Μηχανική Μάθηση

3ο Σετ Ασκήσεων

ΟΝΟΜΑ: Αλέξανδρος
ΕΠΩΝΥΜΟ: Καλέργης
ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ: 1066665
ΕΤΟΣ ΦΟΙΤΗΣΗΣ: 4ο

Πρόβλημα 3.1

Σε αυτό το πρόβλημα θέλουμε να αξιολογήσουμε τις δυνατότητες της πρώτης μεθόδου Kernel στο να προσεγγίζει πυκνότητες πιθανότητας. Για τον σκοπό αυτό δημιουργούμε 1000 υλοποιήσεις μιας τυχαίας μεταβλητής ομοιόμορφα κατανεμημένης στο διάστημα [0, 1]. Προσεγγίζουμε την πυκνότητα πιθανότητας χρησιμοποιώντας το Gaussian kernel:

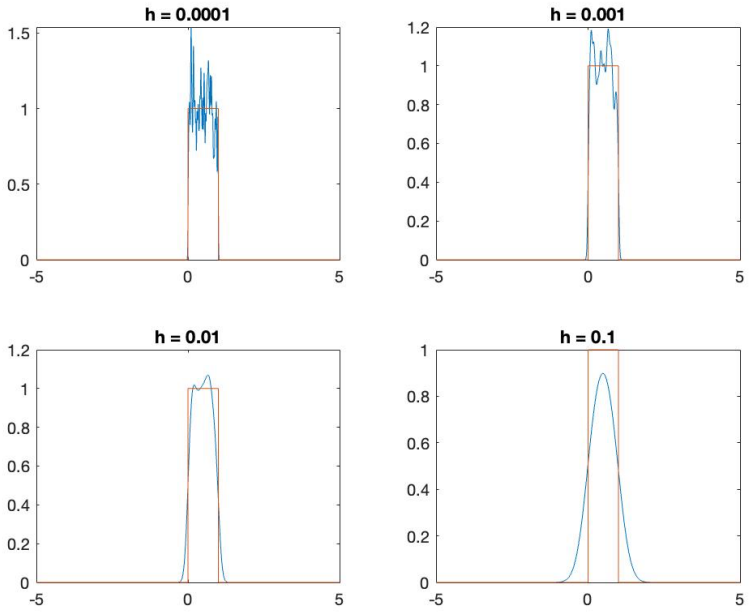
$$K(x,h) = \frac{1}{\sqrt{2\pi h}} e^{-\frac{1}{2h}x^2}$$

Η προσέγγιση της συνάρτησης πυκνότητας πιθανότητας γίνεται μέσω της παρακάτω σχέσης:

$$\hat{f}(x) = \frac{K(x - x_0, h) + \cdots + K(x - x_{N-1}, h)}{N}$$

Όπου x_i είναι οι διαθέσιμες υλοποιήσεις που έχουμε δημιουργήσει και N το πλήθος τους.

Δοκιμάζουμε να προσεγγίσουμε τη πυκνότητα πιθανότητας με h = 0.0001, 0.001, 0.01, 0.1 και παίρνουμε τα αποτελέσματα που φαίνονται στο σχήμα παρακάτω.



Σχήμα 1: Συνάρτηση πυκνότητας πιθανότητας για διάφορα h

Παρατηρούμε ότι όσο πιο μικρό είναι το *h* προσεγγίζουμε καλύτερα το πεδίο ορισμού της *f(x)* αλλά όχι τις τιμές της, ενώ για μεγάλο *h* γίνεται το αντίθετο. Δηλαδή για μικρό *h* βλέπουμε ότι η καμπύλη ξεκινάει στο 0 και τελειώνει στο 1 αλλά οι τιμές της έχουν spikes, ενώ για μεγάλο η τιμή είναι κοντά στο 1 αλλά η καμπύλη ξεκινάει εκτός των 0 και 1. Από τις παραπάνω εικόνες βλέπουμε ότι η καμπύλη για *h=0.01* είναι μια καλή προσέγγιση (με κόκκινο φαίνεται η ιδανική *f(x)*).

Πρόβλημα 3.2

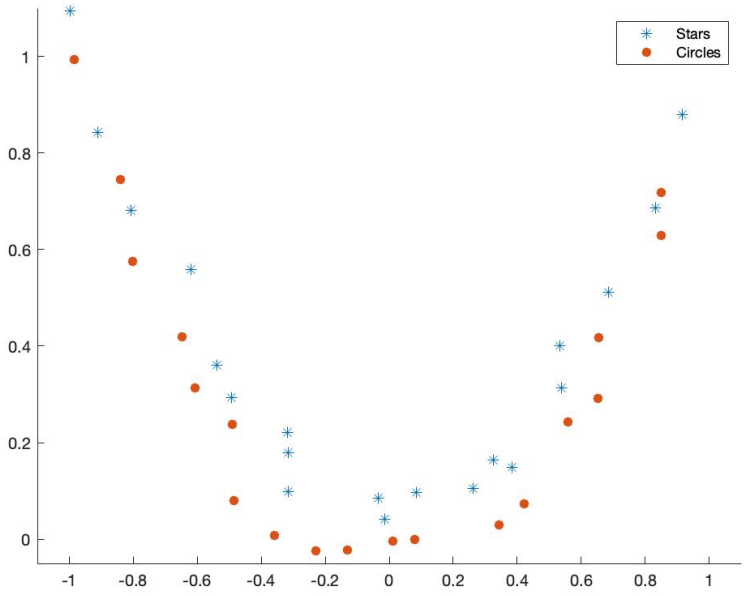
Σε αυτό το πρόβλημα μας δίνεται ένα αρχείο με δεδομένα που περιέχουν δύο δι-διάστατα διανύσματα stars και circles. Επιθυμούμε να κατασκευάσουμε έναν classifier που να διαχωρίζει ποιο στοιχείο ανήκει σε ποιο σύνολο (stars,circles). Γι αυτό το σκοπό αντιστοιχούμε την ετικέτα "1" στο stars και την "-1" στο circles και ψάχνουμε μία συνάρτηση $\varphi(X), X = [x_1, x_2]^T$ που εάν την εφαρμόσουμε στα δεδομένα θα υλοποιεί την εν λόγω αντιστοίχηση. Για να βρούμε τη επιλύουμε το παρακάτω πρόβλημα βελτιστοποίησης:

$$min_{\varphi \in V} \left\{ \sum_{X_i \in stars} \left(1 - \varphi(X_i)\right)^2 + \sum_{X_j \in circles} \left(1 - \varphi(X_j)\right)^2 + \lambda \|\varphi(X)\|^2 \right\} =$$

$$min_{\varphi \in V} \left\{ \sum_{i=0}^{N-1} \left(Y_i - \varphi(X_i)\right)^2 + \lambda \|\varphi(X)\|^2 \right\}, \qquad N = N_{stars} + N_{circles}$$

Όπου V είναι ο διανυσματικός χώρος των συναρτήσεων που ορίζονται με τη βοήθεια του Gaussian Kernel:

$$K(x, h) = e^{-\frac{1}{h}\|X-Y\|^2}$$



Σχήμα 2: Δεδομένα του Προβλήματος 3.2

α) Έχουμε τα $X_0, X_1, \dots, X_{N-1} \in \mathbb{R}^2$,που είναι τα παραδείγματα από αστέρια και κύκλους και τα αντιστοιχίζουμε σε:

$$Z_0, Z_1, \dots, Z_{N-1} = K(X,\ X_0), K(X,\ X_1), \dots, K(X,\ X_{N-1}) \in V$$

Ορίζουμε τη γραμμική θήκη $\Omega = \{c_0 Z_0,\ c_1 Z_1, \dots, c_{N-1} Z_{N-1}\}$.

Αφού $\varphi(X) \in V$, τότε ορίζουμε την κάθετη προβολή $\hat{\varphi}(X) \in \Omega$, έτσι από την αρχή της ορθογωνιότητας έχουμε ότι:

$$\langle\ \varphi(X) - \hat{\varphi}(X),\ K(X,\ X_i)\ \rangle = 0 \iff$$

$$\langle\ \varphi(X)K(X,\ X_i)\ \rangle = \langle\ \hat{\varphi}(X)K(X,\ X_i)\ \rangle \iff$$

$$\varphi(X_i) = \hat{\varphi}(X_i)$$

Άρα μπορούμε σε κάθε σημείο $X_0, X_1, \dots, X_{N-1}$ να αντικαταστήσουμε τη $\varphi(X)$ με την κάθετη προβολή $\hat{\varphi}(X)$, δηλαδή αν:

$$\varphi(X) = \sum_{i=0}^{M-1} \overline{c_i} K(X, Z_i)\,,\ \ M \le N$$

Τότε μπορούμε να το αντικαταστήσουμε με:

$$\hat{\varphi}(X) = \sum_{i=0}^{N-1} c_i K(X, X_i) = \sum_{X_i \in stars} \alpha_i K(X, X_i) \quad + \sum_{X_j \in circles} \beta_i K(X, X_j)$$

β) Ισχύει ότι:

$$\|\varphi(X)\|^2 = \|\hat{\varphi}(X) + \varphi(X) - \hat{\varphi}(X)\|^2 = \|\hat{\varphi}(X)\|^2 + \|\varphi(X) - \hat{\varphi}(X)\|^2 + 2\langle\ \varphi(X), \hat{\varphi}(X) - \hat{\varphi}(X)\rangle$$

Όμως γνωρίζουμε πως το $\hat{\varphi}(X)$ είναι κάθετο με το $\varphi(X) - \hat{\varphi}(X)$, επομένως η παραπάνω σχέση γίνεται:

$\langle\ \varphi(X), \hat{\varphi}(X) - \hat{\varphi}(X)\rangle = 0$ και $\|\varphi(X)\|^2 = \|\hat{\varphi}(X)\|^2 + \|\varphi(X) - \hat{\varphi}(X)\|^2 \ge \|\hat{\varphi}(X)\|^2$

Άρα αφού ισχύει η παραπάνω ανισότητα μπορούμε να αντικαταστήσουμε το $\|\varphi(X)\|^2$ με το $\|\hat{\varphi}(X)\|^2$ αφού είναι μικρότερο και θέλουμε να ελαχιστοποιήσουμε τη σχέση.

γ) Σε αυτό το ερώτημα θέλουμε να λύσουμε το πρόβλημα βελτιστοποίησης

$$min_{\hat{\varphi}\in\Omega}\left\{\sum_{i=0}^{N-1}\left(Y_i-\hat{\varphi}(X_i)\right)^2 \;+\; \lambda\|\hat{\varphi}(X)\|^2\right\}, \qquad (1)$$

$$\hat{\varphi}(X)=\sum_{i=0}^{N-1}c_iK(X,X_i)=c_0K(X,\,X_0)+c_1K(X,\,X_1)+\cdots+c_{N-1}K(X,\,X_{N-1})$$

Έστω ότι ορίζουμε ως διανύσματα **X** και **Y** τα δεδομένα και τα labels αντίστοιχα, τότε:

$$\mathbf{X}=\;[X_0 \quad X_0 \;\ldots\; X_{N-1}]^T$$

$$\mathbf{Y}=\;[X_0 \quad X_0 \;\ldots\; X_{N-1}]^T$$

Τότε

$$\hat{\varphi}(X)=\begin{bmatrix} K(X_0,\,X_0) & K(X_0,\,X_1) & \cdots & K(X_0,\,X_{N-1}) \\ K(X_1,\,X_0) & K(X_1,\,X_1) & & K(X_1,\,X_{N-1}) \\ \vdots & & \ddots & \vdots \\ K(X_{N-1},\,X_0) & K(\,X_{N-1},\,X_1) & \cdots & K(X_{N-1},\,X_{N-1}) \end{bmatrix}\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{bmatrix}=\boldsymbol{KC}$$

Οπότε

$$\sum_{i=0}^{N-1}\left(Y_i-\hat{\varphi}(X_i)\right)^2=\;\|\boldsymbol{Y}-\boldsymbol{KC}\|^2$$

Θα βρούμε τις μερικές παραγώγους της σχέσης που θέλουμε να ελαχιστοποιήσουμε ως προς το διάνυσμα **C** που θέλουμε να βρούμε

$$\frac{\partial}{\partial \boldsymbol{C}}\sum_{i=0}^{N-1}\left(Y_i-\hat{\varphi}(X_i)\right)^2=\;\frac{\partial}{\partial \boldsymbol{C}}\|\boldsymbol{Y}-\boldsymbol{KC}\|^2=\;-2\boldsymbol{K}(\boldsymbol{Y}-\boldsymbol{KC}) \qquad (2)$$

&

$$\frac{\partial}{\partial \boldsymbol{C}}\lambda\|\varphi(X)\|^2=2\lambda\boldsymbol{KC} \qquad (3)$$

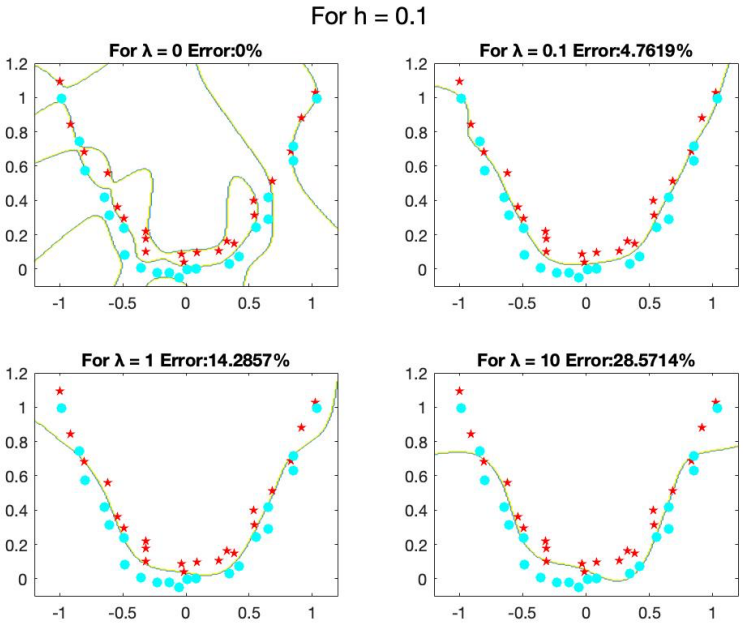
Από τις παραπάνω σχέσεις βρίσκουμε την εξίσωση που πρέπει να λύσουμε για να βρούμε το διάνυσμα παραμέτρων:

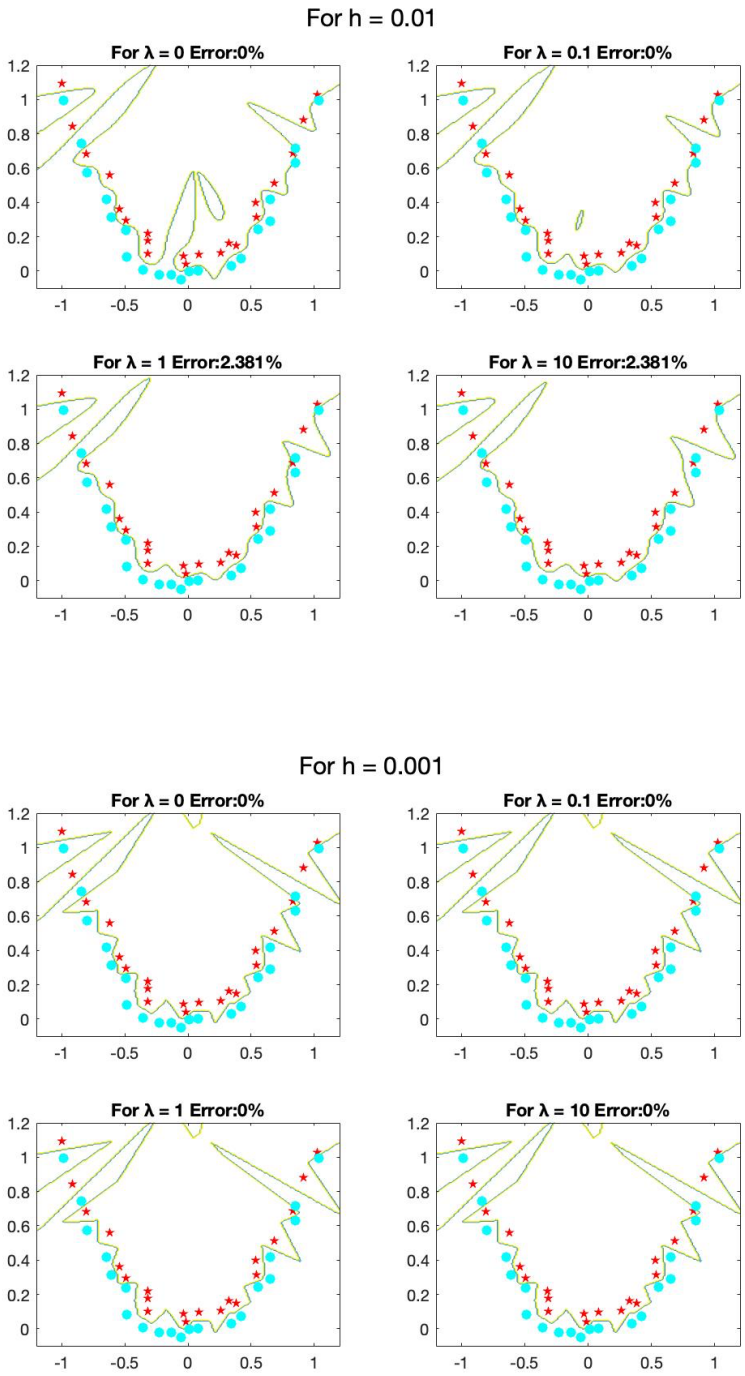
$$-2\boldsymbol{K}(\boldsymbol{Y}-\boldsymbol{KC})+2\lambda\boldsymbol{KC}=0 \Leftrightarrow \boldsymbol{C}=(\boldsymbol{KC}+\lambda\boldsymbol{I})^{-1}\boldsymbol{Y}$$

Επειδή το διάνυσμα X περιέχει πρώτα το σύνολο των stars και μετά σύνολο circles($X = \{X_{stars}, X_{circle}\}$), τα πρώτα 21 c_i που θα υπολογίσουμε αποτελούν τα α_i και τα υπόλοιπα 21 c_i θα αποτελούν τα β_i.

δ) Για να κατηγοριοποιήσουμε ένα νέο σημείο X_{new} θα βρούμε τη τιμή του $\hat{\varphi}(X_{new})$και θα τη συγκρίνουμε με το 0. Εάν είναι μεγαλύτερη σημαίνει ότι το νέο σημείο ανήκει στη κατηγορία starts ενώ αν είναι μικρότερη ανήκει στη κατηγορία circles.

ε) Τα αποτελέσματα του διαχωριστικού συνόρου που υπολογίσαμε για διάφορες τιμές του h και του λ. Τα ποσοστά σφάλματος είναι συγκεντρωμένα στο παρακάτω πίνακα καθώς και η ποιότητα του συνόρου φαίνεται παρακάτω στα σχήματα. Ο τρόπος υλοποίησης φαίνεται στον κώδικα του προβλήματος που βρίσκεται στο τέλος της αναφοράς.



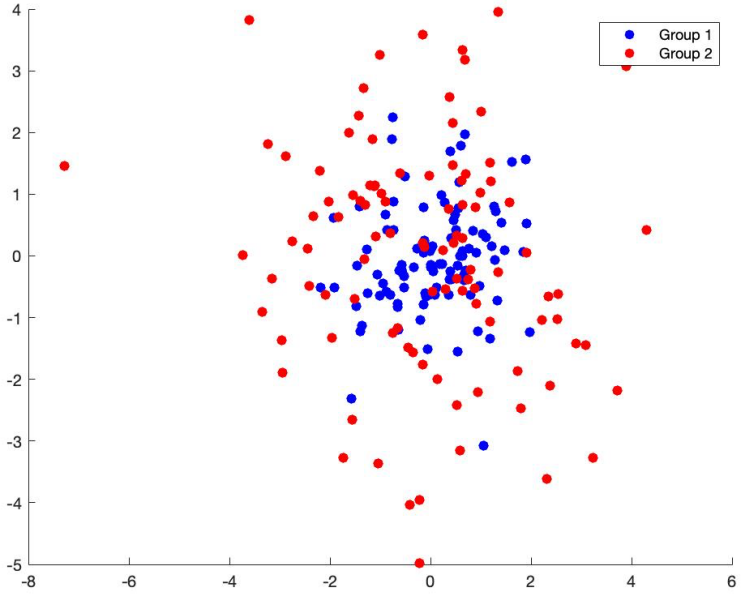


Σχήμα 3: Ποιοτικά αποτελέσματα του διαχωριστικού συνόρου κατηγοριοποίησης

h \ λ	0	0.1	1	10
0.001	0%	0%	0%	0%
0.01	0%	0%	2.38%	2.38%
0.1	0%	4.76%	14.29%	28.57%

Πρόβλημα 3.3

Σε αυτό το πρόβλημα μας δίνεται ένα αρχείο το οποίο περιέχει N = 200 διανύσματα μήκους 2, τα οποία επιθυμούμε να ομαδοποιήσουμε σε δύο ομάδες.



Σχήμα 4: Δεδομένα του Προβλήματος 3.3

α) Σε αυτό το ερώτημα θέλουμε να κατασκευάσουμε τον αλγόριθμο KMeans ο οποίος αποτελείται από τα παρακάτω βήματα και η υλοποίηση του κώδικα φαίνεται στο τέλος της αναφοράς:

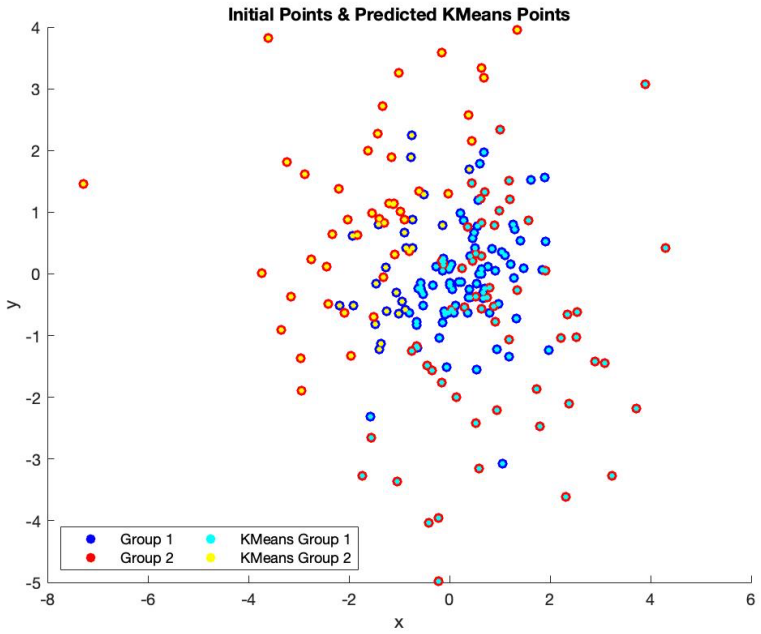
Βήμα 1ο → Επιλέγω K τυχαίους αντιπροσώπους $Z_1, Z_2, \ldots, Z_K$.

Βήμα 2ο → Δημιουργώ τις ομάδες $C_1, C_2, \ldots, C_K$ συμπεριλαμβάνοντας σε κάθε ομάδα C_i όλα τα σημεία που απέχουν λιγότερο από το Z_i από ότι τους άλλους αντιπροσώπους.

Βήμα 3ο → Χρησιμοποιώντας τα σημεία κάθε ομάδας δημιουργώ το νέο αντιπρόσωπο παίρνοντας το μέσο όρο των δεδομένων της ομάδας.

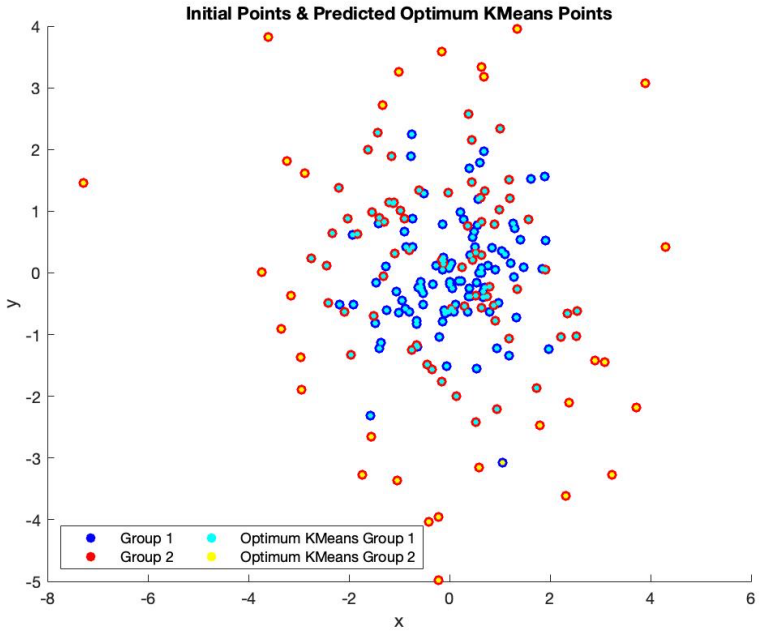
Βήμα 4ο → Εκτελώ επαναληπτικά τα Βήματα 2 και 3 έως ότου ο αλγόριθμος να συγκλίνει.

Τρέχοντας τον αλγόριθμο παίρνω σφάλμα 39.5%. Στο Σχήμα 5 φαίνονται τα αποτελέσματα του αλγορίθμου.



Σχήμα 5: Αλγόριθμος KMeans στα δεδομένα του Προβλήματος 3.3

β) Για να βελτιώσουμε την απόδοση του KMeans επεκτείνουμε τη διάσταση των παραδειγμάτων δημιουργώντας μία νέα συντεταγμένη όπου κάθε δισδιάστατο X_i αντικαθίσταται από το $\{X_i, \|X_i\|^2\}$. Με αυτόν το τρόπο τα σημεία που είναι κοντά στην αρχή των αξόνων στις τρεις διαστάσεις θα είναι πιο κοντά στο οριζόντιο επίπεδο από ό,τι σημεία που βρίσκονται μακρύτερα.Τρέχοντας τον αλγόριθμο παίρνω σφάλμα 35.5%. Στο Σχήμα 6 φαίνονται τα αποτελέσματα του αλγορίθμου.



Κώδικας Προβλήματος 3.1

```
close all;
clear;
clc;

N = 1000;
samples = 4*N;
X = rand(1,N);
xaxis = linspace(-2,3,samples);
X = xaxis' - X;
h = [0.0001 0.001 0.01 0.1];

for i = 1:length(h)
    f = mean(K(X,h(i)),2);
    subplot(2,2,i)
    plot(xaxis,f)
    hold on
    syms x
    fplot(rectangularPulse(0,1,x))
    txt = sprintf("h = %g",h(i));
    title(txt)
end

function GaussianKernel = K(X,h)
    GaussianKernel = ( 1/sqrt(2*pi*h) )*( exp( -1/(2*h)*X.^2 ) );
end
```

Κώδικας Προβλήματος 3.2

```
close all;
clear;
clc;
%========= Load Data =========
data32 = load('data32.mat');
stars = data32.stars;
circles = data32.circles;
%========== Plot Data =========
figure(1)
scatter(stars(:,1),stars(:,2),'*')
hold on
scatter(circles(:,1),circles(:,2),'filled')
legend("Stars","Circles")
xlim([-1.1 1.1])
ylim([-0.05 1.1])
%====== Calculate Errors ======
h = [0.001 0.01 0.1];
l = [0 0.1 1 10];
stars_n = size(stars, 1);
circles_n = size(circles,1);
star_labels = ones([1, stars_n]); % Label Stars as 1
circle_labels = -ones([1, circles_n]); % Label Circles as -1
X = [stars; circles];
Y = [star_labels circle_labels]';
for i=1:size(h,2)
    for j=1:size(l,2)
        x = reshape(X' , [1, 2, 42]) - X;
        T = [];

        for k=1:2*size(circles,1)
            T = [T K(x(:,:,k),h(i))];
        end
        C = inv( T  + l(j).*eye(stars_n + circles_n) ) * Y;
        a = C(1:stars_n)';
        b = C(circles_n+1:end)';

        star_error = Error(stars, stars, circles, a, b, h(i));
        star_error = star_error(2);

        circle_error = Error(circles, stars, circles, a, b, h(i));
        circle_error = circle_error(1);

        total_error = (star_error + circle_error)/(stars_n +circles_n);

        %============= Plot ============
        lin_x = linspace(-1.2, 1.2, 200);
        lin_y = linspace(-0.1, 1.2, 200);
        [x_meshgrid, y_meshgrid] = meshgrid(lin_x, lin_y);

        x = reshape(x_meshgrid', [size(x_meshgrid,1)^2, 1]);
        y = reshape(y_meshgrid', [size(y_meshgrid,1)^2, 1]);
```

```
        x = [x y];
        n = size(x,1);

        T_circ = [];
        x_circ = reshape(x' , [1, 2, n]) - circles;
        T_star = [];
        x_star = reshape(x' , [1, 2, n]) - stars;
        for k=1:n
            T_circ = [T_circ K(x_circ(:,:,k),h(i))];
            T_star = [T_star K(x_star(:,:,k),h(i))];
        end
        t = sum(a * T_star,1) +  sum(b * T_circ,1);
        t = reshape(t,[length(lin_x),length(lin_y)])';
        t(t>0) = 1;
        t(t<0) = -1;

        if i==1
            figure(2)
            subplot(size(l,2)/2, size(l,2)/2, j)
            contour(x_meshgrid,y_meshgrid,t);
            hold on;
            scatter(stars(:,1), stars(:,2), 40, 'rp', 'filled')
            hold on;
            scatter(circles(:,1), circles(:,2), 40, 'cyan', 'filled')
            title(['For λ = ' num2str(l(j)) ' Error:'
num2str(total_error*100) '%']);
            sgtitle(['For h = ' num2str(h(i))])
        end
        if i==2
            figure(3)
            subplot(size(l,2)/2, size(l,2)/2, j)
            contour(x_meshgrid,y_meshgrid,t);
            hold on;
            scatter(stars(:,1), stars(:,2), 40, 'rp', 'filled')
            hold on;
            scatter(circles(:,1), circles(:,2), 40, 'cyan', 'filled')
            title(['For λ = ' num2str(l(j)) ' Error:'
num2str(total_error*100) '%']);
            sgtitle(['For h = ' num2str(h(i))])
        end
        if i==3
            figure(4)
            subplot(size(l,2)/2, size(l,2)/2, j)
            contour(x_meshgrid,y_meshgrid,t);
            hold on;
            scatter(stars(:,1), stars(:,2), 40, 'rp', 'filled')
            hold on;
            scatter(circles(:,1), circles(:,2), 40, 'cyan', 'filled')
            title(['For λ = ' num2str(l(j)) ' Error:'
num2str(total_error*100) '%']);
            sgtitle(['For h = ' num2str(h(i))])
        end
    end
end
```

```
%========== Functions ==========
%----- Kernel Function -----
function y = K(X, h)
    nrm = vecnorm(X,2,2);
    y  = exp((-1/h) * nrm.^2);
end
%----- Error -----
function y = Error(X, stars, circles, a, b, h)
    y = [0 0];
    n = size(X,1);

    T_circ = [];
    x_circ = reshape(X' , [1, 2, 21]) - circles;

    T_star = [];
    x_star = reshape(X' , [1, 2, 21]) - stars;

    for k=1:size(circles,1)
        T_circ = [T_circ K(x_circ(:,:,k),h)];
        T_star = [T_star K(x_star(:,:,k),h)];
    end

    t = sum(a * T_star,1) +  sum(b * T_circ,1);

    y(1) = nnz(t>0); %this is when we have erros for circles
    y(2) = nnz(t<0); %this is when we have erros for stars
end
```

