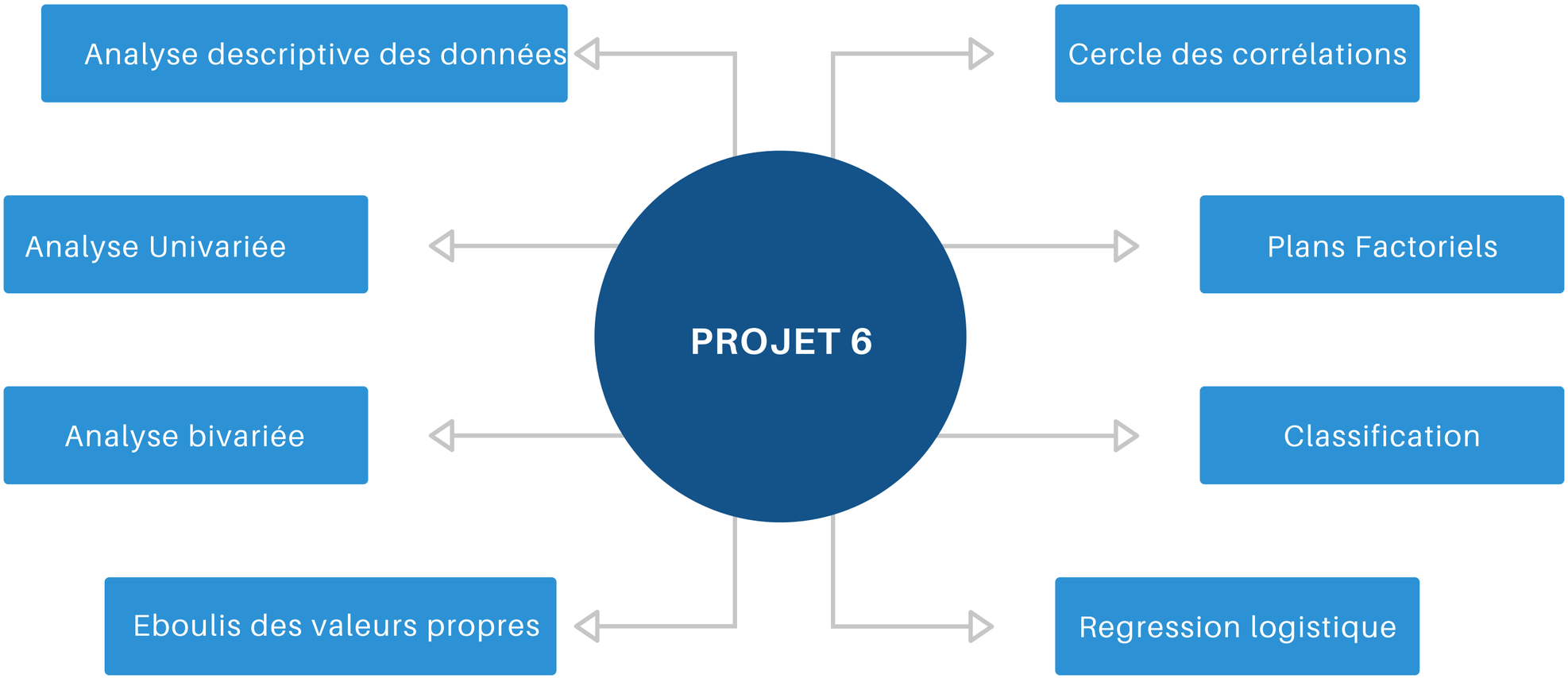
DETECTION DE FAUX BILLETS

Présentateur:
M Cheikhou **FOFANA**
Data Analyst chez
OpenClassrooms

Chef de Projet:
M Alioune Nar **SAMBE**

Date: 20 Octobre 2021

SOMMAIRE


Analyse descriptive des données
Cercle des corrélations
Analyse Univariée
PROJET 6
Plans Factoriels
Analyse bivariée
Classification
Eboulis des valeurs propres
Regression logistique

Analyse descrptive des données

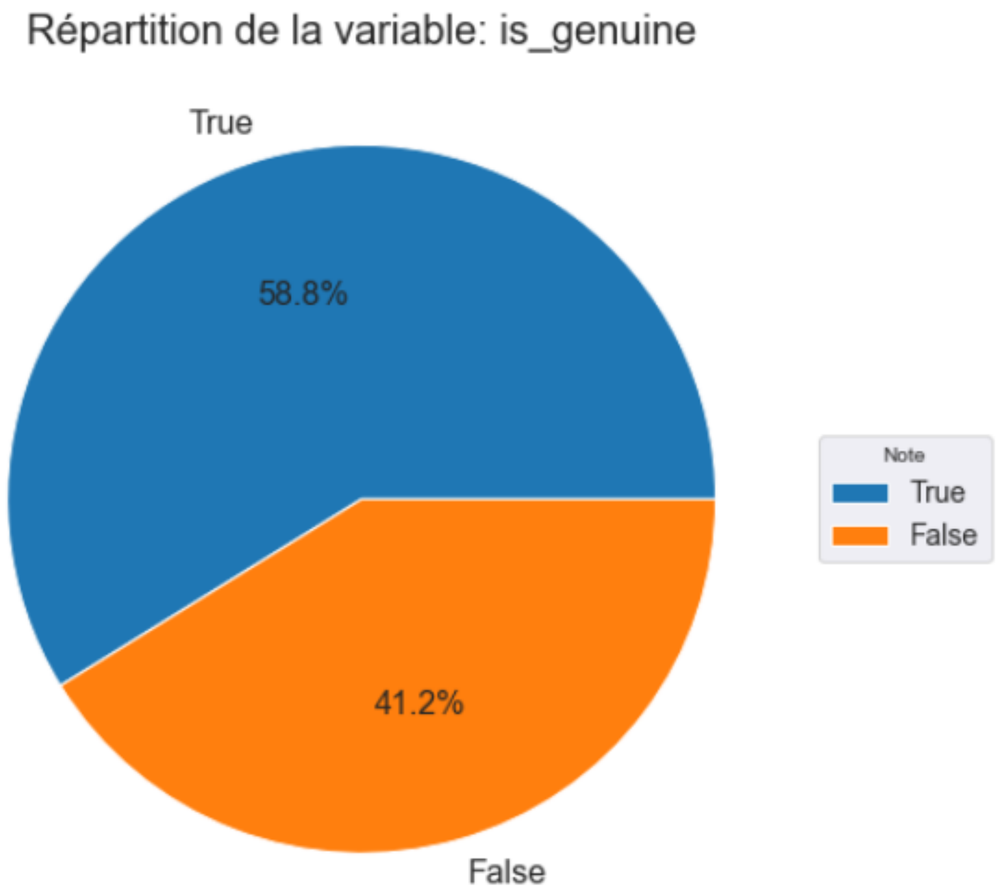
I Analyse de Forme:

- **Target:** is_genuine
- **Nombres de lignes et de colonnes:**170, 7
- **Types de variables:** quantitatives: 6, qualitative: 1
- **Valeurs manquantes:** 0
- **Lignes dupliquées:** 0

Analyse univariées

II Analyse de Fond:

1°) Visualisationde la target:

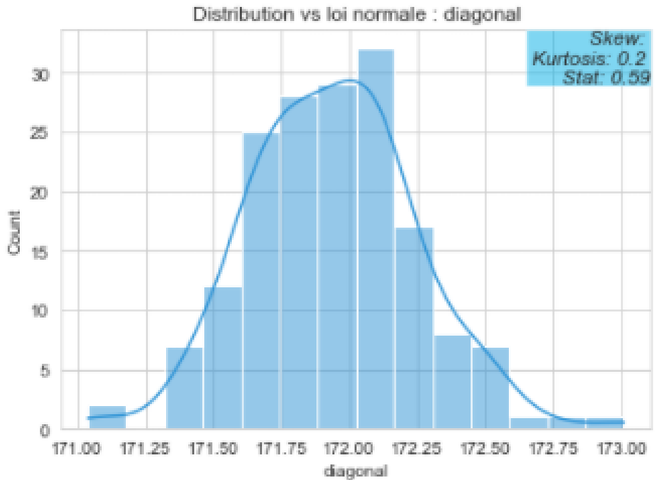


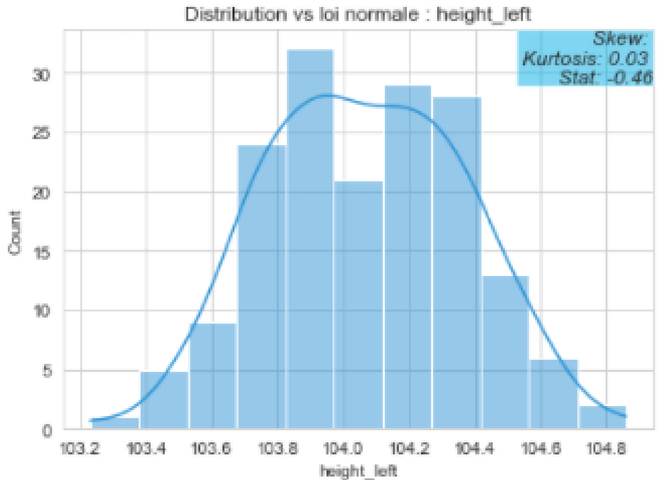
Analyse univariée

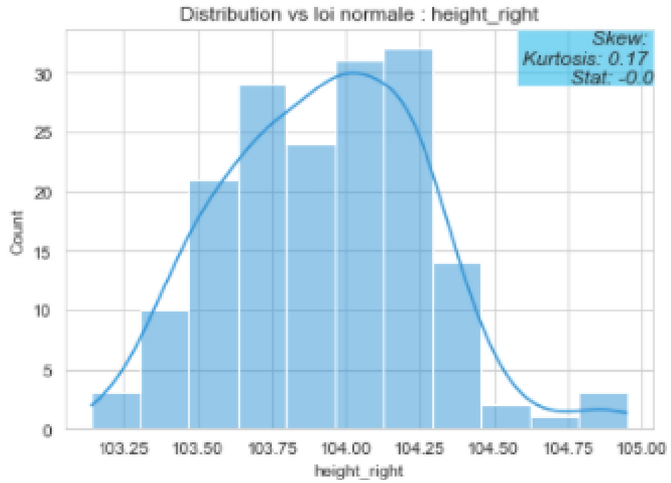
II Analyse de Fond:

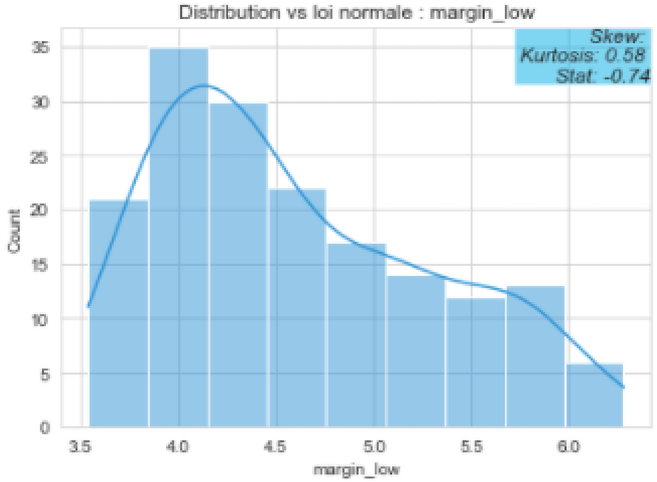
2°) Signification des variables:

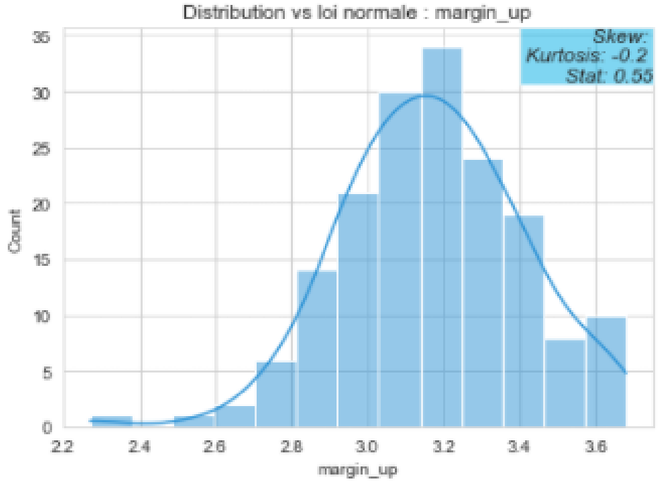
- Variables continues non standardisées;
- Variables catégorique: True, False

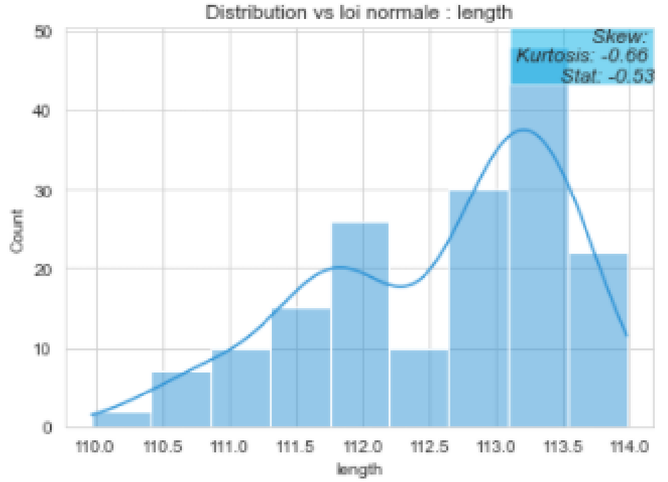








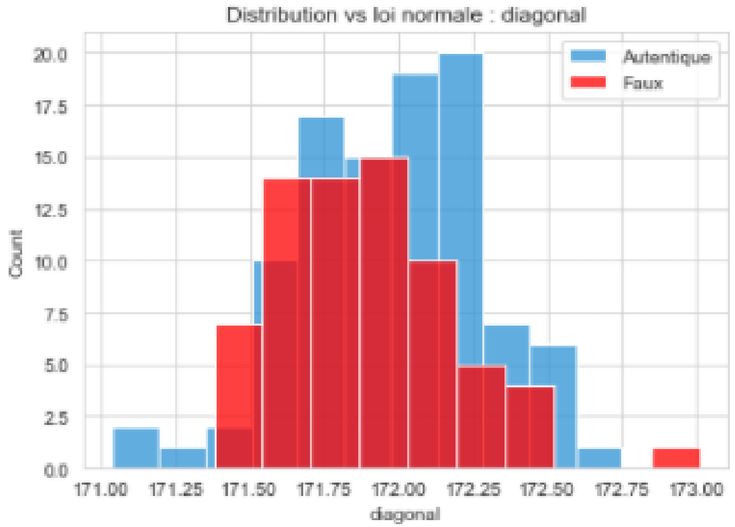


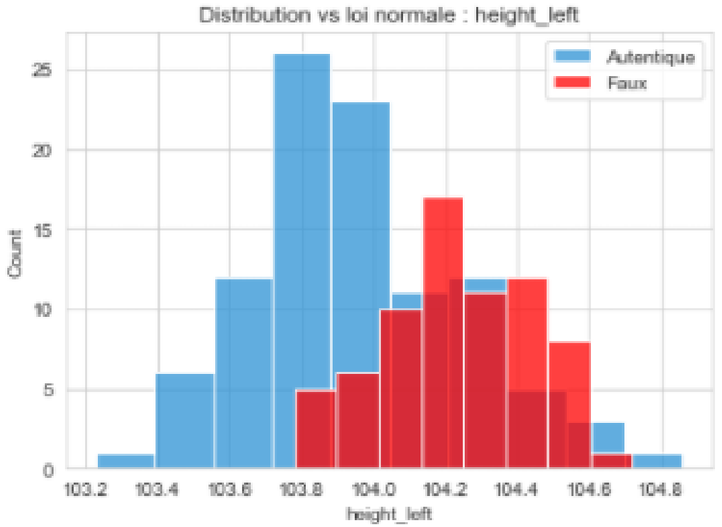


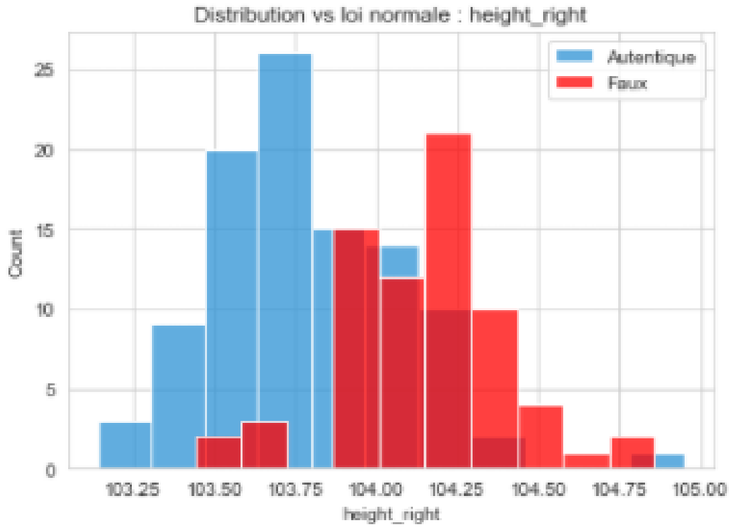
Analyse bivariées

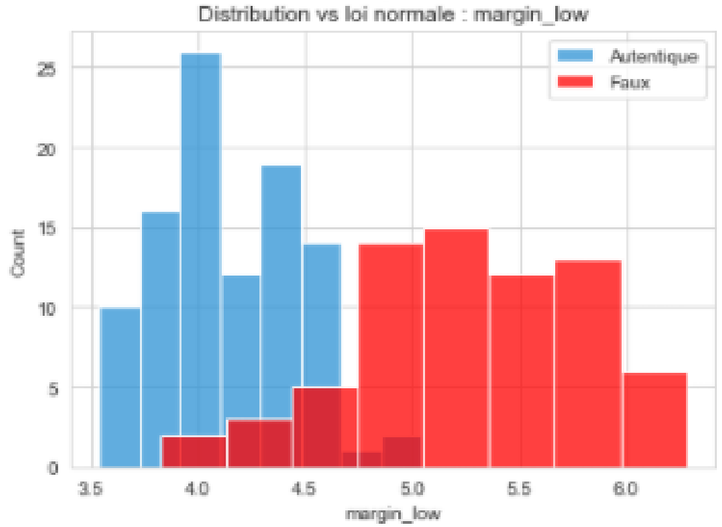
II Analyse de Fond:

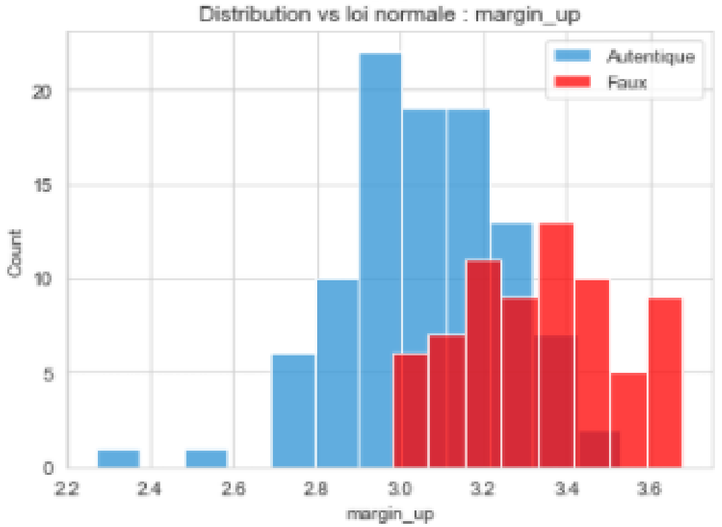
3°) Relation Target / Variables:

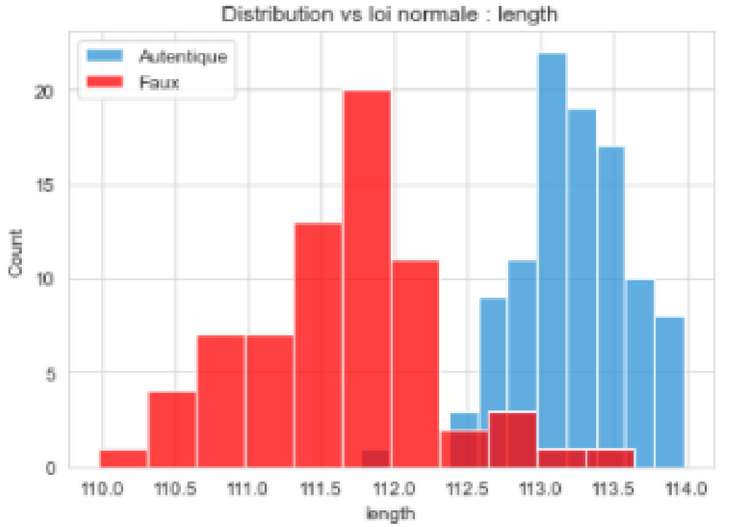












Analyse bivariées

II Analyse de Fond:

4°) Test de Student:

Test de student

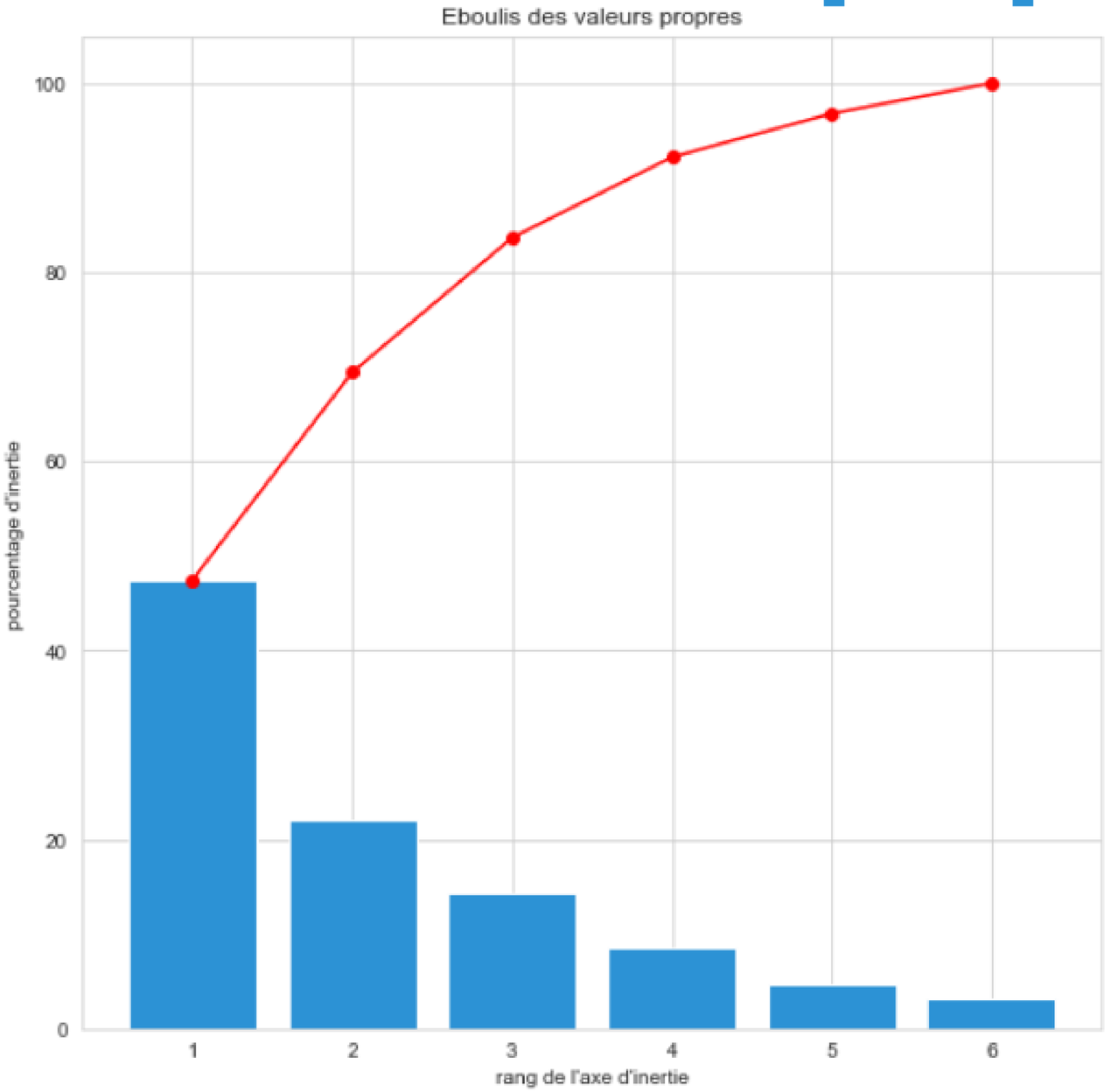
```
1  from scipy.stats import ttest_ind
```

```
1  def t_test(col):
2      from scipy.stats import ttest_ind
3      alpha = 0.05
4      stat, p_val = ttest_ind(df_true.sample(df_false.shape[0])[col], df_false[col])
5      if p_val < alpha:
6          return 'Cette variable influe sur le statut du billet !!!'
7      else:
8          return 'Cette variable n\'a aucune influence sur le statut du billet !!!'
9
```

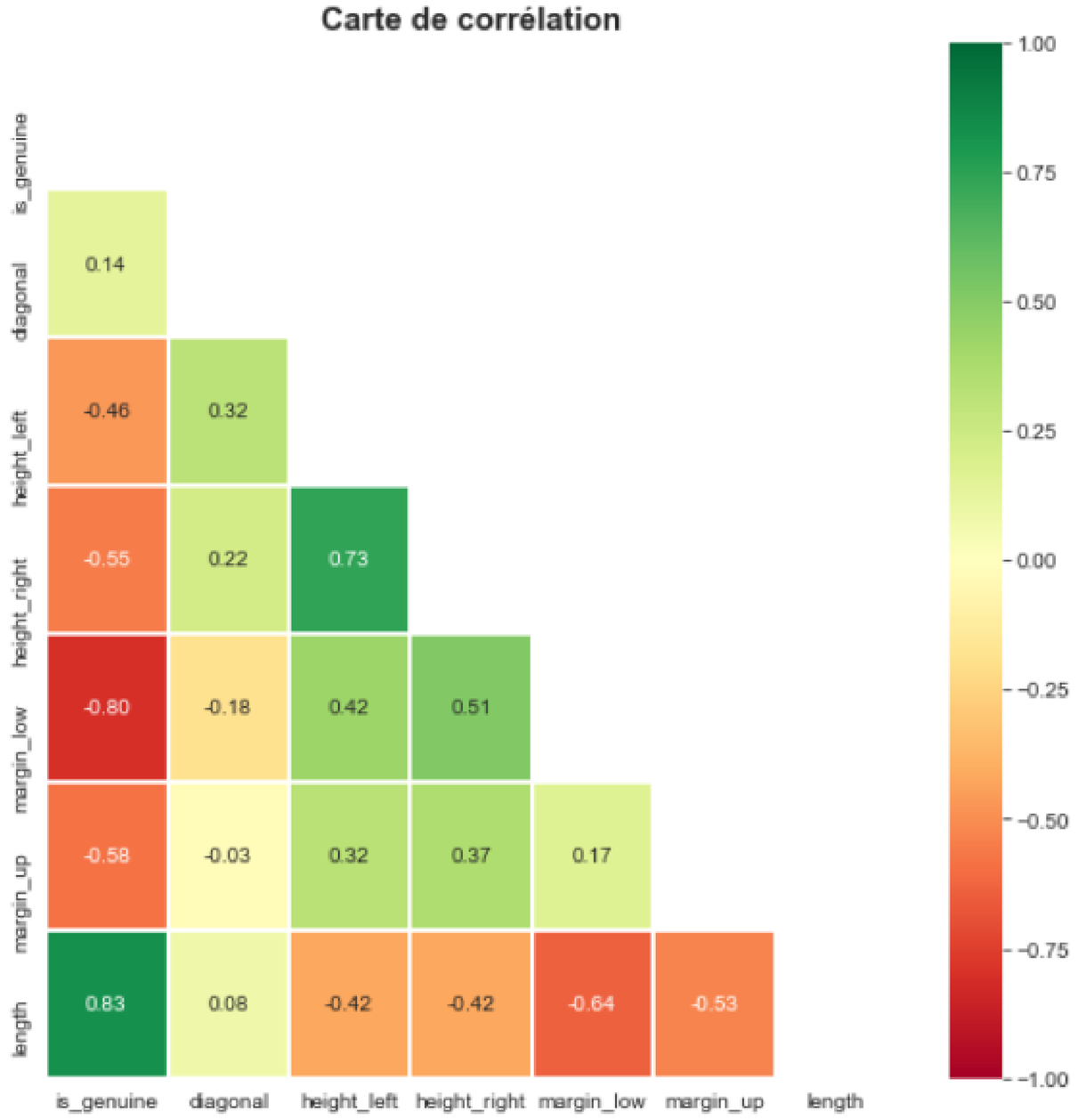
```
1  for col in df.select_dtypes('float'):
2      print(f'{col :-<50} {t_test(col)}')
```

```
diagonal---------------------------------------- Cette variable n'a aucune influence sur le statut du billet !!!
height_left------------------------------------- Cette variable influe sur le statut du billet !!!
height_right------------------------------------ Cette variable influe sur le statut du billet !!!
margin_low-------------------------------------- Cette variable influe sur le statut du billet !!!
margin_up--------------------------------------- Cette variable influe sur le statut du billet !!!
length------------------------------------------ Cette variable influe sur le statut du billet !!!
```

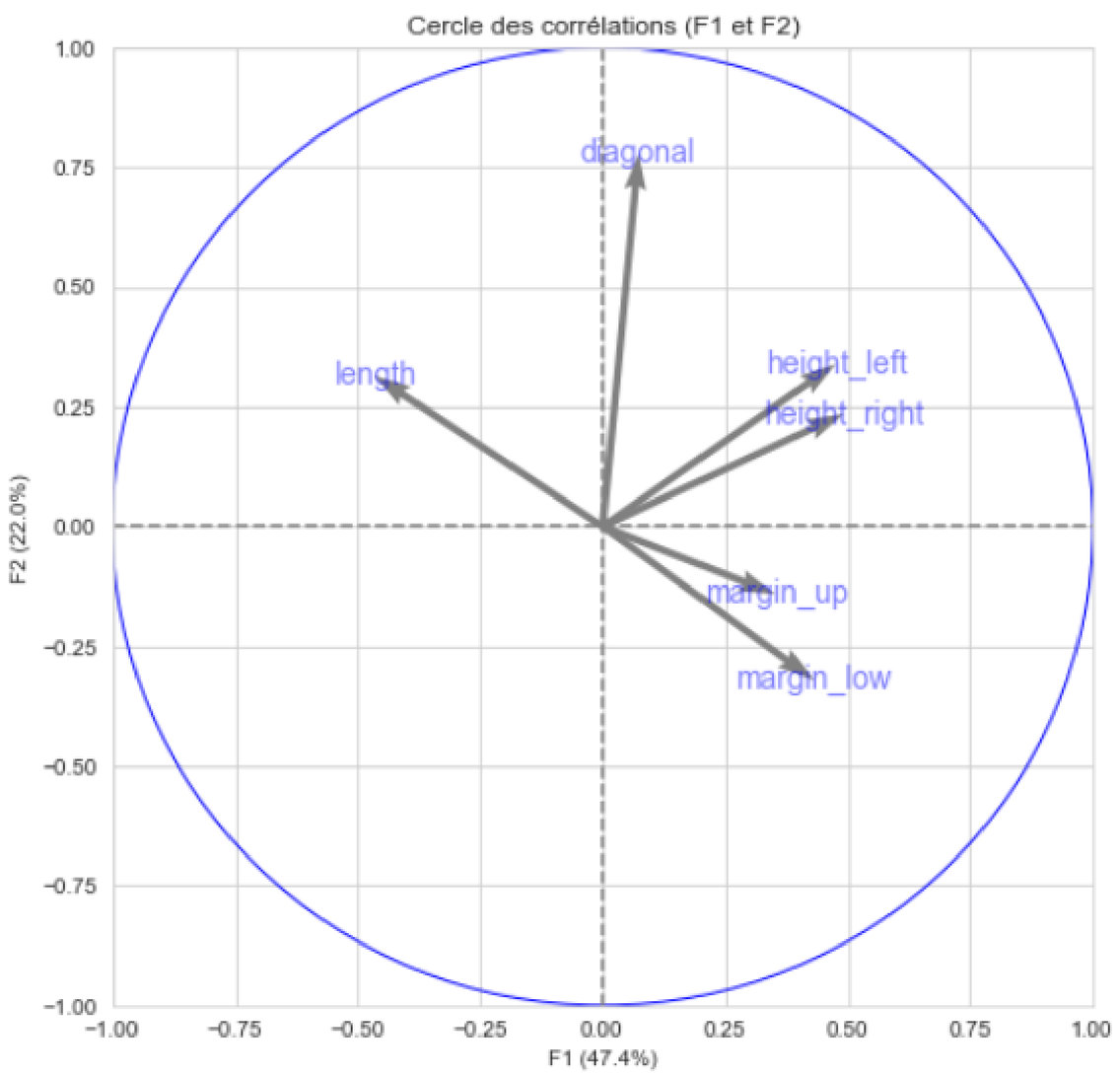
Eboulis des valeurs propres

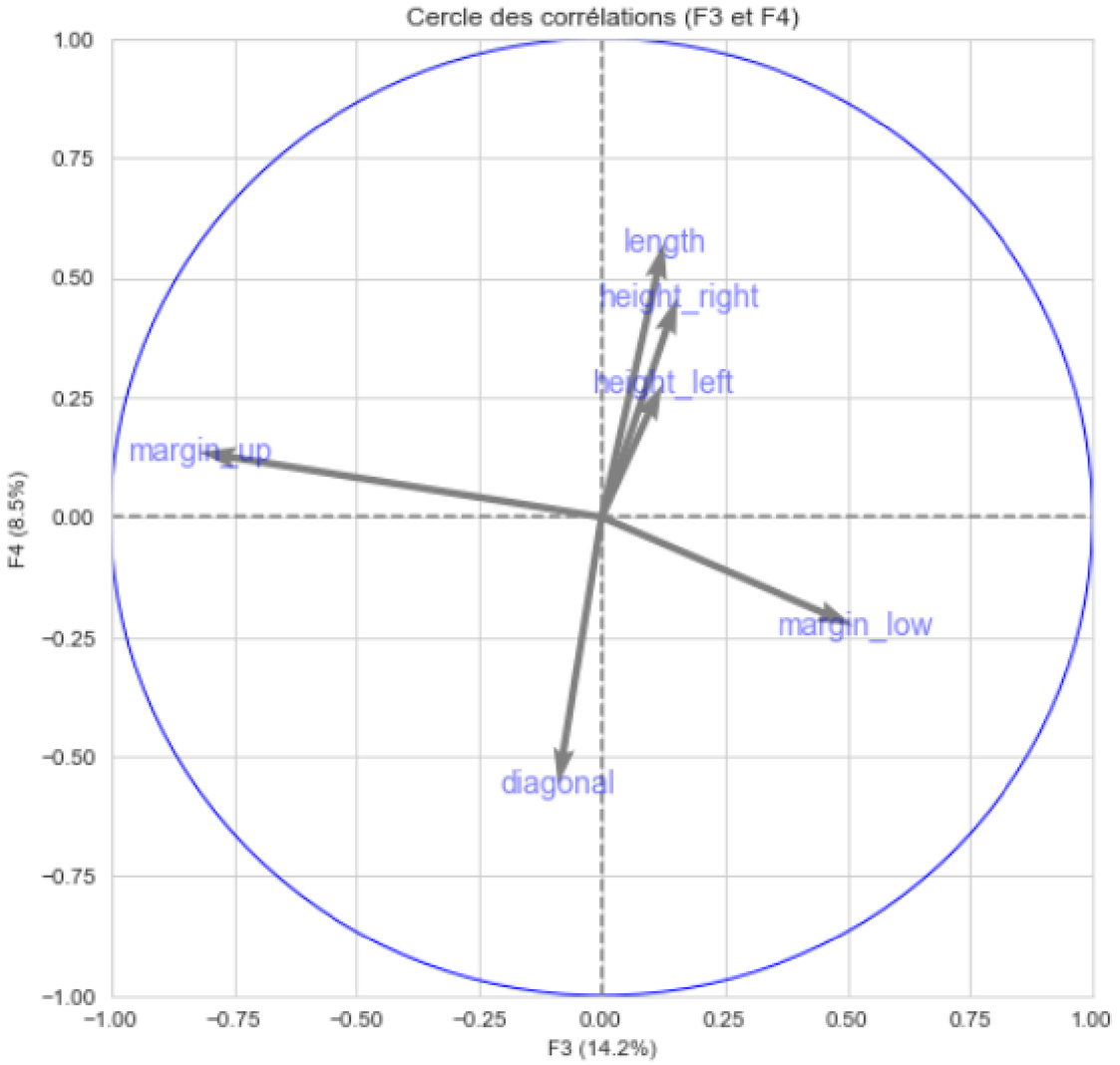


Carte des Corrélations

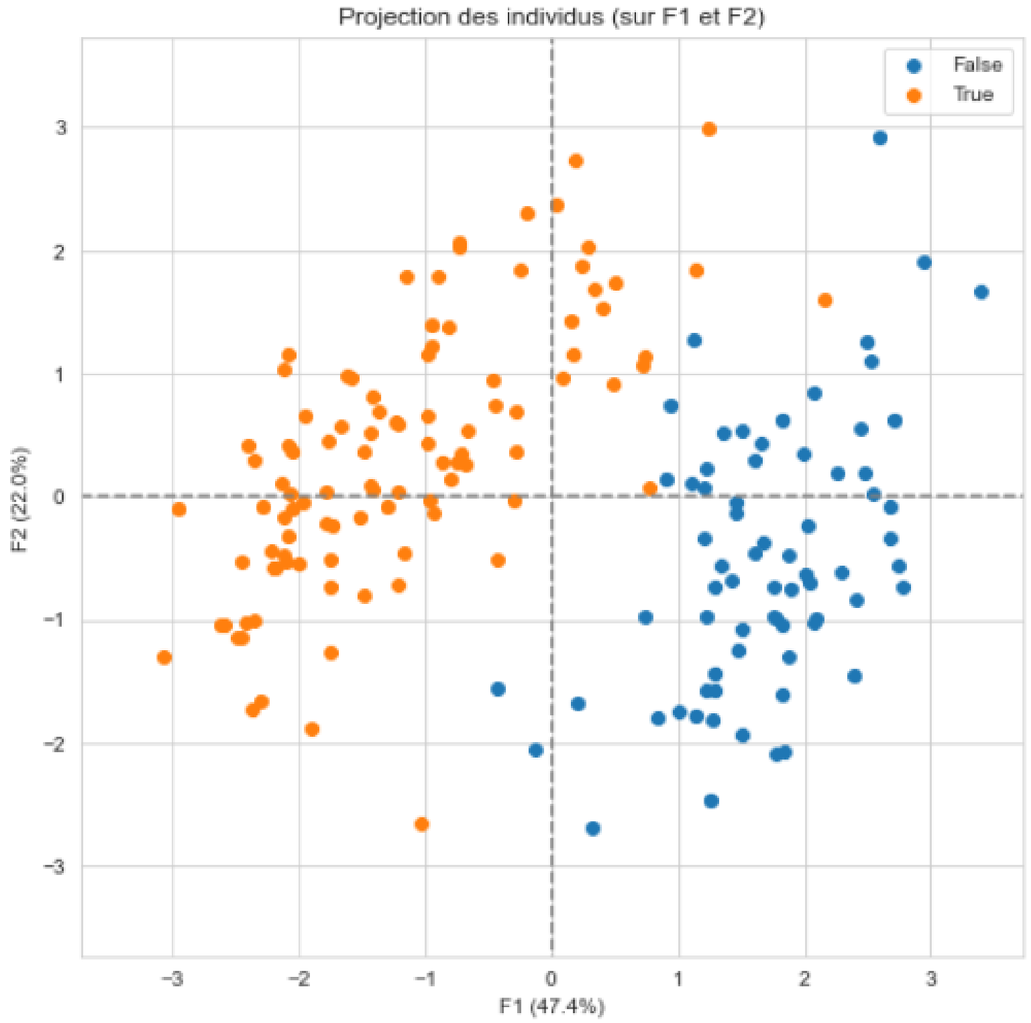


Cercle des Corrélations

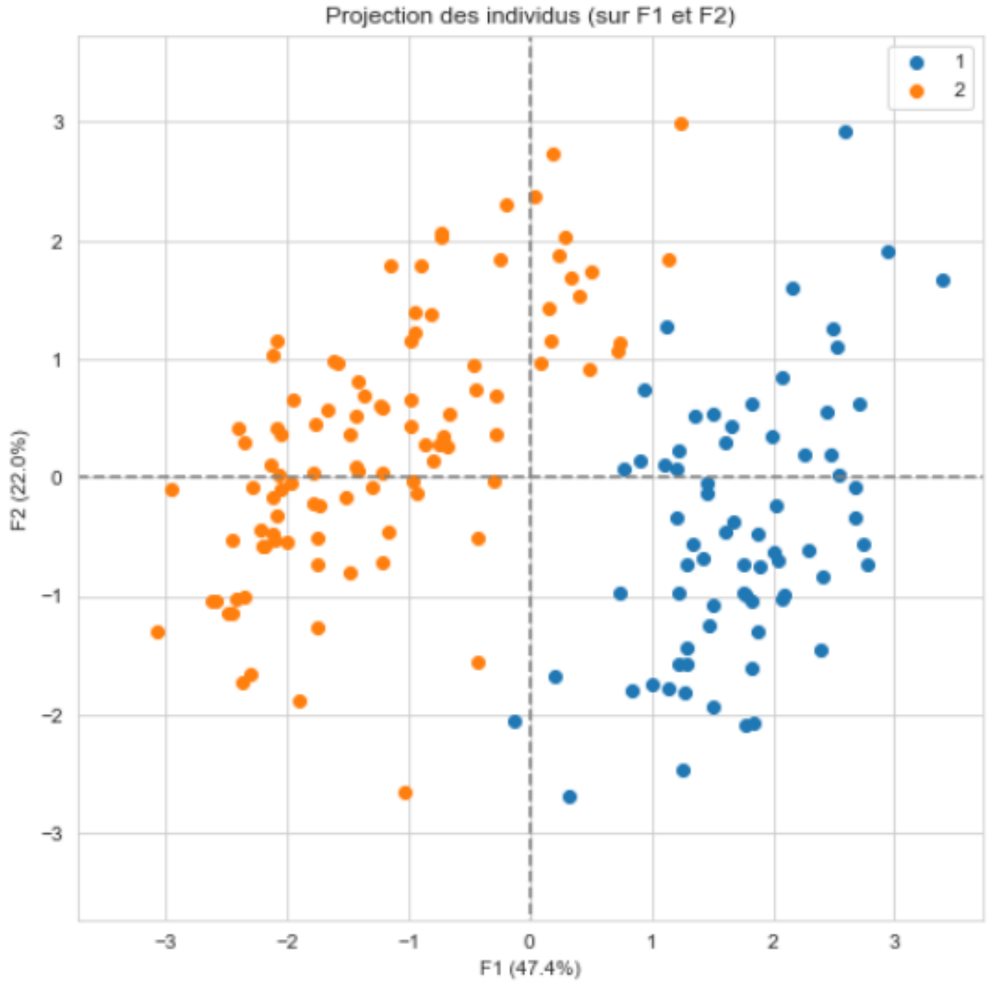




Plan Factoriel

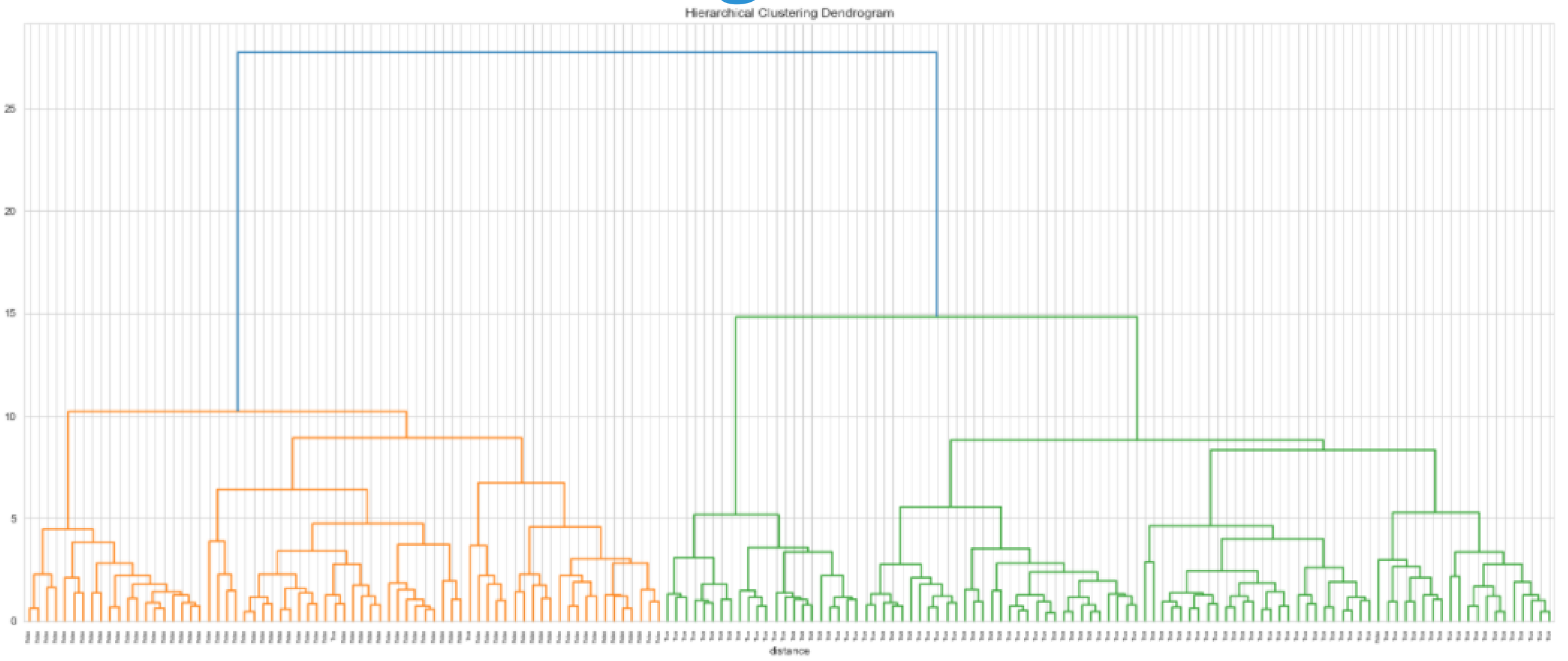


Plan Factoriel selon le **statut** du billet



Plan Factoriel selon le **cluster**

Dendrogramme



Régression logistique

1°) Encodage

Encodage

```python
def encodage(df):
    code = {True :1, False :0}
    for col in df.select_dtypes('bool').columns:
        df.loc[:, col] = df[col].map(code)
    return df
```

```python
def preprocessing(df):

    df = encodage(df)

    X = df.drop('is_genuine', axis =1)
    y = df['is_genuine']

    print(y.value_counts())

    return X, y
```

Régression logistique

2°) Scinder le dataset

TrainTest - Nettoyage - Encodage

```
1  from sklearn.model_selection import train_test_split
2
3  trainset, testset = train_test_split(df, test_size=0.3, random_state=100)
```

```
1  trainset['is_genuine'].value_counts()
```

```
True     71
False    48
Name: is_genuine, dtype: int64
```

```
1  testset['is_genuine'].value_counts()
```

```
True     29
False    22
Name: is_genuine, dtype: int64
```

```
1  X_train, y_train = preprocessing(trainset)
```

```
1    71
0    48
Name: is_genuine, dtype: int64
```

```
1  X_test, y_test = preprocessing(testset)
```

```
1    29
0    22
Name: is_genuine, dtype: int64
```

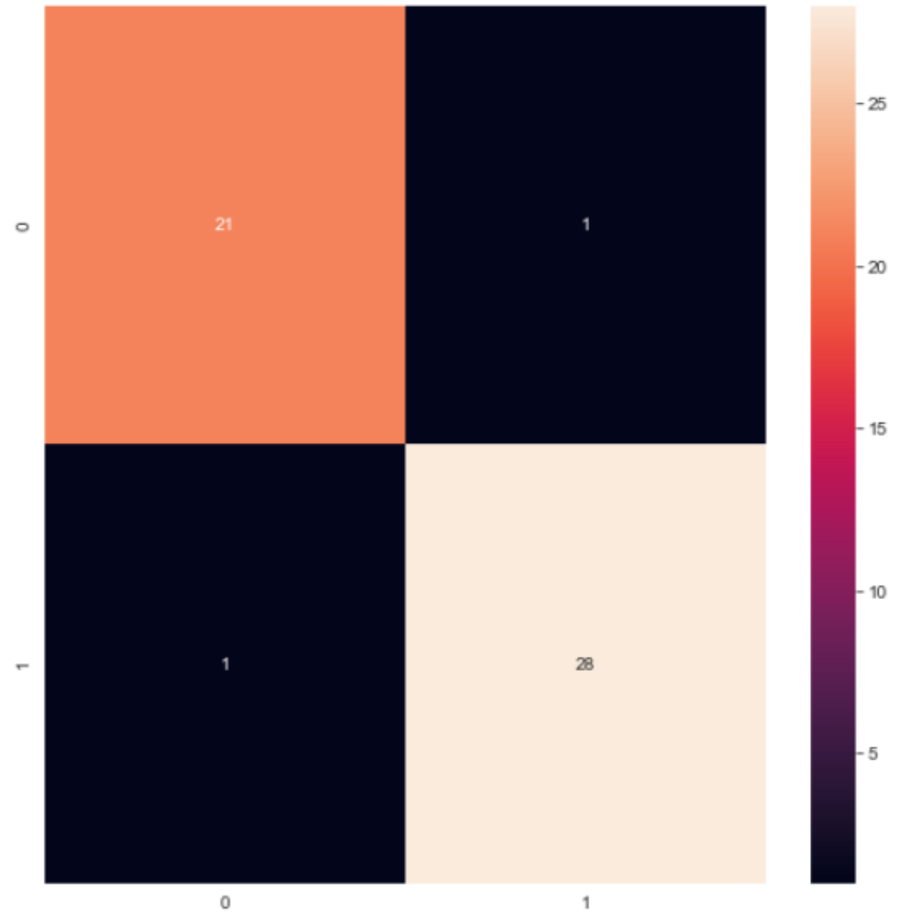
Régression logistique

3°) Création du modèle

Création du modèle

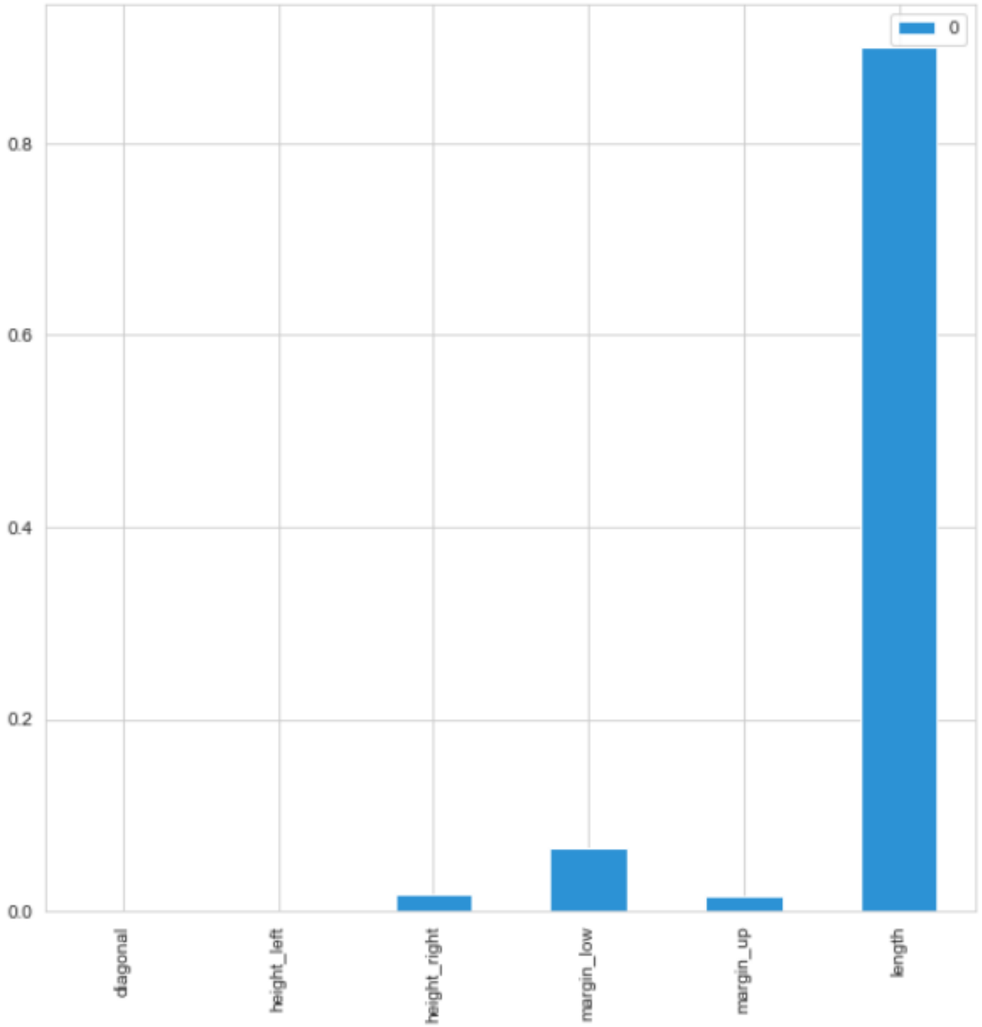
```
1  model = LogisticRegression(random_state=100)
2  model.fit(X_train, y_train)
```

	precision	recall	f1-score	support
0	0.95	0.95	0.95	22
1	0.97	0.97	0.97	29
accuracy			0.96	51
macro avg	0.96	0.96	0.96	51
weighted avg	0.96	0.96	0.96	51



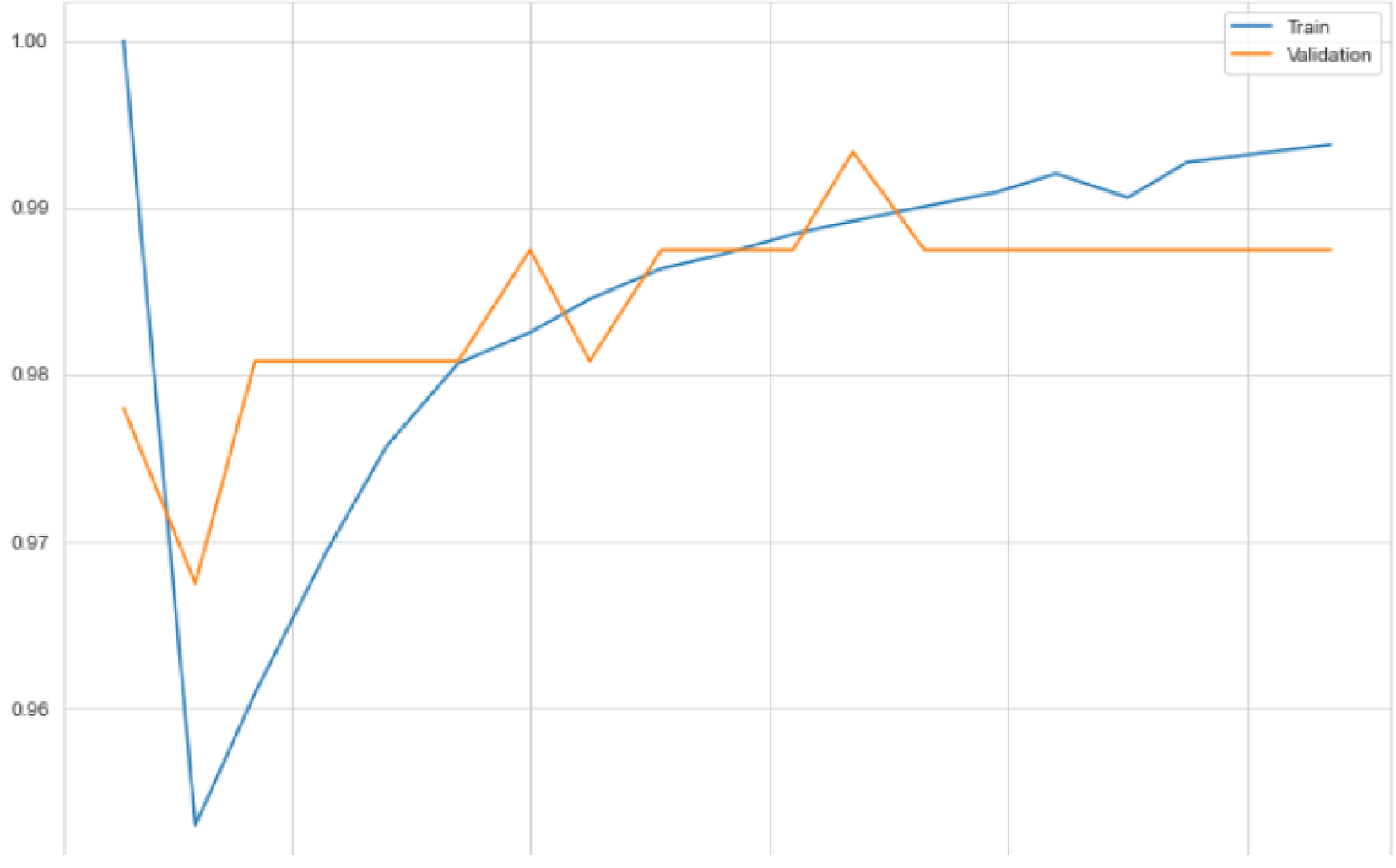
Régression logistique

4°) Importance de l'influence des variables sur la target:



Régression logistique

5°) Learning Curve



Régression logistique

6°) Optimisation du modèle

Optimisation du modèle

```
# Importation de RandomizedSearchCV
from sklearn.model_selection import RandomizedSearchCV
```

```
# Création du preprocessor du pipeline
preprocessor = make_pipeline(PolynomialFeatures(2), SelectKBest(f_classif, k = 20))
```

```
# Choix des paramètres à soumettre à RandomizedSearchCV
params = {
    'pipeline__polynomialfeatures__degree': [2, 3, 4],
    'pipeline__polynomialfeatures__order': ['C', 'F'],
    'pipeline__polynomialfeatures__interaction_only': ['True','False'],
    'pipeline__polynomialfeatures__include_bias': ['False','True'],
    'logisticregression__solver': ["lbfgs", "sag", "saga"],
    'logisticregression__max_iter': [100, 300, 500, 1000],
    'logisticregression__penalty': ['l1', 'l2', 'elasticnet'],
    'pipeline__selectkbest__k': range(40, 60)
}
```

```
# Instanciation du pipeline
LRG = make_pipeline(preprocessor, StandardScaler(), PCA(),
                    LogisticRegression(random_state=0))
```

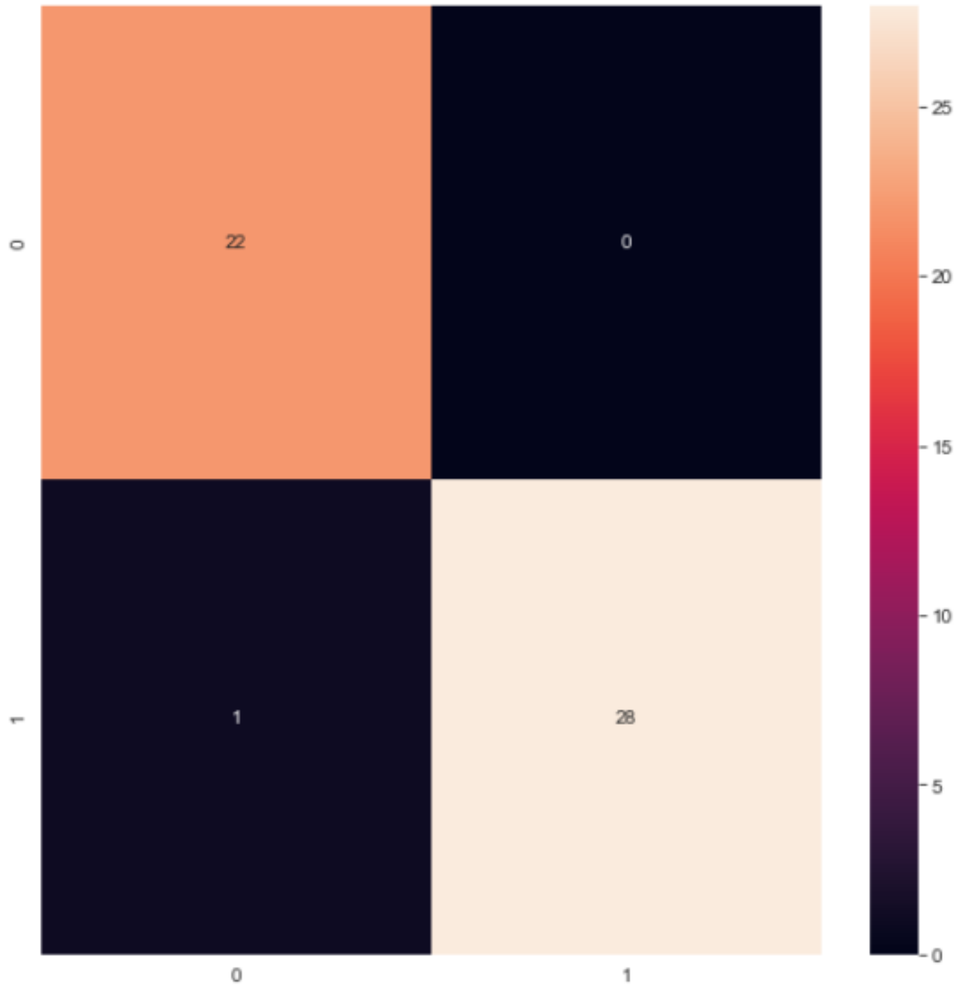
Régression logistique

7°) Création du modèle optimisé

```
#Création du modèle avec RandomizedSearchCV
model_optimized = RandomizedSearchCV(LRG, params, scoring='recall', cv=4, n_iter=40)

model_optimized.fit(X_train, y_train)

y_pred = model_optimized.predict(X_test)

print(classification_report(y_test, y_pred))
```

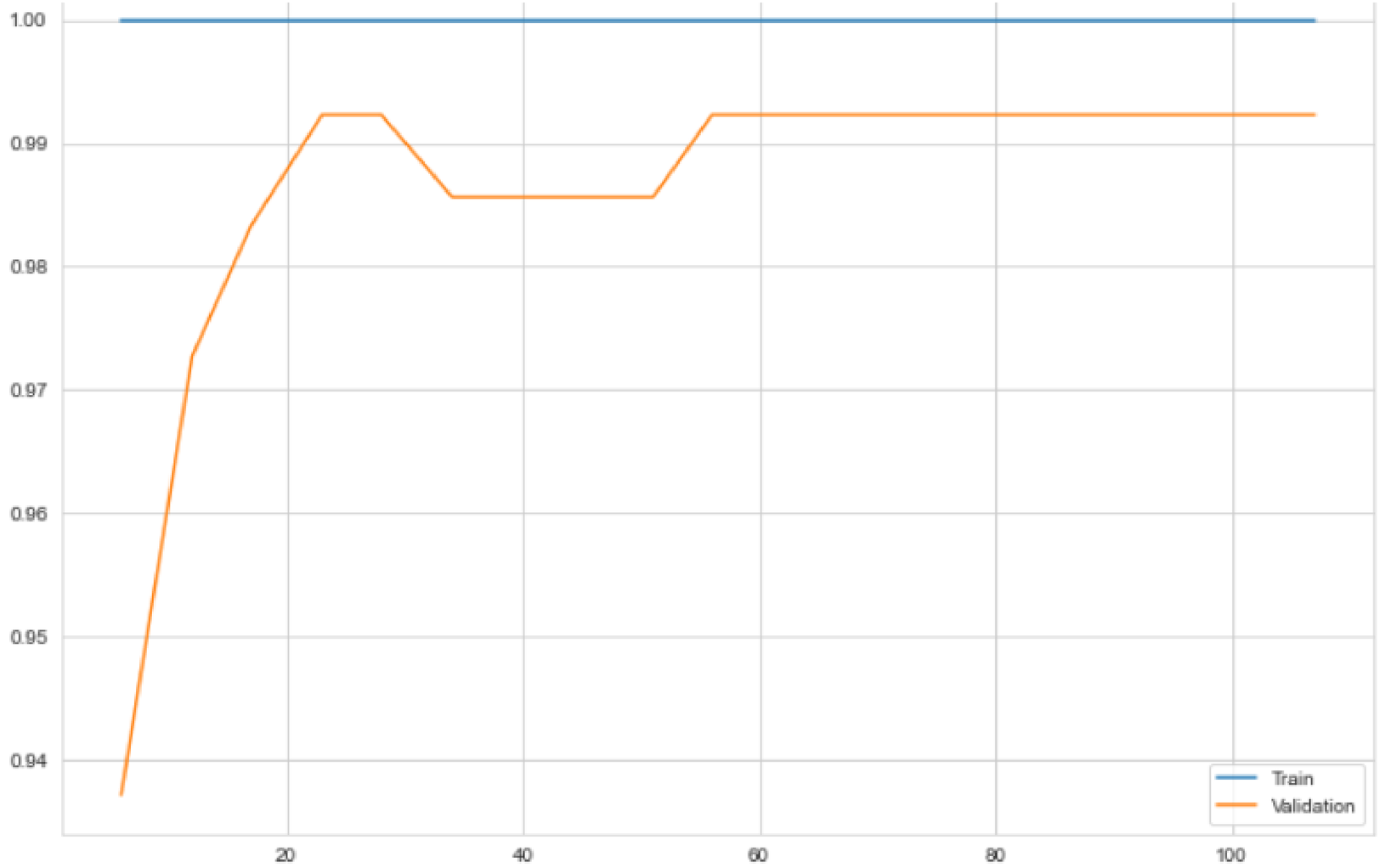
Régression logistique

7°) Création du modèle optimisé



Régression logistique

8°) Learning Curve



FIN

MERCI !!!