Homework 7 for ISYE6501

Dylan Peters

6/30/2017

1. Formulate an optimization model (a linear program) to find the cheapest diet that satisfies the maximum and minimum daily nutrition constraints, and solve it using PuLP. Turn in your code and the solution. (The optimal solution should be a diet of air-popped popcorn, poached eggs, oranges, raw iceberg lettuce, raw celery, and frozen broccoli. UGH!)

We start by loading the data, then setting the variables and constraints

In [1]:
```
import sys
sys.version
```

Out[1]:
```
'2.7.13 |Anaconda custom (64-bit)| (default, May 11 2017, 13:17:26) [MSC v.1500
64 bit (AMD64)]'
```

In [4]:
```
from pulp import *
import pandas as pd
```

In [5]:
```
dietdata = pd.read_excel("diet.xls")
dietdata.dropna(axis=0,inplace=True)
dietdata.head()
```

Out[5]:

	Foods	Price/ Serving	Serving Size	Calories	Cholesterol mg	Total_Fat g	Sodium mg	Carbohydrates g	D g
0	Frozen Broccoli	0.16	10 Oz Pkg	73.8	0.0	0.8	68.2	13.6	8
1	Carrots,Raw	0.07	1/2 Cup Shredded	23.7	0.0	0.1	19.2	5.6	1
2	Celery, Raw	0.04	1 Stalk	6.4	0.0	0.1	34.8	1.5	0
3	Frozen Corn	0.18	1/2 Cup	72.2	0.0	0.6	2.5	17.1	2
4	Lettuce,Iceberg,Raw	0.02	1 Leaf	2.6	0.0	0.0	1.8	0.4	0

Step 1: Define the problem

The goal is to find a set of foods that will minimize the cost while meeting nutritional demands.

The variables are the foods and their nutritional values.

The nutritional requirements are defined here:

	Calories	Cholesterol mg	Total_Fat g	Sodium mg	Carbohydrates g	Dietary_Fiber g	Protein g	Vit_A IU	Vit_C IU	Calcium mg	Iro m
Minimum daily intake	1500	30	20	800	130	125	60	1000	400	700	10
Maximum daily intake	2500	240	70	2000	450	250	100	10000	5000	1500	40

Those are our main constraints.

The list of foods are in dietdata.Foods, the first column. The rest of the columns contain the nutrition content for each food. We can use those arrays.

Step 2: Set up the variables and constraints and objective function

The foods are our variables so create a list for them.

In [198]:
```
dietprob = LpProblem("Army Diet", LpMinimize) # Minimize because we want the lowes
t cost

# Food list for use in list comprehension
FoodList = list(dietdata["Foods"])

# Create variables
food_vars = LpVariable.dicts("Foods",FoodList,0)
```

First set up the objective function. I use list comprehension and indexing the Pandas data frame.

In [199]:
```
dietprob += lpSum([dietdata["Price/ Serving"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]), "Total Cost of food per meal"
```

Now add the other constraints. These constraints will ensure that for each nutrient, the sum of all the nutrient values in the chosen foods bit between the minimum and maximum.

In [200]:
```
# Calories
dietprob += lpSum([dietdata["Calories"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) >= 1500, "Calorie Min"
dietprob += lpSum([dietdata["Calories"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) <= 2500, "Calorie Max"

# Cholesterol
dietprob += lpSum([dietdata["Cholesterol mg"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) >= 30, "Cholesterol Min"
dietprob += lpSum([dietdata["Cholesterol mg"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) <= 240, "Cholesterol Max"

# Total Fat
dietprob += lpSum([dietdata["Total_Fat g"][dietdata.Foods == i] * food_vars[i] for
i in FoodList]) >= 20, "Total Fat Min"
dietprob += lpSum([dietdata["Total_Fat g"][dietdata.Foods == i] * food_vars[i] for
i in FoodList]) <= 70, "Total Fat Max"

# Sodium
dietprob += lpSum([dietdata["Sodium mg"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) >= 800, "Sodium Min"
dietprob += lpSum([dietdata["Sodium mg"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) <= 2000, "Sodium Max"

# Carbohydrates
dietprob += lpSum([dietdata["Carbohydrates g"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) >= 130, "Carbohydrates Min"
dietprob += lpSum([dietdata["Carbohydrates g"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) <= 450, "Carbohydrates Max"

# Dietary Fiber
dietprob += lpSum([dietdata["Dietary_Fiber g"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) >= 125, "Dietary_Fiber Min"
dietprob += lpSum([dietdata["Dietary_Fiber g"][dietdata.Foods == i] * food_vars[i]
for i in FoodList]) <= 250, "Dietary_Fiber Max"

# Protein
dietprob += lpSum([dietdata["Protein g"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) >= 60, "Protein Min"
dietprob += lpSum([dietdata["Protein g"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) <= 100, "Protein Max"

# Vitamin A
dietprob += lpSum([dietdata["Vit_A IU"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) >= 1000, "Vitamin A Min"
dietprob += lpSum([dietdata["Vit_A IU"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) <= 10000, "Vitamin A Max"

# Vitamin C
dietprob += lpSum([dietdata["Vit_C IU"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) >= 400, "Vitamin C Min"
dietprob += lpSum([dietdata["Vit_C IU"][dietdata.Foods == i] * food_vars[i] for i
in FoodList]) <= 5000, "Vitamin C Max"

# Calcium
dietprob += lpSum([dietdata["Calcium mg"][dietdata.Foods == i] * food_vars[i] for
i in FoodList]) >= 700, "Calcium Min"
dietprob += lpSum([dietdata["Calcium mg"][dietdata.Foods == i] * food_vars[i] for
i in FoodList]) <= 1500, "Calcium Max"

# Iron
dietprob += lpSum([dietdata["Iron mg"][dietdata.Foods == i] * food_vars[i] for i i
n FoodList]) >= 10, "Iron Min"
dietprob += lpSum([dietdata["Iron mg"][dietdata.Foods == i] * food_vars[i] for i i
```

