**Preference-Based Daily Diet Optimization**

A dissertation presented

by

Baihan Lin

Daehyun Kim

Xinyuan Liu

Yijun Ma

to

Dr. Matthew Conroy

Department of Mathematics

in partial fulfillment of the requirements for

MATH 381: Discrete Mathematical Modeling

in the subject of

Mathematics

University of Washington

Seattle, Washington, USA

Oct 28th 2016

**Table of Contents**

**1. Introduction** …………………………………………………………….………… 3

**2. Background** ………………………………………………………………….…….. 3

**3. The Model Itself**  .…………...……………………..……………………………..... 4

**4. Solution** ...………..……………………………………………………………….. 12

**5. Commentary on Solution** ……...………………………………………………... 15

**6. Variations** ….…………………………………………………………………….. 17

**7. Conclusion**  ……..……………………………………………………………....... 24

**8. References** .……………...……….……………………………………………….. 25

**9. Appendix** ...……….………………………………………………………………. 26

**1. Introduction**

In this project, our team is doing a diet problem. In particular, we are modeling a daily diet problem to maximize the happiness of every group member in our group. That is, in hope of getting the greatest joy and a health body, from a possible diet we can eat every day, meeting all the nutrition levels each person needs every day and controlling our daily calorie and sodium intakes. In this project, we used the linear programming as our method to solve our model and get an optimal solution for each one in our group. Specifically, we give advice for our group members on what recipe or what combination of food we should eat to make our week the most cheerful, based on the given solution to the linear program of our model.

**2. Background**

When we started discussing the possible problems we want to solve, we came up with several ideas. For instance, we discussed about the sports league ranking optimization that treats seasonal budgets and athletes’ skill levels as constraints to maximize the team’s rank, the students working scheduling problem based on working efficiency, students’ available time, their preference time to work, and their current class credits, and also the movie festival problem to determine possible list for movies on screen to maximize the preference of audience within a certain time available and the cost constraint. However, before we went further with any of these topics, we found that we would like to solve a problem that was closer to our daily life. That is, we were more interested in working on a topic whose result can directly affect our life. Hence, a new topic came to our mind -- the daily diet preference optimization. As students, we wish to handle the busy school work as well as keep fit. Having a healthy diet is critical for us, not only controlling the calories and fats we take in every day but also ensuring the food we eat contains enough amount of essential nutrients, such as several kinds of vitamins, proteins, Calcium, and so on. On the other hand, we probably do not want to eat food that we dislike but is indeed low in calories and high in protein and fiber. And thus, we wish to have a diet that makes us the happiest and achieves our fitness goal at the same time.

Our diet preference maximization problem is basically a knapsack problem. According to the Wikipedia, the knapsack problem is a combinatorial optimization problem. We have to determine from a set of items, each with a weight and a value, the amount of each item to include in the knapsack so that the total weight is less than or equal to a given limit and the total value is as large as possible [11]. The similar thing is that, both our diet preference problem and the knapsack problem are designed to figure out an optimal solution to maximize the total value we wish to get without exceeding the certain constraints. The constraints in the knapsack problem are the limit of the sum weight of the items. In our diet preference maximization problem, we cannot let some kinds of elements, like fats and calories, that we take in each day to exceed upper limits, but we also have to ensure that our daily ingestion of essential nutrients reaches the necessary amounts. Furthermore, the daily nutrition levels for male and female are not identical, so we have to deal with them in separate sets. Hence, our constraints on the intakes of the food are far more complicated than the weight constraint of the items to put in the knapsack.

Besides, our diet preference optimization is related to diet problem. The diet problem is a classic application of linear programming. It was one of the first optimization problems researched in the 1930s and 1940s [4]. The main goal is to select different types of food to meet daily nutritional requirements at the minimum cost. Geroge Stigle, one of the earliest researchers studying the problem, came up with an optimization problem called the Stifler diet. He wanted to find the amount of each of 77 items of food to be eaten per day in order to at least meet the requirements of dietary allowances suggested by National Research Council with minimal cost [12]. He used heuristic methods to get the optimal solution at $39.69 in 1939. Both of diet problem and diet preference optimization fulfill the requirements of different nutrients. However, diet preference optimization is not exactly the same as Stifler diet. While Stifler diet aimed to minimize the cost of food with the constraints of nutrition, our diet preference optimization aims to maximize people’s happiness from having meals with the constraints of nutrition. We define happiness in different categories of food in a Likert scale of 1 to 10, in which 1 implies total dislike and 10 implies total love. Besides, the sample size and sample objects are distinct. Stifler diet‘s sample object is male, and it used 77 items of food to fulfill the nutrition requirement. On the other hand, in our diet preference optimization, we separate sample objects to male and female and we conducted a survey of preferences in our group. Also, the variety of food is as large as 2835 items of food.

Before digging into the process of generating code, we first downloaded an integrated dataset of the composition of foods on the nutrient contents from Public Health England’s governmental website<https://www.gov.uk/government/publications/composition-of-foods-integrated-dataset-cofid> [5]. This entire dataset actually categorizes food into hundreds of different kinds of food groups. We then conducted a survey on the preference of different categories of food in a Likert scale of 1 to 10 within our group.

**3. The Model Itself**

**The Mathematical Model**

The mathematical model we have developed for investigating the diet preference optimization problem was linear programming. Linear programming has an objective function in terms of optimized variables subject to some constraints those variables are linearly related to. For our diet preference optimization model, our objective was to have the maximum preference rating from the optimized set of food items. The amount of each item, therefore, represented each of our variables as . The unit of the amount was in 100 grams because the nutrient data we had was in per 100 grams of food. Each of 2835 food items on the data list was already classified into different food groups by the source. Since it is extremely time-consuming for each member to rate each food item, we decided to rate each of 135 food groups. These ratings were then averaged based on gender due to the difference in nutrient requirements for each gender. These averaged ratings became our coefficients for the objective funtion as , after the automated process - to be discussed in the solution section - to assign each food group’s rating to corresponding food items. The final objective funtion was to maximize the sum of the products of each item’s preference rating and amount of each item in 100 grams as below, in which **H** is defined as the happiness score from the diet:

**H =**

As for the constraints the objective funtion is subject to, we had various nutrient constraint constants excerpted from several websites. The sum of the specific nutrient values among the optimized set could not be smaller than the minimum nor larger than the maximum amount of the daily recommended value for that nutrient. Most of these nutritional constraints were different for each gender because each gender has a different calorie requirement each day, and most constraints are related to the calorie consumption. The calorie requirements for each gender were taken from U.S. Department of Agriculture website [8]. The calorie requirement for moderately active men was from 2200 to 2800kcal, and that for moderately active women was from 1800 to 2200kcal. Using this and the relationship between calorie consumption and daily recommended ranges of nutrients from the websites “SF Gate” and the American Heart Association, we could find the maximum and minimum amounts of each nutrient that is related to calorie [2][1]. For example, 10% to 35% of daily calories has to be from protein, which is 4 grams per kcal. If we convert this relationship, we get that 55 to 245 grams of protein have to be consumed daily for men, 45 to 192 grams for women. Likewise, we could build constraint inequalities for all nutrients that are related to calories such as protein, fat, carbohydrates, saturated fats, and trans-fats. For the other nutrient requirements or limits that are not related to calories such as cholesterol, sodium, potassium, calcium, and all the vitamins, the maximum and/or minimum values for each nutrient could be sourced from other websites such as “ConsumerLab.com” [6]. These values could be used as common values for each gender. The resulting contraints of the model are set for each nutrient individually and are listed as follows:

where is the amount of each nutrient in 100 grams of each food , and

and are the maximum and minimum values for each nutrient.

One of the final constraints was to limit the amount of beverages and alcohol beverages in the optimized food set under 96 ounces according to “SparkPeople” and this is approximately 3000 grams [7]. Another was to limit the amount of each food item in the optmized set under 3000 grams because more than that amount would be unrealistically excessive of the same food. These two types of constraints can be described as below:

for in the group of beverages and alcoholic beverages

where = 30 (hunderd grams of food)

(i = 0, 1, 2 … , 2834)

**Implementation of the Model**

In order to get the preference parameters for the objective function, a survey was conducted on the preference on different food groups. The survey took samples within our team. The preference was defined as a score of 1 to 10 in a Likert scale, in which 1 implies total dislike and 10 implies total satisfaction. Considering the potential physical difference and psychological disparity, this preference dataset based on satisfaction was furthered divided into the preference scores for male and female.

In order to prepare our datasets for further linear programming, we rearranged the data into a single graph with each row indicating an individual type of food. In our final arranged datasheet, these 2835 possible food items featured in the McCance and Widdowson’s dataset each consists of a row with nutrition information on each column.

In order to integrate preference data to correspond each individual food item, we wrote several Bash codes to automate the process in Mac OSX terminal. Preference\_Switch.sh (Appendix) generated proper data format of two columns of preference data for male and female corresponding to each food item. These two columns are later copied and pasted into Excel sheet of all data.

Preference\_Switch.sh consists of two parts: part I aims to generate pattern matching arguments for part II, with input files FoodGroup.list, pref.list, prefMF.list and generate output file pref\_ind\_switch\_final.sh (Appendix). FoodGroup.list contains the list of food group abbrevations in 2835 items in alphabetical order of the food names. pref.list contains the list of sorted food group abbrevations. prefMF.list contains the list of sorted food group abbrevations, as well as the preference scores on these food groups for male and female. pref\_ind\_switch\_final.sh was further copied and pasted into part II for Preference\_Switch.sh to match the preference scores to each food item based on their food group abbreviation. After these rearrangements, the initial dataset was ready for next step’s calculation (Figure 1).

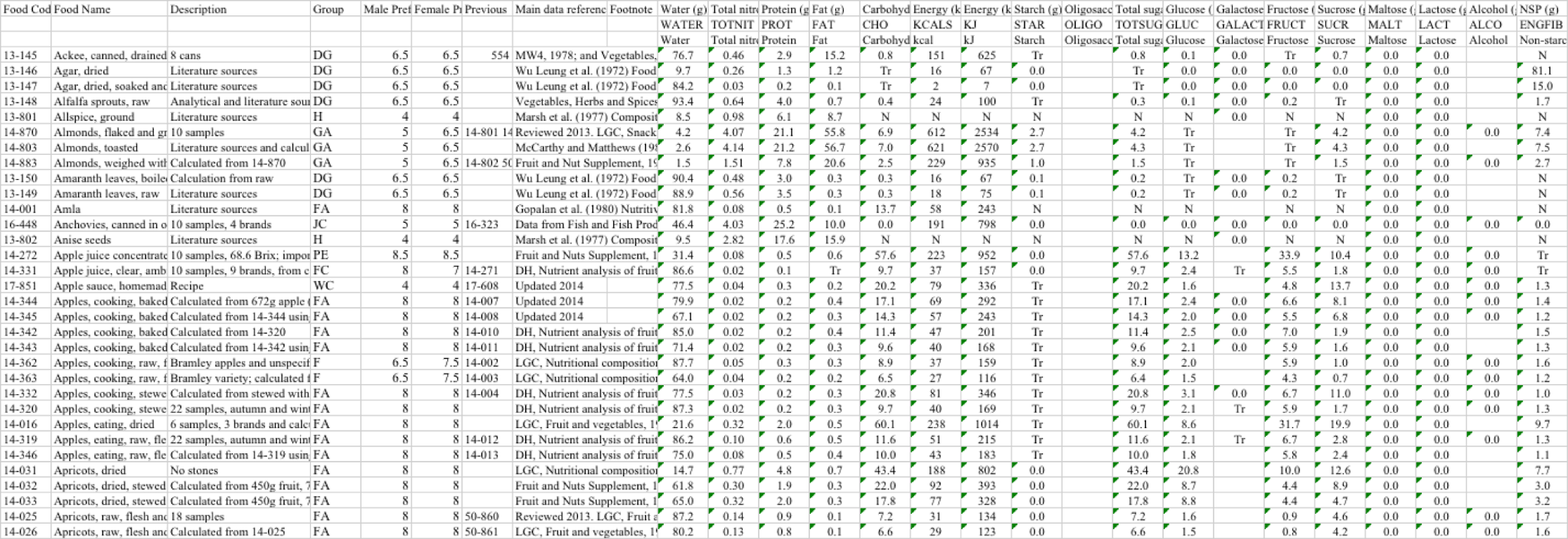


Figure 1. An extraction of the data sheet.

In order to generate the lpsolve script file for lp\_solve as input based on our model, we wrote several Python (3.5.2) codes (which can be run in Windows, Mac, Linux) to automate the process. lp\_diet.py extracted the necessary data from the CSV file using the CSV package and placed the data into a list of lists. This list of lists would essentailly represent a table with each row containing appropriate data for each food item. The gender variable was set as 2 for male, or 3 for female. The code then generated the objective of the Linear Programming script using a for loop that loops through the data table to extract the preference rating for each item depending on the gender variable.

As for the constraints, the minimum and maximum values for each nutrient for each gender were set as variables to certain constants according to the websites discussed above. Then a for-loop looped through the data table for each nutrient constraint, to print out the corresponding inequalities. At the end, lp\_diet.py looped through each variable from male and female preferences, protein (g), fat (g), carb (g), energy (kcal), sat\_fat (g), trans\_fat (g), cholesterol (mg), sodium (mg), potassium (mg), calcium (mg), Vitamin D (mcg), Vitamin E (mg), Vitamin B6 (mg), Vitamin B12 (mcg), and Vitamin C (mg), and the output lpsolve script lp\_diet\_\*.txt (Appendix) was generated with the objective and the constraints encoded.

Then we used lpsolve software [3] to solve our model, and saved its output into a seperate file lp\_output\_\*.txt (Appendix). This process had some challenges because lpsolve software initially failed to give a desired output saying the lp is unbounded. We realized this resulted from our model not having the upper limit for the calories. Because most of the other nutrient constraints were linearly related to the calorie constraints, not having the upper boundary for calories would have caused the optimization problem being unbounded. After we provided a maximum value to the calorie requirements for each gender and changed the other nutrient constraints that are related to calories, the lpsolve software produced a desired output. It took less than a second to generate the LP script as well as the output file. Further descriptions of the coding methods are below in the actual code as well as other supporting files in the Appendix:

**(code) lp\_diet.py**

# The following python code can be run to

# generate an lpsolve script file as outpt, like this:

# python lp\_diet.py > lp\_diet\_\*.txt

#

# It will write a LP script file that can be run on the command line

# to generate an lpoutput file, like this:

# lp\_solve lp\_diet\_\*.txt > lp\_output\_\*.txt

import csv

# checks if the string represents a number (int/float)

# this function will be used to find out whether an element

# is a number or something else like a string.

def is\_number(s):

try:

float(s)

return True

except ValueError:

return False

gender = 2; # int of gender: 2 - male, 3 - female

# open the csv file annd excerpt data from the file

with open('./NewNew\_Data.csv') as f:

reader = csv.reader(f)

next(reader) # skip headers (3 lines)

next(reader)

next(reader)

data = [] # stores necessary food data (going to be a list of lists)

for row in reader:

# stores each item's

# Food Name, Group Name, male prefrence, female preference, protein(g), fat(g),

# carb(g), energy(kcal), sat fat(g), trans fat(g), cholesterol(mg), sodium(mg),

# potassium(mg), calcium(mg), Vitamin D(mcg), Vitamin E(mg), Vitamin B6(mg),

# Vitamin B12(mcg), Vitamin C(mg)

item = [row[1], row[3], row[4], row[5], row[11], row[12], row[13], row[14], row[29], row[47],

row[48], row[49], row[50], row[51], row[64], row[65], row[72], row[73], row[77]]

# append each row(item) to the list

data.append(item)

# Change non-number elements of the columns that should contain numbers such as Tr and N to 0.0

for i in range(len(data)):

for j in range(2, 19):

if (is\_number(data[i][j]) == False):

data[i][j] = 0.0

# print out objective function which optimizes the preference rating based on gender

print('max:', end='')

for i in range(len(data)):

print('+' , data[i][gender], 'x\_', i, sep='', end='')

print(';')

print('')

# Nutrient Requirements

# male (age 19-30) calorie requirement: 2200-2800kcal

# female (age 19-30) calorie requirement: 1800-2200kcal

# protein: 4 grams per kcal, 10% to 35% of daily calories

# male protein requirement: 55g to 245g

# female protein requirement: 45g to 192.5g

# fat: 9 grams per kcal, 20% to 35% of daily calories

# male fat requirement: 48.9g to 108.9g

# female fat requirement: 40g to 85.6g

# carbs: 4 grams per kcal, 45% to 65% of daily calories

# male carb requirement: 247.5g to 455g

# female carb requirement: 202.5g to 357.5g

# saturated fat: up to 7% of daily calories

# male: limit to 21.8g

# female: limit to 17.1g

# trans fat: up to 1% of daily calories

# male: limit to 3.1g

# female: limit to 2.4g

# cholesterol: limit to 300mg

# sodium: limit to 2400mg

# potassium: from 4700mg

# calcium: from 1000mg to 2500mg

# Vitamin D: 600 IU to 4000 IU, 15mcg to 100mcg

# Vitamin E: 22 IU to 1500 IU, 0.55mcg to 37.5mcg, 0.00055mg to 0.0375mg

# Vitamin B6: 1.3mg to 100mg

# Vitamin B12: from 2.4mcg

# Vitamin C: male: 90 to 2000mg | female: 75 to 2000mg

# According to the nutrient requirements above, set each nutrient's

# daily recommended minimum and maximum values based on gender

if gender == 2:

calorie\_min = 2200

calorie\_max = 2800

protein\_min = 55

protein\_max = 245

fat\_min = 48.9

fat\_max = 108.9

carb\_min = 247.5

carb\_max = 455

sat\_max = 21.8

trans\_max = 3.1

VitC\_min = 90

elif gender == 3:

calorie\_min = 1800

calorie\_max = 2200

protein\_min = 55

protein\_max = 192.5

fat\_min = 48.9

fat\_max = 85.6

carb\_min = 247.5

carb\_max = 357.5

sat\_max = 17.1

trans\_max = 2.4

VitC\_min = 75

choles\_max = 300

sodium\_max = 2400

potassium\_min = 4700

calcium\_min = 1000

calcium\_max = 2500

VitD\_min = 15

VitD\_max = 100

VitE\_min = 0.00055

VitE\_max = 0.0375

VitB6\_min = 1.3

VitB6\_max = 100

VitB12\_min = 2.4

VitC\_max = 2000

bev\_max = 30

amount\_max = 30

# print out lp constraints for each daily nutrient requirement

# calorie requirement

for i in range(len(data)):

print('+', data[i][7], 'x\_', i, sep='', end='')

print('>=', calorie\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][7], 'x\_', i, sep='', end='')

print('<=', calorie\_max, ';', sep='')

print('')

# protein requirement

for i in range(len(data)):

print('+', data[i][4], 'x\_', i, sep='', end='')

print('>=', protein\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][4], 'x\_', i, sep='', end='')

print('<=', protein\_max, ';', sep='')

print('')

# fat requirement

for i in range(len(data)):

print('+', data[i][5], 'x\_', i, sep='', end='')

print('>=', fat\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][5], 'x\_', i, sep='', end='')

print('<=', fat\_max, ';', sep='')

print('')

# carbs requirement

for i in range(len(data)):

print('+', data[i][6], 'x\_', i, sep='', end='')

print('>=', carb\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][6], 'x\_', i, sep='', end='')

print('<=', carb\_max, ';', sep='')

print('')

# saturated fat requirement

for i in range(len(data)):

print('+', data[i][8], 'x\_', i, sep='', end='')

print('<=', sat\_max, ';', sep='')

print('')

# trans fat requirement

for i in range(len(data)):

print('+', data[i][9], 'x\_', i, sep='', end='')

print('<=', trans\_max, ';', sep='')

print('')

# cholesterol requirement

for i in range(len(data)):

print('+', data[i][10], 'x\_', i, sep='', end='')

print('<=', choles\_max, ';', sep='')

print('')

# sodium requirement

for i in range(len(data)):

print('+', data[i][11], 'x\_', i, sep='', end='')

print('<=', sodium\_max, ';', sep='')

print('')

# potassium requirement

for i in range(len(data)):

print('+', data[i][12], 'x\_', i, sep='', end='')

print('>=', potassium\_min, ';', sep='')

print('')

# calcium requirement

for i in range(len(data)):

print('+', data[i][13], 'x\_', i, sep='', end='')

print('>=', calcium\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][13], 'x\_', i, sep='', end='')

print('<=', calcium\_max, ';', sep='')

print('')

# Vitamin D requirement

for i in range(len(data)):

print('+', data[i][14], 'x\_', i, sep='', end='')

print('>=', VitD\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][14], 'x\_', i, sep='', end='')

print('<=', VitD\_max, ';', sep='')

print('')

# Vitamin E requirement

for i in range(len(data)):

print('+', data[i][15], 'x\_', i, sep='', end='')

print('>=', VitE\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][15], 'x\_', i, sep='', end='')

print('<=', VitE\_max, ';', sep='')

print('')

# Vitamin B6 requirement

for i in range(len(data)):

print('+', data[i][16], 'x\_', i, sep='', end='')

print('>=', VitB6\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][16], 'x\_', i, sep='', end='')

print('<=', VitB6\_max, ';', sep='')

print('')

# Vitamin B12 requirement

for i in range(len(data)):

print('+', data[i][17], 'x\_', i, sep='', end='')

print('>=', VitB12\_min, ';', sep='')

print('')

# Vitamin C requirement

for i in range(len(data)):

print('+', data[i][18], 'x\_', i, sep='', end='')

print('>=', VitC\_min, ';', sep='')

print('')

for i in range(len(data)):

print('+', data[i][18], 'x\_', i, sep='', end='')

print('<=', VitC\_max, ';', sep='')

print('')

# Beverages limit

for i in range(len(data)):

# if the group name starts with P or Q, which means the item is

# either the beverage or alcohol, add the variable to the constraint

if (data[i][1].startswith('P') or data[i][1].startswith('Q')):

print('+x\_', i, sep='', end='')

print('<=', bev\_max, ';', sep='')

print('')

# amount of food limit

for i in range(len(data)):

print('+x\_', i, '<=', amount\_max, ';', sep='', end='')

print('')

**4. Solution**

Through lpsolve, we got long outputs. Here is an extraction from lp\_output\_female.txt:

Value of objective function: 1639.81681676

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

In order to give us a better idea of what these outputs suggests about our diet, we wrote lp\_output\_process.py to process our output files by extracting all the non-trivial items from the output with non-zero amount of food, and printing out their corresponding food names, group names, and the lpsolve output. The output from this step, lp\_trimmed\_\*.txt, is our final solution for our question; in another words, it is the suggested diet that maximizes our happiness or satisfaction under specified nutritional constraints.

From the trimmed output, we found that, in order to maximize the happiness, an optimal daily diet for female consisted of chicory, cranberries, fennel, gourd, jackfish, jelly, lemon juice, limes, marrow, mushroom, nutmeg, passion fruit, orange juice, rock salmon and tea, with a maximum happiness score **H**max= 1639.81681676. An optimal daily diet for male consisted of chicory, cranberries, fennel, duck, gourd, jackfish, jelly, lemon juice, lime juice, marrow, mushroom, nutmeg, passion fruit, orange juice, rock salmon and tea, with a maximum happiness score **H**max= 1794.33880959. The specific food names and amount of each item to be consumed in 100 grams of food are listed in the trimmed output below.

