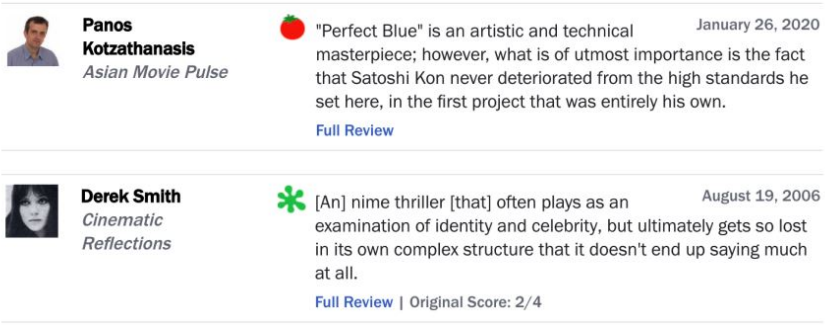
Inteligencia artificial
Lic. en Ciencias de la Computación
Tarea 1

Expediente: 220218775
Nombre: Jazmin Alejandra Escobedo Javalera
Colaboradores: Malcom Hiram Navarro López, Gaspar Armando Dávila Reyes

Desarrollando la intuición

Considera las siguientes dos reseñas de la película Perfect Blue, del sitio Rotten Tomatoes. Rotten



Tomatoes ha clasificado estas reseñas como “positiva” y “negativa” respectivamente, como se indica por el tomate fresco e intacto en la reseña de arriba y la salpicadura de un tomate podrido en la reseña de abajo. En esta tarea, vas a crear un simple sistema de clasificación de texto que puede realizar esta tarea automáticamente. Vamos a considerar el siguiente conjunto de cuatro mini-reseñas, cada una etiquetada como positiva $(+1)$ o negativa (-1):

1. (-1) not good
2. (-1) pretty bad
3. $(+1)$ good plot
4. $(+1)$ pretty scenery

Cada reseña x es asignada a un vector de características $\phi(x)$, el cual asocia cada palabra a la cantidad de veces que aparece en la reseña. Por ejemplo, la segunda reseña se asigna al vector disperso de características $\phi(x) = \{\text{pretty} : 1, \text{bad} : 1\}$. Recuerda la definición de la pérdida de articulación,

$$\text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = \max\{0, 1 - \mathbf{w} \cdot \phi(x)y\},$$

donde x es el texto de la reseña, y es la clasificación correcta y $\mathbf{w}$ es el vector de pesos.

1. Supongamos que corres el descenso de gradiente estocástico una vez por cada una de las cuatro reseñas de arriba en el mismo orden, actualizando los pesos de acuerdo a,

   $$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}).$$

   Después de las actualizaciones, ¿cuáles son los pesos de las seis palabras (”pretty”, ”good”, ”bad”, ”plot”, ”not”, ”scenery”) que aparecen en las reseñas?

- Utiliza $\eta = 0.1$ como tamaño de paso.
- Inicializa $\mathbf{w} = [0, 0, 0, 0, 0, 0]$.
- El gradiente $\nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = 0$ cuando el margen es exactamente 1.

Solución:

- El problema nos indica unas mini-reseñas cada una etiquetada como positiva (+1) o negativa (−1):
  1. (−1) not good
  2. (−1) pretty bad
  3. (+1) good plot
  4. (+1) pretty scenery

Las reseñas son x , y es la clasificación y w es el vector de peso

Vamos a inicializar las peso w en 0 , [0, 0, 0, 0, 0, 0] , después vamos a calcular el gradiente con la formula w←w-η□w Loss hinge (x, y,w). Vamos a ir actualizando los pesos y calculando el gradiente

Y $\eta = 0.1$ como el tamaño de paso.

