A Personalized Meal Recommender System

Jiaqi Xiong xiongjia@usc.edu
Jiayue Chen jiayuec@usc.edu
Zhengan Pan zhenganp@usc.edu
Han Wang hwang378@usc.edu

08-May-2020

Table of Contents

Instructed by:
Dr. Suvrajeet Sen s.sen@usc.edu

Abstract

In daily life, our diet often encounters problems such as malnutrition and lack of time. A complete diet plan is very helpful to us. In this project, we tried to make a weekday recipe for a pair of roommates. First of all, we think that the taste of dishes is the most important for a person, so we aim to evaluate the best. Also, keeping similar cooking time for each person can avoid quarrels. Then, the 5 recipes must meet some conditions, such as healthy weekly nutrition, the schedule of cooking time and the total expenses. We crawled the basic information of the recipes on the yummly.com website and cleaned the data, then built a Mixed-Integer Linear Program and solved it by IBM Decision Optimization on Cloud Service.

Introduction

On the premise of satisfying basic needs, improving one's dining taste is the basic goal, and the higher this goal, the better the recipe. Also, the difference between two users' cooking time should be small.

The impact of price on recipes is very important when making decisions. Recipes that are too expensive are difficult to bear, so prices often only need to meet a budget. Actually, people often set a price interval when doing online shopping, so we add price to the constraint function.

Recipes also have strict time requirements. Due to class or work, the time requirements of recipes are also very limited. The cooking time of a week should be met, but it is not the shorter the better. So time is added as a constraint.

Finally, in order to achieve the most basic health advice, we believe that the basic calories and other nutrients consumed every week need to be considered. But notice here that only weekly nutrition should be satisfied to maintain a balanced diet. The daily and weekly nutritional quantities are not simply a summation relationship. For example, perhaps a person can skip meat for a day, but skipping meat for a week can cause physical harm. So we divide nutrition into five constraints and add it to the model. At the same time, due to the requirements of diet richness, we have added to the constraint that each recipe cannot be selected more than once per week. Furthermore, people may have an allergy to certain food and she may be a picky eater, so we should avoid recommending recipes with these ingredients.

After summing up, we have a main goal to improve the quality of the dishes, while also considering the impact of time and money costs on the recipes. The recipe must also meet personal nutritional needs, as well as a person's preferences and allergies. After adding the above factors to the constraints , Forming a comprehensive model

Objective: Maximize ratings for 5 meals, and minus difference between the cooking time needed for each person
Constraints:

- Weekly Budget
- Weekly Schedule
- Time Equality
- Weekly Nutrition Requirement
- Recipes Diversity
- Ingredients Preference

Overview of Models and Methods

Model

As shown below, mathematical language can be used to formulate the model described in natural language above.

$Indices$:

$Person : i \in \{1,2\}$

$Day : t \in \{1,...,5\}$

$Recipes : k \in \{1,...,K\}$

$Parameters$:

$Available\ Time : T_{it}$

$Budget : B$

$Recipe\ price : \pi_k$

$Nutrition\ Vector : A_k = (c_k, f_k, p_k, s_k, car_k)^T$

$Minimum\ Nutrition\ Requirements : R_l = (c_l, f_l, p_l, s_l, car_l)^T$

$Maximum\ Nutrition\ Requirements : R_u = (c_u, f_u, p_u, s_u, car_u)^T$

$Cooking\ Time : \tau_k$

$Ratings : r_{ik}$

$Filtering : F$ indicate the indices for recipes including allergy or dislike food

$Variables$:

$x_{itk} \in \{0,1\}$ indicate whether person i cook recipe k on day t

$w : inequity$

$$\max \sum_{j=1}^{K} \sum_{t=1}^{5} \sum_{i=1}^{2} (r_{1k} + r_{2k}) x_{itk} - \alpha w$$

subject to

$$\sum_{j=1}^{K}\sum_{t=1}^{5}\sum_{i=1}^{2}\pi_k x_{itk} \le B$$

$$\sum_{j=1}^{K}\tau_k x_{itk} \le T_{it}, \forall i \in \{1,2\}, t \in \{1,...,5\}$$

$$\sum_{j=1}^{K}\sum_{t=1}^{5}\tau_k(x_{1tk} - x_{2tk}) \ge -w$$

$$\sum_{j=1}^{K}\sum_{t=1}^{5}\tau_k(x_{1tk} - x_{2tk}) \le w$$

$$\sum_{j=1}^{K}\sum_{t=1}^{5}\sum_{i=1}^{2}A_k x_{itk} \ge R_l$$

$$\sum_{j=1}^{K}\sum_{t=1}^{5}\sum_{i=1}^{2}A_k x_{itk} \le R_u$$

$$\sum_{k=1}^{K}\sum_{i=1}^{2}x_{itk} = 1, \forall t \in \{1,...,5\}$$

$$\sum_{t=1}^{5}\sum_{i=1}^{2}x_{itk} \le 1, \forall k \in \{1,...,K\}$$

$$x_{itk} = 0, \forall i \in \{1,2\}, t \in \{1,...,5\}, k \in F$$

$$x_{itk} \in \{0,1\}, \forall i \in \{1,2\}, t \in \{1,...,5\}, k \in \{1,...,K\}$$

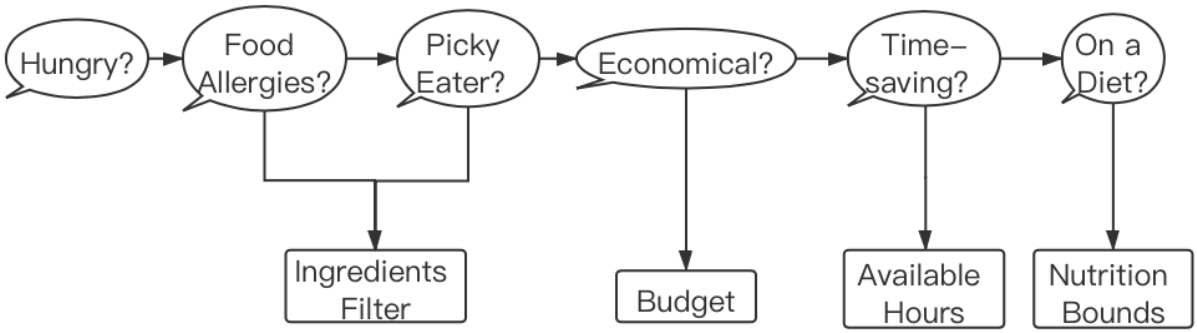
In the objective at the above formula, we maximize the sum of ratings from two users for all recipes being selected and minimize their discrepancy between their total cooking time.

The first constraint makes sure the summation of the cost of the recipes we choose do not exceed our budget limit. The second constraint indicates that every day, someone (either person 1 or person 2) has sufficient time to cook. The reason we want to use third and fourth constraints is that we do not want it is always the person 1 (or person 2) to cook the food. Therefore, we use third and fourth constraints to make sure both of them will cook in these 5 days. The fifth and sixth constraints indicate that the dishes must meet the fixed nutritional requirements and the upper limit of demand, and the vector R represents the nutrition and calories of the dishes, respectively, which means that people cannot consume too much or too little nutrition. The seventh constraint makes sure the diversity of the dishes. The eighth constraint ensures that there is food every day.

Method

Survey: Determine values of Right-Hand Sides

- Input: Survey
- Output: RHS parameters
- Procedure:

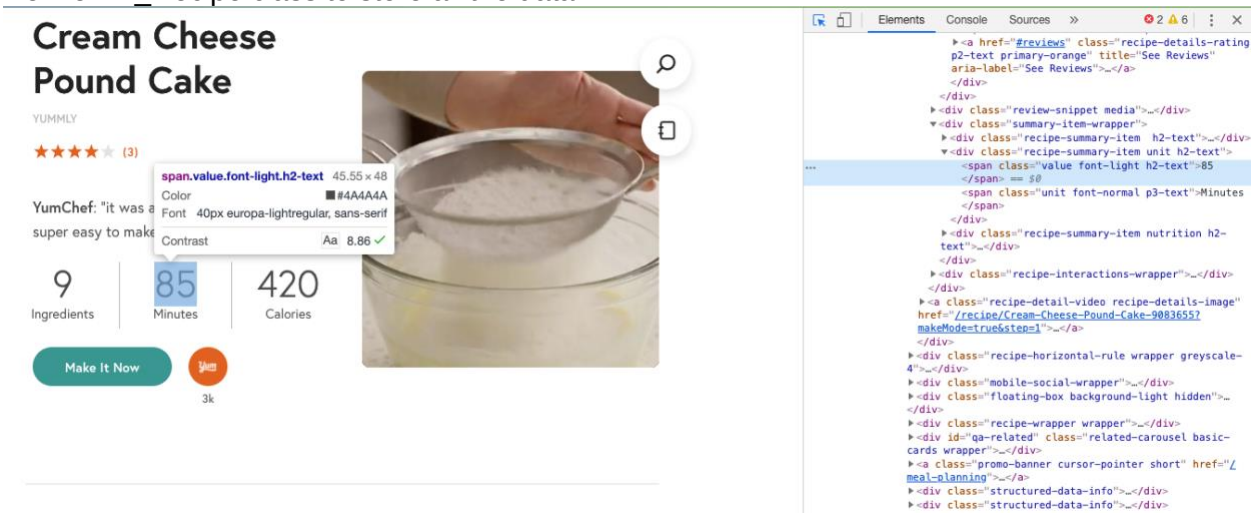


Before coding with the project, we need to do a survey to determine values of right-hand sides in the model. Maybe our friends have an allergy to certain foods, and they may be picky eaters, so we should avoid recommending recipes with these ingredients. Then we should ask for their budget and available hours each day. Also, if they are on a diet, we need to add personalized constraints for their nutrition requirement.

Web Scraping: Collect related recipes data

- Input: https://www.yummly.com/guided-video-recipes
- Output: recipes_data.json
- Tool: Python, chromedriver
- Procedure

1. Define EP_Recipe class to store all the data

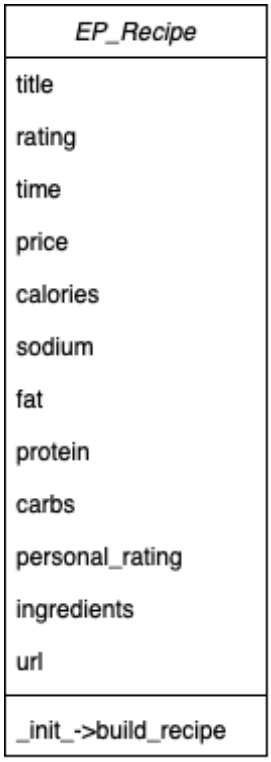


As this picture shows, we inspect a recipe web page, and find the unique location of desired data with the help of class tag. The most difficult feature to be collected is price. The web page shows the price only after clicking the button “shop ingredients”. Thus, with the help of the chromedriver, for each web page, the driver automatically clicks the button and then gets the price.

2. Find all the addresses for recipes and store them in ep_urls

   Again, we need the chromedriver’s help since the website shows only a few recipes before scrolling down. Here, we let the driver scroll down to the end for 20 times to make sure that the number of recipes is enough.

3. Scrape the website of recipes and generate the data. Store the data as recipes_data.json



Data Processing: Store recipes data and construct recipes-user-rating matrix

- Input: recipes_data.json
- Output: recipe_info.csv, final_rating_data.csv
- Tool: Python
- Procedure
    1. Read the data from recipes_data.json
    2. Delete the row with NaN data and only consider the recipes with at least four reviews
    3. Delete the row with “<1” protein or carbs and convert these two features which contain strings due to the existence of ”<1” to numerical values
    4. Combine 50 users' rating as one person's rating

Collaborative Filtering: Fill recipes-user-rating matrix

- Input: final_rating_data.csv
- Output: recipe-user-rating-after-matrix-completion.csv
- Tool: R
- Procedure: Matrix completion via singular value decomposition with the help of softImpute package, that is, predicting missing ratings on the basis of reactions by similar users.

Mixed Integer Programming

- Input: recipe-user-rating-after-matrix-completion.csv, recipe_info.csv, RHS parameters
- Output: Meal recommendation
- Tool: Python, IBM Decision Optimization on Cloud Service
- Procedure:
    1. Rearrange the recipes-user-rating matrix
       The columns order in recipes-user-rating matrix should be matched with the row order in recipe_info.csv, since we need to make sure that for the same index k, the rating[k] is for the recipe[k].

