Implementación de Algoritmo K-Means sin Librerías

ROBBYEL ELIAS, CARLOS JERONIMO, ANDRUS LOPEZ

March 12, 2025

Abstract

Este documento describe la implementación del algoritmo K-Means sin el uso de librerías en Python. Se explican los pasos principales del algoritmo y se proporciona una implementación sencilla y comentada del mismo.

1 Introducción

El algoritmo K-Means es un método de agrupamiento utilizado para dividir un conjunto de datos en k grupos o clusters. Su objetivo es minimizar la distancia intra-cluster y maximizar la distancia inter-cluster. En esta implementación, se presenta una versión simple sin el uso de librerías externas.

2 Descripción del Algoritmo

El algoritmo sigue los siguientes pasos:

1. Seleccionar k puntos iniciales como centroides.
2. Asignar cada punto de datos al centroide más cercano.
3. Recalcular los centroides como el promedio de los puntos asignados.
4. Repetir hasta que los centroides no cambien significativamente.

3 Implementación en Python

A continuación se presenta el código en Python con explicaciones detalladas:

```
# Definimos los datos de entrada
# Cada punto es representado por sus coordenadas (x, y)
datos = [
    [1, 2], [2, 1], [1.5, 1.8],
    [5, 8], [6, 9], [6.5, 8.5],
    [1, 0.6], [9, 11], [8, 10],
    [3, 4], [2.5, 3.5], [3.2, 3.8]
]
```

```
# Número de grupos
k = 3

# Función para calcular la distancia euclidiana entre dos puntos
def distancia(a, b):
    return ((a[0] - b[0]) * 2 + (a[1] - b[1]) * 2) ** 0.5

# Inicializamos los centroides con los primeros k puntos
centroides = datos[:k]

# Número de iteraciones fijas
for iteracion in range(10):
    asignaciones = []  # Lista para guardar el cluster de cada punto

    # Asignamos cada punto al centroide más cercano
    for punto in datos:
        distancias = [distancia(punto, centro) for centro in centroides]
        cluster = distancias.index(min(distancias))
        asignaciones.append(cluster)

    # Recalcular los centroides
    nuevos_centroides = [[0, 0] for _ in range(k)]
    conteo = [0] * k

    for indice, punto in enumerate(datos):
        grupo = asignaciones[indice]
        nuevos_centroides[grupo][0] += punto[0]
        nuevos_centroides[grupo][1] += punto[1]
        conteo[grupo] += 1

    for i in range(k):
        if conteo[i] != 0:
            nuevos_centroides[i][0] /= conteo[i]
            nuevos_centroides[i][1] /= conteo[i]
        else:
            nuevos_centroides[i] = centroides[i]

    centroides = nuevos_centroides

    print(f"Iteración iteracion + 1 - Centroides: centroides")

print("Asignación final de puntos a clusters:", asignaciones)
```

4 Conclusiones

Esta implementación del algoritmo K-Means permite agrupar datos sin necesidad de librerías especializadas. Se logra con un código sencillo y humanizado que puede adaptarse a diferentes aplicaciones.