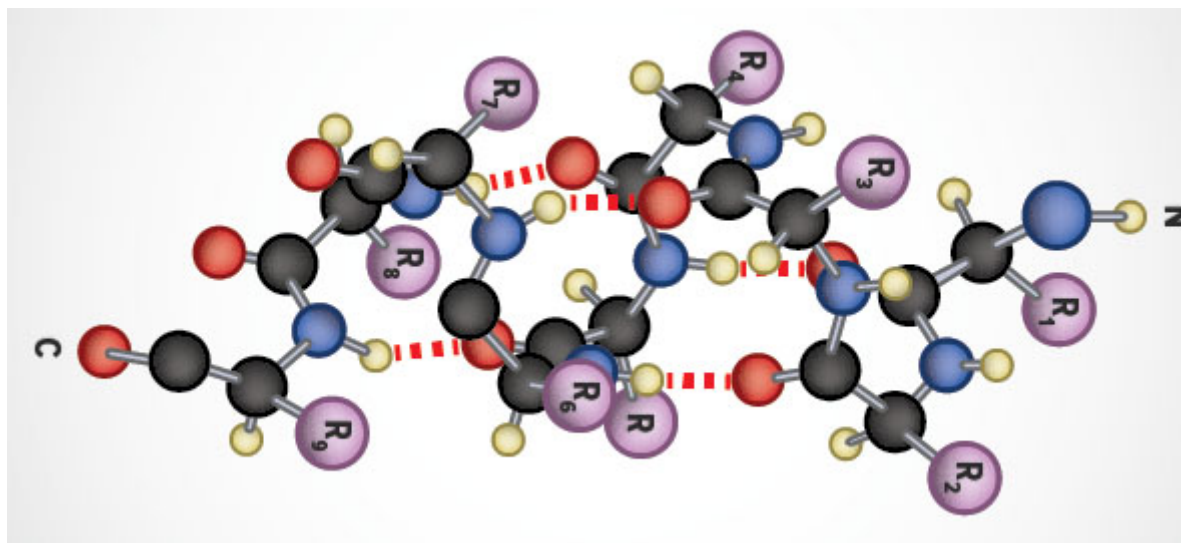


Structural Protein Classification



NGIZULU Edi

[linkedin](#)

DIALLO Sadou Safa

[linkedin](#)

DS Formation continue Mai 2021

Sommaire

###Contexte du projet

I. Analyse des données

1. [Source des données](#)
2. [Nettoyage des données](#)
3. [Analyse Exploratoire des données](#)
 - [Classification](#)
 - [Technique d'extraction de la Proteine](#)
 - [PH Value](#)
 - [macromoleculeType](#)
 - [methodes_crystallizations](#)
 - [residueCount](#)
 - [structureMolecularWeight](#)
 - [publicationYear](#)
 - [Sequence Feature](#)

II-Méthodologie

III. Modélisation

A-MACHINE LEARNING

1. Preprocessing
2. Métriques des tests
3. Itération des Modèles 3.1 Itération 1: Lazypredict() 3.2 Itération 2: Performances prédictives 3.3 Itération 3: Modèles retenus
4. Métrique des modèles
5. Optimisation des paramètres: Tuning
6. Interprétabilité du Modèle

B- DEEP LEARNING

1. Preprocessing
2. Choix des modèles
3. Test des modèles 3.1 Modèle 1: CNN 3.2 Modèle 2: LSTM
4. Analyse des métriques

IV. Bilan et Perspectives

Contexte du Projet

Le projet fil rouge clôturant notre formation continue de data scientiste chez Datascientest porte sur **la Structural Protein Classification**. Ce projet se fixe pour objectif la prédiction de la structure des protéines avec des algorithmes de **Machine Learning** et de Deep **Learning**. Le choix de ce projet qui ne faisait pas partie du catalogue de projets proposés par nos formateurs a été laborieux:

- dans la compréhension du sujet
- sa mise en oeuvre
- son interprétation des résultats etc.

Néanmoins la volumétrie des données ainsi que les diverses catégories des variables nous ont permis de comprendre vers quels algorithmes de machine learning et de deep learning orienter nos recherches et appliquer les connaissances acquises au cours de la formation.

Quels sont les intérêts pour la classification de la structure des protéines?

Les protéines sont des macromolécules complexes, elles sont les plus abondantes des molécules organiques des cellules et constituent à elles seules plus de 50% du poids à sec des êtres vivants.

L'intérêt majeur de la prédiction de la structure des protéines trouve son application en **bio-informatique, en biotechnologie et en médecine** notamment dans la fabrication des enzymes et de nouveaux médicaments. Le projet communautaire **CAMEO3D** évalue les performances continues des serveurs web dédiés à la prédiction de la structure des protéines.

AlphaFold est une IA développée par google pour la prédiction de la structure des protéines et cette application d'intelligence artificielle a notamment servi dans la prédiction de la structure des protéines du **SARS-COV-2**

La prédiction de la structure des protéines est l'inférence de la structure tridimensionnelle d'une protéine à partir de sa séquence d'acides aminés c'est-à-dire la prédiction de son pliage et de sa structure secondaire et tertiaire de sa structure primaire **wikipedia** Dans le cadre de ce projet, il s'agit de prédire la fonction de la molécule, c'est-à-dire la classe à laquelle elle appartient.

I. ANALYSE DES DONNEES

1. Source des données

Les données sont issues de la **Protein Data Bank** (PDB). Il s'agit d'un ensemble de données extrait de la base du groupe d'experts du Research Collaboratory for Structural Bioinformatics. Dans cette base sont stockées les coordonnées atomiques des protéines et des informations relatives à d'autres macromolécules. Le dataset est composé des deux fichiers:

protéine : regroupe toutes les variables décrivant les propriétés physiques des amino-acides (protéines)

séquences: regroupe toutes les variables décrivant les séquences associées à chaque acide aminé (protéine)

1. Dataset proteins

Features	Types	Description
Classification	object	classification de la molécule
crystallizationMethod	object	Méthode de cristallisation de la protéine
crystallizationTempK	float64	température à laquelle la protéine cristallise
densityMatthews	float64	volume cristallin par unité de poids moléculaire
densityPercentSo	float64	% de la densité du solvant dans la protéine
experimentalTechnique	object	Méthode d'obtention de la structure protéique
pdbsDetails	object	divers détails sur la protéine
phValue	object	Ph de la solution (acide, basique, neutre)
publicationYear	float64	année de publication de la structure protéique

Features	Types	Description
residueCount	float64	nombre d'acides aminés dans la séquence
resolution	float64	qualité du cristal contenant la protéine
structureMolecularWeight	float64	masse moléculaire en kilo dalton
structureId	float64	id de la structure
macromoleculeType	object	Type de macromolécule (Protein,DNA, RNA)

2. Dataset sequence

Features	Types	Description
residueCount	float64	nombre d'acides aminés dans la séquence
macromoleculeType	object	Type de macromolécule (Protein,DNA, RNA)
structureId	float64	id de la structure
chainId	float64	id de la sequence
sequence	object	sequence de la proteine

3 variables communes des 2 tables ont permis la fusion en un tableau unique

2. Nettoyage des données

Après la fusion des datasets, nous avons constaté un nombre non négligeable des données manquantes sur certaines variables. Ainsi pour garder un maximum des données, nous avons appliqué la stratégie de gestion des données manquantes suivante:

- remplacement par la médiane des variables numériques
- remplacement par la mode des variables catégorielles

Nous avons supprimé les variables n'ayant aucun impact sur l'analyse des données :

pdpxDetails ----> diverses propriétés de la proteine sans impact sur l'analyse

chainId ----> id sans impact sur l'analyse des données

structureId ----> id sans impact sur l'analyse des données

Nous avons également supprimé la variable **sequence** pour la **partie Machine Learning** et nous l'avons gardé comme input feature en deuxième partie de ce projet consacré au **Deep Learning**.

Les variables quantitatives telles que *residueCount*, *structureMolecularWeight*, *densityMatthews*, *resolution* étaient fortement asymétriques, une correction logarithmique a été opérée.

Target Feature

La variable classification est notre variable cible (target feature), elle comprend **4989 modalités !** . Ne pouvant pour des raisons pratiques faire la classification de toutes ces modalités, dans le modele final nous n'avons gardé que **les classes de fréquence supérieure à 5000 valeurs**

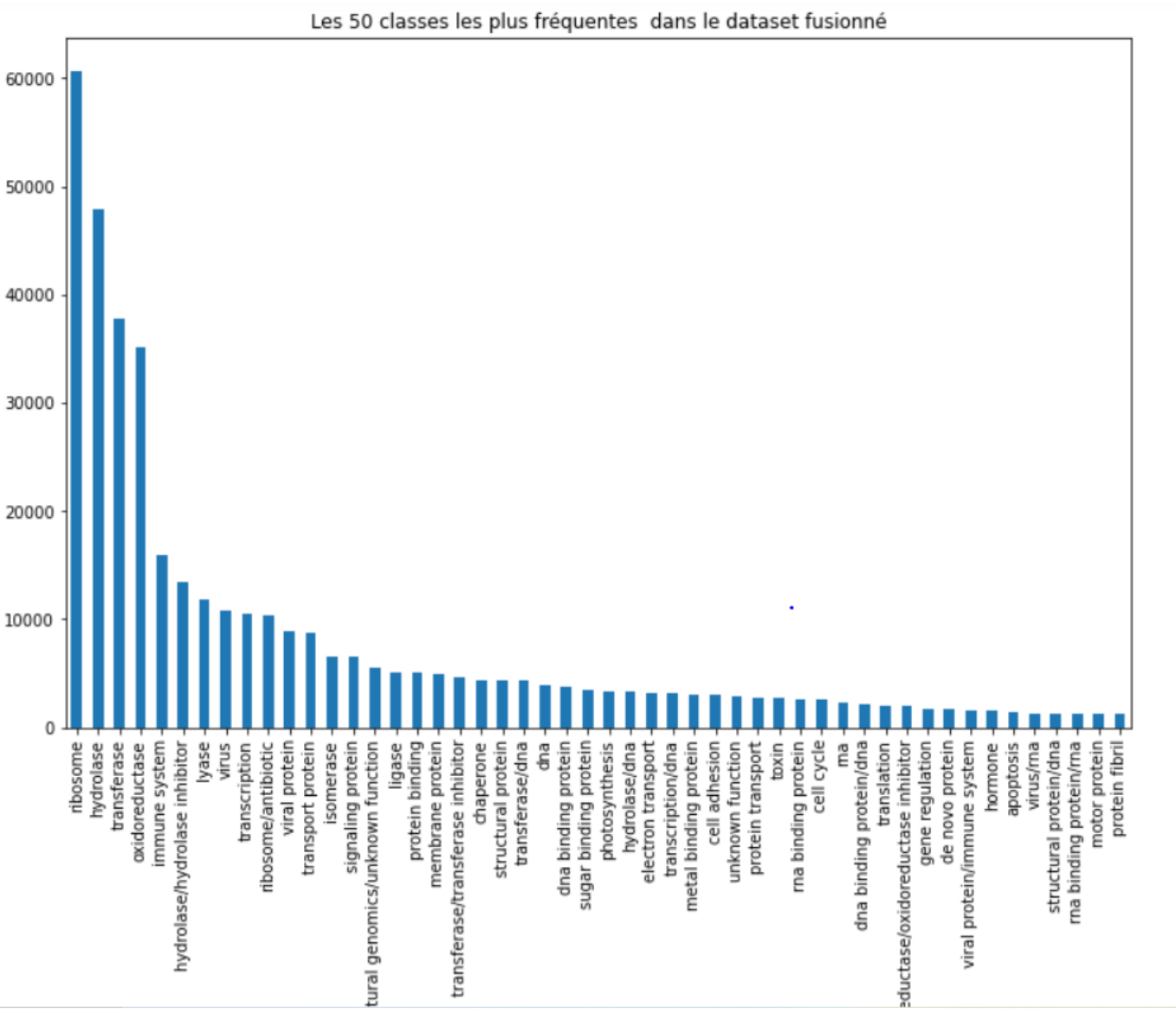
Pour terminer, toutes les données manquantes restantes ont été supprimées.

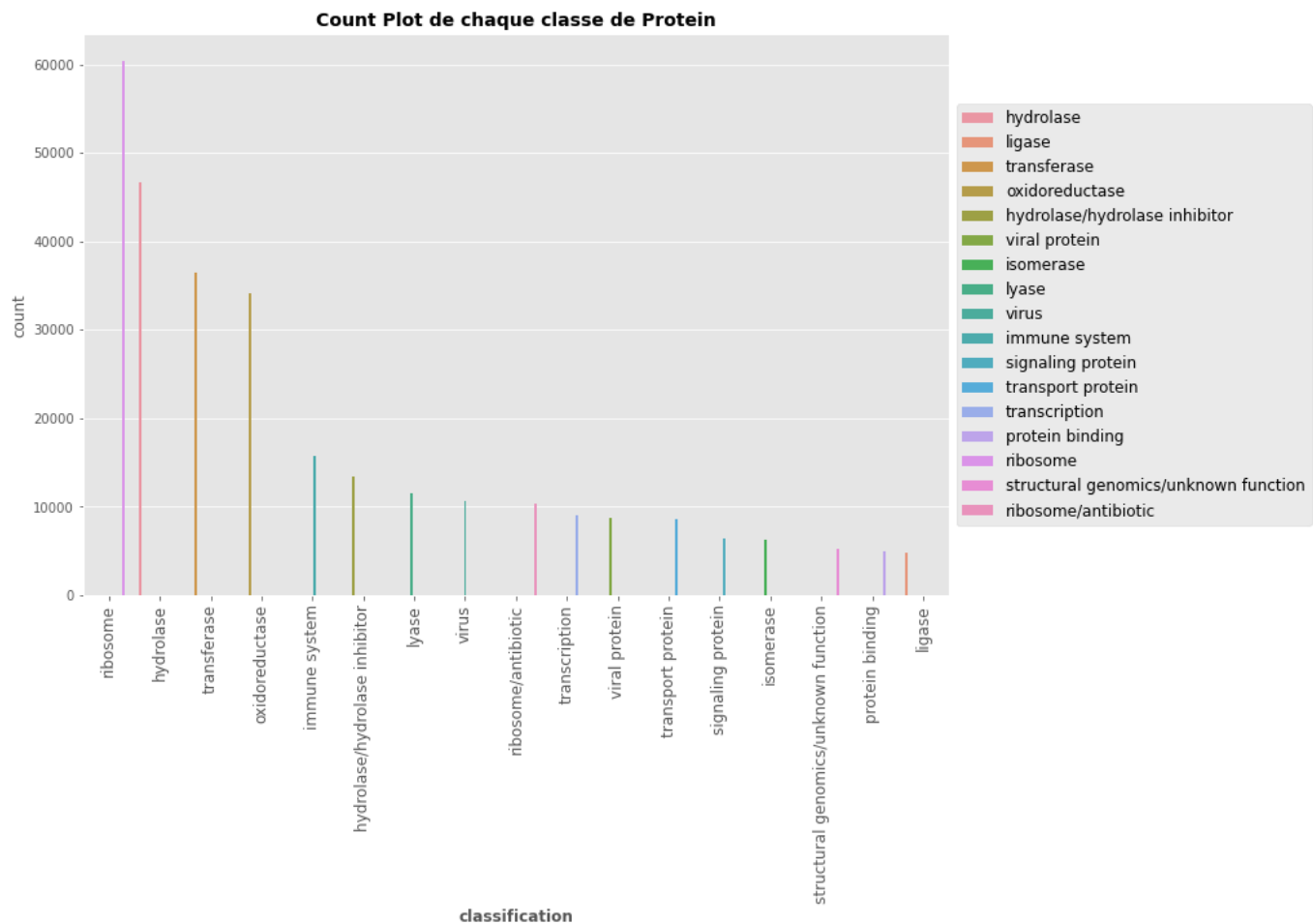
Cette stratégie nous a permis d'avoir un dataset final propre et prêt pour l'analyse des données avec les **algorithmes ML** et le **DL**.

3. Analyse Exploratoire des données

• Classification

La variable classification est notre variable cible, elle comprend 4989 modalités. Nous avons restreint les classes à prédire à 17 correspondant aux classes ayant une fréquence supérieure à 5000 valeurs.



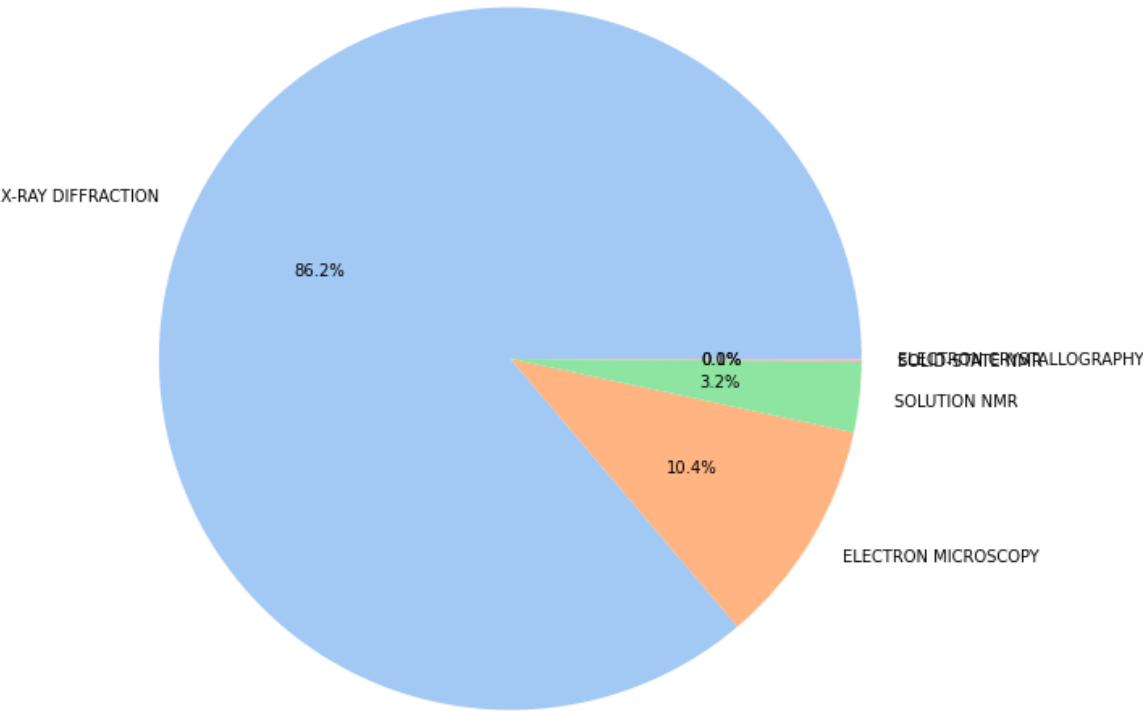


- **Technique d'extraction de la Proteine**

Avant de séquencer la proteine, son extraction s'obtient par plusieurs types de technique (32 techniques) dont la plus utilisée est la **X-RAY-DIFFRACTION** représentant à elle seule **86 %** des techniques utilisées dans la base, deux autres techniques s'ajoutent à celle-ci formant **99%** des solutions techniques utilisées dans le dataset.

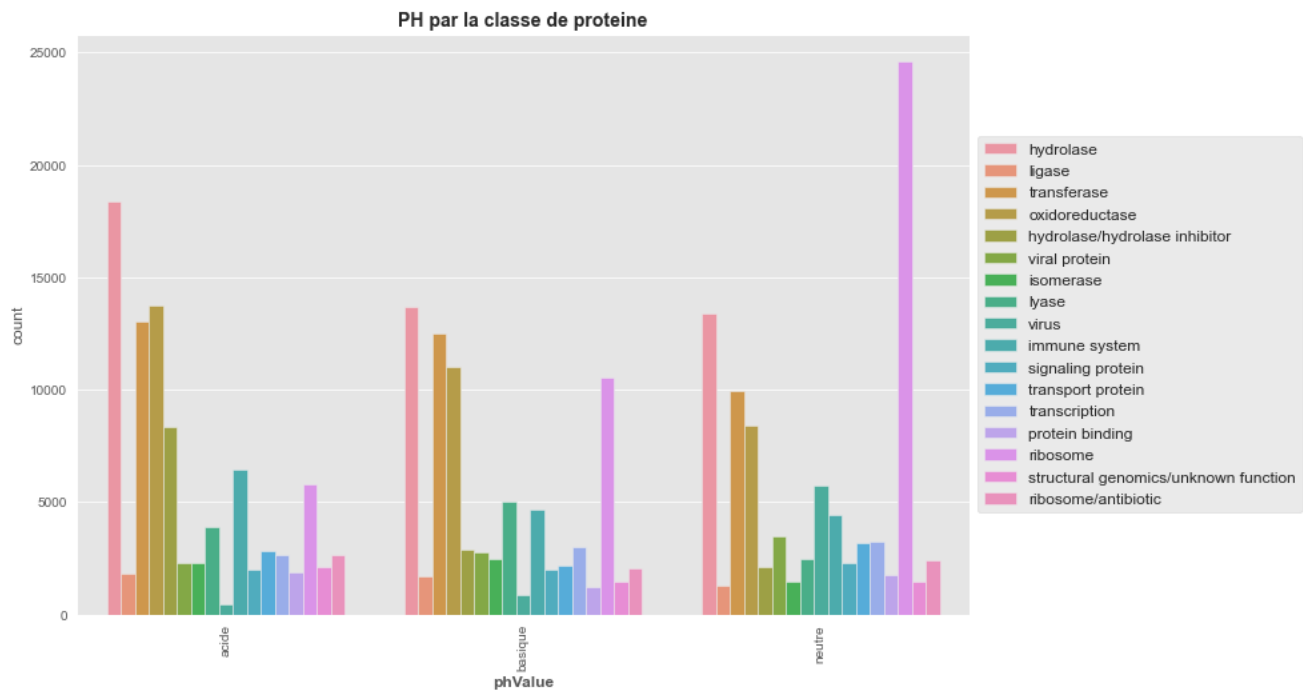
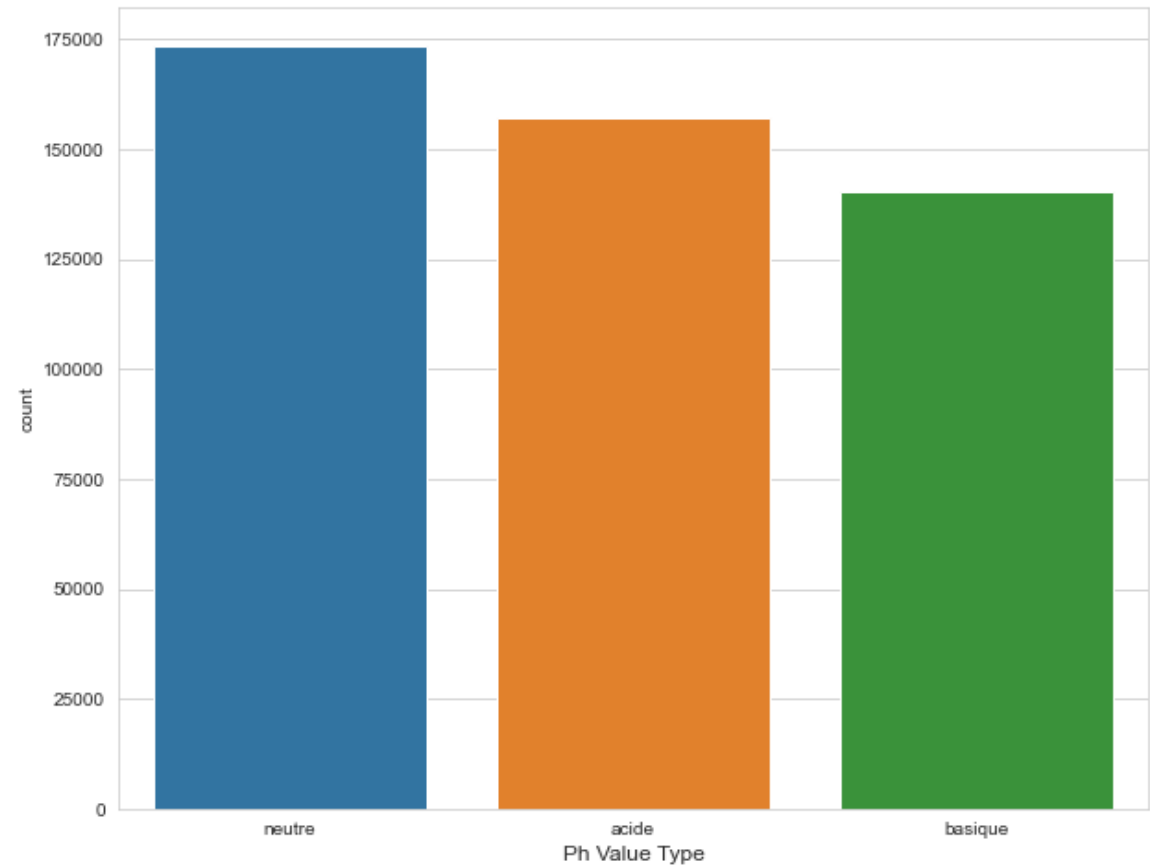
Ce constat nous a amené à restreindre le dataset final à ces 3 techniques et regrouper les autres dans la modalité "other_techniques". Cette étape est importante car elle permet d'obtenir la séquence de la proteine et donc de la classification de sa structure lui affectant un rôle dans la cellule.

Les 5 techniques d'extraction de la proteine les plus fréquentes



• Valeur du Ph

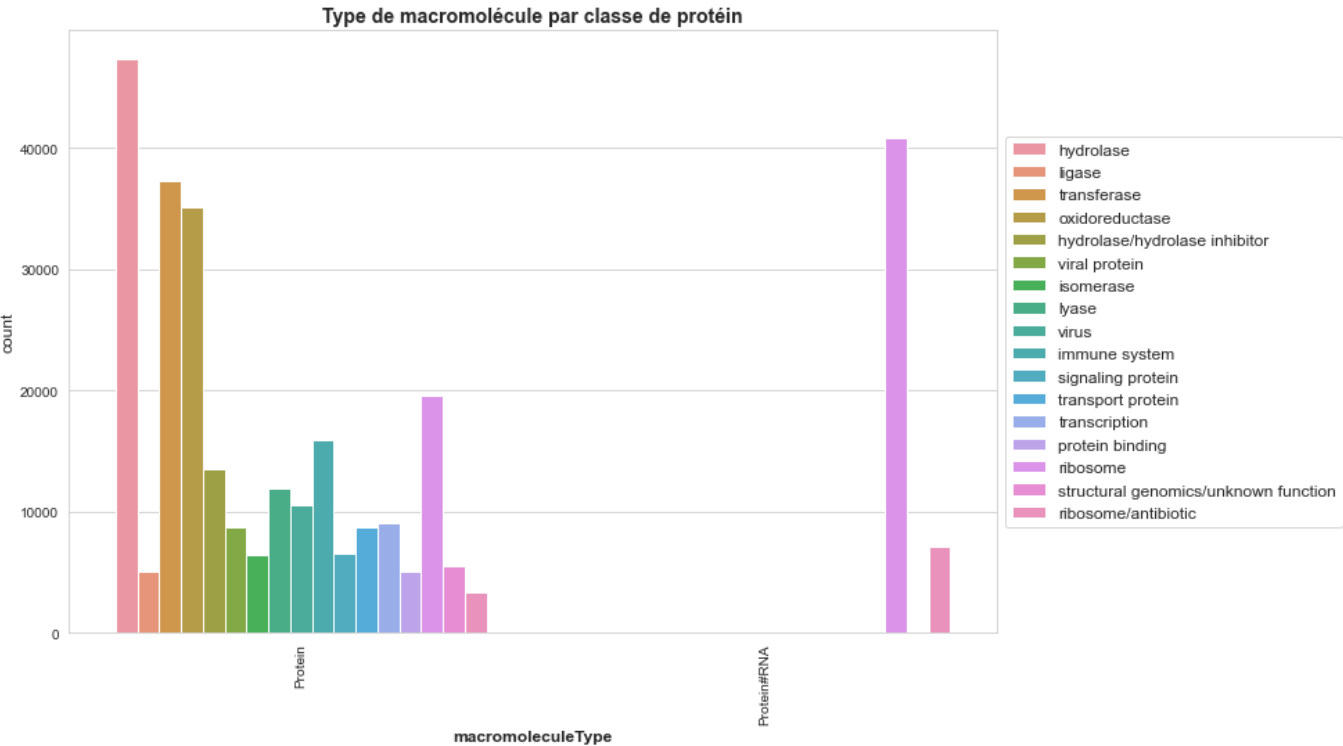
Les liaisons protéiques peuvent être changées voire disloquées par des agents de dénaturation tels que le PH (potentiel hydrogène). Globalement les protéines de la base sont plus neutres que basiques (supérieur à 7). Leur acidité (inférieur à 7) étant à cheval entre les deux premières. Cette variable a été recodée en variable catégorielle.



Nous n'observons pas de tendance dans la distribution des classes de proteine selon les valeurs du PH à part la classe **ribosome** qui ressort beaucoup plus neutre, **l'hydrolase** plus acidulé

• Type de Macromolécule

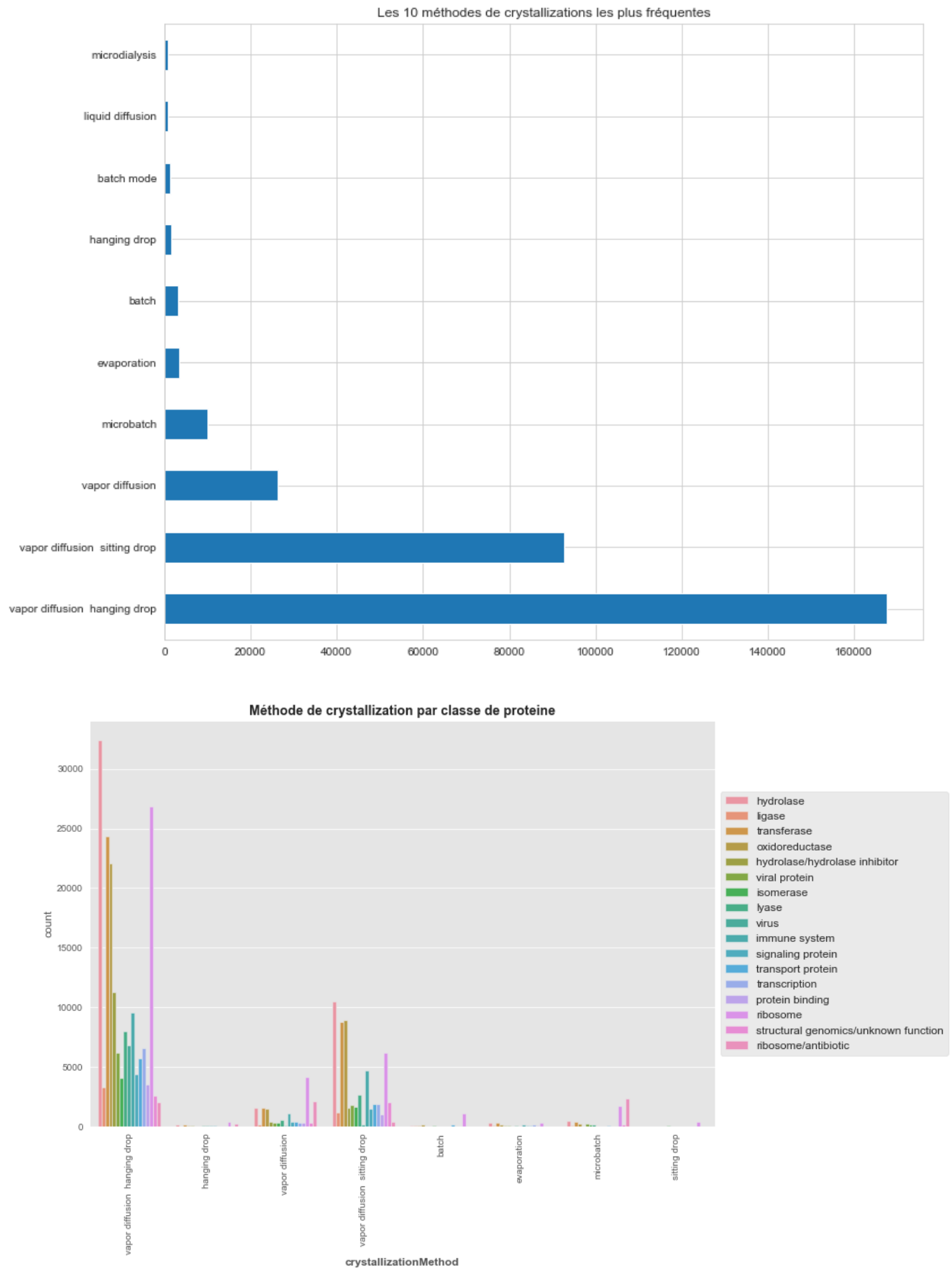
Nous avons environ **80%** du dataset qui est composé de protéine, 18% composé de protéines avec ARN ou ADN. Nous avons fait le choix de garder ces 3 modalités.



Comme attendu nous constatons plus de protéine dans le dataset final

- **Méthode de cristallisation**

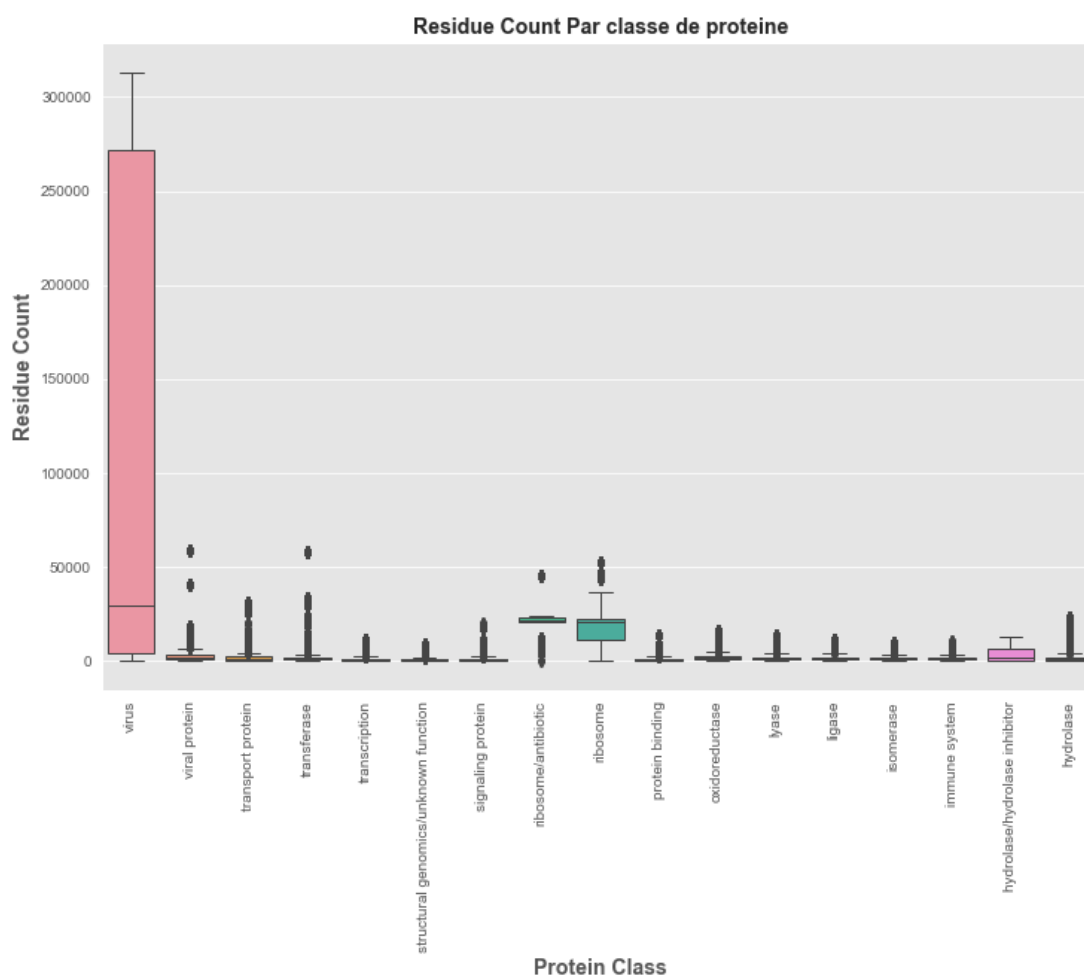
Nous avons dénombrés 418 techniques de cristallisation parmi lesquelles 4 à elles seules représentent 94% du dataset total



Les méthodes de vaporisation comptent pour 90% du dataset, le microbatch 3%. Les techniques de cristallisation par la vapeur sont comme attendues les plus fréquentes dans le dataset final.

- **residueCount par classe de protéine**

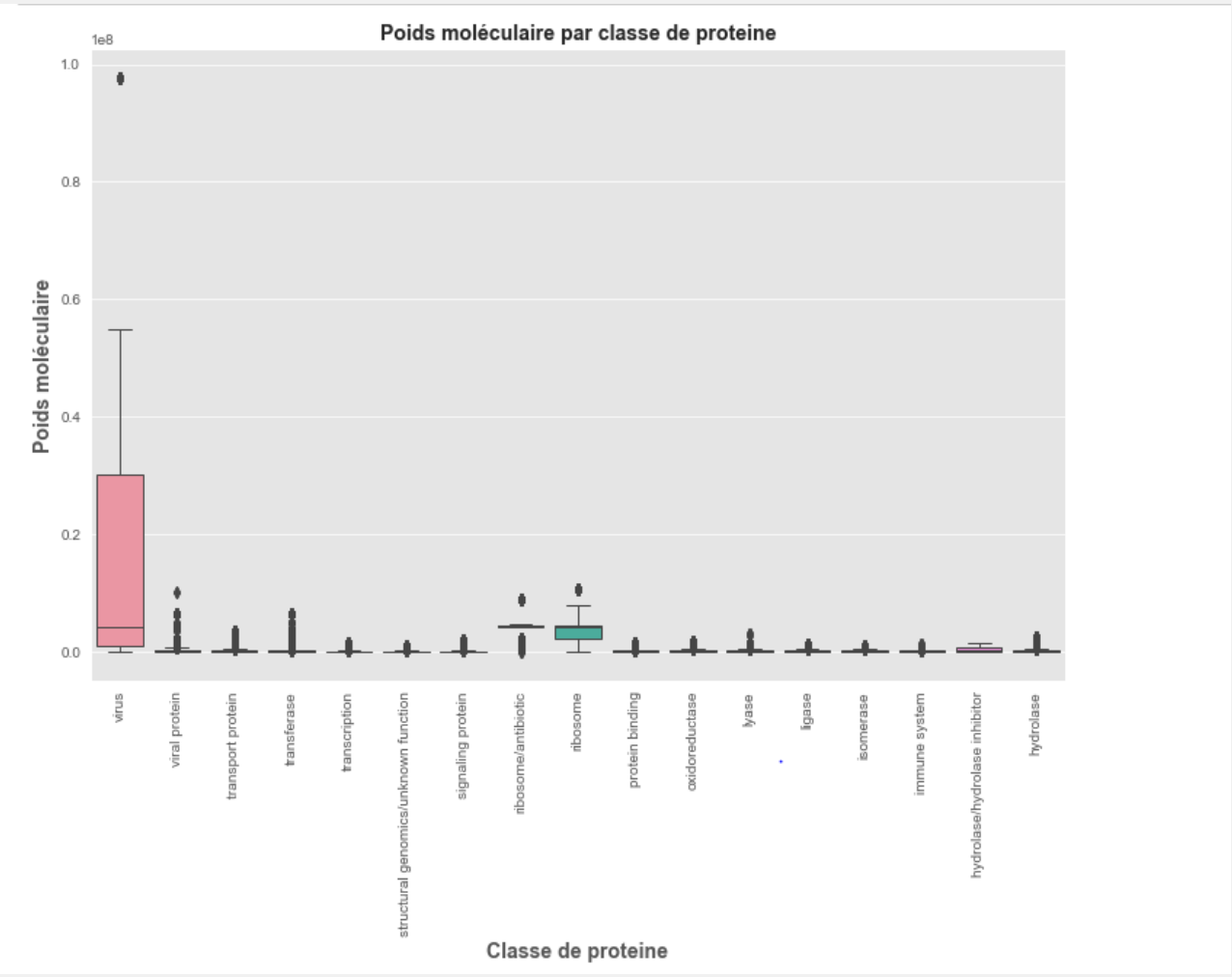
Nous avons comparé les 17 classes de protéine selon le nombre d'acides aminés (feature `residueCount`), nous n'observons pas des différences particulières entre les classes hormis les classes **virus**, **ribosome** et **ribosome/antibiotic**



- **structureMolecularWeight par classe de protéine**

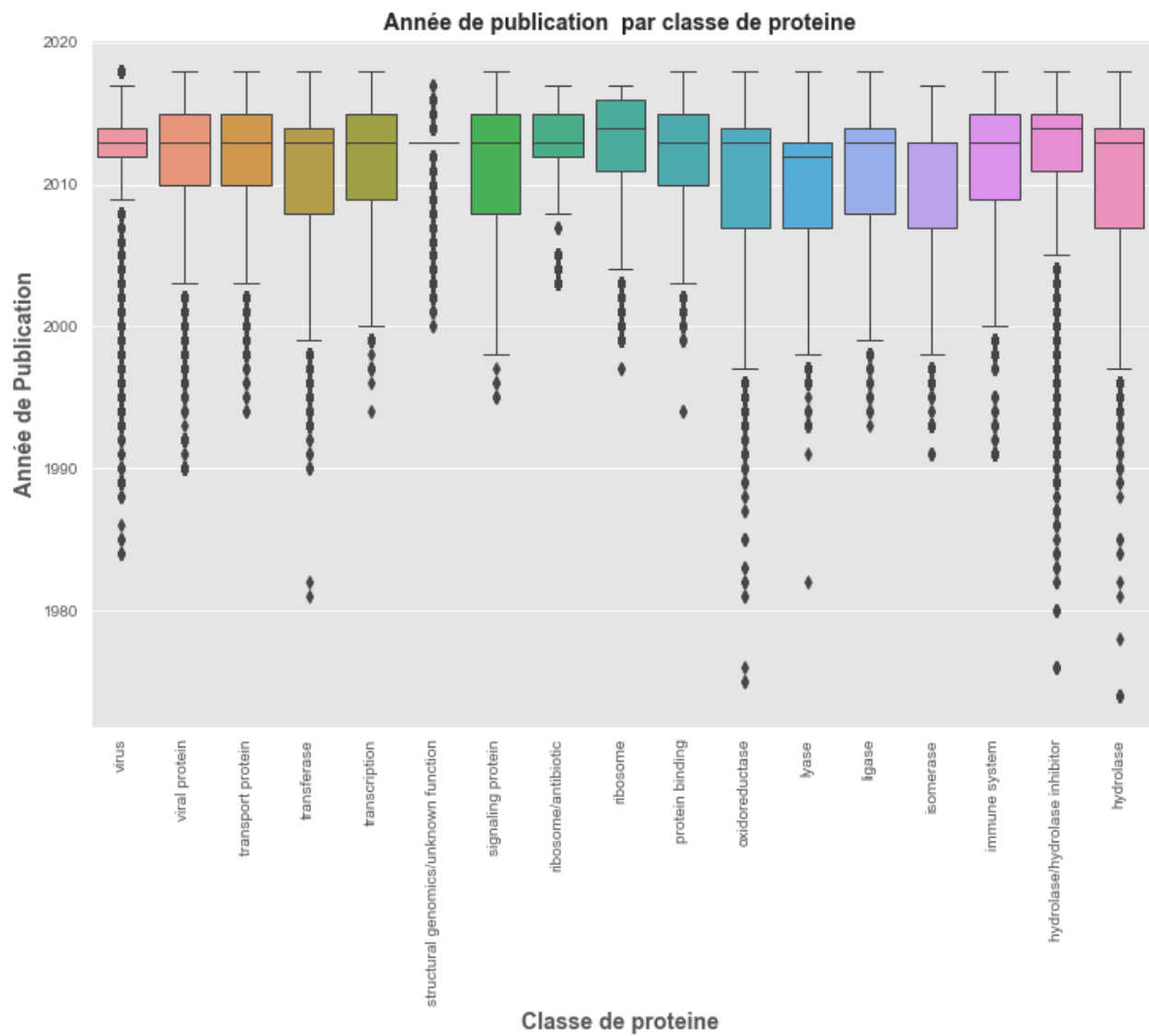
Nous avons fait aussi la comparaison des classes de protéine avec le poids de la structure moléculaire, comme précédemment nous n'observons pas des différences particulières entre les classes, les classes précédentes ressortent(**virus**, **ribosome** et **ribosome/antibiotic**) montrant une corrélation entre ces

deux variables sur ces 3 classes.



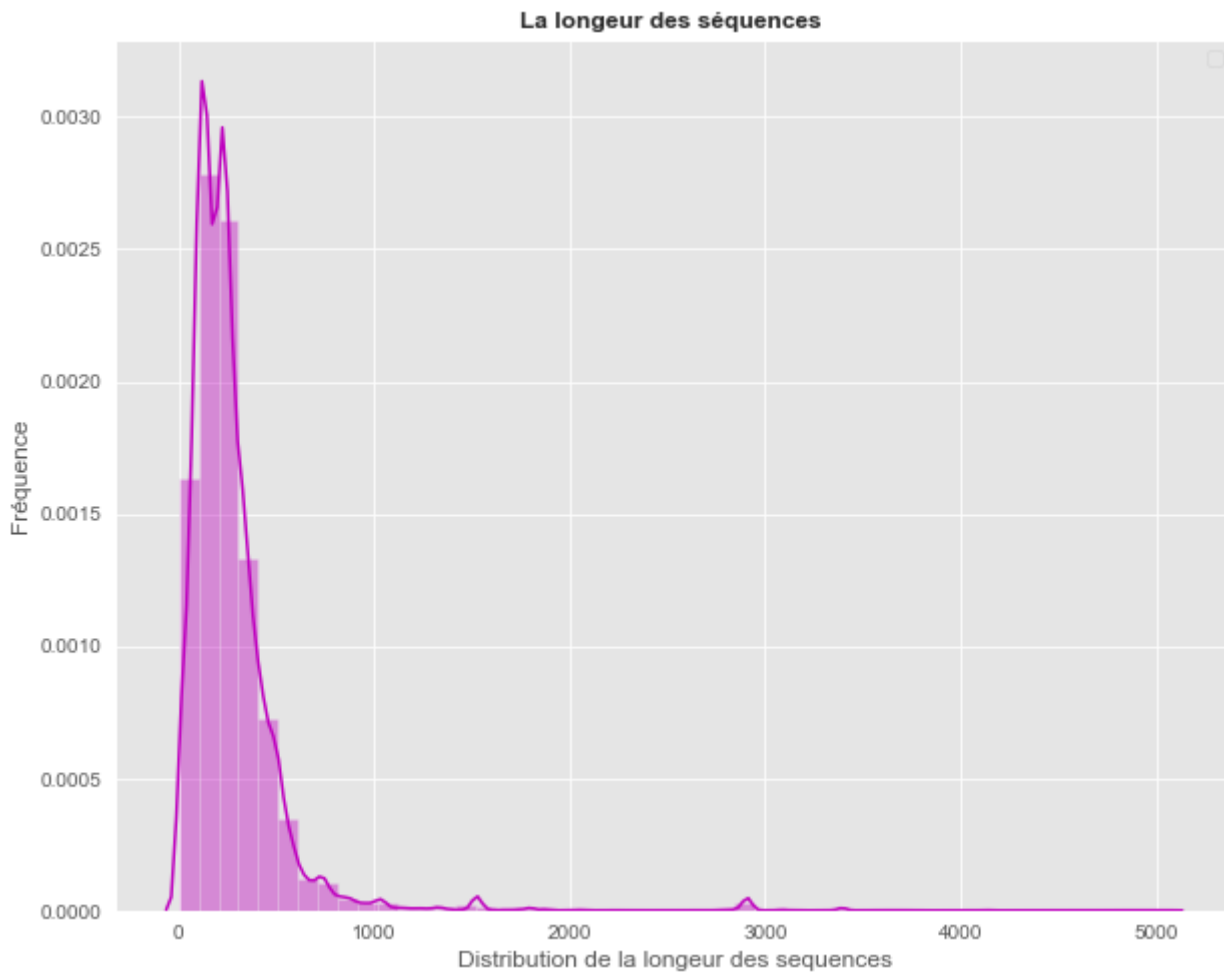
- **publicationYear par classe de protéine**

Cette variable bien qu'intervenant peu dans la prédiction de la structure des protéines nous renseigne sur l'intense activité de la communauté des chercheurs du RCB et l'intérêt grandissant de la thématique des protéines depuis 2015 (année médiane)

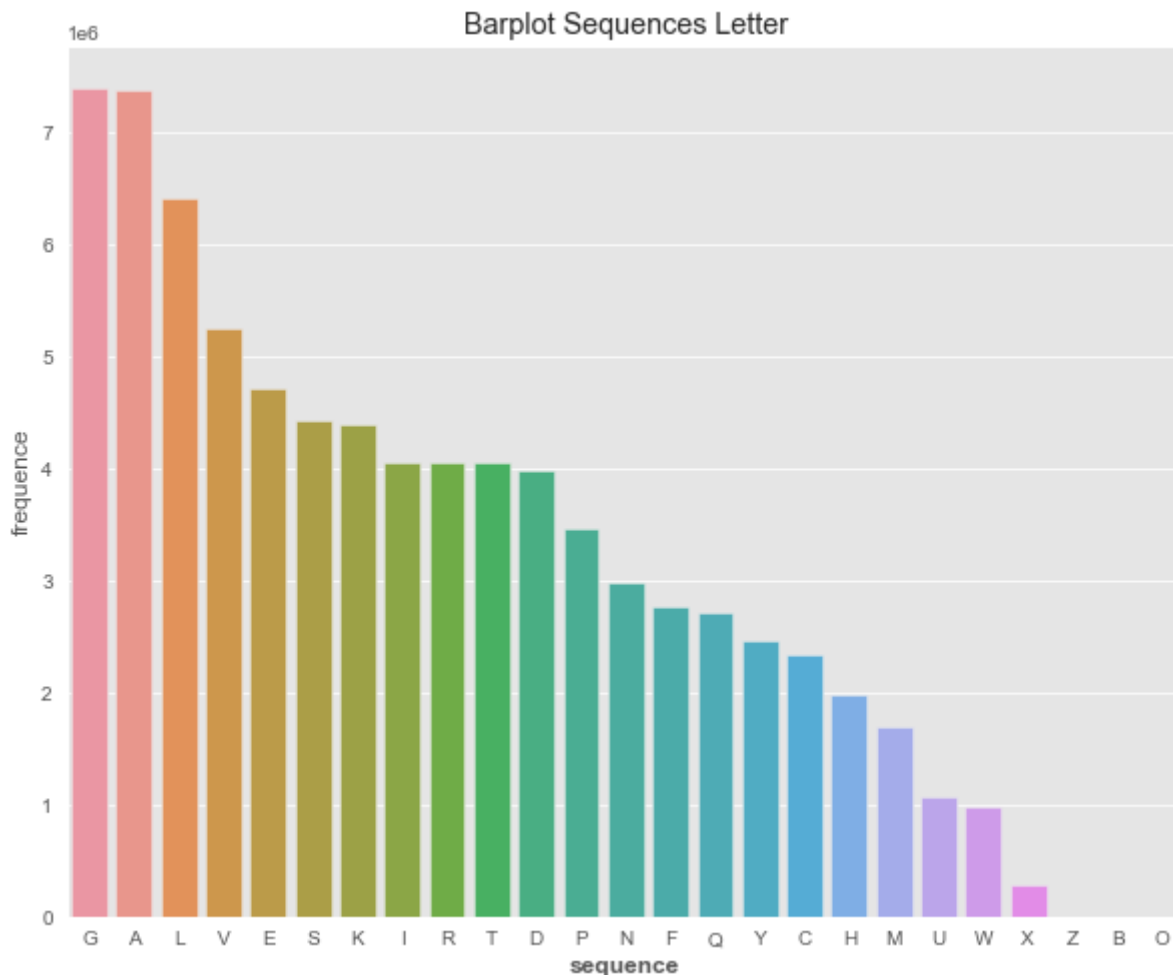


- Sequence Feature

Notre objectif en deuxième partie de ce projet étant le deep learning, la variable sequence sera utilisée comme seule variable explicative; elle est composée de 25 lettres de longueur différente



Les lettres **G** et **A** sont les fréquentes.



II. Méthodologie

Nous avons fait le choix d'adopter une double démarche dans l'analyse des données :

- la prédiction de la structure des protéines avec des algorithmes de **Machine Learning** par l'analyse des caractéristiques physiques des protéines
- la prédiction de la structure des protéines avec des algorithmes de **Deep Learning** en nous basant uniquement cette fois sur l'analyse des séquences protéiques

III. Modélisation

A. Machine Learning

1. Preprocessing des données:

L'exploration et la phase d'analyse terminées, nous avons nettoyé et préparé le jeu de données pour l'apprentissage. Cette étape nous a permis:

- d'identifier les variables non indispensables dans le dataset final, variables que nous avons supprimées
- de gérer les données manquantes en les remplaçant par la médiane pour les variables numériques, la mode pour les variables catégorielles et supprimer les lignes des variables pour lesquelles les données

étaient manquantes.

- de réduire les modalités de certaines variables en les regroupant selon le nombre d'observation dans une nouvelle modalité.
- de définir une stratégie de réduction des classes de protéines à prédire en les ramenant à 17 classes au lieu de 4989 en ne tenant compte que des modalités ayant plus de 5000 observations. Nous avons ainsi gardé 64% des classes protéiques dans le dataset final.
- de discrétiser les variables catégorielles à plusieurs modalités.
- Nous avons procédé par élimination récursive des variables (RFE) selon leur poids en utilisant l'algorithme des forêts aléatoires en ne gardant que 8 variables dans le dataset final.

Les diverses étapes listées ci-dessus nous ont permis d'avoir un jeu données final de **310.000 lignes (68% du dataset initial)** et **8 features**

2. Métriques des tests

Nous avons utilisé les métriques suivantes pour la classification de la structure protéique:

Accuracy: cette métrique nous a permis d'obtenir rapidement la performance de nos modèles

Rapport de classification: les performances fines sur chaque classe protéique ont été obtenues avec cette métrique

Matrices de confusion: en détails, cette métrique nous a permis de comprendre les classifications correctes et incorrectes de des modèles.

3. Itération des Modèles

Nous avons utilisé la librairie *lazypredict* pour gérer le choix difficile de la pléthore des algorithmes de classification existant. En effet, cette bibliothèque par sa simplicité d'utilisation avec peu de codes et sans réglage des hyperparamètres nous a permis de faire le choix des meilleurs modèles à retenir, modèles auxquels seront appliqués des paramètres d'optimisations par la suite.

• 3.1 Itération 1

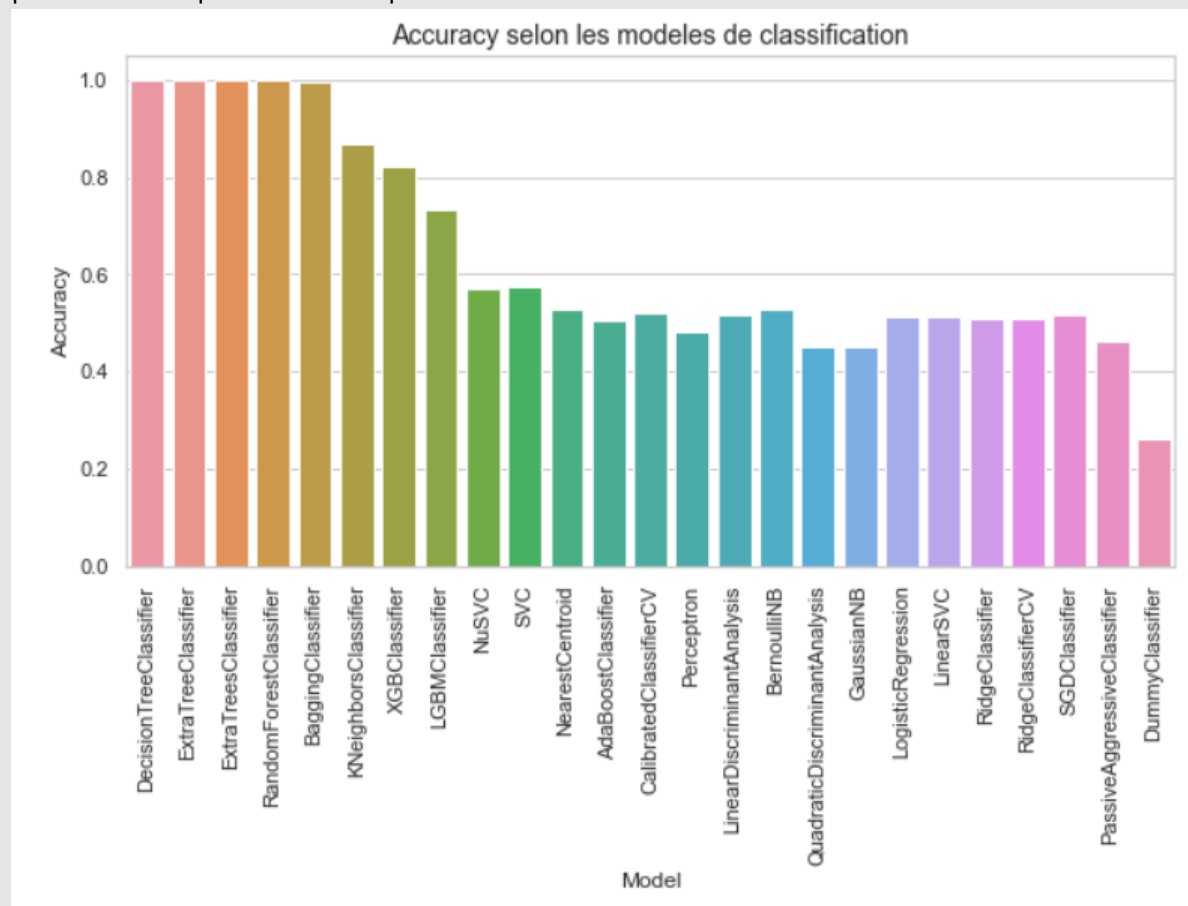
Nous n'avons pas eu à choisir les modèles, le choix a été opéré automatiquement par le package avec des métriques de performance des différents modèles en ordre décroissant.

Out[100]:

	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
Model					
DecisionTreeClassifier	1.00	1.00	None	1.00	0.47
ExtraTreeClassifier	1.00	1.00	None	1.00	0.29
ExtraTreesClassifier	1.00	1.00	None	1.00	8.43
RandomForestClassifier	1.00	1.00	None	1.00	11.41
BaggingClassifier	1.00	1.00	None	1.00	2.13
KNeighborsClassifier	0.87	0.88	None	0.87	6.76
XGBClassifier	0.82	0.83	None	0.82	10.26
LGBMClassifier	0.73	0.75	None	0.73	2.32
NuSVC	0.57	0.61	None	0.57	658.37
SVC	0.58	0.59	None	0.57	246.55
NearestCentroid	0.53	0.56	None	0.53	0.34
AdaBoostClassifier	0.50	0.54	None	0.50	4.39
CalibratedClassifierCV	0.52	0.54	None	0.51	169.20
Perceptron	0.48	0.54	None	0.47	0.43
LinearDiscriminantAnalysis	0.51	0.53	None	0.49	0.82
BernoulliNB	0.53	0.53	None	0.47	0.23
QuadraticDiscriminantAnalysis	0.45	0.53	None	0.37	0.33
GaussianNB	0.45	0.52	None	0.37	0.28
LogisticRegression	0.51	0.52	None	0.48	5.04
LinearSVC	0.51	0.52	None	0.48	62.16
RidgeClassifier	0.51	0.52	None	0.47	0.29
RidgeClassifierCV	0.51	0.52	None	0.47	0.35
SGDClassifier	0.52	0.50	None	0.39	0.79
PassiveAggressiveClassifier	0.46	0.49	None	0.41	0.49
DummyClassifier	0.26	0.25	None	0.26	0.21

Les algorithmes classiques de classification (regression logistique, SGDClassifier, lassoClassifier...) se sont montrés peu performants au contraire des algorithmes d'ensemble qui avec un temps d'entrainement relativement courts affichent des métriques élevées. Sur les 25 modèles testés 4 se sont montrés particulièrement performants et seront utilisés en troisième itération avec des

paramètres d'optimisation adaptés.



• 3.2 Itération 2: Performances prédictives

Pour arrêter définitivement notre choix sur les modèles retenus, nous avons testé avec le package Lazypredict les performances prédictives des 25 modèles et nous avons constaté que ces performances sont les mêmes que celles enregistrées en apprentissage.

Out[103]:

	Accuracy	Balanced Accuracy	ROC AUC	F1 Score	Time Taken
Model					
DecisionTreeClassifier	1.00	1.00	None	1.00	0.47
ExtraTreeClassifier	1.00	1.00	None	1.00	0.29
ExtraTreesClassifier	1.00	1.00	None	1.00	8.43
RandomForestClassifier	1.00	1.00	None	1.00	11.41
BaggingClassifier	1.00	1.00	None	1.00	2.13
KNeighborsClassifier	0.87	0.88	None	0.87	6.76
XGBClassifier	0.82	0.83	None	0.82	10.26
LGBMClassifier	0.73	0.75	None	0.73	2.32
NuSVC	0.57	0.61	None	0.57	658.37
SVC	0.58	0.59	None	0.57	246.55
NearestCentroid	0.53	0.56	None	0.53	0.34
AdaBoostClassifier	0.50	0.54	None	0.50	4.39
CalibratedClassifierCV	0.52	0.54	None	0.51	169.20
Perceptron	0.48	0.54	None	0.47	0.43
LinearDiscriminantAnalysis	0.51	0.53	None	0.49	0.82
BernoulliNB	0.53	0.53	None	0.47	0.23
QuadraticDiscriminantAnalysis	0.45	0.53	None	0.37	0.33
GaussianNB	0.45	0.52	None	0.37	0.28
LogisticRegression	0.51	0.52	None	0.48	5.04
LinearSVC	0.51	0.52	None	0.48	62.16
RidgeClassifier	0.51	0.52	None	0.47	0.29
RidgeClassifierCV	0.51	0.52	None	0.47	0.35
SGDClassifier	0.52	0.50	None	0.39	0.79
PassiveAggressiveClassifier	0.46	0.49	None	0.41	0.49
DummyClassifier	0.26	0.25	None	0.26	0.21

- Difficultés rencontrées:

Dans ces deux itérations, la principale difficulté rencontrée a été l'entraînement des modèles avec le package [Lazypredict](#). Ce package malgré son utilité a des sérieux problèmes de mise à jour qui ne facilitent pas son utilisation notamment les dépendences liées à d'autres bibliothèques. Pour sa mise en oeuvre, il nous a fallu l'installer dans un environnement virtuel dédié.

• 3.3 Itération 3: Modèles retenus

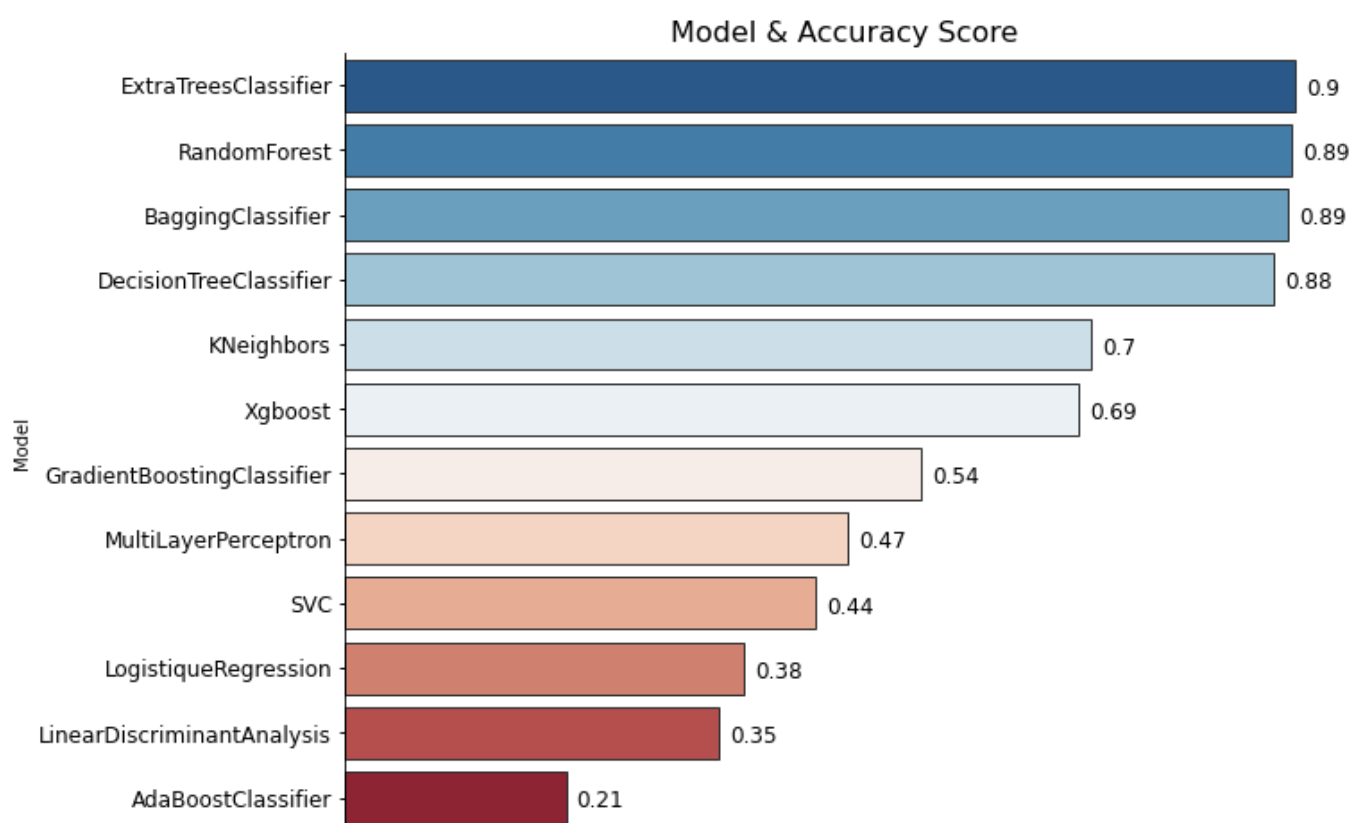
- **Objectif**

confirmer ou infirmer les prédictions du package Lazypredict des modèles prédéfinis précédemment.

