



UNIVERSIDAD ADOLFO IBÁÑEZ

RECONOCIMIENTO DE PATRONES EN IMÁGENES TICS 585

FACULTAD DE INGENIERÍA Y CIENCIAS
UNIVERSIDAD ADOLFO IBÁÑEZ

SEGUNDO SEMESTRE 2021

PROFESOR: MIGUEL CARRASCO

SELECCIÓN DE CARACTERÍSTICAS

■ Discriminante de Fisher

Discriminante de Fisher

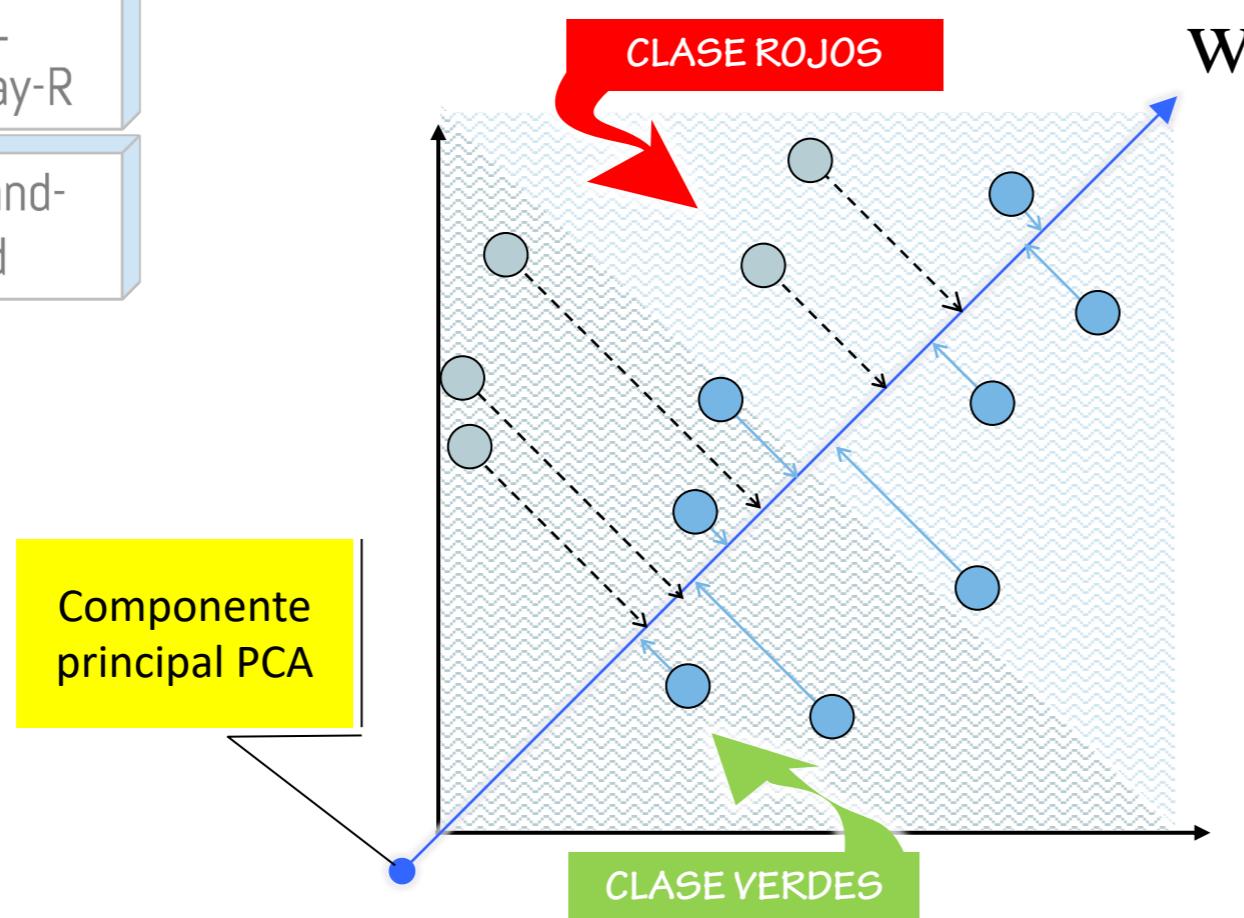
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

■ Preliminares

Como vimos anteriormente, el algoritmo **PCA** realiza una **combinación lineal** de las características para representar el problema **maximizando la varianza**.



Pregunta

¿Podemos asumir que la combinación de PCA es útil en separar las clases?

Respuesta

Una combinación que maximiza la varianza **NO** siempre mejora la separación. Inclusive **puede degradar la clasificación** del problema.



■ Discriminante de Fisher

Discriminante de Fisher

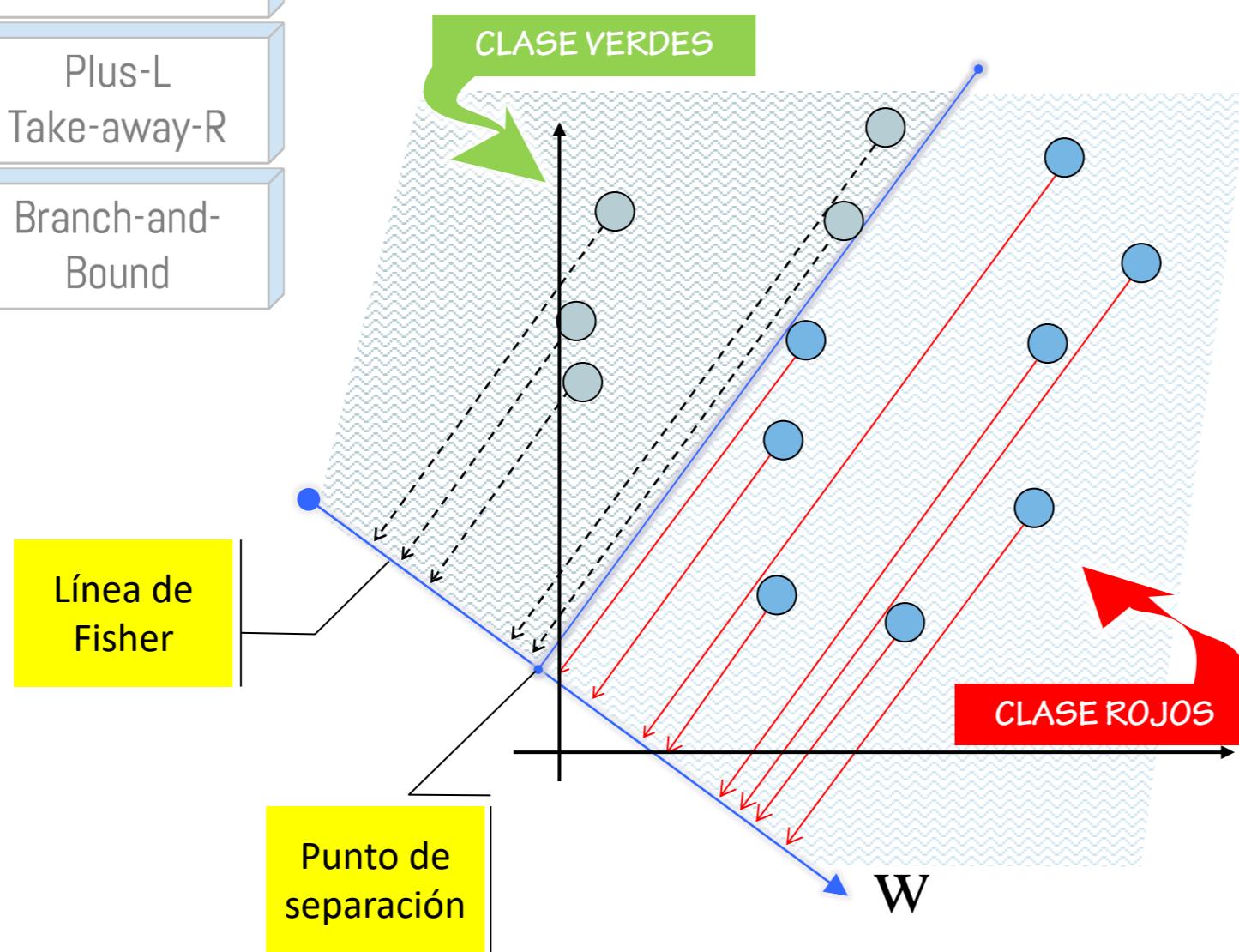
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

■ Objetivo:

Buscar una recta óptima tal que las varianzas **entre clases** sea maximizada y la varianza **en cada clase** sea minimizada



Pregunta

¿Podemos saber si las características que empleamos son buenas para separar las clases?

Respuesta

Si, a través del discriminante de Fisher podemos calcular un clasificador lineal óptimo para clasificar una población



■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

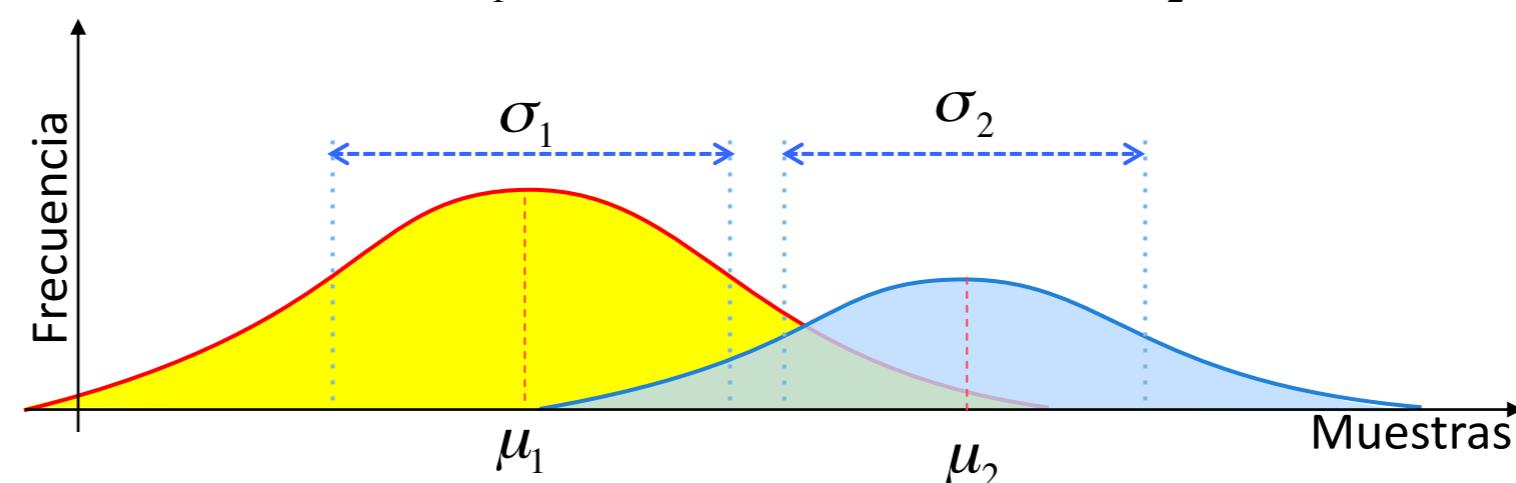
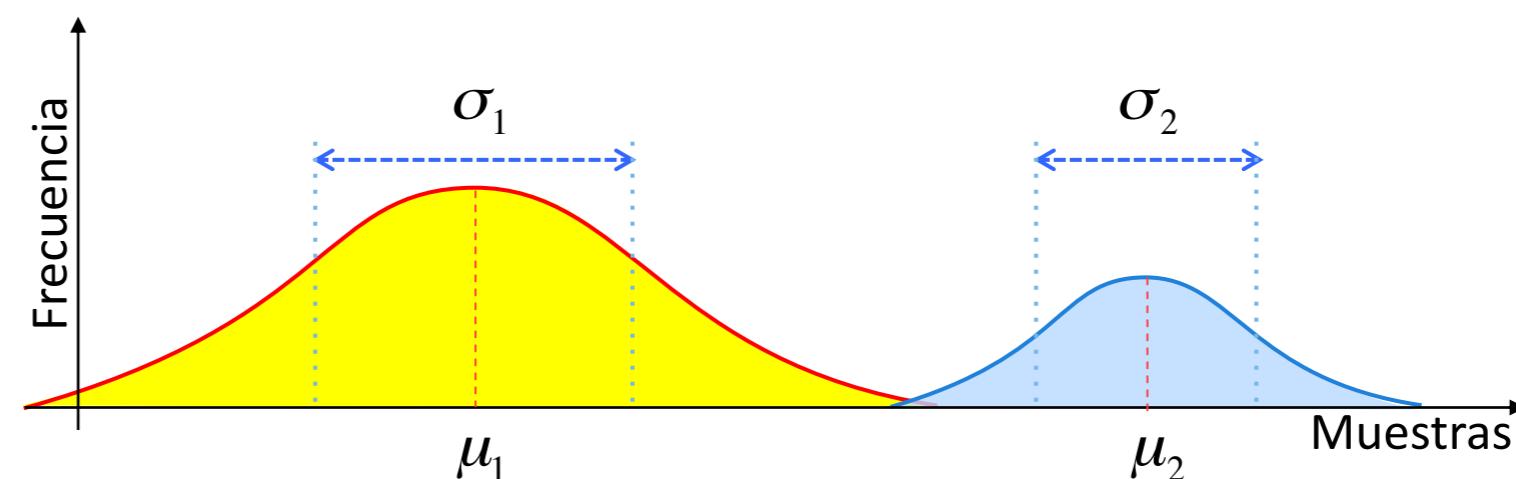
Plus-L
Take-away-R

Branch-and-Bound

■ Problema:

Suponga dos distribuciones con medias μ_1 y μ_2 , y desviaciones estándar σ_1 y σ_2 .

¿Cómo podemos describir cual de las dos tiene una mejor separación en función de las medias y las desviaciones estándar?



■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

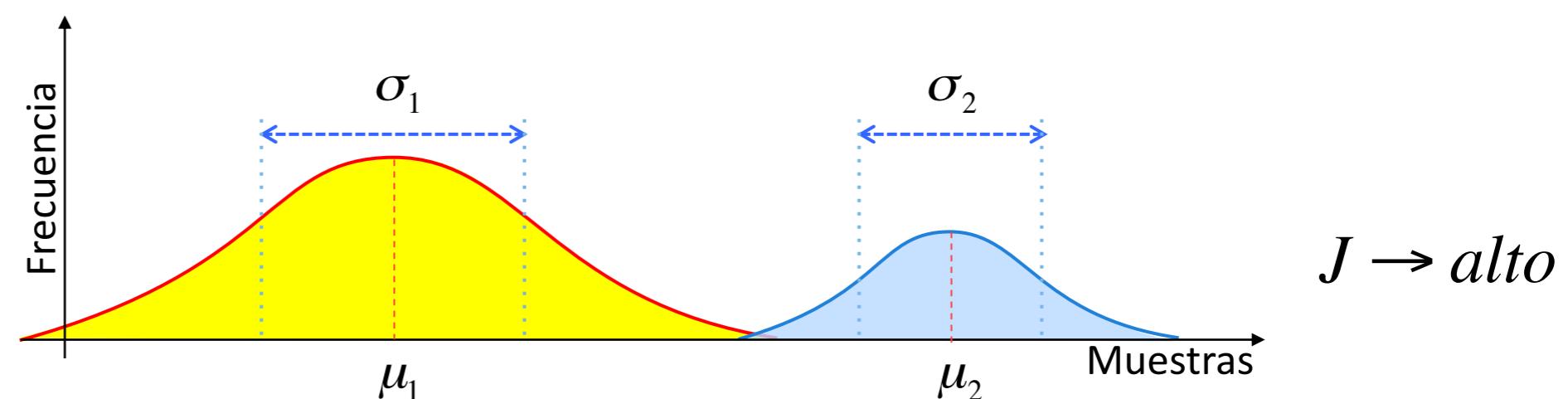
Plus-L
Take-away-R

Branch-and-Bound

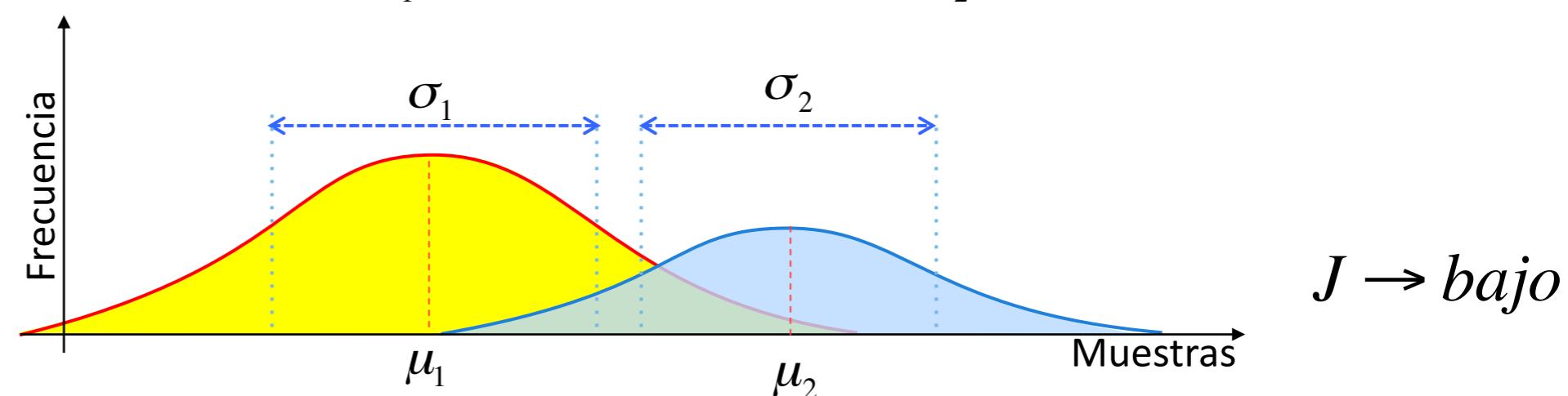
- Índice J de Fisher para **dos** distribuciones:

$$J = \frac{(\mu_2 - \mu_1)^2}{\sigma_1^2 + \sigma_2^2}$$

Si encontramos un J con valor alto garantizamos que las clases están bien separadas.



$J \rightarrow \text{alto}$



$J \rightarrow \text{bajo}$

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

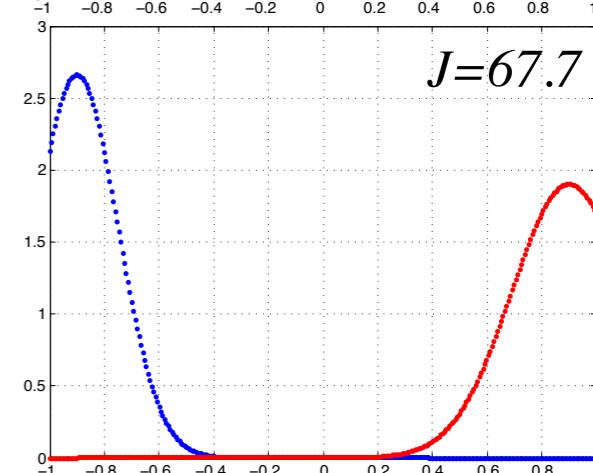
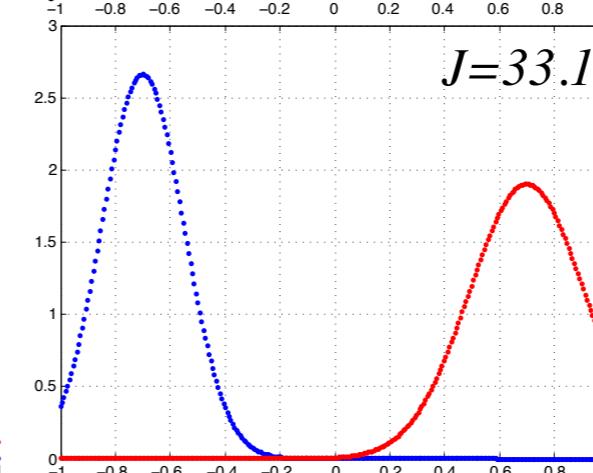
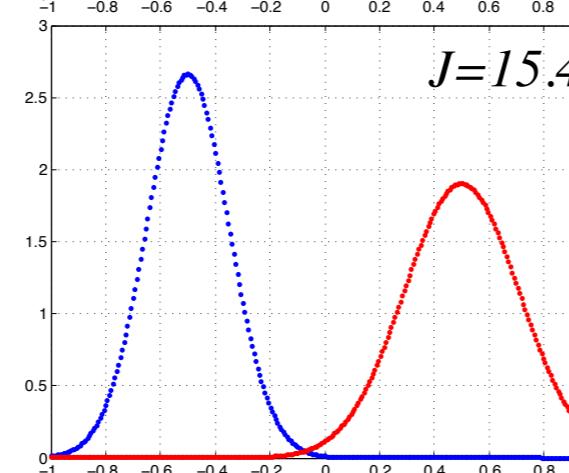
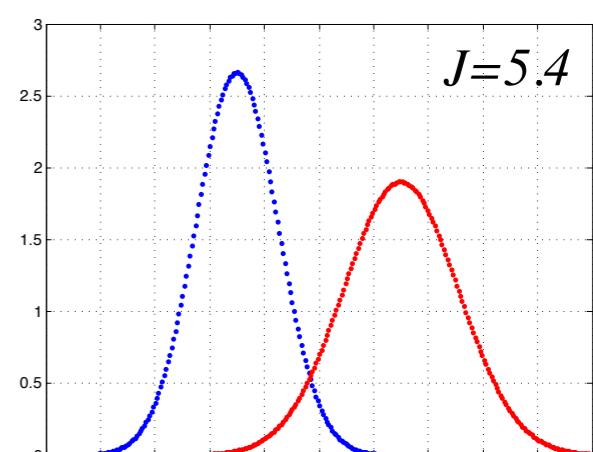
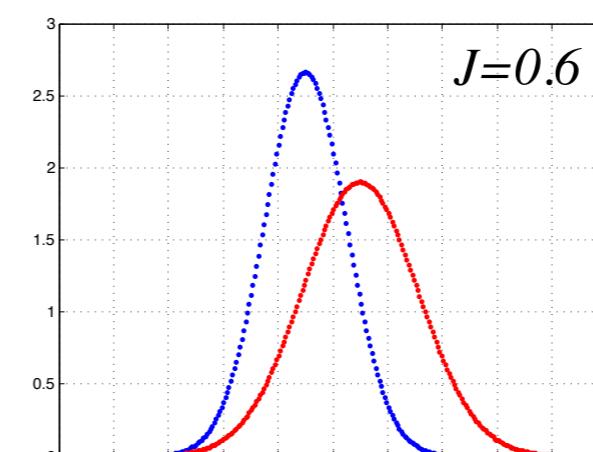
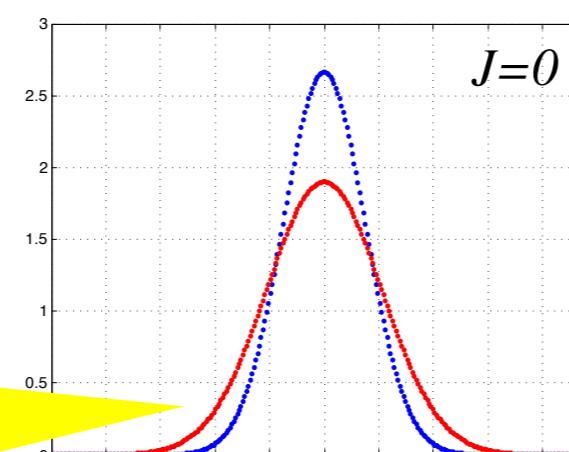
Branch-and-Bound

¿Cuales son la medias de cada distribución?
Observe las distribuciones

- Índice J de Fisher para **dos** distribuciones:

$$J = \frac{(\mu_2 - \mu_1)^2}{\sigma_1^2 + \sigma_2^2}$$

Todas las distribuciones en azul tiene $s=0.15$ y en rojo $s=0.21$



■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- **Ejercicio.** Utilice el siguiente código para inventar datos y luego calcule el índice J de Fisher. Pruebe distintos valores para las dos distribuciones

```
def datg(N, s, mu):  
  
    x = np.linspace(-1,1, N)  
    fx = (1/(s*np.sqrt(2*pi)))*  
    np.exp(-0.5*((x-mu)/s)**2)  
    return x,fx  
  
-----  
  
X1,Y1 = datg(300,0.15,-0.1)  
X2,Y2 = datg(300,0.21,0.1)  
J = fisher(X1,Y1,X2,Y2)  
  
print(f' J:{J}')  
  
plt.figure()  
plt.plot(X1,Y1, color='blue')  
plt.plot(X2,Y2, color='red')  
plt.show()
```

$$J = \frac{(\mu_2 - \mu_1)^2}{\sigma_1^2 + \sigma_2^2}$$

- Implemente la función que calcule el valor de J.

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- **Ejercicio.** Utilice el siguiente código para inventar datos y luego calcule el índice J de Fisher. Pruebe distintos valores para las dos distribuciones

```
def datg(N, s, mu):  
  
    x = np.linspace(-1,1, N)  
    fx = (1/(s*np.sqrt(2*pi)))*  
    np.exp(-0.5*((x-mu)/s)**2)  
    return x,fx  
  
-----  
  
X1,Y1 = datg(300,0.15,-0.1)  
X2,Y2 = datg(300,0.21,0.1)  
J = fisher(X1,Y1,X2,Y2)  
  
print(f' J:{J}')  
  
plt.figure()  
plt.plot(X1,Y1, color='blue')  
plt.plot(X2,Y2, color='red')  
plt.show()
```

```
def fisher(X1, Y1, X2, Y2):  
  
n= len(X1)  
  
# normalizamos la distribucion  
Y1n = Y1/np.sum(Y1)  
Y2n = Y2/np.sum(Y1)  
  
# media mu  
m1 = np.sum(Y1n*X1)  
m2 = np.sum(Y2n*X2)  
  
# desviacion estandar  
s1 = np.sqrt(sum(((X1-m1)**2)*Y1n))  
s2 = np.sqrt(sum(((X2-m2)**2)*Y2n))  
  
# Fisher  
J = ((m1-m2)**2)/ (s1**2+s2**2)  
return J
```

■ Discriminante de Fisher

Discriminante de Fisher

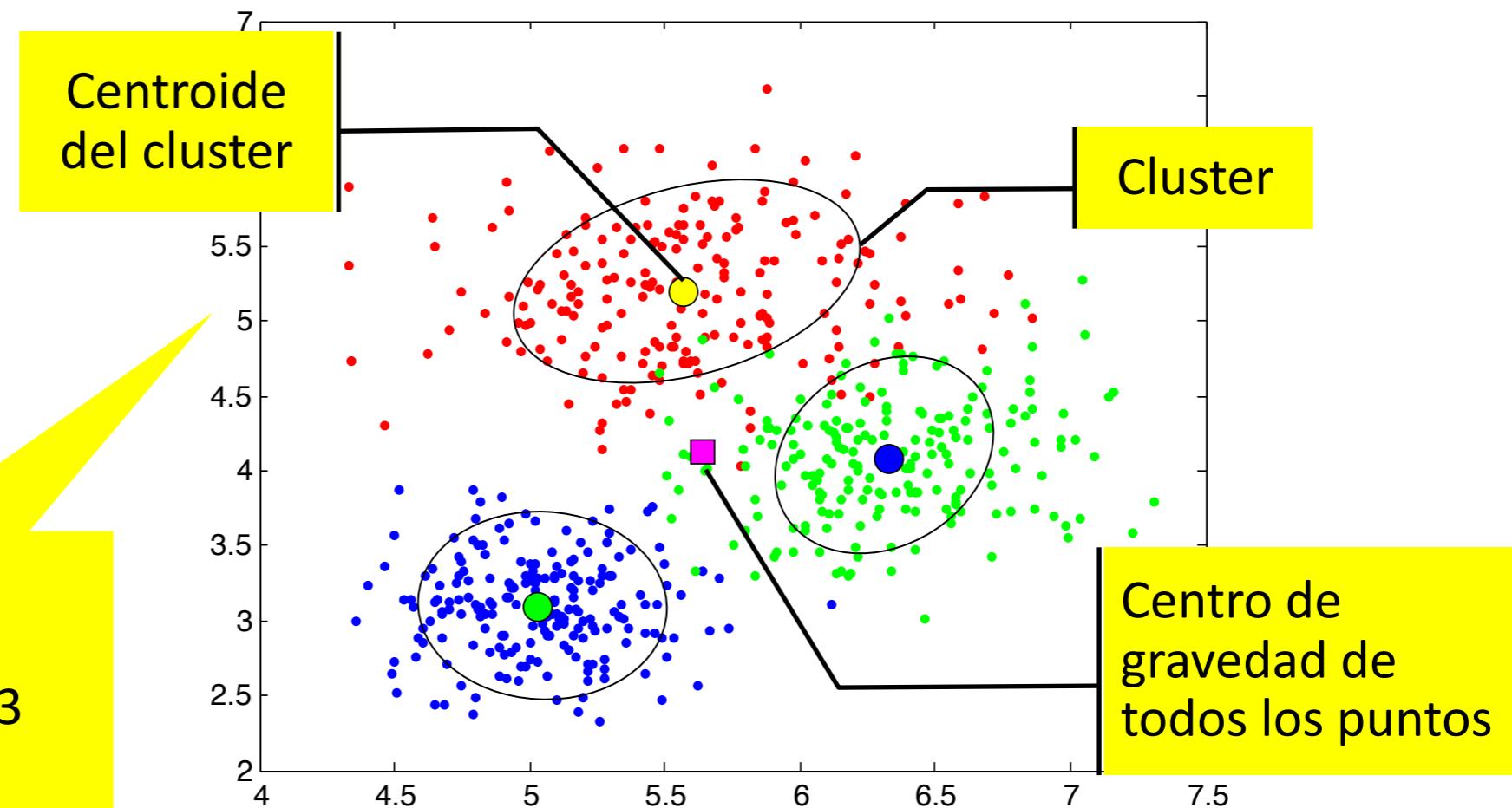
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

En el ejemplo tenemos 2 características y 3 clases o clusters

- Si tenemos más de una característica debemos calcular la covarianza **interclase** e **intraclase**. Para ello debemos emplear una ecuación que considera la distancia entre el centro de gravedad, y la relación entre cada cluster.



■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

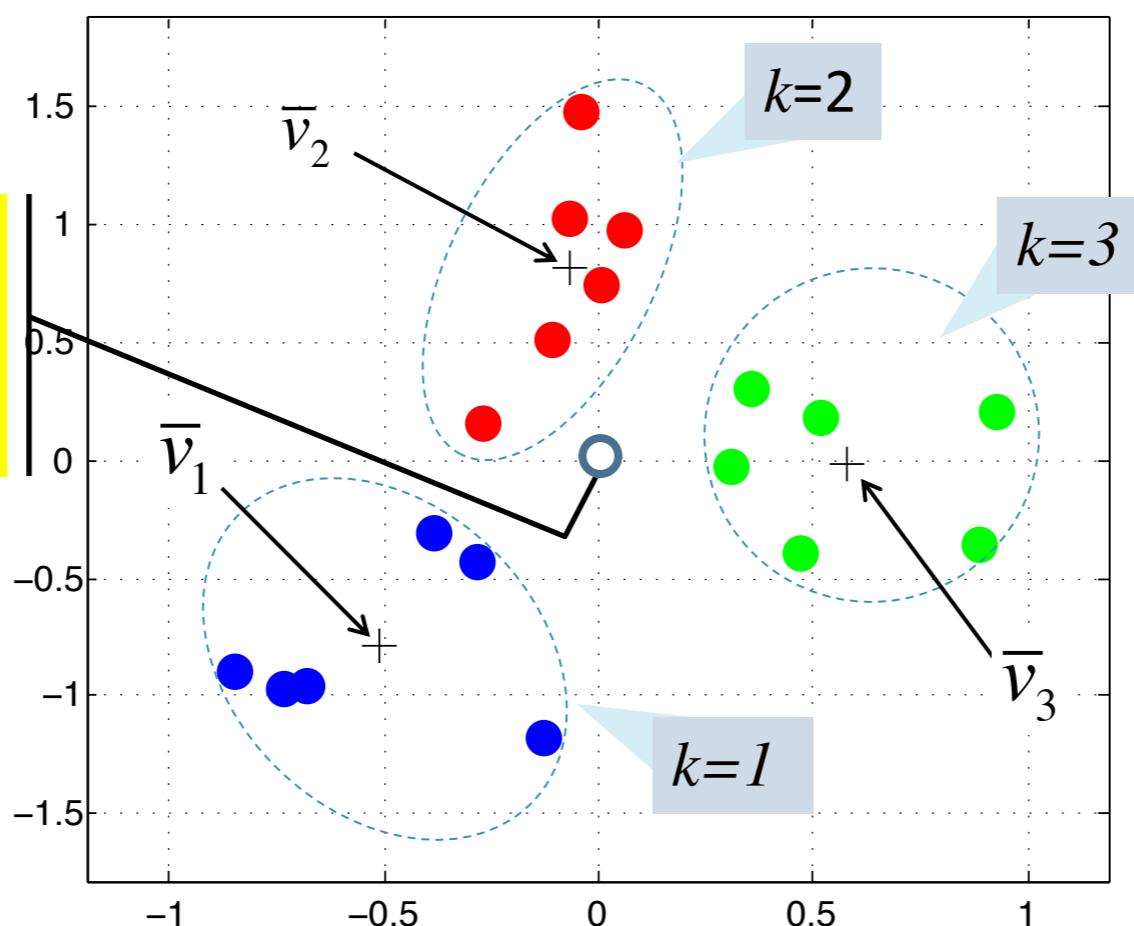
Plus-L Take-away-R

Branch-and-Bound

Centro de gravedad de todos los puntos \bar{v}

- *Interclase*: Mide la dispersión entre todos los clusters. Los parámetros corresponden a (1) las medias de los centroides, (2) la probabilidad a priori de la clase p_k y (3) el centro de gravedad

$$C_b = \sum_{k=1}^n p_k \cdot (\bar{v}_k - \bar{v}) \cdot (\bar{v}_k - \bar{v})^T$$



X	Y
-0.13	-1.18
-0.73	-0.97
-0.85	-0.90
-0.68	-0.96
-0.38	-0.31
-0.28	-0.43

X	Y
0.92	0.16
0.30	-0.03
0.35	0.29
0.46	-0.40
0.51	0.17
0.88	-0.36

Datos de ejemplo

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- **Interclase:** Mide la dispersión entre todos los clusters. Los parámetros corresponden a (1) las medias de los centroides, (2) la probabilidad a priori de la clase p_k y (3) el centro de gravedad

$$C_b = \sum_{k=1}^n p_k \cdot (\bar{v}_k - \bar{v}) \cdot (\bar{v}_k - \bar{v})^T$$

Probabilidad a priori de las clases (ejemplo)

$$p = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

Esto significa que 33 % de los datos son la clase₁, 33% de la clase₂ y un 33% de la clase₃

Centroides

$$\bar{v} = \begin{bmatrix} 0.0000 & 0.000 \end{bmatrix}$$

$$\bar{v}_1 = \begin{bmatrix} -0.5089 & -0.7298 \end{bmatrix}$$

$$\bar{v}_2 = \begin{bmatrix} -0.0706 & 0.8106 \end{bmatrix}$$

$$\bar{v}_3 = \begin{bmatrix} 0.5794 & -0.0178 \end{bmatrix}$$

$$p_1 \cdot (\bar{v}_1 - \bar{v}) \cdot (\bar{v}_1 - \bar{v})^T = \frac{1}{3} \cdot \left(\begin{bmatrix} -0.5089 \\ -0.7298 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right) \cdot \left(\begin{bmatrix} -0.5089 & -0.7298 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix} \right)$$

$$p_2 \cdot (\bar{v}_2 - \bar{v}) \cdot (\bar{v}_2 - \bar{v})^T = \frac{1}{3} \cdot \left(\begin{bmatrix} -0.0706 \\ 0.8106 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right) \cdot \left(\begin{bmatrix} -0.0706 & 0.8106 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix} \right)$$

$$p_3 \cdot (\bar{v}_3 - \bar{v}) \cdot (\bar{v}_3 - \bar{v})^T = \frac{1}{3} \cdot \left(\begin{bmatrix} 0.5794 \\ -0.0178 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right) \cdot \left(\begin{bmatrix} 0.5794 & -0.0178 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix} \right)$$

■ Discriminante de Fisher

Discriminante de Fisher

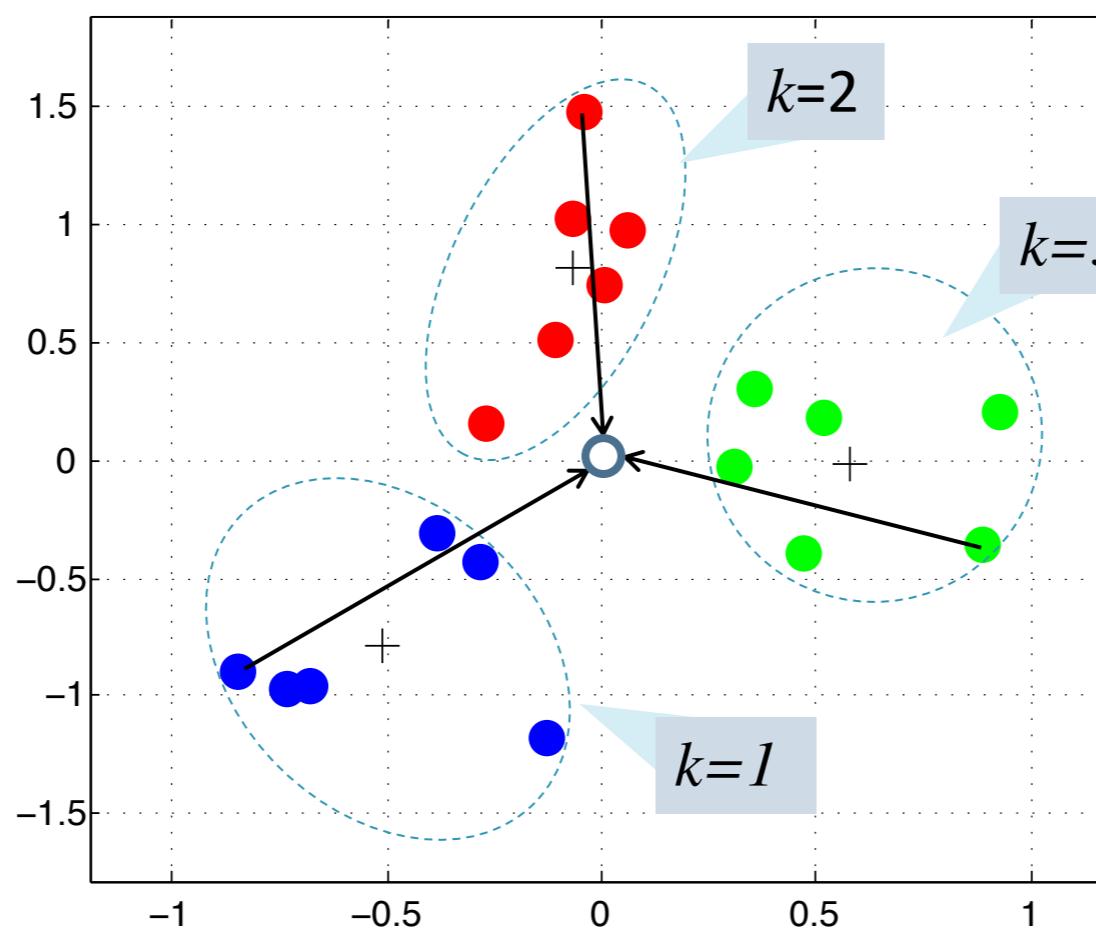
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- **Interclase:** Mide la dispersión entre todos los clusters. Los parámetros corresponden a (1) las medias de los centroides, (2) la probabilidad a priori de la clase p_k y (3) el centro de gravedad

$$C_b = \sum_{k=1}^n p_k \cdot (\bar{v}_k - \bar{v}) \cdot (\bar{v}_k - \bar{v})^T$$



$$C_b = \begin{bmatrix} 0.1999 & 0.1120 \\ 0.1120 & 0.4286 \end{bmatrix}$$

Covarianza entre clases

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

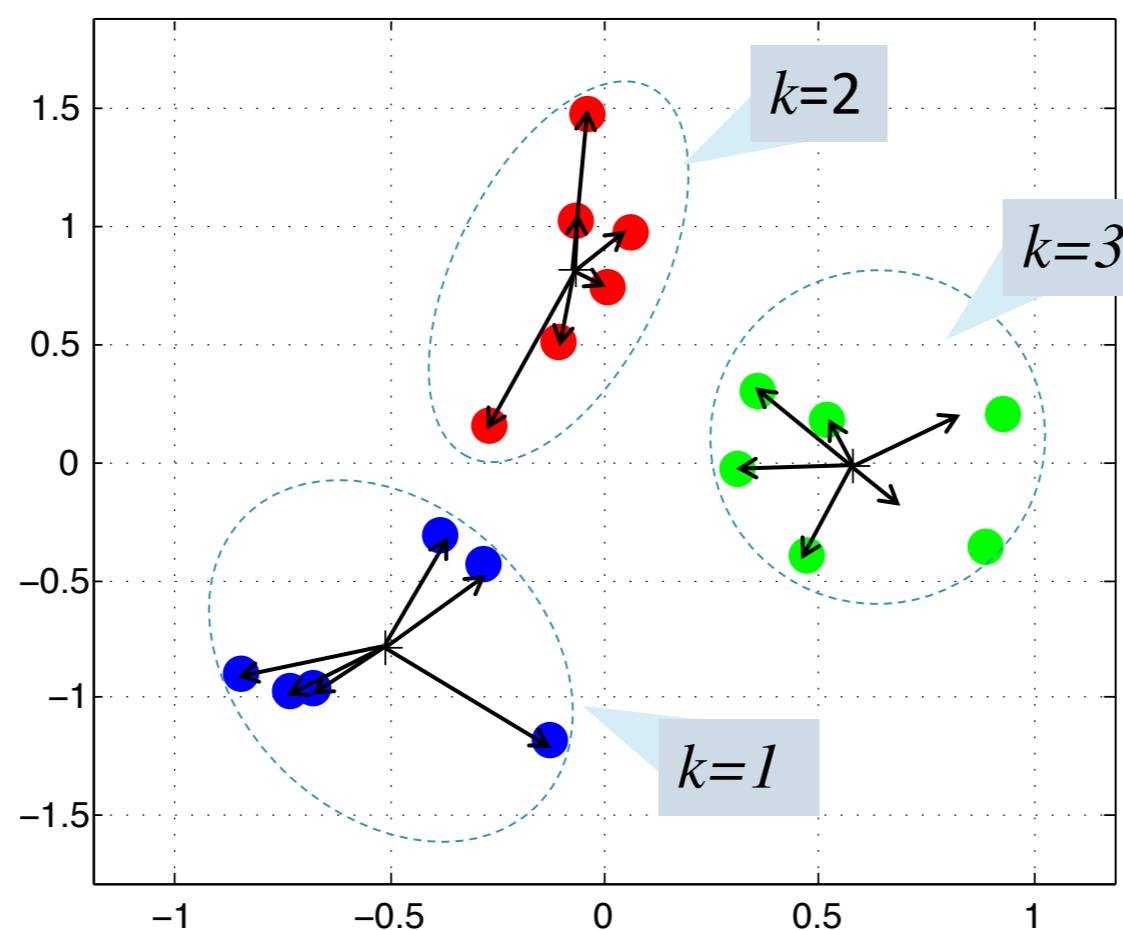
- *Intraclass*: Mide la covarianza o dispersión en cada clase (o cluster)

$$C_w = \sum_{k=1}^n p_k \cdot \text{cov}(G_k)$$

Calculamos la covarianza de cada grupo.

$$C_w = p_1 \cdot \text{cov}(G_1) + p_2 \cdot \text{cov}(G_2) + p_3 \cdot \text{cov}(G_3)$$

$$C_w = \begin{bmatrix} 0.0552 & 0.0146 \\ 0.0146 & 0.1393 \end{bmatrix}$$



La matriz covarianza depende de la dimensión del problema, esto es, según el número de características

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- A través discriminante de Fisher podemos evaluar la separación entre las clases.
- Si el valor J es alto, quiere decir que las clases están bien separadas.

$$J = \text{trace}\left(\left[C_w^{-1} \cdot C_b\right]\right)$$

Evaluamos el índice J para evaluar la separación de las clases.

$$C_b = \begin{bmatrix} 0.1999 & 0.1120 \\ 0.1120 & 0.4286 \end{bmatrix}$$

$$C_w = \begin{bmatrix} 0.0552 & 0.0146 \\ 0.0146 & 0.1393 \end{bmatrix}$$

$$J = \text{traza}\left(C_w^{-1} \cdot C_b\right) = \left(\begin{bmatrix} 0.0552 & 0.0146 \\ 0.0146 & 0.1393 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 0.1999 & 0.1120 \\ 0.1120 & 0.4286 \end{bmatrix} \right)$$

$$J = 6.4491$$

■ Discriminante de Fisher

Discriminante de Fisher

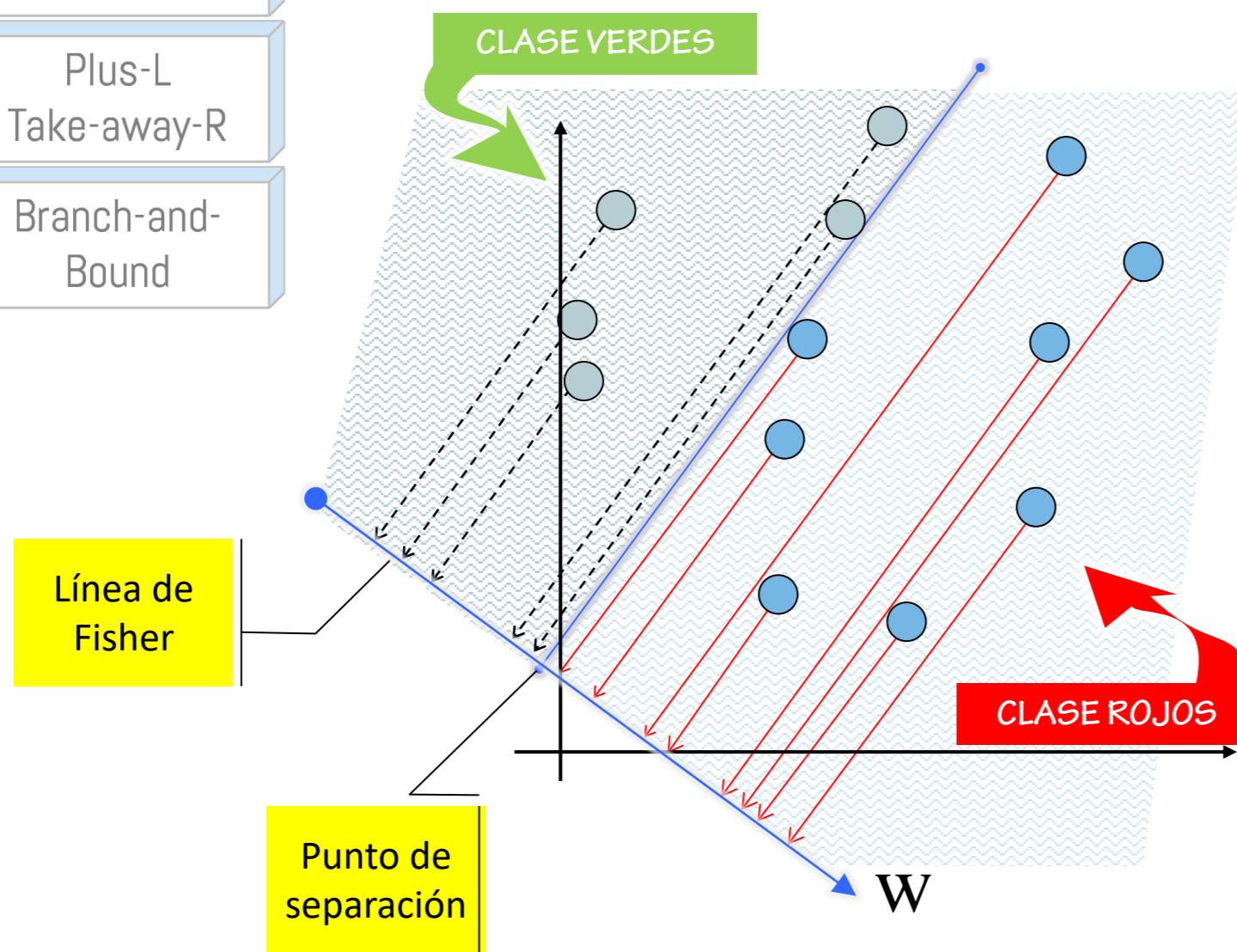
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- Otra característica es podemos calcular la línea de separación entre *dos clases* a través de la siguiente ecuación

$$\mathbf{w} = C_w^{-1} \cdot (\bar{\nu}_i - \bar{\nu}_j)$$



Centro de masa
de la i-ésima
clase

Centro de masa
de la j-ésima
clase

■ Discriminante de Fisher

Discriminante de Fisher

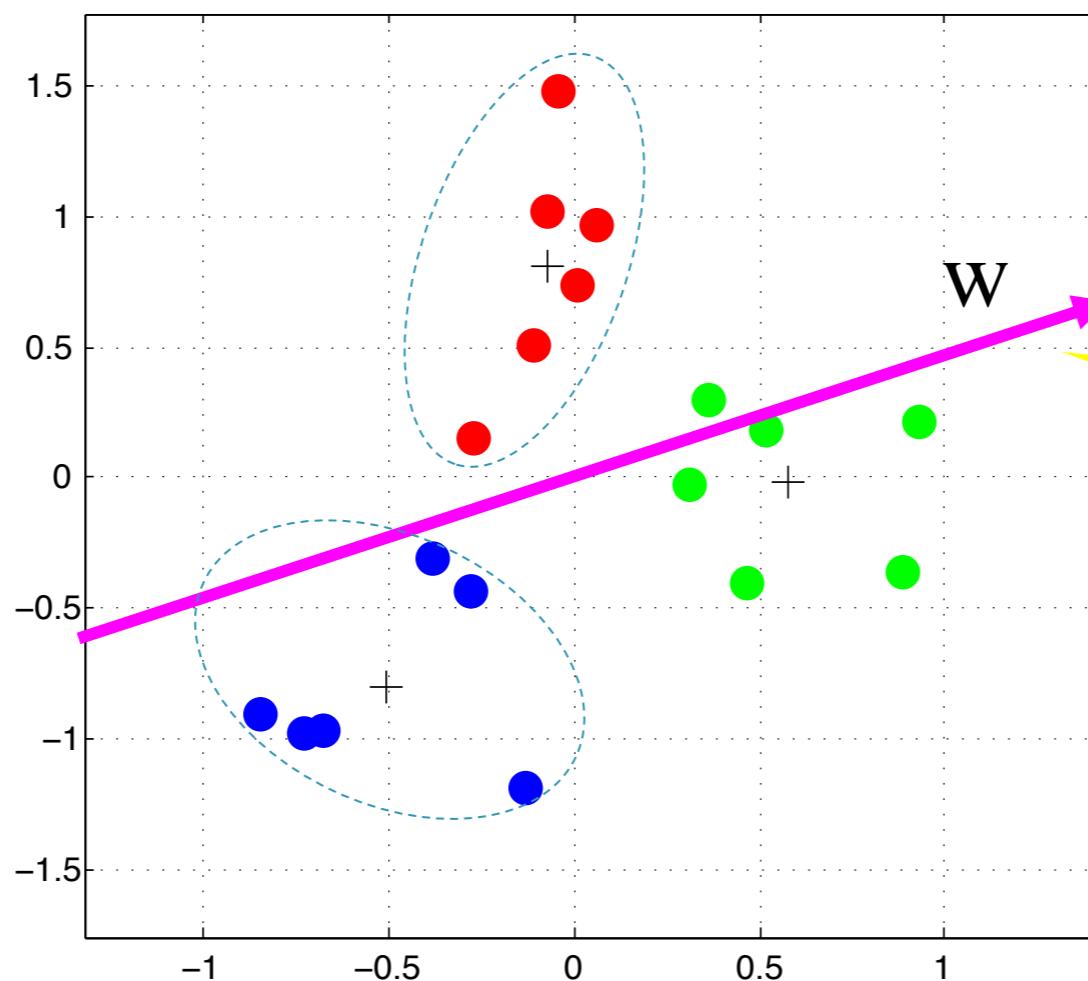
Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- Otra característica es podemos calcular la línea de separación entre *dos clases* a través de la siguiente ecuación

$$\mathbf{W} = C_w^{-1} \cdot (\bar{\mathbf{v}}_i - \bar{\mathbf{v}}_j)$$



OBJETIVO:
Separemos los clusters 1 y 2.

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

- Otra característica es podemos calcular la línea de separación entre *dos clases* a través de la siguiente ecuación

$$\mathbf{W} = C_w^{-1} \cdot (\bar{\mathbf{v}}_i - \bar{\mathbf{v}}_j)$$

$$\rightarrow C_w^{-1} = \begin{bmatrix} 18.6232 & -1.9552 \\ -1.9552 & 7.3827 \end{bmatrix}$$

Centroides

$$\begin{aligned}\bar{\mathbf{v}} &= [0.0000 \quad 0.0000] \\ \bar{\mathbf{v}}_1 &= [-0.5089 \quad -0.7298] \\ \bar{\mathbf{v}}_2 &= [-0.0706 \quad 0.8106] \\ \bar{\mathbf{v}}_3 &= [0.5794 \quad -0.0178]\end{aligned}$$

$$\mathbf{w}_{1-2} = \begin{bmatrix} 18.6232 & -1.9552 \\ -1.9552 & 7.3827 \end{bmatrix} \cdot \left(\begin{bmatrix} -0.5089 \\ -0.0706 \end{bmatrix} - \begin{bmatrix} -0.7298 \\ 0.8106 \end{bmatrix} \right)$$

$$\mathbf{w}_{1-2} = \begin{bmatrix} -5.0283 \\ -10.9798 \end{bmatrix}$$

Centro de Masa del cluster 1

Centro de Masa del cluster 2

Este resultado corresponde a la pendiente de la recta 1-2

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

Plus-L
Take-away-R

Branch-and-Bound

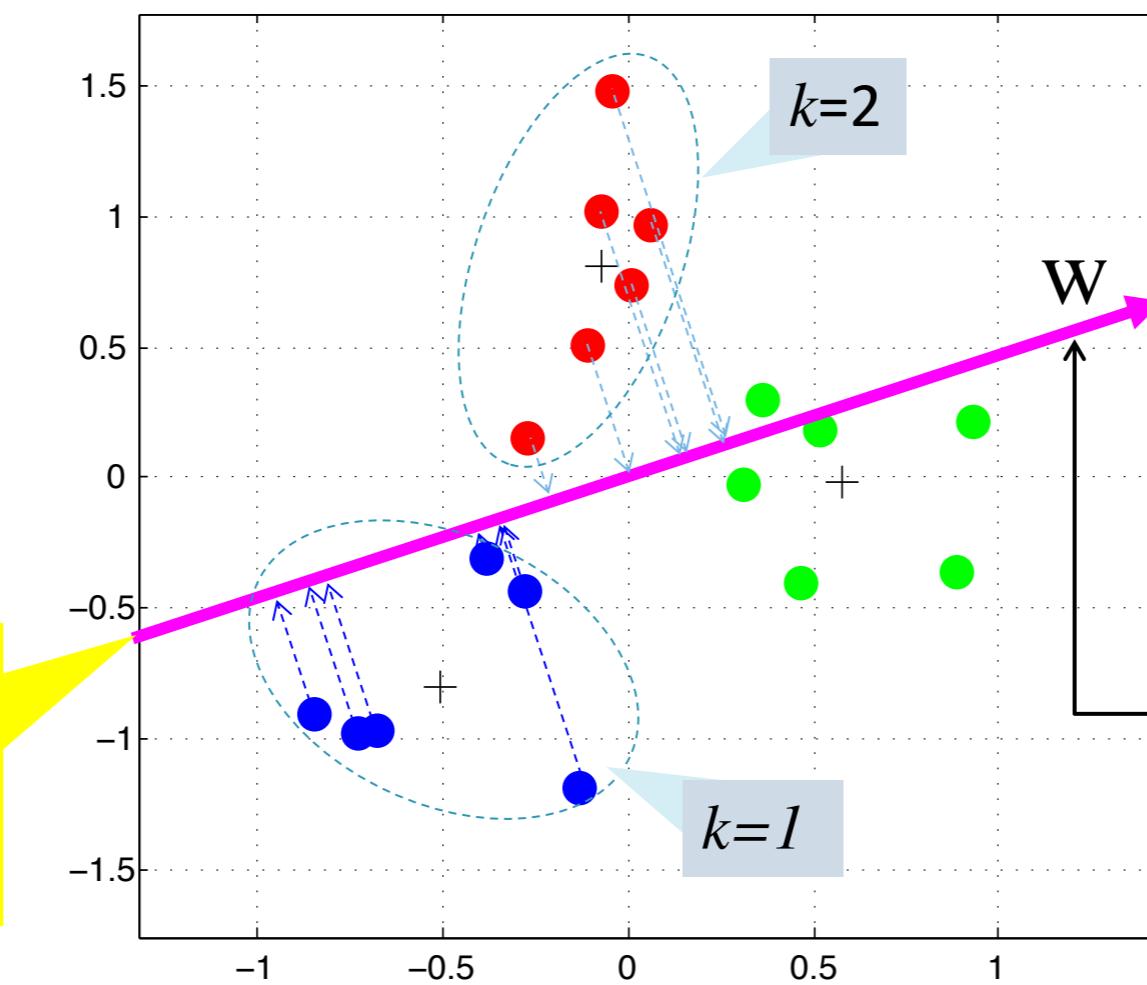
- Otra característica es podemos calcular la línea de separación entre *dos clases* a través de la siguiente ecuación

$$\mathbf{W} = C_w^{-1} \cdot (\bar{\mathbf{v}}_i - \bar{\mathbf{v}}_j)$$

Este resultado corresponde a la pendiente de la recta entre el cluster 1 y el cluster 2

$$\mathbf{w}_{1-2} = \begin{bmatrix} -5.0283 \\ -10.9798 \end{bmatrix}$$

Como los datos están centrados, no existe la constante en la ecuación de la recta



