Chapitre 10

Algorithmique de l'apprentissage et de la fouille de données

L'apprentissage artificiel est essentiellement centré sur l'étude et la mise au point de méthodes permettant la recherche de régularités à partir de données correspondant à des cas particuliers. Pour ce problème général d'induction, de nombreux algorithmes ont été développés depuis les premiers travaux dans les années mille neuf-cent cinquante. Ce chapitre introduit d'abord le problème de l'induction et l'importance en particulier de l'espace des hypothèses considéré, qui est lié à la représentation des connaissances jugée adéquate. C'est alors en fonction des caractéristiques générales de ces espaces d'hypothèses que sont présentées les grandes familles d'algorithmes d'apprentissage existants. La conclusion esquisse les nouvelles questions et pistes de recherche qui définissent la nouvelle frontière de la recherche en apprentissage artificiel.

10.1 Introduction

Apprendre, cela signifie faire sens du monde, y découvrir des régularités, des lois permettant de comprendre et/ou de prédire, cela fait également référence à la capacité d'adaptation, c'est-à-dire d'auto-modification afin d'améliorer sa performance. Le fait que le programme d'un ordinateur puisse, à l'instar du reste de la mémoire, être considéré par la machine comme des données modifiables a très tôt fasciné les pionniers de l'intelligence artificielle (voir l'excellent ouvrage de Nilsson (Nilsson, 2010)), ce qui a conduit à une floraison de travaux exploratoires durant les années cinquante à soixante-dix environ. Cependant, il faut reconnaître que le souci des applications, et en particulier l'exploitation des bases de données, a finalement focalisé l'intérêt du côté de l'apprentissage de motifs ou de régularités au sein de données. C'est donc principalement sur ce type d'apprentissage, le plus étudié et le mieux maîtrisé aujourd'hui, que porte ce chapitre (voir le chapitre I.9 pour ce qui est d'autres algorithmes portant davantage sur l'adaptation et l'apprentissage par renforcement, ainsi que les chapitres III.6 et III.7 pour des techniques d'apprentissage développées pour la bioinformatique ou la vision et la robotique).

Auteurs : ANTOINE CORNUÉJOLS, CHRISTEL VRAIN et JEAN-DANIEL ZUCKER.

Ces algorithmes d'apprentissage sont désormais au cœur des processus de fouille de données et leur compréhension est essentielle pour qui veut appréhender comment sont analysées les quantités de données [1] dont la progression est devenue exponentielle (par exemple les données post-génomiques ou bien toutes les données issues de nos téléphones mobiles).

Les premiers algorithmes d'apprentissage sont apparus en même temps que l'intelligence artificielle (IA) il y a plus de soixante ans. De même que pour l'histoire de l'IA, on peut dégager cinq grandes phases dans le développement de l'apprentissage artificiel. Très brièvement, ce sont les suivantes. La première est celle de l'enthousiasme des années 1950 et 1960 pendant lesquelles les premiers algorithmes d'apprentissage sont inspirés par la nature et la psychologie cognitive. Ainsi sont développés les prolégomènes de l'apprentissage par renforcement (celui du jeu de dames par Samuel en 1959) et de l'apprentissage neuro-mimétiques (celui du perceptron par Rosenblatt 1957 (Rosenblatt, 1958)) voient le jour et semblent promettre un succès rapide pour la réalisation d'apprentissage à partir de percepts (e.g. images de rétine sous forme matricielle, position au jeu de dames, etc.). Les années 1970 sont liées à un « âge de raison ». L'ouvrage de Minsky et Papert (1969) (Minsky et Papert, 1988) montre les limitations de l'algorithme du perceptron soulignant notamment le problème de l'apprentissage de représentations adéquates en amont d'un système adaptatif tel que le perceptron. Ce sont aussi les années d'euphorie en I.A. avec les systèmes experts et les premiers grands systèmes d'apprentissage symboliques, dont ARCH (Winston, 1970), l'apprentissage par espace des versions (Mitchell, 1979), ou encore ACT qui cherche à simuler l'apprentissage des enfants dans le domaine de l'algèbre par exemple (Anderson *et al.*, 1979; Anderson, 1995).

Suivent les années 1980 qui correspondent à une phase de renaissance et d'un nouvel engouement pour toutes les formes de l'apprentissage artificiel : algorithmes d'apprentissage de règles, d'arbre de décision, supervisé, non supervisé, par renforcement, par analogie, etc. De très nombreux « paradigmes » sont ainsi explorés : apprentissage par analogie, par découverte d'heuristiques, par généralisation dans un espace de versions, par analyse d'explications, par chunking, etc. (voir par exemple (Michalski *et al.*, 1986; Kodratoff et Michalski, 1990) et la traduction française (Michalski *et al.*, 1993)). La quatrième phase est celle de la maturité où, d'une part, les algorithmes existant sont mieux compris, mieux analysés et où, d'autre part, la théorie statistique de l'apprentissage fournit de nouveaux outils pour mieux comprendre la convergence des algorithmes. On commence alors à délaisser le modèle humain de l'apprentissage. La phase actuelle, des années 2000 à nos jours, se caractérise par un développement très rapide des techniques d'apprentissage artificiel qui irriguent tous les domaines de l'IA ainsi que de très nombreux secteurs industriels, commerciaux et financiers. D'un point de vue conceptuel, l'apprentissage est considéré comme le déroulement d'algorithmes d'optimisation heuristique ou exact pour lesquels les chercheurs ont maintenant le recul nécessaire pour bien en comprendre leurs complexité et l'origine de leur capacité d'apprentissage.

Que ce soit en vue de prédire ou de comprendre l'environnement, l'apprentissage a pour objectif de découvrir une fonction de décision ou un modèle du monde, celui-ci pouvant prendre la forme d'une théorie ou d'une distribution de probabilités par exemple. Or l'espace des fonctions de décision ou celui des modèles est immensément vaste. Dès lors l'inférence d'une fonction ou d'un modèle particulier à partir des données disponibles pose la double question de l'exploration de cet espace et de l'estimation de la qualité des fonctions ou modèles envi-

1. On parle aujourd'hui de plus en plus de base de données massives ou Big Data pour lesquels l'analyse ne peut se faire sans les algorithmes d'apprentissage et de fouille de données

sageables par cette exploration. Il faut donc, d'une part, définir *un critère inductif* permettant de jauger les fonctions ou modèles en fonction des observations et de toute autre connaissance préalable, et, d'autre part, trouver *un moyen de guider une recherche dans un espace de modèles* du monde. C'est ce double défi qui est au cœur de l'apprentissage et celui qui motive les algorithmes développés.
Les problèmes d'apprentissage étudiés aujourd'hui, les sources d'informations possibles et les tâches visées sont d'une grande variété, même si on est encore loin de pouvoir rendre compte de tous les types d'apprentissages naturels, comme, par exemple, le développement de l'intelligence chez l'enfant (Woolfolk, 2001). Nous avons essayé d'adopter dans ce chapitre un point de vue aussi générique que possible, nous concentrant sur les concepts fondamentaux et les grandes approches algorithmiques, elles-mêmes très liées aux types d'espaces de fonctions ou modèles considérés.

10.1.1 Recherche de régularités dans des données

L'apprentissage met en jeu des observations, notées $\mathbf{x}$, éventuellement associées à une étiquette y (dans ce cas on utilisera parfois aussi la notation $\mathbf{z} = (\mathbf{x}, y)$). Ces observations (et étiquettes) peuvent être de nature très diverses : données sur les clients d'une entreprise, données sur le génome de patients atteints ou non d'une certaine pathologie, descriptions de documents sur internet, etc. Elles peuvent donc être décrites soit sous forme vectorielle, ceci se fera de manière assez naturelle si les données sont stockées dans des bases de données relationnelles classiques, soit non vectorielle, comme c'est le cas pour la description de molécules ou de documents. De manière générale, on notera $\mathcal{X}$ l'espace des observations, et $\mathcal{X} \times \mathcal{Y}$ l'espace des observations et des étiquettes possible si celles-ci sont disponibles. Les données d'apprentissage, aussi appelées *échantillon d'apprentissage*, forment un ensemble (éventuellement un multi-ensemble si il y a répétition), noté $\mathcal{S}$, pris dans $\mathcal{X}^m$ ou dans $(\mathcal{X} \times \mathcal{Y})^m$. Par exemple, un échantillon de données étiquetées prendra la forme : $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$ si il y a m données d'apprentissage.
Le but de l'apprentissage est soit de permettre la prise de décision lorsqu'une nouvelle situation $\mathbf{x}$ arrive, ce qui revient souvent à savoir lui associer une étiquette, soit d'aider à la compréhension du monde par la mise à jour de régularités ou d'hypothèses permettant d'expliquer ou de rendre compte des observations connues. On notera $\mathcal{H}$ l'*espace des hypothèses* candidates.
L'apprentissage revient donc à associer à un échantillon de données $\mathcal{S}$ une ou plusieurs hypothèses les plus à même de nous rendre performant à l'avenir. Un *algorithme d'apprentissage* $\mathcal{A}$ peut ainsi être considéré comme une fonction (déterministe ou non) $\mathcal{A} : \mathcal{S} \mapsto h$ définie de $\mathcal{X}^m$ (ou $(\mathcal{X}^m \times \mathcal{Y})^m$) $\rightarrow \mathcal{H}$ associant à un échantillon d'apprentissage $\mathcal{S}$ de taille m une hypothèse h (ou la combinaison de plusieurs hypohèses).
Ces hypothèses ou régularités sont de nature très variée en fonction du problème posé. Ainsi, on peut chercher :

— *des nouveaux descripteurs*. Par exemple, la recherche d'axes principaux d'inertie (par analyse en composantes principales). En calculant les principaux axes d'inertie du nuage de points correspondant aux données supposées décrites dans un espace géométrisable, elle permettra en particulier de voir si les variables de description sont corrélées, ce qui peut conduire par exemple à une redescription des données selon de nouveaux axes.

— des *motifs fréquents*. Ce sont des motifs (conjonctions d'attributs-valeur) suffisamment représentés (à l'aune d'un seuil prédéfini appelé *support*) dans la table des données. Cela pourrait être par exemple (*âge* > *60*) $\wedge$ (*HDL-cholesterol* > *1,65 mmol/L*).
— des *règles d'association* : règles de la forme $I \rightarrow J$, où I et J sont des motifs et $J \neq \varnothing$. On veut bien sûr des règles « intéressantes », ce que l'on traduit par diverses mesures.
— des *catégories* dans les données. On parle aussi de *classification non supervisée* ou de *clustering* lorsque les catégories prennent la forme de nuages de points ou de structures particulières dans l'espace de description des formes. Selon les critères d'évaluation utilisés, les catégories trouvées peuvent être très variables. On pourra éventuellement être intéressé par une hiérarchie de catégories.
— des *règles de prédiction* (*Apprentissage supervisé*). Si l'une des colonnes de la table est considérée comme une étiquette (par exemple l'*activité chimique* dans la table de la figure 1), alors une tâche peut être d'apprendre à associer une étiquette à n'importe quelle observation (décrite par les autres attributs). Si les étiquettes correspondent à des exemples positifs et négatifs, on parle d'*apprentissage de concept* car on cherche alors à trouver une description caractérisant l'appartenance à ce concept. Si les étiquettes appartiennent à un ensemble fini de possibilités, on parlera de *classification*. Finalement, si les étiquettes sont prises dans $\mathbb{R}$, on parlera généralement de *régression*.
— un *tri des éléments de* $\mathcal{S}$, on parle alors de *ranking*. On peut ainsi vouloir fournir une liste triée de films à destination d'un internaute client d'un site de recommandation. L'apprentissage consiste alors à apprendre à trier correctement des ensembles d'éléments.

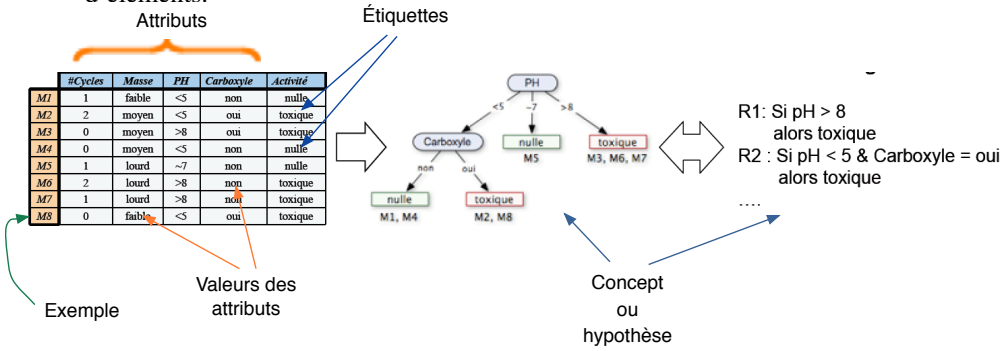

FIGURE 1: *Extraction de régularités, ici sous la forme de règles de prédiction, à partir d'une table de données.*

Les types de régularités recherchées sont donc variables, fonctions des buts de l'apprentissage, par exemple : prédiction ou compréhension.

