Inteligencia Artificial
1INF24

- Dr. Edwin Villanueva Talavera

UNIDAD 2: Fundamentos de *machine learning* y redes neuronales artificiales

Pre-procesamiento y reducción dimensional

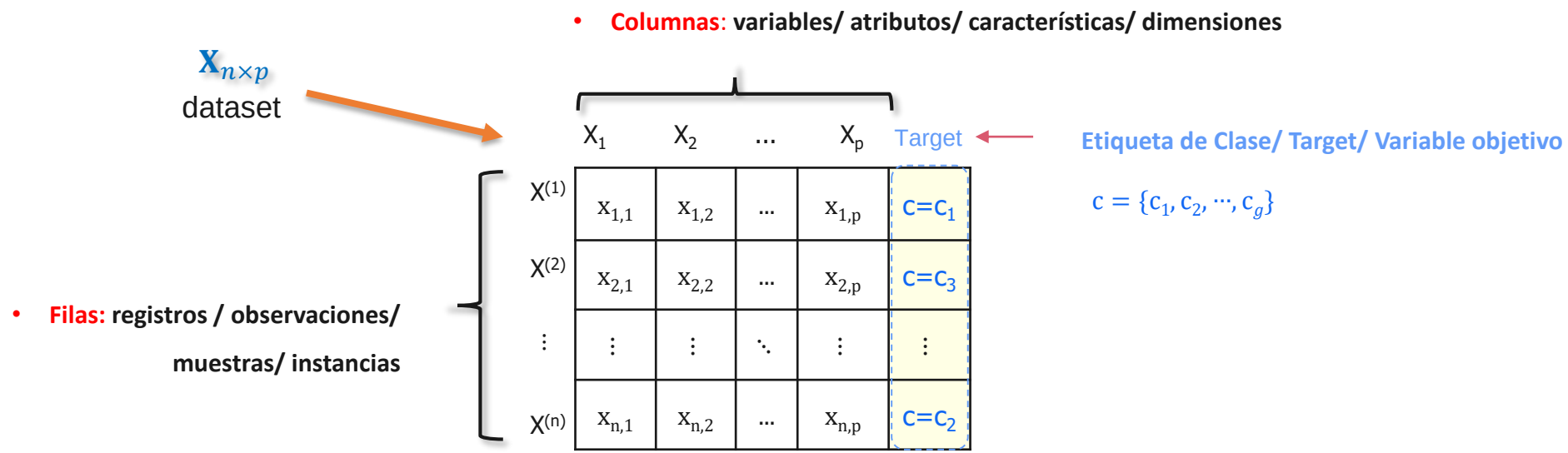
Dr. Edwin Villanueva Talavera

PUCP

Contenido

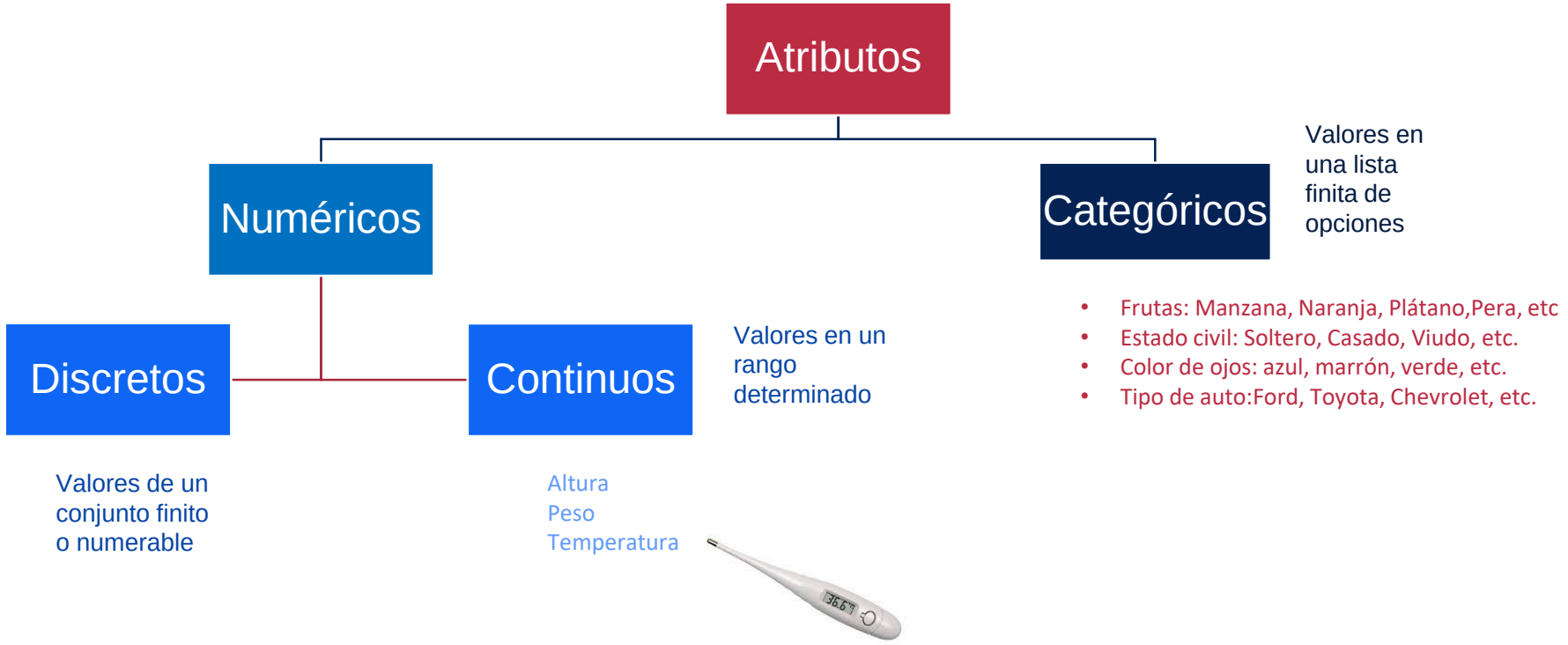
- Exploración inicial
  - Histogramas
  - Densidad
  - Correlación
  - Boxplot
- Limpieza de datos
- Transformación de datos
- Reducción dimensional

Los datos en ML – El dataset



Un conjunto de datos está formado por un grupo finito de registros (entidades o observaciones), cada uno descrito por un conjunto de atributos (o variables)

Tipos de atributos



Fuente: Jessie Kennedy, Edinburgh Napier University

¿Por qué tenemos que preprocesar los datos?

CALIDAD DE DATOS = CALIDAD DE RESULTADOS

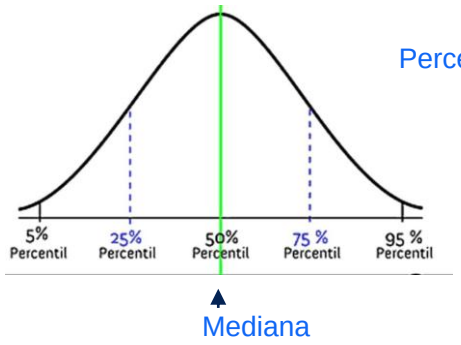
Los datos son: Ruidosos Incompletos Con valores atípicos

Es necesario procesar los datos antes de usarlos para garantizar resultados de calidad

¿Cómo identificamos problemas en la data? → **Análisis Exploratorio de Datos(EDA)**

Exploración inicial de datos

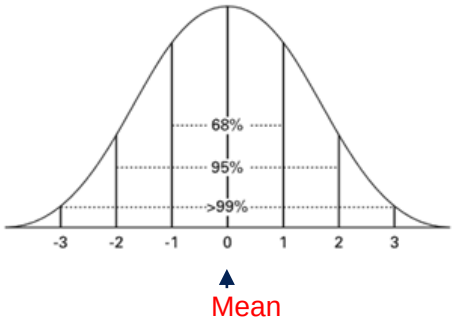
- Estadísticas descriptivas



Percentiles

	temperature	windspeed
count	6.000000	6.000000
mean	30.500000	4.666667
std	3.885872	2.338090
min	24.000000	2.000000
25%	29.000000	2.500000
50%	32.000000	5.000000
75%	32.000000	6.750000
max	35.000000	7.000000