Ahora actualizaremos los pesos por iteraciones:

1. $x =$ "not good", $y = -1$

   ϕ(x) = {"not": 1, "good": 1}

   $$\nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = \mathbf{max}\{0, 1-\}$$

   $$\mathbf{w} \leftarrow \mathbf{0} - 0.1 \cdot (-1, 1) = [0.1, -0.1]$$

2. $x =$ "pretty bad", $y = -1$

   $$\nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = -1 \cdot \{\text{"not"} : 0, \text{"good"} : 0, \text{"bad"} : 1, \text{"plot"} : 0, \text{"pretty"} : 1,$$

   $$\text{"scenery"} : 0 \}$$

   $$\mathbf{w} \leftarrow \mathbf{w} - 0.1 \cdot (0, 0, 1, 0, 1, 0) = [0.1, -0.1, -0.1, 0, 0.1, 0]$$

3. $x =$ "good plot", $y = +1$
   No hay actualización de pesos ya que $\nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = 0$.
4. $x =$ "pretty scenery", $y = +1$
   No hay actualización de pesos ya que $\nabla_{\mathbf{w}} \text{Loss}_{\text{hinge}}(x, y, \mathbf{w}) = 0$.

Por lo tanto, los pesos finales son: $\mathbf{w} = [0.1, -0.1, -0.1, 0, 0.1, 0]$

- not: 0.1
- good: -0.1
- bad: -0.1
- plot: 0
- pretty: 0.1
- scenery: 0

2. (Opcional) Dado el segundo conjunto de datos de reseñas:
   (a) (-1) bad
   (b) $(+1)$ good
   (c) $(+1)$ not bad
   (d) (-1) not good

   Muestra que ningún clasificador lineal que utilice conteo de palabras como sus características puede obtener un error cero sobre este conjunto de datos. Recuerda que esta es una pregunta sobre clasificadores, no sobre algoritmos de optimización; tu prueba debe ser cierta para cualquier clasificador lineal de la forma $f_{\mathbf{w}}(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \phi(\mathbf{x}))$, independientemente de cómo se aprenden los pesos.

   Propón una característica adicional para el conjunto de datos con la que podemos ampliar el vector de características que resolvería este problema.

Solución:
En los ejemplos que nos da el problema, las reseñas donde tenemos problemas serian la (c) y (d), muestran una contradicción en la etiqueta asignada. Esto se debe a que las palabras ”not bad” y ”not good”, con la palabra not hace que cambie el significado de las reseñas, las palabras ”bad” y ”good” tienen el mismo peso en ambas reseñas, pero su combinación con ”not” cambia el significado de la frase.

Una solución que se me ocurre es agregar un característica nueva o adicional que estén relacionas a las reseñas, que se puedan distinguir que es una mala o buena reseña, para que tenga una mejor capacidad de distinguir las palabras. Se puede introducir una característica adicional que capture la negación de la palabra.

Prediciendo calificaciones de películas

Supongamos que ahora nos interesa predecir una calificación numérica para reseñas de películas. Utilizaremos un predictor no-lineal que toma una reseña x y regresa $\sigma(\mathbf{w} \cdot \phi(x))$, donde,

$$\sigma(z) = (1 + e^{-z})^{-1}$$

es la función logística que aplasta un número real al rango $(0, 1)$. Para este problema, supón que la calificación de película y es una variable con valor real en el rango $[0, 1]$.

1. Supongamos que queremos usar la pérdida cuadrática. Escribe la expresión para $\text{Loss}(x, y, \mathbf{w})$ para un dato (x, y).

Solución:
Para este problema la expresión para la pérdida Loss cuadratica(x, y, w) es :

$$\text{Loss}_{\text{cuadratica}}(x, y, \mathbf{w}) = (y - \sigma(\mathbf{w} \cdot \phi(x)))^2$$

Donde:

- x es la reseña de la pelicula .
- y es la calificación de la película, con un el rango $[0, 1]$.
- $\mathbf{w}$ es el vector de pesos del modelo.
- $\phi(x)$ es el vector de características asociado a la reseña x.
- $\sigma(z) = (1 + e^{-z})^{-1}$ es la función logística, que aplasta un número real al rango $(0, 1)$.

La expresión para $\text{Loss}_{\text{cuadratica}}(x, y, \mathbf{w})$ para un dato (x, y) es:

$$\text{Loss}_{\text{cuadratica}}(x, y, \mathbf{w}) = (y - \sigma(\mathbf{w} \cdot \phi(x)))^2$$

Esta función de pérdida mide el cuadrado de la diferencia entre la calificación real y y la calificación predicha $\sigma(\mathbf{w} \cdot \phi(x))$ para una sola observación. El objetivo es minimizar esta función de pérdida en todo el conjunto de datos de entrenamiento, encontrar el vector de pesos w que minimiza la suma de las pérdidas de todas las observaciones en el conjunto de entrenamiento

2. Dada la pérdida $Loss(x, y, \mathbf{w})$ del problema anterior, calcula el gradiente de la pérdida con respecto a $\mathbf{w}$, $\nabla_{\mathbf{w}} Loss(x, y, \mathbf{w})$. Escribe la respuesta en términos del valor predecido $p = \sigma(\mathbf{w} \cdot \phi(x))$.

