ondalys
Donnez du sens à vos données

Etude scientifique

Comparaison de méthodes de Machine Learning pour l'analyse de données spectroscopiques

Les méthodes de Machine Learning (ML) sont de plus en plus populaires dans de nombreux domaines d'analyse de données, et l'analyse de données spectroscopiques n'échappe pas à la tendance !

Ceci dit, lorsque l'on s'intéresse au ML et à ses applications potentielles en chimiométrie, il peut se révéler ardu de trouver des méthodes adaptées aux problématiques que l'on souhaite traiter, en raison du grand nombre d'options possibles…

Afin de donner quelques pistes pour démarrer, Ondalys a comparé quelques méthodes parmi les plus utilisées en chimiométrie, en modélisant le taux de matière grasse d'échantillons de viande à l'aide d'un jeu de données spectroscopiques proche infrarouge. Les performances obtenues ainsi que les avantages et inconvénients des méthodes présentées sont discutés.

En fin d'article, quelques détails scientifiques sont donnés sur les méthodes de Machine Learning : Support Vector Machines (SVM), Réseaux de neurones (ANN) et Forêts aléatoires (RF).

De plus, quelques logiciels de chimiométrie et de programmation mathématique contenant ces méthodes de Machine Learning sont listés.

ONDALYS S.A.S.
S.A.S. au capital de 7 500 €
RCS Montpellier 447 624 024

Siège Social
4, rue Georges Besse
34830 CLAPIERS - France

ence Auvergne Rhône Alpes
Rond-point de l'échangeur - Les Levées
69360 SOLAIZE - France

+33 (0)4 67 67 97 87
contact@ondalys.fr
www.ondalys.fr

1 Introduction

De plus en plus de méthodes de Machine Learning sont utilisées en analyse de données et pour des problématiques de chimiométrie. Parmi les nombreuses méthodes de ML existantes, quelques-unes ont prouvé leur efficacité en chimiométrie en général, et pour le traitement de données spectroscopiques en particulier : les Support Vector Machines (SVM), les réseaux de neurones (ANN) et les arbres de régression (CART/RF). Mais une méthode est-elle plus efficace que les autres ?

L'exemple traité ici montre un jeu de données spectroscopiques avec une relation non-linéaire au paramètre à prédire ; des méthodes de chimiométrie traditionnelles sont comparées aux trois méthodes de ML listées ci-dessus.

2 Application des différentes méthodes

2.1 Echantillons et Spectres

Les différentes méthodes de ML sont appliquées à un jeu de données spectroscopiques. Les spectres (Figure 1) proviennent d'un spectromètre proche infrarouge FOSS Tecator Infratec Food and Feed Analyzer (850 à 1050 nm).[1]

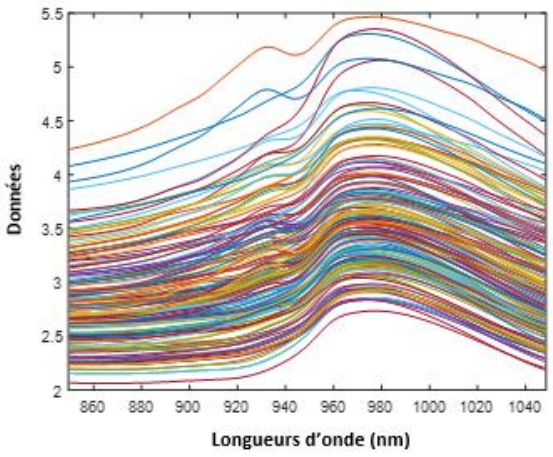

Figure 1: Spectres proche infrarouge de viande

215 échantillons de viande ont été scannés et 3 paramètres de composition ont ensuite été mesurés en référence : la teneur en matières grasses, le taux d'humidité et le taux de protéines.

On ne traitera ici que le taux de matières grasses, présentant une corrélation non linéaire avec l'absorbance proche infrarouge (Figure 2).

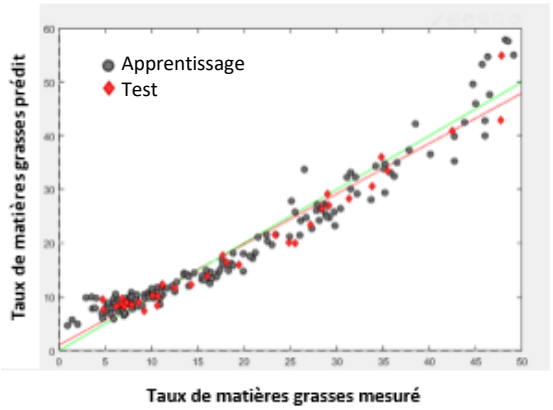

Figure 2: Non-linéarité pour la prédiction du taux de matières grasses

Les échantillons ont été séparés en deux lots :

- 172 constituent le lot d'apprentissage, qui sert à développer et optimiser le modèle ;
- 43 échantillons restants constituent un lot de test indépendant, auquel le modèle optimisé sera appliqué afin d'en évaluer les performances.

2.2 Méthodes testées

(1) Dans un premier temps, une régression linéaire de type PLS – *Partial Least Squares Regression* – a été testée. La PLS est une méthode de régression linéaire, très couramment utilisée dans de nombreux problèmes de chimiométrie. L'objectif de la méthode est de déterminer des composantes qui résument la covariance entre les variables d'entrée X (spectres) et le paramètre à prédire Y. Ces composantes sont appelées des variables latentes.

(2) La PLS se basant sur une relation linéaire, elle est peu efficace pour le traitement de données ayant une corrélation non-linéaires avec Y. Pour contourner ce problème, il est possible de transformer les données : on peut ainsi transformer les X, les Y, ou les deux. Dans un second temps, la PLS a donc été appliquée sur des données transformées.

[1] *Borggaard, Thodberg, Analytical Chemistry, 64 (1992) 545–551.Thodberg, IEEE Transactions on Neural Networks 7 (1996) 56–72.* http://lib.stat.cmu.edu/datasets/tecator

(3) Une troisième possibilité est d'utiliser une méthode de régression locale : le principe est de calculer un modèle – généralement une PLS – pour chaque échantillon en ne prenant en compte qu'un certain nombre de ses voisins, et non la population complète. On obtient ainsi des modèles « linéaires par morceaux » selon la gamme de X et de Y.

(4) Les Support Vector Machines (SVM) ont aussi été testées. Les SVM sont une méthode de ML. L'algorithme est basé sur des calculs de marges. Dans le cas d'une régression quantitative, seuls les échantillons sur et en dehors de la marge vont participer aux calculs du modèle. Ces échantillons sont appelés « vecteurs support ». Par ailleurs, les SVM utilisent des fonctions noyau, permettant entre autres de bien gérer d'éventuelles non linéarités dans les jeux de données en réalisant une transformation. La fonction noyau la plus utilisée est un noyau gaussien.

(5) Une autre méthode de ML assez répandue est celle des réseaux de neurones – ANN (*Artificial Neural Networks*). Les ANN sont basés sur le principe du « perceptron multi-couches », s'inspirant de la structure du système nerveux. L'algorithme utilise des neurones répartis sur plusieurs couches – entrée, sortie et au moins une couche cachée – et des fonctions non linéaires afin de déterminer les poids, ou coefficients, des liaisons entre neurones connectés.

(6) Enfin, les RF (*Random Forest*) sont une méthode d'ensemble, construite à partir de plusieurs arbres de régression (CART – *Classification And Regression Tree*). Le principe du CART est de réaliser des séparations binaires successives : à chaque nœud, l'algorithme sélectionne une variable et un seuil correspondant, de manière à maximiser la variance inter-classes dans le jeu de données. Dans un arbre complet, chaque feuille correspond à un échantillon. Le CART seul a tendance au sur-apprentissage, ce qui est résolu à l'aide de RF, qui regroupent de nombreux arbres.

3 Résultats pour chaque méthode

3.1 Régression linéaire PLS

La PLS donne des résultats insuffisants, avec une non linéarité visiblement non modélisée (Figure 3).

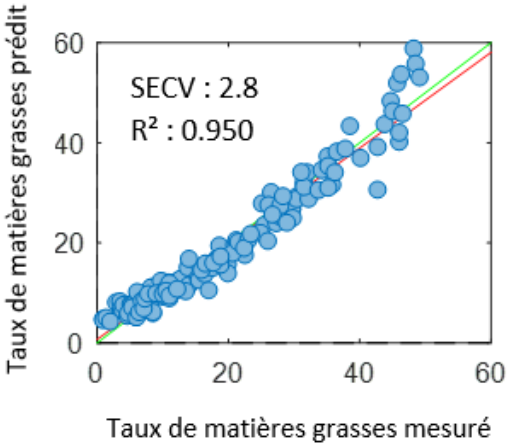

Figure 3 : Résultats de la PLS

3.2 Régression PLS sur variables transformées

La PLS a été recalculée, non plus sur les spectres, mais après transformation de variables X. Les calculs ont été réalisés sur les dix scores d'une Analyse en Composantes Principales (ACP), plus les termes au carré, les termes croisés, et avec une sélection de 14 de ces variables. Cette transformation a permis de faire disparaître la non-linéarité. Les performances sont meilleures qu'avec la PLS basée sur les spectres.

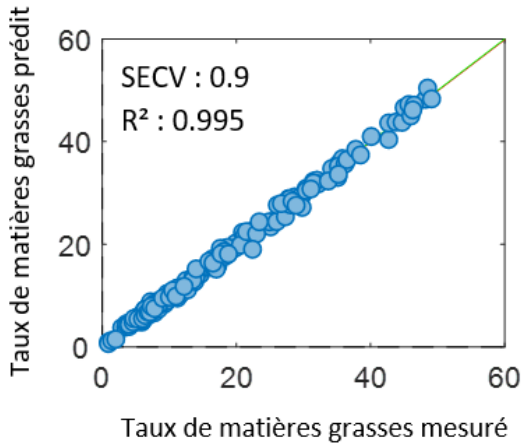

Figure 4 : Résultats de la PLS sur variables transformées

3.3 Régression locale LWR

Dans le cas de la méthode Locale LWR – *Locally Weighted Regression*, on constate également une bonne gestion de la non-linéarité. Les résultats obtenus sont similaires à ceux de la PLS basée sur les X transformés.

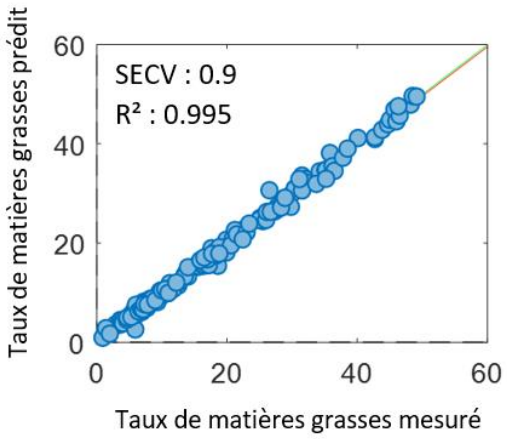

Figure 5 : Résultats de la régression locale LWR

3.4 Support Vector Machines SVR

Concernant les Support Vector Machines en Régression (SVR), on observe une amélioration des résultats, avec toujours une bonne gestion de la non-linéarité. A noter cependant, un risque de sur-apprentissage existe avec cette méthode très puissante. Il est donc important de bien optimiser les paramètres du modèle SVM.

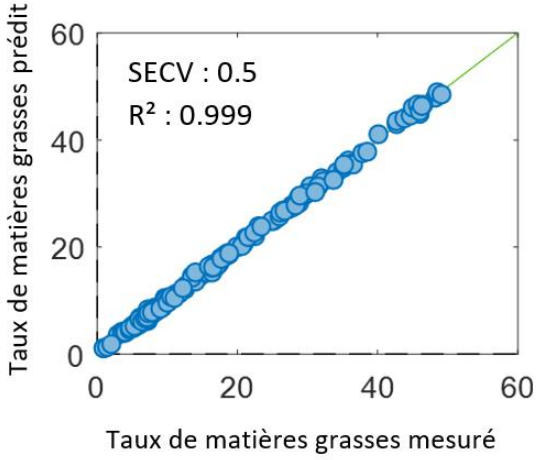

Figure 6 : Résultats des SVM

3.5 Réseaux de neurones ANN

La mise en œuvre des réseaux de neurones (ANN) a permis d'obtenir des résultats similaires aux SVM en termes de performance. Encore une fois, la non-linéarité est bien gérée par le modèle. Ici aussi, il faut noter un fort risque de sur-apprentissage lors de l'implémentation de ce type de modèle. Il est nécessaire de réduire la taille du réseau au maximum afin de limiter le sur-apprentissage. Plus un réseau est simple, plus il est robuste. Une compression des données est ainsi fortement recommandée, par exemple en utilisant les scores d'une ACP ou d'une PLS comme variables d'apprentissage et non les spectres complets.

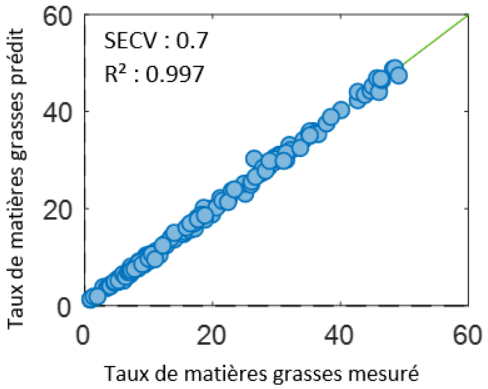

Figure 7 : Résultats des ANN

3.6 Randon Forest RF

Enfin, un modèle RF a été réalisé sur une sélection de longueur d'ondes. Comme pour les autres méthodes de ML, une bonne gestion des non-linéarités est observée. Les performances sont en revanche moins satisfaisantes qu'avec les ANN et SVM, et équivalentes à la méthode locale et PLS avec transformation de variables.

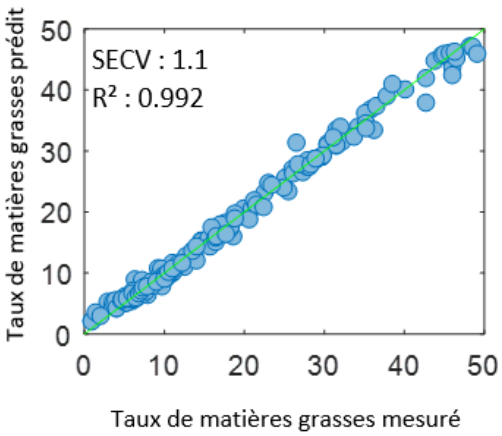

Figure 8 : Résultats des RF

4 Conclusions

Plusieurs modélisations ont été réalisées, à l'aide de méthodes bien connues en chimiométrie telles que la PLS ou la LWR, ainsi qu'avec quelques méthodes de ML qui ont prouvé leur intérêt pour des problématiques d'analyse de données spectrales.

Tableau 1 : Performances de chaque modèle

		Etalonnage				Validation croisée				Test			
Méthode	Nb LVs	SEC	SEC (%)	R²	RPD	SECV	SECV (%)	R²	RPD	SEP	SEP (%)	R²	RPD
1. PLS	5	2.6	18.5%	0.958	5	2.8	20.2%	0.950	4	2.7	19.0%	0.958	5
2. PLS sur X transformé	5	0.7	5.3%	0.997	17	0.9	6.2%	0.995	15	0.9	6.6%	0.996	14
3. LWR	3	0.5	3.6%	0.998	25	0.9	6.5%	0.995	14	1.0	6.9%	0.995	13
4. SVR	-	0.4	2.6%	0.999	35	0.5	3.6%	0.998	25	0.5	3.4%	0.999	27
5. ANN	-	0.5	3.9%	0.998	23	0.6	4.5%	0.998	20	0.4	2.7%	0.999	34
6. RF	-	0.6	4.3%	0.998	21	1.1	8.2%	0.992	11	1.0	7.5%	0.994	12

Le Tableau 1 montre que toutes les méthodes ont permis de gérer la non-linéarité à l'exception de la PLS « classique ».

Les SVM et ANN ont fourni les meilleures performances. Les résultats comparatifs donnés sur cet exemple sont généralisables à bon nombre de données spectroscopiques présentant des relations complexes (non-linéarité, clusters, faibles concentrations, paramètres non chimiques, etc.).

