Universidad de Nacional de San Agustín
Facultad de Ingeniería de Producción y Servicios
Escuela Profesional de Ciencia de la Computación

Laboratorio 01:
La maldición de la dimensionalidad

Link Github:

https://github.com/i-am-sergio/EDA_UNSA/tree/main/Laboratorio_1

Asignatura: **Estructuras de Datos Avanzadas**

Elaborado por: Sergio Daniel Mogollon Caceres

Arequipa - Perú
2023

ACTIVIDAD

- **Generar 100 puntos aleatorios entre 0 y 1 de dimensión d (Hint:https://en.cppreference.com/w/cpp/numeric/random/uniform_real_distribution)**
- **Calcular la distancia entre todos los pares de puntos (Distancia Euclidiana) (Hint 4950 distancias)**
- **Generar un histograma (pueden usar Python) de las distancias obtenidas para cada dimensión.**

Introducción

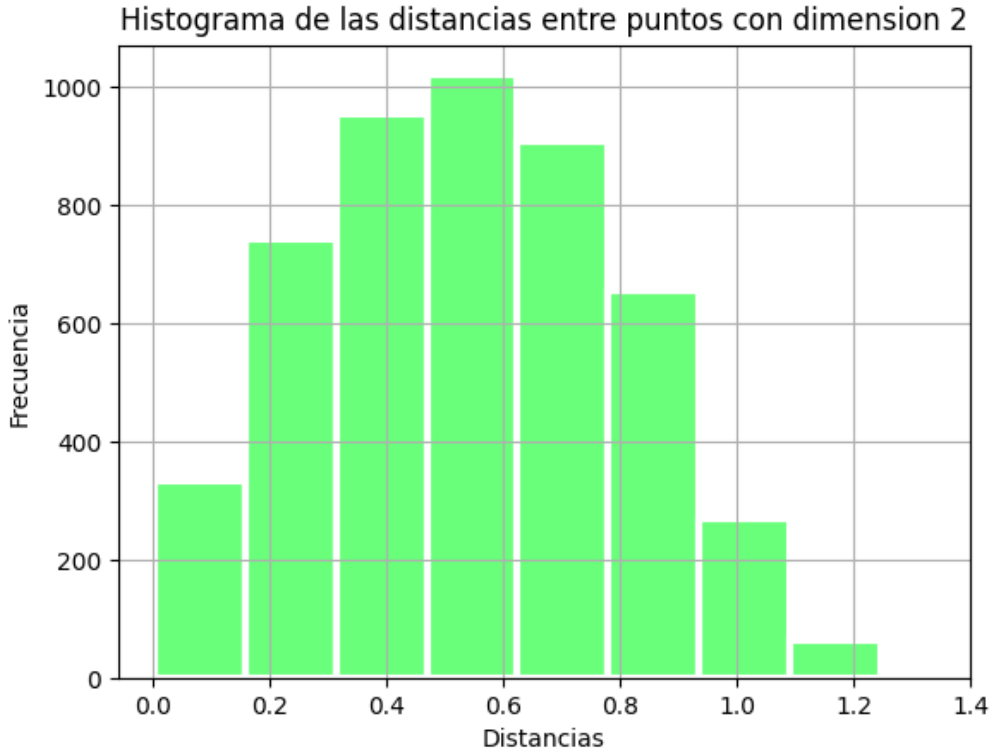
El presente informe tiene como objetivo analizar las distancias entre puntos en diferentes dimensiones, específicamente en dimensiones *10, 50, 100, 500, 1000, 2000, y 5000*. Para llevar a cabo este análisis, se generaron histogramas que representan la distribución de estas distancias y se observaron las tendencias y cambios a medida que se aumenta la dimensión.

Metodología

Para llevar a cabo este análisis, se utilizó un conjunto de datos que consiste en generar 100 puntos en un espacio de alta dimensión. A partir de estos puntos, se calcularon todas las posibles distancias entre ellos y se construyeron los histogramas correspondientes. El código utilizado para este análisis usa C++ y python, y se encuentra disponible en el siguiente repositorio de GitHub: https://github.com/i-am-sergio/EDA_UNSA/tree/main/Laboratorio_1

