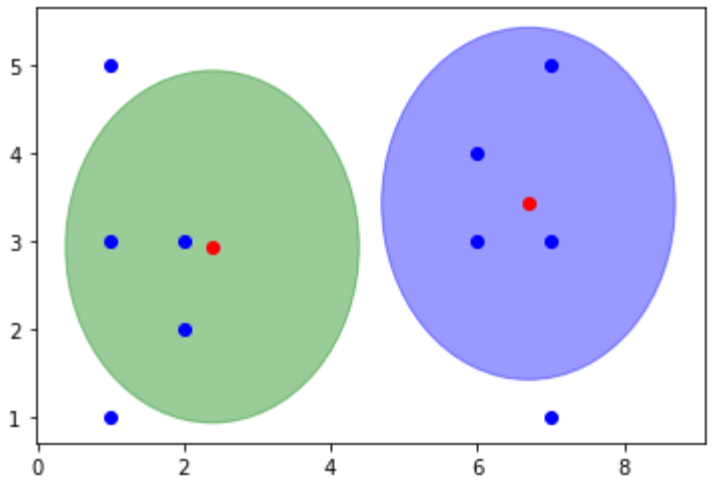
Alejandro Bazaco
Álvaro Ortiz

Memoria

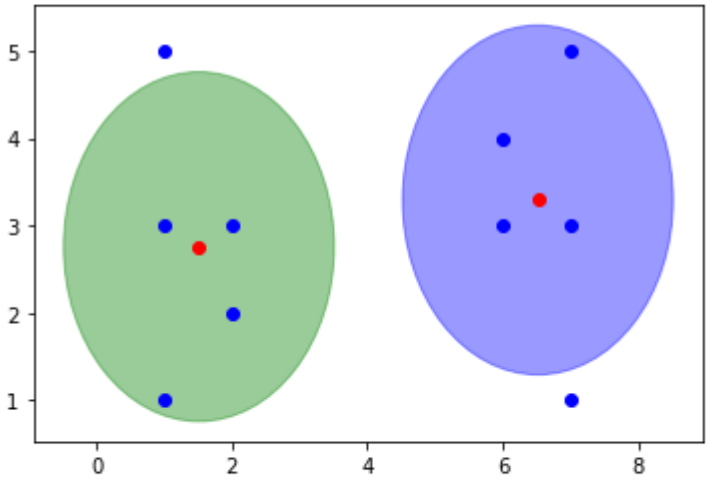
1. Diseño conceptual

```
########## ENTRENAMIENTO DE BAYES ##########

Las covariantes obtenidas son:
[[[ 0.66666667  0.        ]
  [ 0.          0.        ]]

 [[ 0.66666667 -0.66666667]
  [-0.66666667  0.66666667]]

 [[ 0.66666667  0.33333333]
  [ 0.33333333  0.66666667]]]

########## TEST DE BAYES ##########
El archivo prueba_clase1.txt pertenece a la clase
C1

El archivo prueba_clase2.txt pertenece a la clase
C2

El archivo prueba_clase3.txt pertenece a la clase
C3


########## ENTRENAMIENTO DE K-MEDIAS ##########


En la iteración 0 los centroides son:
[[6.7  3.43]
 [2.39 2.94]]
```



