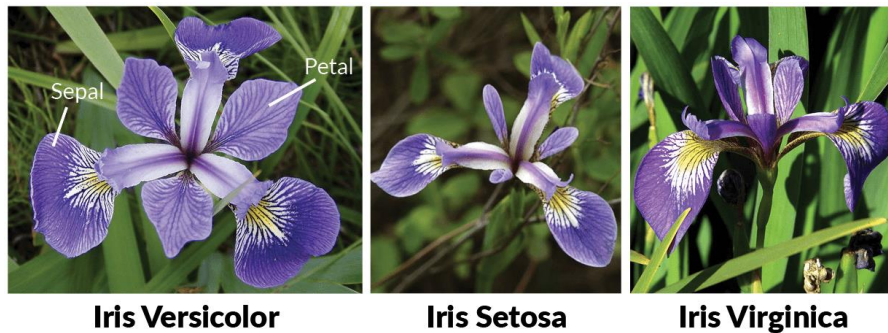


## Búsqueda por Similitud

Profesor Heider Sanchez

El objetivo del laboratorio es analizar el comportamiento de la búsqueda por rango y la búsqueda de los k vecinos más cercano sobre un conjunto de vectores característicos.

Se toma como referencia la colección de imágenes de flores llamada **Iris** en donde cada imagen es representado por un vector característico de 4 dimensiones que recoge información del sépalo y del pétalo. Además, las imágenes están agrupadas en tres categorías: *versicolor*, *setosa* y *virginica*.



### P1. Búsqueda por Rango

Implementar en cualquier lenguaje de programación el algoritmo lineal de búsqueda por rango, el cual recibe como parámetro el objeto de consulta y un radio de cobertura. Luego usando la distancia Euclidiana (ED) se retorna todos los elementos que son cubiertos por el radio.

#### Algorithm RangeSearch(Q, r)

```
1. result = [ ]
2. for all objects Ci in the collection
3.     dist = ED(Q, Ci)
4.     if dist < r
5.         append(result, Ci)
6.     endif
7. endfor
8. return result
```

- Para seleccionar el radio se debe realizar un análisis previo de la distribución de las distancias entre dos los elementos de la colección. Se selecciona tres valores de radio:  $r_1 < r_2 < r_3$ .
- Luego realizar la búsqueda de tres objetos de consulta para cada radio. Nota: el objeto de consulta debe ser retirado de la colección antes de aplicar la búsqueda.
- Para evaluar la efectividad del resultado se debe usar la medida de Precisión ¿Cuántos de los objetos recuperados pertenecen a la misma categoría de la consulta?:

$$PR = \frac{\#ObjetosRelevantesRecuperados}{\#ObjetosRecuperados}$$

A continuación, se proporciona el cuadro que debe ser llenado por el alumno.

$PR$	$Q_{15}$	$Q_{82}$	$Q_{121}$
$r1 =$			
$r2 =$			
$r3 =$			

## P2. Búsqueda KNN

Usando los mismos objetos de consulta del ejercicio anterior, implementar y aplicar el algoritmo lineal de búsqueda de los  $k$  vecinos más cercano (KNN) variando el  $k$  entre  $\{2, 4, 8, 16, 32\}$ .

### Algorithm KnnSearch(Q, k)

```

1. result = [ ]
2. for all objects  $C_i$  in the collection
3.     dist = ED(Q,  $C_i$ )
4.     append(result, { $C_i$ , dist})
5. endfor
6. orderByDist(result)
7. return result[1:k]
```

$PR$	$Q_{15}$	$Q_{82}$	$Q_{121}$
$k = 2$			
$k = 4$			
$k = 8$			
$k = 16$			
$k = 32$			

## Preguntas:

- 1- ¿Cuál de los dos métodos de búsqueda es más fácil de implementar si tengo todos los parámetros definidos? ¿Y cuál es más eficiente?
- 2- ¿Cuál de los dos métodos de búsqueda usted usaría en un ambiente real de recuperación de la información? Sustente su respuesta.