Optimisation in Machine Learning

Lab Assignment 2a

```
Ayush Abrol B20AI052
```

Question 1

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cvxopt as cp
import cvxpy as cvx


c = np.array([20,25,22,28,15,18,23,17,19,17,21,24,25,23,24,24])
b = np.zeros((16, 1))
A = np.negative(np.identity(16,dtype='int'))
beq = np.array([[1],[1],[1],[1],[-1],[-1],[-1],[-1]])
Aeq = np.array([[1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0],
                [0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0],
                [0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0],
                [0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1],
                [-1,-1,-1,-1,0,0,0,0,0,0,0,0,0,0,0,0],
                [0,0,0,0,-1,-1,-1,-1,0,0,0,0,0,0,0,0],
                [0,0,0,0,0,0,0,0,-1,-1,-1,-1,0,0,0,0],
                [0,0,0,0,0,0,0,0,0,0,0,0,-1,-1,-1,-1]])

sol =cp.solvers.lp(cp.matrix(c,tc='d'),cp.matrix(A,tc='d'),cp.matrix(b,tc='d'),
                   cp.matrix(Aeq,tc='d'),cp.matrix(beq,tc='d'))
print(sol['x'],sol['primal objective'])
print("Minimum value of objective function is :",sol['primal objective'])
```

```
     pcost       dcost       gap    pres   dres   k/t
 0:  8.6250e+01  8.6250e+01  2e+01  0e+00  2e-01  1e+00
Terminated (singular KKT matrix).
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
[ 2.50e-01]
 86.25
Minimum value of objective function is : 86.25
```

Question 2

```
A = np.array([[1,1,0,0,0,0,0],
              [1,0,1,-1,0,0,0],
              [0,1,0,0,-1,-1,0],
              [0,0,0,1,0,0,-1],
              [0,0,-1,0,1,0,1],
              [-1,0,0,0,0,0,0],
              [0,-1,0,0,0,0,0],
              [0,0,-1,0,0,0,0],
              [0,0,0,-1,0,0,0],
              [0,0,0,0,-1,0,0],
              [0,0,0,0,0,-1,0],
              [0,0,0,0,0,0,-1]])

c = np.array([[9.2],[-6.0],[-1.3],[4.1],[3.0],[8.0],[-2.1]])

b = np.array([[12],[0],[0],[4],[8],[0],[0],[0],[0],[0],[0],[0]])

sol = cp.solvers.lp(cp.matrix(c,tc='d'), cp.matrix(A,tc='d'),cp.matrix(b,tc='d'))
print(sol['x'],sol['primal objective'])
```

```
     pcost       dcost       gap    pres   dres   k/t
 0:  3.9628e+01 -6.9267e+01  3e+02  8e-01  1e+00  1e+00
 1: -1.7028e+01 -4.4360e+01  8e+01  2e-01  3e-01  4e+00
 2: -2.3828e+01 -2.6997e+01  7e+00  3e-02  3e-02  4e-01
 3: -2.4458e+01 -2.4968e+01  1e+00  4e-03  5e-03  9e-02
 4: -2.4783e+01 -2.4824e+01  1e-01  3e-04  4e-04  9e-03
 5: -2.4800e+01 -2.4800e+01  1e-03  4e-06  4e-06  9e-05
 6: -2.4800e+01 -2.4800e+01  1e-05  4e-08  4e-08  9e-07
Optimal solution found.
[-1.76e-08]
[ 1.20e+01]
[ 4.00e+00]
[ 4.00e+00]
[ 1.20e+01]
[ 5.04e-07]
[ 4.28e-07]
 -24.799998312669278
```

Question 3

```
x1 = cvx.Variable(shape=(2,1), name = 'x')
A1 = np.array([2,3])
```

```
B1 = np.array([4])
r1 = np.array([1,6])
P1 = np.array([[6,2],[2,2]])


constraints1 = [cvx.matmul(A1,x1) >= B1, x1>=0]
objective1 = cvx.Minimize((1/2)*cvx.quad_form(x1,P1)+cvx.matmul(r1,x1)+2)
problem1 = cvx.Problem(objective1,constraints1)
solution = problem1.solve()


print(solution)
print(x1.value)
```

```
11.25
[[0.5]
 [1. ]]
```

Question 4

```
x2 = cvx.Variable(shape=(2,1), name = 'x')
A2 = np.array([[1,1],[2,1]])
B2 = np.array([[2],[3]])
r2 = np.array([2,3])
P2 = np.array([[-2,0],[0,-2]])


constraints2 = [cvx.matmul(A2,x2) <= B2, x2>=0]
objective2 = cvx.Maximize((1/2)*cvx.quad_form(x2,P2)+cvx.matmul(r2,x2))
problem2 = cvx.Problem(objective2,constraints2)
solution = problem2.solve()


print(solution)
print(x2.value)
```

```
3.125
[[0.75]
 [1.25]]
```

● ×