Домашняя работа №2 по курсу «Методы Оптимизации»

Коврижных Дмитрий Б05-903а

17.11.2021

Такое ощущение, что на физтехе со временем тупеешь. Надо не забыть удалить

1 General optimization problems

1 Give an explicit solution of the following LP.
$\mathbf{c}^\top x \to \min\limits_{x \in \mathbb{R}^n}$
s.t. $\mathbf{Ax = b}$
Sol:
Задача выпуклая, условие Слейтера выполняется, так как полупространство, заданное $Ax = b$ не пусто. Поэтому условия ККТ необходимы и достаточны для глобального минимума.

Запишем Лагранжиан: $L(x, \lambda) = c^T x + \lambda^T (Ax - b)$, Для ККТ нужно: $1) c + A^T \lambda = 0 \;\; 2) Ax = b$.
Так как $c^T x$ в подпространстве, заданном $Ax = b$ либо не имеет инфимум всегда, либо константа. Поэтому:

$1) p^* = c^T A^{-1} b$, в случае совместности системы,
$2) - \inf$ в остальных случаях. Доказано.

2 Give an explicit solution of the following LP. $\mathrm{c}^\top x \to \min\limits_{x \in \mathbb{R}^n}$
s.t. $1^T x = 1 \; and \; x \succeq 0$

Sol :
Условие регулярности Слейтера выполнено возьмём вектор, компоненты которого равны $\frac{1}{n}$, выпуклость - тоже. Поэтому условия ККТ необходимы и достаточны для глобального минимума.
Запишем Лагранжиан: $L(x, \lambda, \mu) = c^T x + \lambda(1^T x - 1) - \mu^T x, \lambda \in \mathbb{R}, \mu \in \mathbb{R}^n$
Запишем ККТ: $1) c_k + \lambda - \mu_j = 0, k = range(1, n) \;\; 2) \mu \geq 0 \;\; 3) \mu_k x_k = 0, \; where \; k = range(1, n) \;\; 4) 1^T x = 1 \;\; 5) x \geq 0$.
Тогда из третьего и второго условий получим, что $\forall i, \mu_i = 0$, из первого будет следовать, что $\lambda = -c_i$ а из четвёртого условия, что существует компонента $x > 0$. Добавим второе условие и получим, что $c_k - c_i = \mu_k \;\; c_i = \min c_k$.
Значит, $x_j = 1$ если $k = i$ и 0, если $k \neq i$. Решением будет $x^* = e_i, i = k | c_k = \min c_m, \; p* = c_i$

3 Where α is an integer between 0 and n...
Sol :

Во-первых, равенства аффинные, неравенство и сама функция - выпуклые. Рассмотрим вектор с компонентами $\frac{\alpha}{n}$, которые лежат в $\mathbb{R}$ как действительные числа. Тогда выполняется условие Слейтера.
Запишем функцию Лагранжа: $L(x, \lambda, \mu, \theta) = c^T x + \lambda(1^T x - \alpha) + \theta^T(x - 1) - \mu^T x$ где $\lambda \in \mathbb{R}; \mu, \theta \in \mathbb{R}^n$ Получим ККТ: $1. c_k + \lambda - \mu_j + \theta_k = 0; k\ from\ 1\ to\ n - 1.$ $2. \mu, \theta \geq 0.$ $3. \mu_k x_k = 0; \theta_k(x_k - 1) = 0, k\ from\ 1\ to\ n - 1.$ $4. 0 \leq x \leq 1.$ $5. 1^T x = \alpha.$
Прокидываем навык Visual Calculus и кажется, что оптимальным будет сумма α штук наименьших компонентов c, с оглядкой на предыдущую задачу. Рассмотрим разные случаи. Пусть $\alpha = 0$, то допустимо только $x = 0; p* = 0$. Пусть $\alpha \neq 0$, эти компоненты вектора x равны колу, а остальные нулевые. Отлично, так как по условию 3, первые α компонент у μ нулевые, и у θ это будут последние $n - (\alpha + 1)$. Значит, $\lambda \in [-c_\alpha;\ -c_{\alpha+1}]$
Для $\lambda = -c_\alpha$ выполняются все условия ККТ. Получим оптимальное $p* = \sum_{i=1}^{\alpha} c_i$.
Нецелое α округлим вверх до некоторого h, такого что первые h - 1 равны 1, а остальные компоненты x нулевые. Это можно сделать, смягчив ограничение на α до отрезка. Значение двойственных переменных не меняется, оптимальное $p^* = \sum_{i=1}^{h-1} c_i + (\alpha - (h + 1))c_m$.
!!11ДИМА БЛЯЬЬ НЕ ЗАБУДЬ ДОПИЛТЬ ПОСЛКДНИЙ СЛУЧАЙ11!!1!!