- **Résultats**

Sur les 12 modèles testés, 4 se dégagent nettement avec des accuracy élevés, ces modèles sont essentiellement des algorithmes de classification d'ensemble:

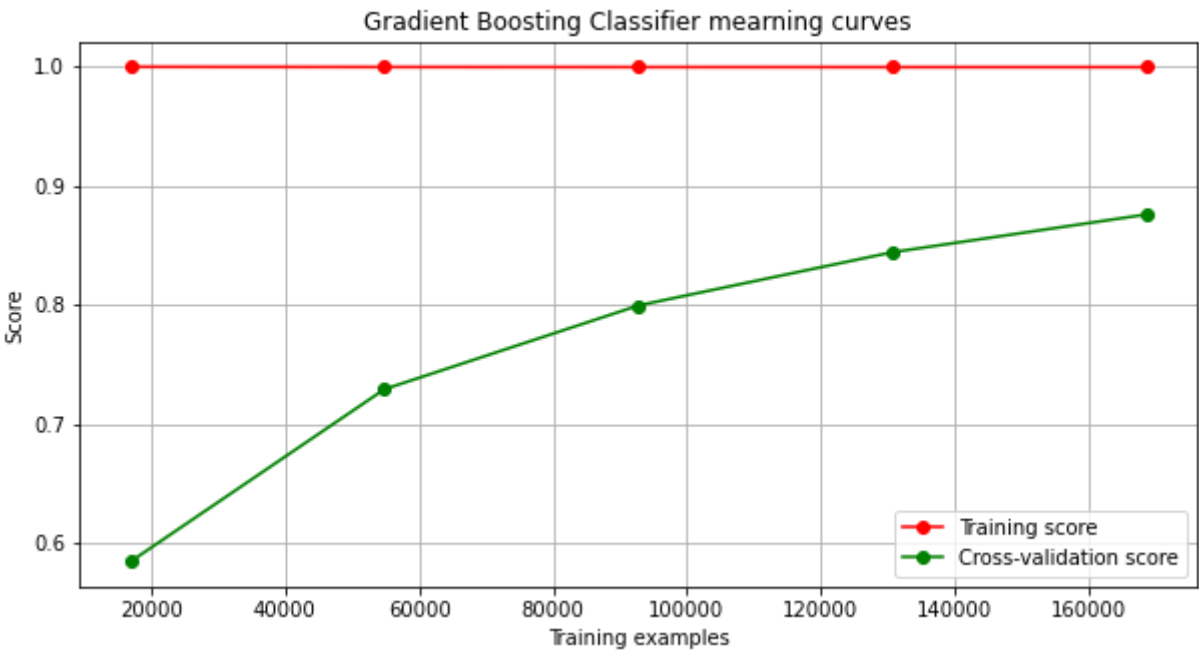
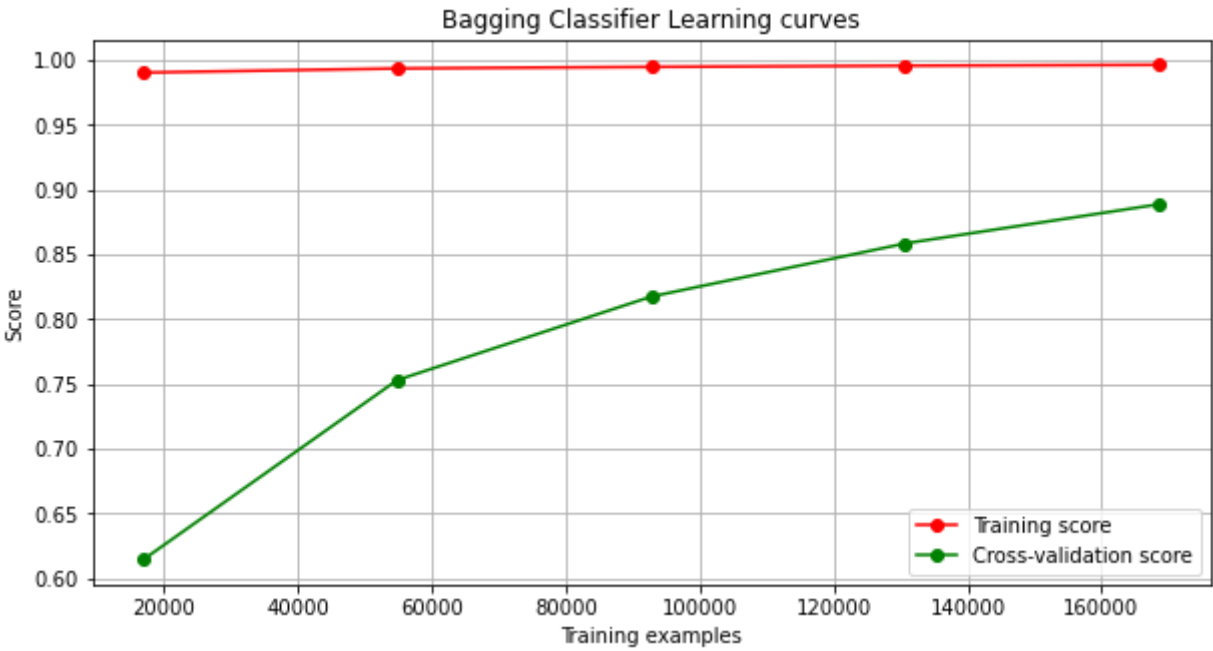
- ExtraTreesClassifier,
- RandomForest,
- Le Bagging Classifier,
- DecisionTreeClassifier

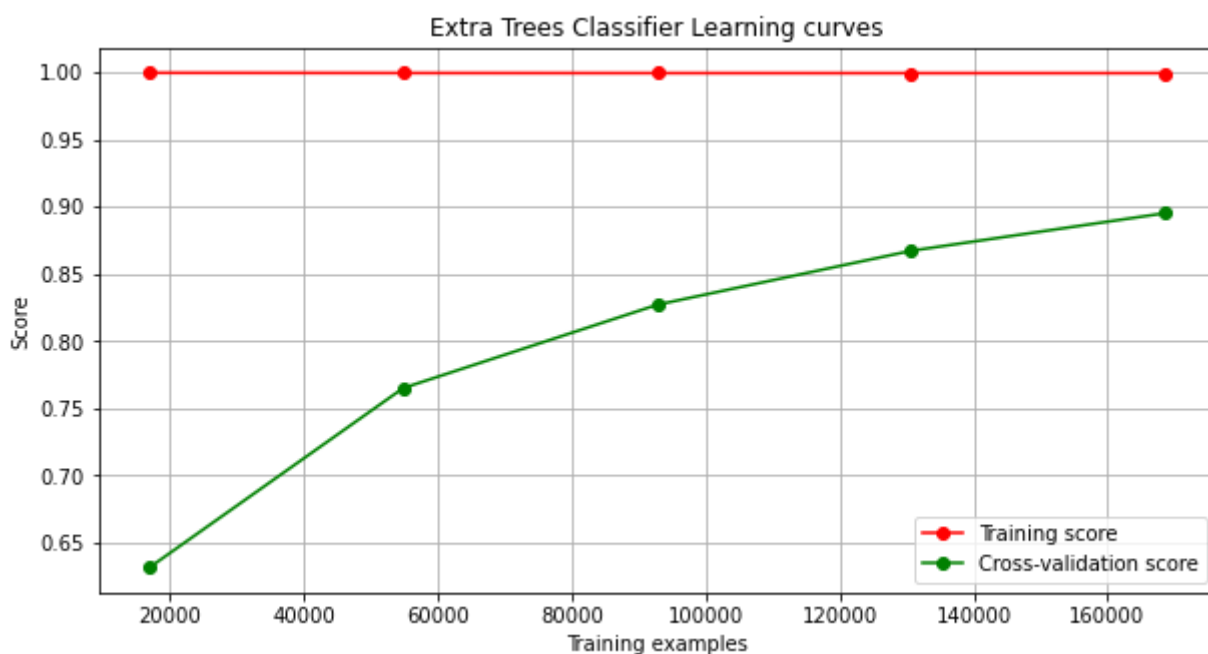
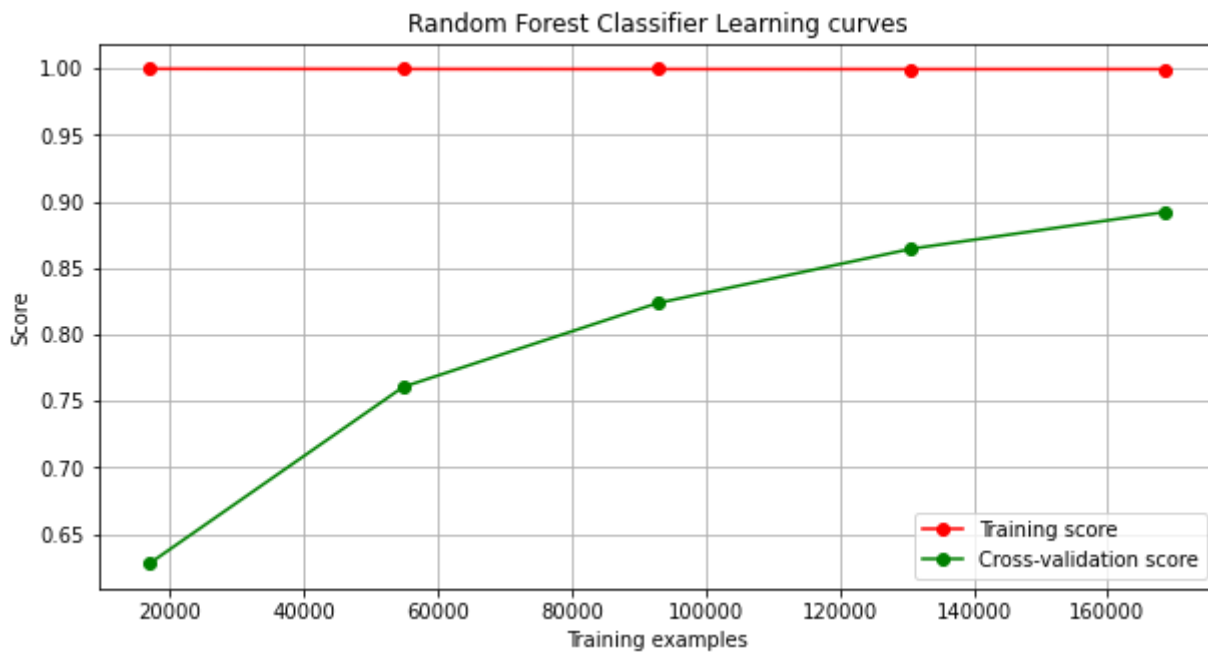


Le modèle **ExtraTrees** s'est montré un plus performant que les autres modèles.

- **Problèmes**

Les performances élevées de nos 4 modèles retenus en terme d'accuracy suscitent de la prudence dans l'interprétation des résultats. En effet, un surapprentissage de nos modèles pourrait avoir pour effet une difficulté de généralisation de ceux-ci sur de nouveaux jeux de données en terme prédictif. Un des meilleurs moyens de voir un effet de surapprentissage sur l'échantillon d'apprentissage et plus globalement sur la taille du jeu de données des modèles est la représentation de ceux-ci en courbe d'apprentissage





La lecture de l'allure des courbes d'apprentissage ne nous permet pas d'exclure complètement un effet d'overfitting (surapprentissage)

- **Difficultés rencontrées**

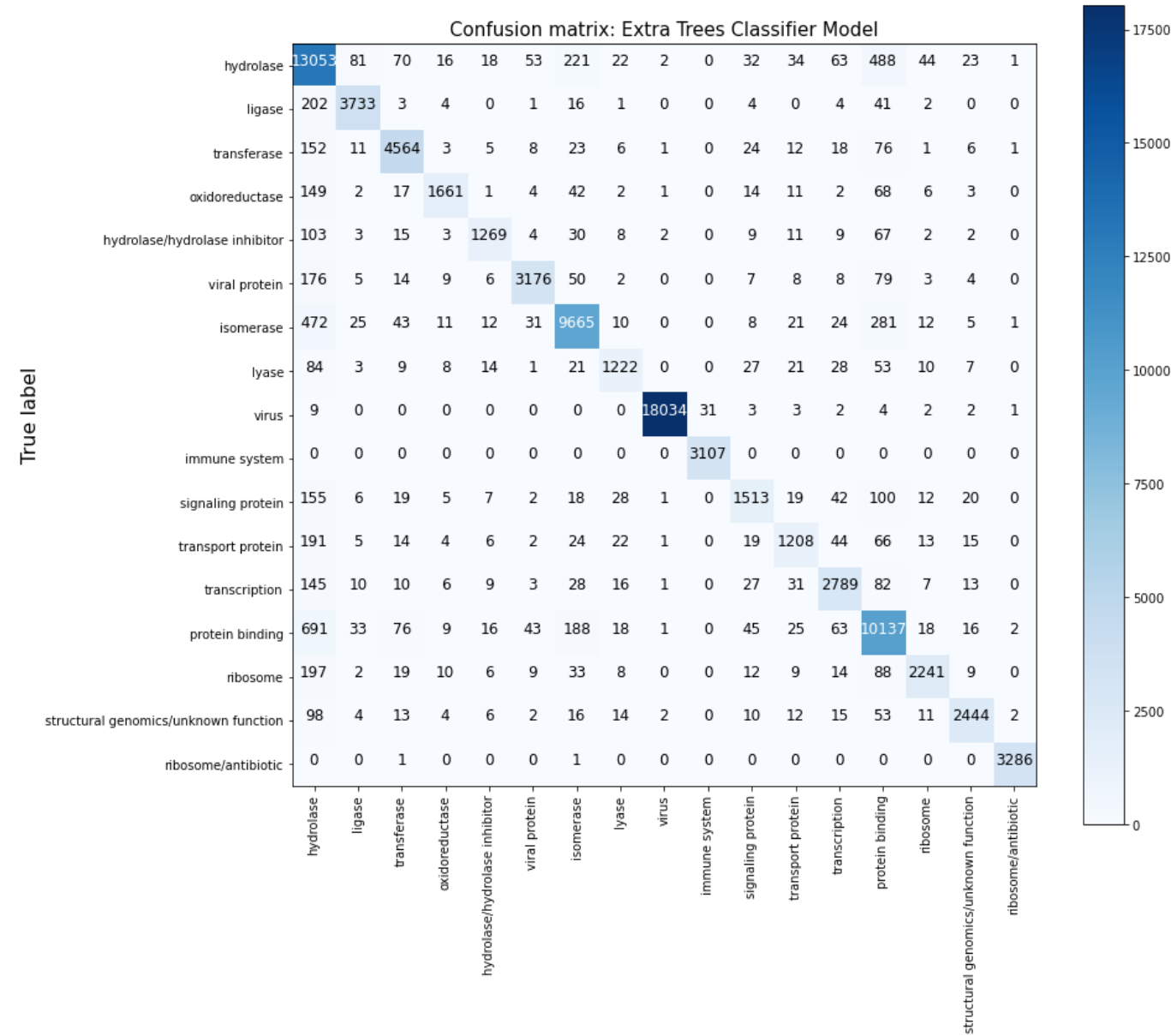
Les difficultés rencontrées ont été principalement le temps d'apprentissage des modèles sur une machine de 8 Go de RAM (16 heures), ce temps a été ramené à 4h50' dans une autre machine plus adaptée avec 16 Go de RAM

4. Métriques des différents modèles

Sur les 4 modèles testés, nous avons ajouté un meta modèle qui est le **voting classifieur**

- **ExtraTreesClassifier**

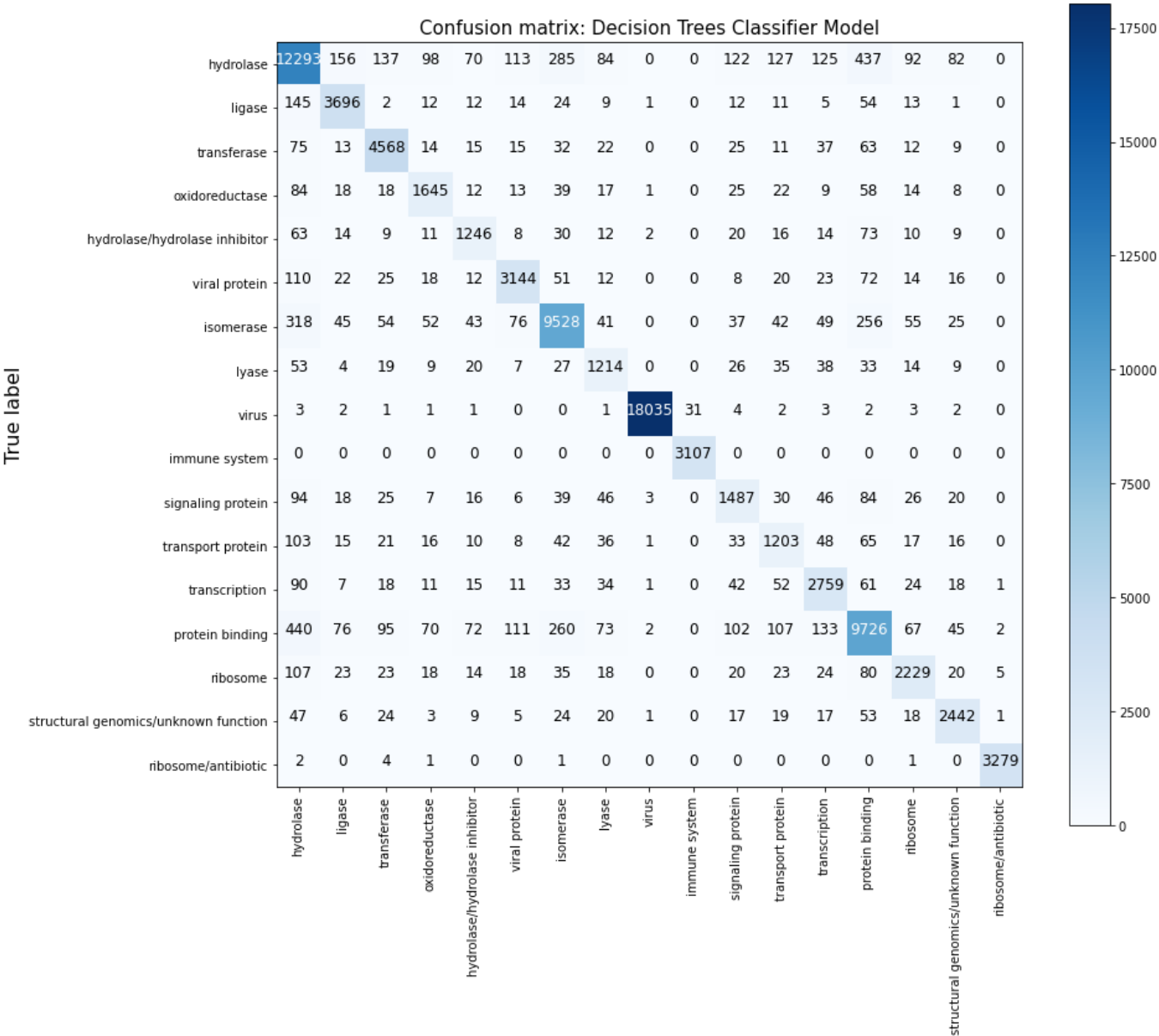
	precision	recall	f1-score	support
hydrolase	0.83	0.92	0.87	14221
hydrolase/hydrolase inhibitor	0.95	0.93	0.94	4011
immune system	0.93	0.93	0.93	4911
isomerase	0.94	0.84	0.89	1983
ligase	0.93	0.82	0.87	1537
lyase	0.96	0.90	0.93	3547
oxidoreductase	0.93	0.91	0.92	10621
protein binding	0.88	0.81	0.84	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	1.00	3107
signaling protein	0.86	0.78	0.82	1947
structural genomics/unknown function	0.86	0.74	0.79	1634
transcription	0.90	0.88	0.89	3177
transferase	0.87	0.89	0.88	11381
transport protein	0.94	0.85	0.89	2657
viral protein	0.95	0.91	0.92	2706
virus	1.00	1.00	1.00	3288
accuracy			0.92	90327
macro avg	0.92	0.89	0.90	90327
weighted avg	0.92	0.92	0.92	90327



Predicted label

- **DecisionTreeClassifier**
-

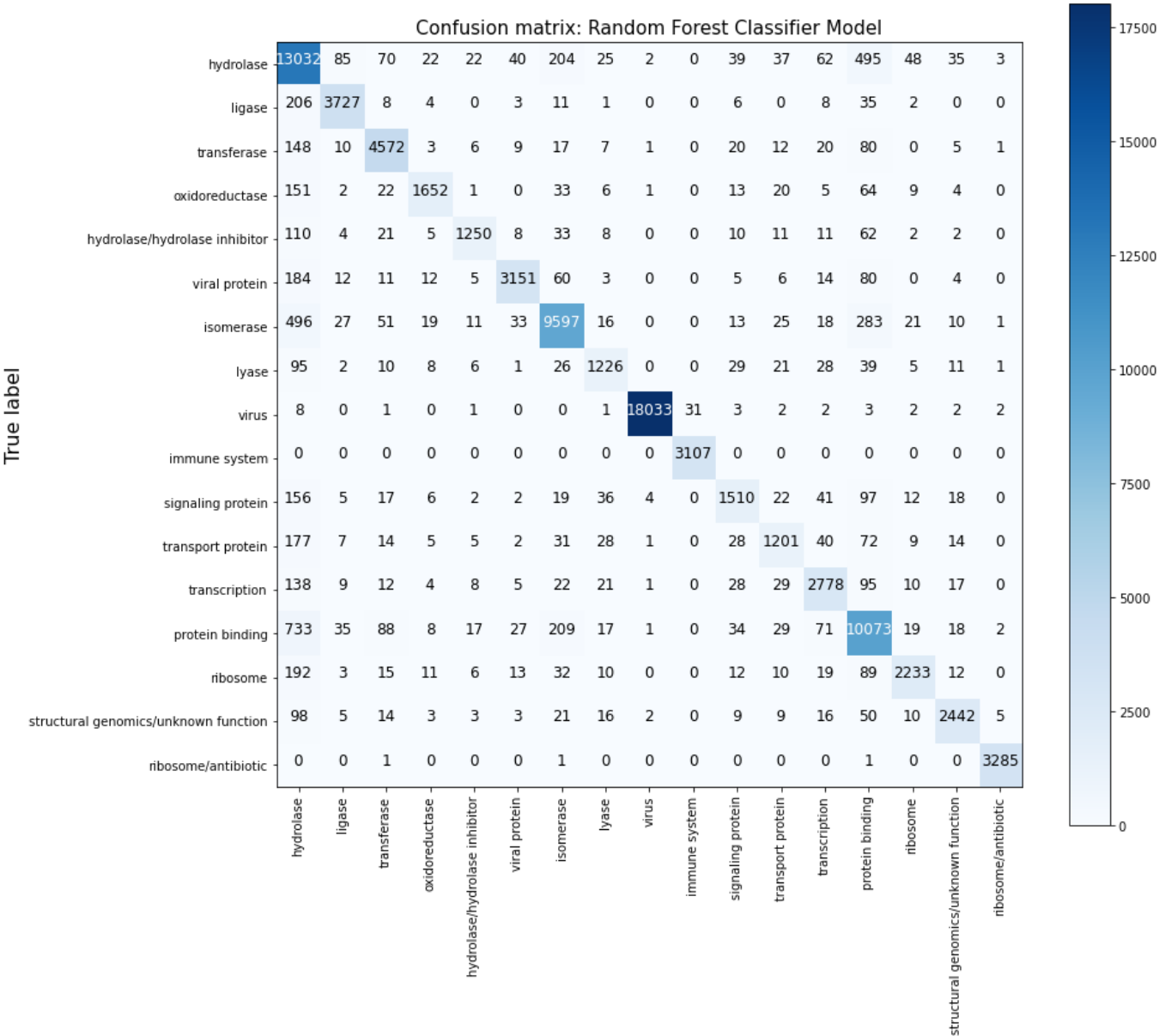
	precision	recall	f1-score	support
hydrolase	0.88	0.86	0.87	14221
hydrolase/hydrolase inhibitor	0.90	0.92	0.91	4011
immune system	0.90	0.93	0.92	4911
isomerase	0.84	0.83	0.83	1983
ligase	0.81	0.81	0.81	1537
lyase	0.88	0.89	0.88	3547
oxidoreductase	0.91	0.90	0.90	10621
protein binding	0.74	0.81	0.77	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	0.99	3107
signaling protein	0.75	0.76	0.76	1947
structural genomics/unknown function	0.70	0.73	0.72	1634
transcription	0.83	0.87	0.85	3177
transferase	0.87	0.85	0.86	11381
transport protein	0.84	0.84	0.84	2657
viral protein	0.89	0.90	0.90	2706
virus	1.00	1.00	1.00	3288
accuracy			0.90	90327
macro avg	0.87	0.88	0.87	90327
weighted avg	0.90	0.90	0.90	90327



Predicted label

- RandomForestClassifier
-

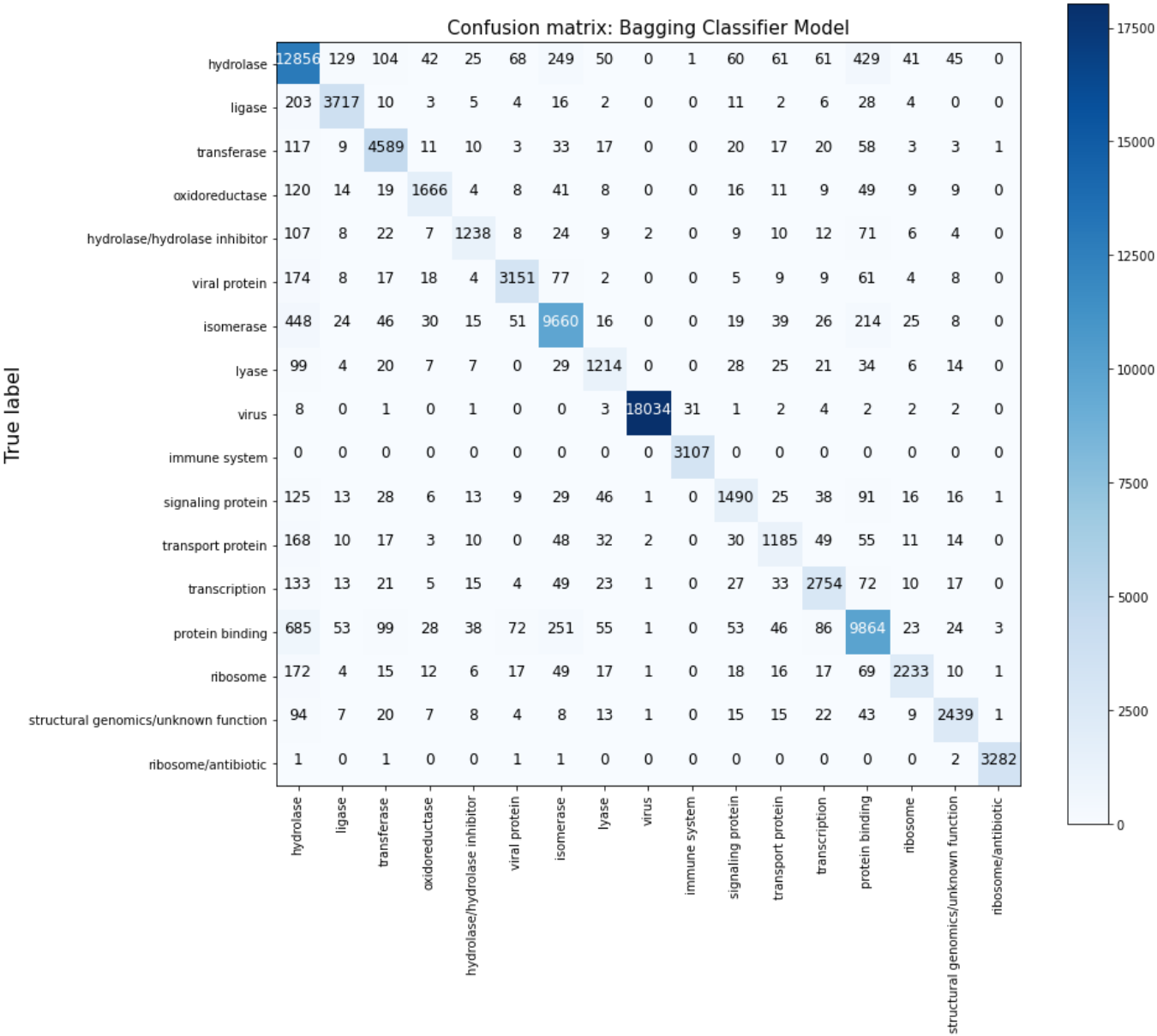
	precision	recall	f1-score	support
hydrolase	0.82	0.92	0.87	14221
hydrolase/hydrolase inhibitor	0.95	0.93	0.94	4011
immune system	0.93	0.93	0.93	4911
isomerase	0.94	0.83	0.88	1983
ligase	0.92	0.81	0.86	1537
lyase	0.95	0.89	0.92	3547
oxidoreductase	0.93	0.91	0.92	10621
protein binding	0.87	0.81	0.84	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	1.00	3107
signaling protein	0.85	0.77	0.81	1947
structural genomics/unknown function	0.84	0.74	0.79	1634
transcription	0.89	0.87	0.88	3177
transferase	0.87	0.89	0.88	11381
transport protein	0.94	0.84	0.89	2657
viral protein	0.95	0.90	0.92	2706
virus	1.00	1.00	1.00	3288
accuracy			0.92	90327
macro avg	0.92	0.88	0.90	90327
weighted avg	0.92	0.92	0.92	90327



Predicted label

- **BaggingClassifier**
-

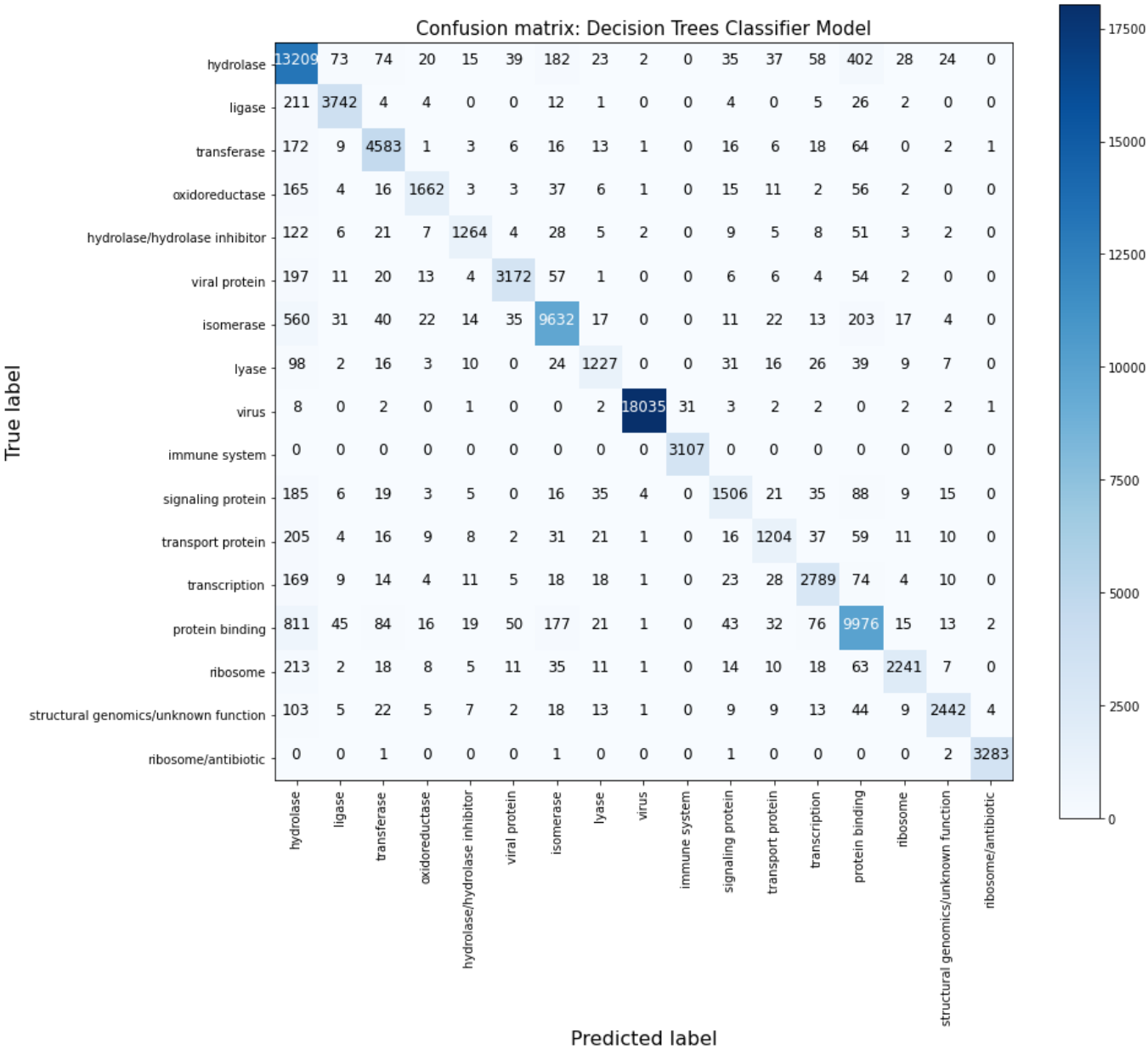
	precision	recall	f1-score	support
hydrolase	0.83	0.90	0.87	14221
hydrolase/hydrolase inhibitor	0.93	0.93	0.93	4011
immune system	0.91	0.93	0.92	4911
isomerase	0.90	0.84	0.87	1983
ligase	0.90	0.81	0.85	1537
lyase	0.92	0.89	0.90	3547
oxidoreductase	0.92	0.91	0.91	10621
protein binding	0.83	0.80	0.81	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	1.00	3107
signaling protein	0.82	0.77	0.79	1947
structural genomics/unknown function	0.79	0.74	0.76	1634
transcription	0.87	0.88	0.87	3177
transferase	0.89	0.86	0.88	11381
transport protein	0.92	0.84	0.88	2657
viral protein	0.94	0.90	0.92	2706
virus	1.00	1.00	1.00	3288
accuracy			0.91	90327
macro avg	0.90	0.88	0.89	90327
weighted avg	0.91	0.91	0.91	90327



predicted label

- **VotingClassifier**
-

	precision	recall	f1-score	support
hydrolase	0.87	0.89	0.88	14221
hydrolase/hydrolase inhibitor	0.92	0.93	0.92	4011
immune system	0.92	0.94	0.93	4911
isomerase	0.88	0.84	0.86	1983
ligase	0.84	0.82	0.83	1537
lyase	0.91	0.89	0.90	3547
oxidoreductase	0.92	0.91	0.91	10621
protein binding	0.78	0.81	0.80	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	0.99	3107
signaling protein	0.79	0.77	0.78	1947
structural genomics/unknown function	0.75	0.74	0.74	1634
transcription	0.85	0.88	0.86	3177
transferase	0.88	0.88	0.88	11381
transport protein	0.87	0.85	0.86	2657
viral protein	0.91	0.91	0.91	2706
virus	1.00	1.00	1.00	3288
accuracy			0.91	90327
macro avg	0.89	0.88	0.89	90327
weighted avg	0.91	0.91	0.91	90327



- Résultats des Performances Globales des Modèles

ExtraTrees Classifier Model

Accuracy Training Set:	1.0 %
F1 Score Training Set:	1.0 %

Accuracy Testing Set:	0.92 %
F1 Score Testing set :	0.921 %

#####

RandomForest Classifier Model

Accuracy Training Set:	1.0 %
F1 Score Training Set:	1.0 %

Accuracy Testing Set:	0.918 %
F1 Score Testing set :	0.918 %

#####

Bagging Classifier Model

Accuracy Training Set:	0.997 %
F1 Score Training Set:	0.997 %

Accuracy Testing Set:	0.913 %
F1 Score Testing set :	0.913 %

#####

DecisionTree Classifier Model

Accuracy Training Set:	1.0 %
F1 Score Training Set:	1.0 %

Accuracy Testing Set:	0.903 %
F1 Score Testing set :	0.903 %

#####

Voting Classifier Model

Accuracy Training Set:	1.0 %
F1 Score Training Set:	1.0 %

Accuracy Testing Set:	0.913 %
F1 Score Testing set :	0.913 %

- Optimisation des paramètres: Tuning

Le modèle qui s'est montré un peu plus performant sur les 4 modèles retenus reste

ExtraTreesClassifier (Extremely Randomized Trees) dont l'accuracy en échantillon test a été de **92%**.

Nous avons pour la partie demo streamlit choisi ce modèle en plus de cette performance sa propension à contrôler le surapprentissage (pas totalement exclus comme vu précédemment).

- Objectif

Optimiser les paramètres du modèle ET (ExtraTreesClassifier) avec GridSearch pour améliorer ses performances.

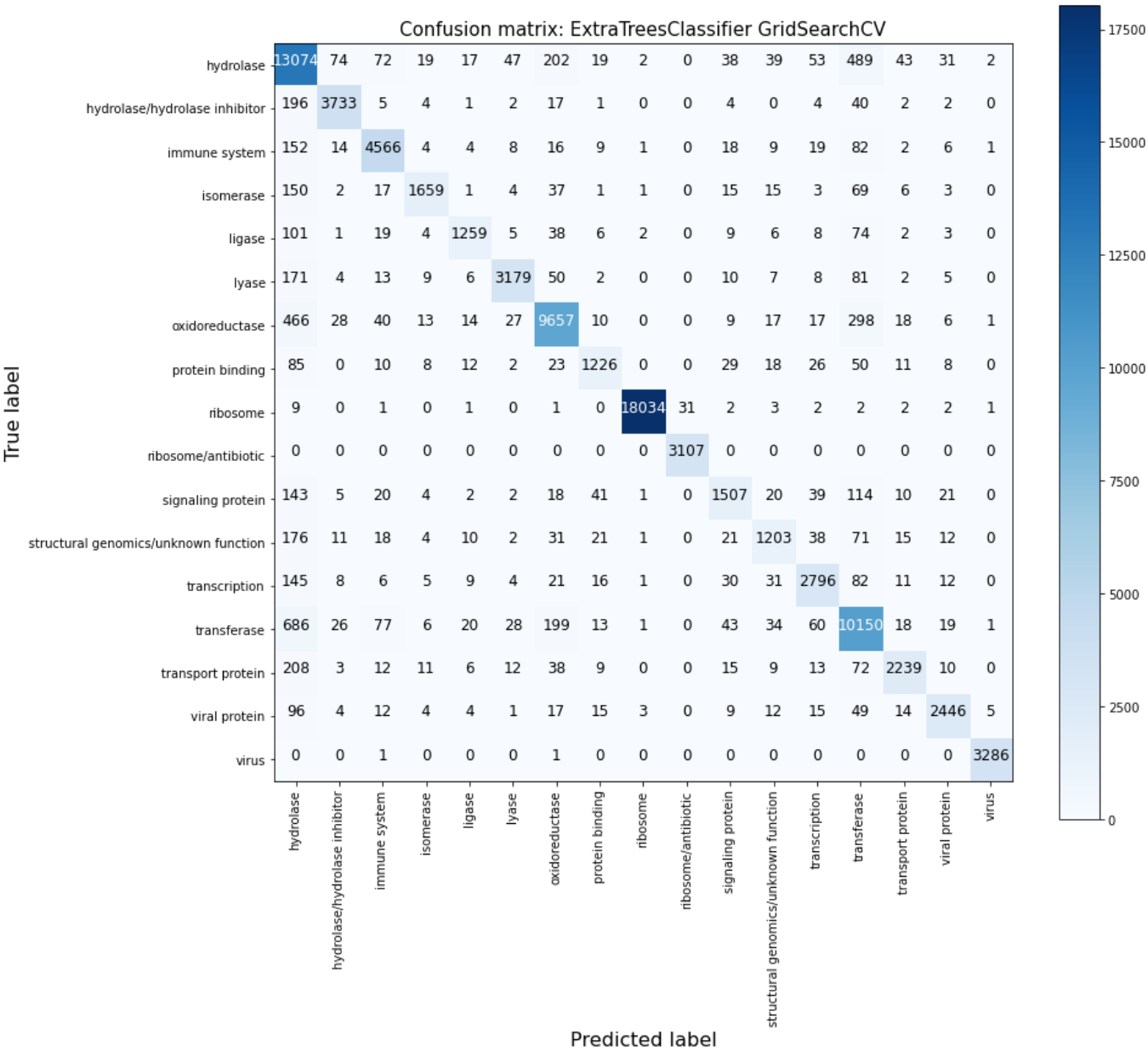
- Résultats:

L'optimisation des paramètres n'améliore pas d'avantage le modèle qui était déjà performant, tout au plus nous constatons pour la classe ribosome une meilleure prédiction.

Fitting 5 folds for each of 36 candidates, totalling 180 fits

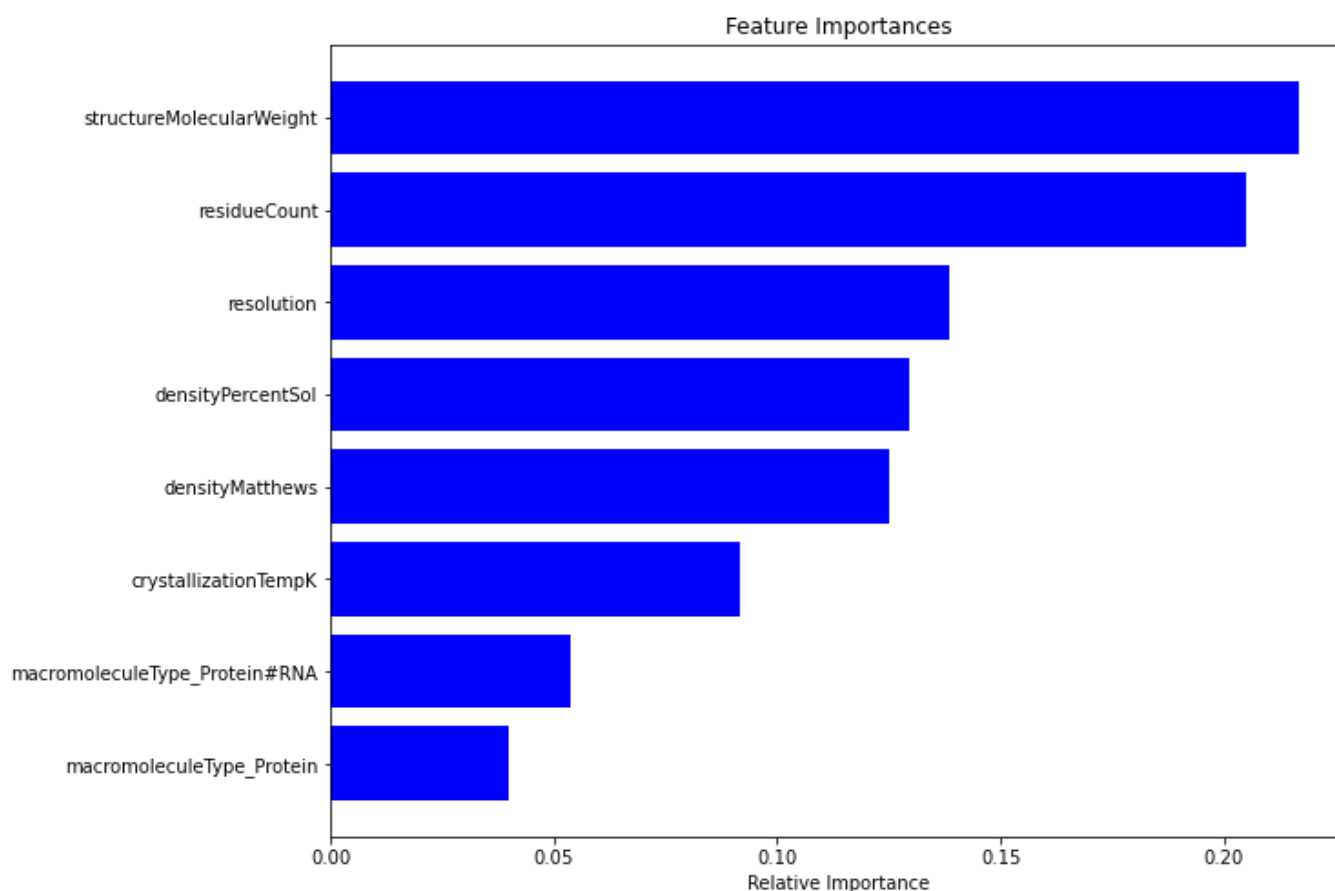
```
GridSearchCV(estimator=ExtraTreesClassifier(), n_jobs=-1,
              param_grid={'criterion': ['gini', 'entropy'],
                           'max_depth': [10, 15, 20, 30, 40, 50],
                           'min_samples_leaf': [1, 2, 5]},
              verbose=3)
```

	precision	recall	f1-score	support
hydrolase	0.82	0.92	0.87	14221
hydrolase/hydrolase inhibitor	0.95	0.93	0.94	4011
immune system	0.93	0.93	0.93	4911
isomerase	0.95	0.84	0.89	1983
ligase	0.92	0.82	0.87	1537
lyase	0.96	0.90	0.93	3547
oxidoreductase	0.93	0.91	0.92	10621
protein binding	0.88	0.81	0.85	1508
ribosome	1.00	1.00	1.00	18091
ribosome/antibiotic	0.99	1.00	1.00	3107
signaling protein	0.86	0.77	0.81	1947
structural genomics/unknown function	0.85	0.74	0.79	1634
transcription	0.90	0.88	0.89	3177
transferase	0.87	0.89	0.88	11381
transport protein	0.93	0.84	0.89	2657
viral protein	0.95	0.90	0.92	2706
virus	1.00	1.00	1.00	3288
accuracy			0.92	90327
macro avg	0.92	0.89	0.90	90327
weighted avg	0.92	0.92	0.92	90327



- Interpretabilité du Modèle

L'algorithme ET s'appuie sur certaines variables importantes pour prédire les classes de protéines. Ici le poids moléculaire a toute son importance dans la prise de décision de l'algorithme, il en est de même du residuecount ou encore de la résolution.



Nous n'avons pas pu utiliser le package shap pour une interprétation fine du modèle ExtraTrees, les shap_values n'ayant pu être extraites de la fonction explain du module, les temps de calcul extrêmement allongés (plus de 24h). Est-ce dû à la volumétrie des données? Les packages eli5 et lime nous ont permis d'avoir une interprétation locale de l'algorithme.

- Eli5:

Les coefficients associés à chaque variable sont de même ordre que les features importances vus précédemment.

Weight	Feature
0.2167 ± 0.0480	structureMolecularWeight
0.2049 ± 0.0474	residueCount
0.1386 ± 0.0218	resolution
0.1296 ± 0.0159	densityPercentSol
0.1252 ± 0.0197	densityMatthews
0.0916 ± 0.0126	crystallizationTempK
0.0538 ± 0.0916	macromoleculeType_Protein#RNA
0.0397 ± 0.0864	macromoleculeType_Protein

- Lime:

Ici avec le package la contribution de chaque variable dans la prédiction de "l'individu 1"

Prediction probabilities

virus	0.00
Other	0.00
	0.00
	0.00
	0.00

NOT undefined

```

structureMolecularWe... 0.01
macromoleculeType_... 0.00
-0.06 < resolution <=... 0.00
densityMatthews > 0.64 0.00
residueCount > 0.76    0.00

```

Feature Value


structureMolecularWeight	1.64
macromoleculeType_Protein#RNA	2.28
resolution	0.18
densityMatthews	0.71
residueCount	1.51

B. Deep Learning

Objectif

Obtenir la classification uniquement en se basant sur la séquence d'acides aminés. Le problème est ramené à un cas de Natural Language Processing (NLP), **Sequence-to-sequence**. Nous aborderons deux modèles de deep learning: un modèle convolutionnel à une dimension et en dernier le modèle d'apprentissage profond LSTM.

Une séquence est constitué d'une suite de lettres représentant des acides aminés, la longueur des séquences à notre disposition va de 1 à un maximum de 5070 caractères

sequence_example

1. Preprocessing

En observant la distribution des séquences suivant le nombre de caractères, on observe que le dataframe contient essentiellement des séquences de tailles inférieure 1000. L'analyse peut être restreinte à ces séquences, pour minimiser le temps de calcul et aussi les tailles des séquences lors du padding.

- Le problème est ramené à une analyse sequence-to-sequence, avec une phase classique de tokenization des séquences, padding sur la taille maximale choisie (1000) puis application de nos modèles.
- La variable classification sera aussi encodée (LabelBinarizer) pour pouvoir la soumettre à nos modèles. Noter que les restrictions appliquées à la variable classification (nb d'échantillons > 5000) sont également appliquées dans cette partie

Du jeu de données nettoyé précédemment on n'a conservé que les variables **sequence** et **target**.

