



UNIVERSIDAD
NACIONAL
DE COLOMBIA

CLASIFICACIÓN Y RECONOCIMIENTO DE PATRONES

Extracción de Características y Reducción de la Dimensionalidad

Sesión 3

Jorge E. Espinosa

Profesor

Departamento de Ciencias de la Computación y de la Decisión

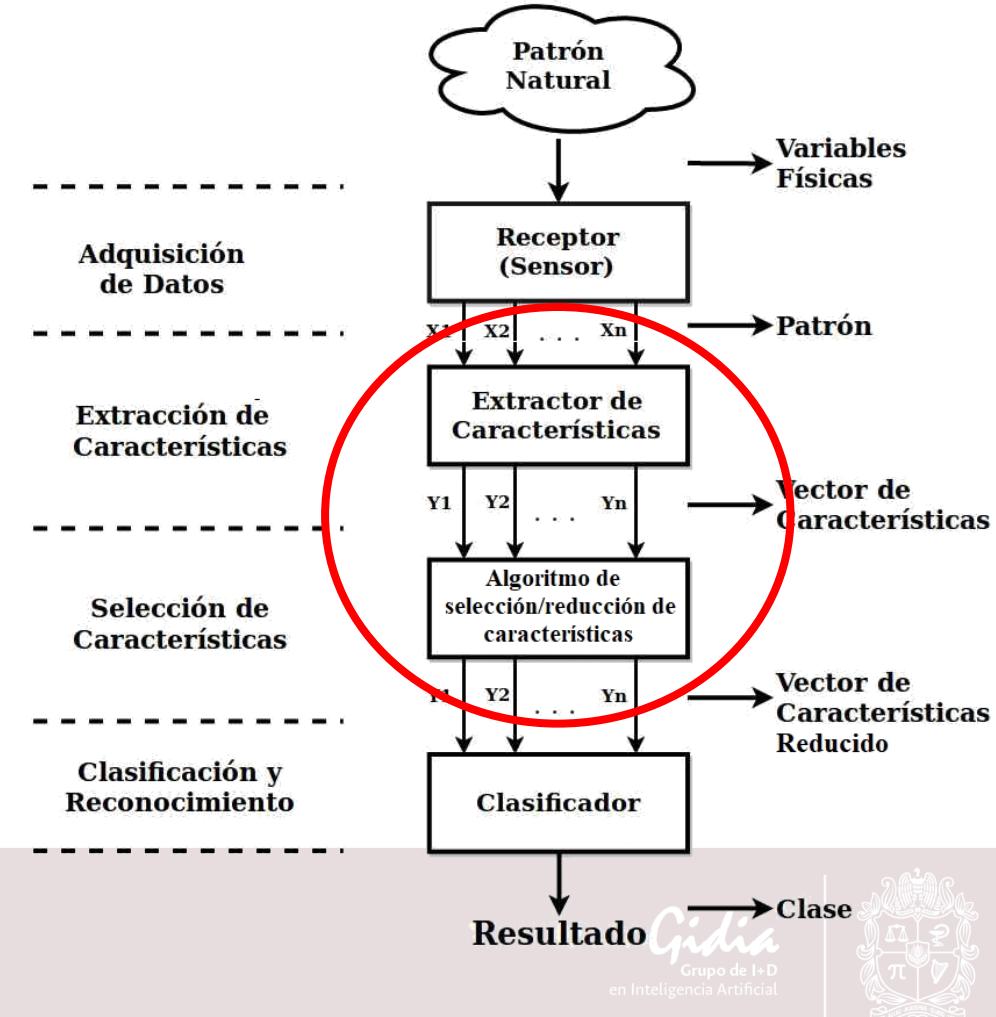
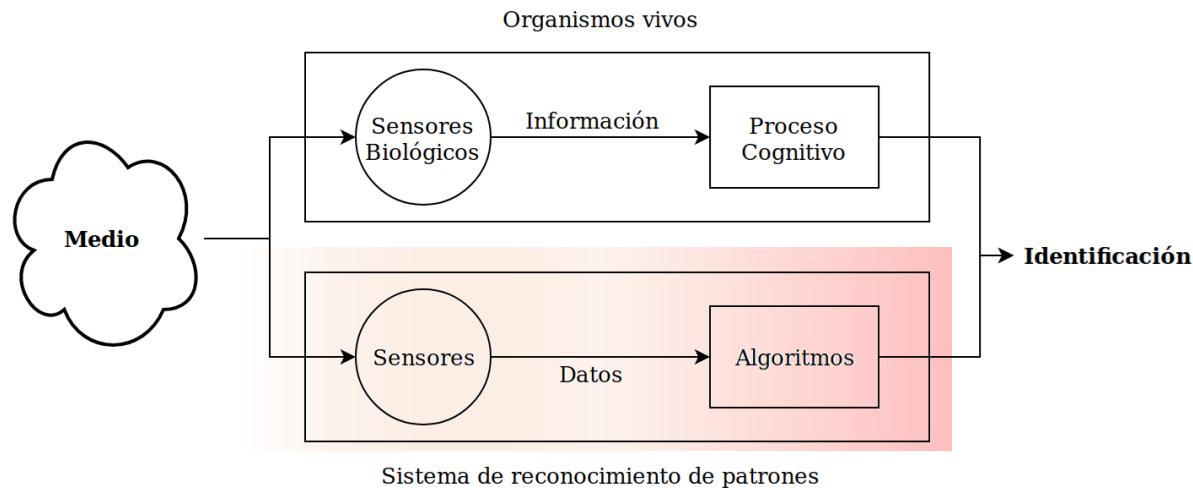
Investigador del Grupo de I+D en Inteligencia Artificial – GIDIA

jeespinosao@unal.edu.co

Contenido

1. Reaso
2. Extracción Vs Selección
3. Extracción de Características
 - a. Generación de Características
 - b. Promedios
4. Análisis de Componente Principal (PCA)
 - a. RFE – Recursive Feature Elimination
5. PCA y Descomposición Simple de Valores (SVD)
6. PCA para la Reducción de la Dimensionalidad
7. PCA para la Exploración de datos
8. PCA para la compresión de Imágenes
9. Lab 4 ☺

Etapas Sistema de Clasificación y Reconocimiento de Patrones



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:

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

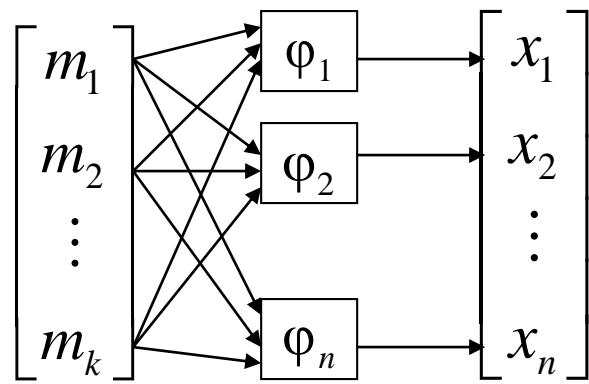
- El criterio para la reducción de características puede ser diferente de acuerdo a las diferentes configuraciones de problemas.
 - Configuración **no supervisada**: minimice la pérdida de información
 - Entorno **supervisado**: maximizar la discriminación de clase

Reducción de características vs. Selección de características

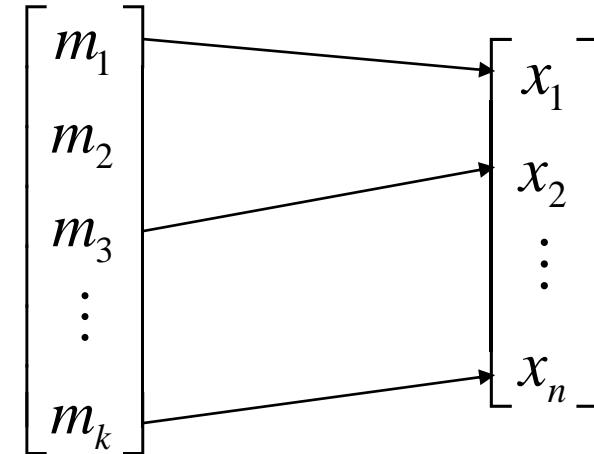
- **Reducción de características**
 - Se utilizan todas las características originales.
 - Las características transformadas son combinaciones lineales (en la mayoría de los casos) de las características originales.
- **Selección de las características**
 - Solo se selecciona un subconjunto de las características originales.

Métodos de extracción de Características

Extracción de Características



Selección de Características

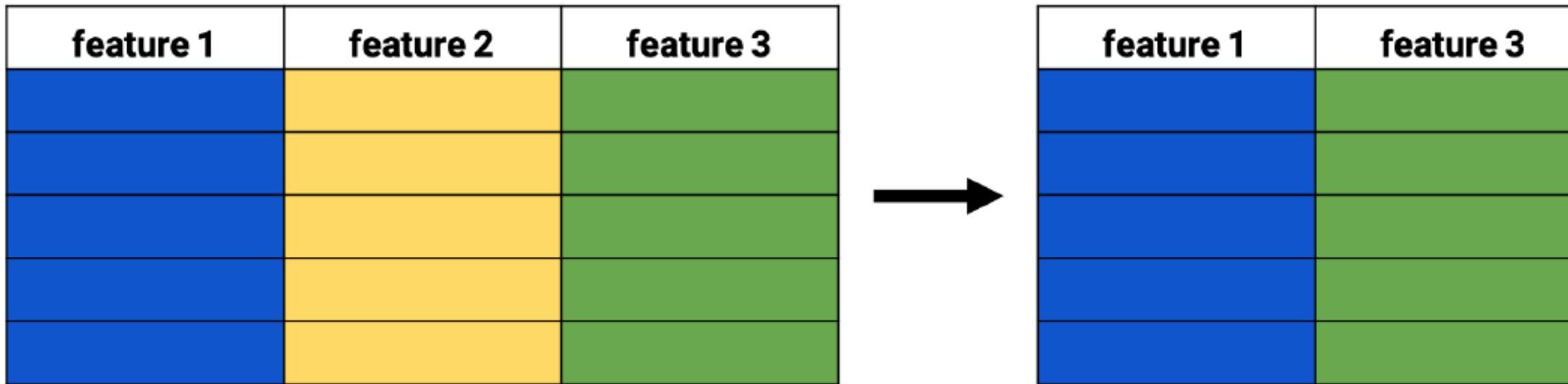


El problema puede expresarse como la optimización de los parámetros $\varphi(\theta)$ del extractor de características.

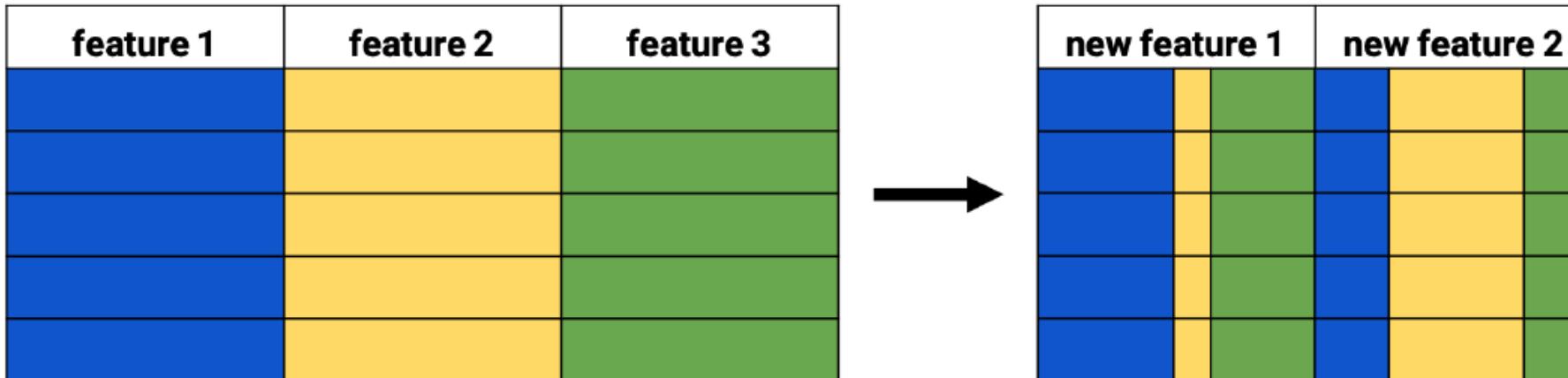
Métodos supervisados: la función objetivo es un criterio de separabilidad (discriminabilidad) de ejemplos etiquetados, por ejemplo, análisis de discriminación lineal (LDA).

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).

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

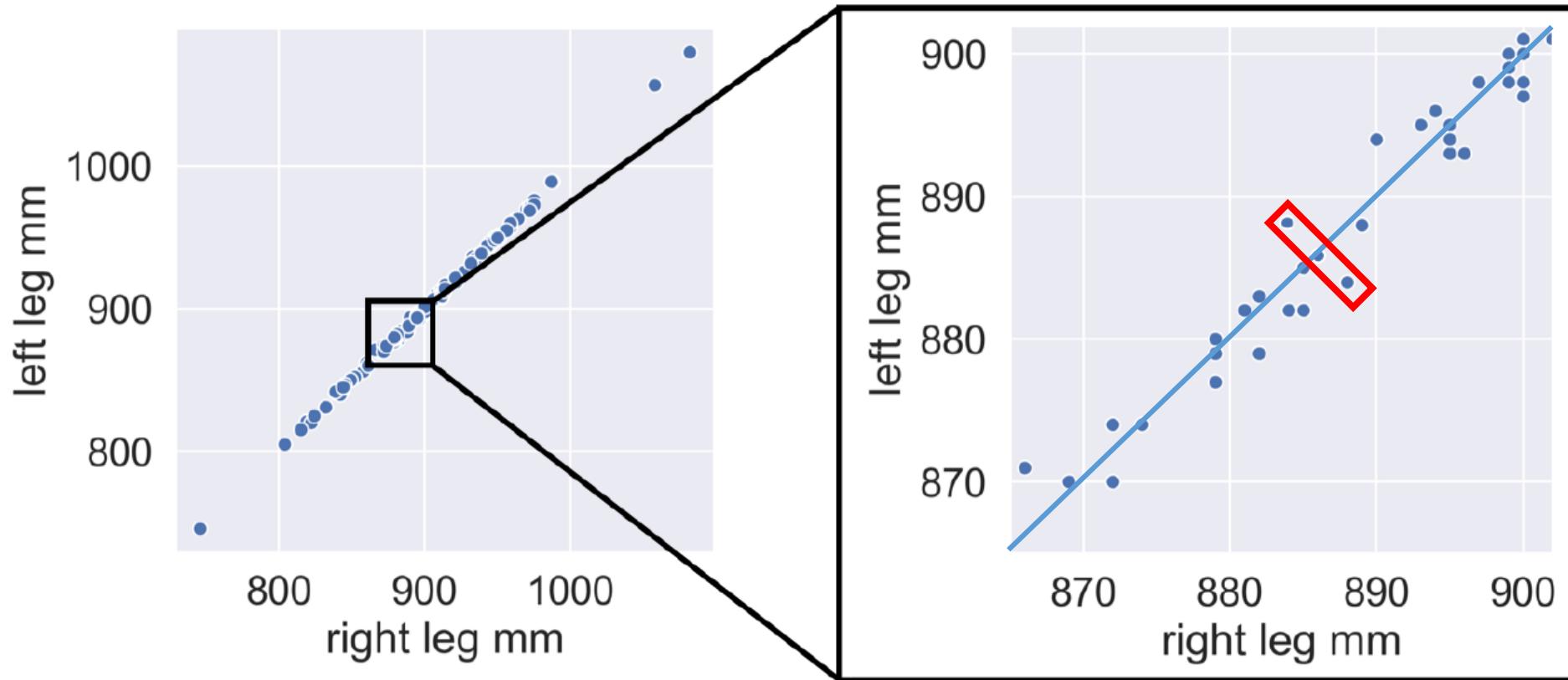
Feature generation - averages

```
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

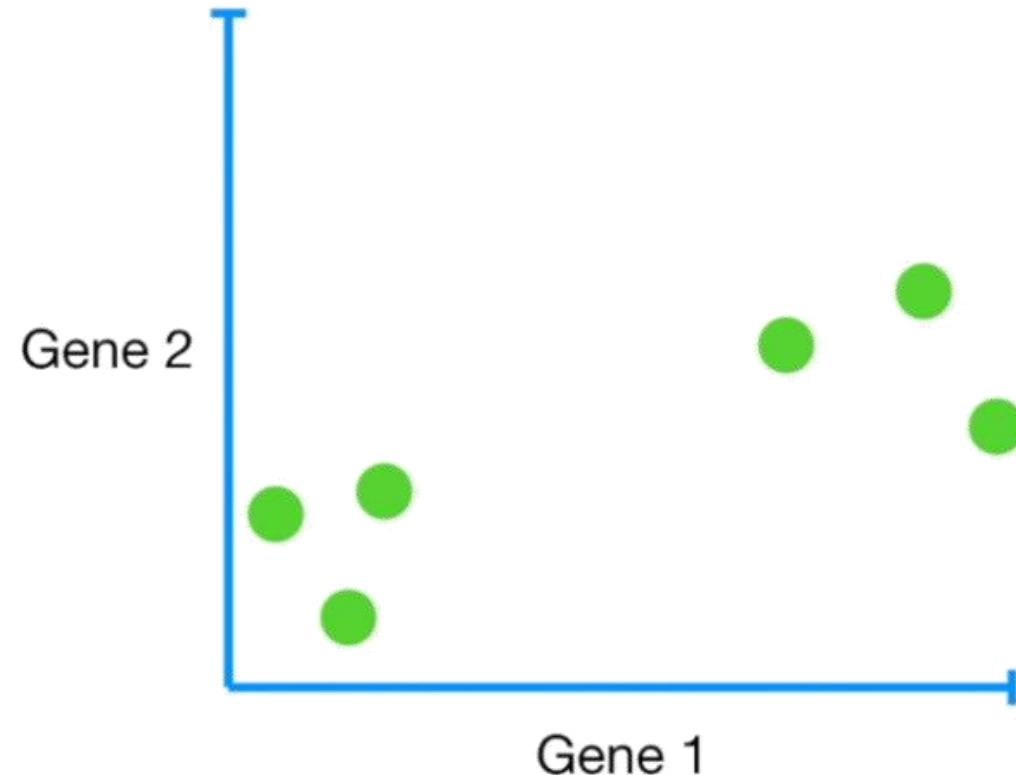
| | Mouse 1 | Mouse 2 | Mouse 3 | Mouse 4 | Mouse 5 | Mouse 6 |
|--------|---------|---------|---------|---------|---------|---------|
| Gene 1 | 10 | 11 | 8 | 3 | 2 | 1 |
| Gene 2 | 6 | 4 | 5 | 3 | 2.8 | 1 |

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 |

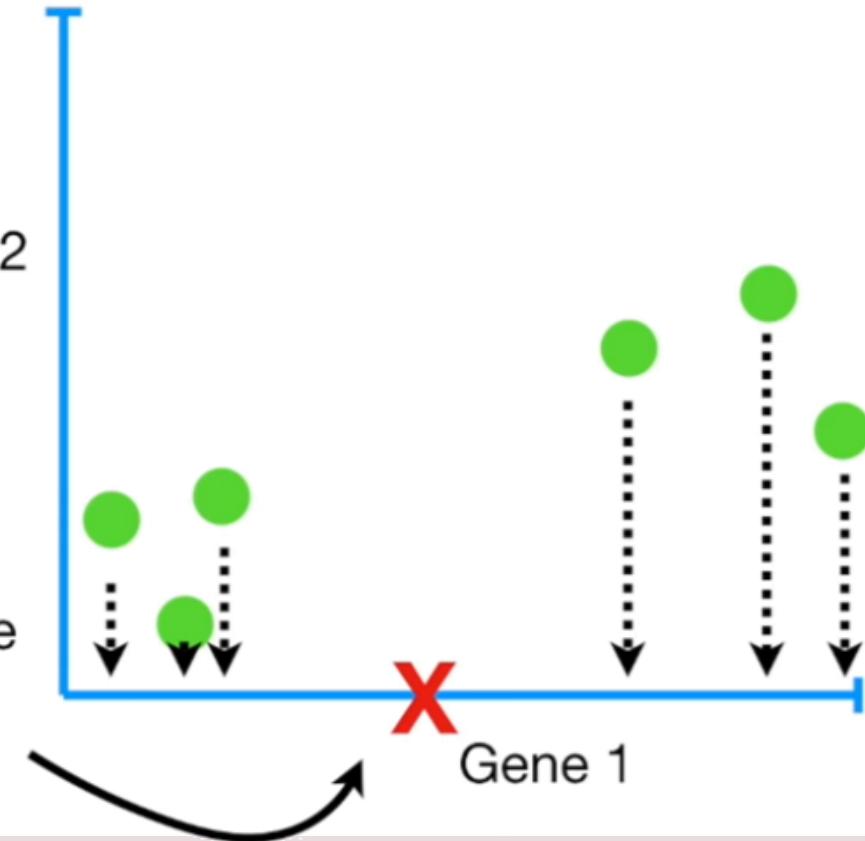
We'll start by plotting the data...



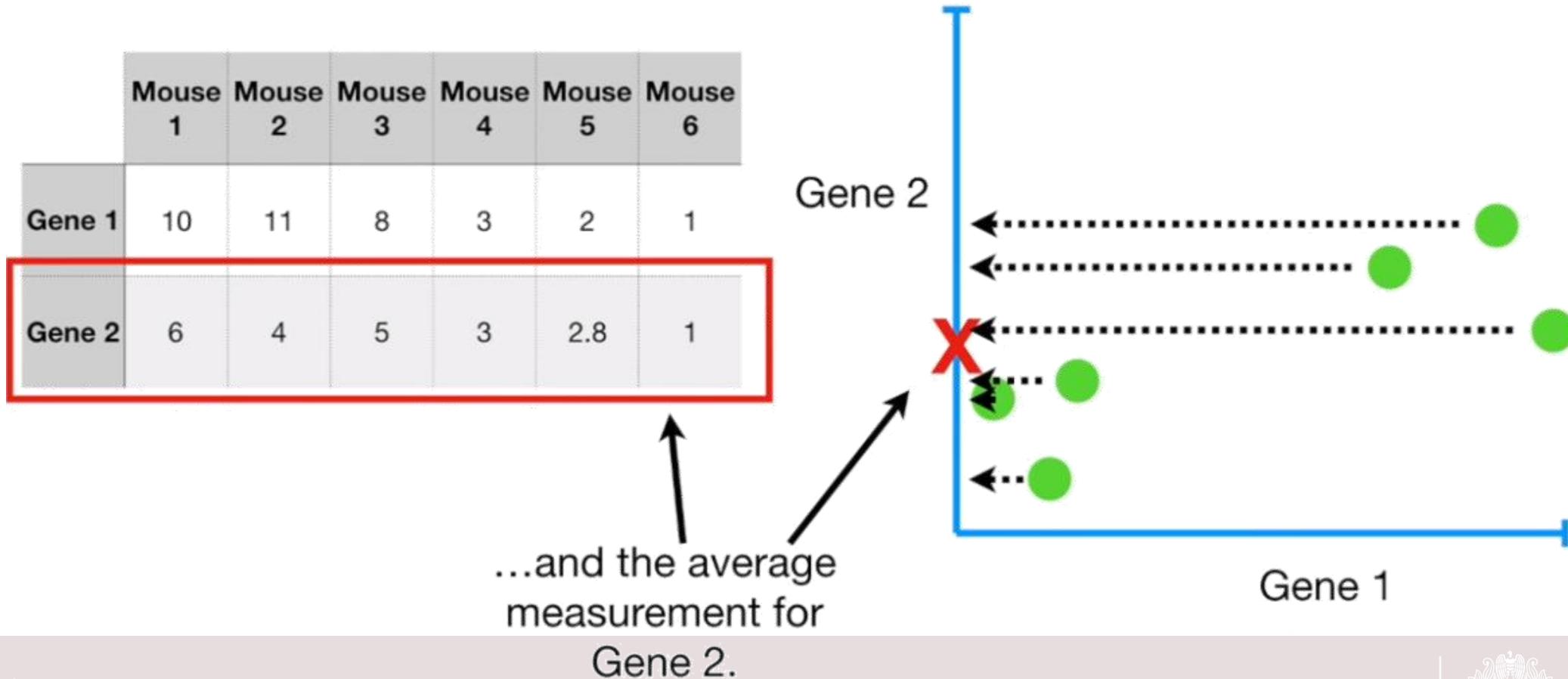
Extracción de Características - PCA

| | 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 |

Then we'll calculate
the average
measurement for
Gene 1...

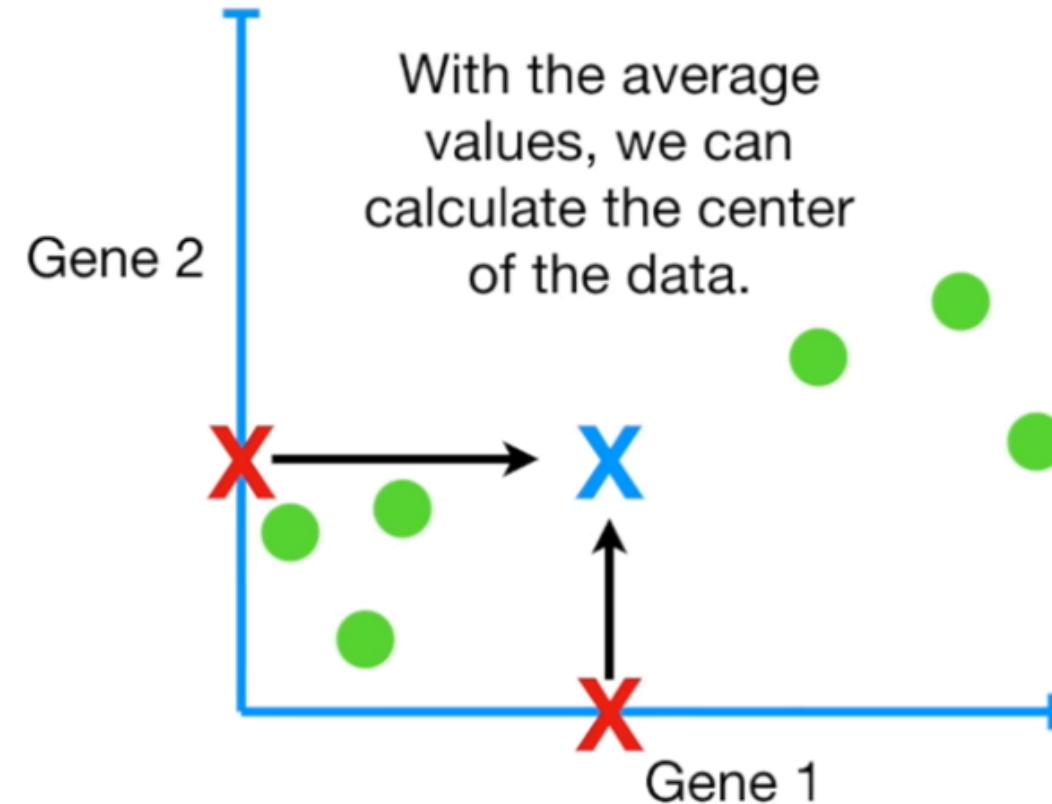


Extracción de Características - PCA