4 Give an explicit solution of the following QP.

Sol :
Снова задача выпукла, снова подставим вектор, скажем с компонентами $\frac{1}{n+1}$, снова условие Слейтера выполнено. Снова пишем Лагранжиан. $L(x, \mu) = c^T x + \mu(x^T A x - 1), \mu \in \mathbb{R}$. Запишем условия ККТ. Снова:
$1. c + 2\mu A x = 0.$ $2. \mu, \theta \geq 0.$ $3. \mu(x^T - 1) = 0.$ $4. x^T x = 1.$
$\mu \neq 0$ так как $c \neq 0$. Заодно выразим $x = \frac{-A^{-1}c}{2\mu}$. У нас есть x, подставим его в третье условие: $x^T A x - 1 = 0$, так как μ сократится. Найдём $\mu = \frac{\sqrt{c^T A^{-1} c}}{2} > 0$. Подставим μ в x и получим оптимальное решение: $x^* = \frac{-A^{-1}c}{\sqrt{c^T A^{-1} c}},\ then\ p* = -\sqrt{c^T A^{-1} c}$.
Доп. вопросы: так как,не изменив допустимое множество, из матрицы можно сделать симметричную, то будем считать нашу матрицу симметричной. При этом она не обязательно положительно определена. Если хотя бы один $\lambda \leq 0$, то допустимое множество неограниченно.
Пруфы: $z^T A z = \lambda z^T z = \lambda ||z||^2 \leq 0 < 1$.
$\forall \theta \in \mathbb{R}, tz \in S.\ p* \to \inf$ if $t \to \inf$. То есть неограниченно. Proofed.

5 Give an explicit solution of the following QP.

Sol :
Снова задача выпукла, снова подставим вектор $x = x_c$, для него выполняются ограничения, а сам он лежит во внутренности. Снова условие Слейтера выполнено. Снова пишем Лагранжиан.
$L(x, \mu) = c^T x + \mu((x - x_c)^T A(x - x_c) - 1), \mu \in \mathbb{R}$
Запишем ККТ:
1. $c + 2\mu A(x - x_c) = 0$ $2. \mu \geq 0$ $3. \mu((x - x_c)^T A(x - x_c) - 1) = 0$ $4. (x - x_c)^T A(x - x_c) \leq 1$. Как в задаче под номером 4: $x = x_c - \frac{A^{-1}c}{2\mu}$. Подставляем в пункт 3 и тогда: $(x - x_c)^T A(x - x_c) = 1 \to \mu = \frac{\sqrt{c^T A^{-1} c}}{2} > 0$. Тогда: $x^* = x_c - \frac{A^{-1}c}{\sqrt{c^T A^{-1} c}}$ и $p^* = c^T x_c - \sqrt{c^T A^{-1} c}$. GG.

6 Give an explicit solution of the following QP. Там где $x^T B x \to min$

Sol :
$X = 0$ принадлежит допустимому множеству, так как лежит в S и выполняется невероятно сложное условие: $0 < 1$. Рассмотрим произвольную квадратичную форму B, она положительно полуопределена. То есть квадратичная форма является положительно полуопределенной, тогда и только тогда, когда все угловые миноры её матрицы неотрицательны.

Значит минимум достигается только для $x* = 0; p* = 0$.

7 Consider the equality constrained least-squares problem

Задача выпуклая, условие регулярности выполнено, поэтому давайте запишем Лагранжиан: $L(x, \lambda) = ||Ax - b||_2^2 + \lambda^T(Cx - d)$. Условия ККТ: 1. $2A^T(Ax - b) + \lambda(Cx - d) = 0$ 2.$C^T x = d$.
Выразим x во втором и подставим в первое: $x = 0.5(A^TA)^{-1}(2A^Tb - C^T\lambda)$. Подставляем этот x во втрое: $\lambda^* = 2(C(A^TA)^{-1}C^T)(C(A^TA)^{-1}A^Tb - d)$ и $x^* = 0.5(A^TA)^{-1}(2A^Tb - C^T\lambda^*)$
В принципе, это и есть решение двойственной задачи, так как при сильной двойственности наша задача становится $L(x(\lambda), \lambda) \to max$ поэтому дифференцировать будем по всем производным. Именно это сделано, когда в ККТ были взяты производные лагранжиана.
Следовательно, λ^* искомое решение.

8 Derive the KKT conditions for the problem

Sol :
Задача выпуклая, для единичной матрицы выполнено условие Слейтера, пишем ККТ: 1.$X \succ 0$ 2.$Xs = y$ 3.$X^{-1} = I + 0.5(\lambda s^T + s\lambda^T)$ Умножим пункт 2 скалярно на y. Так как $s^Ty = 1$ и $s = X^{-1}y$ имеем $s = y + 0.5(\lambda + (s\lambda^Ty)$. Умножаем: $1 = y^Ty + \lambda^Ty$, $\lambda = -2y + (1 + y^Ty)s$
$X^{-1} = I + 0.5((-2y + (1 + y^Ty)s)s^T + s(-2y + (1 + y^Ty)s)^T) = I + (1 + y^Ty)ss^T - ys^T - sy^T$
Проверим первое условие ККТ: X^* - симметричная, это ясно. Для положительной определённости осталось только посмотреть: $X^* = BB^T, B = I + \frac{ys^T}{\sqrt{s^Ts}}\frac{ss^T}{s^Ts}$
Проверим, что X^* из условия, умножением
$(I + (1 + y^Ty)ss^T - ys^T - sy^T)X^* = (I + yy^T - \frac{ss^T}{s^Ts} + (1 + y^Ty)(ss^T + sy^T - ss^T) - (ys^T + yy^T - ys^T) - (sy^T + y^Tysy^T - \frac{ss^T}{s^Ts}) = I$.

9 Supporting hyperplane interpretation of KKT conditions. Consider a convex problem with no equality constraints

Sol :
Запишем ККТ: 1.$\nabla f_0(x^*) + \sum_j^n = 1\mu_j^*\nabla f_j(x^*) = 0$ 2.$\mu \le 0$ 3.$\mu_j f_j(x^*) = 0$ 4.$f_j(x^*) \le 0$
Для $x \in$ "Бюджетное множество"домножим первое условие скалярно на $(x - x^*)$, наложим 3 и 4 и докажем выпуклость f_j: $\nabla f_0(x^*)^T(x - x^*) = -\sum_{j=1}^n \mu_j^*\nabla f_j(x^*)^T(x - x^*) \ge \sum_{j=1}^n \mu_{j=1}^* f_j(x) - \sum_{j=1}^n \mu_j^* f_j(x^*)^T(x - x^*) = \sum_{j=1}^n \mu_j^*(f_j(x) - f(j(x^*) - f_j(x*)^T(x - x^*) \ge 0$. Доказано. Точка.

2 Duality

1 Fenchel + Lagrange = . Express the dual problem of

Sol :
Запишем функцию Лагранжа: $L(x, \mu) = c^T + \mu f(x),\ \mu \in \mathbb{R}$, при $\mu \ne 0$: $g(\mu) = inf_{x\in\mathbb{R}^n}$; $L(x, \mu) = -sup_{x\in\mathbb{R}^n}(-c^Tx - \mu f(x)) = -\mu \cdot sup(-(\frac{c}{\mu})^Tx - f(x)) = -\mu f^*(-\frac{c}{\mu})$. Допустим, что $\mu = 0$, тогда $g(0) = -\infty$. Но он должен быть конечен.
Запишем двойственную задачу: $-\mu f^*(-\frac{c}{\mu}) \to max_{\mu\in\mathbb{R}}, \mu > 0$. Сопряжённая выпукла, следовательно, целевая вогнута всегда при данных условиях. Так как ищем максимум, то задача выпукла.

2 Minimum volume covering ellipsoid. Let we have the primal problem:

Sol :
$L(X, \mu) = \ln\det X^{-1} + \sum_{i=1}^n \mu_i(a_i^TXa_i - 1),\ \mu \in \mathbb{R}^n$. Лагражиан.
$f^* = -\ln\det(-Y) - n, Y \in -S_{++}^n$ и ∞ иначе, что взято из первой домашки. Так как $tr((a_ia_i^T)X) =< a_ia_i^T, X) \le 1$, то ограничения типа неравенств аффинны и, зная f^*, надо найти двойственную функцию. Надо. Потом найду.

3 A penalty method for equality constraints.
Sol :
Так как x^- - минимум $\phi(x)$ и функция дифференцируема, то $\nabla\phi(x^-) = 0 = \nabla f_0(x^-) + 2\alpha A^T(Ax^- - b)$ $L(x,\lambda) = f_0(x) + \lambda^T(Ax-b), \lambda \in R^m$. Возьмём условие на минимум KKT: $\nabla_x L(x,\lambda) = \nabla f_0(x) + A^T\lambda = 0$. Для x^-, подходящий условиям (*), решением будет $\lambda^- = 2\alpha(Ax^- - b)$.
Запишем двойственную: $g(\lambda) = inf_{x\in R^n}(f_0(x) + \lambda^T(Ax-b))$. Поэтому $g(\lambda^-) = inf_{x\in R^n}(f_0(x) + 2\alpha(Ax^- - b)^T(Ax-b)) = f_0(x^- + 2\alpha||Ax^- - b||_2^2$. Для градиента этого равенства, равного 0, градиент $\phi(x^-) = 0$. Поэтому $\forall x$, удовлетворяющим условиям типа равенств, верно, что $f_0(x) \geq g(\lambda^-)$, который мы нашли. Доказано.

4 Analytic centering.
Sol :
Запишем замену из условия: $1.-\sum_{i=1}^m \ln y_i \to min_{y\in R^m_{++}}$ $2. y = -Ax + b$. Ф-я Ла-жа: $L(x,y,\lambda) = -\sum_{i=1}^m \ln y_i + \lambda^T(y + Ax - b), \lambda \in R^m$. Когда мы берём двойственную функцию, то очевидно, берётся инфимум по обеим переменным, но на x ограничений нет, поэтому $\lambda^T A \neq 0$, то будем брать очень маленькие x или $\lambda_j \leq 0 \quad L \to -\infty$. Для корректности нужны: $A^T\lambda = 0, \lambda \geq 0$;. Найдём наименьшее положение через градиент $y_k + \lambda_k = 0 \to y_k = \frac{1}{\lambda_k}$. Ответ: $\sum_{i=1}^m \ln\lambda_- + m - \lambda^T b, for A^T\lambda = 0, \lambda \geq 0, \lambda \in R^m$ иначе $-\infty$. Двойственная задача: $\sum_{i=1}^m \ln\lambda_i + m - \lambda^T b \to max_{\lambda\in R^m_{++}};\ A^T\lambda = 0$

3 Applications

1 Covers manufacturing. Random Corp is producing covers for following products:
Sol :

$$System = \begin{cases} max & 5y_1 + 7y_2 + 12y_3 \\ S.t & \\ Phones\ max\ per\ week & y_2 \leq 15000 \\ Heads\ max\ per\ week & y_1 \leq 10000 \\ Laptops\ max\ per`week & y_3 \leq 8000 \\ Storage\ max\ per\ week & y_1 * 0.05 + y_2 * 0.06 + 0.22 * y_3 \leq 6000 \\ Heads\ min\ per\ week & y_1 \geq 5000 \\ Laptops\ min\ per\ week & y_3 \geq 4000 \\ Production\ max\ per\ week & \frac{y_1}{30000} + \frac{y_2}{25000} + \frac{y_3}{15000} \leq 1 \\ & y_1, y_2, y_3 \geq 0 \end{cases}$$