L'apprentissage dit *supervisé* prend en entrée un échantillon d'observations avec leur étiquette associée : $\mathcal{S} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$ et cherche une dépendance entre observation et étiquette dans le but d'être capable de prédire l'étiquette d'une observation inconnue, et éventuellement d'aider l'expert à comprendre la nature de cette dépendance. Celle-ci peut prendre la forme d'une fonction ou d'une distribution de probabilité jointe $\mathbf{p}_{\mathcal{X}\mathcal{Y}}$.

De même que l'espace de description des observations prend des formes variées, depuis des espaces vectoriels jusqu'à des espaces d'objets structurés, l'*espace des étiquettes* peut être de nature diverse. Dans le cas de la *classification*, il s'agit d'un espace discret contenant un

nombre fini d'éléments interprétés comme étant des classes, par exemple {molécule bioactive, molécule non bioactive}. Dans le cas de la *régression*, il s'agit le plus souvent de l'ensemble des réels $\mathbb{R}$. Mais dans le cas d'*apprentissage multivarié*, l'espace de sortie peut être le produit cartésien de plusieurs espaces. Par exemple, on pourrait vouloir prédire à la fois l'âge et le revenu d'un individu décrit par un certain nombre de caractéristiques. Certains apprentissages visent même à prédire des structures, comme celle d'une molécule ou d'un réseau de régulation par exemple. On parle alors d'apprentissage avec sortie structurée.
Étant donnée sa place centrale à ce jour dans les applications et dans l'analyse théorique de l'apprentissage, nous accordons à l'apprentissage supervisé une attention particulière dans la suite de ce chapitre. Pour les lecteurs intéressés spécifiquement par l'apprentissage non supervisé, souvent assimilé au clustering, nous renvoyons par exemple à (Xu et Wunsch, 2008).

10.1.2 Le critère inductif

L'apprentissage consiste à construire une représentation du monde en adéquation avec les observations disponibles et permettant de prendre des décisions. Cette représentation peut prendre des formes diverses : hyperplan séparateur dans l'espace des observations $\mathcal{X}$ lorsque l'on veut distinguer deux classes, distribution de probabilité, automate à états finis, description logique, etc. Pour trouver cette représentation, il faut explorer un espace de possibilités généralement énorme, l'espace des hypothèses $\mathcal{H}$. Afin d'être capable de comparer les descriptions candidates entre elles, il faut pouvoir évaluer leur mérite. C'est le rôle du critère inductif.
Ce mérite dépend de la tâche considérée. Nous prendrons ici l'exemple de **l'apprentissage supervisé**. Dans ce cas, l'objectif est de trouver une fonction de décision de l'espace des observations $\mathcal{X}$ vers l'espace des étiquettes $\mathcal{Y}$ permettant les meilleures prédictions face aux environnements attendus. On traduit cela par une espérance de coût associée aux décisions prises dans le futur si l'on utilise l'hypothèse h. Les coûts sont définis ponctuellement pour chaque évènement $(\mathbf{x}, y)$ possible. Si l'hypothèse h est utilisée, alors la prédiction associée à l'observation $\mathbf{x}$ sera $h(\mathbf{x})$ tandis que la vraie réponse qu'il aurait fallu produire sera y. Cet écart se traduira par un coût noté $\ell(h(\mathbf{x}), y)$, calculé grâce à la *fonction de perte* $\ell(\cdot, \cdot)$. Cette fonction peut être symétrique, comme c'est le cas de la fonction qui prend la valeur 1 quand $h(\mathbf{x}) \neq y$ et 0 sinon, ou encore dans le cas d'un écart quadratique : $\ell(h(\mathbf{x}), y) = (h(\mathbf{x}) - y)^2$. Mais elle peut être non symétrique si un type d'erreur est plus important que l'autre, comme c'est, par exemple, souvent le cas en médecine. On cherche naturellement à minimiser l'espérance de perte en prenant en compte la densité de probabilité $\mathbf{p}_{\mathcal{X}\times\mathcal{Y}}$ des paires $(\mathbf{x}, y)$ qui caractérise l'environnement dans lequel évoluera le système une fois choisie l'hypothèse h.
Formellement, cela s'exprime par l'espérance de perte que l'on appelle le *risque réel* associé à la fonction de décision $h : \mathcal{X} \to \mathcal{Y}$:

$$R_{\text{Réel}}(h) \;=\; \int_{\mathcal{X}\times\mathcal{Y}} \ell(h(\mathbf{x}), y)\, \mathbf{p}_{\mathcal{X}\times\mathcal{Y}}\, d\mathbf{x}dy$$

Malheureusement, cette densité $\mathbf{p}_{\mathcal{X}\times\mathcal{Y}}$ est inconnue *a priori*, et la seule information disponible lors de l'apprentissage est l'échantillon $\mathcal{S} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$. Une hypothèse souvent faite est que cet échantillon résulte d'un tirage aléatoire suivant la distribution $\mathbf{p}_{\mathcal{X}\times\mathcal{Y}}$. Cela revient à dire que les exemples sont tirés indépendamment les uns des autres et que l'environnement est stationnaire. En adoptant ces présupposés, il devient alors tentant de remplacer le problème de la recherche d'une hypothèse qui minimise le risque réel, inconnu,

par celui de la recherche d'une hypothèse qui minimise la perte moyenne sur l'échantillon d'apprentissage, encore appelé *risque empirique* :

$$R_{\text{Emp}}(h) \;=\; \frac{1}{m}\sum_{i=1}^{m} \ell(h(\mathbf{x}_i), y_i)$$

On remplace alors le critère du risque réel par un critère s'appuyant sur l'échantillon d'apprentissage. C'est ce que l'on appelle le *critère inductif*. Supposer que le risque empirique est indicatif du risque réel, celui qui nous importe, c'est prendre un pari sur le monde. Caractériser et quantifier ce pari a été l'objet principal des nombreuses recherches des dernières décennies dans le cadre de la théorie de l'induction et de la théorie statistique de l'apprentissage. Cette dernière prend fondamentalement appui sur des théorèmes de type centrale limite, qui nécessitent l'hypothèse de données indépendantes et identiquement distribuées.
L'analyse, par une approche statistique, de la pertinence de la minimisation du risque empirique, notablement par Vapnik et Chervonenkis (Vapnik, 1995) dans les années soixante-dix puis quatre-vingt, a ainsi montré qu'il fallait compléter le risque empirique par un terme pénalisant la sélection d'hypothèses h complexes. En effet, il est facile sinon, à l'aide d'hypothèses arbitrairement complexes, de trouver une hypothèse de risque empirique très faible, voire nul, c'est-à-dire une hypothèse se comportant très bien sur l'échantillon d'apprentissage, mais dont le risque réel est sans rapport, ce que l'on appelle « sur-apprentissage » ou sur-adaptation. Dans ce cas, l'apprentissage aura identifié des coïncidences accidentelles dans l'échantillon d'apprentissage au dépend des régularités « vraies » du monde.
Il faut donc optimiser un compromis entre envisager un espace d'hypothèses assez riche pour pouvoir y trouver de bonnes hypothèses, et le prendre cependant assez restreint pour que le risque empirique, mesuré sur l'échantillon d'apprentissage, soit représentatif du risque réel sur les données non encore observées. L'optimisation de ce compromis peut s'opérer par plusieurs approches.

1. La *régularisation*. Elle consiste à pénaliser les hypothèses trop complexes, cette complexité étant par exemple mesurée par la régularité des fonctions de base (e.g. le degré du polynôme utilisé), et/ou par le nombre de fonctions de base impliquées (voir section 10.2.1). La méthode de *minimisation du risque structurel* proposée par Vapnik est liée à cette approche. Mais au lieu de pénaliser l'hypothèse considérée, elle pénalise la richesse de l'espace d'hypothèses dans lequel s'opère la recherche de la meilleure hypothèse. Les techniques d'élagage consistant à réduire la complexité d'une hypothèse après apprentissage entrent dans ce cadre.
2. Le *contrôle du processus d'optimisation*. L'idée est de limiter l'exploration de l'espace des hypothèses en interrompant le processus de recherche pour empêcher d'arriver à des hypothèses apparemment très bonnes mais qui au lieu de capturer seulement les régularités sous-jacentes des données, détectent aussi leurs particularités accidentelles. La *règle d'arrêt prématuré* utilisée pour les perceptrons multicouches, et qui arrête le processus d'optimisation avant sa convergence, en est un exemple.

L'étude du critère inductif dans le cas de l'apprentissage supervisé à partir de données supposées tirées indépendamment selon une distribution stationnaire est la plus avancée actuellement car elle se prête bien aux analyses de type statistique.

En revanche, l'étude formelle de l'apprentissage non supervisé ou d'autres types d'apprentissages, par exemple lorsque le monde évolue et que les données ne sont donc plus indépendantes, n'a pas la même maturité et reste encore exploratoire.

10.1.3 Le choix de l'espace des hypothèses

L'apprentissage inductif consiste à chercher une hypothèse capturant les régularités profondes du monde environnant et permettant donc des prédictions fiables. Une étape essentielle dans ce processus consiste à fixer l'espace des hypothèses $\mathcal{H}$ que l'on est prêt à considérer. C'est ce que l'on appelle la *sélection de modèles*, par référence au vocabulaire employé en statistiques. Pour cela, on s'appuie sur des connaissances préalables plus ou moins précises, des connaissances expertes, ainsi que sur des considérations liées à la facilité d'exploration de l'espace. Ainsi, par exemple, un espace d'hypothèses linéaires est très limitatif car son pouvoir expressif est réduit, mais il permet la mise en jeu de méthodes d'optimisation convexes donc efficaces. L'espace des hypothèses peut également prendre la forme d'une famille de distributions de probabilité, l'apprentissage consistant alors à estimer la valeur des paramètres fixant une distribution donnée. En *apprentissage symbolique*, l'espace des hypothèses sera défini par un langage de description (e.g. des clauses de Horn, des règles de décision, des arbres de décision, etc.) associé éventuellement à un biais de préférence pour des hypothèses « simples ».

Il existe une distinction importante entre les *modèles génératifs* et les modèles qui ne le sont pas et que l'on qualifie fréquemment de *modèles discriminatifs*. (Un ouvrage de facture très théorique focalisé sur cette distinction est (Jebara, 2003)).

1. L'apprentissage d'un modèle *génératif* vise à estimer une distribution de probabilité $\mathbf{p}_{\mathcal{X}\mathcal{Y}}$ sur l'espace conjoint des observations et des étiquettes $\mathcal{X} \times \mathcal{Y}$. Cette distribution peut être décomposée en $\mathbf{p}_{\mathcal{X}}\,\mathbf{p}_{\mathcal{Y}|\mathcal{X}}$ ou en $\mathbf{p}_{\mathcal{Y}}\,\mathbf{p}_{\mathcal{X}|\mathcal{Y}}$. Dans les deux cas, une fois les distributions apprises, il est possible de les utiliser pour générer un ensemble de points $\mathbf{z} = (\mathbf{x}, y)$ qui, si l'estimation est correcte, doit être statistiquement indistinguable de l'échantillon d'apprentissage. C'est la raison pour laquelle on parle d'apprentissage génératif. Les algorithmes utilisés ressortent des techniques statistiques d'estimation de modèles à partir de critères tels que le *maximum de vraisemblance* ou de *maximum a posteriori*. L'algorithme d'Expectation-Maximization (EM) est un exemple typique de ces algorithmes (on peut en trouver une description dans des ouvrages généraux comme (Barber, 2012; Bishop, 2006; Hastie *et al.*, 2009; Koller et Friedman, 2009)).
2. L'apprentissage qualifié de *discriminatif* se donne comme objectif d'associer la bonne étiquette y à toute observation $\mathbf{x}$. Pour ce faire, il n'est pas nécessaire d'estimer les distributions de probabilité sous-jacentes aux données. On peut par exemple déterminer une frontière de décision entre classes d'observations. Si cet apprentissage peut sembler moins satisfaisant puisqu'il cherche une information plus limitée sur le monde, c'est aussi ce qui fonde son avantage dans bien des situations car il demande moins de données et moins d'*a priori* sur le monde sous la forme de distributions de probabilité paramétriques qu'il faut fournir au système d'apprentissage.

Il est évident que la définition de l'espace des hypothèses contrôle l'adéquation possible au vrai modèle sous-jacent du monde, elle dicte aussi le type d'algorithmes mis en jeu. Dans la suite, nous étudions trois grandes familles d'espaces d'hypothèses correspondant à trois grandes

approches algorithmiques. Deux reposent principalement sur des techniques d'optimisation dans des espaces de paramètres réels alors que le troisième met en œuvre des stratégies de recherche dans un espace structuré. Cette perspective orientée sur l'approche algorithmique nous a conduit à une organisation non classique des paradigmes d'apprentissage.

10.2 L'apprentissage selon les espaces d'hypothèses : par dictionnaire ou à partir d'exemples

Cette section regroupe deux familles d'apprentissage fondées sur des techniques d'optimisation mais qui diffèrent dans les modèles sous-jacents mis en œuvre.

1. La première famille d'apprentissage cherche à modéliser les régularités du monde à partir de *combinaisons de fonctions de base*. Dans ce cadre, une hypothèse s'exprime par une telle combinaison, et ne fait pas intervenir dans son expression les exemples d'apprentissage.
2. La seconde famille, par contraste, exprime les hypothèses en *se référant explicitement aux exemples d'apprentissage*, ou à un sous-ensemble de ceux-ci, et utilise des fonctions de similarité avec pondération éventuelle pour calculer la réponse associée à une observation.

10.2.1 Modèles définis à l'aide d'un dictionnaire de régularités

La première famille d'espaces d'hypothèses (ou famille de modèles dans le cadre statistique) concerne les hypothèses décrites comme combinaisons d'un ensemble de fonctions de base. Typiquement, il s'agit de familles de distributions de probabilité, par exemple un mélange de n gaussiennes, ou bien de combinaisons linéaires de fonctions de base :

$$h(\mathbf{x}) \;=\; \sum_{i=1}^{n} w_i \, g_i(\mathbf{x}) + w_0$$

où les $g_i(\cdot)$ sont les fonctions de base et les w_i les coefficients ou paramètres.
Parce que ces méthodes utilisent un ensemble de fonctions de base souvent *données a priori*, à l'instar des séries de Fourier, on les appelle fréquemment des méthodes à base de dictionnaire. Le nombre de fonctions de base utilisées peut servir à contrôler la capacité de l'espace d'hypothèses et donc à régulariser le risque empirique.
Ces fonctions de base $g_i(\cdot)$ permettent en un sens la redescription des entrées, et les paramètres w_i servent alors à sélectionner ou à pondérer l'importance des éléments de cette redescription. Contrairement à l'analyse de Fourier et à l'analyse fonctionnelle en général, les fonctions de base utilisées dans ces méthodes ne sont pas nécessairement orthogonales. Cela peut alors poser des problèmes d'unicité des solutions, et de stabilité, les coefficients w_i pouvant varier fortement lorsque l'échantillon d'apprentissage est légèrement modifié.
Une grande partie des méthodes d'apprentissage concerne cette famille d'espaces d'hypothèses qui combinent un ensemble de fonctions de base prédéfinies. L'enjeu est alors d'apprendre les valeurs des paramètres de combinaison w_i, voire, parfois, des paramètres θ_j des fonctions de

base elles-mêmes si elles sont variables : $g_j(\mathbf{x}, \theta_j)$. Sans être exhaustif, on peut citer les espaces d'hypothèses suivant :

— Les **modèles additifs** correspondent aux combinaisons les plus simples de fonctions de base : des sommes pondérées ce celles-ci. Dans la *régression linéaire*, les fonctions de base sont elles-mêmes juste les coordonnées des entrées. Les *classifieurs linéaires*, tels le perceptron, sont de ce type avec une fonction signe pour décider si l'observation est positive ou non : $h(\mathbf{x}) = \text{signe}(\sum_{i=1}^{n} w_i\, g_i(\mathbf{x}) + w_0)$. Les fonctions $g_i(\mathbf{x})$ correspondent alors à l'extraction de descripteurs ou de formes dans les entrées. Les fonctions de base peuvent aussi prendre la forme de fonctions prises dans des familles plus complexes, par exemple les *splines*. De fait, si l'on dispose d'un dictionnaire de fonctions de base adapté, il est possible d'approcher n'importe quel signal ou n'importe quelle fonction cible (Hastie *et al.*, 2009), la question étant évidemment de trouver ce dictionnaire. L'apprentissage par *boosting*, qui, dans sa version de base, apprend une combinaison linéaire de « classifieurs faibles » (faibles au sens où chaque classifieur $h_t(\cdot)$ de la combinaison peut n'être qu'à peine meilleur qu'un classifieur opérant par tirage aléatoire de la classe à prédire) : $h_T(\mathbf{x}) = \text{signe}\{\sum_{t=1}^{T} w_t\, h_t(\mathbf{x})\}$ est un bel exemple de la puissance de telles méthodes. Ici, chaque fonction de base est alors adaptative et apprise en même temps que les coefficients w_t de la combinaison (Schapire, 2003; Shapire et Freund, 2012; Zou, 2012).

— Les **modèles de mélanges**, qui font partie des approches génératives de l'apprentissage, servent à estimer la densité de probabilité $\mathbf{p}_{\mathcal{X}}$ par une formule du type : $\mathbf{p}(\mathbf{x}) = \sum_{i=1}^{n} \pi_i\, \mathbf{p}_i(\mathbf{x}|\theta_i)$. La densité de probabilité générale sur $\mathcal{X}$ est ainsi décomposée en une somme pondérée de fonctions de densité. Chacune des fonctions de base $\mathbf{p}_i(\mathbf{x}|\theta_i)$ consiste typiquement en une fonction paramétrique simple (de paramètre θ_i) comme une densité normale par exemple. Le coefficient π_i représente la probabilité qu'un point tiré au hasard ait été engendré à partir de la densité i, que l'on interprète parfois comme une classe de formes « expliquant » les observations.

— Les **perceptrons multicouche** combinent des fonctions de base (typiquement des sigmoïdes), réalisées par les « neurones », dans une structure hiérarchique fixée par l'utilisateur entre une couche d'entrée et la couche de sortie via des couches cachées (voir figure 2). L'apprentissage consiste dans ce cadre à apprendre les poids des connexions entre les neurones de couches consécutives, soit encore les coefficients v_k pris en compte dans la fonction d'activation (souvent une sigmoïde) :

$$g(\mathbf{x}) = s\left(\sum_{j=1}^{d} v_k x_k + v_0\right) = s(\mathbf{v}\cdot\mathbf{x})$$

où s est la fonction d'activation (e.g. une sigmoïde : $s(a) = \frac{1}{1+\exp(-a)}$) (Une référence assez complète sur ces méthodes neuronales est (Dreyfus *et al.*, 2008)).

— Les **modèles graphiques à structure fixée**, dont les réseaux bayésiens ou les modèles de Markov font partie, sont des représentations de dépendances conditionnelles entre variables. Lorsque la structure de ces dépendances est fixée (ce qui peut faire l'objet d'un apprentissage, voir section 10.3), l'apprentissage consiste à apprendre les probabilités conditionnelles en chaque nœud du réseau, ce qui revient à apprendre les paramètres des fonctions de base. (Voir (Koller et Friedman, 2009), et (Pearl, 1988) pour l'ouvrage pionnier dans ce domaine).

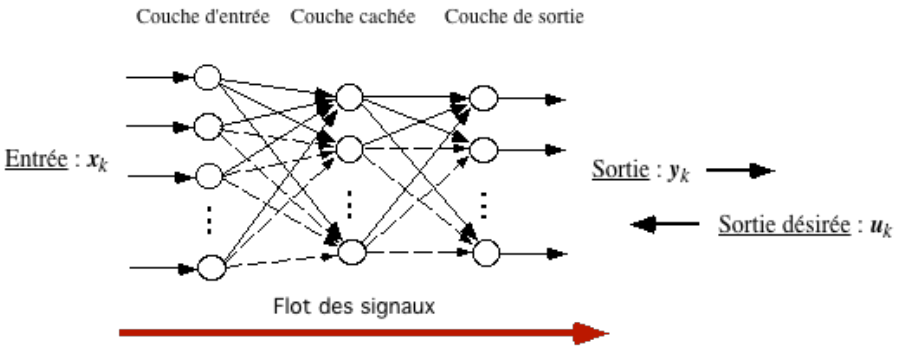

FIGURE 2: *Schéma d'un perceptron multicouche.*

Le premier problème à résoudre dans l'emploi de ces méthodes concerne donc le choix du « dictionnaire » de fonctions de base. Le deuxième problème est celui de l'estimation des paramètres w_i. Plusieurs approches existent pour le résoudre. Elles dépendent du type de fonction de perte utilisée dans la définition du risque empirique et de l'ensemble des fonctions de base. Trois grandes familles d'approches peuvent être distinguées :

1. *Méthodes à base de gradient* (Snyman, 2005). Ces méthodes partent d'une estimation initiale du vecteur de paramètres $\mathbf{w} = (w_0, w_1, \ldots, w_n)^\top$ puis l'adaptent par une méthode de gradient. Lorsque celui-ci est calculé sur le risque empirique (régularisé), on parle de gradient total. Lorsqu'il est calculé après chaque présentation d'une donnée $\mathbf{z}_t$, on parle de gradient stochastique. De nombreuses variantes existent. Le célèbre algorithme de retro-propagation de gradient est ainsi une procédure permettant de calculer le gradient de l'erreur en chaque neurone et chaque connexion d'un perceptron multicouche.
2. *Méthodes itératives* (Kelley, 1987). Les méthodes itératives cherchent également à diminuer la mesure de risque de manière itérative. Cependant, elles ne s'appuient pas sur une estimation de gradient, mais utilisent des fonctions d'approximation ou des fonctions de perte spéciales permettant d'assurer qu'un procédé itératif diminue finalement le risque. Les méthodes du type expectation-maximization (EM), utilisées dans le cas de données non supervisées, sont un exemple d'une telle approche.
3. *Optimisation gloutonne* (Cormen *et al.*, 2001). Les méthodes d'optimisation gloutonnes décomposent le problème d'optimisation en sous-problèmes plus simples. Dans le cas des hypothèses définies par combinaison de fonctions de base, ce genre de méthode consiste à optimiser l'un des termes de la combinaison (e.g. l'un des termes de la combinaison linéaire), puis à le fixer et optimiser un autre terme jusqu'à ce que tous les termes soient optimisés. Un bon optimum n'étant pas nécessairement atteint après une seule passe, on peut éventuellement répéter plusieurs fois ce cycle d'optimisation. Le *boosting* qui opère par une sorte de descente de gradient conjugué est un exemple de cette famille d'approches.

En dehors de la détermination de la méthode d'optimisation la plus adaptée au problème, le principal problème de ces méthodes à partir de dictionnaire est de déterminer le bon diction-

naire de fonctions de base, c'est-à-dire, *in fine*, le bon espace $\mathcal{H}$ de fonctions hypothèses. C'est encore une fois le problème de la sélection de modèle dont l'objectif est en particulier de contrôler le risque de sur-apprentissage c'est-à-dire de sur-adaptation aux données au détriment de l'estimation des « vraies » régularités du monde (voir la fin de la section 10.1.2).

Il faut noter la *place particulière qu'occupent les modèles linéaires* dans les modèles à partir de dictionnaire de fonctions de base. Si ils peuvent paraître limités dans leur pouvoir expressif, ils présentent cependant de nombreux avantages qui les font toujours apprécier à la fois des théoriciens et des praticiens.
D'abord, leur pouvoir expressif dépend en fait du pouvoir expressif des fonctions de base. Si celles-ci sont adaptées au problème, ces modèles peuvent approcher n'importe quelle régularité cible. Surtout, si les fonctions de base sont suffisamment orthogonales entre elles, les coefficients w_i attachés à chaque fonction de base permettent d'en évaluer l'importance dans le signal étudié. Ces modèles se prêtent ainsi plus aisément à une interprétation par l'utilisateur ou l'expert que des modèles plus complexes, non linéaires par exemple. Finalement, il est possible à la fois d'améliorer encore cette interprétabilité et de contrôler la richesse de l'espace des hypothèses, si cruciale pour contrôler le risque de sur-apprentissage, en cherchant à diminuer le nombre de coefficients w_i qui sont non nuls. L'idée est alors de pénaliser les hypothèses en fonction du nombre de fonctions de base mises en jeu. Ces approches sont actuellement très étudiées (voir, par exemple, les méthodes LASSO (Hastie *et al.*, 2009)).
Cependant, contraindre l'expression du modèle du monde à être linéaire peut aussi conduire à un modèle artificiel qui ne rend pas compte de la structure des dépendances entre propriétés du monde. L'engouement actuel pour ces modèles est donc à tempérer comme le font certaines études critiques (voir (Bengio et Cun, 2007)).