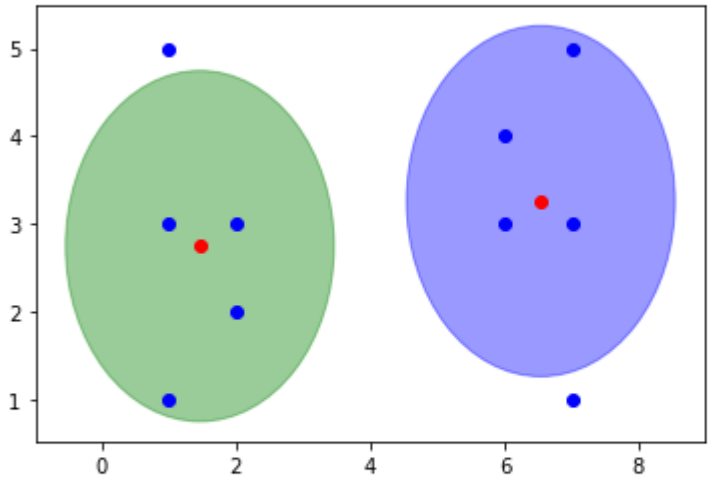
Alejandro Bazaco
Álvaro Ortiz

```
En la iteración 1 los centroides son:
[[6.52000497 3.29226962]
 [1.50997223 2.75818155]]
```




```
Hemos terminado en la iteración 2, con los centroides:
[[6.54008929 3.25987847]
 [1.46108287 2.7468911 ]]
```

Hemos terminado en la iteración 2, con los centroides:
[[6.54008929 3.25987847]
 [1.46108287 2.7468911]]




```
########## TEST DE K-MEDIAS ##########
El archivo prueba_clase1.txt pertenece al centroide situado en
[6.54008929 3.25987847] -> C1

El archivo prueba_clase2.txt pertenece al centroide situado en
[1.46108287 2.7468911 ] -> C2
```

Alejandro Bazaco
Álvaro Ortiz

```
########## ENTRENAMIENTO DE LLOYD ##########

ITERACIÓN 1.
Antiguo:
[[1. 4.]
 [7. 2.]]

Nuevo
[[1.19     3.478143]
 [6.8461   2.4548  ]]

------------------------------
ITERACIÓN 2.
Antiguo:
[[1.19     3.478143]
 [6.8461   2.4548  ]]

Nuevo
[[1.29097379 3.20080679]
 [6.74512621 2.75319428]]

------------------------------
ITERACIÓN 3.
Antiguo:
[[1.29097379 3.20080679]
 [6.74512621 2.75319428]]

Nuevo
[[1.3446354  3.05341896]
 [6.67887731 2.94897077]]

------------------------------
ITERACIÓN 4.
Antiguo:
[[1.3446354  3.05341896]
 [6.67887731 2.94897077]]

Nuevo
[[1.37315338 2.97509103]
 [6.6354114  3.07741972]]

------------------------------
########## TEST DE LLOYD ##########
El archivo prueba_clase1.txt pertenece a la clase
C1

El archivo prueba_clase2.txt pertenece a la clase
C2
```

La salida por pantalla muestra los resultados de llamar a los tres archivos:
Bayes.py Kmedias.py Lloyd.py

2. Instalación: Se debe tener Python instalado en el ordenador, y además, las siguientes librerías:
   - Pandas: pip install pandas
   - Numpy: pip install numpy
   - Matplotlib: pip install matplotlib

3. Diseño del código

3.1. Funciones de Bayes.py:

3.1.1. **obtenerMedias(X, Y, vName)**: Calcula la media de los datos pertenecientes a una clase a partir de un conjunto de datos(X), sus respectivas clases(Y), y un array con el nombre de todas las clases.

3.1.2. **covariante(X, media)**: Calcula el covariante de un dato X[i] y su media.

3.1.3. **obtenerCovariantes(X, Y, vName, medias)**: Calcula todas las covariantes a partir de un conjunto de datos(X), sus etiquetas(Y), un array con todas las etiquetas (vName) y las medias de dichas etiquetas/clases.

3.1.4. **pertenenciaBayes(X, medias)**: Devuelve la posición (en el array) de la distancia más cercana respecto a una media (centroide) a partir de un conjunto de datos (X) y la media de las clases (medias).

3.1.5. **bayes(X, Y, vName)**: Gestiona el algoritmo de Bayes y devuelve las medias (centroides) a partir de un conjunto de datos X, su valor Y, y el nombre de todas las etiquetas (vName).

