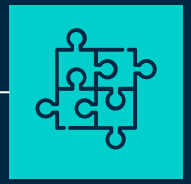
Détection de fraude avec Python

Nasseha Said-Salim
Ariinui Teriitehau
Master 2 Économétrie, Statistiques

MICROPOLE
Driving Distinction

IAE NANTES
ÉCONOMIE & MANAGEMENT

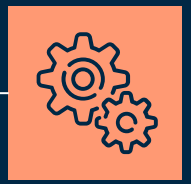
Sommaire



01

Introduction

- Benchmark
- Méthodologie



02

Exploration de données

- Analyse de données

03

Classification

Machine Learning

04

Conclusion

Introduction

Le but du projet est de prédire par probabilité si **une opération de paiement donnée est frauduleuse ou non**. L'ensemble de données provient des transactions réalisées par téléphone sur le continent africain.

Il est nécessaire de trouver une solution pour détecter la fraude selon les caractéristiques des données **de la transaction à temps réels.** Les transactions sont acceptées ou refusées pour différents motifs : ***dépassement de plafond, insolvabilité ou présomption de fraude.***

Les organismes bancaires se doivent de détecter les fraudes pour éviter **des pertes financières trop importantes, garantir la sécurité et la fiabilité des transactions des clients**.

Le projet est divisé en 4 grandes sections et nous utilisons le logiciel Python.

Veille documentaire → Exploration des données → Apprentissage → Conclusion

Veille documentaire et recherche: kaggle

Pour le projet, nous nous sommes aidé des discussions et notebooks sur le site **Kaggle**, avec une compétition traitant également la détection de fraude pour l'entreprise Vesta (lien). Ceci nous a beaucoup apporté pour la construction de nos graphiques, l'analyse de données ,et le type de machine learning utilisé dans la détection de fraude.

Dans un rapport publié en 2018 dans IEEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS, lien), les auteurs trouvent que les modèles **Random Forest et Gradient Boosting** permettent de mieux classifier.

Dans le projet, nous proposons d'utiliser deux méthodes d'apprentissage disponible sur Python par le biais de *scikit-learn*, comme **Decision Tree** et **Gradient Boosting**.

scikit learn XGBoost

Présentation du jeu de données

Base de données

Les données contiennent les transactions en devise locale et des informations des clients émetteurs et destinataires aux **39 unités de temps** *(step).*

Ces données sont divisées en deux bases distinctes : la base **train** et la base **test**, qui sont de taille respective de 500k et 250k observations. La *base* train est composée des step entre 1 et 20 et la *base test* entre 21 et 39. La cible à prédire est isFraud qui est codée à 1 si la transaction est une fraude et 0 sinon. Notons aussi qu'il n'y **a aucune valeur manquante** dans train et test.

Tableau 1 : Description des variables

Nom variables	Description
nameOrig	client qui lance la transaction
nameDest	Client destinataire qui reçoit la transaction
Variables catégorielles	
isFraud	Fraude, variable à prédire prenant 1 si fraude, 0 sinon
type	type de transaction, CASH-IN, CASH-OUT, DEBIT, PAYMENT et TRANSFER
Variables quantitatives (numérique)	
step	unité de temps. Données incrémentale qui permet de positionner dans le temps les transactions.
amount	montant de la transaction de la devise locale
oldbalanceOrg	Solde du client avant transaction
newbalanceOrig	Solde du client après transaction
oldbalanceDest	Solde du destinataire avant transaction.
newbalanceDest	Solde du destinataire après transaction.

Exploration de données

MICROPOLE
Driving Distinction

IAE NANTES
ÉCONOMIE & MANAGEMENT

Exploration de données

Variable cible : isFraud

Les transactions de fraude représentent moins de 0.1% dans *train* et *test*. Nous travaillons sur des données **déséquilibrées** entre la modalité 1 et 0 (tableau 2).

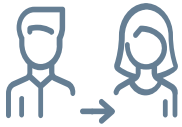
Tableau 2 : modalité de fraude

Echantillon d'apprentissage (Train)

Modalité	Effectif	%
0	499 767	99,95%
1	233	0,05%
Total	500 000	100,00%

Echantillon Test

Modalité	Effectif	%
0	249 779	99,91%
1	221	0,09%
Total	250 000	100,00%



Variable Dest :

Une nouvelle variable catégorielle '**dest**' est crée par l'intermédiaire nameDest. Elle prend la valeur M si le destinataire est un commerçant ou C sinon. Dans la base train, nous retrouvons **335k C** et **164k M**. D'après le tableau 3, la fraude est réalisée **uniquement** pour **la modalité de *Dest* 'C'** dans *train,* de même pour *test.*

Tableau 3 : Croisement entre isFraud - dest

	train		test	
Modalité	**No Fraude**	**Fraude**	**No Fraude**	**Fraude**
C	335735	233	164584	221
M	164032	0	85195	0

Analyse de données

Variable transaction, amount

En **moyenne**, le montant d'une transaction est de **166 393**, avec un **minimum** de transaction de **0,01**. (cf tableau 4 annexe 22)

Nous illustrons figure 1, le montant de transaction de la variable amount en fonction de la fraude. Au niveau du premier graphique, les **lois de distributions diffèrent** en fonction d'une **fraude ou non**. **La queue de distribution** de la courbe de densité dans le cas où il n y a **pas de fraude est plus longue à gauche**.

Les lois de distribution du montant en fonction de la fraude ne suivent pas la loi normale.

Figure 1 : Distribution log(amout) par isFraud

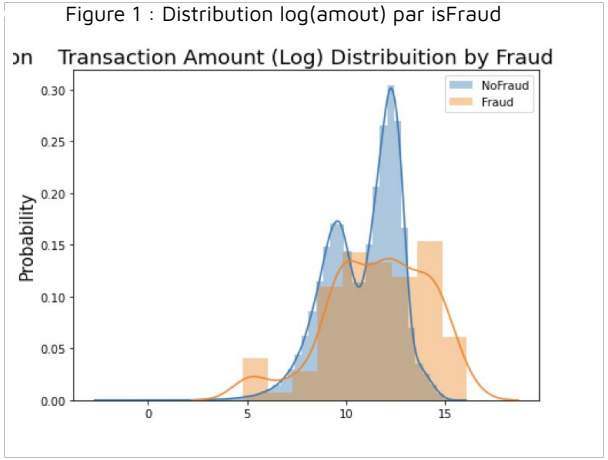
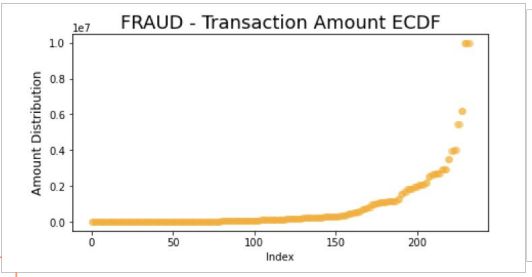
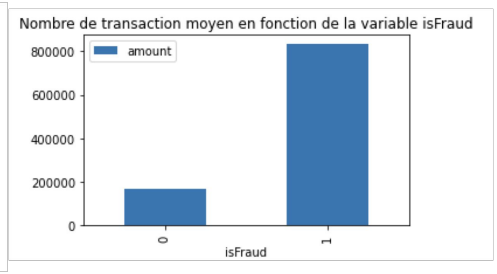


Figure 2 : Distribution isFraud-amount

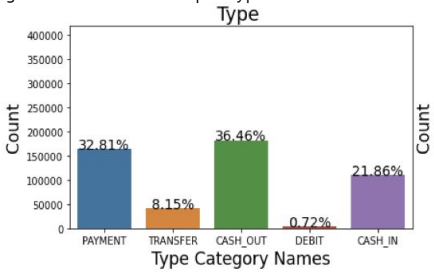




Nous avons une distribution du montant de la transaction de fraude qui a **une tendance croissante**, **plus le montant est conséquent, plus la probabilité de fraude est élevée** (figure 2).

Analyse de données

Figure 3.A : distribution par type de transaction



Variable Type :

Le premier graphique indique le nombre et le pourcentage de transaction pour chaque type. La majorité des transactions sont réalisées en CASH_OUT (36.46%) et PAYMENT (32.81%).

Figure 3.B : Distribution par type et Isfraud

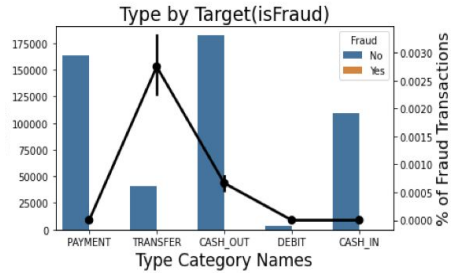


Figure 3.B, nous représentons le groupement par isFraud et type, la fraude se manifeste uniquement pour **TRANSFERT** (0.003%), suivi de **CASH_OUT** (0.0005%).

Figure 3.C : Boxplot type – amount – isFraud

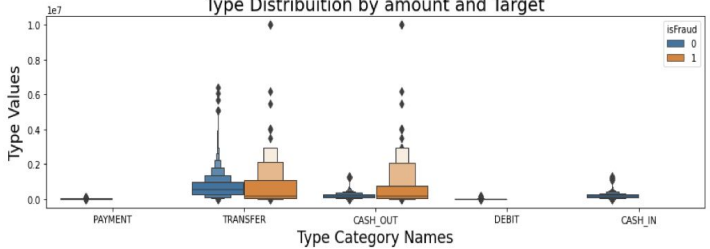


Figure 3.C, nous montrent clairement les valeurs aberrantes pour la fraude ayant pour type **TRANSFERT** et **CASH_OUT** avec un **amount** maximum de 10 millions. Le tableau 5 (annexe 18) montre les 4 valeurs aberrantes du même montant.