Desviación estándar

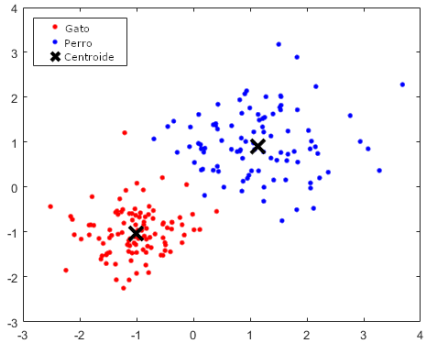


Exploración inicial de datos

- Distribución de clases

```
data.groupby('class').size()
```

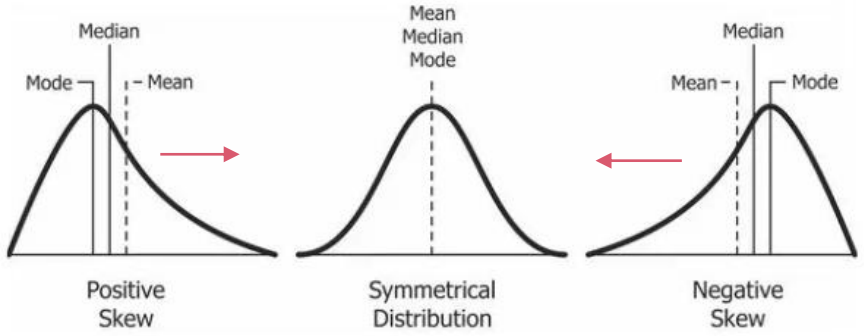
```
class
tested_negative    500
tested_positive    268
dtype: int64
```



- Skewness

```
data.skew()
```

```
preg     0.901674
plas     0.173754
pres    -1.843608
skin     0.109372
insu     2.272251
mass    -0.428982
pedi     1.919911
age      1.129597
```



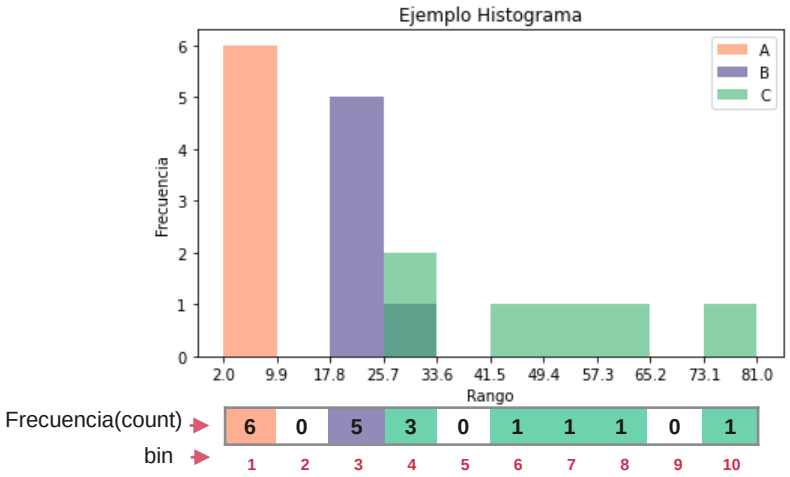
Medidas de tendencia central (media, mediana, moda)

Exploración inicial de datos

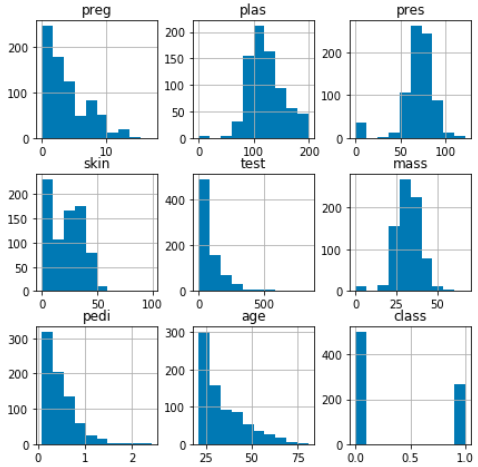
- Histograma

  - Ayuda a comprender la distribución de frecuencias de los atributos.
  - Los "*bins*" indican la frecuencia de valores en un rango.
  - Ayuda a detectar outliers.

A	B	C
6	21	28
7	26	30
5	22	42
2	22	62
2	24	81
2	24	50



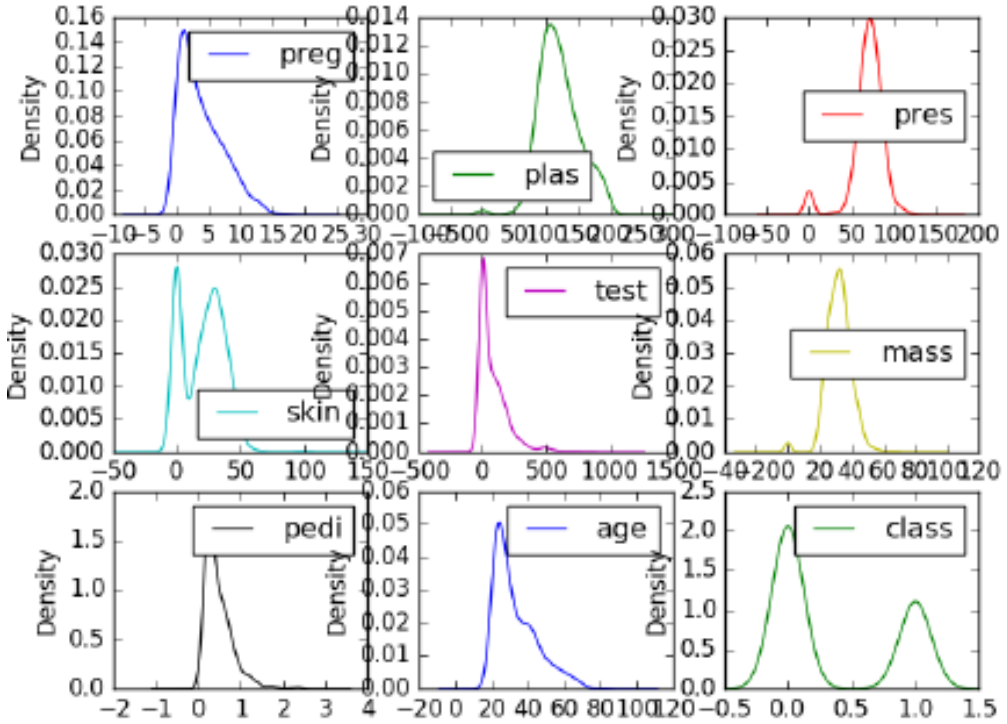

```
data.hist(figsize=[7,7])
plt.show()
```



Exploración inicial de datos

- Diagramas de densidad
  - Muestran la distribución de datos en forma continua.
  - Puede tener diferentes formas: exponencial, gaussiana, multinomial, etc.

```
data.plot(kind='density', subplots=True, layout=(3,3), sharex=False)
pyplot.show()
```



Exploración inicial de datos

- Correlación

  Nos da un indicio de que tan **relacionadas** están dos variables.

