Tarea 3 - Reconocimiento estadístico de patrones

Erick Salvador Alvarez Valencia

CIMAT A.C.,
erick.alvarez@cimat.mx

Resumen En el presente reporte se hablará sobre la resolución de los ejercicios de la tercer tarea de reconocimiento de patrones, de la misma manera se presentará el código de los mismos y los resultados que fueron obtenidos en los problemas que lo requieren.

1. Problema 1

En este ejercicio demostramos lo que faltó del algoritmo EM. Partimos de la siguiente propiedad para dos densidades g_1, g_2:

$$E_{g_1} ln(g_1(X)) \geq E_{g_1} ln(g_2(X)).$$

donde referimos con por ejemplo $E_{g_1} ln(g_1(X))$ a $E_X ln(g_1(X))$ donde X tiene densidad g_1.

Usando la notación de las notas, para θ, θ_n dadas, construye las siguientes densidades:

$$g_1 = fZ^m|Z = z, \theta^n, \ g_2 = fZ^m|Z = z, \theta$$

donde $Z^m|Z, \theta^n$ denota la distribución condicional tomando el valor de los parámetros de esta distribución igual a θ^n. Verifica que

$$Q(\theta|\theta^n) - ln(\theta, Z = z) \leq Q(\theta^n|\theta^n) - ln + (\theta^n, Z = z)$$

Solución : Empezamos definiendo la esperanza de la función g_1

$$\begin{aligned} g_1 &= fZ^m|Z = z, \theta^n \\ E_{g_1}[ln(g_1(X))] &= E_X[ln(g_1(X))] \\ &= E_X[ln(fZ^m|Z = z, \theta^n)] \\ &= E_X[ln(\frac{fZ^m, Z = z, \theta^n}{fZ = z, \theta^n})] \\ &= E_X[ln(fZ^m, Z = z, \theta^n)] - E_X[ln(fZ = z, \theta^n)] \\ &= \int_I g_1(x) ln(fZ^m, Z = z, \theta^n) dx - ln(Z = z, \theta^n) \end{aligned} \tag{1}$$

Para $I \in dom\{X\}$.

Por otra parte tenemos:

$$Q(\theta^n|\theta^n) = E_{\theta^n}[l_0(\theta^n, T)|Z = z] \tag{2}$$

Donde

$$\begin{aligned} E_{\theta^n}[l_0(\theta^n, T)|Z = z] &= E_{\theta^n}[ln(f(X, Z = z, \theta^n))] \\ &= \int_I g_1(x) ln(f(X, Z = z, \theta^n)) dx \end{aligned} \tag{3}$$

Hemos llegado a la expresión que teníamos en (1) por lo cual demostramos que $E_{g_1}[ln(g_1(X))] = Q(\theta^n|\theta^n) - ln(Z = z, \theta^n)$.

Ahora, aplicamos el mismo análisis usando la función g_2.

$$\begin{aligned} g_2 &= fZ^m|Z = z, \theta \\ E_{g_1}[ln(g_2(X))] &= E_X[ln(g_2(X))] \\ &= E_X[ln(fZ^m|Z = z, \theta)] \\ &= E_X[ln(\frac{fZ^m, Z = z, \theta}{fZ = z, \theta})] \\ &= E_X[ln(fZ^m, Z = z, \theta)] - E_X[ln(fZ = z, \theta)] \\ &= \int_I g_1(x) ln(fZ^m, Z = z, \theta) dx - ln(Z = z, \theta) \end{aligned} \tag{4}$$

Para $I \in dom\{X\}$.

Por otra parte tenemos:

$$Q(\theta|\theta^n) = E_\theta[l_0(\theta, T)|Z = z] \tag{5}$$

Donde

$$\begin{aligned} E_\theta[l_0(\theta, T)|Z = z] &= E_\theta[ln(f(X, Z = z, \theta))] \\ &= \int_I g_1(x) ln(f(X, Z = z, \theta)) dx \end{aligned} \tag{6}$$

Hemos llegado a la expresión que teníamos en (3) por lo cual demostramos que $E_{g_1}[ln(g_2(X))] = Q(\theta|\theta^n) - ln(Z = z, \theta)$.

Finalmente, por la desigualdad marcada en la descripción

$$E_{g_1} ln(g_1(X)) \geq E_{g_1} ln(g_2(X))$$

nos queda

$$Q(\theta^n|\theta^n) - ln(\theta^n, Z = z) \geq Q(\theta|\theta^n) - ln(\theta, Z = z)$$

.

Muestra que lo anterior implica que EM es un algoritmo MM.

Solución : En el algoritmo EM definimos una función que minoriza la log-verosimilitud y lo que buscamos es maximizar dicha función, por lo cual se tienen que cumplir los siguientes puntos:

1. $g(\theta^n|\theta^n) = f(\theta^n)$

2. $g(\theta|\theta^n) \geq f(\theta)$

Para $f(\theta)$ y $g(\theta|\theta^n)$ dadas.

