Experiment Objective :

- XGBoost + Bayesian Optimization

Hasil dan Analisa :

- AUC score dengan menggunakan Bayesian Optimization dari library scikit optimize yaitu 0.828. Nilai ini lebih tinggi dibandingkan menggunakan random search atau grid search.

Code :

In [22]:
```
import pandas as pd
from skopt import BayesSearchCV
from skopt.space import Real,Integer
from sklearn.model_selection import StratifiedKFold
from xgboost import XGBClassifier
```

In [23]:
```
print(skopt.__version__)
```

```
0.8.1
```

In [24]:
```
X_bank=pd.read_csv('dataset/X_bank_preprocessed.csv').to_numpy()
y_bank=pd.read_csv('dataset/y_bank_preprocessed.csv').to_numpy().ravel()
```

In [25]:
```
model=XGBClassifier()
```

In [28]:
```
params={
    'eta': Real(0.001,1,'log-uniform'), # learning rate
    'subsample': Real(0,1,'uniform'),
    'max_depth': Integer(5,50),
    'gamma': Real(1e-9,1,'log-uniform'),
    'min_child_weight': Integer(0,15)
}
```

- referensi untuk prior distribution: https://www.kaggle.com/nanomathias/bayesian-optimization-of-xgboost-lb-0-9769 (https://www.kaggle.com/nanomathias/bayesian-optimization-of-xgboost-lb-0-9769)
- penggunaan tipe data pada tiap hyperparameter : https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html (https://scikit-optimize.github.io/stable/modules/generated/skopt.BayesSearchCV.html)
- untuk min value dari gamma tidak bisa jika 0, maka diganti dengan nilai yang sangat kecil mendekati 0

Scikit-optimize

In [30]:
```
bayes_tuner= BayesSearchCV(model,
                           params,
                           n_jobs=-1,
                           n_iter=50,
                           scoring='roc_auc',
                           cv=5,
                           random_state=1)
```

In [31]:
```
bayes_tuner.fit(X_bank,y_bank)
```

Out[31]:
```
BayesSearchCV(cv=5, error_score='raise',
              estimator=XGBClassifier(base_score=None, booster=None,
                                      colsample_bylevel=None,
                                      colsample_bynode=None,
                                      colsample_bytree=None, gamma=None,
                                      gpu_id=None, importance_type='gain',
                                      interaction_constraints=None,
                                      learning_rate=None, max_delta_step=Non
e,
                                      max_depth=None, min_child_weight=None,
                                      missing=nan, monotone_constraints=None,
                                      n_...
              search_spaces={'eta': Real(low=0.001, high=1, prior='log-unifor
m', transform='identity'),
                             'gamma': Real(low=1e-09, high=1, prior='log-unif
orm', transform='identity'),
                             'max_depth': Integer(low=5, high=50, prior='unif
orm', transform='identity'),
                             'min_child_weight': Integer(low=0, high=15, prio
r='uniform', transform='identity'),
                             'subsample': Real(low=0, high=1, prior='unifor
m', transform='identity')},
              verbose=0)
```

In [32]:
```
index=bayes_tuner.best_index_
print("Best params: ")
print(bayes_tuner.best_params_)
print("AUC: ")
print(bayes_tuner.cv_results_['mean_test_score'][index])
print("std: ")
print(bayes_tuner.cv_results_['std_test_score'][index])
```

```
Best params:
OrderedDict([('eta', 0.0032453549635184584), ('gamma', 0.0240750964750453),
('max_depth', 21), ('min_child_weight', 15), ('subsample', 0.0175253259941222
82)])
AUC:
0.8278029873155748
std:
0.07065837411116632
```

In [ ]: