Université de Rouen

Rapport de projet

Minimal Learning Machine

Étudiant Laziz Hamdi
Encadrant M. Simon Bernard

2020 — 2021

Table des matières

Chapitre 1

Introduction

Dans le cadre de mes études en Sciences et Ingénieries des Données, je vous présente dans ce document, le sujet de mon projet annuelle, il s'agit du Minimal Learning Machine ou MLM. Ce document se divise en trois parties principales, chaque partie présente l'un des objectifs de ce TER qui sont dans un premier temps de comprendre et de documenter la méthode ensuite de reprendre les expérimentations présentés dans les articles du MLM pour implémenter une version de la méthode de référence qui serait "kernelizable" et de tester cette version avec des Randome Forest Kernel. Le Minimal Learning Machine fonctionne sur des problèmes à grandes dimensions et capable de résoudre des problèmes de régressions et de classifications.

Chapitre 2

MLM

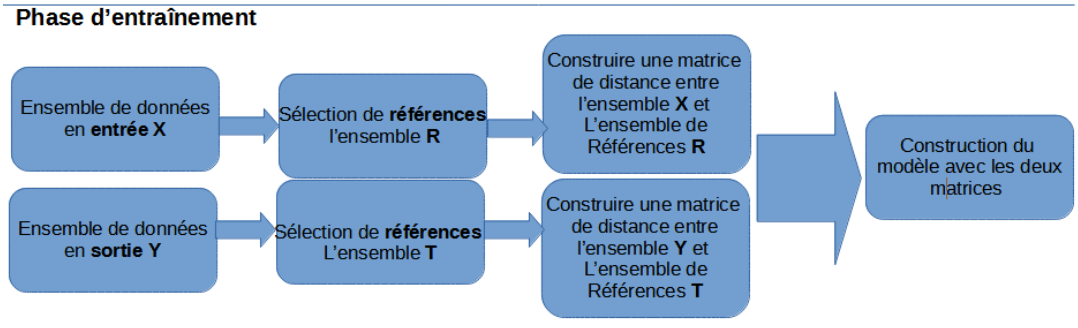
2.1 Définition

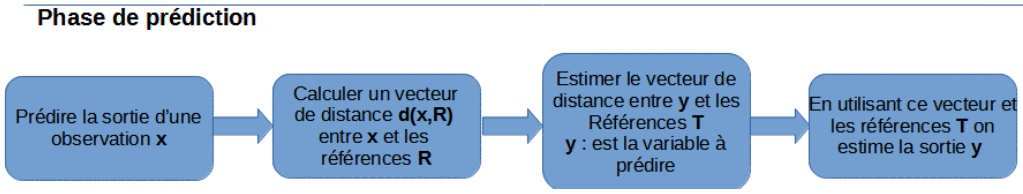
Le Minimal Learning Machine est une technique d'apprentissage supervisé apparu en 2015. Dans sa phase d'entraînement, les données sont projetés dans un nouvel espace. Pour cela des observations aléatoires sont sélectionnées depuis les données, ensuite des distances qu'on appellera "dissimilarités" sont calculées entre ces observations et l'ensemble des données. Dans ce nouvel espace on ne cherche plus à prédire les mêmes données mais plutôt à prédire des distances. Et une fois ces distances prédites, on utilise le processus inverse pour estimer les réelles valeurs.

2.2 Phase d'entraînement

2.2.1 Formulation

Pour un ensemble de données en entrée $X = \{x_i\}_{i=1}^{N}$ avec $x_i \in R$ et sa correspondance en sortie $Y = \{y_i\}_{i=1}^{N}$ avec $y_i \in S$.
On suppose qu'il existe une relation continue entre ces deux espaces qu'on appelle $f : X \to Y$, l'objectif est d'estimer f.
Le processus du MLM se divise en deux étapes principales :





2.2.2 Construire les matrices de distances

Le MLM requière que l'utilisateur précise le nombre de points références k à sélectionner pour construire ce qu'on appelle des matrices de distances ou de dissimilarités. Si on considère D une matrice de distance de taille n, m, la valeur qui se trouve à l'intersection de la ligne i et la colonne j représente la distance euclidienne entre l'observation i et la référence j. De ce fait les lignes de D représentent l'ensemble des observations et les colonnes l'ensemble des points références.
On définie l'ensemble des points références $R = \{m_k\}_{k=1}^{k}$ sélectionnés aléatoirement de l'ensemble X et leurs correspondances $T = \{t_k\}_{k=1}^{k}$ de l'ensemble Y, ensuite on construit la matrice $D_x \in R^{N \times K}$ des distances euclidiennes entre les observations x_i en ligne et les références en colonne, de la même manière est définie la matrice des distances en sortie $\Delta_x \in R^{N \times K}$.
On définie la relation entre ces matrices ainsi $\Delta_y = g(D_x) + E$ avec E qui représente le résidus.
On peut représenter cette relation sous forme matricielle $\boldsymbol{\Delta_y = D_x B + E}$.
$\boldsymbol{B}$ est la matrice des coefficients du modèle.

2.2.3 Construire le modèle

Pour estimer $\boldsymbol{B}$ c'est à dire les coefficients du modèle, plusieurs méthodes peuvent être utiliser comme les moindres carrés moyens, moindres carrés récursifs pour calculer la différence entre les vraies distances et les distances prédites. cette différence est exprimé avec cette fonction.
$\boldsymbol{RSS(B) = tr((\Delta_y - D_x B)'(\Delta_y - D_x B))}$
Minimiser cette fonction revient à chercher le point ou le gradient est null, ce qui conduit à résoudre un système d'équations ou le nombre d'équations est le nombre d'observations $\boldsymbol{N}$ et le nombre d'inconnue est le nombre de références $\boldsymbol{K}$. La solution est différente selon le nombre $\boldsymbol{K}$.

- Pour $\boldsymbol{K < N}$: $\boldsymbol{\hat{B} = (D_x{}' D_x)^{-1} D_x{}' \Delta_y}$
- Pour $\boldsymbol{K = N}$: $\boldsymbol{\hat{B} = D_x{}^{-1} \Delta_y}$
- Pour $\boldsymbol{K > N}$: une infinité de solutions.

Ce dernier cas se produit lorsque après sélection des points références uniquement une partie des données est utilisé pour créer le modèle, ceci donne naissance à un problème indéterminé car le nombre d'équations est plus petit que le nombre d'inconnu avec une infinité de solution.

2.3 Phase de prédiction

Une fois $\boldsymbol{B}$ estimer pour un point en entrée $\boldsymbol{x}$, on construit un vecteur de distances euclidienne entre ce point et l'ensemble des point références $\boldsymbol{d(x, R) = [d(x, m_1)...d(x, m_k)]}$, alors dans le cas ou $\boldsymbol{K = N}$ ou $\boldsymbol{k < N}$, le vecteur des distances en sortie est le produit entre le vecteur des distances en entrée et la matrice des coefficients.
$\boldsymbol{\hat{\delta}(y, T) = d(x, R)\hat{B}}$ avec $\boldsymbol{\hat{\delta}(y, T) = [\hat{\delta}(y, t_1)...\hat{\delta}(y, t_k)]}$

$\boldsymbol{y}$ est estimé en utilisant le vecteur des distances $\boldsymbol{\hat{\delta}(y, T)}$ et les références $\boldsymbol{T}$ avec ce qu'on appelle une multitaréation. C'est une technique qui utilise des mesures de distance pour relever les coordonnées spatiales de positions inconnues. En pratique les distances sont mesurées avec erreur, et les méthodes statistiques peuvent quantifier l'incertitude de l'estimation de la position inconnue. De nombreuses méthodes d'estimation de la position d'un point par multitération peuvent être utiliser comme un estimateur linéaire des moindres carrés, un estimateurs des moindres carrés pondéré de manières itérative et une technique non linéaire des moindres carrés. En général la technique des moindres carrés non linéaire est la plus performante.

