UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Outils Big Data

Projet final

Contexte de la fraude à la carte bancaire

Étudiant :
Antoine QUEVILLART

Enseignant :
M. Bart LAMIROY

Master SEP

28 février 2023

Table des matières

1 Contexte et enjeux de la fraude à la carte bancaire 1

2 Récolte des données 3

3 Stockage et filtre avec MongoDB 4

4 Le Machine Learning au service de la détection de fraude 8

5 Déploiement avec Spark 10

6 Conclusion 12

1 Contexte et enjeux de la fraude à la carte bancaire

La fraude à la carte bancaire concerne l'utilisation malveillante d'une carte de paiement. En effet, l'usage des cartes bancaires a été largement développé en France et dans d'autres pays (En 2018, 67% des Français préféraient payer en carte bancaire selon Statista) et cette part tend à augmenter avec la numérisation de l'acte d'achat. On assiste alors à une hausse des usages frauduleux de carte bancaire pour réaliser des paiements. Il s'agit d'un cas particulier de fraude bancaire contre lequel des organismes comme le C3N (centre de lutte contre les criminalités numériques) de la gendarmerie nationale avec la plateforme *Perceval* mais aussi les banques s'efforcent de lutter, que ce soit pour des raisons légales, financières (rembourser les victimes est très coûteux) ou simplement pour ne pas dégrader leur réputation (les clients souhaitent être protégés des fraudeurs).

Une des différences par rapport aux transactions en monnaie fiduciaire est que la transaction par carte bancaire, qu'elle soit physique ou non, est enregistrée et répertoriée dans une base de données. Cela permet d'archiver toutes les transactions effectuées, parmi lesquelles on retrouve des transactions légitimes mais aussi des fraudes. Les données obtenues lors des opérations de transactions bancaires sont aussi caractérisées par leur récurrence et leur très grand nombre. Selon Statista [1], rien qu'en France, le montant de la fraude à la carte bancaire en France s'élevait à 473 millions d'euros alors que celui-ci n'était que de 269 millions d'euros en 2010. De plus, les flux de données relatifs aux paiements en carte bancaire n'est pas stable, c'est-à-dire que leur distribution dans le temps n'est pas toujours la même. Cela s'explique par des évènements comme les fêtes, périodes de soldes, etc. Donc, si l'on s'intéresse aux transactions effectuées par une carte bancaire, une hausse soudaine du volume de transactions avec cette carte n'est pas nécessairement synonyme d'un cas de fraude.

L'enjeu lorsque l'on souhaite identifier les fraudes à la carte bancaire est de se placer dans la peau d'un fraudeur afin de comprendre les méthodes utilisées pour mieux les contrer. En effet, les fraudeurs peuvent ruser afin de ne pas être détectés par les organismes de contrôle en adoptant des stratégies qui leur permettent de tromper la vigilance des victimes. Certains fraudeurs choisissent d'employer des techniques visant à réaliser un grand nombre de transactions, mais pour des petits montants, ainsi, une victime peu attentive ne remarquerait pas forcément qu'un tiers utilise sa carte bancaire. Aussi, les petits montants peuvent ne pas être suffisants pour déclencher une alerte auprès de la banque et donc ne pas attirer l'attention des algorithmes de détection.

1. Banque d'études statistiques allemande. Etude sur le montant des fraudes à la carte bancaire en France de 2009 à 2020.

Plusieurs techniques de fraude à la carte bancaire existent avec un niveau d'expertise des fraudeurs plus ou moins poussé : elles peuvent être physiques (vol d'une carte bancaire, clonage (ou *skimming*) consistant à reproduire les pistes magnétiques d'une carte) ou numériques (vol des coordonnées bancaires avec le phishing [2]). D'autant plus que la fraude à la carte bancaire peut être un moyen de dissimuler des transactions illicites pouvant bénéficier à d'autres organisations criminelles. On pourrait alors imaginer un montage avec lequel un individu souhaitant effectuer une transaction illégale en ligne utiliserait la carte bancaire d'un tiers afin de ne pas pouvoir être retracé par rapport à ses propres coordonnées bancaires. Les banques se retrouvent alors, involontairement, en position de facilitateur d'organisation de l'activité criminelle. Afin de lutter contre cette utilisation de la banque comme intermédiaire à l'action financière frauduleuse, les organismes bancaires sont tenus de s'assurer de l'identité du client par la présentation d'un document officiel avec photo pour l'ouverture d'un compte conformément à l'article L561-1 Section 3 du Code monétaire et financier [3]. L'enjeu face à cette croissance fulgurante des montants recensés dans les cas de fraude est donc de trouver des techniques permettant de lutter contre la fraude.

La collaboration entre organismes étatiques et organismes privées (banques notamment) fait référence à la notion de *policing*, littéralement « l'action de faire la police ». On retrouve alors l'idée de maintien d'ordre social à travers une mobilisation d'acteurs pour lutter contre la fraude qui ne relève plus seulement de l'ordre des organismes étatiques. Bien que l'intérêt des banques à lutter contre la fraude à la carte bancaire semble être lié à des raisons financières (légalement, les banques doivent rembourser les victimes de fraudes), il faut aussi prendre en compte le fait que leur réputation auprès du public pourrait être entachée par une absence de réponse face aux fraudeurs. Dans ce cas, on peut penser que les clients, en quête de plus de sécurité, opteraient pour les services d'une banque plus soucieuse de la lutte contre la fraude à la carte bancaire. Encore une fois, cela va contre les intérêts économiques de la banque privée qui a besoin de ses clients pour maintenir son activité.

Se pose alors la question des méthodes de lutte contre la fraude à la carte bancaire. S'il existe bien sûr la méthode de détection par l'humain qui consiste pour un expert à passer en revue certaines transactions qui semblent suspectes, d'autres méthodes sont apparues avec le développement de l'intelligence artificielle. La multitude de données permet de mettre en place des méthodes de détection de fraudes automatisées appelées « *data driven fraud detection* ». On entraîne alors un algorithme (algorithme d'apprentissage)

2. Inciter l'utilisateur à transmettre des coordonnées personnelles en se faisant passer pour un tiers de confiance (ex : banque, service public)
3. Article L561-1 Section 3 du Code monétaire et financier

avec des données d'apprentissage et des données de test (avec répartition 80%/20% par exemple) dont on connait les caractéristiques (fraude ou non) afin d'entraîner l'algorithme à détecter de futurs transactions suspectes dont on ignore le statut réel a priori.

2 Récolte des données

Afin d'automatiser la détection de transactions frauduleuses à l'aide du Machine Learning, il faut tout d'abord apprendre à maîtriser les données ainsi qu'à savoir les récolter. Le nombre de transactions étant extrêmement important dans le monde dans un intervalle de temps très petit, il faut prender conscience de la dimension Big Data que représente un tel projet. En effet, le nombre de transactions émanant de différentes sources dont nous parlerons plus tard est très important. Il est par ailleurs nécessaire de traiter ces données quasi-instantanément puisque lors d'un paiement, l'utilisateur d'une carte bancaire ne souhaite pouvoir réaliser une transaction très rapidement. Prenons par exemple le cas d'une vente de place de concert très prisé. Si la communication entre l'organisme de paiement et le vendeur de place est trop longue en raison d'un temps d'analyse de la transaction élevé, alors l'utilisateur pourrait manquer son achat. D'où l'importance de système de traitement des données rapides et performants.

Concernant la source des données, on peut discerner trois point d'origine à une transaction par carte bancaire. Premièrement, nous avons les transactions sur un site e-commerce. Cette source de paiement est parmis les plus risquée puisque la possession de la carte physique n'est pas requise. Seules les informations de paiement sont nécessaire pour réaliser une transaction, cela facilite donc la tâche des fraudeurs puisqu'il leur suffit de mettre en place des stratégie comme le phishing (usurper l'idendité d'une entreprise ou organisation de confiance pour détourner des informations privées à une victime) afin d'utiliser une carte bancaire de façon malveillante. Ensuite, il y a les terminaux de paiement que nous utilisons lors d'achat dans une boutique physique. Cette fois, la possession de la carte physique est nécessaire bien que de nouveaux systèmes de paiement relativement sécurisés comme Apple Pay émergent et permettent de se dispenser de la carte en tant qu'objet. Enfin, il y a le distributeurs bancaire qui permette de réaliser des retraits d'argent liquide ou des dépôts sur le compte bancaire associé à la carte. Cette dernière source de récolte des données semble moins à risque face à la fraude puisque le fraudeur doit d'abord s'emparer de la carte physiquement ainsi que de son code privé.