**(output) lp\_trimmed\_male.txt**

Value of objective function: 1794.33880959

Actual values of the variables:

Food Name Group Name Variable 100g of food

Chicory, pale variety, boiled in unsalted water DG x\_716 30

Cranberries FA x\_853 30

Duck, raw, meat only, weighed with fat, skin and bone MCC x\_1049 2.94369

Fennel, Florence, boiled in unsalted water DG x\_1089 8.29889

Fruit juice drink/squash, no sugar added, diluted PCC x\_1162 30

Gourd, ridge, raw DG x\_1199 21.6347

Jackfish, raw JC x\_1316 0.46012

Jelly, sugar free, made with water BR x\_1327 30

Lemon juice, fresh, weighed as whole fruit FC x\_1492 30

Lime juice, fresh FC x\_1520 15.6523

Marrow, boiled in unsalted water DG x\_1575 30

Mushrooms, white, raw DG x\_1699 3.7122

Orange juice, ambient, UHT FC x\_1760 0.0189024

Passion fruit, flesh and pips, weighed with skin FA x\_1817 30

Rock Salmon/Dogfish, raw JA x\_2346 0.990351

**(output) lp\_trimmed\_female.txt**

Value of objective function: 1639.81681676

Actual values of the variables:

Food Name Group Name Variable 100g of food

Chicory, pale variety, boiled in unsalted water DG x\_716 30

Cranberries FA x\_853 30

Fennel, Florence, boiled in unsalted water DG x\_1089 9.78434

Gourd, ridge, raw DG x\_1199 2.39609

Jackfish, raw JC x\_1316 0.809946

Jelly, sugar free, made with water BR x\_1327 30

Lemon juice, fresh, weighed as whole fruit FC x\_1492 30

Limes, flesh only, weighed with peel and pips FA x\_1522 20.8073

Marrow, boiled in unsalted water DG x\_1575 30

Mushrooms, white, raw DG x\_1699 3.72073

Nutmeg, ground H x\_1731 0.269869

Orange juice, ambient, UHT FC x\_1760 0.0146341

Passion fruit, flesh and pips, weighed with skin FA x\_1817 25.8045

Rock Salmon/Dogfish, raw JA x\_2346 0.490755

Tea, black, infusion, average PAA x\_2647 30

**(code) lp\_output\_process.py**

# The following python code can be run with

# an lpsolve output file as input, like this:

# python lp\_output\_process.py < lp\_output\_\*.txt > lp\_trimmed\_\*.txt

#

# It will extract the lines giving the non-zero values

# and prints out the corresponding food name, group name, and the lp output

import fileinput

import csv

import re

# open the csv file and extract the necessary data into a list of lists

with open('./NewNew\_Data.csv') as f:

reader = csv.reader(f)

next(reader) # skip headers

next(reader)

next(reader)

data = [] # stores necessary food data

for row in reader:

# stores each item's

# Food Name, Group Name, male prefrence, female preference, protein(g), fat(g),

# carb(g), energy(kcal), sat fat (g), trans fat(g), cholesterol(mg), sodium(mg),

# potassium(mg), calcium(mg), Vitamin D(mcg), Vitamin E(mg), Vitamin B6(mg),

# Vitamin B12(mcg), Vitamin C(mg)

item = [row[1], row[3], row[4], row[5], row[11], row[12], row[13], row[14], row[29], row[47],

row[48], row[49], row[50], row[51], row[64], row[65], row[72], row[73], row[77]]

data.append(item)

# get the results from the input file

for line in fileinput.input():

# deals with the lines that print the actual lp output

if line.startswith('x'):

output = line.split() # split into a variable name and an output value

# find lines that have non-zero values

if output[1] != '0':

# extracts the number of the variable name

var\_num = int(re.search(r'\d+', output[0]).group())

# find the corresponding food name and group name using the variable number

item = [data[var\_num][0], data[var\_num][1], output[0], output[1]]

# align the columns appropriately

print ('{0[0]:<60}{0[1]:<15}{0[2]:<15}{0[3]:<30}'.format(item))

else:

if line.startswith('Actual'):

print(line)

# prints out the header

header = ['Food Name', 'Group Name', 'Variable', '100g of food to consume daily (opt value)']

print ('{0[0]:<60}{0[1]:<15}{0[2]:<15}{0[3]:<30}'.format(header))

print('')

else:

print(line, end='')

**5. Commentary on solution**

Firstly, from the result of the diet preference optimization for male, we found that the value of objective function is **H**max= 1794.33880959, which means male in our group reach the highest level of happiness while fulfilling the requirement of nutrition. There are fifteen kinds of food items that male consume. Among these fifteen items, five kinds are vegetables within food group DG, three kinds are fruit juice within FC, two kinds are general fruit within FA, one kind is duck within MCC, one kind is squash and cordials within PCC, one kind is fatty fish within JC, one kind is puddings and chilled desserts within BR, and one type is white fish within JA. From this data, we can know that approximately half items from the recommended food list are vegetables and fruits. These food items are healthy and contain low level of calories and fat.

In order to ensure diversity in our food list, we set that the amount of each type of food can be consumed up to 3000 grams per day. There are seven kinds of foods that reached the maximum amount of consumption: chicory, cranberries, fruit juice drink/squash, jelly, lemon juice, marrow and passion fruit. Most of them are grouped in vegetable (DG) and fruits (FA). That means male in our group feel happier when they eat these seven kinds of foods than eat others. In fact, these two groups were rated by male at 6.5 and 8, which are relatively high. Meanwhile, we found that male consume less on jackfish, orange juice and rock salmon/dogfish, which are less than 100 grams per day. They are grouped in white fish (JC), fruit juices (FC) and fatty fish (JA). This tells us male are less likely to eat them; however, they have to consume some of them per day in order to meet daily requirements of nutrition.

The interesting thing is that male highly prefer to have lemon juice, but dislike drinking orange juice. However, they still need to consume orange juice, although lemon juice and orange juice are both in fruit juices group. The reason is that orange juice contains more energy and water than lemon juice, which are 146 kj and 89.4g separately. Lemon juice contains only 11 kj energy and 32g water. male need to consume orange juice to get enough energy and water. Besides, the result shows that duck, jackfish, mushroom and rock salmon account for relatively small portion of daily consumption for male, which are fewer than 300g. We found that all of them are raw foods, which indicated that male in our group are less likely to eat raw foods.

Second, base on the result we got for female preference maximization, female can reach a happiness score **H**max= 1639.81681676 with eating fifteen different types of food and meeting the nutrition requirement daily. Out of these fifteen kinds of food, there are five types of general vegetables within group DG, three types of general fruits within group FA, two types of fruit juice within group FC), two kinds of fish and fish products, namely one from fatty fish within group JC and one white fish within group JA, one food of puddings and chilled desserts within group BR, which belongs to milk and milk products group, one food from herbs and spices within group H, and one kind of beverage, powered drinks and essences within group PAA. Obviously, we can see that two-thirds of the recommended food list contains vegetables and fruits. From the nutrition data of these food, we found that they are low in calories and fat. So we can consume more units of them than other foods that have higher levels of nutrients that we want to restrict below certain amounts. The two vegetables are both rich in Potassium, which female are required to have as much as 4700 mg per day and cranberries, fresh lemon juice, passion fruit, and orange juice are high in Vitamin C. Besides, there are several types of food like fish and nuts that can fulfill the proteins intake requirement for female. And also, black tea does not give any fat, which means it meets the nutrition constraints as well as the preference by female in our group. In addition, jelly should not be good for a diet. But since based on our group survey of preference, the two female members rated highly for the happiness they could get by eating desserts, so the recommendation contains sugar free and also water made jelly to increase their joy while restraining the sugar and calories they take in.

Moreover, from the result we got, six types of food items, namely unsalted water boiled pale chicory, cranberries, water made sugar free jelly, fresh lemon juice weighed as whole fruit, unsalted water boiled marrow, and black tea all have the value of 30. Since we have restricted that we cannot have any types of food more than 30 units of 100 grams, or 3000 grams per day, having a solution of 30 units of these six types of food indicates that the female in our group love consuming vegetables, fruits, desserts, and beverages and the six types of food listed above best fulfill the nutrient requirements. If we do not restrain the units of food consumed, perhaps female will eat even more of those six types. In other words, the female still remain healthy, but the diversity of food eating per day will drop.

Another interesting result is that both of the two female member in our group do not like cranberries, which means that they will not raise happiness level by eating cranberries. However, our solution gives that the recommended consumption of cranberries is 30 units, i.e., 3000 grams. This seems to be unhappy since they would be a lot of food that they dislike. We suppose that the reason for the optimal solution to contain this much of cranberries is that our survey placed preference ratings on each food group, not on each individual food item. Therefore, even if they do not like cranberries, because entire fruit group was rated as highly as 8, the model will consider cranberries as an item that enhances preference optimization. This is one of the drawbacks from rating food groups instead of individual items, though it was impossible to rate each of 2384 items in the first place.

In conclusion, in order to reach a happiness level as high as possible and also meet daily nutritional requirements, both male and female in our group should eat 15 types of certain food. They all prefer to eat vegetables and fruits but are less likely to eat raw food. However, male are less suggested to consume fruit juice than are female.

**6. Variations**

Variation 1: Unhappy Case -- minimize our satisfaction but in a healthy way

**(code) lp\_diet\_unhappy.py**

From the original lp\_diet.py, we did some modification by changing max to min in line 47.

