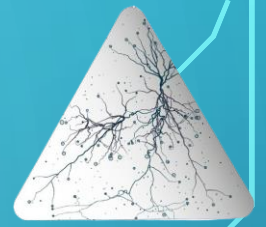
Ensias_ai_club

DATA CLEANING

DATA CELL

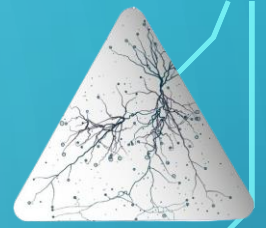
BY : KARMOUCHI ASMAE



Ensias_ai_club

INTRODUCTION:

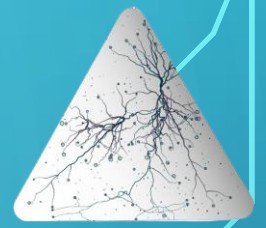
- Le **nettoyage de données** ou le Data Cleaning, vient souvent après la collecte de données .
- les données peuvent être inexactes, incohérentes et redondantes. Mais il est impossible de fournir des données avec certaines anomalies a un algorithme de ML.
- Les étapes de ce processus ne sont pas toujours les mêmes et varient d'une base de données à l'autre.



Ensias_ai_club

DATA CLEANING :

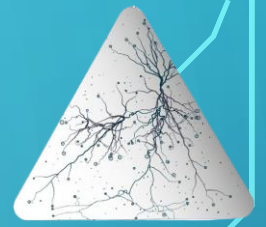
- Importation des données
- Traitement des données manquantes
- Valeurs aberrantes
- Normalisation et standardisation



Ensias_ai_club

IMPORTER LES DONNÉES

IMPORT DES DONNÉES:



Ensias_ai_club

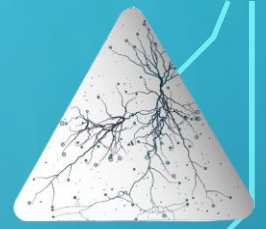
- Nous utilisons le module ***Pandas***. Développé pour apporter à **Python** les outils nécessaires pour manipuler de gros volumes de données.
- Ce module a pour objectif de devenir le meilleur outil de manipulation de données : à la fois performant, facile d'utilisation et flexible.
- **Pandas** est aujourd'hui le module le plus utilisé pour gérer des bases de données aux formats : CSV, Excel, …

Fichier Csv

Ensias_ai_club

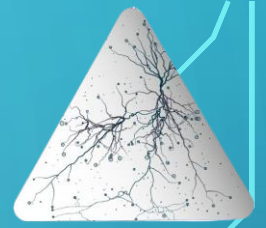
```
1  import pandas as pd #Importation du module pandas sous le raccourci pd.
2
3  data=pd.read_csv('data.csv') #data.csv correspond au chemin menant à votre dataset.
4
5  data.head() #On affiche les 5 premières lignes
6
```

FICHIER EXCEL



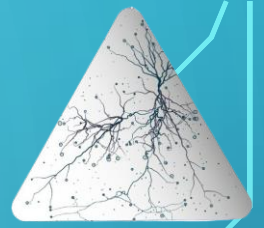
Ensias_ai_club

```
1  data=pd.ExcelFile('data.xlsx') # On lit le fichier Excel dans un DataFrame data
2
3  print(data.sheet_names). # On affiche le nom des feuilles du classeur Excel
4
5  data.parse('sheet name') # On accède à la feuille souhaitée
```



Ensias_ai_club

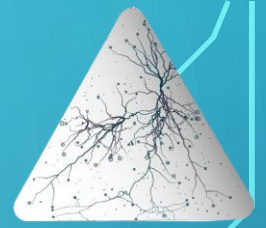
TRAITEMENT DES DONNÉES MANQUANTES



Ensias_ai_club

1- Combien de points de données manquants avons-nous ?

```
# get the number of missing data points per column
missing_values_count = nfl_data.isnull().sum()


# look at the # of missing points in the first ten columns
missing_values_count[0:10]
```



Ensias_ai_club

Pour prendre une meilleure idée de l'ampleur de ce problème

```
# combien de valeurs manquantes au total avons-nous ?
total_cells = np.product(nfl_data.shape)
total_missing = missing_values_count.sum()

# % de données manquantes
percent_missing = (total_missing/total_cells) * 100
print(percent_missing)
```

Entrée [24]:

```
# modules we'll use
import pandas as pd
import numpy as np


# read in all our data
nfl_data = pd.read_csv('gender_submission.csv')


# Regardez les cinq premières lignes du fichier nfl_data.
nfl_data.head()
```

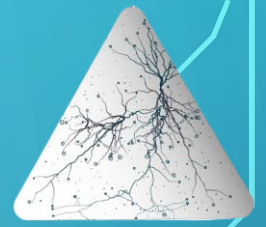
Out[24]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

Entrée [25]:

```
#total des valeurs manquantes
total_missing = missing_values_count.sum()
print(total_missing)
```

```
0
```



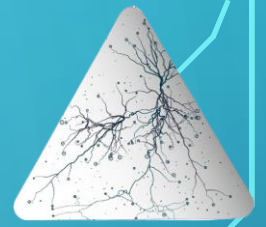
Ensias_ai_club

2- COMPRENDRE POURQUOI LES DONNÉES SONT MANQUANTES

↳ l'intuition des données.

❖ "regarder vraiment vos données et essayer de comprendre pourquoi c'est comme ça et comment cela va affecter votre analyse"

❖ Pour traiter les valeurs manquantes, vous devrez utiliser votre intuition pour comprendre pourquoi la valeur est manquante.

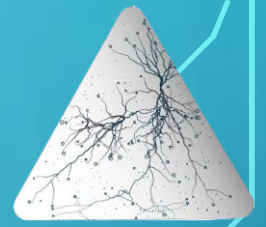


Ensias_ai_club

Cette valeur manque-t-elle parce qu'elle n'a pas été enregistrée ou parce qu'elle n'existe pas ?

- Si une valeur manque parce qu'elle n'existe pas :
  - cela n'a aucun sens d'essayer de deviner ce que cela pourrait être.
  - Ces valeurs que vous souhaitez probablement conserver en tant que NaN
- Si une valeur est manquante parce qu'elle n'a pas été enregistrée:

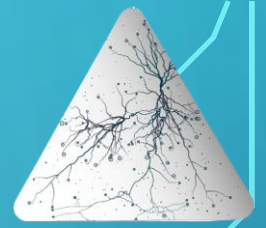
vous pouvez essayer de deviner ce qu'elle aurait pu être en fonction des autres valeurs de cette colonne et de cette ligne. C'est ce qu'on appelle l'imputation.



Ensias_ai_club

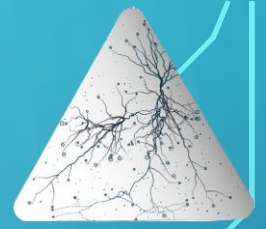
3- SUPPRIMER LES VALEURS MANQUANTES

- La méthode dropna() permet de supprimer les données manquantes
- on peut spécifier l'axe de suppression axis=0 on supprime les lignes où il y a des données manquantes, axis=1 on supprime les colonnes où il y a des données manquantes



Ensias_ai_club

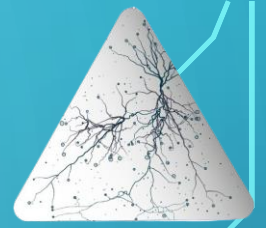
```
# supprimer toutes les lignes contenant une valeur manquante
 nfl_data.dropna()
# supprimer toutes les colonnes avec au moins une valeur manquante
 columns_with_na_dropped = nfl_data.dropna(axe=1)
 colonnes_with_na_dropped.head()
```



Ensias_ai_club

 # combien de données avons-nous perdu ?

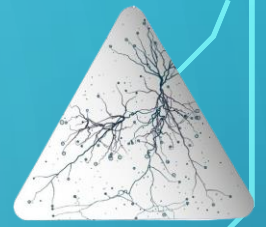
```
 print("Colonnes dans le jeu de données d'origine : %d \n" %
nfl_data.shape[1])

 print("Colonnes avec na supprimées : %d" %
columns_with_na_dropped.shape[1])
```



4-REMPLISSAGE AUTOMATIQUE DES VALEURS MANQUANTES

- Nous pouvons utiliser la fonction **fillna()** de Panda pour remplir les valeurs manquantes
- Spécifier par quoi nous voulons que les valeurs NaN soient remplacées.

