Intervenant	**Rakia Jaziri**
Intitulé TD/TP :	**Atelier : Apprentissage supervisé avec Python**
Contenu	• **Data Preprocessing (données hétérogènes, données manquantes, etc.)** • **Feature engineering** • **Feature selection** • **Classification** • **Evaluation de la qualité d'un classifieur**

Dans cet atelier pratique, vous allez expérimenter des algorithmes de traitement de données pour répondre à différents problèmes liés à l'apprentissage supervisé avec le langage **Python**.

Pour lancer le notebook Python, il faut taper la commande **jupyter notebook** dans votre dossier de travail. Une fenêtre va se lancer dans votre navigateur pour ouvrir l'application Jupyter. Créer un nouveau notebook Python et taper le code suivant dans une nouvelle cellule :

```
import numpy as np
np.set_printoptions(threshold=np.nan)
import pandas as pd
import warnings
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
```

I. Apprentissage supervisé : Feature engineering et Classification

L'objectif dans cette partie est de construire un bon classifieur sur un jeu de données de *crédit scoring* du fichier **"credit_scoring.csv"**.

1. ***Chargement des données et préparation*** **:** Dans un premier temps nous allons importer le jeu de données et analyser ses caractéristiques.
    - Importer ce jeu de données avec la librairie **pandas** (*c.f.* ***read_csv***)
    - Transformer votre jeu de données issue de **pandas** qui sera de type **Data Frame** en **numpy Array** (*c.f.* ***values***) et séparer ensuite les variables caractéristiques de la variable à prédire (**status**) en deux tableaux différents.
    - Analyser les propriétés de vos données : taille de l'échantillon (*c.f.* ***shape***), nombre d'exemples positifs et négatifs (*c.f.* ***hist***).
    - Pour éviter d'avoir un résultat biaisé du classifieur que nous allons construire, séparer les données en deux partie une dite d'apprentissage qui servira à l'apprentissage du classifieur et l'autre dite de test qui servira à son évaluation (*c.f.* ***train_test_split***).
2. ***Apprentissage et évaluation de modèles :*** Utiliser ensuite sur votre jeu de données les algorithmes d'apprentissage supervisé suivants :
    - Un arbre CART
    - k-plus-proches-voisins avec k=5

L'objectif est à présent de comparer les résultats obtenus à l'aide de ces deux simples algorithmes sur ce jeu de données. Cette comparaison s'appuiera sur l'estimation de l'***accuracy***.

3. ***Normalisation des variables continues*** **:** Certains algorithmes d'apprentissage supervisé fonctionneront mieux si les données sont normalisées (centrées autour de 0) pour que toutes les variables caractéristiques auront le même poids dans la phase d'apprentissage. Utiliser le module ***StandardScaler*** de Scikit-learn pour normaliser vos données. Vous pouvez également tester le module ***MinMaxScaler***. Exécuter à nouveau votre code sur vos données une fois normalisées. Interpréter les résultats obtenus en les comparant avec ceux de la question précédente.

4. ***Création de nouvelles variables caractéristiques par combinaisons linéaires des variables initiales*** **:** Il est parfois utile pour certains classifieurs de faire une réduction de dimensions sur les données afin de déceler et créer certaines combinaisons linéaires dans les variables descriptives et augmenter ainsi le pouvoir discriminant du classifieur. Appliquer une **ACP** (module ***PCA*** de Scikit-learn) sur vos données et garder les *k* premières nouvelles dimensions en les concaténant à vos données normalisées de l'étape précédente.

Exécuter à nouveau votre code sur vos nouvelles données. Que se passe t-il ?