

UNIVERSIDAD
NACIONAL
DE COLOMBIA

APRENDIZAJE DE MÁQUINAS

Extracción y Selección de Características

JORGE ESPINOSA

Docente Investigador

Politécnico Colombiano Jaime Isaza Cadavid

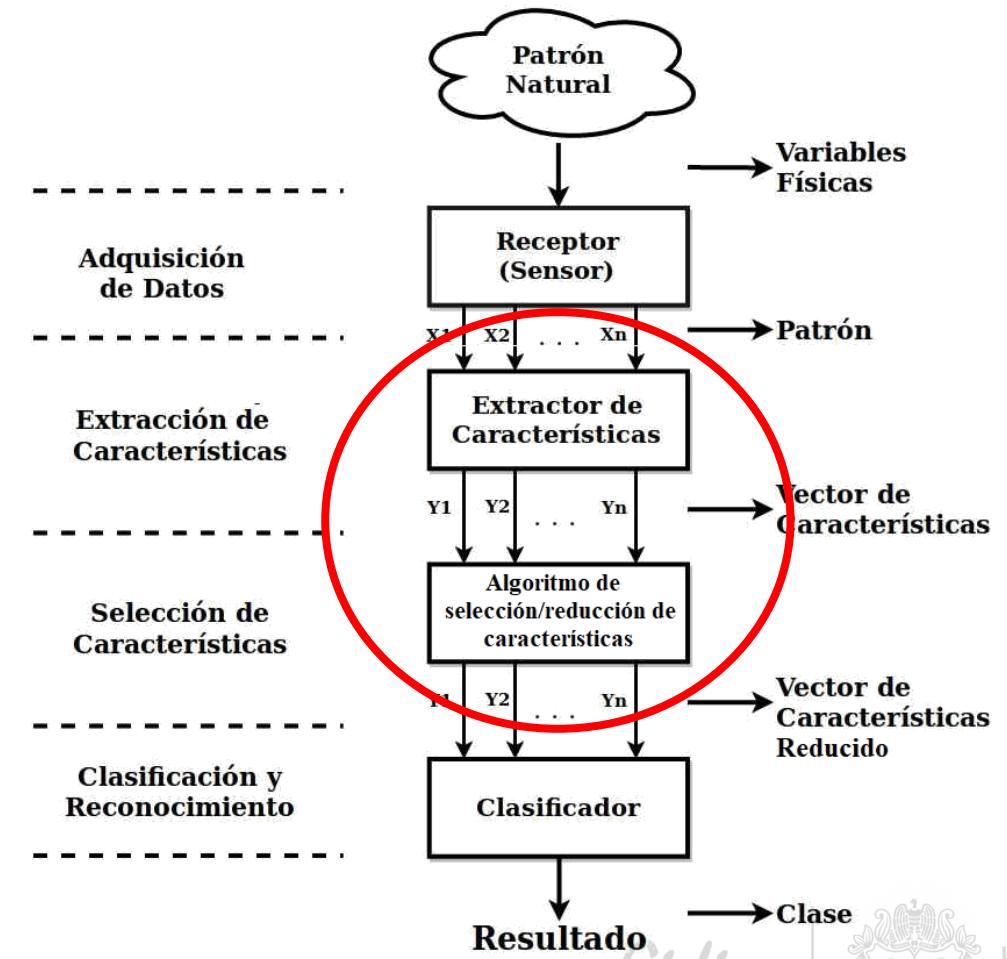
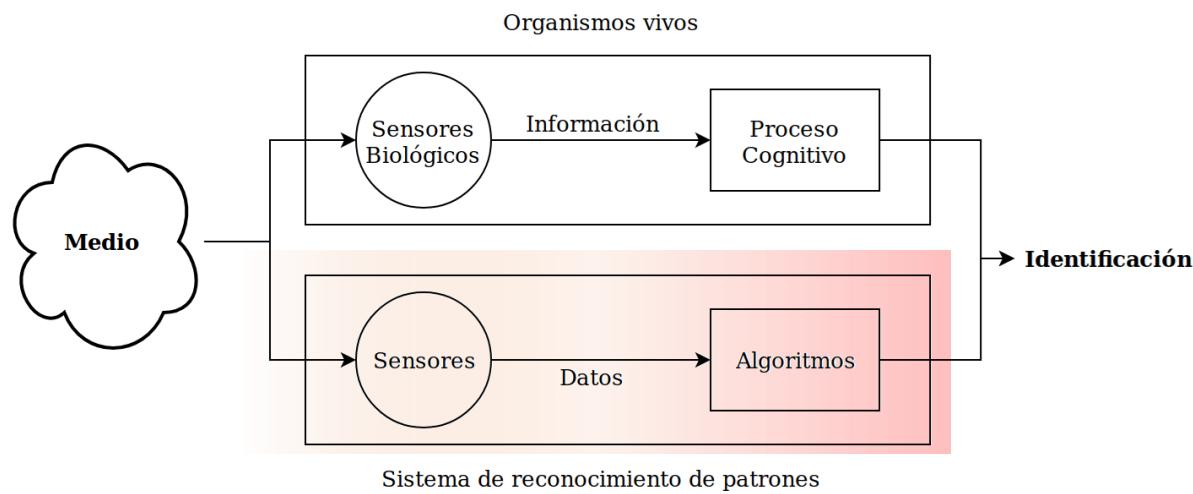
jeespinosa@elpoli.edu.co

<https://github.com/srobles05/3008422-AprendizajeDeMaquinas>

Contenido

1. Generalidades
2. Por qué la Reducción de la Dimensionalidad ?
3. Principales técnicas de reducción de la dimensionalidad
 - a. Selección de Características
 - b. Extracción de Características
4. Reducción de la dimensionalidad en Python
 - a. Selección Vs Extracción
5. La Maldición de la Dimensionalidad (The curse of dimensionality)
6. Correlación de Valores
7. Extracción de características
8. PCA – Análisis de Componente Principal

Etapas Sistema de Aprendizaje de Máquinas

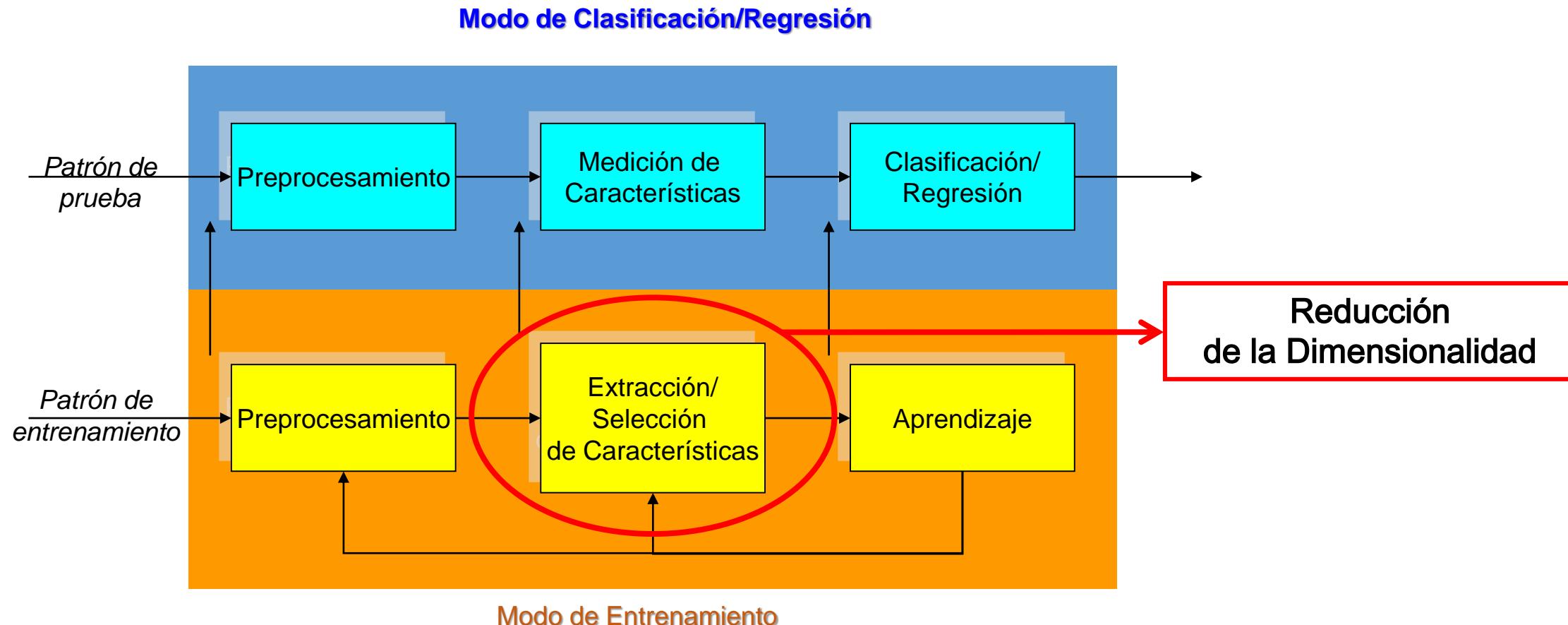


Representación de un Patrón

- Un patrón es representado por un conjunto de *d* **características**, o atributos, visto como un vector *d*-dimensional *vector de características*.

$$\mathbf{x} = (x_1, x_2, \dots, x_d)^T$$

Dos procesos para la Implementación de sistemas de Aprendizaje de Máquinas



Por qué la Reducción de la Dimensionalidad ?

- En la actualidad es muy fácil **recopilar Datos**
- Los datos **no son únicamente recopilados para análisis**
- Los **datos se acumulan con una velocidad sin precedentes**
- El **preprocesamiento de los datos** (Reduciendo Dimensionalidad?) resulta *efectivo* en el aprendizaje de máquinas.
- La **reducción de la dimensionalidad** es una técnica efectiva en la optimización de la información

Por qué la Reducción de la Dimensionalidad ?

- La mayoría de las técnicas de aprendizaje de máquinas **pueden no ser efectivas** en datos con alta dimensionalidad
 - **La Maldición de la dimensionalidad**
 - La precisión y la eficiencia de la consulta se degradan rápidamente a medida que aumenta la dimensión.
- La dimensión **intrínseca** puede ser pequeña.
 - Por ejemplo, el número de genes responsables de cierto tipo de enfermedad puede ser pequeño.

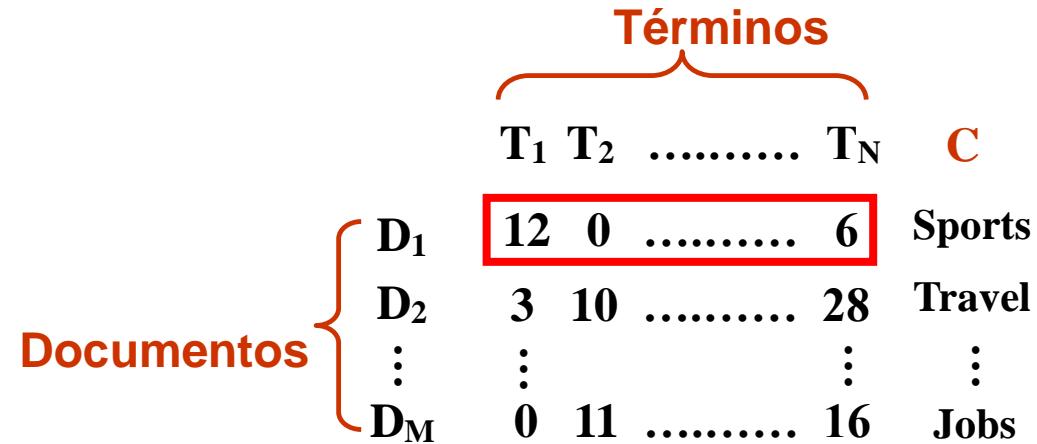
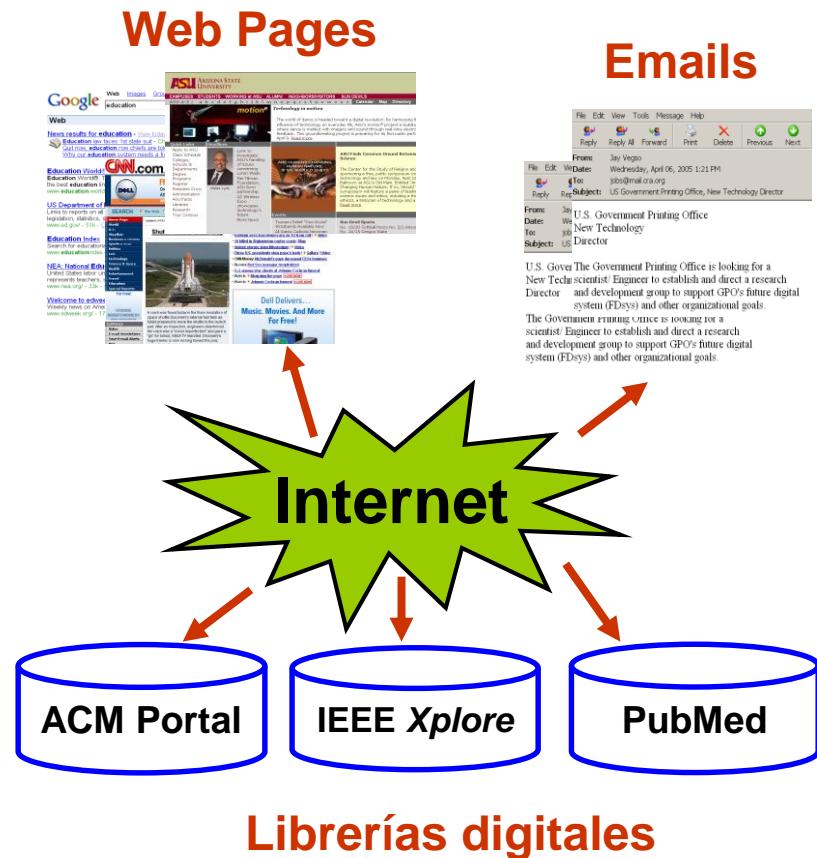
Por qué la Reducción de la Dimensionalidad ?

- **Visualización:** proyección de datos multidimensionales en 2D o 3D.
- **Compresión de datos:** almacenamiento y recuperación eficientes.
- **Eliminación de ruido:** efecto positivo en la precisión de la consulta.

Aplicaciones de la Reducción de la Dimensionalidad

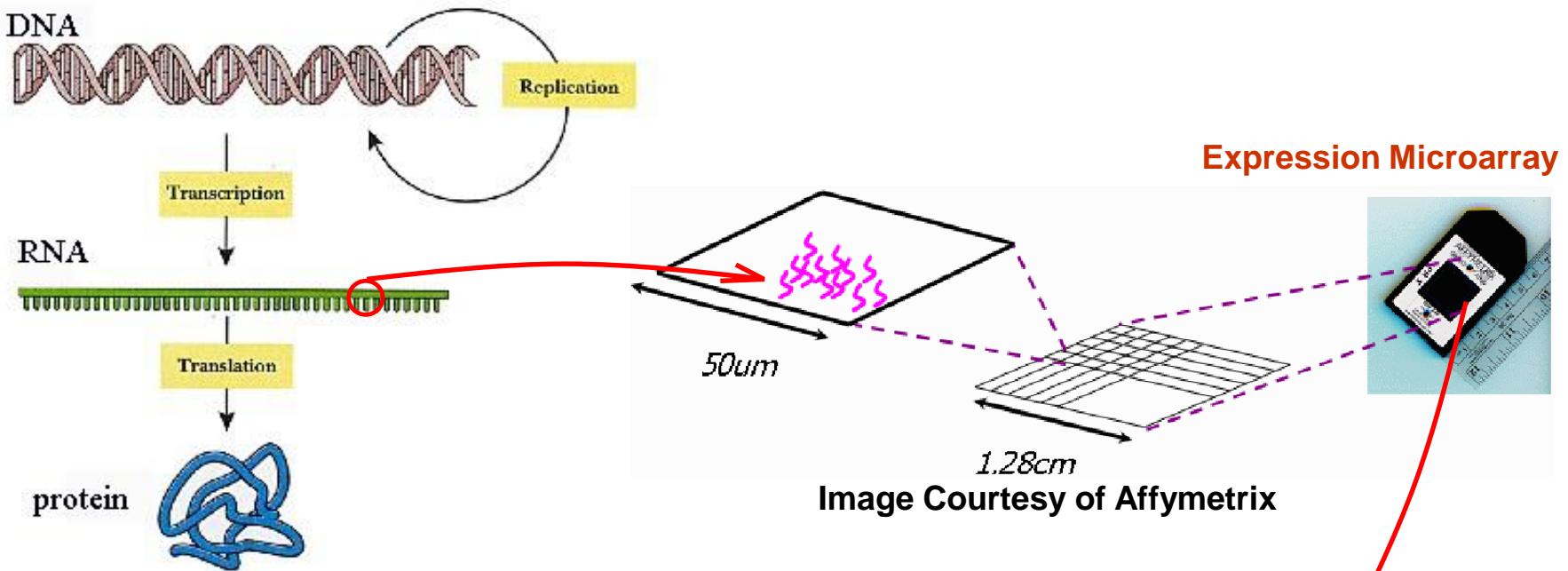
- Gestión de la relación con el cliente
- Extracción de textos
- Recuperación de imágenes
- Análisis de datos de microarrays
- Clasificación de proteínas
- Reconocimiento facial
- Reconocimiento de dígitos escritos a mano
- Detección de intrusiones

Clasificación de documentos



- **Tarea:** clasificar documentos sin etiqueta en categorías
- **Desafío:** miles de términos
- **Solución:** Aplicar reducción de la dimensionalidad

Análisis de microarrays de expresión genética



- **Tarea:** Clasificar nuevas muestras en tipos de enfermedades conocidas (diagnóstico de enfermedades)
- **Desafío:** miles de genes, pocas muestras
- **Solución:** aplicar reducción de dimensionalidad

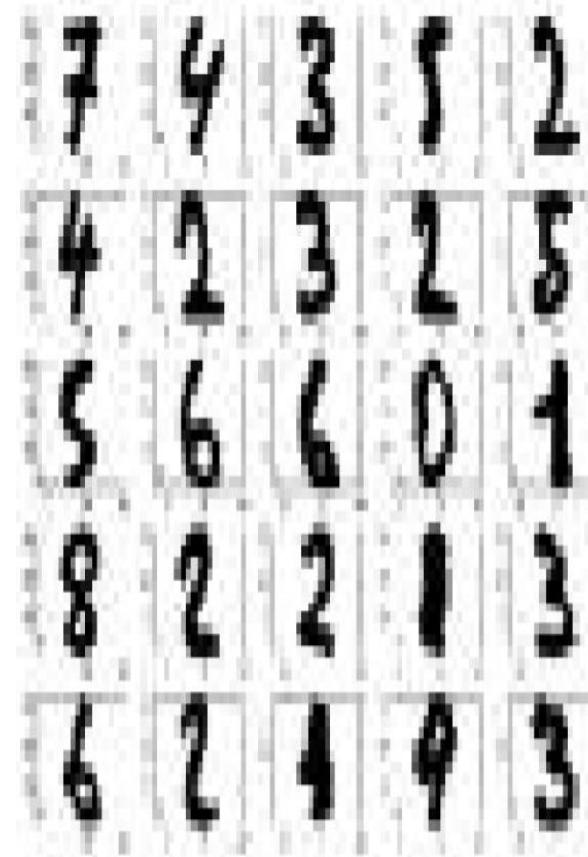
Sample \ Gene	M23197_at	U66497_at	M92287_at	...	Class
Sample 1	261	88	4778	...	ALL
Sample 2	101	74	2700	...	ALL
Sample 3	1450	34	498	...	AML
.
.

Expression Microarray Data Set

Otros tipos de datos multidimensionales



Imágenes de rostros



Dígitos manuscritos

Principales técnicas de reducción de dimensionalidad

- Selección de características
 - Definición
 - Objetivos
- Extracción de características (reducción)
 - Definición
 - Objetivos
- Diferencias entre las dos técnicas

Selección de características

- **Definición**

- Un proceso que elige un subconjunto óptimo de características de acuerdo con una función objetivo

- **Objetivos:**

- Para reducir la dimensionalidad y eliminar el ruido.
- Para mejorar el desempeño del reconocimiento de patrones
 - Velocidad de aprendizaje
 - Precisión predictiva
 - Simplicidad y comprensibilidad de los resultados de reconocimiento de patrones.

Extracción de Características (Reducción)

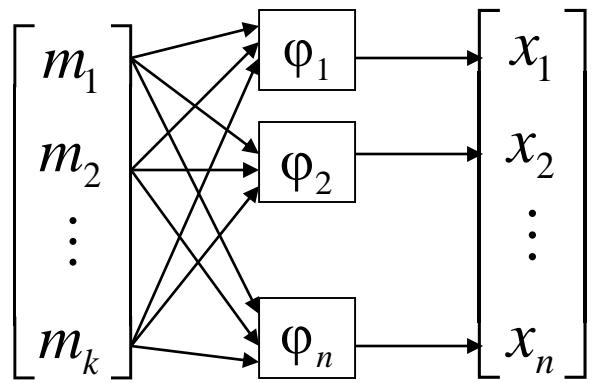
- La reducción de características se refiere al mapeo de los datos originales de alta dimensión en un espacio de menor dimensión
- Dado un conjunto de puntos de datos de p variables $\{x_1, x_2, \dots, x_n\}$
Computar su representación en una dimensión más reducida:

$$x_i \in \Re^d \rightarrow y_i \in \Re^p \quad (p \ll d)$$

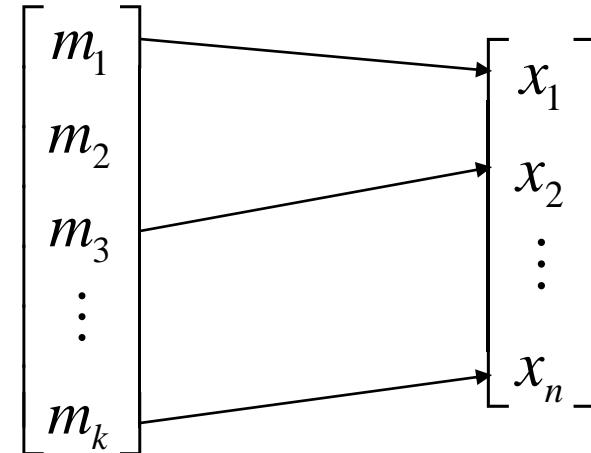
- El criterio para la reducción de características puede ser diferente de acuerdo a las diferentes configuraciones de problemas.
 - Configuración **no supervisada**: minimice la pérdida de información
 - Entorno **supervisado**: maximizar la discriminación de clase

Métodos de extracción de Características

Extracción de Características



Selección de Características



El problema puede expresarse como la optimización de los parámetros $\varphi(\theta)$ del extractor de características.

Métodos supervisados: la función objetivo es un criterio de separabilidad (discriminabilidad) de ejemplos etiquetados, por ejemplo, análisis de discriminación lineal (LDA).

Métodos no supervisados: se busca una representación dimensional más baja que conserve las características importantes de los datos de entrada, por ejemplo, análisis de componentes principales (PCA).

Datos Organizados

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

Datos Organizados

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

Datos Organizados

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

Datos Organizados (comando shape)

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

```
pokemon_df.shape
```

(5, 7)

Cuando Utilizamos Reducción de Características

Name	Type	HP	Attack	Defense	Speed	Generation
Bulbasaur	Grass	45	49	49	45	1
Ivysaur	Grass	60	62	63	60	1
Venusaur	Grass	80	82	83	80	1
Charmander	Fire	39	52	43	65	1
Charmeleon	Fire	58	64	58	80	1

Datos Organizados (comando describe())

	HP	Attack	Defense	Speed	Generation
count	5.0	5.0	5.0	5.0	5.0
mean	56.4	61.8	59.2	66.0	1.0
std	15.9	13.0	15.4	14.7	0.0
min	39.0	49.0	43.0	45.0	1.0
25%	45.0	52.0	49.0	60.0	1.0
50%	58.0	62.0	58.0	65.0	1.0
75%	60.0	64.0	63.0	80.0	1.0
max	80.0	82.0	83.0	80.0	1.0

```
pokemon_df.describe()
```

Datos Organizados (comando describe())

	HP	Attack	Defense	Speed	Generation
count	5.0	5.0	5.0	5.0	5.0
mean	56.4	61.8	59.2	66.0	1.0
std	15.9	13.0	15.4	14.7	0.0
min	39.0	49.0	43.0	45.0	1.0
25%	45.0	52.0	49.0	60.0	1.0
50%	58.0	62.0	58.0	65.0	1.0
75%	60.0	64.0	63.0	80.0	1.0
max	80.0	82.0	83.0	80.0	1.0

```
pokemon_df.describe()
```

Selección de Características

income	age	favorite color
--------	-----	----------------

10000	18	Black
-------	----	-------

50000	47	Blue
-------	----	------

20000	40	Blue
-------	----	------

30000	29	Green
-------	----	-------

20000	22	Purple
-------	----	--------

Selección de Características

income	age	favorite color
10000	18	Black
50000	47	Blue
20000	40	Blue
30000	29	Green
20000	22	Purple

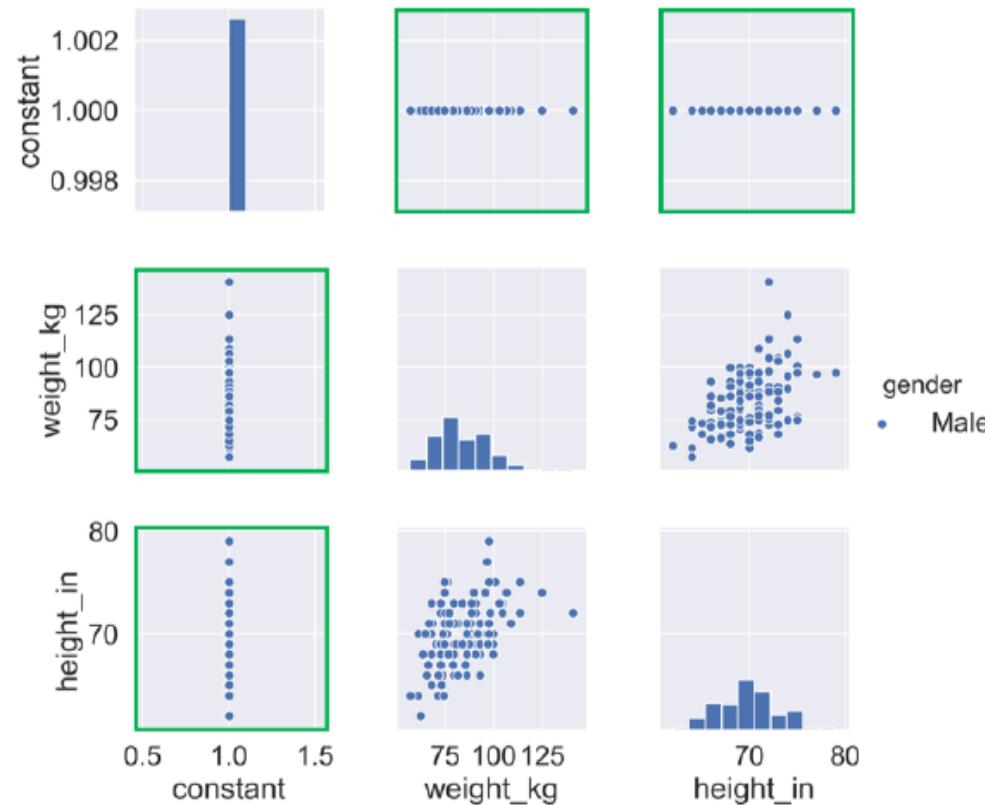


income	age
10000	18
50000	47
20000	40
30000	29
20000	22

```
insurance_df.drop('favorite color', axis=1)
```

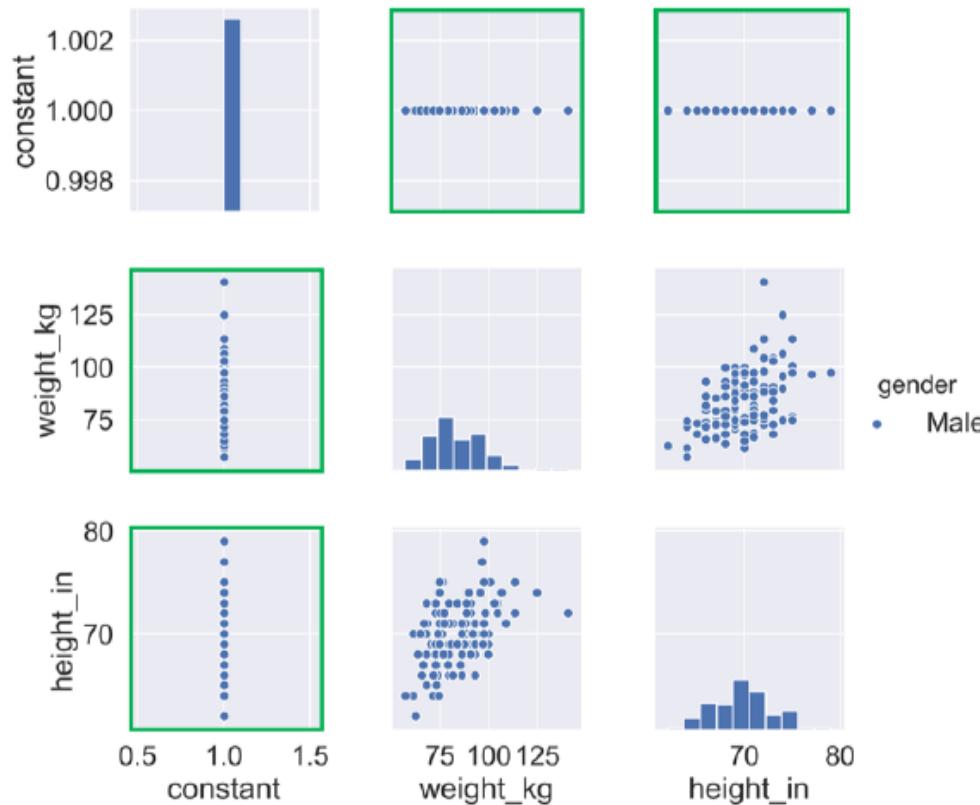
Selección de Características a partir del análisis de los datos

```
sns.pairplot(ansur_df, hue="gender", diag_kind='hist')
```

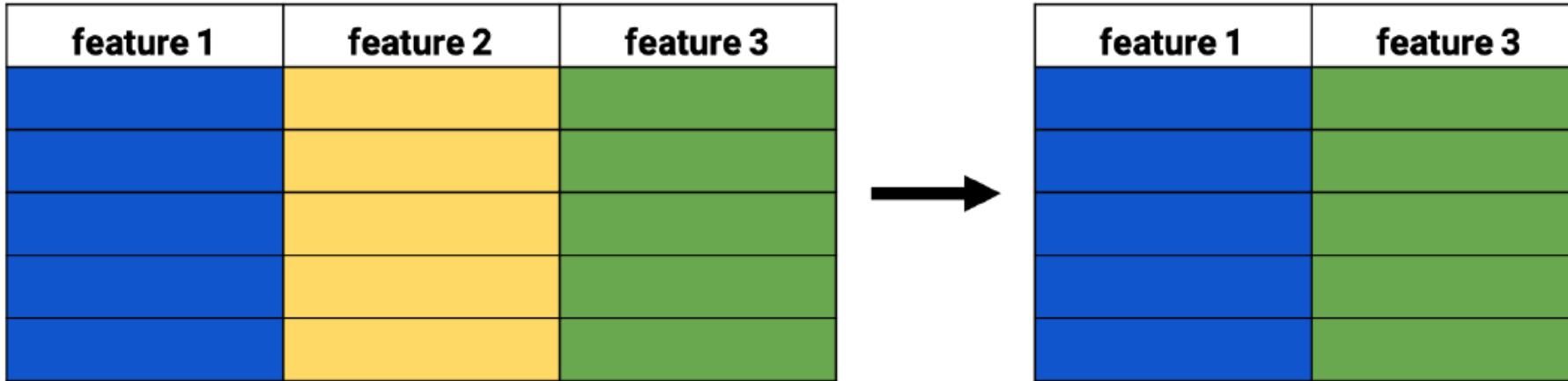


Selección de Características a partir del análisis de los datos

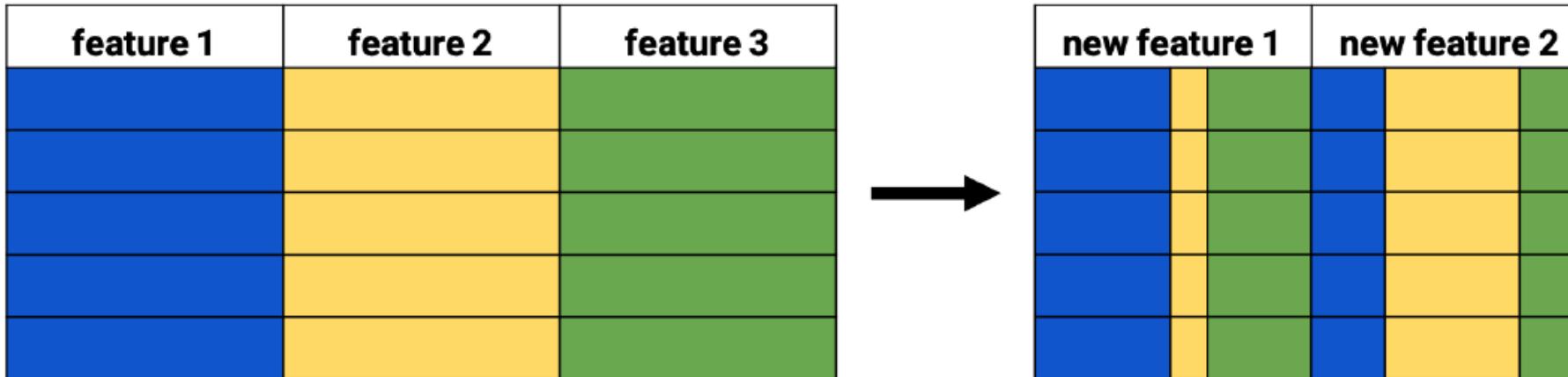
```
sns.pairplot(ansur_df, hue="gender", diag_kind='hist')
```



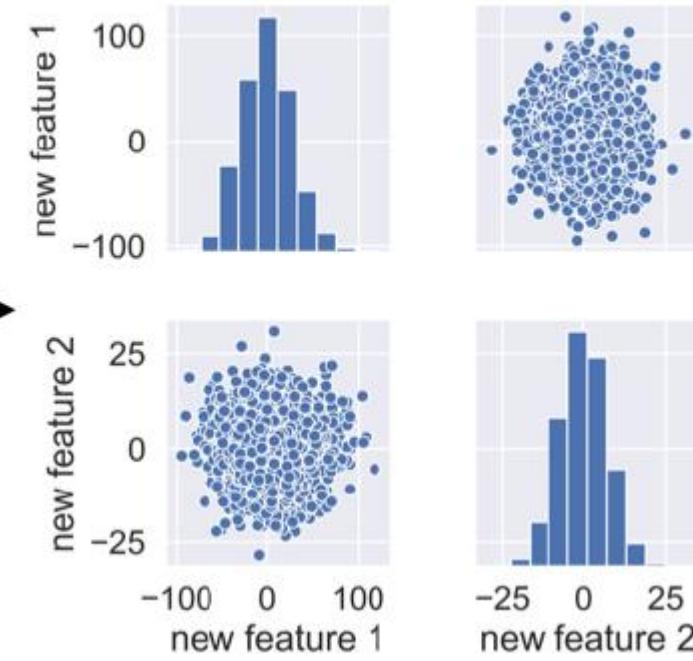
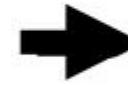
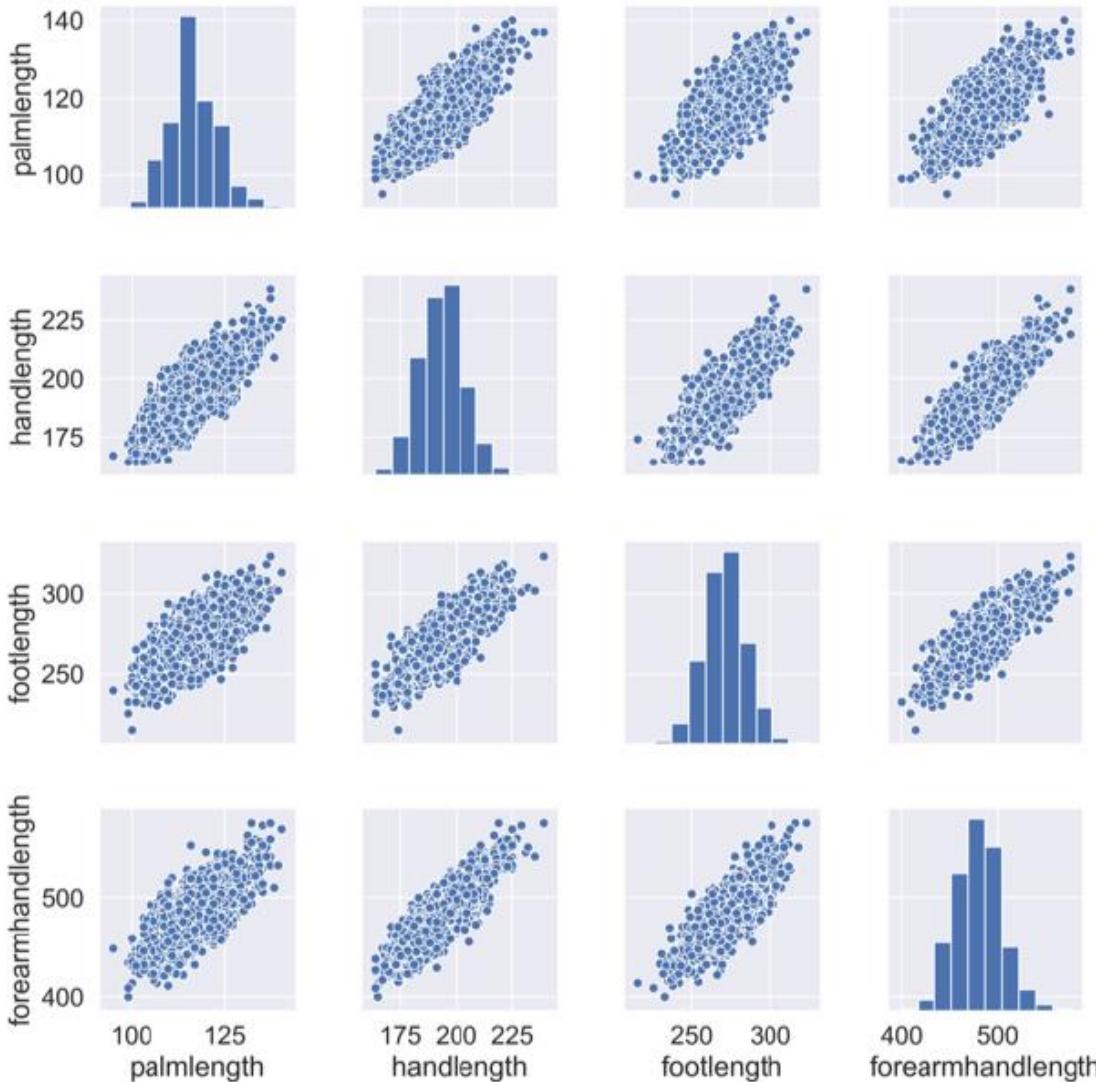
Selección de Características



Extracción de Características



Extracción de Características a partir del análisis de los datos



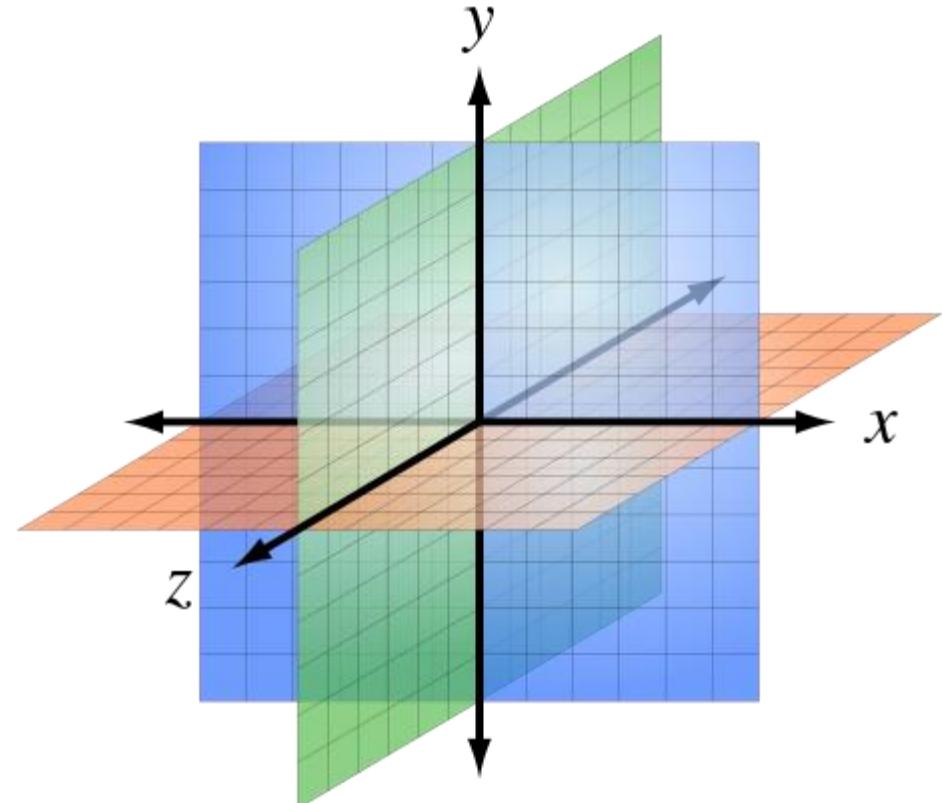
1

30

La maldición de la dimensionalidad (The Curse of Dimensionality)

Cuando el número de características es muy grande en relación con el número de observaciones en su conjunto de datos, ciertos algoritmos luchan por entrenar modelos efectivos.

Esto se llama la "**Maldición de la dimensionalidad**", y es especialmente relevante para los algoritmos de agrupamiento que se basan en cálculos de distancia.



La maldición de la dimensionalidad

(Analogía)

“Digamos que tiene una línea recta de 100 metros de largo y dejó caer una moneda en algún lugar. No sería muy difícil de encontrar. Caminas a lo largo de la línea y te lleva dos minutos.

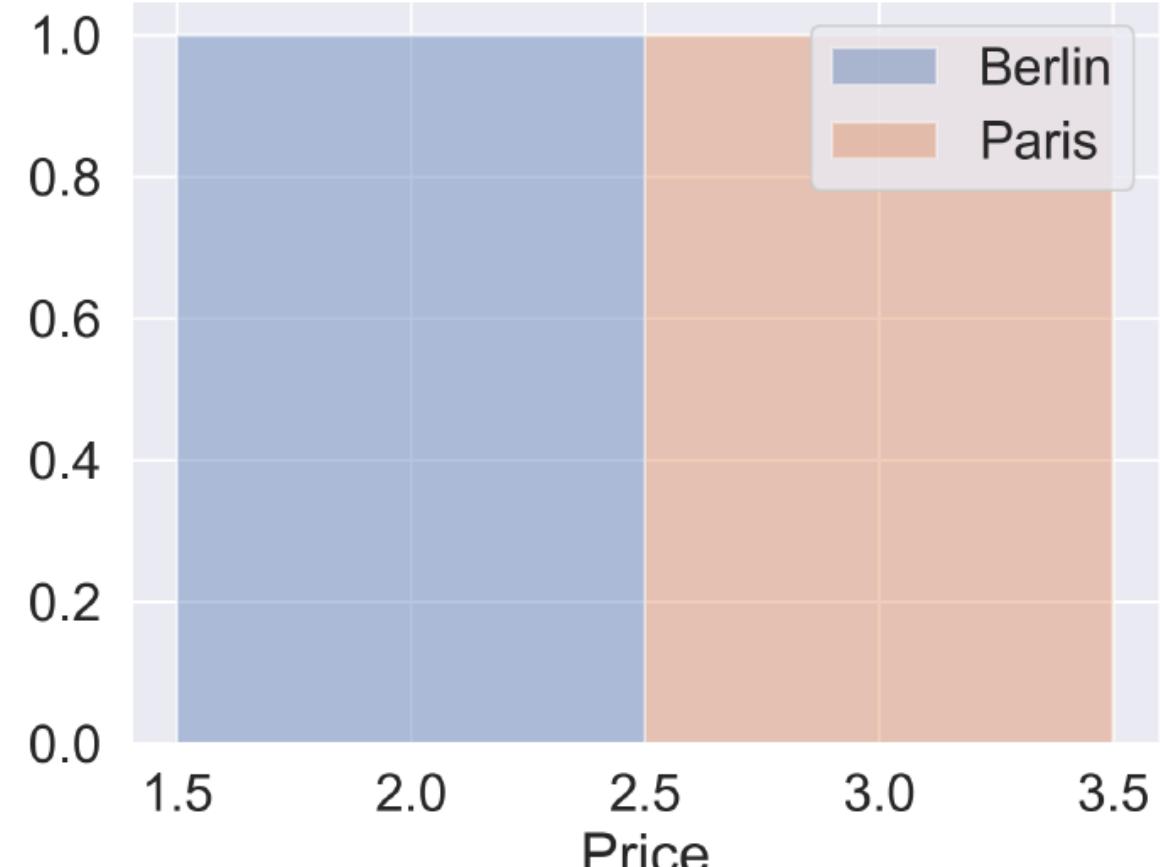
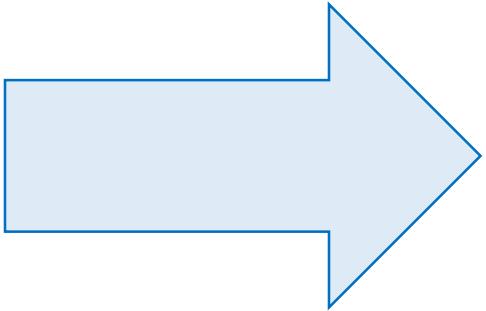
Ahora supongamos que tiene un cuadrado de 100 metros a cada lado y dejó caer un centavo en algún lugar. Sería bastante difícil, como buscar en dos campos de fútbol unidos. Podría llevar días.

Ahora un cubo de 100 metros de arista. Es como buscar en un edificio de 30 pisos del tamaño de un estadio de fútbol. Ugh

La dificultad de buscar en el espacio se vuelve mucho más difícil a medida que tienes más dimensiones.”

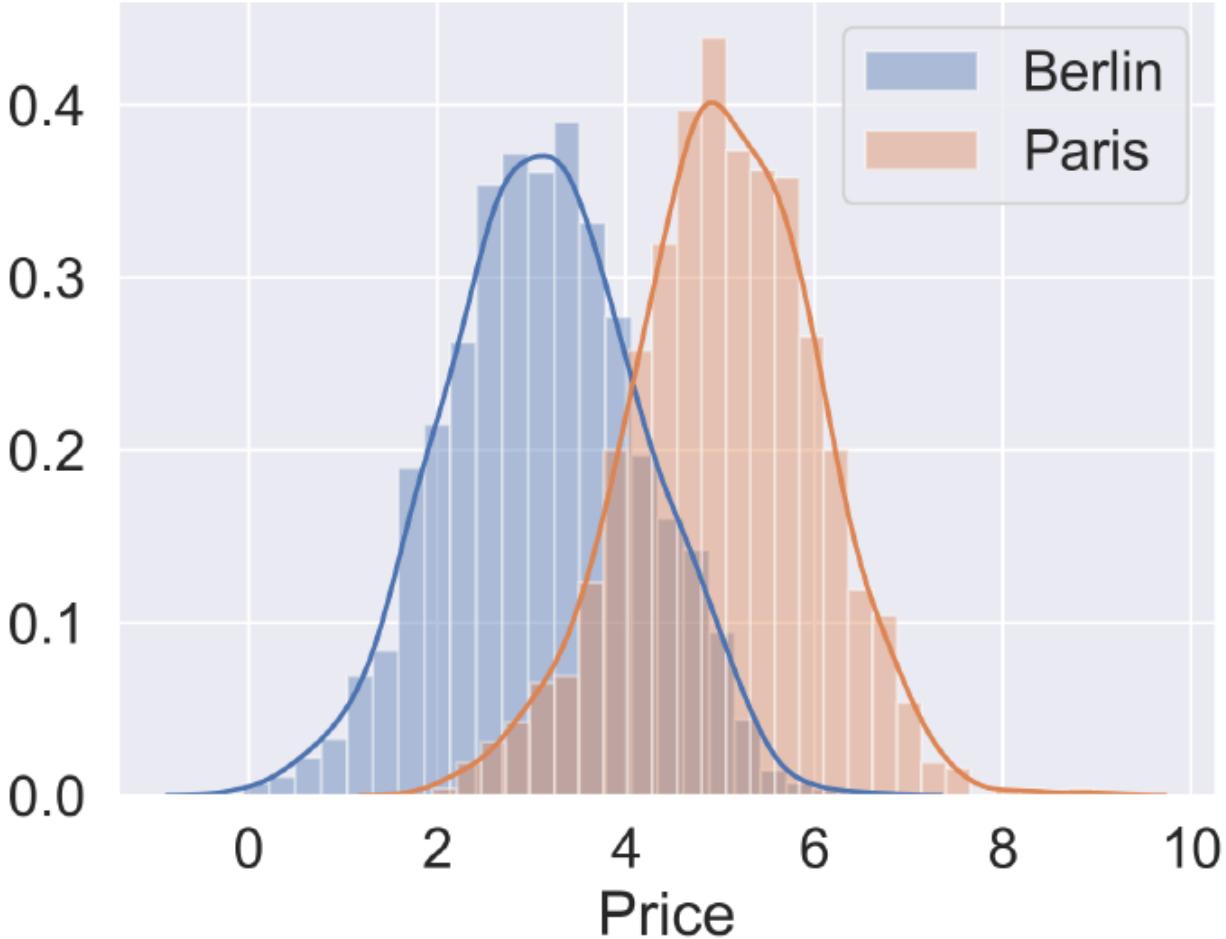
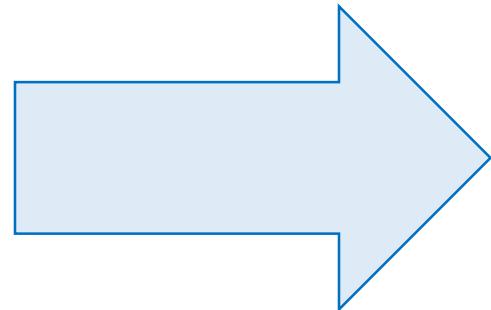
La maldición de la dimensionalidad (Ejemplo)

City	Price
Berlin	2
Paris	3



La maldición de la dimensionalidad (Ejemplo)

City	Price
Berlin	2.0
Berlin	3.1
Berlin	4.3
Paris	3.0
Paris	5.2
...	...



La maldición de la dimensionalidad

Construimos un Clasificador

Separate the feature we want to predict from the ones to train the model on.

```
y = house_df['City']  
  
X = house_df.drop('City', axis=1)
```

Perform a 70% train and 30% test data split

```
from sklearn.model_selection import train_test_split  
  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

La maldición de la dimensionalidad

Construimos un Clasificador

Create a Support Vector Machine Classifier and fit to training data

```
from sklearn.svm import SVC  
  
svc = SVC()  
  
svc.fit(X_train, y_train)
```

La maldición de la dimensionalidad

Evaluamos la calidad de la Predicción

```
from sklearn.metrics import accuracy_score  
  
print(accuracy_score(y_test, svc.predict(X_test)))
```

0.826

```
print(accuracy_score(y_train, svc.predict(X_train)))
```

0.832

La maldición de la dimensionalidad

Incrementamos dimensiones (Características)

City	Price	n_floors	n_bathroom	surface_m2
Berlin	2.0	1	1	190
Berlin	3.1	2	1	187
Berlin	4.3	2	2	240
Paris	3.0	2	1	170
Paris	5.2	2	2	290
...



La maldición de la dimensionalidad

Se debe asegurar que existe una cantidad significativa de ejemplos por cada dimensión adicional que se maneja en nuestro problema.

De otra manera puede haber **sobreajuste (Overfitting)**. No se garantiza la generalización.

Por esto es importante identificar de que manera podemos eliminar características (**feature selection**) que poco aporten en nuestra búsqueda de nuestro modelo de reconocimiento de patrones.

Características (Dimensiones) con valores faltantes o Poca Varianza

En algunos casos debemos identificar estrategias para completar datos faltantes. (Usar el valor medio o por ejemplo (**mean**) - El Valor más frecuente (**mode**))

Para los valores que presenten poca varianza podemos eliminarlos. (Para esto es importante **normalizar la varianza** en todo el dataset)

También es muy útil identificar el valor de **correlación** en los datos.

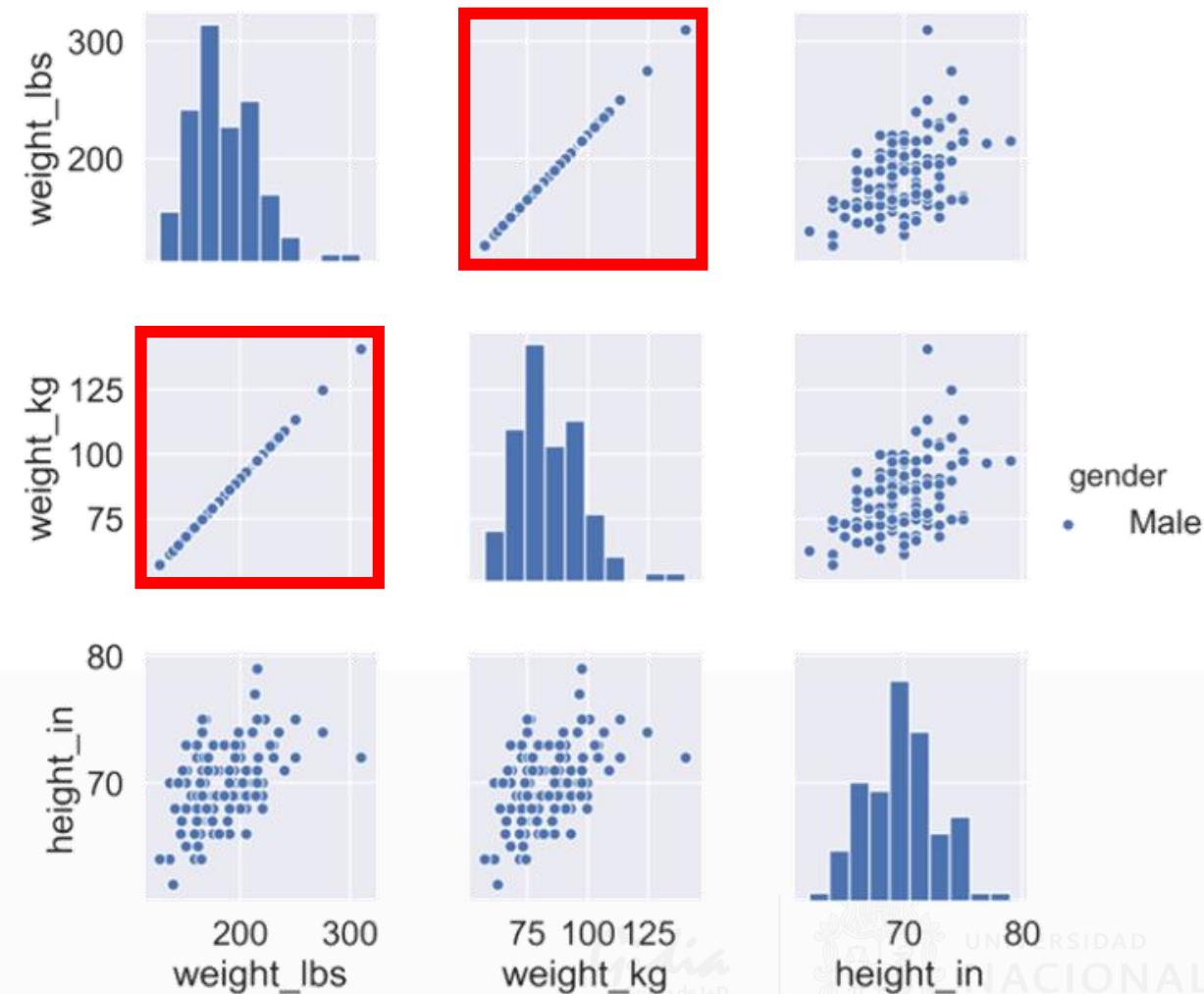
Correlación de Valores

Correlación en parejas

```
sns.pairplot(ansur, hue="gender")
```

$$Cov(X, Y) = \frac{\sum_1^n (x_i - \bar{x})(y_i - \bar{y})}{n}$$

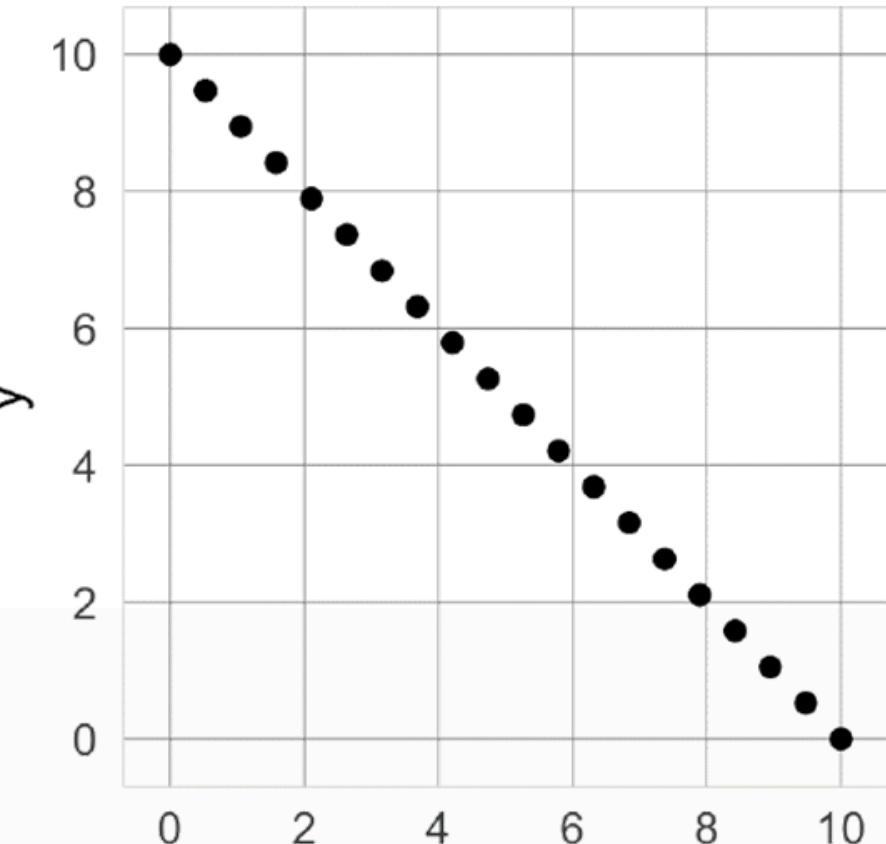
$$\rho_{xy} = \frac{Cov_{xy}}{\sigma_x \sigma_y}$$



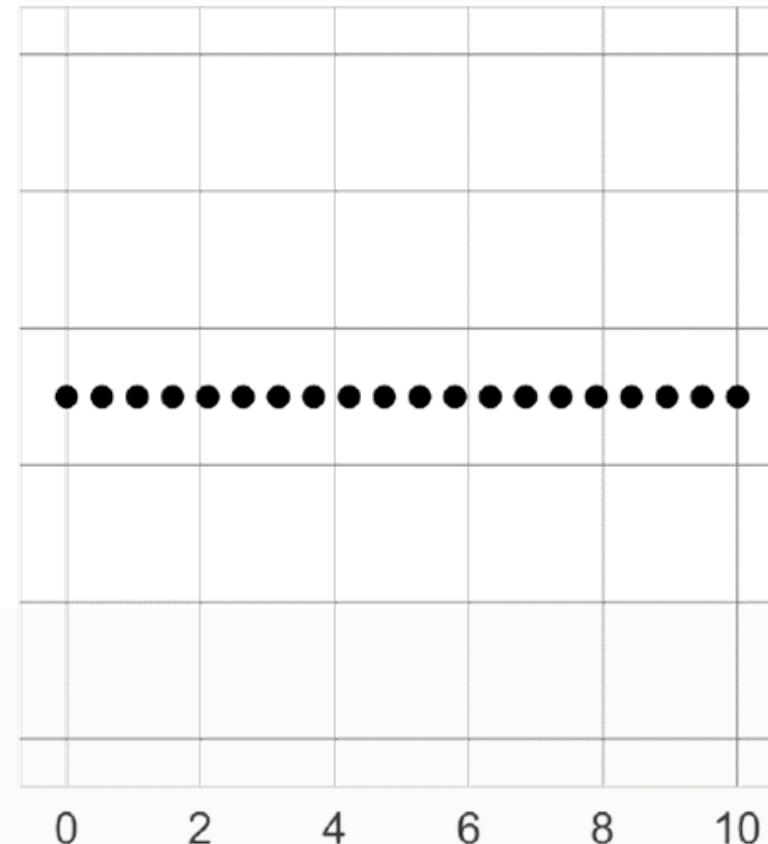
Correlación de Valores

Coeficiente de correlación

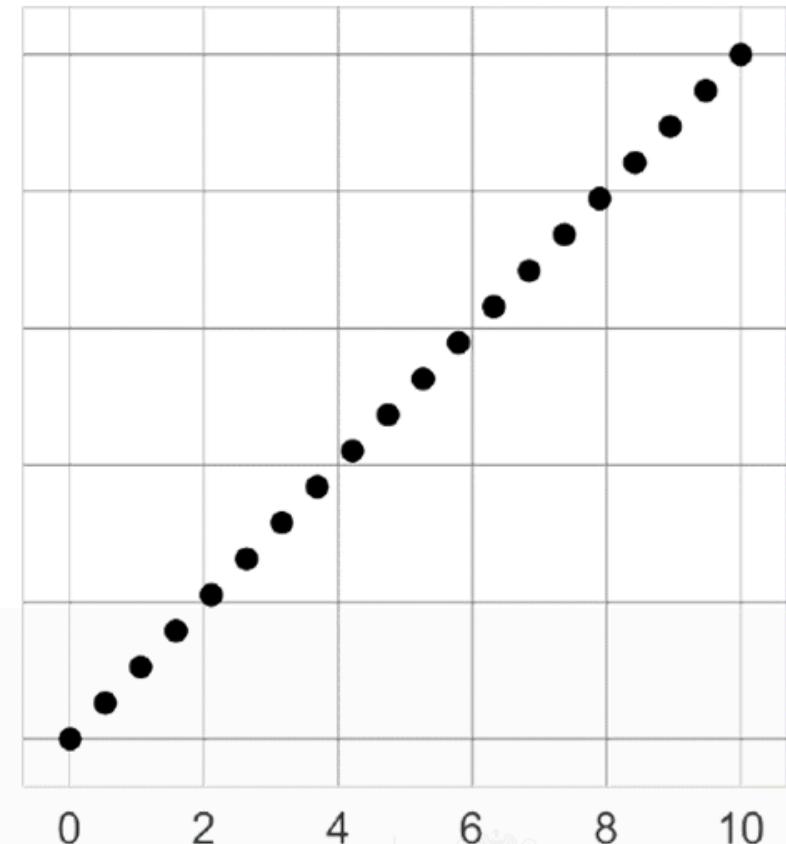
$r = -1$



$r = 0$



$r = 1$



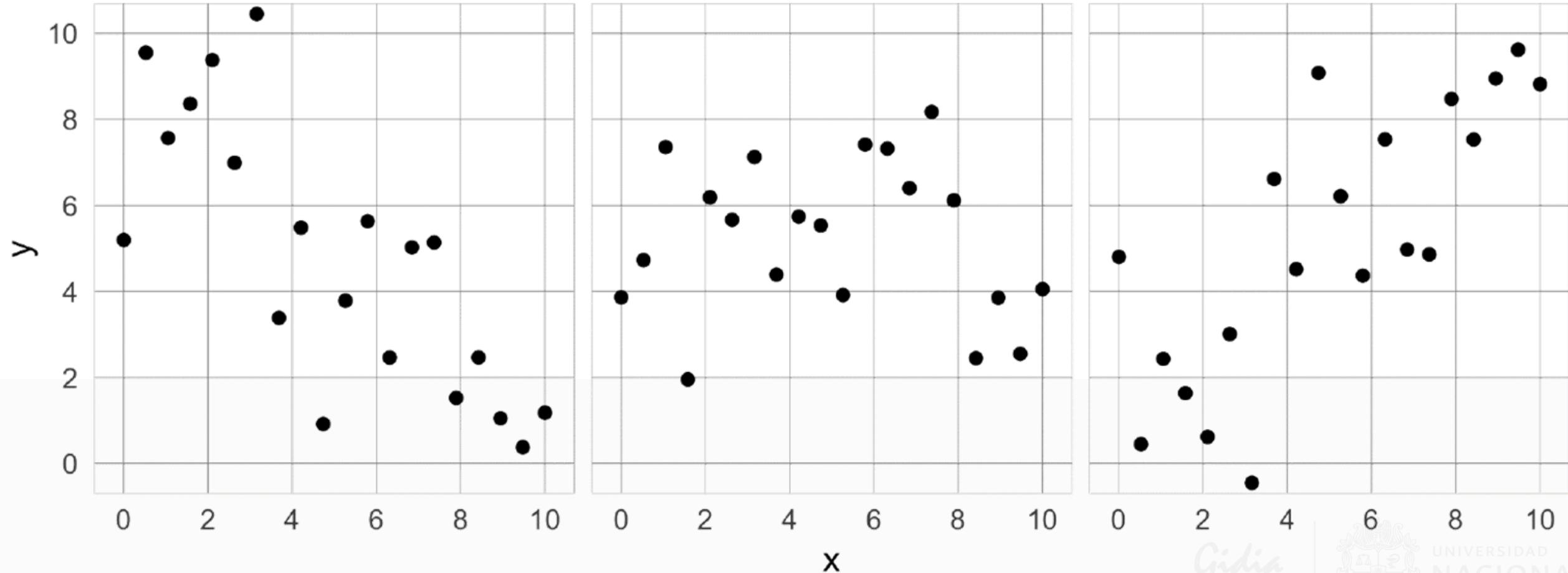
Correlación de Valores

Coeficiente de correlación

$$r = -0.88$$

$$r = 0.05$$

$$r = 0.88$$



Correlación de Valores

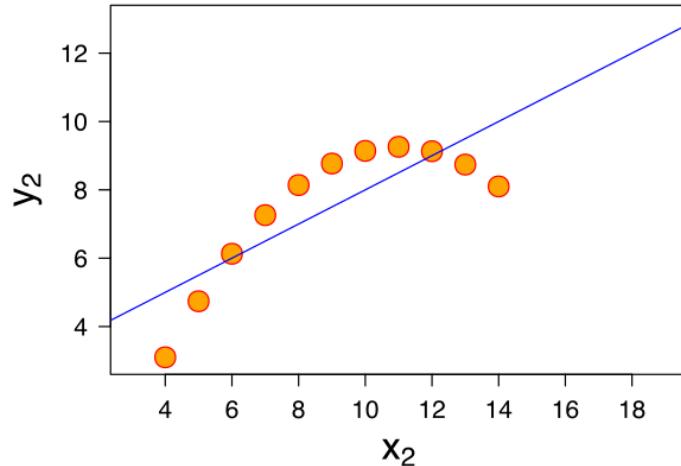
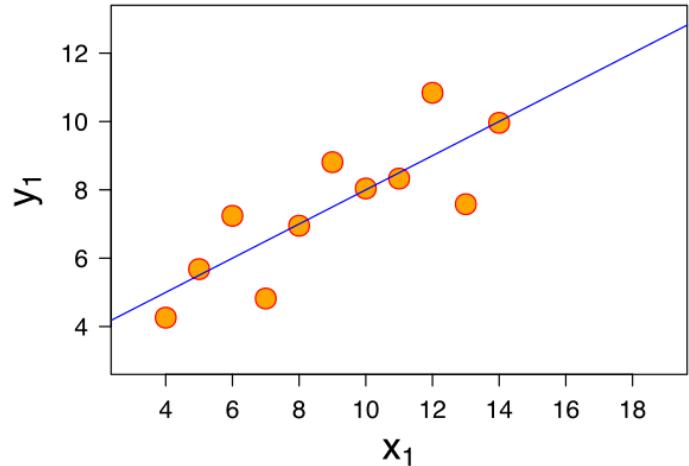
Matriz de Correlación

```
weights_df.corr()
```

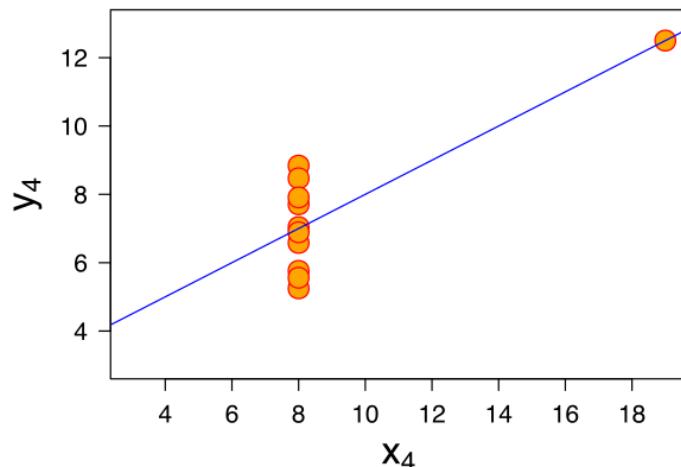
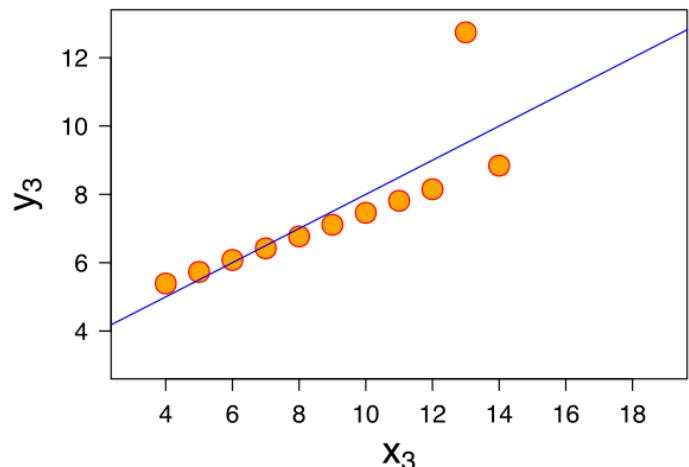
	weight_lbs	weight_kg	height_in
weight_lbs	1.00	1.00	0.47
weight_kg	1.00	1.00	0.47
height_in	0.47	0.47	1.00

Correlación de Valores

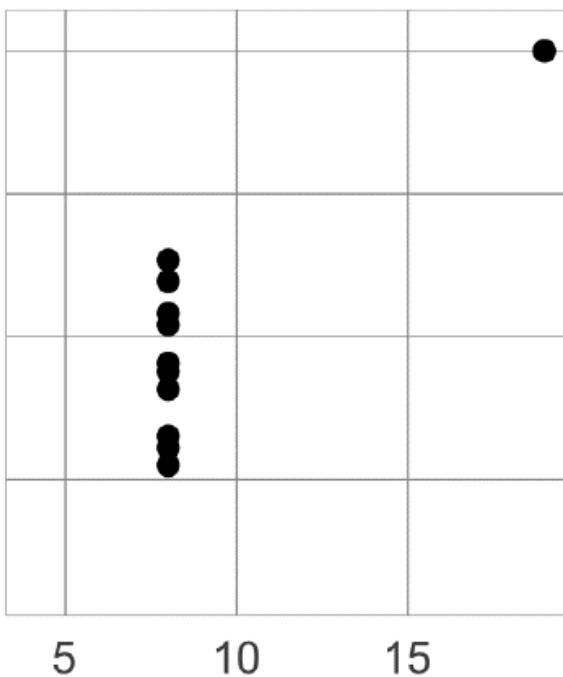
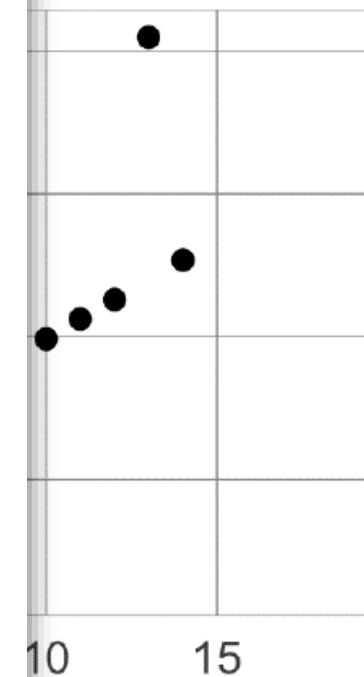
Problemas con la Correlación



= 0.82



$r = 0.82$

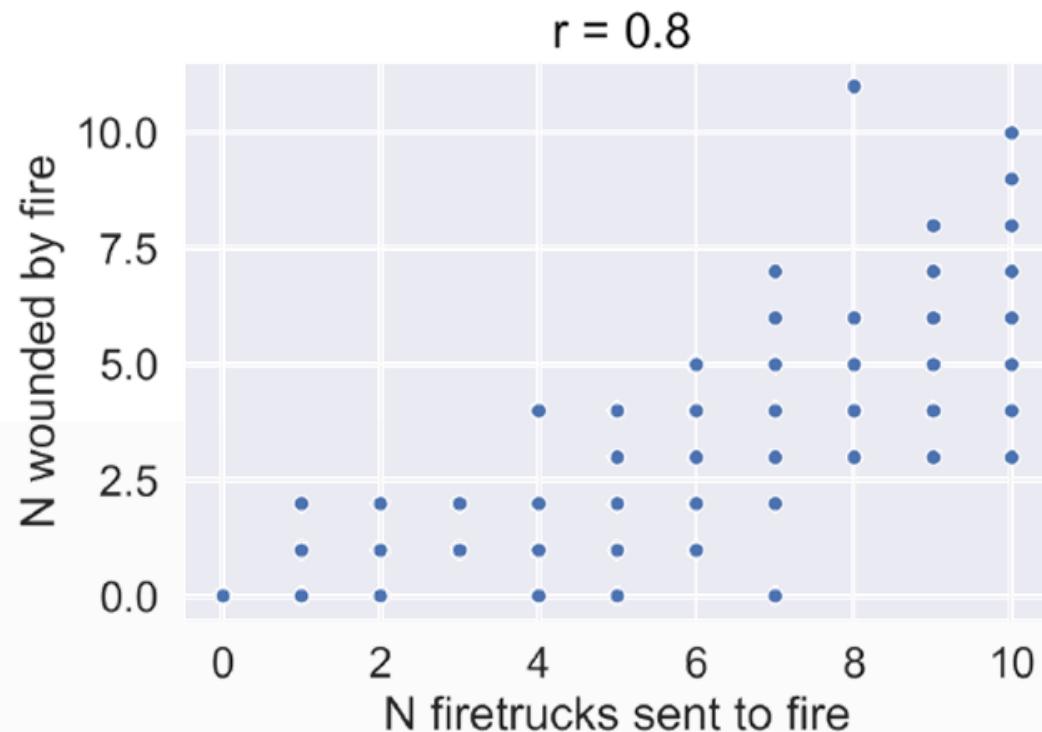


Cuarteto de Anscombe

Correlación de Valores

Problemas con la Correlación

```
sns.scatterplot(x="N firetrucks sent to fire",  
y="N wounded by fire", data=fire_df)
```



Causación

Laboratorios

<https://github.com/srobles05/3008422-AprendizajeDeMaquinas/>

Reemplazo de valores faltantes y Filtrado de Baja Varianza

Considere una variable en nuestro conjunto de datos donde todas las observaciones tienen el mismo valor, digamos 1. Si usamos esta variable, ¿cree que puede mejorar el modelo que construiremos? La respuesta es no, porque esta variable tendrá cero varianza.

Entonces, necesitamos calcular la varianza de cada variable que se nos da. Luego, eliminar las variables que tienen una varianza baja en comparación con otras variables en nuestro conjunto de datos. La razón para hacerlo, como mencionamos anteriormente, es que las variables con una varianza baja no afectarán la variable objetivo.

```
# import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
train=pd.read_csv("Train_data.csv")
# To show rows and columns
train.shape
```

(8523, 12)

Primero, identificamos los datos faltantes en cada dimensión

```
train.isna().sum()
```

Item_Identifier	0
Item_Weight	1463
Item_Fat_Content	0
Item_Visibility	0
Item_Type	0
Item_MRP	0
Outlet_Identifier	0
Outlet_Establishment_Year	0
Outlet_Size	2410
Outlet_Location_Type	0
Outlet_Type	0
Item_Outlet_Sales	0
dtype:	int64

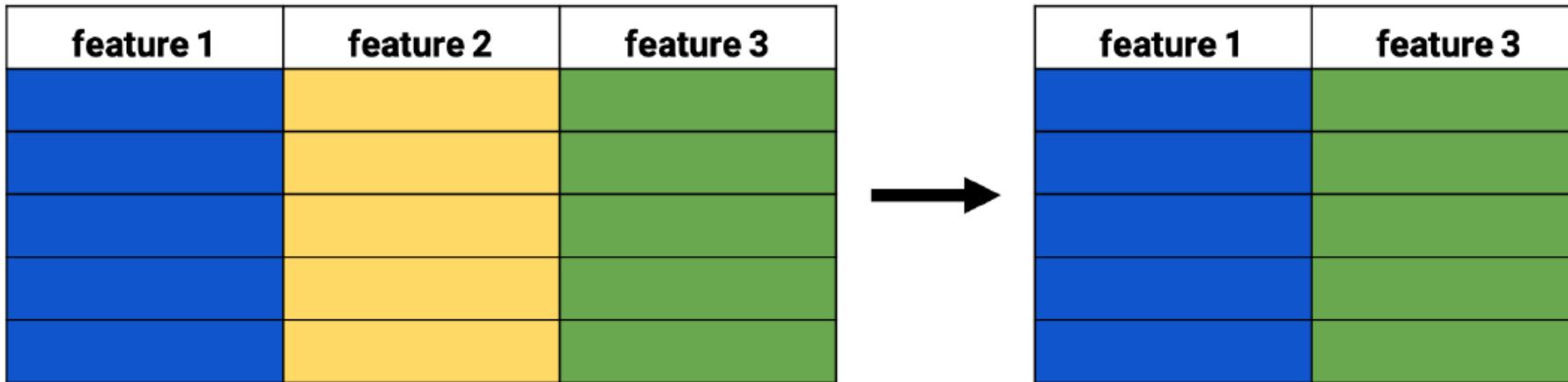
Reemplazemos los valores faltantes en la columna Item_Weight usando el valor medio de las observaciones conocidas de Item_Weight. Para la columna Outlet_Size, usaremos el modo de los valores Outlet_Size conocidos para reemplazar los valores faltantes:

```
train['Outlet_Size'].mode()[0]
'Medium'
```

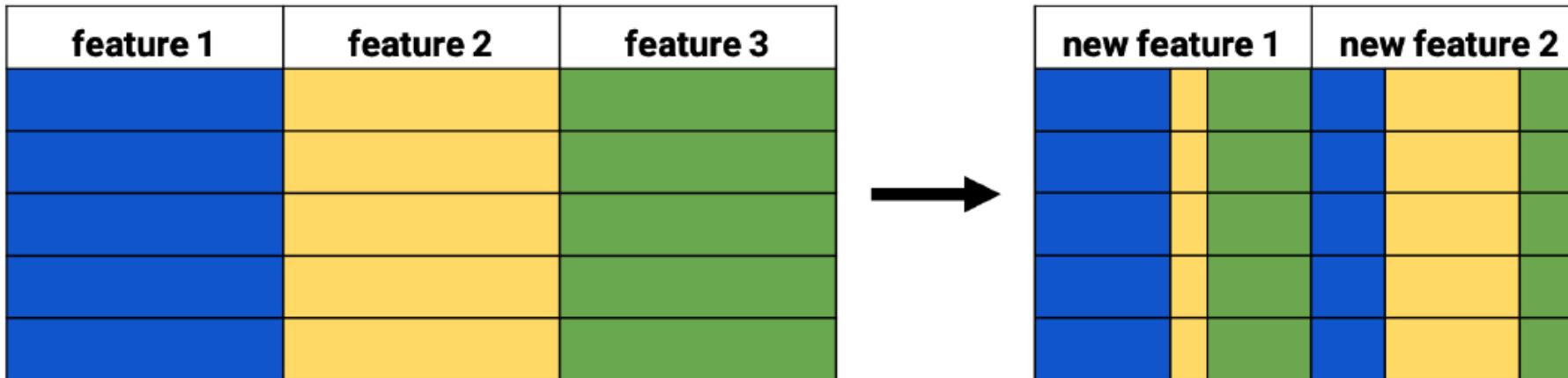
Basic_labs



Selección de Características



Extracción de Características



Extracción de Características

Feature generation - BMI

```
df_body['BMI'] = df_body['Weight kg'] / df_body['Height m'] ** 2
```

Weight kg	Height m	BMI
81.5	1.776	25.84
72.6	1.702	25.06
92.9	1.735	30.86

Extracción de Características

Feature generation - BMI

```
df_body.drop(['Weight kg', 'Height m'], axis=1)
```

BMI
25.84
25.06
30.86

Extracción de Características

Feature generation - averages

left leg mm	right leg mm
882	885
870	869
901	900

```
leg_df['leg_mm'] = leg_df[['right leg mm', 'left leg mm']].mean(axis=1)
```

Extracción de Características

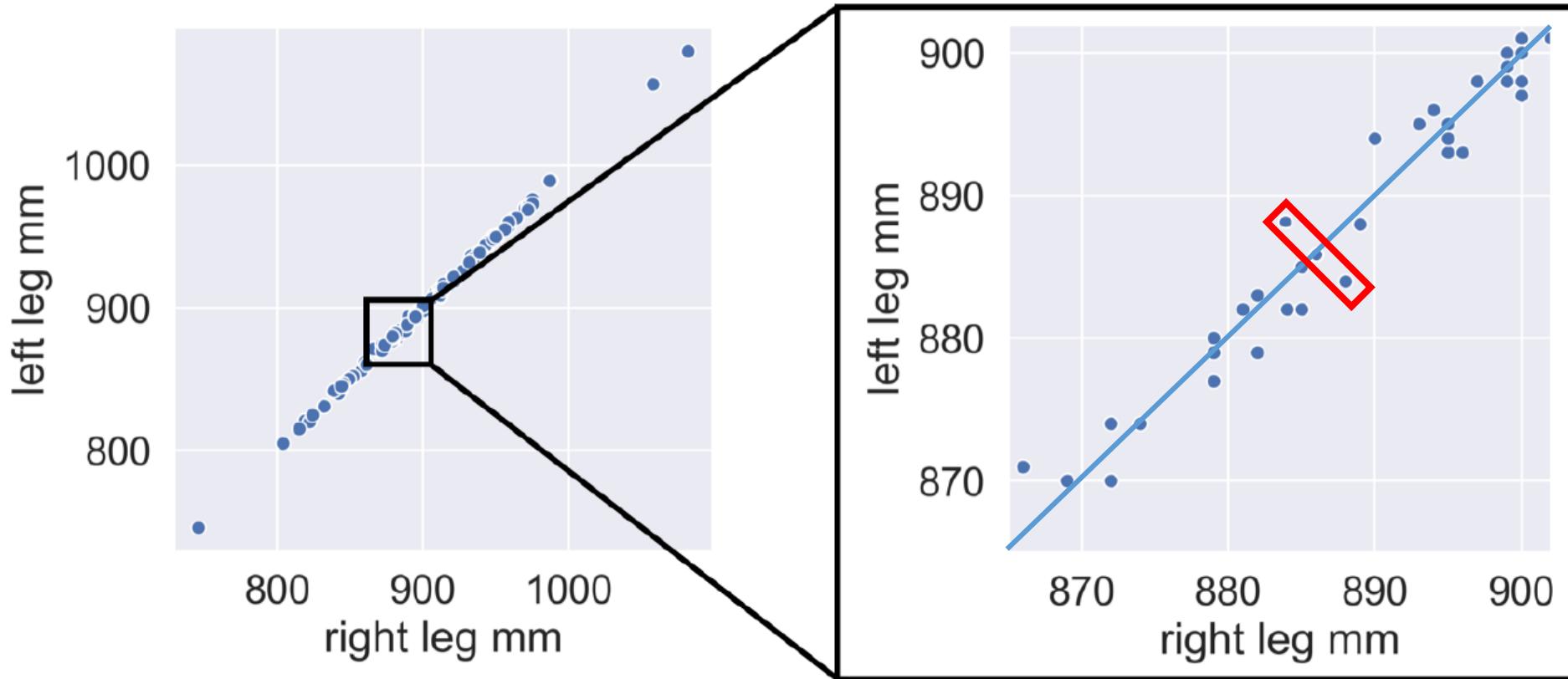
Feature generation - averages

