hw04

March 22, 2022

Name: Allan Gongora

Section: 0131

1 Homework 4: Functions, Histograms, and Groups

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

[1]:
```
pip install gofer-grader
```

```
Requirement already satisfied: gofer-grader in /opt/conda/lib/python3.7/site-
packages (1.1.0)
Requirement already satisfied: tornado in /opt/conda/lib/python3.7/site-packages
(from gofer-grader) (6.1)
Requirement already satisfied: pygments in /opt/conda/lib/python3.7/site-
packages (from gofer-grader) (2.11.2)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.7/site-packages
(from gofer-grader) (3.0.3)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/conda/lib/python3.7/site-
packages (from jinja2->gofer-grader) (2.0.1)
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
# Don't change this cell; just run it.

import numpy as np
from datascience import *

# These lines do some fancy plotting magic.\n",
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')

# These lines load the tests.

from gofer.ok import check
```

Recommended Reading:

- Visualizing Numerical Distributions
- Functions and Tables

1) For all problems that you must write explanations and sentences for, you **must** provide your answer in the designated space. This can include:
   A) Sentence reponses to questions that ask for an explanation
   B) Numeric responses to multiple choice questions
   C) Programming code
2) 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!

Once you're finished, select "Save and Checkpoint" in the File menu. Your name and course section number should be in the first and last cell of the assignment. Be sure you have run all cells with code and that the output from that is showing. Then click "Print Preview" in the File menu. Print a copy from there in pdf format. (This means you right click and choose print and choose "save as pdf" from your printer options.) You will need to submit the pdf in Canvas by the deadline.

The gopher grader output and/or output from your coding are essential to helping your instructor grade your work correctly and in a timely manner.

Files submitted that are missing the required output will lose some to all points so double check your pdf before submitting.

1.1 1. Working with Text Using Functions

The following table contains the words from four chapters of Charles Dickens' *A Tale of Two Cities*. We're going to compute some simple facts about each chapter. Since we're performing the same computation on each chapter, it's best to encapsulate each computational procedure in a function, and then call the function several times. Run the cell to get a table with one column.

[3]:
```
# Just run this cell to load the data.
tale_chapters = Table.read_table("tale.csv")
tale_chapters
```

[3]:
```
Chapter text
I. The Period


It was the best of times,
it was the wor …
II. The Mail


It was the Dover road that lay, on a Frid …
III. The Night Shadows


A wonderful fact to reflect upo …
```

IV. The Preparation

When the mail got successfully to …

[19]:
```
tale_chapters["Chapter text"][0]
```

[19]:
```
'I. The Period\n\n\nIt was the best of times,\nit was the worst of times,\nit
was the age of wisdom,\nit was the age of foolishness,\nit was the epoch of
belief,\nit was the epoch of incredulity,\nit was the season of Light,\nit was
the season of Darkness,\nit was the spring of hope,\nit was the winter of
despair,\nwe had everything before us,\nwe had nothing before us,\nwe were all
going direct to Heaven,\nwe were all going direct the other way--\nin short, the
period was so far like the present period, that some of\nits noisiest
authorities insisted on its being received, for good or for\nevil, in the
superlative degree of comparison only.\n\nThere were a king with a large jaw and
a queen with a plain face, on the\nthrone of England; there were a king with a
large jaw and a queen with\na fair face, on the throne of France. In both
countries it was clearer\nthan crystal to the lords of the State preserves of
loaves and fishes,\nthat things in general were settled for ever.\n\nIt was the
year of Our Lord one thousand seven hundred and seventy-five.\nSpiritual
revelations were conceded to England at that favoured period,\nas at this. Mrs.
Southcott had recently attained her five-and-twentieth\nblessed birthday, of
whom a prophetic private in the Life Guards had\nheralded the sublime appearance
by announcing that arrangements were\nmade for the swallowing up of London and
Westminster. Even the Cock-lane\nghost had been laid only a round dozen of
years, after rapping out its\nmessages, as the spirits of this very year last
past (supernaturally\ndeficient in originality) rapped out theirs. Mere messages
in the\nearthly order of events had lately come to the English Crown and
People,\nfrom a congress of British subjects in America: which, strange\nto
relate, have proved more important to the human race than any\ncommunications
yet received through any of the chickens of the Cock-lane\nbrood.\n\nFrance,
less favoured on the whole as to matters spiritual than her\nsister of the
shield and trident, rolled with exceeding smoothness down\nhill, making paper
money and spending it. Under the guidance of her\nChristian pastors, she
entertained herself, besides, with such humane\nachievements as sentencing a
youth to have his hands cut off, his tongue\ntorn out with pincers, and his body
burned alive, because he had not\nkneeled down in the rain to do honour to a
dirty procession of monks\nwhich passed within his view, at a distance of some
fifty or sixty\nyards. It is likely enough that, rooted in the woods of France
and\nNorway, there were growing trees, when that sufferer was put to
death,\nalready marked by the Woodman, Fate, to come down and be sawn
into\nboards, to make a certain movable framework with a sack and a knife
in\nit, terrible in history. It is likely enough that in the rough outhouses\nof
some tillers of the heavy lands adjacent to Paris, there were\nsheltered from
the weather that very day, rude carts, bespattered with\nrustic mire, snuffed
about by pigs, and roosted in by poultry, which\nthe Farmer, Death, had already
```

```
set apart to be his tumbrils of\nthe Revolution. But that Woodman and that
Farmer, though they work\nunceasingly, work silently, and no one heard them as
they went about\nwith muffled tread: the rather, forasmuch as to entertain any
suspicion\nthat they were awake, was to be atheistical and traitorous.\n\nIn
England, there was scarcely an amount of order and protection to\njustify much
national boasting. Daring burglaries by armed men, and\nhighway robberies, took
place in the capital itself every night;\nfamilies were publicly cautioned not
to go out of town without removing\ntheir furniture to upholsterers\' warehouses
for security; the highwayman\nin the dark was a City tradesman in the light,
and, being recognised and\nchallenged by his fellow-tradesman whom he stopped in
his character of\n"the Captain," gallantly shot him through the head and rode
away; the\nmail was waylaid by seven robbers, and the guard shot three dead,
and\nthen got shot dead himself by the other four, "in consequence of
the\nfailure of his ammunition:" after which the mail was robbed in peace;\nthat
magnificent potentate, the Lord Mayor of London, was made to stand\nand deliver
on Turnham Green, by one highwayman, who despoiled the\nillustrious creature in
sight of all his retinue; prisoners in London\ngaols fought battles with their
turnkeys, and the majesty of the law\nfired blunderbusses in among them, loaded
with rounds of shot and ball;\nthieves snipped off diamond crosses from the
necks of noble lords at\nCourt drawing-rooms; musketeers went into St. Giles\'s,
to search\nfor contraband goods, and the mob fired on the musketeers, and
the\nmusketeers fired on the mob, and nobody thought any of these
occurrences\nmuch out of the common way. In the midst of them, the hangman, ever
busy\nand ever worse than useless, was in constant requisition; now,
stringing\nup long rows of miscellaneous criminals; now, hanging a housebreaker
on\nSaturday who had been taken on Tuesday; now, burning people in the\nhand at
Newgate by the dozen, and now burning pamphlets at the door of\nWestminster
Hall; to-day, taking the life of an atrocious murderer,\nand to-morrow of a
wretched pilferer who had robbed a farmer\'s boy of\nsixpence.\n\nAll these
things, and a thousand like them, came to pass in and close\nupon the dear old
year one thousand seven hundred and seventy-five.\nEnvironed by them, while the
Woodman and the Farmer worked unheeded,\nthose two of the large jaws, and those
other two of the plain and the\nfair faces, trod with stir enough, and carried
their divine rights\nwith a high hand. Thus did the year one thousand seven
hundred\nand seventy-five conduct their Greatnesses, and myriads of
small\ncreatures--the creatures of this chronicle among the rest--along
the\nroads that lay before them.\n\n\n\n'
```

Question 1.1 Write a function called `word_count` that takes a single argument, the text of a single chapter, and returns the number of words in that chapter. Assume that words are separated from each other by spaces.

Hint: Try the string method `split` and the function `len`.

[4]:
```
def word_count(chpt_text: str) -> int:
    return len(chpt_text.split(" "))

word_count(tale_chapters.column("Chapter text").item(0))
```

[4]:
```
911
```

[5]:
```
check('tests/q1_1.py')
```

[5]:
```
<gofer.ok.OKTestsResult at 0x7f0fb83b3150>
```

Question 1.2 Create an array called `chapter_lengths` which contains the length of each chapter in `tale_chapters`.

Hint: Consider using `apply` along with the function you have defined in the previous question.

[6]:
```
chapter_lengths = tale_chapters.apply(word_count, "Chapter text")
chapter_lengths
```

[6]:
```
array([ 911, 1827, 1468, 3994])
```

[7]:
```
check('tests/q1_2.py')
```

[7]:
```
<gofer.ok.OKTestsResult at 0x7f0f70904c10>
```

Question 1.3 Write a function called `character_count`. It should take a string as its argument and return the number of characters in that string that aren't spaces (" "), periods ("."), exclamation marks ("¡'), or question marks ("?"). Remember that `tale_chapters` is a table, and that the function takes in only the text of one chapter as the input.

Hint: Try using the string method `replace` several times to remove the characters we don't want to count.

