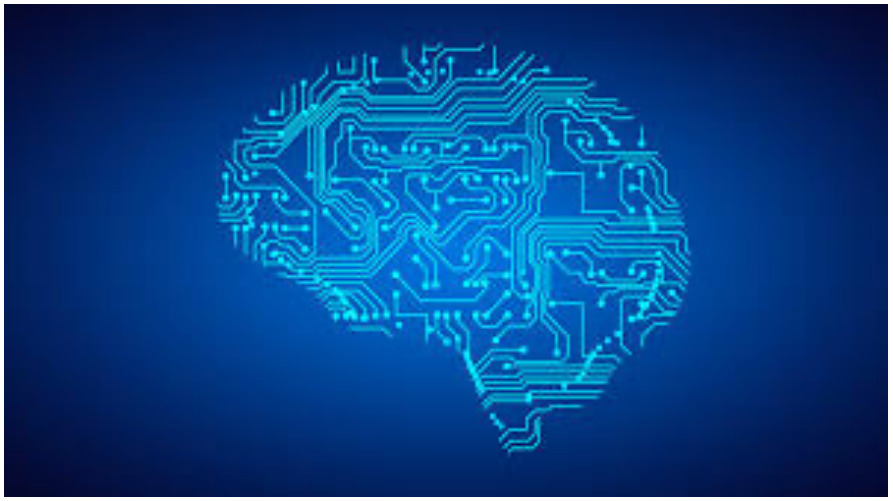
Informatique 3OC

A. Ridard

Machine learning



Gymnase Yverdon

canton de vaud

Table des matières

I. Introduction

Entre statistique et informatique, le **machine learning**[1] est une famille d'algorithmes permettant à une machine de « résoudre » certains problèmes, de manière automatique à partir d'une « grande » quantité de données.

On distingue trois branches qui correspondent à des problèmes différents :

- l'apprentissage **supervisé** cherche à « prédire » une information (inconnue) pour un individu[2], à partir d'un grand nombre d'individus observés pour lesquels on dispose de cette information.
- l'apprentissage **non supervisée**[3] vise à regrouper les individus qui se « ressemblent », qui ont le même « profil ».
- l'apprentissage **par renforcement**[4] permet à un agent autonome, disons une intelligence artificielle comme ChatGPT, de trouver le meilleur comportement, au travers d'expériences itérées fournissant des récompenses positives ou négatives selon les succès ou échecs constatés.

Pour présenter les deux types d'apprentissage supervisé, nous avons besoin de formaliser un peu les choses.

L'étude d'un phénomène réel « complexe » commence souvent par une phase de **modélisation** qui consiste à transformer ce problème de la « vraie vie » en un problème « plus simple ». Cette **approximation simplificatrice** de la réalité se limite à certaines caractéristiques, avec lesquelles on espère pouvoir déterminer une solution.

Ces caractéristiques sont appelées des **variables**, puisque leurs valeurs varient selon les **individus**. Plus précisément, nous supposerons qu'un individu x est caractérisé par p **variables prédictives**[5] $X_1, X_2, \dots, X_p$, autrement dit un individu x sera vu comme un vecteur $(x_1, x_2, \dots, x_p)$.

Si l'on note Y la **variable cible**, appelée aussi **étiquette**, c'est à dire l'information que l'on cherche à prédire, le problème consiste à approcher au mieux la valeur de Y pour un « nouvel » individu $x = (x_1, x_2, \dots, x_p)$, à partir d'un échantillon $x^{(1)}, x^{(2)}, \dots, x^{(N)}$ de N **individus étiquetés**.

En résumé, nous souhaitons approcher le y de $x = (x_1, x_2, \dots, x_p)$ à partir des données :

$$\left(\begin{array}{cccc|c} x_1^{(1)} & x_2^{(1)} & \dots & x_p^{(1)} & y^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_p^{(2)} & y^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{(N)} & x_2^{(N)} & \dots & x_p^{(N)} & y^{(N)} \end{array}\right)$$

