Ayudantía 8

Intro a ML, uso de pandas, numpy, y sklearn

Por Juan José Alonso y Martín José Vial

11 de octubre del 2024

Contenidos

- pandas
- numpy
- Terminología de *machine learning*
- Preprocesamiento de datos
- KNN
- sklearn
- Ejemplo de código


pandas

pandas

¿Qué es pandas?

pandas es una librería de Python para manipular y analizar datos estructurados (tabulares)

¿Para qué vamos a usar pandas?

El *machine learning* se alimenta de una gran cantidad de datos, los cuales se suelen almacenar en formato **.csv (*comma-separated values*)**

Ya vamos a hablar más de esto...

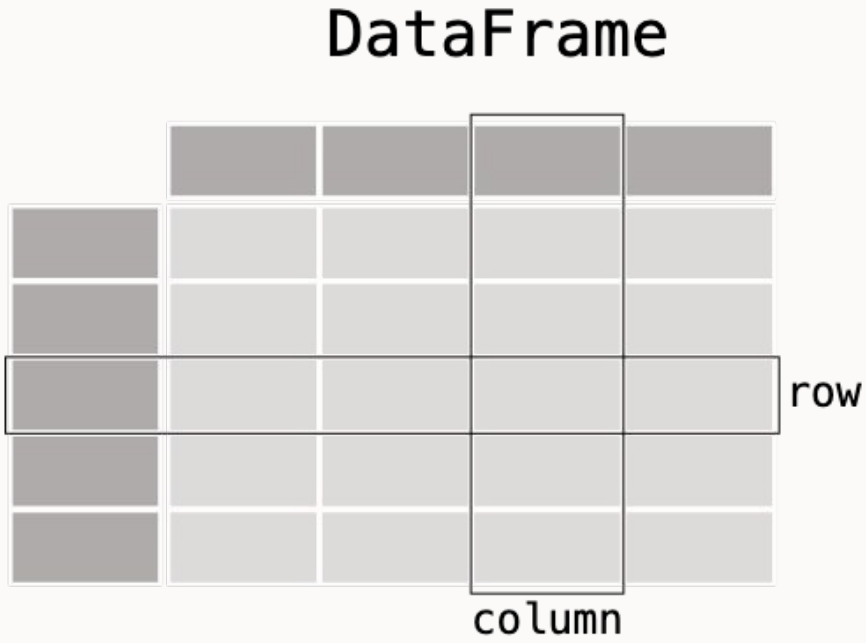
pandas

¿Qué podemos hacer con pandas?

- Cargar un archivo de datos, como **.csv**, y almacenarlo como un **DataFrame**

```
>>> import pandas as pd # Importamos la librería ‘pandas’ bajo el alias ‘pd’
>>> # Supongamos que tenemos un archivo .csv llamado ‘ayudantes.csv’
>>> df = pd.read_csv(‘ayudantes.csv’)
>>> # ¡Y listo! Hemos almacenado los datos en un DataFrame
```

pandas



pandas

¿Qué podemos hacer con pandas?

- Visualizar un DataFrame, sus características (columnas) y sus datos (filas)

```
>>> print(df) # Imprimimos las primeras y últimas 5 filas del DataFrame
```

```
      Nombre     Edad          Major        AñoDeCarrera          RamoFavorito

0  Juan José     22.2        Software                   4              IIC2613
1     Martín     23.9     Computación                   5              IIC2613
2    Ignacio     23.0        Robótica                   5              IIC2613
3    Gonzalo     21.4        Software                   3              IIC2613
4   Josefina     20.8     Transportes                   2              IIC2613
..       ...      ...             ...                 ...                 ...
# Supongamos que aquí se ven las últimas 5 filas… ¿se entiende la idea o no?
```

pandas

¿Qué podemos hacer con pandas?

- Filtrar lo que vemos según ciertos parámetros, como el nombre de una columna específica

```
>>> print(df['Nombre']) # Imprimimos la columna 'Nombre'
```

```
      Nombre

0  Juan José
1     Martín
2    Ignacio
3    Gonzalo
4   Josefina
..       ...
```

pandas

¿Qué podemos hacer con pandas?

- Analizar los datos que contiene un DataFrame, como el valor máximo de una columna

```
>>> print(df['Edad'].max()) # Imprimimos el valor máximo de la columna 'Edad'
```

```
23.9
```

pandas

¿Qué podemos hacer con pandas?

- Analizar los tipos de datos de un DataFrame, para así entender el dominio de sus columnas

```
>>> print(df.dtypes()) # Imprimir el tipo de dato que representa cada columna
```

```
Nombre         object
Edad          float64
Major          object
AñoDeCarrera    int64
RamoFavorito   object
```


NumPy

numpy

¿Qué es numpy?

numpy es una librería de Python nos permite utilizar arreglos y matrices multidimensionales de gran tamaño, además de una gran colección de funciones matemáticas de alto nivel

¿Por qué vamos a usar numpy?

Al igual que en el resto de la computación, el uso de arreglos, matrices, y cálculos numéricos son muy frecuentes en el *machine learning*...