Remplissez les valeurs nulles avec la moyenne de la colonne, la médiane ou autre mode selon les besoins.



Ensias_ai_club

- Remplacer toutes les valeurs NaN par 0:

```
nfl_data.fillna(0)
```

- Moyenne:

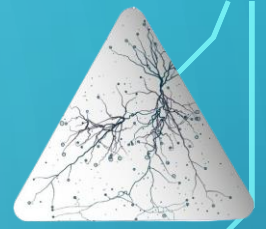
```
nfl_data.fillna(df.mean())
```

- Valeur la plus présente:

```
nfl_data.fillna(df.mode())
```

- Médiane:

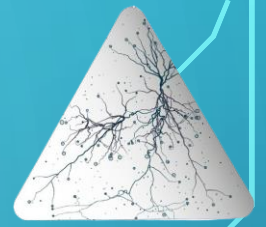
```
nfl_data[‘colonneX'].fillna(nfl_data[‘colonneX'].median(), inplace=True)
```



Ensias_ai_club

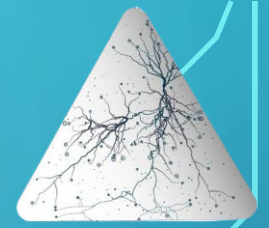
Suppression des colonnes qui n’ont pas de données utiles:

```
train.drop(columns=[‘colonneX', ‘colonneY', ‘assembleurs', ‘simula’ ], axis=1, inplace=True)
```



Ensias_ai_club

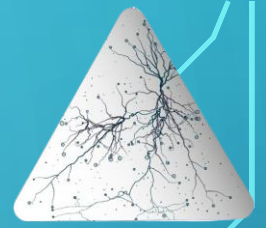
VALEURS ABERRANTES



Ensias_ai_club

- Saisie de données incohérente
- une erreur de mesure

DÉTECTION DES VALEURS ABERRANTES



Ensias_ai_club

L'INCOHÉRENCE DES DONNÉES

- Espaces
- Lettres majuscules
- Correspondance floue …

Asmae.K

Entrée [40]:
```
# modules we'll use
import pandas as pd
import numpy as np
# read in all our data
professors = pd.read_csv("pakistan_intellectual_capital.csv")
```

Entrée [41]:
```
professors.head()
```

Out[41]:

	Unnamed: 0	S#	Teacher Name	University Currently Teaching	Department	Province University Located	Designation	Terminal Degree	Graduated from	Country	Year	Area of Specialization/Research Interests	Other Information
0	2	3	Dr. Abdul Basit	University of Balochistan	Computer Science & IT	Balochistan	Assistant Professor	PhD	Asian Institute of Technology	Thailand	NaN	Software Engineering & DBMS	NaN
1	4	5	Dr. Waheed Noor	University of Balochistan	Computer Science & IT	Balochistan	Assistant Professor	PhD	Asian Institute of Technology	Thailand	NaN	DBMS	NaN
2	5	6	Dr. Junaid Baber	University of Balochistan	Computer Science & IT	Balochistan	Assistant Professor	PhD	Asian Institute of Technology	Thailand	NaN	Information processing, Multimedia mining	NaN
3	6	7	Dr. Maheen Bakhtyar	University of Balochistan	Computer Science & IT	Balochistan	Assistant Professor	PhD	Asian Institute of Technology	Thailand	NaN	NLP, Information Retrieval, Question Answering...	NaN
4	24	25	Samina Azim	Sardar Bahadur Khan Women's	Computer Science	Balochistan	Lecturer	BS	Balochistan University of Information	Pakistan	2005.0	VLSI Electronics DLD Database	NaN

