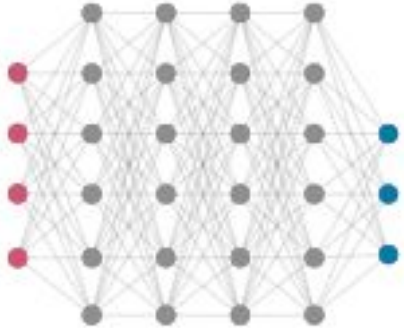
Projet Kaggle : Prédire une réponse biologique des molécules à partir de leurs propriétés chimiques

Emma Soufir
Naïma Ammiche
Thanina Chabane
Noura Nouali

Introduction

Introduction

- ➔ Développement de médicaments
- ➔ Prédiction des réactions bio-moléculaires
- ➔ Apprentissage automatique



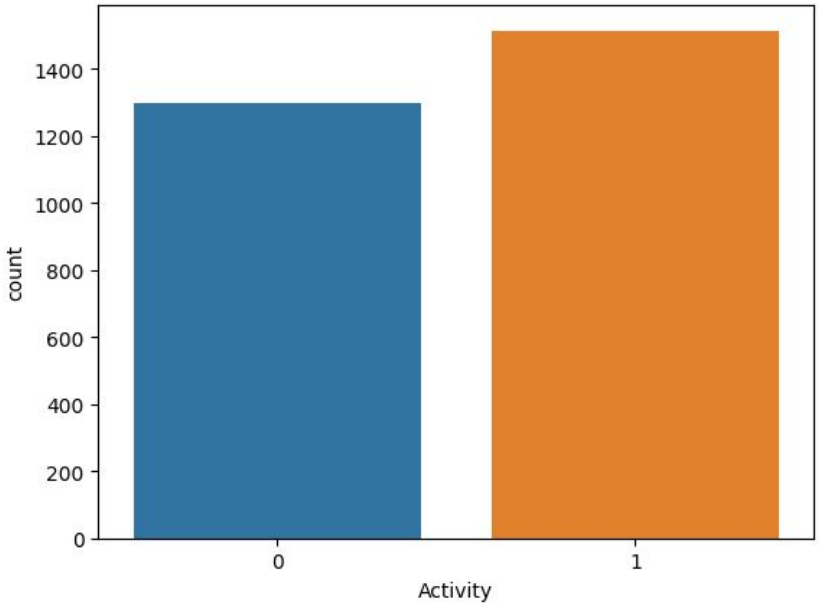
Comment développer un modèle de prédiction de réponse biologique des molécules efficaces ?

Matériel et méthodes

Présentation des données

1776 descripteurs

3751 molécules



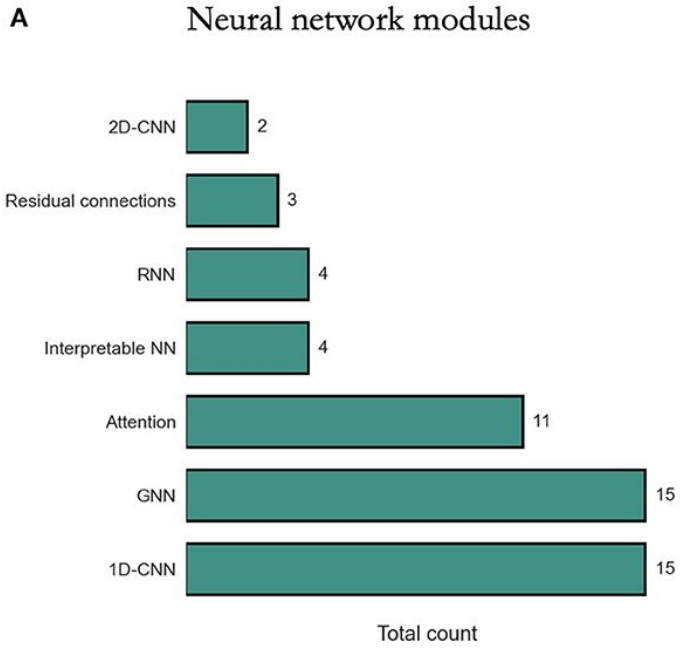
<u>Prétraitement :</u>

- Variables avec peu de variabilité
- Variables fortement corrélées
- MinMax Scaling

⇒ 1533 descripteurs conservés

Modèles

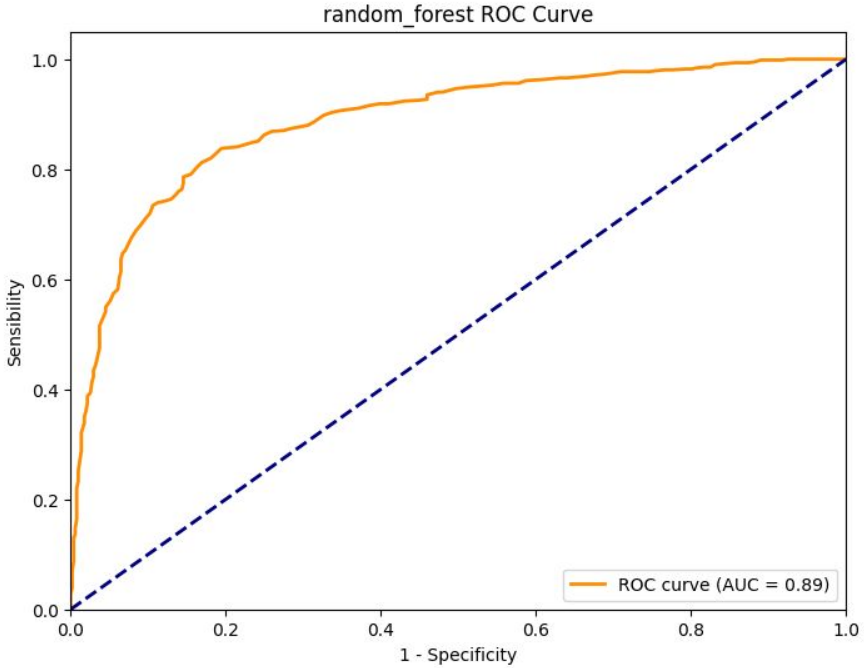
- Random Forest
- MLP
- CNN
- Gradient Boosting
- Autres Modèles *(régression logistique, RNN, MLP Adam, MINN, XGB)*

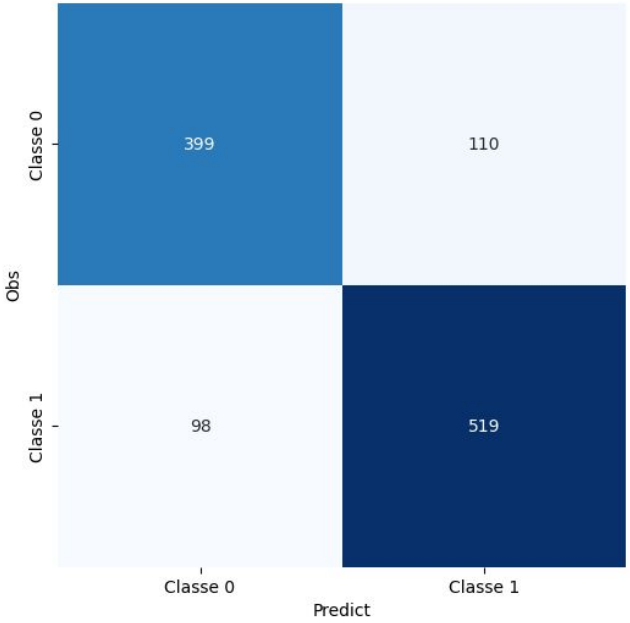


Partin A, et al. 2023

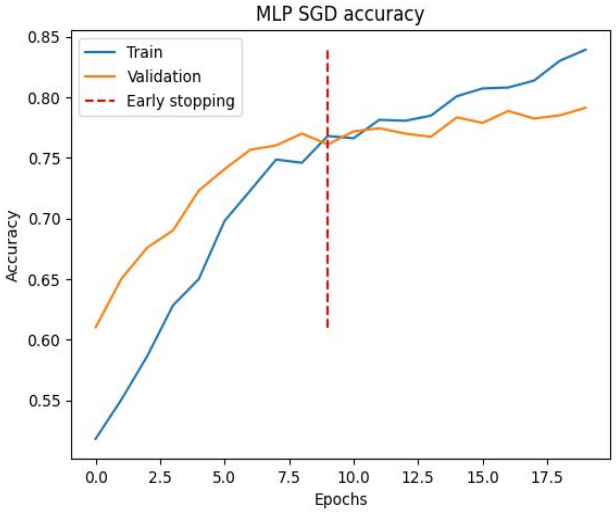
Résultats et Discussion

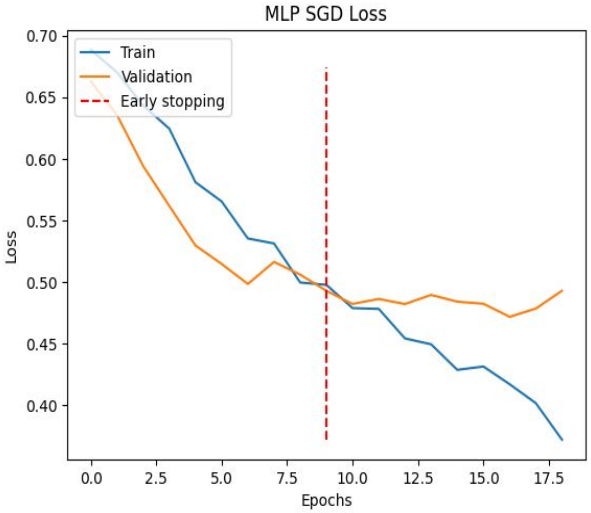
Modèle : Random Forest

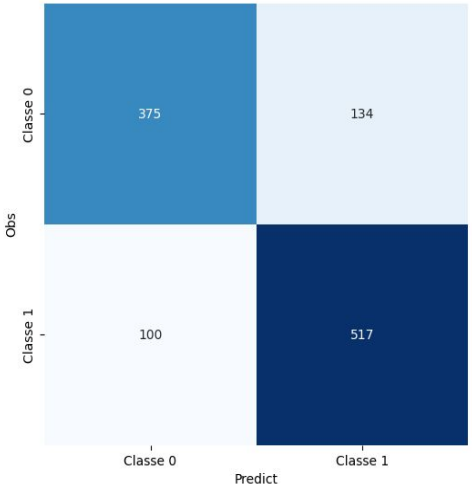




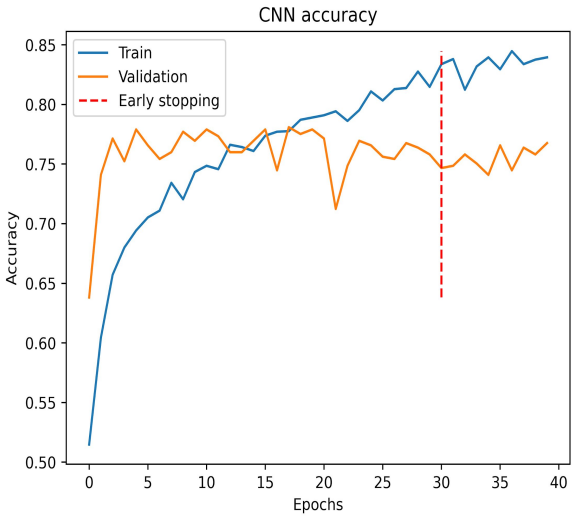
Modèle : Multi Layer Perceptron (MLP) avec SGD

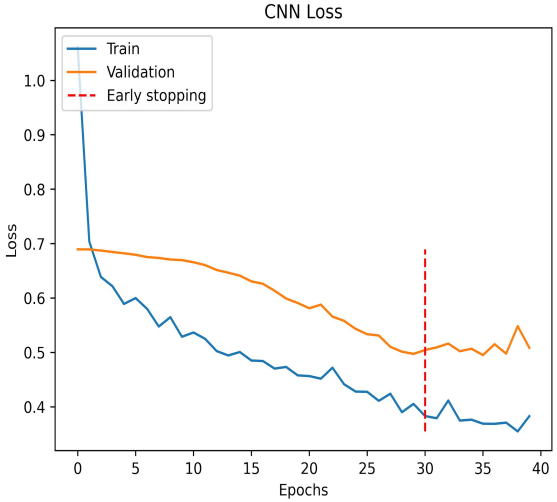


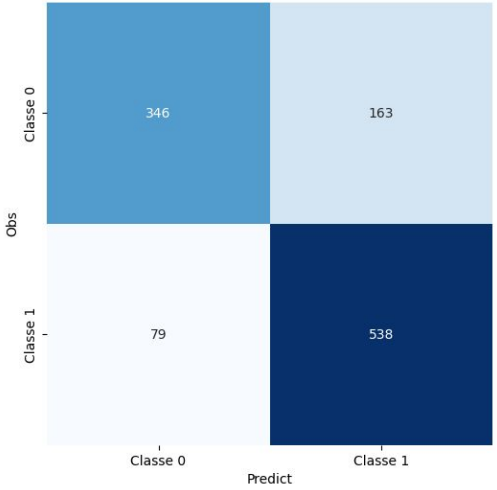




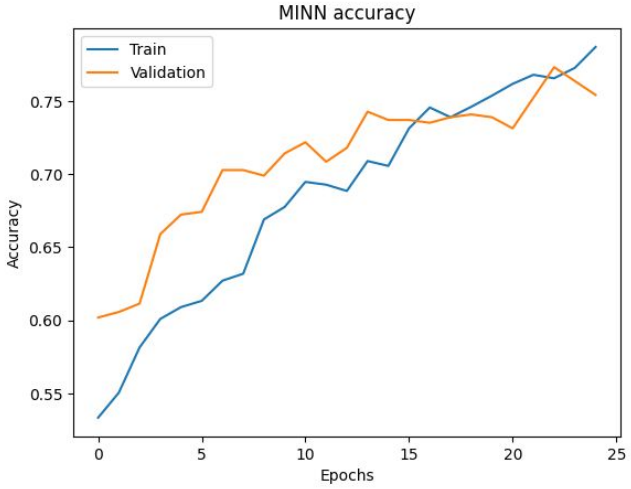
Modèle: Réseaux de Neurones Convolutifs (CNN)

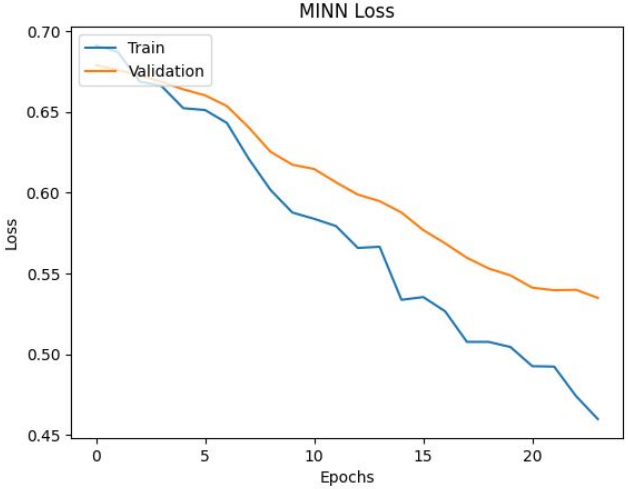


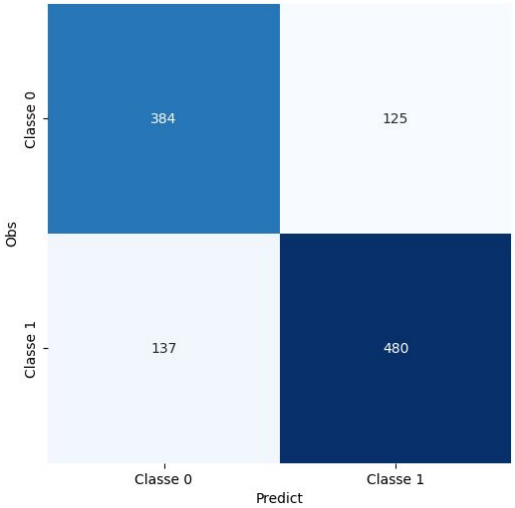




Modèle: Multi-Input Neural Network

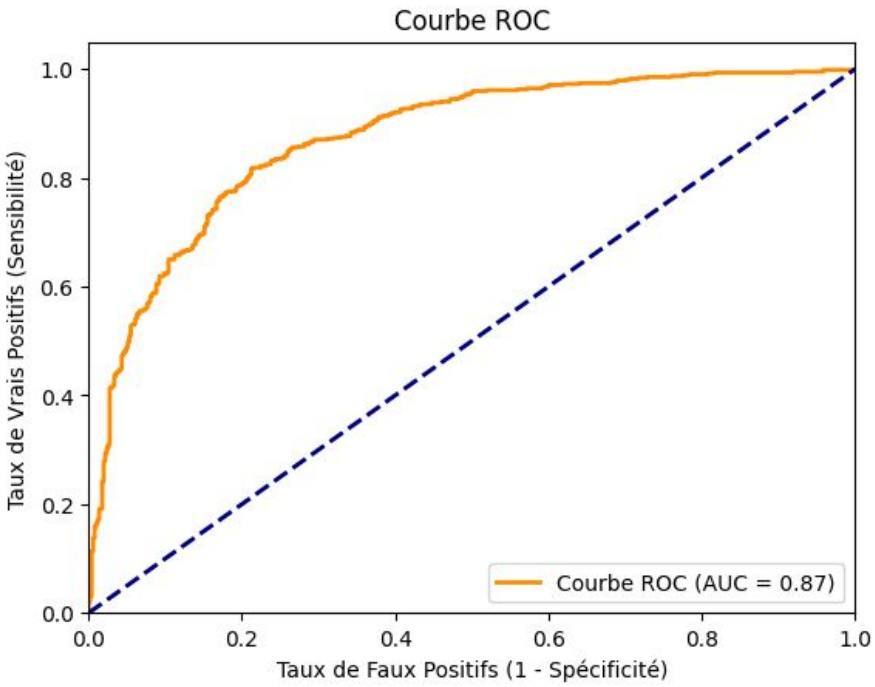




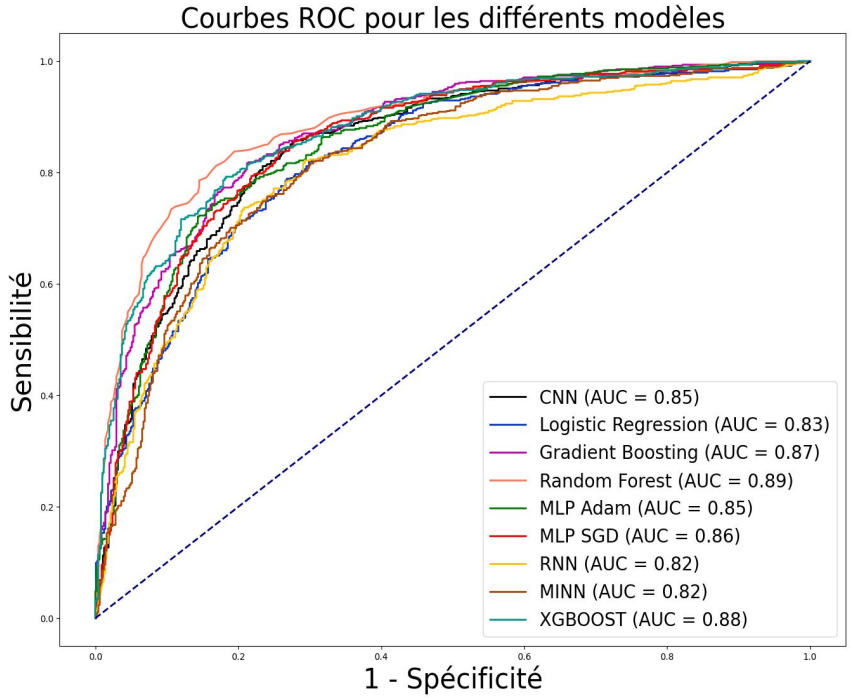


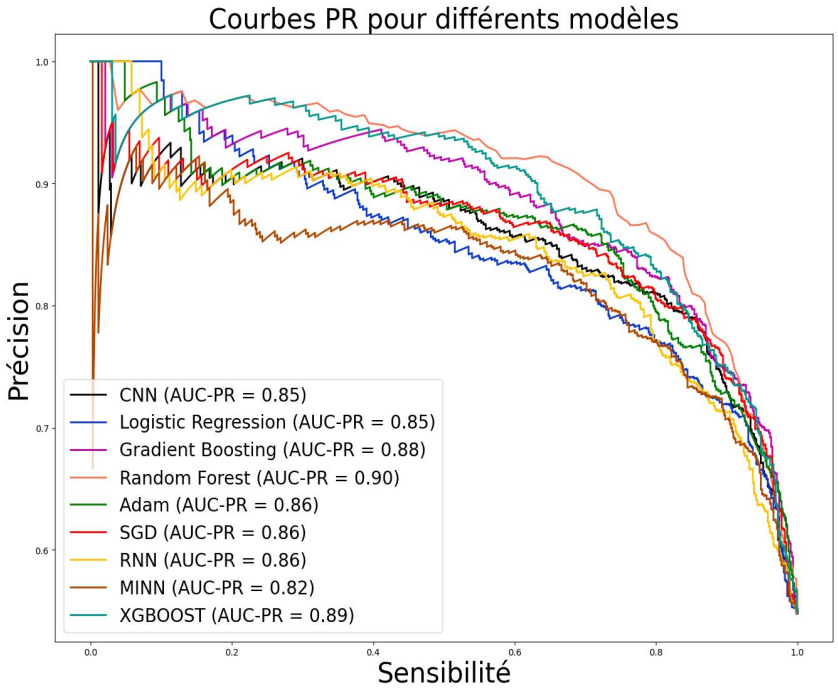
Modèle : Gradient Boosting

Accuracy	0.80
Sensibility	0.85
Specificity	0.74
F1 score	0.82

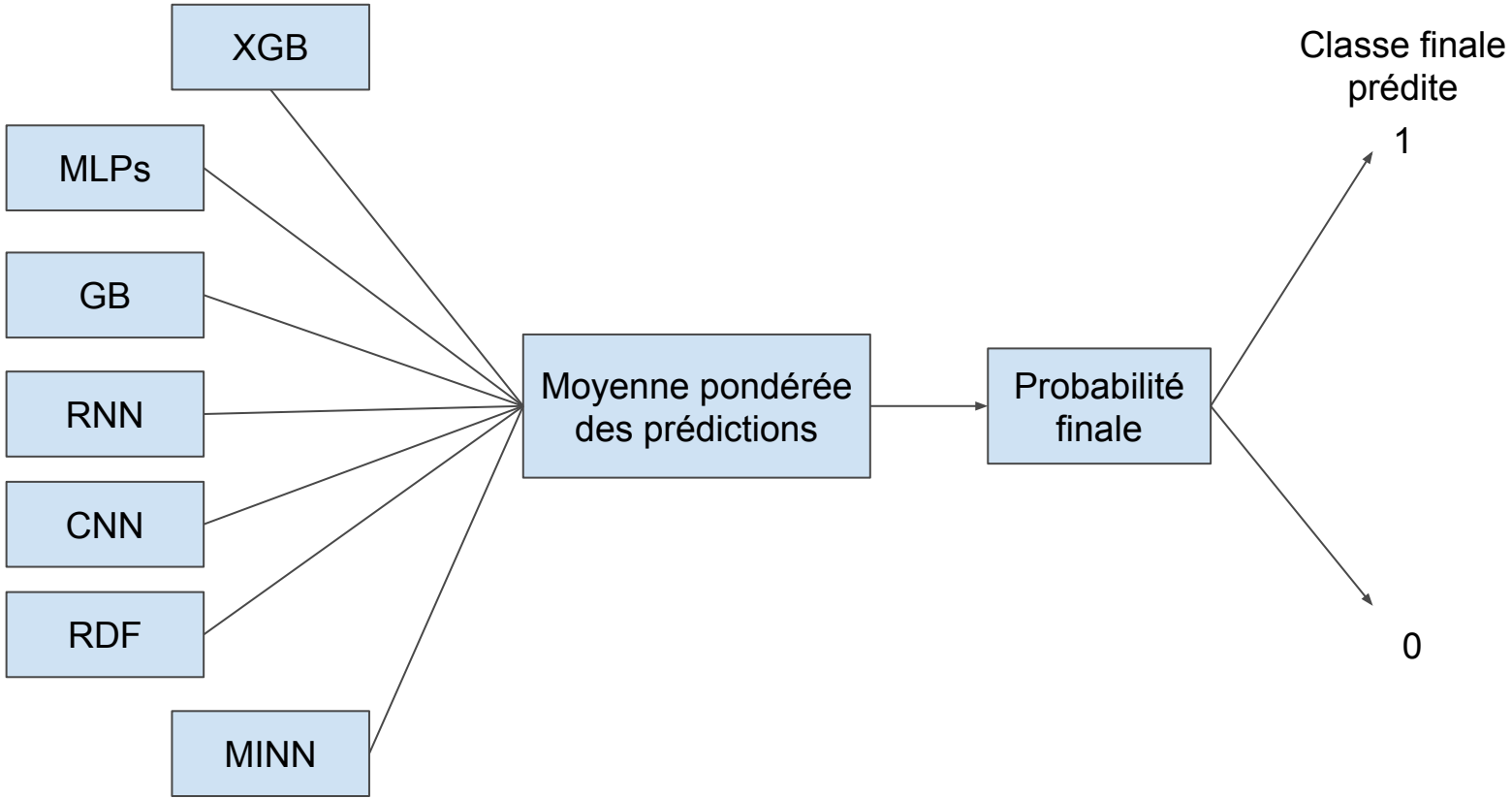


Comparaison des modèles



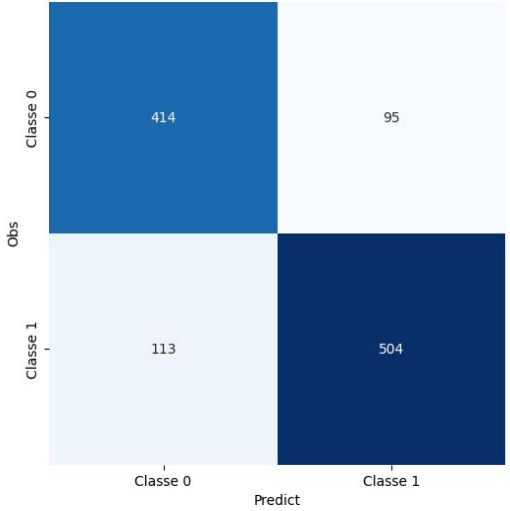


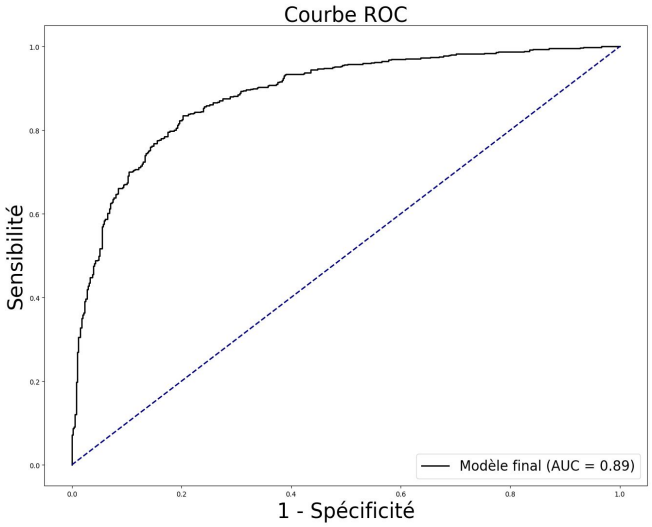
Modèle Final



Modèle Final : Performance

Accuracy	0.82
Sensibilité	0.82
Spécificité	0.81





Soumission Kaggle

$$-\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

	RF	MLP Adam	MLP SGD	CNN	RNN	MINN	Régression logistique	Gradient boosting	XGBoost	Modèle final
Notes à la compétition	0.468	0.500	0.440	0.490	0.598	0.529	0.538	0.472	0.466	0.435

Conclusion et Perspectives

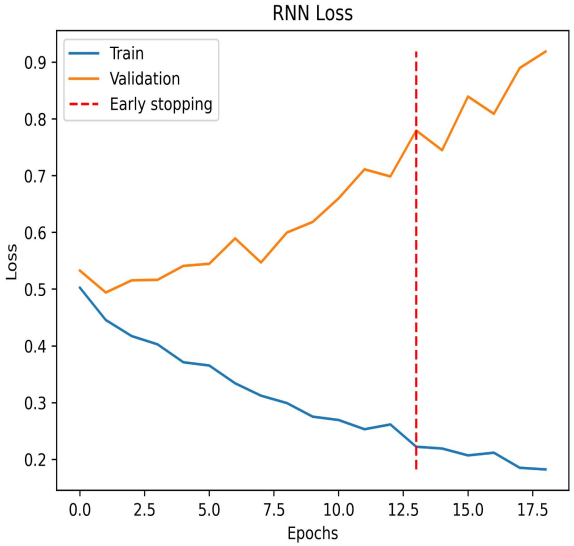
Conclusion

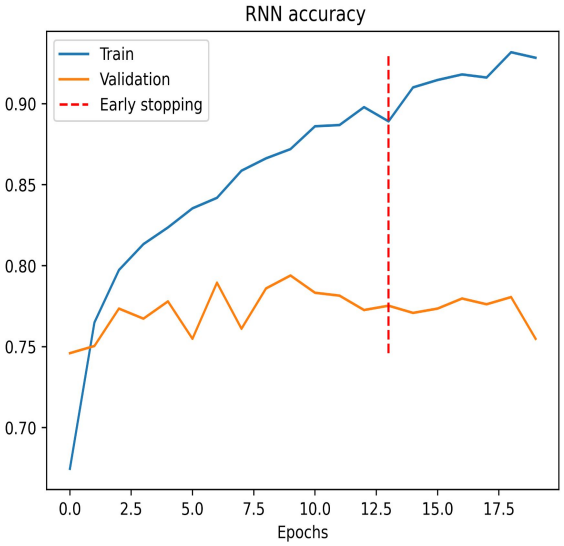
- Modèle final avec le meilleur score
- Manque d’information sur les descripteurs
- Classes déséquilibrées
- Random Forest et Gradient Boosting sont les modèles les plus performants

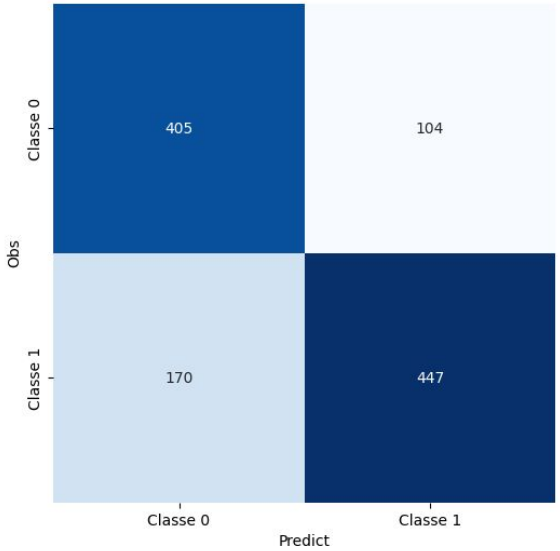
Perspectives

- Choix du nombre de variables
- Autres architectures
- Plus grand spectre de paramètres

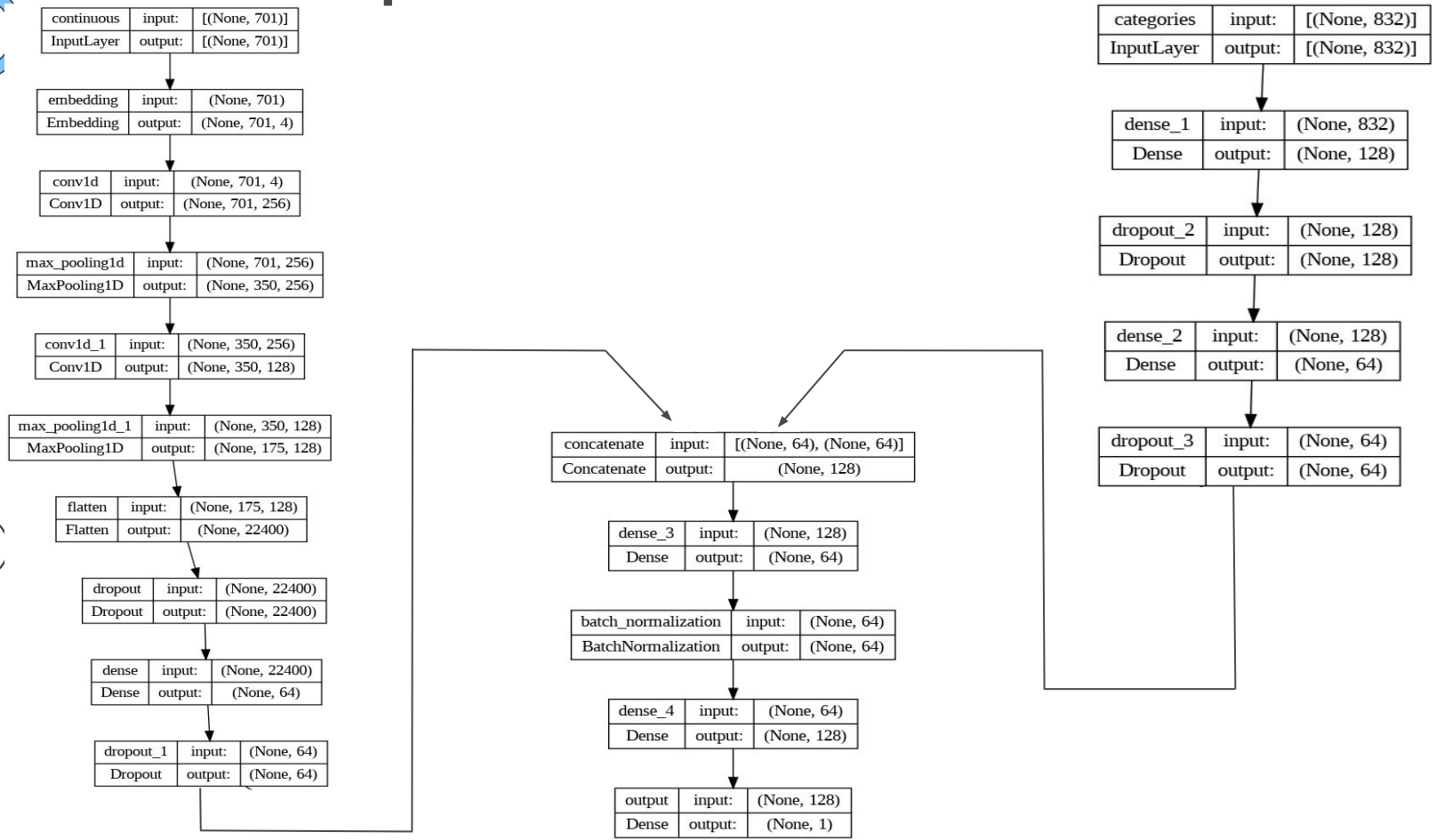
Modèle : Réseaux de Neurones Récurrents (RNN)