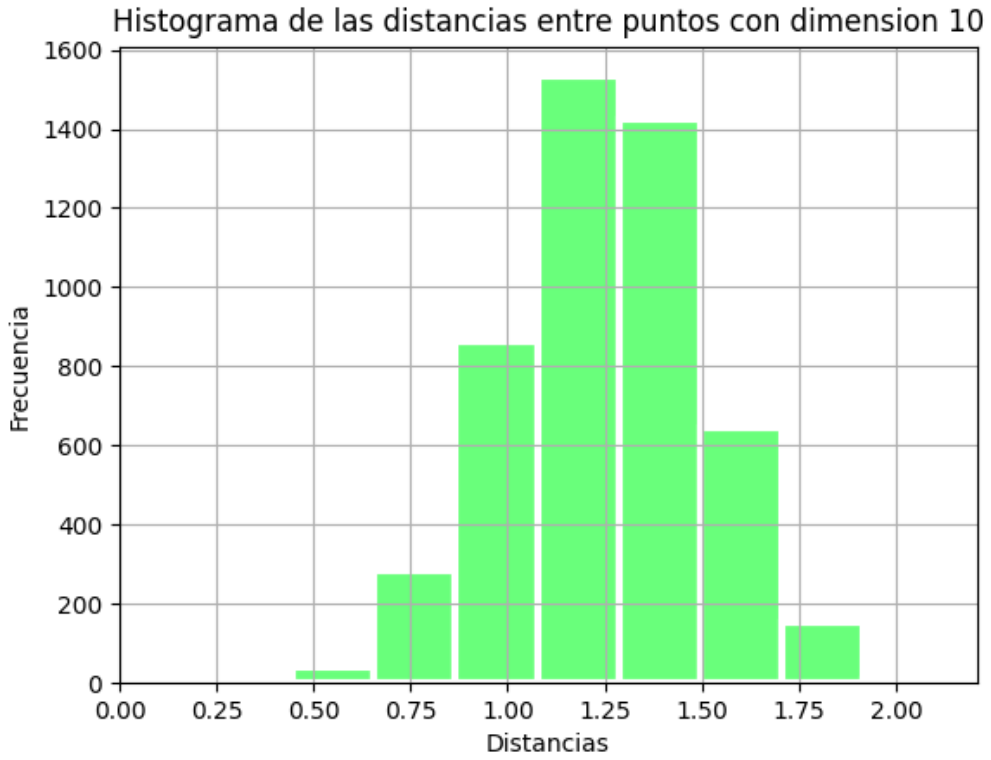
Análisis

1. **Dimensión 2**
   En esta dimensión, observamos una distribución de distancias que es notablemente diferente de las dimensiones superiores. En este caso, el histograma muestra una **concentración de distancias pequeñas**, lo que sugiere que los puntos en dimensiones bajas tienden a estar más cercanos entre sí. A medida que aumentamos la dimensión, esta tendencia cambia, y las distancias comienzan a dispersarse más ampliamente, como se ha descrito en los análisis previos. **Los valores oscilan entre 0 y 1.3**



2. **Dimensión 10**

   En una dimensión baja, como 10, se observa que la mayoría de las distancias **tienden a agruparse cerca de un valor central** en el histograma. Esto sugiere que las distancias entre los puntos no varían significativamente y que la mayoría de los puntos están relativamente cerca entre sí. A pesar de que aumentan un poco, aún se mantienen con distancias pequeñas.