Entrée [42]:

```
# get all the unique values in the 'Country' column
countries = professors['Country'].unique()
# sort them alphabetically and then take a closer look
countries.sort()
countries
```

Out[42]:

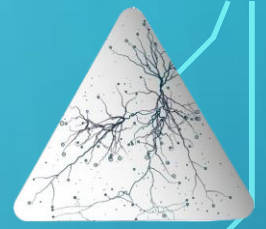
```
array([' Germany', ' New Zealand', ' Sweden', ' USA', 'Australia',
       'Austria', 'Canada', 'China', 'Finland', 'France', 'Greece',
       'HongKong', 'Ireland', 'Italy', 'Japan', 'Macau', 'Malaysia',
       'Mauritius', 'Netherland', 'New Zealand', 'Norway', 'Pakistan',
       'Portugal', 'Russian Federation', 'Saudi Arabia', 'Scotland',
       'Singapore', 'South Korea', 'SouthKorea', 'Spain', 'Sweden',
       'Thailand', 'Turkey', 'UK', 'USA', 'USofA', 'Urbana', 'germany'],
      dtype=object)
```

Entrée [43]:

```
# convert to lower case
professors['Country'] = professors['Country'].str.lower()
# remove trailing white spaces
professors['Country'] = professors['Country'].str.strip()
countries = professors['Country'].unique()
countries.sort()
countries
```

Out[43]:

```
array(['australia', 'austria', 'canada', 'china', 'finland', 'france',
       'germany', 'greece', 'hongkong', 'ireland', 'italy', 'japan',
       'macau', 'malaysia', 'mauritius', 'netherland', 'new zealand',
       'norway', 'pakistan', 'portugal', 'russian federation',
       'saudi arabia', 'scotland', 'singapore', 'south korea',
       'southkorea', 'spain', 'sweden', 'thailand', 'turkey', 'uk',
       'urbana', 'usa', 'usofa'], dtype=object)
```



Ensias_ai_club

- Il semble y avoir une autre incohérence : "Corée du Sud" et "Corée du Sud" devraient être identiques.
- Nous allons utiliser le package **fuzzywuzzy** pour aider à identifier les chaînes les plus proches les unes des autres.
- **Fuzzywuzzy** renvoie un rapport donné pour deux chaînes. Plus le rapport est proche de 100, plus la distance d'édition entre les deux chaînes est petite.

- pip install fuzzywuzzy

Entrée [44]:

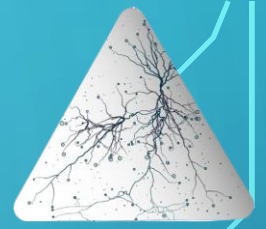
```
# helpful modules
import fuzzywuzzy
from fuzzywuzzy import process
import chardet
```

Entrée [45]:

```
# obtenir le top 10 des correspondances les plus proches de "corée du sud"
matches = fuzzywuzzy.process.extract("south korea", countries, limit=10, scorer=fuzzywuzzy.fuzz.token_sort_ratio)

# take a look at them
matches
```

Out[45]:

```
[('south korea', 100),
 ('southkorea', 48),
 ('saudi arabia', 43),
 ('norway', 35),
 ('ireland', 33),
 ('portugal', 32),
 ('singapore', 30),
 ('netherland', 29),
 ('macau', 25),
 ('usofa', 25)]
```



Ensias_ai_club

- les villes sont très proches de « south korea » : « south korea » et « southkorea ».
- Remplaçons toutes les lignes de notre colonne "Pays" qui ont un rapport > 47 par "Corée du Sud".