Pour estimer y on minimise la fonction objective suivante :
$\jmath(y) = \sum_{k-1}^{k}((y - t_k)\prime(y - t_k) - \hat{\delta}^2(y, t_k))^2$
Cette fonction de coût possède un minimum en 0 qui est atteint seulement si la valeur estimer est égal à la vrai valeur, c'est à dire que les valeurs prédites sont égal au valeurs réels $\prime y = y$. Sinon on approche le plus possible y avec un algorithme de minimisation.

Plusieurs algorithmes de minimisation peuvent être utilisés mais le plus adapter pour ce problème est L'algorithme de Levenberg-Marquardt qui permet de trouver une solution numérique à un problème de minimisation d'une fonction non linéaire dépendant de plusieurs variables. Cet algorithme est plus stable et trouve une solution même s'il démarre très loin du minimum.

2.4 Choix du paramètre K

Le principale avantage du MLM est qu'il ne possède qu'un seul hyper paramètre K à optimiser le nombre de point références que l'utilisateur doit rentrer. Pour optimiser le nombre de point références la validation croisée est utilisée, en divisant l'ensemble des données en F sous-ensembles ensuite en testant avec différentes valeurs de k les taux de réussites sont calculés avec les moindres carrés moyens pour les vecteur de distances en sortie $\hat{\delta}$ et l'estimation des $\hat{y}$

$$AMSE(\delta) = \frac{1}{K}\sum_{k=1}^{k}\frac{1}{N_v}\sum_{i=1}^{N_v}(\delta(y_i, t_k) - \hat{\delta}(y_i, t_k))^2$$
$$AMSE(y) = \frac{1}{4}\sum_{s=1}^{S}\frac{1}{N_v}\sum_{i=1}^{N_v}(y_i^{(s)} - \hat{y}_i^{(s)})^2$$

Les points références sont sélectionnés aléatoirement des données.

2.5 Performances et Complexité

La complexité pour l'étape de formation du modèle dépend fortement de la méthode utiliser pour le calcul des inversions de matrice surtout qu'on construit des matrices de distances toujours plus grandes selon la taille des données et le nombre de points références sélectionnés, puisque on construit des matrices de tailles N, K.
L'une des méthodes les plus connue est l'inverse de Moore-Penrose qui estime une pseudo inverse de la matrice, car dans certains cas les matrices ne sont pas inversibles. L'une des constructions les plus connues de cette méthode est la décomposition en valeurs singulières SVD qui est très précise mais très gourmande en temps de calcule et est plusieurs fois plus élever que le produit matrice-matrice.

Pour accélérer le calcule, plusieurs méthodes ont été proposé comme le produit entre un type spécial de tenseur et une décomposition QR ou encore un algorithme basé sur une factorisation Cholesky.
La complexité de la phase d'entrainement du MLM est $\Theta(K^2N)$ c'est similaire à celle d'un algorithme de machine learning lorsque le nombre de neurones cachés est égal au nombre de références K.

Le MLM est testé sur 12 ensembles de données les plus fréquemment utiliser dans le monde ensuite ces performance sont comparées à celle de cinq autre méthode de références le machine learning extrême ELM, le réseau de fonction à base radial RBF, les machines à vecteur de support SVM, les processus gaussiens GP et le percepteron multicouches MLP. Tous les ensembles de données sont pré traités de la même manière pour reproduire les expériences à l'identique supprimer les données manquantes, supprimer les données catégorielle, normaliser de la même façon et utiliser la même proportion de données pour l'entraînement et le test.

Pour ces tests le seul hyper paramètre K du MLM est optimisé avec une validation croisé sur 10-Fold, avec une sélection aléatoire des références depuis l'ensemble de données pour k allant de 5% à 100% avec un pas de 5%. Tous les modèles sont évalués en utilisant l'erreur quadratique moyenne MSE sur 10 tests indépendants. Le MLM obtient le plus petit taux d'erreur pour 5/8 des problèmes de régressions et pour les autre problèmes il obtient des résultats proches des résultats obtenue par les autres méthodes.
Ces expériences montre qu'en utilisant 20% des points d'apprentissages comme points références semble être un bon choix pour la plus part des ensembles de données.

