DEMO 11 - OPTIMIZATION

RESULT 1 OPTIMIZED SOLUTION

```
Optimized Solution 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
```

CODE FOR 1

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

In [ ]:
```
data = pd.read_excel("diet.xls",header=0)

dt = data[0:64]
dt = dt.values.tolist()

nutrientnames = list(data.columns.values)
minval = data[65:66].values.tolist()
maxval = data[66:67].values.tolist()

foods = [j[0] for j in dt]
costs = dict([(j[0],float(j[1])) for j in dt])

nutrients = []
for i in range(11):
    nutrients.append(dict([(j[0],float(j[i+3])) for j in dt]))

prob = LpProblem("Food Optimization",LpMinimize)

foodvars = LpVariable.dicts("Foods",foods,0)
foodvarssel = LpVariable.dicts("food_select", foods, 0, 1, LpBinary)

#objective function
prob += lpSum([costs[f] * foodvars[f] for f in foods])

#constraints
for i in range(11):

    prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) >= minval[0][i+3]
    prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) <= maxval[0][i+3]

prob.solve()

print()
print("Optimized Solution is: \n")
for var in prob.variables():
    if var.varValue > 0 and "food_select" not in var.name:
        print(str(var.varValue)+" units of "+str(var).replace("Foods_",""))

print(f'{chr(10)}Total cost of food = ${round(value(prob.objective),2)}')
```

RESULT 2 OPTIMIZED SOLUTION - MORE CONSTRAINTS

```
Optimized Solution is:

0.1 units of Bologna,Turkey
42.423026 units of Celery,_Raw
82.673927 units of Lettuce,Iceberg,Raw
3.0856009 units of Oranges
1.9590978 units of Peanut_Butter
0.1 units of Poached_Eggs
13.214473 units of Popcorn,Air_Popped
0.1 units of Scrambled Eggs

Total cost of food = $4.51
```

CODE FOR 2

In [ ]:

```
data = pd.read_excel("diet.xls",header=0)

dt = data[0:64]
dt = dt.values.tolist()

nutrientnames = list(data.columns.values)
minval = data[65:66].values.tolist()
maxval = data[66:67].values.tolist()

foods = [j[0] for j in dt]
costs = dict([(j[0],float(j[1])) for j in dt])

nutrients = []
for i in range(11):
    nutrients.append(dict([(j[0],float(j[i+3])) for j in dt]))

prob = LpProblem("Food Optimization",LpMinimize)

foodvars = LpVariable.dicts("Foods",foods,0)
foodvarssel = LpVariable.dicts("food_select", foods, 0, 1, LpBinary)
#print(foodvars)

#objective function
prob += lpSum([costs[f] * foodvars[f] for f in foods])

#constraint 0
for i in range(11):

    prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) >= minval[0][i+3]
    prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) <= maxval[0][i+3]

#constraint 1
for food in foods:
    prob += foodvars[food] >= foodvarssel[food]*0.1
for food in foods:
    prob += foodvarssel[food] >= foodvars[food]*0.0000001

#constraint 2
prob += foodvarssel['Frozen Broccoli'] + foodvarssel['Celery, Raw'] <= 1

#constraint 3
prob += foodvarssel['Roasted Chicken'] + foodvarssel['Poached Eggs']\
        + foodvarssel['Scrambled Eggs'] + foodvarssel['Bologna,Turkey']\
        + foodvarssel['Frankfurter, Beef'] + foodvarssel['Kielbasa,Prk']\
        + foodvarssel['Pizza W/Pepperoni'] + foodvarssel['Hamburger W/Toppings']\
        + foodvarssel['Hotdog, Plain'] + foodvarssel['Pork']\
        + foodvarssel['Sardines in Oil'] + foodvarssel['White Tuna in Water']\
        + foodvarssel['Chicknoodl Soup'] + foodvarssel['Splt Pea&Hamsoup']\
        + foodvarssel['Vegetbeef Soup'] + foodvarssel['Neweng Clamchwd']\
        + foodvarssel['New E Clamchwd,W/Mlk'] + foodvarssel['Ham,Sliced,Extralean']\
        + foodvarssel['Beanbacn Soup,W/Watr']


#solving the new optimization problem


prob.solve()

print()
print("Optimized Solution is: \n")
for var in prob.variables():
    if var.varValue > 0 and "food_select" not in var.name:
        print(str(var.varValue)+" units of "+str(var).replace("Foods_",""))

print(f'{chr(10)}Total cost of food = ${round(value(prob.objective),2)}')
```