...numpy nos provee herramientas para ello

numpy

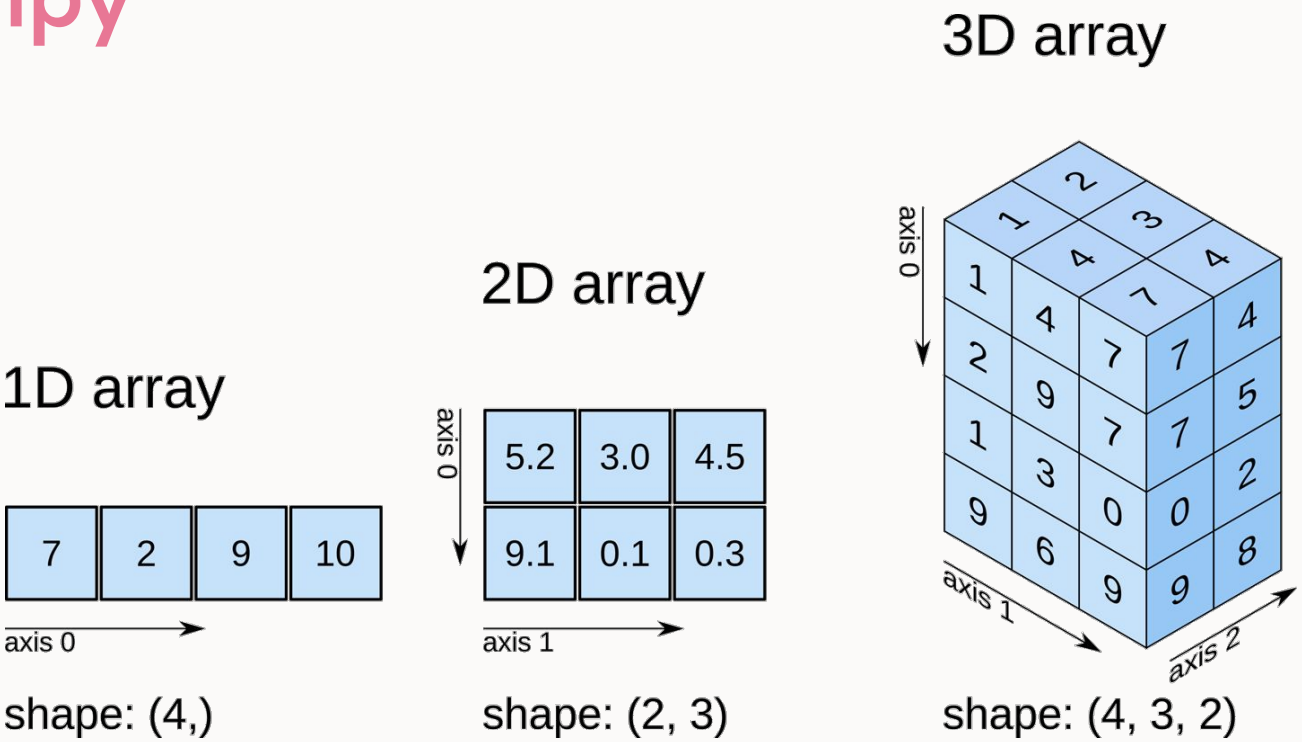
¿Qué podemos hacer con numpy?

- Crear un arreglo, como una matriz 2D (flashbacks de Álgebra Lineal?)

```
>>> import numpy as np # Importamos la librería numpy bajo el alias 'np'
>>>
>>> # Creamos una matriz de 3 x 5 que contenga los primeros 15 números naturales
>>> a = np.arange(15).reshape(3, 5)
>>>
>>> print(a) # Imprimimos la matriz
```

```
[[ 0  1  2  3  4]
 [ 5  6  7  8  9]
 [10 11 12 13 14]]
```

numpy


1D array
7
2
9
10
axis 0
shape: (4,)
2D array
axis 0
5.2
3.0
4.5
9.1
0.1
0.3
axis 1
shape: (2, 3)
3D array
axis 0
axis 1
axis 2
shape: (4, 3, 2)

numpy

¿Qué podemos hacer con numpy?

- Realizar operaciones matemáticas, como calcular la raíz de los elementos de una matriz

```
>>> # Creamos una matriz 1D que contiene los primeros 3 números naturales
>>> b = np.arange(3)
>>> b_sqrt = np.sqrt(B)
>>> b_sqrt
```

```
array([ 0.        , 1.        , 1.41421356        , 1.73205080 ])
```

numpy

¿Qué podemos hacer con numpy?

- Unir diferentes matrices en una única matriz

```
>>> # Aquí, vamos a conectar dos matrices de manera vertical
>>> np.vstack((a, b_sqrt))
```

```
array([[ 0  1           2           3 ]
       [ 4  5           6           7 ]
       [ 8  9          10          11 ]
       [ 0  1  1.41421356  1.73205080 ]
```

¡¡¡MUY IMPORTANTE!!!