Definimos $f(\theta)$ como $ln(\theta, Z = z)$ y $g(\theta|\theta^n)$ como $Q(\theta|\theta^n) - Q(\theta^n|\theta^n) + ln(\theta^n, Z = z)$. Esto anterior es válido ya que para $Q(\theta^n|\theta^n) + ln(\theta^n, Z = z)$ la θ^n ya está dada. Verificamos el primer punto.

$$\begin{aligned} Q(\theta^n|\theta^n) - ln(\theta^n, Z = z) &\leq Q(\theta^n|\theta^n) - ln(\theta^n, Z = z) \\ ln(\theta^n, Z = z) &\geq Q(\theta^n|\theta^n) - Q(\theta^n|\theta^n) + ln(\theta^n, Z = z) \\ ln(\theta^n, Z = z) &= ln(\theta^n, Z = z) \end{aligned} \tag{7}$$

Se demostró que $f(\theta^n) = g(\theta^n|\theta^n)$.
Ahora para el segundo punto tenemos.

$$\begin{aligned} Q(\theta|\theta^n) - ln(\theta, Z = z) &\leq Q(\theta^n|\theta^n) - ln(\theta^n, Z = z) \\ ln(\theta, Z = z) &\geq Q(\theta|\theta^n) - Q(\theta^n|\theta^n) + ln(\theta^n, Z = z) \\ f(\theta) &\geq g(\theta|\theta^n) \end{aligned} \tag{8}$$

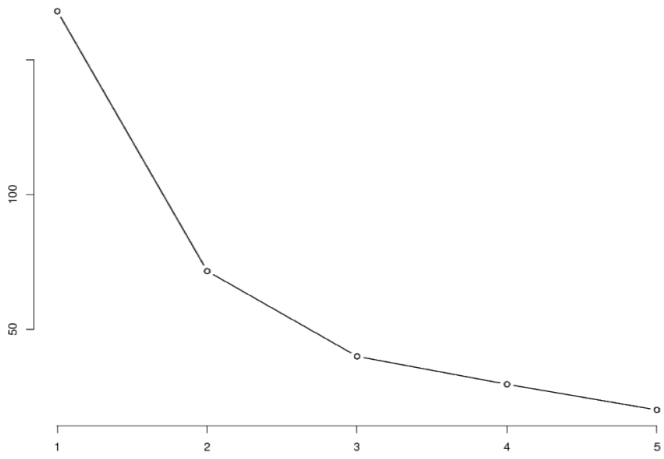
Por lo cual queda demostrado el segundo punto y po lo tanto vemos que EM es un algoritmo MM.

2. Problema 3

Usa los algoritmos de clustering (kmeans, hclust) para entender mejor la estructura de los datos "deport.dat"de la primera tarea. Escribe un pequeño reporte con los resultados más informativos.

Solución : Para este problema se trabajaron los datos del heptatlon de la primer tarea. Lo primero que se hizo fue aplicar el algoritmo de KMeans y para

ello se escalaron los datos usando la función *scale*. Ahora, previo a calcular el KMeans se aplicó el código visto en las ayudantías para ayudarnos a ver qué número de clusters usar. Para lo cual se obtuvo la siguiente gráfica.



(a) Figura 1. Gráfica con relación al número de clusters a usar en KMeans.

Podemos ver que entre el 2 y 3 se encuentra el çodo"de la gráfica, y por lo tanto podemos optar por una de esas dos opciones. Como elección arbitraria se usó el KMeans con 3 clusters. A continuación se mostrará el resultado del algoritmo de KMeans y para ayudar a interpretar un poco más la información se mostrará la tabla de los datos.

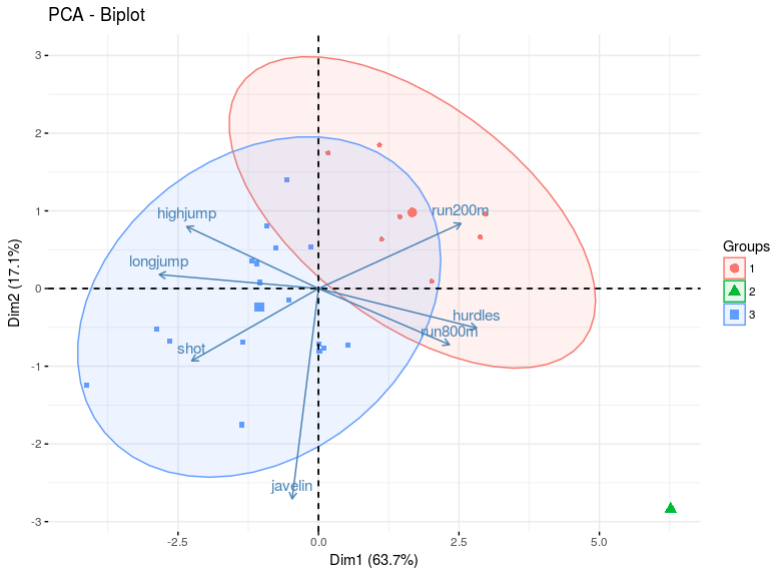
	hurdles	highjump	shot	run200m	longjump	javelin	run800m
Joyner-Kersee (USA)	-1.56112593	1.0007405	1.79799456	-2.154798275	2.35675020	1.1782304	-0.9098926366
John (GDR)	-1.34392580	0.2309401	2.08622065	-1.030573634	1.17584459	0.3039212	-1.1981539571
Behmer (GDR)	-0.86880052	0.6158403	0.72552539	-1.597842949	1.11258179	0.8623510	-1.4297279049
Sablovskaite (URS)	-0.31222519	0.2309401	1.41592742	-0.752095971	0.20581498	0.3659689	-0.4600119984
Choubenkova (URS)	-0.44797527	-0.5388603	1.10088960	-0.741781983	0.35342818	1.6858939	-0.9834656096
Schulz (GDR)	-0.12217507	0.6158403	0.25632013	0.000825119	0.37451578	0.3772503	-1.2379557294
Fleming (AUS)	-0.62445037	0.2309401	-0.15926167	-1.092457559	0.45886618	-0.3391192	-0.4238285691
Greiner (USA)	-0.39367523	0.2309401	0.67860487	-0.174512669	0.66974218	-0.9821595	-0.2899498805
Lajbnerova (CZE)	-0.28507517	0.6158403	0.77914885	0.217418857	-0.08941142	0.2023885	-0.0004824457
Bouraga (URS)	-0.80092548	-0.1539601	-0.33353791	-1.092457559	0.26907778	-0.6832022	-0.1584834205
Wijnsma (HOL)	-0.12217507	1.0007405	-0.07212355	0.392756645	0.39560338	-1.0216445	-0.5504705718
Dimitrova (BUL)	-0.81450048	0.2309401	-0.15926167	-1.092457559	0.45886618	-0.3391192	-0.4238285691
Scheider (SWI)	0.01357501	1.0007405	-1.03064287	0.227732845	-0.21593703	1.6971753	-0.1355672486
Braun (FRG)	-0.17647510	0.6158403	0.02842043	0.134906957	-0.06832382	0.8736324	0.8160569432
Ruotsalainen (FIN)	-0.06787504	0.2309401	-0.53462588	-0.040430831	-0.15267423	1.1161827	0.1213350997
Yuping (CHN)	0.12217507	1.0007405	0.73222833	0.361814683	0.52212898	-0.8129384	1.2804109531
Hagger (GB)	-0.50227530	0.2309401	-0.24639979	0.846572097	0.39560338	-1.6139185	0.2926033320
Brown (USA)	0.31222519	0.6158403	-0.28661739	0.186476895	-0.04723622	0.8059439	1.2514642097
Mulliner (GB)	0.74662544	-0.9237604	-0.29332032	0.279302782	-0.11049902	-1.0498480	0.2371220736
Hautenauve (BEL)	0.27150016	-0.1539601	-0.87647543	0.990967922	-0.34246263	-1.6364813	-0.2597970227
Kytola (FIN)	0.63802538	-0.1539601	-0.97701941	1.073479822	-0.84856503	-0.5647474	-0.3261333098
Geremias (BRA)	0.52942531	-0.9237604	-0.11234115	0.877514059	-1.37575504	-0.5196217	0.9607906606
Hui-Ing (TAI)	1.37107581	-1.3086606	-2.00970618	0.599036396	-1.43901784	-0.6606394	0.1502818432
Jeong-Mi (KOR)	0.93667556	-0.9237604	-1.53336279	2.022366675	-1.37575504	-0.6267951	0.3758252195
Launa (PNG)	3.50235208	-3.6180617	-0.89658423	1.558237236	-2.68318625	1.3812958	3.3018585393

(b) Figura 2. Tabla de los datos.

```
K-means clustering with 3 clusters of sizes 17, 1, 7

Cluster means:
     hurdles  highjump       shot    run200m   longjump    javelin    run800m
1 -0.4407885  0.457352  0.4132476 -0.4851457  0.4576257  0.3112209 -0.2783854
2  3.5023521 -3.618062 -0.8965842  1.5582372 -2.6831862  1.3812958  3.3018585
3  0.5701503 -0.593846 -0.8755179  0.9556057 -0.7280645 -0.9531502  0.2043847

Clustering vector:
Joyner-Kersee (USA)          John (GDR)        Behmer (GDR)  Sablovskaite (URS)   Choubenkova (URS)        Schulz (GDR)       Fleming (AUS)
                  1                   1                   1                   1                   1                   1                   1
      Greiner (USA)    Lajbnerova (CZE)       Bouraga (URS)       Wijnsma (HOL)     Dimitrova (BUL)      Scheider (SWI)         Braun (FRG)
                  1                   1                   1                   1                   1                   1                   1
 Ruotsalainen (FIN)        Yuping (CHN)         Hagger (GB)         Brown (USA)       Mulliner (GB)    Hautenauve (BEL)        Kytola (FIN)
                  1                   1                   3                   1                   3                   3                   3
     Geremias (BRA)       Hui-Ing (TAI)      Jeong-Mi (KOR)         Launa (PNG)
                  3                   3                   3                   2

Within cluster sum of squares by cluster:
[1] 57.59632  0.00000 14.65312
 (between_SS / total_SS =  57.0 %)
```

(c) Figura 3. Resultado del algoritmo KMeans usando 3 clusters.