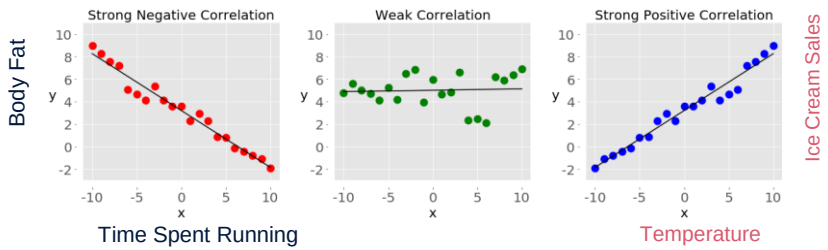
Correlación de Pearson:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

-1
máxima correlación
negativa

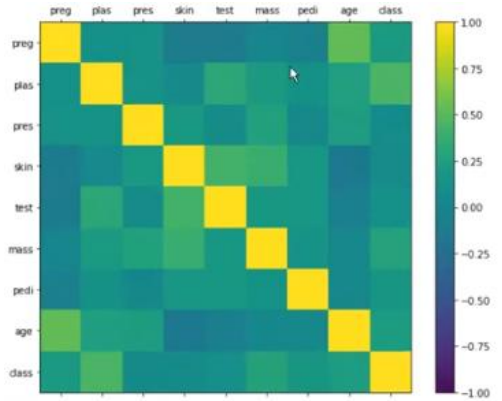
Correlación nula

1
máxima correlación
positiva




```
import matplotlib.pyplot as plt
```

```
data.corr(method='pearson')
```



Correlaciones espurias: https://www.tylervigen.com/spurious-correlations

Exploración inicial de datos

- BoxPlot/Whisker

Muestra los valores de la distribución de cada variable, según la posición de los cuartiles y los valores extremos (outliers)

```
X['edad'].plot(kind = 'box', figsize = (3,4), grid='True')
```

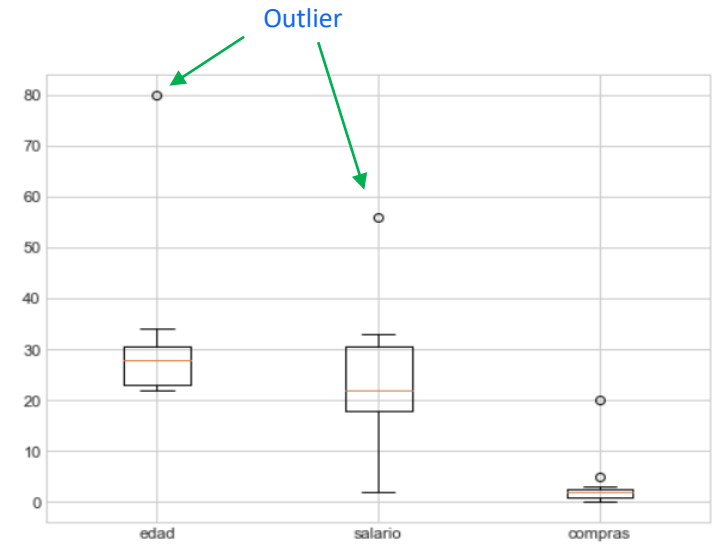
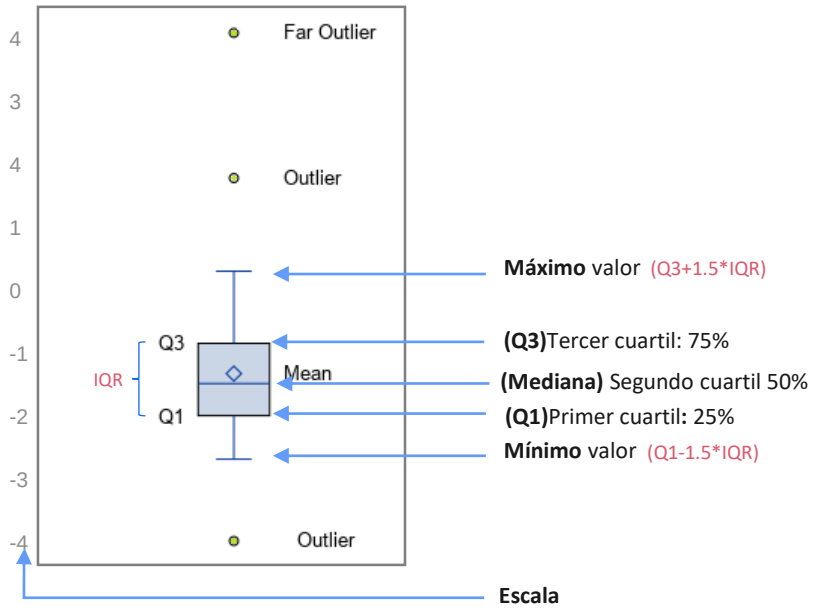


Diagrama esquemático



Limpieza de datos

- Técnicas para el tratamiento de valores faltantes:
- Imputar los valores perdidos, reemplazándolos por estimaciones.
- Eliminar muestras o variables que tienen datos faltantes.

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	NaN	60	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	NaN	50
5	179	84	V	64

Limpieza de datos

- Imputación

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	NaN	60	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	NaN	50
5	179	84	V	64

Sustitución por la media

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	176.6	60	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	NaN	50
5	179	84	V	64

df.fillna(df.mean())

Sustitución por la moda

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	176.6	60	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	V	50
5	179	84	V	64

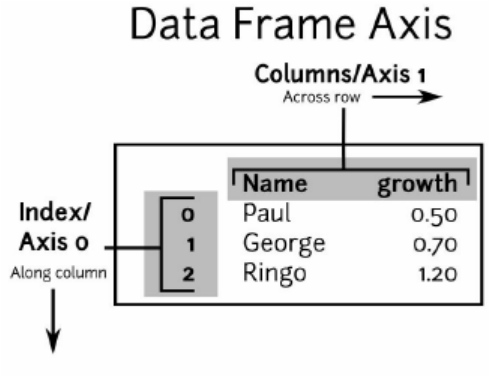
df1.fillna(df1.mode())

Otras opciones:

- Aplicar ML para predecir el valor faltante
- Crear variable nueva para categoría faltante

Limpieza de datos

- Eliminación de filas o columnas con valores perdidos



df.dropna(axis=0)

Eliminar
Filas: 1 y 5

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
2	174	70	V	23
3	162	55	M	21
5	179	84	V	64

Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	NaN	60	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	NaN	50
5	179	84	V	64

Eliminar Columnas:
”Altura”y “Sexo”

df.dropna(axis=1)

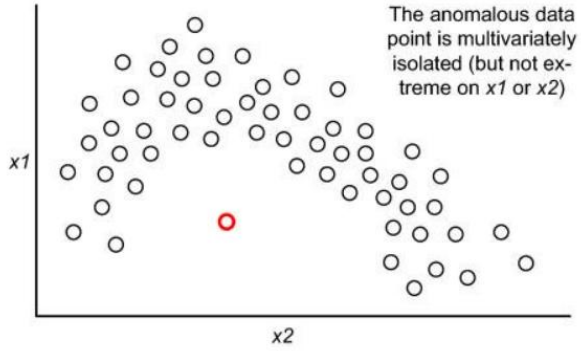
Nº Muestra	Peso (kg)	Edad
0	90	29
1	60	34
2	70	23
3	55	21
4	85	50
5	84	64

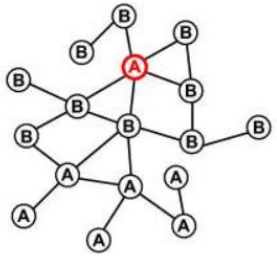
Limpieza de datos

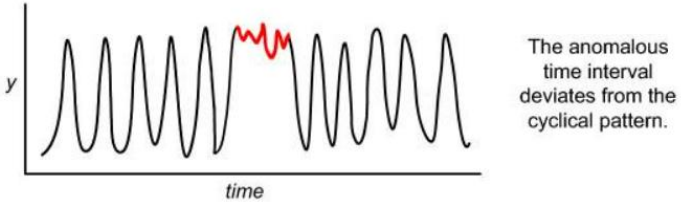
Datos atípicos (*Outliers*)

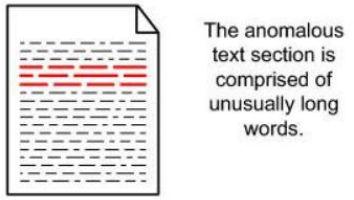
Los *outliers*, son ocurrencias en un conjunto de datos que de alguna manera son inusuales, raros, distintos y no se ajustan a los patrones generales o no encajan. Generalmente, definiremos valores ***outliers*** como valores atípicos, que se “escapan al rango en donde se concentran la mayoría de datos”.