Se pose alors la question des outils adéquats pour permettre la récolte mais aussi le traitement, le filtrage et la gestion des alertes pour l'organisation de la lutte contre la

fraude à la carte bancaire.

3 Stockage et filtre avec MongoDB

MongoDB est une base de données NoSQL, qui permet de stocker des données semi-structurées ou non structurées. Contrairement aux autres bases de données relationnelles traditionnelles comme Access ou PostGre, MongoDB utilise des collections de documents JSON pour stocker les données. Cette approche permet une grande flexibilité dans la gestion des données, ce qui est particulièrement utile pour stocker des données de capteurs, de log ou de transactions. MongoDB est conçu pour être évolutive horizontalement, c'est-à-dire que les performances peuvent être améliorées en ajoutant de nouveaux serveurs au cluster. Cet outil fournit des fonctionnalités telles que l'indexation, les agrégations, les requêtes complexes, le traitement en temps réel, la réplication et la sécurité intégrée, c'est pourquoi MongoDB est utilisé pour le traitement de données massives. C'est donc un outil très prisé par les entreprises qui cherchent une grande flexibilité et une évolutivité horizontale.

Outre son évolutivité horizontale qui permet d'améliorer les performances et le stockage de bases de données, MongoDB utilise aussi des techniques de réplication pour améliorer la disponibilité des données. La réplication consiste à maintenir plusieurs copies des données dans différents serveurs MongoDB, afin de permettre une réduction des temps d'arrêt et de s'assurer de la disponibilité des données en cas de défaillance d'un ou plusieurs serveurs.

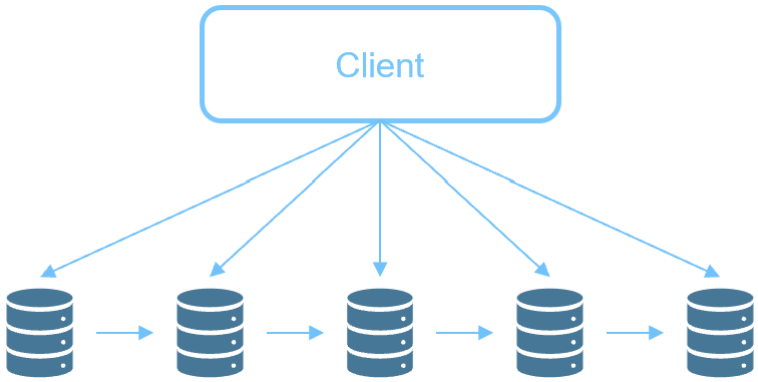

FIGURE 1 – Répartition des données sur différents noeuds

Une des fonctionnalités fondamentales disponible sur MongoDB est le MapReduce. Ce modèle utilise deux principes : la fonction Map et la fonction Reduce. La fonction Map prend une paire clé-valeur en entrée et génère une liste de paires clé-valeur en sortie. La fonction Reduce prend une clé et une liste de valeurs en entrée et agrège ensuite les valeurs pour produire une nouvelle valeur. Le modèle MapReduce est également utilisé dans de nombreux autres systèmes pour le traitement de données volumineuses, tels que Hadoop et Spark. Il est particulièrement utile pour traiter des données structurées et semi-structurées telles que des données de logs, des données de surveillance de la performance, des données de capteurs et des données de transactions bancaires.

On peut également ajouter une partie intermédiaire, le Shuffle durant laquelle les données sont triées. Dans le cas d'une utilisation de MapReduce sur des transactions par carte bancaire afin de faire de la détection de fraude, on pourrait suivre la démarche suivante :

- Map : Les transactions de carte bancaire sont mappées en paires clé-valeur, où la clé est généralement le numéro de carte bancaire et la valeur est le montant de la transaction.
- Shuffle : Les paires clé-valeur sont triées et regroupées par clé. Cela permet de regrouper toutes les transactions associées à une carte bancaire donnée.
- Reduce : Les montants des transactions regroupées sont réduits en une seule valeur, qui représente le montant total dépensé pour chaque carte bancaire.