```
leg_df.drop(['right leg mm', 'left leg mm'], axis=1)
```

leg mm
883.5
869.5
900.5

Extracción de Características

Cost of taking the average



Extracción de Características - PCA

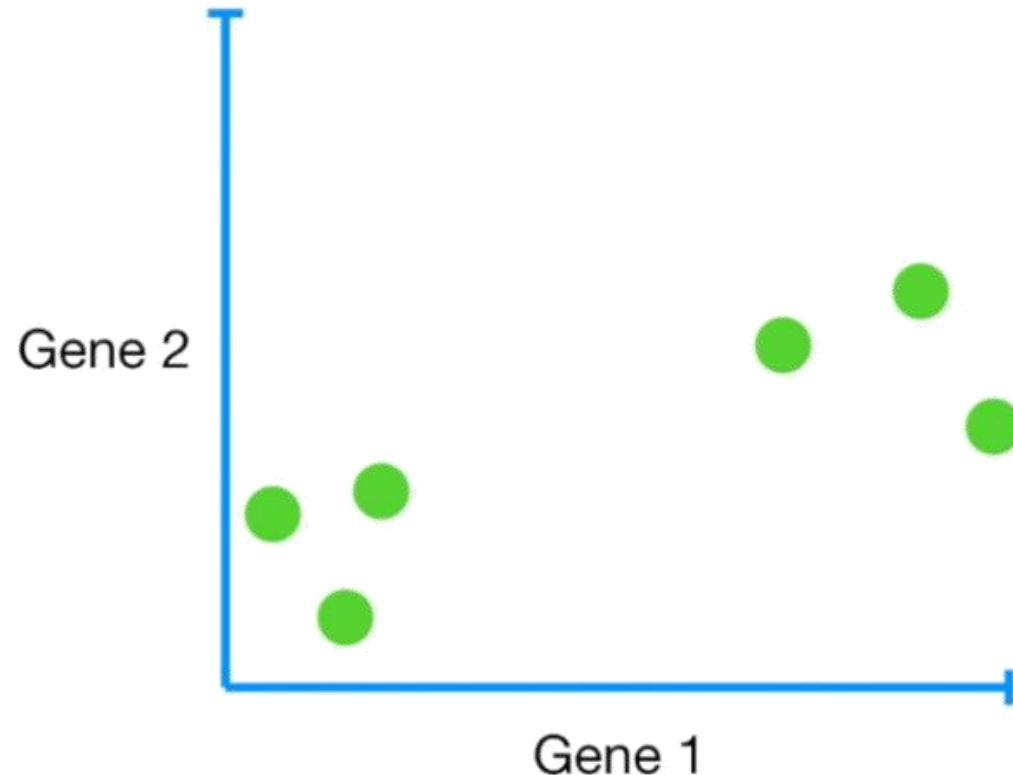
	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

To understand what PCA does and how it works, let's go back to the dataset that only had 2 genes...

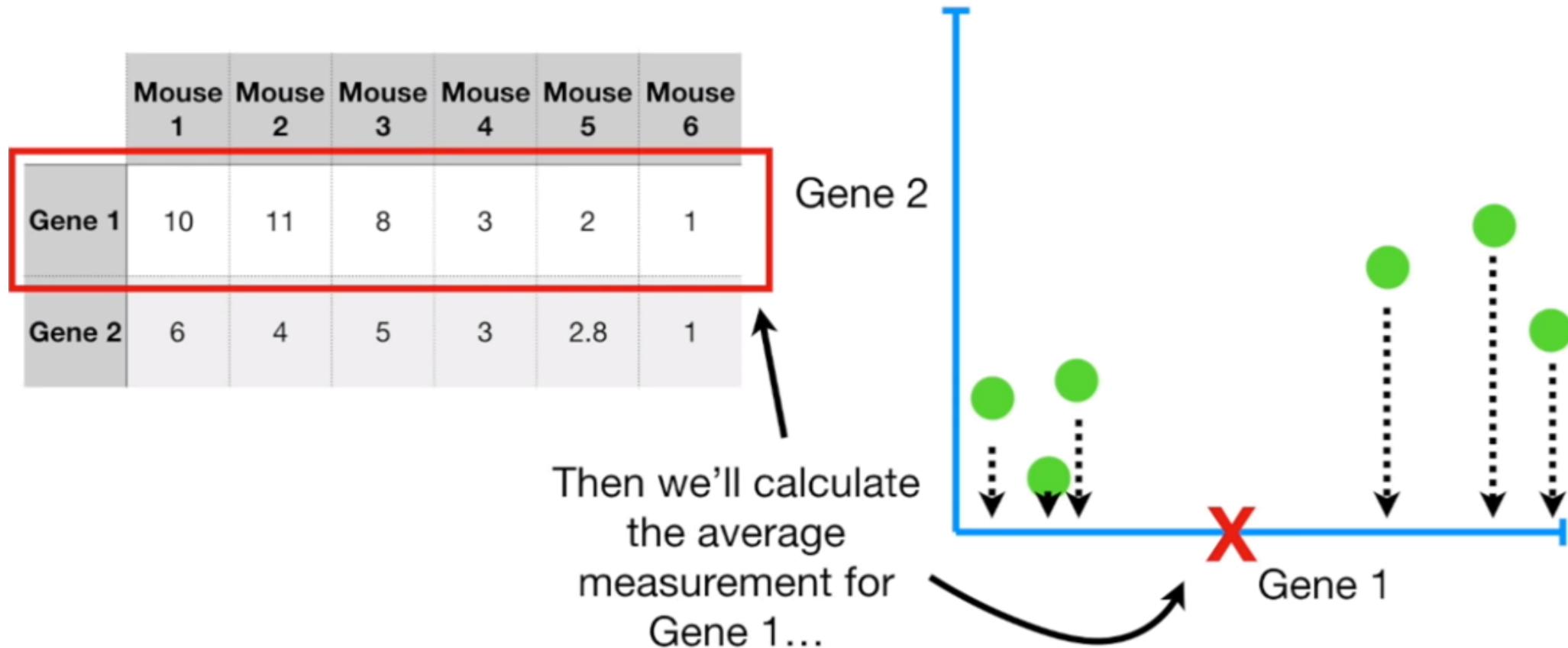
Extracción de Características - PCA

	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1

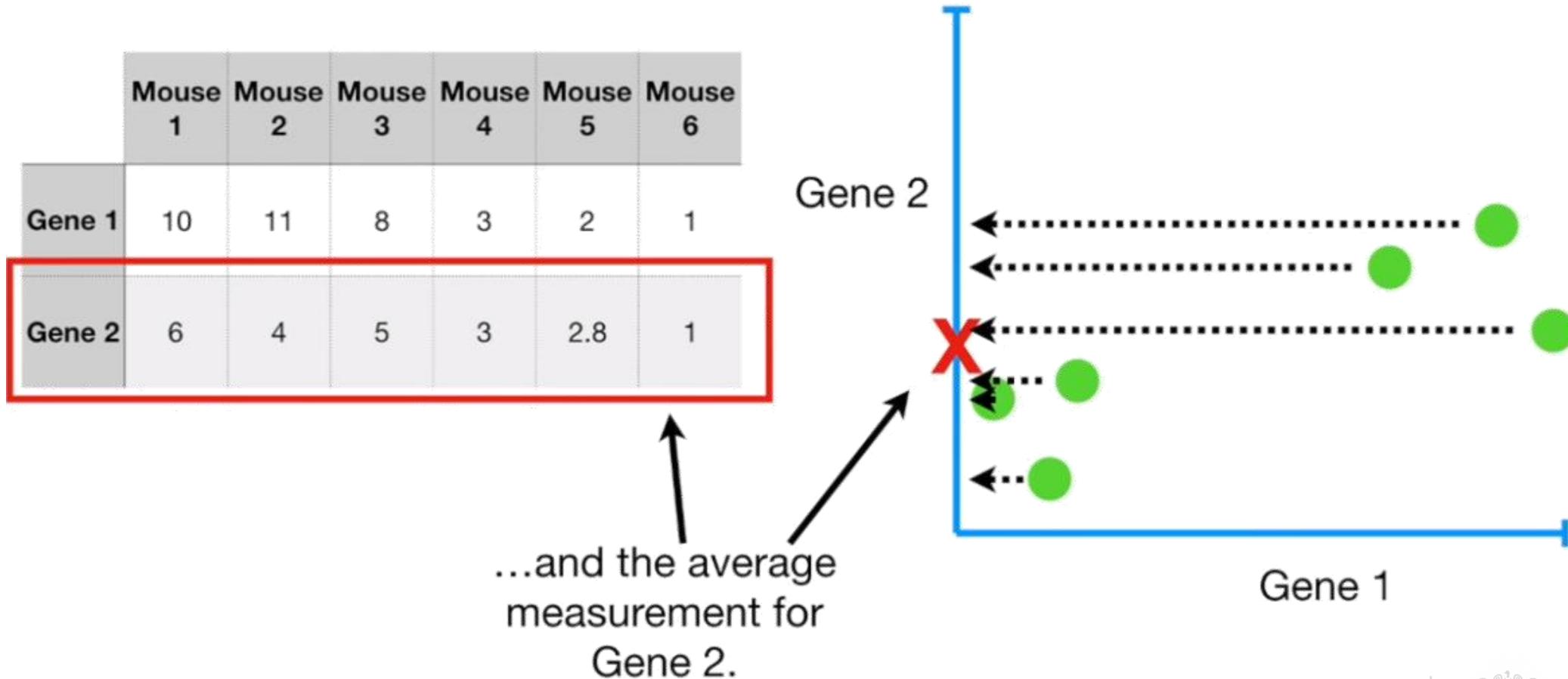
We'll start by plotting the data...



Extracción de Características - PCA

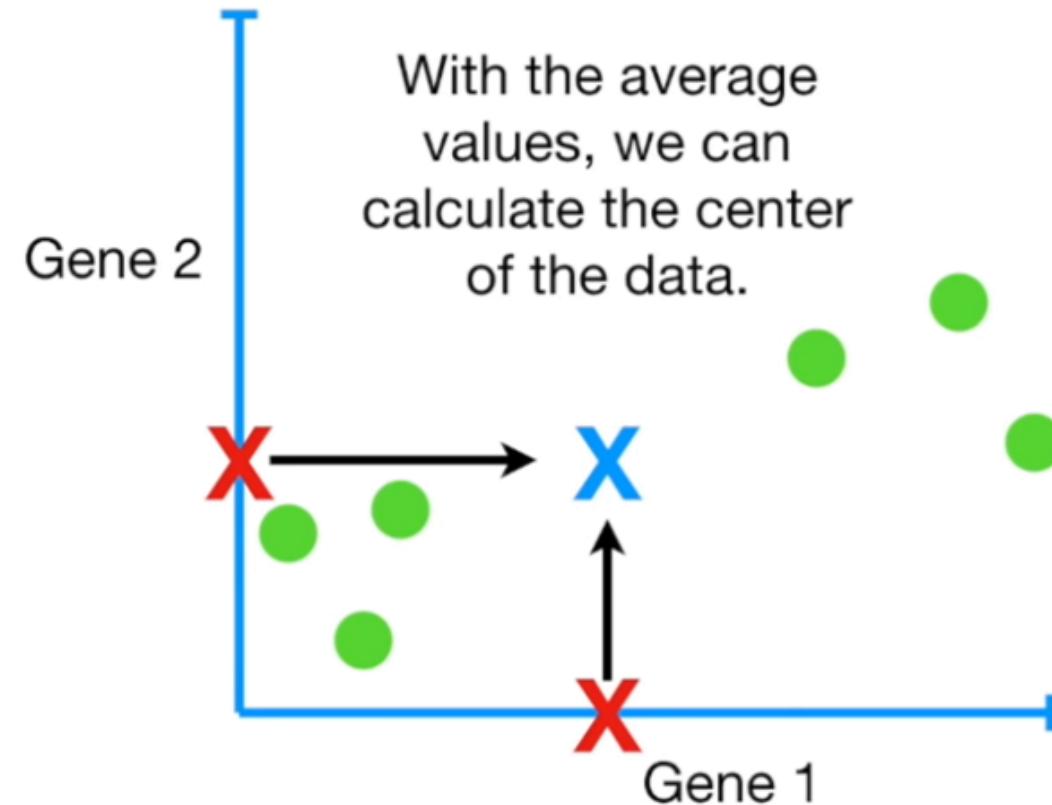


Extracción de Características - PCA

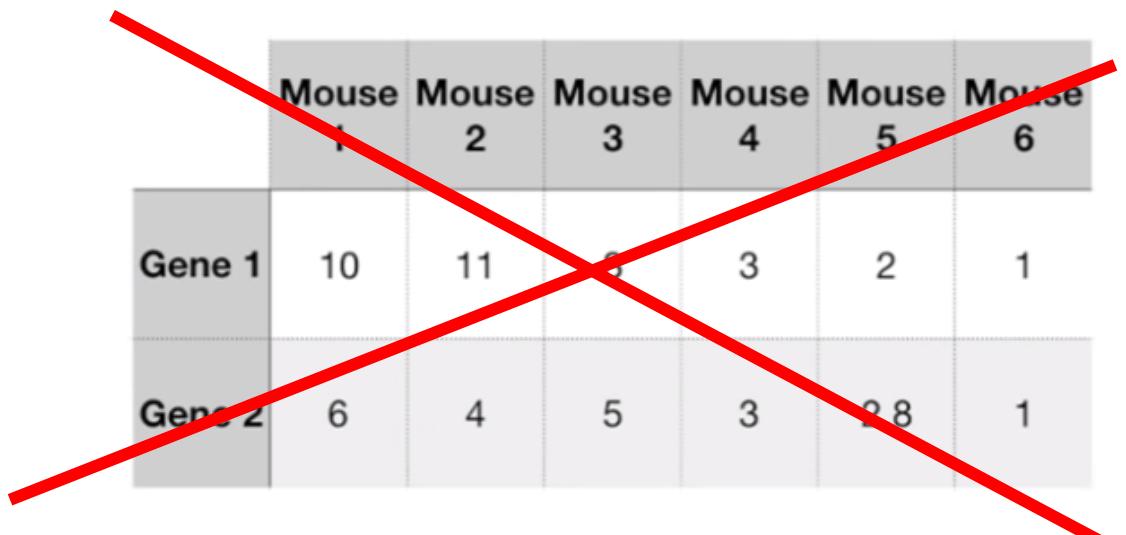


Extracción de Características - PCA

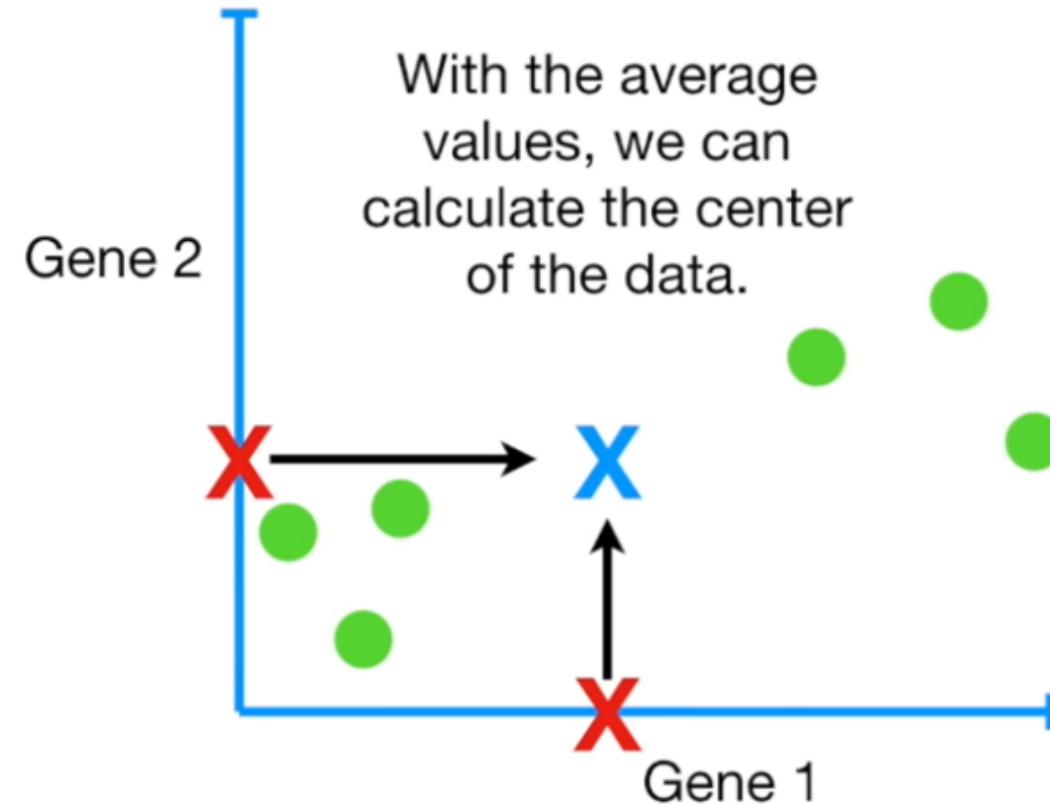
	Mouse 1	Mouse 2	Mouse 3	Mouse 4	Mouse 5	Mouse 6
Gene 1	10	11	8	3	2	1
Gene 2	6	4	5	3	2.8	1



Extracción de Características - PCA

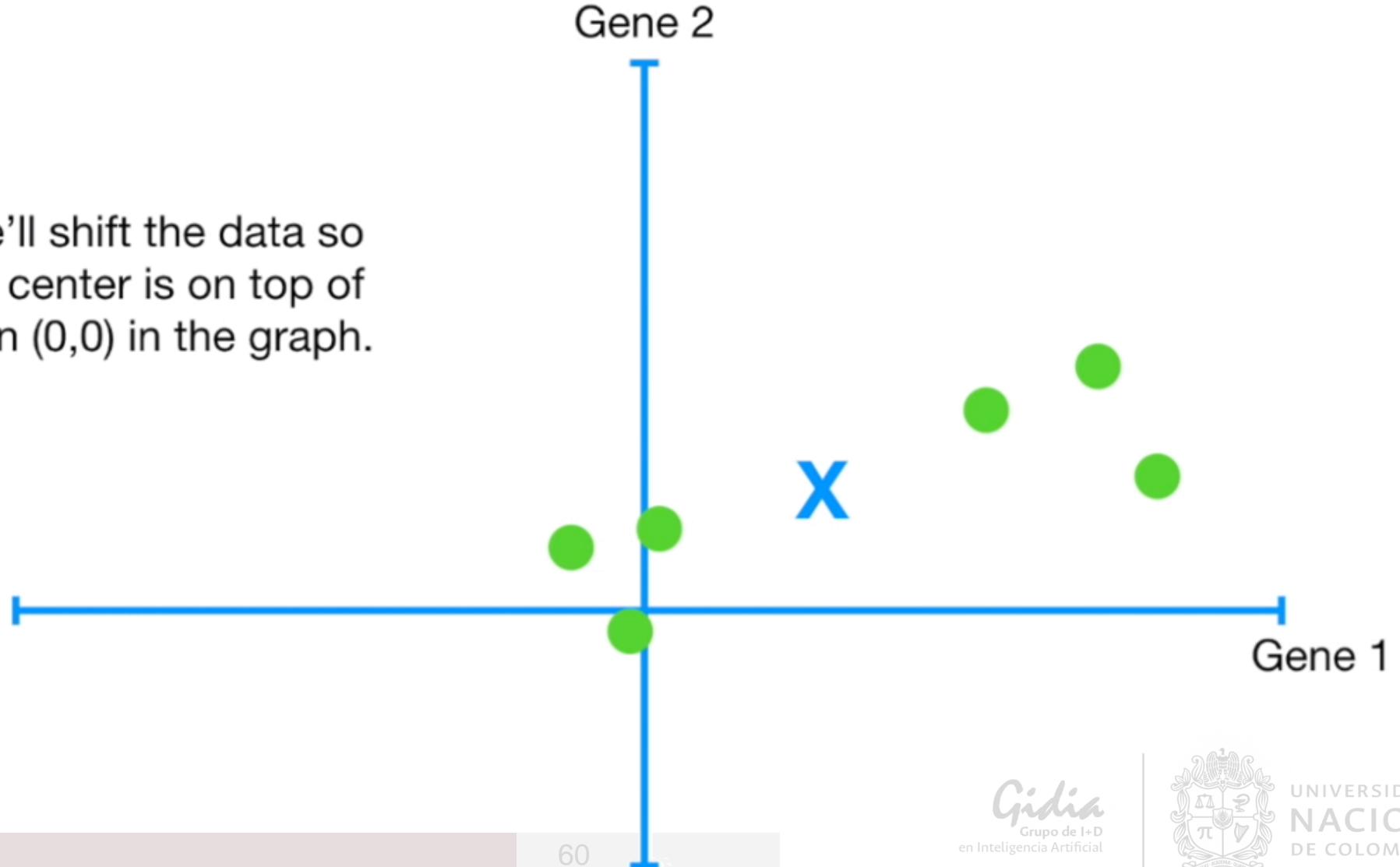


	Mouse					
	1	2	3	4	5	6
Gene 1	10	11	9	3	2	1
Gene 2	6	4	5	3	2.8	1



Extracción de Características - PCA

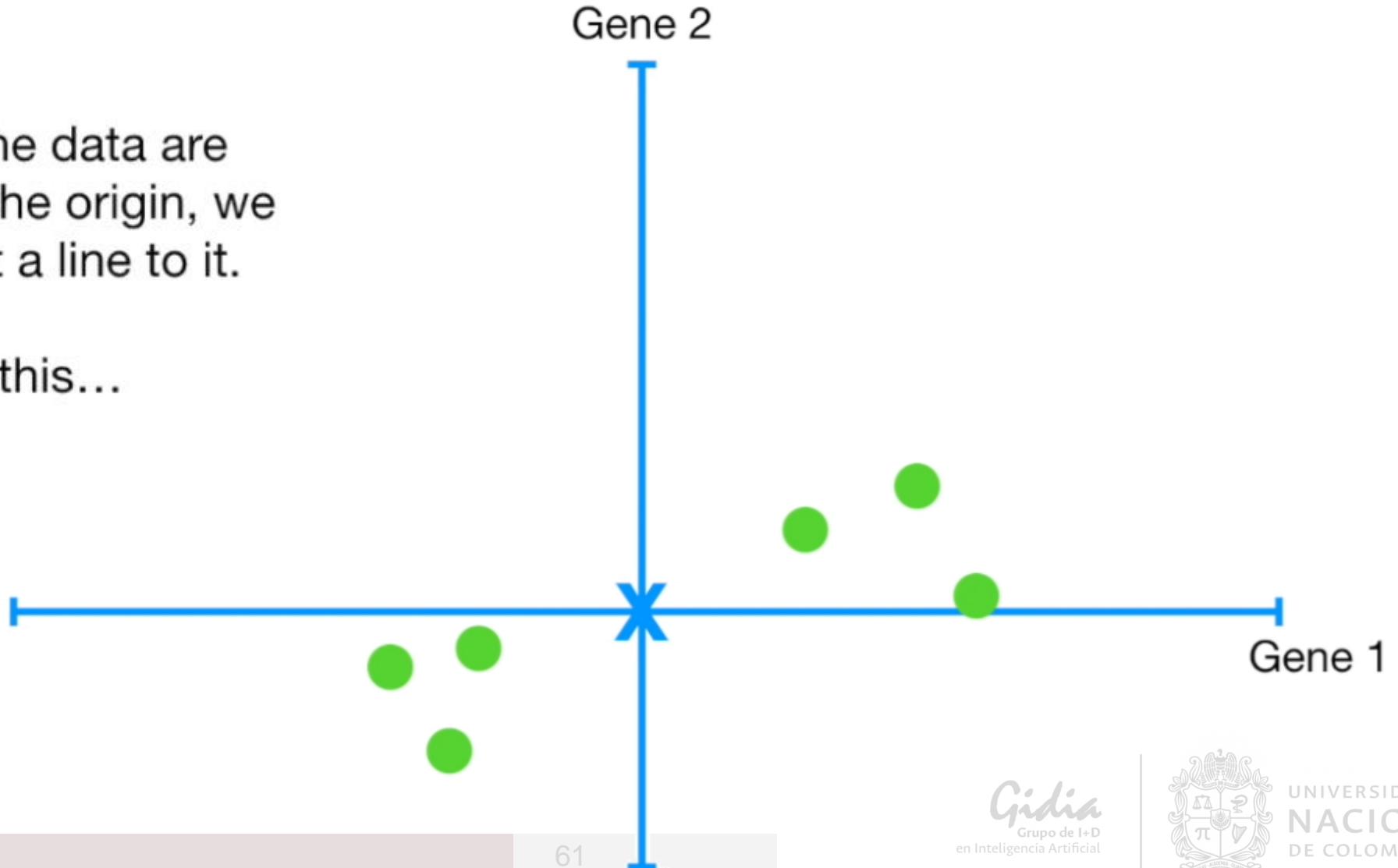
Now we'll shift the data so that the center is on top of the origin (0,0) in the graph.



Extracción de Características - PCA

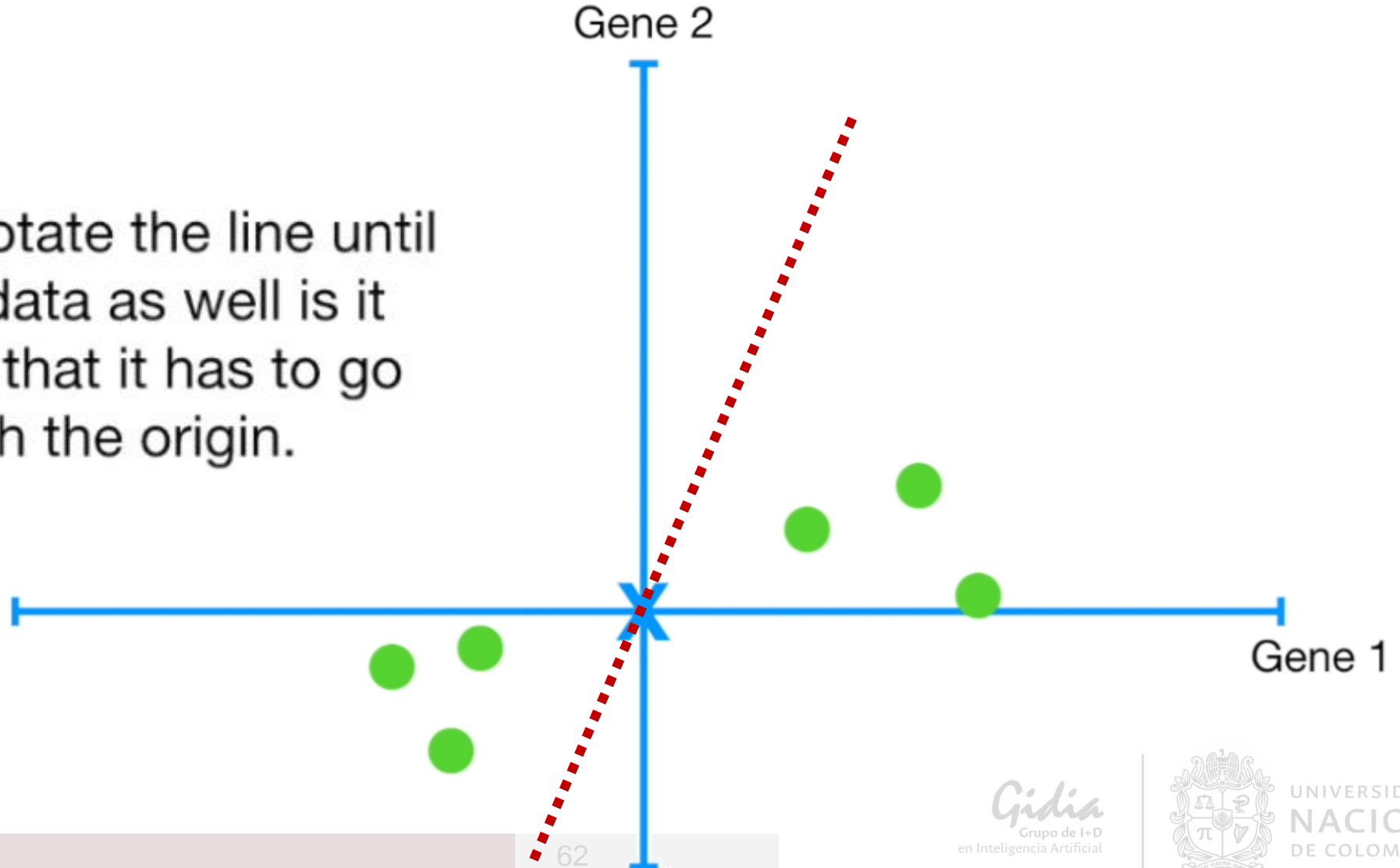
Now that the data are centered on the origin, we can try to fit a line to it.

To do this...



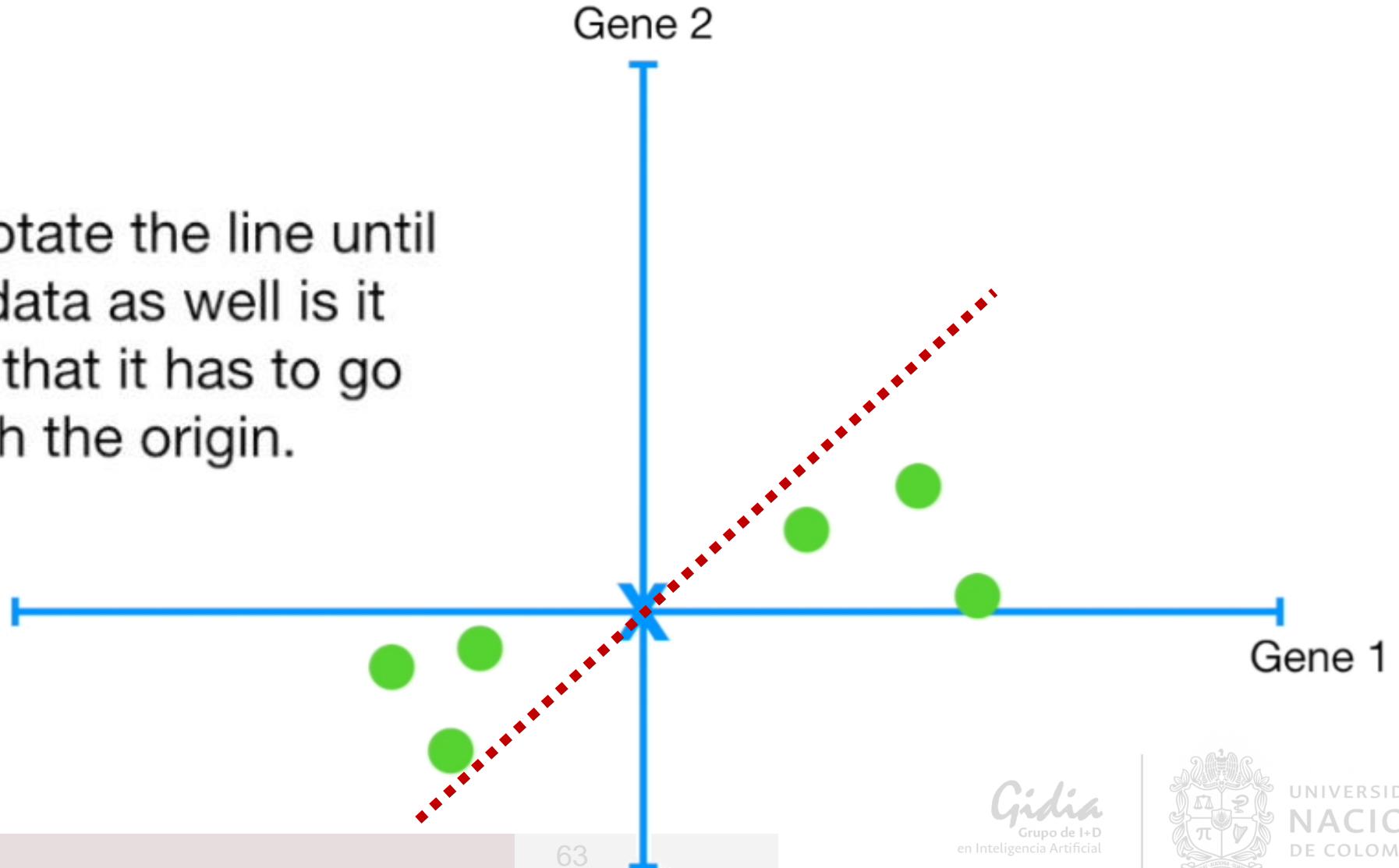
Extracción de Características - PCA

...then we rotate the line until it fits the data as well as it can, given that it has to go through the origin.



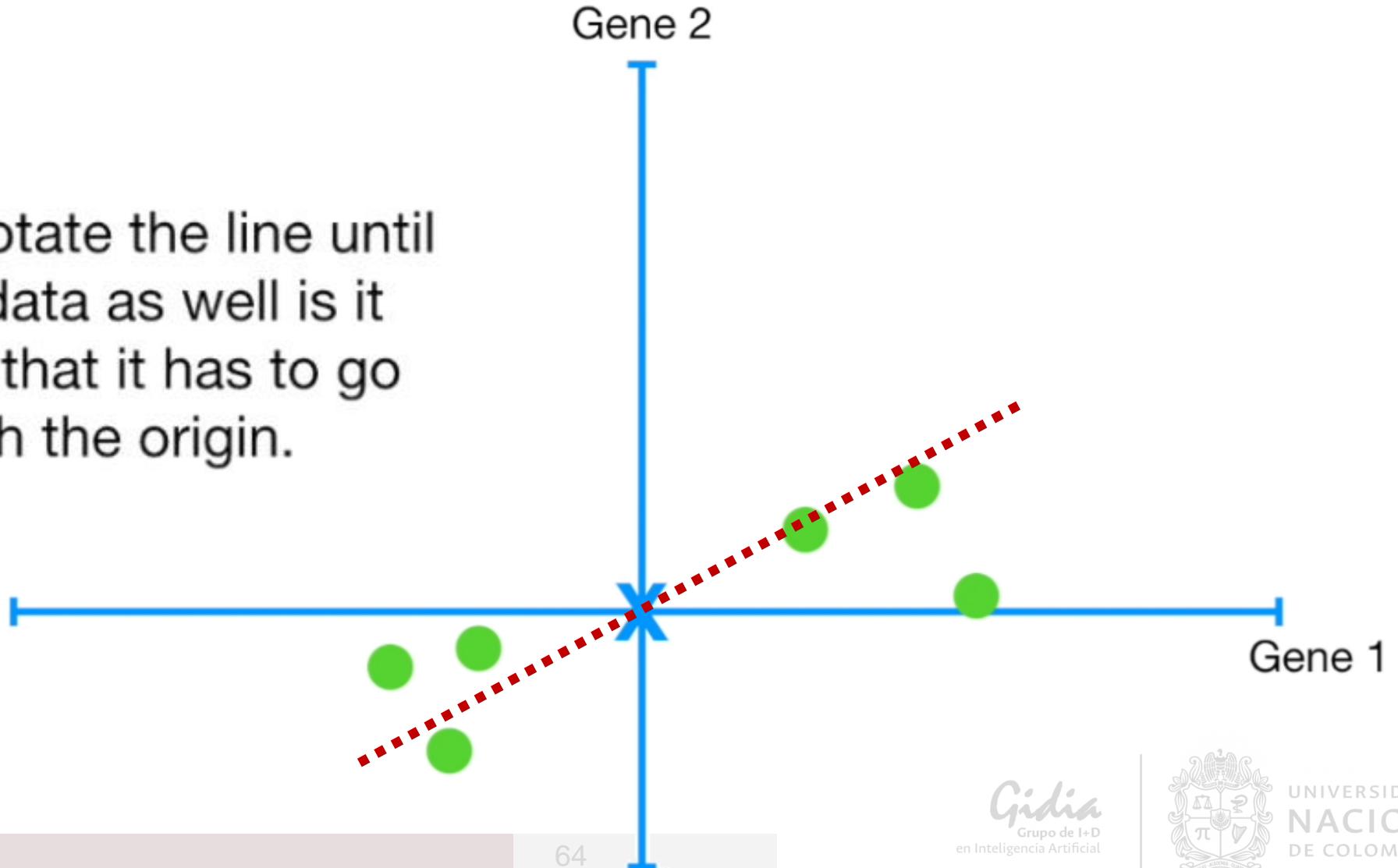
Extracción de Características - PCA

...then we rotate the line until it fits the data as well as it can, given that it has to go through the origin.



Extracción de Características - PCA

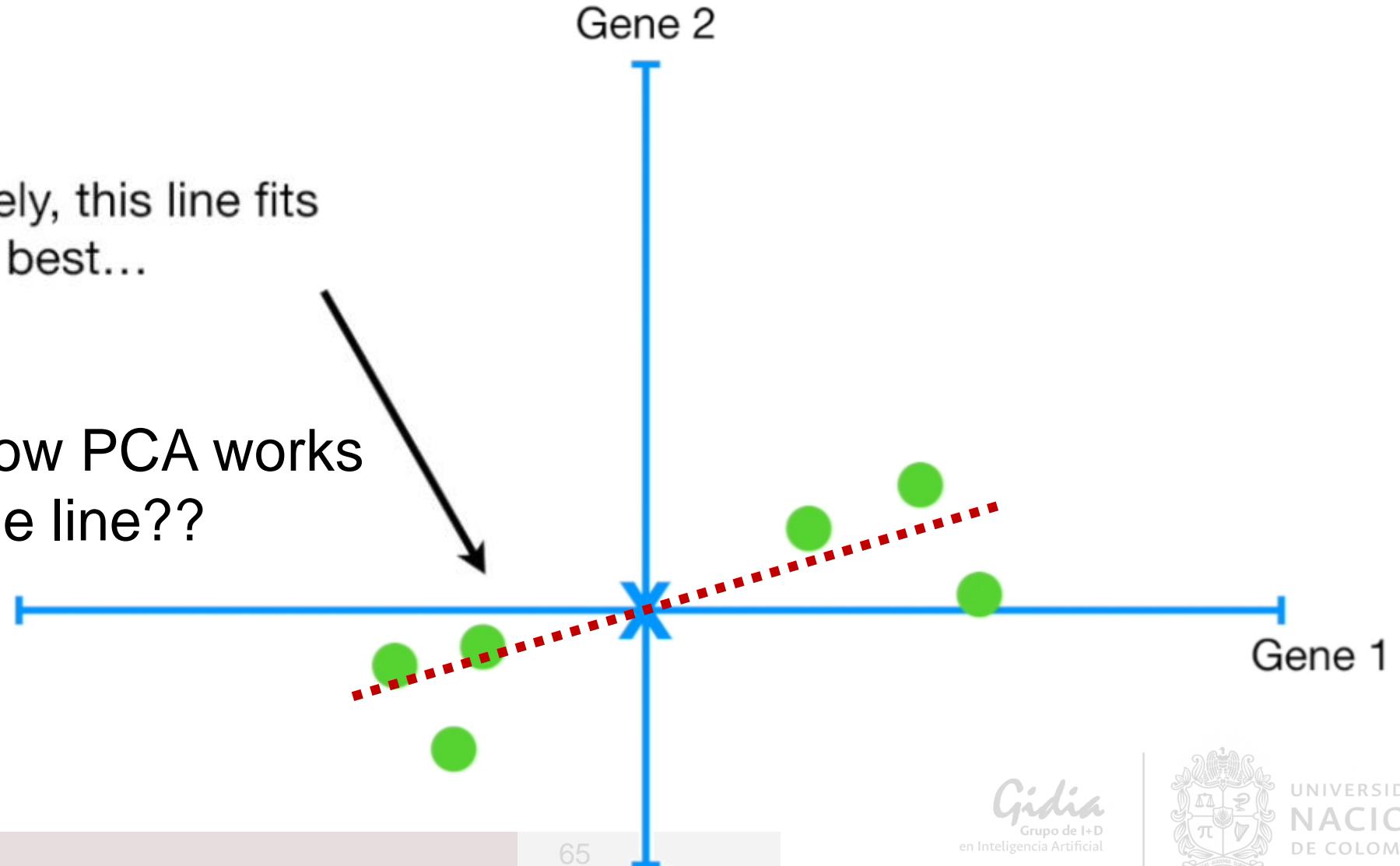
...then we rotate the line until it fits the data as well as it can, given that it has to go through the origin.



Extracción de Características - PCA

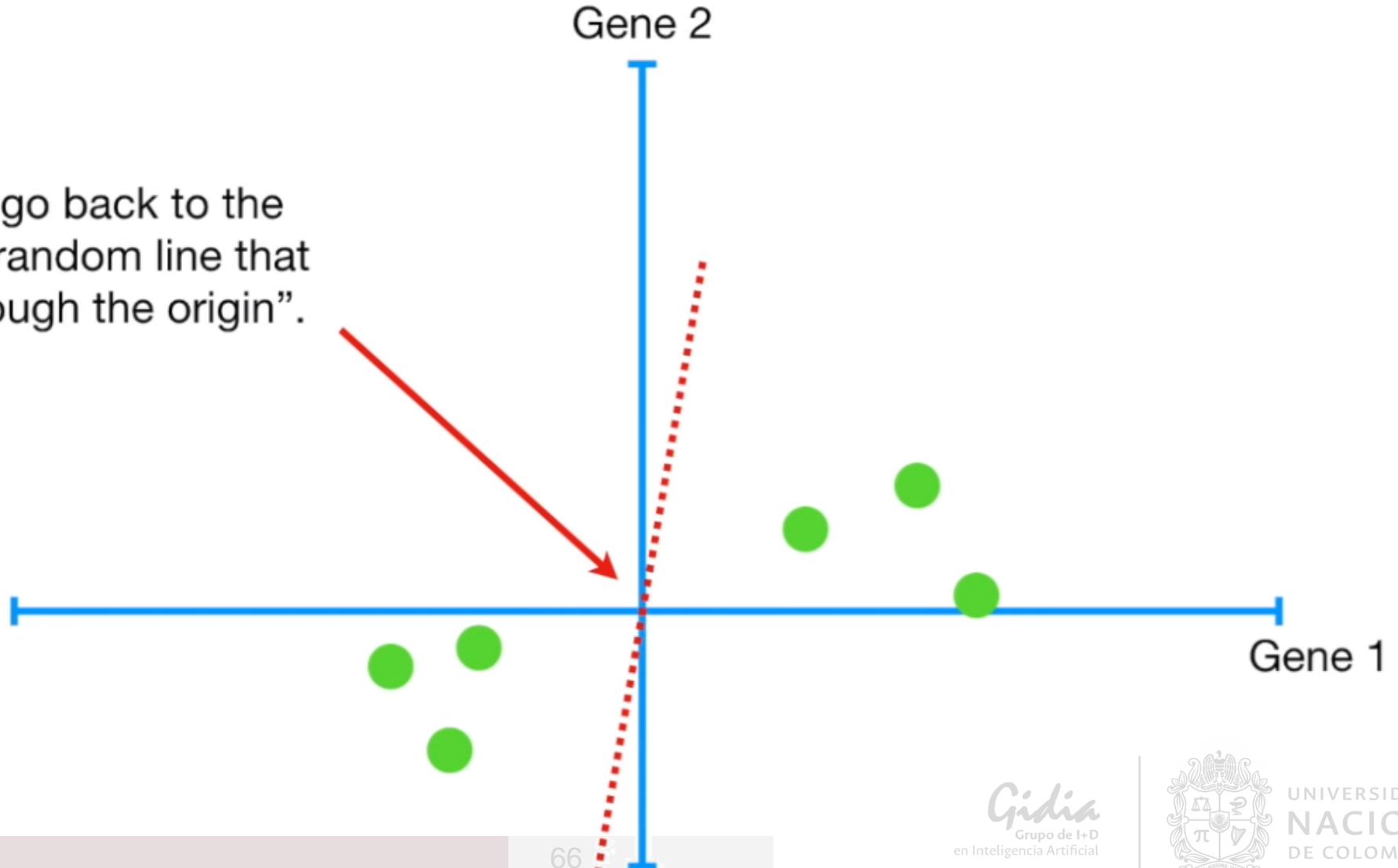
Ultimately, this line fits best...

But; How PCA works to fit the line??



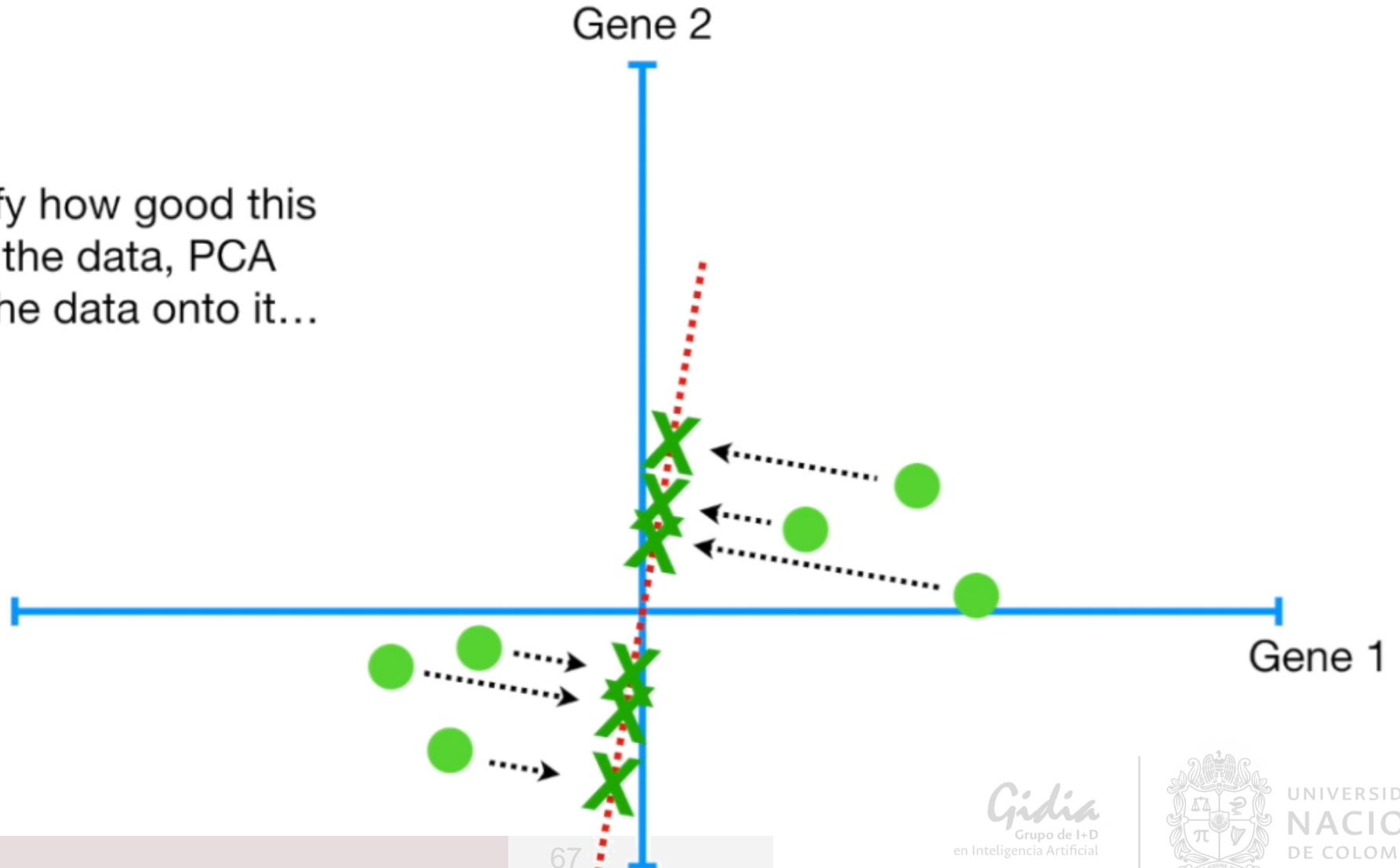
Extracción de Características - PCA

So let's go back to the original “random line that goes through the origin”.

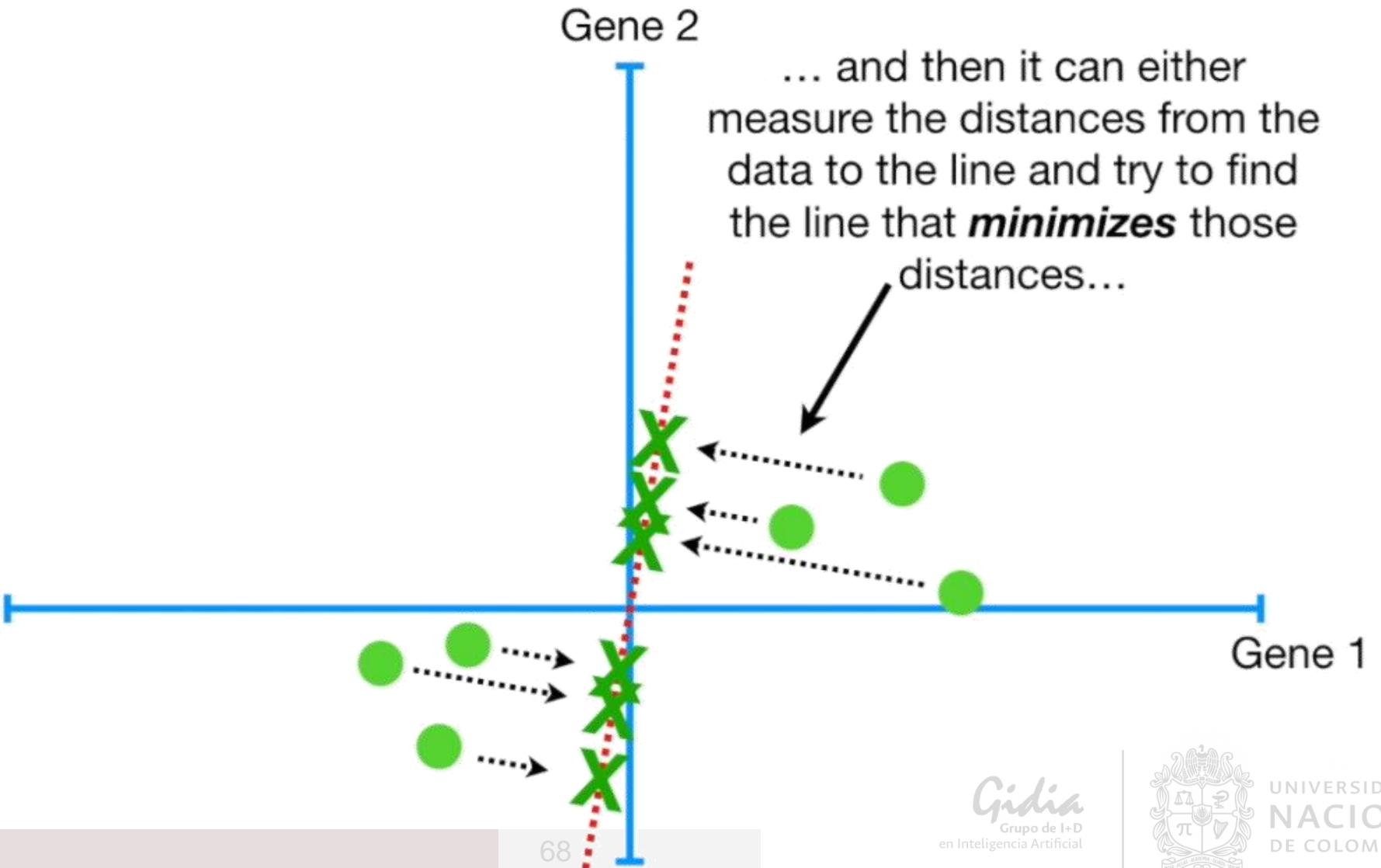


Extracción de Características - PCA

To quantify how good this line fits the data, PCA projects the data onto it...

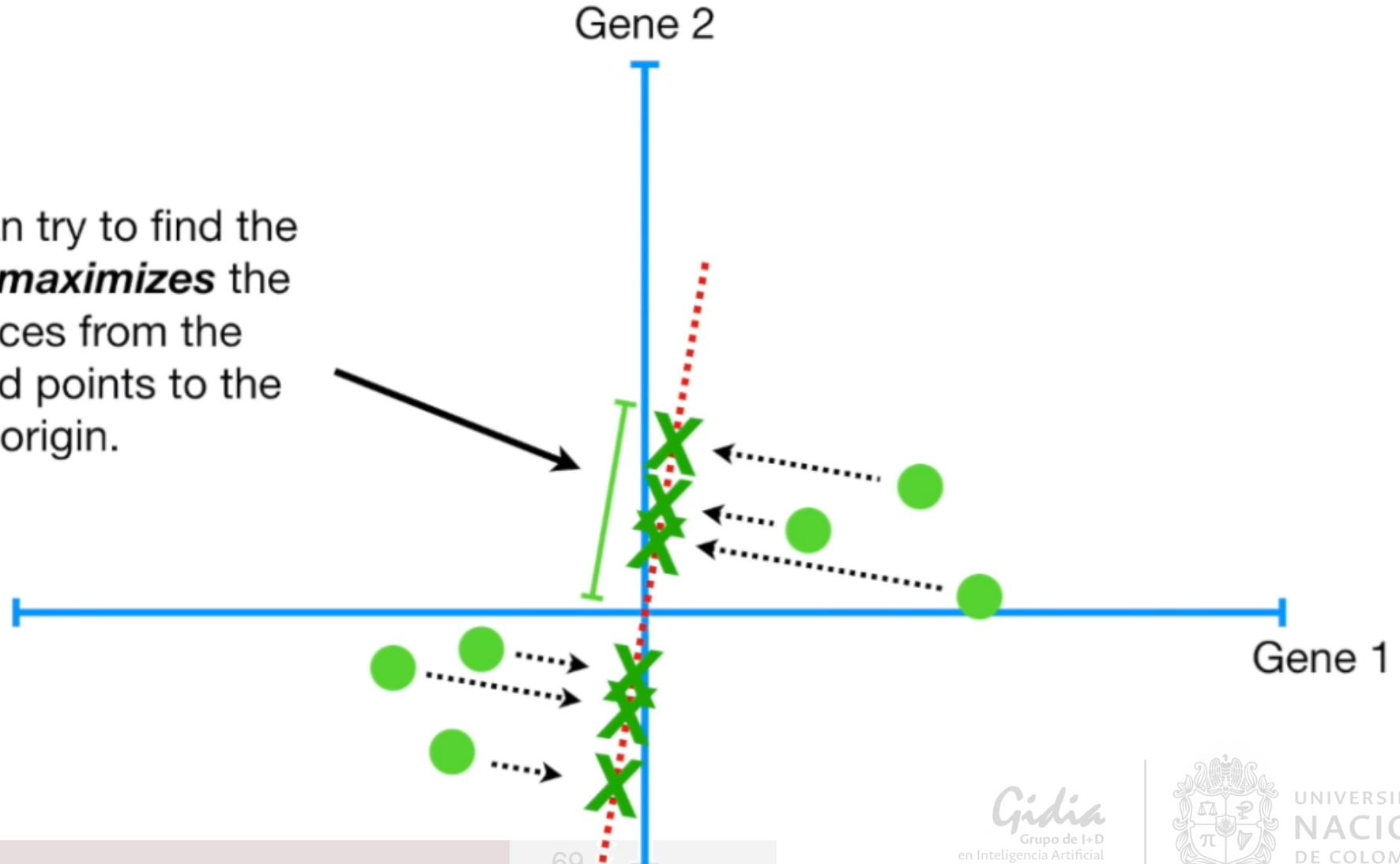


Extracción de Características - PCA



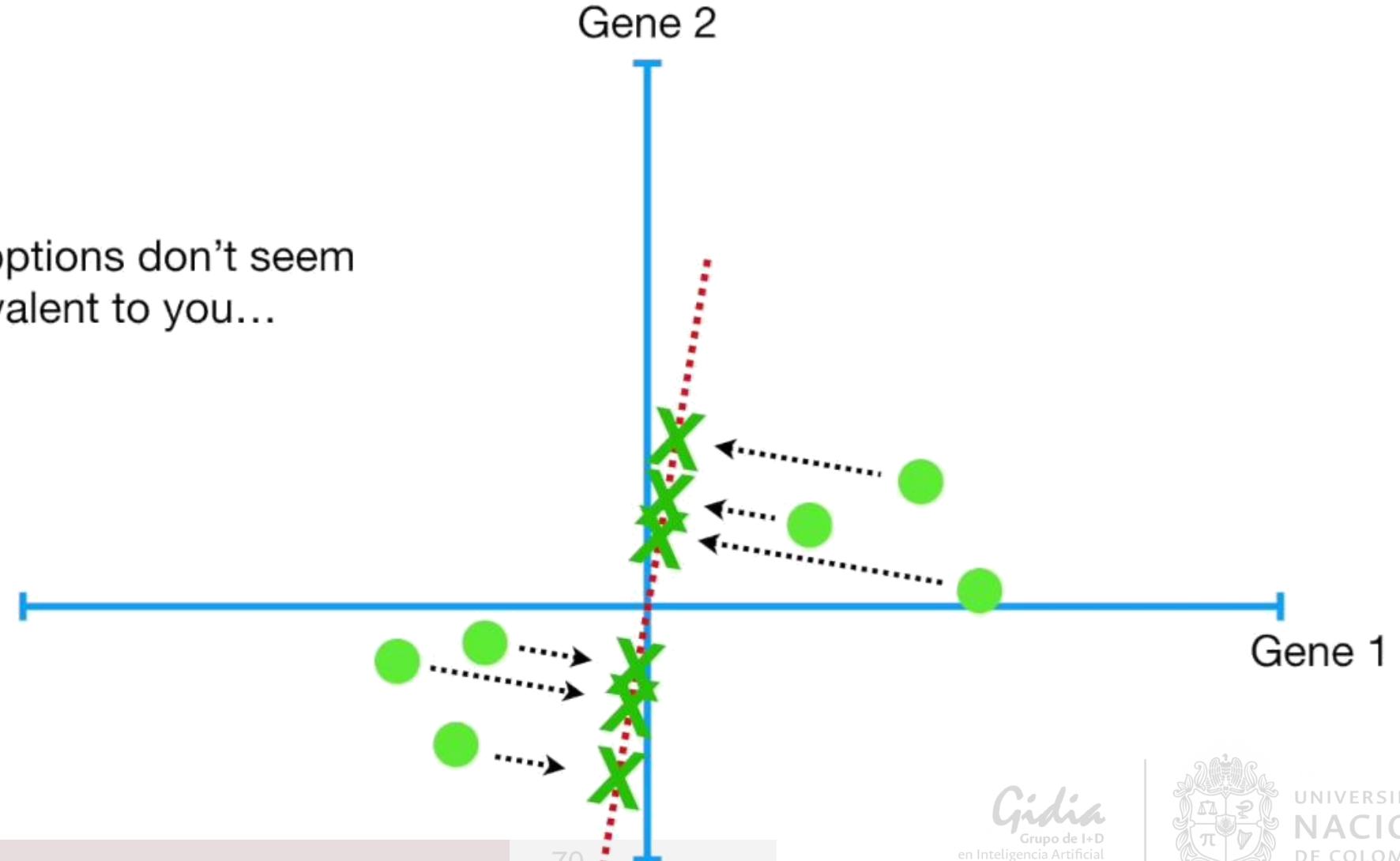
Extracción de Características - PCA

...or it can try to find the line that **maximizes** the distances from the projected points to the origin.

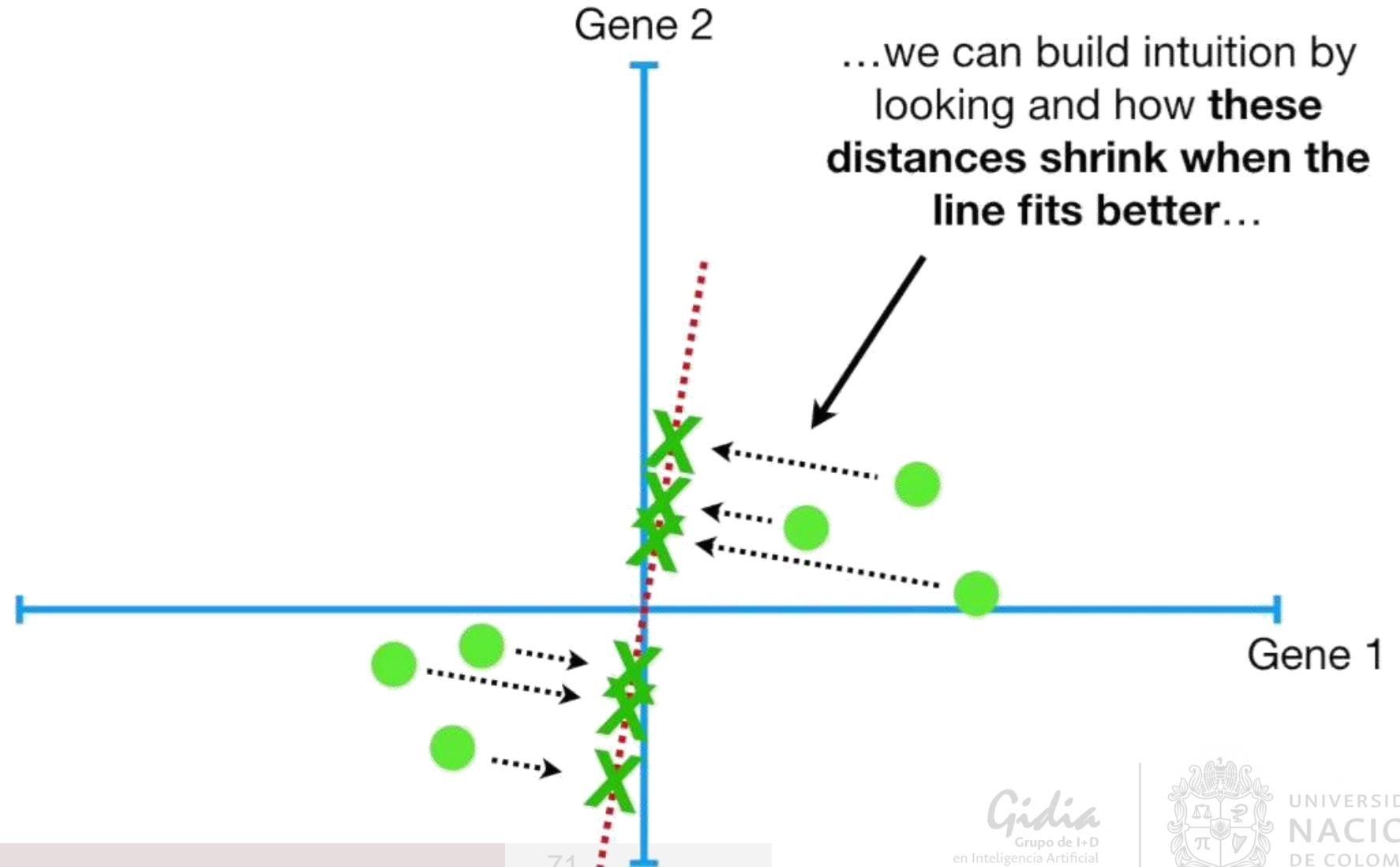