Algunas otras formas de denominar un “outlier” son : valores **Anormales** o **Anómalos**.





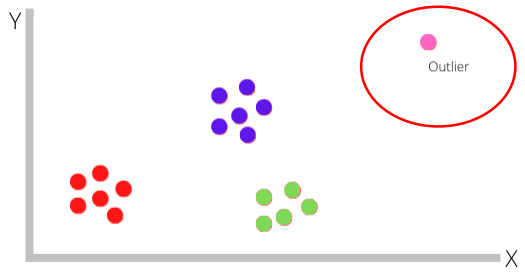




Foorthuis R. (2021). On the nature and types of anomalies: a review of deviations in data. International journal of data science and analytics, 1–35. Advance online publication. https://doi.org/10.1007/s41060-021-00265-1

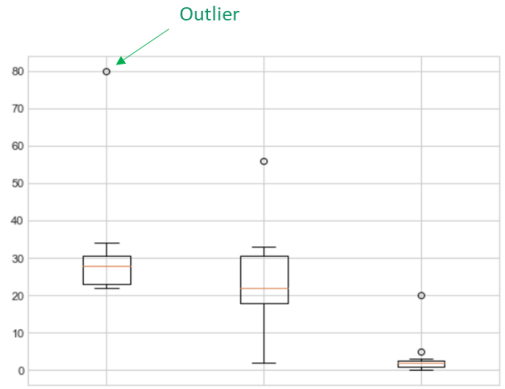
Limpieza de datos

Datos atípicos (*Outliers*)



Nº Muestra	Altura (cm)	Peso (kg)	Sexo	Edad
0	185	90	V	29
1	178	150	M	34
2	174	70	V	23
3	162	55	M	21
4	183	85	V	50
5	179	84	V	80

- Una librería muy recomendada es PyOD.
- Ofrece distintos algoritmos, entre ellos: KNN, PCA, Redes Neuronales.



Lectura sugerida:

https://pyod.readthedocs.io/en/latest/

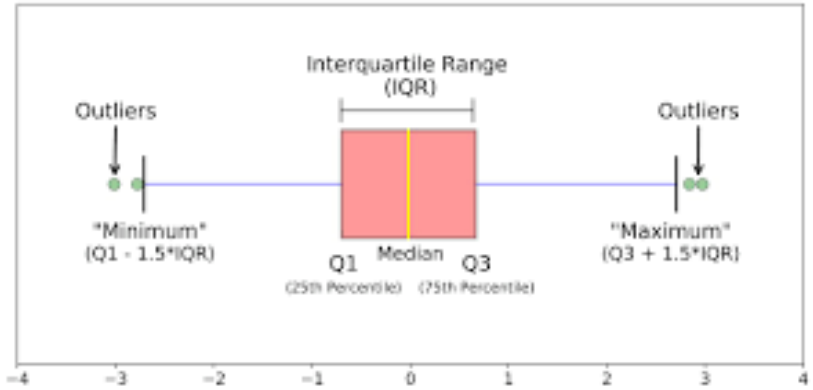
https://github.com/yzhao062/anomaly-detection-resources

Zhao, Y., Nasrullah, Z. **and** Li, Z., 2019. PyOD: A Python Toolbox **for** Scalable Outlier Detection. Journal of machine learning research (JMLR), 20(96), pp.1-7.

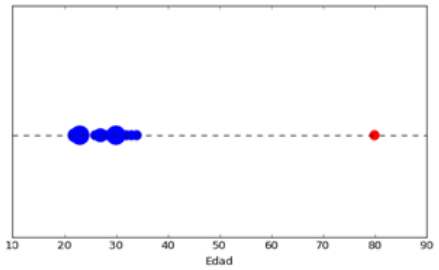
Limpieza de datos

Datos atípicos (*Outliers*)

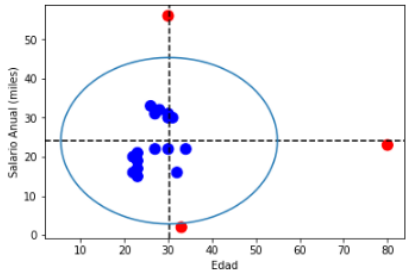
Los *outliers* de un atributo se visualizan fácilmente en un diagrama de cajas Boxplot



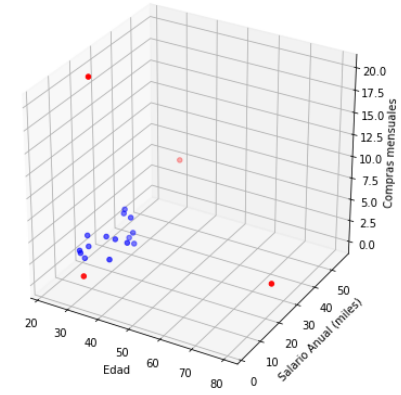
❖ *Outliers* en 1 dimensión



❖ *Outliers* en 2 dimensiones



❖ *Outliers* en 3 dimensiones



Transformación de datos

- Variables nominales - Encoding

  Con frecuencia es necesario convertir la variable nominal en numérica para que pueda ser utilizada por los algoritmos de ML

  Ejemplo: variable REGION (Norte, Centro, Sur)

  - Label encoding

    Asigna un valor numérico a cada una de los niveles de la variable categórica

    REGION

    Norte → 1
    Centro → 2
    Sur → 3

    No se recomienda para variables que no tengan, al menos, escala ordinal.

  - One-hot encoding

    Crea una columna binaria por cada nivel de la variable categórica

    REGION
    Norte
    Centro
    Sur

Es_Norte	Es_Centro	Es_Sur
0	1	0

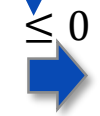
    No se recomienda para variables que tengan muchos niveles de categoría

Transformación de datos

- Binarización
  - Todos los valores iguales o menores a un *threshold* se transforman a 0 y los valores que están por encima se transforman a 1.

```
# binarization
binarizer = Binarizer(threshold=0.0).fit(X)
binaryX = binarizer.transform(X)
```

Sensor 1	Sensor 2	Sensor 3
0.007483	0.097278	0.995229
0.007218	-0.055335	0.998442
0.583487	0.795664	0.162669
0.001062	0.053088	-0.998589



Sensor 1	Sensor 2	Sensor 3
1.0	1.0	1.0
1.0	0.0	1.0
1.0	1.0	1.0
1.0	1.0	0.0

Binarización es útil cuando queremos trabajar con datos categóricos. Por ex. sí estamos trabajando con un problema de clasificación, necesitamos tener valores tipo 0 o 1 para indicar la clase.

Transformación de datos

A nivel de Columnas

- Re-escalamiento

  Min-Max: coloca los valores de cada columna entre [0,1].

	Salario	Edad	Publicaciones
0	2000	23	3
1	5000	30	15
2	10000	41	4
3	15000	35	5

Min-Max

	Salario	Edad	Publicaciones
0	0.000000	0.000000	0.000000
1	0.230769	0.388889	1.000000
2	0.615385	1.000000	0.083333
3	1.000000	0.666667	0.166667

$$z = \frac{\mathrm{x} - min(x)}{[\max(x) - min(x)]}$$

- Aplicable si los datos NO siguen una distribución normal.
- Es útil cuando se desea comparar variables que tienen diferentes escalas o unidades de medida, o cuando se desea mejorar el rendimiento de algunos modelos de ML

  Por ej. algoritmos, como las RN y los modelos de regresión lineal, pueden tener dificultades para trabajar con variables que tienen diferentes escalas.
- **Altera la distribución original**

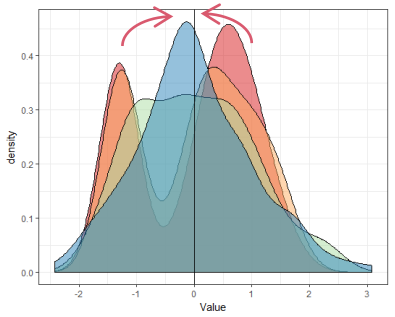
- Estandarización
  - Zscore: coloca los valores de cada columna con media = 0 y desviación estándar = 1

	Bacteria 1	Bacteria 2	Bacteria 3
0	-220	300	123
1	-4500	-200	115
2	30	-450	234
3	-300	500	905

Zscore

	Bacteria 1	Bacteria 2	Bacteria 3
0	0.546028	0.691223	-0.676300
1	-1.728423	-0.625393	-0.700754
2	0.678881	-1.283701	-0.337004
3	0.503515	1.217870	1.714057

$$z = \frac{\mathrm{x} - mean(x)}{stdev(x)}$$



- Aplicable si los datos siguen una distribución normal o próxima a ella.
- Es útil para comparar los valores de la variable en diferentes momentos, se puede analizar su tendencia a lo largo del tiempo.
- **No altera la distribución original**

Transformación de datos

A nivel de Filas

- Normalización
  - Norma unitaria: cada fila (observación) suma 1 como longitud máxima

```
# Normalize data (length of 1)
scaler = Normalizer().fit(X)
normalizedX = scaler.transform(X)
```

```
norm="l1"
norm="l2"
```

Usamos **Normalización** cuando nos interesa conocer la importancia relativa de los valores a nivel de filas respecto a todas las columnas.