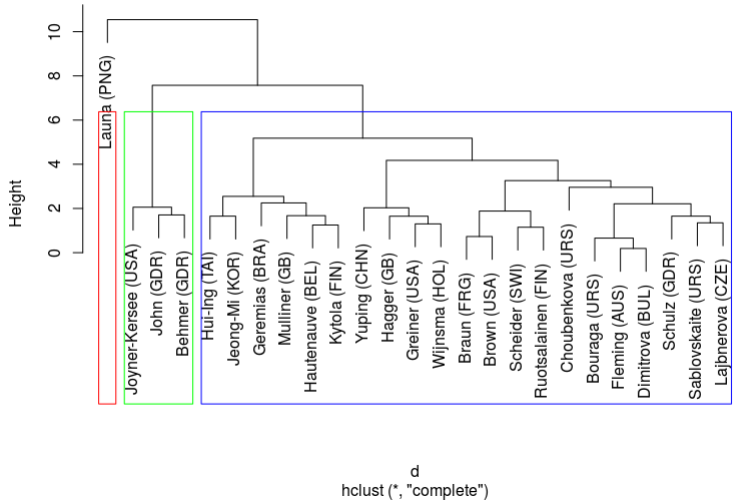
En la tabla de la Figura 2. se muestran a los atletas ordenados de mayor a menor en base a su desempeño, por lo que por ejemplo podemos ver que los mejores atletas fueron Joyner-Kersee, John, Behmer, etc.
Ahora para el resultado de KMeans mostrado en la Figura 3. vemos a los atletas clasificados en los 3 diferentes grupos. Antes de continuar recordemos que la medida de distancia usada por en el algoritmo es la que trae por defecto, osea la distancia euclideana.
Lo primero que podemos notar es que el algoritmo nos clasificó a los mejores atletas en el grupo 1, posteriormente a los atletas que tuvieron un desempeño más bajo en el tercero y finalmente pone a un/una atleta en el segundo grupo. Es fácil notar que este/a atleta tuvo el peor desempeño en muchas de las categorías del evento.
Ahora y para ilustrar un poco más gráficamente los resultados del KMeans se generó una gráfica en la cual se colocan los puntos como en un plano 2D y su centroide. Además se pensó que era buena idea hacer una comparación con lo que nos arroja el algoritmo de PCA aplicado al mismo conjunto de datos (Teniendo en cuenta la normalización de los mismos usando la matríz de correlación). Finalmente se combinó la gráfica anteriormente dicha con el biplot generado por el PCA y este fue el resultado.



(d) Figura 4. Biplot del algoritmo PCA combinado con la gráfica de los grupos generados por KMeans.

En la gráfica mostrada en la Figura 4. podemos notar lo que se mencionó anteriormente, primero tenemos el grupo donde el/la atleta Launa está incluido/a, el cual se aprecia en la esquina inferior derecha de la gráfica, claramente para PCA este punto no genera variabilidad alguna. Posteriormente tenemos los otros dos grupos correspondientes a los atletas con mejor desempeño y a atletas con desempeño medio.
Una cosa interesante que podemos notar es que en el grupo azul podemos ver a los atletas que están más del lado en eventos tipo salto/lanzamiento y en el grupo naranja vemos a los que están más del lado en eventos tipo carreras. Si bien hay que tener en cuenta que los resultados presentados en la tabla de datos son mostrados en tiempo (seg), estos dos conjuntos eventos mencionados se diferencia en los resultados de los eventos de salto son mejores si el tiempo es mayor, y por el contrario, los resultados de los eventos de carreras son mejores entre menos tiempo tenga el mismo.

Para conluir, se generó un dendograma sobre los datos escalados usando como métrica la distancia euclideana y 3 grupos. Esto fue lo que se obtuvo.



(e) Figura 5. Dendograma generado con el conjunto de datos.

En la Figura 5. se muestra el dendograma generado por los datos escalados. Lo que podemos concluir es que coincide ampliamente con los datos obtenidos por KMeans, tenemos los grupos representando a los mejores atletas, los de menor desempeño y finalmente a Launa, el/la que tuvo un peor desempeño.
A continuación se muestra el código utilizado para este ejercicio. Para ello se usó la librería *factoextra*.

```
setwd("Directorio")
d <- read.table("hepatlon")
library("factoextra")

data = d[1:7]
print(data)
data = scale(data)
p = prcomp(data, center = TRUE, scale = TRUE)
set.seed(7)
#biplot(p)
postscript("Biplot_kmeans_separated.eps")
fviz_pca_biplot(p, label="var", habillage=kmeans(as.matrix(data), 3, nstart=20)$
dev.off()

no_clusters <- 3
km <- kmeans(as.matrix(data), no_clusters, nstart = 20, iter.max = 100)
print(km)
```

```
#Numero de grupos
#wss <- rep(0, 6)
wss <- (nrow(data) - 1) * sum(var(as.vector(data)))
for (i in 1:5){
  wss[i] <- sum(kmeans(as.vector(data), centers = i)$withinss)
}

print(wss)
plot(1:5, wss, type = "b", xlab = "Numero de grupos",
     ylab = "Suma de cuadrados entre grupos")

d = dist(data, method = "euclidean") # definimos la distancia
fit = hclust(d)  # definimos el metodo para hacer clusters
# Generamos el dendograma
plot(fit, main = "")
rect.hclust(fit, 3, border = rainbow(3))
```

3. Problema 4

Para el primer ejemplo en las notas del algoritmo MM, verifica que $h_i(\theta|\theta^m)$ mayoriza $|y_i - \theta|$. Limítate al caso $y_i \geq \theta \geq \theta^m$.

Solución : Por definición, para que $h_i(\theta|\theta^m)$ mayorice a $|y_i - \theta|$ se tiene que cumplir:

1. $g(\theta^m|\theta^m) = f(\theta^m)$
2. $g(\theta|\theta^m) \geq f(\theta)$

Para algunas funciones $f(\theta)$, $g(\theta|\theta^m)$ previamente definidas. En nuestro problema tenemos que $g(\theta|\theta^m) = h_i(\theta|\theta^m)$ y $f(\theta) = |y_i - \theta|$.

Para el primer caso tenemos:

$$h_i(\theta^m|\theta^m) = 0{,}5\frac{(y_i - \theta^m)^2}{|y_i - \theta^m|} + 0{,}5(y_i - \theta^m) \tag{9}$$

Sabemos que por hipótesis $y_i \geq \theta \geq \theta^m$ se cumple entonces la cantidad $y_i - \theta^m$ es positiva. Por lo que nos queda:

$$\begin{aligned} 0{,}5\frac{(y_i - \theta^m)^2}{y_i - \theta^m} + 0{,}5(y_i - \theta^m) &= 0{,}5(y_i - \theta^m) + 0{,}5(y_i - \theta^m) \\ &= (y_i - \theta^m) \end{aligned} \tag{10}$$

De esta forma vemos que el primer punto se cumple.

Ahora, el segundo punto se demostrará por contradicción. Suponemos que $h_i(\theta|\theta^m) < |y_i - \theta|$ por lo cual tenemos:

$$\begin{aligned}
0{,}5\frac{(y_i-\theta)^2}{|y_i-\theta^m|} + 0{,}5(y_i-\theta^m) &< y_i - \theta \\
0{,}5(\frac{(y_i-\theta)^2+(y_i-\theta^m)^2}{y_i-\theta^m}) - (y_i-\theta) &< 0 \\
\frac{0{,}5(y_i-\theta)^2+0{,}5(y_i-\theta^m)^2-(y_i-\theta)(y_i-\theta^m)}{y_i-\theta^m} &< 0 \\
0{,}5(y_i^2-2y_i\theta+\theta^2)+0{,}5(y_i^2-2y_i\theta^m+\theta^{2m})-(y_i^2-y_i\theta^m-y_i\theta+\theta\theta^m) &< 0 \\
0{,}5y_i^2-y_i\theta+0{,}5\theta^2+0{,}5y_i^2-y_i\theta^m+0{,}5\theta^{2m}-y_i^2+y_i\theta^m+y_i\theta-\theta\theta^m &< 0 \\
0{,}5\theta^2+0{,}5\theta^{2m}-\theta^{m+1} &< 0 \\
(\sqrt{0{,}5}\theta-\sqrt{0{,}5}\theta^m)^2 &< 0
\end{aligned} \tag{11}$$

Hemos llegado a que $(\sqrt{0{,}5}\theta-\sqrt{0{,}5}\theta^m)^2$ es menor que cero, pero sabemos que a lo más, ese cuadrado puede ser positivo o cero, por lo que esto anterior es una contradicción. De lo cual se deduce que $h_i(\theta|\theta^m) \geq |y_i - \theta|$ y queda demostrado que h_i mayoriza a $|y_i - \theta|$.

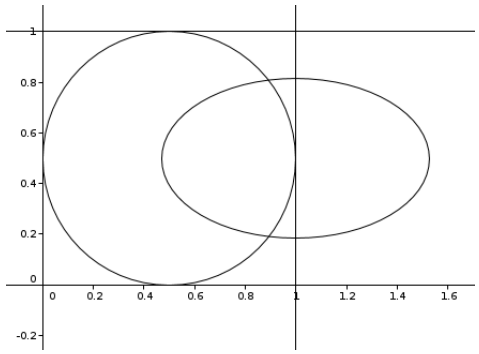
4. Problema 5

Supongamos que elegimos un punto al azar del area formada por una letra C. Usa el algoritmo EM para aproximar esta distribución con una mezcla de K distribuciones Gausianas usando una muestra de tamaño 200 (no está permitido usar la función me o em de R!). Toma K = 5, 10, 15.
Visualiza los resultados (por ejemplo con una gráfica de contornos de la densidad encontrada).