В storage обе части были домножены на 100, а в Production на 150000 при реализации, так как лично мне удобнее работать с большими числами, нежели с дробями.

```
[64] from scipy.optimize import linprog
     data = {
         'Heads': {'cost': -5, 'prod_week':5,'storage':5 , 'kostyl':1},
         'Phones': {'cost': -7, 'prod_week':6,'storage':6, 'kostyl1':1 },
         'Laps': { 'cost': -12, 'prod_week':10,'storage':22, 'kostyl2':1 },
     }

     Prod = 150000
     stor = 600000


     C = data.keys()
     model = ConcreteModel()
     model.x = Var(C, domain=NonNegativeReals)
     for c in C:
         print(model.x[c])
     def beer_blend(Prod, stor, data):
         C = data.keys()
         model = ConcreteModel()
         model.x = Var(C, domain=NonNegativeReals)

         model.cost = Objective(expr = sum(model.x[c]*data[c]['cost'] for c in C))

         model.prod = Constraint(expr = Prod >= sum(model.x[c]*data[c]['prod_week'] for c in C))
         model.Phones_max_per_week = Constraint(expr = model.x['Heads'] <= 10000)
         model.heads_max_per_week = Constraint(expr = model.x['Phones'] <= 15000)
         model.Laps_max_per_week = Constraint(expr = model.x['Laps'] <= 8000)

         model.storage = Constraint(expr = stor >= sum(model.x[c]*data[c]['storage'] for c in C))

         model.head_min = Constraint(expr = model.x['Heads'] >= 5000)
         model.lap_min = Constraint(expr = model.x['Laps'] >= 4000)
         #model.prod = Constraint(expr = Prod== sum(model.x[c]*data[c]['prod_week'] for c in C))


         #model.abv = Constraint(expr = 0 == sum(model.x[c]*(data[c]['abv'] - abv) for c in C))
         #model.vol = Constraint(expr = vol == sum(model.x[c] for c in C))

         solver = SolverFactory('cbc')
         solver.solve(model)

         print('Optimal cond')
         for c in data.keys():
             print('  ', c, ':', model.x[c](), 'comupte')
         print()
         #print('Volume = ', model.vol(), 'gallons')
         print('Cost = $', (-1)*model.cost())

     beer_blend(Prod, stor, data)

     x[Heads]
     x[Phones]
     x[Laps]
     Optimal cond
        Heads : 5000.0 comupte
        Phones : 7500.0 comupte
        Laps : 8000.0 comupte

     Cost = $ 173500.0
```

Компутил в колабе, если вы вставите это в Pivko Blending problem, там где финальный кусок с решением, и скомпилируете всё, что выше, то, мамой клянусь, оно выдаст тот же результат.

Сам код приложен в конце файла.

2 Optimal watching TED talks In this task you are to formulate LP problem for selecting TED talks for watching. Take a look at the example , described in the class *Sol* :

Я хочу посмотреть качественные лекции и толки. То есть это видео от 10 минут до 3 часов с максимальным рейтингом, при этом число просмотров может быть любым. Описание желательно не длиннее 501 символа. 5 штук видосов. Стартуем.

```
[130] # reorder results
      variable_name = []
      variable_value = []

      for v in prob.variables():
          variable_name.append(v.name)
          variable_value.append(v.varValue)

      df = pd.DataFrame({'index': variable_name, 'value': variable_value})
      for rownum, row in df.iterrows():
          value = re.findall(r'(\d+)', row['index'])
          df.loc[rownum, 'index'] = int(value[0])

      # df = df.sort_index(by = 'index')
      df = df.sort_values(by = 'index')
      result = pd.merge(data, df, on = 'index')
      result = result[result['value'] == 1].sort_values(by = 'ratings', ascending = False)
      selected_cols_final = ['name', 'event', 'duration', 'ratings']
      final_set_of_talks_to_watch = result[selected_cols_final]
```

```
[131] from IPython.display import display, HTML
      display(HTML(final_set_of_talks_to_watch.to_html(index=False)))
```

