LABORATORIO 4

Boosting k-Means con K-D Trees

Docente: Rosa Yuliana Gabriela Paccotacya Yanque

1 Competencia del Curso

Conoce e investiga los métodos de acceso multi dimensional, métrico y aproximado

2 Competencia del Laboratorio

- Describir, implementar k-Means usando la estructura de datos KD-Tree
- Analizar el impacto del KD-Tree en el algoritmo de k-Means.

3 Equipos y Materiales

- Un computador
- Lenguaje de Programación C++
- Bibliografía sugerida [1], [2]

4 k-Means

Una de las aplicaciones más conocidas dela búsqueda del vecino más cercano es en los algoritmos de clustering. El clustering es la forma principal de aprendizaje no supervisado. Los algoritmos de clustering toman un conjunto de datos sin etiquetar e intentan recopilar la mayor cantidad de información posible sobre su estructura, agrupando puntos de datos similares y separando los diferentes. El algoritmo más comun de clustering es k-Means.

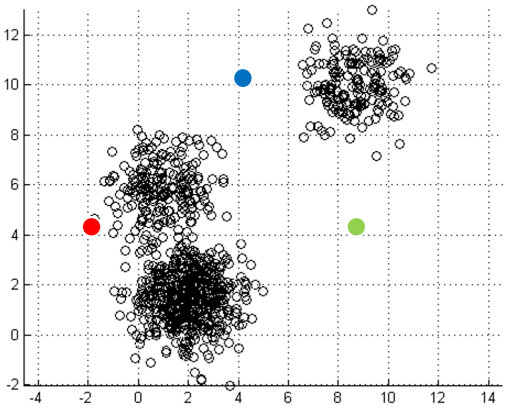
4.1 Problema

Dado un número de k clusters, encuentre los centros de cada cluster que minimice la suma de distancias cuadráticas desde cada punto al centro de su cluster.

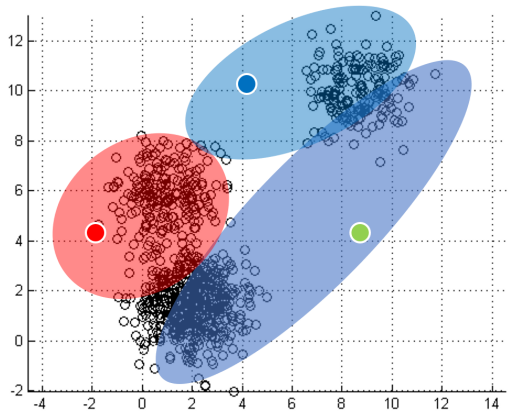
4.2 Algoritmo

Las Figuras 1 y 2 ilustran el algoritmo de K-means.

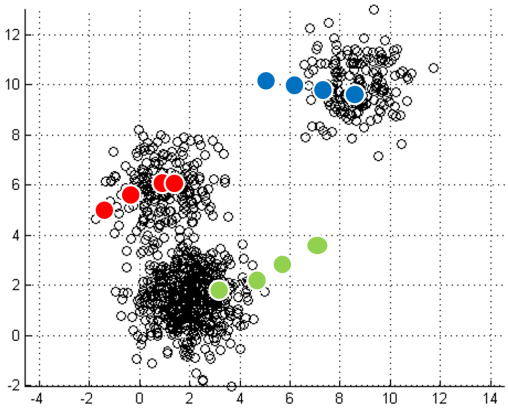
1. Defina los centroides $\boldsymbol{\mu}$ de los clusters iniciales de manera random.
2. Asigne cada punto al cluster $\mathbf{z}$ que tenga el centroide más cercano. Para encontrar el cluster con el centroide más cercano, el algoritmo debe calcular la distancia euclidiana cuadrática entre todos los puntos y cada centroide.
3. Vuelva a calcular los centroides. Cada componente del centroide se actualiza y se establece como el promedio de los componentes correspondientes de los puntos que pertenecen al grupo.
4. Repita los pasos 2 y 3 hasta que los puntos ya no puedan cambiar de grupo(convergencia).



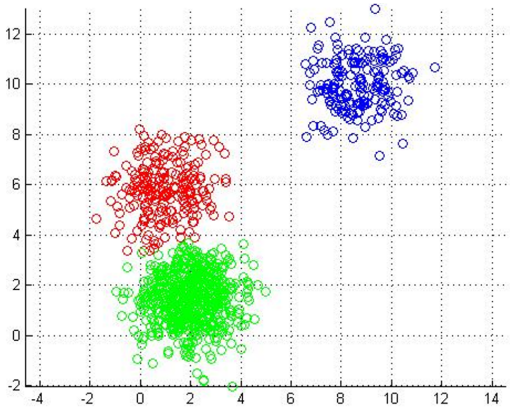
(a) Elige tres centroides al azar



(b) Asigne cada punto a su centroide más cercano



(c) El nuevo centroide es el promedio de todos los puntos que se le asignan. Cada iteración acerca los centroides a su destino.



(d) Los centroides dejan de moverse. Cada punto es etiquetado con su centroide más cercano.

Figure 1: Visualización del comportamiento del algoritmo de k-means

Algorithm: K-means

Initialize $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_K]$ randomly.

For $t = 1, \ldots, T$:

Step 1: set assignments $\mathbf{z}$ given $\boldsymbol{\mu}$

For each point $i = 1, \ldots, n$:

$$z_i \leftarrow \arg\min_{k=1,\ldots,K} \|\phi(x_i) - \mu_k\|^2$$

Step 2: set centroids $\boldsymbol{\mu}$ given $\mathbf{z}$

For each cluster $k = 1, \ldots, K$:

$$\mu_k \leftarrow \frac{1}{|\{i : z_i = k\}|} \sum_{i:z_i=k} \phi(x_i)$$

Figure 2: Algoritmo del k-means

5 Actividad (Este laboratorio pueden realizarlo en grupos de hasta 2 personas)

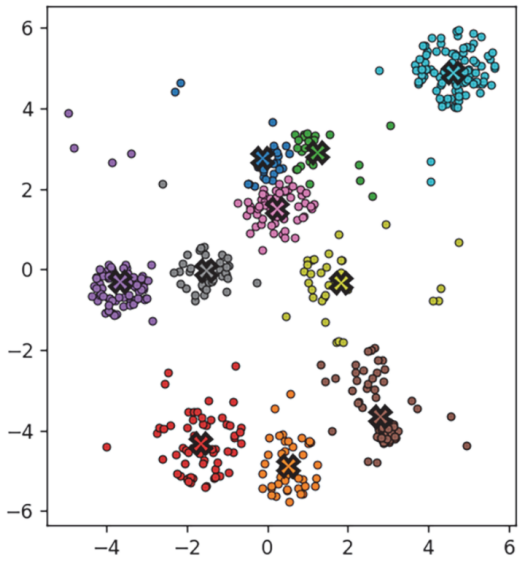
Cuando describimos el código para k-means, vimos que el paso 2, donde asignamos cada punto exactamente a uno de los centroides, es una búsqueda de fuerza bruta entre todas las combinaciones de puntos y centroides. Ahora la pregunta es, ¿podemos acelerarlo de alguna manera?

Si lo piensan bien, para cada punto buscamos su vecino más cercano entre el conjunto de centroides, y ya conocemos al menos una estructura de datos para acelerar esta búsqueda. Entonces, las actividades del laboratorio son:

1. Implementa dos versiones de k-means en C++, una que hace la busqueda delcentroide más cercano con fuerza bruta, y la otra usando un KD-Tree.

   (a) Analizar el **costo computacional** de ambas implmentaciones.

   (b) Ejecutar 10 veces ambas implementaciones de k-means con k = 18 , analizar el cambio de los centroides y tiempo de ejecución. Visualizar algunos resultados(clusters y centroides) usando cualquier lenguaje de programación. Ejemplo:



   (c) Utilizando el conjuntos de archivos adjunto en este trabajo que contiene 2400 puntos, analizar el **tiempo de ejecución** de las dos versiones de k-means con un k fijo $k = \{5, 15, 25, 50, 75\}$ y numero de puntos $n = \{1000, 1150, 1300, 1450, 1600, 1750, 1900, 2050, 2200, 2400\}$. Ejemplo: (Figure 3)

   (d) Analizar el **tiempo de ejecución** de las dos versiones de k-means con $k = \{5, 15, 25, 50, 75, 100, 125, 150, 200\}$ y numero de puntos fijo $n = \{1000, 1450, 1900, 2400\}$. Ejemplo: (Figure 4)

2. Documentar sus hallazgos en un formato de artículo de investigación (formato de libre elección), considerando aspectos del marco teórico, metodología, resultados, conclusiones, discusión y bibliografía.Aseguráte de responder lassiguinetes preguntas en tu documento;

   - ¿Por qué usar KD-Trees y no otras estructuras? Qué otras estructuras podrían usarse?
   - ¿Para qué tamaño de datos en 2D es beneficioso usar un KD-Tree en el k-means? ¿Por qué?
   - ¿Para qué valor de k beneficioso usar un KD-Tree en el k-means? ¿Por qué?

6 Entregables

Al finalizar el estudiante deberá:

- Elaborar un documento en Latex, formato articulo (IEEE u otro), donde registrará todas las actividades desarrolladas y explique las funciones inplementadas.