Solución : Lo primero que se hizo para este problema fue generar la distribución de puntos que cumpliera con la forma de una letra C, para lo cual se siguió la idea de primeramente generar puntos (x_i, y_i) con una distribución uniforme de (0, 1). Lo anterior generará puntos en el un cuadro de área 1. Lo siguiente fue hacer un filtrado con los mismos para que quedaran dentro de una forma de luna, y para ello tenemos que ver si los puntos generados están dentro de una circunferencia con centro en (0,5 0,5) y con radio $r = 0{,}5$, para ello verificamos si la distancia euclideana del punto generado con respecto al centro del círulo es menor que su radio, de esta manera el punto está dentro de la circunferencia.
por otra parte debemos verificar que los puntos generados estén fuera de una elipse definida con centro en (1, 0,5), semieje mayor $a = 0{,}28$ y semiejemenor $b = 0{,}1$. Para ello se tiene que cumplir la siguiente desigualdad:

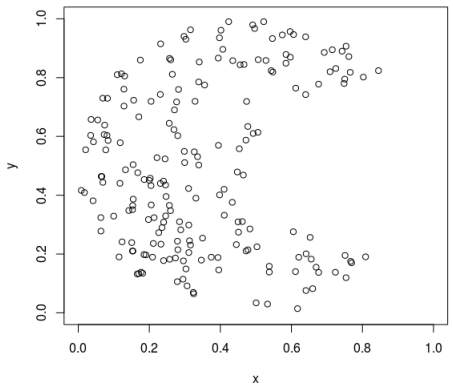
$$\frac{(x-1)^2}{0{,}28} + \frac{(y-0{,}5)^2}{0{,}1} > 1 \tag{12}$$

De cumplir lo anterior, el punto estará fuera de la elipse. Un ejemplo gráfico de lo mencionado anteriormente se muestra a continuación.



(f) Figura 6. Área geométrica donde se encontrarán los puntos de la distribución.

El resultado de la generación de la distribución de puntos con forma de C usando una muestra de tamaño 200 queda de la siguiente manera.



(g) Figura 7. Distribución de puntos con forma de C.

Para la siguiente parte se implementó el algoritmo EM para generar distintas distribuciones normales que se ajustaran a la serie de puntos descritos anteriormente.
La implementación del algoritmo se hace generalmenet en dos partes, la primera es la expectation, y para la cual se calculan los valores de una matriz $W \in R^{NxK}$ donde N es el número de datos y K es el número de clusters. El cálculo de esa matriz se hace de la siguiente manera:

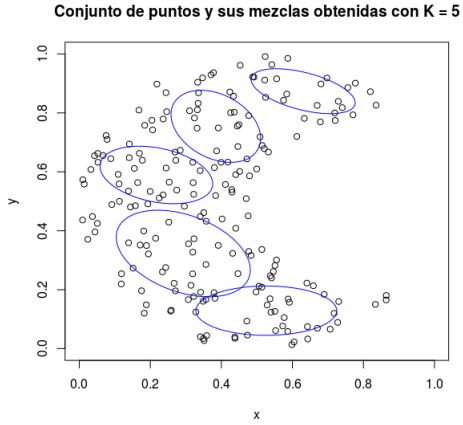
$$W_{i,j} = \frac{P_k(x_i|z_k, \theta_k)\alpha_k}{\sum_{m=1}^{K} P_m(x_i|z_m, \theta_m)\alpha_m}, 1 \leq k \leq K, 1 \leq i \leq N \tag{13}$$

Donde x_i corresponde a cada par $(\hat{x}_i, \hat{y}_i)$ de datos que tenemos y P es la densidad de una distribución normal bivariada.

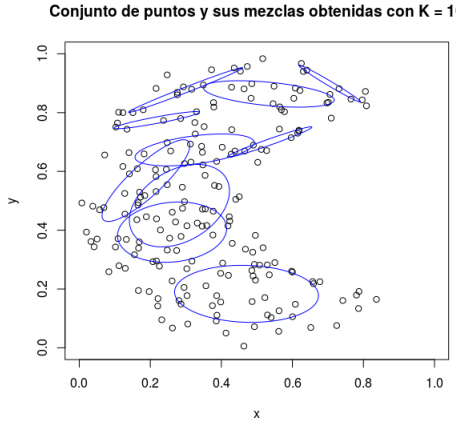
Ahora para la parte de maximización tenemos que estimar los nuevos valores de los parámetros, para ello definimos $N_k = \sum_{i=1}^{N} W_{i,k}$ la suma de las K columnas de la matriz anterior. Ahora la actualización de los paramétros se calcula de la siguiente manera.

$$\begin{aligned} \alpha_{k+1} &= \frac{N_k}{N}, 1 \leq k \leq K \\ \mu_{k+1} &= \frac{1}{N_k}\sum_{i=1}^{N} W_{i,k}x_i, 1 \leq k \leq K \\ \Sigma_{k+1} &= \frac{1}{N_k}\sum_{i=1}^{N} W_{i,k}(x_i - \mu_{k+1})(x_i - \mu_{k+1})^T, 1 \leq k \leq K \end{aligned} \tag{14}$$