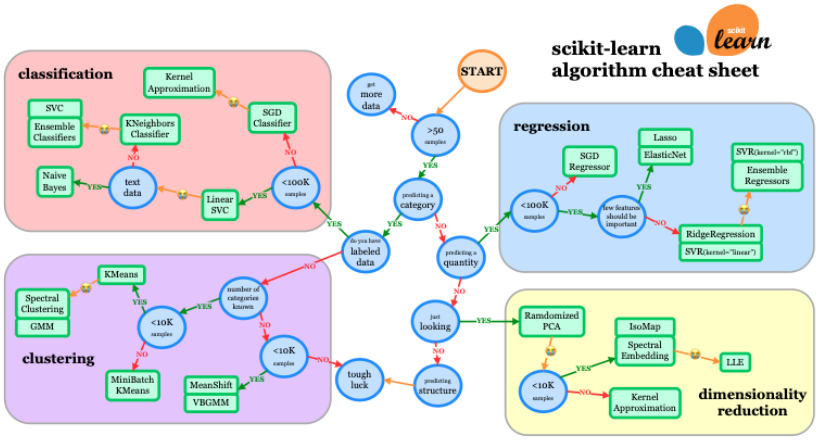
Maintenant que nous avons la forme des données, il convient d'en préciser la nature.
Une variable est dite **quantitative** si ses valeurs sont des nombres, et **qualitative** sinon.

Cette différence permet de distinguer les deux types d'apprentissage supervisée :

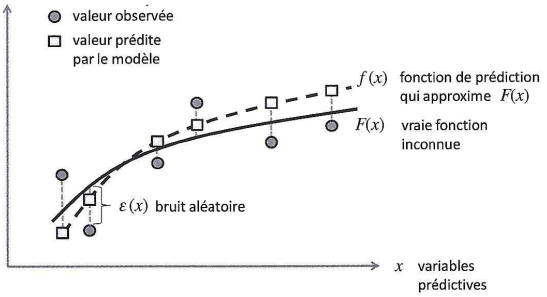
1. la **régression** si la variable cible est quantitative
2. la **classification** si elle est qualitative

[1]. **apprentissage automatique** en français
[2]. Au sens statistique du terme
[3]. Il s'agit du clustering que nous n'aborderons pas dans ce cours
[4]. Nous ne traiterons pas ce type d'apprentissage qui sort largement du cadre de ce cours !
[5]. On note en majuscule les variables et en minuscules les valeurs prises par ces variables

En TP, nous utiliserons la bibliothèque **scikit-learn** qui propose cet arbre de choix :

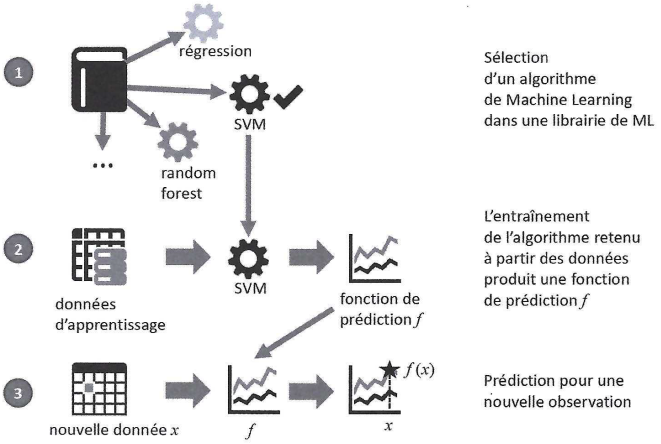


Revenons à l'apprentissage supervisée avec ce schéma [6] qui devrait fixer les idées.



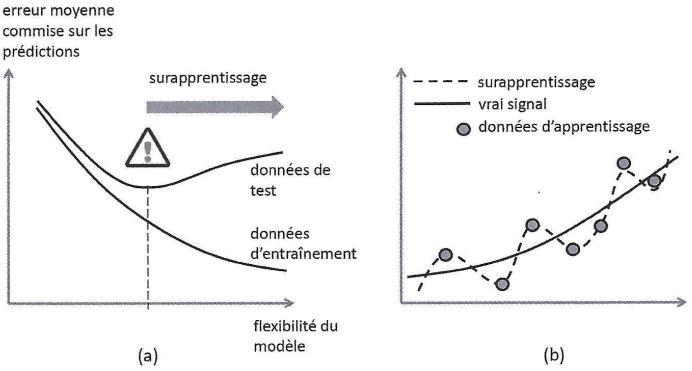
La valeur d'une variable cible est la somme d'une **fonction déterministe** F et d'un **bruit aléatoire** ϵ. L'objectif du ML est de trouver une bonne approximation de F à partir des observations. Cette approximation f est la **fonction de prédiction** qui permet d'obtenir une estimation $f(x)$ de $F(x)$.

Donnons maintenant la démarche générale pour saisir le rôle joué par les données.



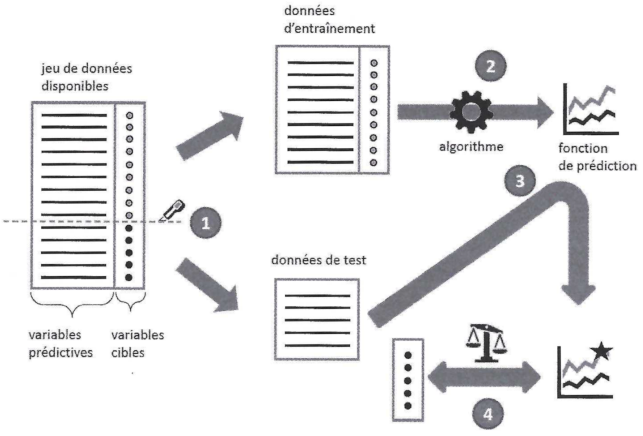
[6]. Dans sa forme, il correspond plus à une régression, mais il peut s'étendre à une classification

Un risque bien connu des *data scientists* est le **surapprentissage** qui caractérise une situation dans laquelle un modèle de ML reproduit avec une grande précision les données d'apprentissage, mais sans parvenir à **généraliser** de manière satisfaisante.



Avec un modèle « trop flexible », on prend le risque de capter les bruits lors de la phase d'entraînement, éloignant ainsi la fonction de prédiction f du signal F à découvrir pour bien généraliser ou prédire par la suite.

D'où proviennent ces **données de test** permettant d'évaluer la qualité du modèle ?



II. Classification

1. *k* plus proches voisins

Pour présenter simplement cet algorithme, nous allons utiliser le problème de la « carte scolaire » qui a pour vocation d'affecter chaque élève dans un gymnase proche de son domicile. Mais d'autres facteurs comme les capacités d'accueil des établissements et les données démographiques des quartiers peuvent intervenir pour équilibrer la répartition globale.

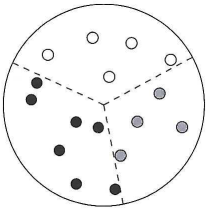
Ce problème est modélisé de la manière suivante :

- Chaque individu (élève ici) est caractérisé par une variable quantitative correspondant à sa position géographique[7].
- La variable cible est le gymnase d'affectation, et peut prendre ici trois valeurs différentes que l'on associera à trois couleurs différentes : blanc, gris et noir.

Remarquons au passage qu'il s'agit bien d'une classification.

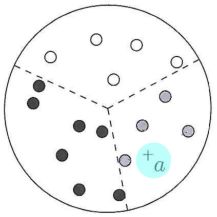
[7]. Voire deux si l'on distingue latitude et longitude

Une phase d'observation qui se traduit ici par un sondage dans le voisinage fournit la situation suivante.



Notons que les pointillés décrivent la réalité **inconnue** de la carte scolaire c'est à dire la fonction déterministe F que l'on cherche à approcher.

Comment prédire la couleur d'un nouvel individu à partir de ces individus étiquetés ?

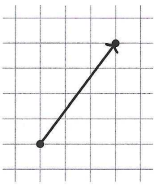


Comme son nom l'indique, nous allons considérer les k plus proches voisins du nouvel individu a, puis lui attribuer la valeur majoritaire [8] pour la variable cible.

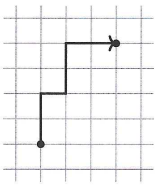
Pour cela, nous devons choisir en amont deux **hyper-paramètres** [9] :

- une **métrique**, c'est à dire un moyen de mesurer une « distance »
- le nombre k de voisins à considérer

Concernant la métrique, nous allons opter naturellement pour la distance euclidienne, mais il en existe d'autres.



Distance euclidienne : 5



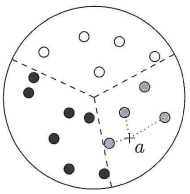
Distance Manhattan : 7

Ces distances sont définies de la manière suivante :

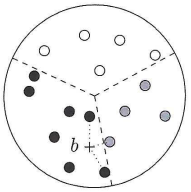
- Distance euclidienne [10] : $d(A,B) = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$
- Distance Manhattan [11] : $d(A,B) = |x_B - x_A| + |y_B - y_A|$

Dans cette méthode, le choix de l'hyper-paramètre k est certainement le plus délicat.

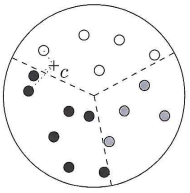
Prenons par exemple $k = 3$:



3 voisins : ●●●
résultat : ●



3 voisins : ●●●
résultat : ●

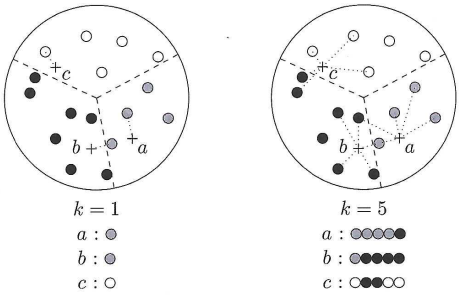


3 voisins : ○●●
résultat : ●

[8]. Si des valeurs sont ex aequo, on en choisit une au hasard.
[9]. Ne pas confondre avec les paramètres des modèles paramétriques que nous verrons plus loin
[10]. A vol d'oiseau
[11]. A travers les rues de la ville de Manhattan

Dans ce cas, seul l'individu c est mal classé.

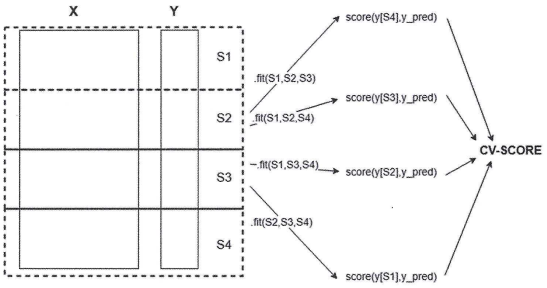
Puis comparons avec $k = 1$ et $k = 5$:



Avec $k = 1$, seul l'individu b est mal classé.
Avec $k = 5$, tous les individus sont bien classés.

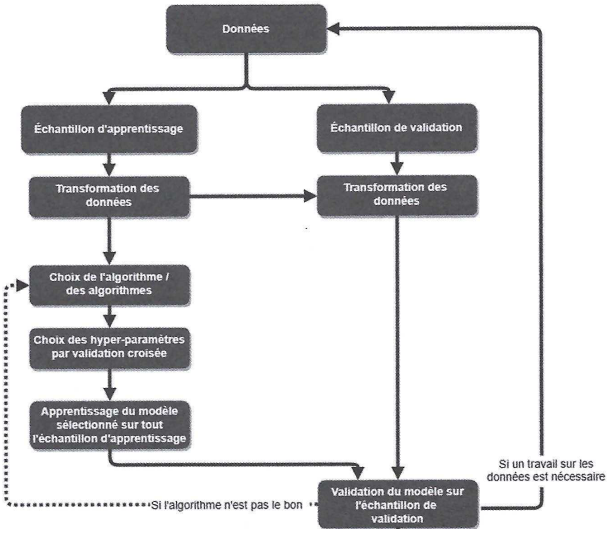
Attention, une trop grande valeur de k a aussi ses défauts.
Que se passe-t-il pour l'individu a avec $k = 9$?

Dans la pratique, pour choisir le meilleur k, nous avons recours à la **validation croisée** qui permet de tester sur une partie des données d'entraînement.



Pour chaque valeur de k, nous obtenons ainsi un **cv-score** qui mesure la qualité [12] de notre modèle, et permet de choisir le meilleur k.

Voici à nouveau la démarche générale, mais un peu plus détaillée.

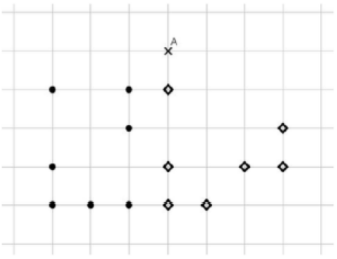


Cet algorithme est apprécié pour sa simplicité conceptuelle, mais il est particulièrement sensible au bruit, et au delà de dix variables prédictives, il est conseillé d'utiliser des techniques de réduction dimensionnelle [13] pour limiter le coût des calculs. Malgré tout, il est qualifié de « paresseux » car sa prédiction est lente !

[12]. Nous préciserons plus tard les différentes manières de mesurer cette qualité
[13]. Tout comme l'apprentissage par renforcement, ces techniques ne seront pas traitées dans ce cours

Dans le quadrillage ci-dessous, 14 points sont représentés :

- 7 de la classe C_1 (ronds)
- 7 de la classe C_2 (losanges)



Pour la distance Manhattan, déterminer la classe du nouveau point A avec :

1. $k = 1$
2. $k = 3$
3. $k = 5$

Considérons les données d'apprentissage suivantes :

$$\left(\begin{array}{ccccc|c} 3 & 5 & 4 & 6 & 1 & 1 \\ 4 & 6 & 10 & 3 & 2 & 2 \\ 8 & 3 & 4 & 2 & 6 & 3 \\ 2 & 1 & 4 & 3 & 6 & 3 \\ 2 & 5 & 1 & 4 & 8 & 2 \end{array}\right)$$

Déterminer[a] la classe (1, 2 ou 3) du nouvel individu $x = (3, 12, 4, 7, 8)$ avec :

1. $k = 1$
2. $k = 3$

a. A l'aide d'un tableur et pour la distance euclidienne

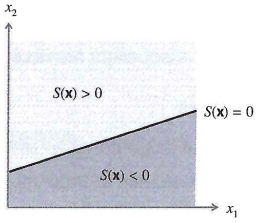
2. Régression logistique

Nous cherchons avec ce modèle à prédire une variable cible **binaire** [14].

Commençons par indiquer qu'il s'agit d'un modèle **linéaire** dans le sens où il utilise une combinaison linéaire des variables prédictives :

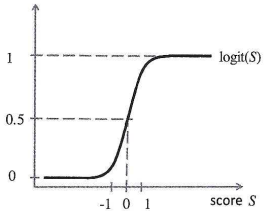
$$S(x) = a_1 x_1 + a_2 x_2 + \cdots + a_p x_p$$

Cette combinaison linéaire, appelée **Score** [15], cherche à partager l'espace des variables prédictives en deux parties, à l'aide d'un hyperplan [16] : l'une espérant capter les individus prenant la valeur 0, et l'autre ceux prenant la valeur 1.



Lors de la phase d'apprentissage (ajustement), la machine va chercher à optimiser [17] les paramètres $a_1, a_2, \ldots, a_p$ du modèle [18] de sorte que $S(x)$ soit positif lorsque les chances d'appartenir au groupe 1 sont grandes [19], et négatif sinon.

Pour calculer cette probabilité d'appartenance au groupe 1, le modèle utilise la fonction logistique appliquée au score S : $logit(S) = \dfrac{1}{1+e^{-S}}$



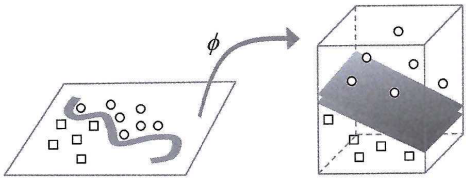
Contrairement à l'algorithme des k plus proches voisins, la classification d'un nouvel individu est ici extrêmement rapide puisqu'elle se résume à l'évaluation de la fonction linéaire S. Mais ce modèle possède aussi l'inconvénient de son avantage, puisqu'il n'est pas toujours possible de séparer correctement les individus de manière linéaire !

3. SVC

Comme précédemment, nous cherchons avec ce modèle à prédire une variable cible binaire, mais cette fois de manière **non linéaire**.

Le principe des **SVM** (séparateurs à vaste marge) consiste à construire une bande séparatrice non linéaire de largeur maximale pour séparer les deux groupes (0 et 1), et à l'utiliser pour faire les prédictions.

Pour y parvenir, les SVM utilisent une transformation ϕ qui envoie les points (individus) $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$ de l'espace initial à p dimensions dans un espace de dimension plus grande, où ils seront plus faciles à séparer de manière linéaire.



Nous ne donnerons pas plus de détails, mais nous serons amenés à utiliser ces modèles extrêmement puissants en TP.