Ecuación de la recta

$$f(x) = \frac{-5.0283}{-10.9798} \cdot x$$

■ Discriminante de Fisher

Discriminante de Fisher

Búsqueda Exhaustiva

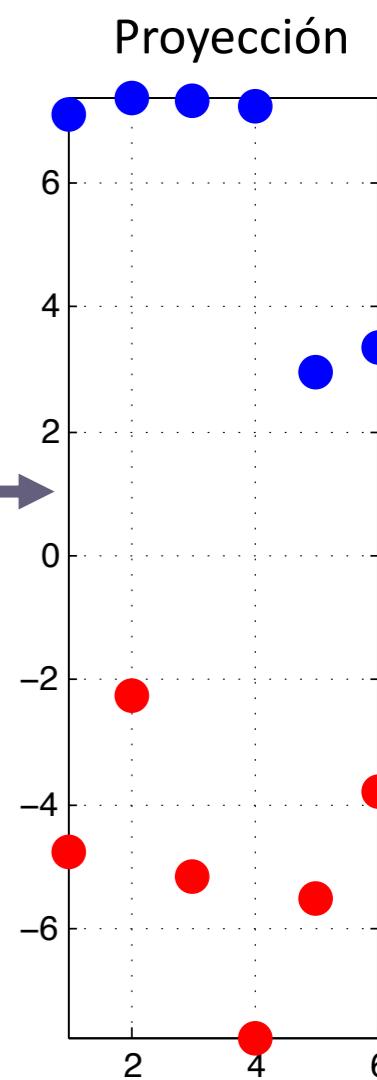
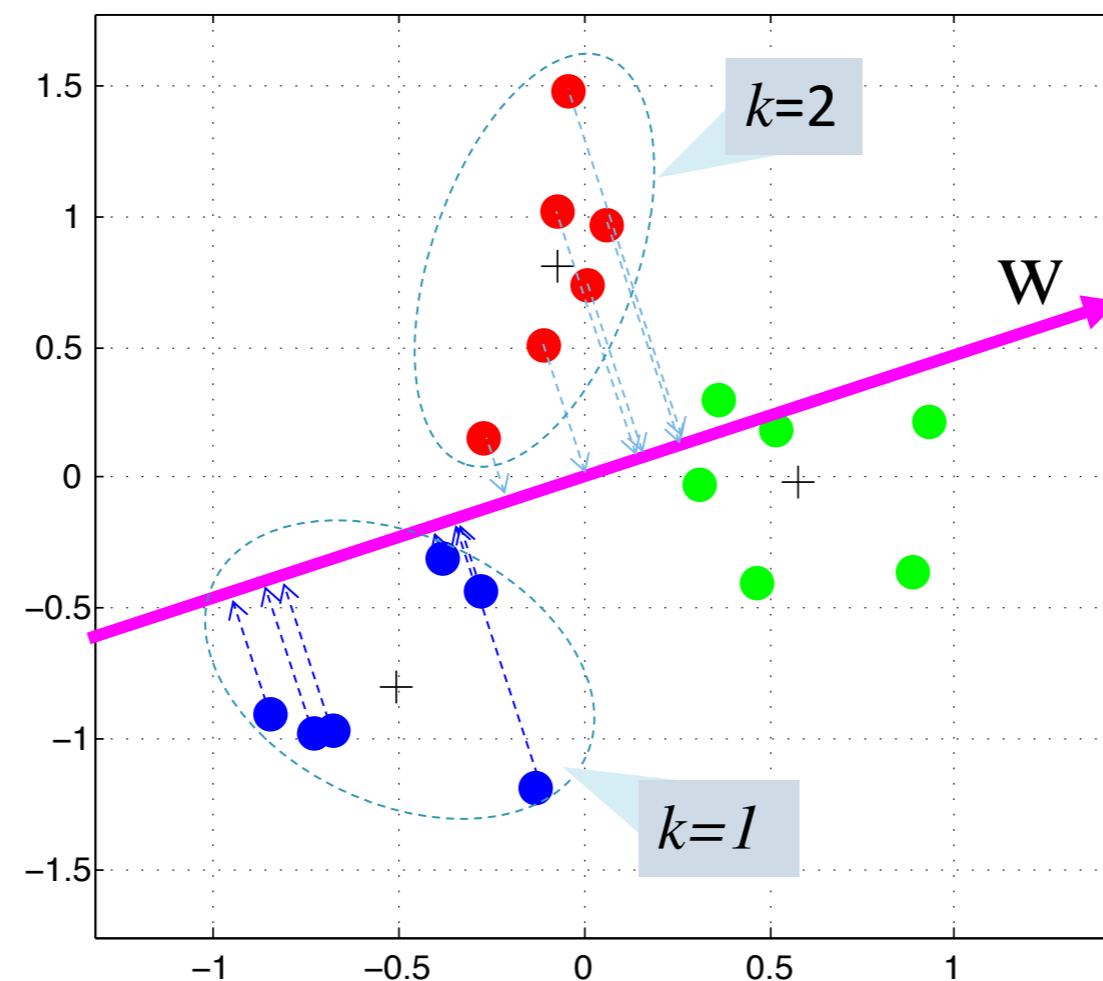
Plus-L
Take-away-R

Branch-and-Bound

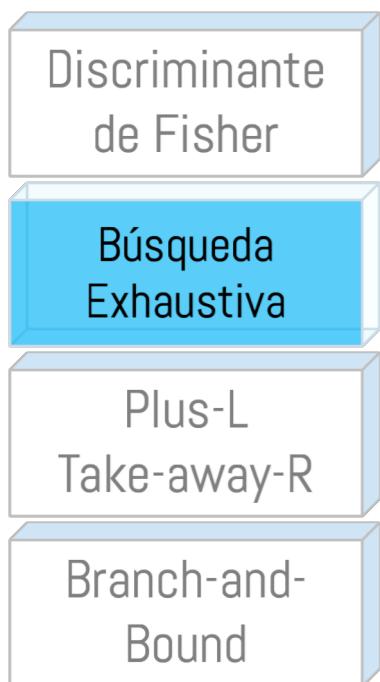
- La última etapa corresponde a la proyección de los datos para su separación. En el ejemplo, suponemos que G_1 y G_2 corresponden a los datos desplazados.

$$Y_1 = W_{1-2} \cdot (G_1)$$

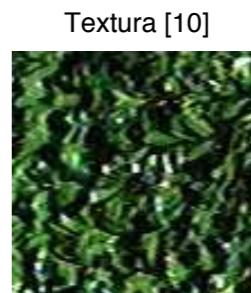
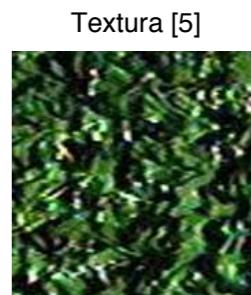
$$Y_2 = W_{1-2} \cdot (G_2)$$



Búsqueda Exhaustiva

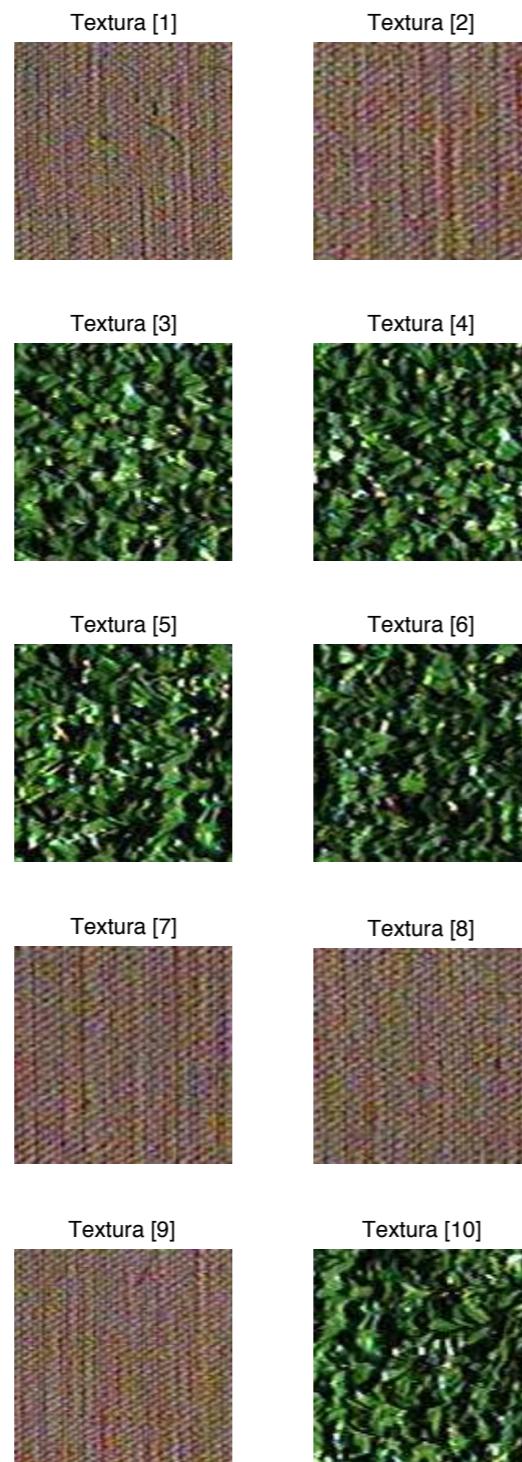
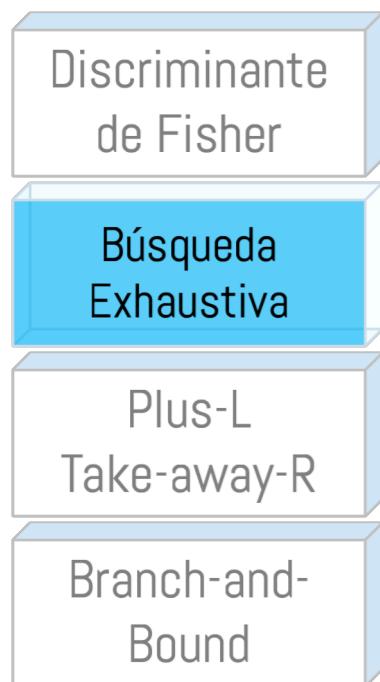


- Suponga las siguientes características para un conjunto de texturas ¿Cuales características ocuparía usted?

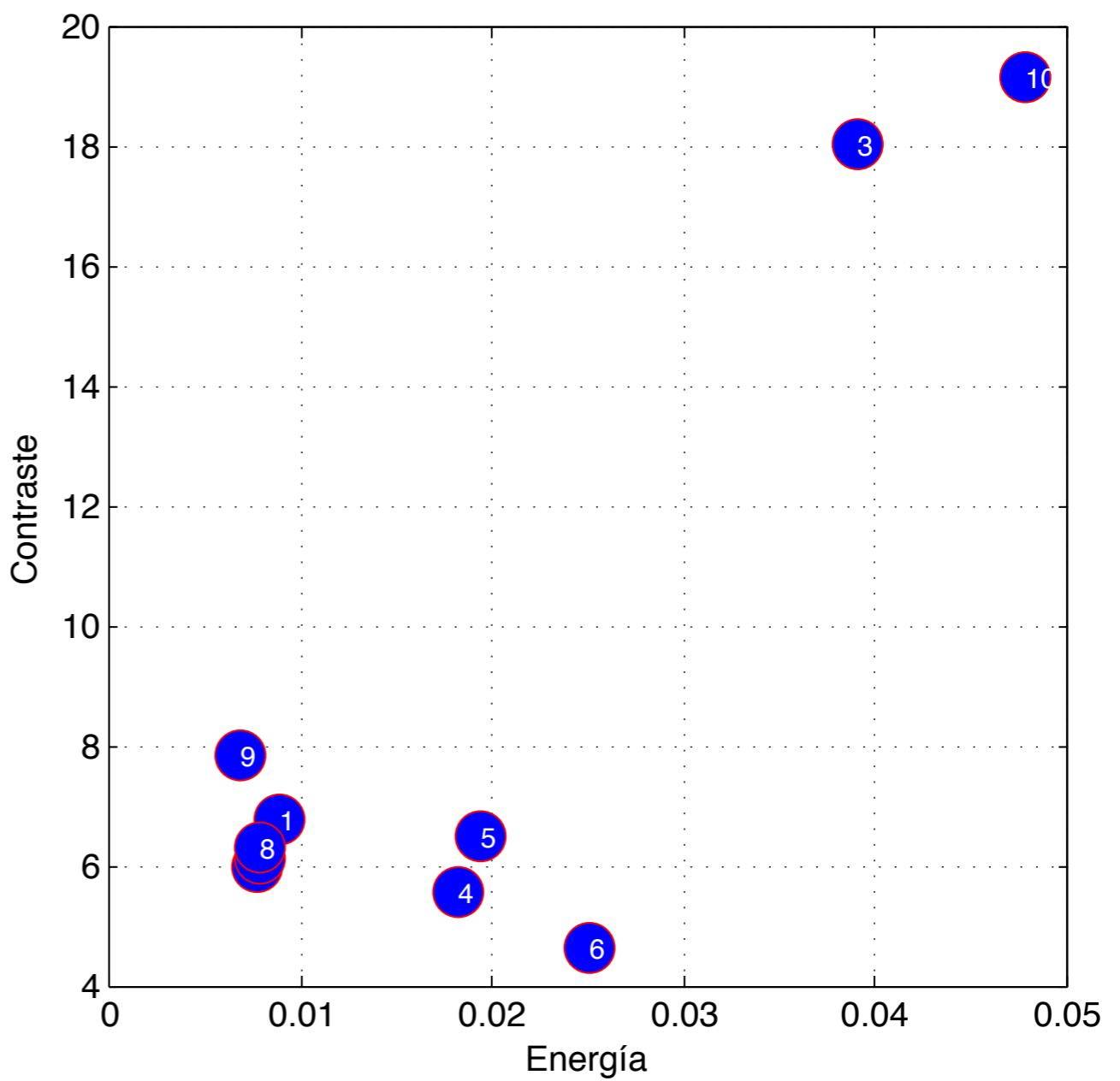


	Energía	Contraste	Correlación	Homogeneidad	Entropía	Momento
Textura [1]:	0.0088	6.7759	0.7531	0.3581	7.2402	0.4369
Textura [2]:	0.0077	5.9886	0.8399	0.3747	7.4072	0.4510
Textura [3]:	0.0390	18.0805	0.7537	0.4648	6.2451	0.5229
Textura [4]:	0.0182	5.5507	0.9141	0.4705	7.0282	0.5280
Textura [5]:	0.0193	6.4887	0.9035	0.4709	7.0618	0.5283
Textura [6]:	0.0251	4.6442	0.9151	0.5107	6.6688	0.5616
Textura [7]:	0.0079	6.1436	0.8235	0.3722	7.3654	0.4488
Textura [8]:	0.0078	6.3304	0.8215	0.3712	7.3984	0.4480
Textura [9]:	0.0069	7.8256	0.7868	0.3376	7.5614	0.4206
Textura [10]:	0.0478	19.1941	0.7418	0.4718	6.1327	0.5278

Búsqueda Exhaustiva



- ¿La característica **Energía v/s Contraste** es útil en separar las clases?



Búsqueda Exhaustiva

Discriminante
de Fisher

Búsqueda
Exhaustiva

Plus-L
Take-away-R

Branch-and-
Bound

- La búsqueda exhaustiva supone buscar todos los grupos de k características dentro de un grupo de n características ($k < n$). Esto último es calculado con el **coeficiente binomial**

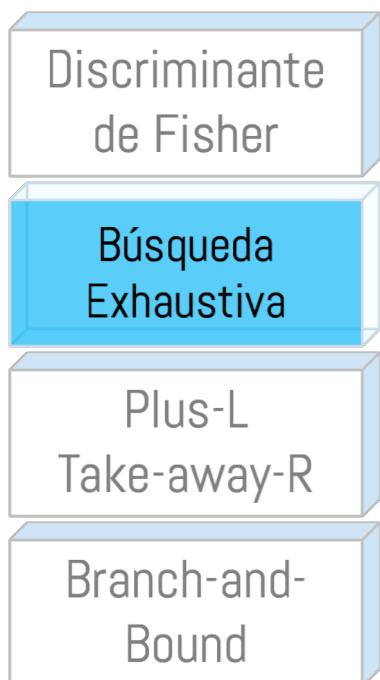
$$\binom{n}{k} = \frac{n!}{(n-k)! \cdot k!}$$

- Ejemplo:** Cuantos grupos de dos características existen en el conjunto anterior.

$$\binom{6}{2} = \frac{6!}{4! \cdot 2!} = \frac{1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6}{1 \cdot 2 \cdot 3 \cdot 4 \cdot 1 \cdot 2} = 15$$

	Energía	Contraste	Correlación	Homogeneidad	Entropía	Momento
Textura [1]:	0.0088	6.7759	0.7531	0.3581	7.2402	0.4369
Textura [2]:	0.0077	5.9886	0.8399	0.3747	7.4072	0.4510
Textura [3]:	0.0390	18.0805	0.7537	0.4648	6.2451	0.5229
Textura [4]:	0.0182	5.5507	0.9141	0.4705	7.0282	0.5280
Textura [5]:	0.0193	6.4887	0.9035	0.4709	7.0618	0.5283
Textura [6]:	0.0251	4.6442	0.9151	0.5107	6.6688	0.5616
Textura [7]:	0.0079	6.1436	0.8235	0.3722	7.3654	0.4488
Textura [8]:	0.0078	6.3304	0.8215	0.3712	7.3984	0.4480
Textura [9]:	0.0069	7.8256	0.7868	0.3376	7.5614	0.4206
Textura [10]:	0.0478	19.1941	0.7418	0.4718	6.1327	0.5278

Búsqueda Exhaustiva



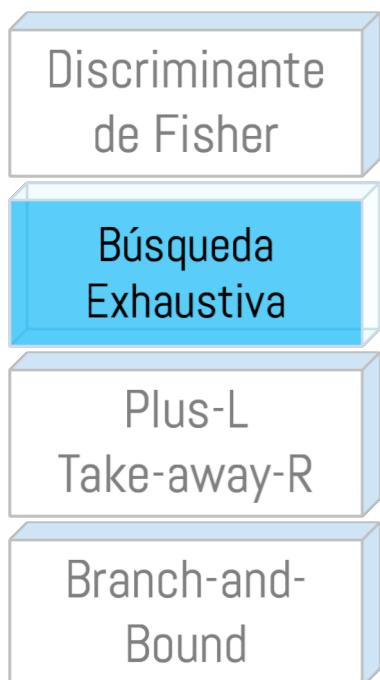
- ¿Cuales son los pares de características que maximizan el índice de Fisher?

		Energía	Contraste	Correlación	Homogeneidad	Entropía	Momento
Textura [1]:		0.0088	6.7759	0.7531	0.3581	7.2402	0.4369
Textura [2]:		0.0077	5.9886	0.8399	0.3747	7.4072	0.4510
Textura [3]:		0.0390	18.081	0.7537	0.4648	6.2451	0.5229
Textura [4]:		0.0182	5.5507	0.9141	0.4705	7.0282	0.5280
Textura [5]:		0.0193	6.4887	0.9035	0.4709	7.0618	0.5283
Textura [6]:		0.0251	4.6442	0.9151	0.5107	6.6688	0.5616
Textura [7]:		0.0079	6.1436	0.8235	0.3722	7.3654	0.4488
Textura [8]:		0.0078	6.3304	0.8215	0.3712	7.3984	0.4480
Textura [9]:		0.0069	7.8256	0.7868	0.3376	7.5614	0.4206
Textura [10]:		0.0478	19.1941	0.7418	0.4718	6.1327	0.5278

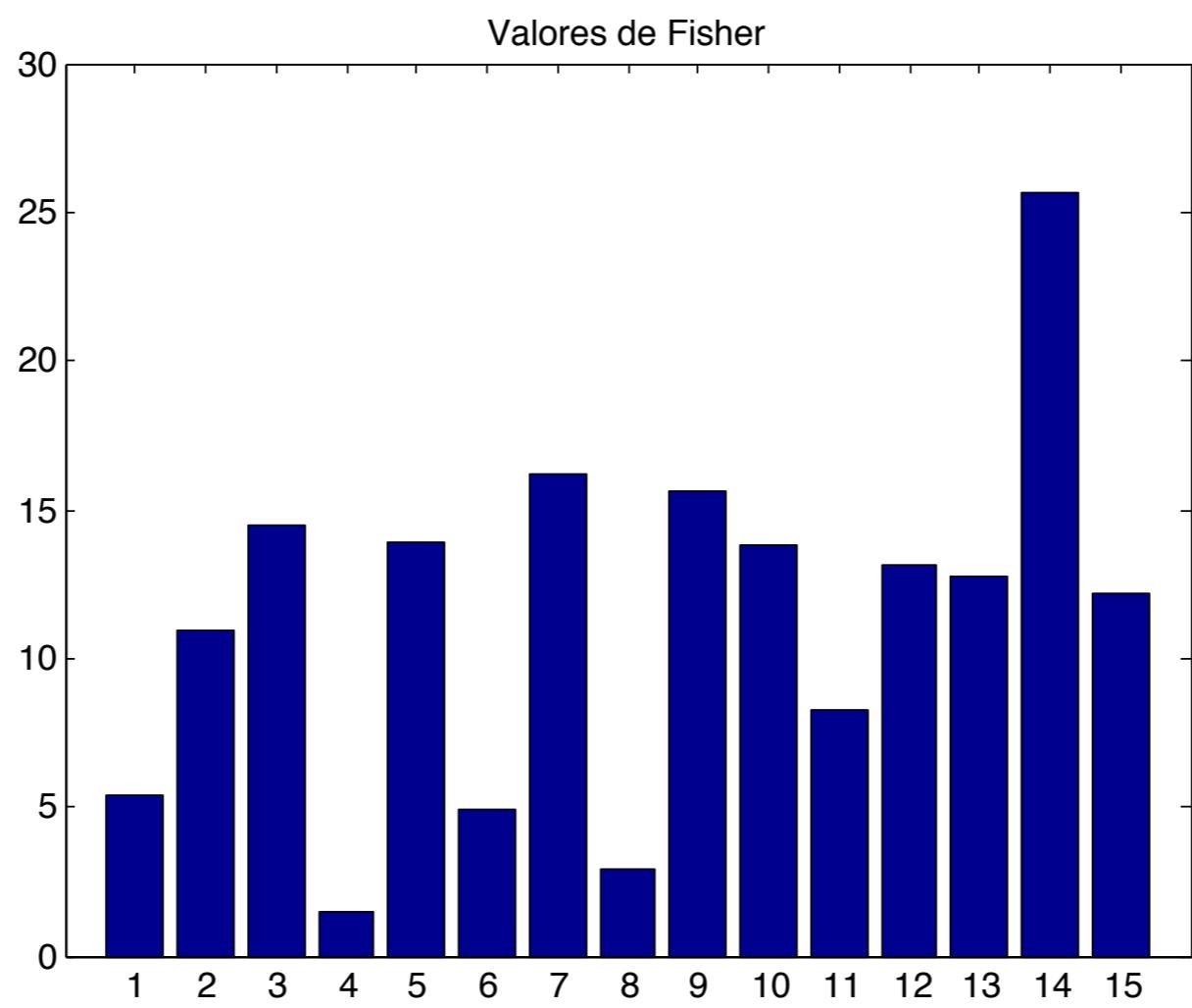
- Veamos cuales son las combinaciones

1	Energía	Contraste	6	Correlación	Homogeneidad	11	Contraste	Correlación
2	Energía	Correlación	7	Correlación	Entropía	12	Contraste	Homogeneidad
3	Energía	Homogeneidad	8	Correlación	Momento	13	Contraste	Entropía
4	Energía	Entropía	9	Homogeneidad	Entropía	14	Contraste	Momento
5	Energía	Momento	10	Homogeneidad	Momento	15	Entropía	Momento

Búsqueda Exhaustiva

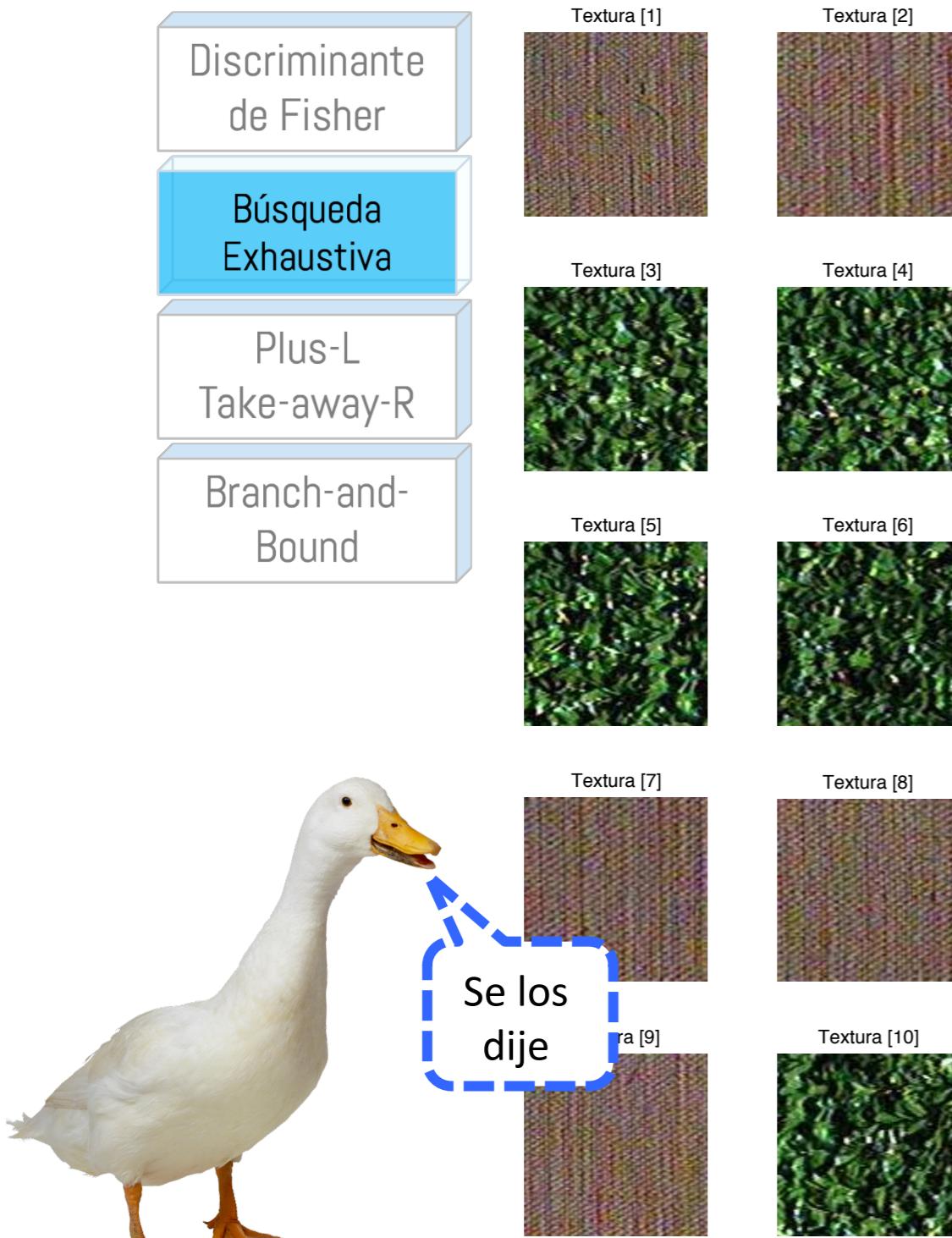


- ¿Cuales son los pares de características que maximizan el índice de Fisher?, es decir, **¿cuál es la mejor combinación que más separa las clases?**

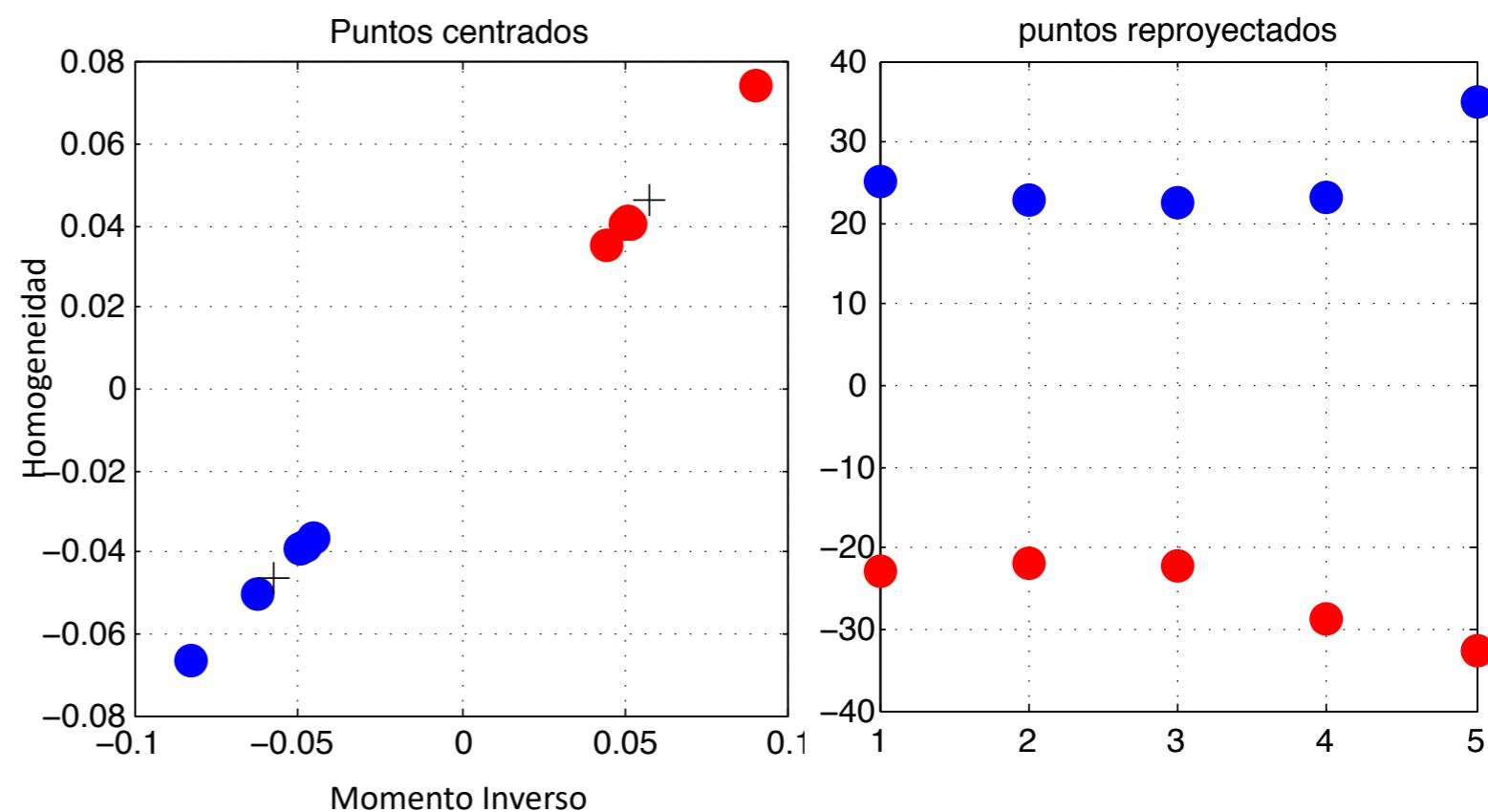


1	Energía	Contraste
2	Energía	Correlación
3	Energía	Homogeneidad
4	Energía	Entropía
5	Energía	Momento
6	Correlación	Homogeneidad
7	Correlación	Entropía
8	Correlación	Momento
9	Homogeneidad	Entropía
10	Contraste	Momento
11	Contraste	Correlación
12	Contraste	Homogeneidad
13	Contraste	Entropía
14	Homogeneidad	Momento
15	Entropía	Momento

Búsqueda Exhaustiva



- El mayor índice de Fisher lo alcanzamos con la combinación de dos características **Homogeneidad vs Momento Inverso** con un índice de Fisher de **25.97**



■ Plus-L, Take-away-R

Discriminante
de Fisher

Búsqueda
Exhaustiva

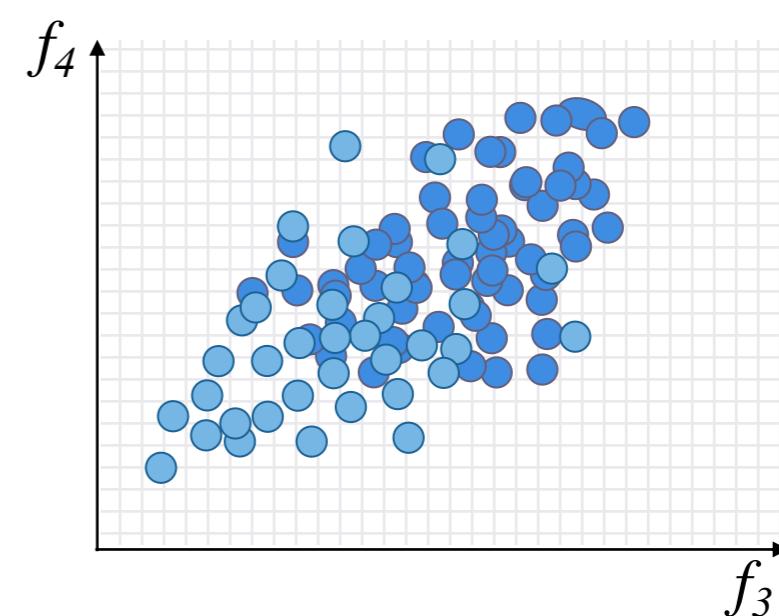
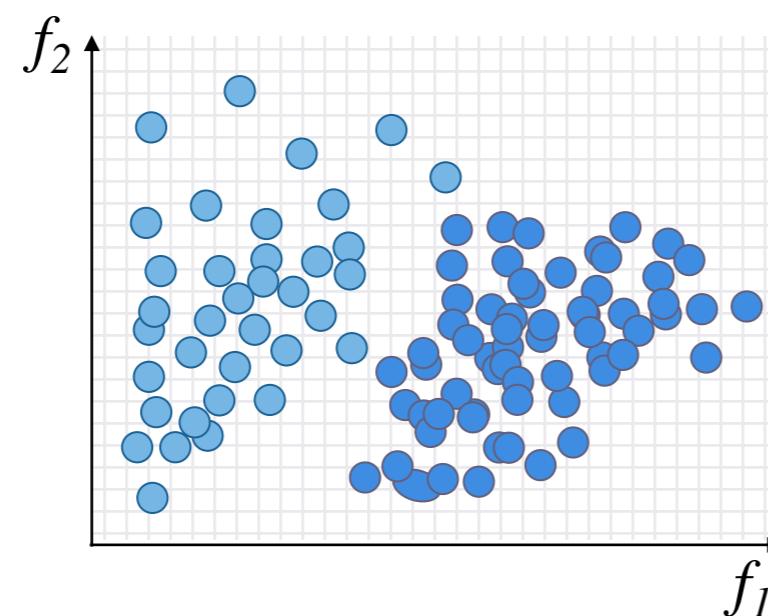
Plus-L
Take-away-R

Branch-and-
Bound

El problema de selección de selección de características se define de la siguiente forma:

Dado un set de d características, elegir un subset de m características ($m < d$), el cual obtiene un alto valor para una función de criterio, asumiendo que un alto valor indica un mejor subset de características.

En la figura se presenta cuatro características, ¿cuál de ellas obtiene un mayor separación?