Par exemple, afin de créer un système d'alerte, on pourrait utiliser la fonction de MapReduce suivante :

```
// fonction map
var mapFunction = function(){
  var amount = this.amount;
  var account = this.account;
  var date = this.date;
  emit({account: account, date: date}, amount);
};

// fonction reduce
var reduceFunction = function(key, values){
  var totalAmount = Array.sum(values);
  if (totalAmount > 10000){
    return{fraud: true, totalAmount: totalAmount};
```

```
  } else {
    return {fraud: false, totalAmount: totalAmount};
  }
};

// fonction MapReduce
db.transactions.mapReduce(
  mapFunction,
  reduceFunction,
  {out: "fraud_detection"}
);
```

On utilise une fonction map pour extraire les champs pertinents de chaque transaction (exemple : montant de la transaction, compte émetteur/receveur, variables temporelles) ainsi qu'une fonction reduce pour calculer le montant total des transactions pour chaque clé (numéro de compte et date). La méthode mapReduce() permet le traitement et le stockage des résultats dans la collection qu'on appelle fraud_detection. Si le montant total dépasse un seuil prédéfini, la transaction est considérée comme frauduleuse et une valeur fraud : true est renvoyée avec le montant total. Sinon, une valeur fraud : false est renvoyée avec le montant total.

Ensuite, une analyse sur les données sorties du Reduce peut permettre de détecter des anomalies dans les dépenses, telles que des transactions inhabituelles ou des schémas de dépenses atypiques.

A l'aide de requêtes MongoDB (voir ci-après pour les exemples), il est donc possible d'effectuer un filtrage des transactions. Avec des critères restrictifs sur plusieurs variables, on peut sélectionner des cas de figure particuliers. Cela peut permettre de fixer des règles de décision sur la nature d'une transaction (fraude ou non) et ainsi enclencher un système d'alerte pour bloquer un paiement si nécessaire. Les règles de décision sont fixées préalablement par l'humain ou à l'aide de techniques de Machine Learning et reposent sur un fondement appelé Complex Event Processing [4]. Il s'agit d'une technique d'analyse en temps réel de données en continu pour détecter des événements complexes ou des motifs de comportement dans les données en temps réel. Cette technique est souvent utilisé afin de surveiller des flux de données en temps réel provenant de plusieurs sources et détecter des modèles de comportement inhabituels ou des événements qui nécessitent une réponse immédiate. On peut par exemple chercher à effectue une agrégation sur les transactions

4. Complex Event Processing with T-REX, Gianpaolo Cugola, Alessandro Margara, *Dip. di Elettronica e Informazione Politecnico di Milano, Italy*, 2012

par compte, puis à compter le nombre de transactions et la somme des montants, et enfin retourner uniquement les comptes ayant effectué plus de 10 transactions pour un montant total supérieur à 50 000 à l'aide de la requête suivante :

```
db.transactions.aggregate([
    {
        $group:{
            _id:"$account_id",
            count:{$sum:1},
            total:{$sum:"$amount"}
        }
    },
    {
        $match:{
            count:{$gt:10},
            total:{$gte:50000}
        }
    }
])
```

Pour obtenir les transactions qui ont un montant supérieur à 10 000, qui représentent plus de 80% du solde du compte, ou qui ont un montant supérieur à 5 000 en dessous du solde du compte on utilise la requête ci-dessous :

```
db.transactions.find({
    $or:[
        {amount:{$gte:10000}},
        {$expr:{$gt:["$amount",{$multiply:["$balance",0.8]}]}},
        {$expr:{$gt:["$amount",{$subtract:["$balance",5000]}]}}
    ]
})
```

On peut également chercher les transactions effectuées en janvier 2022 qui ont un montant supérieur à 50% du solde du compte ou un montant supérieur à 10 000 en dessous du solde du compte avec cette requête :

```
db.transactions.find({
    timestamp:{
```

```
            $gte:ISODate("2022-01-01T00:00:00.000Z"),
            $lt:ISODate("2022-02-01T00:00:00.000Z")
        },
        $or:[
            {$expr:{$gt:["$amount",{$multiply:["$balance",0.5]}]}},
            {$expr:{$gt:["$amount",{$subtract:["$balance", 10000]}]}}
        ]
})
```

La présentation de ces requêtes MongoDB non-éxhaustive pourrait donc représenter un moyen d'établir des critères de blocages de transactions. Cependant, cette méthode repose sur l'élaboration de règles de décision fixées au préalable par l'humain. Il existe des techniques de Machine Learning permettant elles aussi de détecter des transactions frauduleuse.