Solución: Para calcular el gradiente de la pérdida con respecto a $\mathbf{w}$, $\nabla_{\mathbf{w}}\text{Loss}(x, y, \mathbf{w})$, primero necesitamos encontrar la derivada parcial de $\text{Loss}(x, y, \mathbf{w})$ con respecto a $\mathbf{w}$. Utilizaremos la regla de la cadena para diferenciar la pérdida con respecto a $\mathbf{w}$. Primero, notemos que $\text{Loss}(x, y, \mathbf{w})$ se expresa como:

$$\text{Loss}(x, y, \mathbf{w}) = (y - \sigma(\mathbf{w} \cdot \phi(x)))^2$$

Donde $p = \sigma(\mathbf{w} \cdot \phi(x))$ es el valor predicho.

Aplicando la regla de la cadena, obtenemos:

$$\nabla_{\mathbf{w}}\text{Loss}(x, y, \mathbf{w}) = \nabla_{\mathbf{w}}(y - p)^2$$

$$= 2(y - p)\nabla_{\mathbf{w}}(y - p)$$

$$= 2(y - p)\nabla_{\mathbf{w}} y$$

Ahora, como y es una constante con respecto a $\mathbf{w}$, su gradiente es cero, entonces:

$$\nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w}) = 2(y - p)\nabla_{\mathbf{w}} p$$

Para calcular $\nabla_{\mathbf{w}} p$, primero necesitamos encontrar $\nabla_{\mathbf{w}} \sigma(z)$, donde $z = \mathbf{w} \cdot \phi(x)$. La derivada de la función logística $\sigma(z)$ con respecto a z es:

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$$

Entonces, por la regla de la cadena, la derivada de p con respecto a z es:

$$\nabla_z p = \frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$$

Finalmente, por la regla de la cadena nuevamente, la derivada de p con respecto a $\mathbf{w}$ es:

$$\nabla_{\mathbf{w}} p = \nabla_z p \cdot \nabla_{\mathbf{w}} z = \sigma(z)(1 - \sigma(z)) \cdot \phi(x)$$

Por lo tanto, el gradiente de la pérdida con respecto a $\mathbf{w}$, $\nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w})$, en términos del valor predicho $p = \sigma(\mathbf{w} \cdot \phi(x))$, es:

$$\nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w}) = 2(y - p)\sigma(z)(1 - \sigma(z)) \cdot \phi(x)$$

Al final queda que:

$$\nabla_{\mathbf{w}} \text{Loss}(x, y, \mathbf{w}) = 2(y - p) \cdot p \cdot (1 - p) \cdot \phi(x)$$

3. (Opcional) Supongamos que hay un dato (x, y) con algún $\phi(x)$ arbitrario y $y = 1$. Especifica las condiciones para $\mathbf{w}$ para hacer que la magnitud del gradiente de la pérdida con respecto a $\mathbf{w}$ sea arbitrariamente pequeño (es decir, que minimice la magnitud del gradiente). ¿Es posible que la magnitud del gradiente con respecto a $\mathbf{w}$ sea exactamente cero? Puedes considerar que la magnitud de $\mathbf{w}$ sea arbitrariamente grande pero no infinita.

Solución:
En los ejemplos que miramos en clase, los resultados de w, nunca llegaban a 0, las era muy poquito entre lo que cambiaba las w una entre otras, el gradiente de la pérdida con respecto a w no puede ser exactamente cero, se puede utilizar el algoritmo de descenso de gradiente para ajustar los pesos w de manera eficiente y minimizar la magnitud del gradiente.

Clasificación de sentimientos

En este problema vamos a construir un clasificador lineal binario que lee reseñas de películas y adivina si son “positivos” o “negativos”.

1. (Opcional) Corre tu predictor lineal con el extractor de características extractCharacterFeatures . Experimenta con distintos valores de □ para ver cuál produce el menor error de validación. Debes observar que este error es casi tan pequeño como el producido con el uso de conteo de palabras como características. ¿Por qué es este el caso? Construye una reseña (una oración) en donde el conteo de □-gramas probablemente sea mejor que el conteo de palabras y explica brevemente por qué es el caso.

