Interlude: Schools

In this section, we explore a simple code gloss on the stories that in-school data tell. How do they reinforce existing stories? What stories do they hide? As with all the code sections, or, for that matter, as with (quite literally) all communication, ideas or positions are erased, foregrounded, backgrounded, and subject to bias. These are simplistic digital explorations of the nature of complex social interactions. That does not make them less worthy of study or comprehension; ignoring dominant modes of power does not make them go away.

As in all the code in the book, we must start by IMPORTING the appropriate parts of the Python language (called "libraries") so that we can use them as we program. Here, we need: various math functions (basic arithmetic is included by default); the ability to generate pseudorandom numbers (that is, "effectively random" numbers); the NumPy library, which is at the core of most data analysis in Python; and a statistics (and machine learning) package called SCIKIT-LEARN.

```
from random import randint,choice,sample
from math import ceil, floor, log
import numpy as np
from sklearn.linear_model import LinearRegression
```

If we want to print out our very simple graph, we need to handle that ourselves. The following "cell" in this "notebook" prints a textual graph given a list of points on that graph.

```
# this generates a list of random numbers (defaulting to 100)
# with a given mean (def. to 50) and standard deviation (def. to 25)
def generate_simple(n=100, mean=50, sd=25, max=100):
    return list(np.random.normal(mean, sd, n))

# this function makes a readable picture in characters
# (a variable with 1 or more characters is a "string")
# given a graph that is a list of coordinates ("tuple"s)
# for instance, a graph that is [(0,1),(3,4)] includes two points
# (x=0,y=1) and (x=3,y=4).
def graph_to_string(a_graph):
    output = "+\n"
    a_graph.reverse()
    for x in a_graph:
        output += "|{}\n".format(""".join(x))
    output += "+" + "-"*len(a_graph[0])
    return output

# if we are drawing our own graphs, we need to scale
# them to the right size. we don't want all the points
# to be on top of each other, say.
def normalize_datum(y, x, bucket_size_y, bucket_size_x):
    return (floor(y / bucket_size_y), floor(x / bucket_size_x))

graph_size_x = 20 # this is an arbitrary number that i thought looked ok!
graph_size_y = ceil(graph_size_x / 2) # graphs often are wider than tall
def plot_simple(ys, xs = []):
    if ([] == xs) or (len(ys)!= len(xs)):
        xs = range(len(ys))
    bucket_size_y = max(ys) / (float(graph_size_y) - 1)
    bucket_size_x = len(xs) / (float(graph_size_x) - 1)
    graph = [[" " for x in range(graph_size_x)] for y in range(graph_size_y)]
```

```
    for i in range(len(xs)):
        dn = normalize_datum(ys[i],xs[i],bucket_size_y,bucket_size_x)
        graph[dn[0]][dn[1]] = "*"
    return graph_to_string(graph)
```

Now that the preliminaries are out of the way, let's delve into the world of data science. Unfortunately, there is no easy way to explain linear regression without a lot more information; indeed, this is one of those situations in which a little knowledge is more dangerous than none. That said, people use linear regression every single day across thousands of professions. A *linear regression* is an approach for modeling the relationship between two variables (Wikipedia, 2022, among thousands of others). In our case, we will draw a line and the data from two variables and see that a linear regression produces a "meaningfully predictive" model of that relationship (be it strong or weak, positive or negative).

```
max_events = 10

simple_data = generate_simple(max_events) ## generate random data
print(plot_simple(simple_data)) ## plot those data

X=np.array(range(len(simple_data))).reshape(-1,1) ## put those data in the right
format
y=simple_data
model = LinearRegression().fit(X,y) ## perform our linear regression
print(f"linear_model: score: {model.score(X, y):.2f}, coef: {model.coef_[0]:.2f},
intercept: {model.intercept_:.2f}")
# show how we did

pred = model.predict(X) # draw the best fit line
print(plot_simple(pred))
>>>
+
```

```
|    *
|*      *
|  *   *
|        *
|
|           *
|
|   *      *
|
|*
+--------------------
linear_model: score: 0.01, coef: -0.60, intercept: 49.79
+
|*
|*********
|
|
|
|
|
|
|
|
+--------------------
```

Now that we have seen how linear regression is performed, we can play a simple game. The algorithm generates random data, and you tell if it is right or wrong.