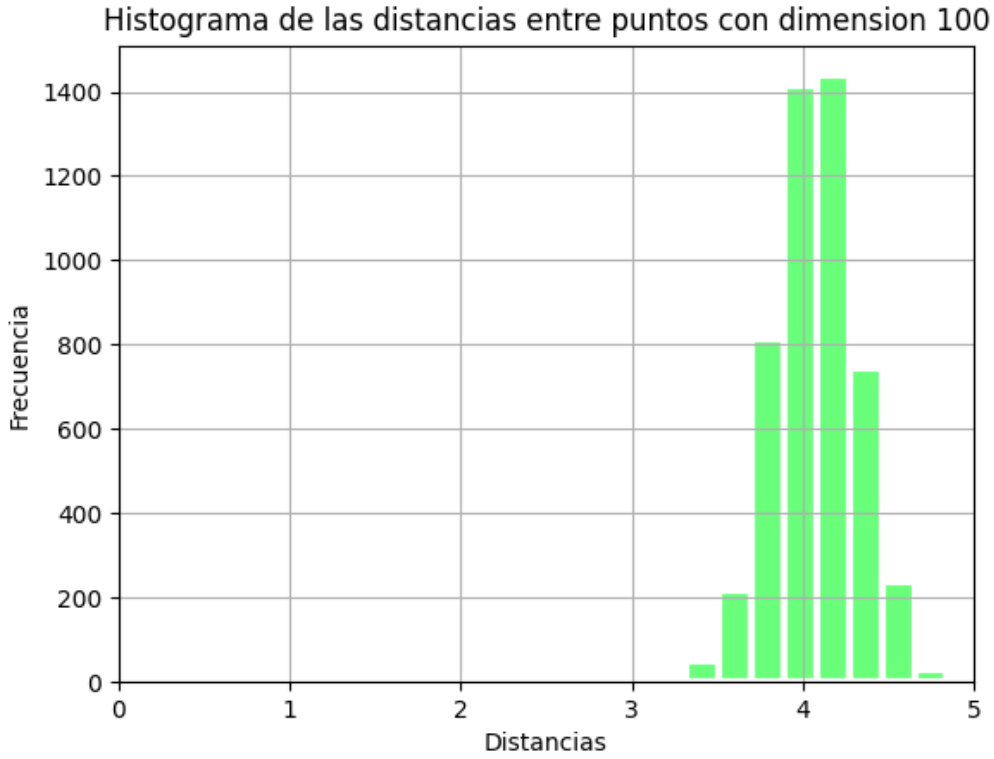


3. **Dimensión 50**

   A medida que aumentamos la dimensión a 50, comenzamos a notar un cambio en la distribución de las distancias. El histograma **muestra un incremento para el valor de las distancias** que oscila entre 3 y 5 (lado derecho), lo que indica que en dimensiones más altas, las distancias tienden a ser más variables y es más probable encontrar distancias significativamente mayores. **Los valores oscilan entre 2 y 4.**

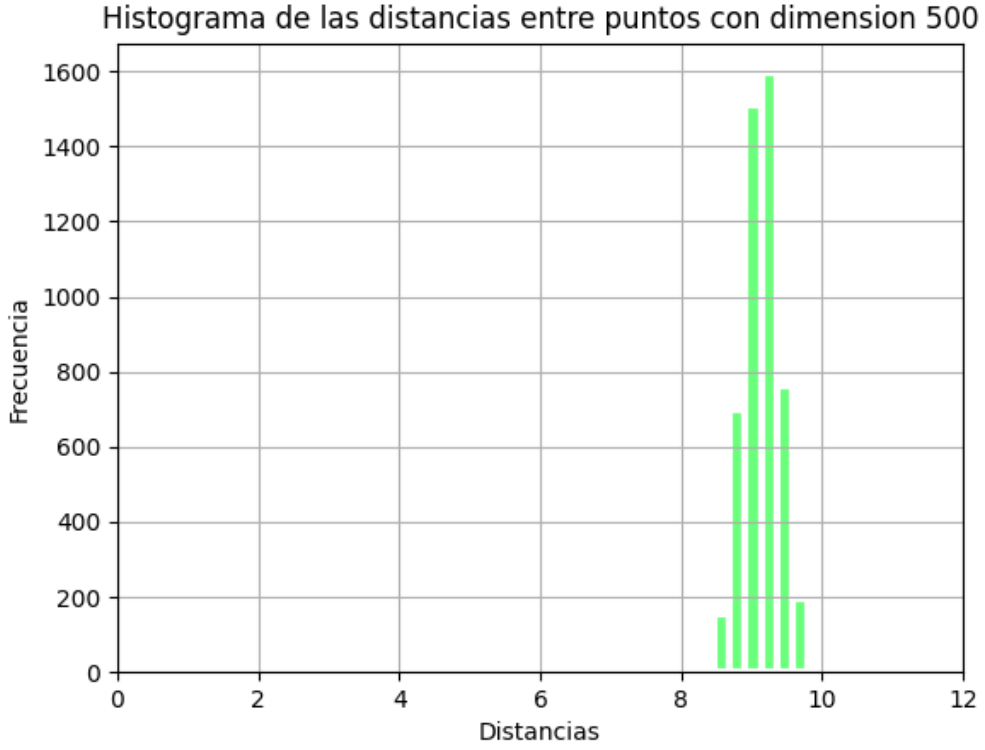
4. **Dimensión 100**

   En dimensiones 100, este patrón de dispersión de distancias se vuelve aún más evidente. El histograma muestra una distribución sesgada hacia la derecha, lo que significa que la mayoría de las distancias tienden a ser pequeñas, pero hay un número significativo de distancias mucho más grandes. **Los valores oscilan entre 3 y 5**



