ISYE 6501, Week 11 HW

Question 1

Describe a situation or problem from your job, everyday life, current events, etc., for which optimization would be appropriate. What data would you need?

Response –

Optimization will be ideal for a business which is trying to maximize the profit among different types of products it sells. Each product requires different amount of labor and materials. By using the constraints on labor and raw materials, the optimization model can help the business decide on how many of each product can be manufactured for maximum profit.

The data needed will be –

Cost of labor
Cost of raw materials
Selling price of each product
Maximum limit of labor hours
Maximum limit of raw materials
Maximum sales of each product possible

Question 2

In the videos, we saw the "diet problem". (The diet problem is one of the first large-scale optimization problems to be studied in practice. Back in the 1930's and 40's, the Army wanted to meet the nutritional requirements of its soldiers while minimizing the cost.) In this homework you get to solve a diet problem with real data. The data is given in the file diet.xls.

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!)

Response –

The diet problem data is read from the excel file and the objective function and constraints are set as fol lows –

```
tut = LpProblem('PuLPTutorial',LpMinimize)

foodVars = LpVariable.dicts("Foods", foods, 0)

chosenVars = LpVariable.dicts("Chosen", foods,0,1,'Binary')

tut += lpSum([cost[f]*foodVars[f] for f in foods]), 'Total Cost'

tut += lpSum([calories[f] * foodVars[f] for f in foods]) >= 1500, 'min Calories'
tut += lpSum([calories[f] * foodVars[f] for f in foods]) <= 2500, 'max Calories'

tut += lpSum([totalFat[f] * foodVars[f] for f in foods]) >= 20, 'min Fat'
tut += lpSum([totalFat[f] * foodVars[f] for f in foods]) <= 70, 'max Fat'

tut += lpSum([cholesterol[f] * foodVars[f] for f in foods]) >= 30, 'min Cholesterol'
tut += lpSum([cholesterol[f] * foodVars[f] for f in foods]) <= 240, 'max Cholesterol'

tut += lpSum([sodium[f] * foodVars[f] for f in foods]) >= 800, 'min Sodium'
tut += lpSum([sodium[f] * foodVars[f] for f in foods]) <= 2000, 'max Sodium'

tut += lpSum([carbohydrates[f] * foodVars[f] for f in foods]) >= 130, 'min Carbohydrates'
tut += lpSum([carbohydrates[f] * foodVars[f] for f in foods]) <= 450, 'max Carbohydrates'

tut += lpSum([dietaryFiber[f] * foodVars[f] for f in foods]) >= 125, 'min Dietary Fiber'
tut += lpSum([dietaryFiber[f] * foodVars[f] for f in foods]) <= 250, 'max Dietary Fiber'

tut += lpSum([protein[f] * foodVars[f] for f in foods]) >= 60, 'min Protein'
tut += lpSum([protein[f] * foodVars[f] for f in foods]) <= 100, 'max Protein'

tut += lpSum([vitA[f] * foodVars[f] for f in foods]) >= 1000, 'min vitamin A'
tut += lpSum([vitA[f] * foodVars[f] for f in foods]) <= 10000, 'max vitamin A'

tut += lpSum([vitC[f] * foodVars[f] for f in foods]) >= 400, 'min vitamin C'
tut += lpSum([vitC[f] * foodVars[f] for f in foods]) <= 5000, 'max vitamin C'

tut += lpSum([calcium[f] * foodVars[f] for f in foods]) >= 700, 'min Calcium'
tut += lpSum([calcium[f] * foodVars[f] for f in foods]) <= 1500, 'max Calcium'

tut += lpSum([iron[f] * foodVars[f] for f in foods]) >= 10, 'min Iron'
tut += lpSum([iron[f] * foodVars[f] for f in foods]) <= 40, 'max Iron'
```

The solution for the linear program to find the cheapest diet that statisfies the nutrition constraints is –

```
Status: Optimal

-------------The solution to the diet problem is--------------
52.64371 units of Celery,_Raw
0.25960653 units of Frozen_Broccoli
63.988506 units of Lettuce,Iceberg,Raw
2.2929389 units of Oranges
0.14184397 units of Poached_Eggs
13.869322 units of Popcorn,Air_Popped

Total cost of food = $4.34
```

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.

Response –

The constraints are introduced as –

for f in foods:

    tut += 0.1*chosenVars[f] <= foodVars[f]

    tut += foodVars[f] <= 10000000*chosenVars[f]

tut += chosenVars['Celery, Raw'] + chosenVars['Frozen Broccoli'] <= 1, 'Either celery or frozen broccoli'

tut += chosenVars['Roasted Chicken'] + chosenVars['Pizza W/Pepperoni'] + chosenVars['Scrambled Eggs'] + chosenVars['Hamburger W/Toppings'] + chosenVars['Hotdog, Plain'] + chosenVars['Pork'] + chosenVars['White Tuna in Water'] + chosenVars['Splt Pea&Hamsoup'] + chosenVars['Vegetbeef Soup'] + chosenVars['Neweng Clamchwd'] >= 3, 'Atleast 3 meat'

The output of the model is -

```
Status: Optimal

-------------The solution to the diet problem is--------------
1.0 units of Celery,_Raw
1.0 units of Hotdog,_Plain
1.0 units of Lettuce,Iceberg,Raw
1.0 units of Oranges
1.0 units of Peanut_Butter
1.0 units of Popcorn,Air_Popped
1.0 units of Scrambled_Eggs
1.0 units of White_Tuna_in_Water
40.945972 units of Celery,_Raw
0.1 units of Hotdog,_Plain
87.305329 units of Lettuce,Iceberg,Raw
3.0916297 units of Oranges
1.8363895 units of Peanut_Butter
13.187384 units of Popcorn,Air_Popped
0.10426136 units of Scrambled_Eggs
0.1 units of White_Tuna_in_Water

Total cost of food = $4.62
```