2. Ingredients Filter
   Save all the indices of recipes that contain ingredients without permission, in the model, we set all x[itk] with k= indices to 0.
3. Check tags of recipes
   Save all the indices of recipes that are dessert, in the model, we set all x[itk] with k= indices to 0.
4. Create a model instance
5. Define the variables, parameters, constraints, and objective function
6. Solve the problem
   Here we need to get the service URL and our personal API key by subscribing to DOcplexcloudor Decision Optimization on Cloud solve service. Considering the scale for this problem, the CPLEX Community Edition is not enough to solve since the error "**** Promotional version. Problem size limits exceeded, CPLEX code=1016" will occur.
7. Save the result to a dataframe and display.

Data Sources and Data Science

All recipes data comes from https://www.yummly.com/guided-video-recipes website.
All parameters values in the survey and nutrition bounds in the Scenario 1 come from Ashley and Mico.
Nutrition bounds in the Scenario 2 come from [Appendix 7. Nutritional Goals for Age-Sex Groups Based on Dietary Reference Intakes and Dietary Guidelines Recommendations.] in [US Department of Health and Human Services. (2017). Dietary guidelines for Americans 2015-2020. Skyhorse Publishing Inc..] [https://health.gov/our-work/food-nutrition/2015-2020-dietary-guidelines/guidelines/appendix-7/]
Nutrition bounds in the Scenario 3 come from https://www.nhlbi.nih.gov/health/educational/lose_wt/eat/menus_asian.htm.

After extracting the ratings and raw materials on the website, we quantified it according to the needs of our unit and converted it into the format we need. For example, we changed the stars indicating rating on the web page to specific integers.

At the same time, after crawling the data, we found that some of the data did not comply with the specifications. For example, the price of a small portion of the data could not be captured due to network problems. We removed these NaN data. For some nutrients, we delete the recipes with string type data to keep all nutrients data being numerical values. Also, we only consider the recipes with at least four reviews.

Discussion of Results

After the survey, we got these parameters' values from Ashley and Mico.

Person/Available Time	Mon	Tue	Wed	Thu	Fri

Ashley	200	50	100	50	200
Mico	0	100	100	100	100

B = 120 #budget
alpha = 0.1 #parameter for inequity in objective function
allergy = "salmon"
picky = "peanut" #dislike

Scenario 1: Personalized Choice

Nutrition Bounds Choice

Nutrition	calories	protein	fat	sodium	carbs
Lower bound	2700	140	150	2400	50
Upper bound	3600	200	250	3000	300

Result

Objective value : 47.850
w=6.000 # inequity

	person	day	title	url	price	calories	time
0	1	1	Herb & Garlic Roasted Chicken	https://www.yummly.com/recipe/Herb-_-Garlic-Roasted-Chicken-2696765	19.25	550.0	100.0
1	1	5	Cheesy Egg And Hashbrown Casserole	https://www.yummly.com/recipe/Cheesy-Egg-And-Hashbrown-Casserole-9029559	12.15	300.0	50.0
2	2	2	Garlic-Butter Baked Chicken Breasts with Cheese	https://www.yummly.com/recipe/Garlic-Butter-Baked-Chicken-Breasts-with-Cheese-9092202	21.47	300.0	35.0
3	2	3	Basic Roasted Chicken	https://www.yummly.com/recipe/Basic-Roasted-Chicken-2376893	18.99	570.0	100.0
4	2	4	Restaurant-Style Loaded Chicken Nachos	https://www.yummly.com/recipe/Restaurant-Style-Loaded-Chicken-Nachos-9029506	42.43	990.0	21.0

Scenario 2: Dietary Guidelines Recommendations

Nutrition Bounds Choice

Nutrition	calories	protein	fat	sodium	carbs
Lower bound	1200	25	22	2200	112
Upper bound	3000	88	39	2400	163

