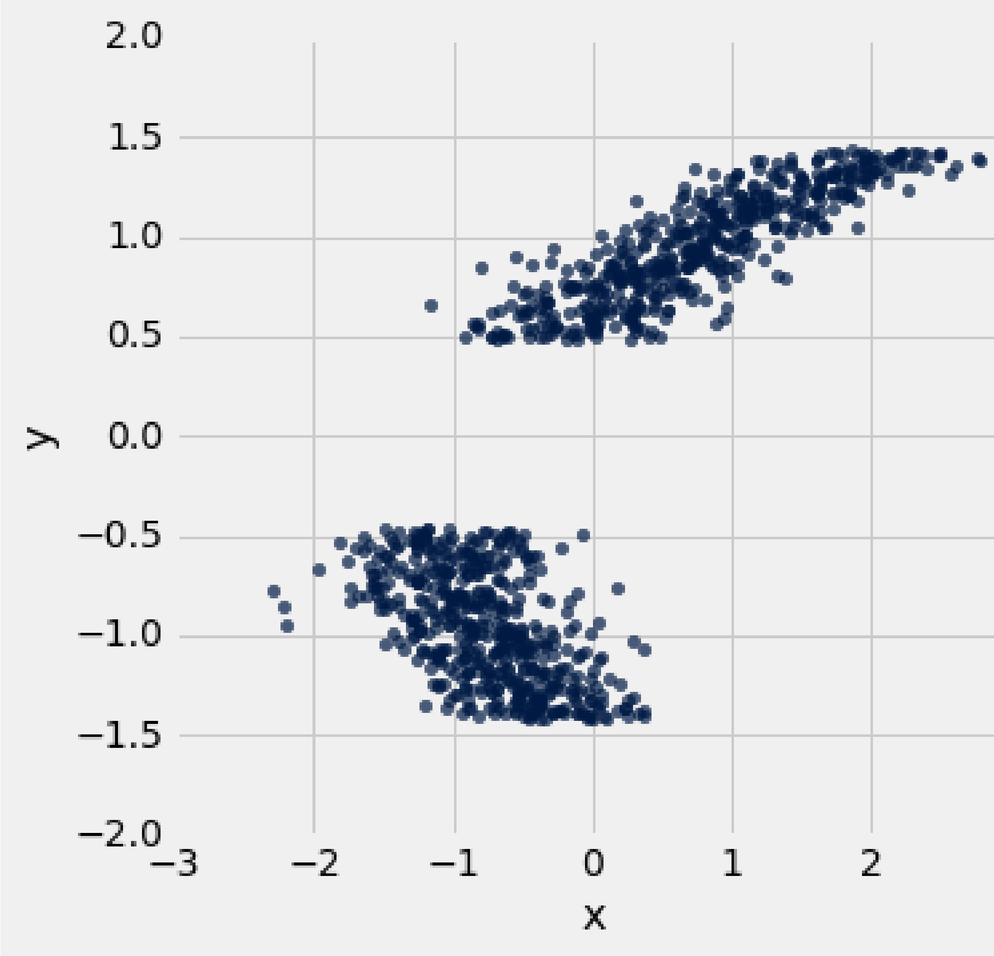
Question 6. Make a bar chart like the one above, with one difference: The size of each company's bar should be the *proportion* (among *all complaints* made against any company in `complaints`) that were made against that company.

Note: Graphs aren't very useful without accurate labels. Make sure that the text on the horizontal axis of the graph makes sense.

```
[35]: complaints_per_company.with_column("proportion of all complaints",␣
       ↪complaints_per_company.column("number of complaints") / complaints.num_rows)\
      .sort("proportion of all complaints", descending=True)\
      .drop("number of complaints")\
      .take(np.arange(10))\
      .barh("company") #SOLUTION
```



1.4 4. Marginal Histograms



Consider the following scatter plot:

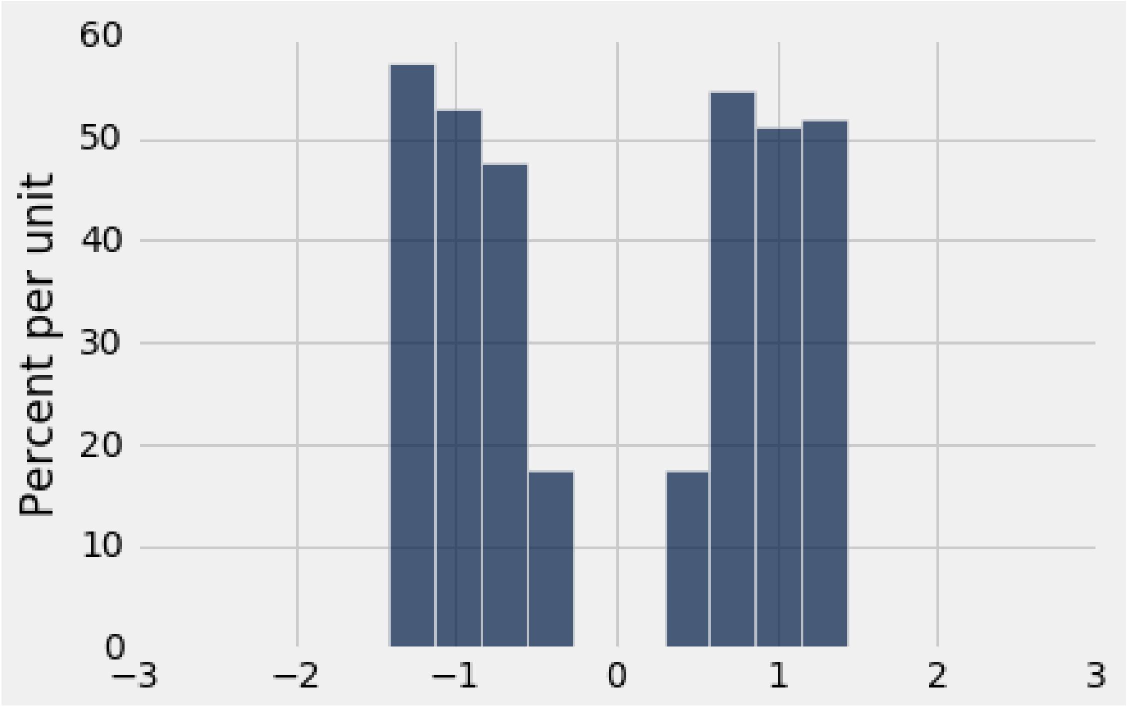
The axes of the plot represent values of two variables: x and y.

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

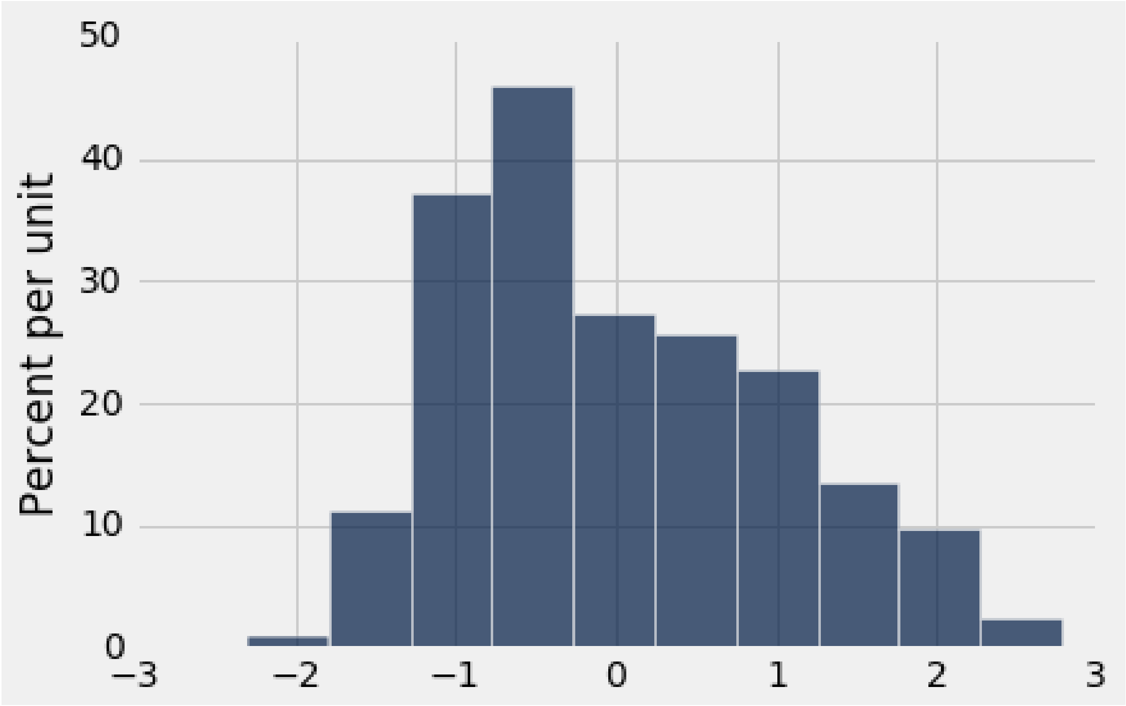
- `x`: a column containing the x-values of the points in the scatter plot
- `y`: a column containing the y-values of the points in the scatter plot

Question 1: Match each of the following histograms to the code that produced them. Explain your reasoning.

Histogram A:



Histogram B:



Line 1: `t.hist('x')`

Histogram for Line 1: Histogram B #SOLUTION

Explanation: Because there are no gaps in the X-variable, we would expect the histogram for X to have no gaps in it. Also, because the two masses overlap at the left side of the plot, we would expect there to be more mass on the left end of the histogram, since each vertical slice at the lower end of the range contains more points. Also, the values of the X-variable range from -2 to 2, which fits the range of values in histogram B. #SOLUTION

Line 2: `t.hist('y')`

Histogram for Line 2: Histogram A #SOLUTION

Explanation: There is a gap in the points in the Y-direction, so we would expect a gap in the histogram of those values. Also, the range of values covered by the Y-variable range from -1.5 to 1.5, which fits the range of values in histogram A. #SOLUTION

[ ]:

```

```