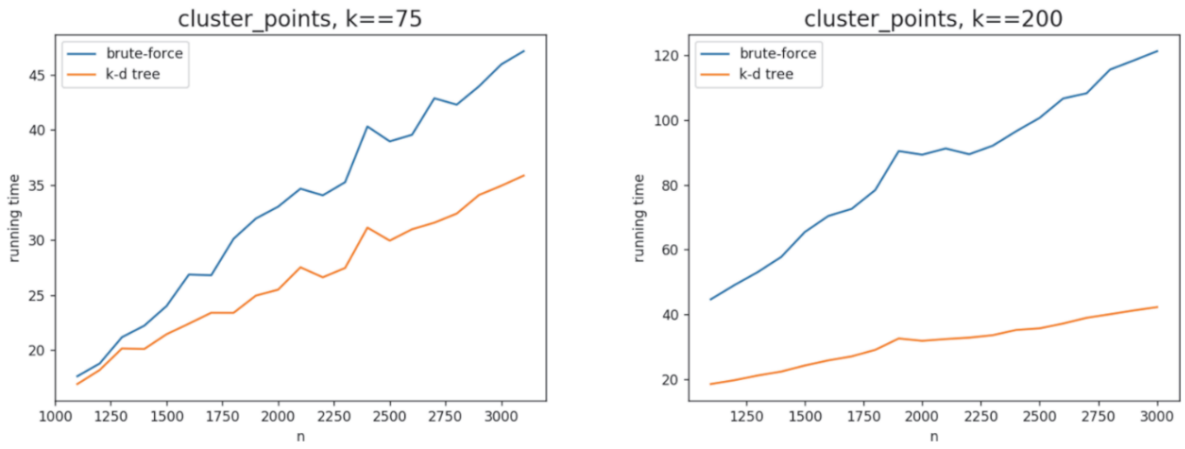


Figure 3: Tiempo de ejecución con diferentes versiones del k-means con un número fijo de centroides y variando el número de puntos.

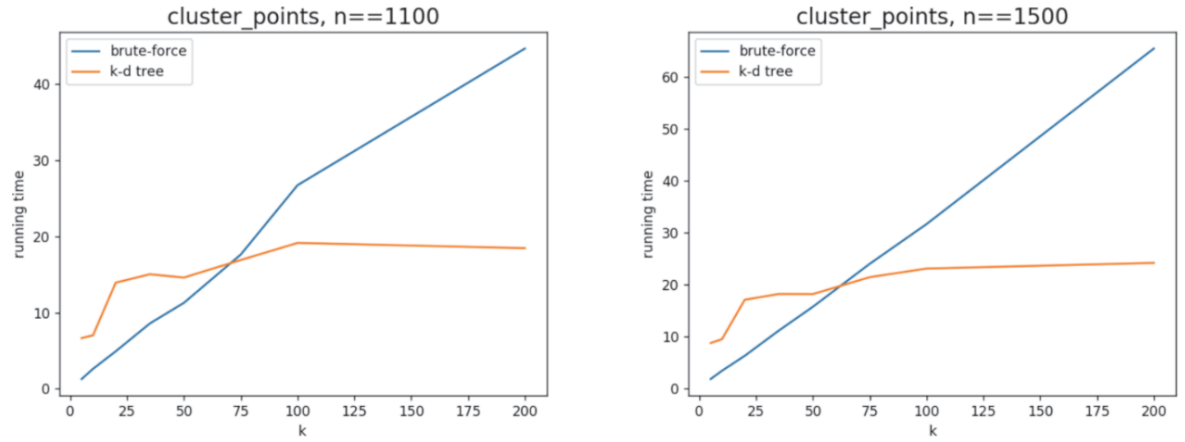


Figure 4: Tiempo de ejecución con diferentes versiones del k-means con un número fijo de puntos y variando el número de centroides.

- Podrá incluir en el documento elaborado solo capturas de pantalla de la ejecución de su algoritmo en el documento elaborado
- Agregar **solo los archivos fuente de los código desarrollados** en un archivo comprimido con su Apellido (LABXXApellido.zip)
- Deberán de subir a la plataforma Moodle a tiempo el documento elaborado en **formato PDF** y el archivo comprimido con los códigos elaborados.

7 Rúbrica de Evaluación

La evaluación constará de una evaluación oral (exposición del código y ejecución) y del informe elaborado.

- **IMPORTANTE** En caso de copia o plagio o similares todos los alumnos implicados tendrán sanción en toda la evaluación del curso.

References

[1] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[2] Hanan Samet. An overview of quadtrees, octrees, and related hierarchical data structures. *Theoretical Foundations of Computer Graphics and CAD*, pages 51–68, 1988.