Bien utilisées, ces méthodes de Machine Learning sont donc d'excellentes alternatives à des méthodes classiques pour le traitement de données spectroscopiques.

Cependant, au-delà des performances, d'autres critères sont à considérer (C.f. Tableau ci-dessous) : le risque de sur-apprentissage est l'un des écueils principaux des méthodes de Machine Learning. Ce risque est largement supérieur avec les ANN par rapport aux SVM, les premiers nécessitant un nombre d'échantillons d'apprentissage bien plus important. De plus, les ANN sont nettement plus complexes à mettre en œuvre que les SVM, nécessitant de nombreux apprentissages afin de déterminer le modèle optimal.

Tableau 2 : Avantages et inconvénients

Méthode	Gestion non-linéarité	Performance	Complexité de mise en œuvre	Risques de sur-apprentissage
1. PLS	-	-	-	-
2. PLS sur X transformé	+	+	-	+
3. LWR	+	+	+	+
4. SVR	+	++	++	++
5. ANN	+	++	+++	+++
6. RF	+	+	+	+

5 Quelques détails sur les méthodes de ML

5.1 Les Support Vector Machines (SVM)

Les SVM sont des **algorithmes supervisés**, c'est-à-dire dans lesquels la référence (classe ou concentration) est connue.

Ils ont été développés initialement à des fins de classification, basé sur des calculs de « marge ». L'algorithme cherche la plus grande distance (**marge**) permettant de séparer deux classes de manière optimale. Cette séparation sera alors basée sur la sélection d'un petit nombre d'échantillons seulement, appelés « **vecteurs supports** », situés en bordure de marge. Cet algorithme se suffit donc d'un petit nombre d'échantillons d'apprentissage « bien placés ».

Par ailleurs, les calculs des SVM ne sont pas basés sur les variables initiales (absorbances en fonction de longueurs d'onde) mais sur des distances entre les échantillons 2 à 2. Ces distances sont calculées grâce à des fonctions de **noyaux** (***Kernel***), qui peuvent être des noyaux linéaires (distance euclidienne) ou plus généralement des noyaux gaussiens.

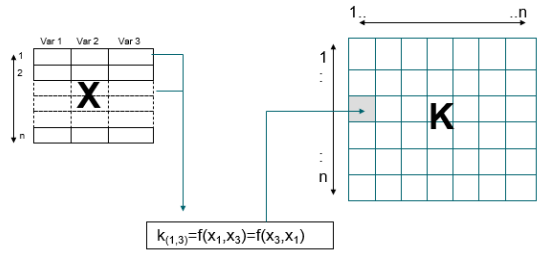

Figure 9 : Transformation de la matrice X en une matrice K à l'aide d'une fonction noyau f

Plusieurs types de fonctions peuvent être mises en œuvre pour le calcul du noyau :

- Linéaire : $K(x_i, x_j) = x_i . x_j$
- Polynomial : $K(x_i, x_j) = (x_i . x_j + 1)^p$
- Gaussienne : $K(x_i, x_j) = e^{\frac{-\|x_i - x_j\|^2}{\sigma''}}$

Ce « kernel-trick », qui transforme les variables en distances, rend généralement les classes linéairement séparables dans ce nouvel espace comme le montre la Figure 10 : (a) 3 classes non séparables linéairement dans l'espace de départ et (b) discrimination linéaire possible avec de larges séparations (à vaste marge) après transformation par un noyau Gaussien.

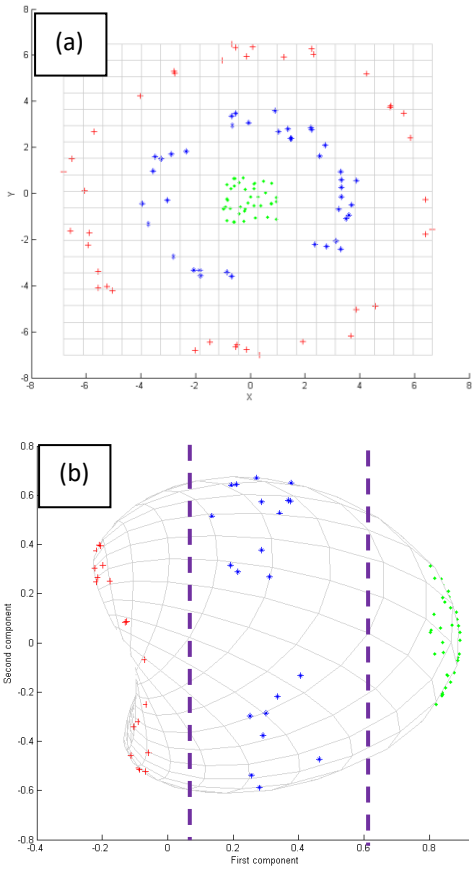

Figure 10 : Effet de la transformation des données par un noyau Gaussien

---

Les SVM pour la prédiction quantitative

Les SVM ont initialement été développés pour la discrimination (*Pattern Recognition*) de 2 classes. Cependant, ils s'adaptent très bien à des problématiques de prédiction quantitative.

Dans ce cas, les vecteurs supports sont alors les échantillons d'apprentissage qui présentent de forts résidus de prédiction, au-delà d'un seuil qui définit alors la marge.

De plus, le calcul de noyau est alors capable de rendre linéaire la plupart des problématiques de prédiction non-linéaires.

Comme pour toute méthode d'apprentissage, un certain nombre de paramètres doivent être optimisés.

➢ Comment optimiser les paramètres des SVM

Dans les cas d'application des SVM en régression, et d'un noyau gaussien, trois paramètres doivent être optimisés :

(a) un paramètre modifiant la taille de la **marge** (ε)

(b) un paramètre réglant **l'écart-type du kernel** Gaussien (σ^2 ou γ selon les logiciels)

(c) un paramètre de **régularisation** ou de coût de l'erreur (C, pour *Cost*).

(a) La taille de la marge (ε)

ε a un effet direct sur la taille de la marge : plus il est faible, plus la marge est petite.

Ce paramètre est directement corrélé au nombre de vecteurs supports sélectionnés comme le montre la Figure suivante (échantillons en bleu).

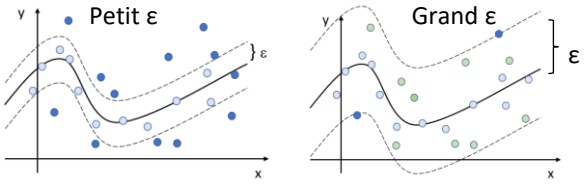

Figure 11: Effet de l'augmentation de ε

(b) Ecart-type du kernel Gaussien (σ^2)

σ^2 est l'écart-type du kernel Gaussien et détermine le degré de non-linéarité du modèle.

Un écart-type plus petit (gaussienne plus étroite et voisinage plus petit) permettra à l'algorithme de représenter de plus fortes non linéarités (Figure 12), en prenant en compte un voisinage plus petit.

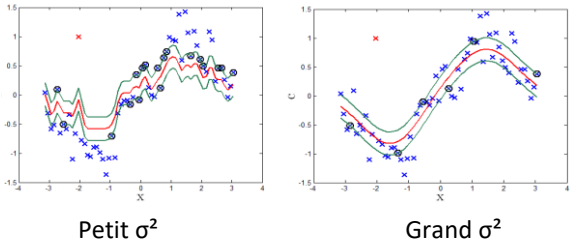

Figure 12: Effet de l'augmentation de σ^2

(c) Régularisation ou coût de l'erreur (C)

Le paramètre de coût ou de « régularisation » (C) pondère l'erreur de prédiction et le niveau des coefficients du modèle lors du calcul du critère d'optimisation (C.f. équation ci-dessous).

$$min\left(C\sum_{i=1}^{n}\xi_i+\frac{b^Tb}{2}\right) \quad \text{Avec} \quad \xi_i=\begin{cases}0 & si\ \ |y_i-\hat{y}_i|<\varepsilon\\ |y_i-\hat{y}_i| & sinon\end{cases}$$

b – coefficients de régression

S'il est élevé, il entraînera un impact important des erreurs de prédiction dans le calcul de la fonction de performance ; ceci peut être dangereux en présence d'*outliers*.

Un coût trop faible peut en revanche mener à un sous-apprentissage, en donnant trop d'importance à la norme des coefficients du modèle par rapport aux erreurs des vecteurs supports.

5.2 Les réseaux de neurones (ANN)

Les réseaux de neurones sont inspirés des neurones biologiques. Ils sont ainsi schématisés par des neurones, liés par des « synapses ». Ces connexions représentent des fonctions de transfert (aussi appelés fonctions d'activation).

A noter que ces *Artificial Neural Networks* (ANNs) sont parfois appelés « shallow ANNs », en opposition aux « Deep ANNs » qui sont les CNN – *Convolutional Neural Networks*. Les CNN ne seront pas abordés ici.

Le type de réseau le plus répandu dans le cas d'apprentissage supervisé est le MLP – Multilayer Perceptron (Figure 13).

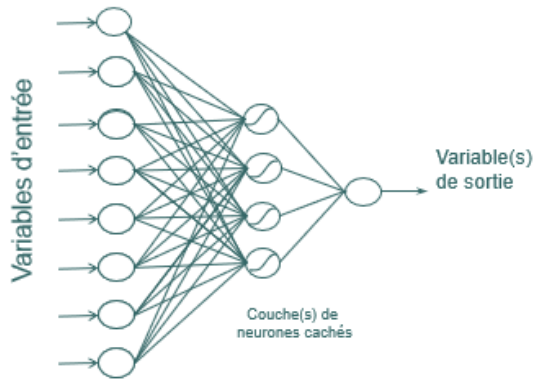

Figure 13 : Architecture d'un réseau de neurones MLP

Le MLP est structuré en plusieurs couches. Une couche est un groupe de neurones uniformes, sans connexion entre eux. La première couche correspond aux variables d'entrée, avec une variable = un neurone.

- Une couche reçoit une variable d'entrée et la transforme en variable de sortie
- Au moins une couche est non linéaire (généralement la(les) couche(s) cachée(s))
- Les dimensions d'entrée et de sortie peuvent être différentes
- A minima deux couches : une couche cachée et une couche de sortie.

Le fonctionnement simplifié de chaque neurone est le suivant :

- A chaque variable entrant sur le neurone (chacune des lignes connectées à celui-ci), un poids est attribué.
- Toutes les entrées d'un même neurone sont combinées
- Une fonction d'activation est appliquée à chaque neurone – parmi les fonctions les plus courantes sont : la sigmoïde, la tangente hyperbolique.
- Le neurone propage son nouvel état vers le neurone suivant.

L'un des inconvénients majeurs des ANN est le risque de sur-apprentissage. Afin de limiter cet effet, il faut contrôler au maximum la complexité du réseau.

Il est par exemple fortement recommandé de compresser les données d'entrée, afin de limiter le nombre de poids d'entrée. Le réseau doit être le plus parcimonieux possible. En pratique en spectroscopie, il est généralement observé qu'une seule couche cachée est suffisante, avec un maximum de 10 neurones.

D'autres moyens de limiter le surapprentissage existent :

- « Early stopping » : critère similaire à la validation croisée – on arrête l'apprentissage lorsque l'erreur de validation augmente.
- Régularisation : une pénalité est ajoutée à la fonction d'erreur des poids.
- Ajout de bruit dans les données d'entraînement.

Les ANN sont complexes à mettre en œuvre et nécessitent de nombreux essais pour le réglage des paramètres et une bonne optimisation.

5.3 Les CART et Random Forest (RF)

5.3.1 CART – Arbre simple

Les arbres de classification et régression (CART) sont ainsi nommés en raison de leurs formes rappelant un arbre : le principe consiste à réaliser des divisions binaires successives d'un jeu de données.

A chaque nœud, une variable est sélectionnée et le jeu de données est séparé en deux selon un seuil déterminé. Lorsque l'arbre est complet, on obtient autant de feuilles que d'échantillons. L'arbre est ensuite « élagué » par validation croisée, afin de réduire le sur-apprentissage. Un exemple d'arbre est présenté Figure 14 : des Iris sont séparés selon la taille et la largeur de leurs pétales.

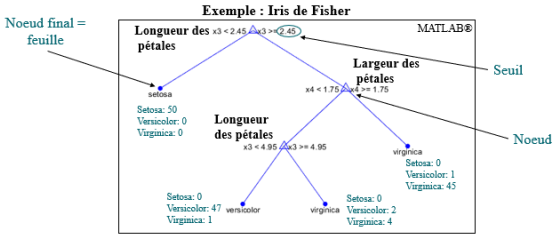

Figure 14 : Exemple de CART - Classification d'Iris

Dans le cas d'une régression, la valeur prédite d'un échantillon correspond à la moyenne des échantillons de la feuille à laquelle il est attribué.

Plusieurs choix doivent dont être faits lorsque l'on cherche à optimiser un CART :

- La variable de séparation de chaque nœud
- Le seuil de chaque variable
- A quel moment arrêter de construire de nouvelles branches

Concernant le choix de la variable discriminante et son seuil, les étapes sont les suivantes :

1. Toutes les combinaisons possibles sont testées
2. L'impureté (mélange de classes) du nœud est calculée
3. La variable discriminante et son seuil associé donnant la diminution maximum de l'impureté sont choisis.

➢ Dans le cadre de la régression, l'impureté d'un nœud est la variance de la variable à prédire dans ce nœud. Plus cette variance est faible, plus le nœud est homogène.

Il est également nécessaire de savoir quand arrêter de construire l'arbre : si l'on construit un arbre complet, pour lequel une feuille = un échantillon, on obtient un modèle sur-appris.

Deux paramètres sont à optimiser pour déterminer l'arrêt :

- Le nombre maximum de divisions dans l'arbre
- La taille minimum d'une feuille

Une autre option consiste à réaliser un arbre complet, puis à l'élaguer à l'aide de l'erreur de validation croisée.

Si les CART présentent de nombreux avantages (facilité d'interprétation, gestion de non-linéarité, gestion de distribution atypique des échantillons, etc.), ils ont également des inconvénients majeurs : une tendance très prononcée au sur-apprentissage, des résultats de prédiction « en escalier », ou encore une forte dépendance au jeu de données.

➢ **Pour régler ces problèmes, une solution est d'utiliser des forêts aléatoires (RF).**

5.3.2 RF – Ensemble d'arbres

Le principe des *Random Forests* (RF) est de combiner un grand nombre de CART afin d'obtenir des modèles plus robustes. Ce type de modèle permet également d'obtenir une meilleure précision des prédictions, et d'éliminer le problème des prédictions « en escalier » généré par les CART.

Les RF utilisent une double randomisation :

- Randomisation des échantillons : des échantillonnages par bootstrap sont réalisés à partir du lot d'apprentissage.
- Randomisation des variables : à chaque division, un nombre restreint de variables est sélectionné aléatoirement.

➢ Cette double randomisation permet de construire différents arbres indépendants et de constituer ainsi une « méthode d'ensemble ».

Comme pour les CART, il est nécessaire d'optimiser quelques paramètres lors de la construction du modèle :

- Le nombre d'arbres construits
- Le nombre maximum de divisions par arbre
- La taille minimum d'une feuille
- Le nombre de variables à sélectionner

6 Implémentation logicielle

Plusieurs logiciels permettent de mettre en œuvre les méthodes de ML présentées.

- PLS_Toolbox - EigenVector Research Inc.:
  - SVM
  - ANN
- Unscrambler – AspenTech
  - SVM
- MATLAB, The Mathworks (avec la Statistics & Machine Learning Toolbox)
  - SVM
  - ANN
  - CART & RF
- Python (avec la bibliothèque Scikit-Learn)
  - SVM
  - ANN
  - CART & RF

7 Références bibliographiques

Cortes, C., Vapnik, V. *Support-vector networks*. Mach Learn 20, 273–297 (1995).

Rosenblatt, F. (1958). *The perceptron: A probabilistic model for information storage and organization in the brain*. Psychological Review, *65*(6), 386–408.

Breiman L, Friedman JH, Olshen RA, Stone CJ. *Classification and Regression Trees*. CRC Press. 1984.

Breiman L. Random Forests. Machine Learning, 45, 5-32 (2001).