المعهد الوطني للبريد والمواصلات
ⵓⵙⴷⵉⵡ ⵏⵜⵎⵙⵙⵓⵔⵜ ⴷ ⵜⵜⵉⵍⵉⵎⵓⵔⵜ
Institut National des Postes et Télécommunications

Mini-projets Machine learning
Fraud detection

Réalisé par :

Ayoub HRAZEM
Hamza HASIB
Youssef MOUSSAOUI

Encadré par :

Mr Abdelhak MAHMOUDI

Table des matières

I. Introduction……03
1. Généralités……...…03
2. Problématique……..03
II. Cadre contextuel……..04
1. Criminalité financière…….04
2. Fraude……..05
3. Anomalie…….05
III. Conception…….....06
1. Détection automatique non supervisée des anomalies…...06
a. OC-SVM: One Class Support Vector Machines….06
b. Isolation Forest…….06
c. LOF: Local Outlier Factor……07
d. K-means……07
2. Modélisation…….07
IV. Réalisation……10
1. Environnement de travail……..10
2. Outils de travail…….10
a. Python……...10
b. Jeu de données……..11
3. Processus de réalisation……11
a. Chargement des bibliothèques……11
b. Stockage des résultats……..12
c. Chargement des données et séparation entre les données X et les étiquettes Y……12
d. Algorithmes……..13
e. Résultats……...14

I. Introduction

1. Généralités :

De nos jours, la Data Science est de plus en plus présente dans notre vie. Avec l'émergence de la technologie, beaucoup de taches difficiles ou impossibles à faire manuellement sont maintenant possible grâce au processus d'automatisation.

Dans le domaine bancaire, l'utilisation de la Data Science est plus qu'une tendance, c'est devenu une nécessité pour faire face à la concurrence. Les banques se sont rendu compte que l'intelligence artificielle peut les aider à concentrer efficacement leurs ressources, à prendre des décisions plus judicieuses et à améliorer leur performance.

Naturellement, les banques cherchent de meilleures façons de comprendre les clients et de les fidéliser. Ils examinent les tendances à l'intérieur des données pour comprendre les données transactionnelles et s'engager avec les clients de façon plus significative. Les banques utilisent la Data Science dans les domaines du service à la clientèle, de la détection des fraudes, des prévisions, de la compréhension du sentiment des consommateurs, du profilage des clients et du marketing ciblé, entre autres. Les données dont disposent les banques grâce aux transactions sont nombreuses. Les banques commencent à comprendre l'importance de rassembler et d'utiliser leurs données internes telles que les transactions de débit et de crédit, l'historique et les habitudes d'achat dans la gestion des risques et des fraudes.

Face à cet énorme volume de données qui ne peut être traité manuellement, ou par des manipulations basiques, le Machine Learning et le Deep Learning interviennent afin de faciliter le processus de traitement des données. Ces nouvelles technologies décisionnelles automatisent le processus décisionnel et surtout économisent le coût et le temps de la recherche des informations.

Dans ce mini-projet, nous nous intéressons à la mise en place d'un dispositif de détection de transactions bancaires frauduleuses sur les transactions des clients d'une banque.

2. Problématique :

La fraude à la carte bancaire engendre chaque année, des pertes de plusieurs millions de dollars.

En France, l'Observatoire National de la Délinquance et des Réponses Pénales (ONDRP) a publié une étude, en mai 2018, selon laquelle plus de 1,2 million de ménages français avaient affirmé que leur carte bancaire avait été piratée en 2016.

En 2017, la fraude aux transactions scripturales représente un montant global de 744 millions d'euros pour 5,1 millions de transactions frauduleuses.