name	event	duration	
Jill Sobule: Global warming's theme song, "Manhattan in January"	TED2006	2.7	[{'id': 9, 'name': 'Ingenious', 'count': 45}, {'id': 7, 'name': 'Funny', 'count': 259}, {'id': 1, 'nar 'name': 'Courageous', 'count': 16}, {'id': 23, 'name': 'Jaw-dropping', 'count': 5}, {'id': 26, 'n
Inara George: "Family Tree"	TEDxGreatPacificGarbagePatch	3.3	[{'id': 7, 'name': 'Funny', 'count': 8}, {'id': 23, 'name': 'Jaw-dropping', 'count': 4}, {'id': 'name': 'Courageous', 'count': 5}, {'id': 21, 'name': 'Unconvincing', 'count': 4}, {'id': 26,
Rives: A story of mixed emoticons	TED2008	3.3	[{'id': 7, 'name': 'Funny', 'count': 630}, {'id': 1, 'name': 'Beautiful', 'count': 211}, {'id': 21, 'na {'id': 9, 'name': 'Ingenious', 'count': 258}, {'id': 8, 'name': 'Informative', 'count': 32}, {'id': 2
Andy Hobsbawm: Do the green thing	TED2008	3.4	[{'id': 3, 'name': 'Courageous', 'count': 28}, {'id': 26, 'name': 'Obnoxious', 'count': 7}, { 121}, {'id': 25, 'name': 'OK', 'count': 47}, {'id': 2, 'name': 'Confusing', 'count': 10}, {'id': 2
Raul Midon: "Peace on Earth"	TED2007	9.3	[{'id': 10, 'name': 'Inspiring', 'count': 150}, {'id': 1, 'name': 'Beautiful', 'count': 243}, {'id': {'id': 21, 'name': 'Unconvincing', 'count': 8}, {'id': 23, 'name': 'Jaw-dropping', 'count': 65}, {
Amy Tan: Where does creativity hide?	TED2008	22.9	[{'id': 1, 'name': 'Beautiful', 'count': 416}, {'id': 3, 'name': 'Courageous', 'count': 157}, {'id 'count': 444}, {'id': 25, 'name': 'OK', 'count': 298}, {'id': 2, 'name': 'Confusing', 'count': 204}

4 Covers manufacturing. Random Corp is producing covers for following products:

Посмотрите другую, пожалуйста. Например, первую, она вышла просто отличной, всё компилируется, никаких warnings. Красота!

Кстати, тикеры российских компаний работают некорректно, например, для Фосагро, он же PHOR, значения в таблице были 'nan'.

Спасибо, что проверяете наши работы. :)

In [4]:

```
from scipy.optimize import linprog
data = {
    'Heads': {'cost': -5, 'prod_week':5,'storage':5 , 'kostyl':1},
    'Phones': {'cost': -7, 'prod_week':6,'storage':6, 'kostyl1':1 },
    'Laps': { 'cost': -12, 'prod_week':10,'storage':22, 'kostyl2':1 },
}

Prod = 150000
stor = 600000


C = data.keys()
model = ConcreteModel()
model.x = Var(C, domain=NonNegativeReals)
for c in C:
    print(model.x[c])
def beer_blend(Prod, stor, data):
    C = data.keys()
    model = ConcreteModel()
    model.x = Var(C, domain=NonNegativeReals)

    model.cost = Objective(expr = sum(model.x[c]*data[c]['cost'] for c in C))

    model.prod = Constraint(expr = Prod >= sum(model.x[c]*data[c]['prod_week'] for c in C))
    model.Phones_max_per_week = Constraint(expr = model.x['Heads'] <= 10000)
    model.heads_max_per_week = Constraint(expr = model.x['Phones'] <= 15000)
    model.Laps_max_per_week = Constraint(expr = model.x['Laps'] <= 8000)

    model.storage = Constraint(expr = stor >= sum(model.x[c]*data[c]['storage'] for c in C))

    model.head_min = Constraint(expr = model.x['Heads'] >= 5000)
    model.lap_min = Constraint(expr = model.x['Laps'] >= 4000)
    #model.prod = Constraint(expr = Prod== sum(model.x[c]*data[c]['prod_week'] for c in C))


    #model.abv = Constraint(expr = 0 == sum(model.x[c]*(data[c]['abv'] - abv) for c in C))
    #model.vol = Constraint(expr = vol == sum(model.x[c] for c in C))

    solver = SolverFactory('cbc')
    solver.solve(model)

    print('Optimal cond')
    for c in data.keys():
        print('  ', c, ':', model.x[c](), 'comupte')
    print()
    #print('Volume = ', model.vol(), 'gallons')
```

```
    print('Cost = $', (-1)*model.cost())

beer_blend(Prod, stor, data)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_15808/3232535437.py in <module>
     12 from IPython.display import Image
     13
---> 14 model = ConcreteModel()
     15
     16 # declare decision variables

NameError: name 'ConcreteModel' is not defined
```

In [ ]:

In [ ]:

In [ ]:

Linear programming with `pulp`

PuLP library example (Optimal watching TED talks)

```
!pip install pulp
```

```
Collecting pulp
  Downloading PuLP-2.6.0-py3-none-any.whl (14.2 MB)
     |████████████████████████████████| 14.2 MB 8.9 MB/s
Installing collected packages: pulp
Successfully installed pulp-2.6.0
```

```
%matplotlib inline

import pulp
import numpy as np
import pandas as pd
import re
import matplotlib.pyplot as plt
from IPython.display import Image


# Download the dataset from https://www.kaggle.com/rounakbanik/ted-talks

# Read the dataset into pandas dataframe, convert duration from seconds to minutes
ted = pd.read_csv('https://raw.githubusercontent.com/MerkulovDaniil/sber21_fmin/sources/data/ted_main.csv', encoding='ISO-8859-1')
ted['duration'] = ted['duration'] / 60
ted = ted.round({'duration': 1})

# Select subset of columns & rows (if required)
# data = ted.sample(n=1000) # 'n' can be changed as required
data = ted
selected_cols = ['name', 'event', 'duration', 'ratings']
data.reset_index(inplace=True)
data.head()
```

	index	comments	description	duration	event	film_date	languages	main_speaker	name	num_speaker	published_date	ratin
0	0	4553	Sir Ken Robinson makes an entertaining and pro...	19.4	TED2006	1140825600	60	Ken Robinson	Ken Robinson: Do schools kill creativity?	1	1151367060	[{'id' 'nan 'Funı 'cou 19645},
1	1	265	With the same humor and humanity he exuded in ...	16.3	TED2006	1140825600	43	Al Gore	Al Gore: Averting the climate crisis	1	1151367060	[{'id' 'nan 'Funı 'count': 54
2	2	124	New York Times columnist David Pogue takes aim...	21.4	TED2006	1140739200	26	David Pogue	David Pogue: Simplicity sells	1	1151367060	[{'id' 'nan 'Funı 'count': 96
			In an						.. .			[{'id'

```
# create LP object,
# set up as a maximization problem --> since we want to maximize the number of TED talks to watch
prob = pulp.LpProblem('WatchingTEDTalks', pulp.LpMaximize)


# create decision - yes or no to watch the talk?
decision_variables = []
for rownum, row in data.iterrows():
    variable = str('x' + str(row['index']))
    variable = pulp.LpVariable(str(variable), lowBound = 0, upBound = 1) # make variable binary
    decision_variables.append(variable)

print('Total number of decision variables: ' + str(len(decision_variables)))
```

```
Total number of decision variables: 2550
```

> ***YOUR TASK IS TO CHOOSE YOUR FAVORITE LINEAR LOSS FUNCTION AND BUDGET CONSTRAINTS***

```
# Create optimization Function
```

```
total_views = ''
for rownum, row in data.iterrows():
    for i,talk in enumerate(decision_variables):
        if rownum == i:
            formula = (len(row['description']) + (0)*row['views']) * talk
            total_views += formula

prob += total_views
# print('Optimization function: ' + str(total_views))



# Contraints
total_time_available_for_talks = 3*60
total_time_min_for_talks = 10 # Total time available is 5 hours . Converted to minutes
total_talks_can_watch = 5 # Don't want an overload information


# Create Constraint 1 - Time for talks
total_time_talks = ''
for rownum, row in data.iterrows():
    for i,  talk in enumerate(decision_variables):
        if rownum == i:
            formula = row['duration']*talk
            total_time_talks += formula

prob += (total_time_talks <= total_time_available_for_talks)
prob += (total_time_talks >= total_time_min_for_talks)


# Create Constraint 1 - Time for talks
description_of_talks = ''
for rownum, row in data.iterrows():
    for i,  talk in enumerate(decision_variables):
        if rownum == i:
            formula = len(row['description'])*talk
            description_of_talks += formula

prob += (description_of_talks)
```

```
/usr/local/lib/python3.7/dist-packages/pulp/pulp.py:1704: UserWarning: Overwriting previously set objective.
  warnings.warn("Overwriting previously set objective.")
```

```
# Create Constraint 2 - Number of talks
total_talks = ''

for rownum, row in data.iterrows():
    for i, talk in enumerate(decision_variables):
        if rownum == i:
            formula = talk
            total_talks += formula

prob += (total_talks == total_talks_can_watch)


# Be careful, the output will be huge
# print(prob)
prob.writeLP('WatchingTEDTalks.lp')
print('🤔 The problem has successfully formulated')
```

```
 🤔 The problem has successfully formulated
```

```
optimization_result = prob.solve()

assert optimization_result == pulp.LpStatusOptimal
print('Status:', pulp.LpStatus[prob.status])
print('Optimal Solution to the problem: ', pulp.value(prob.objective))
print('Individual decision variables: ')


for v in prob.variables():
    if v.varValue > 0:
        print(v.name, '=', v.varValue)
```

```
 Status: Optimal
 Optimal Solution to the problem:  3557.0
 Individual decision variables:
 x1998 = 1.0
 x2084 = 1.0
 x2128 = 1.0
```

```
      x2147 = 1.0
      x2153 = 1.0

# reorder results
variable_name = []
variable_value = []

for v in prob.variables():
    variable_name.append(v.name)
    variable_value.append(v.varValue)

df = pd.DataFrame({'index': variable_name, 'value': variable_value})
for rownum, row in df.iterrows():
    value = re.findall(r'(\d+)', row['index'])
    df.loc[rownum, 'index'] = int(value[0])

# df = df.sort_index(by = 'index')
df = df.sort_values(by = 'index')
result = pd.merge(data, df, on = 'index')
result = result[result['value'] == 1].sort_values(by = 'ratings', ascending = False)
selected_cols_final = ['name', 'event', 'duration', 'ratings']
final_set_of_talks_to_watch = result[selected_cols_final]


from IPython.display import display, HTML
display(HTML(final_set_of_talks_to_watch.to_html(index=False)))
```

name	event	duration	ratings
Mary Bassett: Why your doctor should care about social justice	TEDMED 2015	13.8	[{'id': 8, 'name': 'Informative', 'count': 98}, {'id': 21, 'name': 'Unconvincing', 'count': 21}, {'id': 25, 'name': 'OK', 'count': 42}, {'id': 11, 'name': 'Longwinded', 'count': 14}, {'id': 2, 'name': 'Confusing', 'count': 3}, {'id': 26, 'name': 'Obnoxious', 'count': 10}, {'id': 10, 'name': 'Inspiring', 'count': 82}, {'id': 3, 'name': 'Courageous', 'count': 48}, {'id': 24, 'name': 'Persuasive', 'count': 66}, {'id': 1, 'name': 'Beautiful', 'count': 18}, {'id': 22, 'name': 'Fascinating', 'count': 14}, {'id': 7, 'name': 'Funny', 'count': 6}, {'id': 23, 'name': 'Jaw-dropping', 'count': 5}, {'id': 9, 'name': 'Ingenious', 'count': 3}]
Travis Kalanick: Uber's plan to get more people into fewer cars	TED2016	19.3	[{'id': 8, 'name': 'Informative', 'count': 320}, {'id': 9, 'name': 'Ingenious', 'count': 100}, {'id': 22, 'name': 'Fascinating', 'count': 160}, {'id': 10, 'name': 'Inspiring', 'count': 190}, {'id': 24, 'name': 'Persuasive', 'count': 106}, {'id': 25, 'name': 'OK', 'count': 66}, {'id': 21, 'name': 'Unconvincing', 'count': 53}, {'id': 26, 'name': 'Obnoxious', 'count': 40}, {'id': 3, 'name': 'Courageous', 'count': 31}, {'id': 2, 'name': 'Confusing', 'count': 10}, {'id': 11, 'name': 'Longwinded', 'count': 13}, {'id': 1, 'name': 'Beautiful', 'count': 14}, {'id': 23, 'name': 'Jaw-dropping', 'count': 21}, {'id': 7, 'name': 'Funny', 'count': 29}]
Chelsea			[{'id': 11, 'name': 'Longwinded', 'count': 42}, {'id': 21, 'name': 'Unconvincing', 'count': 134}, {'id': 26, 'name': 'Obnoxious', 'count': 60}, {'id': 1, 'name': 'Beautiful', 'count': 116}, {'id': 3, 'name': 'Courageous', 'count': 228}, {'id': 10

✓ 1 сек. выполнено в 15:40