Colby Lamson-Gordon

Coding Markets

10 May 2019

Professor Sethi

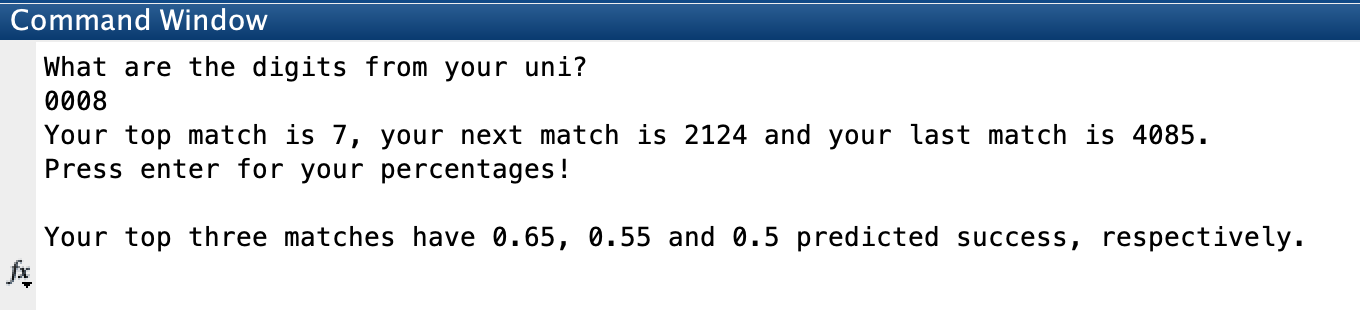
Coding Final Project: Research Summary

I wanted to explore and challenge the notions of compatibility for my final coding project. As a kid in middle school, I remember fondly the Love Calculator—an online website that allows people to input two names to see the predicted percentage of success of the relationship between the two people. I knew it was fake, but it still left me wondering if it was possible to create a love calculator. When I learned about the New York Times’ quiz—36 Questions That Lead to Love—in high school, my middle school hopes returned. Even though I have never taken this quiz with anyone for the purpose of falling in love, I did find that it built trust authentically; it actually brought me closer to some of my friends. Thus, after learning the tools in this class, I found the opportunity of this project to make my dream of creating a real love calculator true. Using questions adapted from the New York Times’ quiz, I create a user-facing program that quizzes them and returns three of their highest matches—calculated from counting the number of identical responses. I do not consider any individuals’ sexuality because I wanted to consider critically platonic intimacy and compatibility. Rather than solving a problem, I hope this program gives insight into addressing do you need face-to-face interaction to find authentic connections with people and does likeness indicate compatibility?

The critical aspects that inform my code are twofold: the survey questions and how I define compatibility. While the New York Times’ quiz has 36 questions broken up into three sections, my quiz uses 20 questions pulled from each of the three sections. All of the questions in the New York Times’ quiz are phrased to open dialogue and conversation. Because I am attempting to control for that, I picked questions that could be answered as either, yes/no, a duration of time, or three to four general ideas. I know that these questions were carefully crafted by psychologists, so I tried to keep the wording as much as possible. An assumption of my quiz is that I trust that these psychologists picked and crafted these questions with purpose of not only stirring vulnerable conversations, but ultimatlye in determining compatibility.

Leading to my next point: compatibility. I measure compatibility in my program as sameness in whether or not the last respondent’s answers are identical to all of the previous answers in my code. I will explain how I coded this later on. I decided to do this because it allowed for straightforward analysis, but more importantly tapped into a factor that people actually do weight heavily in looking for compatible people in their lives: similarities. While I do not know a lot about it, I do know that, generally, “like marries like,” and that it is true that marriages of similar income brackets are one perpetuator of financial inequality because of the passing-down of familial wealth. Another stereotype is “opposites attract.” If given more time, I could also make matches based off exactly opposite responses and compare to my current. I would like to do this and see my code as a working program. My program is informed by the questions from the New York Times’ quiz designed by psychologists to most authentically build connection and determine compatibility, and the fact that I use compatibility in my code as sameness.

My code is comprised of a figure display, a user’s response in the command window, analysis, and a summary of what their results were including a histogram. The user answers the quiz with mouse clicks in a figure and inputs the digit portion of their uni in the command window. At the end, they see their results in the command window and a new figure—a histogram showing the frequency of the number of identical responses between the user and the database of people. As the user goes through the quiz, my program builds a matrix that tracks the responses, coded as 1s, 2s, 3s, and 4s, and writes them to a file called DataFile.txt with the rows as each respondent and the columns as the question. I unfortunately could not create the quiz portion within a streamlined for loop because there was no pattern to the questions in terms of a sequence of yes/no type questions vs other questions with categorical responses. Each needed to be programmed individually, which adds substantially to the length of my program. I then read the DataFile.txt and create a matrix of ones and zeros that compares the coded values of the user to the existing database of values within the DataFile.txt by indicating 1 for identical and 0 for different. This matrix is called matches, with the rows being people and the columns being questions. To find the top three matches, I sum each row in matches. The highest possible score is twenty, which would indicate perfectly identical responses. The raw number indicates that number of identical responses. To complete the matching process, I create a new matrix called unitopmatches, which adds the uni of that respondent to their respective number of identical responses. This matrix tells me the number of similar responses each person in the database has with the user. I then sort it by descending order and designate the top matches as unis named best, next, and last. Each have their respective percent success – measured as percent of identical answers – as well. I then create a histogram for the user, to see the number of people who share the same number of responses as them. For example, a bar with bin five at height six means that six people shared five identical responses to you (out of twenty). Thus, the percentage of success would be 25%.

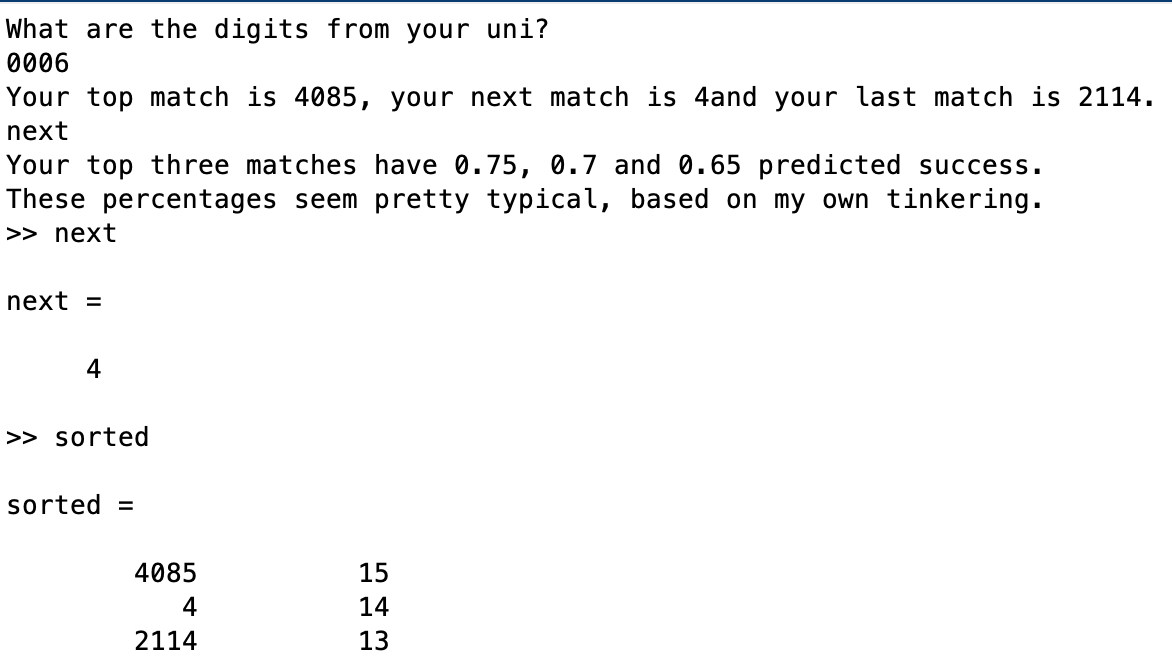
I feel strongly in that I used and draw from real data. While I could’ve done this with simulated responses, I was genuinely curious to see how people would respond. While my sample pool is relatively small, my takeaway from this is a surprise of how dissimilar people are. I ran trials myself and found that most times the highest matches were only of 50%-70% similarity; I anticipated higher response. My own test responses can be found in the DataFile.txt as responses 0001 through 0008 (I ran them as I was debugging and as I was finishing my code, so I apologize for the error messages). Here are the following results from one of my test results as user 0008:

In this response, the three highest matches had 65%, 55%, and 50% identicalness respectively. Here is the accompanying histogram as user 0008:

number of identical responses between user and database 


I could not get the labels to add to my histogram, even though I tried. The x-axis shows the number of identical responses between the user and the database and the y-axis shows the frequency of these occurrences. So, this graph shows that four people shared five identical responses, eight people shared both six and seven identical responses…and so on.

In a trial as user 0006, the top three matches had a much higher rate of success. Of note: please ignore the comment that these percentages were typical. I had that coded in before, as when I was users 0005, 0004, 0003, 0002, and 0001 my top matches were between 50% and 60%.



In conclusion, I learned that people are more different than I expected them to be. This fact questions a key assumption of my program: that compatibility is determined by likeness. It is something to continue to consider. I think it is also important to recognize that coding inherently controls for emotional factors particularly relevant in any dialogue in which people are open to love. By nature of coding, this code succeeds by only using the responses to intimate and/or revealing questions given through clicks on a screen. I tried to show that you do not really need face-to-face interaction but based off the top matches having lower than expected percentages of likeness, I think that 1) people overcompensate their similarities face to face to build connection or 2) people base compatibility more from emotions seen face-to-face than and how someone engages rather than what they say. This final project has been really interesting for me to think about not only ways that society engages with notions of compatibility, but also how I do it. I watched many of my peers take this test, and it was interesting in itself to watch them take it—and the dialogue fostered did give more insight into platonic compatibility between acquaintances and friends I got to take the quiz. While I believe this quiz is viral not because of the questions themselves but rather the dialogues fostered from them, I think that this program successfully questions notions of compatibility and I walk away questioning if I, personally, should seek compatibility based off likeness.

Side note:

I feel like I have learned a lot in this class. I came into this class with no coding experience, and it was immensely gratifying to write this program, in particular the code to find identical matches (lines 692-701). I remember struggling with this in the beginning, and to be able to do it for a program that I was passionate about was really fun. Also, I apologize that my function is so long. I prioritized having my program run correctly so I didn’t get to create functions to shorten the main script.