Κώδικας Προβλήματος 3.3

```
close all;
clear;
clc;

%========= Load Data =========
data33 = load('data33.mat');
X = data33.X;
%========== Plot Data =========
figure()
scatter(X(1,1:100), X(2,1:100), 40, 'blue', 'filled')
hold on;
scatter(X(1,101:end), X(2,101:end), 40, 'red', 'filled')
hold on;
legend("Group 1","Group 2")
```

```
%========== KMeans ==========
K = 2;
iterations = 30;
[C,Z] = KMeans(X, K, iterations);
disp(['KMeans Classification
Error:',num2str(classification_error(X,Z)*100),'%'])
%======= Optimum_KMeans =======
X_norm = vecnorm(X,2,1);
X_optimum = [X; X_norm.^2];
[C_optimum,Z_optimum] = KMeans(X_optimum, K, iterations);
disp(['Optimum KMeans Classification
Error:',num2str(classification_error(X_optimum,Z_optimum)*100),'%'])
%========== Plots ==========
Plot_points = cell(1,K);
Plot_optimum_points = cell(1,K);
for i=1:size(C,2)
    Group_Points_Index = C{i};
    Group_opt_Points_Index = C_optimum{i};
    Group_Points = X(:, Group_Points_Index);
    Class_optimum_points = X(:,Group_opt_Points_Index);
    Plot_points{i} = Group_Points;
    Plot_optimum_points{i} = Class_optimum_points;
end
%----- KMeans -----
figure()
scatter(X(1,1:100), X(2,1:100), 40, 'blue', 'filled')
hold on;
scatter(X(1,101:end), X(2,101:end), 40, 'red', 'filled')
hold on;
scatter(Plot_points{1}(1,:), Plot_points{1}(2,:), 12, 'cyan', 'filled')
hold on;
scatter(Plot_points{2}(1,:), Plot_points{2}(2,:), 12, 'yellow', 'filled')
title('Initial Points & Predicted KMeans Points')
xlabel('x')
ylabel('y')
legend({'Group 1','Group 2', 'KMeans Group 1', 'KMeans Group 2'},...
    'Location','southwest','NumColumns',2)
%----- Optimum_KMeans -----
figure()
scatter(X(1,1:100), X(2,1:100), 40, 'blue', 'filled')
hold on;
scatter(X(1,101:end), X(2,101:end), 40, 'red', 'filled')
hold on;
scatter(Plot_optimum_points{1}(1,:), Plot_optimum_points{1}(2,:), 12,
'cyan', 'filled')
hold on;
scatter(Plot_optimum_points{2}(1,:), Plot_optimum_points{2}(2,:), 12,
'yellow', 'filled')
title('Initial Points & Predicted Optimum KMeans Points')
xlabel('x')
ylabel('y')
legend({'Group 1','Group 2', 'Optimum KMeans Group 1', 'Optimum KMeans
Group 2'},...
    'Location','southwest','NumColumns',2)
```

```
%========== Functions ==========
%----- KMeans -----
function [c,y] = KMeans(X, K, iterations)
    sz = size(X);
    Z = zeros(sz(1), K);
    Xt = X';
    for i=1:K
        Z(:,i) = X(:, randi(sz(2)) );
    end
    for k = 1:iterations
        Zt = Z';
        C = cell(1:K);
        for i=1:length(Xt)
            nrm = vecnorm(Zt-Xt(i,:),2,2);
            mikro = min(nrm);
            class_i = find(nrm == mikro);
            C{class_i} = [C{class_i} i];
        end
        % Changing the centers
        for i=1:K
            Xmo = [];
            for j = 1:length(C{i})
                mo = C{i}(j);
                Xmo = [Xmo X(:,mo)];
            end
            Z(:,i) = mean(Xmo,2);
        end
    end
    y = Z;
    c = C;
end
%----- Classification Error -----
function y = classification_error(X,Z)
    sz = size(X);
    Zt = Z';
    Xt = X';
    tot_err = 0;
    for i=1:length(Xt)
        nrm = vecnorm(Zt-Xt(i,:),2,2);
        mikro = min(nrm);
        class_i = find(nrm == mikro);
        if class_i == 1 && i >= 100
            tot_err = tot_err + 1;
        elseif class_i == 2 && i < 100
            tot_err = tot_err + 1;
        end
    end
    y = tot_err/sz(2);
end
```