Année universitaire	**2023-2024**		
Filière	**Data Science**	**Année**	**M2**
Matière	**Machine Learning**		
Enseignant	**Haytham Elghazel**		
Intitulé TD/TP :	**Atelier 1 : Détection d'anomalies avec Python**		
Contenu	• **Détection d'anomalies** • **Approches non supervisées : Isolation Forest, LOF** • **Approches supervisées** • **Evaluation** • **Mise en production d'algorithmes de Machine learning**		

Dans cet atelier pratique, vous allez expérimenter des algorithmes de traitement de données pour répondre à différents problèmes liés à la détection d'anomalies 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=10000, suppress = True)
import pandas as pd
import warnings
import matplotlib.pyplot as plt
warnings.filterwarnings('ignore')
```

La détection d'anomalies (dite aussi détection d'outliers ou détection de nouveauté dans certains cas) est une tâche de l'apprentissage automatique qui consiste à déceler dans les données, les instances (individus) ayant un comportement différent (inhabituel) des autres instances de la base dites normales. Dans le cas de la détection des fraudes par exemple, toute dépense très étrange par carte de crédit est suspecte. Les variations possibles sont si nombreuses et les exemples de formation si rares, qu'il n'est pas possible de savoir à quoi ressemble une activité frauduleuse ou un incident. L'approche de la détection des anomalies consiste simplement à apprendre à quoi ressemble l'activité normale (à l'aide d'un historique de transactions supposées non-frauduleuses) et d'identifier tout ce qui est très différent. Différents algorithmes ont été proposées dans la littérature dont certaines sont supervisées et d'autres non supervisées. Pour plus de détails considérant ces approches, vous pouvez vous référer vers les liens suivants : https://ngoix.github.io/nicolas_goix_osi_presentation.pdf et https://en.wikipedia.org/wiki/Anomaly_detection

Sickit-learn propose différentes approches pour la détection d'anomalies (outliers et nouveautés). Nous nous intéressons ici à plusieurs approches (voir le cours sur Moodle) non supervisées et supervisées. Nous allons appliquer cette approche sur trois jeux de données **"mouse.txt"** (https://elki-project.github.io/datasets/) et le fichier **"creditcard.csv"** du challenge Kaggle (https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud) et la **Kddcup99** pour la détection d'intrusions dans les réseaux (accessibles sur sklearn pour la version complète corrigée, ou à partir du fichier moodle pour la version réduite ou sur directement sur kaggle également).

1. Sur la base de données Mouse

Ce fichier contient 500 instances décrites par deux variables x1 et x2 représentant des points de la tête de Mickey Mouse. Les 10 dernières instances du fichier sont aberrantes (*outliers*).

- Télécharger ce jeu de données et analyser le.
- Donner une représentation graphique des données.

- Appliquer la technique Isolation Forest pour détecter les outliers dans ce jeu de données.
- Appliquer la technique Local Outlier Factor pour détecter les outliers dans ce jeu de données.
- Pour chacune des approches, proposer une approche pour mieux choisir le seuil de contamination.
- Modifier votre représentation graphique précédente pour visualiser les données aberrantes avec les deux approches.
- Comparer les résultats obtenus numériquement et visuellement.

2. Sur le jeu de données des cartes de crédits et de détection d'intrusions dans les réseaux

L'objectif dans le début de cette partie est de déceler les fraudes dans les transactions de cartes bancaires. Pour plus d'informations sur ce jeu de données, vous pouvez visiter le lien suivant (https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud).

Ce jeu de données est très déséquilibré avec seulement 0.172% de fraudes (492 fraudes sur 284 807 transactions). Si vous risquez d'avoir des problèmes de performances, n'hésitez à prendre un échantillon aléatoire stratifié de ce jeu de données. Vous pouvez également travailler sur la totalité des données.

a) Préparer ce jeu de données (ne pas utiliser la variable Time).

b) Proposer une méthodologie pour **la détection d'outliers** en comparant plusieurs approches entre elles :

   - Des approches supervisées classiques (deux approches) mais aussi certaines qui gère le déséquilibre dans les données (trois approches).
   - Les deux approches Isolation Forest et Local Outlier Factor.

NB :

Votre méthodologie proposée devrait prendre en compte les points suivants :

- Un protocole de comparaison se basant sur un **échantillonnage aléatoire stratifié** ou sur une **validation croisée stratifiée** (si pas de problème de performances).
- L'évaluation des approches comparées avec des **critères de performances adéquats** (voir le cours sur Moodle).
- La **normalisation des données** si besoin (StandardScaler, MinMaxScaler, RobustSclaer).
- La **prise en compte des données catégorielles** si besoin (OneHotEncoder, OrdinalEncoder).
- L'affichage de la **courbe ROC**.
- Le **choix des meilleurs paramètres** des approches Isolation Forest (nombre d'arbres) et Local Outlier Factor (nombre de voisins les plus proches) ainsi que le choix du meilleur seuil de décision.
- La factorisation de **votre code avec des fonctions** et son automatisation.

c) Appliquer votre méthodologie sur le jeu de données **de détection d'intrusions dans les réseaux** (Kddcup99).

d) Proposer une méthodologie pour la **détection de nouveautés** en comparant les deux approches Isolation Forest et Local Outlier Factor.

Bonus :

Proposer une approche d'explicabilité des résultats retournés par l'Isolation Forest permettant d'analyser les variables les importantes pour la détection d'anomalies (Utilisation de la bibliothèque **SHAPE**).