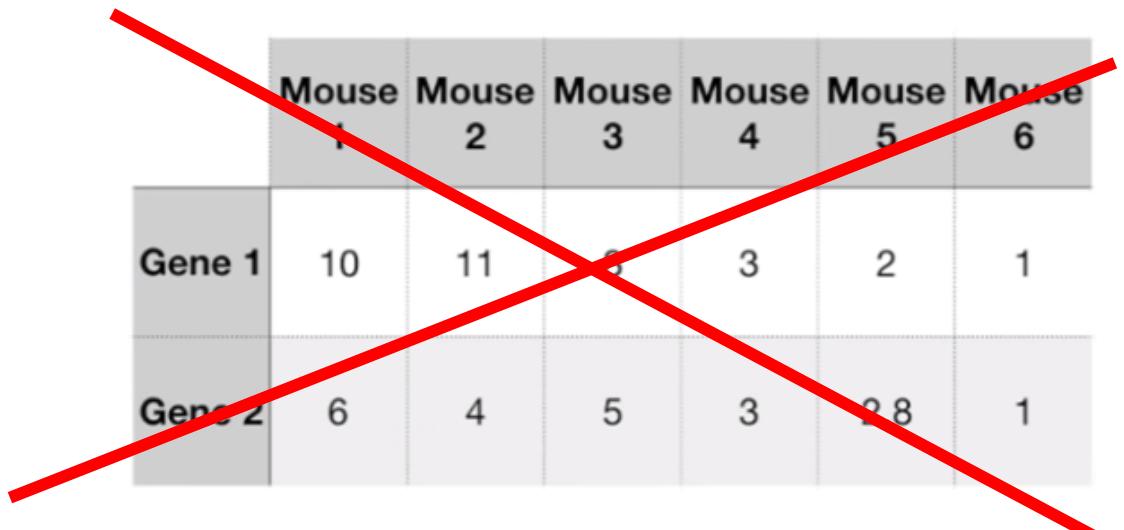


Extracción de Características - PCA

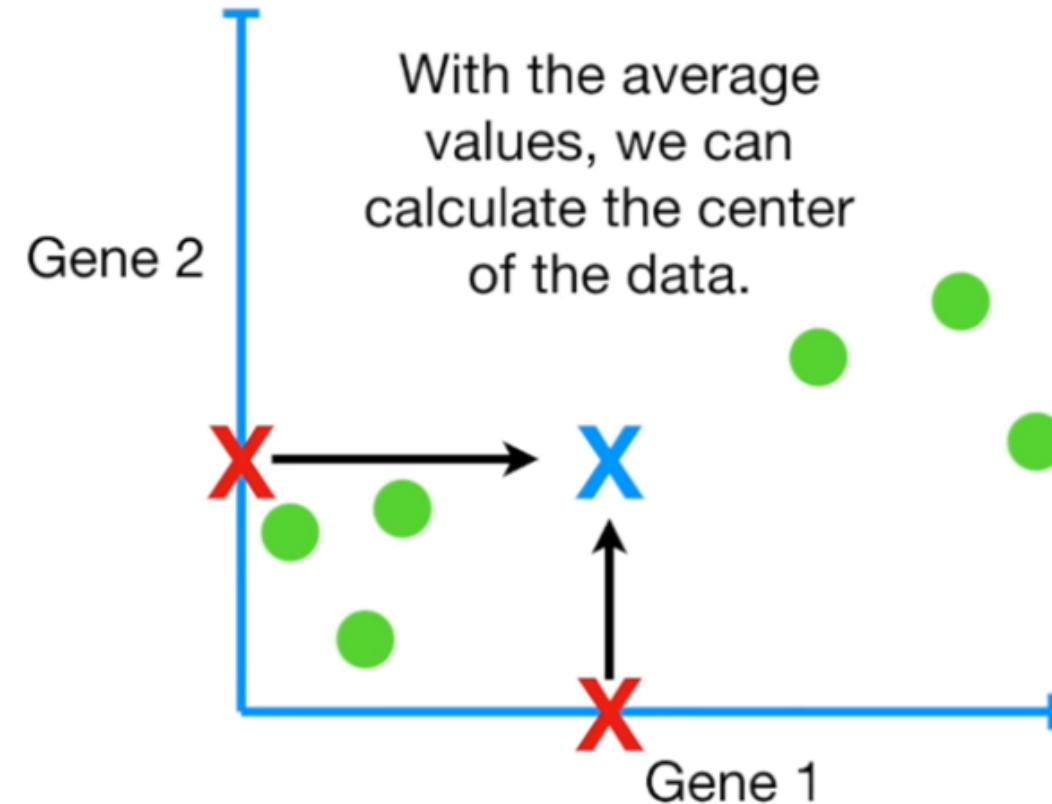
| | Mouse 1 | Mouse 2 | Mouse 3 | Mouse 4 | Mouse 5 | Mouse 6 |
|--------|---------|---------|---------|---------|---------|---------|
| Gene 1 | 10 | 11 | 8 | 3 | 2 | 1 |
| Gene 2 | 6 | 4 | 5 | 3 | 2.8 | 1 |



Extracción de Características - PCA

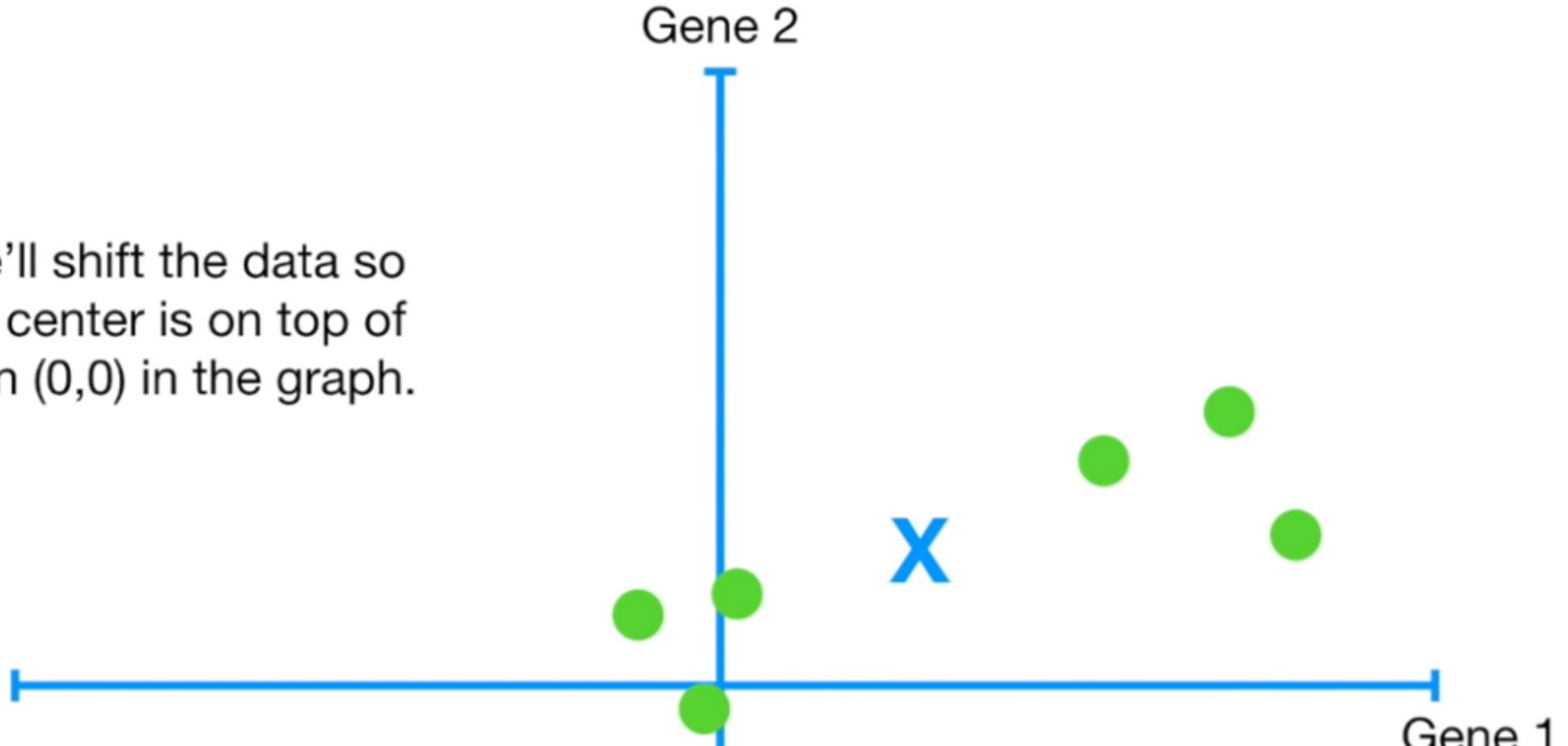


| | Mouse | | | | | |
|--------|-------|----|---|---|-----|---|
| | 1 | 2 | 3 | 4 | 5 | 6 |
| Gene 1 | 10 | 11 | 9 | 3 | 2 | 1 |
| Gene 2 | 6 | 4 | 5 | 3 | 2.8 | 1 |



Extracción de Características - PCA

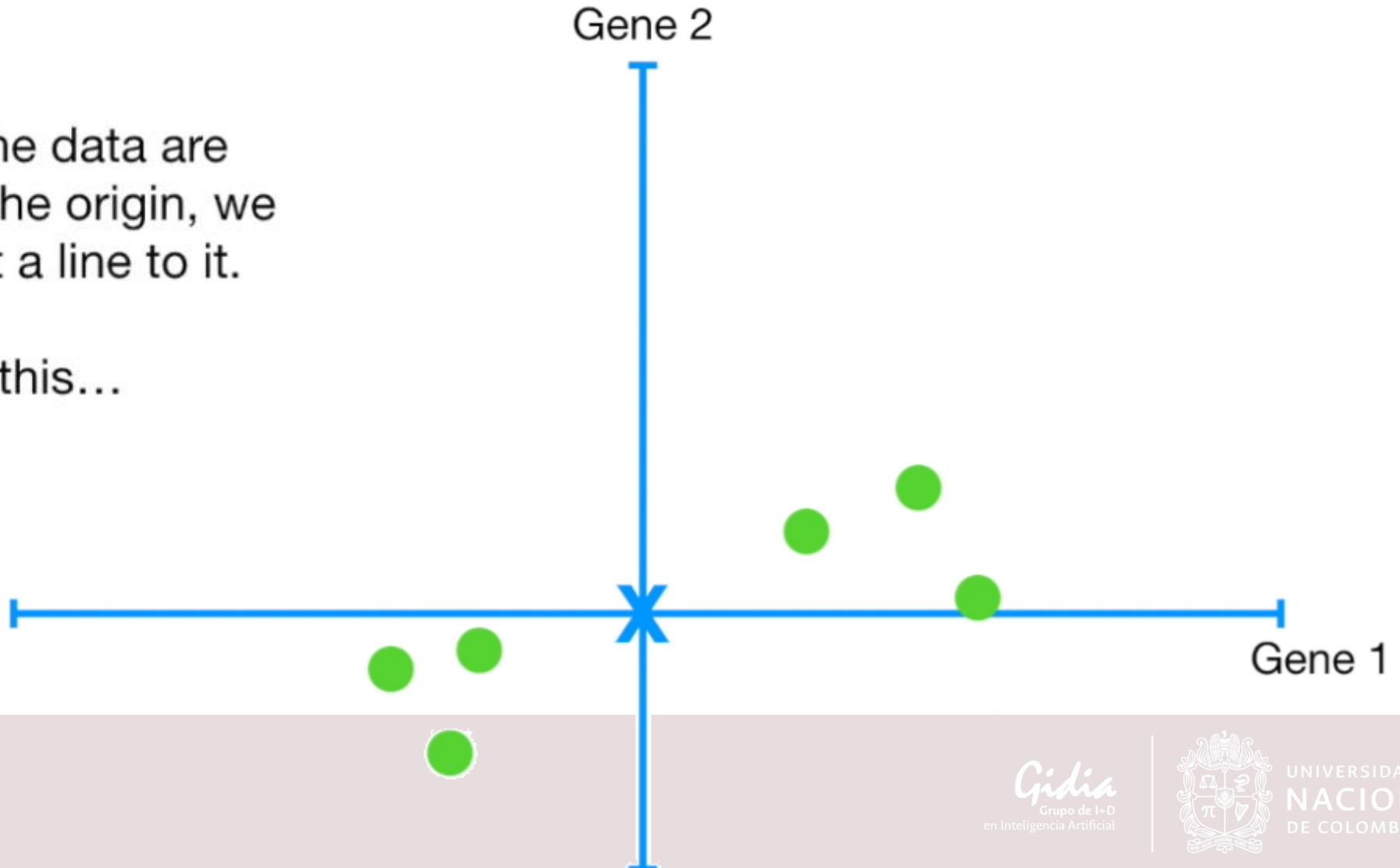
Now we'll shift the data so that the center is on top of the origin (0,0) in the graph.



Extracción de Características - PCA

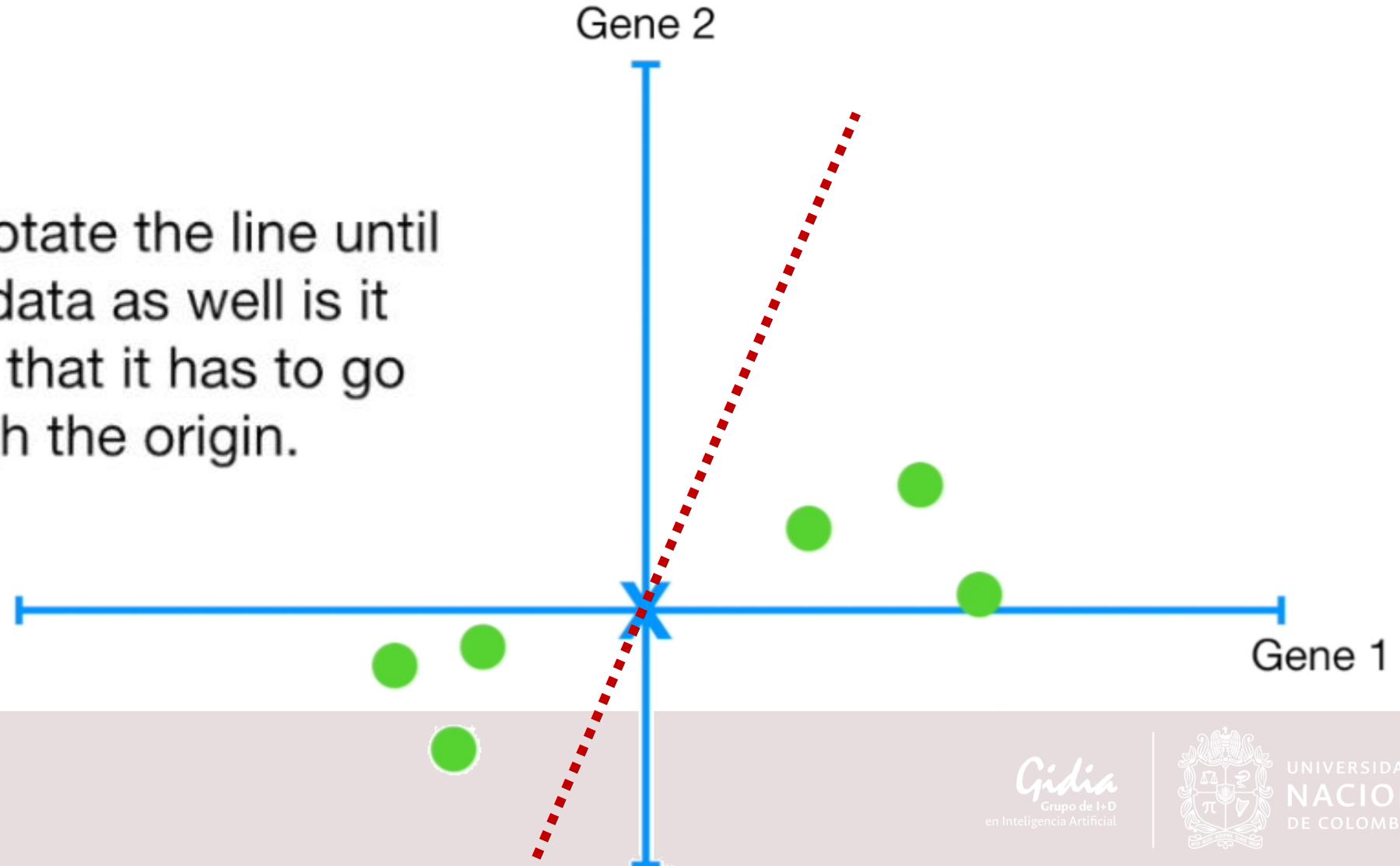
Now that the data are centered on the origin, we can try to fit a line to it.

