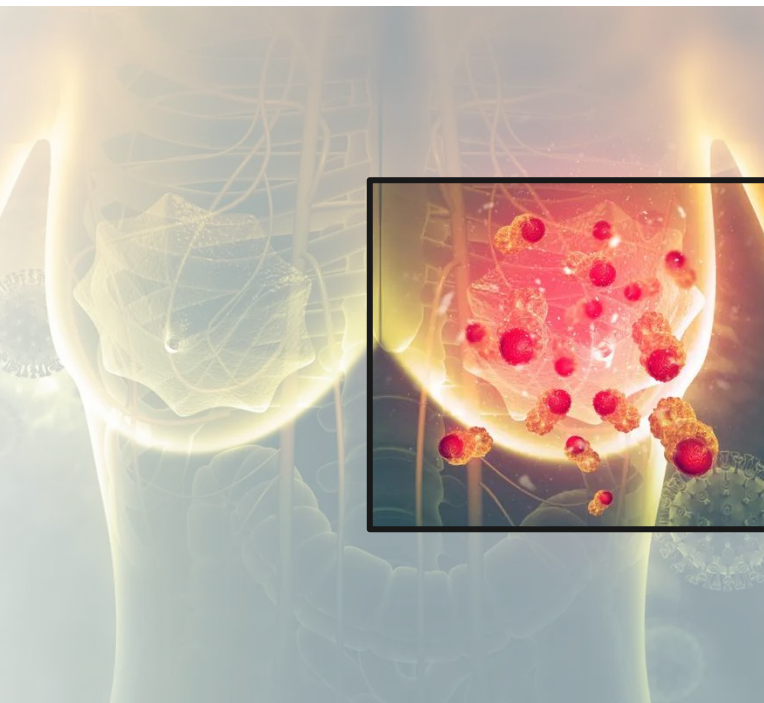


# Exámen Final

## PCA y AdaBoost

---

Grupo 3: Guadalupe Sosa Ferro y Florencia Denisse Costa



# 01

# Dataset

---

# Breast Cancer Coimbra

- Dataset obtenido de UC Irvine Machine Learning Repository.
- 116 muestras
- 10 atributos
- Características clínicas de 64 pacientes con cáncer de mama y 52 controles sanos.
- **Clase:** control sano o paciente.



# Atributos

Variables cuantitativas discretas:

**Edad** (años)

**Glucosa** (mg/dL) → azúcar en sangre

Variables cuantitativas continuas:

- **BMI** (kg/m<sup>2</sup>) → índice de masa corporal
- **Insulina** (μU/mL) → liberada en respuesta a la glucosa para su absorción
- **HOMA** → *Homeostatic model assessment of insulin resistance*
- **Leptina** (ng/mL) → regula el hambre
- **Adiponectina** (μg/mL) → metabolismo de la glucosa
- **Resistina** (ng/mL) → asociada a la insulinoresistencia
- **MCP.1** (pg/dL) → disminuye la captación de glucosa

$$HOMA = \frac{glucosa \times insulina}{405}$$

**Clasificación** → 1: control sano 2: paciente con cáncer de mama

# Procedimiento

01 - - - - - 02 - - - - - 03 - - - - - 04 - - - - - 05

**Pre-  
procesamiento**

**Análisis  
univariado**

**Reducción de  
dimensionalidad**

- PCA
- Elección propia

**Modelos**

- AdaBoost
- Random Forest

**Evaluación**

# Preprocesamiento

## Duplicados

```
Datos duplicados: False
```

## Datos faltantes

```
Datos faltantes:  
Age           False  
BMI           False  
Glucose       False  
Insulin       False  
HOMA          False  
Leptin        False  
Adiponectin   False  
Resistin      False  
MCP.1         False  
Classification False
```

## Reemplazo de valores

1 → 0 (control sano)  
2 → 1 (paciente enfermo)