Entrée [47]:

```
# fonction pour remplacer les lignes dans la colonne fournie du dataframe fourni
# qui correspondent à la chaîne fournie au-dessus du ratio fourni avec la chaîne fournie
```

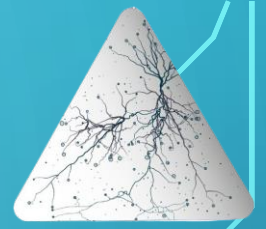
Entrée [54]:

```
def replace_matches_in_column(df, column, string_to_match, min_ratio = 47):
    # obtenir une liste de chaînes uniques
    strings = df[column].unique()

    # obtenir les 10 correspondances les plus proches de notre chaîne d'entrée
    matches = fuzzywuzzy.process.extract(string_to_match, strings,
                                         limit=10, scorer=fuzzywuzzy.fuzz.token_sort_ratio)

    # que les matchs avec un ratio > min_ratio
    close_matches = [matches[0] for matches in matches if matches[1] >= min_ratio]

    # obtenir les lignes de toutes les correspondances proches dans notre dataframe
    rows_with_matches = df[column].isin(close_matches)

    # remplacer toutes les lignes par des correspondances proches par les correspondances d'entrée
    df.loc[rows_with_matches, column] = string_to_match

    # let us know the function's done
    print("All done!")
```

Entrée [55]:

```
#utilisez la fonction pour remplacer les correspondances proches de "south korea" par "south korea"
replace_matches_in_column(df=professors, column='Country', string_to_match="south korea")
```

```
All done!
```



Ensias_ai_club

SCORE Z

- ✓ Z-Score est aussi appelé score standard. Cette valeur/score aide à comprendre à quelle distance se trouve le point de données de la moyenne.
- ✓ Et après avoir défini une valeur seuil, on peut utiliser les valeurs de score z des points de données pour définir les valeurs aberrantes.

Zscore = (data_point -mean) / std. déviation

Entrée [2]:

```
import pandas as pd
import numpy as np
train = pd.read_csv("train.csv")
train.head()
```

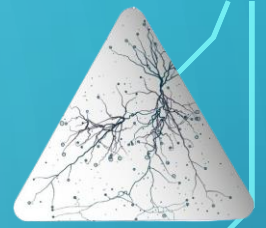
Out[2]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Entrée [6]:

```
# Z score
from scipy import stats
z = np.abs(stats.zscore(train['Fare']))
print(z)
```

```
[5.02445171e-01 7.86845294e-01 4.88854258e-01 4.20730236e-01
 4.86337422e-01 4.78116429e-01 3.95813561e-01 2.24083121e-01
 4.24256141e-01 4.29555021e-02 3.12172378e-01 1.13845709e-01
 4.86337422e-01 1.87093118e-02 4.90279793e-01 3.26266659e-01
 6.19988892e-02 3.86670720e-01 2.85997284e-01 5.02948539e-01
 1.24919787e-01 3.86670720e-01 4.86756223e-01 6.63597416e-02
 2.24083121e-01 1.64441595e-02 5.02948539e-01 4.64700108e+00
 4.89776426e-01 4.89442190e-01 9.02720170e-02 2.30172882e+00
 4.92377828e-01 4.37007438e-01 1.00606170e+00 3.98582080e-01
```

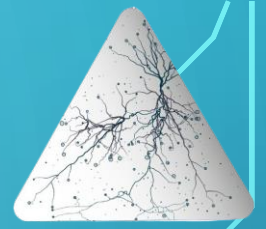


Ensias_ai_club

- **Autre méthode(selon le besoin):**

  * **visualisation**

```
# Box Plot

        import seaborn as sns

        sns.boxplot(df_boston['DIS'])

# Position of the Outlier

        print(np.where(df_boston['DIS']>10))
```



Ensias_ai_club

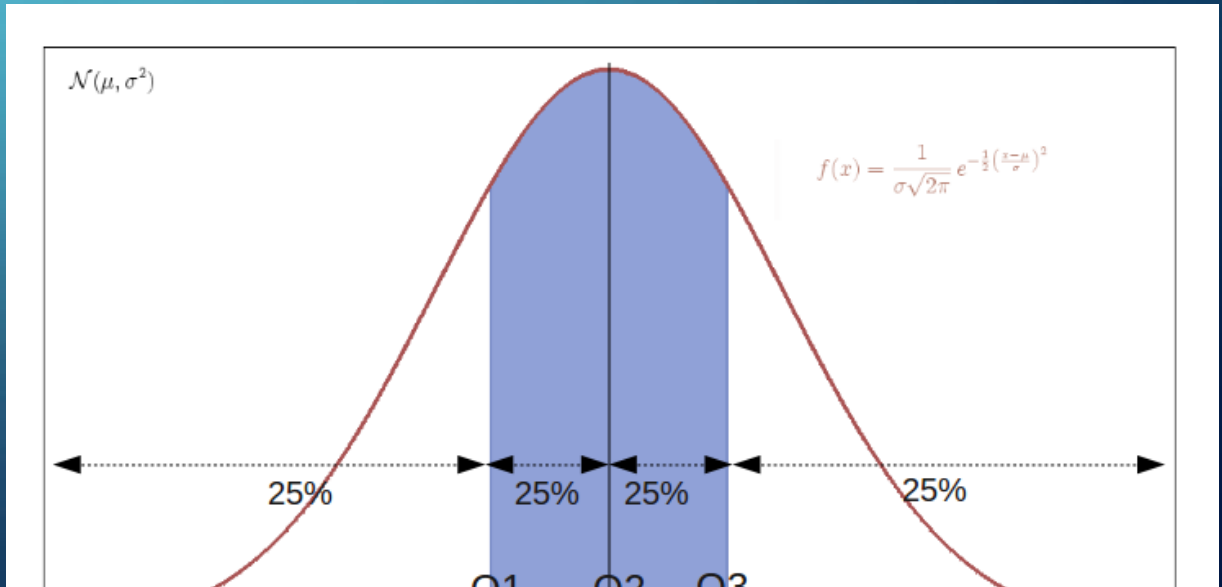
* IQR (gamme interquartile):

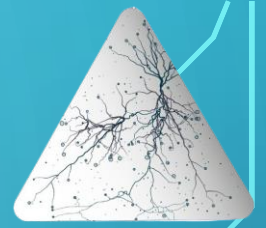
```
    # IQR
Q1 = np.percentile(df_boston['DIS'], 25, interpolation = 'midpoint')
Q3 = np.percentile(df_boston['DIS'], 75, interpolation = 'midpoint')
IQR = Q3 - Q1
```

Pour définir la valeur de base aberrante est définie au-dessus et en dessous de la plage normale des ensembles de données, à savoir les limites supérieure et inférieure, définissez la limite supérieure et la limite inférieure (la valeur 1,5 * IQR est prise en compte) :

supérieur = Q3 +1,5*IQR

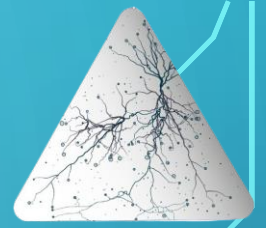
inférieur = Q1 – 1,5*IQR





Ensias_ai_club

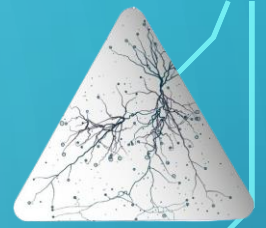
NORMALISATION ET STANDARDISATION



Ensias_ai_club

- Pour équilibrer le poids de chacune des variables au sein du Data
- Mettre variables à la même échelle

NORMALISATION



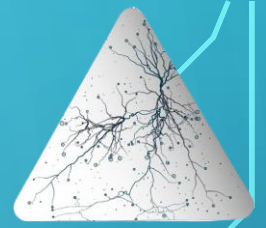
Ensias_ai_club

- Le procédé de normalisation n'a besoin que du min et du max.

- L'idée est la suivante, on ramène toutes les valeurs de la variable entre 0 et 1, tout en conservant les distances entre les valeurs.

