

CC2 Analyse des données et introduction au machine learning

Souleiman ALDJ, Arnaud BUCCAFURI et Solène GAILLOT.

Nous commençons par importer les différentes bibliothèques python que nous allons utiliser tout au long du Notebook.

```
import numpy as np  
import pandas as pd
```

```
from sklearn.impute import SimpleImputer  
from sklearn.preprocessing import OneHotEncoder  
from sklearn.compose import ColumnTransformer  
from sklearn.model_selection import train_test_split,cross_val_score  
from sklearn.pipeline import Pipeline  
from sklearn.preprocessing import LabelEncoder  
import optuna  
  
import xgboost as xgb  
from xgboost import XGBClassifier  
  
import os  
for dirname, _, filenames in os.walk('/kaggle/input'):  
    for filename in filenames:  
        print(os.path.join(dirname, filename))  
/usr/local/lib/python3.12/dist-packages/sqlalchemy/orm/query.py:195: SyntaxWarning: "is not" with 'tuple' literal. Did you mean "!="?  
    if entities is not ():  
/kaggle/input/master-econometrie-et-statistiques/sample_submission.csv  
/kaggle/input/master-econometrie-et-statistiques/train.csv  
/kaggle/input/master-econometrie-et-statistiques/test.csv
```

Nous continuons par nettoyer les données de la base "train". Ce qui nous servira ensuite de support pour nettoyer la base de données "test".

```
def wrangle(chemin_fichier: str):
```

```
    """
```

- Transformer la base de données en type pandas dataframe.

- Nettoyage des variables qualitatives : Créer une catégorie "Missing" pour les valeurs manquantes et aberrantes.

- Nettoyage des variables continues : Créer une variable binaire qui prend 1 si la valeur est manquante. Clipper les valeurs non manquantes aux quantiles 0.01 et 0.99. Appliquer la fonction logarithme sur certaines variables.

- Retourne la base de données nettoyée au type pandas dataframe.

```
    """
```

```
df = pd.read_csv(chemin_fichier)
```

```
#"Drug":  
df['Drug'] = df['Drug'].fillna('Missing')  
K_drugs = ['Placebo', 'D-penicillamine', 'Missing']  
df.loc[~df['Drug'].isin(K_drugs), 'Drug'] = 'Missing'
```

```
#"Sex":
```

```

df['Sex'] = df['Sex'].fillna('Missing')
K_sex = ['F', 'H', 'Missing']
df.loc[~df['Sex'].isin(K_sex), 'Sex'] = 'Missing'

#"Ascites":
df['Ascites'] = df['Ascites'].fillna('Missing')
K_ascites = ['N', 'Y', 'Missing']
df.loc[~df['Ascites'].isin(K_ascites), 'Ascites'] = 'Missing'

#"Hepatomegaly":
df['Hepatomegaly'] = df['Hepatomegaly'].fillna('Missing')
K_hepatomegaly = ['N', 'Y', 'Missing']
df.loc[~df['Hepatomegaly'].isin(K_hepatomegaly), 'Hepatomegaly'] = 'Missing'

#"Spiders":
df['Spiders'] = df['Spiders'].fillna('Missing')
K_spiders = ['N', 'Y', 'Missing']
df.loc[~df['Spiders'].isin(K_spiders), 'Spiders'] = 'Missing'

#"Edema":
df['Edema'] = df['Edema'].fillna('Missing')
K_edema = ['N', 'Y', 'S','Missing']
df.loc[~df['Edema'].isin(K_edema), 'Edema'] = 'Missing'

#"Stage":
df['Stage'] = df['Stage'].fillna('Missing')
df['Stage'] = df['Stage'].astype(str)
K_stage = ['1', '2', '3','4', 'Missing']
df.loc[~df['Stage'].isin(K_stage), 'Stage'] = 'Missing'

#"N_Days":
df["N_Days_missing"] = df['N_Days'].isna().astype(int)

quantile_bas = 77
quantile_haut = 4500
quantiles = df['N_Days'].notna()
df.loc[quantiles, 'N_Days'] = df.loc[quantiles, 'N_Days'].clip(quantile_bas, quantile_haut)

#"Age":
df["Age_missing"] = df['Age'].isna().astype(int)

quantile_bas = 11773
quantile_haut = 25899
quantiles = df['Age'].notna()
df.loc[quantiles, 'Age'] = df.loc[quantiles, 'Age'].clip(quantile_bas, quantile_haut)

#"Bilirubin":
df["Bilirubin_missing"] = df['Bilirubin'].isna().astype(int)

quantile_bas = 0.4
quantile_haut = 16.2
quantiles = df['Bilirubin'].notna()
df.loc[quantiles, 'Bilirubin'] = df.loc[quantiles, 'Bilirubin'].clip(quantile_bas, quantile_haut)

```

```

df.loc[quantiles, 'Bilirubin_log'] = np.log(df.loc[quantiles, 'Bilirubin'])
df = df.drop(columns = "Bilirubin")

#"Cholesterol":
df["Cholesterol_missing"] = df['Cholesterol'].isna().astype(int)

quantile_bas = 132
quantile_haut = 1276
quantiles = df['Cholesterol'].notna()
df.loc[quantiles, 'Cholesterol'] = df.loc[quantiles, 'Cholesterol'].clip(quantile_bas, quantile_haut)

df.loc[quantiles, 'Cholesterol_log'] = np.log(df.loc[quantiles, 'Cholesterol'])
df = df.drop(columns = "Cholesterol")

#"Albumin":
df["Albumin_missing"] = df['Albumin'].isna().astype(int)

quantile_bas = 2.48
quantile_haut = 4.38
quantiles = df['Albumin'].notna()
df.loc[quantiles, 'Albumin'] = df.loc[quantiles, 'Albumin'].clip(quantile_bas, quantile_haut)

#"Copper":
df["Copper_missing"] = df['Copper'].isna().astype(int)

quantile_bas = 10
quantile_haut = 444
quantiles = df['Copper'].notna()
df.loc[quantiles, 'Copper'] = df.loc[quantiles, 'Copper'].clip(quantile_bas, quantile_haut)

df.loc[quantiles, 'Copper_log'] = np.log(df.loc[quantiles, 'Copper'])
df = df.drop(columns = "Copper")

#"Alk_Phosphat":
df["Alk_Phosphat_missing"] = df['Alk_Phosphat'].isna().astype(int)

quantile_bas = 377
quantile_haut = 9933.2
quantiles = df['Alk_Phosphat'].notna()
df.loc[quantiles, 'Alk_Phosphat'] = df.loc[quantiles, 'Alk_Phosphat'].clip(quantile_bas, quantile_haut)

df.loc[quantiles, 'Alk_Phosphat_log'] = np.log(df.loc[quantiles, 'Alk_Phosphat'])
df = df.drop(columns = "Alk_Phosphat")

#"SGOT":
df["SGOT_missing"] = df['SGOT'].isna().astype(int)

quantile_bas = 41.85
quantile_haut = 299.15
quantiles = df['SGOT'].notna()
df.loc[quantiles, 'SGOT'] = df.loc[quantiles, 'SGOT'].clip(quantile_bas, quantile_haut)

df.loc[quantiles, 'SGOT_log'] = np.log(df.loc[quantiles, 'SGOT'])

```

```

df = df.drop(columns = "SGOT")

#"Tryglicerides":
df["Tryglicerides_missing"] = df["Tryglicerides"].isna().astype(int)

quantile_bas = 46
quantile_haut = 272
quantiles = df['Tryglicerides'].notna()
df.loc[quantiles, 'Tryglicerides'] = df.loc[quantiles, 'Tryglicerides'].clip(quantile_bas, quantile_haut)

df.loc[quantiles, 'Tryglicerides_log'] = np.log(df.loc[quantiles, 'Tryglicerides'])
df = df.drop(columns = "Tryglicerides")

#"Platelets":
df["Platelets_missing"] = df["Platelets"].isna().astype(int)

quantile_bas = 80
quantile_haut = 514
quantiles = df['Platelets'].notna()
df.loc[quantiles, 'Platelets'] = df.loc[quantiles, 'Platelets'].clip(quantile_bas, quantile_haut)

#"Prothrombin":
df["Prothrombin_missing"] = df["Prothrombin"].isna().astype(int)

quantile_bas = 9.5
quantile_haut = 13.2
quantiles = df['Prothrombin'].notna()
df.loc[quantiles, 'Prothrombin'] = df.loc[quantiles, 'Prothrombin'].clip(quantile_bas, quantile_haut)

return df

```

Nous Séparons la base de données "train" entre les variables explicatives et expliquée. Puis nous encodons les catégories de la variable expliquée.

```

df = wrangle("/kaggle/input/master-econometrie-et-statistiques/train.csv")

```

```

var_exp = "Status"
X = df.drop(columns=[var_exp, 'id'])
y = df[var_exp]

le = LabelEncoder()
y_enc = le.fit_transform(y)

```

Nous remplaçons, pour les variables continues, les valeurs manquantes par la moyenne des autres valeurs. Et, pour les variables qualitatives, nous encodons les catégories.

```

var_num = X.select_dtypes(include=['int64', 'float64']).columns
var_qual = X.select_dtypes(include=['object', 'category']).columns

```

```

num_imputer = SimpleImputer(strategy='mean')
qual_encoder = OneHotEncoder(handle_unknown='ignore')

```

```

preprocessor = ColumnTransformer(
    transformers=[
        ('num', num_imputer, var_num),
        ('qual', qual_encoder, var_qual)
    ]
)

```

```

        )
Nous décidons d'utiliser un modèle de GradientBoosting plus précisément XGBoost. Nous
utilisons la bibliothèque Optuna, pour choisir de manière optimale les meilleurs paramètres
possibles.
def objective(trial):
    model = XGBClassifier(
        n_estimators=800,
        max_depth=trial.suggest_int("max_depth", 3, 10),
        min_child_weight=trial.suggest_int("min_child_weight", 1, 10),
        gamma=trial.suggest_float("gamma", 0.0, 5.0),
        learning_rate=trial.suggest_float("learning_rate", 0.01, 0.3, log=True),
        subsample=trial.suggest_float("subsample", 0.6, 1.0),
        colsample_bytree=trial.suggest_float("colsample_bytree", 0.6, 1.0),
        reg_alpha=trial.suggest_float("reg_alpha", 0.0, 10.0),
        reg_lambda=trial.suggest_float("reg_lambda", 0.1, 10.0),
        eval_metric="logloss",
        random_state=42,
        n_jobs=-1
    )

    pipeline = Pipeline([
        ("preprocessor", preprocessor),
        ("model", model)
    ])

    score = cross_val_score(
        pipeline,
        X,
        y_enc,
        cv=5,
        scoring="neg_log_loss",
        n_jobs=-1
    ).mean()

    return score

tuning = optuna.create_study(direction="maximize")
tuning.optimize(objective, n_trials=50)
[I 2026-01-15 21:49:58,947] A new study created in memory with name: no-name-e
25f3828-80ff-4d48-a923-697994c5f3c1
[I 2026-01-15 21:50:12,922] Trial 0 finished with value: -0.3886804472433882 a
nd parameters: {'max_depth': 7, 'min_child_weight': 9, 'gamma': 1.26254
06765479663, 'learning_rate': 0.0260217644807647, 'subsample': 0.624362
424508789, 'colsample_bytree': 0.8045221723741838, 'reg_alpha': 6.65435
7509327032, 'reg_lambda': 2.8968472579289464}. Best is trial 0 with val
ue: -0.3886804472433882.
[I 2026-01-15 21:50:20,359] Trial 1 finished with value: -0.3908119118198553 a
nd parameters: {'max_depth': 9, 'min_child_weight': 6, 'gamma': 3.06042
5826494715, 'learning_rate': 0.1101038896780815, 'subsample': 0.6647766
482975273, 'colsample_bytree': 0.6161857363705489, 'reg_alpha': 1.84286
26633514034, 'reg_lambda': 6.875740116417861}. Best is trial 0 with val
ue: -0.3886804472433882.
[I 2026-01-15 21:50:31,456] Trial 2 finished with value: -0.37338994901658873
and parameters: {'max_depth': 6, 'min_child_weight': 2, 'gamma': 0.5975

```

575753783213, 'learning_rate': 0.05849470022329364, 'subsample': 0.8275
065630486671, 'colsample_bytree': 0.7121096926372564, 'reg_alpha': 3.09
421047689201, 'reg_lambda': 2.331979710118583}. Best is trial 2 with va
lue: -0.37338994901658873.

[I 2026-01-15 21:50:41,508] Trial 3 finished with value: -0.37265681096683356
and parameters: {'max_depth': 3, 'min_child_weight': 4, 'gamma': 0.3543
50520749504, 'learning_rate': 0.1883113455972794, 'subsample': 0.862911
1369785152, 'colsample_bytree': 0.7818022821347483, 'reg_alpha': 4.2603
70538824551, 'reg_lambda': 4.812227148436427}. Best is trial 3 with val
ue: -0.37265681096683356.

[I 2026-01-15 21:50:53,757] Trial 4 finished with value: -0.39246938920747815
and parameters: {'max_depth': 9, 'min_child_weight': 7, 'gamma': 1.3799
559659301956, 'learning_rate': 0.019193133424926546, 'subsample': 0.785
4731138506541, 'colsample_bytree': 0.7830261990684816, 'reg_alpha': 7.4
82956493753737, 'reg_lambda': 5.463276547604166}. Best is trial 3 with
value: -0.37265681096683356.

[I 2026-01-15 21:51:00,631] Trial 5 finished with value: -0.40744713581944325
and parameters: {'max_depth': 3, 'min_child_weight': 9, 'gamma': 3.6668
00835047791, 'learning_rate': 0.06407809383605377, 'subsample': 0.98431
18052359505, 'colsample_bytree': 0.6182828581795313, 'reg_alpha': 5.480
39143911317, 'reg_lambda': 3.2053674894588005}. Best is trial 3 with va
lue: -0.37265681096683356.

[I 2026-01-15 21:51:08,023] Trial 6 finished with value: -0.3939770480543906 a
nd parameters: {'max_depth': 4, 'min_child_weight': 5, 'gamma': 1.54638
99524870066, 'learning_rate': 0.12100411001361333, 'subsample': 0.74185
84231399338, 'colsample_bytree': 0.9076426087647904, 'reg_alpha': 9.543
045991326565, 'reg_lambda': 4.160976955529648}. Best is trial 3 with va
lue: -0.37265681096683356.

[I 2026-01-15 21:51:19,303] Trial 7 finished with value: -0.3905095185936173 a
nd parameters: {'max_depth': 5, 'min_child_weight': 6, 'gamma': 2.21971
79746205293, 'learning_rate': 0.018776765803437657, 'subsample': 0.9618
998638854018, 'colsample_bytree': 0.9885468262943462, 'reg_alpha': 0.35
992884933765623, 'reg_lambda': 1.3466316724432748}. Best is trial 3 wit
h value: -0.37265681096683356.

[I 2026-01-15 21:51:32,111] Trial 8 finished with value: -0.39490511314616317
and parameters: {'max_depth': 6, 'min_child_weight': 7, 'gamma': 1.3117
086029104619, 'learning_rate': 0.01625890045248442, 'subsample': 0.8422
541376221566, 'colsample_bytree': 0.9413133045114273, 'reg_alpha': 8.72
870694170418, 'reg_lambda': 3.841176163399544}. Best is trial 3 with va
lue: -0.37265681096683356.

[I 2026-01-15 21:51:39,041] Trial 9 finished with value: -0.40447109079451044
and parameters: {'max_depth': 10, 'min_child_weight': 4, 'gamma': 4.946
124090460758, 'learning_rate': 0.27927468476808454, 'subsample': 0.6478
549610630818, 'colsample_bytree': 0.6349530907256382, 'reg_alpha': 4.12
2720612984666, 'reg_lambda': 7.021795909833408}. Best is trial 3 with v
alue: -0.37265681096683356.

[I 2026-01-15 21:51:48,360] Trial 10 finished with value: -0.37560878120188845
and parameters: {'max_depth': 3, 'min_child_weight': 1, 'gamma': 0.1094
6792910167968, 'learning_rate': 0.2823515985622814, 'subsample': 0.9013
858015697441, 'colsample_bytree': 0.8323288946222455, 'reg_alpha': 5.41

4649805631087, 'reg_lambda': 9.233439718847805}. Best is trial 3 with value: -0.37265681096683356.

[I 2026-01-15 21:52:09,806] Trial 11 finished with value: -0.37766591953207324 and parameters: {'max_depth': 7, 'min_child_weight': 2, 'gamma': 0.08590369207099069, 'learning_rate': 0.04630361831215987, 'subsample': 0.8705950705454188, 'colsample_bytree': 0.7246795885367169, 'reg_alpha': 3.0766785421533114, 'reg_lambda': 0.8531192666692817}. Best is trial 3 with value: -0.37265681096683356.

[I 2026-01-15 21:52:19,636] Trial 12 finished with value: -0.37445831135205054 and parameters: {'max_depth': 5, 'min_child_weight': 3, 'gamma': 0.5858228451745237, 'learning_rate': 0.12836934783459325, 'subsample': 0.7506642690913737, 'colsample_bytree': 0.7166729214128106, 'reg_alpha': 2.6015415864759035, 'reg_lambda': 2.1194780856820667}. Best is trial 3 with value: -0.37265681096683356.

[I 2026-01-15 21:52:29,335] Trial 13 finished with value: -0.37982677720734176 and parameters: {'max_depth': 5, 'min_child_weight': 3, 'gamma': 0.7212655478535347, 'learning_rate': 0.04326910911069326, 'subsample': 0.9159141613010713, 'colsample_bytree': 0.7107560328555227, 'reg_alpha': 4.441388597886524, 'reg_lambda': 5.815381340124637}. Best is trial 3 with value: -0.37265681096683356.

[I 2026-01-15 21:52:36,740] Trial 14 finished with value: -0.3843636235811713 and parameters: {'max_depth': 8, 'min_child_weight': 1, 'gamma': 2.2269019583221676, 'learning_rate': 0.1775432432540394, 'subsample': 0.8239826680726947, 'colsample_bytree': 0.8623897541900952, 'reg_alpha': 0.5634028301395699, 'reg_lambda': 0.35688209106396496}. Best is trial 3 with value: -0.37265681096683356.

[I 2026-01-15 21:52:47,659] Trial 15 finished with value: -0.3720638975956825 and parameters: {'max_depth': 4, 'min_child_weight': 4, 'gamma': 0.46961633938122976, 'learning_rate': 0.06957655587471777, 'subsample': 0.7839094857138693, 'colsample_bytree': 0.7597541295648416, 'reg_alpha': 3.2594483061523083, 'reg_lambda': 4.646747868529857}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:52:55,106] Trial 16 finished with value: -0.3959418927308989 and parameters: {'max_depth': 4, 'min_child_weight': 5, 'gamma': 3.688684361663831, 'learning_rate': 0.0842090186861907, 'subsample': 0.6882195688453033, 'colsample_bytree': 0.7726998549839373, 'reg_alpha': 1.719800120981299, 'reg_lambda': 8.41718562280153}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:05,445] Trial 17 finished with value: -0.40627609202992454 and parameters: {'max_depth': 3, 'min_child_weight': 4, 'gamma': 1.8760635036422193, 'learning_rate': 0.010001986945650408, 'subsample': 0.7773129784722851, 'colsample_bytree': 0.6678857788719847, 'reg_alpha': 6.406453660813229, 'reg_lambda': 4.694759136869631}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:12,472] Trial 18 finished with value: -0.3935011314544533 and parameters: {'max_depth': 4, 'min_child_weight': 4, 'gamma': 2.8340420567053535, 'learning_rate': 0.1824327034007405, 'subsample': 0.7112411555309504, 'colsample_bytree': 0.8577845245877058, 'reg_alpha': 3.943061747570987, 'reg_lambda': 6.526641580687851}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:22,819] Trial 19 finished with value: -0.37747962689428893 and parameters: {'max_depth': 4, 'min_child_weight': 7, 'gamma': 0.7910038636709606, 'learning_rate': 0.031272717022155526, 'subsample': 0.9279019417155216, 'colsample_bytree': 0.7571282196436854, 'reg_alpha': 1.8563319808442837, 'reg_lambda': 5.035086456290394}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:32,613] Trial 20 finished with value: -0.37286276050538264 and parameters: {'max_depth': 3, 'min_child_weight': 10, 'gamma': 0.08589954287612017, 'learning_rate': 0.17863182041749628, 'subsample': 0.8783443706969066, 'colsample_bytree': 0.6680209505850502, 'reg_alpha': 5.032598942789081, 'reg_lambda': 6.048665400287299}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:43,023] Trial 21 finished with value: -0.3735041264716113 and parameters: {'max_depth': 3, 'min_child_weight': 10, 'gamma': 0.01667695127103462, 'learning_rate': 0.18613809367527687, 'subsample': 0.8686942396371777, 'colsample_bytree': 0.674035607994361, 'reg_alpha': 4.877235546482136, 'reg_lambda': 8.117711160611048}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:53:50,435] Trial 22 finished with value: -0.38544602601566924 and parameters: {'max_depth': 3, 'min_child_weight': 8, 'gamma': 0.9308235533260028, 'learning_rate': 0.08852311937508765, 'subsample': 0.8731141761501332, 'colsample_bytree': 0.6681276353489173, 'reg_alpha': 6.100740241051966, 'reg_lambda': 6.08557777039848}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:54:00,643] Trial 23 finished with value: -0.37575637976272835 and parameters: {'max_depth': 4, 'min_child_weight': 3, 'gamma': 0.30985582283442276, 'learning_rate': 0.15873199130443646, 'subsample': 0.8058405620986135, 'colsample_bytree': 0.7511869064033881, 'reg_alpha': 3.554139983933715, 'reg_lambda': 4.336814897753011}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:54:08,704] Trial 24 finished with value: -0.38715149038400265 and parameters: {'max_depth': 5, 'min_child_weight': 5, 'gamma': 1.0295821235680858, 'learning_rate': 0.07864811488947812, 'subsample': 0.8488203307897634, 'colsample_bytree': 0.8100564465058387, 'reg_alpha': 7.2801674450570575, 'reg_lambda': 7.666956820263463}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:54:15,679] Trial 25 finished with value: -0.3759473114846309 and parameters: {'max_depth': 3, 'min_child_weight': 10, 'gamma': 0.3972645111838357, 'learning_rate': 0.23012097895081654, 'subsample': 0.9460445512516922, 'colsample_bytree': 0.7423710560803978, 'reg_alpha': 4.897715178997141, 'reg_lambda': 3.537066704357537}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:54:22,494] Trial 26 finished with value: -0.3920502102516166 and parameters: {'max_depth': 4, 'min_child_weight': 4, 'gamma': 1.7464054081827882, 'learning_rate': 0.1444115484106143, 'subsample': 0.8988469044326146, 'colsample_bytree': 0.6841012852682075, 'reg_alpha': 5.75671172743542, 'reg_lambda': 5.140062543064758}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:54:29,282] Trial 27 finished with value: -0.4026613019425908 and parameters: {'max_depth': 3, 'min_child_weight': 6, 'gamma': 4.8016

7943623213, 'learning_rate': 0.21849601247698197, 'subsample': 0.771554
4604884824, 'colsample_bytree': 0.8313071746961169, 'reg_alpha': 2.3385
26390256149, 'reg_lambda': 5.807069485942546}. Best is trial 15 with va
lue: -0.3720638975956825.

[I 2026-01-15 21:54:38,307] Trial 28 finished with value: -0.3729282754947091
and parameters: {'max_depth': 6, 'min_child_weight': 2, 'gamma': 1.0790
68895265574, 'learning_rate': 0.1054496821621514, 'subsample': 0.729763
0774983528, 'colsample_bytree': 0.6409813872623883, 'reg_alpha': 1.1123
725458909433, 'reg_lambda': 7.4677757941665455}. Best is trial 15 with
value: -0.3720638975956825.

[I 2026-01-15 21:54:53,351] Trial 29 finished with value: -0.3773116432241846
and parameters: {'max_depth': 7, 'min_child_weight': 9, 'gamma': 0.3712
5602606950725, 'learning_rate': 0.0374519023492725, 'subsample': 0.6049
512563590811, 'colsample_bytree': 0.8097816786916638, 'reg_alpha': 7.12
9318849222457, 'reg_lambda': 2.4441686943005774}. Best is trial 15 with
value: -0.3720638975956825.

[I 2026-01-15 21:55:00,808] Trial 30 finished with value: -0.37923426347256756
and parameters: {'max_depth': 5, 'min_child_weight': 8, 'gamma': 1.1256
879133967572, 'learning_rate': 0.23582154317653595, 'subsample': 0.8087
665582815857, 'colsample_bytree': 0.8873925227013255, 'reg_alpha': 3.64
99060362226032, 'reg_lambda': 6.335165331472403}. Best is trial 15 with
value: -0.3720638975956825.

[I 2026-01-15 21:55:11,483] Trial 31 finished with value: -0.37288393015447674
and parameters: {'max_depth': 6, 'min_child_weight': 2, 'gamma': 0.4815
8557564211263, 'learning_rate': 0.1023299436901957, 'subsample': 0.7262
55669262165, 'colsample_bytree': 0.6402717654608393, 'reg_alpha': 4.602
253322243703, 'reg_lambda': 7.425287134812347}. Best is trial 15 with v
alue: -0.3720638975956825.

[I 2026-01-15 21:55:21,533] Trial 32 finished with value: -0.37289809260912193
and parameters: {'max_depth': 4, 'min_child_weight': 3, 'gamma': 0.4905
651876952845, 'learning_rate': 0.09884162670594808, 'subsample': 0.7140
379883075073, 'colsample_bytree': 0.6488073620189053, 'reg_alpha': 4.50
8804268923725, 'reg_lambda': 8.93961849992316}. Best is trial 15 with v
alue: -0.3720638975956825.

[I 2026-01-15 21:55:45,019] Trial 33 finished with value: -0.40569486592122067
and parameters: {'max_depth': 8, 'min_child_weight': 2, 'gamma': 0.0021
113645027073247, 'learning_rate': 0.1394107618050133, 'subsample': 0.76
00276283413656, 'colsample_bytree': 0.6118987732223758, 'reg_alpha': 3.
0465502599635563, 'reg_lambda': 7.0942598480989005}. Best is trial 15 w
ith value: -0.3720638975956825.

[I 2026-01-15 21:55:55,792] Trial 34 finished with value: -0.37653076142233755
and parameters: {'max_depth': 6, 'min_child_weight': 1, 'gamma': 0.6950
006001489266, 'learning_rate': 0.07125716860298927, 'subsample': 0.6712
83109260447, 'colsample_bytree': 0.7038899117523182, 'reg_alpha': 5.213
520285721291, 'reg_lambda': 4.896492244395911}. Best is trial 15 with v
alue: -0.3720638975956825.

[I 2026-01-15 21:56:06,303] Trial 35 finished with value: -0.3728449614165209
and parameters: {'max_depth': 3, 'min_child_weight': 5, 'gamma': 0.3124
453880133517, 'learning_rate': 0.054913932231582925, 'subsample': 0.793
9671892768347, 'colsample_bytree': 0.7897855967126697, 'reg_alpha': 3.4

44192280596659, 'reg_lambda': 5.671882147998293}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:14,675] Trial 36 finished with value: -0.3839297334363369 and parameters: {'max_depth': 3, 'min_child_weight': 6, 'gamma': 1.3959525850609522, 'learning_rate': 0.0514970045561201, 'subsample': 0.7919017618012679, 'colsample_bytree': 0.7833655330857161, 'reg_alpha': 3.422998498495628, 'reg_lambda': 5.482229770418971}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:22,093] Trial 37 finished with value: -0.40016742114897663 and parameters: {'max_depth': 3, 'min_child_weight': 5, 'gamma': 3.593204453190096, 'learning_rate': 0.058042806495391744, 'subsample': 0.8294909260329725, 'colsample_bytree': 0.7965429707452746, 'reg_alpha': 2.5359067688418744, 'reg_lambda': 9.868311992182182}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:34,413] Trial 38 finished with value: -0.38228695576367977 and parameters: {'max_depth': 4, 'min_child_weight': 5, 'gamma': 0.2690605969747919, 'learning_rate': 0.024485371238539652, 'subsample': 0.8510833830360467, 'colsample_bytree': 0.7361875466473289, 'reg_alpha': 8.21869133814489, 'reg_lambda': 4.173144324956719}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:42,443] Trial 39 finished with value: -0.37981187534370564 and parameters: {'max_depth': 3, 'min_child_weight': 7, 'gamma': 0.8006235389237154, 'learning_rate': 0.06606208841712034, 'subsample': 0.8832969827407506, 'colsample_bytree': 0.768838849382922, 'reg_alpha': 4.27637185724969, 'reg_lambda': 2.8095357560070076}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:50,968] Trial 40 finished with value: -0.38958698568131334 and parameters: {'max_depth': 4, 'min_child_weight': 4, 'gamma': 1.5767719859781804, 'learning_rate': 0.034568005523307514, 'subsample': 0.9926796519525006, 'colsample_bytree': 0.8311566192690505, 'reg_alpha': 1.8121987299239188, 'reg_lambda': 3.4751224067224835}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:56:59,630] Trial 41 finished with value: -0.3757324592145183 and parameters: {'max_depth': 5, 'min_child_weight': 3, 'gamma': 0.5287321854647109, 'learning_rate': 0.109645066228864, 'subsample': 0.8113506424532357, 'colsample_bytree': 0.6000727301719802, 'reg_alpha': 5.99820758304069, 'reg_lambda': 6.5841241536526764}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:57:09,781] Trial 42 finished with value: -0.3726819348057765 and parameters: {'max_depth': 3, 'min_child_weight': 6, 'gamma': 0.2711153271041241, 'learning_rate': 0.07238314837467037, 'subsample': 0.7208942317504491, 'colsample_bytree': 0.6314333661991816, 'reg_alpha': 4.761037948167271, 'reg_lambda': 4.650721225650912}. Best is trial 15 with value: -0.3720638975956825.