SmartMetric, le développeur et fabricant de cartes crédit/débit et cartes de sécurité biométriques, rapporte que, selon des recherches publiées en janvier 2019 par le rapport Nilson, $ 24,26 milliards a

été perdu dans les 12 derniers mois en raison de la fraude par carte de paiement dans le monde entier.
Il est aussi prévu que les pertes dues aux fraudes par carte de paiement ne diminueront pas dans le futur proche. Selon le même rapport, les pertes globales dues aux fraudes de cartes bancaires augmentera de presque $ 10 milliards au cours des 3 prochaines années. Ce qui voudra dire qu'en 2020, le montant global de pertes dépassera $ 35,54 milliards.
Selon SmartMetric, la fraude par carte de paiement s'est élevée à $ 24,26 milliards dans les 12 derniers mois dans le monde entier et devrait continuer à augmenter au cours des 3 prochaines années.
Ce projet a pour but de réaliser un dispositif de détection des fraudes bancaires liées aux comptes courant.

II. Cadre contextuel

1. Criminalité financière :

La criminalité financière est une question fondamentale sur la scène internationale depuis plusieurs décennies. Les autorités cherchent constamment de nouveaux moyens pour traquer et lutter contre la criminalité financière alors que les criminels sont toujours en train de développer de leur côté, des tactiques innovatrices afin de contrer les stratégies anti-crime. La criminalité financière est définie comme la criminalité qui s'engage contre les biens. Ces crimes sont presque toujours au profit personnel du criminel, et qui impliquent une conversion illégale de propriété du bien qui est en cause. Les crimes financiers peuvent se produire sous différentes formes et ils se produisent partout dans le monde. Les crimes les plus communs auxquels fait face le secteur financier sont le blanchiment d'argent, le financement du terrorisme, la fraude, l'évasion fiscale, le détournement de fonds, la falsification, la contrefaçon et l'usurpation d'identité (qui est fortement liée à la fraude). Des tentatives de crimes sont constatées tous les jours, et aussi bien les gouvernements que les banques sont en recherche constante de moyens de lutter contre celles-ci.

3. Fraude :

En droit, la fraude est une tromperie intentionnelle pour garantir des gains malhonnêtes ou illégales, ou pour priver la victime d'un droit légal. La fraude peut être une violation du droit civil (c'est-à-dire, une victime de fraude peut poursuivre l'auteur de la fraude pour éviter la fraude ou pour recouvrer une indemnité pécuniaire), un droit pénal (c'est-à-dire, un auteur de fraude peut être poursuivi et emprisonné par les autorités gouvernementales) Il existe plusieurs types de fraudes financières : Usurpation d'identité : une personne emprunte votre identité et utilise vos• informations personnelles pour voler de l'argent 19 Phishing : les clients des services bancaires en ligne reçoivent des e-mails qui peuvent• être délicats en leur demandant d'accéder à leur compte, en utilisant leur mot de passe & leurs données personnelles sur un site Web qui ressemble au site de leur banque légitime. Vol physique de la carte bancaire : le criminel utilise une carte bancaire volée ou• perdue jusqu'à ce que son propriétaire notifie sa banque pour qu'elle puisse la bloquer. Skimming : il s'agit de voler des informations sur une carte de crédit lors d'une• opération légitime. Les Fraudeurs glissement la carte dans un dispositif électronique appelé « Skimming device » ou « dispositif d'écrémage » qui enregistre toutes les informations figurant sur la bande magnétique. Les fraudeurs utilisent les informations volées pour faire des achats en ligne ou pour reproduire les cartes bancaires. Contrefaçon des cartes bancaires : Les fraudeurs volent les informations de la carte• bancaires pour fabriquer de fausses cartes. La victime de ce genre de fraude s'aperçoit rarement de cette situation puisqu'elle a encore sa carte d'origine en sa possession. Ce sont ces types de fraudes qui seront traités dans ce projet.

3. Anomalie :

Les anomalies ou valeurs aberrantes sont des valeurs extrêmes qui s'écartent d'autres observations sur des données. Elles peuvent être dues à une variabilité dans une mesure, des erreurs expérimentales ou une nouveauté (novelty). En d'autres termes, une valeur aberrante est une observation qui ne se conforme pas à une notion bien définie de _comportement

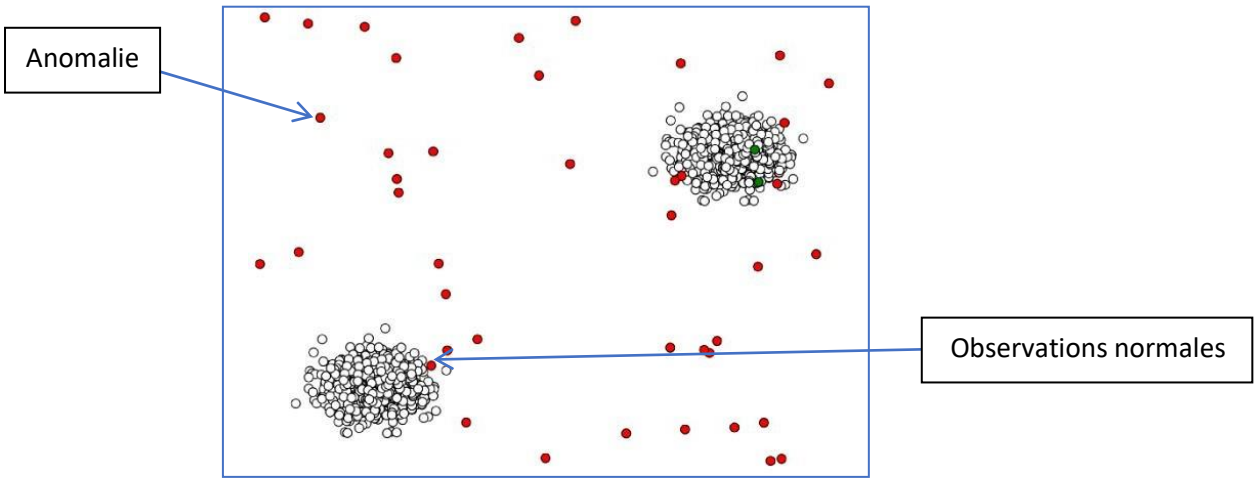

Figure 1 : Représentation des données normales et des anomalies.

III. Conception

1. Détection automatique non supervisée des anomalies

La détection d'anomalies (fraudes) est le processus d'identification des intrusions. Elle repose sur l'hypothèse que le comportement de l'intrus qui génère une anomalie (fraudes) est considérablement différent des comportements normaux ou légitimes.
Les deux modes de détection sont :
La détection supervisée, qui nécessite des données d'entrainement étiquetées.
La détection non supervisée, qu'on utilise dans le cas où les données étiquetées sont difficile à collecter. La difficulté se manifeste aussi pour les algorithmes dans le choix de la métrique du scoring des anomalies ainsi que la définition du seuil selon lequel les données seront classées comme données normales ou anomalies.
Il existe plusieurs approches pour la détection non supervisée d'anomalie :

a. OC-SVM: One Class Support Vector Machines

Cette méthode estime le SV « support vector » d'une distribution en identifiant les régions dans l'espace d'entrée où se trouvent la plupart des cas. Pour ce faire, on projette les données de manière non linéaire dans un espace de caractéristiques et sépare les données de leur origine par une marge aussi large que possible. Tous les points de données se situant en dehors de cette marge sont considérés comme des anomalies.