To do this...



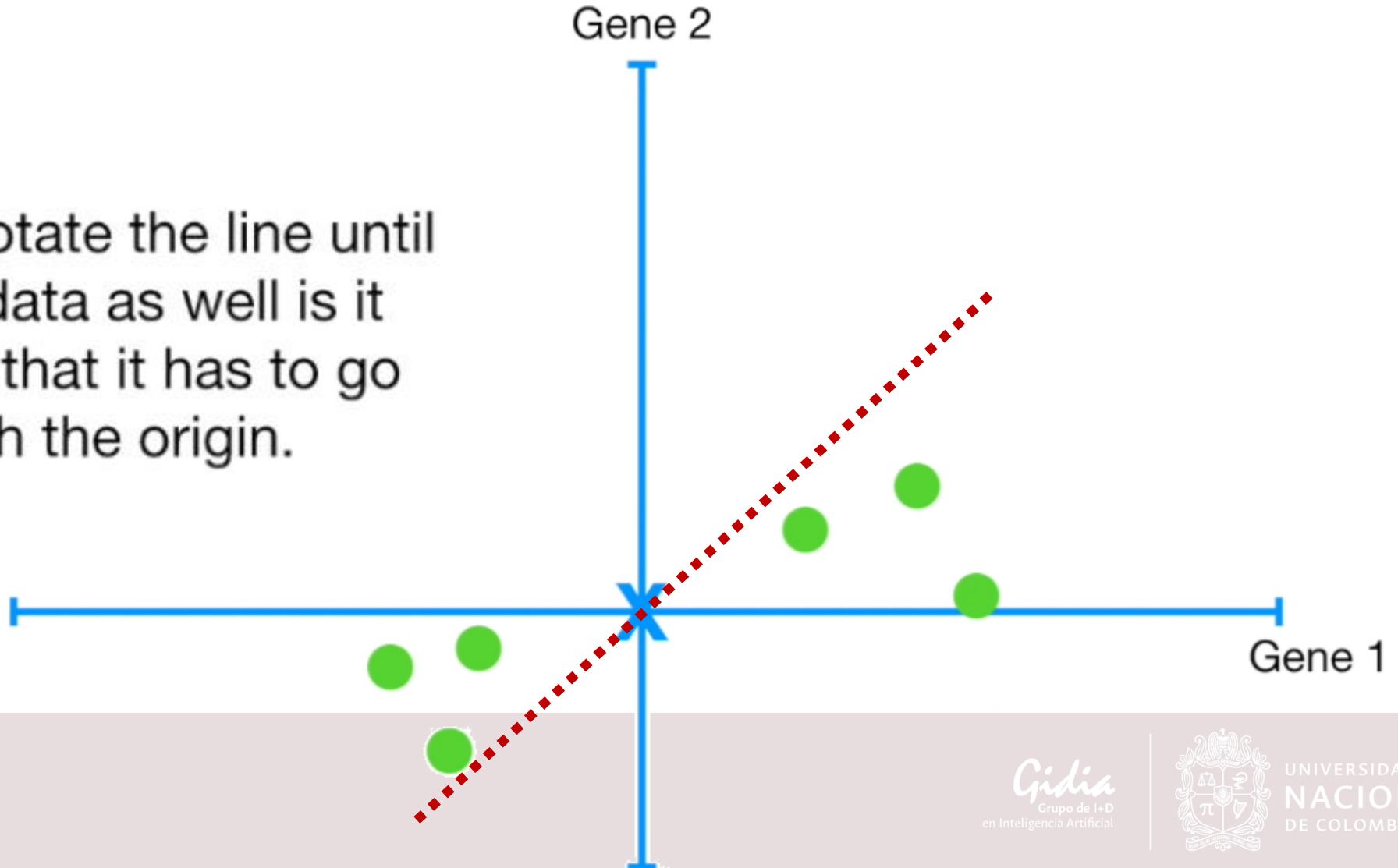
Extracción de Características - PCA

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



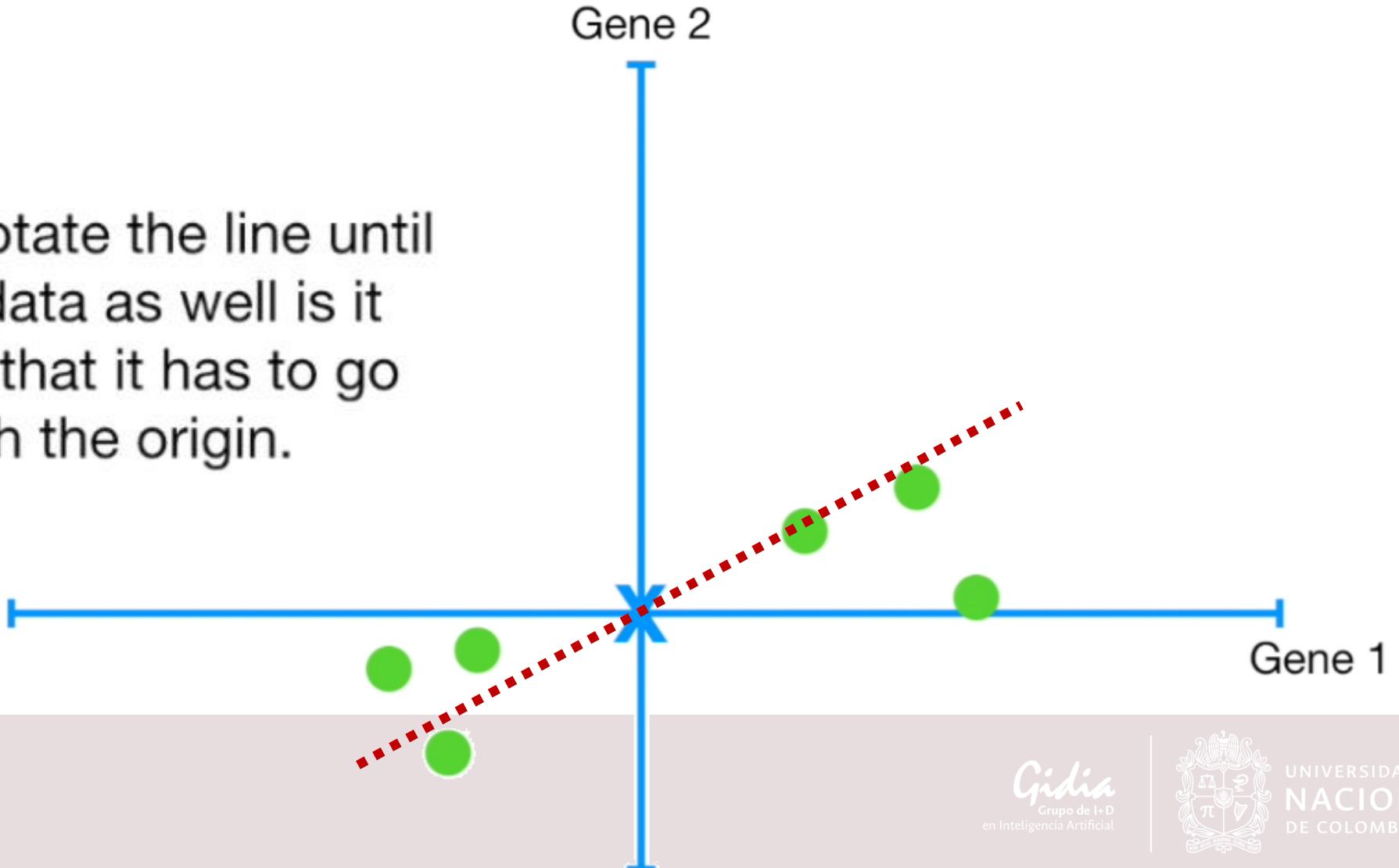
Extracción de Características - PCA

...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

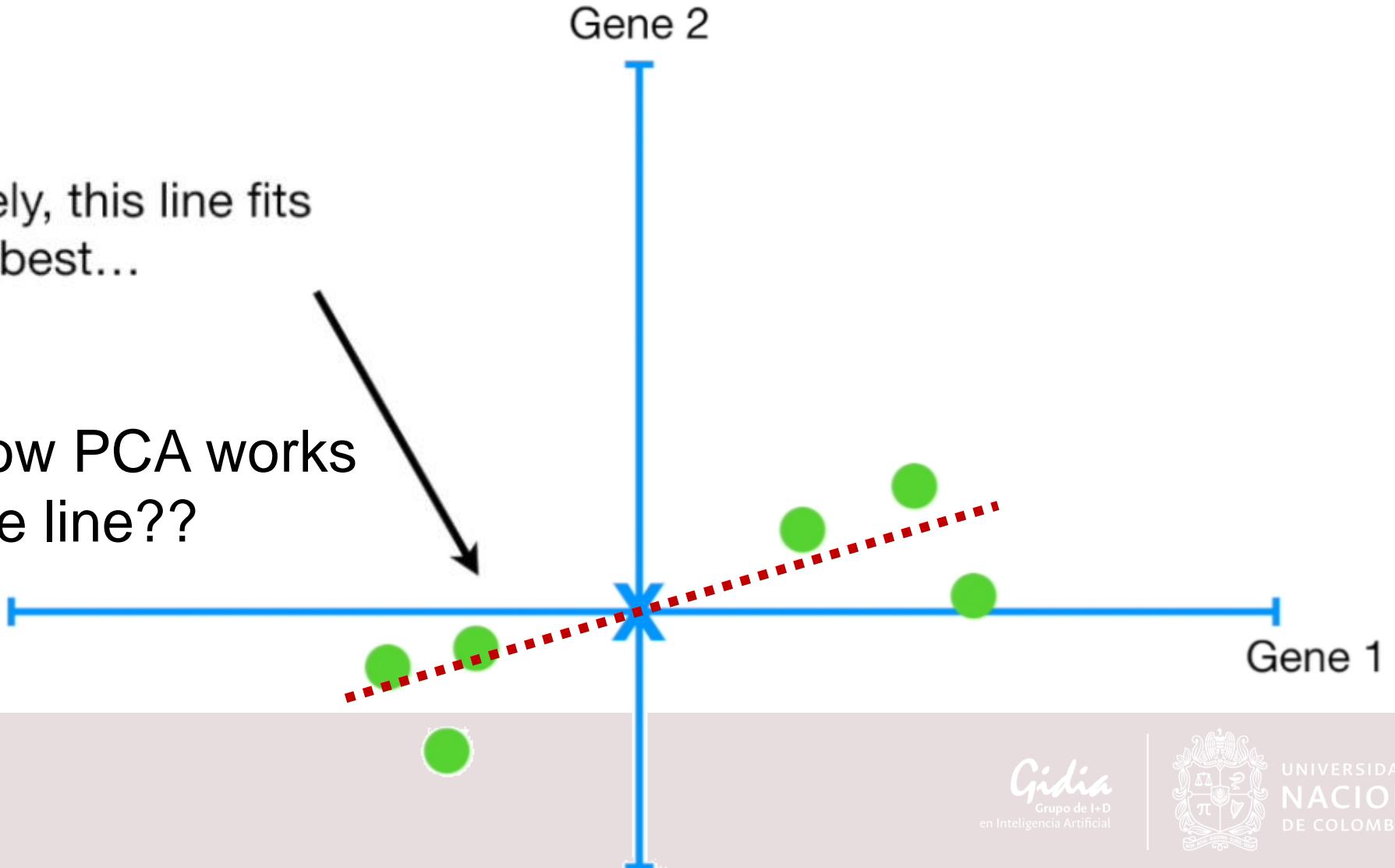
...then we rotate the line until it fits the data as well as it can, given that it has to go through the origin.



Extracción de Características - PCA

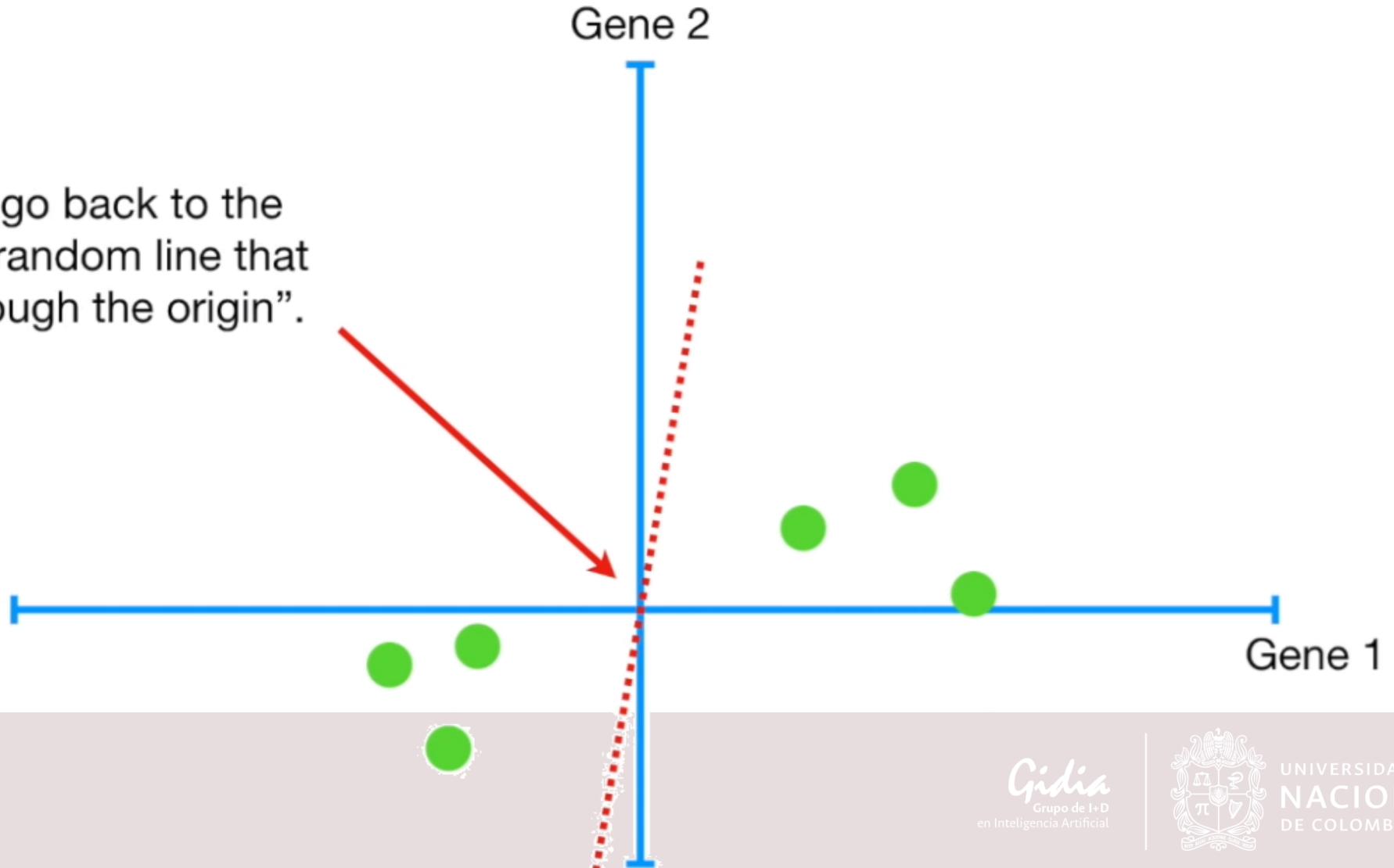
Ultimately, this line fits best...

But; How PCA works to fit the line??



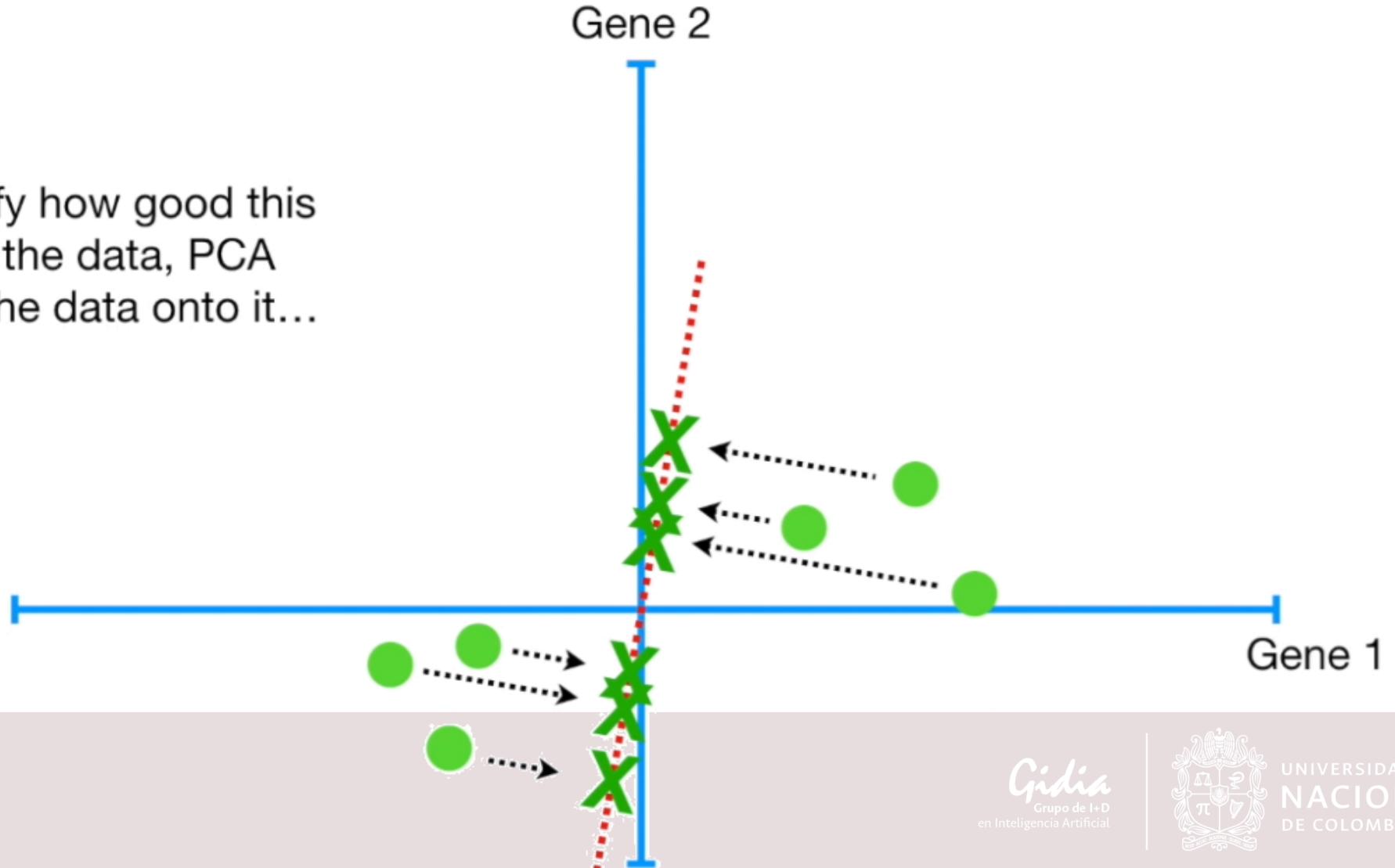
Extracción de Características - PCA

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

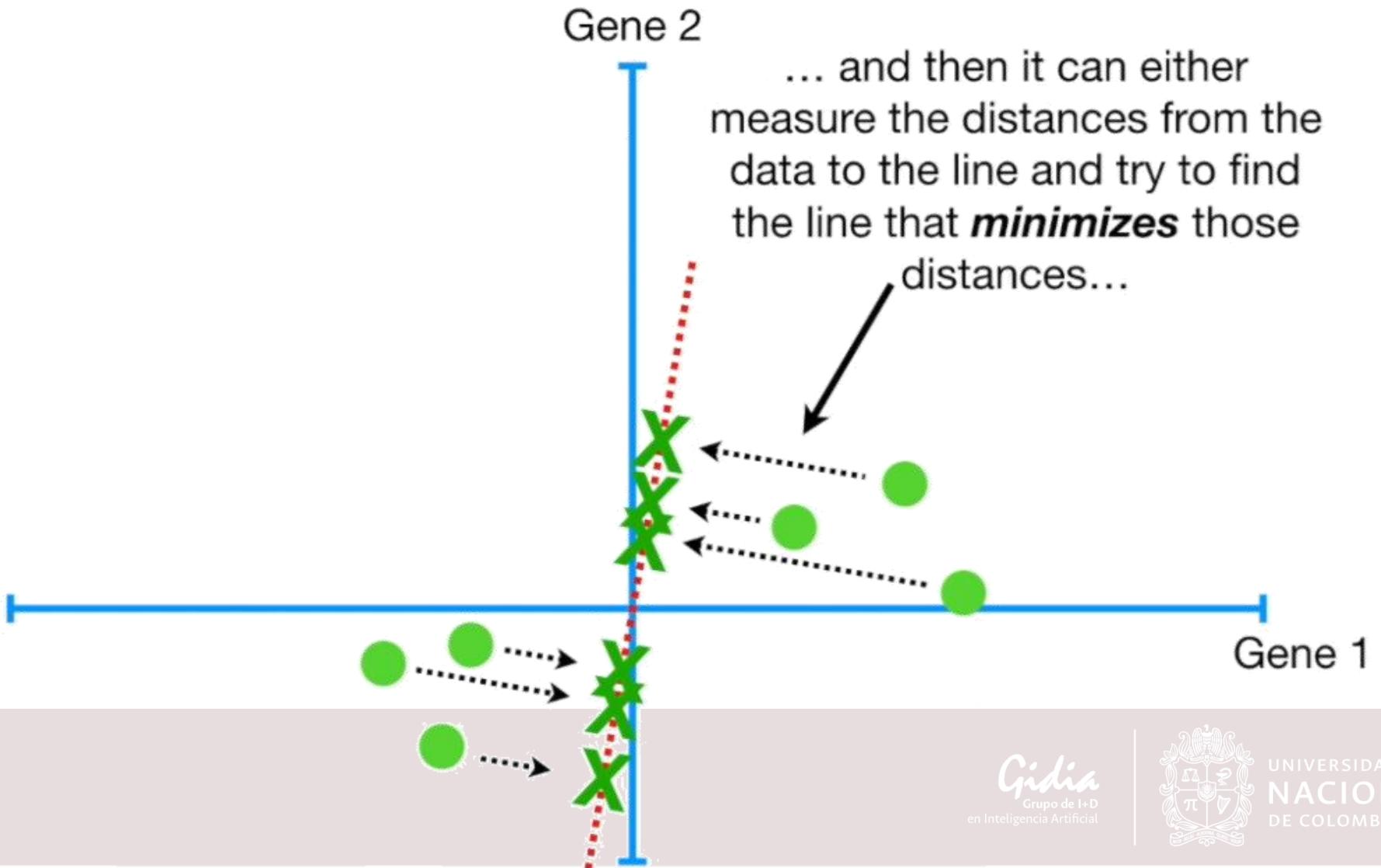


Extracción de Características - PCA

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

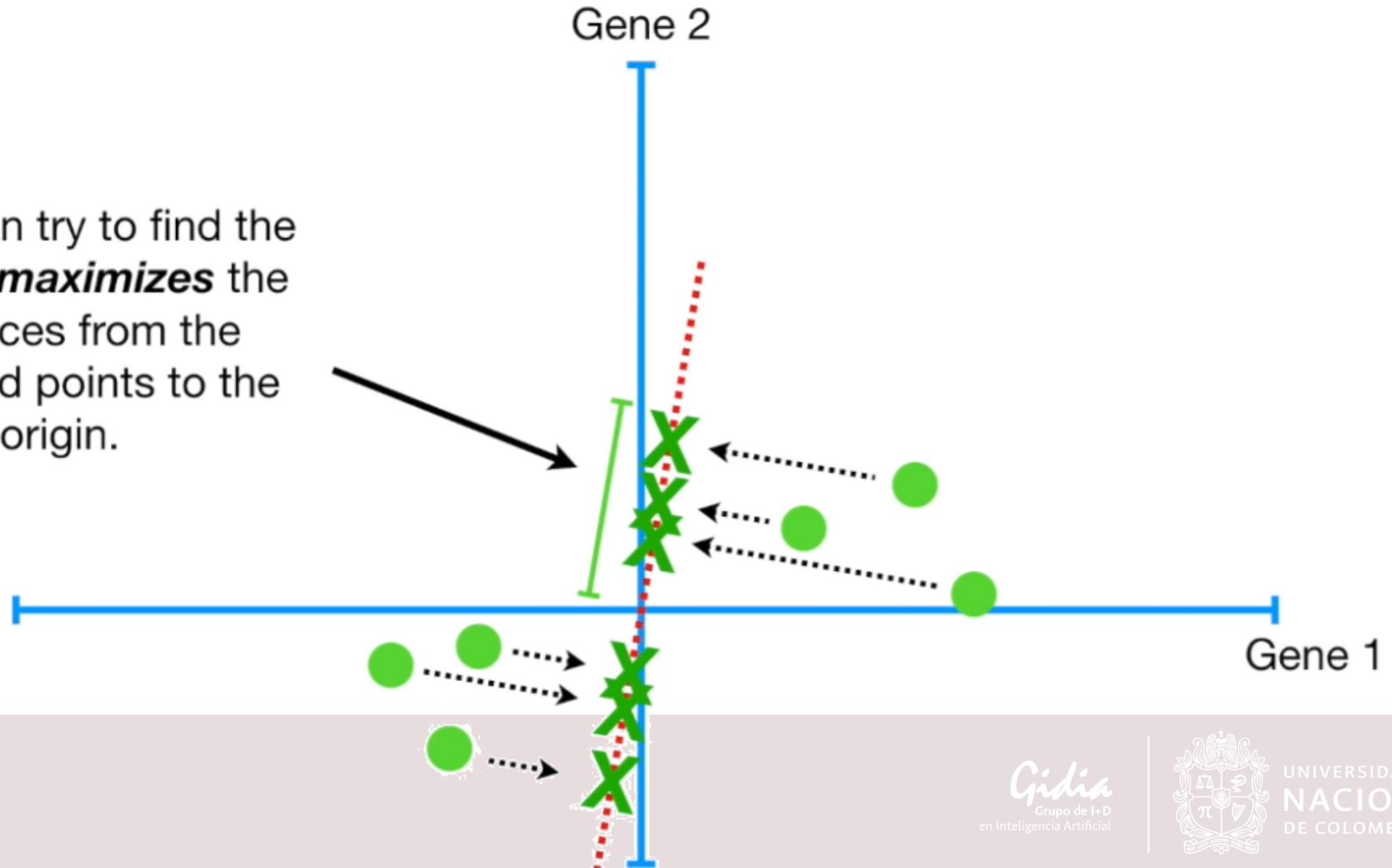


Extracción de Características - PCA



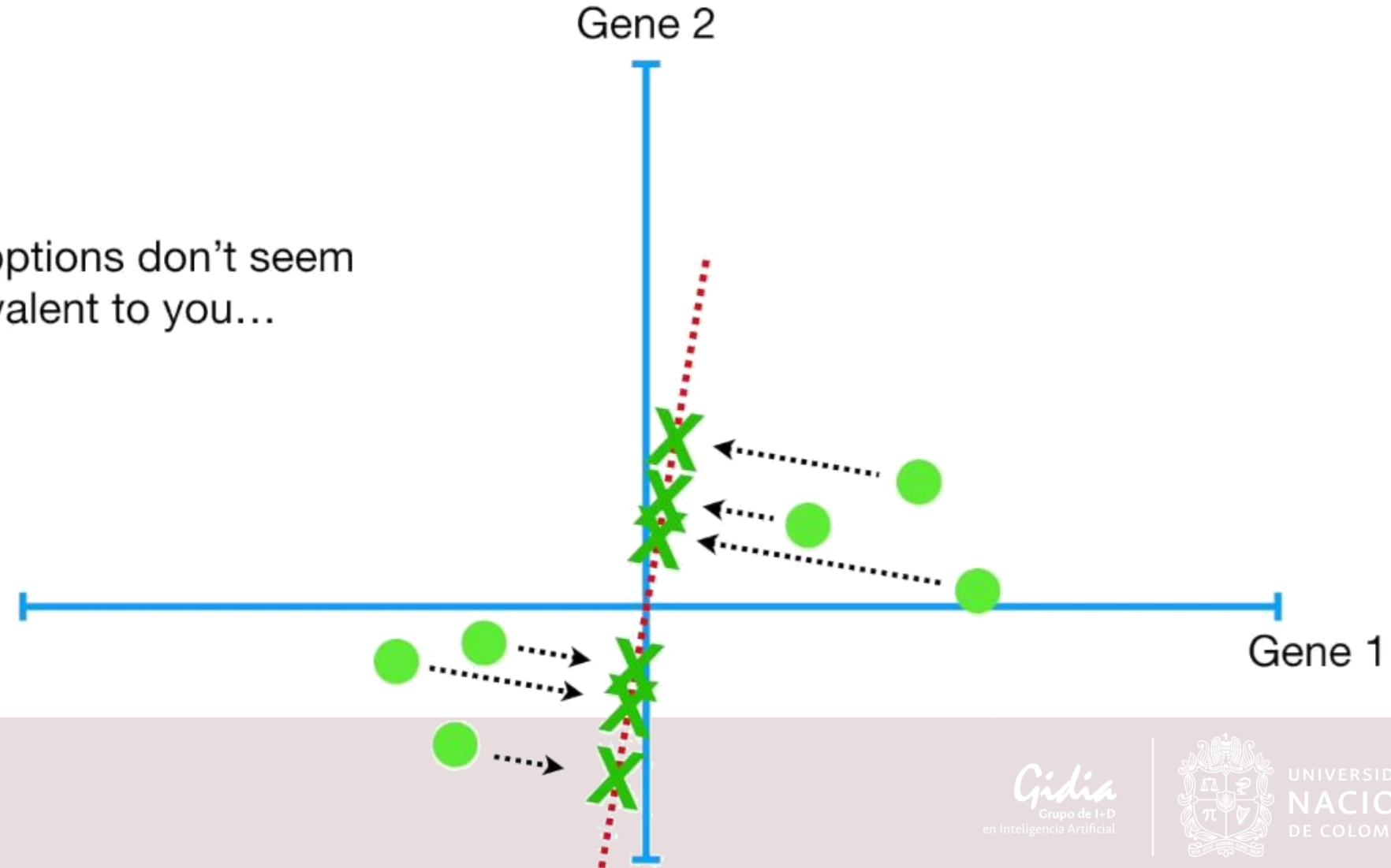
Extracción de Características - PCA

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

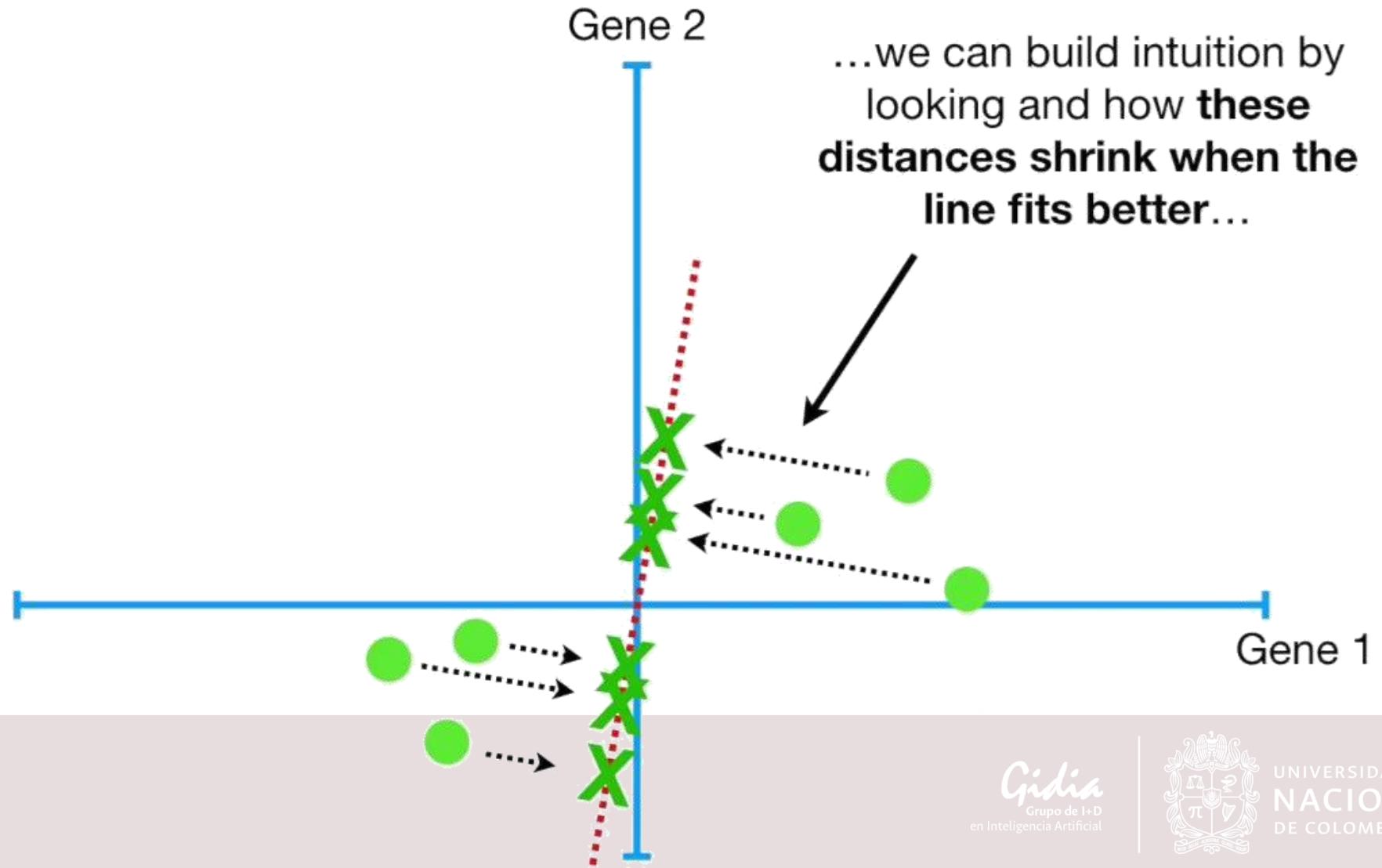


Extracción de Características - PCA

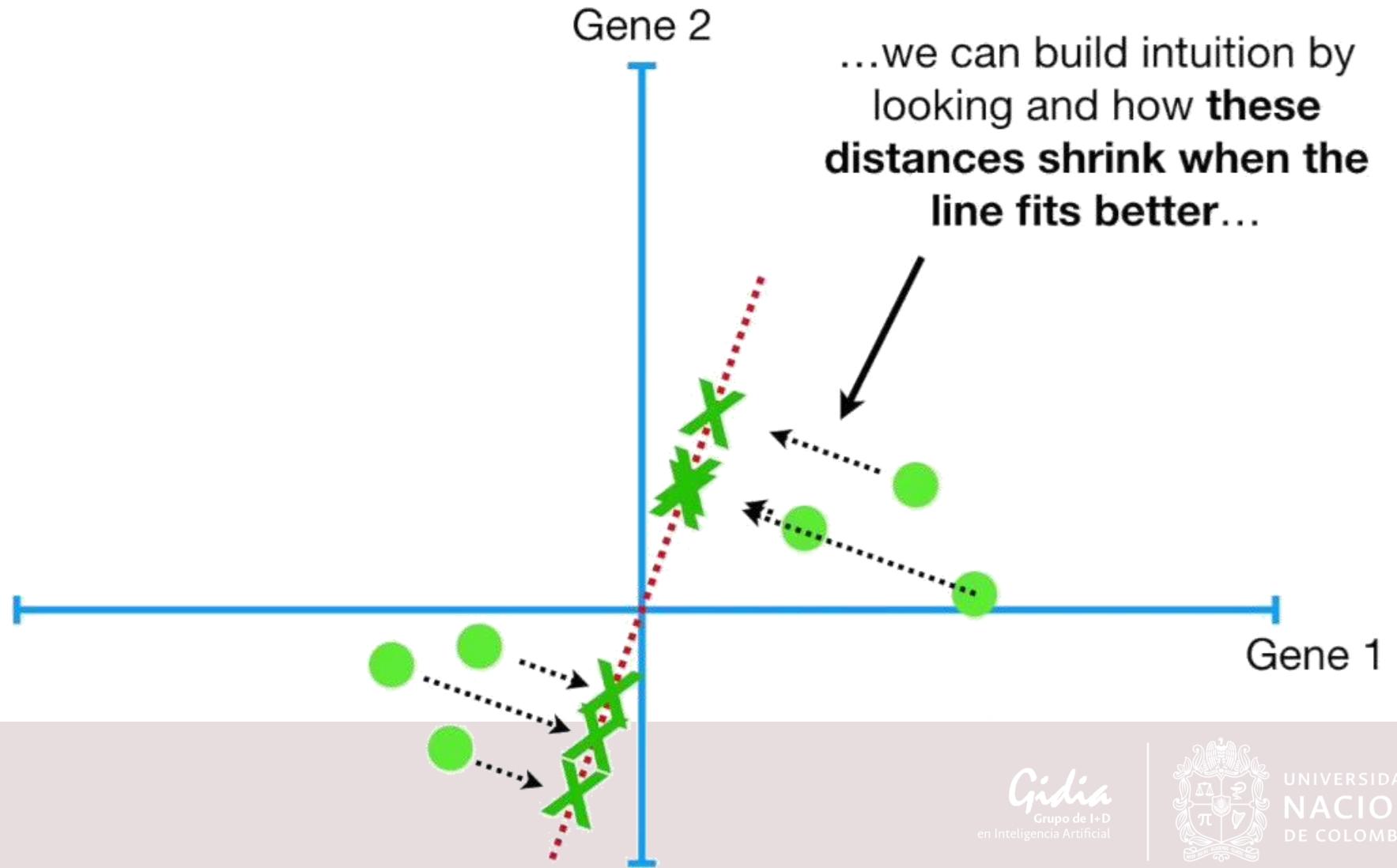
If those options don't seem equivalent to you...



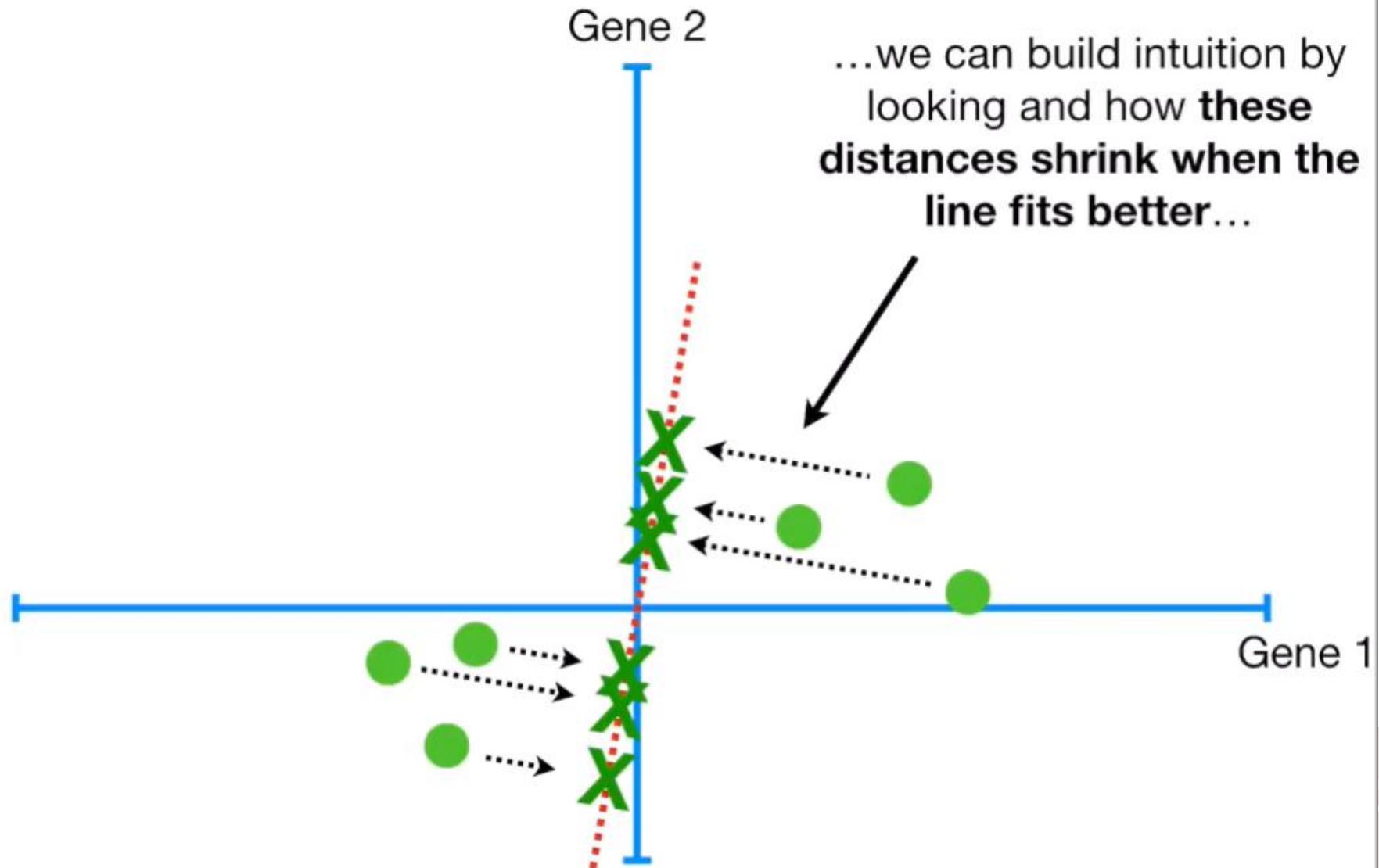
Extracción de Características - PCA



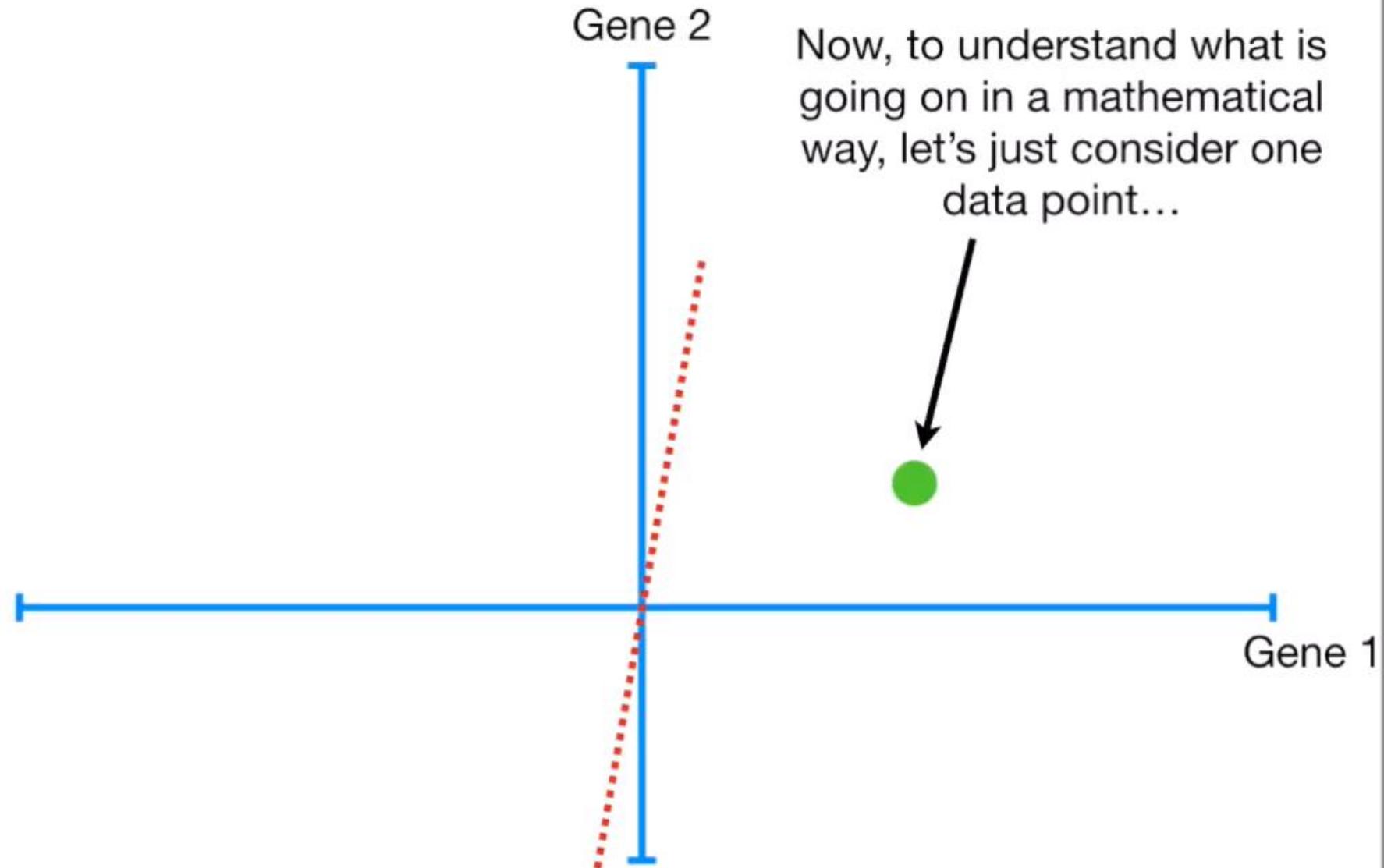
Extracción de Características - PCA



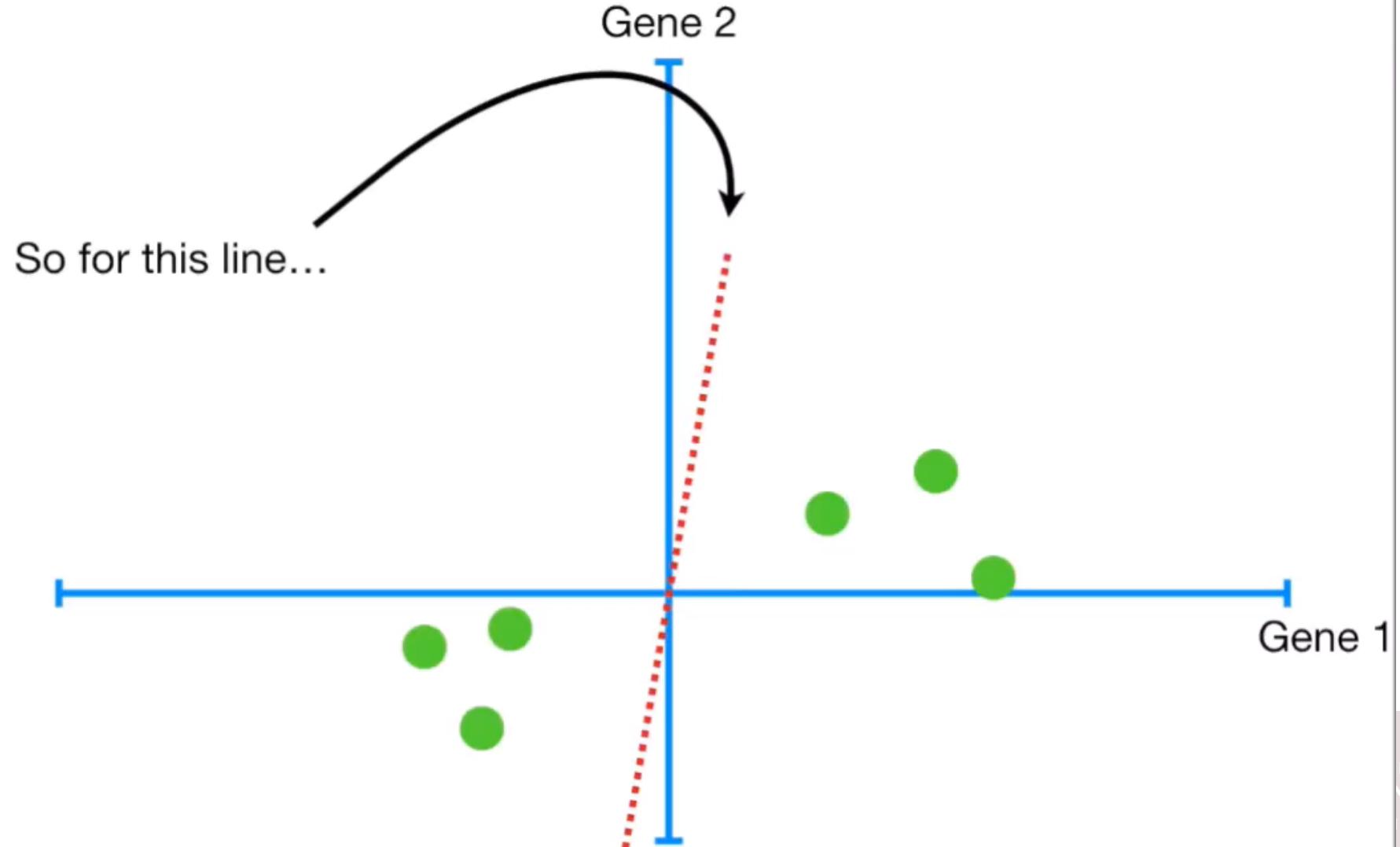
Extracción de Características - PCA



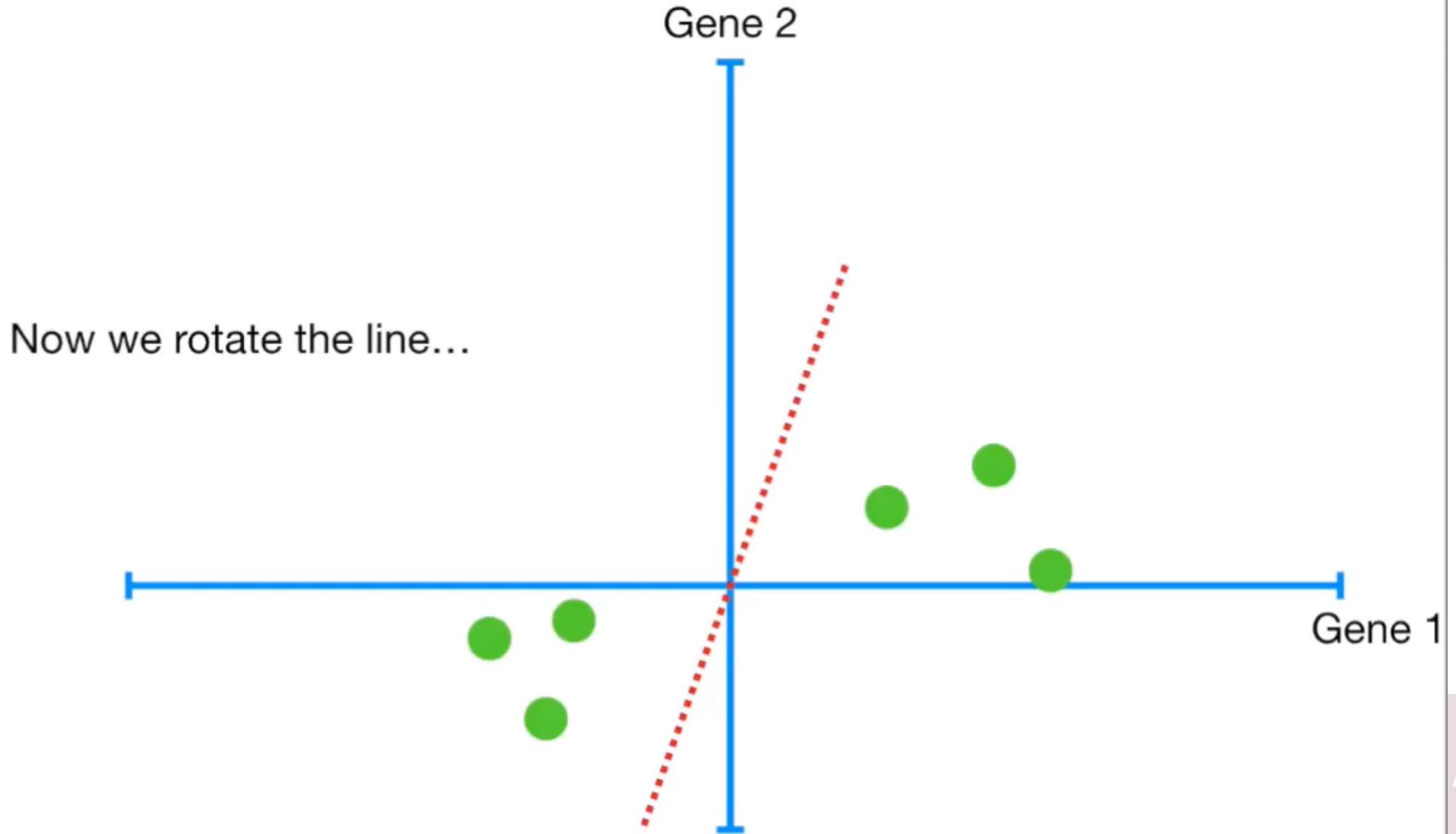
Extracción de Características - PCA



Extracción de Características - PCA

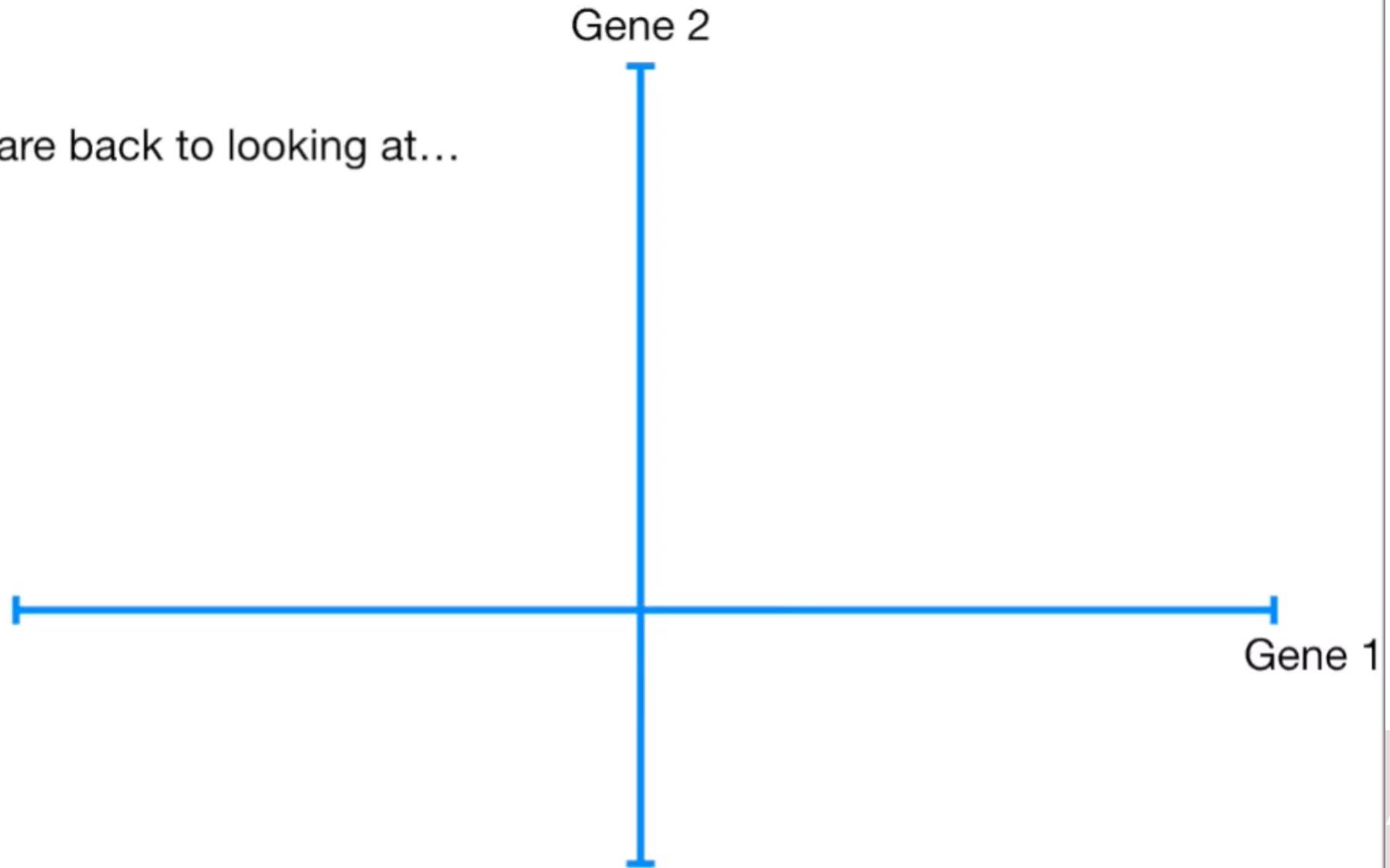


Extracción de Características - PCA



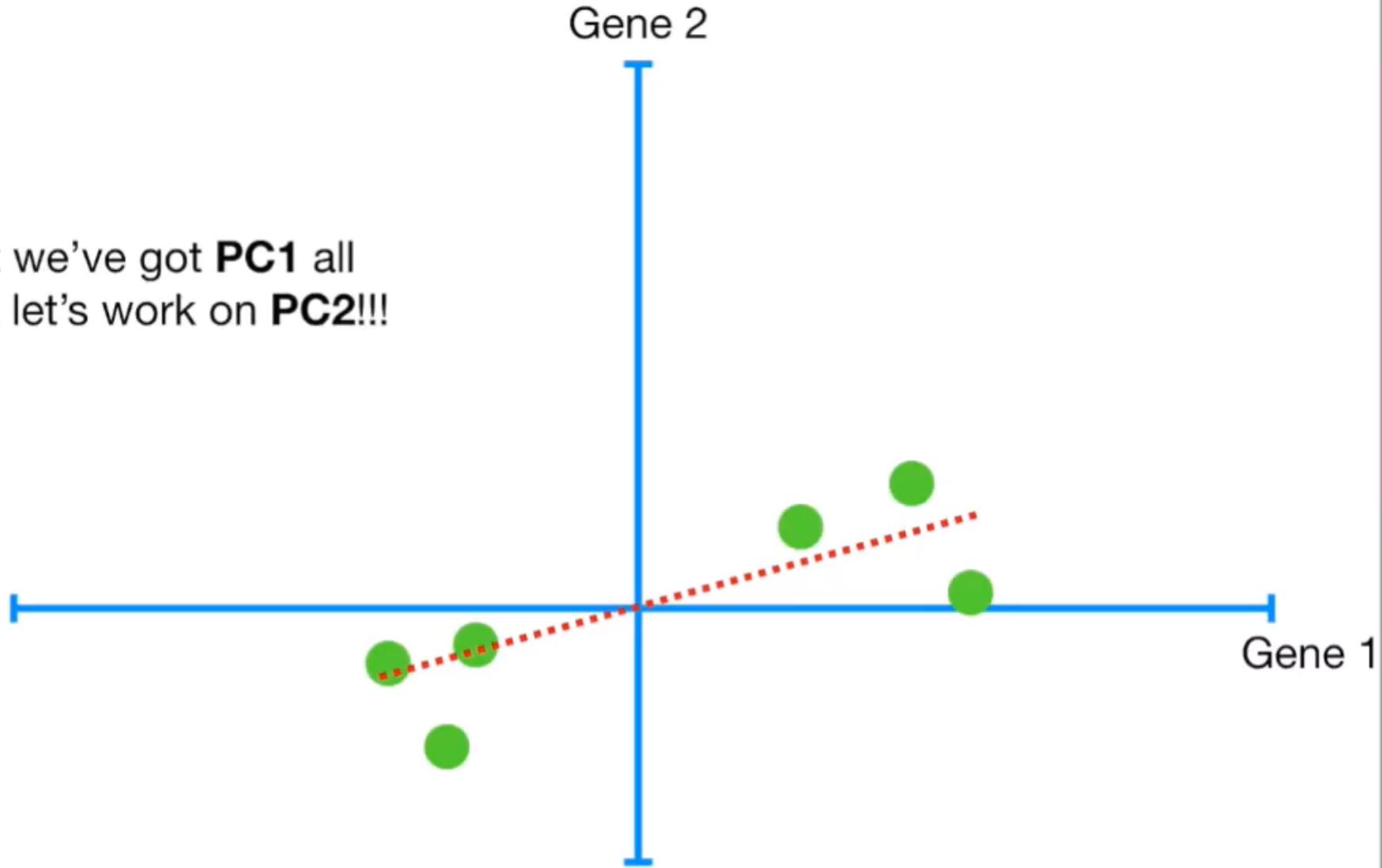