Tanto **pandas** como **numpy** tienen una muy buena documentación *online*

ÚSENLA

Terminología de *Machine Learning* 🤓

Terminología Machine Learning

- Dato/instancia

Una pieza individual de información. Puede ser un número, texto, imagen, etc.

id	Altura	Ancho	Color	Clase
1	12	3	Rojo	Auto
2	34	5	Rojo	Auto
3	16	64	Azul	Moto
4	7	1	Verde	Moto

Terminología Machine Learning

- Conjunto de datos/dataset

Colección de datos estructurados, generalmente organizados en filas (instancias) y columnas (atributos/características).

id	Altura	Ancho	Color	Clase
1	12	3	Rojo	Auto
2	34	5	Rojo	Auto
3	16	64	Azul	Moto
4	7	1	Verde	Moto

Terminología Machine Learning

- Atributos/Features/Características/Columnas

Una propiedad o dimensión que describe algún aspecto de los datos. Por ejemplo, en un conjunto de datos de viviendas, podrían ser "tamaño", "número de habitaciones", etc.

id	Altura	Ancho	Color	Clase
1	12	3	Rojo	Auto
2	34	5	Rojo	Auto
3	16	64	Azul	Moto
4	7	1	Verde	Moto

Terminología Machine Learning

- Objetivo

El valor real que queremos predecir con nuestro modelo. También se le puede llamar "target", “clase” o "variable de salida".

id	Altura	Ancho	Color	Clase
1	12	3	Rojo	Auto
2	34	5	Rojo	Auto
3	16	64	Azul	Moto
4	7	1	Verde	Moto

Terminología Machine Learning

- Predicción

El valor que el modelo intenta estimar o predecir.

id	Altura	Ancho	Color	Clase
1	12	3	Rojo	?

Terminología Machine Learning

- Etiqueta

El valor categórico que se asigna a una instancia en clasificación.

Etiqueta 1: Auto.

Etiqueta 2: Moto.

Terminología Machine Learning

- Parámetro

Valores internos que el modelo ajusta (solito, nosotros no hacemos nada) durante el proceso de entrenamiento para minimizar el error del modelo.

Terminología Machine Learning

- Hiperparámetro

Valores que nosotros podemos modificar para que el modelo tenga un mejor rendimiento.

Terminología Machine Learning

- Entrenamiento

Proceso en que el modelo ajusta sus parámetros

Terminología Machine Learning

- Validación

Proceso en que evaluamos el rendimiento del modelo entrenado con un conjunto de datos separado para ajustar hiperparámetros y evitar sobreajuste

Terminología Machine Learning

- Test/prueba

proceso de evaluar la precisión final del modelo usando un conjunto de datos separado que no se usó durante el entrenamiento o la validación.

Preprocesamiento de datos

- La calidad de los datos impacta en la efectividad del modelo.
- Puede ser conveniente procesarlos antes de entrenar

Preprocesamiento de datos

Métodos, técnicas o algoritmos para **limpiar** y **ordenar** los datos antes de realizar la clasificación.

- Representación de características categóricas
- Normalización para dejar en misma escala
- Extracción de características relevantes
- Manejo de datos faltantes

Imputación de datos nulos

Técnicas para manejar datos con atributos faltantes, que pueden hacer fallar o arrojar errores al aplicar algoritmos.

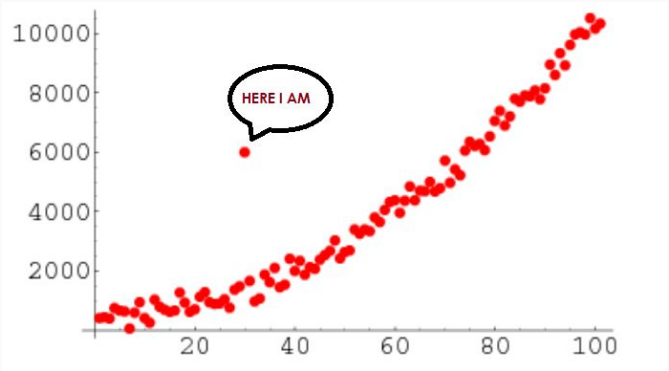
- Se pueden eliminar directamente
- Se pueden reemplazar por media, mediana o moda de la columna
- Se pueden usar algoritmos de machine learning u otros métodos para predecir los datos faltantes por

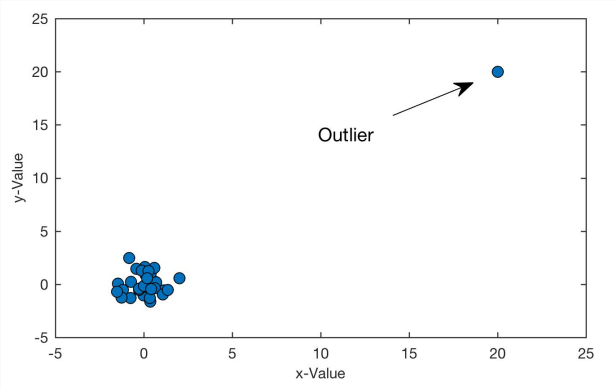
```
# 1. Eliminar filas con datos nulos
df_clean = df.dropna()
```

Outliers

Técnicas para manejar datos atípicos que se encuentran significativamente alejado de la mayoría de los valores del conjunto de datos y pueden afectar negativamente nuestros resultados.

Para eliminar outliers se requieren un análisis más exhaustivo de los datos y otras técnicas de preprocesamiento.