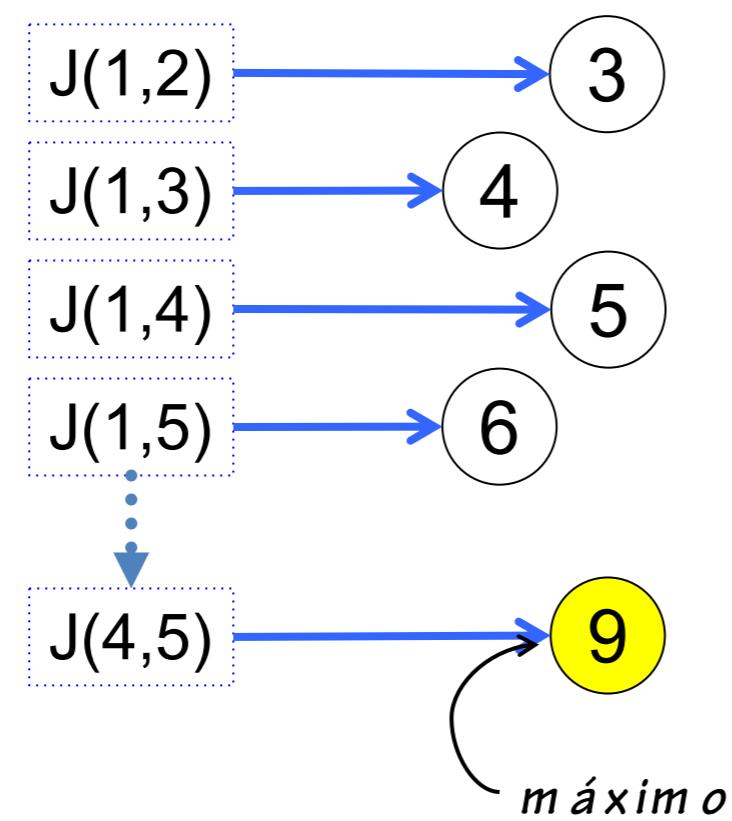
■ Plus-L, Take-away-R

- Discriminante de Fisher
- Búsqueda Exhaustiva
- Plus-L Take-away-R**
- Branch-and-Bound

La forma más obvia para seleccionar características es efectuando una búsqueda exhaustiva, con la desventaja de que es normalmente muy costoso examinar todas las combinaciones.

Ejemplo: Se tienen los números {1,2,3,4,5}. Se necesita escoger dos números cuya suma sea máxima.

Búsqueda Exhaustiva



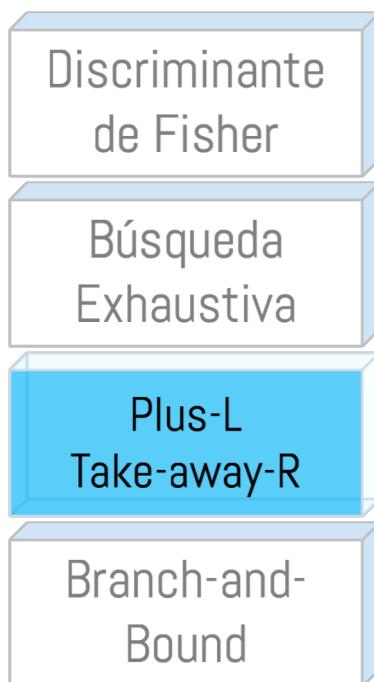
Combinaciones

Número de operaciones para encontrar el máximo

$$\binom{5}{2} = \frac{5!}{3!2!} = 10$$

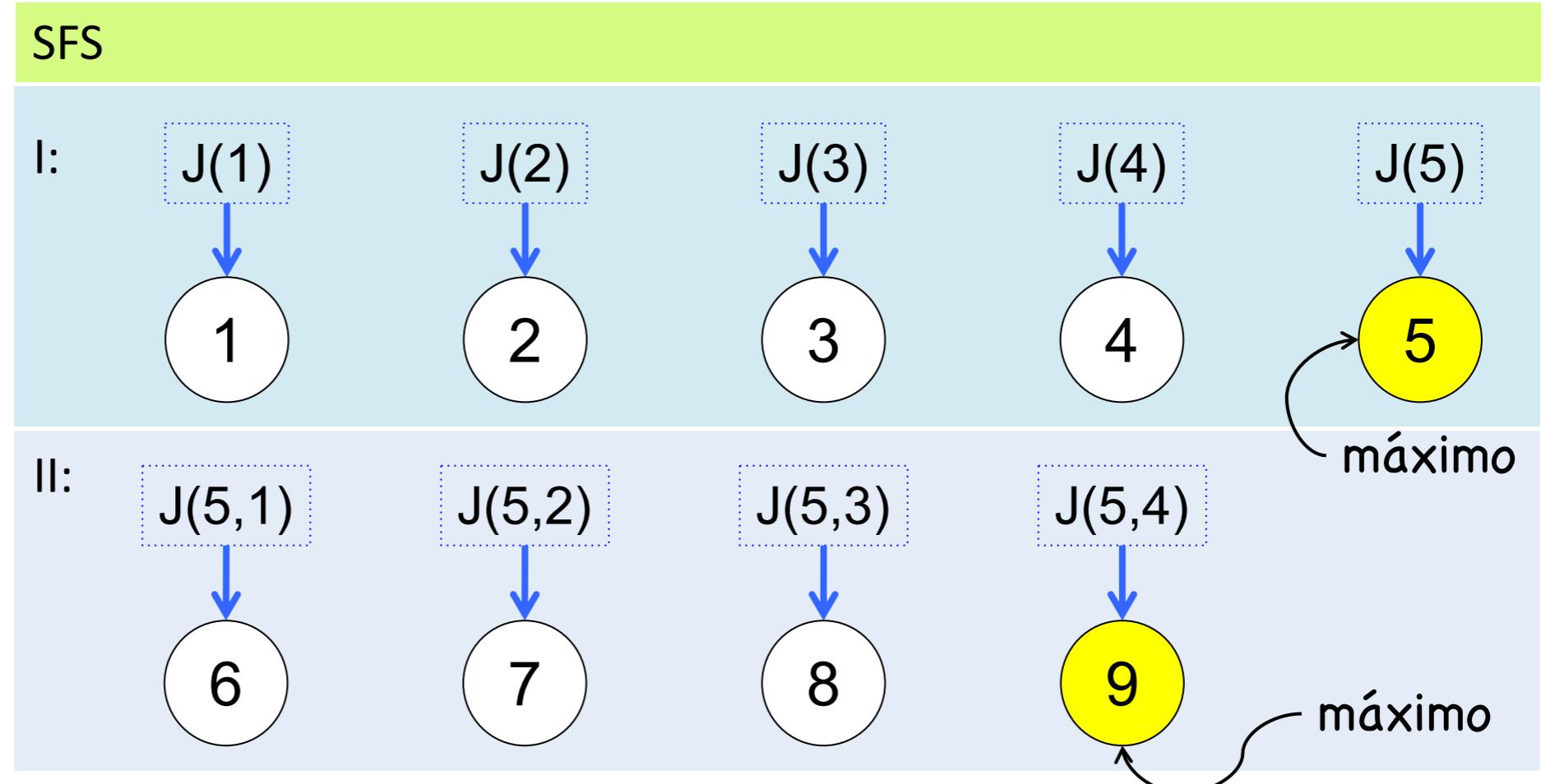
Son necesarias 10 evaluaciones

■ Plus-L, Take-away-R



Sequential Forward Selection (SFS): empieza seleccionando la mejor opción y luego va agregando la que mejor rendimiento tenga. Luego sigue este procedimiento hasta que cumpla algún criterio de parada

Ejemplo: Se tienen los números $\{1,2,3,4,5\}$. Se necesita escoger dos números cuya suma sea máxima.



■ Plus-L, Take-away-R

Discriminante
de Fisher

Búsqueda
Exhaustiva

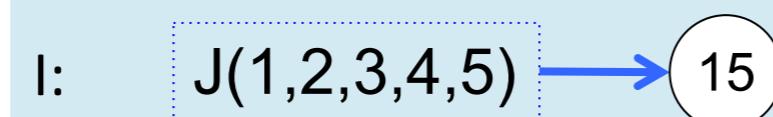
Plus-L
Take-away-R

Branch-and-
Bound

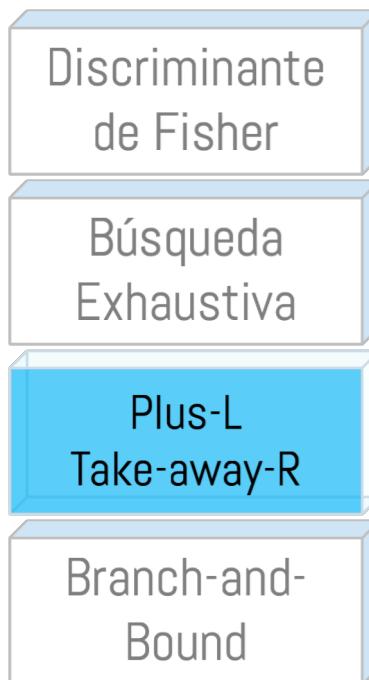
Ejemplo: Se tienen los números $\{1,2,3,4,5\}$. Se necesita escoger dos números cuya suma sea máxima.

Sequential Backward Selection (SBS): comienza seleccionando d características y va eliminando la que menos aporte al rendimiento. Este procedimiento hasta obtener algún criterio de parada.

SBS: Sequential Backward Selection



■ Plus-L, Take away-R

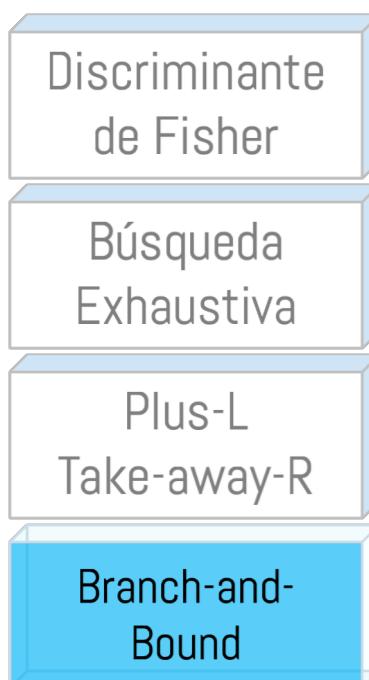


"Plus-L, take away-R" es un procedimiento que combina las dos técnicas previas. El procedimiento consiste en aplicar los siguientes pasos:

- (1) Aplicar un SFS para L características,
- (2) Aplicar SBS para R características del resultado anterior ($L > R$).

El espacio de búsqueda de éste método es mayor que el dado por SBS y SFS, además de que es relativamente práctico para implementarlo, por lo cual es un método adecuado.

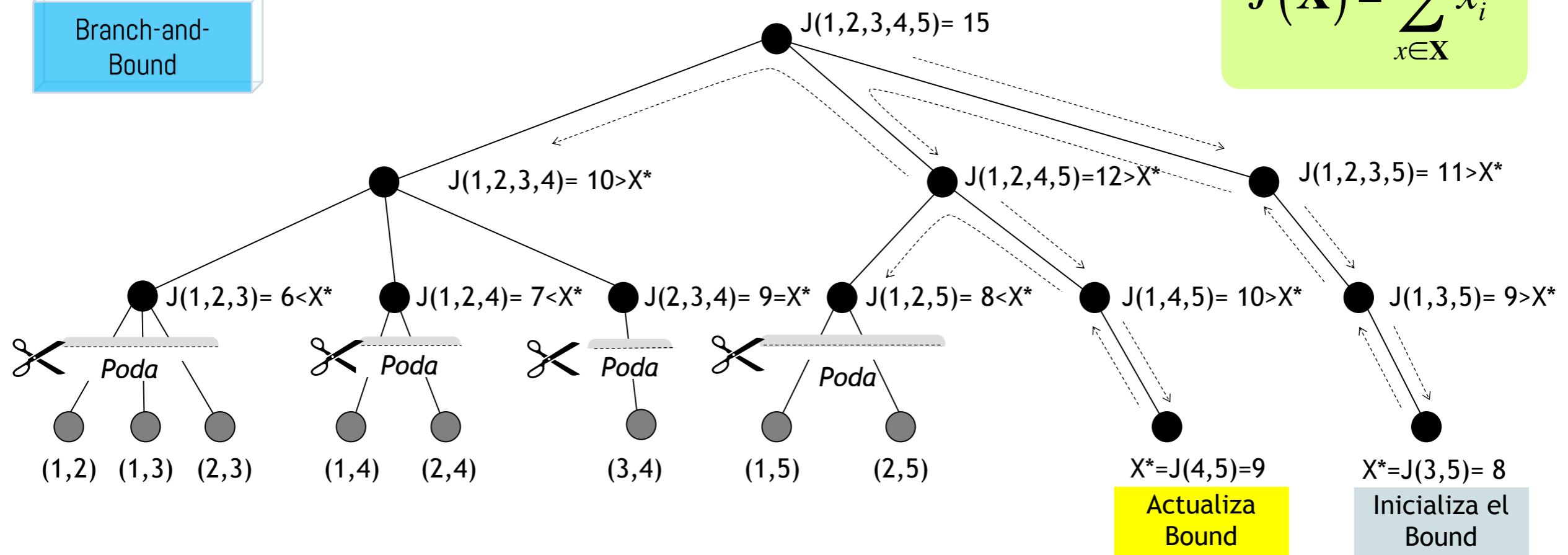
■ Branch and Bound (ramificación y poda)



- B&B es una heurística que **busca en todo el espacio de solución** de un problema. Normalmente es muy costoso (o impracticable) buscar todas las combinaciones, por ello B&B elimina (o poda) aquellas soluciones que no mejoran el resultado.

Función de criterio:

$$J(\mathbf{X}) = \sum_{x \in \mathbf{X}} x_i$$



■ Branch and Bound (ramificación y poda)

Discriminante
de Fisher

Búsqueda
Exhaustiva

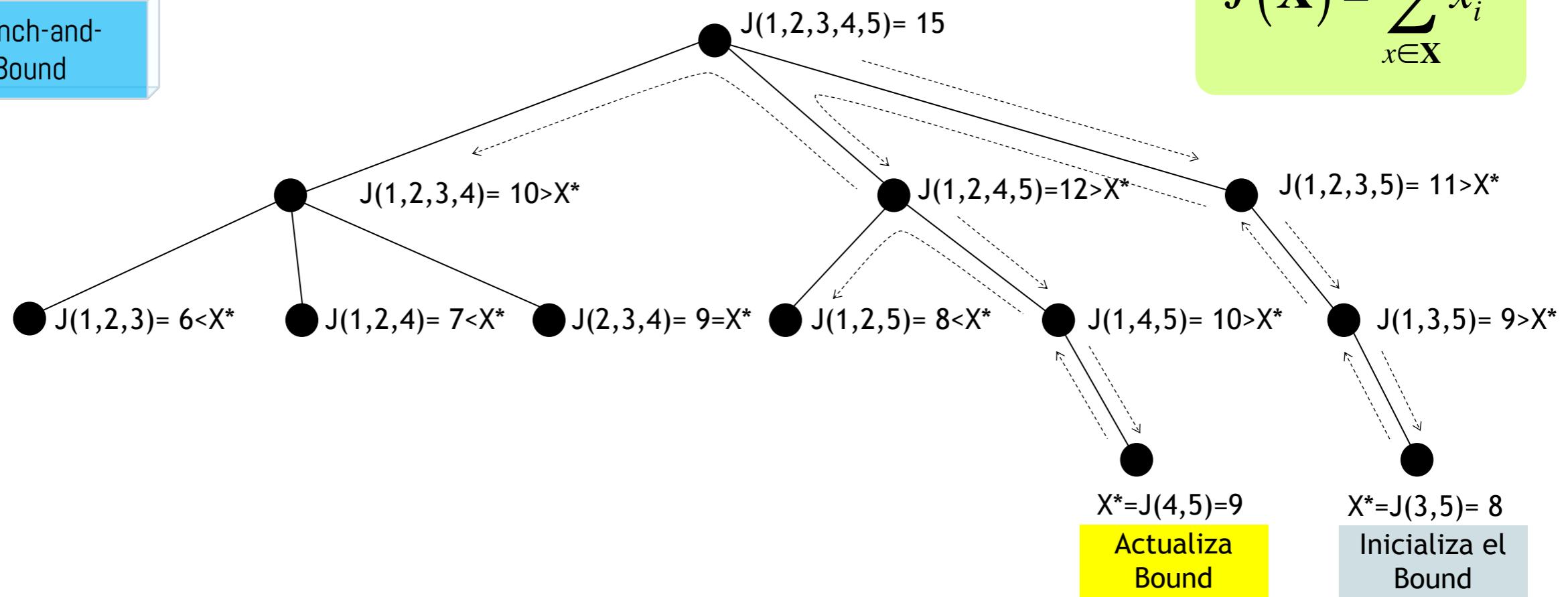
Plus-L
Take-away-R

Branch-and-
Bound

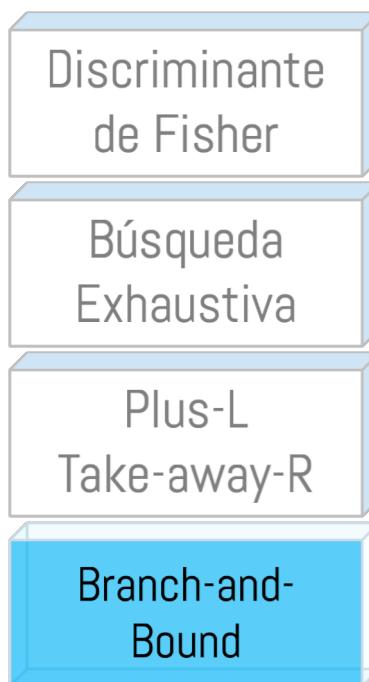
- La principal ventaja de este algoritmo es centrar su búsqueda en sólo aquellas ramas que mejoren el resultado. De esta forma podamos las ramas que aseguren no mejorar el resultado global.