- Norma L2; divide cada fila por su longitud para que tenga una longitud de 1. Permite comparar la similitud entre filas, ya que permite medir la distancia coseno entre dos vectores.
- Norma L1; normaliza los datos de tal manera que la suma de los valores absolutos de cada fila sea igual a 1. Es útil si se desea que los datos sean escalados de manera proporcional a su valor absoluto, en lugar de su valor cuadrático (como en la normalización L2).

  Por ejemplo, si se están analizando datos de frecuencia de palabras en un corpus de texto, la normalización L1 puede ser útil para asegurarse de que las palabras menos frecuentes no sean ignoradas en el análisis.

	Sensor 1 (PM10)	Sensor 1(PM2.5)	Sensor 2 (PM10)	Sensor 2(PM2.5)
SJL	36.77	15.57	49.05	19.69
Comas	48.73	19.04	58.47	24.75
Ate	59.55	26.98	61.36	26.31
Miraflores	63.28	28.82	72.80	30.82

Normalización

```
[[0.55508249 0.23504581 0.74046223 0.29724162]
 [0.59231426 0.23143163 0.71070418 0.30083681]
 [0.63729441 0.28873557 0.65666473 0.28156534]
 [0.60104255 0.27373651 0.69146488 0.2927328 ]]
```

Transformación de datos

Escalamiento

- Cambia la escala de los valores a un rango entre 0 y 1.
- Útil cuando la distribución de los datos es desconocida o no gaussiana
- Sensible a valores atípicos
- Mantiene la forma de la distribución original
- Puede que no conserve las relaciones entre los puntos de datos Conserva las relaciones entre los puntos de datos

Ecuación:

(x – min)/(max – min)

Estandarización

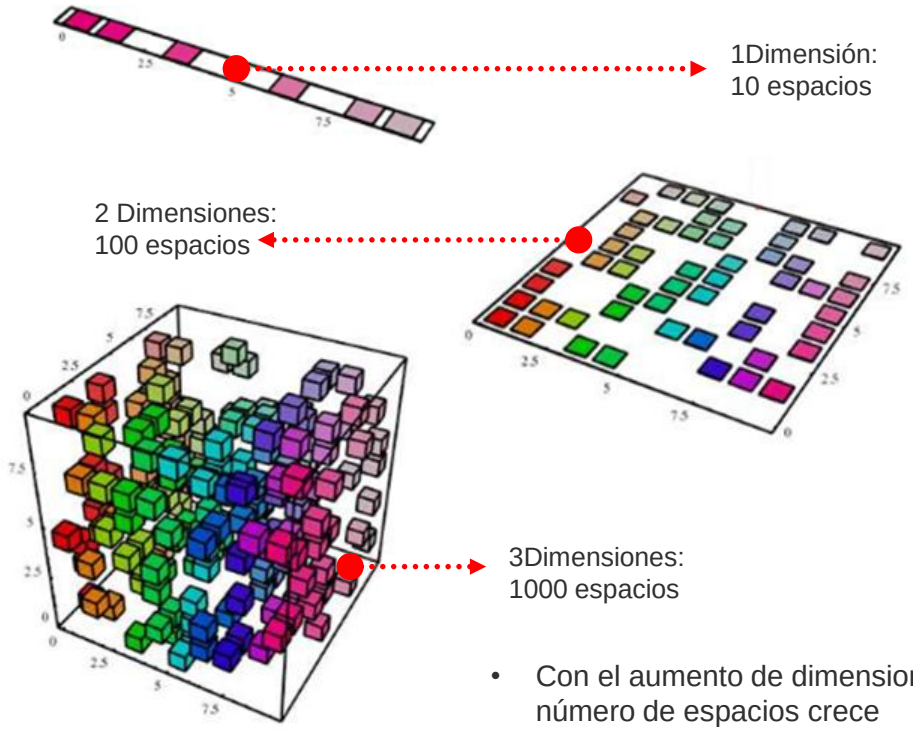
- Centra los datos alrededor de la media y escala a una desviación estándar de 1.
- Útil cuando la distribución de los datos es gaussiana o desconocida
- Menos sensible a valores atípicos
- Cambia la forma de la distribución original
- Conserva las relaciones entre los puntos de datos

Ecuación:

(x – media)/desviación estándar

Reducción de la dimensionalidad

Alta Dimensionalidad

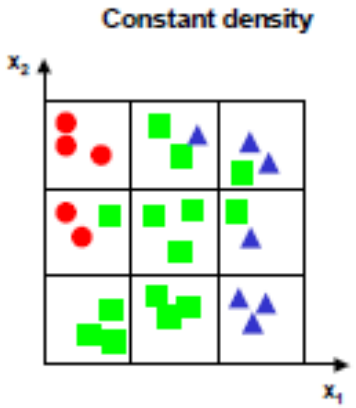


- Con el aumento de dimensiones, el número de espacios crece exponencialmente.

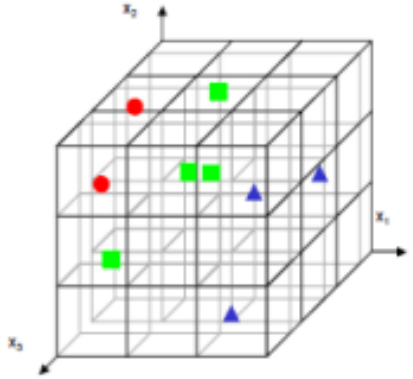
Reducción de la dimensionalidad

Alta Dimensionalidad

- Si las muestras incrementan cuando aumenta la dimensionalidad, es probable que ninguna celda quede vacía.



- El problema aparece cuando el número de muestras se mantiene constante o crece poco, mientras la dimensionalidad aumenta.
  - Con el aumento de dimensiones, el nro. de celdas crece exponencialmente.
  - Las muestras se vuelven muy escasas.



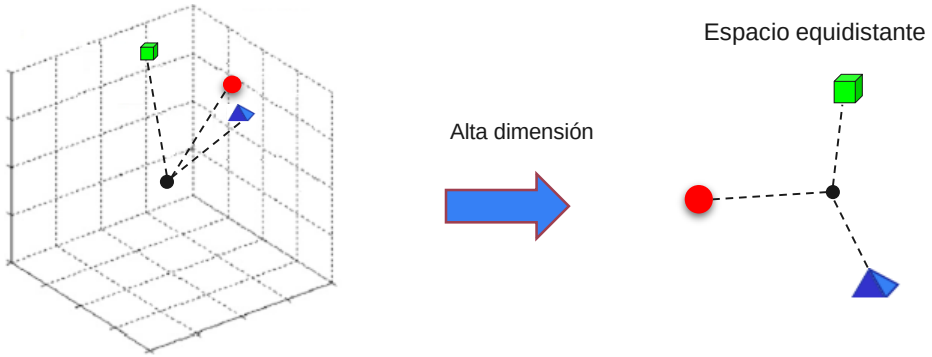
Problema de "*large p, small n*": muchas dimensiones y pocas instancias

Reducción de la dimensionalidad

Maldición de la Dimensionalidad

- Dados esparsos: Las instancias parecen equidistantes

"A medida que aumenta la relación p/n , el espacio donde se representan se vuelve más vacío y las distancias tienden a ser equidistantes."