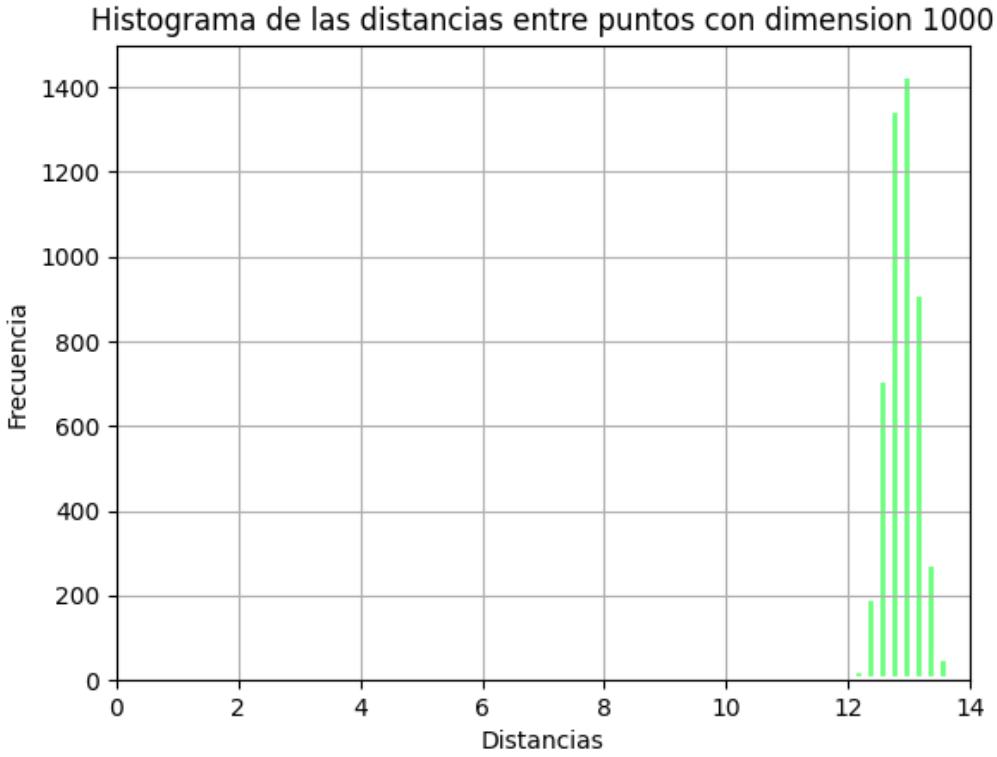
5. **Dimensión 500**

   Al aumentar la dimensión a 500, la maldición de la dimensionalidad se hace aún más pronunciada. El histograma muestra una distribución altamente sesgada hacia la derecha, con la mayoría de las distancias siendo pequeñas, pero una proporción considerable de distancias muy grandes. Esto indica que en dimensiones muy altas, las distancias tienden a ser poco informativas y no proporcionan una representación clara de la similitud entre puntos. **Los valores oscilan entre 8 y 10.**