Normalización

	Muestra	A
Peras	1	13234
	:	12129
	50	11957
Manzanas	51	12911
	:	
	125	17288

Columna original

Normalización

	Muestra	A
Peras	1	0.4981
	:	0.3681
	50	0.3479
Manzanas	51	0.4601
	:	
	125	0.9751

Columna normalizada

Algoritmos comunes en Sklearn

MinMax scaler:

- Escala los datos para que estén dentro de un rango específico, típicamente entre 0 y 1
- Útil cuando se desea que todos los valores estén dentro de un rango específico, manteniendo la proporcionalidad entre ellos

Standard Scaler:

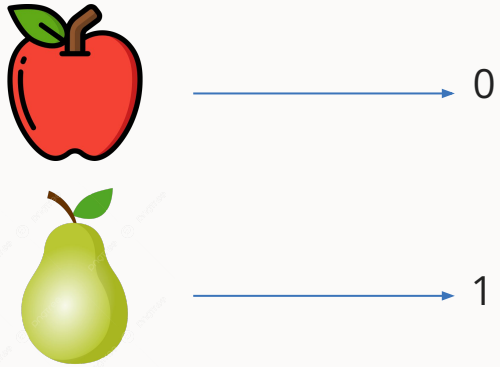
- Estandariza los datos restando la media y dividiendo por la desviación estándar, dando como resultado una distribución con media 0 y desviación estándar 1.
- Utilizado cuando se desea que los datos tengan una distribución normal estándar.

Robust Scaler:

- Escala los datos utilizando los cuartiles (mediana y rango intercuartílico) en lugar de la media y desviación estándar, lo que lo hace robusto frente a outliers.

Codificación de los datos

- Proceso de transformar variables categóricas (como colores, ciudades, o tipos) en un formato numérico que los algoritmos puedan interpretar.
- Los modelos de machine learning solo pueden procesar datos numéricos, por lo que es necesario convertir los datos categóricos en números.



Codificación de los datos

Pandas:

```
1   pd.get_dummies(df['columna_categórica'])
```

Numpy:

- Se debe hacer de manera manual con lógica

Sklearn:

```
from sklearn.preprocessing import OneHotEncoder
encoder = OneHotEncoder()
encoded_data = encoder.fit_transform(df[['columna_categórica']]).toarray()
```

Reducción de dimensionalidad

- Proceso de disminuir el número de variables o características que se utilizan para entrenar un algoritmo de machine learning, sin perder información relevante.

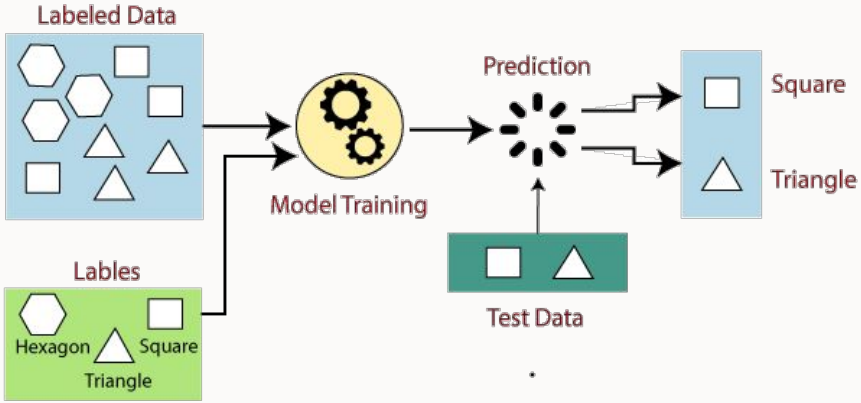
- Puede ser eliminando manualmente columnas que no aporten información relevante (Ej: color de ojos para predecir el peso de una persona), columnas que aportan información muy similar (Ej: Peso en gramos y peso en kilogramos) o utilizar técnicas más avanzadas como PCA para ello.

- Puede mejorar la eficiencia del modelo, reducir el ruido en los datos y evitar realizar clasificaciones en base a datos que no aportan realmente información al problema.

Aprendizaje Supervisado

Definición

Algoritmos entrenados para **predecir** o **clasificar** datos basándose en el aprendizaje previo sobre ejemplos **etiquetados**.



Necesita un conjunto de datos de entrenamiento que consta de entradas (características) y las salidas deseadas (etiquetas).

Set de datos (pera o manzana)

id	Color	Volumen	Área	....	Peso
1	Verde	2	2		200
2	Verde	3	1		300
3	Café	4	3		250
4	Verde	5	2		210
5	Roja	2	2		280
6	Roja	3	1		350
7	Verde	4	3		100

X_train

X_test

Etiqueta
pera
manzana
pera
pera
manzana
manzana
pera

Set de testeo

id	Color	Volumen	Área	....	Peso
5	Roja	2	2		280
6	Roja	3	1		350
7	Verde	4	3		100

Matriz X_test

Etiqueta
manzana
manzana
pera

Vector y_test
(oculto al algoritmo)

Set de entrenamiento

id	Color	Volumen	Área	....	Peso
1	Verde	2	2		200
2	Verde	3	1		300
3	Café	4	3		250
4	Verde	5	2		210

