

PYTHON FOR DATA ANALYSIS

IBO A5 2020/2021

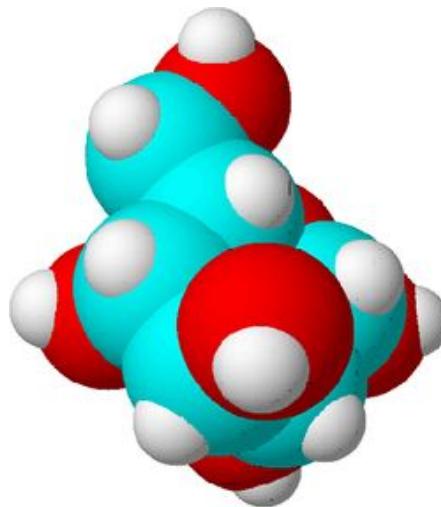
- Galbez Flavien
- Galasso Vincent

SOMMAIRE

- I. Compréhension du dataset
- II. Objectif visé
- III. Méthodes appliqués
 - A. Import
 - B. Nettoyage des données
 - C. Datavisualisation
 - D. Modélisation
 - 1. Régression Logistique
 - 2. kNN
 - 3. Random Forest
 - 4. Support Vector Machine (SVM)
 - 5. Neural Networks rectified linear unit
 - 6. Neural Networks logistic sigmoid
- IV. Résultats obtenus
- V. API

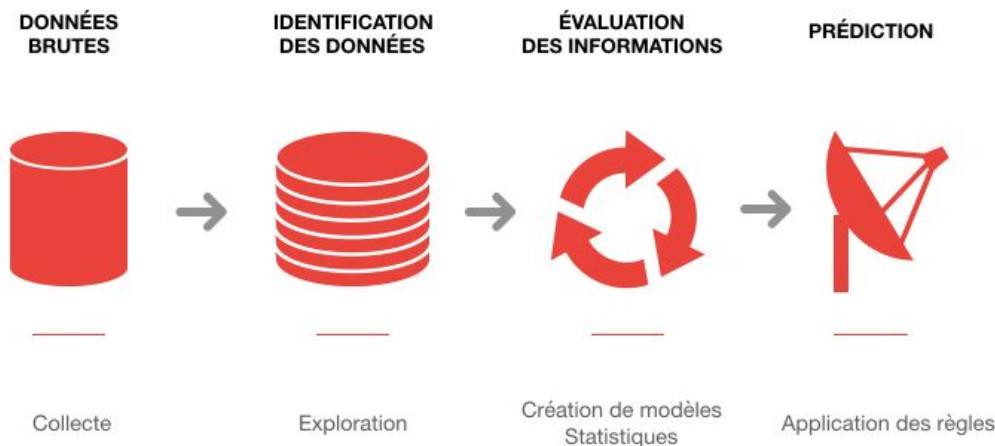
I. Compréhension du dataset

- Notre dataset fait 1055 lignes et 42 colonnes.
- Lignes : Produits chimiques et colonnes : Caractéristiques (type d'atomes...).
- Les produits chimiques sont catégorisés en 2 classes : ready and not ready biodegradable
- Plus de 60 % de non-biodégradables (Dataset non équilibré (qui pourra induire en erreur nos modèles))



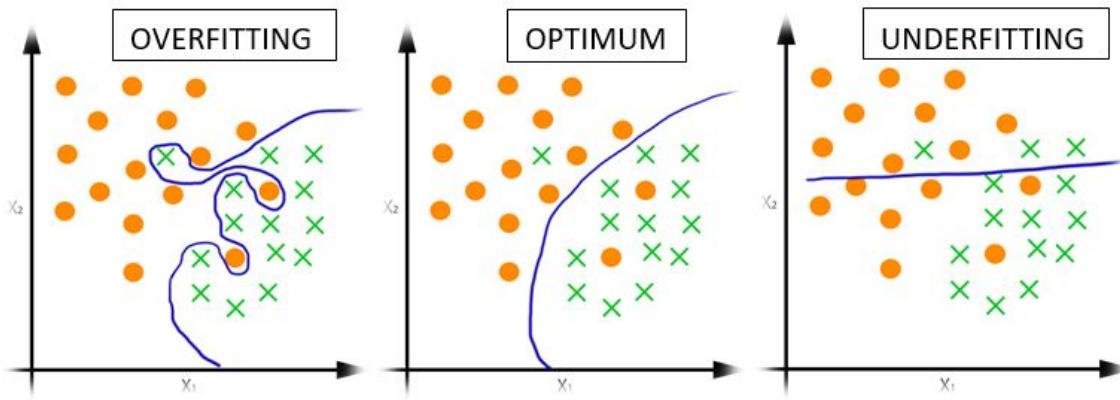
II. Objectif visé

- Objectif principal -> Prédire si un produit chimique est “ready or not ready biodégradable”
- Pour cela on va devoir créer/entraîner un “modèle”
- Le modèle doit être de bonne qualité afin d'être le plus fiable possible



III. Méthodes appliqués

- Dans un premier temps on regarde le dataset en général
 - *Colonnes et lignes (nom, valeurs, etc ...)*
- Nettoyage des données si besoin.
- Potentiellement des variables corrélées dans nos 40 colonnes
- Séparation du dataset en Train et Test (80% Train et 20% Test pour être Optimum)
- Création de différents modèles / Recherche des meilleurs hyperparamètres pour chaque modèle.



III.A Import

- Importation des données grâce à Pandas
- Ajout sur les différentes colonnes de leur nom (ajout fait grâce à la création d'un fichier csv obtenu grâce au descriptif du dataset du site (cf fichier .ipynb), on joint aussi leur description car le titre ne sont pas forcément très compréhensibles seul)

titre	SpMax_L	J_Dz(e)	nHM	F01[N-N]	F04[C-N]	NssssC	nCb-	C%	nCp	nO	F03[C-N]	SdssC	HyWi_B(m)	LOC	SM6_L	F03[C-O]	Me	Mi	nN-N	nArNO2	nCRX3
0	3.919	2.6909	0	0	0	0	0	31.4	2	0	0	0.000	3.106	2.550	9.002	0	0.960	1.142	0	0	0
1	4.170	2.1144	0	0	0	0	0	30.8	1	1	0	0.000	2.461	1.393	8.723	1	0.989	1.144	0	0	0
2	3.932	3.2512	0	0	0	0	0	26.7	2	4	0	0.000	3.279	2.585	9.110	0	1.009	1.152	0	0	0
3	3.000	2.7098	0	0	0	0	0	20.0	0	2	0	0.000	2.100	0.918	6.594	0	1.108	1.167	0	0	0
4	4.236	3.3944	0	0	0	0	0	29.4	2	4	0	-0.271	3.449	2.753	9.528	2	1.004	1.147	0	0	0

III.B Nettoyage des données

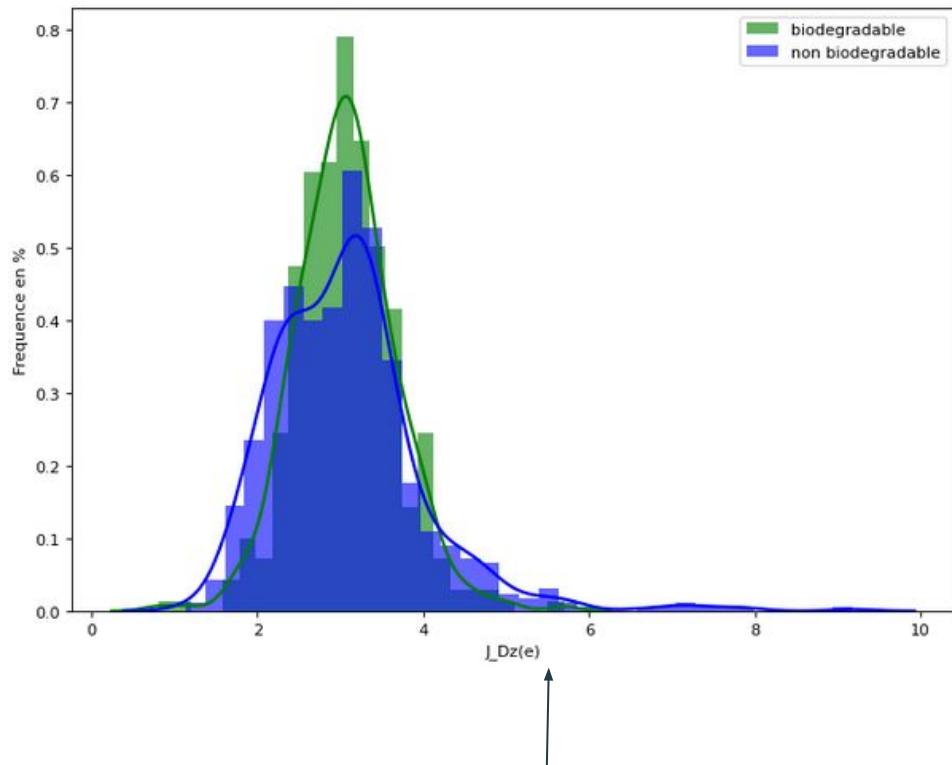
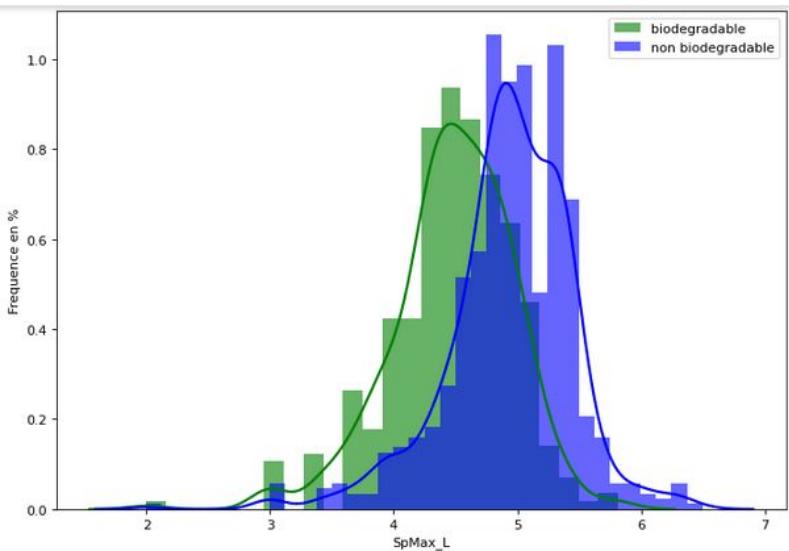
- Suppression des lignes avec des données NULL (aucune ligne à supprimer)
- Vérifications du format des données (int, float)
- Modification du format de la colonne “experimental class”:
object : NRB & RB -> int : 0 & 1 pour la représenter en booléen
(joint à un renommage de la colonne en “biodegradable”)

nArCOOR	nX	biodegradable
0	0	1
0	0	1
0	0	1
0	0	1
0	0	1

titre		
SpMax_L	float64	
J_Dz(e)	float64	
nHM	int64	
F01[N-N]	int64	
F04[C-N]	int64	
NssssC	int64	
nCb-	int64	
C%	float64	
nCp	int64	
nO	int64	
F03[C-N]	int64	
SdsSC	float64	
HyWi_B(m)	float64	
LOC	float64	
SM6_L	float64	
F03[C-O]	int64	
Me	float64	
Mi	float64	
nN-N	int64	
nArNO2	int64	
nCRX3	int64	
SpPosA_B(p)	float64	
nCIR	int64	
B01[C-Br]	int64	
B03[C-C1]	int64	
N-073	int64	
SpMax_A	float64	
Psi_i_1d	float64	
B04[C-Br]	int64	
SdO	float64	
experimental class	object	
	dtype: object	

III.C Datavisualisation

- Observer quels paramètres sont les plus significatifs pour différencier Non-Biodégradable et Biodégradable



Paramètre très significatif

Paramètre peu significatif

III.C Datavisualisation

On remarque grâce aux différents histogrammes que les variables différenciant le plus biodégradable et non biodégradable sont :

- SpMax_L
- C%
- HyWi_B(m)
- SM6_L
- SpPosA_B(p)
- SpMax_A
- SpMax_B(m)
- SM6_B(m)

	titre	description
0	SpMax_L	Leading eigenvalue from Laplace matrix
7	C%	Percentage of C atoms
12	HyWi_B(m)	Hyper-Wiener-like index (log function) from Burden matrix
14	SM6_L	Spectral moment of order 6 from Laplace matrix
21	SpPosA_B(p)	Normalized spectral positive sum from Burden matrix
26	SpMax_A	Leading eigenvalue from adjacency matrix (Lovasz number)
35	SpMax_B(m)	Leading eigenvalue from Burden matrix weighted by atomic numbers
38	SM6_B(m)	Spectral moment of order 6 from Burden matrix weighted by atomic numbers

III.C Datavisualisation

- Recherche des variables corrélés (>0.90)
- Les valeurs corrélés peuvent fausser les résultats des modèles.

titre	SpMax_L	C%	HyWi_B(m)	SM6_L	SpPosA_B(p)	SpMax_A	SpMax_B(m)	SM6_B(m)
titre								
SpMax_L	1.000000	0.381464	0.642858	0.911546	0.230438	0.918928	0.305638	0.489765
C%	0.381464	1.000000	0.352416	0.429790	0.623108	0.568842	0.149713	0.280505
HyWi_B(m)	0.642858	0.352416	1.000000	0.805254	0.421537	0.673059	0.565551	0.821144
SM6_L	0.911546	0.429790	0.805254	1.000000	0.231165	0.922649	0.252469	0.514928
SpPosA_B(p)	0.230438	0.623108	0.421537	0.231165	1.000000	0.311172	0.527758	0.590619
SpMax_A	0.918928	0.568842	0.673059	0.922649	0.311172	1.000000	0.259384	0.471681
SpMax_B(m)	0.305638	0.149713	0.565551	0.252469	0.527758	0.259384	1.000000	0.917443
SM6_B(m)	0.489765	0.280505	0.821144	0.514928	0.590619	0.471681	0.917443	1.000000

III.C Datavisualisation

- On retire les ne prend en compte seulement un parametres de chaque binôme de paramètres corrélés
- Ainsi les variables suivantes ne seront pas prise en compte dans nos modèles :

```
parameter_correlated=["SM6_L", "SpMax_A", "SM6_B(m)", "F04[C-N]"]
```

III.D Modélisation

Données retournées de chaque modèle pour comparaison (Les modèles sont testés avec différents hyper paramètres afin d'obtenir le meilleur F1_score) :

- Accuracy
- Précision
- Sensitivity
- F1 score
- Matrice de confusion

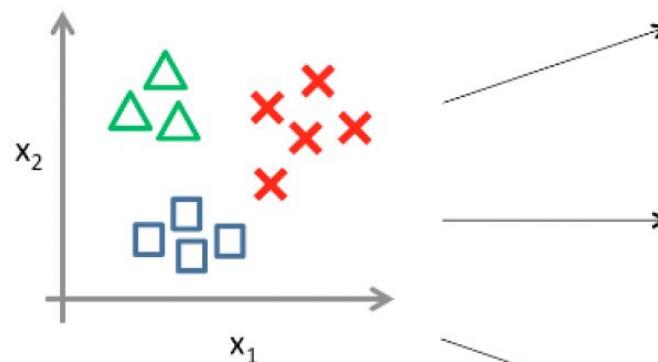
		Classe réelle	
		-	+
Classe prédictive	-	True Negatives <i>(vrais négatifs)</i>	False Negatives <i>(faux négatifs)</i>
	+	False Positives <i>(faux positifs)</i>	True Positives <i>(vrais positifs)</i>

III.D Modélisation

1.Régression Logistique

La régression logistique ou modèle logit est un modèle de régression binomiale. Comme pour tous les modèles de régression binomiale, il s'agit de modéliser au mieux un modèle mathématique simple à des observations réelles nombreuses.

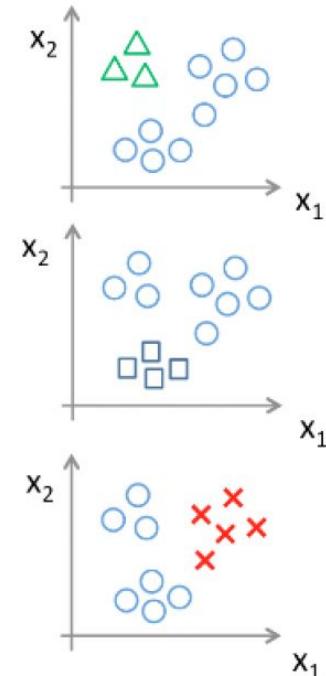
One-vs-all (one-vs-rest):



Class 1:

Class 2:

Class 3:



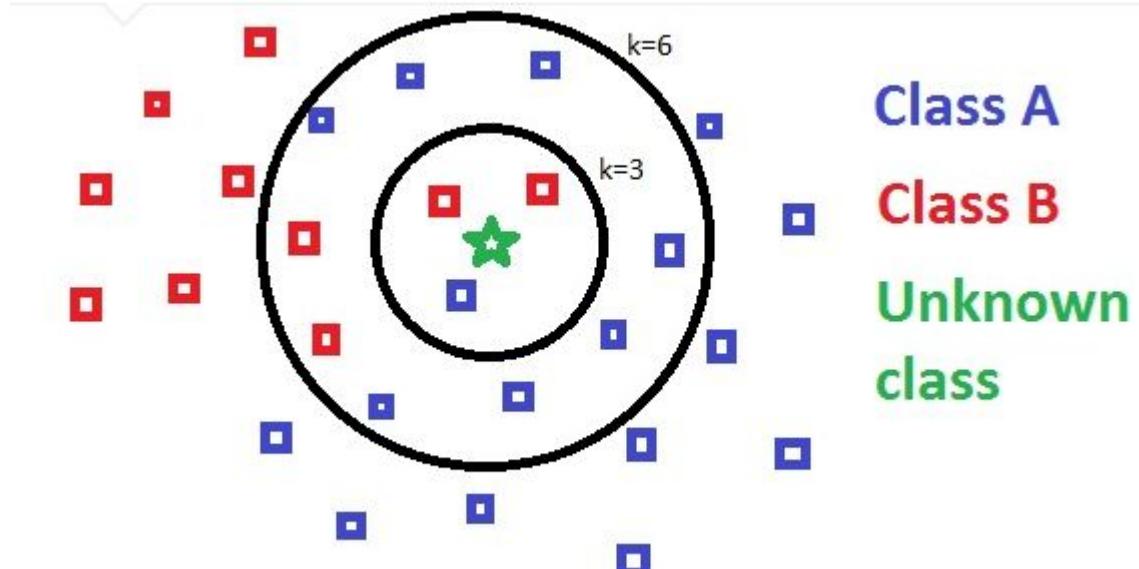
III.D Modélisation

2.kNN

La méthode des k plus proches voisins est une méthode d'apprentissage supervisé.

Meilleur hyperparamètre :
k=6 (nombre de voisins explorés)

Le choix de cet hyperparamètre s'est fait en faisant une boucle sur k et en gardant le k fournissant le meilleur modèle selon le F1_score



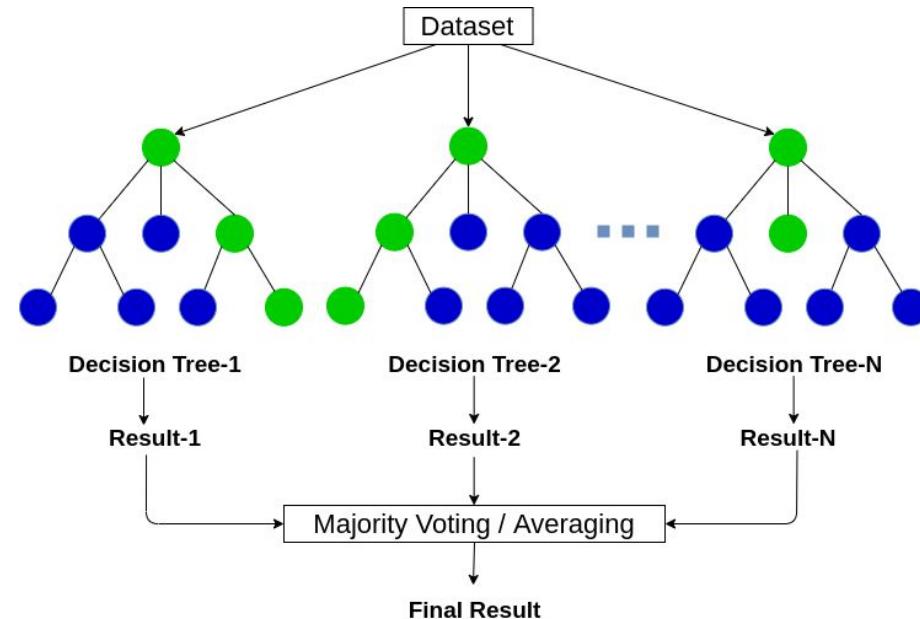
III.D Modélisation

3.Random Forest

Meilleur hyperparamètre :
 $n=61$ (nombre d'arbre)

Le choix de cet hyperparamètre s'est fait en faisant une boucle sur n et en gardant le n fournissant le meilleur modèle selon le `F1_score`

Les forêts d'arbres décisionnels (ou forêts aléatoires de l'anglais random forest classifier). L'algorithme des forêts d'arbres décisionnels effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents.



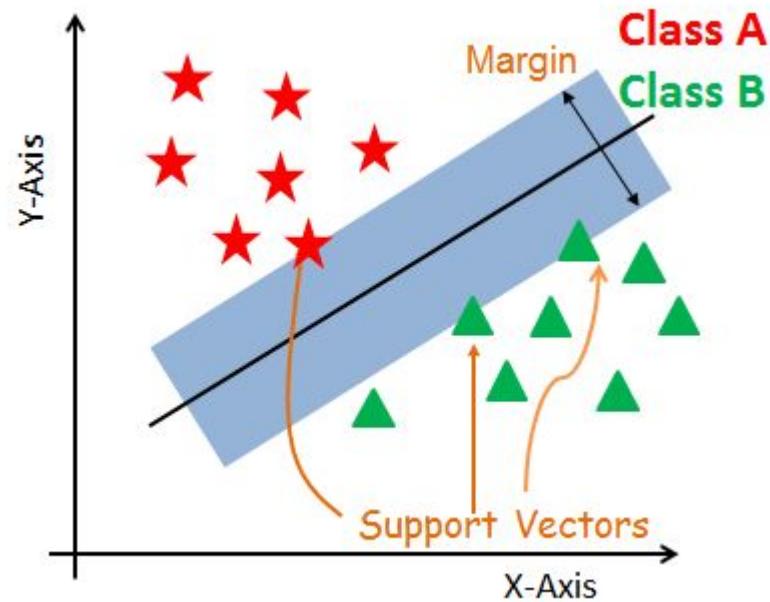
III.D Modélisation

4. Support Vector Machine (SVM)

Les machines à vecteurs de support ou séparateurs à vaste marge (en anglais support vector machine, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination et de régression. Les SVM sont une généralisation des classificateurs linéaires.

Meilleur hyperparamètre : $C=0.9$

Le choix de cet hyperparamètre s'est fait en observant le modèle pour différentes valeur de C , $C=0.9$ était la valeur qui fournissait le meilleur modèle



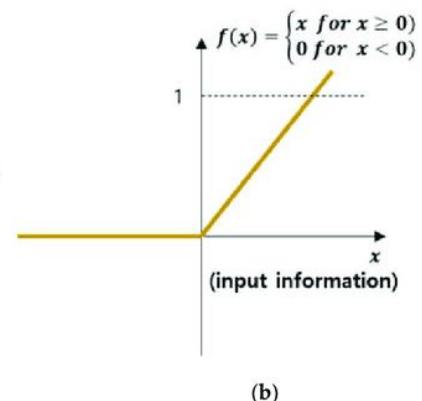
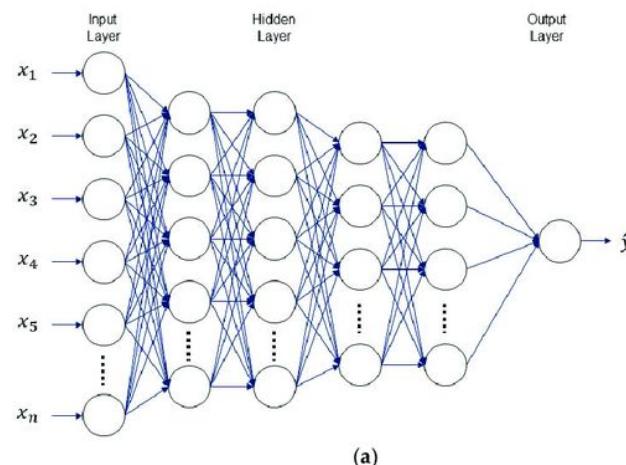
III.D Modélisation

5. Neural Networks rectified linear unit (ReLU)

La méthode Rectified Linear Unit est la fonction d'activation la plus utilisée dans les modèles d'apprentissage profond. La fonction renvoie 0 si elle reçoit une entrée négative, mais pour toute valeur positive x , elle renvoie cette valeur. Elle peut donc être écrite sous la forme $f(x)=\max(0,x)$.

Meilleur hyperparamètre : $h1=(85,85)$

Le choix de cet hyperparamètre s'est fait en faisant une boucle sur i avec $h1=(i,i)$ et en gardant le i fournissant le meilleur modèle selon le F1_score



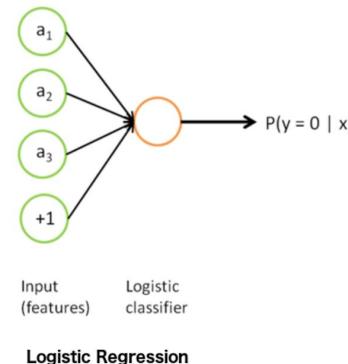
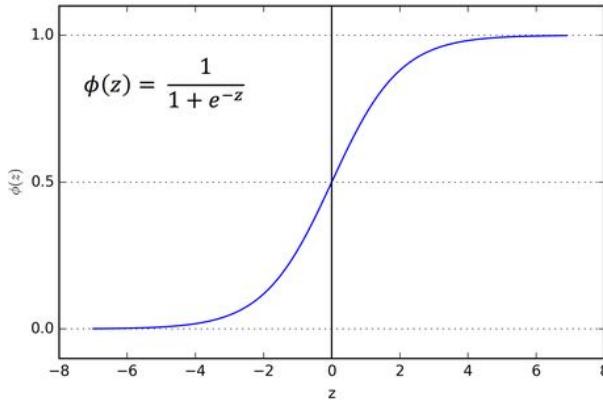
III.D Modélisation

6. neural networks logistic sigmoid

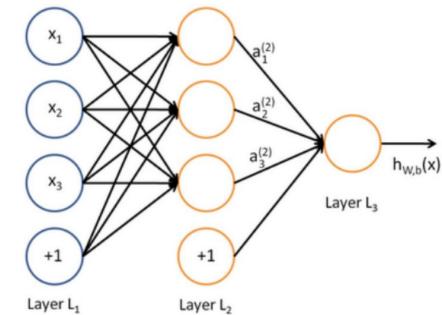
La fonction sigmoïde, contrairement à la fonction pas à pas, introduit la non-linéarité dans notre modèle de réseau neuronal. ... Cette fonction d'activation non linéaire, lorsqu'elle est utilisée par chaque neurone dans un réseau neuronal multicouche, produit une nouvelle "représentation" des données d'origine, et permet finalement une limite de décision non linéaire, telle que le XOR.

Meilleur hyperparamètre : $h2=(128,128)$

Le choix de cet hyperparamètre s'est fait en faisant une boucle sur i avec $h2=(i,i)$ et en gardant le i fournissant le meilleur modèle selon le F1_score



Logistic Regression



Neural Network

IV. Résultats Obtenus

Neural Networks Logistic:

Matrice de confusion :
[[0.89393939 0.10606061]
 [0.17721519 0.82278481]]

Accuracy : 0.8672985781990521

Precision : 0.8227848101265823

Sensitivity : 0.8227848101265823

F1 score : 0.8227848101265823

Neural Networks Relu :

Matrice de confusion :
[[0.90151515 0.09848485]
 [0.17721519 0.82278481]]

Accuracy : 0.8720379146919431

Precision : 0.8333333333333334

Sensitivity : 0.8227848101265823

F1 score : 0.8280254777070064

SVM :

Matrice de confusion :
[[0.85606061 0.14393939]
 [0.13924051 0.86075949]]

Accuracy : 0.8578199052132701

Precision : 0.7816091954022989

Sensitivity : 0.8607594936708861

F1 score : 0.8192771084337349

Random Forest :

Matrice de confusion :
[[0.92424242 0.07575758]
 [0.24050633 0.75949367]]

Accuracy : 0.8625592417061612

Precision : 0.8571428571428571

Sensitivity : 0.759493670886076

F1 score : 0.8053691275167785

KNN :

Matrice de confusion :
[[0.88636364 0.11363636]
 [0.29113924 0.70886076]]

Accuracy : 0.8199052132701422

Precision : 0.7887323943661971

Sensitivity : 0.7088607594936709

F1 score : 0.7466666666666666

Regression Logistique :

Matrice de confusion :
[[0.87121212 0.12878788]
 [0.18987342 0.81012658]]

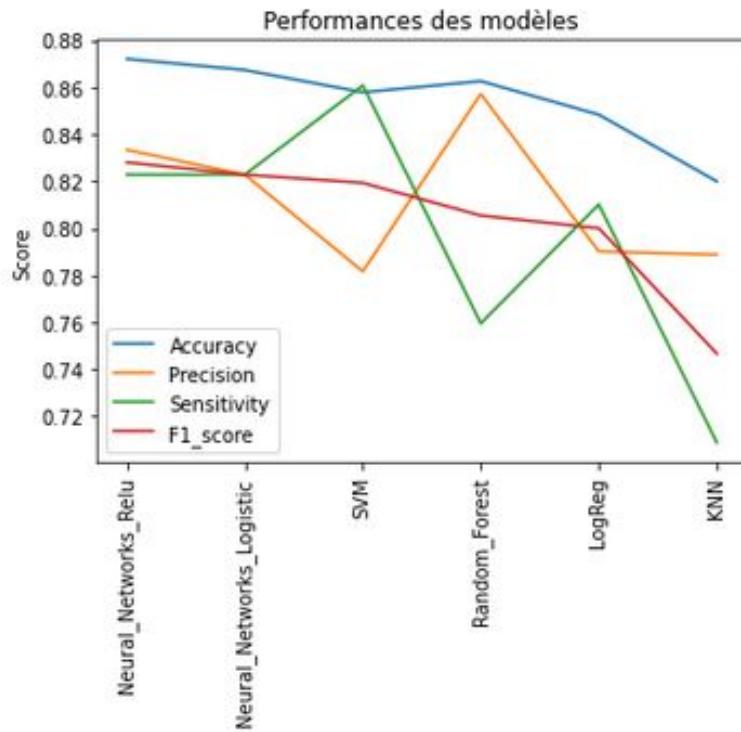
Accuracy : 0.8483412322274881

Precision : 0.7901234567901234

Sensitivity : 0.810126582278481

F1 score : 0.8

IV. Résultats Obtenus



On remarque les points suivants :

De manière générale, tous les modèles (à l'exception du kNN) nous donnent des prédictions assez bonnes dans l'ensemble mais pas non plus excelentes.

Les modèles les plus précis sont les réseaux de neurones ici, notamment la version rectified linear unit. On peut tout de même noter que le SVM se démarque par sa très bonne sensibilité (proportion de vrais positifs). De plus, la random forest se démarque par une grande exactitude et une grande précision.

Ces deux derniers modèles ne sont clairement pas intéressants car ils dépassent ou égalisent le réseau de neurones rectified linear unit sur certains points. Une combinaison de ces modèles pourrait être envisagé afin d'obtenir d'encore meilleurs résultats. Cependant le réseau de neurones rectified linear unit offre déjà de bons et plus constants résultats, il sera donc à utiliser en priorité (si aucune combinaison n'est faite).

IV. Résultats Obtenus

Conclusion de la modélisation :

Le modèle utilisé est donc le réseau de neurones rectified linear unit. La combinaison des différents modèles serait une piste d'amélioration (à faire via la librairie ensemble de sklearn)

IV. API (Flask)

Le but de notre API est que d'autre utilisateurs se servent de notre travail pour classer un produit chimique en Biodégradable ou non biodégradable.

Lorsque nous faisons la commande “python run App.py”, notre API se lance.

On peut exécuter une requête à cette dernière pour tester une molécule.

Pour cela on doit renseigner les caractéristiques de la molécule. Les valeurs doivent être misent dans un fichier similaire à “[Test_request.py](#)”. Le fichier est dans notre GitHub.

Nous avons renseigné la deuxième molécule de notre dataset dans ce fichier test (qui est indiquée comme biodégradable)



Voici l'output qu'on obtient en le faisant tourner : L'output montre que les données renseignées sont celles d'une molécule biodégradable d'après nos 6 modèles

Out : {‘Pred_Log_Reg’:1, ‘Pred_kNN’ : 1,
‘Pred_Rand_Forest’ : 1, ‘Pred_SVM’:1,
‘Pred_Neural_Networks_ReLu’: 1,
‘Pred_Neural_Networks_Log’ : 1}