# print out objective function which minimizes the preference rating based on gender

print('min:', end='')

for i in range(len(data)):

print('+' , data[i][gender], 'x\_', i, sep='', end='')

print(';')

print('')

According to our original model, we try to maximize our happiness and meet the nutritional requirement at the same time. There also exists another interesting situation that we may want to punish ourselves when we do worse in the exams and works. We can choose eating unhappy as a form of punishment. Besides, sometimes, one has to be harsh on him/herself, and this is the classy way to do so. Thus, in this variation, we try to minimize our happiness and keep enough daily nutrition. Based on our original model, we changed maximizing happiness to minimizing happiness and kept all constraints unchanged.

From the result of male’s and female's preference minimization, we can see the value of objective functions are 8.33620465 and 15.27183148. Compared with maximum happiness scores in the original model, both of the happiness levels in the variation are much lower. This is the result we want to get, which is to minimize our satisfaction in a healthy way. Also, the numbers of food in the optimal food list for male and female are less than original model. In male’s recommended food list, there are only seven kinds of food. Female’s food list contains nine kinds of food. All types of food in the food list for male are different from the ones we got form the original model except orange juice that belongs to the type of fruit juices. This is because orange juice has a large amount of energy and water is essential to male’s body under these two situations. A concrete evidence that indicates this diet indeed punishes male is that it has three different types of breakfast cereal, which was rated by male at 1. In female’s food list, there are two types of food that were in the original model: jackfish and orange juice. This is because jackfish and orange juice mainly provide energy for female body. As similar as the unhappy diet for male, the unhappy diet for female contains the types of food they rated very low, such as cheese within group BL at 1.5 unit and processed milk within group BAR at 2.5 unit.

Besides, both value of male’ and female’ daily consumption are smaller than original model. In the food list for male, the values of five types of food are under 100 grams, one is about 130 grams and one is about 564 grams. Since male in our group are less likely eating these foods, they want to consume less of them naturally. Meanwhile, our model needs to ensure male meet the requirements of nutrition. So, the consumption of each food is low, while at least meeting minimum daily nutritional requirements. It is the same condition in female’ food list.

In conclusion, this variation is an inverse version of our original model with same constraints. Both male and female will consume food that makes them unhappy. As a result, they will eat few types of food and small values of consumption to at least meet the minimum amount of daily nutritional requirements.

**(output) lp\_trimmed\_male\_unhappy.txt**

Value of objective function: 8.33620465

Actual values of the variables:

Food Name Group Name Variable 100g of food to consume daily (opt value)

Breakfast cereal, bran type cereal, fortified AI x\_408 5.64612

Breakfast cereal, honey loops and hoops AI x\_416 1.30292

Breakfast cereal, instant hot oat, plain, raw, fortified AI x\_417 0.0928571

Chervil, dried H x\_635 0.0201999

Creme fraiche, full fat BJC x\_870 0.0115916

Dressing, oil and lemon, homemade WC x\_1031 0.30048

Orange juice, ambient, UHT FC x\_1760 0.0184186

**(output) lp\_trimmed\_female\_unhappy.txt**  
Value of objective function: 15.27183148  
  
Actual values of the variables:  
  
Food Name Group Name Variable 100g of food to consume daily (opt value)  
  
Bloater, flesh only, grilled JC x\_352 0.0197297   
Cheese, Brie, rind only BL x\_579 0.448212   
Cherries, West Indian, flesh only FA x\_634 0.0303948   
Coffeemate, whitener powder BAR x\_802 0.619319   
Courgette, dried DI x\_839 0.151279   
Jackfish, raw JC x\_1316 0.913868   
Milk, skimmed, dried, fortified BAR x\_1635 1.74777   
Orange juice, ambient, UHT FC x\_1760 0.0146341   
Pizza base, raw AE x\_1991 2.00951

Variation 2: Vegetarian Case -- maximize our happiness with a vegeterian diet

**(code) lp\_diet\_veg.py**

From the original lp\_diet.py, we did some modification by adding this constraint in line 282.

# vegetarian diet: no meat or fish

for i in range(len(data)):

# the group name starts with M or J, which means the item is

# either meat or fish

if (data[i][1].startswith('M') or data[i][1].startswith('J')):

print('+x\_', i, sep='', end='')

print('=0;') # the sum of meat and fish varaibles is 0

print('')

Perhaps there are some days that we do not want to have meat or fish, but we indeed wish to keep on a healthy diet. In our original model, we maximized our happiness score based on restrictions considering every kind of food as a possible choice. Now, we modify our model and add some constraints that set the amount of all the food that made with meat and fish to zero. In this case, we could have a vegetarian diet on the days when we are not willing to eat meat and fish.

The outcome gives optimal happiness scores **H**max= 1296.64981001 for female and **H**max= 1454.78168382 for male. The optimal values of our objective function decrease for both male and female. Since our original problem includes meat and fish to maximize our preference and we have raw duck, jackfish, and rock salmon in optimal solution for male, and jackfish and rock salmon in the optimal solution for female, substituting these food will reduce our happiness when we have the same nutrient constraints as before. For both of the results for male and female, the solution also contains a maximum amount of 30 units of sugar free jelly, fresh lemon juice, and marrow. Female do not eat fennel for this new solution, while male increase about 16.4 units of that since they drop about 21.6 units of gourd and have to complement more vegetables. Meanwhile, this solution includes breakfast cereal, reduced fat coconut milk, coffee, and eggs, which we do not have any in the previous result. Although their consumption amounts are less than most of the suggested vegetables and fruits, they are recommend to eat to replenish proteins, carbohydrate, calcium, and cholesterol, which are originally supplied by meat and fish.

In addition, it is interesting that we do not have cranberries for both male and female for fruit in this vegetarian diet. Instead, the solution adds 8.13063 units of fresh lime juice and 10 more units of fresh limes for female. Based on our group survey, since both of the female in our group dislike cranberries and prefer lime juice rather, taking cranberries off the list and add lime juice can actually increase their happiness. However, because we are calculating happiness score based on the group categorization but not on each specific food, our optimal value will not change since cranberries and lime juice are both in one categorization.

Overall, the happiness drops for both male and female if we want to have a vegetarian day. We have to consume more cereals and dairy to fulfill the nutrient need that was provided by meat and fish in our original diet problem.

**(output) lp\_trimmed\_male\_veg.txt**

Value of objective function: 1454.78168382

Actual values of the variables:

Food Name Group Name Variable 100g of food to consume daily (opt value)

Breakfast cereal, Ricicles, Kellogg's AI x\_427 1.5107

Chicory, pale variety, boiled in unsalted water DG x\_716 9.32401

Coconut milk, reduced fat, retail GA x\_768 1.28691

Coffee, cappuccino, latte P x\_790 9.4863

Eggs, duck, whole, raw CA x\_1077 0.401817

Fennel, Florence, boiled in unsalted water DG x\_1089 24.6649

Fruit juice drink/squash, no sugar added, diluted PCC x\_1162 20.5137

Jelly, sugar free, made with water BR x\_1327 30

Lemon juice, fresh, weighed as whole fruit FC x\_1492 30

Lime juice, fresh FC x\_1520 27.4038

Marrow, boiled in unsalted water DG x\_1575 30

Mushrooms, white, stewed in unsalted water DG x\_1700 3.71616

Orange juice, ambient, UHT FC x\_1760 0.0169221

Passion fruit, flesh and pips, weighed with skin FA x\_1817 30

**(output) lp\_trimmed\_female\_veg.txt**

Value of objective function: 1296.64981001

Actual values of the variables:

Food Name Group Name Variable 100g of food to consume daily (opt value)

Breakfast cereal, Ricicles, Kellogg's AI x\_427 1.42263

Chicory, pale variety, boiled in unsalted water DG x\_716 22.7102

Coconut milk, reduced fat, retail GA x\_768 1.2542

Coffee, cappuccino, latte P x\_790 10.0323

Dressing, oil and lemon, homemade WC x\_1031 0.0131536

Eggs, duck, whole, raw CA x\_1077 0.399646

Jelly, sugar free, made with water BR x\_1327 30

Lemon juice, fresh, weighed as whole fruit FC x\_1492 30

Lime juice, fresh FC x\_1520 8.13063

Limes, flesh only, weighed with peel and pips FA x\_1522 30

Marrow, boiled in unsalted water DG x\_1575 30

Mushrooms, white, stewed in unsalted water DG x\_1700 3.72489

Orange juice, ambient, UHT FC x\_1760 0.0125553

Passion fruit, flesh and pips, weighed with skin FA x\_1817 12.0148

Tea, black, infusion, average PAA x\_2647 19.9677

Variation 3: Diverse Case -- maximize our happiness with plenty of diversity of foods

**(code) lp\_diet\_grouplim.py**

From the original lp\_diet.py, we did some modification by adding this constraint in line 289.

# amount of group limit

# create a list containing alphabets in uppercase

groups = string.ascii\_uppercase

# loops through the list to check each alphabet

for j in range(len(groups)):

# binary to check if the alphabet represents one of the groups

group\_bool = 0

for i in range(len(data)):

# prints out the sum of items on each group

if (data[i][1].startswith(groups[j])):

group\_bool = 1

print('+x\_', i, sep='', end='')

# only print this if the letter is one of the groups

if group\_bool == 1:

# the sum of items on each group is less than 1000g

print('<=', group\_max, ';', sep='')

print('')

In our daily life, we sometimes want to eat food to make us happy and try as many different types of food as possible within a day. This action can let us taste various types of food as well as ingest different nutritions from varied food. In this variation, we modify the original model to enlarge our possibility to getting more different types of food. That is, originally both male and female eat food from 8 different groups, but after our modification, male can now consume food from 14 kind of groups and female can consume food from 13 different groups.

We set new constraints on each group of food so that no matter what specific food we eat, the total amount of food for each group will be restricted less than a certain amount. In this case, we bound each group with 10 units as an upper limit. Hence, we cannot eat more than 1000 grams of any food for a single day. From the recommended food list for male, we obeserve that the value of consuming coconut, lager, lemon juice, lemon juice that weighed as whole fruit, lollies and marrow reached 10 units. It shows that male are happy to eat these food, but for taking more different types of food, they can only consume 10 units of these food. It is the same situation in the recommended food list for female. The value of cider, coconut, lemon juice, limes and marrow is up to 10 units, which means that the female members in our group prefer to consume them. For sake of having more different food, female can only consume 10 units of each of them. By doing so, we decrease the total amount of vegetables and fruits for every day and add several new food items in the list. For example, we now have low alcohol cider, coconut milk, roast duck, grilled eel, lollies with real fruit juice, fresh oregano, turmeric, homemade carrot and orange soup, and chicken noodle soup for female. This is a big change in the food list since we only have four out of fourteen specific food identical to our original optimal solution. Besides the white fish, fatty fish, puddings and chilled desserts, general fruits, general vegetables, and herbs and spices, female can eat food in the other five groups: ciders, nuts and seeds, duck, juices (which is a different group from the fruit juices group), and ice cream. Ingesting nuts and seeds, juices, and ice cream can increase female happiness in our group, since their rates on these three groups are relatively high. In particular, female have an average rate of 8.5 in juices, 6.5 in nuts and seeds, and 8.5 in ice cream, out of a maximum rate of 10. Similar things happen for male. They can now ingest ten specific new food with nine new groups to the original problem. As female, the male in our group prefer juice at an average rate of 8.5 out of 10 and have a perfect rate of 10 on ice cream. So adding these two groups to the optimal solution will give them more happiness comparing to eating vegetables with a rating of 4.75.

However, with the limitations on the consumption for each group of food, both the male and the female happiness decrease sharply. The maximum happiness score drop from 1639.81681676 to 558.85880275 for female and from 1794.33880959 to 618.16189370 for male. This is basically because with the several constraints on nutrient levels, we can not eat much ice cream, nuts, or beverage juice since all of them are rich in calories. Moreover, limiting the total consumption for each group indeed decreases the total amount of food we consume in the optimal solution, and thus reduces our happiness. In addition, we suppose another reason for the dropping in our joy is that both of male and female in our group like fruits and fruit juice, with an average rating of 8 and 7.5 out of 10. This new solution to the variation on diversity does not give as much the amount of fruit or fruit juice groups as the one we get in the original problem. Hence, our optimal happiness values drop.

**(output) lp\_trimmed\_male\_grouplim.txt**

Value of objective function: 618.16189370

Actual values of the variables:

Food Name Group Name Variable 100g of food to consume daily (opt value)

Cardamom, ground H x\_517 1.12797

Chicken, stir-fried with rice and vegetables, retail, reheated MR x\_695 0.500352

Coconut milk GA x\_767 10

Duck, roasted, meat only, weighed with fat, skin and bone MCC x\_1052 9.49965

Eel, conger, flesh only, grilled, weighed with bones and skin JA x\_1058 8.84615

Jackfish, raw JC x\_1316 1.15385

Lager, alcohol-free QA x\_1360 10

Lemon juice, fresh PE x\_1491 10

Lemon juice, fresh, weighed as whole fruit FC x\_1492 10

Lollies, with real fruit juice BP x\_1548 10

Marrow, boiled in unsalted water DG x\_1575 10

Oregano, fresh H x\_1768 1.59442

Rice, Thai fragrant, boiled in unsalted water AC x\_2310 2.24167

Soup, carrot and orange, homemade WAA x\_2516 0.251388

Soup, low calorie, canned WAC x\_2529 2.68501

Turmeric, ground H x\_2743 4.1343

**(output) lp\_trimmed\_female\_grouplim.txt**

Value of objective function: 558.85880275

Actual values of the variables:

Food Name Group Name Variable 100g of food to consume daily (opt value)

Cider, low alcohol QC x\_753 10

Coconut milk GA x\_767 10

Duck, roasted, meat only, weighed with fat, skin and bone MCC x\_1052 9.48769

Eel, conger, flesh only, grilled, weighed with bones and skin JA x\_1058 6.35424

Jackfish, raw JC x\_1316 1.15385

Jelly, sugar free, made with water BR x\_1327 2.51637

Lemon juice, fresh PE x\_1491 10

Limes, flesh only, weighed with peel and pips FA x\_1522 10

Lollies, with real fruit juice BP x\_1548 7.48363

Marrow, boiled in unsalted water DG x\_1575 10

Oregano, fresh H x\_1768 2.74731

Soup, carrot and orange, homemade WAA x\_2516 0.625

Soup, chicken noodle, dried, as served WAE x\_2518 2.70183

Turmeric, ground H x\_2743 2.65146

**7. Conclusion**

Through this study of exploring the optimal diet based on preference, we concluded that, in order to maximize our happiness, generally a daily diet should consist of mostly fruits and vegetables. For non-vegetarians, fish seems to be the most nutritious meat type in an optimal daily diet. Debatably, from our results, male seem to be more easily satisfied than female, with maximized happiness scores higher than female in every set. From our variation studies, we found that to make oneself unhappy, breakfast cereal and milk/cheese might be good choices for male and female respectively. For vegetarians, with vegetables and fruits they could never be alone and sick. Having a diverse diet can help introducing a more balanced diet while maintaining a high happiness level.

Despite these interesting findings, we have to admit that there are some limitations. First, our preference scores for male and female groups are solely based on the small survey within our group with a sample size of 4. This small sample cannot be representative of the entire male and female populations. Further study can be conducted based on a survey with a much bigger sample size across different populations, and in that case, the generalizability of our model can greatly increase. Second, the survey was inevitably focused on each food group, instead of each individual food item, which resulted in some food item that one actually dislikes being rated highly. This can be fixed with more time and patience of the survey subjects to actually rate each item. Last but not least, we based our model on the dataset by Public Health England, which covers a wide variety of food types but cannot cover every possible food in the world espeically the U.S. we currently live in. This may, in a very rare chance, imply possible better solutions.

Even with limitations due to our small sample size and dataset selection as described above, our model still provides profound insights into diet planning. With our model, not only can the suggested diets be healthy, they can bring satisfaction and happiness into people’s daily life. Therefore, we feel our efforts into this project are very meaningful and rewarding.