Extracción de Características - PCA

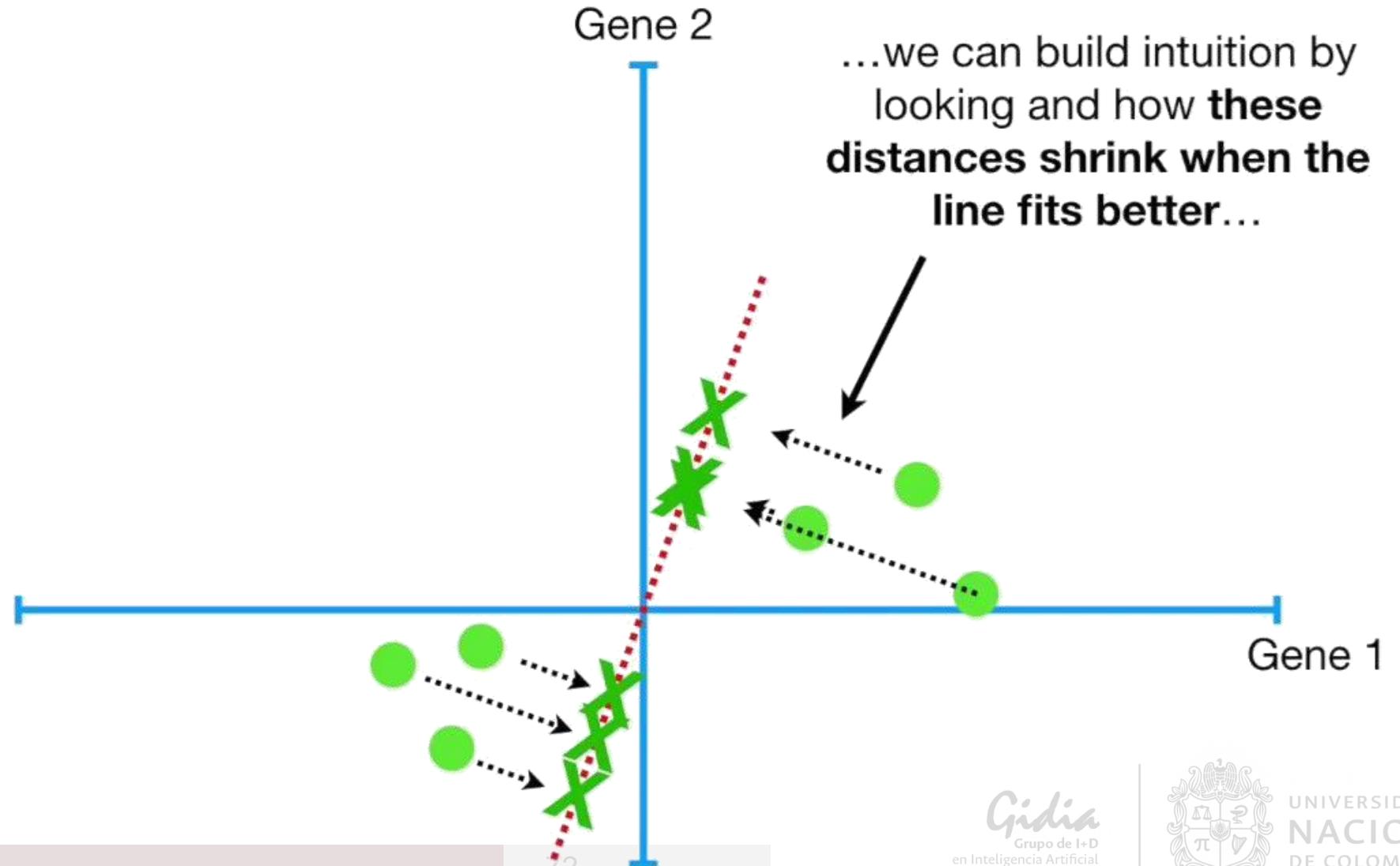
If those options don't seem equivalent to you...



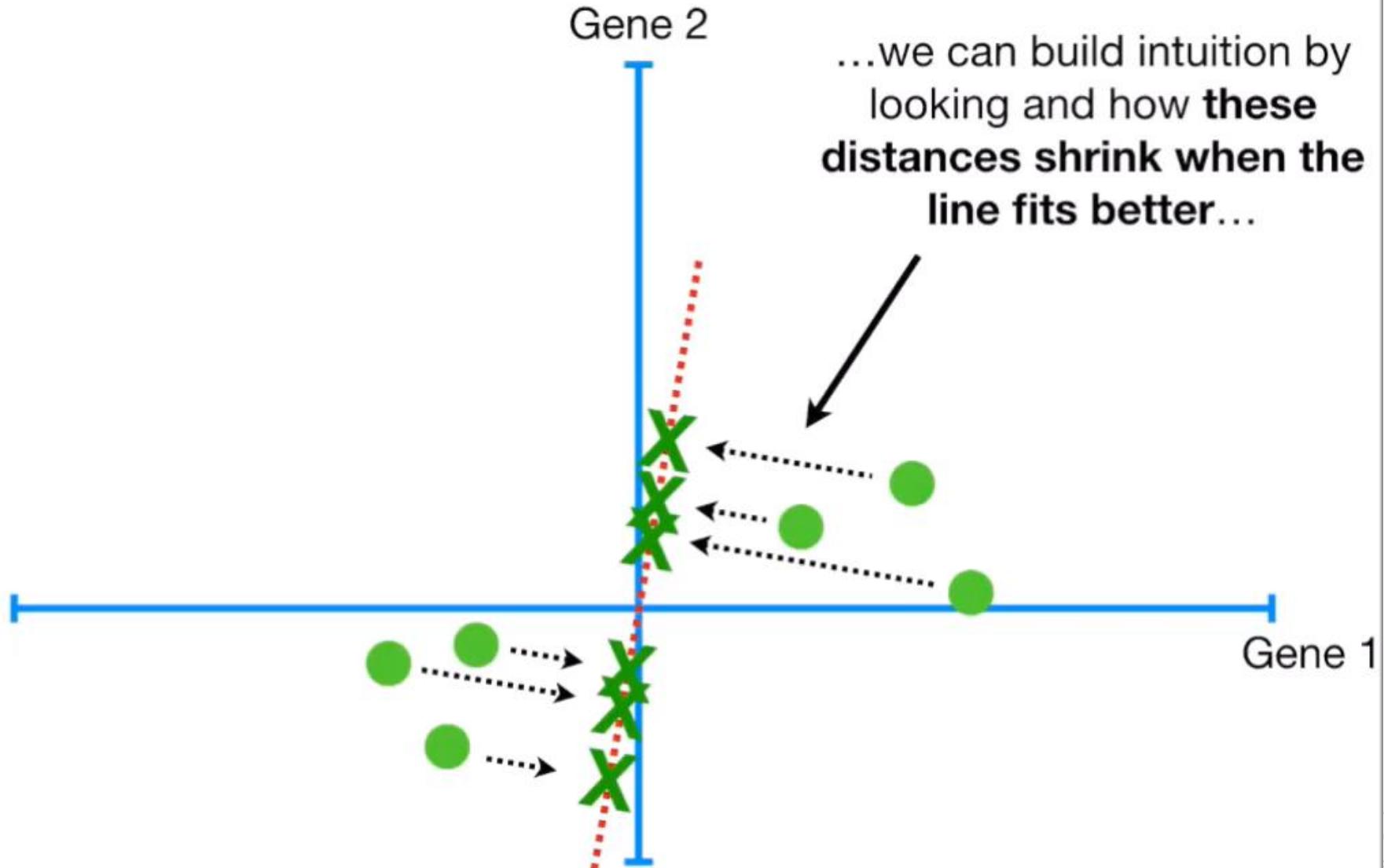
Extracción de Características - PCA



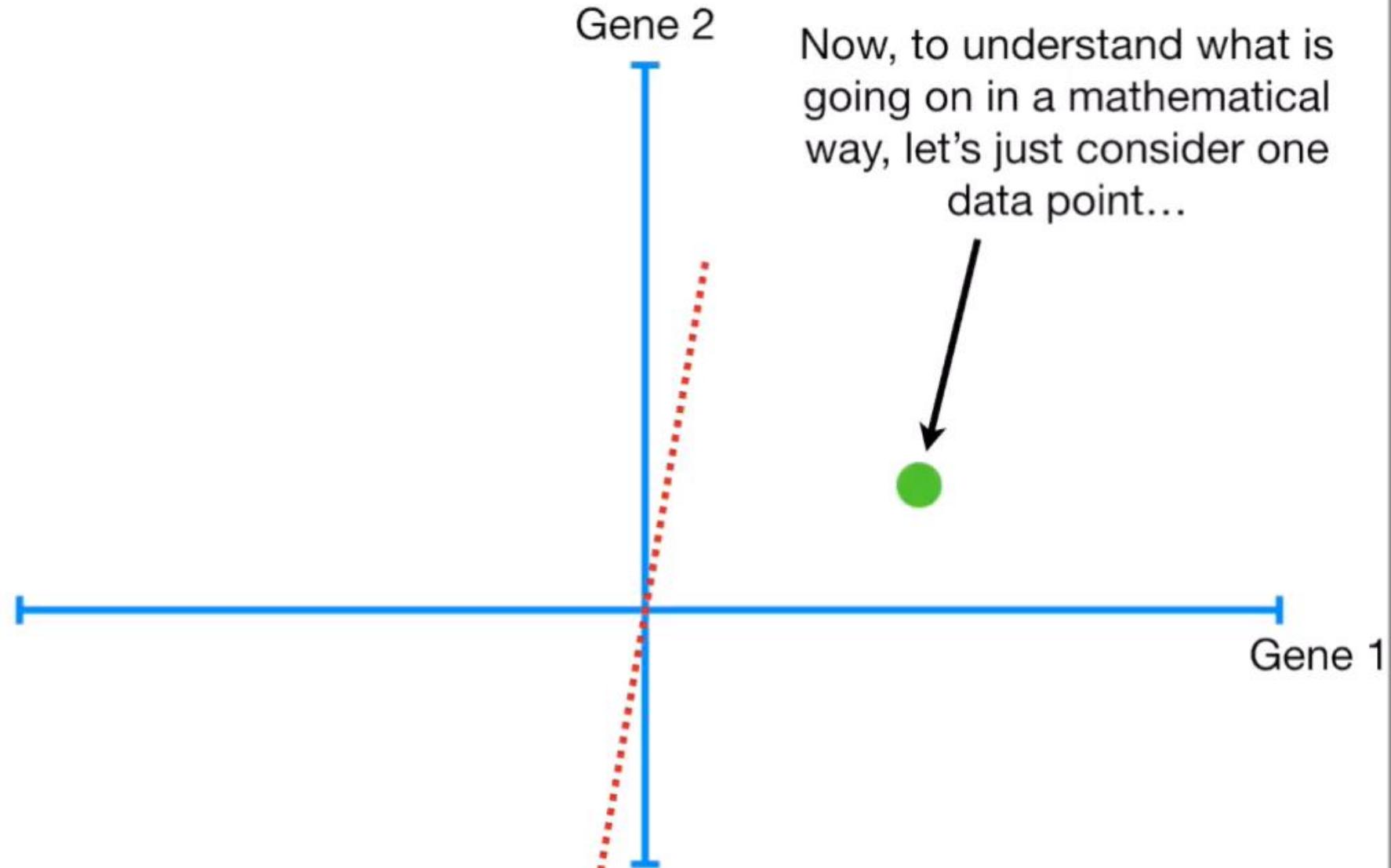
Extracción de Características - PCA



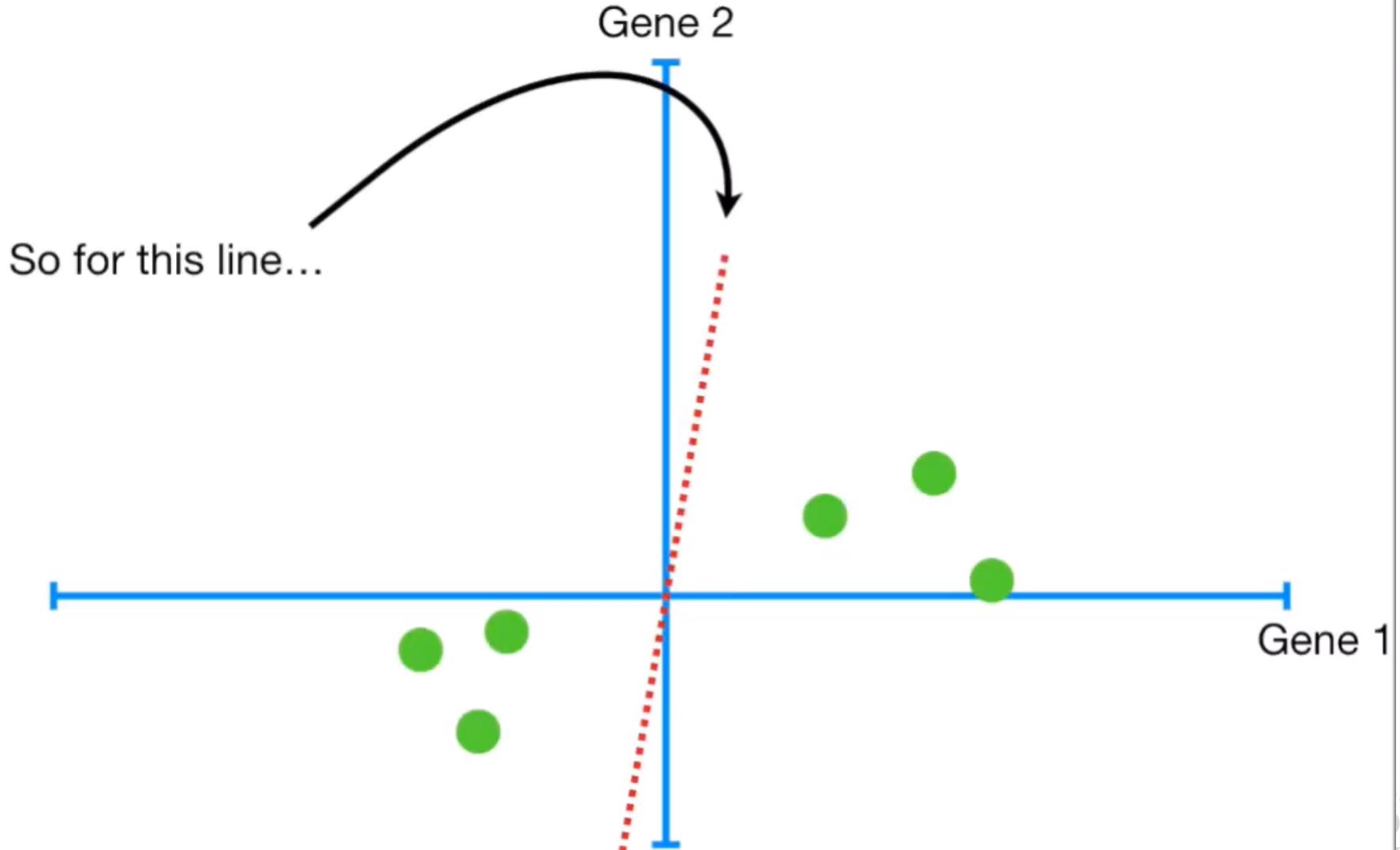
Extracción de Características - PCA



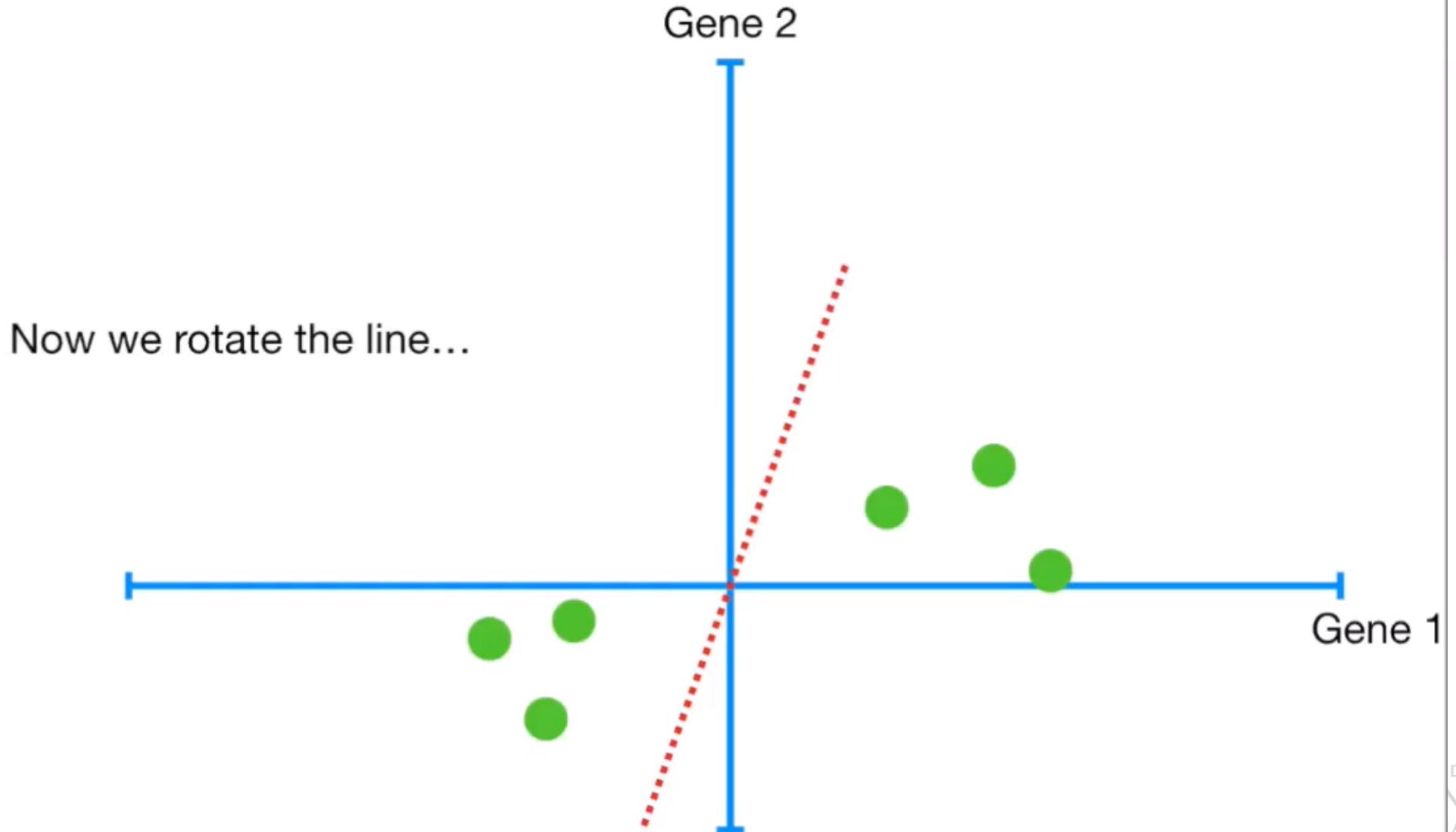
Extracción de Características - PCA



Extracción de Características - PCA

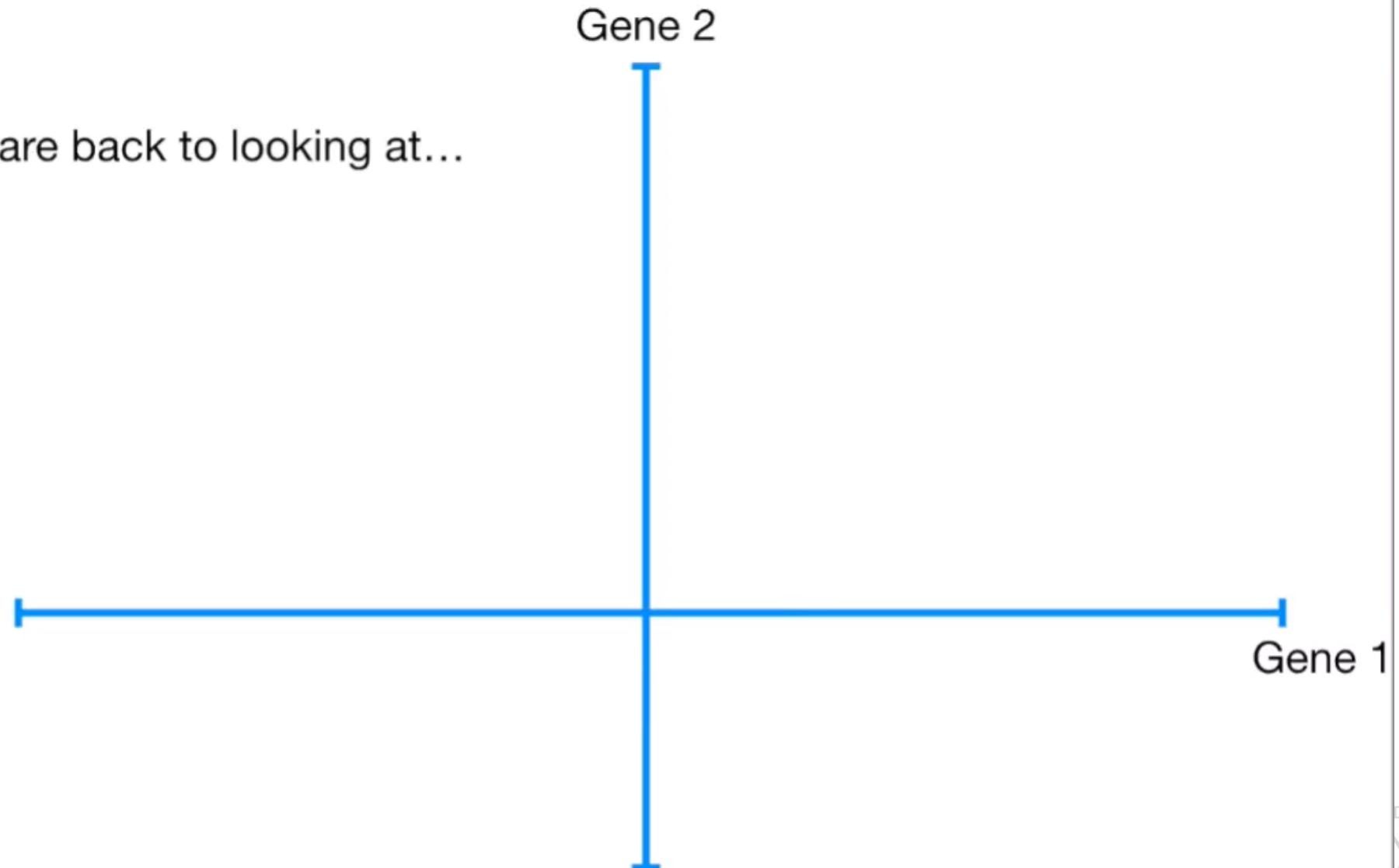


Extracción de Características - PCA



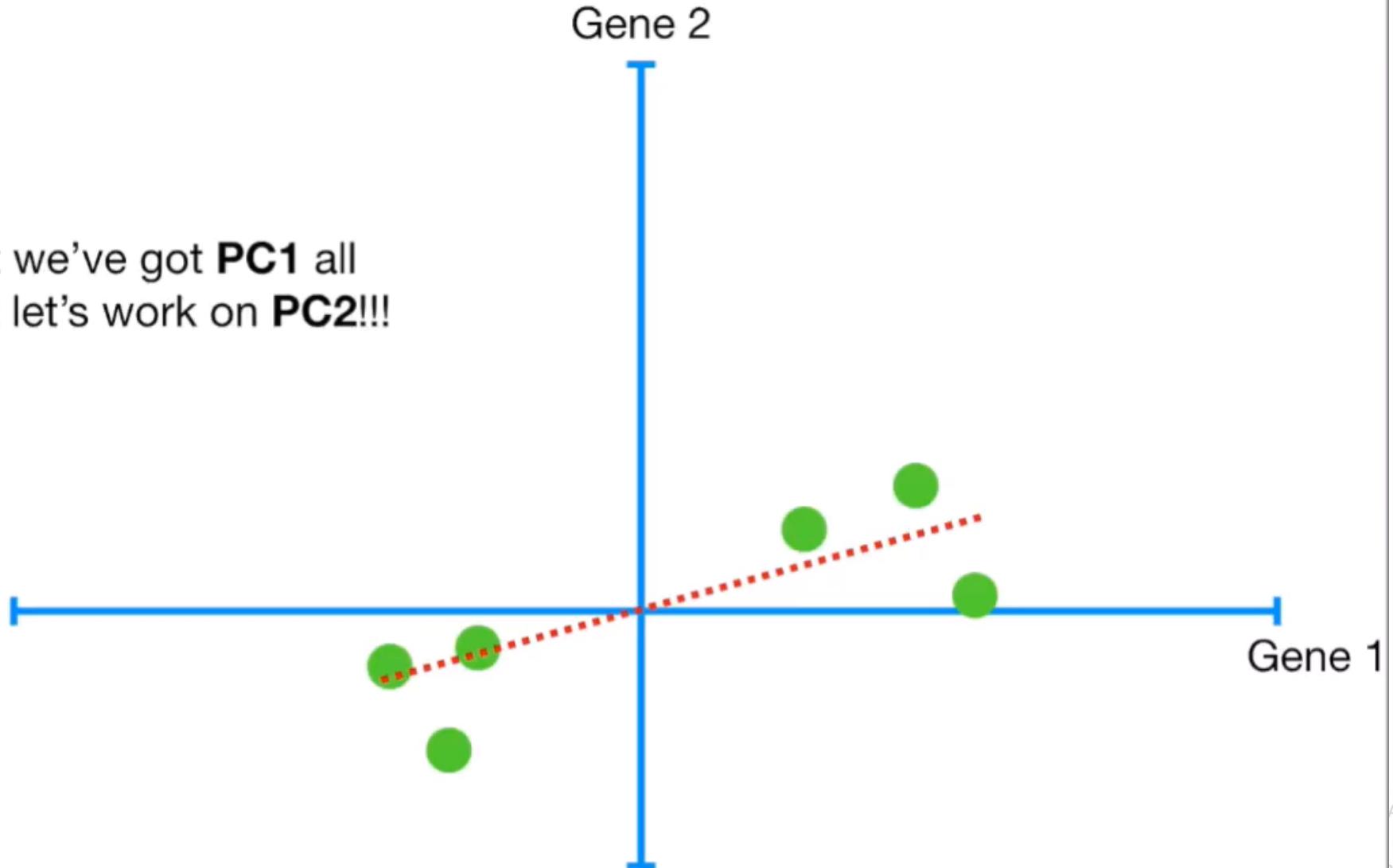
Extracción de Características - PCA

So now we are back to looking at...

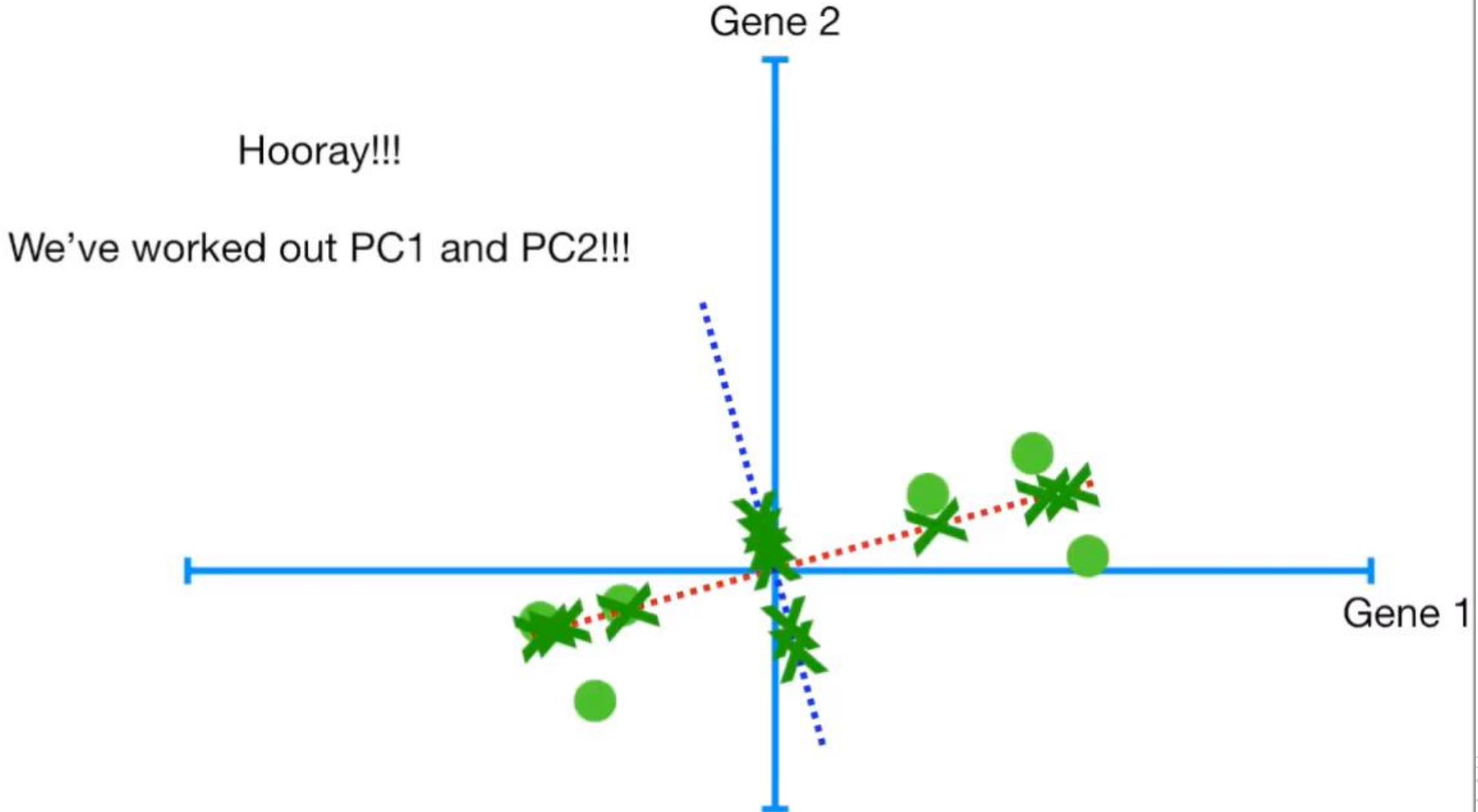


Extracción de Características - PCA

Now that we've got **PC1** all figured out let's work on **PC2!!!**



Extracción de Características - PCA

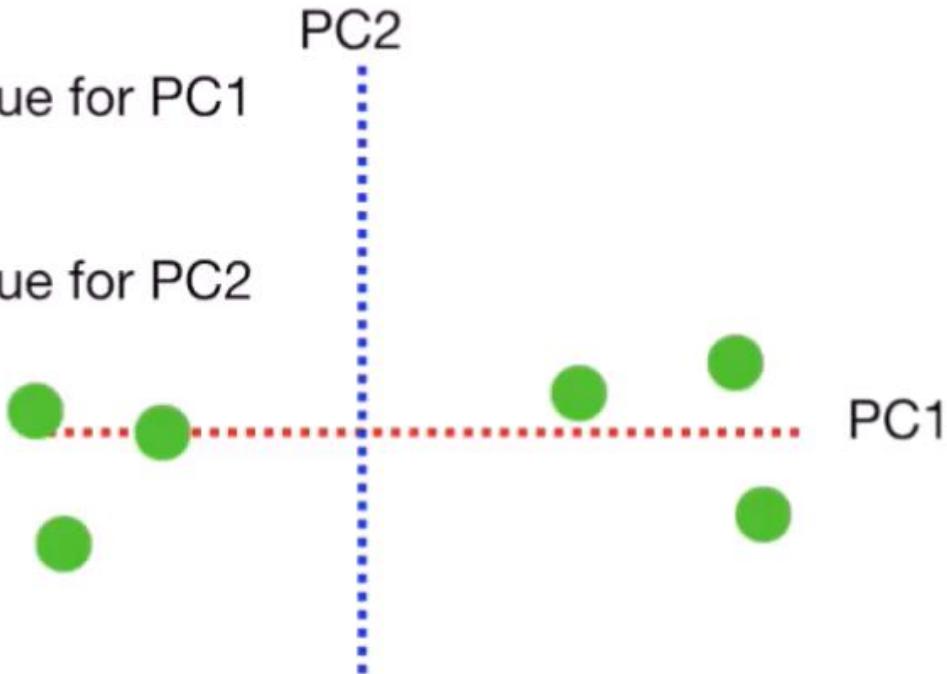


Extracción de Características - PCA

Remember the eigenvalues?

$SS(\text{distances for PC1}) = \text{Eigenvalue for PC1}$

$SS(\text{distances for PC2}) = \text{Eigenvalue for PC2}$



PCA y SVD

$$X_{n \times m} = \begin{bmatrix} & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \end{bmatrix} \begin{array}{l} n: \text{ejemplos} \\ \\ m: \text{dimensiones (measurements)} \end{array}$$

$$T = X W$$

"Scores" $(n \times m)$ "Loadings"
 $(n \times m)$ $(m \times m)$

Cada columna de W es un "Componente Principal"
.. columnas ordenadas por el valor de λ

Principal Component Analysis

eigen decomposition de

$$(X^T X) \xrightarrow{(m \times m)} \lambda$$

$$T_r = X W_r$$

Scores $(n \times m)$ $(m \times r)$
 $(n \times r)$

W es susceptible de ser TRUNCADA

W_r primeras r columnas de W

$$W = \begin{bmatrix} | & | & | & | \\ W_1 & W_2 & \dots & W_m \\ | & | & | & | \end{bmatrix}$$

PCA y SVD

Principal Component Analysis

eigen decomposition de

$$(X^T X) \rightarrow W \\ (mxm) \qquad \lambda$$

$$X_{n \times m} = \begin{bmatrix} & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \end{bmatrix} n: ejemplos$$

m: dimensiones (measurements)

Singular Value Decomposition

$$X = U \Sigma V^*$$

vectores Singulares Izquierdos Valores Singulares
vectores Singulares derechos

En su Diagonal

$$W = V$$

$$T = X W$$

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & \dots & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_m \end{bmatrix}$$

V Transpuesta de la conjugada*

$$V^* V = I$$

$$U^T U = I$$

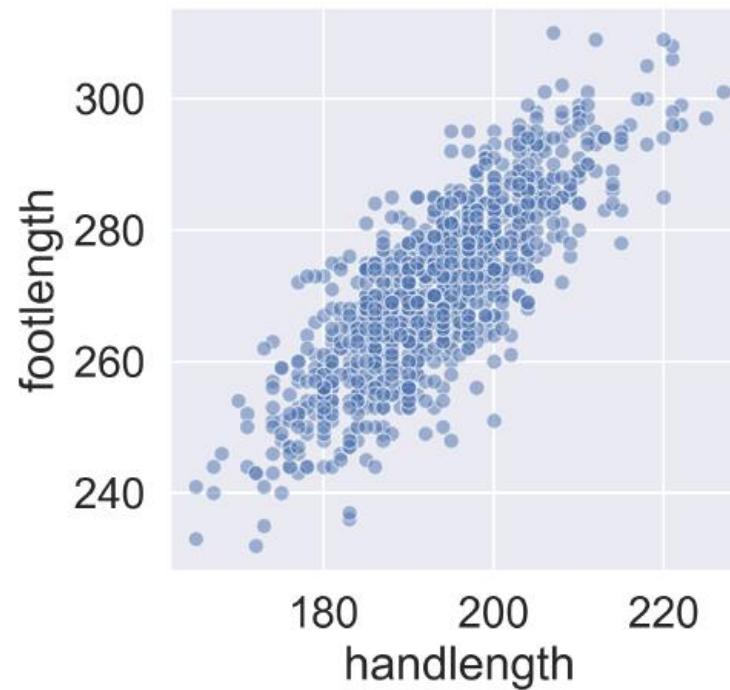
$$X V = U \Sigma V^* V$$

$$T = U \Sigma$$

Extracción de Características - PCA

Intro to PCA

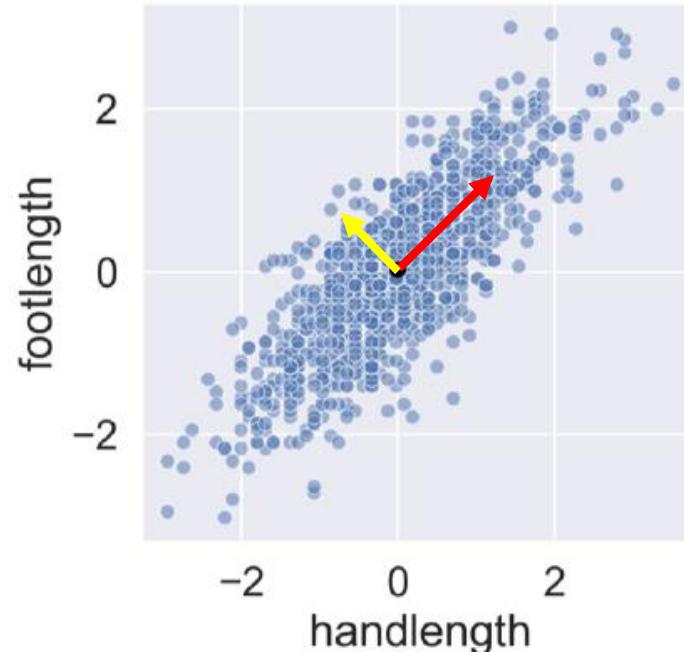
```
sns.scatterplot(data=df, x='handlength', y='footlength')
```



Extracción de Características - PCA

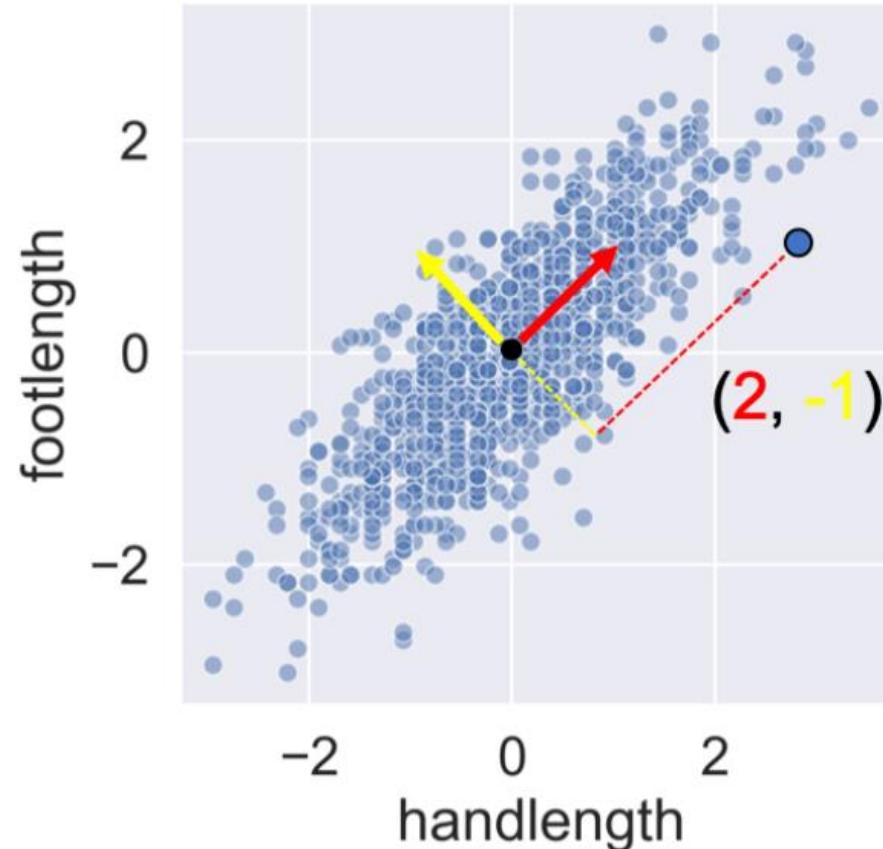
Intro to PCA

```
scaler = StandardScaler()  
df_std = pd.DataFrame(scaler.fit_transform(df), columns = df.columns)
```



Extracción de Características - PCA

PCA concept



Extracción de Características - PCA

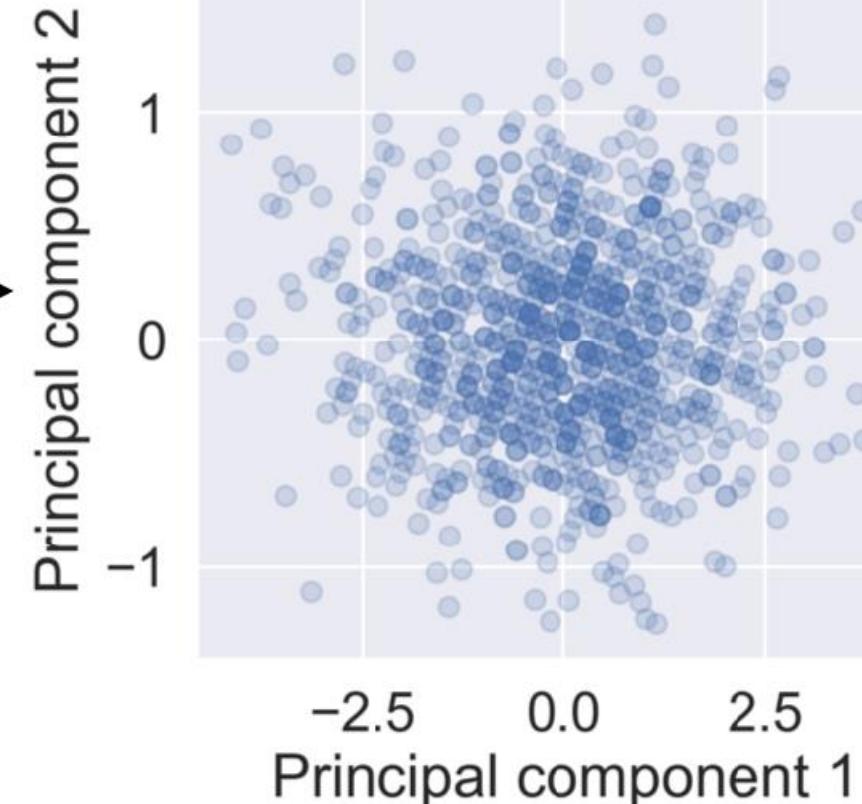
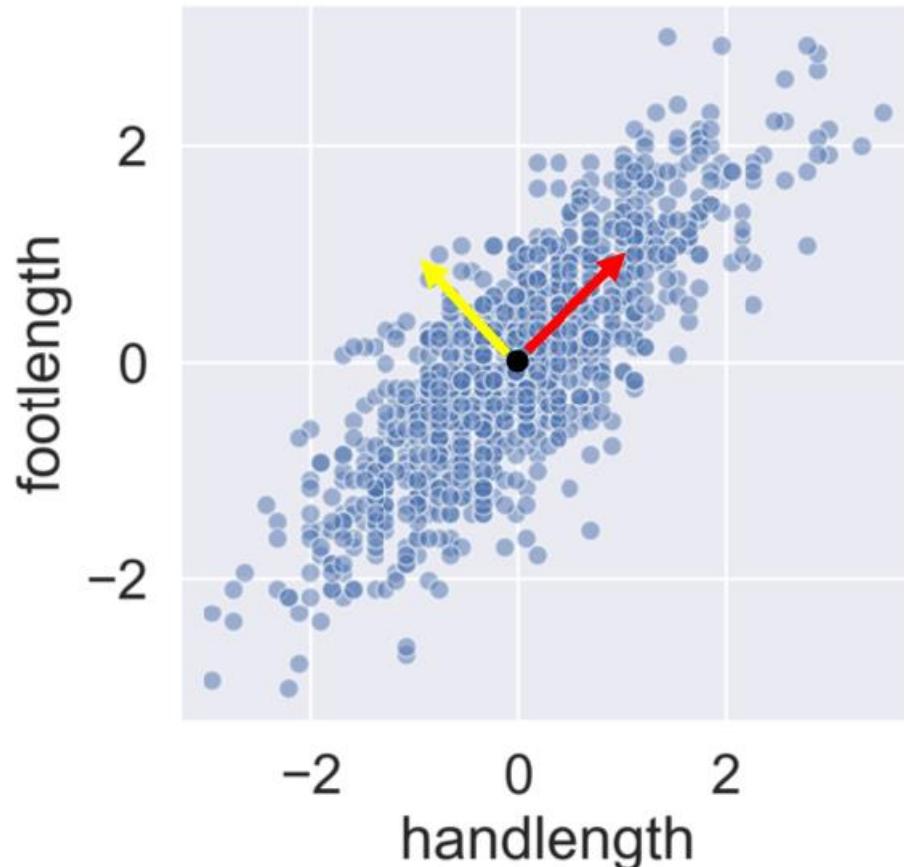
Calculating the principal components

```
from sklearn.preprocessing import StandardScaler  
  
scaler = StandardScaler()  
std_df = scaler.fit_transform(df)  
  
from sklearn.decomposition import PCA  
  
pca = PCA()  
print(pca.fit_transform(std_df))
```

```
[[-0.08320426 -0.12242952]  
 [ 0.31478004  0.57048158]  
 ...  
 [-0.5609523   0.13713944]  
 [-0.0448304  -0.37898246]]
```

Extracción de Características - PCA

PCA removes correlation



Extracción de Características - PCA

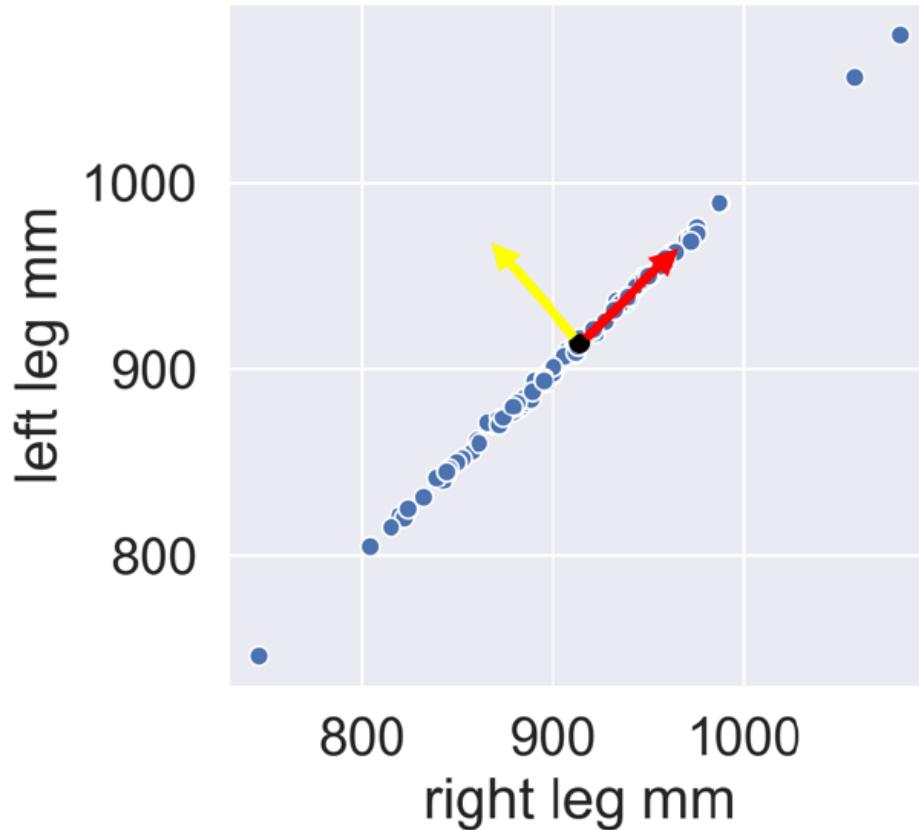
Principal component explained variance ratio

```
from sklearn.decomposition import PCA  
  
pca = PCA()  
  
pca.fit(std_df)  
  
print(pca.explained_variance_ratio_)
```

```
array([0.90, 0.10])
```

Extracción de Características - PCA

PCA for dimensionality reduction



```
print(pca.explained_variance_ratio_)
```

```
array([0.9997, 0.0003])
```

Extracción de Características - PCA

PCA for dimensionality reduction

```
pca = PCA()  
  
pca.fit(ansur_std_df)  
  
print(pca.explained_variance_ratio_.cumsum())
```

```
array([0.44, 0.62, 0.66, 0.69, 0.72, 0.74, 0.76, 0.77, 0.79, 0.8 , 0.81,  
     0.82, 0.83, 0.84, 0.85, 0.86, 0.87, 0.87, 0.88, 0.89, 0.89, 0.9 ,  
     0.9 , 0.91, 0.92, 0.92, 0.92, 0.93, 0.93, 0.94, 0.94, 0.94, 0.95,  
     ...  
     0.99, 0.99, 0.99, 0.99, 0.99, 1. , 1. , 1. , 1. , 1. , 1. , 1. ,  
     1. , 1. , 1. , 1. , 1. , 1. , 1. , 1. , 1. , 1. , 1. , 1. ,  
     1. , 1. , 1. , 1. , 1. , 1. ])
```

Aplicaciones de PCA

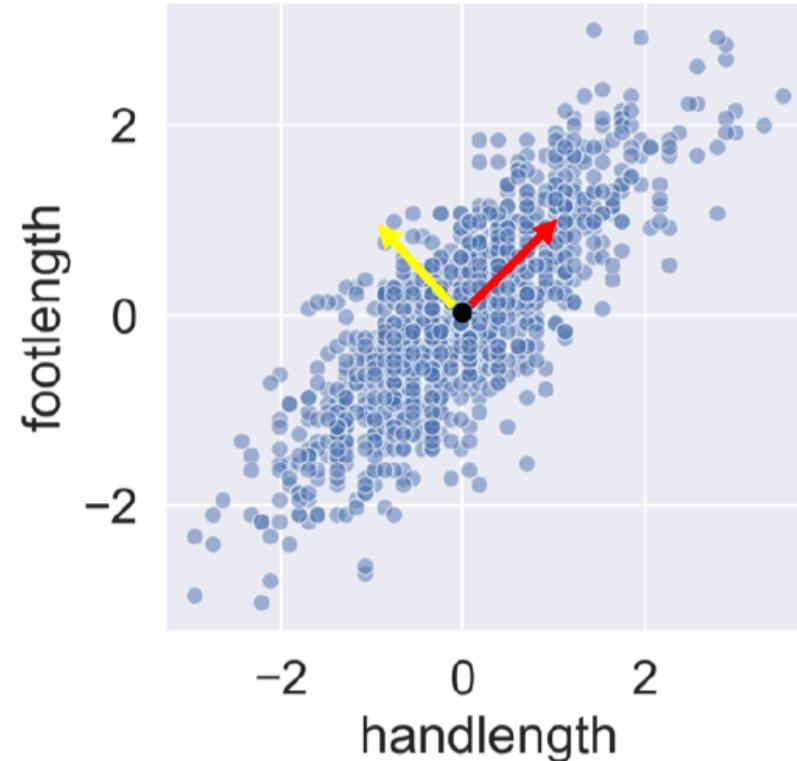
Understanding the components

```
print(pca.components_)
```

```
array([[ 0.71,  0.71],  
       [-0.71,  0.71]])
```

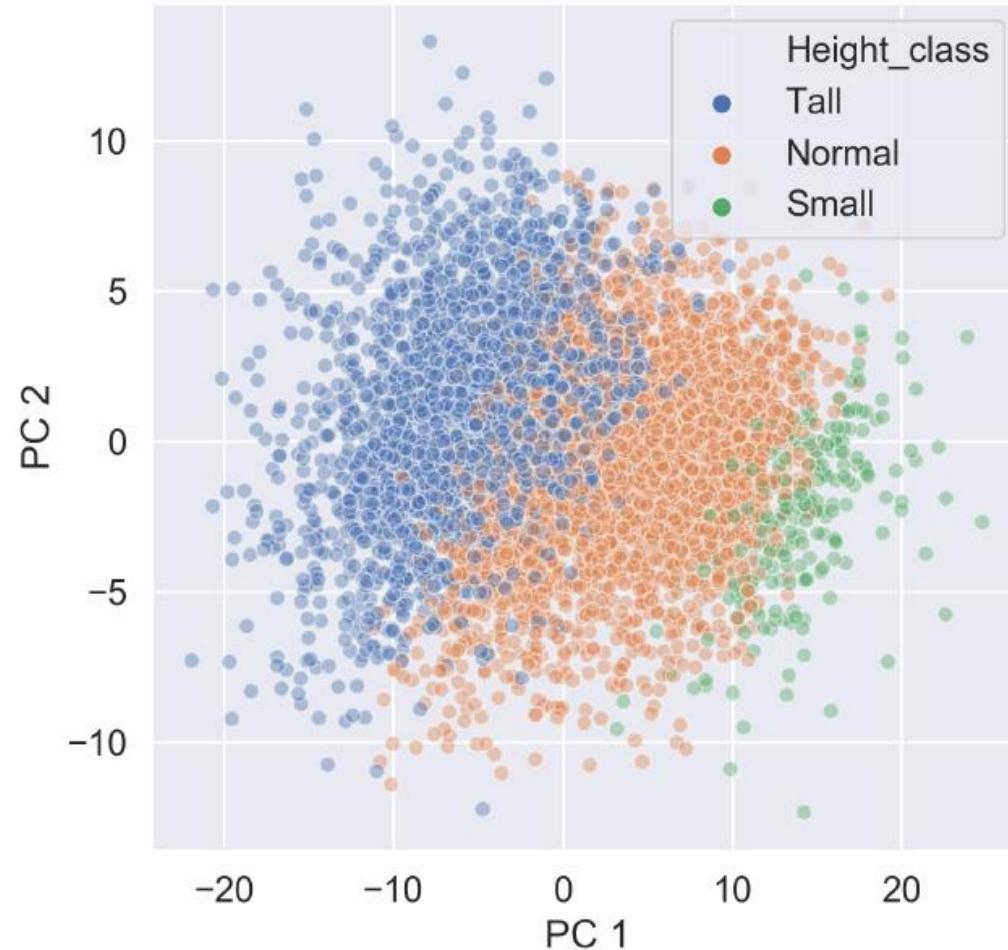
PC 1 = 0.71 x Hand length + 0.71 x Foot length

PC 2 = -0.71 x Hand length + 0.71 x Foot length



Aplicaciones de PCA

PCA for data exploration



Aplicaciones de PCA

PCA in a pipeline

```
from sklearn.preprocessing import StandardScaler  
from sklearn.decomposition import PCA  
from sklearn.pipeline import Pipeline  
  
pipe = Pipeline([  
    ('scaler', StandardScaler()),  
    ('reducer', PCA())])  
pc = pipe.fit_transform(ansur_df)  
  
print(pc[:, :2])
```

```
array([[-3.46114925,  1.5785215 ],  
      [ 0.90860615,  2.02379935],  
      ...,  
      [10.7569818 , -1.40222755],  
      [ 7.64802025,  1.07406209]])
```

Aplicaciones de PCA

Checking the effect of categorical features

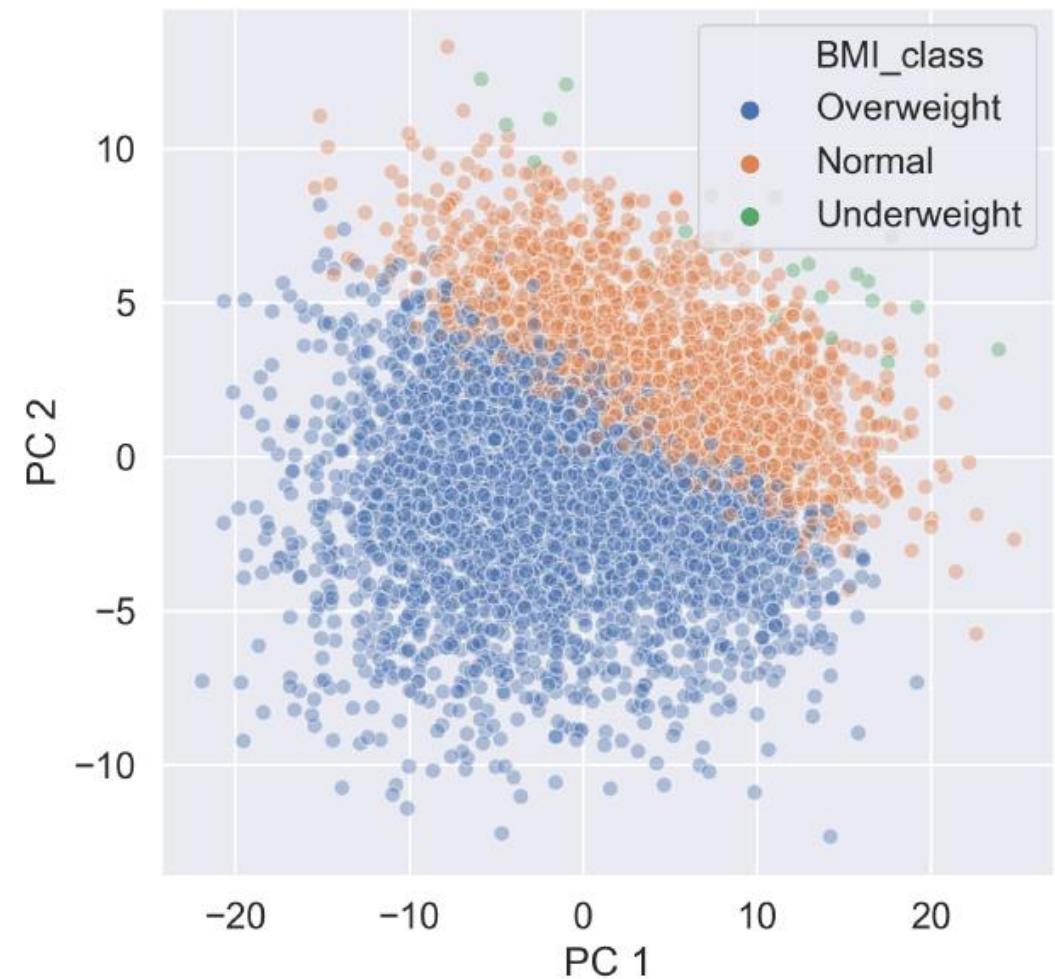
```
print(ansur_categories.head())
```

Branch	Component	Gender	BMI_class	Height_class
Combat Arms	Regular Army	Male	Overweight	Tall
Combat Support	Regular Army	Male	Overweight	Normal
Combat Support	Regular Army	Male	Overweight	Normal
Combat Service Support	Regular Army	Male	Overweight	Normal
Combat Service Support	Regular Army	Male	Overweight	Tall

Aplicaciones de PCA

Checking the effect of categorical features

```
sns.scatterplot(data=ansur_categories,  
                 x='PC 1', y='PC 2',  
                 hue='BMI_class', alpha=0.4)
```



Aplicaciones de PCA

PCA in a model pipeline

```
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('reducer', PCA(n_components=3)),
    ('classifier', RandomForestClassifier())])
pipe.fit(X_train, y_train)
print(pipe.steps[1])
```

```
('reducer',
PCA(copy=True, iterated_power='auto', n_components=3, random_state=None,
svd_solver='auto', tol=0.0, whiten=False))
```

Aplicaciones de PCA

PCA in a model pipeline

```
pipe.steps[1][1].explained_variance_ratio_.cumsum()
```

```
array([0.56, 0.69, 0.74])
```

```
print(pipe.score(X_test, y_test))
```

```
0.986
```

Setting an explained variance threshold

```
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('reducer', PCA(n_components=0.9))])

# Fit the pipe to the data
pipe.fit(poke_df)

print(len(pipe.steps[1][1].components_))
```

5

Selección de PCA

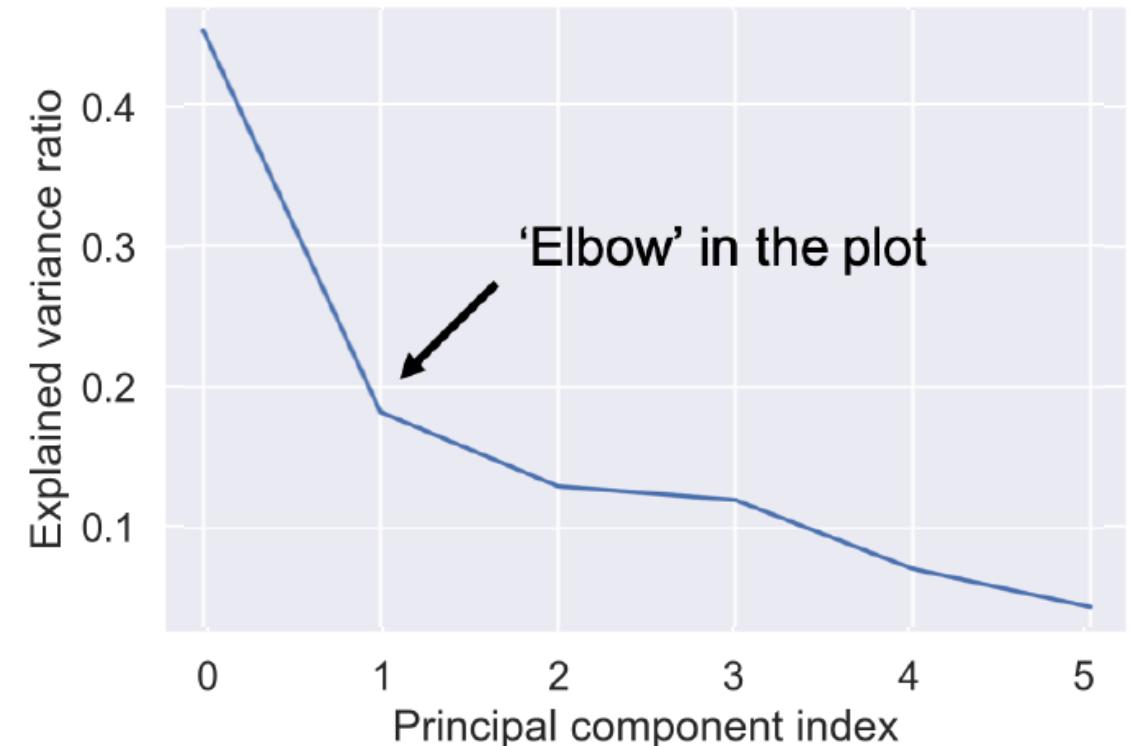
An optimal number of components

```
pipe.fit(poke_df)

var = pipe.steps[1][1].explained_variance_ratio_

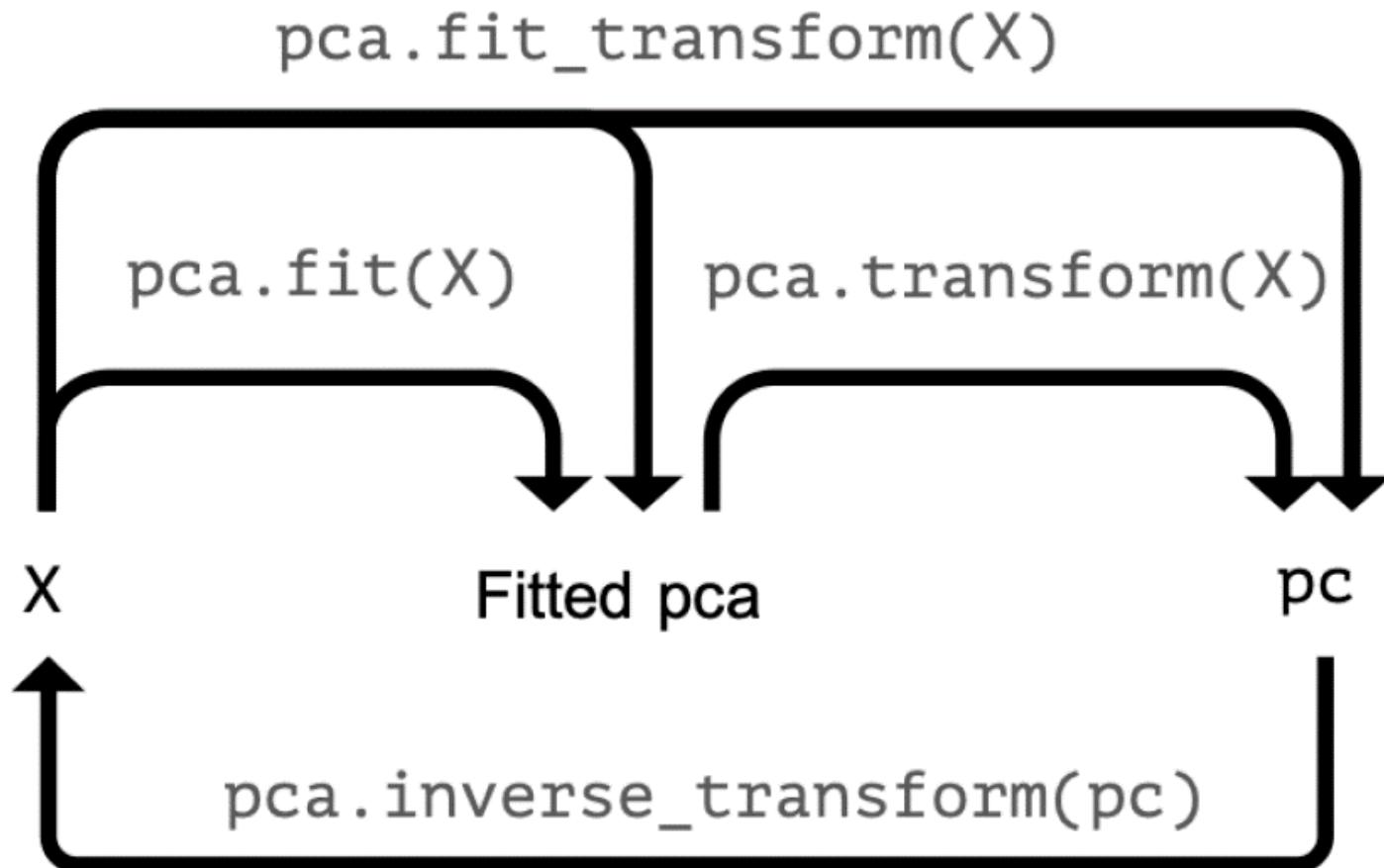
plt.plot(var)

plt.xlabel('Principal component index')
plt.ylabel('Explained variance ratio')
plt.show()
```



Selección de PCA

PCA operations



Selección de PCA

Compressing images



Selección de PCA

Compressing images

```
print(X_test.shape)
```

```
(15, 2914)
```

62 x 47 pixels = 2914 grayscale values

```
print(X_train.shape)
```

```
(1333, 2914)
```

Compressing images

```
pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('reducer', PCA(n_components=290))])

pipe.fit(X_train)

pc = pipe.fit_transform(X_test)

print(pc.shape)
```

(15, 290)

Selección de PCA

Rebuilding images



Laboratorio 2

<https://github.com/srobles05/3008422-AprendizajeDeMaquinas/>



Extracción manual de características

Se desea comparar precios de productos específicos entre tiendas. Las características del conjunto de datos pre-cargado sales_df son: storeID, product, quantity y revenue. Las características quantity y revenue indican cuántos artículos de un producto en particular se vendieron en una tienda y cuáles fueron los ingresos totales. Para el propósito de su análisis, es más interesante saber el precio promedio por producto.

```
In [3]: # import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
sales_df=pd.read_csv("lab4-1.csv")#reading a dataset in a dataframe using pandas
print("sales_df",sales_df.shape)

sales_df (78, 4)
```

Feature Extraction

Calcule el precio del producto a partir de la cantidad vendida (quantity) y los ingresos totales (revenue).

```
In [4]: # Calculate the price from the quantity sold and revenue
sales_df['price'] = ____
print("sales_df",sales_df.shape)

sales_df (78, 5)
```

Descarte las características cantidad vendida (quantity) y ingresos totales (revenue) del conjunto de datos.

```
In [5]: # Drop the quantity and revenue features
reduced_df = sales_df.drop(___, axis=1)

print(reduced_df.head())
```

Gracias !!!



© Man Bouncing Question Mark Towards Doctor - Artist: [Art Glazer](#)