Homework 7

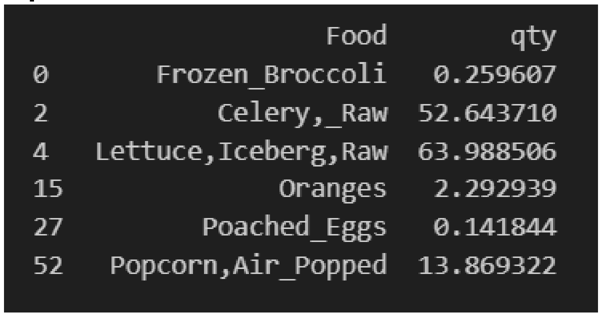
Question 15.2.1

For this problem, I started by importing the nutrition information for the 64 items into a dataframe, dropping blank rows and the min/max daily values for specific nutrients table. I opted to bring those min/max values separately and manually created another dataframe, which will be used when constructing constraints for my LP model. I then initialized my model with food names as the xi variable. I extracted it from the main dataframe and passed it through as an LpVariable. Because the problem was concerned with minimizing total cost of food, I extracted the cost/serving of food into its own series. With all pieces in place, I defined my objective function by multiplying each food variable with its price and summing all the food items together, specifying to the model that I wanted to minimize the total. I added only two constraints per type of nutrient: 1) nutrient value has to be greater than or equal to its corresponding minimum and 2) nutrient value has to be less than or equal to its corresponding maximum. With all constraints in place (2 per nutrient X 11 nutrients), **I solved the minimization problem and found the optimal solution of .25 servings of frozen broccoli, 52 servings of celery, 63.99 units of iceberg lettuce, 2.29 servings of oranges, .14 servings of poached eggs, and 13.87 servings of air-popped popcorn. This minimizes cost subject to the constraints that individuals get more than the minimum value of each nutrient and less than the maximum value of each nutrient.**

```
from pulp import *
import pandas as pd

#import df
df = pd.read_excel('diet.xls').dropna(axis=0,how='any')

#Create df of min/max values for each nutrient
values = [
    [1500,30,20,800,130,125,60,1000,400,700,10],
    [2500,240,70,2000,450,250,100,10000,5000,1500,40]]

nutrients = pd.DataFrame(
    columns = ['Calories','Cholesterol mg','Total_Fat g','Sodium mg','Carbohydrates
g','Dietary_Fiber g','Protein g','Vit_A IU','Vit_C IU','Calcium mg','Iron mg'],
    data= values)

#Convert and transpose to df
nutrients_df = nutrients.transpose().rename(columns = {0:'min',1:'max'})

#list of all nutrient names to use in loop
nutrients = nutrients_df.index.values.tolist()

#Initialize lp
lp = LpProblem("Diet_Problem",LpMinimize)

#xi variables
foods = df['Foods'].tolist()
variables = [LpVariable(name,lowBound=0) for name in foods]

#Cost per serving
cost_per_serving = df['Price/ Serving']

#Set objective function
lp.setObjective(sum([variables[i] * cost_per_serving.iloc[i] for i in
range(len(variables))]))
```

```
#loop to add constraints
for nutrient in nutrients:
    min = nutrients_df.loc[nutrient]['min']
    max = nutrients_df.loc[nutrient]['max']
    nut_val = df[nutrient]
    tot_nut_val = sum([variables[i] * nut_val.iloc[i] for i in range(len(variables))])
    lp.addConstraint(tot_nut_val >= min,f"Min_{nutrient}")
    lp.addConstraint(tot_nut_val <= max,f"Max_{nutrient}")

lp.solve()

# Construct new dataframe based on food_names and quantity values given by
food_names = pd.Series(variables)
qty = pd.Series([food.value() for food in variables])
results_df = pd.DataFrame({'Food':food_names, 'qty':qty})

#Select foods where the quantity was greater than 0
results_df = results_df[results_df['qty']>0]
print(results_df)
```

Optimal Solution



```
                  Food        qty
0      Frozen_Broccoli   0.259607
2          Celery,_Raw  52.643710
4   Lettuce,Iceberg,Raw  63.988506
15             Oranges   2.292939
27         Poached_Eggs   0.141844
52   Popcorn,Air_Popped  13.869322
```

Question 15.2.2

a) Reusing the code from the first part of the problem, I began building up the model using additional constraints. The first requirement is that 1/10 of a serving minimum must be selected if a food is selected at all. This required a for loop to create, with two constraints for each food. The first constraint links the xi variable (quantity of each food) to the yi variable (binary variables indicating if a food is selected as a solution or not). I used the equation **xi[food]<=yi[food]*10000** – if a food is selected, yi[food] will be equal to 1 and the right side of the equation will be 10,000 meaning that the amount of each food selected cannot be greater than 10,000. Conversely, if a food is not selected in the optimal solution, yi[food] = 0, and the xi[food] will be constrained to be less than or equal to 0 (0 X 10,000 = 0). The second constraint, **xi[food]>=yi[food]*.1** sets the minimum amount of food to be either 0 or 0.1, depending on if it is selected in the optimal solution or not.

b) After linking the binary variable and amount of each food variable in the previous section, it became much easier to write additional constraints. Using **lp.addConstraint(yi['Frozen Broccoli'] + yi['Celery, Raw'] <= 1)**, I was able to add in the constraint that either Frozen Broccoli OR Celery, Raw were selected for the model, but not both. This works because of the binary variable, yi; if frozen broccoli is selected, yi['Frozen Broccoli'] is set to 1, and yi['Celery,Raw'] cannot also be set to 1 because it would violate the constraint.

c) I decided to classify a subset of the data as protein if they were pure meat or tofu options and opted to exclude items such as bacon soup and clam chowder. I defined my protein types as a list, proteins. I then used list comprehension to create a new list made up of the binary values of the proteins. I then summed these up and set it as a constraint, **lp.addConstraint(protein_types >= 3,"Protein Variety")**. This constraint sets the minimum sum of the binary values to be equal to 3, effectively forcing the model to find an optimal solution to include 3 or more protein varieties, as defined in my list.

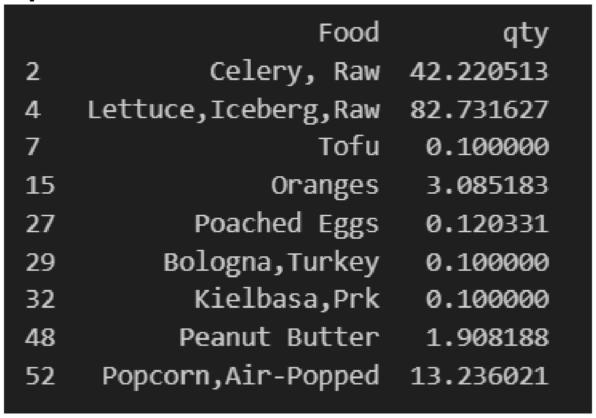
While slightly more appealing than the first diet, this new diet minimized costs while still including 3 types of protein by providing the minimum amount, .1 servings, per meat type. **The optimal solution consists of a diet of 42 servings of celery, 82 servings of lettuce, .1 servings of tofu, 3.1 servings of oranges, .12 servings of poached eggs, .1 servings of turkey bologna, .1 servings of Kielbasa, 1.9 servings of peanut butter, and 13 servings of air-popped popcorn.**

```
from pulp import *
import pandas as pd

#import df
df = pd.read_excel('diet.xls').dropna(axis=0,how='any')

#Create df of min/max values for each nutrient
values = [
    [1500,30,20,800,130,125,60,1000,400,700,10],
    [2500,240,70,2000,450,250,100,10000,5000,1500,40]]

nutrients = pd.DataFrame(
    columns = ['Calories','Cholesterol mg','Total_Fat g','Sodium mg','Carbohydrates 
g','Dietary_Fiber g','Protein g','Vit_A IU','Vit_C IU','Calcium mg','Iron mg'],
    data= values)

#Convert and transpose to df
nutrients_df = nutrients.transpose().rename(columns = {0:'min',1:'max'})

#list of all nutrient names to use in loop
nutrients = nutrients_df.index.values.tolist()
```

```
#Initialize lp
lp = LpProblem("Diet_Problem",LpMinimize)

#variables
foods = df['Foods'].tolist()
xi = LpVariable.dicts("Amounts",foods,0)
yi = LpVariable.dicts("Binaries",foods,cat=LpBinary)

#Cost per serving
cost_per_serving = df.copy().set_index(df['Foods']) #
cost_df = cost_per_serving['Price/ Serving']

#Set objective function
lp.setObjective(sum(xi[food] * cost_df.loc[food] for food in foods))

#loop to add min/max nutrient constraints
for nutrient in nutrients:
    min = nutrients_df.loc[nutrient]['min']
    max = nutrients_df.loc[nutrient]['max']
    nut_val = df.copy().set_index(df['Foods'])
```

```python
    nut_val = nut_val[nutrient]
    tot_nut_val = sum([xi[food] * nut_val.loc[food] for food in foods])
    lp.addConstraint(tot_nut_val >= min,f"Min_{nutrient}")
    lp.addConstraint(tot_nut_val <= max,f"Max_{nutrient}")

#link xi to yi
#if food chosen, quantity >=.1
for food in foods:
    lp.addConstraint(xi[food]<=yi[food]*10000)
    lp.addConstraint(xi[food]>=yi[food]*.1)

#frozen broccoli or celery but not both
lp.addConstraint(yi['Frozen Broccoli'] + yi['Celery, Raw'] <= 1)

#At least 3 different types of protein
proteins = ['Tofu','Roasted Chicken','Bologna,Turkey',
           'Frankfurter, Beef','Ham,Sliced,Extralean',
           'Kielbasa,Prk','Hamburger W/Toppings','Hotdog, Plain',
           'Pork','Sardines in Oil','White Tuna in Water']

protein_types = sum([yi[protein] for protein in proteins])
lp.addConstraint(protein_types >= 3,"Protein Variety")

lp.solve()

# Construct new dataframe based on food_names and quantity values given by lp.solve()
food_names = pd.Series(foods)
qty = pd.Series([xi[food].value() for food in foods])
results_df = pd.DataFrame({'Food':food_names, 'qty':qty})

#Select foods where the quantity was greater than 0.1
results_df = results_df[results_df['qty']>=0.00001]
print(results_df)
```

Optimal Solution



```
                   Food        qty
2           Celery, Raw  42.220513
4   Lettuce,Iceberg,Raw  82.731627
7                  Tofu   0.100000
15              Oranges   3.085183
27        Poached Eggs    0.120331
29       Bologna,Turkey   0.100000
32         Kielbasa,Prk   0.100000
48        Peanut Butter   1.908188
52   Popcorn,Air-Popped  13.236021
```