Solución:

Clasificación de toxicidad y pérdida máxima de grupo

Recordemos que los modelos entrenados (de la forma estándar) para minimizar la pérdida promedio funcionan bien en promedio pero mal para ciertos grupos, y que podemos mitigar este problema minimizando la pérdida máxima de grupo. En este problema, vamos a comparar las funciones objetivo de la pérdida promedio y la pérdida máxima de grupo con un contexto de juguete inspirado en un problema real con modelos de clasificación de toxicidad.

Los clasificadores de toxicidad son diseñados para asistir en la moderación de foros en línea prediciendo si un comentario es tóxico o no, de tal forma que los comentarios predecidos como tóxicos puedan ser marcados para revisión humana. Desafortunadamente, se ha mostrado que algunos modelos clasifican comentarios no-tóxicos que mencionan identidades demográficas como tóxicos. Este comportamiento puede surgir si suponemos que los comentarios tóxicos en los datos mencionan frecuentemente identidades demográficas, y como resultado, los modelos aprenden a correlacionar falsamente la toxicidad con la mención de estas identidades.

En este problema, vamos a estudiar un contexto de juguete que ilustra el problema de la correlación espuria: La entrada x es un comentario (como cadena) realizado en un foro virtual; la etiqueta $y \in \{-1, +1\}$ es la toxicidad del comentario ($y = +1$ es tóxico, $y = -1$ es no-tóxico); $d \in \{0, 1\}$ indica si el texto contiene una palabra que se refiere a una identidad demográfica; y $t \in \{0, 1\}$ indica si el comentario incluye ciertas palabras "tóxicas". El comentario x es asignado a un vector de características $\phi(x) = [1, d, t]$ donde 1 es el término de sesgo (para prevenir el caso $\mathbf{w} \cdot \phi(x) = 0$ en las preguntas de abajo). Se proveen algunos ejemplos simples abajo, donde se subraya las palabras tóxicas y las que se refieren a una identidad demográfica:

Comentario (x)	Toxicidad (y)	Menciona	
		Identidad (d)	Tóxicas (t)
"Hermosillo sucks!"	+1	0	1
"I'm a woman in computer science"	-1	1	0
"The hummingbird sucks nectar from the flower"	-1	0	1

Supongamos que se nos proveen los siguientes datos de entrenamiento, resumidos con la cantidad de veces que cada combinación (y, d, t) aparece en el conjunto de entrenamiento.

y	d	t	# datos
-1	0	0	63
-1	0	1	27
-1	1	0	7
-1	1	1	3
1	0	0	3
1	0	1	7
1	1	0	27
1	1	1	63

De la tabla de arriba, podemos ver que 70 de los 100 comentarios tóxicos incluyen palabras tóxicas, y 70 de los 100 comentarios no-tóxicos no contienen palabras tóxicas. Adicionalmente, la toxicidad del comentario t está altamente correlacionada con menciones de identidades demográficas d (bajo el supuesto de que los comentarios tóxicos tienden a dirigirse a ellas): 90 de los 100 comentarios tóxicos incluyen menciones de identidades demográficas, y 90 de los 100 comentarios no-tóxicos no las incluyen.
Vamos a considerar clasificadores lineales de la forma:

$$f_{\mathbf{w}}(x) = \text{sign}(\mathbf{w} \cdot \phi(x)),$$

donde se define a $\phi(x)$ como arriba. Normalmente entrenaríamos los clasificadores para minimizar ya sea la pérdida promedio o la pérdida máxima de grupo, pero por simplicidad, vamos a comparar dos clasificadores lineales (los cuales pudieran no minimizar ninguno de los objetivos):

- Clasificador **D**: $\mathbf{w} = [-0.1, 1, 0]$
- Clasificador **T**: $\mathbf{w} = [-0.1, 0, 1]$

Para nuestra función de pérdida, vamos a estar usando la pérdida cero-uno, de tal forma que la pérdida por grupo es:

$$TrainLoss_g(w) = \frac{1}{|Dtrain(g)|} \sum_{(x,y) \in Dtrain(g)} 1[f_{\mathbf{w}}(x) \neq y]$$

Recuerda la definición de la pérdida máxima de grupo:

$$TrainLoss_{max}(\mathbf{w}) = \max_{\mathcal{G}} TrainLoss_{\mathcal{G}}(w).$$

Para capturar el problema de la correlación espuria, definamos grupos basados en el valor de (y, d). Entonces tenemos cuatro grupos: $(y = +1, d = 1)$, $(y = +1, d = 0)$, $(y = -1, d = 1)$, y $(y = -1, d = 0)$. Por ejemplo, el grupo $(y = -1, d = 1)$ se refiere a comentarios no-tóxicos con menciones a identidades demográficas.