10.2.2 Modèles définis à partir des exemples d'apprentissage

Si apprendre c'est souvent généraliser pour mieux oublier les exemples et les représenter sous la forme d'un modèle, on peut aussi construire des modèles qui conservent explicitement les exemples d'apprentissage. C'est le cas de la méthode élémentaire qui consiste à étiqueter une nouvelle observation en fonction de l'étiquette de ses plus proches voisins. C'est plus généralement le principe mis en œuvre dans le raisonnement à partir de cas (cf. Chapitre sur le CBR : I.7) où les cas sont mémorisés puis réutilisés, et éventuellement adaptés, selon les besoins. Il est bien connu que ces approches ne sont efficaces qu'à proportion qu'un échantillon suffisant et suffisamment représentatif de cas soit connu et que la fonction de distance, ou plus généralement de similarité, soit adéquate. C'est justement l'objet de l'apprentissage d'essayer d'obtenir ces conditions à partir d'un échantillon d'exemples pour apprendre une bonne fonction de décision.
Si l'on se place dans le cadre des modèles linéaires, les méthodes à base d'exemples par voisinage utilisent une représentation des fonctions de décision de la forme :

$$h(\mathbf{x}) \;=\; \sum_{i=1}^{n} \alpha_i \, \kappa(\mathbf{x}, \mathbf{x}_i) \, y_i$$

La différence avec les méthodes à base de dictionnaires est évidente. On voit en effet que la fonction hypothèse est définie ici directement à l'aide des points d'apprentissage $\mathbf{x}_i$, d'une part,

par l'intervention de la fonction de similarité $\kappa(\mathbf{x}, \mathbf{x}_i)$, et, d'autre part, par les étiquettes des points d'apprentissage y_i. *L'hypothèse est maintenant une combinaison linéaire des étiquettes des points d'apprentissage*, pondérées par les similarités, là où elle était précédemment une combinaison linéaire de fonctions de base.
L'apprentissage consiste donc désormais à sélectionner les bons exemples de référence $\mathbf{x}_i$. Mais avant cela, il faut se préoccuper du choix de la fonction de similarité $\kappa(\mathbf{x}, \mathbf{x}')$.

Mesures de similarité et exemples de référence
Un problème essentiel des méthodes à base d'exemples est celui du choix de la mesure de similarité, voire de distance, appropriée. Ainsi, tout joueur d'échec sait très bien que deux situations de jeu exactement identiques à l'exception de la position d'un pion, peuvent en fait être totalement différentes pour l'issue du jeu. Ce qui est vrai pour le jeu d'échec l'est également lorsque l'on compare des objets variés comme des molécules, des textes, des images, etc. À chaque fois, le choix de la bonne mesure de similarité est crucial pour qu'une méthode basée sur des similarités à des exemples connus soit performante.
En dehors même de la bonne adéquation de la fonction de similarité considérée à la sémantique du domaine se pose le problème de sa définition formelle. Si les mathématiques nous fournissent de nombreuses mesures adaptées aux données vectorielles numériques, par exemple sous la forme de distances, le problème est bien plus ouvert lorsque les données font intervenir des descripteurs symboliques et/ou sont définies dans des espaces non vectoriels, par exemple des textes. C'est pourquoi une partie importante des contributions actuelles en apprentissage artificiel concernent la définition et le test de nouvelles mesures de similarité appropriées pour des types de données particuliers : séquences, fichiers en format XML, données structurées, etc.
Une *distance* est une fonction symétrique de deux arguments, nulle seulement lorsque les deux arguments sont égaux et obéissant à l'inégalité triangulaire. Lorsque l'une des propriétés n'est pas vérifiée, on parle souvent plus librement de *dissimilarité*. Un exemple de distance est celui de la norme euclidienne entre deux vecteurs $||\mathbf{x} - \mathbf{y}|| = \sqrt{\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle}$ dans lequel $\langle \mathbf{x}, \mathbf{x}' \rangle$ est le produit scalaire entre les vecteurs $\mathbf{x}$ et $\mathbf{x}'$. Ce produit scalaire peut servir de base à des mesures de similarité comme le cosinus entre deux vecteurs. Plus les vecteurs sont « alignés », plus ils sont proches selon cette similarité.

Il se trouve que si l'on cherche un séparateur linéaire entre deux nuages de points qui maximise la marge entre ces nuages (en supposant qu'un tel séparateur existe), on peut exprimer ce séparateur comme une fonction de points particuliers $\mathbf{x}_i$ dans l'échantillon d'apprentissage appelés par Vapnik *vecteurs de support* :

$$h(\mathbf{x}) \; = \; \sum_{i=1}^{n} \alpha_i \, \langle \mathbf{x}, \mathbf{x}_i \rangle \, y_i$$

Il s'agit là d'une découverte importante car elle signifie que la solution qui optimise le risque empirique régularisé (au sens où on cherche une marge maximale et non n'importe quel séparateur linéaire) prend la forme d'une combinaison linéaire définie à partir de certains exemples d'apprentissage.
Cependant, cette observation, si intéressante soit-elle, n'aurait pas eu un grand impact si il n'avait pas été réalisé, encore une fois par Vapnik et ses collègues, que l'on pouvait généraliser ce type de solution à des problèmes de séparation non linéaire en remplaçant le produit scalaire

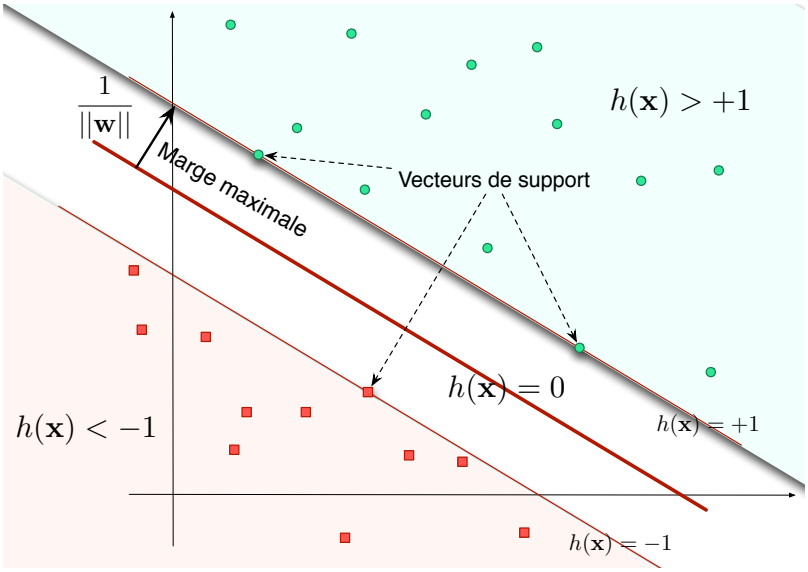

FIGURE 3: *L'hyperplan optimal maximise la distance aux points d'apprentissage des deux classes. Cette distance est entièrement déterminée par les vecteurs de support.*

$\langle \mathbf{x}, \mathbf{x}' \rangle$ par une fonction vérifiant certaines propriétés particulières que l'on appelle fonction noyau : $\kappa(\mathbf{x}, \mathbf{x}')$.

On peut alors montrer qu'un problème de séparation *non linéaire* entre deux classes d'objets dans l'espace d'entrée $\mathcal{X}$ peut s'exprimer sous la f*orme d'une combinaison linéaire* : $h(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \, \kappa(\mathbf{x}, \mathbf{x}_i) \, y_i$ où $\kappa(\mathbf{x}, \mathbf{x}')$ est une fonction noyau et les $\mathbf{x}_i$ sont des exemples d'apprentissage particuliers permettant de définir la séparatrice entre les nuages de points.

L'algorithme des *Séparateurs à Vastes Marges* (SVM ou « Support Vector Machines » en anglais) trouve les points supports et les coefficients α_i optimaux. Ils sont solution d'un problème d'optimisation quadratique, c'est-à-dire d'un problème convexe admettant un optimum unique, ce qui est un énorme avantage par rapport à d'autres méthodes, comme par exemple les perceptrons multicouches. Sans entrer dans les détails, la complexité de cet algorithme est dominée par le nombre d'exemples d'apprentissage, et non par la dimension de l'espace considéré, comme c'est généralement le cas pour les autres approches. Afin de pouvoir traiter des données en grand nombre et des données complexes, on cherche à rendre plus efficace le processus d'optimisation, par exemple en remplaçant le problème d'optimisation par un autre, plus facile à calculer, mais de même solution.

Cependant, la qualité de la fonction apprise repose fondamentalement sur le choix de la fonction noyau utilisée. Sans surprise, on peut montrer ici encore que ce choix est profondément lié au problème de la sélection de modèle et donc au contrôle du sur-apprentissage. De nombreux travaux de recherche portent sur la sélection automatique de fonctions noyau. Cependant, celle-ci est encore largement du ressort de l'expert, particulièrement lorsque les données à comparer sont complexes, comme des molécules, des documents ou des objets structurés en général.

Il est intéressant de noter que le développement des Séparateurs à Vastes Marges, initié pour

le problème de la classification en deux classes, a conduit à la réalisation que de nombreuses autres méthodes linéaires, telles l'analyse en composantes principales ou les filtres de Kalman, pouvaient aisément se généraliser au cas non linéaire. En fait, à chaque fois qu'une méthode conduit à des solutions mettant en jeu des produits scalaires $\langle \mathbf{x}, \mathbf{x}' \rangle$, elle peut se généraliser au cas non linéaire en remplaçant ce produit scalaire par une fonction noyau $\kappa(\mathbf{x}, \mathbf{x}')$. Cette observation a stimulé un vaste mouvement de réutilisation et de développement de nombreuses méthodes linéaires dans le cadre non linéaire grâce à ce que l'on appelle les *méthodes à noyaux* (voir (Schölkhopf et Smola, 2002; Shawe-Taylor et Cristianini, 2004)).

Points de référence
Nous venons de voir que, dans le cas de la classification, et plus précisément de la recherche d'une frontière entre classes, les points supports jouaient un rôle capital. Il se trouve que ces points supports, calculés dans les algorithmes de type SVM, sont les plus proches de la frontière cherchée. Ce n'est pas surprenant car, en un certain sens, ce sont eux qui contraignent cette frontière et permettent de déterminer la « marge » entre les nuages de points. Il serait cependant possible d'imaginer que d'autres points jouent un rôle essentiel. Ainsi, dans le cas de l'apprentissage non supervisé, et spécialement du clustering, les points de référence sont généralement au contraire des points qui représentent le « centre de gravité » des classes. Ces points, correspondant à des points d'apprentissage ou bien calculés, jouent alors le rôle de prototypes.

10.3 Espace des hypothèses : modèles à structure variable

Dans le cas des méthodes à base de dictionnaire, le choix du bon espace d'hypothèses, c'est-à-dire du bon dictionnaire se traduit par le problème dit de la sélection de modèles. Cette sélection peut se faire à la main, par l'expert, ou par des méthodes plus systématiques, mais l'espace des hypothèses $\mathcal{H}$ est donné *a priori* : celui de toutes les fonctions de base potentielles, de la même manière que l'on peut chercher la meilleure base de fonctions dans une analyse en ondelettes. Les méthodes à base d'exemples introduisent une souplesse supplémentaire dans la mesure où l'expression des hypothèses candidates dépend désormais directement des exemples d'apprentissage. L'adaptation au problème se fait ainsi plus naturellement, au prix cependant du bon choix de la fonction de similarité ou de la fonction noyau permettant la comparaison entre les exemples. Dans tous les cas, l'apprentissage consiste essentiellement à explorer un espace d'hypothèses que l'on peut qualifier de paramétré. D'où des techniques spécifiques de recherche. De plus, ces deux familles de méthodes sont souvent utilisées dans le cadre de modèles linéaires, ce qui autorise des algorithmes efficaces mais limite aussi la possibilité de rendre compte de régularités complexes, du moins de manière intelligible.
Il est heureusement possible d'aller plus loin dans l'adaptation de l'espace des hypothèses au problème d'apprentissage étudié. C'est ce que permettent les modèles à structure variable. Dans ces modèles, la forme des hypothèses n'est plus fixée *a priori* comme, par exemple, une combinaison linéaire de fonctions ou de sorties pondérées par des ressemblances aux points d'apprentissage, mais elle est variable. Il peut exister des *hiérarchies de descripteurs* ou de *concepts intermédiaires* ou encore des expressions relationnelles arbitrairement complexes. C'est le cas des réseaux bayésiens ou modèles graphiques, des modèles par arbre de décisions, ou encore celui des modèles à base de règles.

Dans ces approches, l'apprentissage comporte naturellement deux aspects. D'une part, il doit permettre de *découvrir une bonne structure* pour représenter les hypothèses, d'autre part, il doit *estimer les valeurs optimales des paramètres* éventuels.

La recherche dans ce cas de la meilleure approximation des régularités cibles inconnues est encore plus difficile et peut paraître désespérée. Le nombre de structures possibles est en effet généralement gigantesque, chacune possédant ses propres paramètres. Comme on ne dispose plus dans cet espace de propriétés de continuité ou d'analycité comme dans l'espace des modèles paramétrés, une exploration bien guidée paraît impossible. De surcroît, toujours afin de contrôler le risque de sur-apprentissage, il faut pouvoir estimer la qualité des espaces d'hypothèses considérés dans le processus constructif et cela à chaque pas. Les difficultés sont donc magnifiées par rapport aux apprentissages décrits précédemment.

Il n'existe pas actuellement de méthodes génériques s'appliquant à tous les espaces de structures. Chaque famille d'espace a ses spécificités et des approches particulières qui font encore souvent l'objet de recherches. Ainsi, dans les réseaux bayésiens, il existe des approches fondées sur les contraintes tirant profit du test d'indépendance entre variables. D'autres approches sont fondées sur des fonctions de score prenant en compte à la fois l'adéquation aux données, mais aussi une mesure de complexité du modèle candidat et des techniques d'exploration de graphe très variées. Nous allons voir cependant que dans certains cas, essentiellement l'apprentissage de concept utilisant un langage d'expression symbolique des hypothèses, il est possible d'exploiter certaines structures dans l'espace des hypothèses qui rendent l'apprentissage efficace.

Avant d'aller plus loin et devant la diversité des modèles, précisons quelques notions de base sur les différents langages de représentation utilisés dans la suite de cette section. Considérons d'abord l'apprentissage de règles.

Une règle est constituée d'une prémisse spécifiant les conditions d'application de la règle et d'une conclusion spécifiant ce qui est vrai lorsque la prémisse est réalisée. Plusieurs langages peuvent être utilisés pour exprimer les règles : la logique des propositions, composée uniquement de symboles de proposition, comme par exemple dans la règle `rouge` $\wedge$ `bruyant` $\wedge$ `cher` $\rightarrow$ `voiture_bling_bling`, la représentation attribut-valeur comme dans la règle `température` $\geq$ `37,5` $\rightarrow$ `fièvre` ou encore la logique du 1er ordre plus riche et permettant d'exprimer des relations plus complexes entre les objets, comme par exemple `père(X,Y), père(Y,Z)` $\rightarrow$ `grand_père(X,Z)`. Parfois, lorsque le concept à apprendre est implicite (par exemple, apprentissage du concept `voiture_bling_bling`), on omet la conclusion de la règle et on s'intéresse uniquement à la conjonction de propriétés définissant le concept. Notons enfin que des expressions plus complexes peuvent aussi être recherchées, comme des clauses générales permettant de spécifier des alternatives dans la conclusion ou d'introduire de la négation. L'apprentissage de connaissances exprimées en logique du 1er ordre est le domaine d'étude de la *Programmation Logique Inductive* (Raedt, 2008; Dzeroski et Lavrac, 2001), qui a connu un grand essor dans les années 80. Une autre famille concerne les *modèles graphiques* (Koller et Friedman, 2009) : ce sont des graphes dont les nœuds sont les variables et permettant de représenter des dépendances entre variables. On distingue principalement les réseaux bayésiens, graphes orientés associant à chaque nœud la probabilité conditionnelle de ce nœud, étant donnés ses parents et les modèles de Markov, graphes non orientés pour lesquelles une variable est indépendante des autres, étant donnés ses voisins. Un courant actuel, dénommé *Apprentissage Relationnel Statistique* (Raedt *et al.*, 2008; Getoor et Taskar,

2007) tend à coupler la Programmation Logique Inductive et les approches probabilistes. Enfin, l'*inférence grammaticale* ((de la Higuera, 2010), (Cornuéjols et Miclet, 2010) ch.7, ou encore (Miclet, 1990)) s'intéresse à l'apprentissage de grammaires ou de langages à partir de données ; la famille de modèles considérée est souvent celle des automates, éventuellement probabilistes. Nous supposons connues ici les notions d'automates et de grammaires.
Notons que souvent on ne cherche pas une hypothèse mais un ensemble, c'est-à-dire une disjonction d'hypothèses. Considérons ainsi l'apprentissage d'un concept à partir d'exemples positifs et négatifs. Il peut être irréaliste de chercher une unique règle couvrant les exemples positifs et rejetant les négatifs, puisque cela supposerait que tous les exemples positifs suivent le même modèle : on recherche alors plutôt un ensemble de règles. Cependant, étendre le modèle à un ensemble d'hypothèses introduit à nouveau le compromis nécessaire au sein du critère inductif touchant à la complexité de l'espace des hypothèses. En effet, la disjonction des exemples positifs, $\mathbf{x}_1 \vee \ldots \vee \mathbf{x}_m$, représente un ensemble d'hypothèses qui couvre tous les exemples positifs et rejette tous les négatifs, dès lors que ces derniers diffèrent des positifs. On a alors une hypothèse arbitrairement complexe, variant au gré des exemples d'apprentissage, et d'erreur empirique nulle. Néanmoins, cet apprentissage par cœur ne correspond à aucune généralisation. Pour pallier ce problème, il faut introduire ici aussi des contraintes sur les hypothèses, c'est-à-dire faire de la régularisation comme évoqué en section 10.1.2.

10.3.1 Espace des hypothèses - relation de généralité et opérateurs

Comment, lorsque l'on ne considère plus un espace paramétré, effectuer une exploration informée de l'espace des hypothèses ? Une possibilité très intéressante existe dans le cas de l'apprentissage de concept, c'est-à-dire de la description d'une classe d'observations (contre toutes les autres). Dans ce cas en effet, on recherche une hypothèse correspondant à une partie de l'espace des observations $\mathcal{X}$ qui couvre les exemples positifs (et exclut les exemples négatifs si ceux-là sont disponibles). Deux hypothèses peuvent alors être comparées en fonction des parties de l'espace $\mathcal{X}$ qu'elles décrivent respectivement, ou encore en fonction des exemples qu'elles *couvrent*. On définit la *couverture d'une hypothèse* comme l'ensemble des observations de $\mathcal{X}$ que cette hypothèse couvre. Une hypothèse est plus générale qu'une autre si sa couverture contient la couverture de la seconde.
La relation d'inclusion définie sur $\mathcal{X}$ induit ainsi une relation de généralité sur $\mathcal{H}$ qui est un préordre partiel. La figure 4 illustre cette notion. On peut noter que ce préordre peut être transformé en une relation d'ordre en considérant que deux hypothèses sont équivalentes si et seulement si l'une est plus générale que l'autre et vice versa et en considérant l'ensemble quotient de $\mathcal{H}$ par rapport à cette relation d'équivalence. Cela revient à ne considérer qu'un représentant parmi les hypothèses équivalentes. Cette relation est partielle et non totale, ce qui signifie que deux éléments quelconques dans l'espace considéré peuvent ne pas être liés par cette relation.
La relation de couverture est fondamentale pour le problème de l'induction. En effet, une hypothèse incorrecte (donc couvrant indûment des exemples négatifs) devra être spécialisée pour que sa couverture exclut ces exemples, alors qu'une hypothèse incomplète (ne couvrant pas tous les exemples positifs connus) devra être généralisée pour que ces exemples deviennent éléments de sa couverture. Il est donc naturel que le processus d'induction soit guidé par la couverture des hypothèses et la relation d'inclusion éventuelle entre ces couvertures.

Nous avons parlé de la relation d'inclusion dans l'espace des observations $\mathcal{X}$, alors que

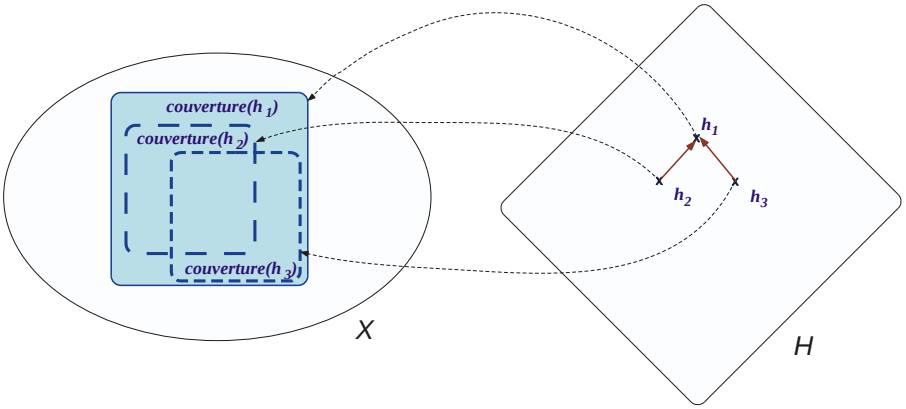

FIGURE 4: *La relation d'inclusion dans $\mathcal{X}$ induit une relation de généralisation dans $\mathcal{H}$. Il s'agit d'une relation de préordre partiel : ici, les hypothèses h_2 et h_3 sont incomparables entre elles, mais elles sont toutes les deux plus spécifiques que h_1.*

l'apprentissage s'effectue par une exploration de l'espace des hypothèses $\mathcal{H}$. Il faut donc, d'une part, définir une relation de généralité permettant de structurer l'espace des hypothèses et modélisant la relation d'inclusion dans $\mathcal{X}$ et, d'autre part, définir des opérateurs de changement d'hypothèses qui respectent la relation de généralité définie dans $\mathcal{H}$ ou la relation d'inclusion dans $\mathcal{X}$.

Relations de généralité

La définition de la relation de généralité et le test de couverture sont étroitement liés. Si la relation de généralité définie sur l'espace des hypothèses respecte la relation d'inclusion sur $\mathcal{X}$, alors la recherche peut être guidée par le test de couverture, et des critères quantitatifs, comme le nombre d'exemples positifs ou négatifs couverts, satisfaisant des propriété de monotonie, peuvent être définis afin de guider et d'élaguer la recherche. Cependant, définir une relation de généralité dépend du langage de représentation des hypothèses et des exemples. Si une telle relation est évidente en logique des propositions, elle devient plus problématique en logique des prédicats. En effet, en logique du 1er ordre, une définition naturelle serait l'implication logique entre deux formules, mais ce problème est non décidable et c'est la raison pour laquelle a été introduite par Plotkin (Plotkin, 1970) la notion de θ-subsomption entre clauses: soient deux clauses A et B, A est plus générale que B s'il existe une susbtitution θ telle que $A.\theta \subseteq B$ [2]. Par exemple, la clause `père(X,Y), père(Y,Z)` $\rightarrow$ `grand_père(X,Z)` est plus générale que la clause `père(jean, paul), père(paul,marie), mère(anne,marie)` $\rightarrow$ `grand_père(jean,marie)`. En effet ces deux clauses sont respectivement réécrites `¬ père(X,Y) ∨ ¬ père(Y,Z) ∨ grand_père(X,Z)` et `¬ père(jean,paul) ∨ ¬ père(paul,marie) ∨ ¬ mère(anne,marie) ∨ grand_père(jean,marie).` Si l'on considère ces clauses comme des ensembles de littéraux et si l'on instancie X par $jean$, Y par $paul$ et Z par $marie$, la première ainsi instanciée est bien incluse dans la seconde. Cette rela-

2. Une clause est une disjonction de littéraux, assimilée dans cette définition à un ensemble de littéraux.

tion de généralité permet de définir des opérateurs de généralisation permettant de transformer une clause en une clause plus générale, comme supprimer un littéral, transformer une constante en variable ou transformer deux occurrences d'une même variable en des variables différentes. Cette définition ne tient pas compte des connaissances que l'on peut avoir sur le domaine, comme par exemple qu'un père ou une mère sont des parents. Mais même cette définion est problématique car elle implique des comparaisons éventuellement coûteuses entre hypothèses.
Il est parfois possible d'inclure l'espace des exemples dans celui des hypothèses, en faisant d'un exemple une sorte d'hypothèse atomique. On parle alors de *single representation trick* ou d'astuce de la représentation unique. Cela simplifie en général le test de couverture. Par exemple, une hypothèse pourra être une clause et un exemple pourra être une clause complètement instanciée, comme dans l'illustration donnée ci-dessus. Il est important de réaliser que, même dans ce cas, la comparaison entre une hypothèse et un exemple, afin de savoir si l'hypothèse couvre l'exemple, peut également être source de complexité algorithmique. La complexité du *test de couverture* est souvent un élément déterminant dans le choix du codage des données et du langage des hypothèses. Par exemple, dans le cas de l'apprentissage d'un ensemble de clauses de Horn (équivalent à un programme Prolog), le test de couverture peut impliquer un démonstrateur de théorème ou, à tout le moins, un programme de satisfaction de contraintes.
Il arrive que les données résultent de mesures imprécises ou approximatives, et, en général, il est possible que la précision sur les valeurs des attributs de description soit illusoire. Dans ce cas, il peut être intéressant de changer de représentation pour rendre compte d'une précision plus réduite et permettre éventuellement d'obtenir des descriptions de concept plus compréhensibles. Le formalisme des *rough sets* (Suraj, 2004) offre un outil pour le raisonnement approximatif applicable en particulier pour la sélection d'attributs informatifs, la catégorisation, et la recherche de règles de classification ou de décision. Sans entrer dans les détails, le formalisme des rough sets permet de chercher une redescription de l'espace des exemples tenant compte de relations d'équivalence induites par les descripteurs sur les exemples disponibles (voir chapitre I.3). Puis, étant donnée cette nouvelle granularité de description, les concepts peuvent être décrits en termes d'approximation inférieure et supérieure (voir figure 5). Cela conduit à de nouvelles définitions des notions de couverture d'un exemple par un concept et de relation de généralité entre concepts.
On voit donc l'*importance du langage de représentation des hypothèses* dans la détermination d'une relation de préordre permettant l'exploration efficace de l'espace des hypothèses et dans la définition des opérateurs de changement d'hypothèses (ou de motifs).
Parmi les relations d'ordre possibles, on privilégie celles qui induisent une structure de *treillis* sur $\mathcal{H}$. Cela signifie que pour tout couple d'hypothèses h_i et h_j, il existe au moins une hypothèse qui soit plus générale que chacune d'entre elles et qu'il n'est pas possible de la spécifier sans perdre cette propriété. L'ensemble de ces hypothèses est appelé le *généralisé maximalement spécifique* de h_i et h_j et noté $gms(h_i, h_j)$. De même, il existe un ensemble d'hypothèses plus spécifiques que h_i et h_j qu'il n'est pas possible de généraliser sans perdre cette propriété. On appelle cet ensemble le *spécialisé maximalement général* et on le note $smg(h_i, h_j)$.
Par une extension facile au cas de plus de deux hypothèses, on peut définir de même un ensemble $gms(h_i, h_j, h_k, \ldots)$ et un ensemble $smg(h_i, h_j, h_k, \ldots)$.
Finalement, nous supposons [3] qu'il existe dans $\mathcal{H}$ une hypothèse plus générale que toutes les

3. C'est en général le cas en pratique.

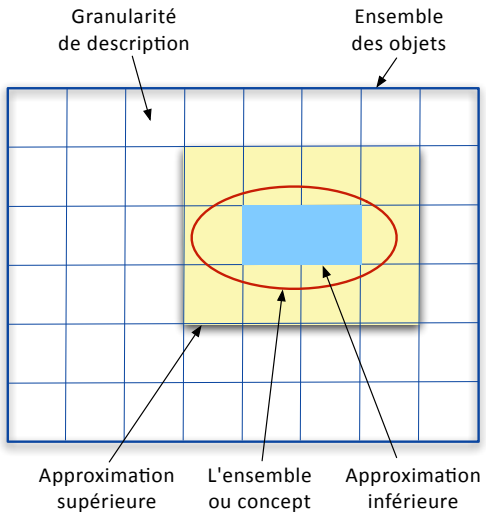

FIGURE 5: *Les rough sets : changement de granularité de description et redéfinition d'un concept par une approximation inférieure et une approximation supérieure.*

autres (ou élément maximal) notée $\top$ et une hypothèse plus spécifique que toutes les autres (ou élément minimal) notée $\bot$ (voir la figure 6).

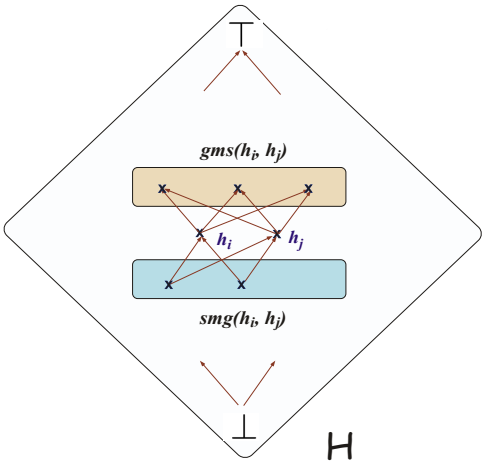

FIGURE 6: *Une vision schématique et partielle du treillis de généralisation sur $\mathcal{H}$ induit par la relation d'inclusion dans $\mathcal{X}$. Chaque flèche indique la relation de généralité (notée $\preceq$ dans le texte). L'élément le plus spécifique du treillis est $\bot$ et le plus général est $\top$.*

Un domaine dans lequel l'apprentissage repose sur la relation de généralité dans l'espace des hypothèses est celui de l'induction de langages ou de grammaires à partir d'exemples, et éventuellement de contre-exemples, de séquences. Dans le cas de ce que l'on appelle l'*inférence*

grammaticale, la description des hypothèses prend généralement la forme d'automates. La plupart des travaux ont porté sur l'induction de langages réguliers qui correspondent aux automates à états finis. Il se trouve que l'utilisation de cette représentation conduit naturellement à une opération associée à une relation de généralité entre automates. Sans entrer dans les détails formels, si l'on considère un automate à états finis et que l'on fusionne de manière appropriée deux états de cet automate, on obtient un nouvel automate qui accepte au moins toutes les séquences acceptées par le premier automate. Cet automate dérivé est donc plus général. Utilisant cette opération de généralisation, la plupart des méthodes d'inférence grammaticale partent ainsi d'un automate particulier, acceptant exactement les séquences positives, et le généralisent, par une succession de fusions d'états, en s'arrêtant soit quand une séquence négative est couverte soit quand un critère d'arrêt sur l'automate courant est vérifié. L'ouvrage de Colin de la Higuera (de la Higuera, 2010) décrit très complètement toutes les techniques de l'inférence grammaticale.

Opérateurs de généralisation/spécialisation

Une fois définie une relation de généralité sur l'espace des hypothèses, il faut définir des opérateurs qui permettent de parcourir cet espace des hypothèses. Ainsi, par exemple, supposons que l'espace des hypothèses soit celui de conjonctions en logique des propositions. Une description du concept de `voiture bling-bling` pourra ainsi être : `rouge` $\wedge$ `bruyant` $\wedge$ `cher`. Par ailleurs, on supposera que les opérateurs de changement d'hypothèse sont l'*ajout* ou le *retrait* d'un terme de la conjonction. On pourrait ainsi produire la description `rouge` $\wedge$ `bruyant` à partir de la description précédente. Il est évident que les véhicules vérifiant cette description incluent ceux qui vérifient la première description, elle est donc plus générale. On peut montrer que l'opérateur de retrait d'un terme de la conjonction dans ce langage est associé à un gain en généralité de l'hypothèse produite par rapport à l'hypothèse de départ. Cette association entre opérateur de changement d'hypothèse et degré de généralité peut alors être utilisée pour guider efficacement l'exploration de l'espace des hypothèses. En effet, si une hypothèse candidate courante ne couvre pas tous les exemples positifs, il faudra envisager les opérateurs associés à une généralisation de cette hypothèse, tandis, qu'inversement, si des exemples négatifs sont couverts, il faudra utiliser des opérateurs associés à une spécialisation. Cela conduit à des *stratégies* d'exploration de l'espace des hypothèses, *descendante* (par spécialisation), *ascendante* (par généralisation) ou mixte.

Les opérateurs de généralisation/spécialisation dépendent du langage de description des hypothèses. Si c'est assez simple en logique des propositions (e.g. pour spécialiser, ajouter une condition à la règle), cela devient plus compliqué si l'on suppose que l'on dispose de connaissances sur le domaine permettant par exemple de spécialiser une condition, ou dans des langages plus complexes comme la logique du 1er ordre. Un opérateur pour être intéressant doit vérifier des propriétés comme être localement fini, i.e. engendrer un nombre fini et calculable de raffinements, ou encore être complet, i.e. si deux hypothèses sont comparables du point de vue de leur généralité, on peut passer de l'une à l'autre par l'application d'un nombre fini de raffinements, etc.

Exploration de l'espace des hypothèses

Pour apprendre un concept à partir d'exemples positifs et négatifs, la méthode la plus courante consiste à apprendre une hypothèse couvrant des exemples positifs puis itérer le processus sur les exemples restants. D'autres méthodes sont envisageables comme séparer les exemples positifs en classes (classification non supervisée) et apprendre une règle pour chaque classe.
Pour construire une règle couvrant des exemples positifs, plusieurs stratégies sont utilisées : une approche gloutonne, qui construit la règle itérativement ajoutant des conditions dans le corps de la règle de manière heuristique (comme par exemple dans le système Foil) ou une approche dirigée par les données qui à partir d'un exemple positif construit une règle couvrant cet exemple (comme dans le système Progol).
Des approches récentes proposent de rechercher de manière exhaustive toutes les règles avec un support et une confiance suffisants (on retrouve alors la problématique de recherche de règles d'association) et de construire un classifieur à partir de ces règles : par exemple, chaque règle applicable vote pour la classe correspondante avec un poids fonction de sa confiance.

Analyse formelle de concepts

Comme nous l'avons vu, il existe une dualité entre l'espace des observations et l'espace des hypothèses. Cette dualité est particulièrement mise en évidence par la notion de connexion de Galois mise en œuvre dans l'Analyse Formelle de Concepts (Ganter *et al.*, 1998, 2005). Etant donné un ensemble d'objets $\mathcal{O}$, un ensemble de descripteurs propositionnels $\mathcal{A}$ et une relation exprimant que tel objet satisfait tel descripteur, on construit deux opérateurs. Le premier, noté f, correspond à la notion de couverture : étant donné un ensemble de descripteurs A, $f(A)$ retourne l'extension de A, i.e. l'ensemble des objets satisfaisant A. Le second, noté g, correspond à la notion de généralisation : étant donné un ensemble d'objets O, $g(O)$ retourne l'intention de O, i.e., l'ensemble des descripteurs vérifiés par tous les objets de O. Considérons à nouveau le concept de 'voiture'. Une voiture peut être décrite par les descripteurs `bruyante`, `silencieuse`, `chère`, `bon_marché`, `couleur_vive`, `couleur_claire`. Supposons que l'on ait trois exemples de voitures définies par : `bruyante` $\wedge$ `chère` $\wedge$ `couleur_vive` pour la première, `bruyante` $\wedge$ `chère` $\wedge$ `couleur_claire` pour la deuxième et `bruyante` $\wedge$ `bon_ marché` $\wedge$ `couleur_claire` pour la troisième, le descripteur `bruyante` a pour extension les trois voitures alors que l'ensemble formé par les deux descripteurs `bruyante` et `chère` n'a pour extension que les deux premières voitures. L'intension des deux premières voitures, i.e., les descripteurs qu'elles ont en commun est composée de `bruyante` et `chère`. Ce couple d'opérateurs forme une correspondance de Galois : f et g sont anti-monotones (si $A_1 \subseteq A_2$ alors $f(A_2) \subseteq f(A_1)$) et si $O_1 \subseteq O_2$ alors $g(O_2) \subseteq g(O_1)$), tout ensemble de descripteurs est inclus dans l'extension de son intension (pour tout A de $\mathcal{A}$, on a $A \subseteq g(f(A))$) et tout ensemble d'objets est inclus dans l'extension de son intension (pour tout O de $\mathcal{O}$, $O \subseteq f(g(0))$). Enfin, on a les propriétés suivantes : $f(A) = f(g(f(A)))$ et $g(O) = g(f(g(O)))$. L'opérateur gof a reçu une attention particulière dans le domaine de la recherche de motifs fréquents et est appelée *fermeture* d'un ensemble de descripteurs.
Un *concept* est alors défini comme un couple d'objets et de descripteurs, (O, A), tel que O est l'extension de A et A est l'intention de O. Dans l'exemple précédent, le couple formé des deux premières voitures et des descripteurs `bruyante` et `chère` forme un concept. Cette notion

est particulièrement utilisée lors de la recherche d'itemsets fréquents puisque si (O, A) est un concept alors A est un ensemble fermé ($A = g(f(A))$) et l'ensemble des itemsets fermés muni de $\subseteq$ est un treillis.
Pour conclure, notons que toutes les stratégies de recherche sont fondées sur le lien entre relation de généralité dans l'espace des hypothèses et couverture dans l'espace des observations. Dans le cas de l'apprentissage à partir d'exemples positifs et négatifs, on va généraliser (resp. spécialiser) une hypothèse pour couvrir plus d'exemples positifs (resp. rejeter plus d'exemples négatifs). Dans celui de la recherche de motifs fréquents (couvrant un pourcentage donné d'observations) la propriété de monotonie de la couverture des motifs (tout sous-ensemble d'un motif fréquent, qui peut aussi être vu comme une conjonction de conditions, est nécessairement également fréquent) est fondamentale ; elle permet d'organiser efficacement la recherche en rendant possible l'élagage de directions de recherche. L'algorithme Apriori, le plus connu en fouille de données, exploite astucieusement cette propriété (voir (Han et Kamber, 2006)).

10.3.2 Quatre illustrations

Induction d'arbre de décision.

L'induction d'arbres de décision est certainement le cas le plus connu d'apprentissage de modèles à structure variable. Notons cependant que contrairement aux approches décrites précédemment, l'algorithme d'apprentissage ne repose pas sur l'exploitation d'une relation de généralité entre modèles, mais il utilise une stratégie gloutonne construisant un arbre de plus en plus complexe correspondant à un découpage de plus en plus fin de l'espace $\mathcal{X}$ des observations.
En apprentissage artificiel, un arbre de décision comporte des nœuds et des arcs entre ces nœuds. Chaque nœud interne correspond à un test sur l'un des attributs de description des données, par exemple `taille > 1.70m`, alors que les feuilles correspondent à l'une des étiquettes de classe. Pour classer une observation, on effectue le test à la racine de l'arbre et en fonction de la réponse, on suit la branche correspondante vers l'un des sous-arbres, et ce jusqu'à arriver à une feuille qui est alors l'étiquette prédite. Il est notable qu'un arbre de décision peut s'exprimer sous la forme d'une disjonction de règles de prédiction, chacune d'entre elles ayant pour condition une conjonction de tests depuis la racine de l'arbre jusqu'à une feuille et pour conclusion l'étiquette de classe associée (voir figure 7).

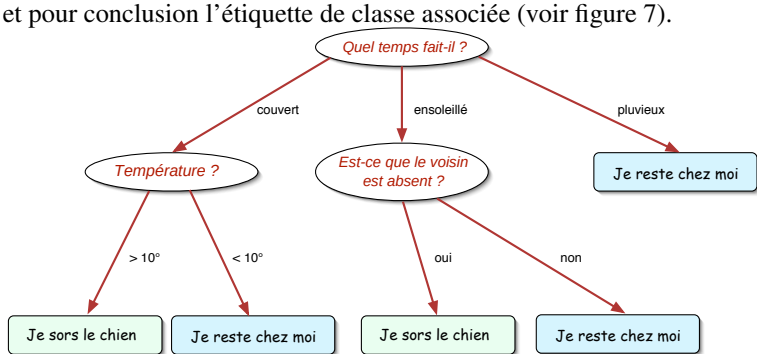

FIGURE 7: *Un arbre de décision concernant la promenade du chien.*

Un arbre de décision est ainsi l'expression symbolique d'une partition de l'espace des entrées. L'apprentissage consiste à trouver une telle partition en cherchant à optimiser un critère inductif. Les algorithmes existants opèrent par partitions successives en sous-espaces, chaque partition correspondant à un test sur l'un des attributs. Le critère inductif global est remplacé par un critère local qui optimise l'homogénéité, en termes d'étiquette, des classes produites dans la partition. Les critères les plus utilisés sont certainement le gain d'information utilisé dans C5.0 (Quinlan, 1993; Kotsiantis, 2007) et fondé sur la notion d'entropie ou l'indice de Gini, utilisé dans le système Cart (Breiman *et al.*, 1984). On ne peut donc obtenir que des partitions *parallèles aux axes* dans la mesure où les tests sur des variables numériques sont en général de la forme $X \geq seuil$. Des techniques d'élagage de l'arbre construit sont ensuite appliquées afin d'éviter la sur-adaptation aux données.

Cet algorithme est de complexité réduite, en $\mathcal{O}(m \cdot d \cdot \log(m))$ où m est la taille de l'échantillon d'apprentissage et d le nombre d'attributs de description. De plus, il permet d'obtenir des hypothèses généralement faciles à interpréter. C'est l'exemple type d'un algorithme de type *diviser pour règner* avec exploration gloutonne.

Espace des versions et algorithme d'élimination des candidats

Tom Mitchell a montré à la fin des années soixante-dix (Mitchell, 1982, 1997) comment on pouvait exploiter de manière très intéressante une relation d'ordre partiel de généralité pour apprendre l'*espace des versions* c'est-à-dire l'ensemble de toutes les hypothèses qui sont cohérentes avec les données connues à un instant donné, c'est-à-dire qui sont complètes (couvrent tous les exemples positifs) et correctes (ne couvrent aucun exemple négatif). Ces hypothèses sont donc de risque empirique nul. Sous certaines conditions en effet, on peut montrer que l'ensemble des hypothèses cohérentes avec les données d'apprentissage sont bornées par deux ensembles frontière dans le treillis de généralisation défini sur $\mathcal{H}$. L'un, dénommé le *S-set* est l'ensemble des hypothèses les plus spécifiques couvrant les exemples positifs et excluant les exemples négatifs. L'autre, le *G-set*, est l'ensemble des hypothèses maximalement générales également cohérentes avec les données d'apprentissage.

L'algorithme d'élimination des candidats conçu par Tom Mitchell permet de mettre à jour ces deux ensembles frontière en considérant séquentiellement les exemples d'apprentissage. Il correspond à une recherche bidirectionnelle en largeur d'abord qui maintient incrémentalement, après chaque nouvel exemple, le *S-set* et le *G-set*. Il élimine les candidats, c'est-à-dire les hypothèses, car chaque nouvel exemple ajoute des contraintes sur les hypothèses possibles, celles de risque empirique nul. En supposant que le langage de description des hypothèses soit parfaitement choisi, et que les données soient suffisantes et non bruitées, l'algorithme peut en principe converger vers une hypothèse unique qui est le concept cible. Une excellente description de cet algorithme se trouve dans le chapitre 2 du livre de Tom Mitchell (Mitchell, 1997).

Alors que cette technique, qui est à la base de nombreux algorithmes d'apprentissage d'expressions en logique ainsi que d'automates à états finis, connaissait une baisse d'intérêt avec l'avénement des méthodes plus numériques, elle est redevenue d'actualité grâce à l'émergence du domaine de la fouille de données et aux travaux de chercheurs provenant davantage de la communauté des bases de données.

Programmation Logique Inductive.

La Programmation Logique Inductive (PLI) a fait l'objet d'une attention particulière depuis les années 80. Initialement étudiée pour la synthèse de programmes logiques à partir d'exemples, elle s'est progressivement orientée vers l'apprentissage de connaissances dans des formalismes du 1er ordre à partir de données relationnelles, dépassant ainsi le cadre classique de données décrites dans des espaces vectoriels et permettant de prendre en compte les relations entre les données. Lors de la synthèse de programmes logiques, un problème clef concernait l'apprentissage de concepts récursifs ou mutuellement récursifs. Un nouvel intérêt pour ce problème s'est fait jour avec les récents développements en ASP (Answer Set Programming) (voir le chapitre II.4) L'évolution de la PLI l'a conduite à s'attaquer à d'autres problèmes comme le traitement des données numériques et le traitement des données bruitées.

L'un des intérêts de la PLI est la possibilité de décrire des connaissances du domaine, permettant d'obtenir des définitions de concepts plus intéressantes. Par exemple si l'on souhaite apprendre le concept de 'grand-père' à partir d'exemples de personnes liées par la relation père ou mère, introduire le concept de 'parent' permettra d'obtenir une définition plus concise du concept. La définition de θ-subsomption définie en Section 10.3.1 doit alors être modifiée en conséquence.

L'une des principales difficultés à laquelle doit s'attaquer la PLI est la complexité due à la taille de l'espace des hypothèses et la complexité du test de couverture. Pour pallier ces problèmes, on définit des biais syntaxiques permettant de réduire la taille de l'espace de recherche ou des biais sémantiques. On a aussi cherché à profiter pleinement des travaux menés en apprentissage propositionnel. Dans ce cadre, une technique assez couramment utilisée, appelée propositionnalisation, consiste à transformer le problème d'apprentissage en 1er ordre en un problème propositionnel ou attribut-valeur (voir un exemple dans (Zelezný et Lavrac, 2006)). Toute la difficulté réside alors dans la construction de caractéristiques pertinentes reflétant le caractère relationnel des données et minimisant la perte d'information.

Les systèmes les plus connus de PLI sont certainement les systèmes FOIL (Quinlan, 1996) et PROGOL (Muggleton, 1995). FOIL adopte une stratégie descendante : la clause la plus générale (sans condition dans le corps de la règle) est successivement raffinée pour rejeter tous les exemples négatifs. Il repose sur une stratégie gloutonne : une heuristique, proche du gain d'information et fondée sur le nombre d'exemples positifs couverts et le nombre d'exemples négatifs rejetés mesure la qualité des spécialisations, la meilleure est choisie sans retour arrière. Cette stratégie souffre d'un problème connu : il peut être nécessaire d'ajouter des raffinements, correspondant à des dépendances fonctionnelles, qui ne permettent pas de rejeter des exemples négatifs mais indispensables pour construire une clause cohérente. PROGOL quant à lui choisit un exemple positif, construit sa clause saturée (la plus spécifique couvrant cet exemple) et effectue sa recherche dans l'espace des généralisations de cette clause saturée. Ses résultats dépendent de l'ordre dans lequel les exemples positifs sont traités. Intéressant en termes de stratégies de recherche, le système ALEPH peut être considéré comme une plate-forme permettant d'implanter différentes stratégies.

Durant les années quatre-vingt-dix, certains travaux en informatique ont montré qu'une grande classe de problèmes de satisfaction de contraintes présentait un phénomène de *transition de phase*, à savoir une variation brutale de la probabilité de trouver une solution quand on varie les paramètres du problème. Peu de temps après, il a été noté que l'apprentissage de programmes logiques peut être réduit à un problème de satisfaction de contraintes (trouver une hypothèse

couvrant les exemples positifs, mais pas les négatifs). Il a alors été montré empiriquement qu'un phénomène de transition de phase apparaît effectivement dans la programmation logique inductive dès que le problème n'est plus trivial. Cette découverte a été étendue à certains types de problèmes en inférence grammaticale. L'impact de ce phénomène de transition de phase et les moyens d'y faire face font l'objet de débats (Saitta *et al.*, 2011).
Notons enfin que si la classification supervisée a été longtemps la principale tâche étudiée en ILP, il existe actuellement des travaux sur la recherche de motifs fréquents dans les bases de données relationnelles, ou la découverte de sous-groupes.
Parmi les références importantes, citons (Lavrac et Dzeroski, 1994; Dzeroski et Lavrac, 2001; Raedt, 2008; Fürnkranz *et al.*, 2012).

La recherche de motifs fréquents et de règles d'association : l'algorithme Apriori.