[20]:
```
def character_count(text: str) -> int:
    bad_chars = {" ", ".", "!", "?"}
    for i in bad_chars:
        text = text.replace(i, "")
    return len(text)
```

[21]:
```
check('tests/q1_3.py')
```

[21]:
```
<gofer.ok.OKTestsResult at 0x7f0f7062dc90>
```

Question 1.4 Write a function called `chapter_number`. It should take a single argument, the text of a chapter from our dataset, and return the number of that chapter, as a Roman numeral. (For example, it should return the string "I" for the first chapter and "II" for the second.) If the argument doesn't have a chapter number in the same place as the chapters in our dataset, `chapter_number` can return whatever you like.

To help you with this, we've included a function called `text_before`. Its documentation describes what it does.

[22]:
```
def text_before(full_text, pattern):
    """Finds all the text that occurs in full_text before the specified pattern.
```

```
    Parameters
    ----------
    full_text : str
        The text we want to search within.
    pattern : str
        The thing we want to search for.

    Returns
    -------
    str
        All the text that occurs in full_text before pattern.  If pattern
        doesn't appear anywhere, all of full_text is returned.

    Examples
    --------

    >>> text_before("The rain in Spain falls mainly on the plain.", "Spain")
    'The rain in '
    >>> text_before("The rain in Spain falls mainly on the plain.", "ain")
    'The r'
    >>> text_before("The rain in Spain falls mainly on the plain.", "Portugal")
    'The rain in Spain falls mainly on the plain.'
    """
    return np.array(full_text.split(pattern)).item(0)

def chapter_number(chapter_text):
    roman_numerals = {"I", "V", "X", "L", "C", "D", "M"}
    chpt_num = ""
    i = 0
    while chapter_text[i] in roman_numerals:
        chpt_num += chapter_text[i]
        i += 1
    return chpt_num if chpt_num != "" else None
```

[23]:
```
check('tests/q1_4.py')
```

[23]:
```
<gofer.ok.OKTestsResult at 0x7f0f6f6be890>
```

1.2 2. 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` table contains the length of the ride, in minutes.

[24]:
```
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
```

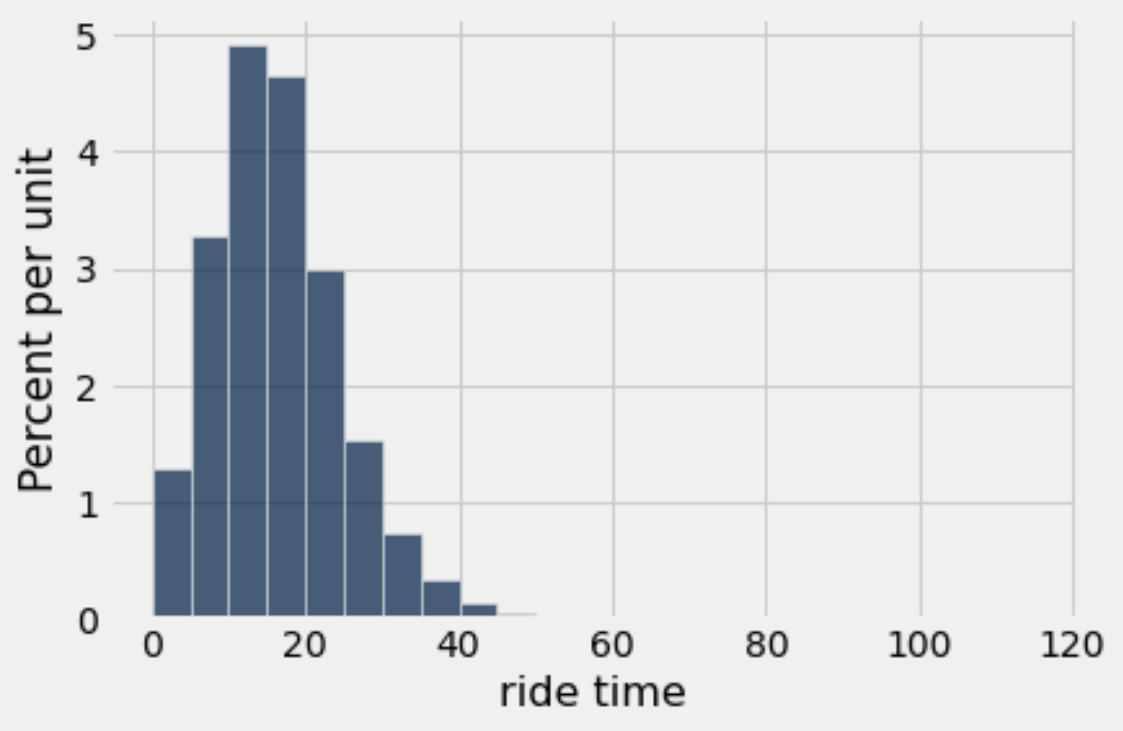
```
<IPython.core.display.HTML object>
```