Extracción de Características - PCA

So now we are back to looking at...

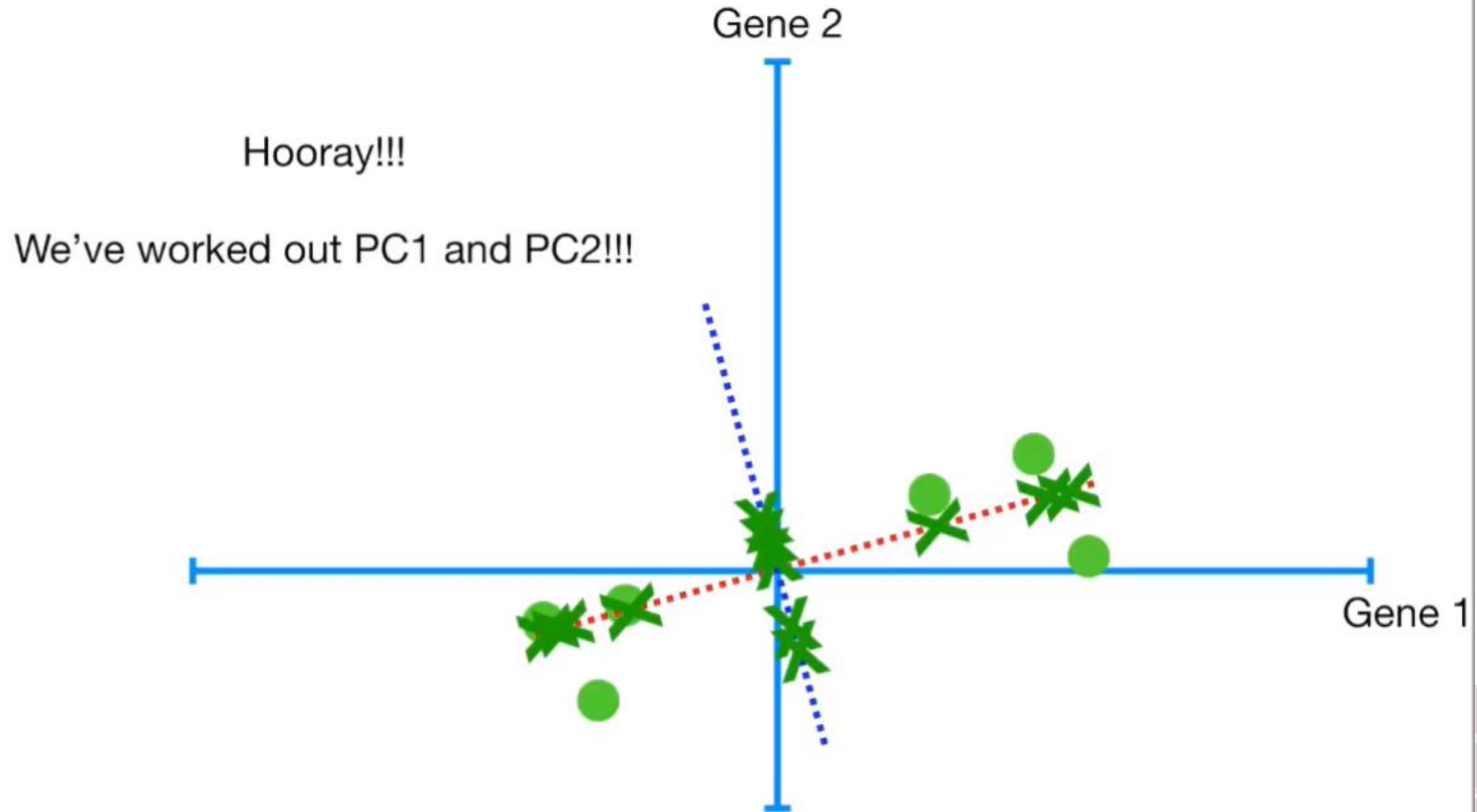


Extracción de Características - PCA

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



Extracción de Características - PCA

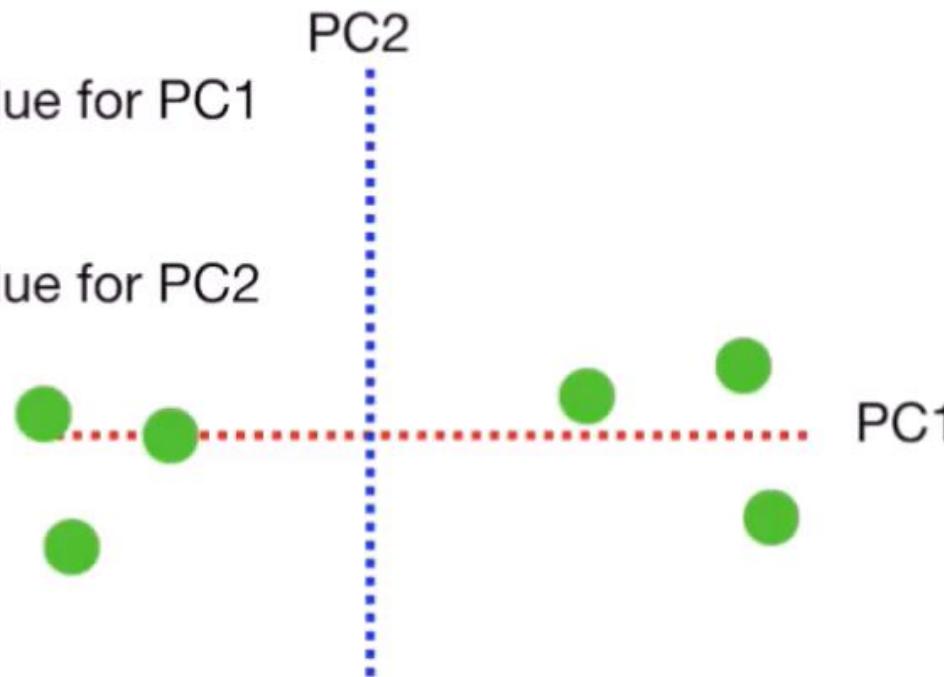


Extracción de Características - PCA

Remember the eigenvalues?

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

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



PCA y SVD

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

m: dimensiones (measurements)

$$T = X W$$

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

Principal Component Analysis

eigen decomposition de

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

$$T_r = X W_r$$

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

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