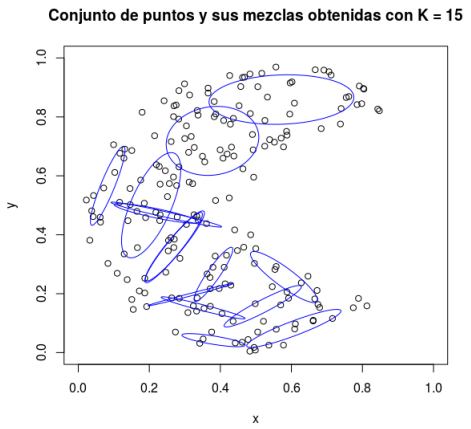
Y de esta manera realizamos el paso de maximización. Como criterio de paro se indicó un número de iteraciones máximo y de la misma forma se demandó que cuando el cambio en los valores de la log-verosimilitud ya no fueran tan grandes (menor a una tolerancia) se detuviera el algoritmo. Los valores iniciales de μ_0, α_0 y Σ_0 fueron dados aleatoriamente con una distribución uniforme entre 0 y 1. Finalmente el algoritmo regresará los parámetros estimados, los cuales corresponden a los parámetros de las Gaussianas bivariadas que tienen media en los K clusters, y las alphas. Posteriormente se usó la librería *mixtools* para dibujar los contornos de las Gaussianas encontradas en el conjunto de puntos.
A continuación se muestran los resultados de ejecutar el algoritmo EM con $K = 5, 10, 15$ clusters, 100 iteraciones como máximo y una tolerancia de $1e^{-3}$.



(h) Figura 8. Resultado de EM usando 5 clusters.

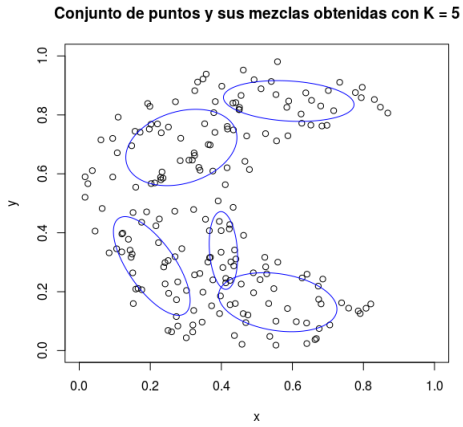


(i) Figura 9. Resultado de EM usando 10 clusters.

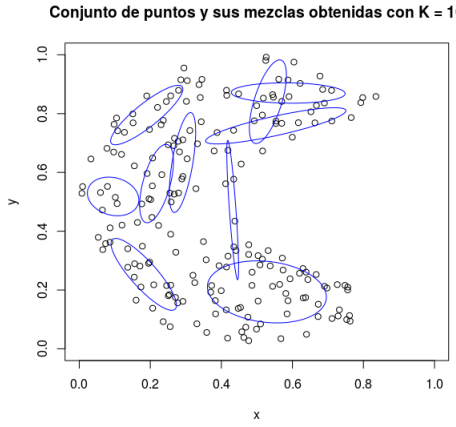


(j) Figura 10. Resultado de EM usando 15 clusters.

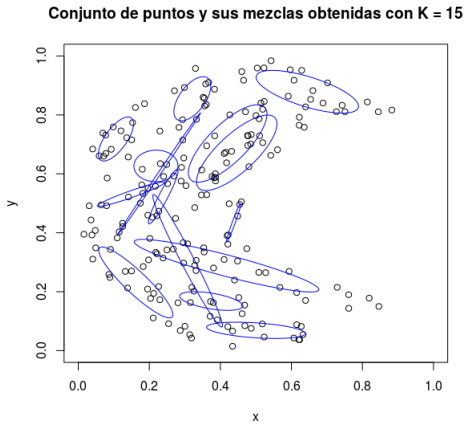
Podemos ver en las tres Figuras anteriores el resultado de aplicar EM con $K = 5, 10, 15$ clusters. Como se comentó previamente, para la visualización de las gaussianas resultantes, se usó la librería mixtools para el dibujo de los contornos, se puede apreciar claramente que las Gaussianas abarcan en su mayoría a la distribución de puntos generada al comienzo del programa.
Una modificación interesante al algoritmo podría ser inicializar los centros μ de las Gaussianas con la salida que nos da el algoritmo $KMeans$. A continuación se muestran los resultados del algoritmo EM usando la misma cantidad de clusters pero implementando el enfoque mencionado sobre $KMeans$.



(k) Figura 11. Resultado de EM usando 5 clusters (Inicialización con KMeans).



(l) Figura 12. Resultado de EM usando 10 clusters (Inicialización con KMeans).



(m) Figura 13. Resultado de EM usando 15 clusters (Inicialización con KMeans).