b. Isolation Forest

L'idée principale, et qu'Isolation Forest identifie explicitement les anomalies au lieu de faire un profilage de points de données normal. Isolation Forest, comme toute autre méthode d'« Ensemble » d'arbre, est construit sur la base des arbres de décision. Dans ces arbres, les partitions sont créées par tout d'abord sélectionner au hasard un élément, puis en sélectionnant une valeur aléatoire de split entre les valeurs minimale et maximale de la fonction sélectionnée. Comme avec d'autres méthodes de détection d'anomalies, un score d'anomalie est nécessaire pour prendre des décisions. Dans le cas d'Isolation Forest, il se définit comme suit :

$$s(x,n) = 2^{-\frac{E(h(x))}{c(n)}}$$

C. LOF: Local Outlier Factor

Dans ce travail, un nouvel algorithme de détection d'anomalies est proposé : LOF Le LOF est l'algorithme le plus connu pour la détection d'anomalie locale et tout d'abord a également introduit l'idée des anomalies locales. Pour calculer le score de LOF, trois étapes doivent être calculées : 1) Les k plus proches voisins doivent être trouvés pour chaque point de données x. En cas d'égalité de distance du k-ième voisin, plus de voisins sont utilisés. 2) La densité locale pour un enregistrement à l'aide de ces k plus proches voisins Nk, est estimée par le calcul de la densité locale d'accessibilité (LRD : the local reachability density). 3) Enfin, le score LOF est calculé en comparant le LRD d'un enregistrement avec les LRDs de ses K voisins. Le score de LOF est donc essentiellement un rapport de la densité locale. Cela se traduit par le fait que, les instances normales, dont la densité est aussi grande que la densité de leurs voisins, obtiennent un score d'environ 1,0. Les anomalies, qui ont une faible densité locale, auront un plus grand score. À ce stade on voit aussi pourquoi cet algorithme est local : il ne repose que sur son voisinage direct, et le score est un ratio principalement basé sur les k voisins seulement. Il est important de noter que, dans les tâches de détection d'anomalie, lorsque les anomalies locales ne sont pas d'intérêt, cet algorithme peut générer beaucoup de fausses alertes.

d. KMeans

K-means est une méthode de clustering utilisée pour la détection automatique des instances de données similaires. K-Means commence en définissant au hasard k centroïdes. A partir de là, des étapes itérations sont exécutées : Affecter chaque point de données au centroïde correspondant le plus proche, en utilisant la distance euclidienne standard. Pour chaque centroïde, calculer la moyenne des valeurs de tous les points lui appartenant. La valeur moyenne devient la nouvelle valeur du centroïde. Ces itérations sont répétées jusqu'à ce que les valeurs des centroïdes ne changent plus. Les valeurs n'appartenant à aucun cluster sont donc considérées non similaires aux valeurs des clusters et donc sont des anomalies. Le résultat du benchmark est le suivant : Les méthodes basées sur l'algorithme K-means sont plutôt rapides Par contre, ils affichent un taux élevé de FP (Faux Positifs)

2. Modélisation :

Afin de pouvoir comparer plusieurs algorithmes de détection non supervisée d'anomalies, nous avons choisi un algorithme de chaque approche, les plus communément utilisées dans le domaine bancaire, Nous allons donc modéliser les 4 algorithmes suivants: OCSVM, IForest, LOF, KMeans:

One-Class Support VectorMachines:

Cet algorithme calcule un hyperplan de séparation de données dans un espace de grande dimension induit par des noyaux exécutant des produits de points entre des points de l'espace d'entrée dans un espace de grande dimension.
Un noyau « kernel » aide à transformer des données linéairement non séparables (comme vu dans la figure input space) des données séparables linéairement (comme vu dans la figure FeatureSpace).

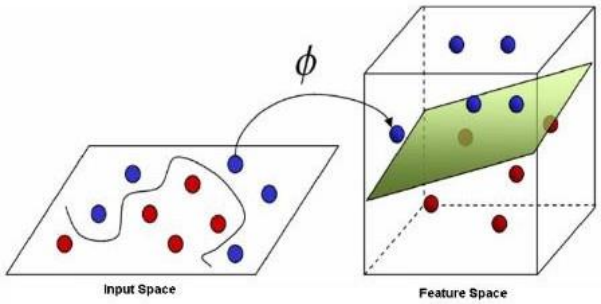

Figure 2 : © 2017, Haydar Ali Ismail, Medium.com Illustration of Support Vector Machine

Ce même hyperplan sépare les données normales, en laissant les anomalies tomber en dehors de la frontière.
Pour utiliser cet algorithme il faut définir la fonction noyau (kernel), ici « Sigmoid » ainsi que son coefficient, gamma = 0,3.

IsolationForest:

Cet algorithme construit une arborescence binaire aléatoire, ensuite choisit un des attributs au hasard, ainsi qu'un seuil de fractionnement, puis il fractionne les données récursivement jusqu'à ce que chaque point de données soit dans sa propre feuille dans l'arbre.
Ce qui est pris en compte, c'est la profondeur d'un point dans l'arbre. L'intuition est que si un point est une valeur aberrante, il suffit de quelques divisions aléatoires pour l'isoler. Donc les données normales sont plus profondes que les données aberrantes

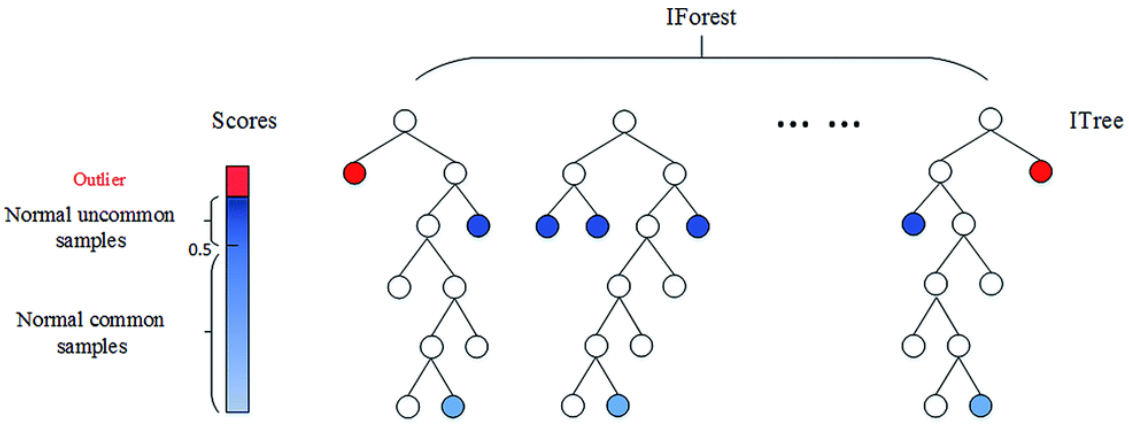

Figure 3 : Isolation Forest pour la détection d'anomalies (Source: Representative subset selection and outlier detection via isolation forest)

Local Outlier Factor:

Cet algorithme mesure la densité locale de chaque « data point » ainsi que celle de ces K voisins. En comparant la densité locale d'un point X avec celles de ses voisins, il identifie les données qui ont des densités beaucoup plus faibles que leurs voisins. Elles sont donc considérées comme des anomalies.

Pour utiliser cet algorithme, il nous faut définir : le nombre k des voisins (k = 35), l'algorithme pour calculer les voisins les plus proches ainsi que lamétrique.

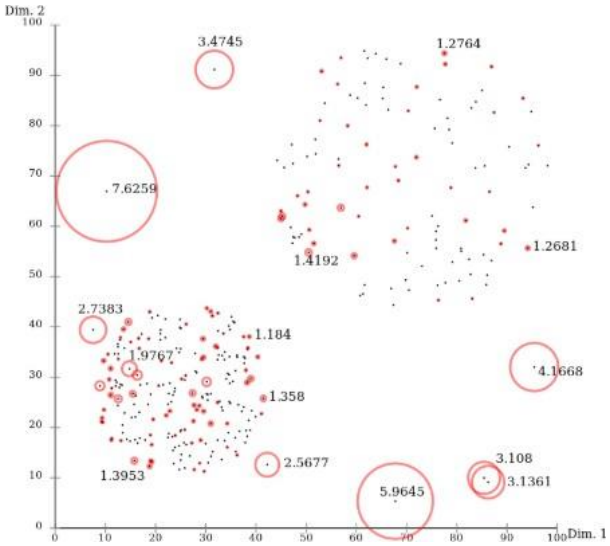

Figure 4 : Visualisation des scores du LOF (Source: https://www.wikiwand.com/en/Local_outlier_factor)

K-means:

Cet algorithme essaie d'affecter chaque ‑data point**||** à un cluster. Les données qui n'appartiennent à aucun de ces clusters sont définie donc comme des anomalies. Pour utiliser cet algorithme, il est nécessaire de définir le nombre initial de clusters. Puisque nous n'avons pas d'informations préalables sur ce nombre, nous allons essayer avec plusieurs nombres pour trouver la meilleure configuration.

IV. Réalisation

1. Environnement de travail :

a. Anaconda :

Une plateforme de gestion de paquet open source, destinée principalement à la science de données. Anaconda regroupe plusieurs bibliothèques antérieurement installées simplifie l'utilisation et la gestion des paquetages.

2. Outils de travail :

b. Python:

Pour la réalisation des modèles nous avons opté pour python comme langage de programmation.

Ce langage de programmation crée par Guido Van Rossem, s'appuie sur plusieurs bibliothèques solides et surtout facilement réutilisables et permet d'écrire des scripts rapidement et facilement. Il est considéré parmi le langage le plus puissant de programmation utilisé par la communauté des Data Scientist.

Vu la simplicité de son usage, la rapidité d'exécution des scripts et la nature des données dont on dispose, Python est plus adéquat pour notre étude.

Python offre un grand panel de bibliothèques traitant différents domaines. Lors de ce premier palier, nous allons avoir besoin de :

-Pandas : Analyse des données
-Scikit-learn : apprentissageautomatique
-NumPy : calculsscientifiques
-Matplotlib : visualisation

c. Jeu de données.

Vu la nature confidentielle des transactions bancaires collectées des différentes entités, et le fait que les jeux de données internes ne soient pas étiquetés, nous allons utiliser pour cette étude des jeux de données (dataset) de détection d'anomalies présentes dans le UCI Machine Learning Repository, ainsi qu'un autre jeu de données « creditcard » qui représentent des transactions réelles anonymisées et publiés pour une utilisation libre.

Les jeux de données utilisés sont tous labélisés, mais les labels ne seront utilisés que dans la phase de l'évaluation, où on appliquera des mesures de comparaisons entre les étiquettes (labels) prédites et les étiquettes (réelles).

Dataset	Taille	Dimension	Contamination
Onecluster	1000	2	2
Annthyroid	7129	21	7.4
Spambase	4207	57	39.9
WDBC	367	30	2.7
Waveform	3443	21	2.9
Credicard	6916	18	4.8

Figure 5 : Jeux de données utilisées dans l'étude

Ces jeux de données sont différents tailles, dimensions, contaminations, distributions ainsi que dans les types de clusters. Ceci est une manière de reproduire la différence entre les caractéristiques des bases de données réelles recueillies par le Groupe Société Générale.

3. Processus de réalisation

a. Chargement des bibliothèques:

```
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt

from sklearn import svm
from sklearn.datasets import make_moons, make_blobs
from sklearn.covariance import EllipticEnvelope
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor
from sklearn.metrics import roc_auc_score

from sklearn.cluster import KMeans, DBSCAN
```

Ce script permet de charger les bibliothèques dont on aura besoin. La bibliothèque SKLEARN permet d'importer les différents algorithmes de détection d'anomalies : SVM,

IsolationForest, LocalOutilerFactor, Kmeans et ensuite la métrique de mesure de performance que nous avons choisie, AUC dans roc_auc_score.