# Procedimiento

01 ————— 02 - - - - - 03 - - - - - 04 - - - - - 05

**Pre-  
procesamiento**

**Análisis  
univariado**

**Reducción de  
dimensionalidad**

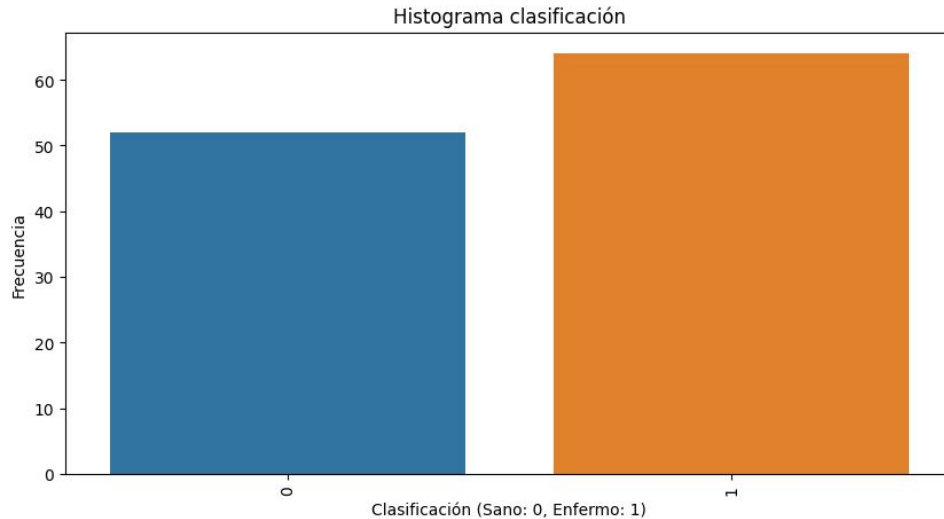
- PCA
- Elección propia

**Modelos**

- AdaBoost
- Random Forest

**Evaluación**

# Análisis univariado

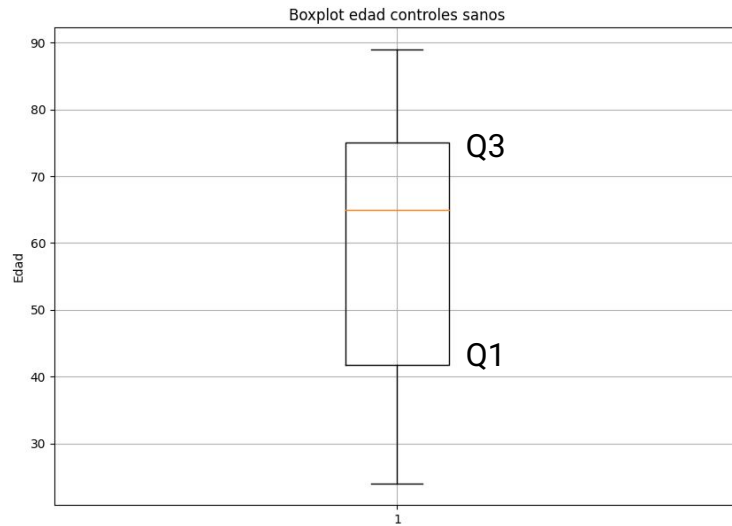


Promedio edad sanos: 58.08 años

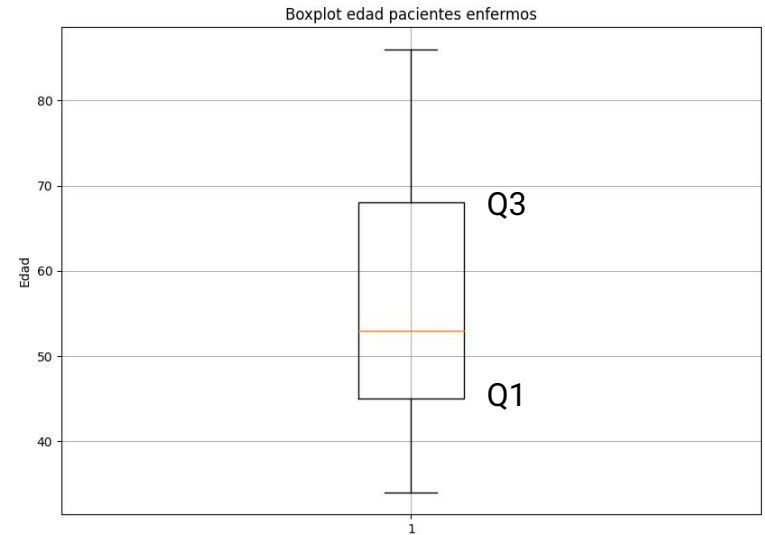
Promedio edad enfermos: 56.67 años



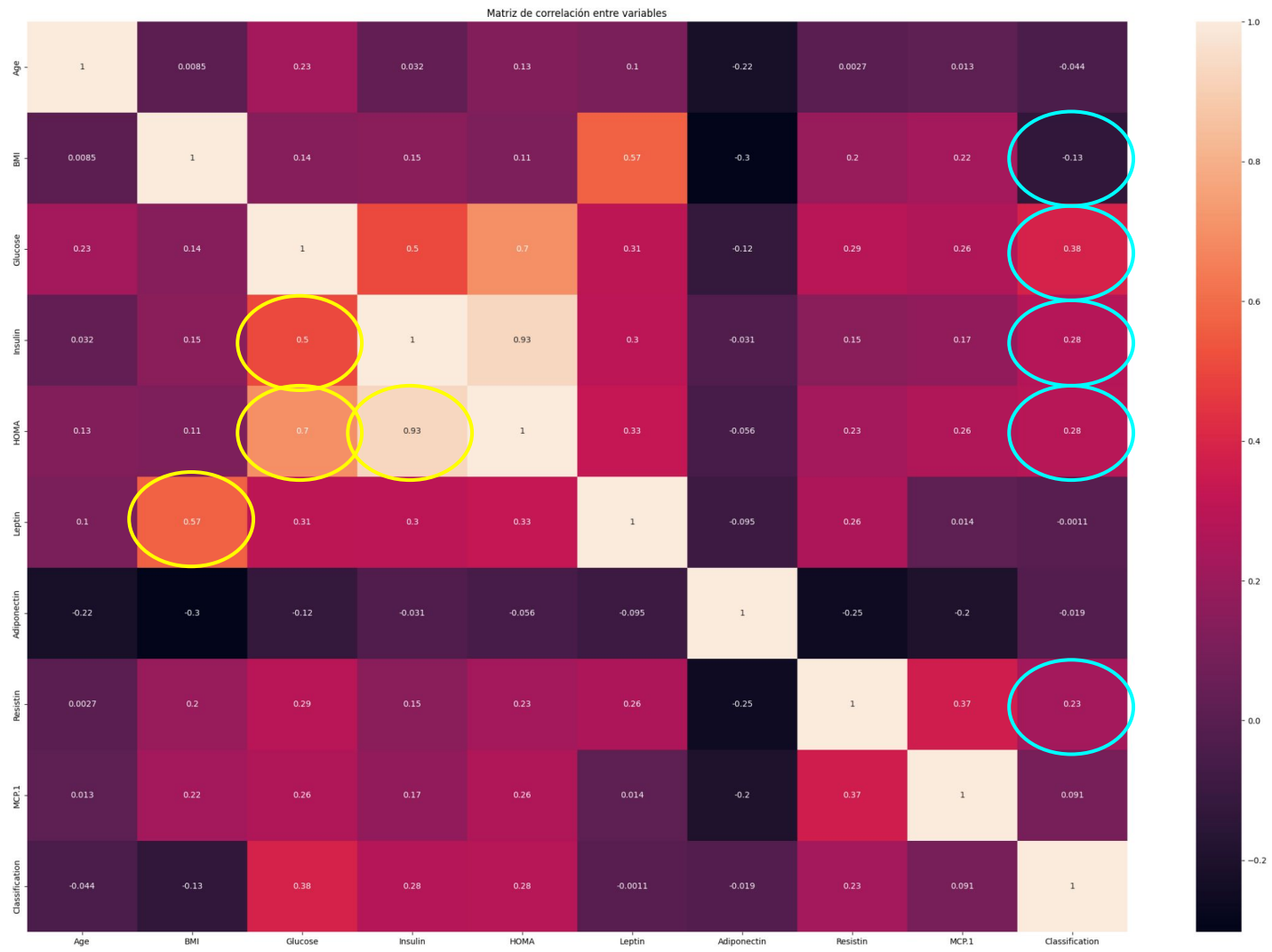
# Análisis univariado

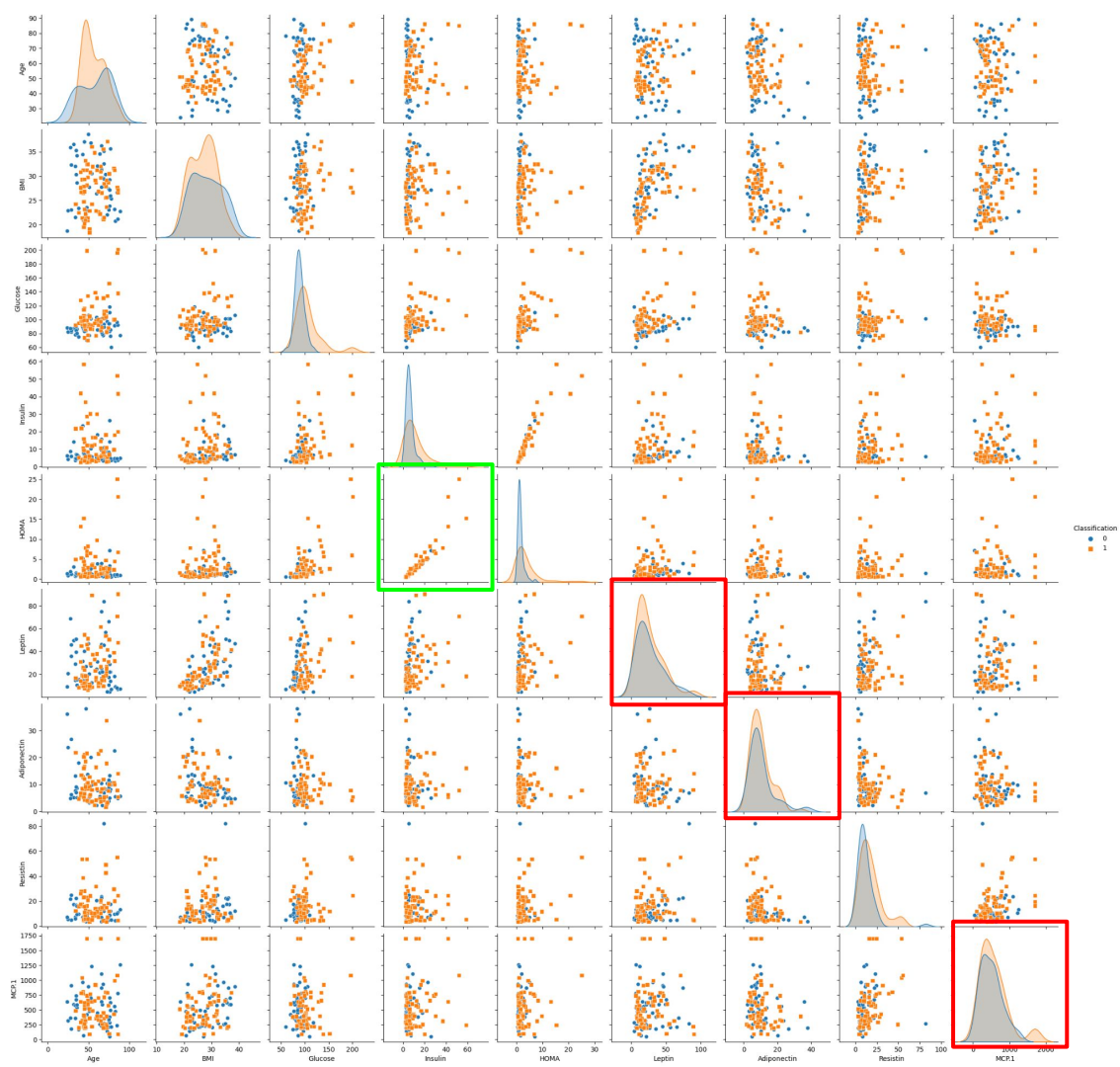


Asimetría negativa



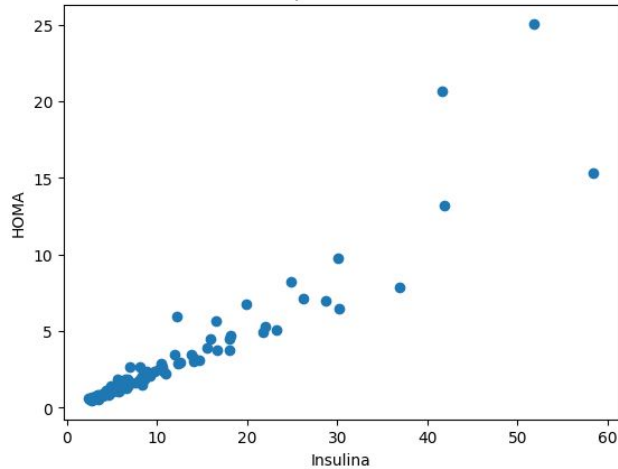
Asimetría positiva





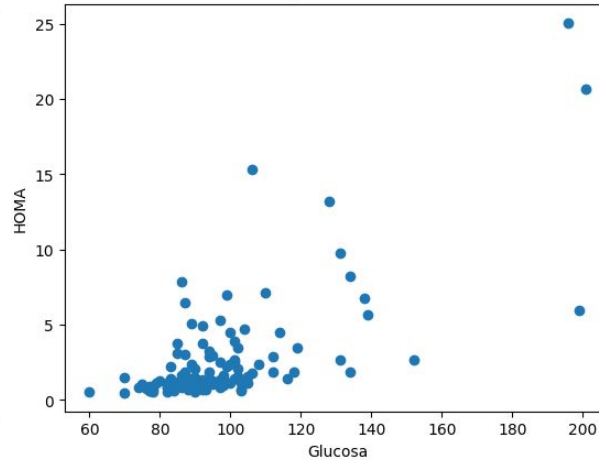
# Análisis univariado

Gráfico de dispersión Insulina - HOMA



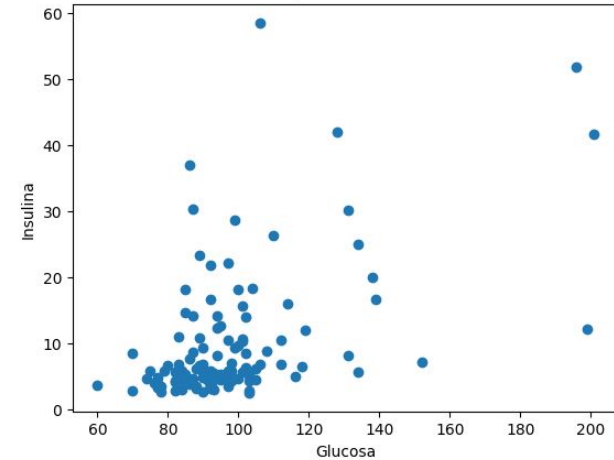
$r = 0,93$

Gráfico de dispersión Glucosa - HOMA



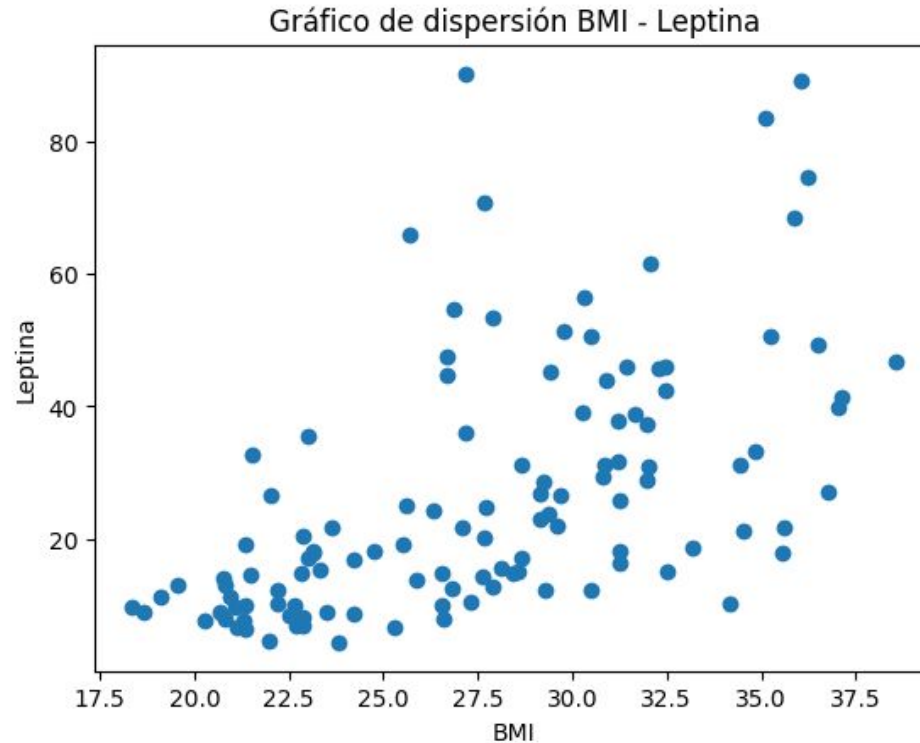
$r = 0,7$

Gráfico de dispersión Insulina - Glucosa



$r = 0,5$

# Análisis univariado



$r = 0,57$

# Procedimiento

01 ————— 02 ————— 03 - - - - - 04 - - - - - 05

**Pre-  
procesamiento**

**Análisis  
univariado**

**Reducción de  
dimensionalidad**

- PCA
- Elección propia

**Modelos**

- AdaBoost
- Random Forest

**Evaluación**

# Reducción de dimensionalidad

## **PRINCIPAL COMPONENT ANALYSIS (PCA)**

1. División del dataset
2. Estandarización
3. Aplicación de PCA

## **SELECCIÓN DE VARIABLES**

1. Selección de variables
2. División del dataset
3. Estandarización

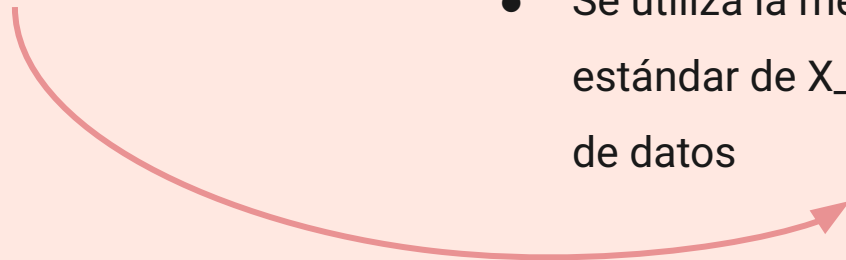
# PCA

## División del dataset

- Método `train_test_split` de Sklearn.
- 80% entrenamiento, 20% prueba.
- Random state = 0

## Estandarización

- Función `StandardScaler()` de Sklearn.
- Transformación solamente de `X_train` y `X_test`.
- Se utiliza la media y desvío estándar de `X_train` → evitar fuga de datos





# PCA - Teoría

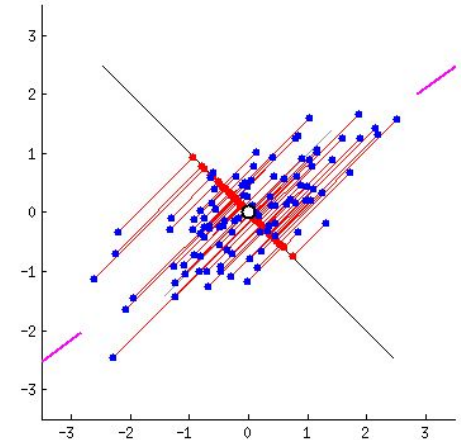
- Inventado en 1901 por Karl Pearson.
- **Idea principal:** poder proyectar datos de alta dimensionalidad en un espacio de menor dimensión, preservando la mayor cantidad de información posible.
- Transforma las variables originales en un nuevo conjunto de variables llamadas **componentes principales**, que son **combinaciones lineales** de las **variables originales**.

Ejemplo       $\text{Var}_1, \text{Var}_2, \text{Var}_3, \dots \text{Var}_n$

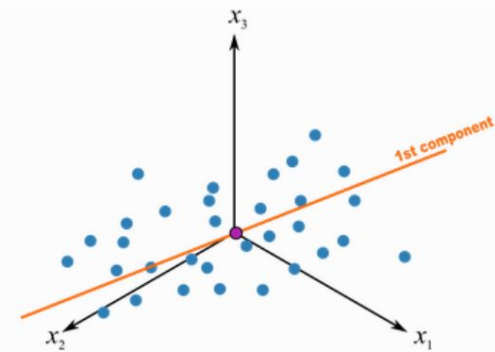
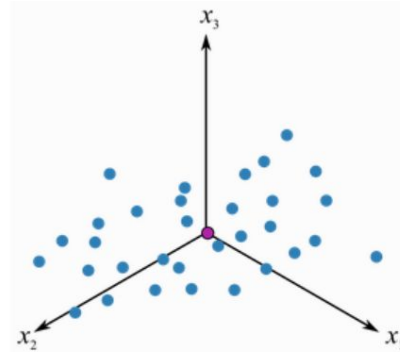
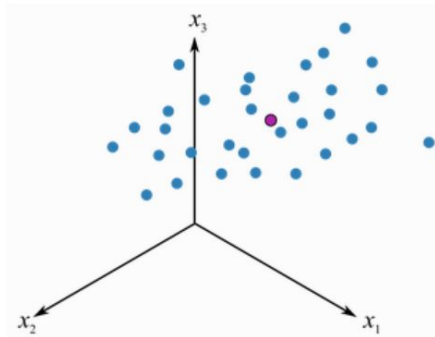
$$\text{PC}_1 = a_1 * \text{Var}_1 + a_2 * \text{Var}_2 + a_3 * \text{Var}_3 + \dots + a_n * \text{Var}_n$$

$$\text{PC}_2 = b_1 * \text{Var}_1 + b_2 * \text{Var}_2 + b_3 * \text{Var}_3 + \dots + b_n * \text{Var}_n$$

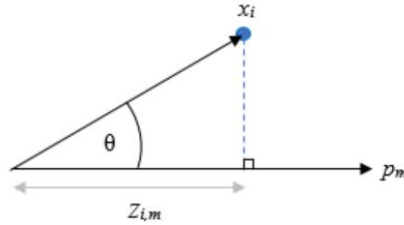
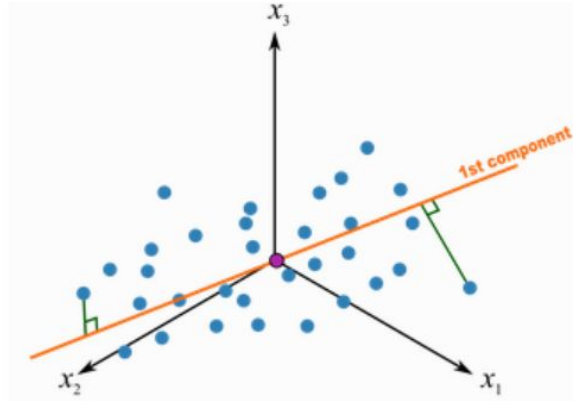
$$\text{PC}_n = c_1 * \text{Var}_1 + c_2 * \text{Var}_2 + c_3 * \text{Var}_3 + \dots + c_n * \text{Var}_n$$



# PCA - Explicación geométrica



# PCA - Explicación geométrica



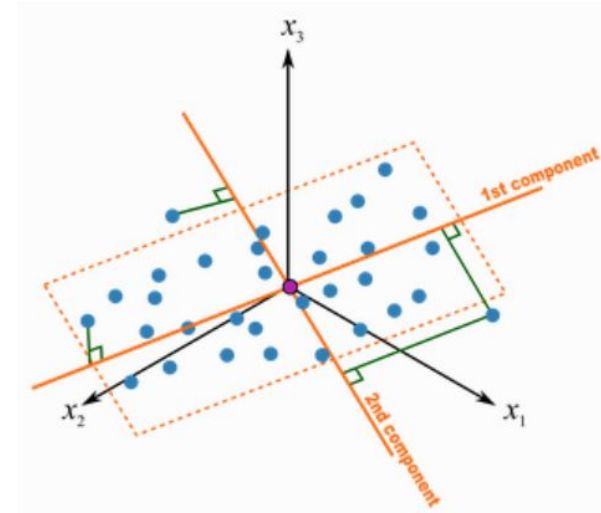
$$\cos(\theta) = \frac{z_{i,m}}{\|x_i\|} = \frac{x'_i p_m}{\|x_i\| \|p_m\|}$$

$$z_{im} = x'_i p_m$$

$z_i$  = score

$x_i$  = observación

$p_m$  = peso (loadings)



# PCA - Algoritmo

1. **Estandarización de datos**  $z = \frac{value - mean}{standard\ deviation}$

2. **Cálculo de la matriz de covarianza**  $\Sigma = \begin{bmatrix} \text{Var}(X_1) & \text{Cov}(X_1, X_2) & \cdots & \text{Cov}(X_1, X_n) \\ \text{Cov}(X_2, X_1) & \text{Var}(X_2) & \cdots & \text{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(X_n, X_1) & \text{Cov}(X_n, X_2) & \cdots & \text{Var}(X_n) \end{bmatrix}$   $Cov(X, Y) = \frac{\sum_1^n (x_i - \bar{x})(y_i - \bar{y})}{n}$

3. **Descomposición SVD de matriz de covarianza**

- Autovectores:** direcciones de variables originales que definen PC (*loadings*)
- Autovalores:** varianza explicada en cada PC

$$A = U D V^T$$

Left singular vectors      Singular values      Right singular vectors

4. **Selección de componentes principales.**

$$\frac{SS(\text{distances for PC1})}{n - 1} = \text{Eigenvalue for PC1}$$
$$\sqrt{SS(\text{distances for PC1})} = \text{Singular Value for PC1}$$

Autovectores de  $A = V * \text{Autovectores de } S * V^T$

5. **Proyección de datos.** Transformamos los resultados a un nuevo espacio dimensional (n, d) donde “n” son el número de componentes y “d” el número de datos.

# PCA - Sklearn

`sklearn.decomposition.PCA`

- Centra los datos pero no los escala → *StandardScaler()*

Hiperparámetros:

- **n\_components:** probamos con 2 y 7 (0.95)
- **svd\_solver:** 'auto'
- Resto *default*