4 Le Machine Learning au service de la détection de fraude

Dans le cas de la détection des fraudes, le problème posé suit une logique binaire. Il s'agit de prédire à l'aide de variables explicatives si une transaction donnée est une fraude ($Y = 1$) ou si c'est une transaction légitime ($Y = 0$). Pour identifier la nature d'un événement, plusieurs modèles de familles différentes peuvent être appliqués. Ici nous allons nous intéresser au modèle « logit », ou régression logistique qui est un modèle de classification. Dans notre cas, il s'agit plus spécifiquement d'un modèle de classification binaire (ou binomiale).

Le modèle de régression logistique repose sur le principe de probabilité conditionnelle (théorème de Bayes), on rappelle que cela consiste à trouver $\mathbb{P}(Y = y_k|X) = \frac{\mathbb{P}(Y=y_k)\times\mathbb{P}(X|Y=y_k)}{\mathbb{P}(X)}$. La fonction logistique renvoie une valeur comprise entre 0 et 1, donc cela revient à probabiliser la nature d'un événement (typiquement une transaction).

Lorsque l'on a seulement deux classes (typiquement fraude / pas fraude), on a [5] :

$$\frac{\mathbb{P}(Y=+|X)}{\mathbb{P}(Y=-|X)} = \frac{\mathbb{P}(Y=+)}{\mathbb{P}(Y=-)} \times \frac{\mathbb{P}(X|Y=+)}{\mathbb{P}(X|Y=-)}$$

Cela revient à identifier la probabilité que la variable cible Y prenne une des deux

5. Les notations utilisées dans cette sous-partie sont tirées du cours sur la régression logistique binaire publié par Ricco Rakotomalala sur le site web http://eric.univ-lyon2.fr/

valeurs en fonction des variables explicatives qui la caractérise.

Lorsque l'on souhaite établir un outil de détection des fraudes à la carte bancaire (et pour d'autres types de fraude), nous devons faire face à des contraintes en termes de répartition des données.

En conservant ce déséquilibre pour l'application du modèle, on risque de ne pas obtenir des résultats satisfaisants sur des données réelles car l'algorithme n'aura pas suffisant appris d'au moins une des classes (celles qui compte le moins d'individus). L'enjeu est alors de trouver une méthode de rééchantillonnage capable de corriger le déséquilibre pour ensuite permettre au modèle de s'entraîner sur des données de différentes classes suffisamment nombreuses (et donc également réparties). Pour cela, il existe plusieurs techniques dites « naïves » afin de réajuster les ratios des différentes classes d'individus. Nous allons pour cela nous intéresser à une méthode proposée par Chawla et al. (2002) [6] en 2002 : la méthode SMOTE (Synthetic Minority Over-sampling Technique).

La méthode SMOTE permet de réduire l'écart entre les deux classes en générant un échantillon de classe minoritaire grâce à une sélection aléatoire de k-voisins proche d'une observation de la classe minoritaire, puis en prenant une valeur située entre l'observation et un des k-voisins. L'échantillon obtenu est alors un échantillon synthétique et non une copie de l'échantillon réel comme dans le cas du sur-échantillonnage. La méthode permet de créer des individus fictifs ayant des caractéristiques similaires aux vrais individus, sans pour autant être totalement identique.

Pour générer un nouvel échantillon, on choisit d'abord un vecteur qui caractérise notre classe minoritaire que l'on note vm. Ensuite, on détermine les k-voisins les plus proches (avec par exemple $k = 10$). On sélectionne ensuite un de ces points aléatoirement que l'on note kv. Pour chaque valeur caractéristique i, on calcule la différence pour chacune des valeurs caractéristiques tel que :

$$vm_i - kv_i$$

Puis, on multiplie le résultat par un nombre aléatoire dans [0 ;1] On ajoute ensuite la nouvelle valeur obtenue correspondant à la valeur caractéristique i de vm et on recommence l'opération. Une des limites de cette technique est qu'elle ne prend pas en compte les labels de k-voisins. En effet, puisque la méthode SMOTE consiste à créer de nouveaux individus synthétiques à partir d'un point donné (qui lui-même peut être synthétique), il se peut qu'une multitude d'individus soient créés avec des caractéristiques propres aux

6. Chawla et al. *Smote : Synthetic minority over-sampling technique*. 2002. [Journal of Artificial Intelligence Research]

individus de l'autre classe, sans que le modèle ne l'interprète comme tel. En effet, si sur le plan un des k-voisins appartenant à la classe minoritaire se trouve entouré d'individus de la classe majoritaire, celui-ci aura tendance à estomper la segmentation entre les deux classes. Cela peut entraîner des confusions entre les individus de la classe majoritaire et ceux de la classe minoritaire.

Une fois que le modèle a été entraîné et que ces performances sont satisfaisantes. Nous pouvons passer à la phase de déploiement de celui-ci.

5 Déploiement avec Spark

Afin de permettre l'utilisation d'un modèle de ML par divers outils, il est préferable de convertir le format. Par exemple, un modèle crée avec Python pourra subir une transformation vers un format PMML (Predictive Model Markup Language). Il s'agit en fait de traduire les paramètres et les caractéristiques du modèle, qui sont généralement stockés dans un format spécifique à l'outil de machine learning utilisé, dans le format standard PMML. Une fois que le modèle est converti en PMML, il peut être utilisé par d'autres outils de traitement de données tel que Spark.

Afin d'intégrer le modèle dans une application Spark. Il est nécessaire de faire quelques ajustements. Par exemple, écrire du code Spark qui charge les nouvelles transactions bancaires depuis une source de données, applique le modèle pour prédire les étiquettes de fraude et enfin stocke les résultats dans une base de données.

On peut trouver plusieurs avantages à l'intégration du modèle de détection de transactions frauduleuses dans une application Spark. Tout d'abord Spark offre des grandes performances en utilisant la mémoire vive pour stocker les données intermédiaires, ce qui permet d'accélérer les traitements et d'améliorer les performances. De plus, Spark permet le raitement en temps réel, c'est essentiel pour la détection de fraudes où il est important d'agir rapidement. Spark est aussi conçu pour être scalable, c'est-à-dire qu'il peut gérer de grandes quantités de données et de nombreuses sources de données simultanément. Cet outil est également accessible à travers plusieurs langages dont Python. Par exemple, si nous avons un modèle de régression logistique pré-entraîné pour la détection de fraude, on peut suivre la démarche suivante pour transformer le modèle en modèle Spark ML :

```
# modele de base avec Scikit-learn
reglog = LogisticRegression(max_iter=1000)

# package mleap pour la conversion
import mleap

# chemin du modele enregistre
chemin_model = "../reglog.pkl"

# sauvegarde du modele en format compatible Spark
save_to_bundle(reglog, chemin_model)
```

Pour charger le modèle dans une session Spark, on utilisera la démarche suivante :

```
# package mleap pour le chargement
from mleap.spark import load_spark_bundle

# creation de la session Spark
spark = SparkSession.builder.appName("load_model").getOrCreate()

# recuperation du chemin
chemin_bundle = "../bundle.zip"

# affecation du modele
model = load_spark_bundle(spark, chemin_bundle)
```

L'utilisation de Spark pour le déploiement de modèle est donc intéressante en ce sens qu'il est possible de convertir les modèles dans des formats compatibles avec d'autres outils facilement, par exemple Hadoop.

6 Conclusion

La lutte contre la fraude à la carte bancaire est un enjeu majeur pour les consommateurs et les entreprises. Les outils Big Data tels que MongoDB et Spark, ont considérablement amélioré la capacité à détecter et prévenir les fraudes à grande échelle. En utilisant des techniques telles que le MapReduce, les entreprises et les banques sont en mesure d'analyser de vastes quantités de données de transaction en temps réel pour détecter les fraudes et prendre des mesures immédiates pour protéger leurs clients et leurs intérêts. On peut s'attendre à des avancées supplémentaires dans la lutte contre la fraude à la carte bancaire grâce aux améliorations continues de l'intelligence artificielle et du traitement du langage naturel (NLP). La combinaison de la prévention de la fraude en sensibilisant les utilisateurs de carte bancaire et de l'utilisation de technologies telles que MongoDB et Spark est essentielle pour protéger les consommateurs et les entreprises contre les pertes financières et les atteintes à la sécurité.