2. Choix des modèles

Nous avons testé deux modèles :

- CNN1D : Convolution 1D**, dans le but de considérer le problème comme une prédiction de données de séquence (image 1D)
- LSTM : L'ordre des acides aminés étant crucial dans la détermination de la classification, une approche avec un algorithme LSTM permet de traiter le problème comme une prédiction basée notamment sur les états précédents(acides aminés dans la séquence)**

2. Convolutional Neural Network (CNN1D)

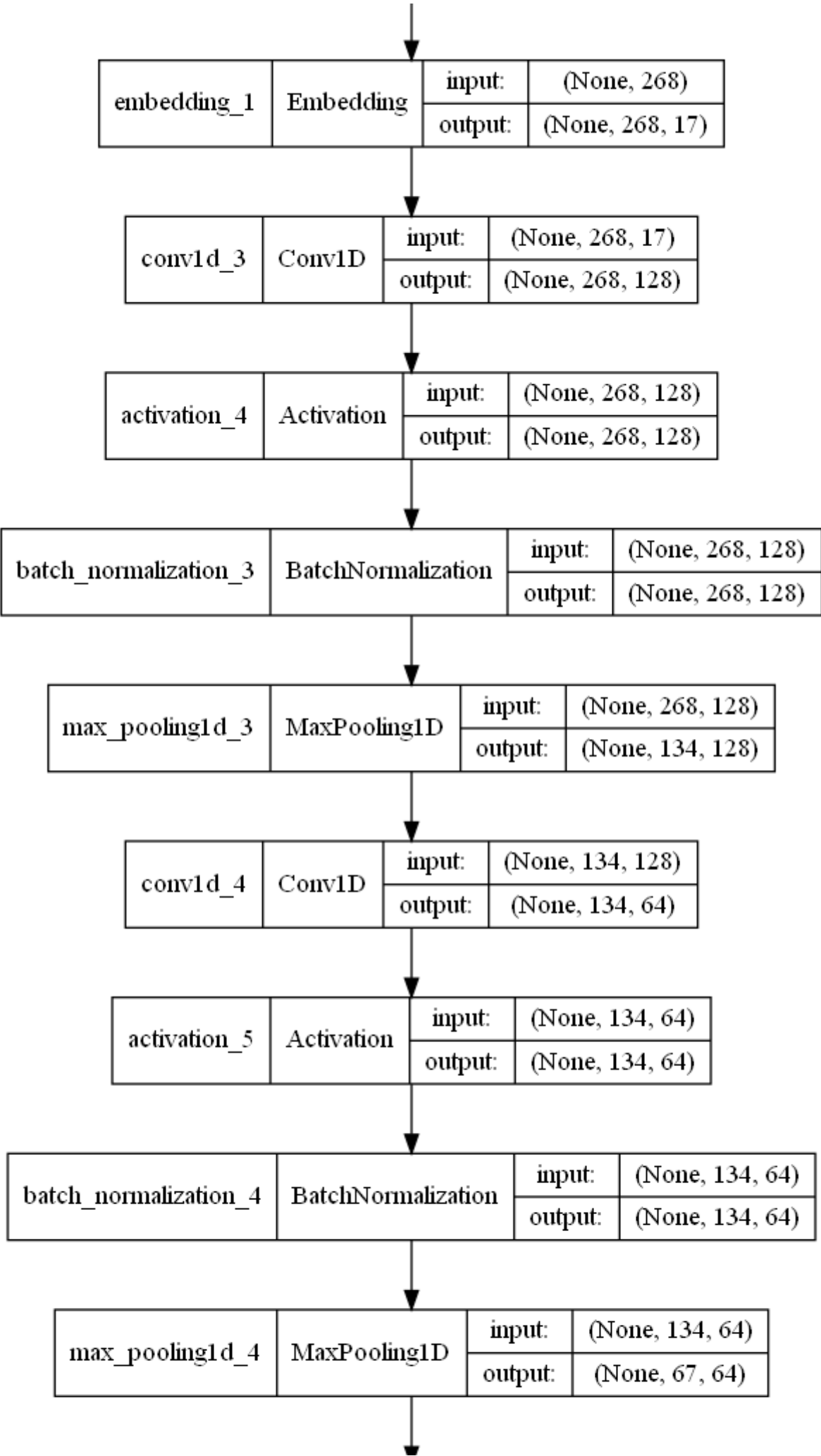
Les réseaux de neurones convolutionnels bien que souvent appliqués en imagerie pour la classification, peuvent aussi être utilisés dans la classification des séquences. Ici la séquence d'entrée est utilisée comme une image 1D.

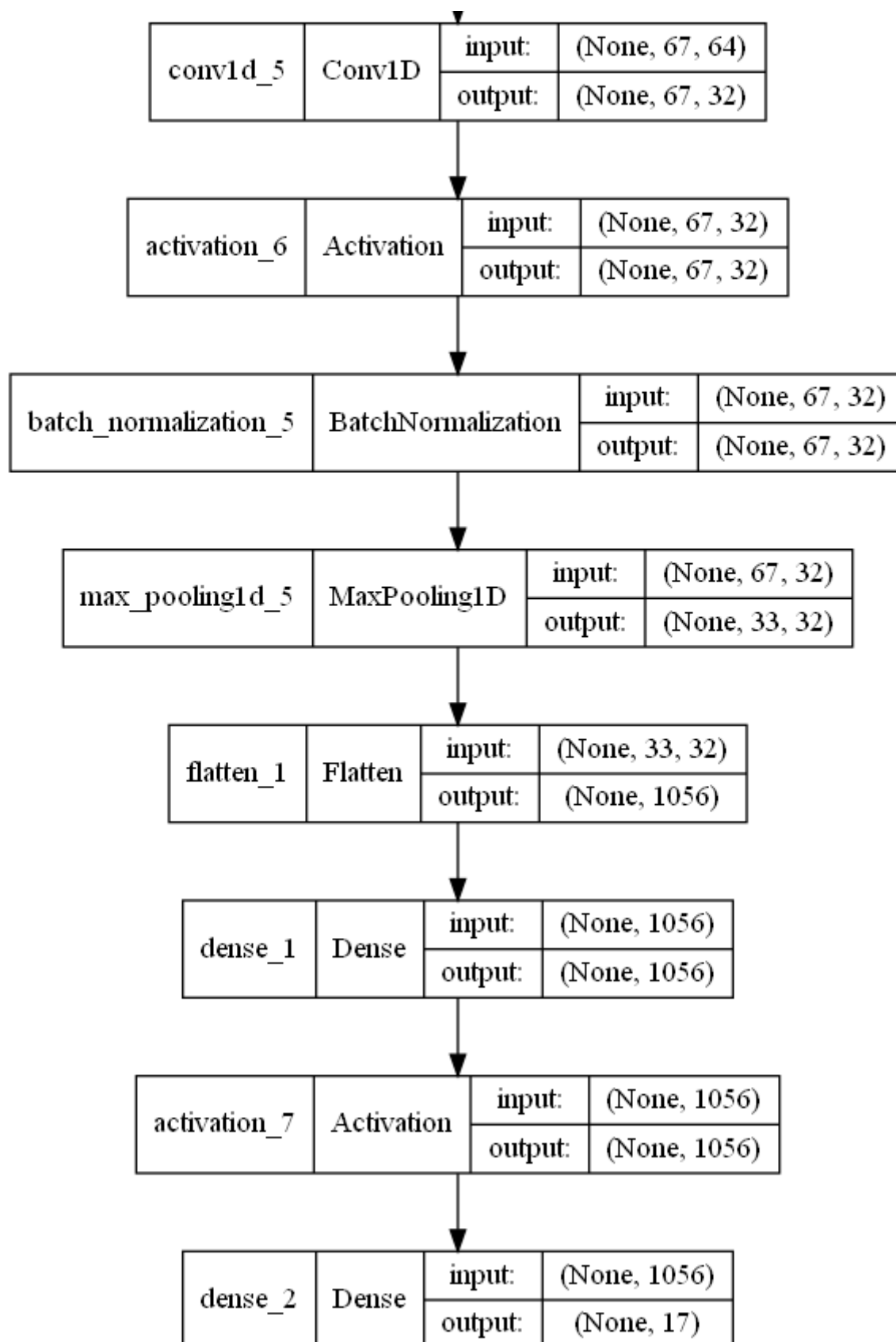
Comme précédemment les métriques utilisées sont les mêmes (accuracy, classification report, matrice de confusion). Nous avons construit le modèle convolutionnel de façon séquentielle avec des couches denses de batchnormalization, de Maxpooling1D et des couches denses full connected.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 268, 17)	442
conv1d_3 (Conv1D)	(None, 268, 128)	26240
activation_4 (Activation)	(None, 268, 128)	0
batch_normalization_3 (Batch Normalization)	(None, 268, 128)	512
max_pooling1d_3 (MaxPooling1D)	(None, 134, 128)	0
conv1d_4 (Conv1D)	(None, 134, 64)	49216
activation_5 (Activation)	(None, 134, 64)	0
batch_normalization_4 (Batch Normalization)	(None, 134, 64)	256
max_pooling1d_4 (MaxPooling1D)	(None, 67, 64)	0
conv1d_5 (Conv1D)	(None, 67, 32)	6176
activation_6 (Activation)	(None, 67, 32)	0
batch_normalization_5 (Batch Normalization)	(None, 67, 32)	128
max_pooling1d_5 (MaxPooling1D)	(None, 33, 32)	0
flatten_1 (Flatten)	(None, 1056)	0
dense_1 (Dense)	(None, 1056)	1116192
activation_7 (Activation)	(None, 1056)	0
dense_2 (Dense)	(None, 17)	17969

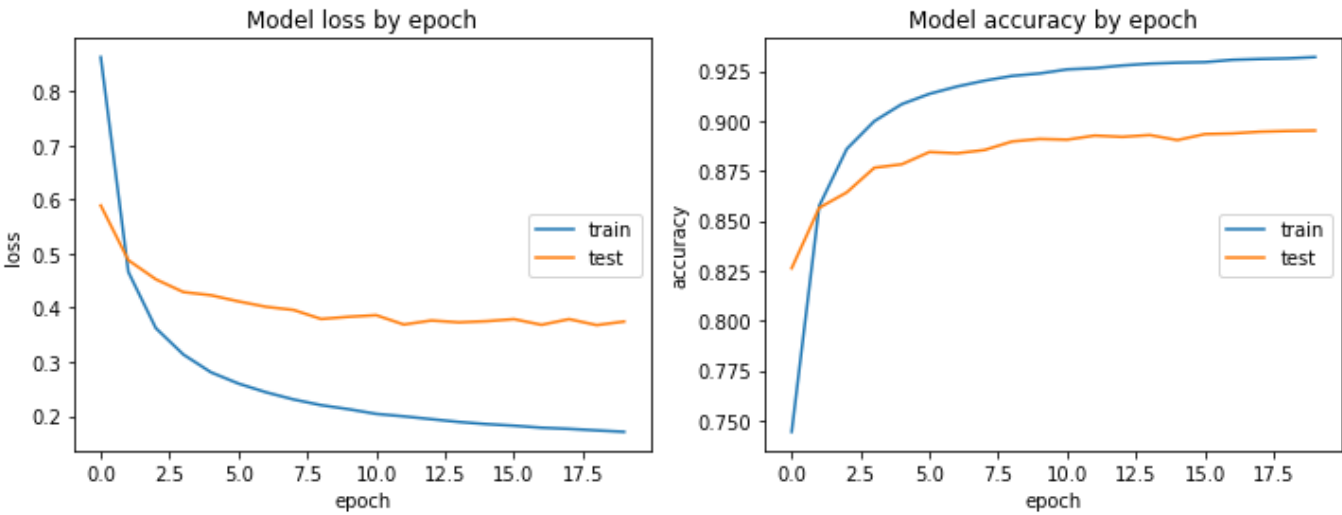
embedding_1_input	InputLayer	input:	[(None, 268)]
		output:	[(None, 268)]





3. Tests des modèles:

• 3.1 Modèle 1: CNN1D



• train accuracy

```
7528/7528 [=====] - 276s 35ms/step - loss: 0.1627 - accuracy: 0.9347
[0.16269294917583466, 0.934674859046936]
```

• test accuracy

```
1882/1882 [=====] - 80s 35ms/step - loss: 0.3743 - accuracy: 0.8952
: [0.37432369589805603, 0.8951642513275146]
```

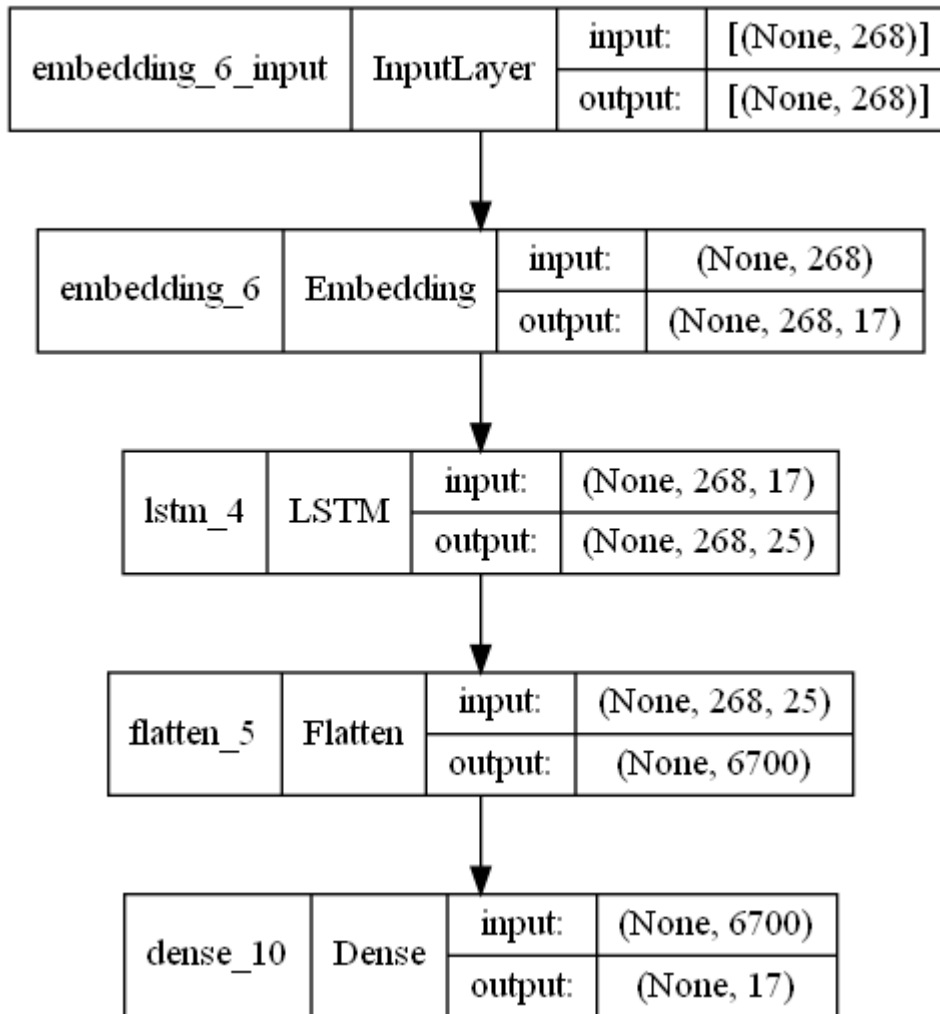
L'accuracy sur les données d'apprentissage et test sont proches et globalement le réseau convolutionnel reste moins efficace que le model ExtraTrees mais à priori il n'y a pas d'overfitting, ce qui est non négligeable dans la généralisation du modèle.

• 3.2 Modèle 2: LSTM

Model: "sequential_6"

Layer (type)	Output Shape	Param #
embedding_6 (Embedding)	(None, 268, 17)	442
lstm_4 (LSTM)	(None, 268, 25)	4300
flatten_5 (Flatten)	(None, 6700)	0
dense_10 (Dense)	(None, 17)	113917

```
=====
Total params: 118,659
Trainable params: 118,659
Non-trainable params: 0
```



IV. Bilan et Perspectives

Ce jeu de données extrait de la base de données du Protein Data Bank a été une opportunité pour nous de faire l'application pratique des connaissances acquises en cours de formation.

En effet, nous savions, en début de formation que la donnée (tabulaire, images, text et sons) est l'outil de travail par excellence du data scientist, l'appivoiser et en extraire des connaissances ne nous étaient pas évidents.

La nature sensible des données, attachés à un secteur d'activité qui ne nous était pas forcément familier, nous a poussé à explorer en profondeur chaque variable et comprendre de manière globale son intérêt pour le sujet suivi. Nous avons donc testé plusieurs configuration de preprocessing (regroupement des données, filtrages des données, suppression des valeurs nulles, remplacement des valeurs nulles, analyses de certaines variables en détails...)

Ce projet nous a permis d'explorer la plupart des modèles de classification en machine learning et en Deep Learning et ce dernier a montré une bonne efficacité en classification.

Pour y arriver nous avons suivi différentes étapes:

- exploration du dataset final après fusion: dans cette étape, nous avons cherché à comprendre notre sujet d'étude mais aussi la signification de toutes les variables du dataset, nous avons cherché les

corrélations entre les différentes variables et avons exploré les liens entre la variable target (nous avons gardé toutes les modalités de cette variable) et les autres variables.

- L'étape d'exploration a été complétée par:
 - la dataviz,
 - la data transformation,
 - le feature engeneering,
 - la modélisation des données,
 - la recherche des modèles optimaux de classification,
 - choix des modèles , entraînement, recherche des hyperparamètres optimaux...
- **Pour la partie ML, nous avons choisi toutes les variables à l'exception de la variable séquence et en DL nous avons choisi la variable sequence et classification.**
- Une autre découverte au cours de l'exécution de ce projet : les possibilités offertes par le package Lazypredict qui en amont de l'entraînement nous a permis de choisir les modèles appropriés avec les accuracy attendus.

Les difficultés n'ont pas manqué dans la réalisation de ce projet:

- la non maîtrise du secteur abordé, causant quelques difficultés sur les choix à opérer notamment lors du preprocessing, et dans la détermination des classes à analyser.
- les temps de calculs longs avec le choix assumé de garder un maximum des données
- les difficultés techniques liées notamment au manque d'expérience en feature engeneering

En dépit de ces difficultés et malgré le fait qu'on a travaillé à deux, nous avons enregistré des résultats intéressants.

En définitive, l'écart entre les accuracy d'entraînement et de test incite à plus de prudence car nous sommes sans doute dans un phénomène d'overfitting quoi que modeste pour le ML, en deep learning cet écart a été légèrement réduit.