3.1.6. **testBayes(medias, vName, directorio='test')**: Función que calcula la distancia de todas las pruebas ubicadas en la carpeta 'directorio' a partir del vector de centroides, su etiqueta, y el nombre de la carpeta. Muestra por pantalla el resultado.

3.2. Funciones de Kmedias.py:

3.2.1. ***formula(d, b)***: Esta función devuelve el cálculo de la fórmula:

$$\frac{1/d}{1/b-1}$$

3.2.2. **dist(x, v)**: Calcula la distancia de un punto X y respecto a su centroide V.

3.2.3. **calcularDivisor(Xj, V, b = 2):** Calcula el divisor sumando todos valores obtenidos de la función *formula* (3.1.1)

3.2.4. **calcularP(X, V, b = 2):** Calcula la variable P a partir de un conjunto de datos X, los centroides V, y un valor B.

3.2.5. **recalcularCentros(X, U):** Calcula otra vez los centros a partir del array X y el array U.

3.2.6. **cumpleEpsilon(vAntiguo, vNuevo, epsilon):** Función que calcula la distancia de un centroide y comprueba si es mayor o menor que epsilon (máximo error permitido).

3.2.7. **seguirActualizando(vAntiguo, vNuevo, epsilon):** Función que comprueba si hay algún valor superior a epsilon en alguna de las componentes.

3.2.8. **dibujarGrafica(X, V):** Dibuja una gráfica con los centroides y los valores de X sólo cuando los centroides V tienen dos dimensiones.

3.2.9. **kMedias(X, V, b, epsilon:** Gestiona el algoritmo de KMedias, devuelve los centroides entrenados a partir de un conjunto de datos X, los centroides iniciales, un valor b y un error mínimo epsilon.

3.2.10. **testKMedias(V, vName, directorio='test'):** Función que calcula la distancia de todas las pruebas ubicadas en la carpeta 'directorio' a partir del vector de centroides, su etiqueta, y el nombre de la carpeta. Muestra por pantalla el resultado.

3.3. **Funciones de Lloyd.py:**

3.3.1. **calcularDistancias(x, V):** Función que calcula la distancia de un punto X[i] respecto a los centroides.

3.3.2. **distEuclidea(A, B):** Calcula la distancia euclidea de un punto A respecto a B.

3.3.3. **lloyd(X, V, gamma, kMax, epsilon):** Gestiona el algoritmo de Lloyd, devuelve los centroides entrenados a partir de un conjunto de datos X, los centroides iniciales, un valor gamma, unas iteraciones máximas y un error mínimo epsilon.

3.3.4. **testLloyd(V, vName, directorio='test'):** Función que calcula la distancia de todas las pruebas ubicadas en la carpeta 'directorio' a partir del vector de centroides, su etiqueta, y el nombre de la carpeta. Muestra por pantalla el resultado.

4. Líneas de código adicionales:

4.1. main.py:

Leemos el archivo

df = pd.read_csv('entrenamiento.txt', header=None)

Dividimos en X e Y

X = np.array(df.iloc[:, :-1])
Y = np.array(df.iloc[:, -1])

Inicializamos los centroides

V = np.array([[4.5, 3.0, 4.0, 0.0], [6.8, 3.4, 4.6, 0.7]])
vName = np.unique(Y)

Valores para K-Medias

b = 2
epsilonKmedias = 0.02

Valores para Lloyd

*epsilonLloyd = 10**-10*
kMax = 10
gamma = .1

4.2. Líneas para llamar a Bayes.py:

medias = bayes(X, Y, vName)
testBayes(medias, vName)

4.3. Líneas para llamar a KMedias.py:

V_KMedias = trainKMedias(X, V, b, epsilonKmedias)
testKMedias(V_KMedias, vName)

4.4. Líneas para llamar a Lloyd.py:
vLloyd = lloyd(X, V, gamma, kMax, epsilonLloyd)
testLloyd(vLloyd, vName)