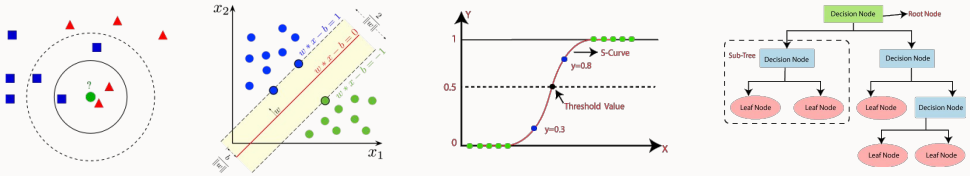
Matriz X_train

Etiqueta
pera
manzana
pera
pera

Vector y_train

Diseño y entrenamiento del clasificador

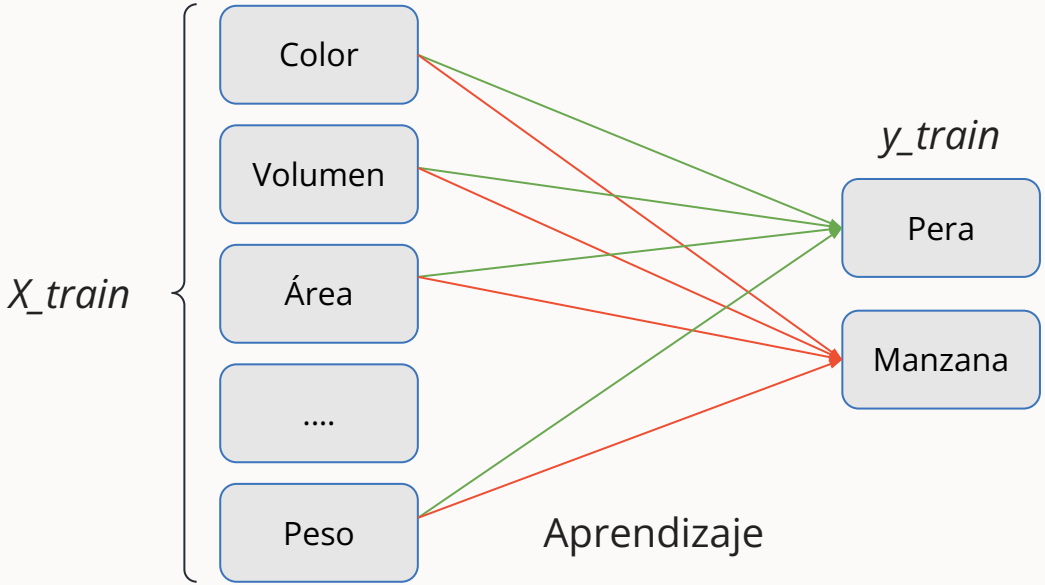
- **Selección de algoritmo:** Se elige un algoritmo de clasificación según el tipo de problema y los datos disponibles.



- **Entrenamiento del algoritmo:** El algoritmo se entrena con el set de entrenamiento y sus etiquetas para saber cómo realizar las clasificaciones.

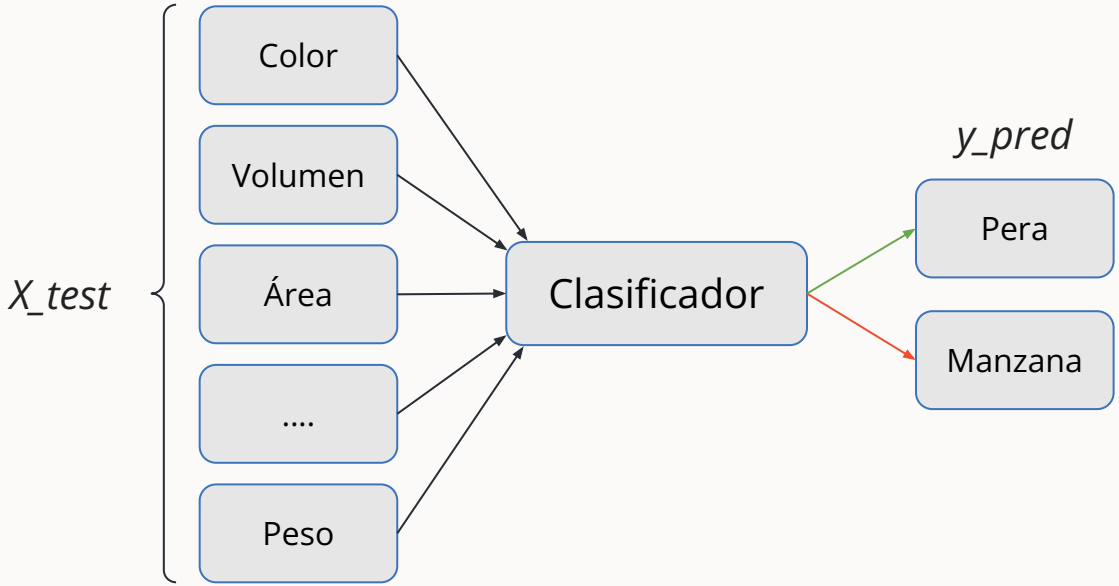
Entrenamiento del clasificador

Aprende **relaciones** y cálculos entre los **atributos** y las **etiquetas**:



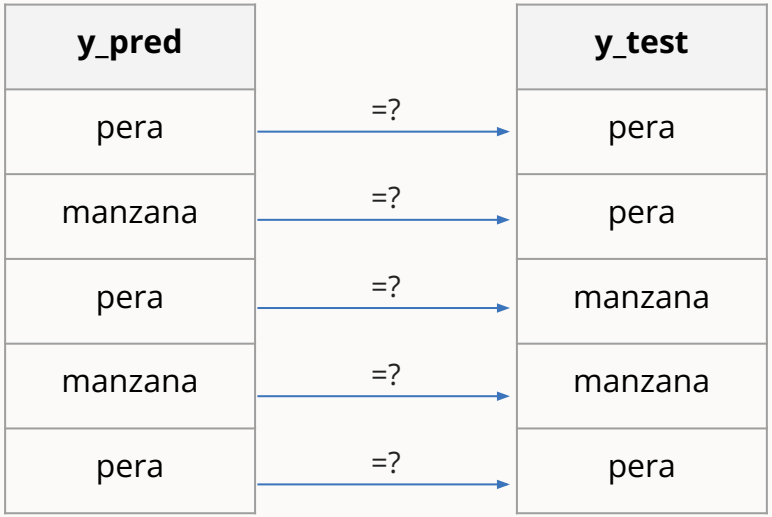
Clasificación y predicciones

Entregamos al algoritmo ya entrenado los **nuevos datos** de testeo para que haga las **predicciones** de la etiqueta de esos datos.



Evaluación

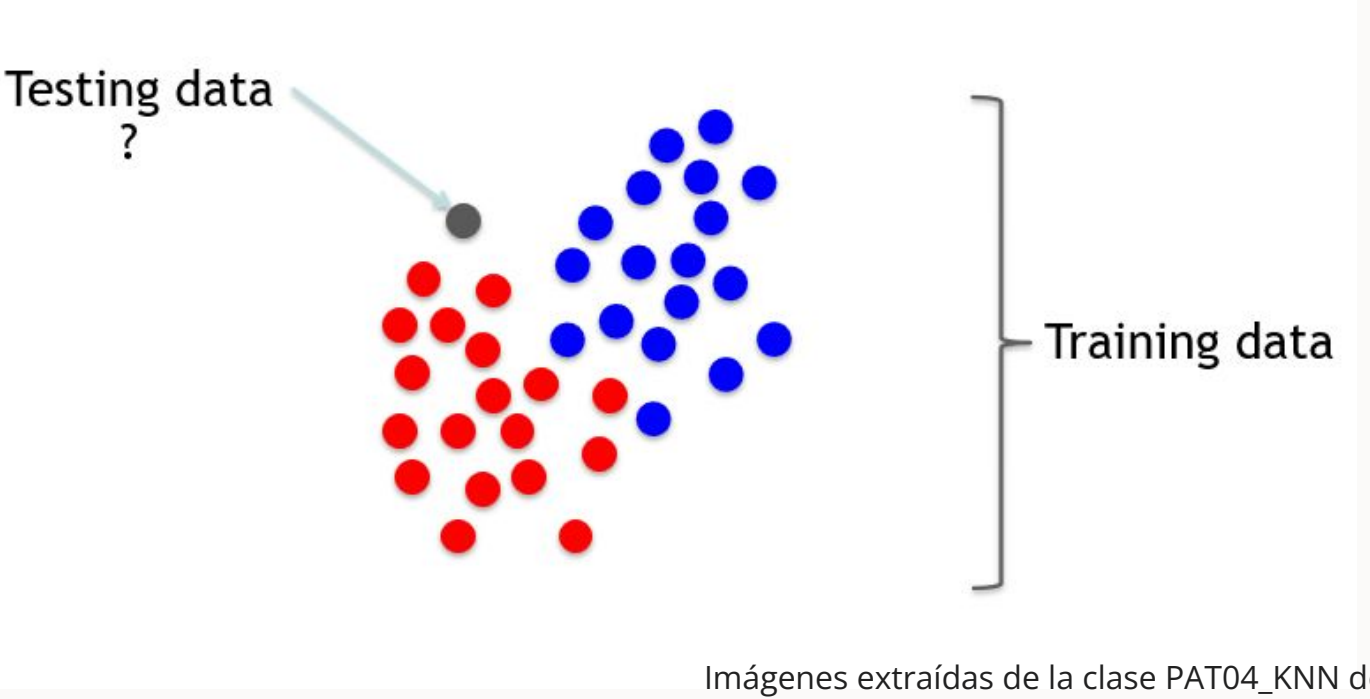
Comparamos la **predicción** de la clasificación del algoritmo con los **valores reales** de la etiqueta del conjunto de testeo y obtenemos distintas **métricas de rendimiento**.