RESULT 3 - WORKING WITH THE FULL DATASET FOR LOW-CHOLESTROL DIET

Solution after adding constraints -

```
Optimized Solution is:

0.2106 units of Beans,_adzuki,_mature_seeds,_raw
0.4786 units of Cocoa_mix,_no_sugar_added,_powder
0.0675 units of Coriander_(cilantro)_leaves,_raw
0.096 units of Egg,_white,_dried,_flakes,_glucose_reduced
0.8 units of Infant_formula,_MEAD_JOHNSON,_ENFAMIL,_NUTRAMIGEN,_with_iron,_p
0.29 units of Infant_formula,_NESTLE,_GOOD_START_ESSENTIALS__SOY,_with_iron,
```

```
0.0022 units of Margarine,_industrial,_non_dairy,_cottonseed,_soy_oil_(partiall
0.9962 units of Oil,_vegetable,_sunflower,_linoleic,_(hydrogenated)
0.0821 units of Sauce,_mole_poblano,_dry_mix,_single_brand
0.4117 units of Snacks,_potato_chips,_barbecue_flavor
0.0592 units of Snacks,_potato_chips,_plain,_salted
0.1872 units of Snacks,_potato_sticks
0.2269 units of Soybeans,_mature_seeds,_roasted,_no_salt_added
0.1293 units of Spaghetti,_spinach,_dry
0.425 units of Tomato_powder
0.0869 units of Tomatoes,_sun_dried
0.0093 units of Veal,_variety_meats_and_by_products,_pancreas,_cooked,_braised
9999.7849 units of Water,_bottled,_non_carbonated,_CALISTOGA


Total Cholesterol: 0.000000
```

CODE FOR 3

In [ ]:
```
data = pd.read_excel("diet_large.xls",skiprows = 1, header=0)

dt = data[0:7146]
dt = dt.values.tolist()

nutrientnames = list(data.columns.values)
numnut = len(nutrientnames) - 1

for i in range(0, 7146):
    for j in range(1, numnut):
        if np.isnan(dt[i][j]):
            dt[i][j]=0

minval = data[7147:7148].values.tolist()
maxval = data[7149:7151].values.tolist()

foods = [j[0] for j in dt]
costs = dict([(j[0],float(j[nutrientnames.index('Cholesterol')])) for j in dt])

nutrients = []
for i in range(numnut):
    nutrients.append(dict([(j[0],float(j[i+1])) for j in dt]))

prob = LpProblem("Food Optimization",LpMinimize)

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

#objective function
prob += lpSum([costs[f] * foodvars[f] for f in foods])

#constraints
for i in range(0, numnut):

    if (not np.isnan(minval[0][i+1])) and (not np.isnan(maxval[0][i+1])):
        #print("\nAdding constriant for ",nutrientnames[i+1])
        prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) >= minval[0][i+1]
        prob += lpSum([nutrients[i][j] * foodvars[j] for j in foods]) <= maxval[0][i+1]
```

In [ ]:
```
#solving the new optimization problem

prob.solve()

print()
print("Optimized Solution is: \n")
for var in prob.variables():
    if var.varValue > 0:
        print(str(round(var.varValue,4))+" units of "+str(var).replace("Foods_",""))
print()
print("\nTotal Cholesterol: %f"%value(prob.objective))
```

THE END----------