Chapitre 3

Expérimentations

Pour tester le Minimal Learning Machine sur des problèmes de régressions, les datasets Abalone, Ailerons, Housing, Servo, Auto Price, Elevators sont utilisés et pour des problèmes de classifications, on utilise les datasets Breast Cancer, Iris, Wine.

Tous les datasets sont pré traités de la même façon, les données sont centrés et réduites, les colonnes catégorielles ainsi que les observations qui contiennent des données manquantes sont supprimées. Dix différentes permutations aléatoires sont appliquées pour chaque dataset. 2/3 de l'ensemble de données sont utilisés pour la phase d'entrainement du modèle et 1/3 pour la phase de test. Pour avoir une idées des performances du Minimal Learning Machine sur ces datasets, on compare les résultats obtenu avec cette méthode au résultats obtenu avec les Machines à Vecteur de Support (SVR), pour la régression et (SVC) pour la classification.

Le seul hyper paramètre du Minimal Learning Machine c'est à dire le nombre de point référence, est optimisé avec une validation croisé sur 10 cv, en allant de 5% à 100% de taille totale de l'ensemble de données. Pour les SVM on utilise le noyau gaussien (RBF) car il donne de meilleurs résultats. Les autres hyper paramètres des SVM sont optimisés avec un GridSearch.

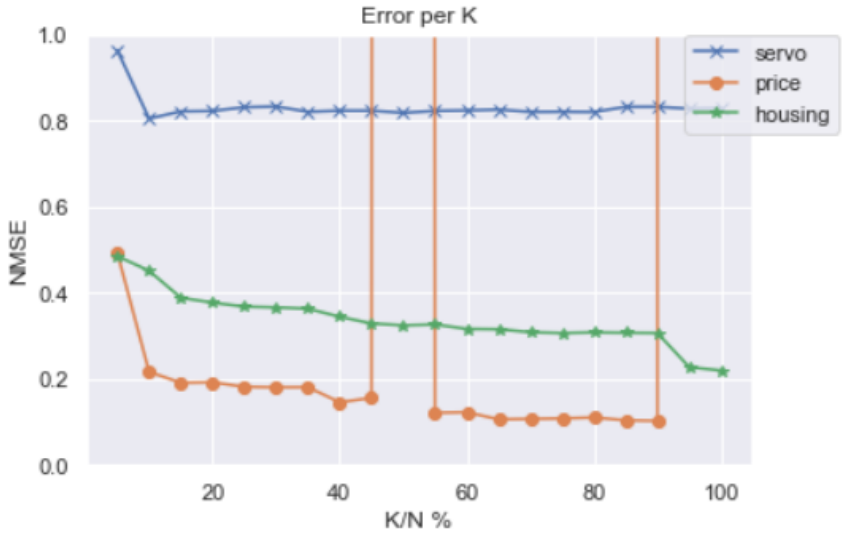
3.1 Sur des problèmes de régressions

Pour calculer les taux de précision le Mean Square Error est utilisé, les résultats sont affichés dans le tableau suivant :

	Servo	Auto price	Boston housing
MLM (MSE)	0.41	0.23	0.11
SVM (MSE)	0.46	0.19	0.16