- Pour ce faire rien de plus simple, la formule est la suivante :

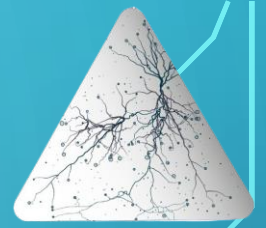
$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \in [0\ ,\ 1]$$



Ensias_ai_club

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

train1 = scaler.fit_transform(train)
```

STANDARDISATION



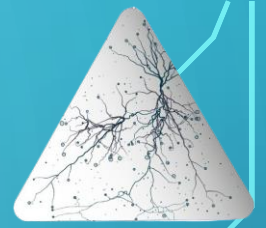
Ensias_ai_club

La Standardisation est le processus de transformer une feature en une autre qui répondra à la loi normale (Gaussian Distribution) $X \sim \mathcal{N}(\mu, \sigma)$

$\mu = 0$ La moyenne de la loi de distribution

$\sigma = 1$ est l'Écart-type (Standard Deviation)

$$X_{standard} = \frac{X - \mu}{\sigma} ,$$



Ensias_ai_club

```
from sklearn.preprocessing import StandardScaler

Scaler = StandardScaler().fit(train)

Data_train_scaled = Scaler.transform(train)

Data_train_scaled
```

```python
#Scale train datas

from sklearn.preprocessing import MinMaxScaler

#2-Standart MinMaxScaler
scaler = MinMaxScaler()
x_train_mm = scaler.fit_transform(x_train)
x_test_mm = scaler.fit_transform(x_test)
```

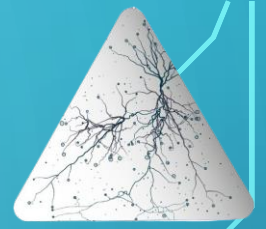
```python
x_train_mm
```

```
array([[0.52103744, 0.37030774, 0.51143667, ..., 0.67835052, 0.29469742,
        0.19670733],
       [0.34450282, 0.28610078, 0.32775897, ..., 0.25876289, 0.21111768,
        0.1424636 ],
       [0.58777036, 0.28643896, 0.57639417, ..., 0.77388316, 0.39917209,
        0.1497442 ],
       ...,
       [0.25457901, 0.40886033, 0.24870431, ..., 0.41408935, 0.27735068,
        0.24642529],
       [0.22334233, 0.31788975, 0.21104278, ..., 0.19728522, 0.37886852,
        0.09536928],
       [0.1566094 , 0.60534325, 0.15119895, ..., 0.31364261, 0.13029765,
        0.18227732]])
```

```python
#Import the function
from sklearn.preprocessing import StandardScaler
Scaler = StandardScaler().fit(x_train)
Data_train_scaled = Scaler.transform(x_train)
Data_train_scaled
```

```
StandardScaler()
```

```
array([[ 1.11369962,  0.32228604,  1.07850587, ...,  1.28735999,
         0.23491182,  0.02863219],
       [ 0.0504453 , -0.24584785, -0.01954056, ..., -0.59926636,
        -0.43000188, -0.41418997],
       [ 1.51562685, -0.24356619,  1.46682929, ...,  1.71691128,
         1.06605395, -0.35475435],
       ...,
       [-0.49115878,  0.58239553, -0.49213842, ...,  0.09914077,
         0.09691086,  0.43450788],
       [-0.67929493, -0.0313716 , -0.71728338, ..., -0.87569343,
         0.9045301 , -0.79864743],
       [-1.08122217,  1.90804128, -1.07503665, ..., -0.35250614,
        -1.07296089, -0.08916814]])
```



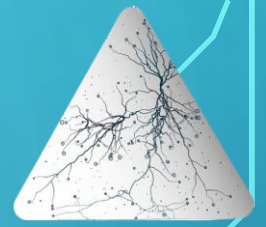
Ensias_ai_club

Python3

```
1  # declaring numeric variables
2  num= 2021
3  n=2.3
4
5
6  # concatenating numeric value within string
7  print("%d is here!! %d " %(num,n))
8
```

```
2021 is here!! 2
```

Improve your Coding Skills with Practice



Ensias_ai_club

MERCI POUR VOTRE ATTENTION