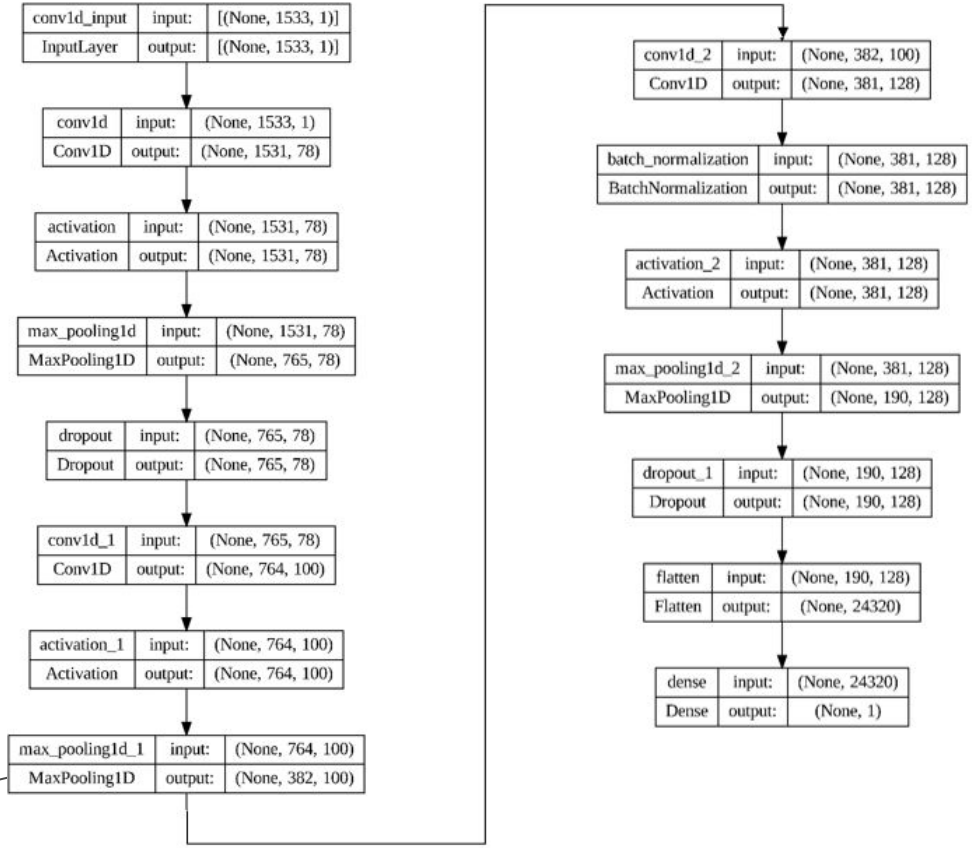




Modèle: Multi-Input Neural Network



Modèle: CNN


conv1d_input | input: | [(None, 1533, 1)]
InputLayer | output: | [(None, 1533, 1)]
conv1d | input: | (None, 1533, 1)
Conv1D | output: | (None, 1531, 78)
activation | input: | (None, 1531, 78)
Activation | output: | (None, 1531, 78)
max_pooling1d | input: | (None, 1531, 78)
MaxPooling1D | output: | (None, 765, 78)
dropout | input: | (None, 765, 78)
Dropout | output: | (None, 765, 78)
conv1d_1 | input: | (None, 765, 78)
Conv1D | output: | (None, 764, 100)
activation_1 | input: | (None, 764, 100)
Activation | output: | (None, 764, 100)
max_pooling1d_1 | input: | (None, 764, 100)
MaxPooling1D | output: | (None, 382, 100)
conv1d_2 | input: | (None, 382, 100)
Conv1D | output: | (None, 381, 128)
batch_normalization | input: | (None, 381, 128)
BatchNormalization | output: | (None, 381, 128)
activation_2 | input: | (None, 381, 128)
Activation | output: | (None, 381, 128)
max_pooling1d_2 | input: | (None, 381, 128)
MaxPooling1D | output: | (None, 190, 128)
dropout_1 | input: | (None, 190, 128)
Dropout | output: | (None, 190, 128)
flatten | input: | (None, 190, 128)
Flatten | output: | (None, 24320)
dense | input: | (None, 24320)
Dense | output: | (None, 1)