1. (Opcional) Describe el comportamiento del clasificador **D** y del clasificador **T**.

Solución:
Si y solo si el clasificador D , tiene la forma w = [-0.1, 1, 0] , cuando $f_w(x)$= 1 , la salida del clasificador será 1 (tóxico) , el clasificador D etiquetará el comentario como tóxico si y solo si la entrada x menciona una identidad demográfica (d = 1)

Si y solo si el clasificador T , tiene la forma w = [-0.1, 0, 1], entonces $f_w(x)$= 1 implica que la salida del clasificador será 1 (tóxico) , clasificador T etiquetará el comentario como tóxico si y solo si la entrada x contiene palabras tóxicas (t = 1) .

2. (Opcional) Calcula las siguientes cantidades del clasificador **D** usando el conjunto de datos de arriba:

   (a) La pérdida promedio del clasificador **D** en general.

   (b) La pérdida promedio del clasificador **D** por cada grupo.

   (c) La pérdida máxima de grupo del clasificador **D**.

Solución:

3. (Opcional) Calcula las siguientes cantidades del clasificador **T** usando el conjunto de datos de arriba:

   (a) La pérdida promedio del clasificador **T** en general.

   (b) La pérdida promedio del clasificador **T** por cada grupo.

   (c) La pérdida máxima de grupo del clasificador **T**.

   ¿Qué clasificador tiene la menor pérdida promedio? ¿Qué clasificador tiene la menor pérdida máxima de grupo?

Solución:

4. (Opcional) Diferentes clasificadores nos llevan a diferentes predicciones acertadas y a diferentes comentarios siendo incorrectamente rechazados (falsos positivos). La clasificación correcta de un comentario no-tóxico es bueno para quien escribe el comentario, pero cuando ningún clasificador tiene precisión perfecta, ¿Cómo deben distribuirse las clasificaciones correctas entre los comentaristas? Recuerda los principios de distribución justa discutidos en clase y con base en ellos: Elabora un argumento para el uso de la pérdida promedio como función objetivo. ¿Cuál clasificador (**D** o **T**) elegirías para una red social real con la finalidad de identificar publicaciones tóxicas para revisión humana? Luego haz lo mismo pero argumentando para el uso de la pérdida máxima de grupo como función objetivo, igualmente apela a alguno de los principios de distribución justa.

Solución:
Cuando vamos a elegir un clasificador para las redes sociales, se debe considerar impacto de las clasificaciones, se debe de tomar en cuenta el principio de equidad, el usuario debe de recibir un trato justo en las plataformas de redes sociales, no sentir excluido, atacado, el principio de minimización del daño**,** este principio implica minimizar el daño causado por las decisiones incorrectas del clasificador.

La pérdida promedio garantiza que el impacto promedio de las clasificaciones incorrectas se distribuya de manera equitativa entre los usuarios. el clasificador minimiza el riesgo de sesgo contra grupos específicos de usuarios, lo que resulta en una distribución justa de los errores entre todos los usuarios.
La pérdida máxima de grupo se enfoca en identificar el peor escenario posible para un grupo de usuarios, garantiza que el clasificador no cometa demasiados errores en grupos minoritarios o subrepresentados. Minimizando la pérdida máxima de grupo, el clasificador asegura una precisión aceptable para todos los subgrupos.