En las tres Figuras anteriores se aprecian los resultados del algoritmo EM pero con la inicialización de los centros usando la salida de $KMeans$. Es claro que las Gaussianas se ajustan al modelo de los puntos como vimos en el caso anterior pero no se nota algún patrón interesante de las mismas, y al ejecutar el algoritmo no se tuvo convergencia más rápida.
A continuación se adjunta el código desarrollado en el lenguaje R sobre este ejercicio, de igual manera se mandará en conjunto con el reporte.

```
library("mvtnorm")
library("mixtools")

#Distancia euclideana
euc <- function(x1, y1, x2, y2) {
  return(sqrt((x2 - x1)^2 + (y2 - y1)^2))
}

#log-verosimilitud
loglikelihood <- function(x, y, alpha_s, mu_x, mu_y, sigma, N, K) {
  res <- 0
  for(i in 1:N) {
    ll <- 0
    for(k in 1:K) {
      ll <- ll + (dmvnorm(c(x[i], y[i]), c(mu_x[k], mu_y[k]), sigma[[k]]) * alph
    }
    res <- res + log(ll)
  }
  return(res)
}
```

```
#Plotear los contornos de las Gaussianas generadas por EM
plot_EM_res <- function(x, y, mu_x, mu_y, sigma, K) {
  plot(x, y, xlim = c(0, 1), ylim = c(0, 1), title("Conjunto de puntos y sus me
  for(k in 1:K) {
    mmu <- c(mu_x[k], mu_y[k])
    ellipse(mu = mmu, sigma = sigma[[k]], alpha = 0.5, col = "blue")
  }
}

#Funcion principal. Algoritmo EM
EM <- function(x, y, N, K, it, use_km) {
  alpha_s <- rep(1 / K, K)
  mu_x <- rep(runif(1, 0, 1), K)
  mu_y <- rep(runif(1, 0, 1), K)
  sigma <- list()
  #Anteriores
  lalpha_s <- vector()
  lmu_x <- vector()
  lmu_y <- vector()
  lsigma <- list()

  #Inicializar los centros usando K-Means
  if(use_km) {
    m_aux <- array(0, dim = c(length(x), 2))
    m_aux[, 1] <- x
    m_aux[, 2] <- y
    k_clust <- kmeans(m_aux, K, nstart = 20)
    for(k in 1:K) {
      mu_x[k] <- k_clust$centers[k, 1]
      mu_y[k] <- k_clust$centers[k, 2]
    }
  }

  for(i in 1:K) {
    sigma[[i]] <- array(0, dim = c(2, 2))
    sigma[[i]][1, 1] <- sigma[[i]][2, 2] <- runif(1, 0, 1)
  }
  W <- matrix(0, nrow = N, ncol = K)
  tol <- 0.0001
  loglike <- 0

  flag <- TRUE
  #Paro por iteraciones
  for(t in 1:it) {
```

```
  if(!flag) {
    break
  }
  #Expectation
  for(i in 1:N) {
    for(j in 1:K) {
      W[i, j] <- (dmvnorm(c(x[i], y[i]), c(mu_x[j], mu_y[j]), sigma[[j]]) * al
    }
  }
  W <- sweep(W, 1, rowSums(W), '/')

  #Maximization
  for(k in 1:K) {
    Nk <- 0
    Nxk <- c(0, 0)
    S_aux <- array(0, dim = c(2, 2))
    for(i in 1:N) {
      Nk <- Nk + W[i, k]
      Nxk <- Nxk + (W[i, k] * c(x[i], y[i]))
    }
    alpha_s[k] <- Nk / N
    mu_x[k] <- Nxk[1] / Nk
    mu_y[k] <- Nxk[2] / Nk
    for(i in 1:N) {
      v_aux <- c(x[i] - mu_x[k], y[i] - mu_y[k])
      S_aux <- S_aux + (W[i, k] * (v_aux %*% t(v_aux)))
    }
    sigma[[k]] <- S_aux / Nk
  }
  if(t > 1) {
    #Paro porque la log-verosimilud del theta actual es muy parecida a la de l
    loglike <- abs(loglikelihood(x, y, alpha_s, mu_x, mu_y, sigma, N, K) - log
    if(loglike < tol) {
      flag <- FALSE
    }
  }
  lalpha_s = alpha_s
  lmu_x = mu_x
  lmu_y = mu_y
  lsigma = sigma
}

#Generar la grafica
cat("Diferencia de la Log-verosimilitud final:", loglike, sep = " ")
plot_EM_res(x, y, mu_x, mu_y, sigma, K)
```

```
}

cnt <- 0
n <- 200
x <- vector()
y <- vector()

cx1 <- 1
cy1 <- 0.5
r1 <- 0.5

cx2 <- 0.5
cy2 <- 0.5
r2 <- 0.5

while(cnt < n) {
  xi <- runif(1, 0, 1)
  yi <- runif(1, 0, 1)
  if(((xi - cx1)^2 / 0.28) + ((yi - cy1)^2 / 0.1) > 1 && euc(xi, yi, cx2, cy2) <
    x <- c(x, xi)
    y <- c(y, yi)
    cnt <- cnt + 1
  }
}

EM(x, y, n, 15, 100, TRUE)
```