ONCFM

Détection de faux billets

Présentation des données

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
0	True	171.81	104.86	104.95	4.52	2.89	112.83
1	True	171.46	103.36	103.66	3.77	2.99	113.09
2	True	172.69	104.48	103.50	4.40	2.94	113.16
3	True	171.36	103.91	103.94	3.62	3.01	113.51
4	True	171.73	104.28	103.46	4.04	3.48	112.54

Le fichier comprend 1500 billets avec les marges, leurs longueurs, les hauteurs et la diagonal.

Nous savons que 1000 billets sont classés vrais et 500 faux.

valeurs manquantes

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   is_genuine     1500 non-null   bool   
 1   diagonal       1500 non-null   float64
 2   height_left    1500 non-null   float64
 3   height_right   1500 non-null   float64
 4   margin_low     1463 non-null   float64
 5   margin_up      1500 non-null   float64
 6   length         1500 non-null   float64
dtypes: bool(1), float64(6)
memory usage: 71.9 KB
```

Des valeurs manquantes ont été repérées sur la colonne margin_low !!

Pas de panique !!

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1500 entries, 72 to 1499
Data columns (total 7 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   is_genuine     1500 non-null   bool   
 1   diagonal       1500 non-null   float64
 2   height_left    1500 non-null   float64
 3   height_right   1500 non-null   float64
 4   margin_up      1500 non-null   float64
 5   length         1500 non-null   float64
 6   margin_low     1500 non-null   float64
dtypes: bool(1), float64(6)
memory usage: 83.5 KB
```

Des solutions existent pour combler ce manque !

J'ai choisi la régression linéaire pour remplacer les valeurs manquantes, ce qui consiste à établir une relation linéaire entre une variable, dite expliquée, et une ou plusieurs variables, dites explicatives.

Analyse de la regression

OLS Regression Results

Dep. Variable:	margin_low	R-squared:	0.617
Model:	OLS	Adj. R-squared:	0.615
Method:	Least Squares	F-statistic:	390.7
Date:	Wed, 15 Dec 2021	Prob (F-statistic):	4.75e-299
Time:	17:44:10	Log-Likelihood:	-774.14
No. Observations:	1463	AIC:	1562.
Df Residuals:	1456	BIC:	1599.
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.8668	8.316	0.345	0.730	-13.445	19.179
is_genuine[T.True]	-1.1406	0.050	-23.028	0.000	-1.238	-1.043
diagonal	-0.0130	0.036	-0.364	0.716	-0.083	0.057
height_left	0.0283	0.039	0.727	0.468	-0.048	0.105
height_right	0.0267	0.038	0.701	0.484	-0.048	0.102
margin_up	-0.2128	0.059	-3.621	0.000	-0.328	-0.098
length	-0.0039	0.023	-0.166	0.868	-0.050	0.042

Omnibus:	21.975	Durbin-Watson:	2.038
Prob(Omnibus):	0.000	Jarque-Bera (JB):	37.993
Skew:	0.061	Prob(JB):	5.62e-09
Kurtosis:	3.780	Cond. No.	1.95e+05

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.95e+05. This might indicate that there are
strong multicollinearity or other numerical problems.

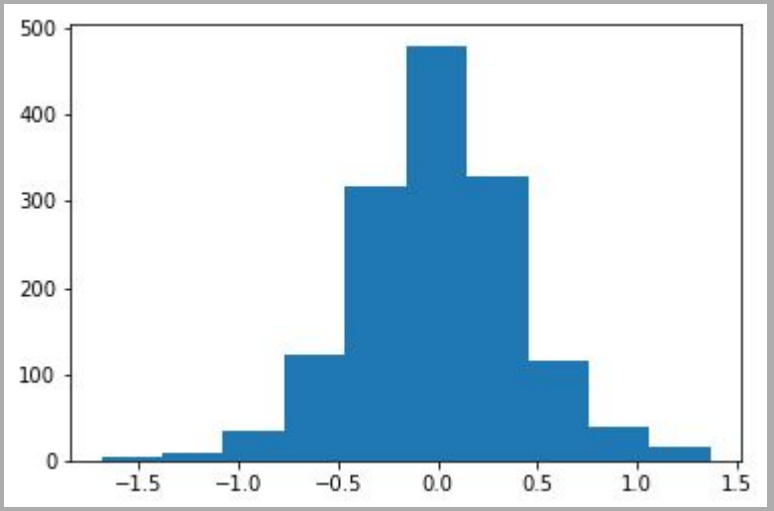
Avec un R^2 de 0.617, nous n'avons pas un modèle de bonne qualité, nous recherchons plus 0.90.

Les variables margin_low et is_genuine sont statistiquement significative car leur ρ>|t| est égal à 0.00.

Le fait de garder que ces 2 variables pourraient influer sur le R^2.(ce qui n'est pas le cas)

coef : La modélisation de margin_low permet de trouver margin_low par rapport aux autres variables

Analyse de la regression



Pour la normalité des résidus

Nous avons un modèle linéaire non faussé car la distribution suit une loi normale.
(histogramme en forme de cloche)

Analyse de la regression

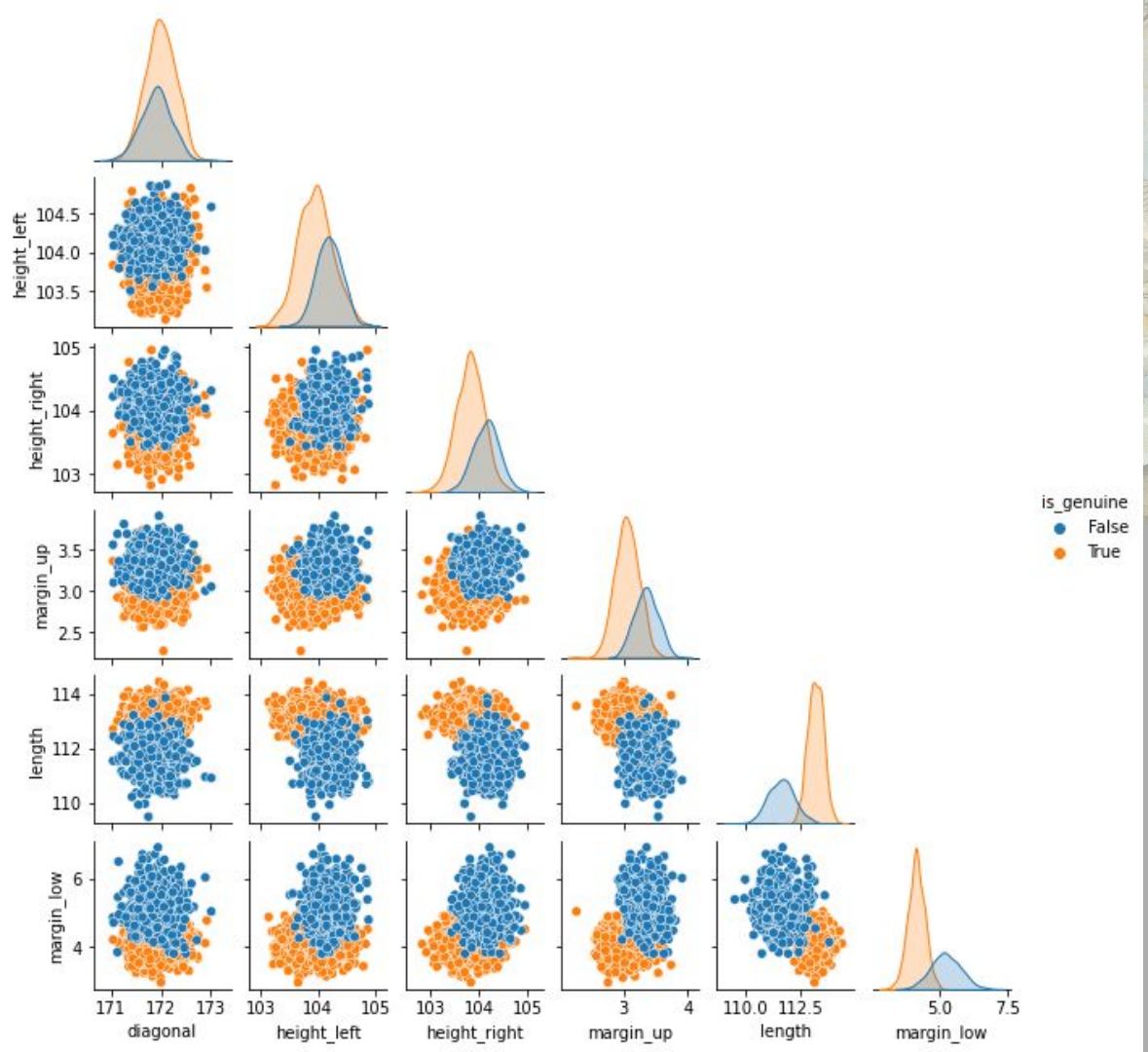
	feature	VIF
0	diagonal	170808.246898
1	height_left	114373.426488
2	height_right	105157.499668
3	margin_up	264.908266
4	margin_low	89.003990
5	length	31205.638468

Calcul du VIF
(variance inflation factor en anglais)

VIF évalue si les factures sont corrélés les uns aux autres(multi-colinéarité), ce qui pourraient influencer les autres facteurs et réduire la fiabilité du modèle.

Pour qu'un VIF soit acceptable nous avons la formule
$VIF < MAX(10,1/1*R^2)$
Dans notre cas, VIF doit être entre 10 et 2.56
Cela confirme que la régression linéaire n'est pas de bonne qualité

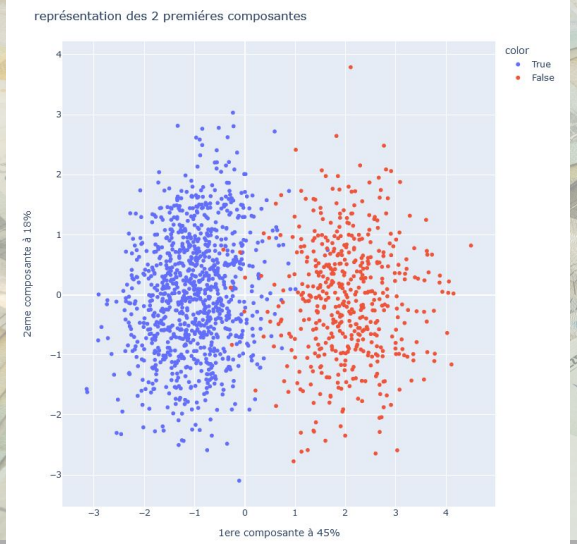
analyse de nos billets



Le Pairplot

Ce graphique permet en un seul coup d'oeil de distinguer les différences entre variables

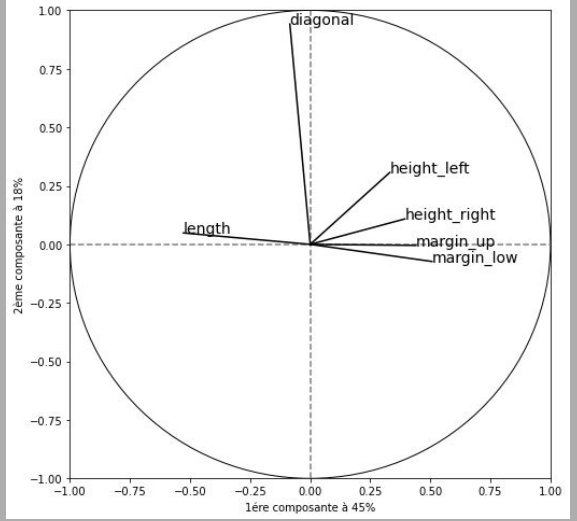
Dans nos données, nous pouvons remarquer que les vrais billets ont tendance à être plus long que les faux.
De plus les faux billets auraient une marge plus grande.
Par contre, il est difficile de voir un vrai ou faux billets avec les diagonales



