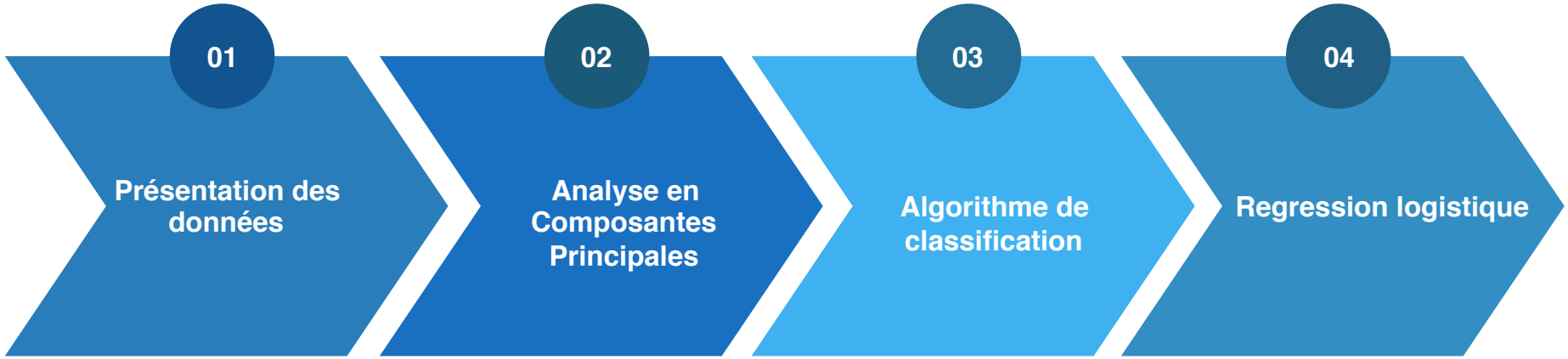
Projet 6 : Détecter des faux billets

Année 2018-2019

Sommaire


01
Présentation des données
02
Analyse en Composantes Principales
03
Algorithme de classification
04
Regression logistique


H2
Présentation des données

Notre jeu de données

7 variables

Height : hauteur

Height_left

Height : hauteur

Height_right

Margin: marge

margin_low :

Margin: marge

margin_up :

Diagonal : diagonale

Lenght : longueur

La valeur du billet

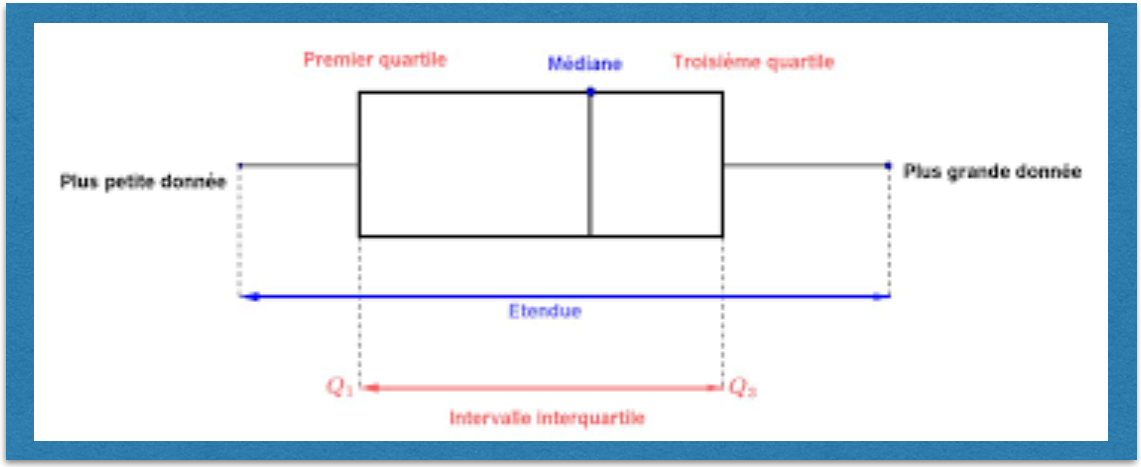
VRAI	FAUX
100	70

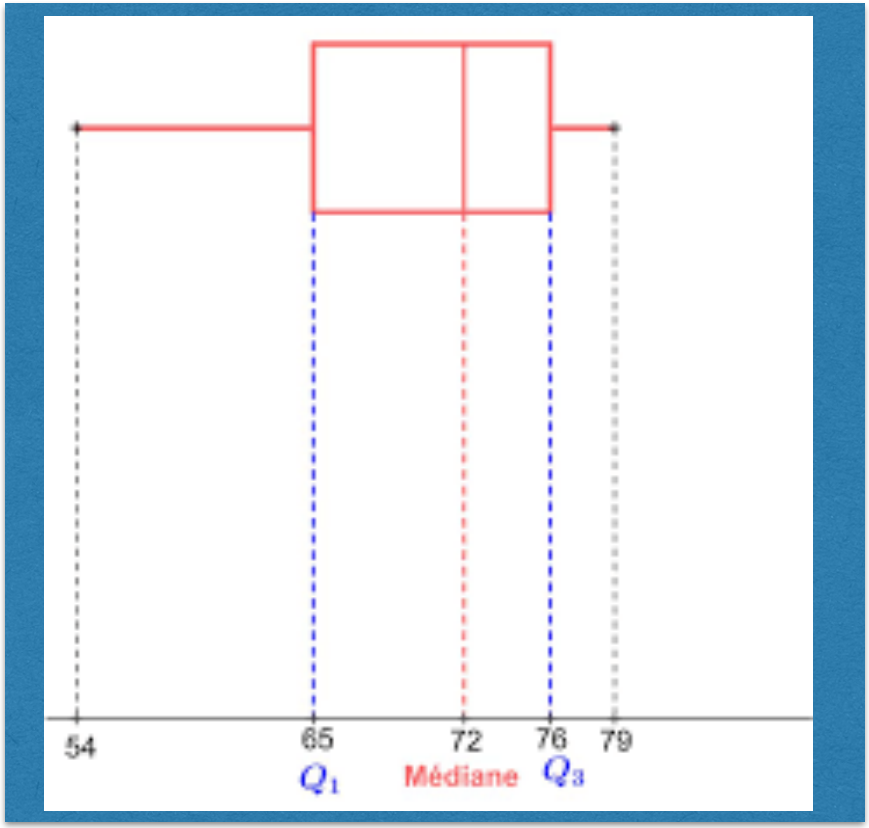
Analyse

	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	171.67	103.74	103.70	4.01	2.87	113.29
2	True	171.83	103.76	103.76	4.40	2.88	113.84
3	True	171.80	103.78	103.65	3.73	3.12	113.63
4	True	172.05	103.70	103.75	5.04	2.27	113.55

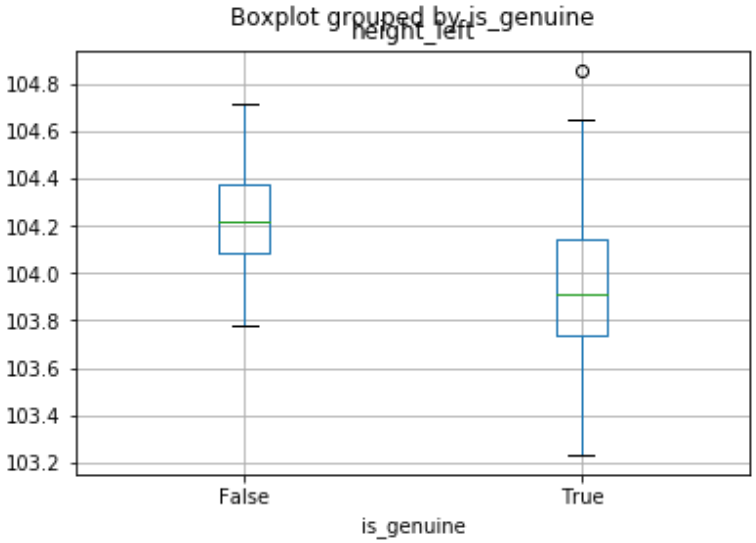
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 170 entries, 0 to 169
Data columns (total 7 columns):
is_genuine      170 non-null bool
diagonal        170 non-null float64
height_left     170 non-null float64
height_right    170 non-null float64
margin_low      170 non-null float64
margin_up       170 non-null float64
length          170 non-null float64
dtypes: bool(1), float64(6)
memory usage: 8.2 KB
```

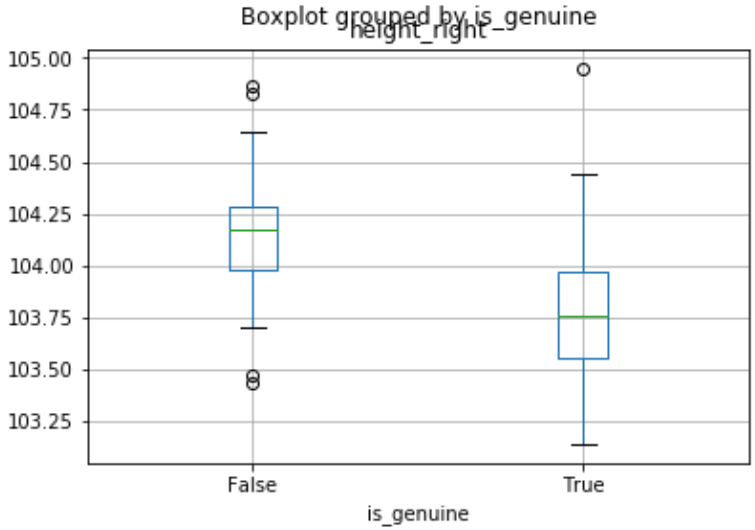
Boxplot rappel





La hauteur du billet





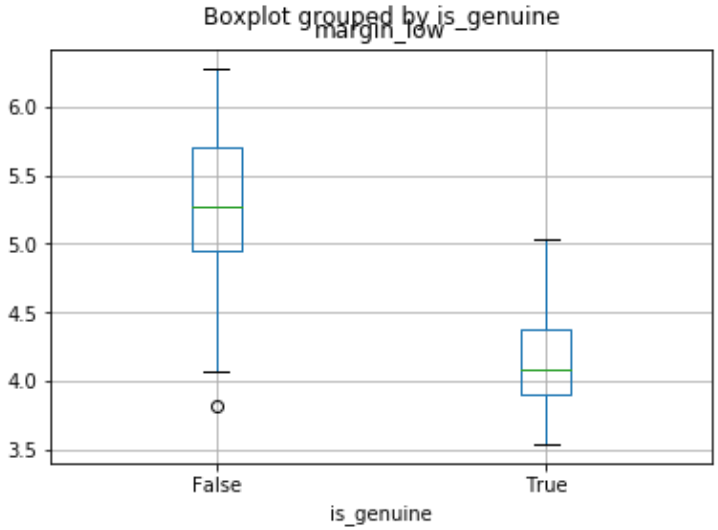
Hauteur gauche

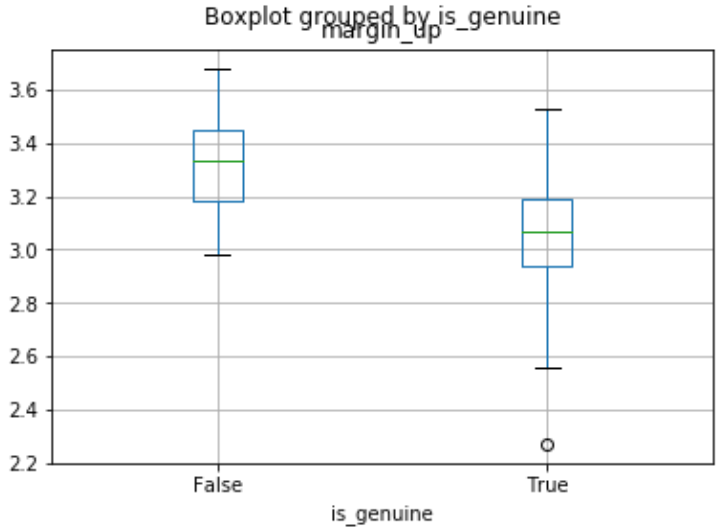
Rien de très marquant

Hauteur droite

Rien de très marquant

La marge du billet





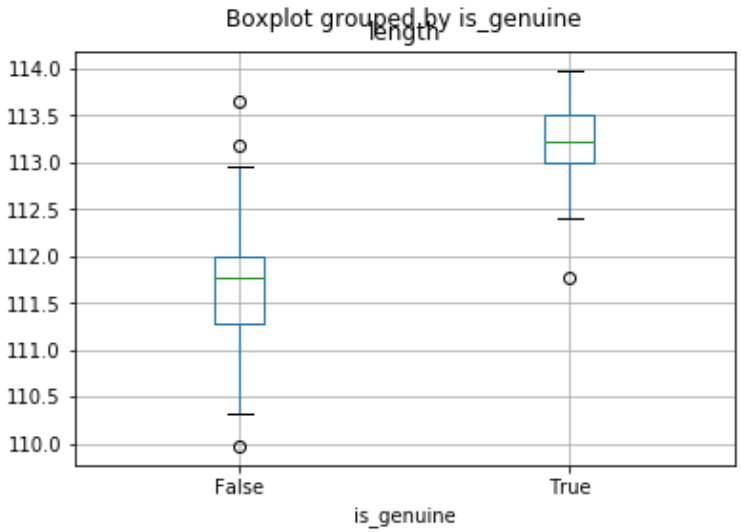
Marge Basse

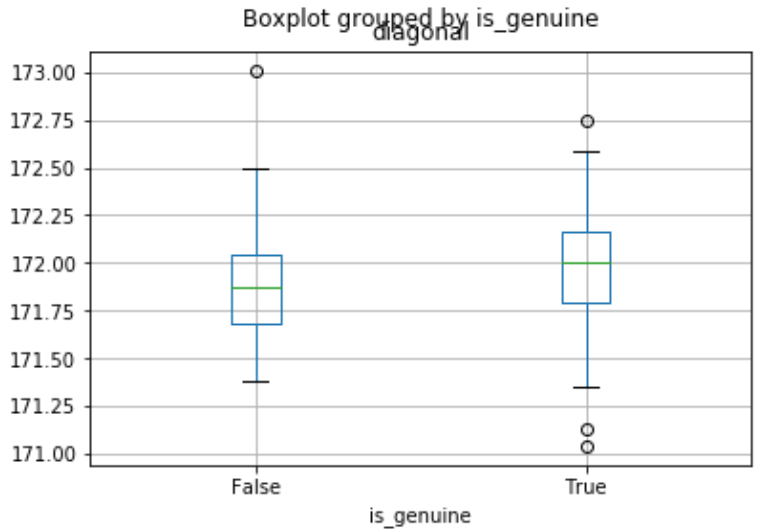
Une différence significative

Marge Haute

Une petite différence

La longueur et la diagonale du billet





La longueur du billet

Une différence significative

Diagonale

Rien de très marquant

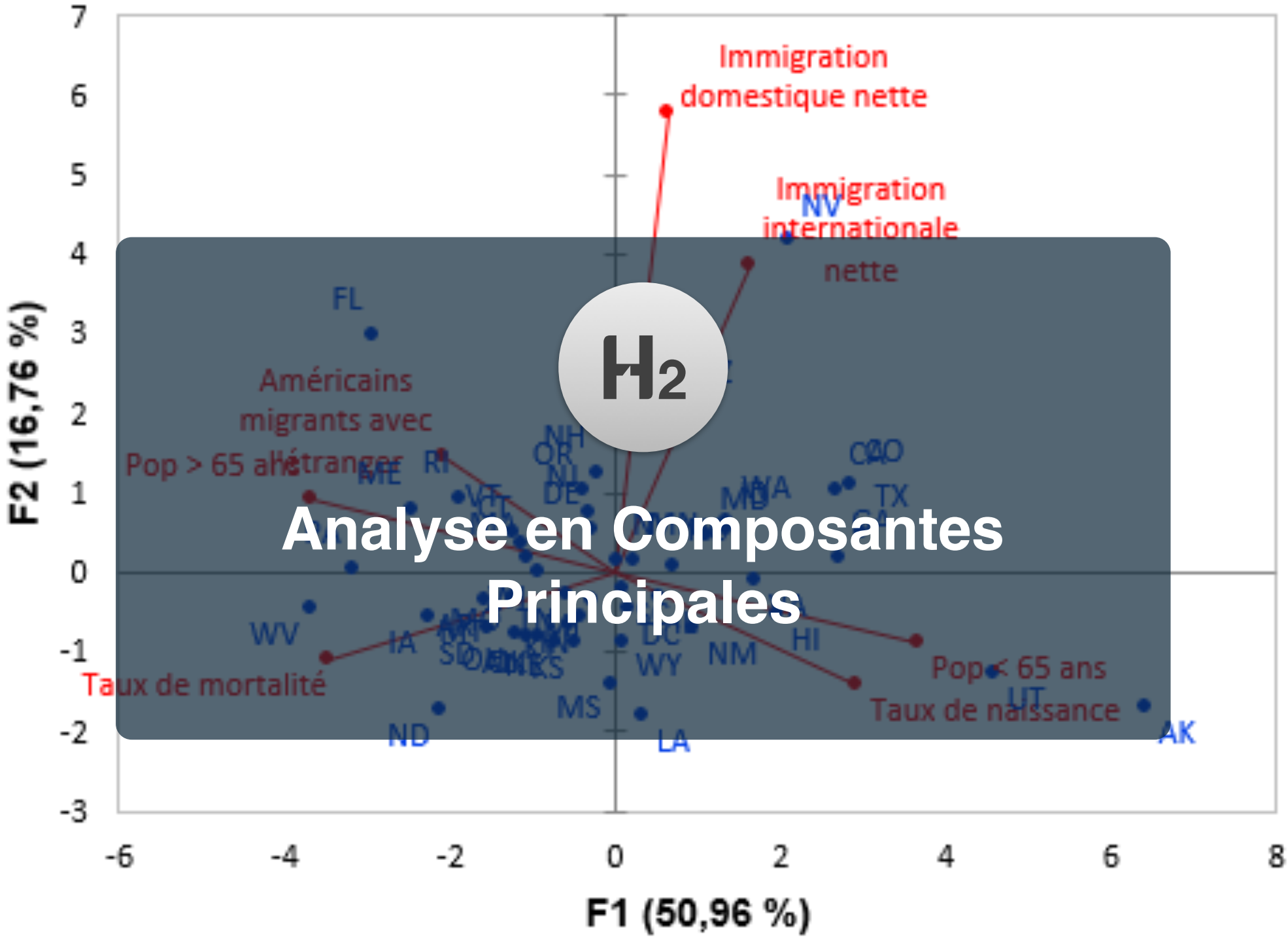
Les variables importantes

is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
False	171.889857	104.230429	104.145571	5.281571	3.334571	111.660714
True	171.976100	103.951500	103.775900	4.143500	3.055500	113.207200

0,087	0,279	0,37	1,138	0,279	1,547

Marge basse

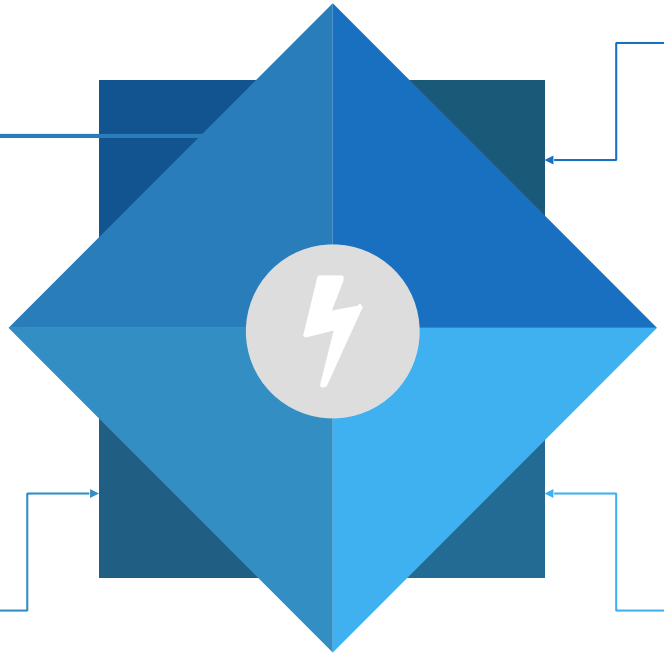
Longueur


Immigration domestique nette
Immigration internationale nette
Américains migrants avec étranger
Pop > 65 ans
Taux de mortalité
Pop < 65 ans
Taux de naissance
H2
Analyse en Composantes Principales
F1 (50,96 %)
F2 (16,76 %)
FL
NV
WV
IA
ND
LA
MS
WY
NM
HI
UT
AK
TX
RI
ME
VT
DE
OR
NH
-6
-4
-2
0
2
4
6
8
7
6
5
4
3
2
1
0
-1
-2
-3

ACP

Analyse de l'éboulis des valeurs propres

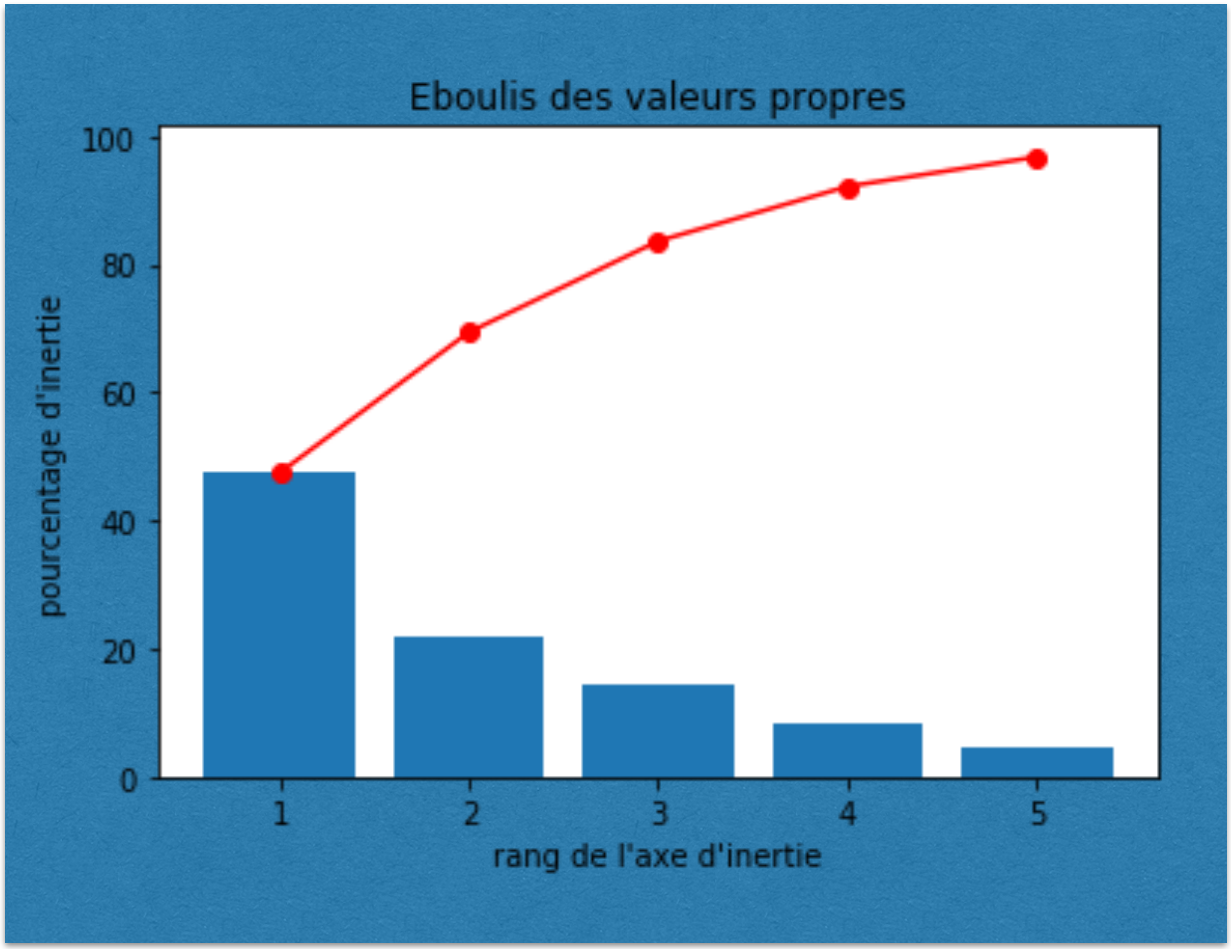
Représentation des variables par le cercle de corrélation



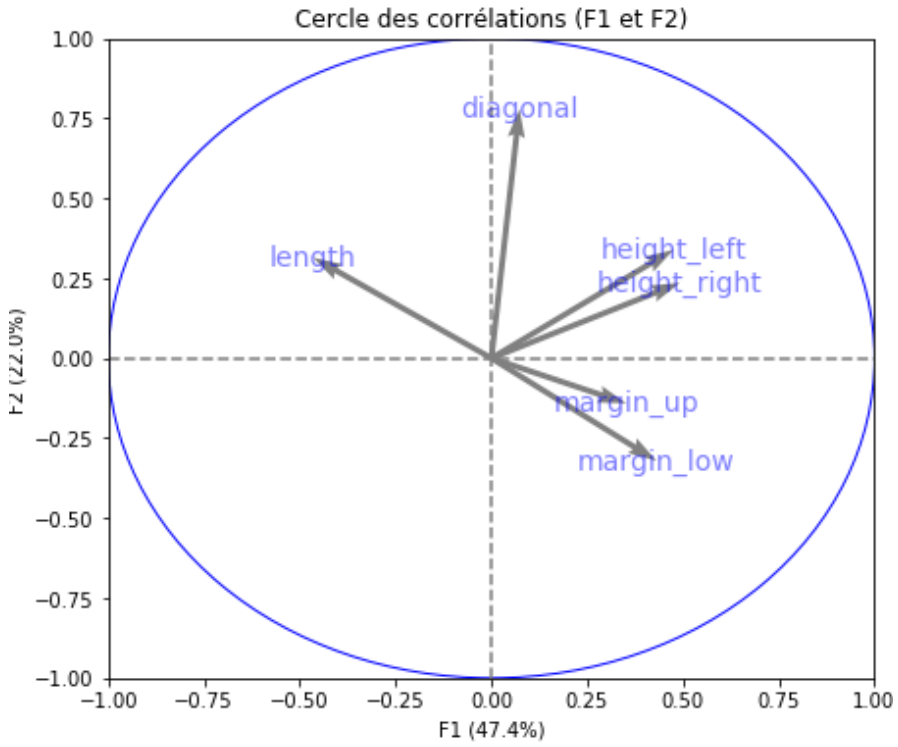
Analyser la qualité de représentation et la contribution des individus

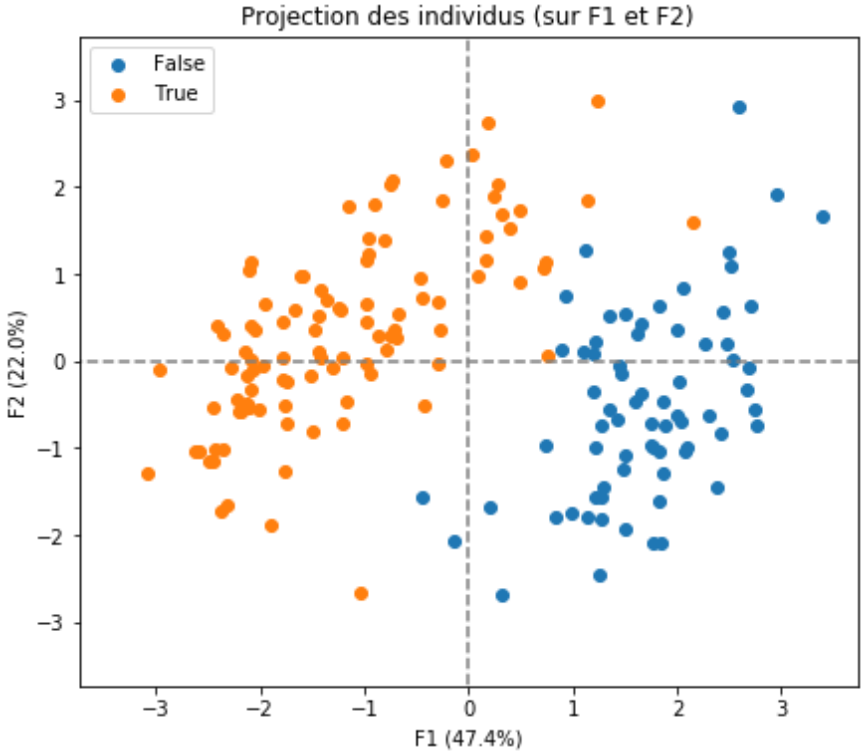
Représentation des individus par les plans factoriels

Analyse de l’éboulis des valeurs propres



Cercle des corrélations

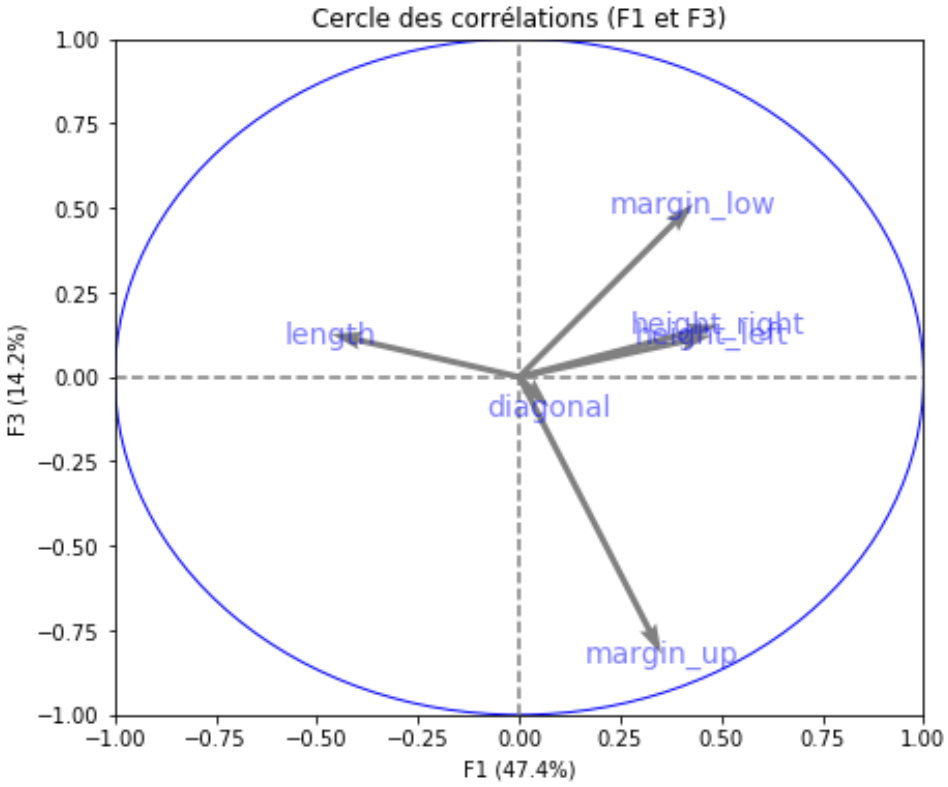


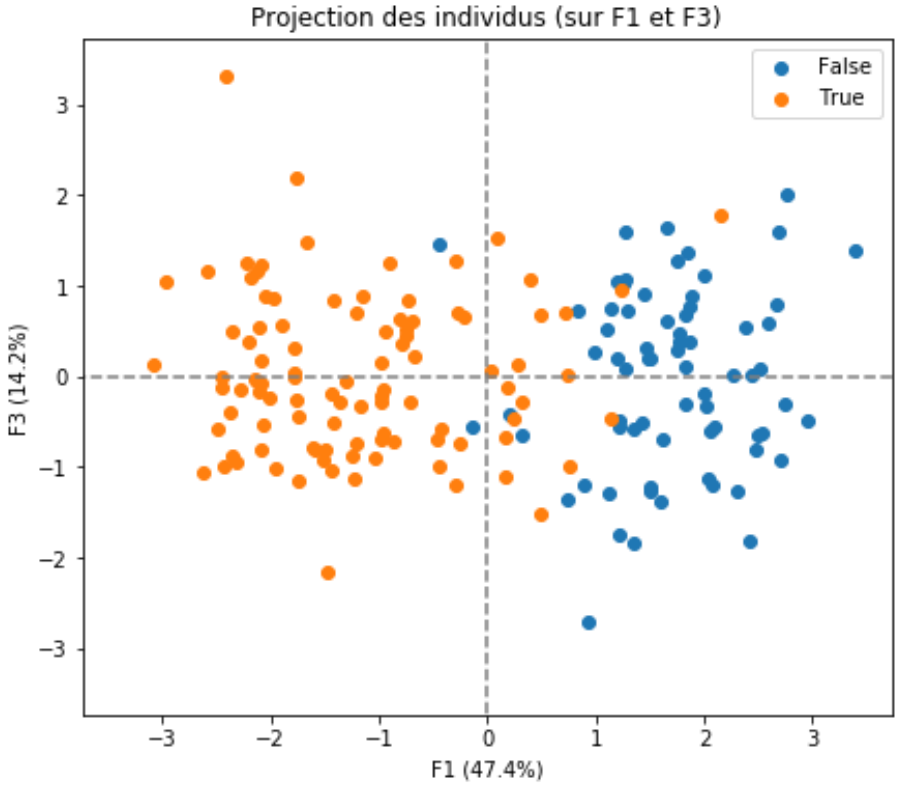


Cercle de corrélation de (F1 et F2)

Projection des individus sur (F1 et F2)

Cercle des corrélations





Cercle de corrélation de (F1 et F3)

Projection des individus sur (F1 et F3)

Analyse

	ID	d_i
166	False	20.625650
0	True	18.410598
4	True	18.039567
122	False	16.790944
39	True	15.052608
112	False	14.422407
151	False	13.637281
49	True	12.917538

La contribution des individus

La contribution : indique l'influence de l'individu dans la définition du facteur

Qualité de représentation des individus - COS2:

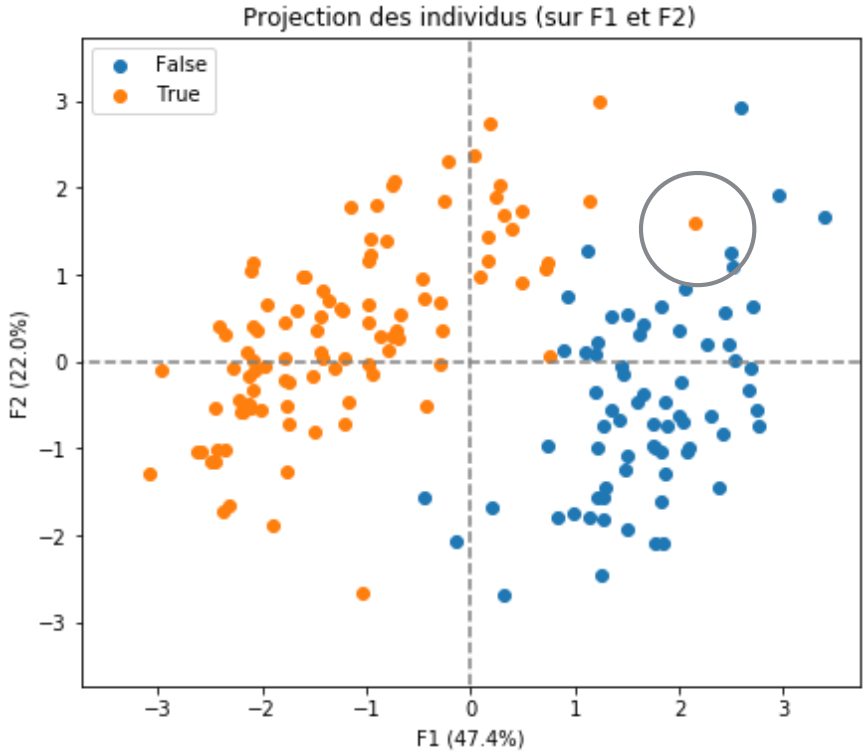
Indique la qualité de la représentation de l'individu sur le facteur

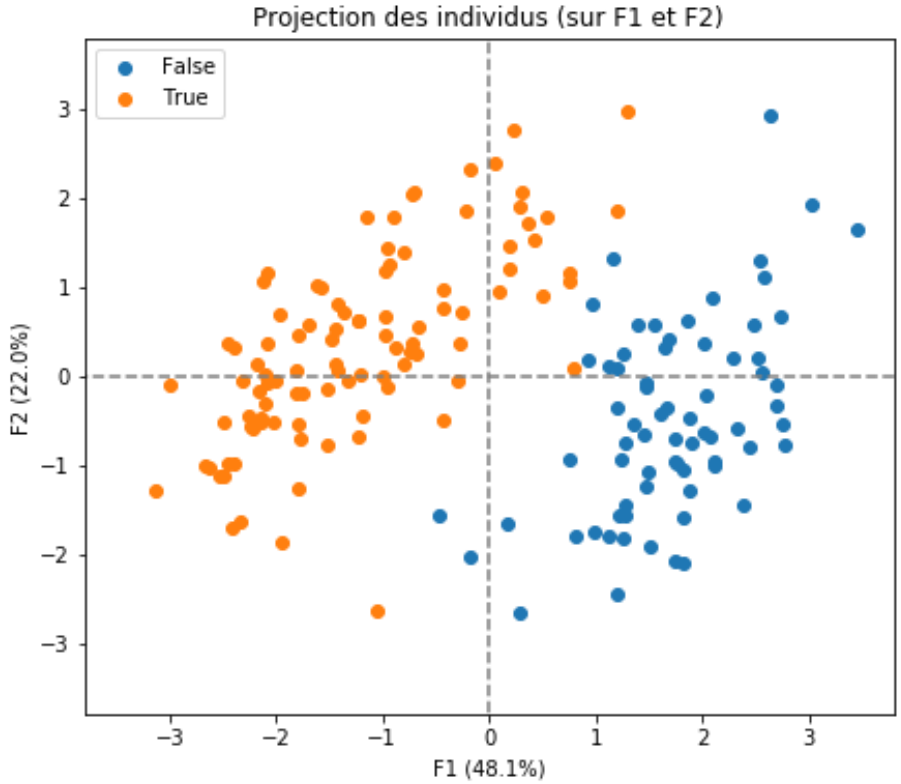
	id	COS2_1	
0	True	4.957420e-07	7
65	True	3.070454e-03	1
51	True	6.715803e-03	9
144	False	7.037218e-03	2
80	True	7.539206e-03	7
9	True	9.917182e-03	6
112	False	1.038201e-02	5
70	True	1.099983e-02	6
102	False	1.277381e-02	8
5	True	1.289523e-02	1
43	True	1.402046e-02	7

id	COS2_1	COS2_2
True	0.395390	8.993649e-07
True	0.149231	3.962597e-06
True	0.227387	4.711665e-05
True	0.136255	1.003096e-04
True	0.113332	1.102244e-04
alse	0.381547	1.182013e-04
True	0.003070	1.304295e-04
alse	0.172407	2.075332e-04
alse	0.117415	2.714030e-04
True	0.350657	3.003629e-04
True	0.039177	3.892855e-04

COS2_2

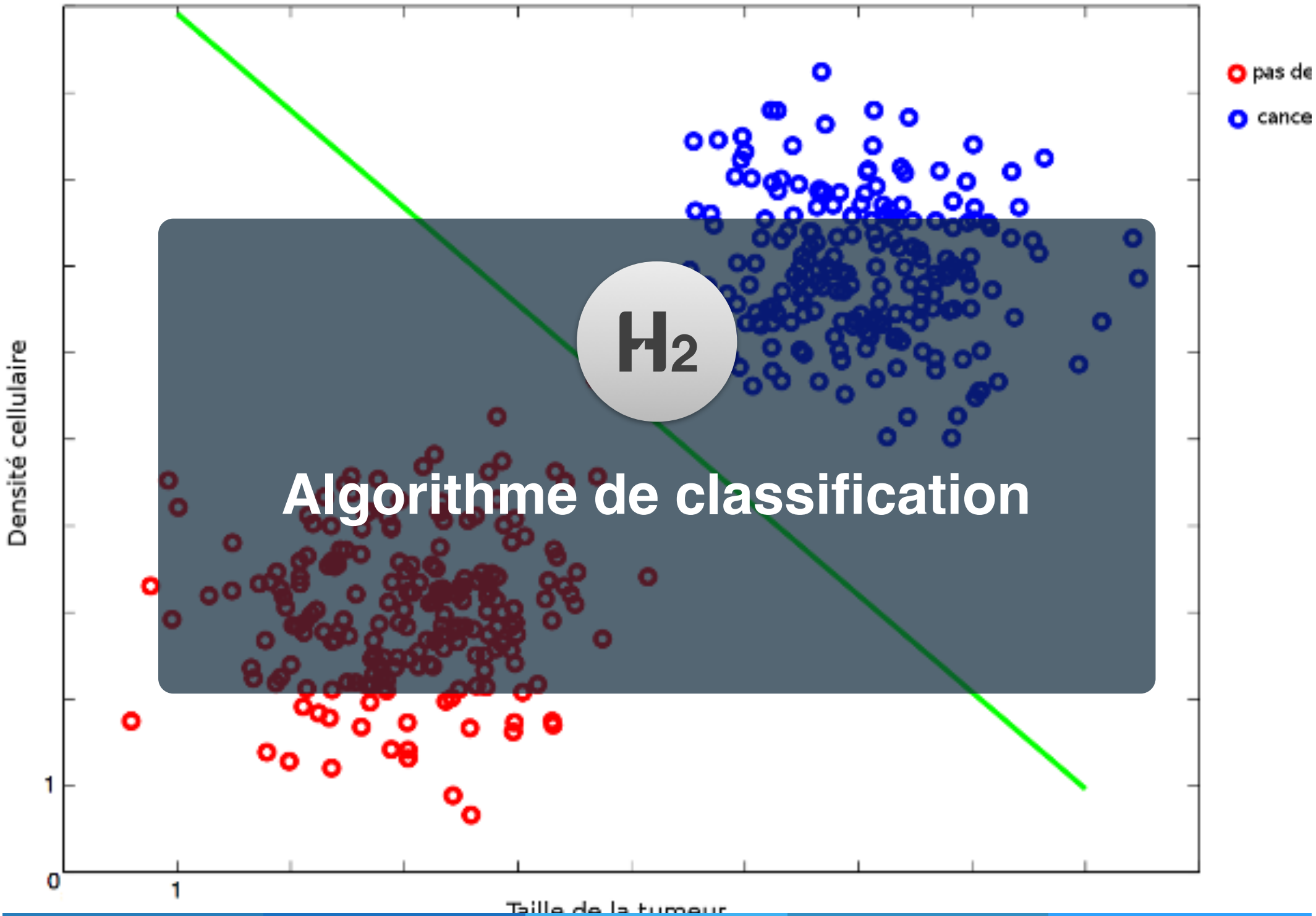
ACP



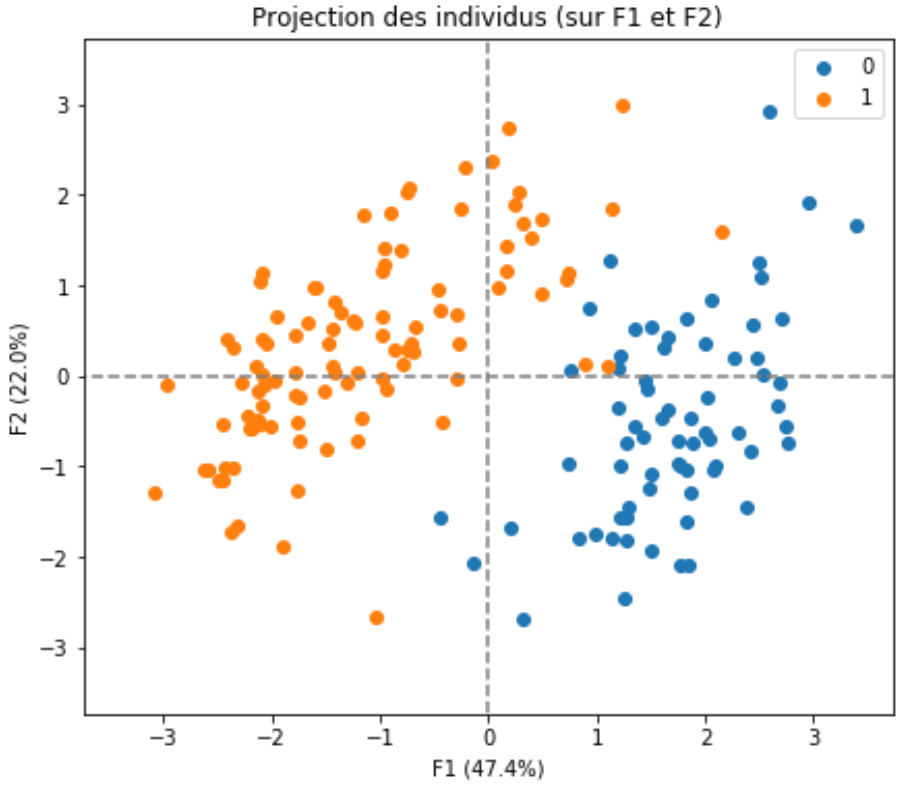


Ancienne ACP

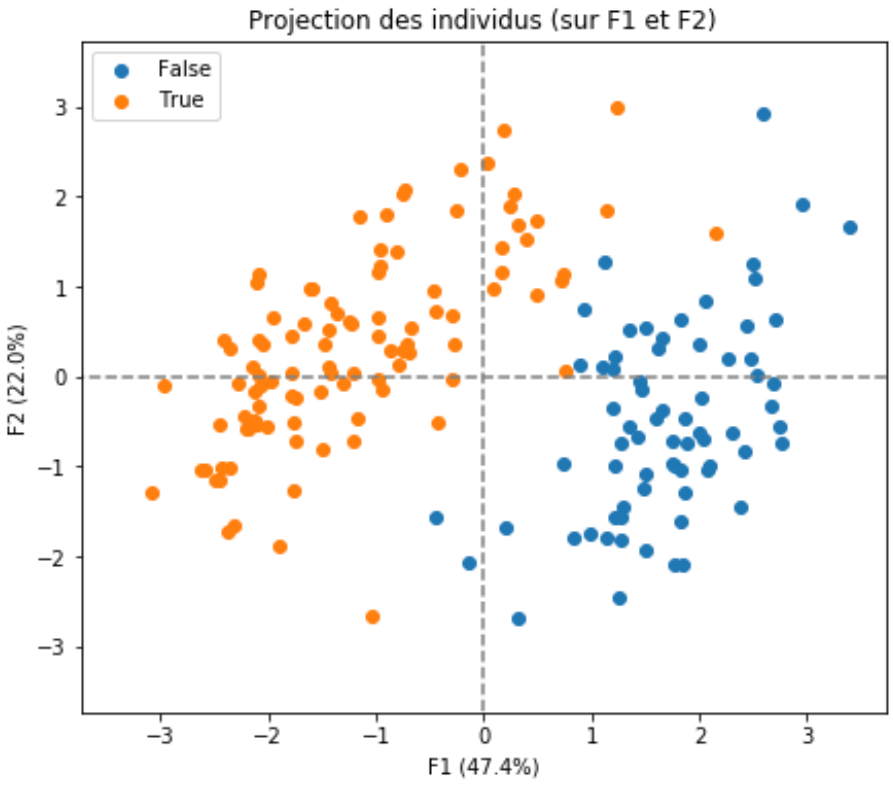
ACP avec le jeu de données pure


pas de
cance
Densité cellulaire
1
0
1
Taille de la tumeur
H2
Algorithme de classification

Résultat du k-means



Kmeans



ACP de base

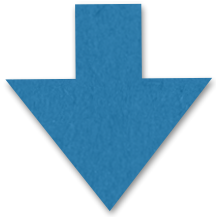
Matrice de confusion

In [110]:
```
print("Matrice de confusion")
ok = pd.crosstab(clusters,notes.index)
ok.index = ['Cluster 1','Cluster 2']
ok.columns = ['Faux','Vrais']
ok
```

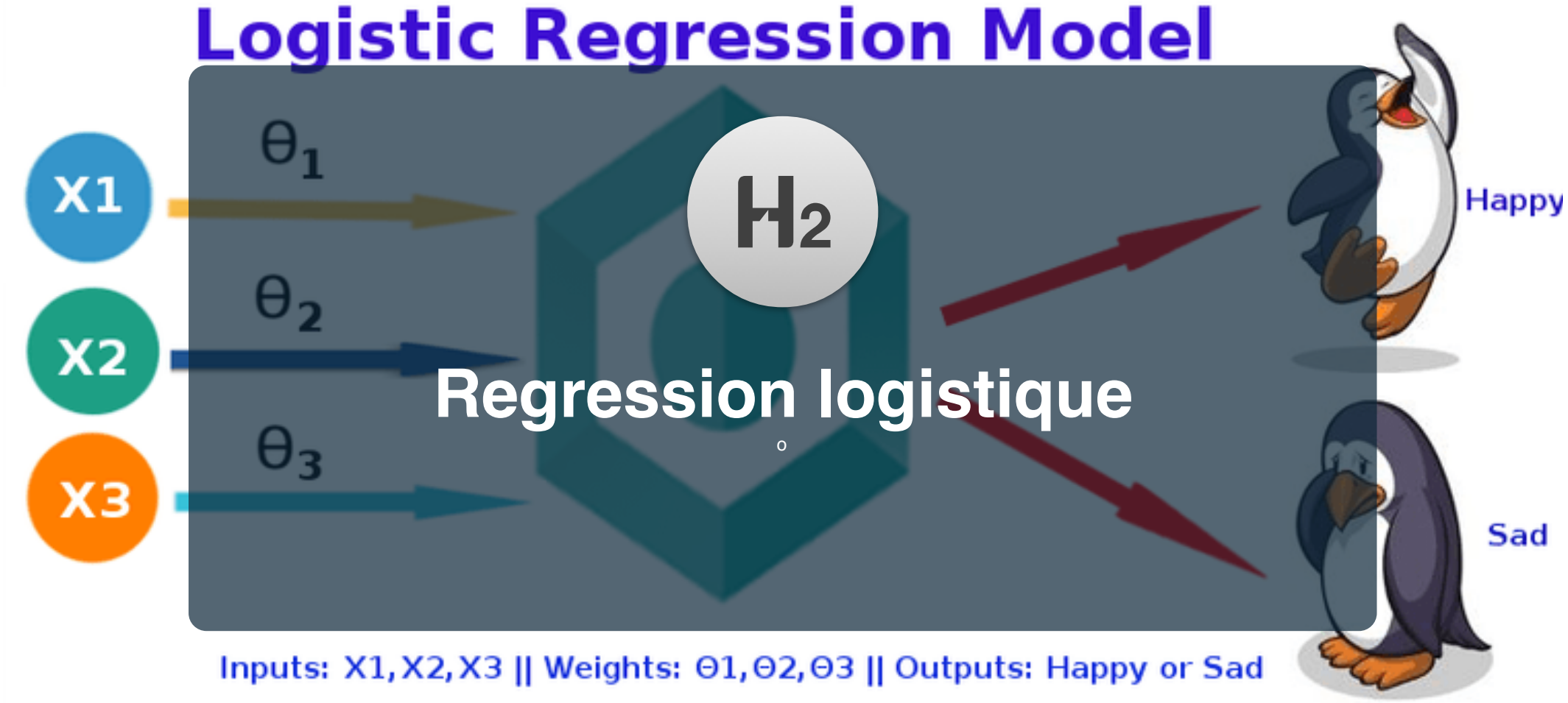
```
Matrice de confusion
```

Out[110]:

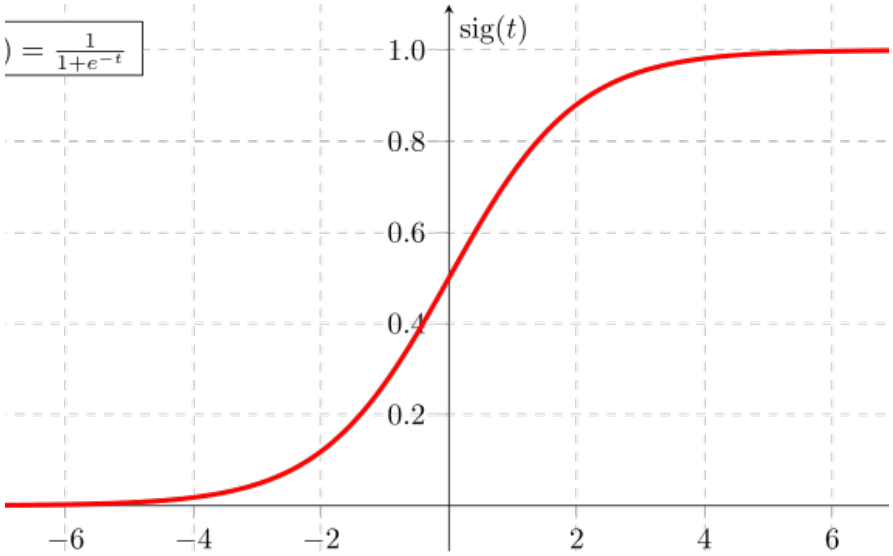
	Faux	Vrais
Cluster 1	68	1
Cluster 2	2	99



FAUX	Vrais
70	100


Logistic Regression Model
X1
X2
X3
θ1
θ2
θ3
H2
Regression logistique
Happy
Sad
Inputs: X1, X2, X3 || Weights: Θ1, Θ2, Θ3 || Outputs: Happy or Sad

Régression logistique



Une régression qui prend deux états

Vrai billet

Faux billet

Les variables qu'on utilise pour notre régression sont : margin_low, margin_up, height_left, height_right, diagonale, length

Précision du modèle

Les vrais et faux positifs ….

```
pd.crosstab(notes['is_genuine'],notes['prediction'])
```

prediction	False	True
is_genuine		
False	69	1
True	1	99

Le taux d'erreur de notre modèle

```
(2/170)#taux d'erreur
```

```
0.011764705882352941
```

La précision de notre modèle

```
from sklearn.metrics import accuracy_score
accuracy_score(notes['is_genuine'],notes['prediction'])#performance du modele good
```

```
0.9882352941176471
```

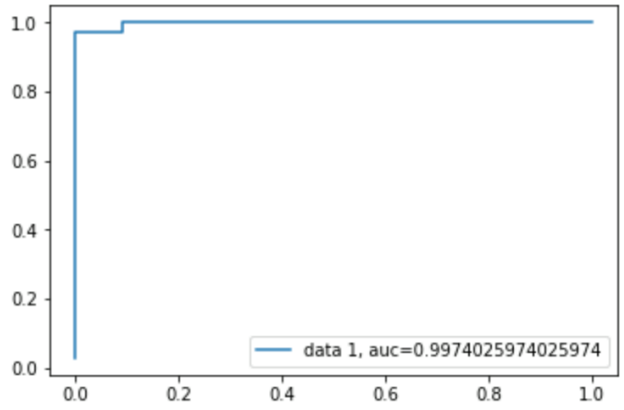
Test de la courbe de ROC

Chargement des données → **Division du jeu de données 80-20** → **Affichage de la précision du modèle** → **Affichage de la courbe**

```
print("Précision", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy 0.9649122807017544
```

```
y_pred_proba = clf.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()
```



Précision du modèle

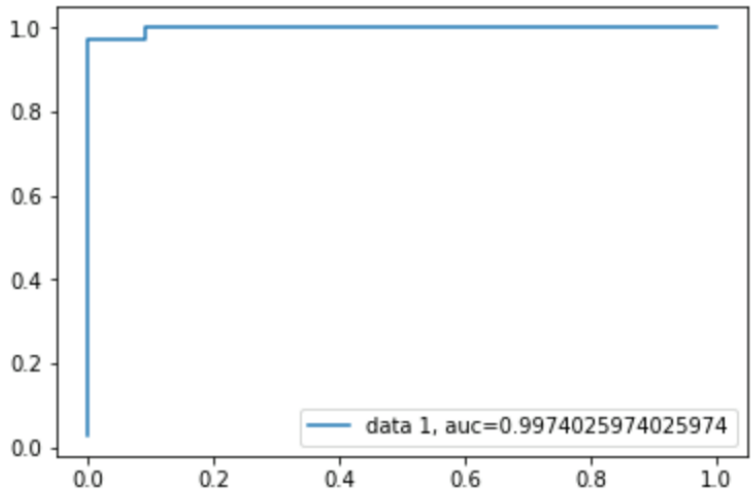
```
from sklearn.metrics import accuracy_score
accuracy_score(notes['is_genuine'],notes['prediction'])#performance du modele good
```

```
0.9882352941176471
```

```
print("Précision", metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy 0.9649122807017544
```

```
y_pred_proba = clf.predict_proba(X_test)[::,1]
fpr, tpr, _ = metrics.roc_curve(y_test,  y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)
plt.plot(fpr,tpr,label="data 1, auc="+str(auc))
plt.legend(loc=4)
plt.show()
```



RFE : recursive feature elimination

```
from sklearn.feature_selection import RFE
estimator = LogisticRegression()
selector = RFE(estimator, step=1)
selector = selector.fit(X_train, y_train)
selector.support_
```

```
array([False, False,  True,  True, False,  True])
```

Diagonal | Height_ left | Height _True | Margi n _low | Margin _up | Lenght

```
print("Accuracy", metrics.accuracy_score(selector.predict(X_test),y_test))
```

```
Accuracy 1.0
```

Place à la pratique

Le jeu test :

```
example=pd.read_csv("example.csv")
```

```
example.head()
```

	diagonal	height_left	height_right	margin_low	margin_up	length	id
0	171.76	104.01	103.54	5.21	3.30	111.42	A_1
1	171.87	104.17	104.13	6.00	3.31	112.09	A_2
2	172.00	104.58	104.29	4.99	3.39	111.57	A_3
3	172.49	104.55	104.34	4.44	3.03	113.20	A_4
4	171.65	103.63	103.56	3.77	3.16	113.33	A_5

Le jeu test :

La précision de notre modèle est de : 98%

```
from sklearn.linear_model import LogisticRegression
X = example.drop(['id'], axis=1).values
#y = notes['id']
probabilites = logistic.predict(X)
#logistic = LogisticRegression()
#logistic.fit(X,y)
```

```
probabilites
```

```
array([False, False, False,  True,  True])
```

```
print(pd.DataFrame({'ID':example['id'],'Nature':probabilites}))
```

```
    ID  Nature
0  A_1   False
1  A_2   False
2  A_3   False
3  A_4    True
4  A_5    True
```