Result

Objective value : 49.863
w=0 # inequity

	person	day	title	url	price	calories	time
0	1	1	Vegan Baked Ziti	https://www.yummly.com/recipe/Vegan-Baked-Ziti-2684090	34.57	360.0	75.0
1	1	4	Garlic Roasted Broccoli	https://www.yummly.com/recipe/Garlic-Roasted-Broccoli-2684142	7.47	130.0	30.0
2	1	5	Easy Honey-Mustard Chicken Thighs	https://www.yummly.com/recipe/Easy-Honey-Mustard-Chicken-Thighs-9029460	19.60	310.0	30.0
3	2	2	Vegan Chickpea "Meatloaf"	https://www.yummly.com/recipe/Vegan-Chickpea-_Meatloaf_-9083673	32.35	380.0	75.0
4	2	3	Vegan Black Bean Enchiladas	https://www.yummly.com/recipe/Vegan-Black-Bean-Enchiladas-9116010	24.05	100.0	60.0

This result is perfect with regard to equity in time. Also, notice that the objective value is high enough considering the full rating is 50.

Scenario 3: Lose Weight

Nutrition Bounds Choice

Nutrition	calories	protein	fat	sodium	carbs
Lower bound	700	25	22	348	46
Upper bound	1220	53	30	1043	138

Result

Objective value : 42.890

w=4 # inequity

	person	day	title	url	price	calories	time
0	1	1	Easy White Cheese & Garlic Pizzas	https://www.yummly.com/recipe/Easy-White-Cheese-_-Garlic-Pizzas-9029496	32.54	110.0	20.0
1	1	5	Vegan Chickpea "Meatloaf"	https://www.yummly.com/recipe/Vegan-Chickpea-_Meatloaf_-9083673	32.35	380.0	75.0
2	2	2	Honey Balsamic Glazed Brussels Sprouts	https://www.yummly.com/recipe/Honey-Balsamic-Glazed-Brussels-Sprouts-2684149	16.89	130.0	30.0
3	2	3	Creamy Spinach Stuffed Mushrooms	https://www.yummly.com/recipe/Creamy-Spinach-Stuffed-Mushrooms-2483585	17.99	70.0	33.0
4	2	4	Dairy-Free Herb-Stuffed Mushrooms	https://www.yummly.com/recipe/Dairy-Free-Herb-Stuffed-Mushrooms-9029483	19.39	20.0	36.0

Summary

Lessons Learned

From this project, we learned how to scrape websites with Python and BeautifulSoup, put the basic theory of Singular Value Decomposition to do rating prediction in practice with R and softImpute, and how to program mixed-integer problem with Python and Docplex. Furthermore, since this project has multiple steps coded in three different files, we understand the importance of teamwork and considering a problem comprehensively at the macroscopic level.

However, during this project, we found that there are several things we could have done to avoid some issues. Firstly, the learning of the new tool chrome driver took too much effort. Every time we check the correctness of our program, it will take several minutes to several hours to get the result. Also, we failed to run our program on the colab at first, after multiple attempts, we accidentally realized that the same code could be run successfully locally. All these wasted our time. Next time if we encountered such a difficult problem, we might try to change a method or source, thus improving our work efficiency.

Future Work

- GUI
  In this project, every time we want to change the scenario, we need to modify the parameters in the MnM_MIP.ipynb and then run it. This is not a user-friendly way. A black-box GUI is a more reasonable strategy. That is, a user just types in their budget, schedule, nutrition requirement type(lose weight, keep fit or strengthen muscles) and

personal diet requirement (allergy and dislike). Then, with one-click, the GUI will display recommending recipes information.

- Expand sources of recipes
  Besides the yummly.com website, we can explore more websites with detailed recipes data. Actually, there are two shortcomings for yummly. One is the indirectly shown price. Right now, we need to use chromedriver to imitate the click action, which costs a large amount of time considering the number of recipes. The other is the sparse recipes-user-rating matrix because most users only rate one recipe. Therefore, if we could find such a website overcoming the disadvantages mentioned before, the performance(accuracy for recommendation and time-consuming) will extremely increase. Even though we could not find such a website, more recipes collected will also increase our accuracy for recommendation at the cost of time.
- Convert the number of days and the number of people to parameters rather than a constant
  Currently, we set the day to 5 and people to 2, but if we let the user determine the quantity of day and people, then the flexibility and applicability of this recommendation system will heavily increase.