Analyse de données

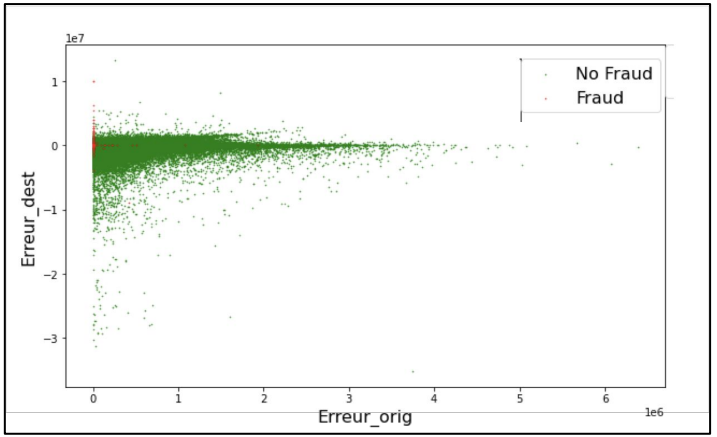
Les variables Erreur_orig et Erreur_dest

Lors de l'exploration des données, nous constatons d'abord que dans les transactions frauduleuses, le montant de la transaction et le solde du client avant transaction sont en moyenne similaires. (figure 4.A annexe 23)

De nos recherches kaggle (lien), certains trouvent que le nouveau solde de l'émetteur (newbalanceOrig) est à 0, même après **une transaction pour 98% des fraudes**. Ce qui mène à une forte suspicion sur chaque transaction, en supposant que les soldes nuls servent à différencier les fraudes des non fraudes.

A priori lorsqu'il n y a pas de fraudes, l'erreur est nulle. L'idée est de créer de nouvelles variables Erreur_orig et Erreur_dest qui tient les erreurs dans le compte d'origine et de destination.

Figure 4.B : représentation graphique Erreur_dest et Erreur_orig/ isFraud



Calcul des variables :

Erreur_orig = newbalanceOrig – oldbalanceOrig + amount

Erreur_Dest = oldbalanceDest - newbalanceDest + amount

Erreur_orig est la différence des soldes avant et après la transaction de l'émetteur (origine) en ajoutant le montant de la transaction.

Analyse de données

Variables step

Figure 5.A : Distribution step + fraud

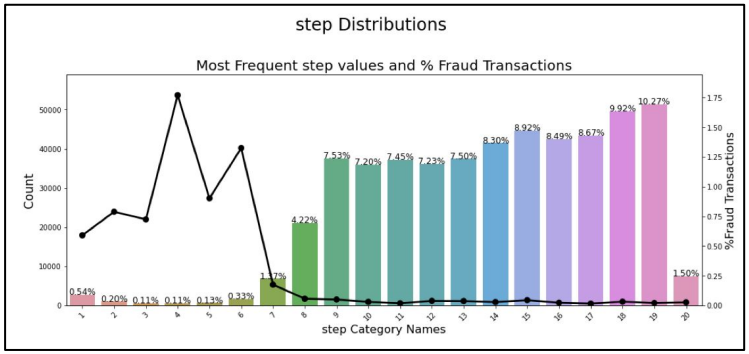


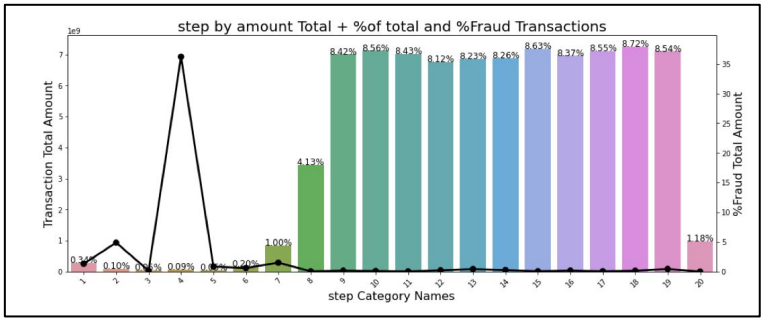
Figure 5.A et B sont basées sur step concernant l'unité de temps de l'échantillon train. Il illustre la distribution de step selon la fréquence de transaction et le taux de fraude dans le graphique du haut, puis, nous ajoutons le montant (amount) de transaction total dans le graphique du bas.

Ainsi, nous remarquons que **le nombre de transaction est faible dans l'unité de temps** (step) entre 1 et 6. À contrario, le **nombre de transaction augmente** et reste élevée entre step 7 à 20.

Step semble être des transactions sur **chaque heure**, par exemple, la tranche entre 1 et 7 peut être des heures entre 1H00 et 7H00 expliquant le nombre de transaction faible dans ce créneau horaire.

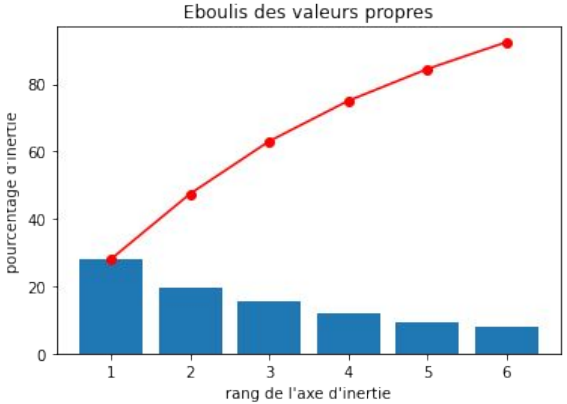
Alors qu'entre step 9 et 19, cela peut correspondre à un créneau horaire **entre 9H00 et 19H00,** en pleine activité dans la journée et début de soirée. Pour base test (figure.5.C annexe 23), nous constatons une diminution des transactions entre step 21 et 32, qui pourrait correspondre aux **horaires entre 21H00 et 8H00 pour jour +1.**

Figure 5.B : Distribution step + fraud + amount total



Analyse en composantes principale

Figure 6 : Valeurs propres



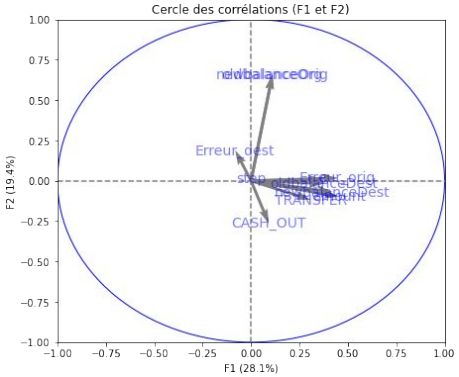
Au niveau du graphique 6, nous illustrons la part d'inertie de chacune des composantes, où nous voyons que les deux axes les plus importants sont les deux premiers.

Chacune de nos variables contribue à la définition d'un axe en particulier. L'utilité de l'analyse en composante principale est de regrouper les variables contribuant au même axe, de sorte à les redéfinir en une autre variable.

Nous regroupons ensuite l'ensemble de nos variables explicatives dans un cercle de corrélation (figure 6a). La part de l'inertie expliquée par les deux axes est de 47.5%.

Certaines variables s'opposent comme newbalanceOrig et Err_dest, d'autres au contraire se rejoignent, Err_orig, oldbalanceDest.
Dans la suite du projet nous n'utilisons pas le regroupement des variables par axe

Figure 6a : Cercle de corrélation



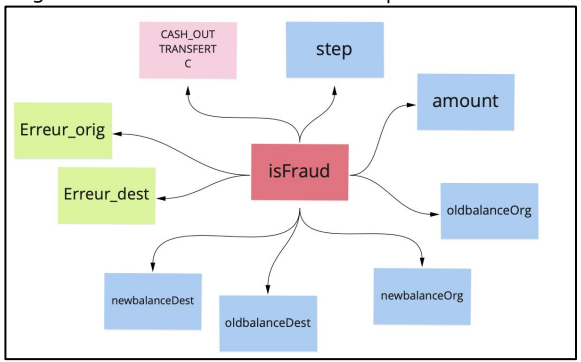
Machine Learning

MICROPOLE
Driving Distinction

IAE NANTES
ÉCONOMIE & MANAGEMENT

Méthode d'apprentissage

Figure 7: Schéma des variables explicatives



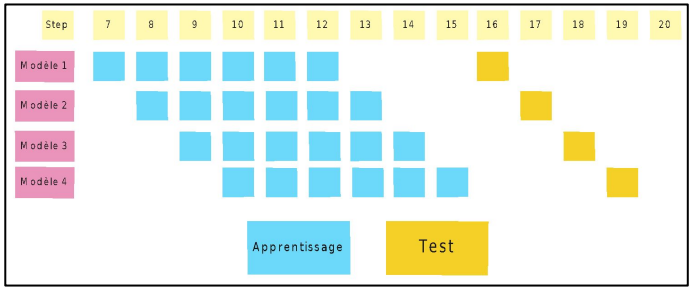
Remarque : l'ensemble des variables explicatives et la cible

Apprentissage des données

Pour l'apprentissage de nos modèles, nous procédons en deux temps. Premièrement, nous réalisons des **prédictions** sur la base train et test. Dans un second temps, nous modélisons la détection de fraude par trois hypothèses : 3 step de délai de remontée de la fraude, 6 step d'historique pour prédire la fraude ,et périodicité du rafraîchissement du modèle par step (voir figure 7a).

Nous utilisons 2 algorithmes : **Xgboost et Arbre de décision (Decision Tree).**

Figure 7a: Schéma des variables



Source : Détection de fraude

Évaluation métrique

Dans notre projet, nous décidons d'utiliser de nombreux indicateurs de performances dans le choix du meilleur modèle pour la prédiction frauduleuse avec choix du seuil de **0.001 pour AUC**.

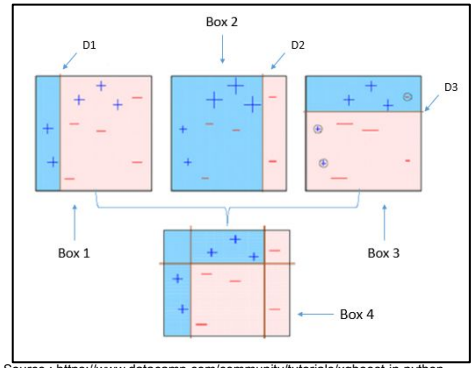
Machine Learning

XGBoost

C’est une implémentation du Gradient boosting, où ce dernier consiste à établir un premier modèle, où **chaque individu est associé à un poids**, et d’attribuer les poids les plus importants aux individus mal classés pour améliorer et corriger le modèle suivant (figure 8).

Il combine des techniques d’optimisation pour donner de meilleures performances, en un meilleur délais.

Figure 8 : Schéma XGBoost



Source : https://www.datacamp.com/community/tutorials/xgboost-in-python

Decision Tree

C’est une représentation des données des séquences de décisions en vue de prédire une classe. Chaque observation est dans une classe et est décrite par **un ensemble de variables testé dans chaque nœud.**

L'algorithme va devoir étudier toutes les décisions qu'il est possible de prendre et les comparer entre elles.

Dans notre cas, nous avons un arbre de classification, où dans les paramètres **l’indice de GINI** mesure l’erreur de classification. Dès qu’une branche d’arbre est ajoutée, l’indice est recalculé.

Résultats et interprétation

Indicateurs de performances :

Les prédictions de classifications sont de très bonne qualité pour l'ensemble des indicateurs sur la base test (tableau 6). Nous n'avons pas de problèmes de sur et sous apprentissage (tableau 7, annexe 24).

Matthews cor. indique une valeur de 0.98 et 0.97 respectivement pour Xgboost et Decision Tree, représentant une prédiction quasi-parfaite.

Kappa de Cohen permet de mesurer l'accord du codage des fraudes en binaire. L'indicateur indique une forte valeur proche de 1 pour les deux modèles, donc, nous avons un accord quasi-parfaite sur le codage de la fraude.

Figure 9 : Matrice de confusion

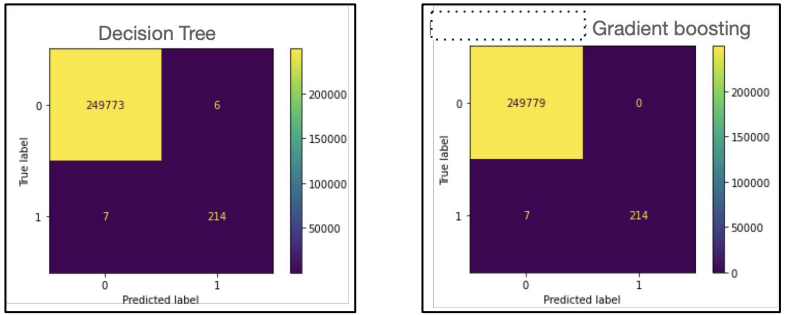


Tableau 6: Indices de performances sur la base test

Indicateur de performance	Xgboost	Decision Tree
F1-score	0,9919	0,9852
PR-AUC	0,973	0,971
Accuracy	0.999	0,9999
Recall	0,9683	0,9683
Precision	1,0000	0,9727
Cohen K	0,9839	0,9705
Matthews Corr	0,9840	0,9705
Jouden	0.9683	0.9683

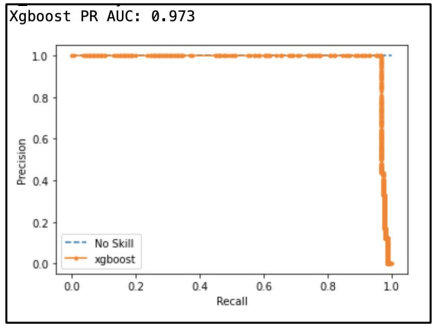
Le choix du meilleur modèle va se faire sur l'**indicateur F1-score** qui combine le **recall** et la **précision**, étant donné que cet indicateur est insensible au déséquilibre de classe dans le cas du projet.

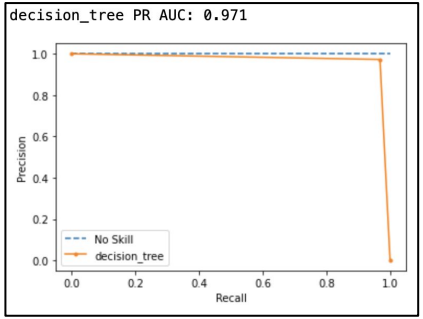
Decision Tree a **F1-score la plus faible avec 0.9852**, sa matrice de confusion présente 13 observations mal classées, dont **7 comme n' étant pas une fraude et 6 comme une fraude** (figure 9).

Xgboost a **F1-score proche de 1**, la matrice de confusion **montre 7 observations mal classées dans la non fraude** (figure 9).

Résultats et interprétation

Figure 10.A : Courbe AUC-PR





Classification déséquilibrée :

Le rapport disproportionné d'observations dans chaque classe est un problème courant dans la classification. Nous utilisons la métrique AUC-PR, étant plus adaptée dans cette situation de détection de fraude par rapport à la AUC-ROC (figure 10.B, annexe 24). La valeur de AUC-PR de Xgboost est légèrement (0.973) plus élevé que Decision tree (0.971).

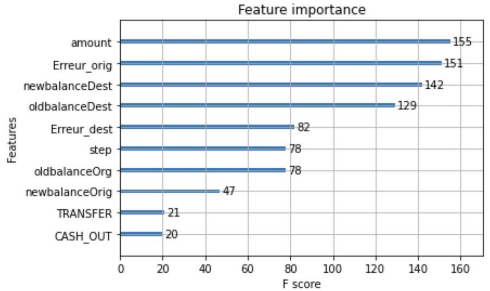
Dans la figure 10, nous avons pratiquement des courbes droites pour Xgboost. Et Decision Tree ne permet de calculer des probabilités de prédiction, d'où, le fait que courbes toutes droites.

Finalement, Xgboost est le modèle le plus performant selon l'ensemble des indicateurs.

Evaluation de l'importance des variables :

Dans Xgboost, les variables l'importance des variables diffèrent l'une de l'autre. Amount, Error_orig, newbalanceDest et oldbalanceDest sont les quatre variables contribuant au mieux à la prédiction du modèle. Nous avons uniquement la variable 'destC' qui n'a pas été retenu (figure 11).

Figure 11 : Importance des variables



Résultats et interprétation

Caractéristiques des prédictions de fraude par XGboost :

Le tableau 8 montre les 7 fraudes mal classées, les transactions sont toutes de type CASH_OUT variant entre 89k et 577k amount. Erreur_dest contient des valeurs nulles et Erreur_orig est équivalent à amount (à l'exception d'une transaction). OldbalanceDest prend aussi de grandes valeurs et newbalanceDest représente bien la somme de oldbalanceDest et amount.

Dans le tableau 9, nous avons trié les observations par prédiction de probabilités avec Xgboost, nous constatons que Erreur_orig est nulle pour les 10 premières observations. Il y a une concordance entre le type de TRANSFERT et l'Erreur_dest prenant d'importantes valeurs d'ordre 10e+06. Et pour CASH_OUT, Erreur_dest est nulle.

Remarque :
D'une part, les transaction frauduleuse pour **les observations mal classées sont réalisées en interne de la banque** car nous connaissons l'information du solde du destinataire.

D'autre part, le fait d'avoir oldbalanceDest et newbalanceDest comme valeur nulle suppose que la transaction a été réalisée vers une **banque différente** de l'original, car nous n'avons aucune information sur le solde du destinataire. Les fraudeurs profitent d'obtenir l'argent **pour réaliser des transferts vers des banques différentes**.

Tableau 8 : 7 observations mal classées de Gradient boosting

	step	type	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	Erreur_orig	Erreur_dest
10820	21	CASH_OUT	23292.30	0.0	0.0	392364.62	415656.92	1	23292.30	0.00
26871	22	CASH_OUT	89571.46	4505.6	0.0	1929428.01	2018999.47	1	85065.86	0.00
110563	35	CASH_OUT	112280.88	0.0	0.0	40512.49	152793.36	1	112280.88	0.01
161443	36	CASH_OUT	234377.29	0.0	0.0	34937.86	269315.15	1	234377.29	0.00
199783	37	CASH_OUT	112486.46	0.0	0.0	257274.47	369760.93	1	112486.46	0.00
217647	38	CASH_OUT	577418.98	0.0	0.0	0.00	577418.98	1	577418.98	0.00
231079	38	CASH_OUT	407005.78	0.0	0.0	0.00	407005.78	1	407005.78	0.00

Tableau 9: 10 premiers observations les mieux classées en terme de probabilités

	step	type	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	dest	Erreur_orig	Erreur_dest
133919	36	CASH_OUT	2352524.18	2352524.18	0.0	1227902.12	3580426.31	1	C	0.0	-1.000000e-02
43226	28	TRANSFER	3832058.34	3832058.34	0.0	0.00	0.00	1	C	0.0	3.832058e+06
43256	32	TRANSFER	2023920.09	2023920.09	0.0	0.00	0.00	1	C	0.0	2.023920e+06
22681	22	TRANSFER	2068118.36	2068118.36	0.0	0.00	0.00	1	C	0.0	2.068118e+06
133918	36	TRANSFER	2352524.18	2352524.18	0.0	0.00	0.00	1	C	0.0	2.352524e+06
47341	33	TRANSFER	3295227.84	3295227.84	0.0	0.00	0.00	1	C	0.0	3.295228e+06
110327	35	TRANSFER	3606943.31	3606943.31	0.0	0.00	0.00	1	C	0.0	3.606943e+06
233519	39	TRANSFER	2070814.27	2070814.27	0.0	0.00	0.00	1	C	0.0	2.070814e+06
43227	28	CASH_OUT	3832058.34	3832058.34	0.0	531662.14	4363720.48	1	C	0.0	-9.313226e-10
43265	32	CASH_OUT	604933.67	604933.67	0.0	35839.70	640773.38	1	C	0.0	-1.000000e-02

Apprentissage par step

Tableau 11 : Qualité des modèles Accuracy et F1 score avec Gradient Boosting

	Train		Test	
Score	**Accuracy**	**F1-score**	**Accuracy**	**F1-score**
Modèle 1	0.9999	0.9966	0.9999	0.9761
Modèle 2	0.9999	0.9967	0.9999	0.9615
Modèle 3	0.9999	0.9967	0.9999	0.9666
Modèle 4	0.9999	0.9807	0.9999	0.9761

Nous estimons avec Xgboost sur 4 modèles différents selon le schéma et les 3 hypothèses (voir page 13), permettant d'avoir une cohérence dans le temps. L'indicateur F1-score montre que la détection de fraude est de bonne qualité dans les 4 modèles sur la base test.

Nous remarquons également que nous n'avons aucun problème de sur et sous-apprentissage dans nos résultats.

Apprentissage sur base test par step

Utilisation de Xgboost sur l'ensemble de la base test pour chaque step entre 21 et 39, pour vérifier la pertinence du modèle dans le temps. Les prédiction de fraude sont réalisées par l'apprentissage du modèle 4. Le tableau 12 dans l'annexe 21 indique les résultats d'estimations par step.

Nous retrouvons bien les 7 observations mal prédites vu précédemment, et l'ensemble des F1-score sont de bonne qualités avec une valeur moyenne de 0.98.

Ces résultats nous permettent de dire que Xgboost est efficace pour détecter des transactions frauduleuses a l'instant t, malgré que l'estimation n'est pas réalisé sur des données d'apprentissage adaptés selon les trois hypothèses.

Conclusion

Conclusion

La plupart des fraudes se font uniquement sur **les destinataires de type C (clients)**.

XGBoost

Le meilleur modèle illustrant au mieux la fraude est le **Xgboost** dont le F1- score est de 0,9919.

La création de la variable **Erreur_orig** était pertinente, car elle est **importante** pour les différents modèles établis.

Le **montant de la transaction**, **les erreurs** sur **les comptes d'origine** ainsi que le **solde après transaction** sont les variables les **plus importantes des modèles.**

Lorsque nous avons odlbalance à 0 et newbalance 0, nous **soupçonnons une fraude.**

Annexe

Annexe

Tableau 4: description statistique des variables quantitatives

	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest
count	5.000000e+05	5.000000e+05	5.000000e+05	5.000000e+05	5.000000e+05
mean	1.663937e+05	9.116928e+05	9.314261e+05	9.827739e+05	1.162668e+06
std	2.725841e+05	3.016901e+06	3.054015e+06	2.336426e+06	2.510610e+06
min	1.000000e-01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	1.335032e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	8.137560e+04	1.856900e+04	0.000000e+00	1.192711e+05	2.221441e+05
75%	2.226441e+05	1.702941e+05	2.101716e+05	8.962267e+05	1.201403e+06
max	1.000000e+07	3.893942e+07	3.894623e+07	4.148270e+07	4.148270e+07

Tableau 5 : 4 observations outliers avec amount maximum

	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud
4440	4	TRANSFER	10000000.0	C7162498	12930418.44	2930418.44	C945327594	0.00	0.00	1
4441	4	CASH_OUT	10000000.0	C351297720	10000000.00	0.00	C766681183	0.00	9941904.21	1
481250	19	TRANSFER	10000000.0	C416779475	11861008.32	1861008.32	C380259496	0.00	0.00	1
481251	19	CASH_OUT	10000000.0	C2050703310	10000000.00	0.00	C1622860679	504326.62	10342417.90	1