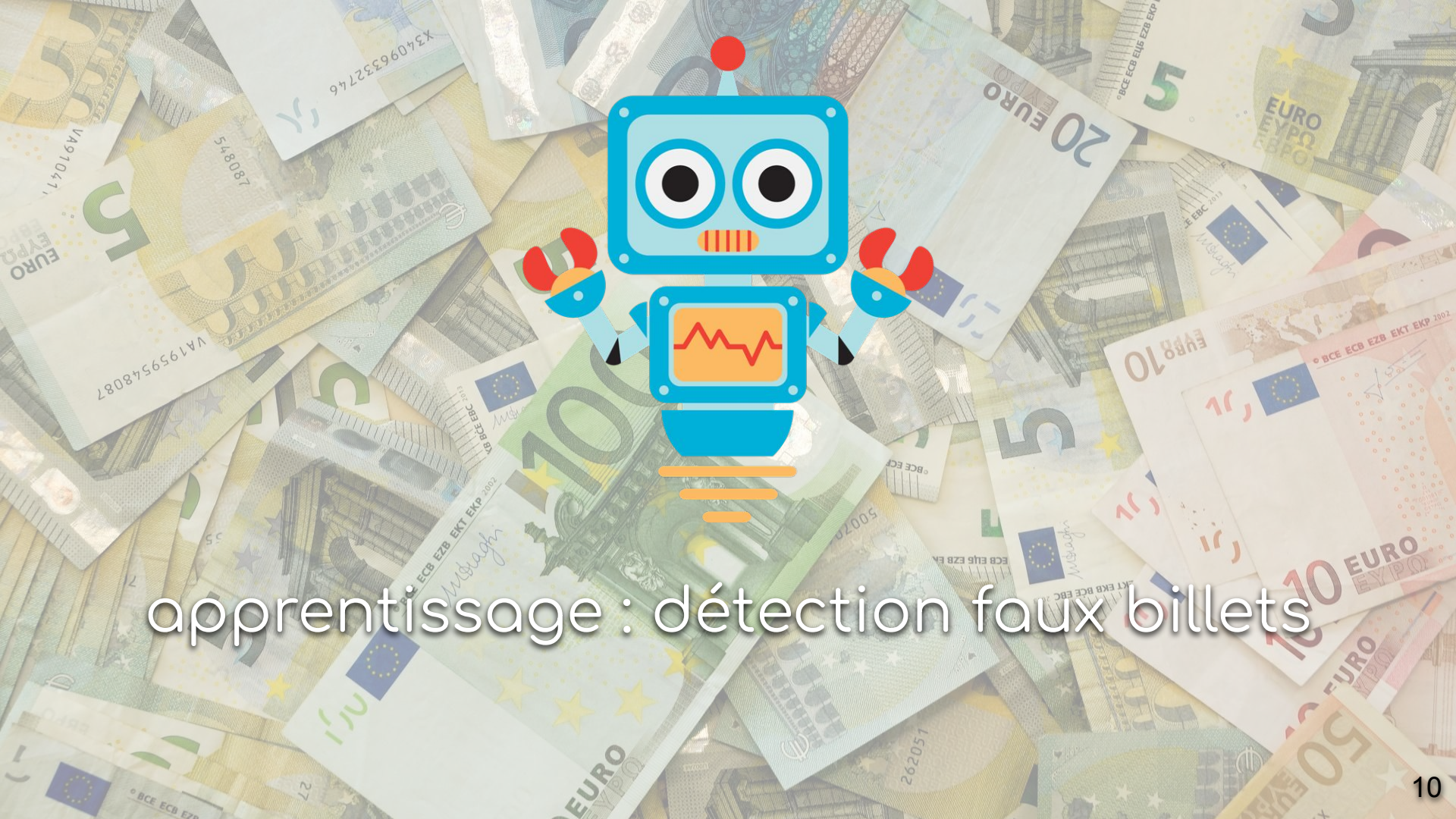
L'ACP (Analyse en composante principale)

Elle permet de réduire le nombre de variables

et de rendre l'information moins redondante.



La longueur serait une tendance pour les 'vrais' billets.
Les hauteurs et les marges pencheraient plus pour les 'faux' billets.
On peut constater aussi qu'il serait difficile de distinguer les vrais et faux billets par rapport à la diagonal.
Comme nous l'avez indiqué le pairplot



apprentissage : détection faux billets

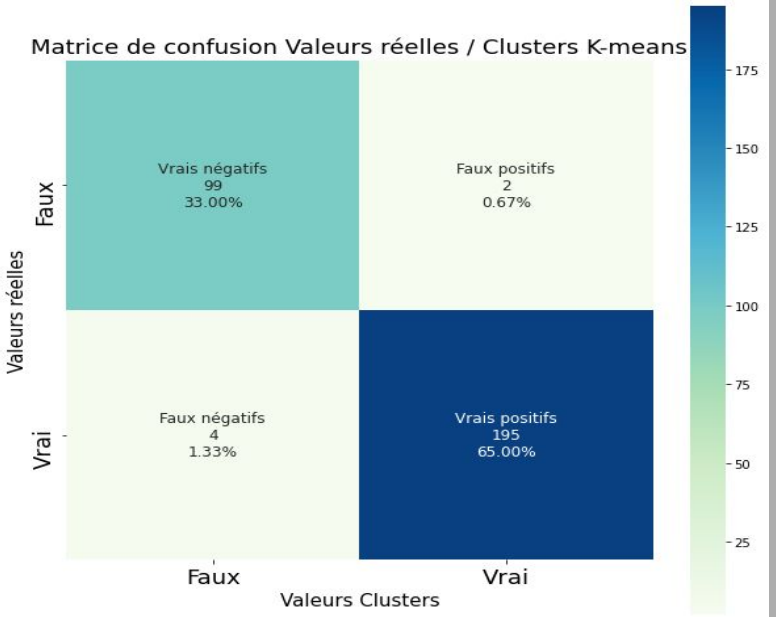
Split des données

Le Split des données consiste à séparer nos données en un jeu d'entraînement et un jeu de test.
Ici notre jeu de test correspond à 20% de nos données soit 300 lignes.

```
x = X_projected
y = billet['is_genuine']
xtrain, xtest, ytrain, ytest = train_test_split(x, y, train_size=0.8, random_state= 1)
```

Les K-means

Il permet d'analyser un jeu de données caractérisées par un ensemble de descripteurs, afin de regrouper les données "similaires" en groupes (ou clusters).



Avec l'algorithme des K-means, sur les 300 billets analysés, 198 sont classés 'vrai' et 97 sont classés 'faux', d'après notre jeu de données.

Cet algorithme se trompe sur 6 billets
Ce qui donne de bon résultats

```
Int64Index([728, 1104, 669, 1482, 626, 946],
```

Les KNN(k-nearest neighbors)

La méthode des K plus proches voisins (KNN) a pour but de classifier des points cibles (classe méconnue) en fonction de leurs distances par rapport à des points constituant un échantillon d'apprentissage (c'est-à-dire dont la classe est connue a priori)

```
: pred = knn.predict(xtest)
  knn.score(xtest, ytest)
```

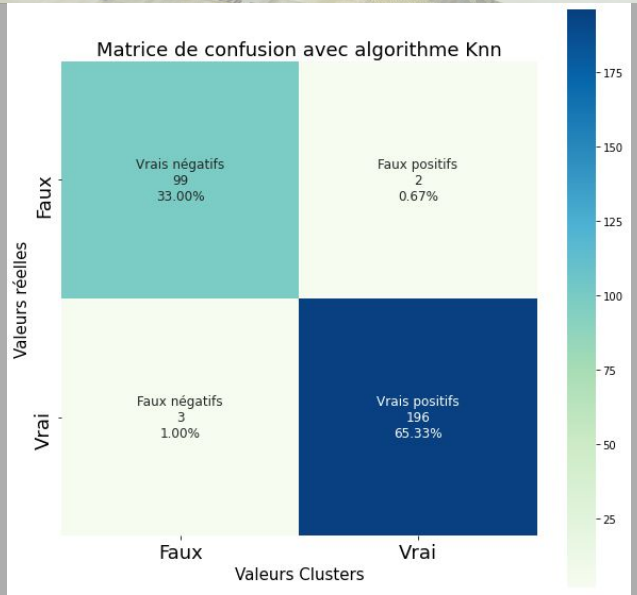
executed in 47ms, finished 12:02:05 2021-

```
0.9866666666666667
```

Avec cette méthode, et après affinement des paramètres, nous arrivons à presque 99% de réussite sur notre jeu de données.

Les KNN(k-nearest neighbors)

La méthode des K plus proches voisins (KNN) a pour but de classifier des points cibles (classe méconnue) en fonction de leurs distances par rapport à des points constituant un échantillon d'apprentissage (c'est-à-dire dont la classe est connue a priori)

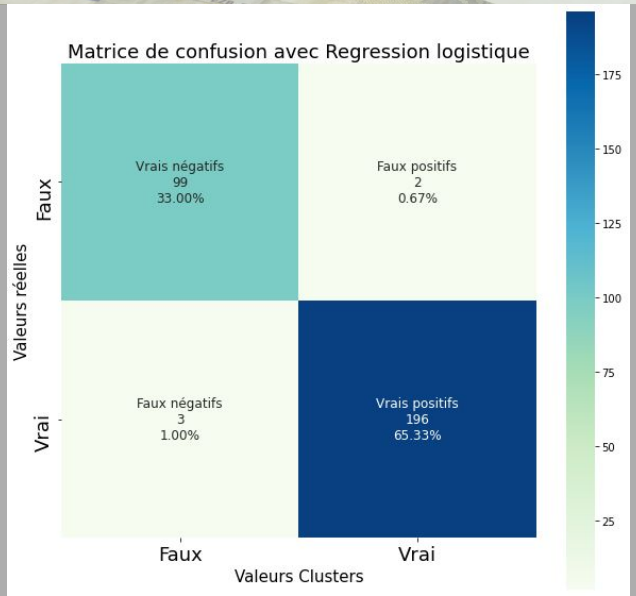


Avec un échantillon d'entraînement de 300 données, nous arrivons à 5 erreurs sur un échantillon de test !

```
Int64Index([728, 1104, 1482, 626, 946],
```

La régression logistique

La régression logistique est une méthode d'analyse statistique qui consiste à prédire une valeur de données d'après les observations réelles d'un jeu de données.



Avec le même échantillon d'entraînement que les Knn, nous arrivons à 5 erreurs sur un échantillon de test !!

```
Int64Index([728, 1104, 1482, 626, 946],
```

Récapitulatif

```
Int64Index([728, 1104, 1482, 626, 946], dtype='int64') sont les index erreurs des KNN
Int64Index([728, 1104, 1482, 626, 946], dtype='int64') sont les index erreurs de la regression logistique
Int64Index([728, 1104, 669, 1482, 626, 946], dtype='int64') sont les index erreurs des Kmeans
```

Nous pouvons remarquer que ces 3 algorithmes se trompent sur les mêmes billets.
Cependant, je choisirai la regression logistique pour la fonction.

POURQUOI ??

La regression logistique est plus adapté sur les gros datasets,(je suppose qu'il y aura beaucoup de billets à analyser),

La régression logistique est un modèle paramétrique qui suit des lois de probabilités normales.

Test sur un autre jeu de données