Step 3: Solve

Now we can solve the problem:

In [201]:
```
dietprob.solve()
```

Out[201]: 1

Now we can loop through the variables and get the results:

In [202]:
```
print LpStatus[dietprob.status]

for food in dietprob.variables():
    if food.varValue > 0:
        print food.name, " = ", food.varValue
```

```
Optimal
Foods_Celery,_Raw  =  52.64371
Foods_Frozen_Broccoli  =  0.25960653
Foods_Lettuce,Iceberg,Raw  =  63.988506
Foods_Oranges  =  2.2929389
Foods_Poached_Eggs  =  0.14184397
Foods_Popcorn,Air_Popped  =  13.869322
```

This matches our expectations.

Part 2

Please add to your model the following constraints (which might require adding more variables) and solve the new model:

a. If a food is selected, then a minimum of 1/10 serving must be chosen. (Hint: now you will need two variables for each food i: whether it is chosen, and how much is part of the diet. You'll also need to write a constraint to link them.)

b. Many people dislike celery and frozen broccoli. So at most one, but not both, can be selected.

c. To get day-to-day variety in protein, at least 3 kinds of meat/poultry/fish/eggs must be selected.

In [203]:
```
# a: enforce at least 1/10 serving
food_selected_vars = LpVariable.dicts("FoodSelected", FoodList, 0, 1, cat=LpBinary
)
for food in FoodList:
    dietprob += food_vars[food] >= 0.1 * food_selected_vars[food], "Min serving si
ze %s" % food
    dietprob += food_vars[food] <= 1000 * food_selected_vars[food], "Max serving s
ize %s" % food
```

In [204]:
```
# b: enforce not both celery and frozen broccoli
dietprob += LpConstraint(food_selected_vars["Celery, Raw"] + food_selected_vars["F
rozen Broccoli"] <= 1), "Not both Celery and Broccoli"
```

In [205]:
```
# c: at least three meats etc.
Meats = ["Roasted Chicken","Poached Eggs","Scrambled Eggs","Bologna,Turkey","Frankfurter, Beef",
         "Ham,Sliced,Extralean","Kielbasa,Prk","Pizza W/Pepperoni","Taco","Hamburger W/Toppings",
         "Hotdog, Plain","Pork","Sardines in Oil","White Tuna in Water","Chicknoodl Soup",
         "Splt Pea&Hamsoup","Vegetbeef Soup","Beanbacn Soup,W/Watr"]
dietprob += lpSum([food_selected_vars[meat] for meat in Meats]) >= 3, "At least 3 meats"
```

In [206]:
```
dietprob.solve()
```

Out[206]: 1

In [209]:
```
print LpStatus[dietprob.status]

for food in dietprob.variables():
    if food.varValue > 0:
        print food.name, " = ", food.varValue, " ", food.cat
```

```
Optimal
FoodSelected_Celery,_Raw  =  1.0   Integer
FoodSelected_Kielbasa,Prk  =  1.0   Integer
FoodSelected_Lettuce,Iceberg,Raw  =  1.0   Integer
FoodSelected_Oranges  =  1.0   Integer
FoodSelected_Peanut_Butter  =  1.0   Integer
FoodSelected_Poached_Eggs  =  1.0   Integer
FoodSelected_Popcorn,Air_Popped  =  1.0   Integer
FoodSelected_Scrambled_Eggs  =  1.0   Integer
Foods_Celery,_Raw  =  42.399358   Continuous
Foods_Kielbasa,Prk  =  0.1   Continuous
Foods_Lettuce,Iceberg,Raw  =  82.802586   Continuous
Foods_Oranges  =  3.0771841   Continuous
Foods_Peanut_Butter  =  1.9429716   Continuous
Foods_Poached_Eggs  =  0.1   Continuous
Foods_Popcorn,Air_Popped  =  13.223294   Continuous
Foods_Scrambled_Eggs  =  0.1   Continuous
```

That's still a lot of celery! There are 3 "meats": poached eggs, scrambled eggs, and pork. And there's celery but no brocoli. Success!