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

$$(X^T X) \rightarrow W \\ (m \times m) \quad \lambda$$

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

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*

$$V^* V = I \\ U^T U = I$$

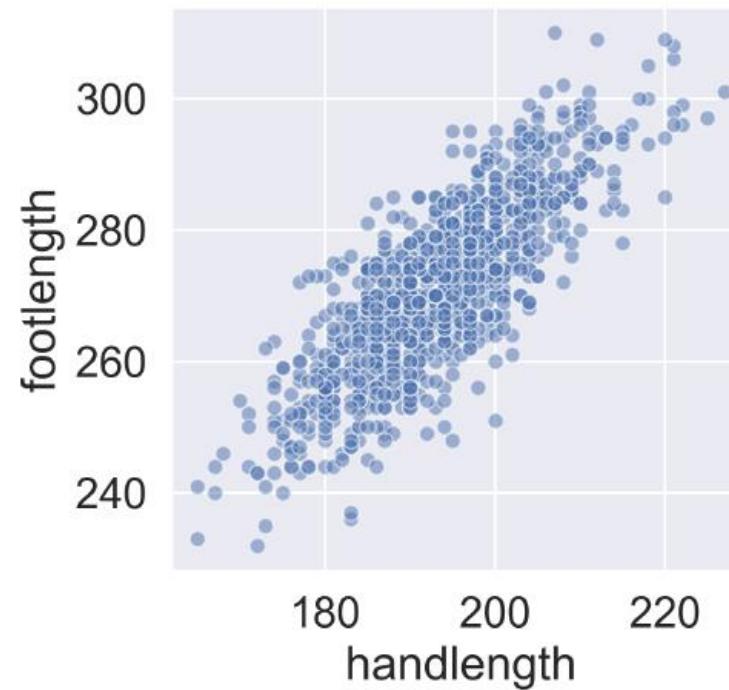
$$X V = U \Sigma \boxed{V^* V}$$

$$T = U \Sigma$$

Extracción de Características - PCA

Intro to PCA

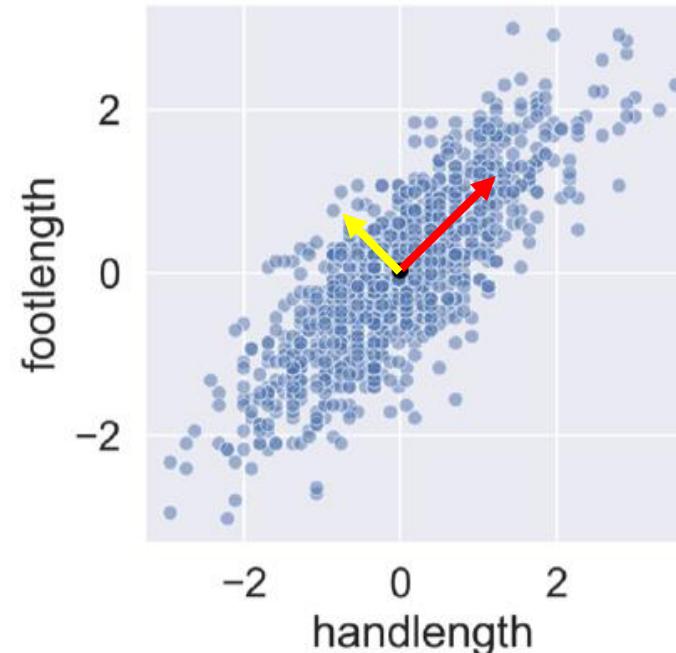
```
sns.scatterplot(data=df, x='handlength', y='footlength')
```



Extracción de Características - PCA

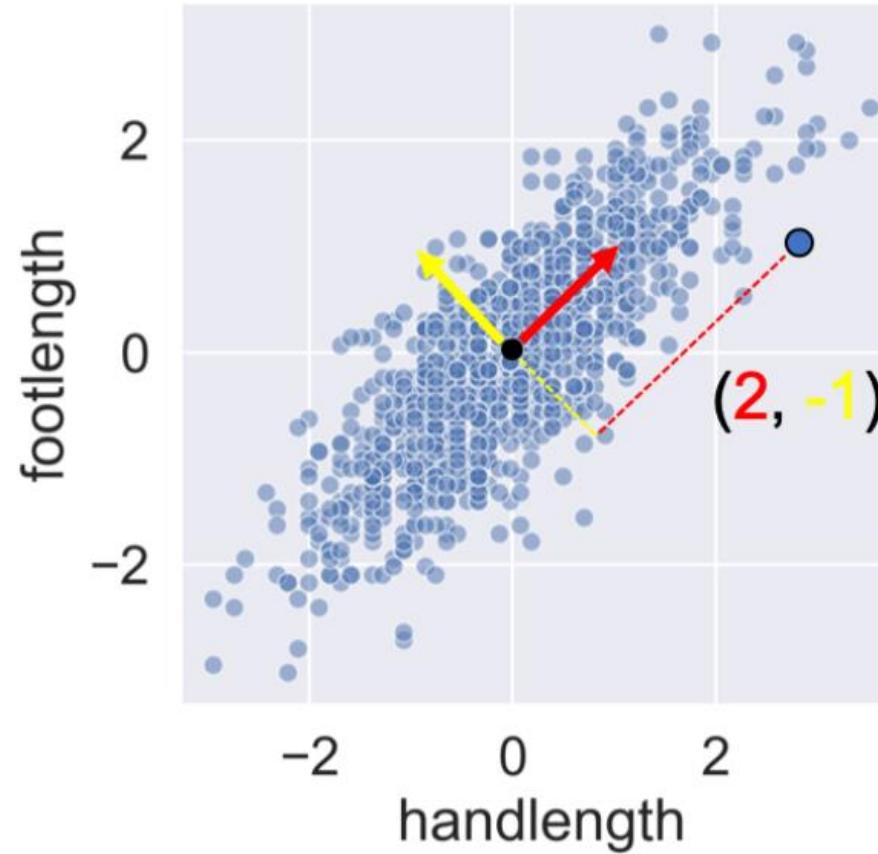
Intro to PCA

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



Extracción de Características - PCA

PCA concept



Extracción de Características - PCA

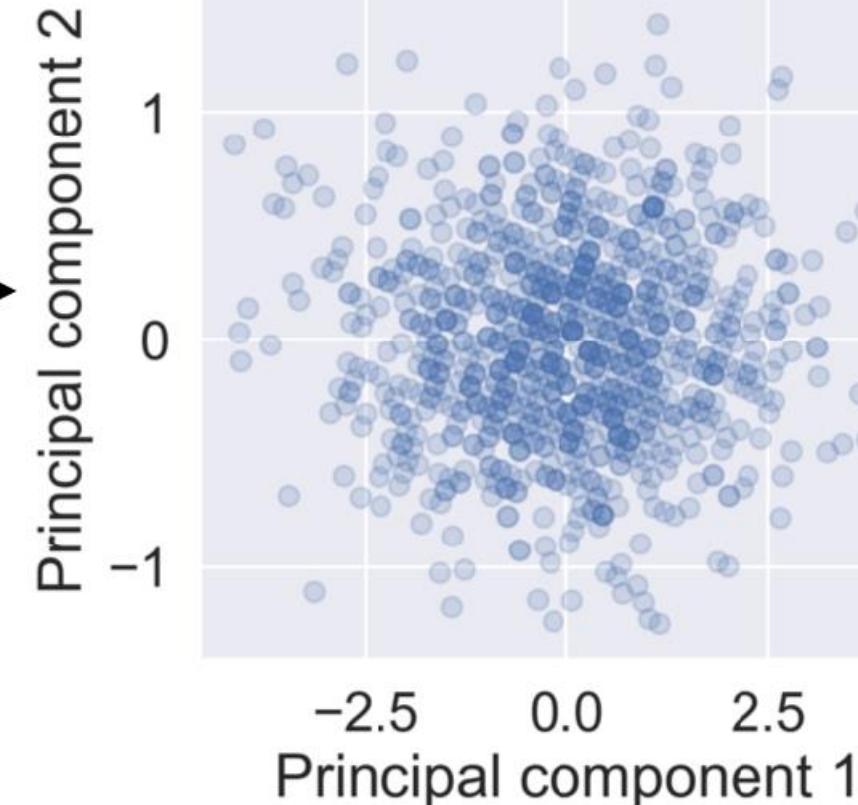
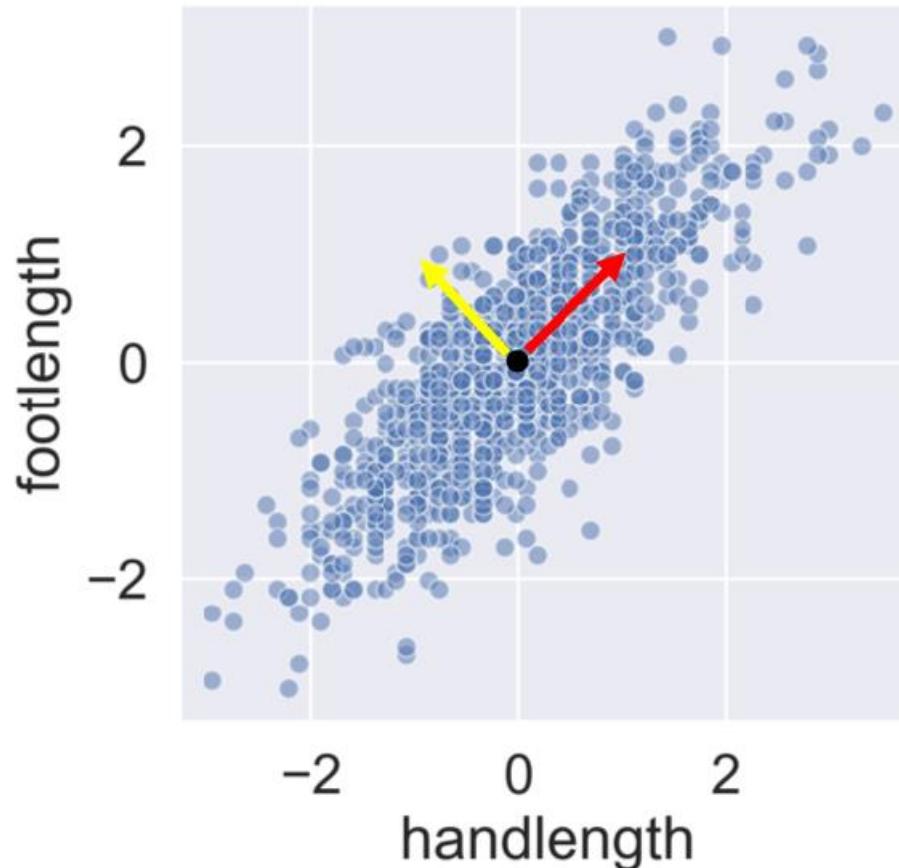
Calculating the principal components