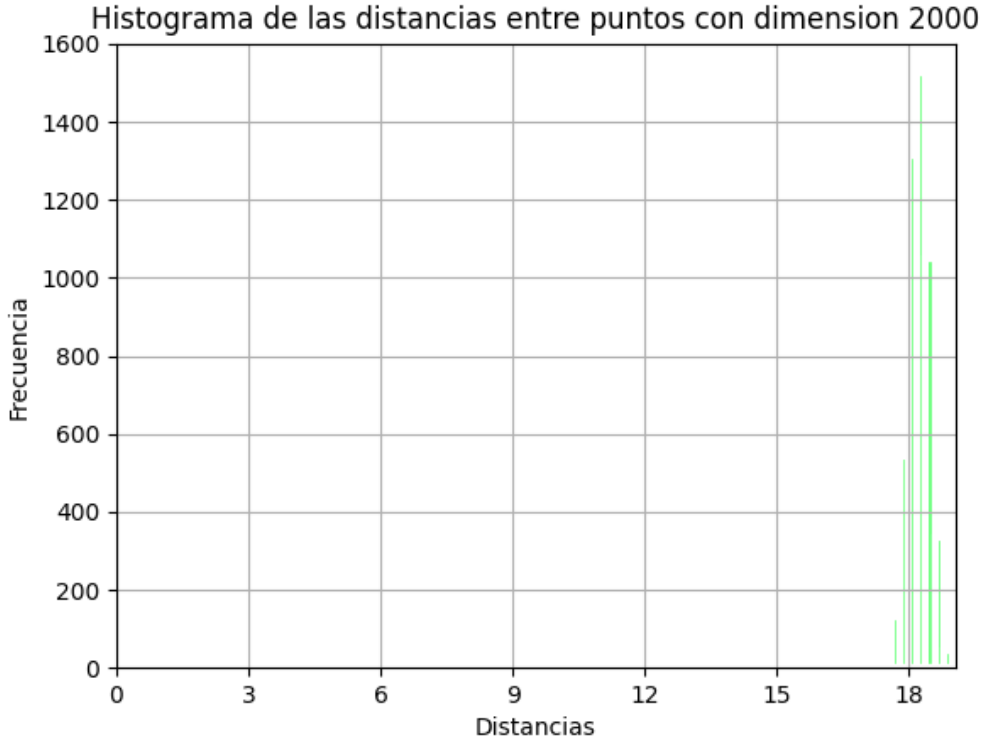
6. **Dimensión 1000**

   En dimensiones 1000, el fenómeno de la maldición de la dimensionalidad continúa, lo que significa que en dimensiones muy altas, las distancias pierden su capacidad para discriminar entre puntos de manera efectiva.



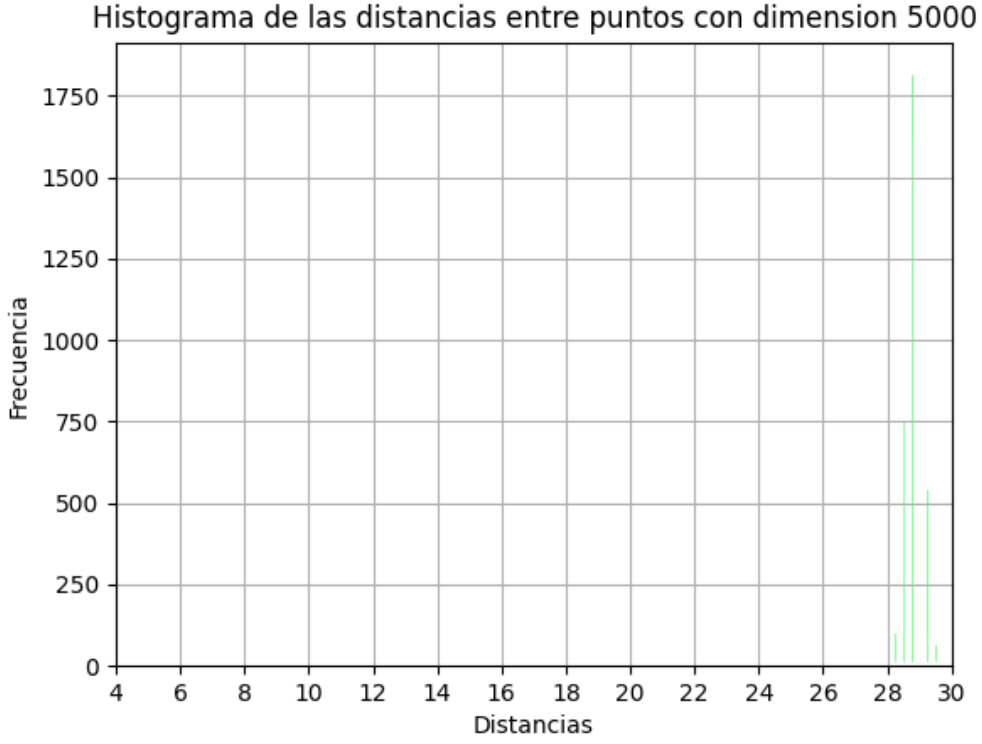
7. **Dimensión 2000**

   La tendencia se mantiene en dimensiones 2000, con una distribución altamente sesgada hacia la derecha en el histograma de distancias. Esto sugiere que en dimensiones extremadamente altas, las distancias tienden a converger en valores similares. **Los valores oscilan entre 17 y 20.**



8. **Dimensión 5000**

   Finalmente, en la dimensión 5000, la maldición de la dimensionalidad se hace aún más evidente. El histograma muestra una concentración aún mayor de distancias, y extremadamente sesgadas hacia la derecha. **Los valores oscilan entre 28 y 29.**



Código

C++ (Algoritmos):