The obvious first modification (on your part) would be to input your real-world data. We provide a list (called VARIABLES) of possible data you could input; these are variables that real schools use every day. You would be shocked. Real schools and corporations use data that are often this simple (with the addition of your identifying information).

```
print("I will tell you a story about your data, then you tell me if it is correct.")
variables = ["current grade","actions per minute","things turned in","lateness",
"attendance","other browser windows open"]
salient_variables = sample(variables,4)
real_answers = ["My mom was sick.","My dad helped me.","My job made me
come in, so I could not get any sleep."]

X=np.array(range(max_events)).reshape(-1,1)
y=generate_simple(max_events)
model = LinearRegression().fit(X,y)
m = model.coef_[0]
if m < -1:
    judgment = "going down."
elif m > 1:
    judgment = "going up."
else:
    judgment = "staying the same."
print(f"The variable '{choice(salient_variables)}' is {judgment}")
print("What is the real story?")
print("Working only off this data, it might look like: ")
for col in salient_variables:
    print(f"\t{col}")
print("Some stories I might tell are:")
for col in real_answers:
    print(f"\t{col}")
print("Why can't those data tell that story?")
>>>
I will tell you a story about your data, then you tell me if it is correct.
The variable 'other browser windows open' is going down.
What is the real story?
Working only off this data, it might look like:
        lateness
        other browser windows open
        things turned in
        attendance
```

Some stories I might tell are:

My mom was sick.

My dad helped me.

My job made me come in, so I could not get any sleep.

Why can't that data tell that story?

Real lives are hard to contain in a limited set of variables; they make the world much messier and the story, well, less straightforward. It is easier to get mad at a kid because their homework is late. It is harder to be mad if you know that it was late because that kid's mom was sick. Is there a set of variables that makes these analyses humane? How do we interpret them humanely?

This is a section of doi:10.7551/mitpress/14381.001.0001

The Left Hand of Data

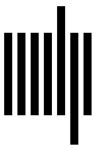
Designing Education Data for Justice

By: Matthew Berland, Antero Garcia

Citation:

The Left Hand of Data: Designing Education Data for Justice
By: Matthew Berland, Antero Garcia
DOI: 10.7551/mitpress/14381.001.0001
ISBN (electronic): 9780262377645
Publisher: The MIT Press
Published: 2024

The open access edition of this book was made possible by generous funding and support from MIT Press Direct to Open



The MIT Press

The MIT Press would like to thank the anonymous peer reviewers who provided comments on drafts of this book. The generous work of academic experts is essential for establishing the authority and quality of our publications. We acknowledge with gratitude the contributions of these otherwise uncredited readers.

This book was set in Stone Serif and Stone Sans by Westchester Publishing Services.

Library of Congress Cataloging-in-Publication Data

Names: Berland, Matthew, author. | Garcia, Antero, author.
Title: The left hand of data : designing education data for justice / Matthew Berland and Antero Garcia.
Description: Cambridge, Massachusetts : The MIT Press, [2024] | Includes bibliographical references and index.
Identifiers: LCCN 2023030088 (print) | LCCN 2023030089 (ebook) | ISBN 9780262547529 (paperback) | ISBN 9780262377652 (epub) | ISBN 9780262377645 (pdf)
Subjects: LCSH: Education—Data processing. | Education—Research. | Educational evaluation.
Classification: LCC LB1028.43 .B45 2024 (print) | LCC LB1028.43 (ebook) | DDC 370.285—dc23/eng/20230718
LC record available at https://lccn.loc.gov/2023030088
LC ebook record available at https://lccn.loc.gov/2023030089