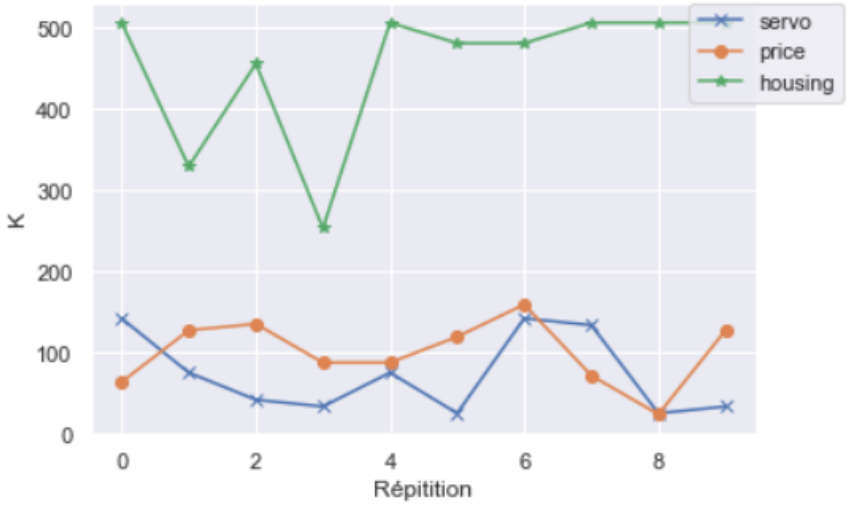
On observe que sur 2/3 des datasets (Servo et Boston housing) le Minimal Learning Machine obtient le meilleur score. Sur ces résultats il manque les dataset Abalone, Elevators, Ailerons car leur exécution prends beaucoup de temps (plusieurs heures).

La figure suivante représente l'évolution de l'erreur en fonction du nombre de références $\boldsymbol{K}$ c'est à dire pour chaque valeur que prend $\boldsymbol{K}$ on calcule le Mean Square Error entre les valeurs prédites et les valeurs réelles ensuite cette valeur est normalisé en la divisant sur le Mean Square Error de ces valeurs prédites.



On voit que les trois courbes évoluent presque de la même façon or mis le changement brusque pour le dataset Auto Price entre les valeurs 40% et 60% de l'axe des abscisses.

Dans la figure qui suit, on affiche le nombre optimal de point références pour chaque dataset sur 10 différentes exécutions.



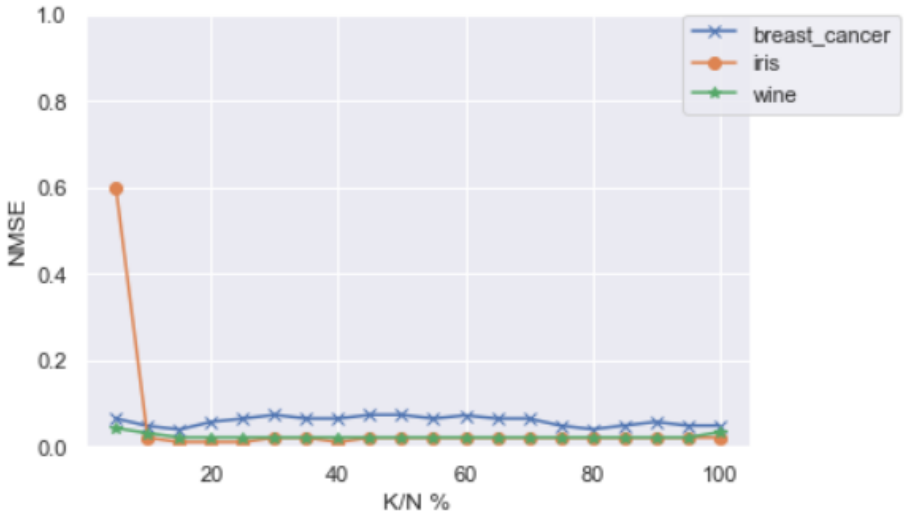
3.2 Sur des problèmes de classifications

Pour la classification on utilise le même processus utilisé pour la régression sauf pour le calcule du taux de précision on utilise l'accuracy.

