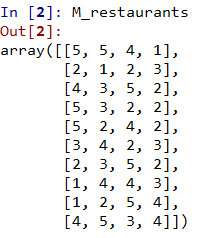
Note: I randomized the input values for people criteria and restaurant criteria. So, each time the program is run, findings could be different. I have taken screenshots of my test run for explanation.

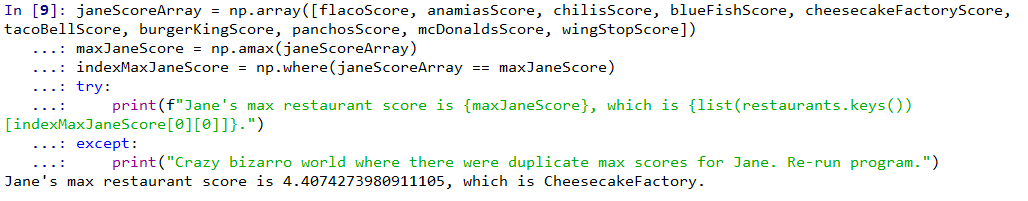
* Transform the restaurant data into a matrix(M\_restaurants) use the same column index.



* The most important idea in this project is the idea of a linear combination. Informally describe what a linear combination is and how it will relate to our restaurant matrix.

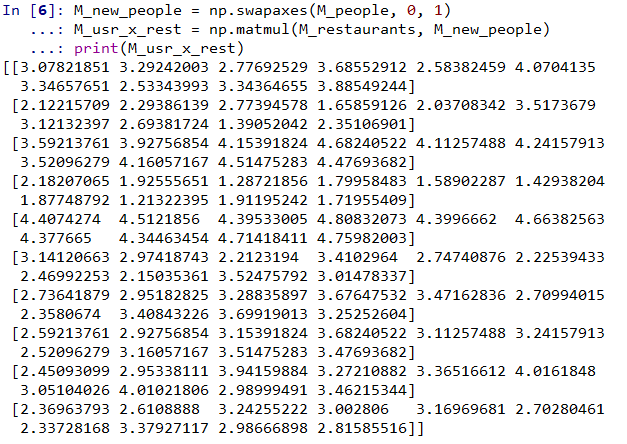
In a linear combination we are multiplying each term by a constant and summing the results. This relates to our restaurant matrix because multiplying our people matrix by the restaurant matrix will give us a "restaurant score" for each person that we can use to optimize lunch choice.

* Choose a person and compute(using a linear combination) the top restaurant for them. What does each entry in the resulting vector represent.



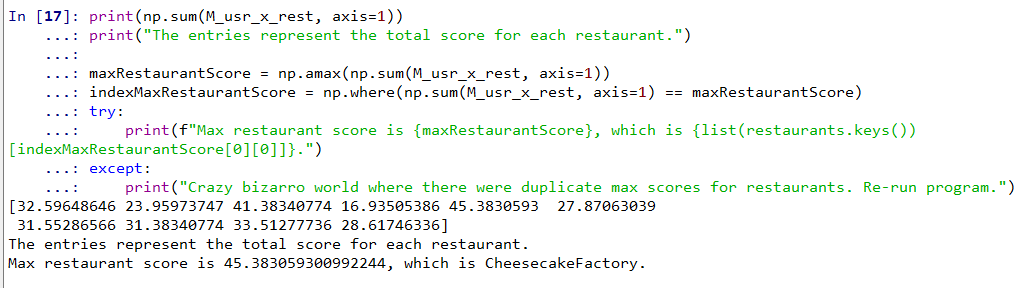
Jane's max restaurant score is 4.4074273980911105, which is CheesecakeFactory.

* Next compute a new matrix (M\_usr\_x\_rest i.e. an user by restaurant) from all people. What does the a\_ij matrix represent?



In M\_usr\_x\_rest, the rows represent restaurants and the columns represent people.

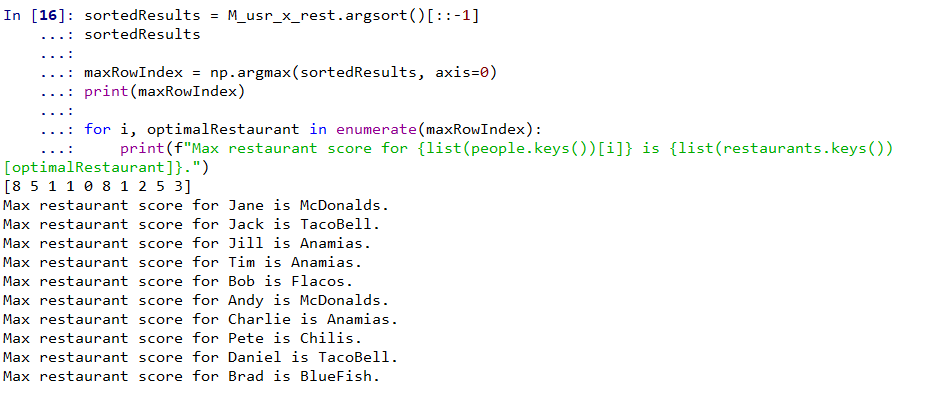
* Sum all columns in M\_usr\_x\_rest to get optimal restaurant for all users. What do the entry’s represent?



Max restaurant score is 45.383059300992244, which is CheesecakeFactory.

The entries represent the total score for each restaurant.

* Now convert each row in the M\_usr\_x\_rest into a ranking for each user and call it M\_usr\_x\_rest\_rank. Do the same as above to generate the optimal restaurant choice.



