**Data Preprocessing**

**The Steps of Data Preprocessing:**

The machine learning process involves 3 steps:

1. **Data Pre-Processing:**
   1. Import the data
   2. Clean the data
   3. Split into training and test sets
2. **Modelling:**
   1. Build the model
   2. Train the model
   3. Make predictions
3. **Evaluation:**
   1. Calculate performance metrics
   2. Make a verdict

**Feature Scaling:** It is the process of converting character values into relational quantities, basically they become values in comparison to each other.

Feature scaling is always applied to columns, never rows.

Two main methods of feature scaling: Normalisation and Standardisation.

Normalisation:

It is the process of taking the minimum inside a column, subtracting that from every value in the column, and dividing by the difference of the maximum and minimum.

Range: [0:1]

Standardisation:

­­Similar to Normalisation, we just subtract the average and divide by the standard deviation.

Range: [-3:+3]

**Importing the files:**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

numpy allows us to create and manage arrays  
pyplot module from matplotlib library allows us to make graphs and charts  
pandas allows us to perform data preprocessing very efficiently

**Importing the dataset:**

dataset = pd.read\_csv('Data.csv')

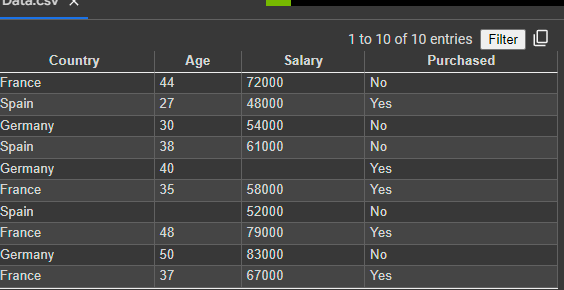
X = dataset.iloc[:, :-1].values

y = dataset.iloc[:, -1].values

read\_csv is a pandas function that reads the csv file

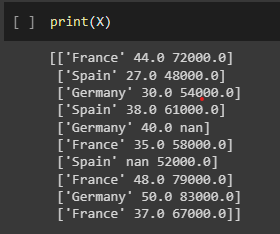
iloc: Purely integer-location based indexing for selection by position.

.iloc[] is primarily integer position based (from 0 to length-1 of the axis), but may also be used with a boolean array.

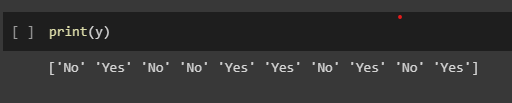


X is getting the independent variables from the dataset, which are:

Country, Age Salary



y is getting the dependent variable, which is Purchased.



We do X = dataset.iloc[ :, :-1].values to make sure it is a new list as opposed to a subset of the csv file

Same for y.

**Taking care of missing data:** We cannot leave empty data entries, since it leads to errors in the results.

A good solution is to replace the missing data with the mean of the variables.

from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing\_values=np.nan, strategy='mean')

imputer.fit(X[:, 1:3])

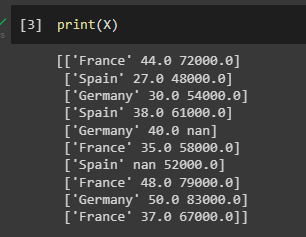
X[:, 1:3] = imputer.transform(X[:, 1:3])

Using SimpleImputer module from sklearn, we pass 2 arguments:  
All the values that are missing as np.nan, and the strategy to resolve those missing values as mean.

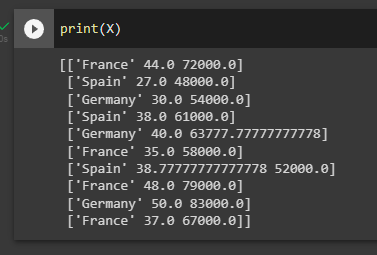
We take our imputer object and perform a “fit” method on it, for columns 1 and 2, which computes the means for the columns.

We then call our imputer object and perform a transform method on, which fills in the missing data with our computed means.

Before:



After:



**Encoding Categorical Data:**

Our data has 2 categorical variables: Country – France, Spain and Germany; and Purchased – Yes, No.

We need to encode them to have a smooth training.

There is no relational order between the countries: If we were to encode the countries as:

France = 0  
Spain = 1  
Germany = 2

It could lead to an establishment of a hierarchical relationship between the countries and the encoding during the training of the machine learning model, which is undesirable, therefore we must use a different method.