[I 2026-01-15 21:57:20,108] Trial 43 finished with value: -0.37357170685415947 and parameters: {'max_depth': 3, 'min_child_weight': 6, 'gamma': 0.29578182908667283, 'learning_rate': 0.05182798104616481, 'subsample': 0.7889502374084936, 'colsample_bytree': 0.7879890138423946, 'reg_alpha': 3.9827179469510323, 'reg_lambda': 4.424249721943031}. Best is trial 15 with value: -0.3720638975956825.

```
[I 2026-01-15 21:57:30,881] Trial 44 finished with value: -0.37729379387221507  
and parameters: {'max_depth': 3, 'min_child_weight': 8, 'gamma': 0.1647  
572450878722, 'learning_rate': 0.040930955702245024, 'subsample': 0.758  
5392860561544, 'colsample_bytree': 0.6951024320850596, 'reg_alpha': 5.3  
3700447462476, 'reg_lambda': 5.469656576498563}. Best is trial 15 with  
value: -0.3720638975956825.
```

```
[I 2026-01-15 21:57:41,677] Trial 45 finished with value: -0.37320444847972634  
and parameters: {'max_depth': 4, 'min_child_weight': 4, 'gamma': 0.9108  
359156290893, 'learning_rate': 0.07595564819850507, 'subsample': 0.6982  
181432947606, 'colsample_bytree': 0.7330008903385189, 'reg_alpha': 2.98  
66894881284267, 'reg_lambda': 3.9625908570677346}. Best is trial 15 wit  
h value: -0.3720638975956825.
```

```
[I 2026-01-15 21:57:49,167] Trial 46 finished with value: -0.3811395769771201  
and parameters: {'max_depth': 3, 'min_child_weight': 5, 'gamma': 1.1897  
721279374895, 'learning_rate': 0.2972869234601859, 'subsample': 0.63537  
85012458778, 'colsample_bytree': 0.9397076038519964, 'reg_alpha': 6.641  
368652087468, 'reg_lambda': 4.63564348180965}. Best is trial 15 with va  
lue: -0.3720638975956825.
```

```
[I 2026-01-15 21:58:00,720] Trial 47 finished with value: -0.37807851879126597  
and parameters: {'max_depth': 4, 'min_child_weight': 9, 'gamma': 0.6270  
967521670547, 'learning_rate': 0.027380056169423957, 'subsample': 0.824  
3778979463041, 'colsample_bytree': 0.7631715726246977, 'reg_alpha': 3.7  
918883904663434, 'reg_lambda': 5.934791597097885}. Best is trial 15 wit  
h value: -0.3720638975956825.
```

```
[I 2026-01-15 21:58:09,232] Trial 48 finished with value: -0.3933691381993708  
and parameters: {'max_depth': 3, 'min_child_weight': 6, 'gamma': 2.1333  
286482268803, 'learning_rate': 0.0629475790525285, 'subsample': 0.73909  
27986600755, 'colsample_bytree': 0.6224048385758171, 'reg_alpha': 4.940  
019232650667, 'reg_lambda': 5.259039432981991}. Best is trial 15 with v  
alue: -0.3720638975956825.
```

```
[I 2026-01-15 21:58:24,434] Trial 49 finished with value: -0.37853856175705414  
and parameters: {'max_depth': 10, 'min_child_weight': 4, 'gamma': 0.176  
39900170799772, 'learning_rate': 0.09019751382454541, 'subsample': 0.84  
25682670333119, 'colsample_bytree': 0.6552314067493566, 'reg_alpha': 3.  
23606300889923, 'reg_lambda': 3.2051370894066427}. Best is trial 15 wit  
h value: -0.3720638975956825.
```

D'après Optuna, les meilleurs paramètres sont les suivants :

```
parametres = tuning.best_params
```

```
Les paramètres utilisés pour la submission sont : {'max_depth': 7, 'min_child_weight': 6, 'gamma':  
0.5567327212493922, 'learning_rate': 0.034154565256344305, 'subsample':  
0.7227286052244662, 'colsample_bytree': 0.6745599697255247, 'reg_alpha':  
1.7139816149193734, 'reg_lambda': 5.86860017256115}
```

Nous entraînons donc notre modèle avec les meilleurs paramètres sur la base de données "train".

```
model = XGBClassifier(  
    **parametres,  
    n_estimators=800,  
    eval_metric="logloss",  
    random_state=42,  
    n_jobs=-1  
)
```

```
pipeline = Pipeline([
    ("preprocessor", preprocessor),
    ("model", model)
])

pipeline.fit(X, y_enc)

Pipeline
?
preprocessor: ColumnTransformer
?
    num
SimpleImputer
?
    qual
OneHotEncoder
?
XGBClassifier
?

Et nous appliquons le modèle sur la base de données "test".
X_test = wrangle("/kaggle/input/master-econometrie-et-statistiques/test.csv")

y_pred_proba_test = pipeline.predict_proba(X_test.drop(columns=['id']))

submission = pd.DataFrame(
    y_pred_proba_test,
    columns=['C', 'CL', 'D']
)
submission['id'] = X_test['id']
submission = submission[['id', 'C', 'CL', 'D']]
submission.to_csv("submission.csv", index=False)
```