Problem Set 4

4.5 (b) Data Set

Firstly, I choose the hyperparameter (λ & π) for different cluster number K.
When K = 1, we set hyperparameter π = 1, λ = 1.

K = 1	π	λ
London	1	0.92882
Antwerp	1	0.89583

When K = 2, we set hyperparameter π = [0.5,0.5], λ =[1,2].

K = 1	π	λ
London	[0.57883,0.42117]	[0.86540,1.01598]
Antwerp	[0.66110,0.33890]	[0.22974,2.19520]

When K = 3, we set hyperparameter π = [0.33,0.33,0.34], λ =[1,2,3].

K = 1	π	λ
London	[0.475,0.326,0.199]	[0.835,1.006,1.028]
Antwerp	[0.401,0.314, 0.285]	[0.089,0.613,2.344]

When K = 4, we set hyperparameter π = [0.25,0.25,0.25,0.25], λ =[1,2,3,4].

K = 1	π	λ
London	[0.441,0.295,0.171,0.093]	[0.827,0.997,1.020,1.027]
Antwerp	[0.412,0.302,0.158,0.128]	[0.096,0.619,2.339,2.339]

When K = 5, we set hyperparameter π = [0.2,0.2,0.2,0.2, 0.2], λ =[1,2,3,4,5].

K = 1	π	λ
London	[0.426,0.282,0.161,0.087,0.045]	[0.824,0.992,1.016,1.023,1.027]
Antwerp	[0.420,0.293,0.122,0.098,0.067]	[0.101,0.623,2.336,2.336,2.336]

Conclusion

Observe the experiment results, we can find that:

(i) For London, the cells could be approximately divided into two clusters: 45%(π) 0.8(λ); 55%(π) 1.0(λ).

(ii) For Antwerp, the cells could be approximately divided into two clusters: 40%(π) 0.85(λ); 60%(π) 2.3(λ).

Here, λ denotes the mean of number of hits. Therefore, compared to London, Antwerp has a large space that gets more hits. The hit frequency of areas in Antwerp has big difference. Enemies have some target areas in Antwerp. Citizens in Antwerp should avoid staying in this dangerous region. The hit frequency of areas in London tends to be the same (0.8 vs 1.0). It seems that enemies have no target areas in London. Citizens in London should also avoid staying in the more dangerous region.

Codes

Source code can be found at https://github.com/yangji12138/machine-learning/tree/master/PS4.

```
function [cost] = costFunction(X,lambda,pi,gamma)
% Compute cost function
% logq(k,l) N*K
K = length(pi);
N = length(X);
cat = zeros(N,K);
for i = 1:N
    for j = 1:K
        cat(i,j) = pi(1,j).*poisspdf(X(1,i),lambda(1,j));
    end
end
J = gamma.*log(cat);
cost = sum(J(:));
end
```

```
function [gamma] = computeGamma(X,lambda,pi)
% Data Size
N = length(X);
% Number of clusters
K = length(lambda);
gamma = zeros(N,K);
for i = 1:K
    gamma(:,i) = pi(1,i).*poisspdf(X',lambda(1,i));
end
denominator = sum(gamma,2);
for m = 1:K
    for n = 1:N
        gamma(n,m) = gamma(n,m)./denominator(n,1);
    end
end
end
```

```
function [lambda] = computelambda(X,gamma)
% Get K dim
K = size(gamma,2);
% convert X into N*1 vector
input = X';
Data = repmat(input,1,K);
Data_sum = Data.*gamma;
% numerator 1*K vector
numerator = sum(Data_sum);
% Z(k) 1*k
denominator = sum(gamma);
```

```
% divide by elementwise
lambda = numerator./denominator;
end
```

```
function [pi] = computePi(X,gamma)
% Data Size
N = length(X);
% sum by column
temp = sum(gamma);
pi = temp./N;
end
```

```
%% Clear
clear all;
close all;
clc;


% %% London
% u0 = repelem(0,229);
% u1 = repelem(1,211);
% u2 = repelem(2,93);
% u3 = repelem(3,35);
% u4 = repelem(4,7);
% u5 = repelem(5,1);
% X = [u0, u1, u2, u3, u4, u5];

%% Antwerp
u0 = repelem(0,325);
u1 = repelem(1,115);
u2 = repelem(2,67);
u3 = repelem(3,30);
u4 = repelem(4,18);
u5 = repelem(5,21);
X = [u0, u1, u2, u3, u4, u5];

%% Main
cluster = 4;
%pi = repelem(1/cluster,cluster);
%pi = [0.25,0.25,0.25,0.25];
%lambda = [1,1,1.5,2];
pi = [0.2,0.2,0.2,0.2, 0.2];
lambda = [1,2,3,4,5];

gamma = computeGamma(X,lambda,pi);
pi = computePi(X,gamma);
lambda = computelambda(X,gamma);
```

```
J1 = costFunction(X,lambda,pi,gamma);

gamma = computeGamma(X,lambda,pi);
pi = computePi(X,gamma);
lambda = computelambda(X,gamma);
J2 = costFunction(X,lambda,pi,gamma);

% iteration number
iter = 2;
while abs(J2-J1) > 0.01
    gamma = computeGamma(X,lambda,pi);
    pi = computePi(X,gamma);
    lambda = computelambda(X,gamma);
    J1 = J2;
    J2 = costFunction(X,lambda,pi,gamma);
    iter = iter + 1;
end

%g_pi = sprintf('%d',pi);
fprintf('Pi: %8.3f\n', pi);
%g_lambda = sprintf('%d',lambda);
fprintf('Lambda: %8.3f\n', lambda);
g_iter = sprintf('%d',iter);
fprintf('iteration number is:%s\n', g_iter);
```