Algoritmo: k-nearest neighbor (KNN)

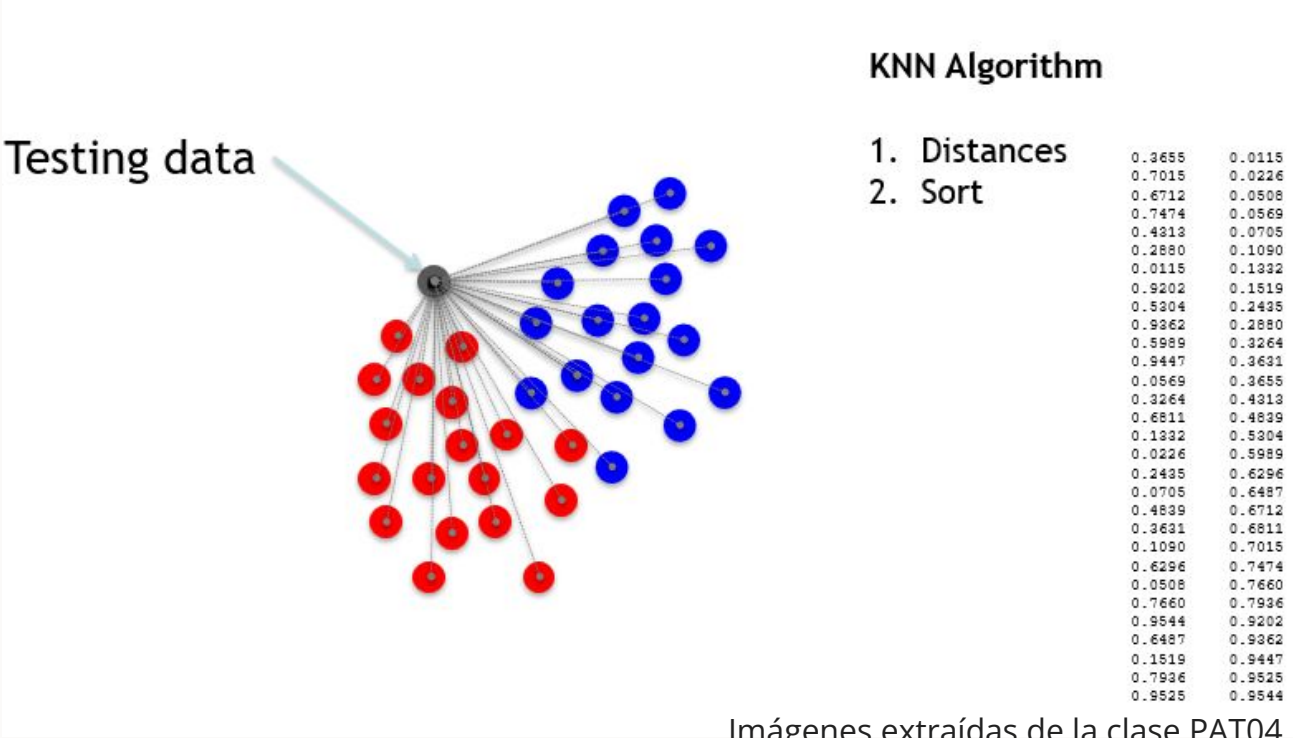
- Clasificación de datos nuevos
- No hay entrenamiento como tal
- De los algoritmos más simples para probar experimentos rápido
- Basado en distancias euclidianas. Preprocesamiento y normalización es fundamental.

Algoritmo: k-nearest neighbor (KNN)



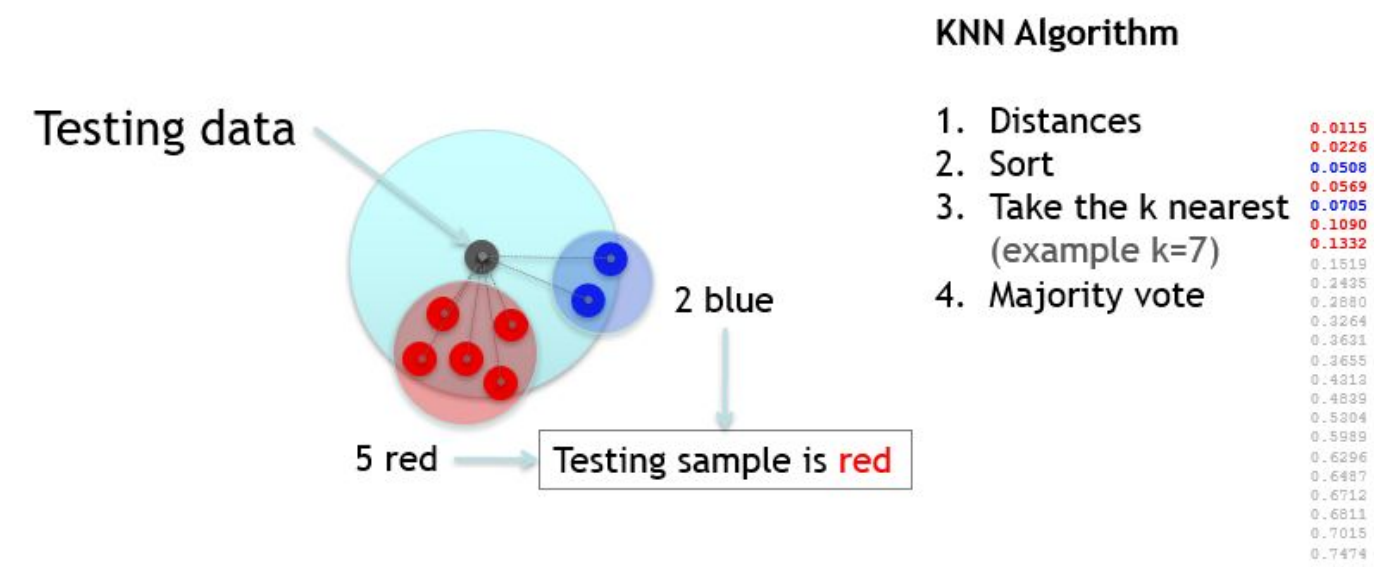
Imágenes extraídas de la clase PAT04_KNN del profesor Domingo Mery

Algoritmo: k-nearest neighbor (KNN)



Imágenes extraídas de la clase PAT04_KNN del profesor Domingo Mery

Algoritmo: k-nearest neighbor (KNN)



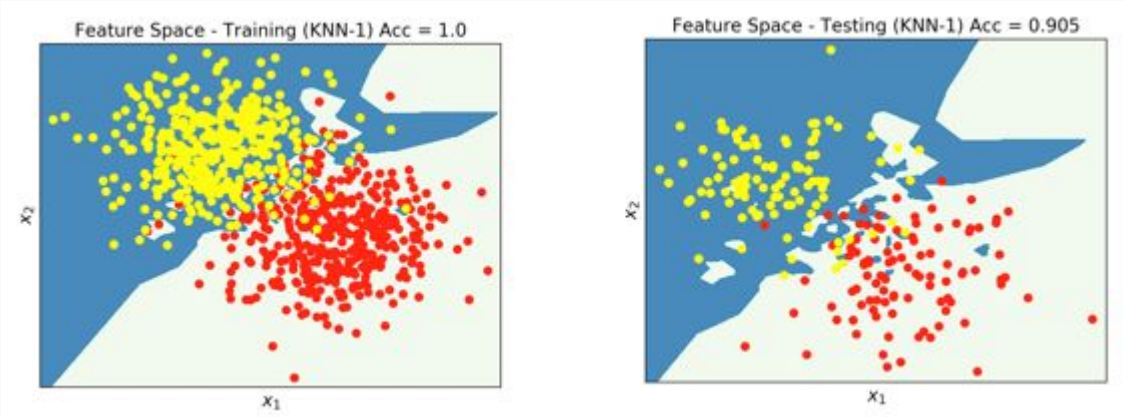
Imágenes extraídas de la clase PAT04_KNN del profesor Domingo Mery

Algoritmo: k-nearest neighbor (KNN)

- ¿Qué parámetros tiene este modelo?
- Y ¿Qué hiper parámetros tiene?

Algoritmo: k-nearest neighbor (KNN)

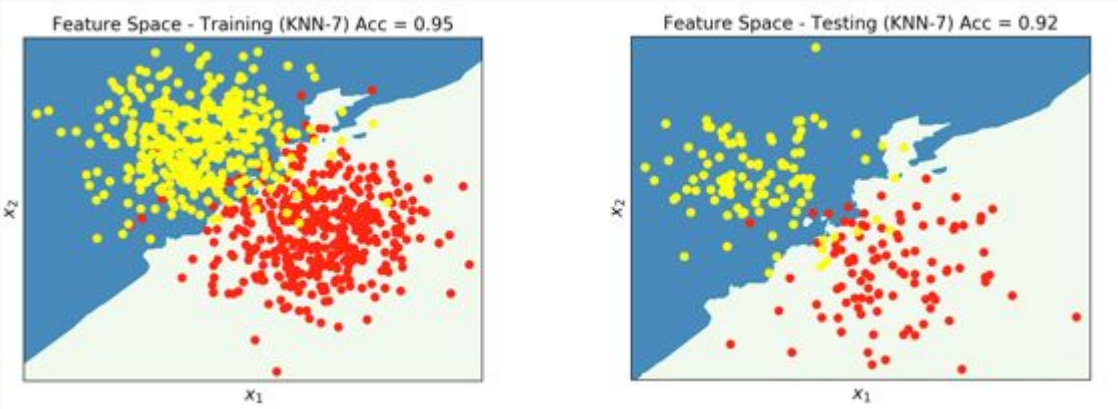
K=1



Imágenes extraídas de la clase PAT04_KNN del profesor Domingo Mery

Algoritmo: k-nearest neighbor (KNN)

K=7



Imágenes extraídas de la clase PAT04_KNN del profesor Domingo Mery

Sklearn

- Librería con algoritmos de machine learning ya definidos y listos para implementar.
- Cada algoritmo es un objeto de python con métodos para entrenar y clasificar en base a un set de datos.

Para instalar en consola:

```
$ pip install -U scikit-learn
```

Sklearn

Importación de algoritmos e instanciación en script:

```
from sklearn import algoritmo_de_clasificacion
from sklearn import train_test_split

# Instanciamos el algoritmo
clasificador = algoritmo_de_clasificacion()
```

Sklearn

Separación de conjunto de datos:

```
X = nuestro_data_Set["atributos"]
y = nuestro_data_Set["variable_objetivo"]

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=porcentaje_de_test)
```

Sklearn

Uso de métodos para entrenar y clasificar

```
# Entrenamos al algoritmo
clasificador.fit(X_train, y_train)

# Realizamos predicciones de testeo
y_pred = clasificador.predict(X_test)

# Evaluamos el rendimiento
lógica para comparar y_pred con y_test
```

Ejemplo en código