Annexe

figure 4.A : Sortie informatique python

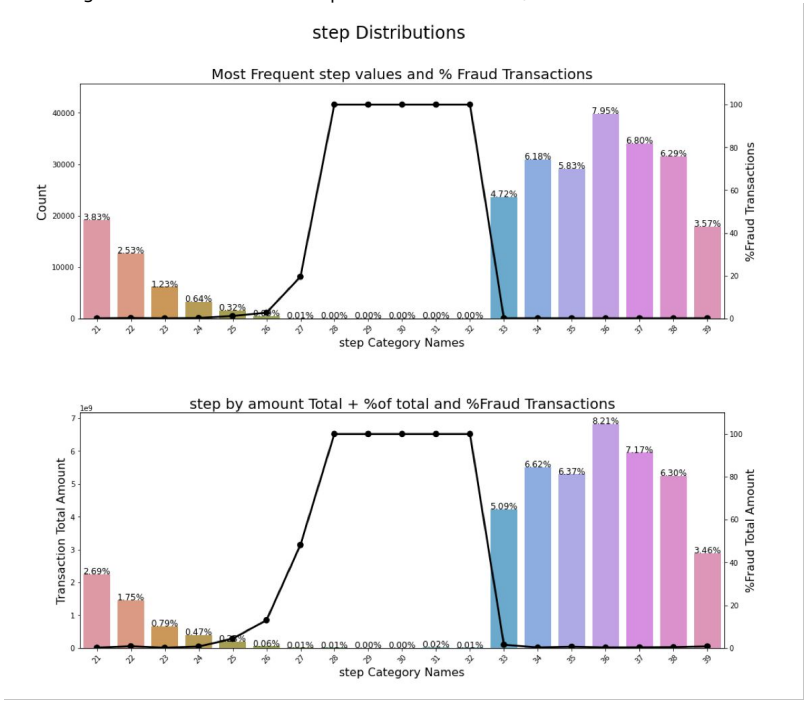
```python
#Les ind qui ont newbalanceOrig zero car la majorité de la fraude provient d'eux
#quand le destinaire est un C
df_train['Orig_newbal_zero_destC'] = 0
df_train.loc[(df_train['newbalanceOrig']==0) &
             (df_train['dest']=='C'),'Orig_newbal_zero_destC']=1

df_test['Orig_newbal_zero_destC'] = 0
df_test.loc[(df_test['newbalanceOrig']==0) &
             (df_test['dest']=='C'),'Orig_newbal_zero_destC']=1
```

```python
pd.crosstab(df_train['Orig_newbal_zero_destC'], df_train['isFraud'])
```

isFraud	0	1
Orig_newbal_zero_destC		
0	301619	4
1	198148	229

Remarque : L'ensemble des transactions dont newbalanceOrig et avec dest 'C' sont une fraude avec 229 observations (233 en total)

Figure 5.C : Distribution step + fraud + amount / total

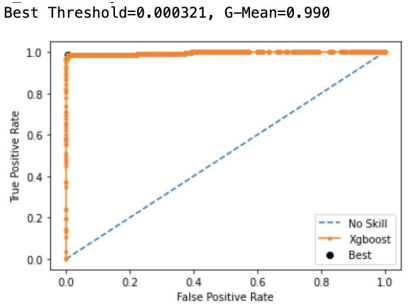


Annexe

Tableau 7 : Evaluation métrique test et train

Indicateur de performance	Gradient Boosting	Random Forest	Light	Decision Tree
Train F1-score	0,9968	0,9957	0,9799	1,0000
Test F1-score	0,9919	0,9919	0,9908	0,9852
AUC train	0,9936	0,9914	0,9614	1,0000
AUC test	0,9842	0,9842	0,9819	0,9842
Accuracy train	0.999	0.999	0.999	1,0000
Accuracy test	0.999	0.999	0.999	0,9999
Recall train	0,9871	0,9828	0,9227	1,0000
Recall test	0,9683	0,9683	0,9638	0,9683
Precision train	1,0000	1,0000	1,0000	1,0000
Precision test	1,0000	1,0000	1,0000	0,9727
Cohen train	0,9935	0,9913	0,9598	1,0000
Cohen test	0,9839	0,9839	0,9816	0,9705
Matthews Cohen train	0,9935	0,9914	0,9606	1,0000
Matthews Cohen test	0,9840	0,9840	0,9817	0,9705

Figure 10.B : Courbe AUC-ROC



Remarque : Nous pouvons voir que le point noir pour le seuil optimal peut sembler être proche du coin supérier gauche et TPR = 1. Il se rapproche d'un classificateur parfait qui ne commet aucune erreur de type fraude ou non.

Annexe

Tableau 12 : Qualité de prédiction sur base test par step

Step	false_no_fraud	true_no_fraud	false_fraud	true_fraud	false_expect	F1_SCORE
21	1,0	19147,0	0,0	4,0	1,0	0,888889
22	1,0	12612,0	0,0	22,0	1,0	0,977778
23	0,0	6142,0	0,0	2,0	0,0	1,000000
24	0,0	3210,0	0,0	6,0	0,0	1,000000
25	0,0	1580,0	0,0	18,0	0,0	1,000000
26	0,0	428,0	0,0	12,0	0,0	1,000000
27	0,0	33,0	0,0	8,0	0,0	1,000000
28	0,0	0,0	0,0	4,0	0,0	1,000000
29	0,0	0,0	0,0	4,0	0,0	1,000000
30	0,0	0,0	0,0	8,0	0,0	1,000000
31	0,0	0,0	0,0	12,0	0,0	1,000000
32	0,0	0,0	0,0	12,0	0,0	1,000000
33	0,0	23604,0	0,0	12,0	0,0	1,000000
34	0,0	30882,0	0,0	22,0	0,0	1,000000
35	1,0	29142,0	0,0	14,0	1,0	0,965517
36	1,0	39755,0	0,0	18,0	1,0	0,972973
37	1,0	33987,0	0,0	12,0	1,0	0,960000
38	2,0	31437,0	0,0	14,0	2,0	0,933333
39	0,0	17820,0	0,0	10,0	0,0	1,000000
				Somme true_fraud	**Somme false_expect**	**Moyenne F1_SCORE**
				214,0	7,0	0,984131