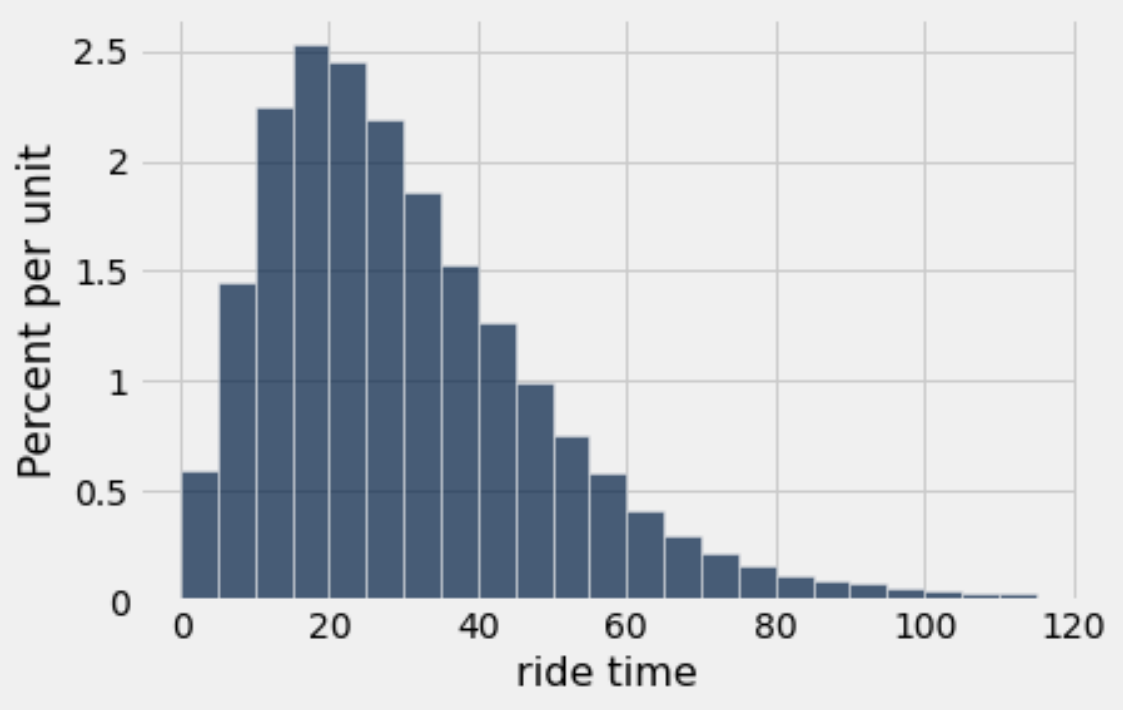
```
Manila Table
```

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

Question 2.1 Produce histograms of all ride times in Boston and in Manila, using the given bins. Please put the code for both of them in the following cell, putting the ride times for Boston first.

[25]:
```
bins = np.arange(0, 120, 5)
boston.hist("ride time", bins=bins)
manila.hist("ride time", bins=bins)
```





Question 2.2 Set the two variables below to estimates of what percentage of rides are less than 10 minutes in Boston and Manila. Find your estimates by visually assessing the histograms. Your solution should consist of only mathematical operations and numbers.

[26]:
```
boston_under_10 = 4.5
manila_under_10 = 2.0
```

Question 2.3 Comment on the main difference between the two histograms. What might be causing this?

Your Answer Here: Manila histogram has a longer right hand tail. Maybe because things are further, traffic is worse, or the infastructure isn't there.

The following two questions are optional!

Please do make an attempt at them, but they will not be incorporated into the final grading of this homework.

Optional Question 2.4 The `hod` column in each table represents the hour of the day during which the Uber was called (24-hour military time). 0 corresponds to 12-1 AM, 1 to 1-2 AM, 13 to 1-2 PM, etc. Write a function which takes in a table like `boston` or `manila`, and an `hod` number between 0 and 23, and displays a histogram of ride lengths from that hour in that city. Use the same bins as before.

[ ]:
```
def hist_for_time(tbl, hod):
    bins = np.arange(0, 120, 5)
    ...
```

```
#DO NOT DELETE THIS LINE!
hist_for_time(boston, 12)
```

Optional Question 2.5 Which city has a larger difference between Uber ride times at 10 AM vs. 10 PM? In other words, which is larger: the difference between 10 AM and 10 PM Uber ride times in Manila or the difference between 10 AM and 10 PM uber ride times in Boston. Use the function you just created to answer this question. You do not need to calculate an actual difference.

Assign `larger_diff` to the number 1 if the answer is Manila, and 2 if the answer is Boston.

[2]:
```
larger_diff = ...
```

[3]:
```
check('tests/q2_5.py')
```

1.3 3. Faculty Salaries

In the next cell, we load a dataset created by the Daily Cal which contains Berkeley faculty, their departments, their positions, and their gross salaries in 2015.

[27]:
```
raw_profs = Table.read_table("faculty.csv").where("year", are.equal_to(2015)).
 ↪drop("year", "title")
profs = raw_profs.relabeled("title_category", "position")
profs
```

[27]:
```
name            | department                       | position            |
gross_salary
CYNTHIA ABAN    | South & Southeast Asian Studies | lecturer            | 64450
PIETER ABBEEL   | Computer Science                 | associate professor | 184998
SALLY ABEL      | Law                              | lecturer            | 3466
ELIZABETH ABEL  | English                          | professor           | 138775
DOR ABRAHAMSON  | Education                        | associate professor | 100300
KATHRYN ABRAMS  | Law                              | professor           | 319693
BARBARA ABRAMS  | Public Health                    | professor           | 191162
SARAH ACCOMAZZO | Social Welfare                   | lecturer            | 14779
CHARISMA ACEY   | City and Regional Planning       | assistant professor | 101567
DAVID ACKERLY   | Biology                          | professor           | 182288
… (2049 rows omitted)
```

We want to use this table to generate arrays with the names of each professor in each department.

Question 3.1 Set `prof_names` to a table with two columns. The first column should be called "department" and have the name of every department once, and the second column should be called "faculty" and contain an *array* of the names of all faculty members in that department.

Hint: Think about how `group` works: it collects values into an array and then applies a function to that array. We have defined two functions below for you, and you will need to use one of them in your call to `group`.

[28]:
```
# Pick between the two functions defined below
def identity(array):
    return array

def first(array):
    return array.item(0)
```

[30]:
```
profs.where("department", "African American Studies").column("name")
```

[30]:
```
array(['AYA DE LEON', 'CHIYUMA ELLIOTT', 'NIKKI JONES', 'DAVID KYEU',
       'MICHEL LAGUERRE', 'JOVAN LEWIS', 'SAM MCHOMBO', 'APARAJITA NANDA',
       'G NWOKEJI', 'TIANNA PASCHEL', 'LEIGH RAIFORD', 'DARIECK SCOTT',
       'STEPHEN SMALL', 'ULA TAYLOR', 'JAMES TAYLOR'], dtype='<U25')
```

[42]:
```
prof_names = profs.drop("position", "gross_salary").group("department",␣
 ↪identity).relabel("name identity", "faculty")
prof_names
```

[42]:
```
department                                          | faculty
African American Studies                            | ['AYA DE LEON' 'CHIYUMA
ELLIOTT' 'NIKKI JONES' 'DAVID KY …
Agricultural and Resource Economics and Policy | ['MAXIMILIAN AUFFHAMMER'
'CHARLES GIBBONS' 'JEFFREY PERL …
Anthroplogy                                         | ['SABRINA AGARWAL' 'STANLEY
BRANDES' 'CHARLES BRIGGS'
 ' …
Architecture                                        | ['MARK ANDERSON' 'JACOB
ATHERTON' 'WILLIAM ATWOOD' 'R.GA …
Art History                                         | ['DILIANA ANGELOVA' 'PATRICIA
BERGER' 'JULIA BRYAN-WILSO …
Art Practice                                        | ['ALLAN DESOUZA' 'AIDA GAMEZ'
'RANDY HUSSONG' 'JENNIFER  …
Astronomy                                           | ['GIBOR BASRI' 'STEVEN
BECKWITH' 'LEO BLITZ' 'EUGENE CHI …
Bioengineering                                      | ['ADAM ARKIN' 'IRINA CONBOY'
'STEVEN CONOLLY' 'JOHN DUEB …
Biology                                             | ['DAVID ACKERLY' 'HILLEL
ADESNIK' 'KELLY AGNEW' 'DORIS B …
Buddhist Studies                                    | ['JANN RONIS']
… (61 rows omitted)
```

[36]:
```
check('tests/q3_1.py')
```

[36]:
```
<gofer.ok.OKTestsResult at 0x7f0f6f49a750>
```

Question 3.2 At the moment, the `name` column is sorted by last name. Would the arrays you generated in the previous part be the same if we had sorted by first name instead before generating

them? Two arrays are the **same** if they contain the same number of elements and the elements located at corresponding indexes in the two arrays are identical. Explain your answer. If you feel you need to make certain assumptions about the data, feel free to state them in your response.

Write your answer here, replacing this text. NO, they would be different because the order would change since we are sorting by a different key

Question 3.3 Set `biggest_range_dept` to the name of the department with the largest salary range, where range is defined as the **difference between the lowest and highest salaries in the department**.

Hint: First you'll need to define a new function `salary_range` which takes in an array of salaries and returns the salary range of the corresponding department. Then, set `department_ranges` to a table containing the names and salary ranges of each department.

[47]:
```
# Define salary_range in this cell
def salary_range(salaries: np.array) -> float:
    return max(salaries) - min(salaries)
```

[63]:
```
temp = profs.drop("name", "position").group("department", identity)
department_ranges = temp.with_column("salary_range", temp.apply(salary_range,␣
 ↪"gross_salary identity"))
biggest_range_dept = str(department_ranges.sort("salary_range",␣
 ↪descending=True)["department"][0])
biggest_range_dept
```

[63]:
```
'Economics'
```

[64]:
```
check('tests/q3_3.py')
```

[64]:
```
<gofer.ok.OKTestsResult at 0x7f0f6f43acd0>
```

1.4 4. Submission

Once you're finished, select "Save and Checkpoint" in the File menu. Your name and course section number should be in the first and last cell of the assignment. Be sure you have run all cells with code and that the output from that is showing.

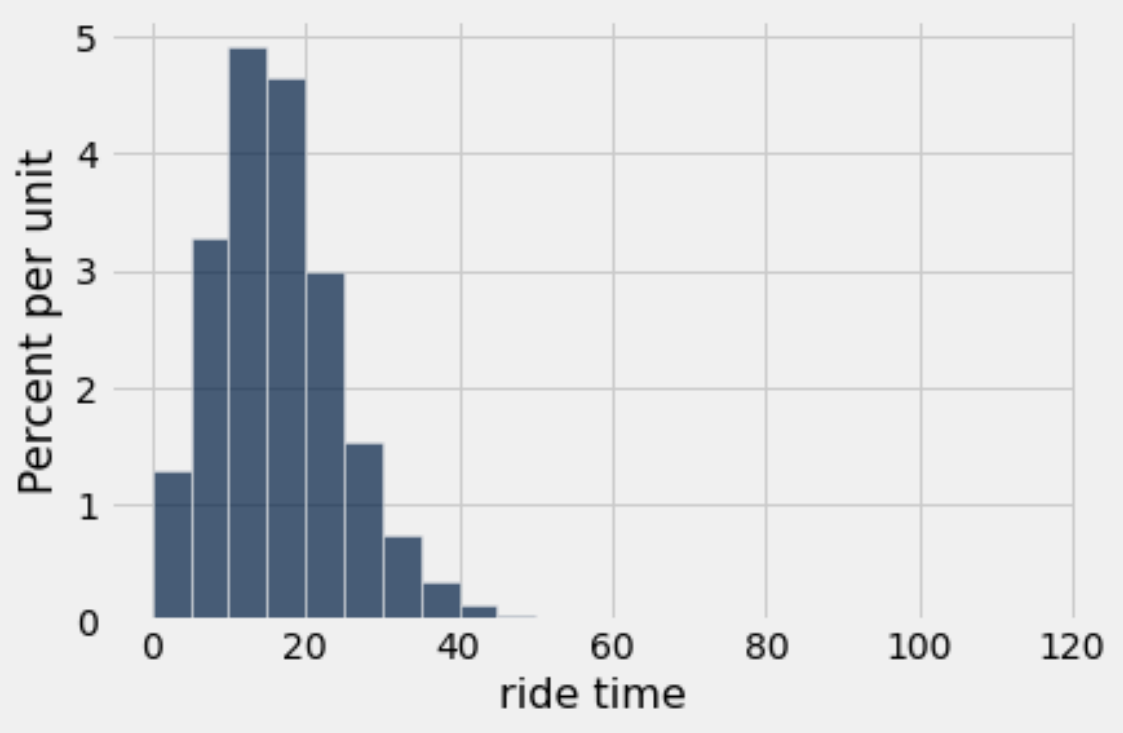
Double check that you have completed all of the free response questions as the auto-grader does NOT check that and YOU are responsible for knowing those questions are there and completing them as part of the grade for this homework. When ready, click "Print Preview" in the File menu. Print a copy from there in pdf format. (This means you right click and choose print and choose "save as pdf" from your printer options.) You will need to submit the pdf in Canvas by the deadline.

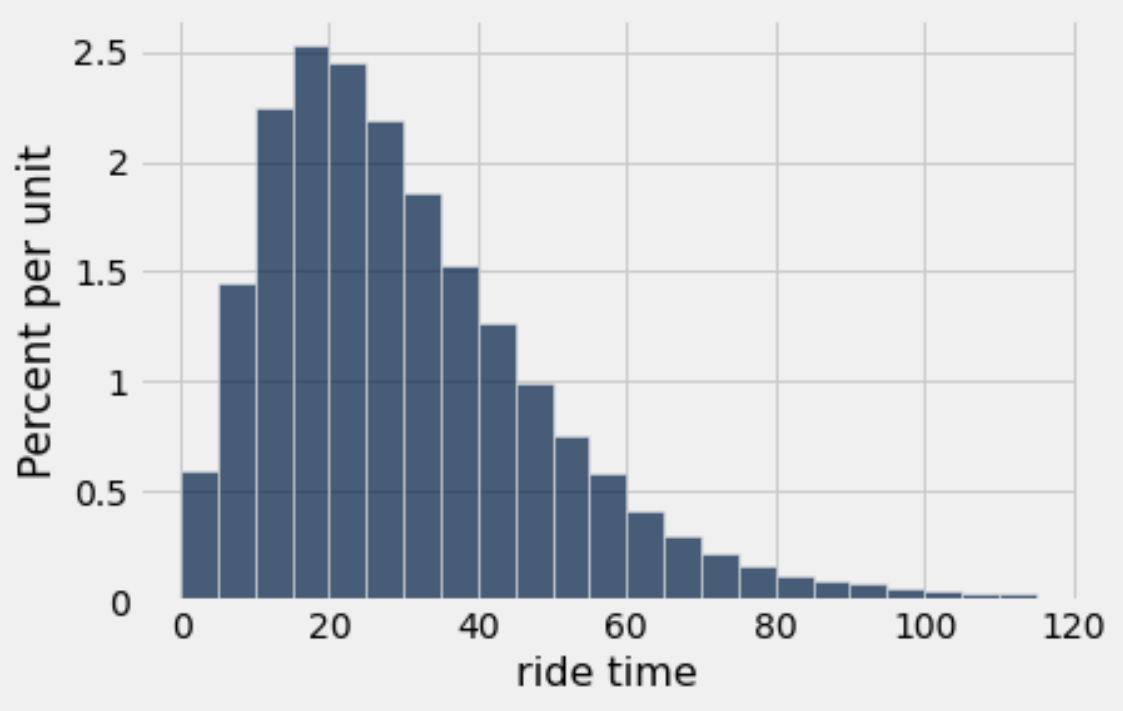
The gopher grader output and/or output from your coding are essential to helping your instructor grade your work correctly and in a timely manner.

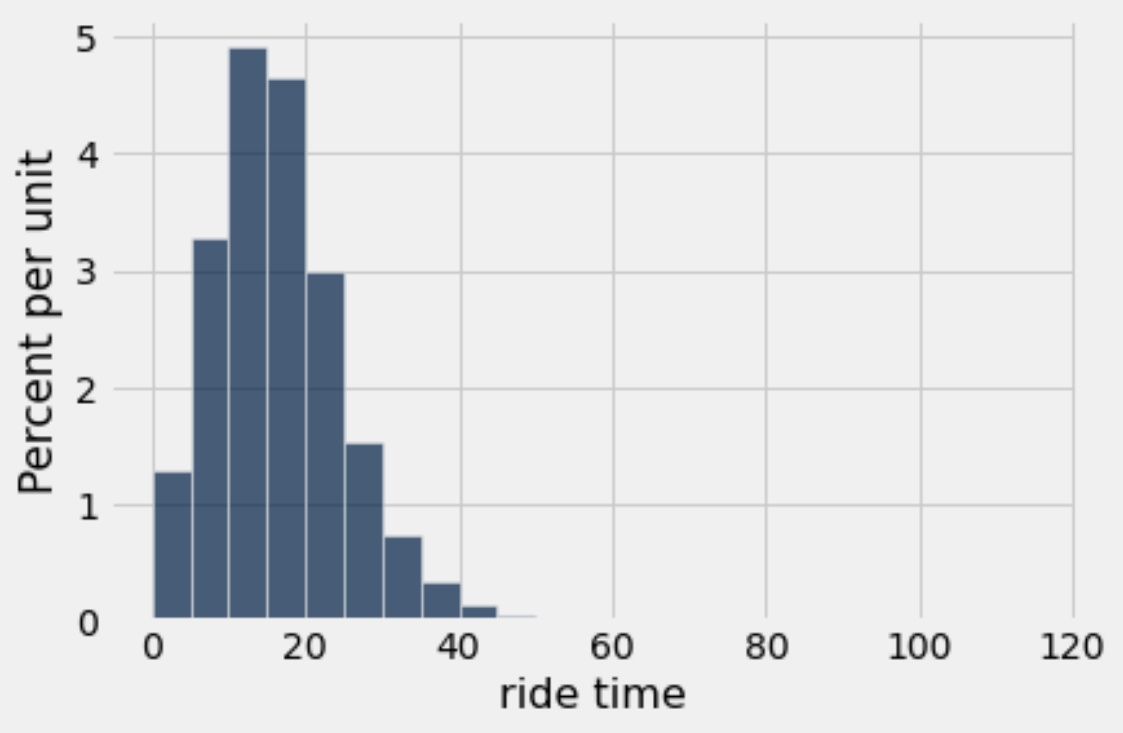
Files submitted that are missing the required output will lose some to all points so double check your pdf before submitting.

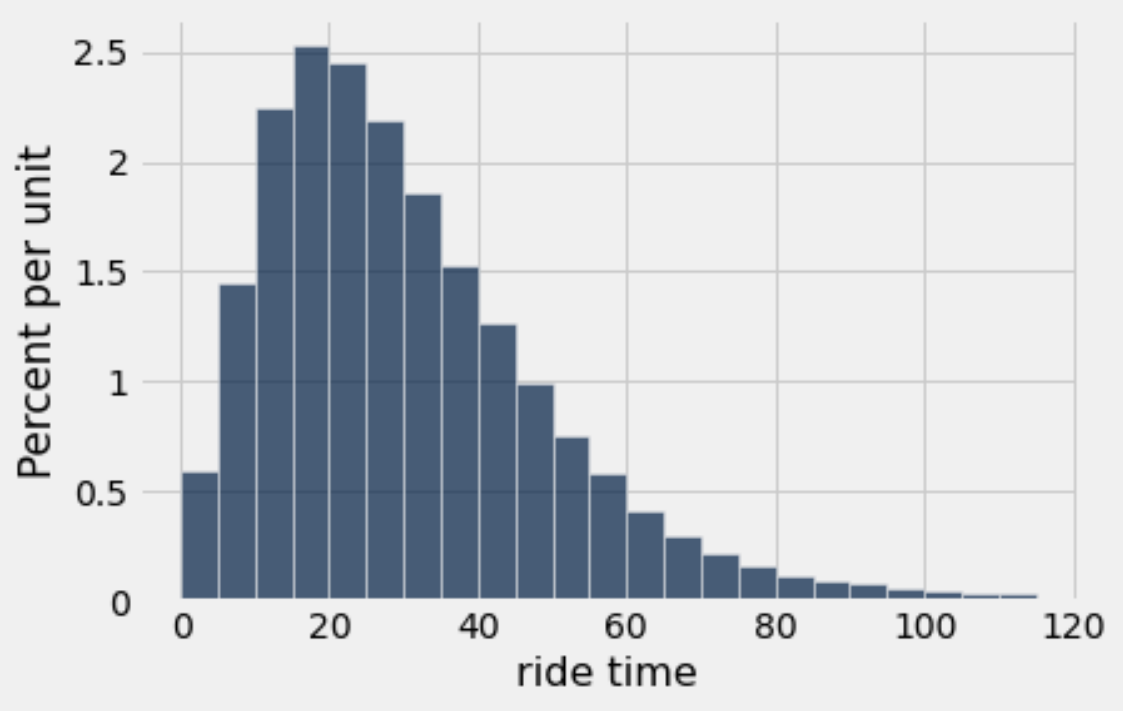
[65]:
```
# For your convenience, you can run this cell to run all the tests at once!
import glob
from gofer.ok import grade_notebook
if not globals().get('__GOFER_GRADER__', False):
    display(grade_notebook('hw04.ipynb', sorted(glob.glob('tests/q*.py'))))
```

```
Boston Table
Manila Table
['tests/q1_1.py', 'tests/q1_2.py', 'tests/q1_3.py', 'tests/q1_4.py',
'tests/q2_5.py', 'tests/q3_1.py', 'tests/q3_3.py']
Question 1:

<gofer.ok.OKTestsResult at 0x7f0f6cea5fd0>

Question 2:

<gofer.ok.OKTestsResult at 0x7f0f6cf8a190>

Question 3:

<gofer.ok.OKTestsResult at 0x7f0f6ce9b350>

Question 4:

<gofer.ok.OKTestsResult at 0x7f0f6d058f50>

Question 5:

<gofer.ok.OKTestsResult at 0x7f0f6cfcd290>

Question 6:

<gofer.ok.OKTestsResult at 0x7f0f6d075590>

Question 7:

<gofer.ok.OKTestsResult at 0x7f0f6f26cf50>

0.8571428571428571
```


Percent per unit
0
1
2
3
4
5
0
20
40
60
80
100
120
ride time


Percent per unit
0
0.5
1
1.5
2
2.5
0
20
40
60
80
100
120
ride time





Name: Allan Gongora

Section: 0131