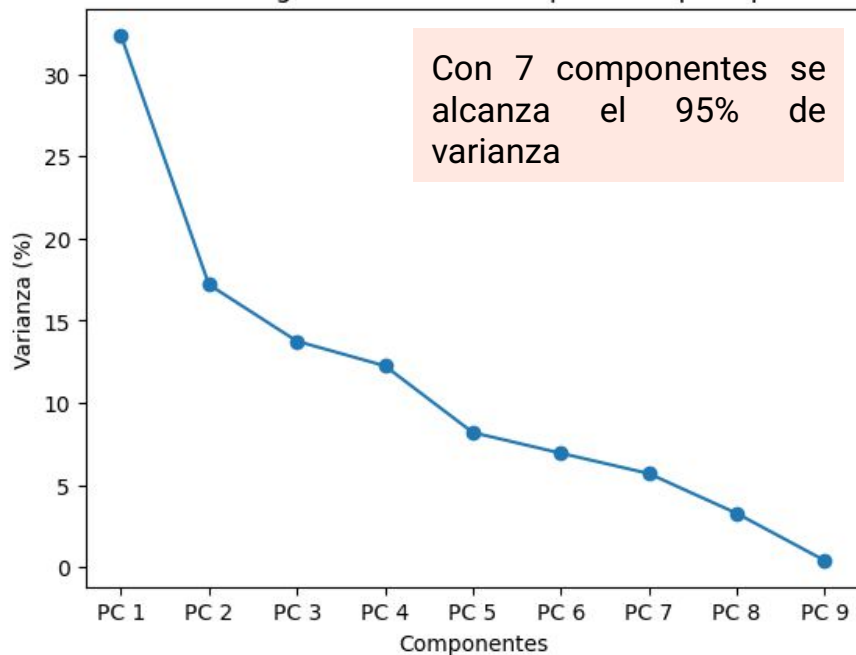
# PCA - Resultados



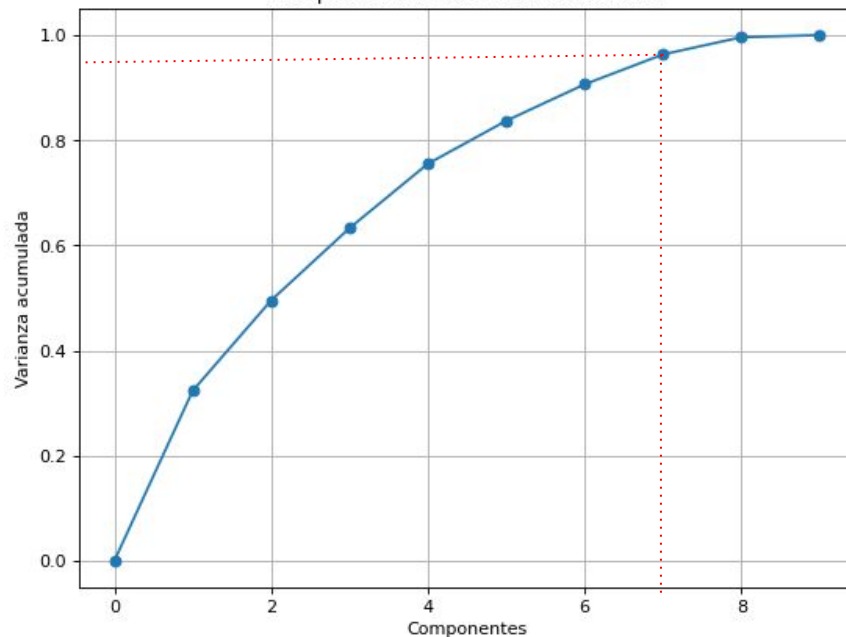
49.5% de la información

# PCA - Resultados

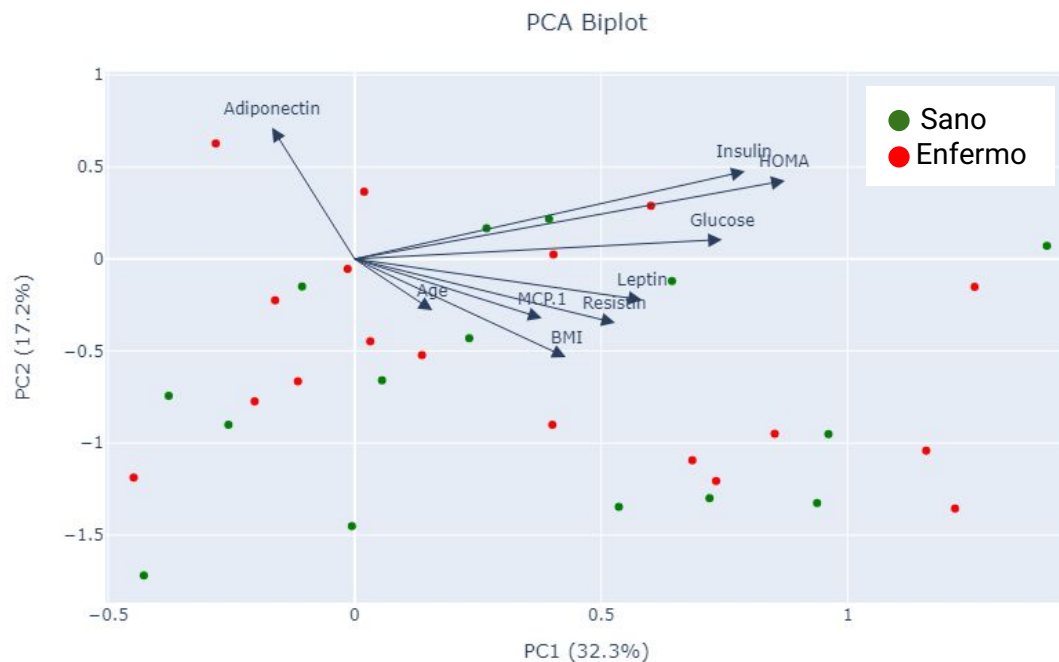
Varianza según cantidad de componentes principales



Componentes vs. Varianza acumulada



# PCA - Resultados



## Variables que más contribuyen

**PC1:** HOMA, Insulina, Glucosa, Leptina, Resistina, BM1, MCP.1, Adiponectina, Age.

**PC2:** Adiponectin, BMI, Insulin, HOMA, Resistin, MCP.1, Age, Leptin, Glucose

*pca.components\_*



# Selección de variables

Variables con  $|r|$  más grande con respecto a la Clase:

- Glucosa  $\rightarrow r = 0,38$
- HOMA  $\rightarrow r = 0,28$
- Insulina  $\rightarrow r = 0,28$
- Resistina  $\rightarrow r = 0,23$
- BMI  $\rightarrow r = -0,13$

HOMA vs Insulina  $\rightarrow r = 0,93$

Glucosa vs HOMA  $\rightarrow r = 0,7$

Glucosa vs Insulina  $\rightarrow r = 0,5$

- ❖ HOMA depende directamente de Glucosa e Insulina, es redundante.

Selección final: glucosa, insulina, resistina y BMI.

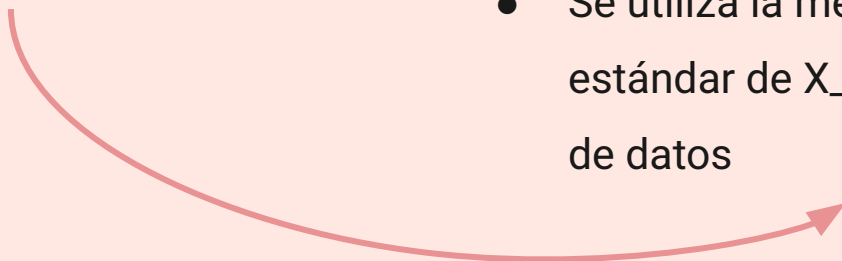
# Selección de variables (cont.)

## División del dataset

- Método `train_test_split` de Sklearn.
- 80% entrenamiento, 20% prueba.
- Random state = 0

## Estandarización

- Función `StandardScaler()` de Sklearn.
- Transformación solamente de `X_train` y `X_test`.
- Se utiliza la media y desvío estándar de `X_train` → evitar fuga de datos



# Procedimiento

01 ——— 02 ——— 03 ——— 04 - - - - 05

**Pre-  
procesamiento**

**Análisis  
univariado**

**Reducción de  
dimensionalidad**

- PCA
- Elección propia

**Modelos**

- AdaBoost
- Random Forest

**Evaluación**

# AdaBoost

- Desarrollado por Yoav Freund y Robert Schapire en 1995.
  - Aprendizaje supervisado, método de ensamble.
  - *Adaptive boosting*
  - Clasificación binaria (también multiclase) y regresión.
  - Sensible al ruido y outliers.
- 
- ★ Clasificadores débiles → **múltiples árboles de un nodo y dos hojas**
  - ★ Distinta influencia en la clasificación final → **Cada árbol tiene asociado un peso**
  - ★ **Los errores de un árbol influyen en la generación del siguiente**

# AdaBoost vs Random Forest

## SIMILITUDES

- Ambos son métodos de ensamble
- Ambos utilizan árboles de decisión

## DIFERENCIAS

- Tamaño de los árboles
- Peso de cada árbol en la clasificación final
- Orden de construcción de los árboles
- Tipo de muestreo (bagging vs boosting)
- Adaboost es más sensible al ruido
- Adaboost generalmente es más preciso

# AdaBoost - Algoritmo


1. Se le asigna un **peso** a las  $n$  instancias de entrenamiento  $\rightarrow w_i = \frac{1}{n}$
2. Se selecciona la **variable** que mejor clasifique a las instancias y se genera el **árbol** (1 nodo y 2 hojas).  $h_t: X \rightarrow \{-1, 1\}$
3. Se calcula el **error** asociado al árbol  $\rightarrow$  suma de los pesos de las instancias clasificadas incorrectamente.
4. Se calcula el **peso del árbol**  $\rightarrow \alpha_t = \frac{1}{2} \log\left(\frac{1 - \text{error}}{\text{error}}\right)$
5. Se calculan los **nuevos pesos** de las instancias y se normalizan
  - a. Muestra clasificada incorrectamente  $\rightarrow w_{i+1} = w_i \times e^{\alpha_t}$
  - b. Muestra clasificada correctamente  $\rightarrow w_{i+1} = w_i \times e^{-\alpha_t}$
6. Repetir los pasos 2 a 5 para generar **más árboles**.
7. **Clasificación final:** combinación ponderada de los clasificadores  $H_{\text{final}} = \text{sign}\left(\sum_t \alpha_t h_t(x)\right)$