[14]. A deux valeurs que l'on peut toujours *renommer* 0 et 1
[15]. Ne pas confondre avec celui qui mesure la qualité d'un modèle !
[16]. Une droite dans le plan, un plan dans l'espace, ...
[17]. A l'aide de méthodes numériques
[18]. Voici donc un modèle paramétrique !
[19]. Supérieures à 0,5

III. Régression

1. k plus proches voisins

Revenons à notre premier algorithme des k plus proches voisins, et voyons comment il peut être adapté pour résoudre un problème de régression cette fois.

Imaginons que nous souhaitions estimer le prix de vente d'un bien immobilier, la variable cible (le prix au m^2) n'est plus qualitative, mais bien quantitative. Comme ce prix dépend de l'emplacement, de l'emplacement, et aussi de l'emplacement [20], c'est plutôt naturel d'utiliser le prix observé des ventes voisines. De manière à comparer tout de même ce qui est comparable, on peut enrichir notre description d'un individu (bien immobilier) en considérant d'autres variables que l'emplacement comme le type de bien (maison ou appartement), sa superficie, son année de construction et sa classe énergétique. Les voisins ne seront plus simplement des voisins au sens géographique du terme, mais des biens qui se ressemblent compte tenu de toutes les caractéristiques.

Maintenant que nous sommes en mesure de choisir les k plus proches voisins, et en imaginant que nous travaillons avec $k = 5$, comment le modèle doit-il estimer le prix à partir des 5 prix voisins suivants ?

2240, 2240, 2339, 2340, 2341

La règle de la majorité n'a évidemment plus beaucoup de sens ici, on préfèrera utiliser la moyenne de ces valeurs, éventuellement pondérée pour donner plus d'importance aux voisins très proches.

Ce modèle a les mêmes avantages et inconvénients que dans le cas de la classification.

2. Régression linéaire (régularisée)

Ce modèle paramétrique suppose [21] que la fonction de prédiction f est de la forme :

$$f(x) = a_1 x_1 + a_2 x_2 + \cdots + a_p x_p + b$$

que l'on peut réécrire de manière affine à l'aide du produit scalaire :

$$f(x) = a \cdot x + b$$

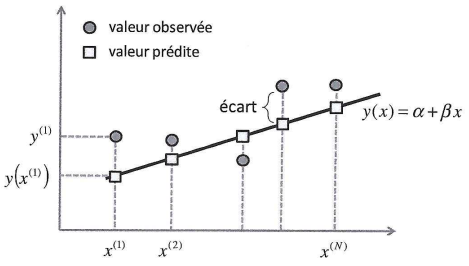
et même de manière linéaire en ajoutant une variable X_0 constante égale à 1 :

$$f(x) = a_0 x_0 + a_1 x_1 + a_2 x_2 + \cdots + a_p x_p = a \cdot x$$

avec $a_0 = b$.

Le caractère linéaire du modèle est ainsi justifié !

Lors de la phase d'apprentissage (ajustement), la machine va chercher à optimiser [22] les paramètres $a_0, a_1, a_2, \ldots, a_p$ du modèle, de manière à minimiser les erreurs [23] de prédiction (sur le jeu de données d'apprentissage).



Sans contrainte supplémentaire sur l'optimisation des paramètres, ces derniers peuvent prendre de grandes valeurs, rendant l'ajustement du modèle particulièrement sensible dans le sens où de faibles changements dans les données d'apprentissage peuvent produire des modèles très différents !

[20]. La règle d'or pour un achat immobilier ;)
[21]. Il s'agit d'une hypothèse très forte qui aura tendance à faire du sous-apprentissage !
[22]. En résolvant un système d'équations linéaires
[23]. On cherche à minimiser l'erreur quadratique moyenne, c'est à dire à obtenir les *moindres carrés* entre valeur observée et valeur prédite

Pour limiter ce problème, on a recours aux méthodes de régularisation :

- Ridge [24] diminue l'importance de certains paramètres
- Lasso [25] est plus sévère que Ridge, elle annule certains paramètres
- ElasticNet est une combinaison des deux précédentes

3. SVR

Comme pour la classification (SVC), les SVM fournissent des modèles de régression (SVR) non linéaires extrêmement puissants que nous utiliserons en TP.

[24]. basée sur la norme 2
[25]. basée sur la norme 1