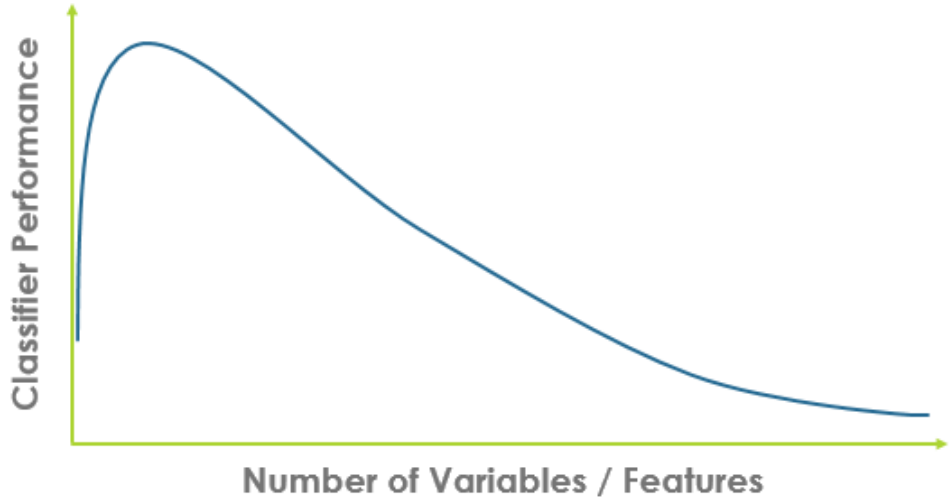


- Afecta el rendimiento de algoritmos que miden la similitud de datos por distancia
  - Ejemplo: algoritmos de agrupación y detección de outliers

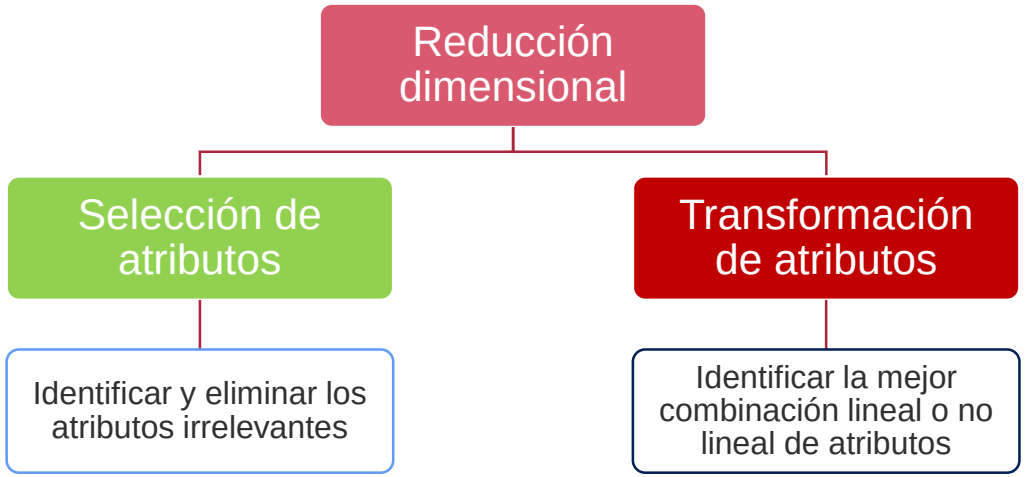
Reducción de la dimensionalidad

Maldición de la Dimensionalidad

- El desempeño de un algoritmo de ML tiende a degradarse para una gran cantidad de atributos



Reducción de la dimensionalidad



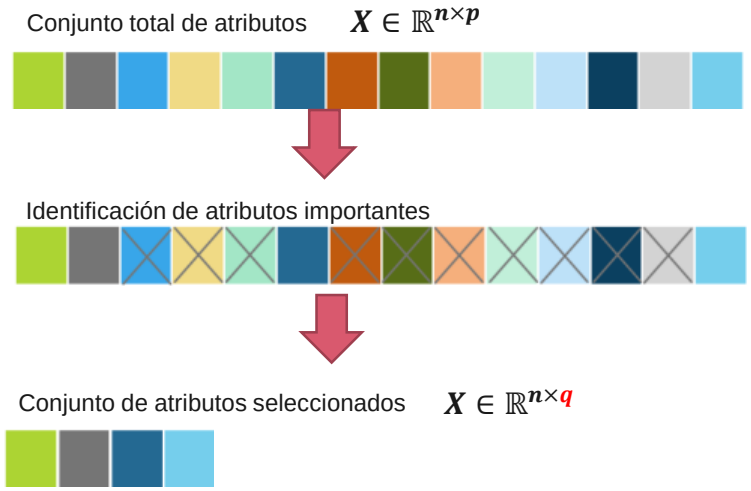
Beneficios:

- Mayor interpretación de los datos
- Ayuda a identificar atributos importantes
- Mejorar la precisión del modelo
- Reduce el tiempo de entrenamiento y costo computacional

Reducción de la dimensionalidad

Selección de atributos

- Proceso de elección de un subconjunto óptimo de atributos de acuerdo con un determinado criterio de selección.

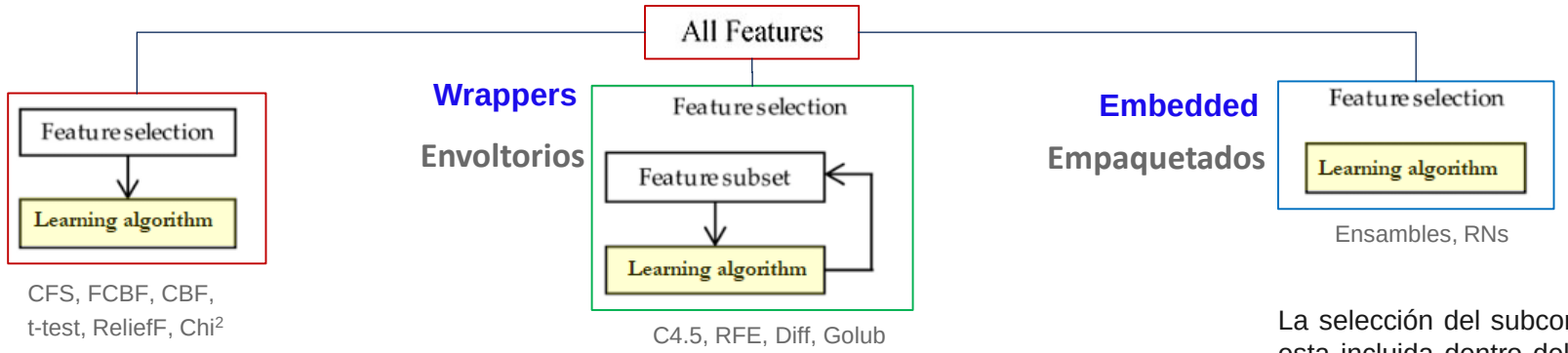


Ventaja

- Mantiene el significado de los atributos.

Reducción de la dimensionalidad

Selección de atributos



Se evalúa la relevancia de los atributos de forma independiente o respecto al target, no implica el uso de un algoritmo ML.
Ej.:

- Chi^2 :mide la dependencia entre variables. Se le usa para seleccionar un subconjunto de atributos que mejor contribuyan a predecir el target.

Utilizan un algoritmo de ML y evalúan la presión del modelo según el subconjunto de atributos seleccionado
Ej.:

- Diff: mide la diferencia entre la media de los valores de dos clases.
- Golub, mide diferencia entre dos clases, con respecto a su desviación estándar.