# AdaBoost - Sklearn

`sklearn.ensemble.AdaBoostClassifier`

Hiperparámetros:

- **estimator**: DecisionTreeClassifier con max\_depth = 1
- **n\_estimators**: entre 50 y 500, paso de 50
- **learning\_rate**: entre 0,01 y 1, paso de 0,11
- **random\_state**: None
- **algorithm**: SAMME.R



**GridSearchCV**  
validación cruzada de  
5 pliegues para  
encontrar los mejores  
hiperparámetros

# AdaBoost - Resultados

PCA

Mejores hiperparámetros:

**n\_estimators** = 400

**learning\_rate** = 0,01

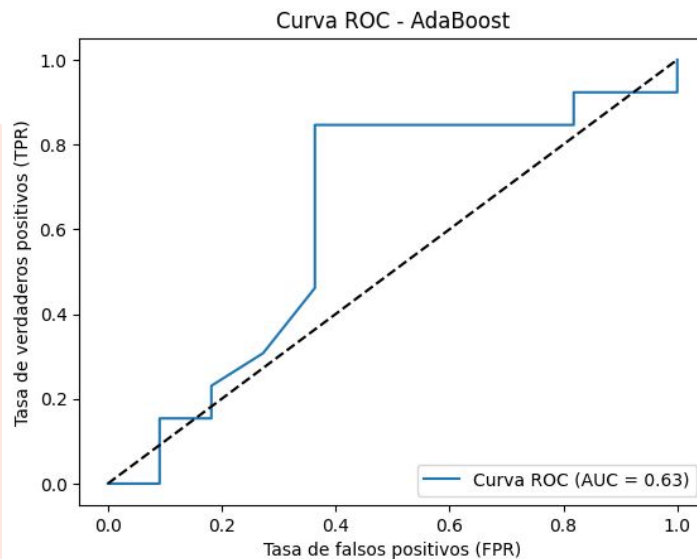
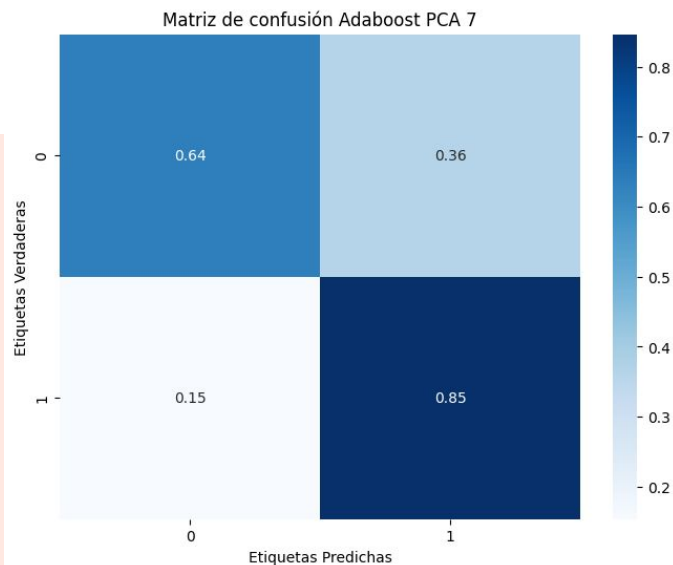
Métricas:

**Accuracy** = 0,75

**Recall** = 0,75

**Precisión** = 0,75

**f1-score** = 0,75





# AdaBoost - Resultados

## Selección de variables

Mejores hiperparámetros:

**n\_estimators** = 100

**learning\_rate** = 0,78

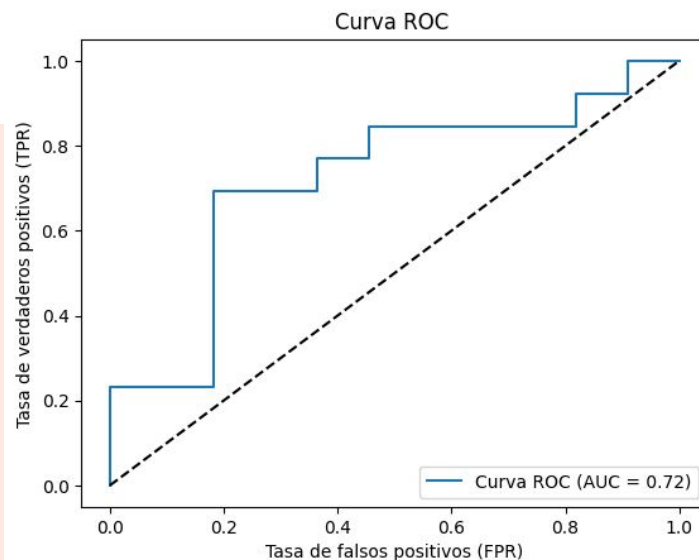
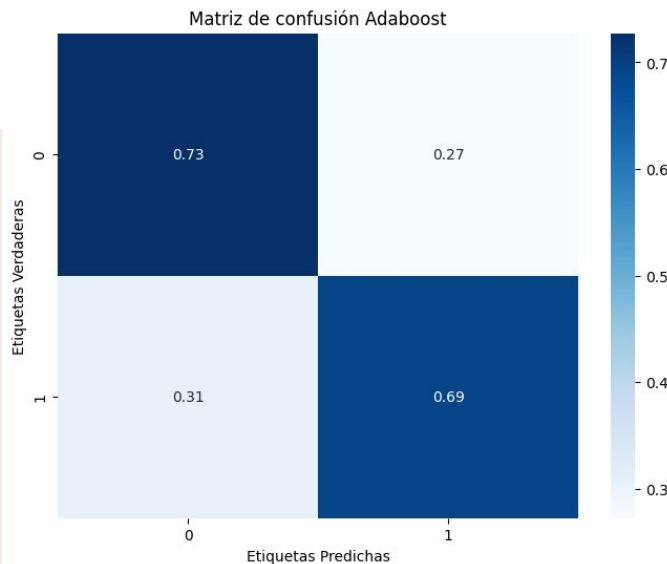
Métricas:

**Accuracy** = 0,71

**Recall** = 0,71

**Precisión** = 0,71

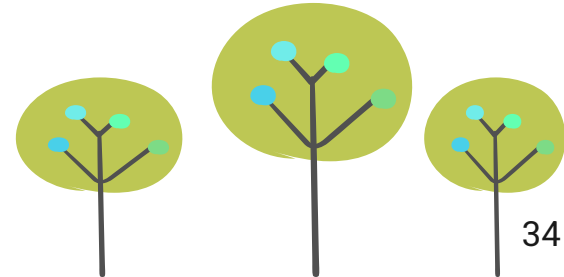
**f1-score** = 0,71



# Random Forest - Algoritmo

Recordamos los pasos:

1. Se **divide aleatoriamente** el dataset en  $k$  subconjuntos de  $N$  muestras con **reemplazo**.
2. Se crea un **árbol** con cada subconjunto.
3. Cada árbol realiza una **clasificación**.
4. La **clasificación final** es la opción más votada por todos los árboles.




# Random Forest - Sklearn

`sklearn.ensemble.RandomForestClassifier`

Hiperparámetros:

- **n\_estimators**: entre 3 y 20, paso de 1
- **max\_depth**: entre 2 y 7, paso de 1
- **random\_state**: entre 0 y 10, paso de 1
- Resto *default*



**GridSearchCV**  
validación cruzada de  
5 pliegues para  
encontrar los mejores  
hiperparámetros

# Random Forest - Resultados

PCA

Mejores hiperparámetros:

**n\_estimators** = 9

**max\_depth** = 3      **random\_state** = 2

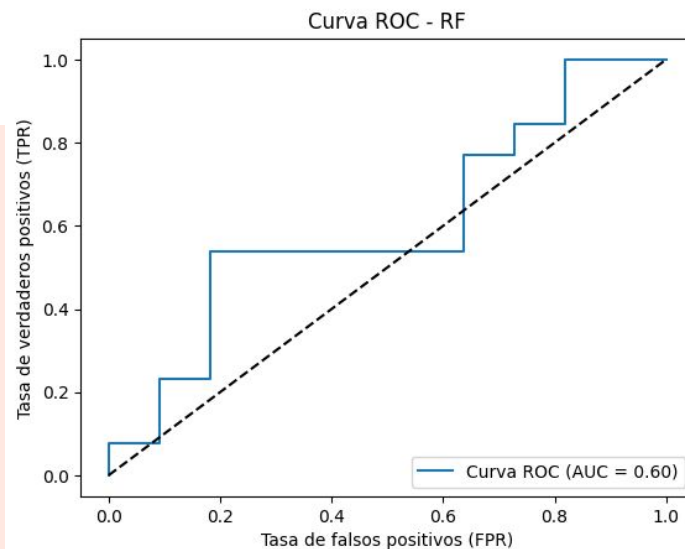
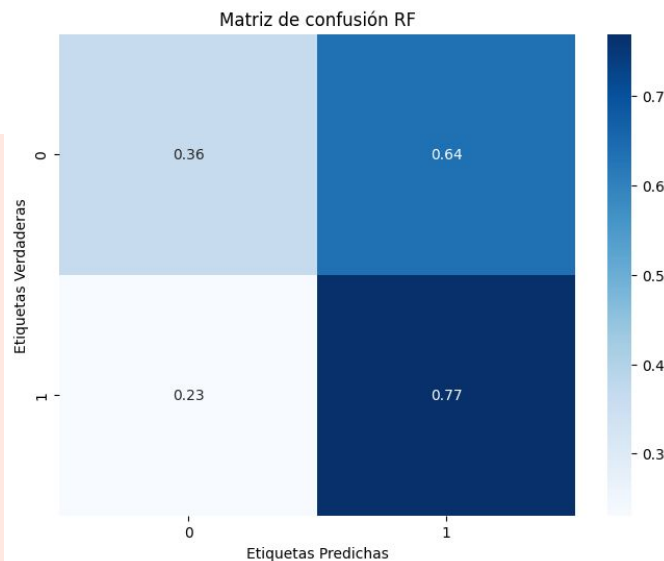
Métricas:

**Accuracy** = 0,58

**Recall** = 0,58

**Precisión** = 0,58

**f1-score** = 0,56



# Random Forest - Resultados selección de variables

Mejores hiperparámetros:

**n\_estimators** = 6

**max\_depth** = 3      **random\_state** = 4

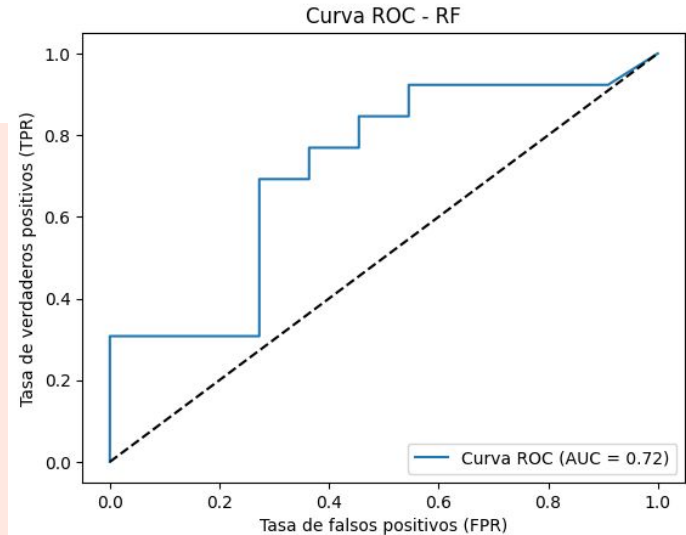
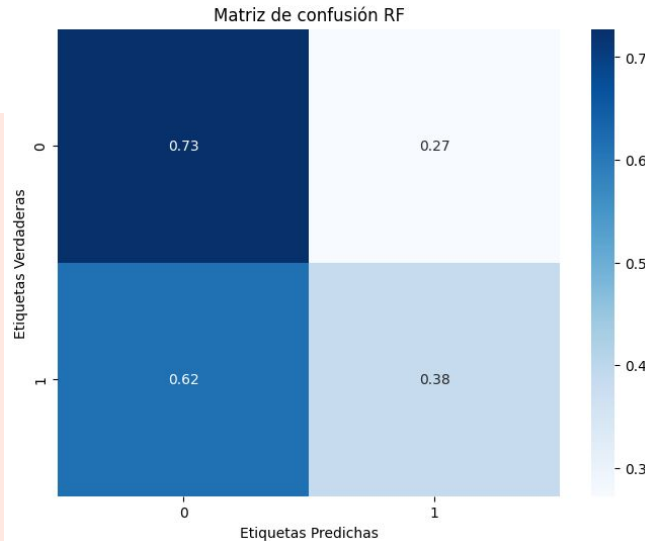
Métricas:

**Accuracy** = 0,54

**Recall** = 0,54

**Precisión** = 0,57

**f1-score** = 0,53



# Conclusiones

- Se generaron 4 modelos de clasificación para el dataset *Breast Cancer Coimbra*
  - 2 de AdaBoost y 2 de Random Forest
    - En cada caso, 1 con PCA y 1 con selección de variables
- Los modelos de AdaBoost son mejores que los de Random Forest
- El uso de PCA para disminuir la dimensionalidad no necesariamente es mejor que la selección de variables
  - Variables y PC sin relación lineal
  - Importancia de entender las variables
  - Pérdida de información
  - Dataset “pequeño”

# Muchas gracias

---

## ¿Preguntas?

Grupo 3: Guadalupe Sosa Ferro y Florencia Denisse Costa

72.75 - Aprendizaje Automático - 2023

**CREDITS:** This presentation template was created by [Slidesgo](#), including icons by [Flaticon](#) and infographics & images by [Freepik](#)

Please keep this slide for attribution