- **Función generarPuntosAleatorios(int, int, Matrix<double>): recibe la dimension y los n puntos que solicite el usuario**

```cpp
#include <iostream>
#include <fstream>
#include <random>
#include <vector>
#include <cmath>
#include <string>

template <typename T>
using Matrix = std::vector<std::vector<T>>;

void generarPuntosAleatorios(int n, int dimension, Matrix<double> &puntos)
{
    std::ofstream datos;
    datos.open("datos_dim" + std::to_string(dimension));
    // Semilla aleatoria
    std::random_device rd;
    std::mt19937 gen(rd());

    // Rango de generacion (0 a 1)
    std::uniform_real_distribution<double> distribucion(0, 1);
    std::vector<double> auxPunto;

    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < dimension; j++)
        {
            double coordenada = distribucion(gen);
            datos << coordenada << ",";
            auxPunto.emplace_back(coordenada);
        }
        datos << "\n";
        puntos.emplace_back(auxPunto);
        auxPunto.clear();
    }
    datos.close();
}
```

- **Funcion distanciaEuclidiana(vector<double>, vector<double>): recibe las coordenadas de dos puntos y calcula la distancia entre ellos**

```cpp
double distanciaEuclidiana(const std::vector<double> &puntoA, const std::vector<double> &puntoB)
{
    double sumaCuadrados = 0;
    for (int i = 0; i < puntoA.size(); ++i)
    {
        double diferencia = puntoA[i] - puntoB[i];
        sumaCuadrados += diferencia * diferencia;
    }
    return std::sqrt(sumaCuadrados);
}
```

- **Función calcularDistancias(Matrix<double>): recibe un vector de puntos y calcula todas sus distancias entre ellos (4950 distancias, Combinatoria(100,2))**

```cpp
void calcularDistancias(Matrix<double> &puntos) // 4950 distancias
{
    std::ofstream distancias;
    distancias.open("distancias_dim" + std::to_string(puntos[0].size()));
    for (int i = 0; i < puntos.size(); i++)
    {
        for (int j = i + 1; j < puntos.size(); j++)
        {
            double distancia = distanciaEuclidiana(puntos[i], puntos[j]);
            // std::cout << "Distancia entre Punto " << i + 1 << " y Punto " << j + 1 << ": " << distancia << std::endl;
            distancias << distancia << "\n";
        }
    }
    distancias.close();
}
```

Python (Graficación):

```python
import matplotlib.pyplot as plt


def graficarhistograma(dimension):
    datos = []
    dimensiones = {
        2: [0, 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4],
        10: [0, 0.25, 0.50, 0.75, 1.00, 1.25, 1.50, 1.75, 2.0],
        50: [0, 1, 2, 3, 4, 5],
        100: [0, 1, 2, 3, 4, 5],
        500: [0, 2, 4, 6, 8, 10, 12],
        1000: [0, 2, 4, 6, 8, 10, 12, 14],
        2000: [0, 3, 6, 9, 12, 15, 18],
        5000: [4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30]
    }

    valores_x = dimensiones.get(dimension, [])

    # Abrir el archivo y leer los datos línea por línea
    with open(f"distancias_dim{dimension}", 'r') as file:
        for line in file:
            dato = float(line.strip())
            datos.append(dato)

    # Crear el histograma
    plt.hist(datos, bins=8, color='#69FF7B',
             edgecolor='white', linewidth=3)
    plt.grid(True)
    plt.xlabel('Distancias')
    plt.ylabel('Frecuencia')
    plt.title(
        f'Histograma de las distancias entre puntos con dimension {dimension}')
    plt.xticks(valores_x)
    plt.savefig(f"dimension{dimension}.png")


with open('env', 'r') as archivo:
    contenido = archivo.read()
    dimension = int(contenido)

graficarhistograma(dimension)
```

Conclusiones

En este análisis de distancias en diferentes dimensiones, se observa claramente el impacto de la maldición de la dimensionalidad. A medida que aumentamos la dimensión, las distancias tienden a agruparse en valores cada vez más similares, lo que hace que las distancias pierdan su capacidad discriminativa. Esto tiene importantes implicaciones para problemas de alta dimensionalidad, como la clasificación y el clustering, donde la elección de métricas de distancia adecuadas se vuelve crítica. La alta dimensionalidad puede ser un problema y es necesario entender el efecto que tiene en los datos y como se ven afectados los algoritmos por lo mismo[1]

Referencias

[1] N. A. Landa Cosio. “La maldición de la dimensionalidad”. Medium. Accedido el 15 de septiembre de 2023. [En línea]. Disponible: https://medium.com/@nicolasarrioja/la-maldición-de-la-dimensionalidad-f7a6248cf9a#:~:text=Esta%20frase%20se%20atribuye%20a,surgen%20problemas%20a%20nivel%20estadístico.