1. Méthode utilisée :

1.1. Prétraitement :

Pour le prétraitement, nous avons décider de ne pas considérer ces variables :

- *fulltext* : Variable non numérique
- $MATTR$ et $MSTTR$: Quantité trop importante de valeur NaN ($\simeq 80\%$)

1.2. Algorithme d'apprentissage :

Nous avons utilisé l'algorithme SVM (Support Vector Machine) implémenter par la lib python *scikit-learn*.

SVM dispose de deux paramètres : C et $gamma$, qui peuvent être affinés en fonction de la performance des prédictions sur les données de test.

1.3. Mesure :

La mesure de performance est celle fournie par CAp 2018 Competition: Call for Participation.

1.4. Optimisation des hyper-paramètres

Pour maximiser la performance du SVM, nous avons utilisé l'optimisation bayésienne afin de d'optimiser les paramètres C et $gamma$.

Objectif : En considérant l'ensemble sur $\mathbb{R}^2$ formé par les paramètres C et $gamma$, on cherche le couple $(C; gamma)$ qui maximise la performance du modèle d'apprentissage.

Principe : L'optimisation bayésienne construit une distribution probabiliste à posteriori pour la fonction à optimiser en explorant ou en exploitant (d'après les observations passées) l'espace formé par C et $gamma$. L'optimisateur devrait s'améliorer après chaque observation et trouver les zones de $\mathbb{R}^2$ les plus intéressantes.

Pratique : Nous avons utilisé une lib pour l'optimisation bayésienne : BayesianOptimization. Les intervalles utilisées pour la recherche de C et $gamma$ sont respectivement $[0.001; 100]$ et $[0.0001; 0.1]$

2. Sources

Bayesian optimization with scikit-learn