La selección del subconjunto de atributos esta incluida dentro del modelo, es decir, las diferentes combinaciones de subconjuntos se evalúan y comparan dentro del modelo.
Ej.:

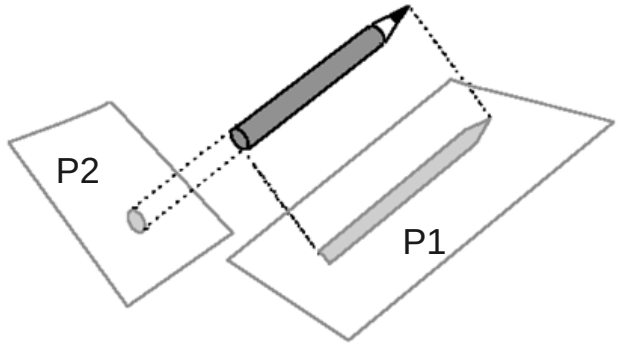
- Ensambles: combinan modelos que asignan implícitamente puntuaciones a subconjuntos de atributos, basados en la precisión del modelo.

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_selection

Reducción de la dimensionalidad

Transformación de atributos

- Buscan espacios de menor dimensión, que represente la información relevante.



Ventajas

- Simplifican el aprendizaje de los modelos de ML
- Facilitan la visualización, interpretación y almacenamiento

Desventajas

- No mantienen el significado de los atributos.

- Tipos:
  a) Lineales: Extraen información buscando Proyecciones lineales.
     - Ejemplo: PCA, LDA, Factor Analysis, ICA
  b) No lineales: Extraen información buscando preservar la geometría o la topología de los dados.
     - Ejemplo. Kernel PCA, MDS, Isomap, CDA

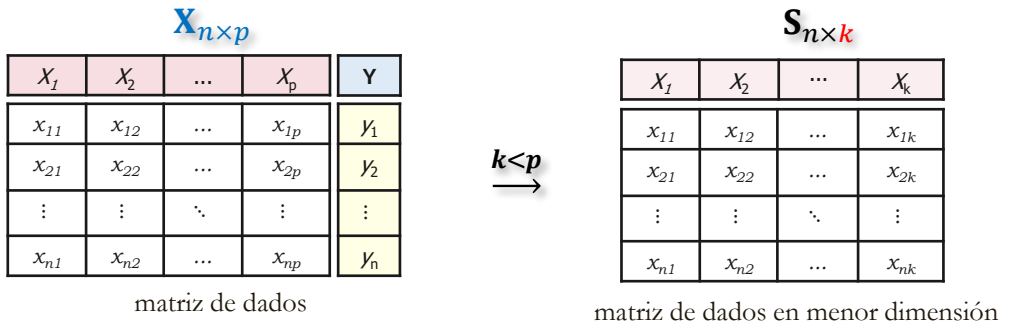
https://scikit-learn.org/stable/modules/decomposition.html#principal-component-analysis-pca

https://scikit-learn.org/stable/modules/manifold.html

Reducción de la dimensionalidad

Análisis de Componentes Principales (PCA)

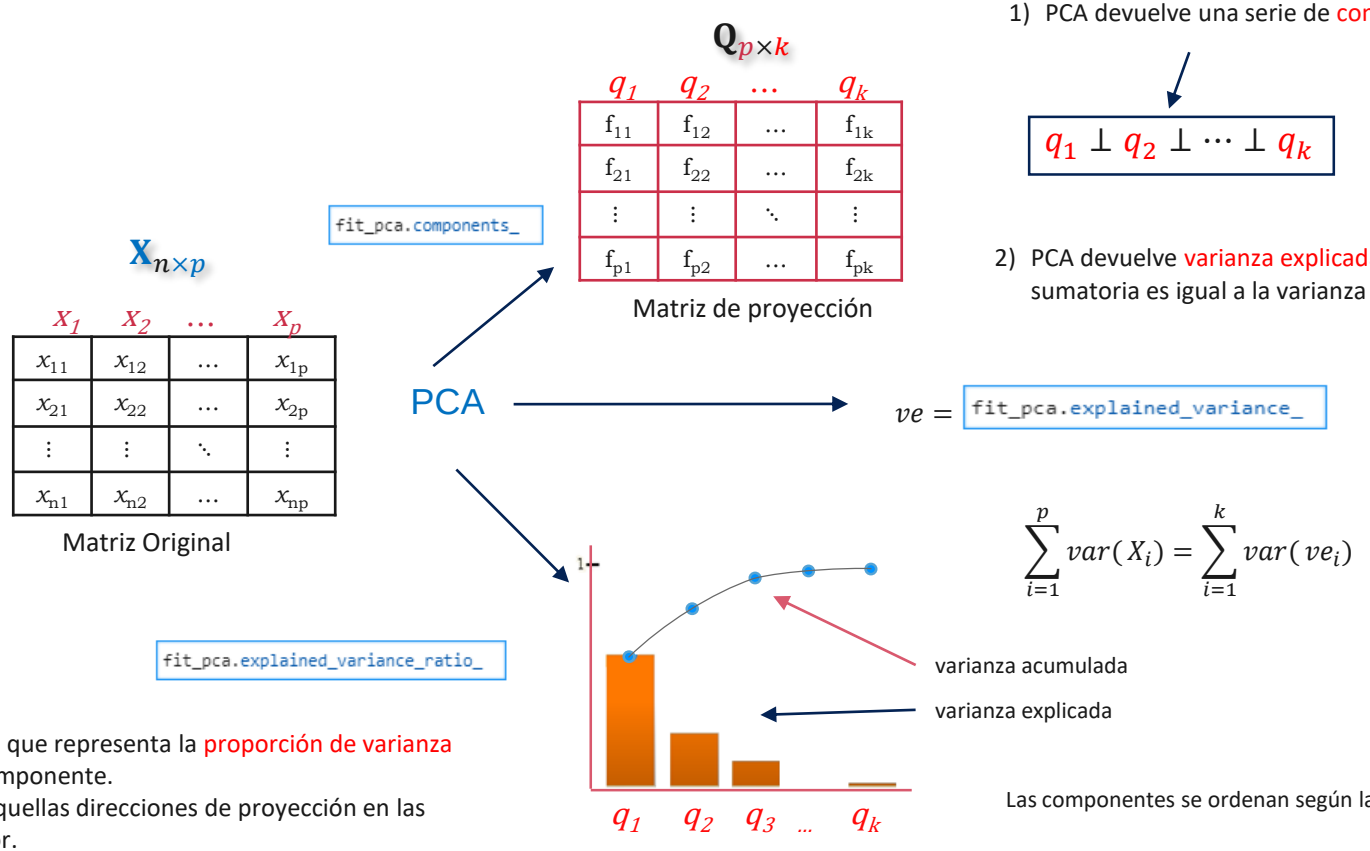
- El Análisis de Componentes Principales (PCA) es una técnica ampliamente utilizada en el análisis, compactación y visualización de datos, que permite transformar un conjunto de observaciones con atributos numéricos correlacionadas entre sí en un conjunto de valores compuestos de atributos no correlacionadas entre sí.



- PCA consigue pasar de p atributos a k, donde k<p.
- PCA devuelve un conjunto de bases de proyección, llamados componentes principales, que contienen la mayor varianza de los datos.
- Los componentes principales obtenidos se ordenan según la proporción de la varianza total explicada por cada uno.

Reducción de la dimensionalidad

Análisis de Componentes Principales (PCA)



1) PCA devuelve una serie de componentes ortogonales entre sí.

$$q_1 \perp q_2 \perp \cdots \perp q_k$$

2) PCA devuelve varianza explicada de las componentes, cuya sumatoria es igual a la varianza total de la matriz X original.

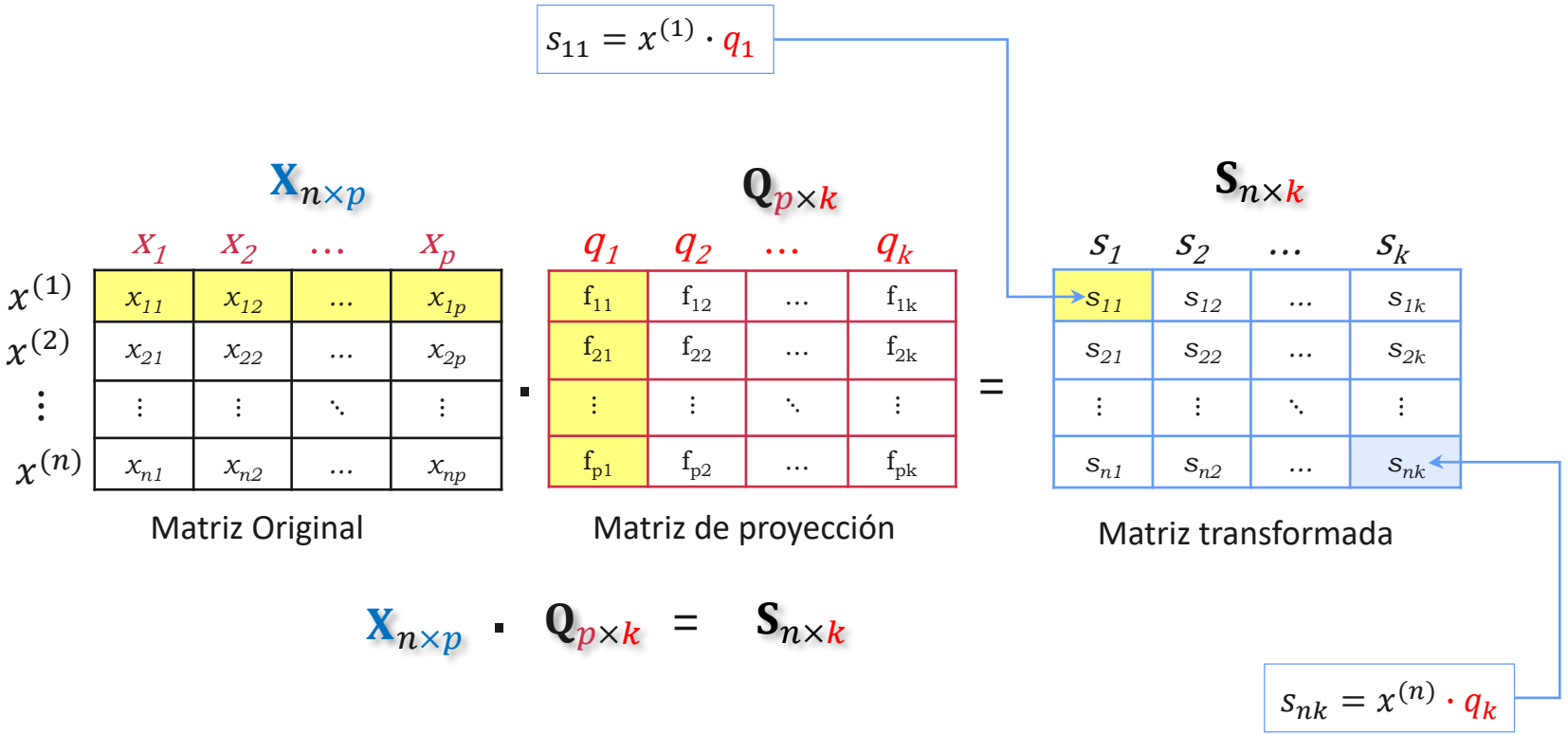
$$\sum_{i=1}^{p} var(X_i) = \sum_{i=1}^{k} var(ve_i)$$

3) PCA devuelve un vector que representa la proporción de varianza acumulada por cada componente.
Con esto se identifica aquellas direcciones de proyección en las que la varianza es mayor.

Las componentes se ordenan según la proporción de varianza explicada.

Reducción de la dimensionalidad

Análisis de Componentes Principales (PCA)



Reducción de la dimensionalidad

Procedimiento de PCA

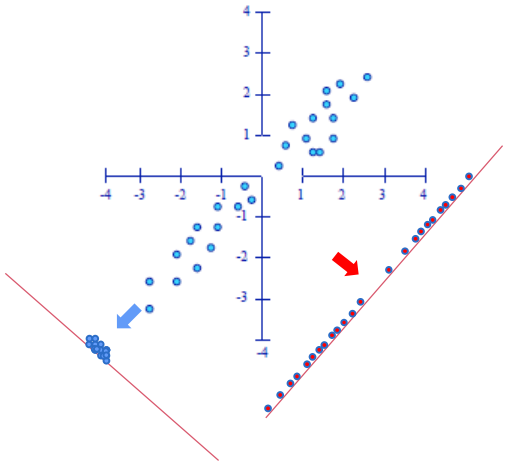
- Estandarizar los datos $\mathbf{X}_{n\times p}$
- Calcular la matriz de covarianza: $\text{Cov} = \frac{1}{n-1} X^T X$
- Calcular autovalores y autovectores de la matriz de covarianza resolviendo: (**Cov** - λ**I**)**v** = 0, donde: λ son los autovalores y **v** los autovectores correspondientes, **I** es matriz identidad
- Ordenar los autovectores por sus autovalores correspondientes en orden descendente
- Seleccionar los k primeros autovectores para formar la matriz de proyección **Q**
- Proyectar los datos: $\mathbf{S}_{n\times k} = \mathbf{X}_{n\times p} \cdot \mathbf{Q}_{p\times k}$

En Python

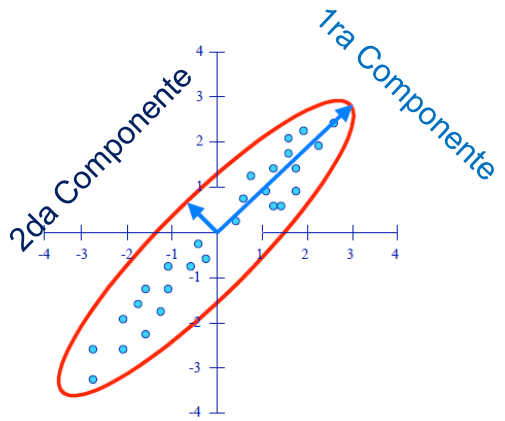
```
from sklearn.decomposition import PCA

# Utilizando PCA de scikit-learn para reducir a 2 componentes
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_std)
```

Reducción de la dimensionalidad

Análisis de Componentes Principales (PCA)



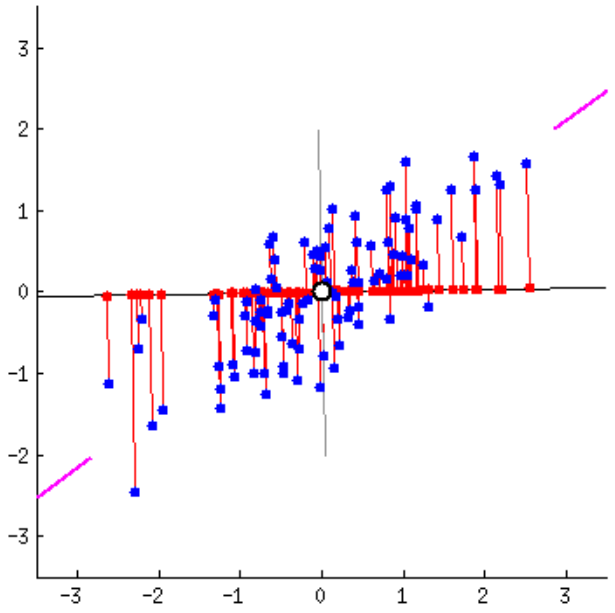
Identificación de la dirección de mayor varianza



Re expresa las coordenadas en términos de los mayores componentes principales. Elimina la menor componente.

Reducción de la dimensionalidad

Análisis de Componentes Principales (PCA)



Consideraciones finales

- Antes de aplicar algoritmos de ML a un conjunto de datos, es importante que los datos sean analizados.
  - Ese análisis, que puede ser realizado por técnicas estadísticas y visualización, porque permiten tener una mejor comprensión de la distribución de los datos y se puede dar soporte a la decisión de cómo vamos a abordar el problema.

Preguntas?