b. Stockage des résultats:

```
25 df_columns = ['Data', '#Samples', '# Dimensions', 'Outlier Perc',
26                'OCSVM','IForest', 'LOF', 'KMeans']
27 roc_df = pd.DataFrame(columns=df_columns)
```

Ce script crée un dataframe pour pouvoir stocker l'ensemble des résultats des différents algorithmes utilisés.

c. Chargement des données et séparation entre les données et les étiquettes Y:

- Chargement des jeux de données labélisés
- séparation des variables (features) X, et des étiquettes Y
- calcul du pourcentage des anomalies pour l'utiliser dans l'évaluation du benchmark
- construction des conteneurs pour l'enregistrement des résultats de l'AUC

```
# Define data file and read X and y
file_list = ['onecluster_1.csv',
             'Annthyroid_1.csv',
             'SpamBase_1.csv',
             'WDBC_1.csv',
             'Waveform_1.csv',
             'creditcard_1.csv']

rng = np.random.RandomState(42)

for csv_file in file_list:
    print("\n... Processing", csv_file, '...')
    data = pd.read_csv(os.path.join(path,csv_file), index_col= 0, header = None)

    X = data.iloc[:, :-1]
    X = X.to_numpy()
    y = data[data.columns[-1]]
    y = y.to_numpy().ravel()

    outliers_fraction = np.count_nonzero(y) / len(y)
    outliers_percentage = 100 - round(outliers_fraction * 100, ndigits=4)

    # construct containers for saving results
    roc_list = [csv_file[:-4], X.shape[0], X.shape[1], outliers_percentage]
```

d. Algorithmes:

Le script suivant nous permet de lister les algorithmes utilisés avec les différents paramètres choisis :

```
# define outlier/anomaly detection methods to be compared
anomaly_algorithms = [
("One-Class SVM", svm.OneClassSVM(kernel="sigmoid",
                                  gamma=0.3)),
("Isolation Forest", IsolationForest(behaviour='old',
                                     random_state=42)),
("Local Outlier Factor", LocalOutlierFactor(
    n_neighbors=35)),
('KMeans',KMeans(n_clusters=2, random_state=0))]
```

avec celles prédites en utilisant l'AUC, et enfin stocker les résultats dans un dataframe

Ensuite, nous allons entrainer ces algorithmes, prédire les classes (enregistrements normaux ou anomalies), comparer les classes réelles

```
for name, algorithm in anomaly_algorithms:
        algorithm.fit(X)
        # fit the data and tag outliers
        if name == "Local Outlier Factor":
            y_pred = algorithm.fit_predict(X)
            y_pred[y_pred == -1] = 0
        else:
            y_pred = algorithm.fit(X).predict(X)
            y_pred[y_pred == -1] = 0
            print(np.unique(y_pred))

        roc = round(roc_auc_score(y, y_pred), ndigits=4)

        print('{name} ROC:{roc}, '
              'execution time: {duration}s'.format(name=name, roc=roc, duration=duration))

        roc_list.append(roc)

    temp_df = pd.DataFrame(roc_list).transpose()
    temp_df.columns = df_columns
    roc_df = pd.concat([roc_df, temp_df], axis=0)
```

e. Résultats :

Les résultats de tous les algorithmes sur chaque jeu de données sont ensuite affichés :

```
print('ROC Performance')
roc_df
```

Data	#Samples	# Dimensions	Outlier Perc	OCSVM	IForest	LOF	KMeans
onecluster_1	1000	2	2	0.4235	0.9592	0.9592	0.5061
Annthyroid_1	7129	21	7.4905	0.5628	0.5563	0.5391	0.5494
SpamBase_1	4207	57	39.9097	0.3477	0.502	0.4772	0.5053
WDBC_1	367	30	2.7248	0.2423	0.8594	0.8594	0.9024
Waveform_1	3443	21	2.9044	0.572	0.5771	0.6286	0.4857
creditcard_1	6916	18	4.8438	0.5808	0.9117	0.6372	0.525