Función de criterio:

$$J(\mathbf{X}) = \sum_{x \in \mathbf{X}} x_i$$



■ Branch and Bound (ramificación y poda)

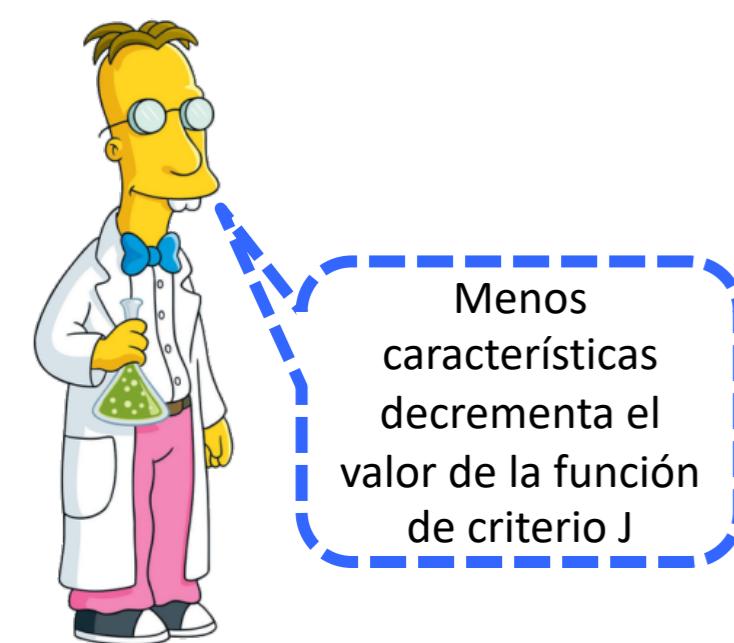


- Para que el algoritmo obtenga un resultado es condición necesaria que la función cumpla una **condición de monotonidad**.
- Esto significa que para un conjunto de n características, a medida que removemos una característica del conjunto, el valor de la función de criterio J va decreciendo.

$$J(a, b, c, d, e) \geq J(a, b, c, d) \geq J(a, b, c) \geq J(a, b) \geq J(a)$$

Características $\{a, b, c, d, e\}$ de un conjunto de n características

Menos características decrementa el valor de la función de criterio J



■ Compresión

- Análisis de Componentes Principales (PCA)
- Vector Quantisation

■ Selección

- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound

Me quedo una
duda, no
entiendo nada



■ Compresión

- Análisis de Componentes Principales (PCA)
- Vector Quantisation

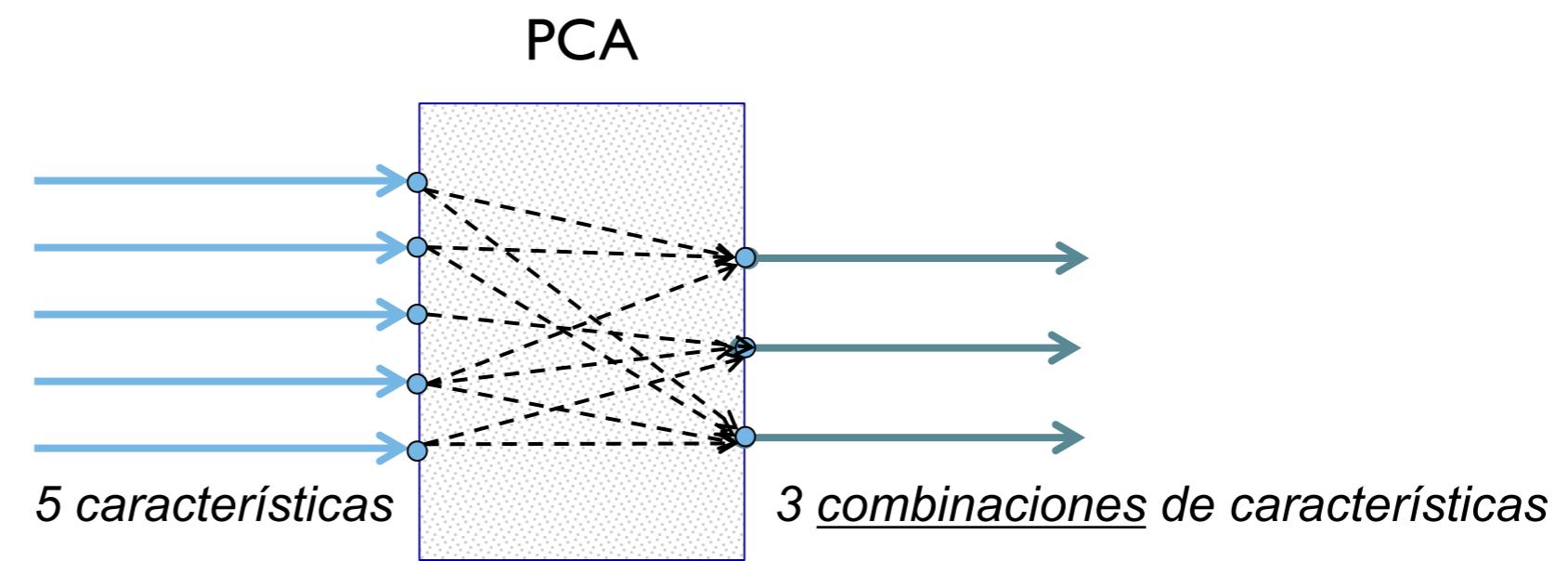
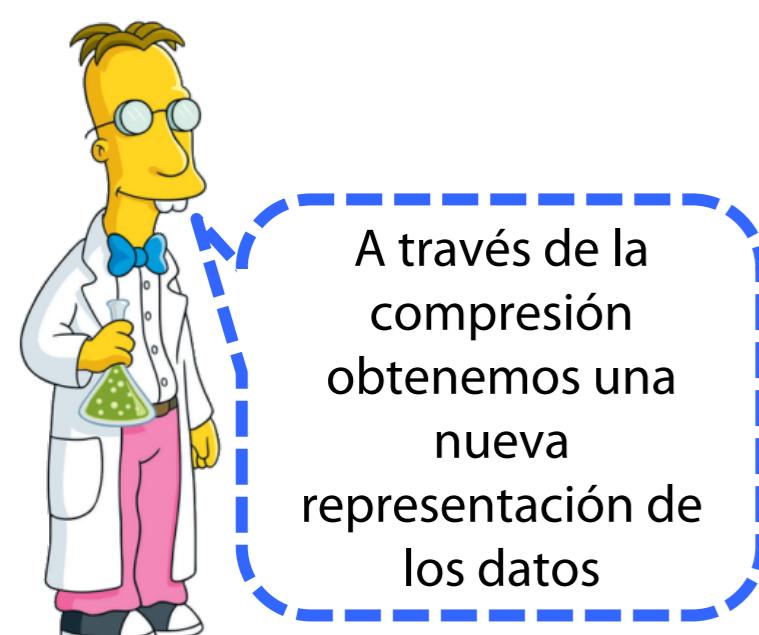
OBJETIVO:

Del conjunto de características, vamos a determinar una representación de la información original



■ Compresión

- Análisis de Componentes Principales (PCA)
- Vector Quantisation
- *Análisis de componentes principales*: El objetivo es transformar el espacio de características originales a través de una combinación lineal de las características maximizando la varianza de los datos.



■ Compresión

- *Análisis de componentes principales:* El objetivo es transformar el espacio de características originales a través de una **combinación lineal de las características** maximizando la varianza de los datos.

X	
2.5	2.4
0.5	0.7
2.2	2.9
1.9	2.2
3.1	3.0
2.3	2.7
2	1.6
1	1.1
1.5	1.6
1.1	0.9

B = X - \bar{X}	
0.69	0.49
-1.31	-1.21
.039	0.99
0.09	0.29
1.29	1.09
0.49	0.79
0.19	-0.31
-0.81	-0.81
-0.31	-0.31
-0.71	-1.01

Datos originales

En este ejemplo seleccionamos 1 vector propio con más energía

W
0.6779
0.7352

Si tenemos m características este vector será de largo m

$$w_1 \quad w_2 \\ \{0.6779\} \cdot (-0.71) + \{0.7352\} \cdot (-1.01)$$

Característica 1
Característica 2

Al multiplicar B con W estamos calculando una combinación lineal entre las características originales

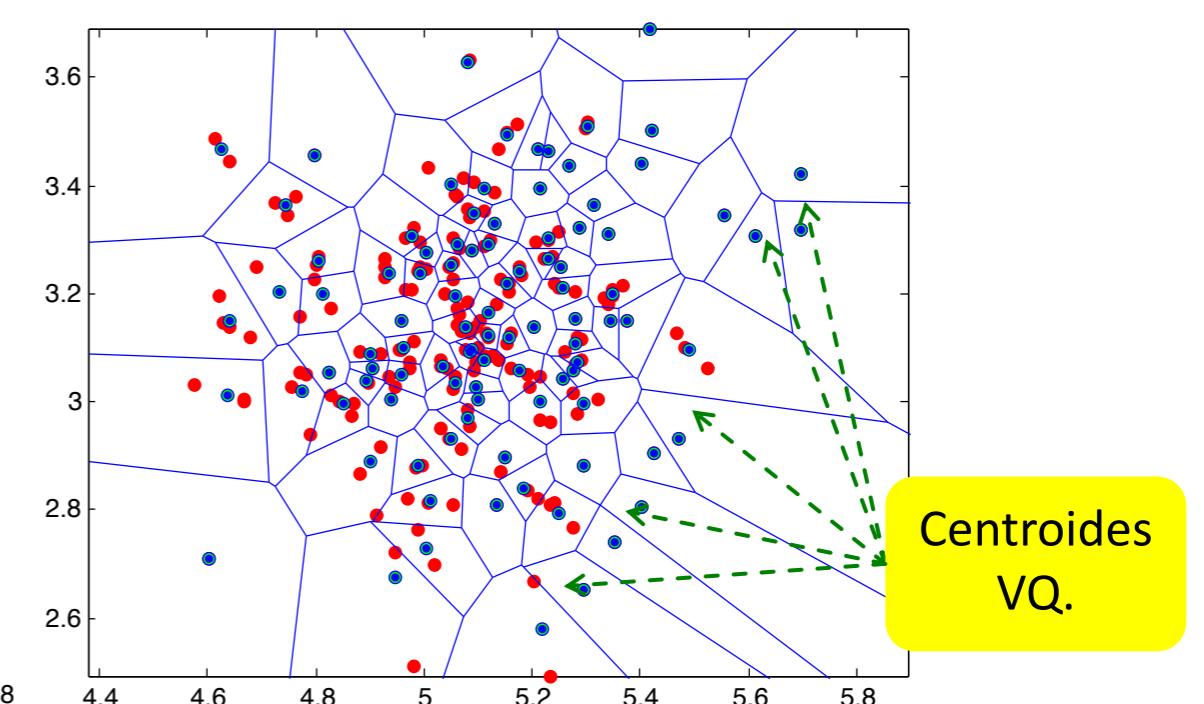
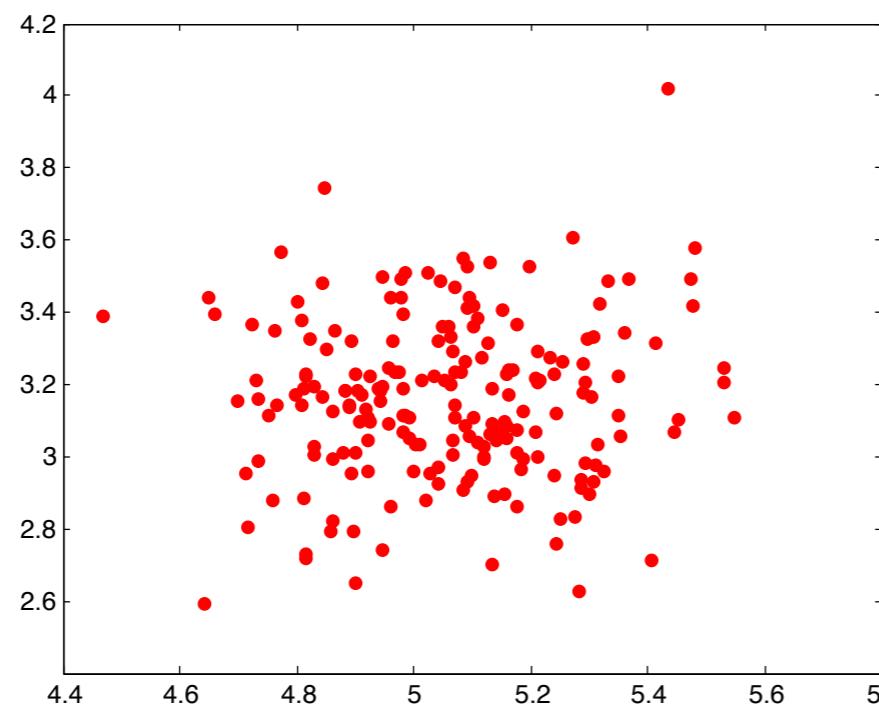
Combinación lineal

KLT
0.8280
-1.7776
0.9922
0.2742
1.6758
0.9129
-0.0991
-1.1446
-0.4380
-1.2238

Datos centrados

■ Compresión

- Análisis de Componentes Principales (PCA)
- Vector Quantisation
- *Vector Quantization*: El objetivo es representar buscar puntos representativos en los datos, reduciendo la cantidad de información en los datos. También puede ser visto como una técnica de compresión. En este caso no reducimos las características, sino los datos.



■ Selección

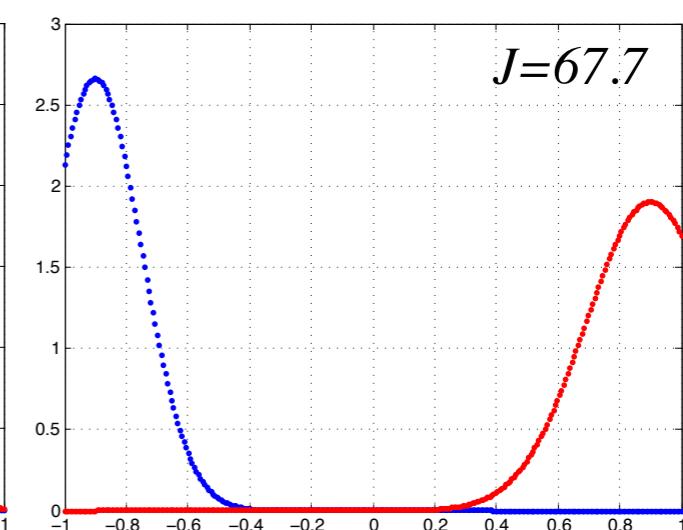
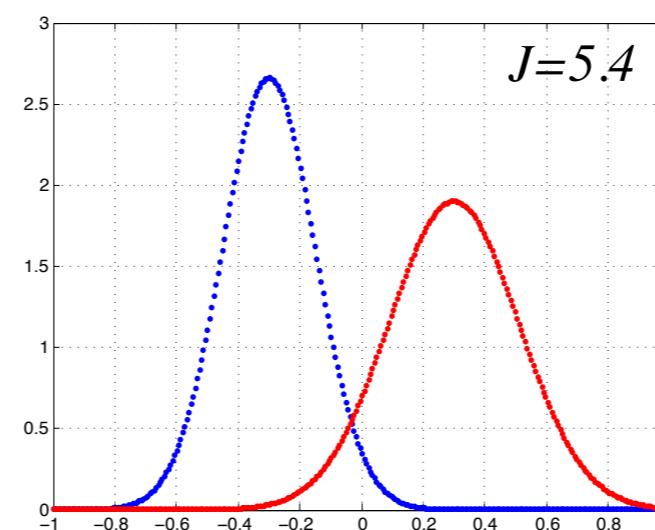
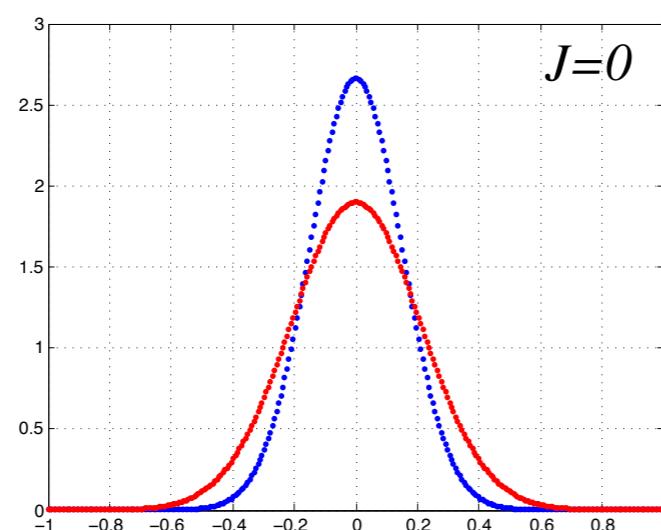
- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound

OBJETIVO:
Del conjunto de características,
vamos a escoger aquellas que
maximicen la separación entre las
clases



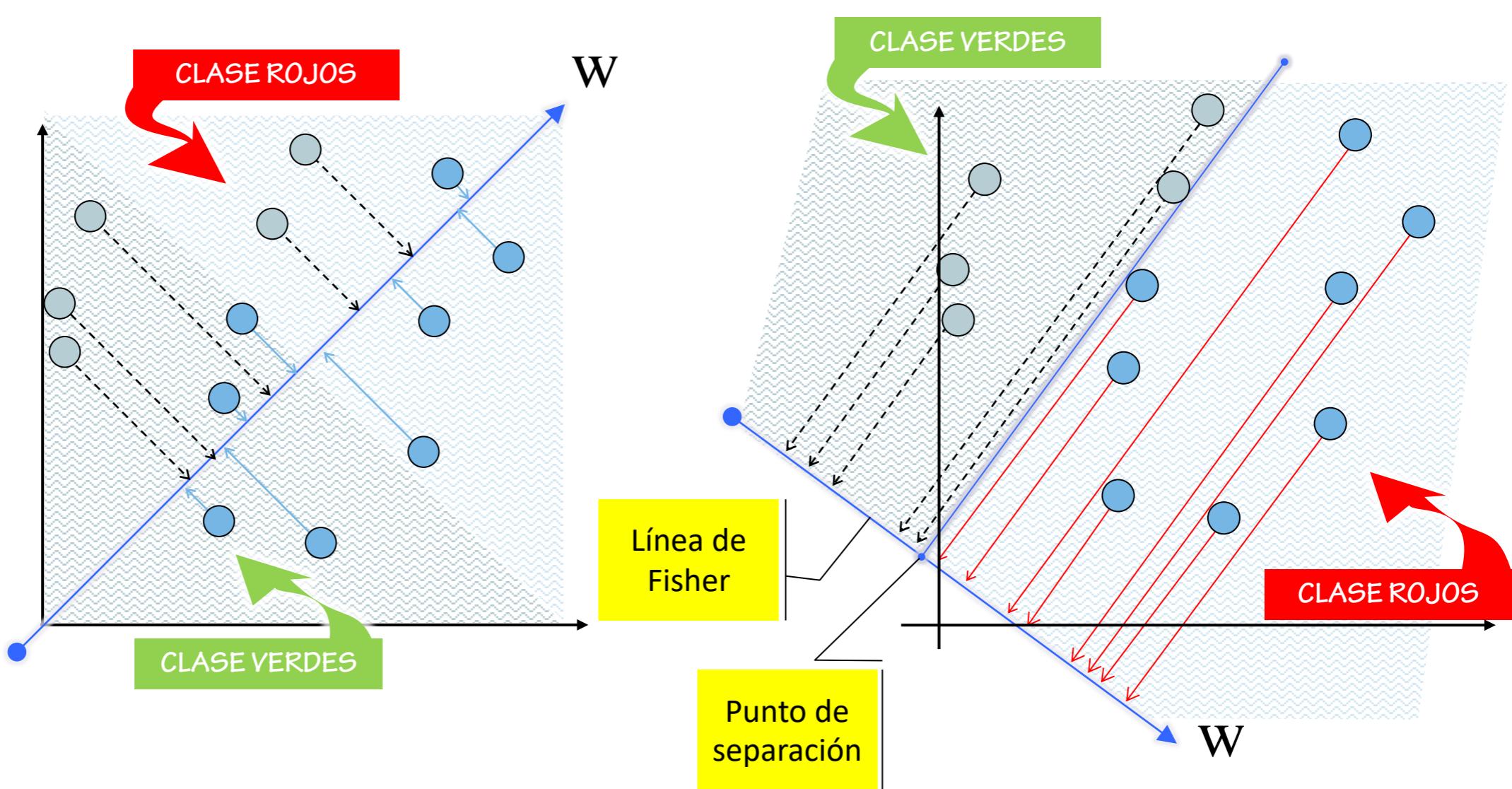
■ Selección

- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound
- *Discriminante de Fisher*: Determinar un índice de separación entre las clases a través de la covarianza entre **interclase** e **intraclasa**. Además Fisher puede ser calculado en un problema de m dimensiones con n clases.



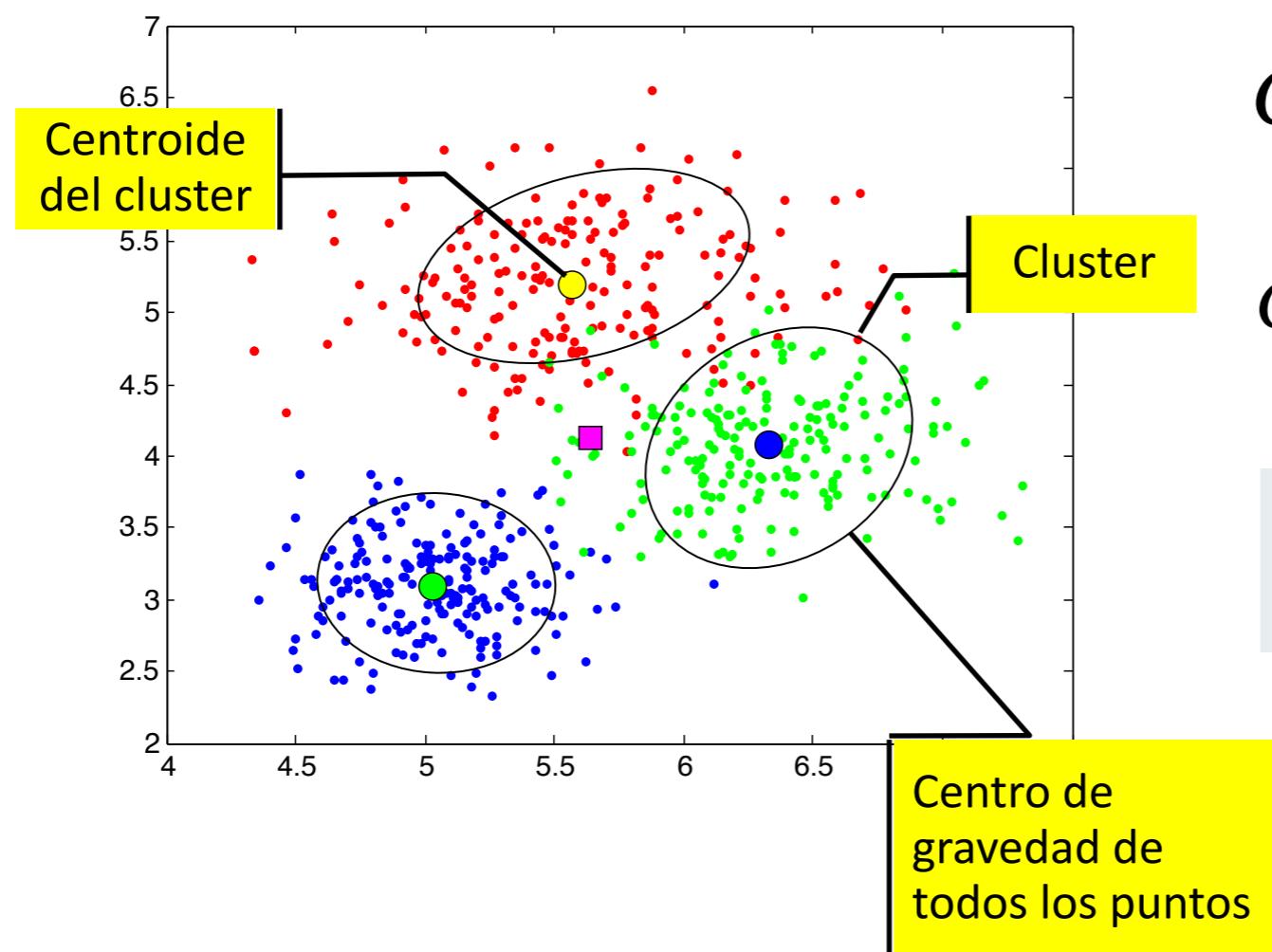
■ Selección

- *Discriminante de Fisher*: Determinar un índice de separación entre las clases a través de la covarianza **interclase** e **intraclasa**. Además Fisher puede ser calculado en un problema de m dimensiones con n clases.



■ Selección

- *Discriminante de Fisher*: Determinar un índice de separación entre las clases a través de la covarianza **interclase** e **interclase**. Además Fisher puede ser calculado en un problema de m dimensiones con n clases.



$$C_b = \sum_{k=1}^n p_k \cdot (\bar{v}_k - \bar{v}) \cdot (\bar{v}_k - \bar{v})^T$$

$$C_w = \sum_{k=1}^n p_k \cdot \text{cov}(G_k)$$

$$J = \text{trace}\left([C_w^{-1} \cdot C_b]\right)$$

■ Selección

- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound
- *Búsqueda exhaustiva*: El objetivo es buscar todas las combinaciones del espacio de característica aquel que maximice una función criterio, en este caso, la función J de Fisher.
- La búsqueda exhaustiva supone buscar todos los grupos de k características dentro de un grupo de n características ($k < n$). Esto último es calculado con el *coeficiente binomial*

$$\binom{n}{k} = \frac{n!}{(n-k)! \cdot k!}$$

■ Selección

¿Cuales son los pares de características que maximizan el índice de Fisher?

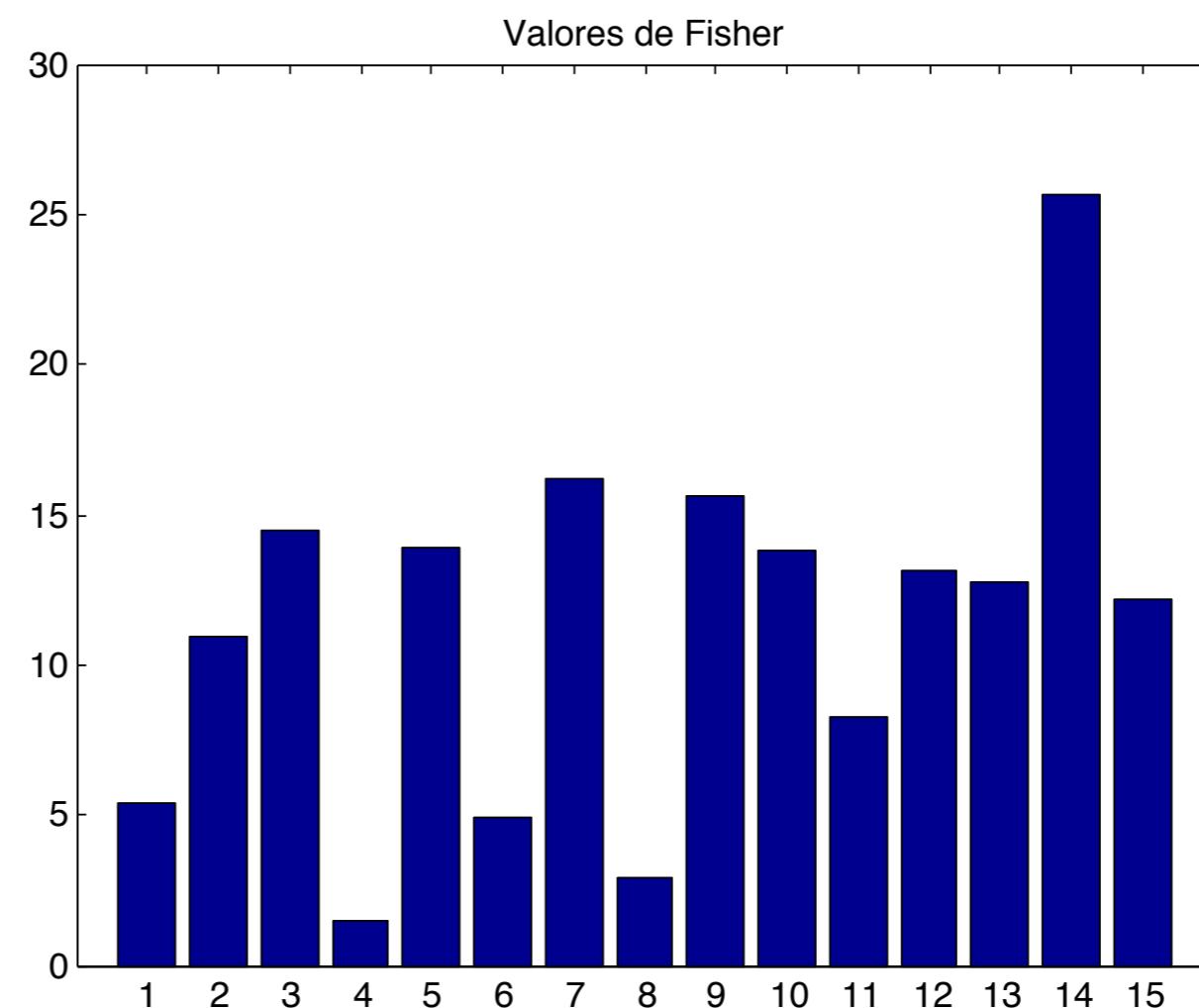
- La búsqueda exhaustiva supone buscar todos los grupos de k características dentro de un grupo de n características ($k < n$). Esto último es calculado con el [coeficiente binomial](#)

	Energía	Contraste	Correlación	Homogeneidad	Entropía	Momento
Textura [1]:	0.0088	6.7759	0.7531	0.3581	7.2402	0.4369
Textura [2]:	0.0077	5.9886	0.8399	0.3747	7.4072	0.4510
Textura [3]:	0.0390	18.081	0.7537	0.4648	6.2451	0.5229
Textura [4]:	0.0182	5.5507	0.9141	0.4705	7.0282	0.5280
Textura [5]:	0.0193	6.4887	0.9035	0.4709	7.0618	0.5283
Textura [6]:	0.0251	4.6442	0.9151	0.5107	6.6688	0.5616
Textura [7]:	0.0079	6.1436	0.8235	0.3722	7.3654	0.4488
Textura [8]:	0.0078	6.3304	0.8215	0.3712	7.3984	0.4480
Textura [9]:	0.0069	7.8256	0.7868	0.3376	7.5614	0.4206
Textura [10]:	0.0478	19.1941	0.7418	0.4718	6.1327	0.5278

■ Selección

1	Energía	Contraste
2	Energía	Correlación
3	Energía	Homogeneidad
4	Energía	Entropía
5	Energía	Momento
6	Correlación	Homogeneidad
7	Correlación	Entropía
8	Correlación	Momento
9	Contraste	Momento
10	Contraste	Correlación
11	Contraste	Homogeneidad
12	Contraste	Entropía
13	Homogeneidad	Momento
14	Entropía	Momento

- Buscar todo el espacio de soluciones puede ser extremadamente costoso, sino imposible, por ello muy pocas veces será posible evaluar todo el espacio combinatorial



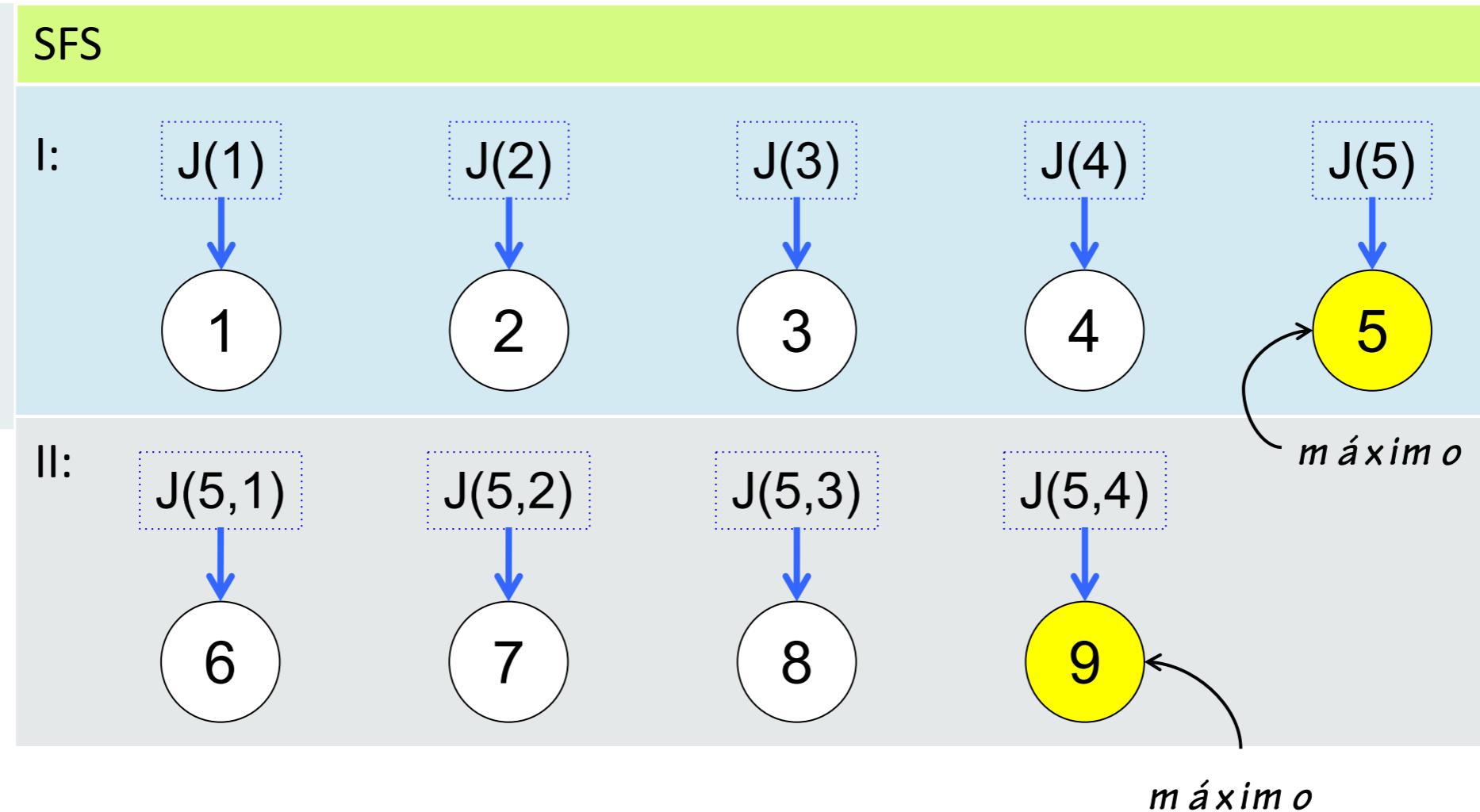
■ Selección

- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound
- *Plus-L Take-R*: El objetivo es seleccionar L características por medio del algoritmo SFS y con el resultado anterior reducimos el espacio de búsqueda en R características.
- *Sequential Forward Selection* (SFS): Comienza seleccionando la mejor opción y luego va agregando la que mejor rendimiento tenga. Este procedimiento se realiza hasta cumplir algún criterio de parada
- *Sequential Backward Selection* (SBS): comienza seleccionando d características y va eliminando la que menos aporte al rendimiento. Este procedimiento hasta obtener algún criterio de parada

■ Selección

- *Sequential Forward Selection* (SFS): Comienza seleccionando la mejor opción y luego va agregando la que mejor rendimiento tenga. Luego sigue este procedimiento hasta que cumpla algún criterio de parada

Ejemplo: Se tienen los números $\{1,2,3,4,5\}$. Se necesita escoger dos números cuya suma sea máxima.

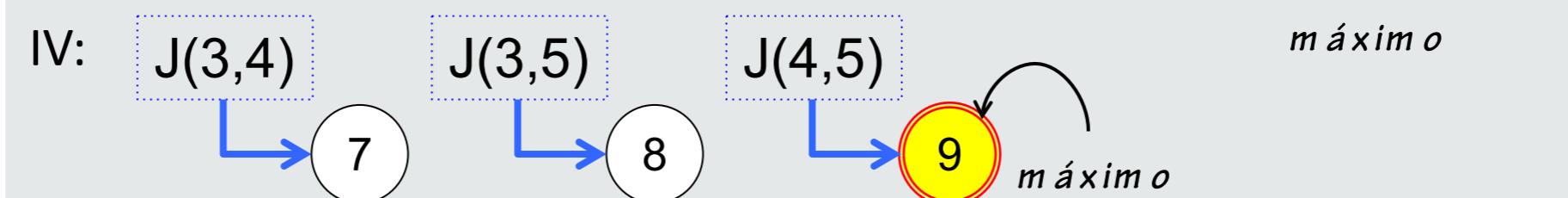
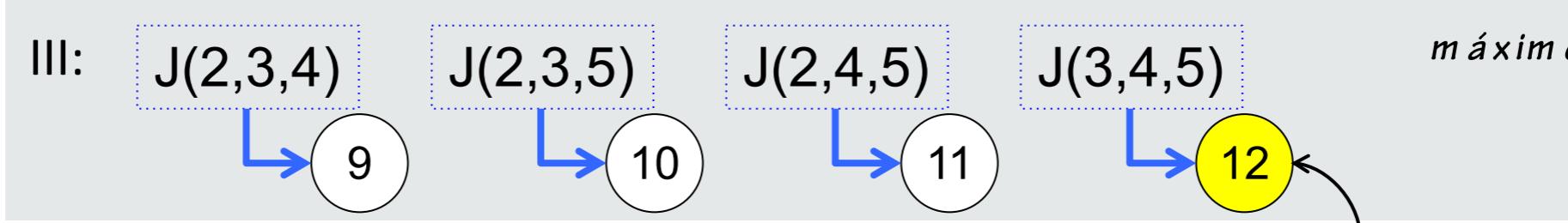


■ Selección

- *Sequential Backward Selection* (SBS): comienza seleccionando d características y va eliminando la que menos aporte al rendimiento. Este procedimiento hasta obtener algún criterio de parada

Ejemplo: Se tienen los números $\{1,2,3,4,5\}$. Se necesita escoger dos números cuya suma sea máxima.

SBS: Sequential Backward Selection



■ Selección

- Discriminante de Fisher
- Búsqueda exhaustiva
- “Plus L-take away R”
- Branch & Bound
- *Branch-and-Bound*: Es una herística de búsqueda normalmente empleado en problemas de optimización lineal. Para nuestro problema, consiste en buscar la combinación de características que maximice una función criterio (como Fisher) ya que B&B explora todo el espacio combinatorio.
- Ejemplo: Buscar las mejores 10 características de un conjunto de 100

$$\binom{100}{10} = \frac{100!}{(90)! \cdot 10!} = 1.73 \times 10^{13}$$

Requiere buscar 17.3 trillones de combinaciones (1 trillón es un millón de millones)

■ Selección

- **Branch-and-Bound:** Es una heruística de búsqueda normalmente empleado en problemas de optimización lineal. Para nuestro problema, consiste en buscar la combinación de características que maximice una función criterio (como Fisher) ya que B&B explora todo el espacio combinatorio.

