Dating Project Code Details

This paper explains what to do with every datapoint from the blind dating survey.

There are two steps with the data

1. Cosine Similarity (finds matching scores)
2. Maximum Matching algorithm (maximum matching of a graph with mutual score threshold) (will explain more later)

Note: I am referring to “a person” in this paper as “perX”, where X indicates their index in either the hash map or vector storing each sample perX.

<u>General Overview</u>

Cosine similarity should result in each person having a ranking vector for their top preferences. For example, if n = 4:

rankingPer4 = {‘per1’: 0.80, ‘ per2’: 0.75, ‘per0’: 0.23, ‘per3’: -0.50}

There may be a better way to organize this, but it is crucial that it considers every person besides the person themself. This can easily generalize to everyone if we let rankings with impossible matches (like straight guy to straight guy) be = -1.

Remember, the cos(x) is between -1 and 1, so -1 is our worst possible match and 1 is our best possible match.

Now I want to mention the semantics of how I’ll be referencing data in this survey. To keep things short and sweet, I’ll be referring to column name in the .csv file you can download from the survey. This goes from A -> JO, with many of these instances blank. You should be able to perform modulo26 arithmetic to transform any of these lettered columns into numerical indices. I will write what I think it should be as a number next to, but if I'm wrong, you should only need to shift my given number by a fixed amount. I'm assuming A = index 0. However, since AA comes after Z, the counting is a little weird. Please check.

For example: if the question "How many years younger" (D | 3) is actually in index 4, then just assume every number I give corresponding to the column is off by 1 in your code.

Rows are different, and I imagine less important for precision as we only need to call all of them, not specifically match questions together, but I will be assuming that "Rook!" is Per1 (and row one in the survey) just to keep it easy matching person numbers to row numbers.

<u>Cosine Similarity</u>

Firstly, I want to explain what cosine similarity mathematically so you can make sense of everything else as you read this document.

Cosine Similarity uses a distance metric (cos(x)) to determine how close (or similar) two vectors are. This is a great way to talk about similarity with multi-dimensional vectors as opposed to just a normal distance formula d = |a - b|.

The cosine between two angles comes from the following formula, which I will refer to as the CSF (Cosine Similarity Formula).

Given A, B vectors of the same size…

CSF: $$\cos(\Theta) = \frac{\mathrm{A} \bullet \mathrm{B}}{|\mathrm{A}|\ |\mathrm{B}|}$$

You can find more information about this equation online if you look up cosine similarity. This is an equation from linear algebra. Note that the magnitude |A| is just the square root of the sum of squares. The value of the fraction WILL be between -1 and 1. We never actually have to use the cosine function in the code.

How does this help us with our project?
We have designed our questions in a way that we can quantify all of our data and then find the similarity between two people. Also note that we have set it up that each person has a "Self" and "Want" vector.

"Self" pertains to who they are themself.

i.e. I wake up early.

"Want" pertains to what they want in someone else.

i.e. I prefer someone who wakes up early.

I will be using per1 and per2 as examples for now. This is not based on real data.

vectorMap = {'per1': [per1Self, per1Want], 'per2': [per2Self, per2Want]}

This follows typical JSON formatting.

Note that every vector is of the same size! Otherwise, the CSF won't work.

The majority of this paper will be detailing how we get the numbers we do, but let's assume that we already have them, and it looks like this.

per1Self = [1, 6, 7, -2, -4]

per1Want = [1, 7,-2, -5, -3]

per2Self = [0, 2, 5, -2, 5]

per2Want = [5, 7, -3, -2, 1]

I chose these completely randomly. Also I'm not going back to do this math again, but values on the 7 point scale are between (-3) and (3), not (-7) and (7).

We compare what someone WANTS with what someone else HAS (Self)

Applying CSF to per1Want and per2Self,

$$\frac{\text{per1Want} \bullet \text{per2Self}}{|\text{per1Want}||\text{per2Self}|} = \frac{[1,7,-2,-5,-3] \bullet [0,2,5,-2,5]}{|[1,7,-2,-5,-3]|\,|[0,2,5,-2,5]|} = \frac{-1}{\sqrt{88} * \sqrt{58}} = -0.014$$

Applying CSF to per2Want and per1Self,

$$\frac{\text{per2Want} \bullet \text{per1Self}}{|\text{per2Want}||\text{per1Self}|} = \frac{[5,7,-3,-2,1] \bullet [1,6,7,-2,-4]}{|[5,7,-3,-2,1]|\,|[1,6,7,-2,-4]|} = \frac{26}{\sqrt{88} * \sqrt{106}} = 0.269$$

These numbers suggest that per2 wants per1 less than per1 wants per2.

This score would be placed in each person's ranking (as mentioned above).

per1Ranking = {'per2': -0.014, …, 'perX': 0.025}

per2Ranking = {'per1': 0.269, …, 'perX': -0.30}

Again, this hash map would have all but the person themself with a score. This is THE output we are trying to get from cosine similarity. I have a suggested plan for how to use it, but we have a lot of options we could use to match people after we obtain this metric.

There are n people each trying to match with n-1 other people, and since "self" is different than "want", there aren't duplicate operations we can systematically skip. This leaves "Step 1: Cosine Similarity" as a O(n*(n-1)) = O(n^2) problem.

Hopefully this all makes sense. Please text me if you need any clarification on the motivation behind using cosine similarity

Our application of Cosine Similarity

All of this said, the next (and biggest problem) is to determine how we get these numbers in our data. This WILL be tedious as we haven't streamlined inputs in a way that we systematically match them to each other. This is why we created the "Sex 100" google sheet to show which questions pair with each other and which questions need to flip polarity to pair correctly. For questions like "body type preference", we have different ways to fix these questions, which I will call "Amends".

Amend 1: What and Why do we fix the “polarity” of questions?

Consider a “self” question and a “want” question with opposite polarity.

> “self”: You cry very often.
>
> “want”: You are uncomfortable consoling someone who is emotionally unwell.

The “self” question to something about YOU and the “want” question implicitly asks that the other person is able to console you if you cry often.

Responses for most questions are on a 7-point scale. The csv file reads this as (1) – (7), but we immediately want to translate this as (-3) – (3). So, subtract every value on those types of questions by 4 (i.e. 7-4 = 3 (max score)). I will be referencing score responses by what it should be mathematically (x-4).

If someone answered “you cry very often” with a score of (-3), then they DISAGREE and do NOT “cry very often”. Good matches multiply to positive scores. Look at the dot product in the CSF to make sense of this (it increases the score). So, someone who does NOT “cry very often” should be paired with someone who IS “uncomfortable consoling someone”. However, a high score for somebody who IS “uncomfortable consoling someone” would be (3), but then (-3)*(3) = (-9), a BAD pairing. So in order to fix this, EVERYONE’s score for one of the columns should be made the opposite sign of what it is.

i.e. IS “uncomfortable consoling someone” = (-3) is changed to –(-3) = (3)

Amend 2: Kill questions. Hard yes and hard no.

Some questions like religious requirements, drug use, etc. sexuality, ask that the other person fits their needs EXACTLY. These will be conditional statements that happen at the beginning of analysis. If any condition is not met, we don’t even need to consider the rest of the questions. Immediately set the matching to -1 (our lowest possible score). Any lower is unnecessary and this will still guarantee no matching. (Skips CSF, sets = -1)

Amend 3: Preference "distance" and "alignment" (D&A)

For questions like height, we can't guarantee girls get paired with their 7 foot king, so we don't want to make it an all or nothing pairing. This type of question is one of the HARDEST to make sense of, and is up to a lot of personal judgement on our end.

Just a reminder: "self" responds with a single input and "want" puts multiple options. These aren't numeric, so we want to come up with a way to make this matching preference numeric.

My thought is that if someone aligns with a preference, let it be (+3), a positive alignment. If it's one off, let it be (+0), indifference. For each thing "further" than that, -3, -6, etc.

How can we do this?

Consider the following

5'1 or shorter	= 0
5'2 – 5'5	= 3
5'6 – 5'9	= 6
5'10 – 5'12	= 9
6'1 – 6'3	= 12
6'4 or taller	= 15

per1 says they ARE "5'6 – 5'9"

per2 says they WANT "5'10 - 5'12", "6'1 – 6'3", "6'4 or taller"

Numerically, we are going to translate this into distance as…

per1 says they ARE 6

per2 says they WANT 9, 12, 15

The following code assumes that any height is just as good as the others to go on a date with, and that any deviation from that window reduces the score.

```
# Determining compatibility for per2 with per1
if heightPer1 < min(heightPrefPer2):
    # Finds the distance + 3 for preference adjustment between the tw
    score = |heightPer1 – min(heightPrefPer2)| + 3
if heightPer1 > max(heightPrefPer2):
    # Finds the distance + 3 for preference adjustment between the two
    score = |heightPer1 – min(heightPrefPer2)| + 3
else:
    score = 3
```

These values can change, and likely will for weighting purposes. The difference in segments 1 and 2 is to determine which value of the range to compare with. This also assumes people don't have a height gap in their preferences. A bold, yet reasonable assumption.

"score" will go in the per2Want vector and be conditionally ADDED and to the other person's score and then *2. This is genuinely one of the hardest questions to ask. Multiplying two negative scores would imply a match, which is not true, which is why we add them, as they'd either meet in the middle, or inflate each other's interest or disinterest. The *2 inflates the result no matter what but could change under our discretion.

<u>Amend 4:</u>

Main Formula:

I'm going to skip over anything that is not included in the provided formula. If you see a way to include it I'd so go for it. Any columns I skip I will include at the bottom.

Also, I am going to assume that every category is of equal weight unless specified differently at the end. This is useful since we can add the pieces one by one. Here's how this will work. HOWEVER, I do think this is not as good of a result as weighting before CSF, since this could allow weighted scores to surpass 1, and for the person who puts weight preferences, it would make matching more problematic. I'm going to explain it anyways as it is easier, and the Categories are still a useful concept.

Consider each category as C_i. For instance, we have 6 categories which I will list here.

C_1: Emotion

C_2: Conflict

C_3: Extraversion

C_4: Lifestyle

C_5: Communication

C_6: Partner Interaction

A potential 7th category could be "Looks", but per the messages I sent in the GroupMe I think this may be left alone for consideration by us individually. If we can get a metric for everything else, I think we will be very close everything else aside.

If we don't categorize, we would have "self" and "want vectors like this"

$W1_{all}$ = [1, 2, -1, -1, -1, 2, 0, 1, 1]

$S2_{all}$ = [3, 2, -2, -2, 0, -1, 0, 1, -3]

CSF for C1 and C2 = 0.33 (I recommend checking this to make sure we are on the same page with the calculation. It's small enough to also check by hand.

But if we categorize as three categories

	C_1	C_2	C_3
$W1_{sum}$ =	[1, 2, -1]	[-1, -1, 2]	[0, 1, 1]
$S2_{sum.}$ =	[3, 2, -2]	[-2, 0, -1]	[0, 1, -3]

Applying CSF to each

$CSF_1 = 0.98$

$CSF_2 = 0.00$

$CSF_3 = -0.44$

This means that the two greatly align on C_1, are indifferent about C_2, and show problems with C_3. If we add these up in thirds though, we get.

$$\frac{1}{3} * CSF_1 + \frac{1}{3} * CSF_2 + \frac{1}{3} * CSF_3 = 0.18$$

It's important to realize that this is different score, but it is more interpretable, and it allows for an easier adjustment to matching scores by letting add multipliers. I will write down the two options in detail for considering the scores.

Method 1 (weight pre-CSF):

Create your 6 (or any k) categories C_i.

Assign the values (explained after this) to the appropriate category and index.

Before comparison, adjust the weight of each important and unimportant category people specified in column CM and CN (index 90 and 91 I think). I recommend *2 and *1/2 respectively.

You should have 6 (or k) categories vectors. Now append them together.

i.e. if you had $C_1 = [1, 2, -1]$ and $C_2 = [-1, -1, 2]$ with "importance" on C_1, then $C_1 = [2, 4, -2]$ and $C_2 = [-1, -1, 2]$, and $C_{all} = [2, 4, -2, -1, -1, 2]$

THEN apply CSF for every element in the same vector C_{all}.

Benefits:

Weighting is done 1 time for each person instead of $O(n^2)$

Scores are normalized between -1 and 1.

Cons:

Less interpretable

Hard to fix later.

<u>Method 2 (weight post-CSF):</u>

Create your 6 (or any k) categories C_i.

Assign the values (explained after this) to the appropriate category and index.

Apply CSF between relative categories on each "self" and "want" vector.

Multiply obtained CSF_i scores relatively by "importance" rating.

Sum all $(1/k) * CSF_i$. (The idea here is that without weights, this adds up to something between -1 and 1.

Benefits:

Easily interpretable

Easy to amend later

Cons:

Longer processing time.

Not normalized

Suggested Variable Assignment and Categories

This is the real “meat and potatoes” of it all. Please be thorough and check over this yourself to make sure everything your pulling is going in the right places. Negatives (-) indicate a polarity flip.

For verification, reference the “sex 100” sheet. Question indices are pulled from that and CTRL+F checking the survey. Expected index is hard to figure out. Please check my accuracy, but I am doing the best that I can.

Categories:

C_1: Emotion

C_2: Conflict

C_3: Extraversion

C_4: Lifestyle

C_5: Communication

C_6: Partner Interaction

C_7: Humor

C_1 (Emotion):

per_Self = [BI, BL, AZ, AY, BC]

per_Want = [-BH, -BN, AZ, AY, BC]

Indexed (hopefully).

per_Self = [60, 63, 51, 50, 54]

per_Want = [-59, -65, 51, 50, 54]

C_2 (Conflict):

per_Self = [BD, BF, BJ, BO, BM]

per_Want = [-BE, -BG, BJ, BO, BM]

Indexed (hopefully)

per_Self = [55, 57, 61, 66, 64]

per_Want = [-56, -58, 61, 66, 64]

C_3 (Extraversion):

per_Self = [AH, AL, AM, AP, AQ, AR]

per_Want = [AI, -AO, AN, AP, AQ, AR]

Indexed (hopefully)

per_Self = [33, 37, 38, 41, 42, 43]

per_Want = [34, -40, 39, 41, 42, 43]

C_4 (Lifestyle):

per_Self = [AJ, AK, AV, AW, AX, BY, BX, AT, AS]

per_Want = [AJ, AK, -AU, AW, AX, BY, -CL, AT, AS]

Indexed (hopefully)

per_Self = [35, 36, 47, 48, 49, 76, 75, 45, 44]

per_Want = [35, 36, -46, 48, 49, 76, -89, 45, 44]

C_5 (Communication):

per_Self = [BA, BB, BK, BZ, BU, BV, CB]

per_Want = [BA, BB, BK, -CA, BU, BV, CC]

Indexed (hopefully)

per_Self = [52, 53, 62, 77, 73, 74, 79]

per_Want = [52, 53, 62, -78, 73, 74, 80]

C_6 (Partner Interaction):

per_Self = [BP, BQ, BR, BW, BS, BT]

per_Want = [BP, BQ, BR, BW, BS, BT]

Indexed (hopefully)

per_Self = [67, 68, 69, 74, 70, 71]

per_Want = [67, 68, 69, 74, 70, 71]

C_7 (Humor):

per_Self = [DB, DC, DD, …, DX]

per_Want = [DB, DC, DD, …, DX]

Indexed (hopefully)

per_Self = [105, 106, 107, …, 127]

per_Want = [105, 106, 107, …, 127]

Column “CM” or “90” Mentions preferences, but as a comma separated list…

Just parse through it. Not ideal, but that’s what we’ve got.

Also, I want to mention some important columns not mentioned…

Anything here that says “set it as a hard no”, these questions are probably best considered before any other computation. IF it’s a hard no, skip CSF test and just set compatibility rating to -1.

C, D, E (2, 3, 4)

Personal age, age younger and older to date

F, G (5, 6)

Sexuality and looking for

If someone is bisexual include both

If someone is trans or other, only let them date bisexual. Idk what else.

AC, AD (28, 29)

If someone put 1 in AD as to say they DON’T want someone with tattoos, set it as a hard no.

AE, AG (30, 31)

If someone is in Greek life or adjacent, and someone puts 1 for interested in dating Greek Life, set it as a hard no.

If someone believes in the inherent goodness of man remove them from the survey.

CG (84)

Recharge battery. Conditionally eliminate any 6:1 from being with a 1:6 social battery. This would not work.

CO (92)

Date availability. This is a parsing problem, so be careful with this but eliminate anyone who can’t be there on the same day as someone else.

CU, CV (98, 99)

Does someone drink and how okay is someone else with it.

Don't worry about precision in matching, just don't let someone who drinks at all be with someone who put 1 (a hard no).

CW, CY (100, 102)

Weed and Nic // Are you okay if they have

If someone is not okay, set it as a hard no

CZ, IQ (103, 250)

Have you done X drug // Are you okay if they have

If they are not okay, set it as a hard no.

FM, EB (168, 131)

Are you religious // Are you okay dating if they aren't religious

If they are not open to dating the other kind, set it to a hard no.

DY, FN (128, 169)

Political Orientation // Open to Dating

If not open to a person's orientation, set it to a hard no.

I would look over everything about yourself, and again PLEASE CHECK MY INDICES. I think this is right but I do not know exactly what your preprocessing looks like.

I skipped some things like aesthetics, looks, height, etc. that we gathered. If you complete everything and feel you have time to incorporate those confidently, I'd say go I ahead. I would approach it with Amend 3 (D&A), but as mentioned in our text chat, I think those sort of things can be manually looked over by Tyler and shouldn't have a great weight on our code. We have a LOT of information.

Example Comparison

Now the part you're REALLY interested in, I want to go over an example comparison between two people who I feel get along well just to see what happens when we check everything. **This is assuming you use the preferred Method 1 – Pre-Weight.**

Cydney Barbee (row 11) vs. Sutherland Wood (row 61).

Step 0: Weights

We will look at what this means for the numbers in a second, but every one should have columns corresponding to a "want" vector weighted according to their preferences.

In this case we have that…

Cydney:

Values more (x 2): Conflict, Lifestyle

Values less(x (1/2)): Communication

Sutherland:

Values more (x 2): Conflict, Partner Interaction

Values less (x (1/2)): Extraversion, Lifestyle

Note that both valuing conflict response will heavily align them on that!
In the dot product part of CSF, that results in a x 4 increase in those values importance to the final scoring. We will understand this better in a second.

Step 1: Conditionals

Cydney Personal

1. Age 22
2. Gender: Woman
3. No Tattoos
4. Not in Greek Life
5. 3:4
6. M30, A1, A2, A3, A4, A5
7. Drinks here and there
8. Vapes
9. Has done psychedelics and weed
10. Not religious
11. Liberal

Cydney Needs

1. 1 younger or 3 older
2. Man
3. Score 3 about tattoos (don’t worry about it then)
4. Not open to Greek life
5. N/A
6. M30, A1, A2, A3, A4, A5
7. Okay with drinks here and there (don’t worry about it then)
8. Okay with Weed, Tobacco, and Vapes
9. Okay if they tried anything as long as it wasn’t hard
10. Open to any
11. Liberal

Sutherland Personal

1. 21
2. Gender: Man
3. No Tattoos
4. Not in Greek Life
5. 6:1

6. M30, A1, A2, A3, A4, A5
7. Drinks here and there
8. No weed, nic, or tobacco
9. Never done hard drugs
10. Religious
11. Moderate

Sutherland Needs

1. 2 younger or 4 older
2. Needs: Man
3. Score 3 about tattoos (don't worry about it then)
4. Score 2 (indifferent then)
5. N/A (but note Sutherland won't be able to pair with a 1:6)
6. M30, A1, A2, A3, A4, A5
7. Okay with a sober person, or here and there drinker
8. Okay with any weed, nic, or tobacco use
9. Okay if they've tried a drug
10. She must be religious
11. Open to dating moderate

They align on everything EXCEPT for part 10 and 11. Unfortunately then, this could never be a match, but I am going to go ahead with it as if it were because I don't want to manually search all of this again. Remember, I am manually checking this too be thorough, but the idea is that a computer would have conditionals and parse through carefully to check this all.

Again though, in a real computer analysis, this would skip the next step and set CSF = -1 without checking anything.

Step 2: CSF Vector Creation

Firstly, I want to remind you that it is imperative that you fix the 7-point scale to being between -3 and 3. That is how I will be referring to the scores written down here. I am writing the value preferences as well. If this complicates things too much, skip it. I do think it is something exciting though.

I'm going to write this out the best I can, but this is going to work out to be a very high dimension vector, which is why it's great to code this stuff hahaha.

First, let's look at Cydney "want" vs Sutherland "self".

Cydney

cValueMore = [Conflict, Lifestyle] or [C_2, C_4]

cValueLess = [Communication] or [C_5]

cWant = {C_1, C_2, C_3, C_4, C_5, C_6, C_7}

cWant = {[-BH, -BN, AZ, AY, BC], [-BE, -BG, BJ, BO, BM]*2,
[AI, -AO, AN, AP, AQ, AR], [AJ, AK, -AU, AW, AX, BY, -CL, AT, AS]*2,
[BA, BB, BK, -CA, BU, BV, CC]*(1/2), [BP, BQ, BR, BW, BS, BT],
[DB, DC, DD, ..., DX]}

cWant = {[1, -2, 1, 0, 1], [2, 0, 2, 1, 2]*2,
[2, 1, -1, 0, 0, -1], [1, 2, -2, 2, 1, 1, 2, 0, 0]*2,
[1, -2, 1, 0, 0, 0, 0]*(1/2), [1, 1, 2, 0, 1, -1],   [...]}
I'm skipping C_7 since there's a lot of humor questions and I'm doing this by hand. Please be sure to include it though. It follows the exact same reasoning.

cWant = {[1, -2, 1, 0, 4], [4, 0, 4, 2, 4],
[2, 1, -1, 0, 0, -1], [2, 4, -4, 4, 2, 2, 4, 0, 0],
[1/2, -1, 1/2, 0, 0, 0, 0], [1, 1, 2, 0, 1, -1],   [...]}

And now after value multipliers have been applied, append them together to make one long vector for the CSF (ORDER OF EVERYTHING MATTERS

cWant = [1, -2, 1, 0, 4, 4, 0, 4, 2, 4, 2, 1, -1, 0, 0, -1, 2, 4, -4, 4, 2, 2, 4, 0, 0, 1/2, -1, 1/2, 0, 0, 0, 0, 1, 1, 2, 0, 1, -1]   (dropped C_7 humor values)

Sutherland

# Multipliers are only added for the “want” vector

sSelf = {C_1, C_2, C_3, C_4, C_5, C_6, C_7}

You may better realize this next step in terms of indices. The letters are in reference to the google sheet columns.

sSelf = {[BI, BL, AZ, AY, BC], [BD, BF, BJ, BO, BM],
[AH, AL, AM, AP, AQ, AR], [AJ, AK, AV, AW, AX, BY, BX, AT, AS]
[BA, BB, BK, BZ, BU, BV, CB], [BP, BQ, BR, BW, BS, BT],
[DB, DC, DD, …, DX]}

sSelf = {[-3, -2, 2, -1, 2], [-2, 3, 3, -1, 3],
[2, 0, -2, -1, 0, 0]), [2, 2, 1, 1, 0, 2, 1, 1, 2],
[2, -2, 1, -1, 2, -3, 3], [3, 3, 3, 0, 3, 0], […]}

Now append them together to make one long vector for the CSF (ORDER OF EVERYTHING MATTERS

sSelf = [-3, -2, 2, -1, 2, -2, 3, 3, -1, 3, 2, 0, -2, -1, 0, 0, 2, 2, 1, 1, 0, 2, 1, 1, 2, 2, -2, 1, -1, 2, -3, 3, 3, 3, 3, 0, 3, 0] (dropped C_7 humor values)

Step 3: CSF Calculation

Now you have the two things needed for CSF calculation: cWant and sSelf.

Remember, this is what Cydney WANTS out of somebody. We are calculating a rating here for her matching with Sutherland, but what Sutherland WANTS would create a different number (sWant vs. cSelf). I'm using an online calculator to do this, but the idea is that you could code a CSF method to do this given to vectors.

So we have…

cWant = [1, -2, 1, 0, 4, 4, 0, 4, 2, 4, 2, 1, -1, 0, 0, -1, 2, 4, -4, 4, 2, 2, 4, 0, 0, 1/2, -1, 1/2, 0, 0, 0, 0, 1, 1, 2, 0, 1, -1]

sSelf = [-3, -2, 2, -1, 2, -4, 6, 6, -2, 6, 1, 0, -1, -1/2, 0, 0, 1, 1, 1/2, 1/2, 0, 1, 1/2, 1/2, 1, 2, -2, 1, -1, 2, -3, 3, 6, 6, 6, 0, 6, 0]

$$\cos(\Theta) = \frac{\text{cWant} \bullet \text{sSelf}}{|\text{cWant}|\ |\text{sSelf}|} = \frac{63.5}{12.9 * 12.29} = \frac{63.5}{158.54} = 0.40$$

This means that Cydney would have a 0.338 match rating with Sutherland. I would not look at this as "percentage match", because it is unlikely anyone will be above something like 0.75. Again, this is a gauge, but honestly, I would say this is a pretty good rating. I don't want to go through all of the steps again to get Sutherland's score for Cydney, but once it's in the computer it'll be easy.

Final Thoughts and Comments

The hardest part of this is two things.

1. Tediously matching the right indices together – this is largely a fault of our methods, which we can change for next time.
2. Setting the conditionals for the hard no questions – just a lot of parsing and such.

A third thing might be searching for the indices, but honestly I think I've done that already and done it well. Check me on it by looking up which index corresponds to which question in your code before locking it in, but I think if it's wrong it's only off by one or two to the left.

Also, I realized that if you are smart with your variables, you should only have to calculate |xSelf| one time since it isn't subject to weights. |xWant| only has to be recomputed because of the weights.

As of finishing this paper, I will be going to New York tomorrow. I hope this fully encapsulates everything but if you have any questions please reach out. If you make any personal decisions about the math please let me know as well. I'm likely to agree, I just want to know what's going on.

Again, this ignores aesthetics, but I think getting this metric for everyone first would be useful. Going back to something I mentioned on page 1, every person should end with a vector like this.
rankingPer4 = {'per1': 0.80, ' per2': 0.75, 'per0': 0.23, 'per3': -0.50}

Ideally, we can create a new .csv file with every person on the spreadsheet (in the same order as the poll spreadsheet) and then each cell next to their name with the ordered list of match ratings.

For example:

Row # (1)	Name			
2	Rook!	Griffin: 0.80	Jack Ward: 0.6	…
3	Caroline	Jayden: 0.79	Zander: 0.77	…
4	Blakely Chandler	Best	2nd Best	…

Once we have this file ^^^, it'll be a much easier problem to just figure out a maximum matching for a graph connecting mutually interested names together since there is one variable lol.

Best of luck! I wish I could be with you to work on, but we will catch up later.