Selección de instancias

La selección de instancias es otra forma de reducir los datos. En vez de reducir las características (columnas), en este caso, intentamos reducir el número de ejemplos (filas) sin perder información, por ejemplo, eliminando ejemplos duplicados o muy similares.

Lectura: Instance Selection: The myth behind Data Sampling (https://towardsdatascience.com/instance-selection-the-myth-behind-data-sampling-d3556ea2e37d)

Los algoritmos de clustering que se estudiarán en otros cursos pueden ser utilizados para elegir algunos ejemplos representativos de nuestro conjunto de datos. Como ejemplo, aqui usaremos el método *Condensed Nearest Neighbour* que es adecuado para problemas de clasificación.

In [41]:
```
# Importar los paquetes que usaremos
import pandas as pd
import numpy as np
from sklearn import datasets

# Cargar el conjunto de datos
dataset = datasets.fetch_openml(name='balance-scale', version=2, as_frame=True)
tabla = dataset.frame
tabla
```

Out[41]:

	left-weight	left-distance	right-weight	right-distance	binaryClass
0	1.0	1.0	1.0	1.0	N
1	1.0	1.0	1.0	2.0	N
2	1.0	1.0	1.0	3.0	N
3	1.0	1.0	1.0	4.0	N
4	1.0	1.0	1.0	5.0	N
...	...	...	...	...	...
620	5.0	5.0	5.0	1.0	P
621	5.0	5.0	5.0	2.0	P
622	5.0	5.0	5.0	3.0	P
623	5.0	5.0	5.0	4.0	P
624	5.0	5.0	5.0	5.0	N

625 rows × 5 columns

In [42]:
```
tabla.describe()
```

Out[42]:

	left-weight	left-distance	right-weight	right-distance
count	625.000000	625.000000	625.000000	625.000000
mean	3.000000	3.000000	3.000000	3.000000
std	1.415346	1.415346	1.415346	1.415346
min	1.000000	1.000000	1.000000	1.000000
25%	2.000000	2.000000	2.000000	2.000000
50%	3.000000	3.000000	3.000000	3.000000
75%	4.000000	4.000000	4.000000	4.000000
max	5.000000	5.000000	5.000000	5.000000

Aplicamos *Condensed Nearest Neighbour*:

In [37]:
```
from imblearn.under_sampling import CondensedNearestNeighbour
X = dataset.data
y = dataset.target
cnn = CondensedNearestNeighbour()
X_resampled, y_resampled = cnn.fit_resample(X, y)
tabla_reducida = X_resampled.join(y_resampled)
tabla_reducida
```

Out[37]:

	left-weight	left-distance	right-weight	right-distance	binaryClass
0	4.0	2.0	5.0	4.0	N
1	1.0	1.0	1.0	1.0	N
2	1.0	1.0	1.0	3.0	N
3	1.0	1.0	3.0	1.0	N
4	1.0	2.0	1.0	2.0	N
...	...	...	...	...	...
373	5.0	5.0	4.0	5.0	P
374	5.0	5.0	5.0	1.0	P
375	5.0	5.0	5.0	2.0	P
376	5.0	5.0	5.0	3.0	P
377	5.0	5.0	5.0	4.0	P

378 rows × 5 columns

In [38]:
```
tabla_reducida.describe()
```

Out[38]:

	left-weight	left-distance	right-weight	right-distance
count	378.000000	378.000000	378.000000	378.000000
mean	3.399471	3.417989	2.552910	2.563492
std	1.327809	1.319068	1.344447	1.361634
min	1.000000	1.000000	1.000000	1.000000
25%	2.000000	2.000000	1.000000	1.000000
50%	4.000000	4.000000	2.000000	2.000000
75%	5.000000	5.000000	4.000000	4.000000
max	5.000000	5.000000	5.000000	5.000000

Vemos como la tabla se ha reducido entorno a un 40% manteniendo una distribución estadística similar a la original.

Opcional, ver este ejemplo con ilustración gráfica: Condensed nearest-neighbour (http://glemaitre.github.io/imbalanced-learn/auto_examples/under-sampling/plot_condensed_nearest_neighbour.html)

Ejercicio: aplicar sobre otro conjunto de datos.