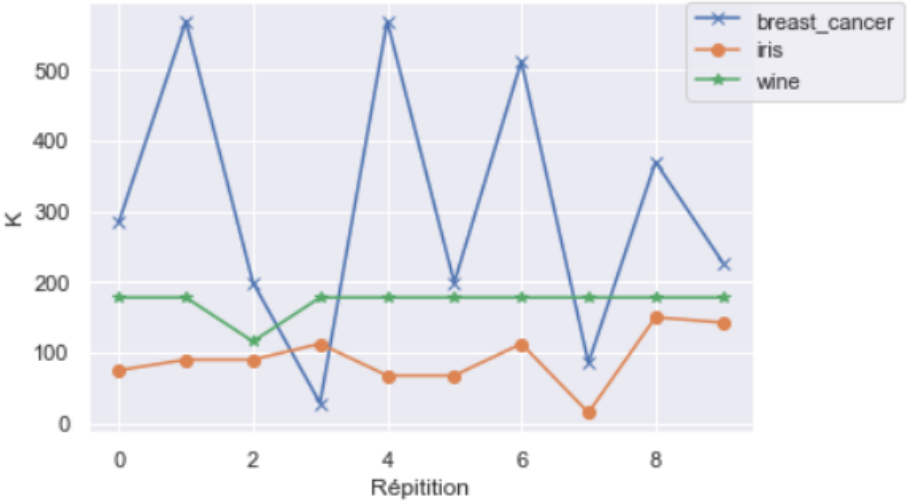
	Boston house	Iris	Wine
MLM (Accuracy)	0.91	0.92	0.99
SVM (Accuracy)	0.95	0.94	0.97

Le Minimal Learning Machine obtient le meilleur score pour le dataset Wine et il obtient un score proche des résultats du SVM pour les autres datasets

L'évolution de l'erreur en fonction du nombre de références :

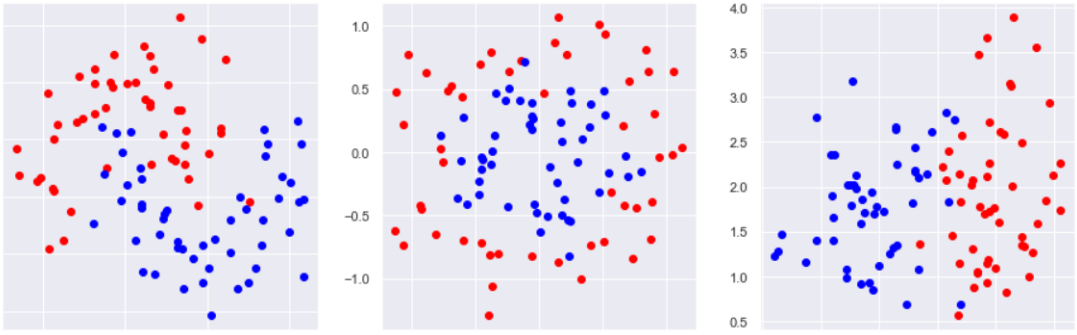


Le nombre optimal de point références pour chaque dataset sur 10 différentes exécutions :



Comme c'est difficile de faire des tests sur des données réelles car l'exécution peut prendre énormément de temps, surtout pour optimiser les hyper paramètre, avec une validation croisée ou une recherche en grille. Donc trois nouvelles ensembles de données de 100 observations à deux dimensions, sont générés, pour comparer les performances du Minimal Learning Machine au méthodes d'apprentissages supervisés les plus connues. Les données sont pré traités toujours de la même façon.

Voici une représentation graphique de la forme des données :



Les résultats obtenus avec les différents modèles sont affichés dans ce tableau :

	Nearest Neighbors	RBF SVM	Gaussian Process	Decision Tree	Neural Net	Naive Bayes	MLMC	NN_MLM
make moon	0.875	0.900	0.900	0.850	0.875	0.875	0.701225	0.70000
make circle	0.875	0.925	0.950	0.900	0.925	0.875	0.702947	0.69697
linearly separable	0.950	0.950	0.975	0.875	0.950	0.950	0.844479	0.90000

Pour ce test le Minimal Learning Machine pour la classification (MLMC) classique est utilisé ainsi qu'une autre variante pour la classification qui se base sur une approche au plus proche voisin.

Chapitre 4

Conclusion

Les résultats des expériences montre que le Minimal Learning Machine peut être un réel atout pour résoudre des problèmes d'apprentissages supervisés. La complexité du MLM lors de la formation du modèle est faible en concurrant les techniques de machine learning les plus rapides. Pour la phase de test une sélections aléatoire des points références et une optimisation avec l'algorithme de Levenberg-Marquardt permet d'atteindre des performances de pointes.