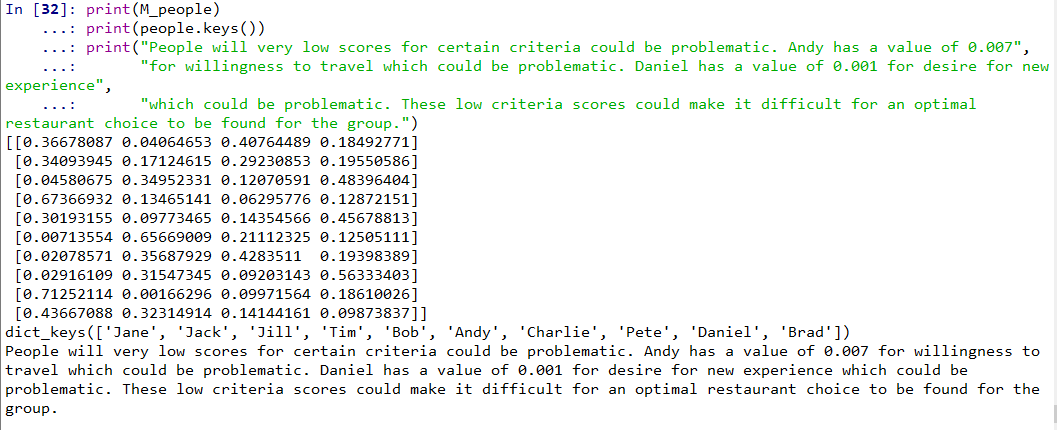
* Why is there a difference between the two? What problem arrives? What does represent in the real world?

There is a difference because there may be groups of people with similar optimal choices. This represents the real world situation where taking all people into account, the entire group cannot make an optimal choice as each restaurant may be optimal for a sub group of people.

* How should you preprocess your data to remove this problem.

We should pre-process our data so we take groups of people into account.

* Find user profiles that are problematic, explain why?



People will very low scores for certain criteria could be problematic. Andy has a value of 0.007 for willingness to travel which could be problematic. Daniel has a value of 0.001 for desire for new experience which could be problematic. These low criteria scores could make it difficult for an optimal restaurant choice to be found for the group.

* Think of two metrics to compute the dissatisfaction with the group.

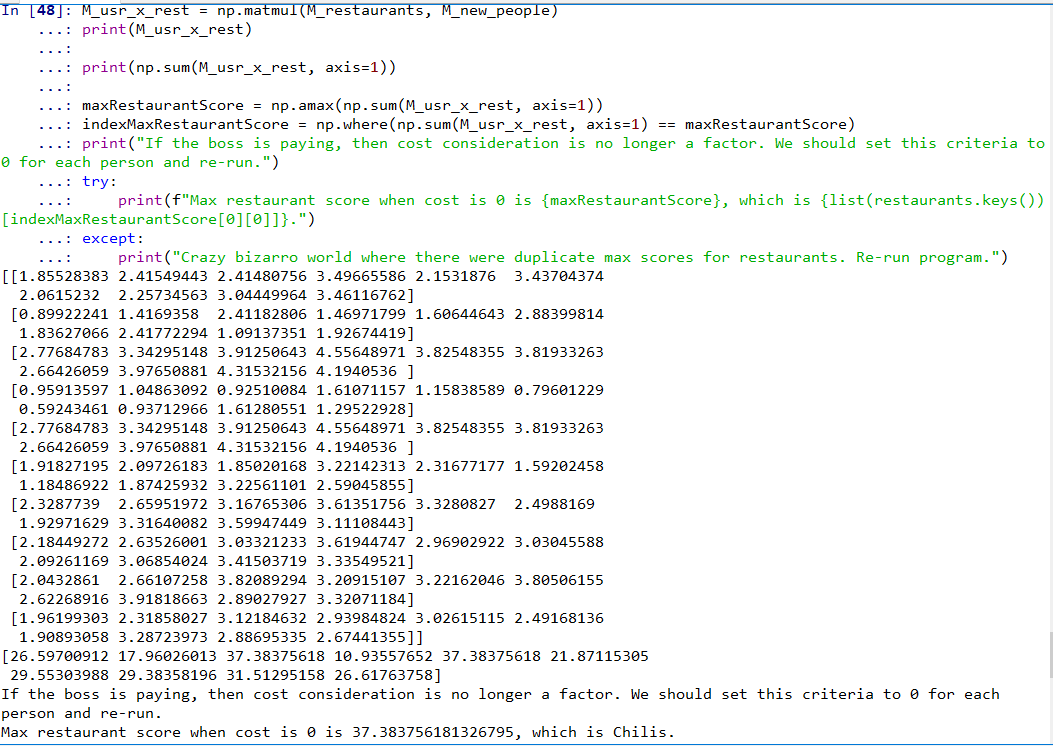
We could take the difference between the group’s optimal restaurant score, and each individual’s optimal restaurant score. The difference between these values could be considered a “dissatisfaction score” for each person.

We could also look at highly ranked criteria for each person and determine if that was in alignment with the group's restaurant choice. For example, if cost > .75 for an individual, if the chosen restaurant is not highly ranked with cost we could consider that dissatisfaction.

* Should you split in two groups today?

Since the group's optimal restaurant choice of Cheesecake Factory is not in alignment with any individual's optimal restaurant choice, we should split into 2 groups.

* Ok. Now you just found out the boss is paying for the meal. How should you adjust. Now what is best restaurant?



If the boss is paying, then cost consideration is no longer a factor. We should set this criteria to 0 for each person and re-run.

Max restaurant score when cost is 0 is 37.383756181326795, which is Chilis.

* Tomorrow you visit another team. You have the same restaurants and they told you their optimal ordering for restaurants. Can you find their weight matrix?

Yes, you could multiply the restaurant matrix by the transpose of the ordering matrix to find the weight matrix.