```
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

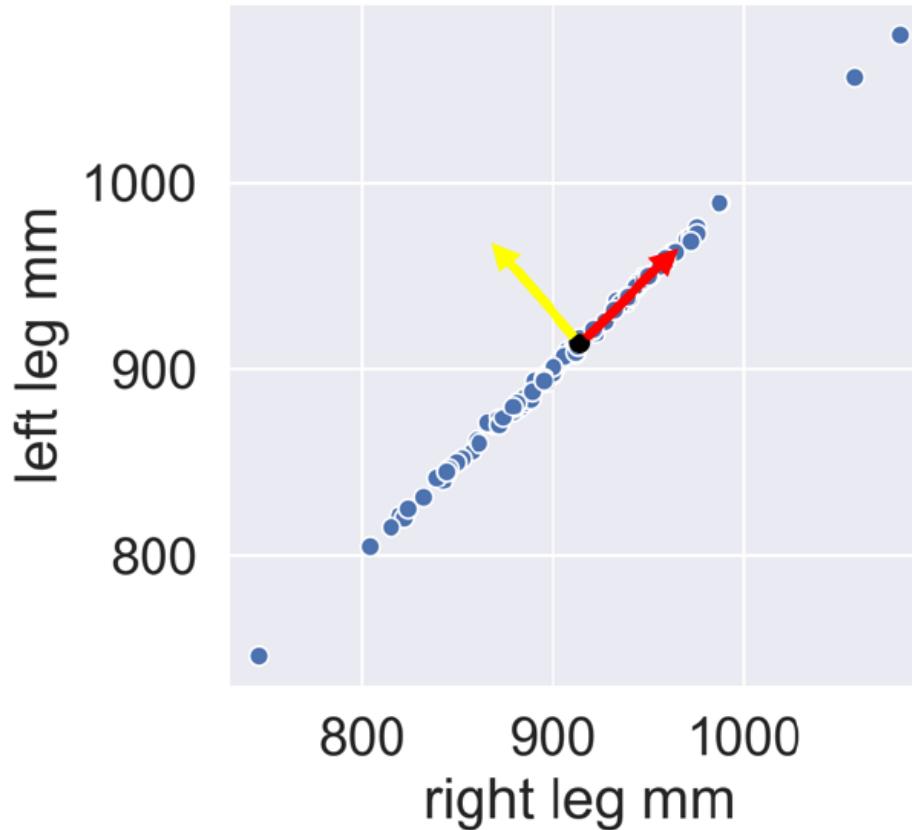
Principal component explained variance ratio

```
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

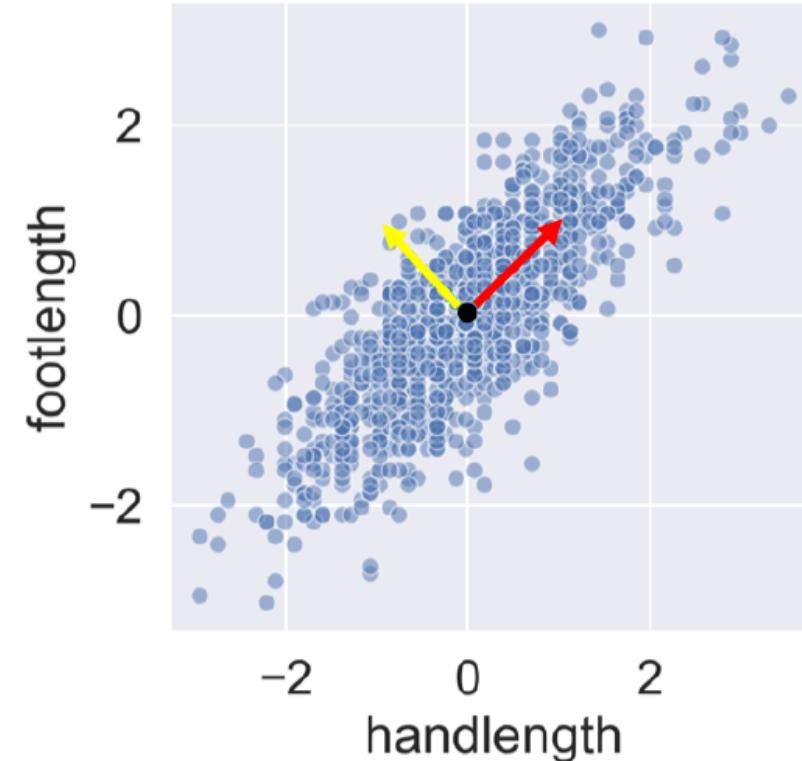
Understanding the components

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

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

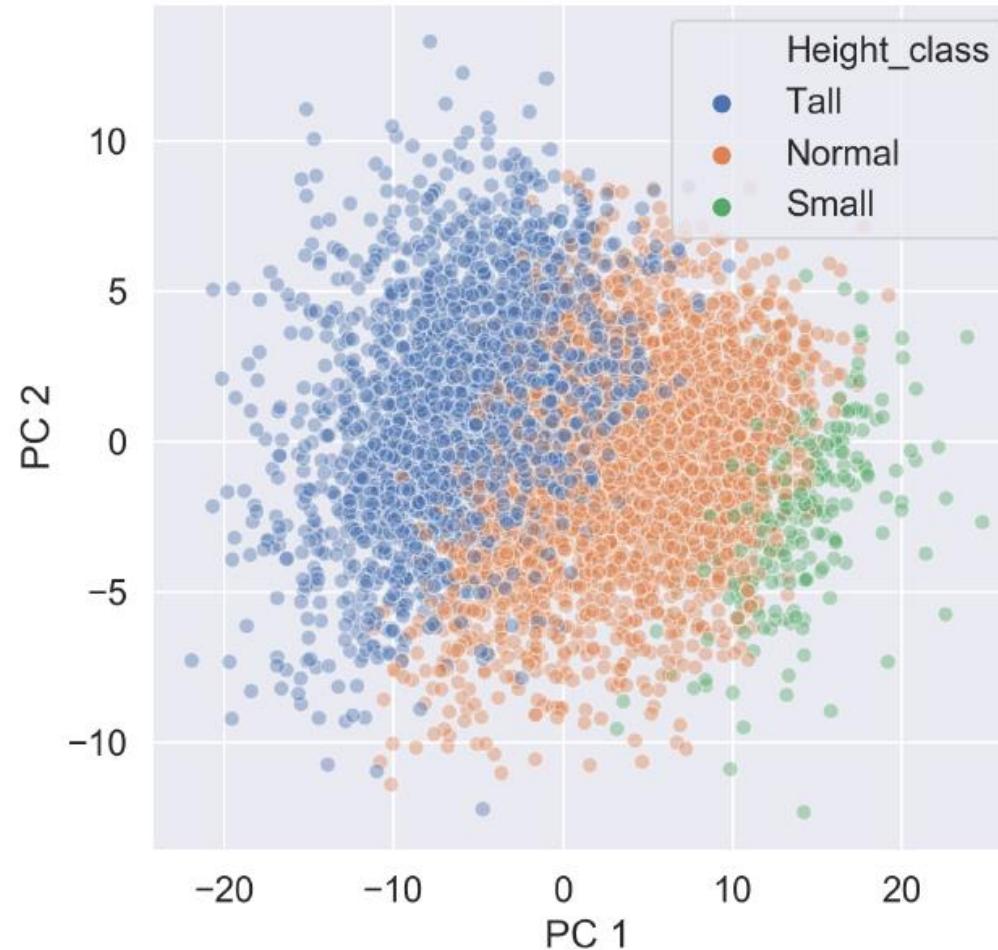
PC 1 = 0.71 x Hand length + 0.71 x Foot length

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

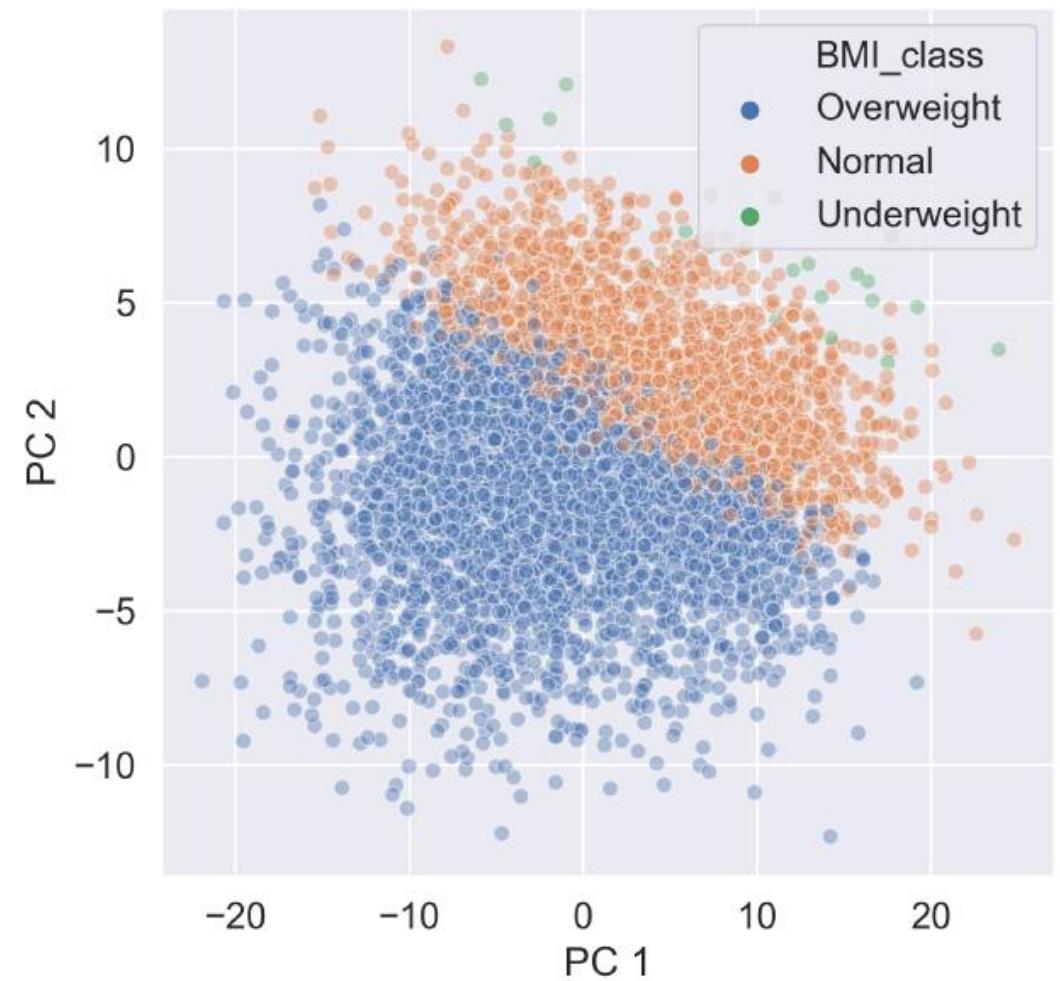
```
print(ansur_categories.head())
```

| 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
```

Selección de PCA

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_

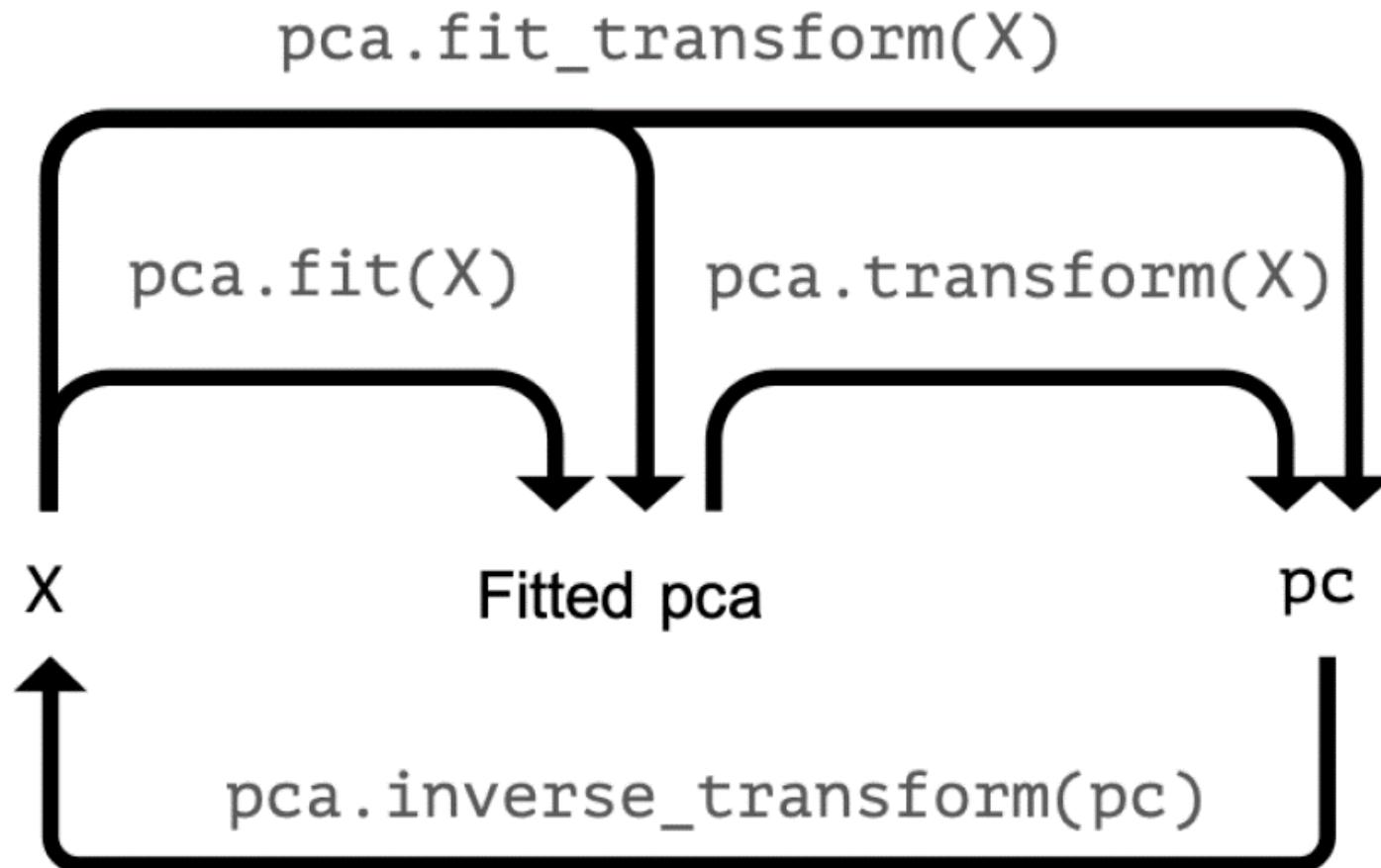
plt.plot(var)

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)
```

Selección de PCA

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 4

<https://github.com/srobles05/CRP-2019S2/>



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.

Feature Extraction

```
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)
```

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](#)