**8. References** (in the alphabetical order)

1. American Heart Association, “Know Your Facts” <http://www.heart.org/HEARTORG/Conditions/Cholesterol/PreventionTreatmentofHighCholesterol/Know-Your-Fats_UCM_305628_Article.jsp>
2. ConsumerLab, “Recommended Daily Intakes and Upper Limits for Nutrients” <http://www.consumerlab.com/RDAs/>
3. Sourceforge, “lpsolve” <https://sourceforge.net/projects/lpsolve/>
4. NEOS, “The Diet Problem” <http://www.neos-guide.org/content/diet-problem>
5. Public Health England, “McCance and Widdowson’s The Composition of Foods Integrated Dataset 2015” <https://www.gov.uk/government/publications/composition-of-foods-integrated-dataset-cofid>
6. SF Gate, “Daily Amounts of Carbs, Fat, Fiber, Sodium & Protein” <http://healthyeating.sfgate.com/daily-amounts-carbs-fat-fiber-sodium-protein-4230.html>
7. Sparkpeople, “Healthy Beverage Guidelines” <http://www.sparkpeople.com/resource/nutrition_articles.asp?id=605>
8. US Department of Argriculture, “Estimated Calorie Needs per Day by Age, Gender, and Physical Activity Level” <https://www.cnpp.usda.gov/sites/default/files/usda_food_patterns/EstimatedCalorieNeedsPerDayTable.pdf>
9. US Food and Drug Administration, “Guidance for Industry: A Food Labeling Guide (14. Appendix F: Calculate the Percent Daily Value for the Appropriate Nutrients)” <http://www.fda.gov/Food/GuidanceRegulation/GuidanceDocumentsRegulatoryInformation/LabelingNutrition/ucm064928.htm>
10. Vegetarian Resource Group, “Veganism in a Nutshell” <http://www.vrg.org/nutshell/vegan.htm#what>
11. Wikipedia, “Knapsack problem” <https://en.wikipedia.org/wiki/Knapsack_problem>
12. Wikipedia, “Stigler Diet” <https://en.wikipedia.org/wiki/Stigler_diet>

**9. Appendix: Codes, Inputs and Outputs**

* **Data arrangements** <https://goo.gl/GnIS24>
* (code) Preference\_Switch.sh
* (code) pref\_ind\_switch\_final.sh
* (input) FoodGroup.list
* (input) pref.list
* (input) prefMF.list
* (output) new\_pref.out
* **Main Solution** <https://goo.gl/lhWq0X>
* (code) lp\_diet.py
* (code) lp\_output\_process.py
* (out/input) lp\_diet\_female.txt
* (out/input) lp\_diet\_male.txt
* (out/input) lp\_output\_female.txt
* (out/input) lp\_output\_male.txt
* (output) lp\_trimmed\_female.txt
* (output) lp\_trimmed\_male.txt
* **Variation 1: Unhappy Case** <https://goo.gl/MSypoq>
* (code) lp\_diet\_unhappy.py
* (out/input) lp\_diet\_female\_unhappy.txt
* (out/input) lp\_diet\_male\_unhappy.txt
* (out/input) lp\_output\_female\_unhappy.txt
* (out/input) lp\_output\_male\_unhappy.txt
* (output) lp\_trimmed\_female\_unhappy.txt
* (output) lp\_trimmed\_male\_unhappy.txt
* **Variation 2: Vegetarian Case** <https://goo.gl/MSypoq>
* (code) lp\_diet\_veg.py
* (out/input) lp\_diet\_female\_veg.txt
* (out/input) lp\_diet\_male\_veg.txt
* (out/input) lp\_output\_female\_veg.txt
* (out/input) lp\_output\_male\_veg.txt
* (output) lp\_trimmed\_female\_veg.txt
* (output) lp\_trimmed\_male\_veg.txt
* **Variation 3: Diverse Case** <https://goo.gl/MSypoq>
* (code) lp\_diet\_grouplim.py
* (out/input) lp\_diet\_female\_grouplim.txt
* (out/input) lp\_diet\_male\_grouplim.txt
* (out/input) lp\_output\_female\_grouplim.txt
* (out/input) lp\_output\_male\_grouplim.txt
* (output) lp\_trimmed\_female\_grouplim.txt
* (output) lp\_trimmed\_male\_grouplim.txt

**(code) Preference\_Switch.sh**

#!/usr/local/bin/

# Baihan Lin, October 2016

# The following code was to generate proper data format of

# two columns of preference data for male and female

# corresponding to each food item. These two columns

# are later copied and pasted into Excel sheet of all data.

# Part I: Generate pattern matching arguments

# input file: FoodGroup.list, pref.list, prefMF.list

# output file: pref\_ind\_switch\_final.sh

rm pref\*.run\*;

rm pref\_ind\_switch.sh\*;

rm pref\_ind\_switch\_final.sh\*;

rm new\_pref.out\*;

cp pref.list pref1.run;

sed -i -e 's/^/elif [[ $j = /' pref1.run;

cp prefMF.list pref2.run;

sed -i -e 's/^/ ]];then echo /' pref2.run;

paste pref1.run pref2.run > pref\_ind\_switch.sh;

sed -i '' 's#$# >> new\_pref.out;#' pref\_ind\_switch.sh;

cat pref\_ind\_switch.sh | tr -d '\011' > pref\_ind\_switch\_final.sh;

sed -i '' "s/+/ /" pref\_ind\_switch\_final.sh;

sed -i '' "s/+/ /" pref\_ind\_switch\_final.sh;

# Part II: Generate columns with matched preference data

# input file: pref\_ind\_switch\_final.sh

# output file: new\_pref.out

for j in `cat FoodGroup.list`; do

#sh pref\_ind\_switch\_final.sh

if [[ $j = A ]];then echo A 5 3.5 >> new\_pref.out;

elif [[ $j = AA ]];then echo AA 3.5 3.5 >> new\_pref.out;

elif [[ $j = AB ]];then echo AB 8 2.5 >> new\_pref.out;

elif [[ $j = AC ]];then echo AC 6 3.5 >> new\_pref.out;

elif [[ $j = AD ]];then echo AD 4.5 3 >> new\_pref.out;

elif [[ $j = AE ]];then echo AE 6.5 1.5 >> new\_pref.out;

elif [[ $j = AF ]];then echo AF 2.5 4.5 >> new\_pref.out;

elif [[ $j = AG ]];then echo AG 2 4 >> new\_pref.out;

...

and 128 more rows like that...

elif [[ $j = WE ]];then echo WE 4 3.5 >> new\_pref.out;

else

echo $j not found >> new\_pref.out;

fi;

done

Here is the link to the full file: <https://goo.gl/GnIS24>

**(code) pref\_ind\_switch\_final.sh**

#!/usr/local/bin/

# Baihan Lin, October 2016

# The following code was generated by Preference\_Switch.sh

# in order to be copied and pasted into Part II of

# Preference\_Switch.sh.

elif [[ $j = A ]];then echo A 5 3.5 >> new\_pref.out;

elif [[ $j = AA ]];then echo AA 3.5 3.5 >> new\_pref.out;

elif [[ $j = AB ]];then echo AB 8 2.5 >> new\_pref.out;

...

and 135 more rows like that...

Here is the link to the full file: <https://goo.gl/GnIS24>

**(input) FoodGroup.list**

Too big to display, 2898 rows.

Here is the link to the full file: <https://goo.gl/GnIS24>

**(input) pref.list**

A

AA

AB

AC

AD

AE

AF

AG

AI

AK

AM

AN

AO

AP

AS

AT

B

BA

BAB

BAE

BAH

BAK

BAN

BAR

BC

BF

BFD

BFG

BFJ

BFP

BH

BJ

BJC

BJF

BJL

BJP

BJS

BL

BN

BNE

BNH

BNS

BP

BR

BV

C

CA

CD

CDE

CDH

D

DA

DAE

DAM

DAP

DAR

DB

DF

DG

DI

DR

F

FA

FC

G

GA

H

IF

IFB

IFC

J

JA

JC

JK

JM

JR

M

MA

MAA

MAC

MAE

MAG

MAI

MC

MCA

MCC

MCE

MCG

MCI

MCK

MCM

MCO

ME

MEA

MEC

MEE

MG

MBG

MI

MIG

MR

O

OA

OB

OC

OE

OF

P

PA

PAA

PAC

PC

PCA

PCC

PE

Q

QA

QC

QE

QF

QG

QI

QK

S

SC

SE

SEA

SEC

SN

SNA

SNB

SNC

W

WA

WAA

WAC

WAE

WC

WCD

WCG

WCN

WE

For environment protection (saving trees is saving paper), here we displayed it in 3 columns.

**(input) prefMF.list**

A + 5 + 3.5

AA + 3.5 + 3.5

AB + 8 + 2.5

AC + 6 + 3.5

AD + 4.5 + 3

AE + 6.5 + 1.5

AF + 2.5 + 4.5

AG + 2 + 4

AI + 1 + 4.5

AK + 2 + 2

AM + 3.5 + 4.5

AN + 7 + 5.5

AO + 3 + 3.5

AP + 4 + 3.5

AS + 4.5 + 4.5

AT + 2 + 3

B + 4 + 4

BA + 3 + 4

BAB + 3 + 4.5

BAE + 2 + 4.5

BAH + 2 + 4.5

BAK + 5 + 4

BAN + 3 + 3.5

BAR + 4.5 + 2.5

BC + 4 + 3

BF + 1.5 + 0.5

BFD + 5 + 3

BFG + 4.5 + 3

BFJ + 6 + 6.5

BFP + 3.5 + 2.5

BH + 4.5 + 4

BJ + 1 + 2.5

BJC + 1 + 3.5

BJF + 1 + 2

BJL + 1 + 2.5

BJP + 1 + 2.5

BJS + 1 + 2.5

BL + 5 + 1.5

BN + 8 + 6

BNE + 8 + 5

BNH + 7 + 7

BNS + 6.5 + 7

BP + 10 + 8.5

BR + 4 + 5.5

BV + 3.5 + 3.5

C + 5.5 + 5.5

CA + 5 + 5.5

CD + 7 + 5.5

CDE + 7 + 5

CDH + 4 + 4.5

D + 4.5 + 5.5

DA + 3.5 + 7.5

DAE + 4 + 4.5

DAM + 4.5 + 5

DAP + 4 + 5

DAR + 4.5 + 5.5

DB + 4 + 5

DF + 4 + 5.5

DG + 6.5 + 6.5

DI + 5 + 5

DR + 7 + 6.5

F + 6.5 + 7.5

FA + 8 + 8

FC + 8 + 7

G + 4.5 + 6.5

GA + 5 + 6.5

H + 4 + 4

IF + 1.5 + 1.5

IFB + 1.5 + 1.5

IFC + 1.5 + 1.5

J + 5.5 + 5.5

JA + 7 + 7

JC + 5 + 5

JK + 5 + 4.5

JM + 5 + 4

JR + 7 + 7

M + 6.5 + 5

MA + 8.5 + 6

MAA + 6.5 + 3.5

MAC + 8.5 + 5.5

MAE + 7.5 + 4.5

MAG + 5.5 + 5.5

MAI + 5 + 6

MC + 5 + 5

MCA + 5 + 6

MCC + 5.5 + 6.5

MCE + 5.5 + 4.5

MCG + 2.5 + 3

MCI + 2 + 3

MCK + 2 + 3.5

MCM + 3 + 3.5

MCO + 5 + 6.5

ME + 2 + 4.5

MEA + 2.5 + 1.5

MEC + 3 + 1.5

MEE + 2 + 1.5

MG + 1.5 + 2.5

MBG + 7 + 3.5

MI + 7 + 4.5

MIG + 6 + 4

MR + 7 + 4.5

O + 3 + 2

OA + 2 + 1.5

OB + 3.5 + 2

OC + 2 + 2

OE + 4.5 + 2.5

OF + 1.5 + 2

P + 5.5 + 4.5

PA + 5.5 + 4

PAA + 5.5 + 5

PAC + 3 + 4

PC + 5 + 4.5

PCA + 5 + 3.5

PCC + 6.5 + 2.5

PE + 8.5 + 8.5

Q + 4.5 + 4

QA + 6 + 3.5

QC + 4.5 + 4

QE + 3 + 5

QF + 3 + 4.5

QG + 3 + 2.5

QI + 4.5 + 2.5

QK + 7 + 4.5

S + 4.5 + 6.5

SC + 4 + 7

SE + 2 + 6.5

SEA + 4.5 + 6

SEC + 4.5 + 6

SN + 5 + 7

SNA + 7 + 7

SNB + 6 + 6

SNC + 4.5 + 6

W + 5.5 + 4.5

WA + 6.5 + 4.5

WAA + 7 + 5.5

WAC + 4.5 + 3

WAE + 4 + 3.5

WC + 4 + 4

WCD + 5 + 5

WCG + 6.5 + 6.5

WCN + 4.5 + 5

WE + 4 + 3.5

For environment protection (saving trees is saving paper), here we displayed it in 3 columns.

**(output) new\_pref.out**

Too big to display, 2898 rows.

Here is the link to the full file: <https://goo.gl/GnIS24>

**(code) lp\_diet.py**

This part is already included in the main text.

**(code) lp\_output\_process.py**

This part is already included in the main text.

**(out/input) lp\_diet\_female.txt**

Too big to display, thus here is only an extraction:

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

+151x\_016x\_1...+211x\_2834>=2200;

+151x\_016x\_1...+211x\_2834<=2800;

+2.9x\_01.3x\_1...+6.7x\_28326.8x\_2833+6.7x\_2834>=55;

+2.9x\_01.3x\_1...+6.7x\_28326.8x\_2833+6.7x\_2834<=245;

+15.2x\_01.2x\_1...9.8x\_2834>=48.9;

15.2x\_01.2x\_1...9.8x\_2834<=108.9;

+0.8x\_0+0.0x\_1+...+25.8x\_2834>=247.5;

+0.8x\_0+0.0x\_1+...+25.8x\_2834<=455;

+0.0x\_0+0.3x\_1+...+2.78x\_2834<=21.8;

+0x\_0+0.0x\_1+...+0.09x\_2834<=3.1;

+0x\_0+0x\_1+0x\_2+...+57.3x\_2834<=300;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Here is the link to the full file: <https://goo.gl/lhWq0X>

**(out/input) lp\_diet\_male.txt**

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Too big to display, similar to female case, but with different constraint values.

Here is the link to the full file: <https://goo.gl/lhWq0X>

**(out/input) lp\_output\_female.txt**

Value of objective function: 1639.81681676

Actual values of the variables:

x\_0 0

...

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/lhWq0X>

**(out/input) lp\_output\_male.txt**

Value of objective function: 1794.33880959

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/lhWq0X>

**(output) lp\_trimmed\_female.txt**

This part is already included in the main text.

**(output) lp\_trimmed\_male.txt**

This part is already included in the main text.

**(code) lp\_diet\_unhappy.py**

This part is already partly included in the main text.

Here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_female\_unhappy.txt**

min:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we changed the “max” to “min”.

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_male\_unhappy.txt**

min:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we changed the “max” to “min”.

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_female\_unhappy.txt**

Value of objective function: 15.27183148

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_male\_unhappy.txt**

Value of objective function: 8.33620465

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(output) lp\_trimmed\_female\_unhappy.txt**  
This part is already included in the main text.

**(output) lp\_trimmed\_male\_unhappy.txt**

This part is already included in the main text.

**(code) lp\_diet\_veg.py**

This part is already partly included in the main text.

Here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_female\_veg.txt**

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we added a constraint.

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_male\_veg.txt**

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we added a constraint.Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_female\_veg.txt**

Value of objective function: 1296.64981001

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_male\_veg.txt**

Value of objective function: 1454.78168382

Actual values of the variables:

x\_0 0

…

and 2835 more lines like above...Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(output) lp\_trimmed\_female\_veg.txt**This part is already included in the main text.

**(output) lp\_trimmed\_male\_veg.txt**

This part is already included in the main text.

**(code) lp\_diet\_grouplim.py**

This part is already partly included in the main text.

Here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_female\_grouplim.txt**

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we added a constraint.

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_diet\_male\_grouplim.txt**

max:+6.5x\_0+6.5x\_1+...+2x\_2834;

....

+x\_0<=30;+x\_1<=30;...+x\_2834<=30;

and 53 more lines like above...

Similar to original case, except that we added a constraint.

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_female\_grouplim.txt**

Value of objective function: 558.85880275

Actual values of the variables:

x\_0 0

…

and 2835 more lines like above...Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(out/input) lp\_output\_male\_grouplim.txt**

Value of objective function: 618.16189370

Actual values of the variables:

x\_0 0

…

and 2834 more lines like above...

Too big to display, thus here is the link to the full file: <https://goo.gl/MSypoq>

**(output) lp\_trimmed\_female\_grouplim.txt**

This part is already included in the main text.

**(output) lp\_trimmed\_male\_grouplim.txt**

This part is already included in the main text.

**(Data) Dataset in Excel format**

Here is the link to the full file: <https://goo.gl/4umRKN>

**(Folder) link to our google drive**

<https://drive.google.com/open?id=0B6BZ-0nxfo4YaW5sS1dKWTZMWmc>