5. El aprendizaje máquina puede pensarse como el proceso de transformar datos en modelos, pero ¿de dónde vienen los datos? En el contexto de recolección de datos para el entrenamiento de los modelos de aprendizaje máquina para la clasificación de toxicidad, quién determina si un comentario debe ser identificado como tóxico o no es muy importante (es decir, si $y = 1$ o si $y = -1$ en las tablas de datos de arriba). Algunas de las opciones comúnmente empleadas son:

   - Reclutar personas en una plataforma de crowdsourcing para anotar cada comentario.
   - Contratar científicos sociales para establecer criterios de toxicidad y anotar cada comentario.
   - Solicitarle a los usuarios de la plataforma que clasifiquen comentarios.
   - Solicitarle a los usuarios de la plataforma que deliberen acerca de y decidan sobre estándares comunitarios y criterios de toxicidad, utilizando un proceso de diseño participativo.

   ¿Qué métodos usarías para determinar la toxicidad de comentarios para usarlos como parte del entrenamiento de un clasificador? ¿Por qué? Explica por qué los métodos que eliges son mejores que los otros enlistados.

Solución:
Para la recolección de datos para el entrenamiento de aprendizaje, usaría varios métodos para así tener más información, como Solicitarles a los usuarios de la plataforma que deliberen acerca de y decidan sobre estándares comunitarios y criterios de toxicidad, utilizando un proceso de diseño participativo. Por qué se involucrarían los usuarios para definir los criterios, aparte ahora la gente usa más el internet para hacer en cuentas y más rápido de hacer. Otro método que se puede usar un algoritmo para determinar la toxicidad de los comentarios, sería más eficiente y automatizada.

Agrupamiento con K-medias

Supongamos que tienes un extractor de características que produce vectores bidimensionales de características, y un conjunto de entrenamiento de juguete $\mathcal{D}_{\text{train}} = \{x_1, x_2, x_3, x_4\}$ donde,

$$(x_1) = [0, 0]$$
$$(x_2) = [4, 0]$$
$$(x_3) = [6, 0]$$
$$(x_4) = [11, 0]$$

1. Corre el método de 2-medias sobre este conjunto de datos hasta llegar a la convergencia. Escribe el desarrollo de tu trabajo. ¿Cuáles son las asignaciones finales z y los centros de los grupos μ? Corre este algoritmo dos veces con los siguientes centros iniciales:

   (a) $\mu_1 = \phi(x_1) = [0, 0]$ y $\mu_2 = \phi(x_4) = [11, 0]$

   (b) $\mu_1 = \phi(x_1) = [0, 0]$ y $\mu_2 = \phi(x_2) = [4, 0]$

Solución:
El problema nos indica que tenemos un conjuto de datos que son los siguientes
$(x_1) = [0, 0]$
$(x_2) = [4, 0]$
$(x_3) = [6, 0]$
$(x_4) = [11, 0]$
Distancia de los centros :

D_{train}	μ_1 [0,0]	μ_2 [11,0]	
x_1	0	11	μ_1
x_2	4	7	μ_1
x_3	6	5	μ_2
x_4	11	0	μ_2

Actualización de los centros de los grupos:
El promedio de los puntos asignados al grupo 1, [(0 + 4) / 2, 0] = [2, 0].
El promedio de los puntos asignados al grupo 2, [(6 + 11) / 2, 0] = [8.5, 0].
se vuelve hacer otra vez la iteracion y sale lo mismos grupos entonces se quda con los mismo centros actualizados que son [2,0] y [8.5,0]

Sacamos la dintancia para los otros centros:

D_{train}	μ_1 [0,0]	μ_2 [4,0]	
x_1	0	4	μ_1
x_2	4	0	μ_2
x_3	6	2	μ_2
x_4	11	7	μ_2

Actualización de los centros de los grupos:
El promedio de los puntos asignados al grupo 1, que es [0, 0].
El promedio de los puntos asignados al grupo 2, que es [(4 + 6 + 11) / 3, 0] = [7, 0].
se vuelve hacer otra vez la iteracion y sale lo mismos grupos entonces se quda con los mismo centros actualizados que son [0,0] y [7,0]

2. Implementa la función kmeans.

Solución: codigo en archivo.py

3. (Opcional) Si escalamos todas las dimensiones de nuestros centroides iniciales y los datos de los puntos por algún factor distinto de cero, ¿garantizamos que podremos reconstruir las mismas agrupaciones después de correr k-medias? Es decir, ¿los mismos puntos en los datos pertenecerán a los mismos grupos antes y después de escalarlos? ¿Y si escalamos solo algunas dimensiones? Si tu respuesta es sí, provee una explicación breve. Si tu respuesta es no, provee un contraejemplo.

Solución: