Trabajo práctico 0: Algoritmo de Maximización de la Esperanza

Ph. D. Saúl Calderón Ramírez
Instituto Tecnológico de Costa Rica,
Escuela de Computación
PAttern Recongition and MAchine Learning Group (PARMA-Group)

22 de septiembre de 2022

Fecha de entrega: Domingo 2 de Octubre del 2022.
Entrega: Un archivo .zip con el código fuente LaTeX o Lyx, el pdf, y un jupyter en Pytorch, debidamente documentado, con una función definida por ejercicio. A través del TEC-digital.
Modo de trabajo: Grupos de 3 personas.

Resumen

En el presente trabajo práctico se repasarán aspectos básicos del algebra lineal, relacionados con los conceptos a desarrollar a lo largo del curso, mezclando aspectos teóricos y prácticos, usando el lenguaje Python.

1. (100 puntos) Probabilidades: Algoritmo de Maximización de la Esperanza

A continuación, implemente el algoritmo de maximización de la esperanza (descrito en el material del curso), usando la definición y descripción de las siguientes funciones como base:

1. **(10 puntos)** Implemente la función *generate_data* la cual reciba la cantidad de observaciones unidimensionales total a generar N, y los parámetros correspondientes a $K = 2$ funciones de densidad Gaussianas. Genere los datos siguiendo tales distribuciones, y retorne tal matriz de datos $X \in \mathbb{R}^N$.

    a) Grafique los datos usando un *scatter plot* junto con las gráficas de la función de densidad de probabilidad, en la misma figura (gráfico), usando $\mu_1 = 10$, $\sigma_1 = 1{,}5$, $\mu_2 = 20$, $\sigma_2 = 3$.

2. **(10 puntos)** Implemente la función *init_random_parameters* la cual genere una matriz de $W \in \mathbb{R}^{K\times 2}$ dimensiones, con los parámetros de las funciones de densidad Gaussiana generados completamente al azar.

    a) Muestre un pantallazo donde verifique su funcionamiento correcto con los comentarios asociados.

    b) Muestre una grafica de las funciones de densidad con los parametros inicializados aleatoriamente.

3. **(10 puntos)**Implemente la función *calculate_likelihood_gaussian_observation(x_n, mu_k, sigma_k)* la cual calcule la verosimilitud de una observación específica x_n, para una función de densidad Gaussiana con parámetros μ_k y σ_k.

    a) Muestre un pantallazo donde verifique su funcionamiento correcto con los comentarios asociados, usando los datos anteriormente generados con $\mu_1 = 10$, $\sigma_1 = 1{,}5$, $\mu_2 = 20$, $\sigma_2 = 3$.

4. **(10 puntos)** Implemente la función *calculate_membership_dataset(X_dataset, Parameters_matrix)*, la cual, usando la matriz de parámetros W y la función anteriormente implementada *calculate_likelihood_gaussian_observation*, defina por cada observación $x_n \in X$ la pertenencia o membresía a cada cluster $k = 1, \dots K$, en una matriz binaria $M \in \mathbb{R}^{N\times K}$. Retorne tal matriz de membresía M.

    a) Muestre un pantallazo donde verifique su funcionamiento correcto con los comentarios asociados, usando los datos de prueba anteriormente generados.

5. **(20 puntos)** Implemente la función *recalculate_parameters(X_dataset, Membership_data)*, la cual recalcule los parámetros de las funciones de densidad Gaussianas representandas en la matriz W, de acuerdo a lo representado en la matriz de membresía M. Debe retornar la matriz con los parárametros W.

    a) Use las funciones *mean* y *std* de pytorch para ello. Intente prescindir al máximo de estructuras de repetición tipo *for*.

    b) Muestre el resultado para un conjunto de datos de prueba y comente los resultados.

6. **(20 puntos)** Ejecute 5 corridas diferentes del algoritmo, donde por cada una documente los parámetros a los que se arribó. Ejecute cada corrida con $R = 20$ iteraciones.

    a) Grafique las funciones de densidad de probabilidad a las que convergió el algoritmo. Puede graficar también las funciones de densidad obtenidas en 2 o 3 pasos intermedios.

b) Comente los resultados, usando algún criterio de correctitud de los mismos.

7. **(20 puntos)** Proponga una mejor heurística para inicializar los parámetros del modelo aleatoriamente.

   a) Compruebe la mejora obtenida con el método propuesto, corriendo las pruebas del punto anterior.