On appelle *motif* (ou *itemset*) toute conjonction de propositions ou de relations attribut-valeur, par exemple (*âge > 60*) $\wedge$ (*HDL-cholesterol > 1,65 mmol/L*), suivant le langage de description choisi et on appelle *motif fréquent* tout motif dont la fréquence d'apparition dans une base de données dépasse un certain seuil appelé *support*. Un problème essentiel en fouille de données est de rechercher les motifs fréquents. Entre autres, cela permet ensuite d'identifier des règles d'association que l'on espère intéressantes, une règle d'association étant de la forme $I \rightarrow J$ où I et J sont des motifs disjoints.
Comme dans la technique de calcul de l'espace des versions, il est naturel de considérer la relation de généralité entre motifs. Un motif est plus général qu'un autre si les exemples dans lesquels il apparaît contient les exemples dans lesquels apparaît le second. L'opérateur de généralisation (resp. spécialisation) est ici la suppression (resp. ajout) d'un terme de la conjonction du motif. On peut alors définir un treillis modélisant la relation de généralité dans l'espace des motifs. A ce treillis est associée une relation d'anti-monotonie : tout motif non fréquent ne peut être spécialisé en un motif fréquent, c'est-à-dire que si un motif n'est pas fréquent, il est inutile de lui ajouter des termes, il restera non fréquent.
Cette observation est à la base de l'algorithme Apriori (Agrawal et Srikant, 1994), certainement le plus connu des algorithmes de recherche de motifs fréquents et de règles d'association. L'algorithme se décompose en deux étapes : la recherche des motifs fréquents puis la construction à partir de ces motifs des règles d'association. La première étape nécessite de confronter les hypothèses à la base de données, elle pose des problèmes de complexité importants et c'est donc celle qui a reçu le plus d'attention.
Dans le cadre le plus simple, Apriori effectue une recherche en largeur dans le treillis, considérant d'abord les motifs de taille 1, puis ceux de taille 2 et ainsi de suite. La base de données n'est parcourue qu'une seule fois par niveaux pour calculer le support des motifs de ce niveau. Pour élaguer la recherche, la contrainte d'anti-monotonie est utilisée. La complexité dépend de la taille de la base de données (parcourue pour calculer les supports) et du nombre de parcours de la bases de données, que l'on peut estimer comme de l'ordre de la taille des plus grands motifs fréquents.
La complexité d'Apriori restant trop élevée, de nombreux algorithmes ont été développés, soit proposant de nouvelles stratégies de recherche (par partitionnement de la base de données ou par échantillonnage), soit reposant sur des représentations différentes des bases de données : association à chaque itemset des identifiants de la base de données (APriori-Tid (Agrawal et Srikant, 1994)), représentation sous forme d'un arbre (FP-Growth, (Han *et al.*, 2004)), ...

Enfin, notons qu'une autre source de complexité est le nombre de motifs fréquents engendrés. On s'est donc intéressé à définir des représentations condensées de ces motifs. La plupart des travaux reposent sur la notion de connexion de Galois et de fermeture définie comme suit : un motif est fermé si les observations qui le contiennent n'ont pas d'autres éléments en commun. On retrouve la notion de concept sous-jacente à l'Analyse Formelle de Concepts, décrite en Section 10.3.1. On peut remarquer que le support d'un motif peut alors se définir comme le cardinal de son extension (le nombre d'éléments de la base de données qui le vérifient) et on a alors des propriétés exprimant par exemple que deux motifs qui ont la même fermeture ont le même support, ou encore que si un motif est inclus dans un autre et a même support alors les deux motifs ont même fermeture. Ces propriétés permettent de montrer que l'ensemble des motifs fermés avec leur support forment une représentation exacte de l'ensemble des motifs et qu'il suffit de ne stocker en mémoire que les motifs fermés (voir par exemple (Zaki, 2000) et (Bastide *et al.*, 2000) pour un travail sur la recherche de fermés).

10.4 Méta-apprentissages

Un célèbre théorème, le *no-free lunch theorem*, affirme qu'aucune méthode d'apprentissage n'est uniformément supérieure à n'importe quelle autre méthode sur tous les problèmes d'induction possibles. La raison profonde en est qu'*a priori* il n'existe aucun lien entre les données d'apprentissage et les données test. Si l'apprentissage revient à jouer contre un adversaire sans contrainte. il ne peut gagner en moyenne. Il est donc nécessaire de s'appuyer sur l'hypothèse qu'il existe une « continuité » entre le passé et l'avenir autorisant l'extrapolation des régularités détectées dans le passé au futur. Concrètement, la conséquence de ce théorème est que chaque méthode est adaptée à une classe de problèmes, mais est mauvaise en dehors de cette classe, c'est-à-dire éventuellement pire qu'une méthode de prédiction opérant au hasard. Et il est tout à fait illusoire de vouloir démontrer la supériorité d'un algorithme d'apprentissage dans l'absolu (en dehors de considération d'efficacité). Il est à noter que ce théorème n'est pas propre à l'apprentissage, mais a des contreparties dans plusieurs domaines, en optimisation en particulier, ainsi que pour le problème de la détermination de la meilleure distance dans des problèmes de clustering (voir par exemple : (Wolpert, 1996b,a)).
Puisqu'il est illusoire de chercher un algorithme idéal, il devient intéressant de déterminer automatiquement l'algorithme adapté au problème courant, voire de combiner des algorithmes. Parce que l'on suppose donnée une famille de méthodes d'apprentissage et qu'il s'agit de trouver comment les appliquer à meilleur escient, on parle de « méta-apprentissage ». Un exemple minimal de méta-apprentissage aurait comme argument deux méthodes d'apprentissage de concept $\mathcal{A}_1$ et $\mathcal{A}_2$ (par exemple un SVM et une méthode d'induction d'arbre de décision) et produirait une hypothèse $H(\mathbf{x}) = \text{signe}\{\alpha\, h_1(\mathbf{x}) + (1-\alpha)\, h_2(\mathbf{x})\}$ à partir des deux hypothèses h_1 et h_2 produites par $\mathcal{A}_1$ et $\mathcal{A}_2$ à partir d'un échantillon d'apprentissage. Le problème étant ici de déterminer le meilleur coefficient $\alpha \in [0, 1]$.
Il existe de multiples méthodes de méta-apprentissage. La plus large famille concerne des méthodes d'apprentissage de combinaisons d'hypothèses, à l'instar de l'exemple cité plus haut. Elles incluent les méthodes de *boosting* et de *bagging*. Une autre famille consiste à combiner des apprentissages s'appuyant sur des descriptions des données différentes, comme par exemple l'en-tête des mails, d'une part, et leur texte, d'autre part. On parle alors de *co-apprentissage* (Blum et Mitchell, 1998).

Un autre type de méta-apprentissage consiste à régler automatiquement les méta-paramètres de l'algorithme utilisé, ceux qui contrôlent l'espace d'hypothèses considéré (encore appelé *biais de représentation*), ou bien la stratégie d'exploration de cet espace (appelé *biais de recherche*).

10.4.1 Méta-apprentissage par comités d'experts

Le boosting Peut-être le plus célèbre des algorithmes de méta-apprentissage, le boosting est du à Bob Shapire et à Yoav Freund dans les années quatre-vingt-dix (Shapire et Freund, 2012). Stimulé par une question de son directeur de thèse, Michael Kearns, Shapire a d'abord montré qu'il était possible de prendre un algorithme d'apprentissage de concept à peine meilleur que la prédiction au hasard et d'en tirer une règle de prédiction supérieure en découpant astucieusement l'échantillon d'apprentissage initial en trois sous-échantillons à partir desquels sont appris trois hypothèses h_1, h_2 et h_3 qui sont alors combinées de manière linéaire : $H(\mathbf{x}) =$ vote majoritaire $\{h_1(\mathbf{x}), h_2(\mathbf{x}), h_3(\mathbf{x})\}$. L'idée essentielle est d'apprendre trois hypothèses qui se complètent au sens où la deuxième est bonne là où la première n'était pas meilleure que le hasard, et la troisième est apprise pour départager les deux premières là où elles sont en désaccord. Après ce premier résultat d'un intérêt surtout théorique, Schapire et Freund ont généralisé cette méthode dans un algorithme, Adaboost (Freund et Schapire, 1996), qui apprend itérativement T hypothèses en focalisant à chaque étape t, l'hypothèse courante sur les régions de l'espace des entrées où les hypothèses précédentes étaient mauvaises.

1. Apprentissage d'une hypothèse h_t sur l'échantillon d'apprentissage courant $\mathcal{S}_t$
2. Modification de l'échantillon courant pour produire $\mathcal{S}_{t+1}$ en surpondérant les exemples sur lesquels h_t s'est trompée et en sous-pondérant les autres.

Ces étapes sont répétées T fois pour obtenir finalement l'hypothèse combinée :

$$H(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t \, h_t(\mathbf{x})$$

Les coefficients α_t sont calculés en fonction de la performance de l'hypothèse h_t sur l'échantillon courant $\mathcal{S}_t$. Meilleure est la performance, plus grand est ce coefficient, ce qui revient naturellement à donner plus de poids aux « experts » qui se sont révélés les meilleurs. Nous renvoyons à (Cornuéjols et Miclet, 2010; Shapire et Freund, 2012; Zou, 2012) par exemple pour plus de détails.
Cet algorithme, très simple à implanter, donne souvent des résultats excellents, dans la mesure où les hypothèses sont suffisamment variables quand l'échantillon d'apprentissage varie (par pondération des exemples) et où les données sont peu bruitées. De plus, il est remarquablement robuste au sur-apprentissage, ce que des études théoriques ont expliqué par un argument de type « recherche d'une marge maximale », à l'instar des méthodes à noyaux.

Le bagging. Du à Breimann (Breiman, 1996), le bagging est une autre méthode très usitée de combinaison d'hypothèses dans laquelle la manipulation de l'échantillon d'apprentissage à chaque étape se fait par un tirage aléatoire avec remise dans l'échantillon initial et où l'hypothèse finale est simplement la moyenne des hypothèses apprises.

Les arbres de décision. Il est remarquable que les méthodes décrites précédemment combinent des « experts » (hypothèses) locaux, au sens où on a focalisé leur apprentissage sur des régions particulières de l'espace, par un vote applicable en n'importe quel point de l'espace d'entrée $\mathcal{X}$. Intuitivement, il semblerait plus adapté de ne consulter les experts que sur leur domaine d'expertise et pas ailleurs. C'est exactement ce qui est réalisé dans l'apprentissage par arbre de décision. Chaque nœud de l'arbre engendre une subdivision de l'espace $\mathcal{X}$. Ces subdivisions sont poursuivies jusqu'à ce qu'un expert local soit capable de prédire l'étiquette ou la valeur pour toute forme $\mathbf{x}$ de la sous-région correspondante. Il s'agit d'un certain côté d'une forme duale de méta-apprentissage par combinaison d'experts.

10.4.2 Apprendre à paramétrer les algorithmes

Chaque algorithme d'apprentissage est associé à un principe inductif (e.g. minimisation du risque empirique) et à une famille de méthodes (e.g. perceptron multicouche) et met en jeu des méta-paramètres (e.g. nombre de couches et de neurones cachés). Ces paramètres sont susceptibles de gouverner l'espace des hypothèses exploré par l'apprenant, la stratégie exploratoire, la taille de la mémoire, etc. Ils sont fréquemment réglés par l'utilisateur, à partir de son expérience et en fonction des caractéristiques de la tâche étudiée afin de maximiser la performance en généralisation de l'algorithme utilisé. Pourquoi cependant ne pas étendre l'apprentissage au contrôle de ces méta-paramètres ?

Tel qu'il est conçu actuellement, l'apprentissage est associé à un problème d'optimisation lié au principe inductif mis en jeu. On cherche un modèle du monde (ou hypothèse) optimisant une certaine fonction définie sur l'échantillon d'apprentissage disponible avec l'espoir que le modèle appris sera performant dans de nouvelles situations, ce que l'on appelle la performance en généralisation. L'apprentissage des méta-paramètres consiste à jouer sur le problème d'optimisation lui-même en le soumettant à un méta-problème d'optimisation dans lequel l'optimisation porte sur la performance en généralisation (voir algorithme 17).

Algorithme 17 : ALGORITHME DE MÉTA-APPRENTISSAGE

```
1 début
2     répéter
3         Réglage des méta-paramètres
4         répéter
5             Apprentissage de l'hypothèse la meilleure sur S_n : ĥ(S_n)
6         jusqu'à Optimisation du critère inductif sur l'échantillon S_n
7     jusqu'à Optimisation de la performance en généralisation
```

Ainsi, une approche générale mise en avant par Vapnik, appelée *Structural Risk Minimization*, consiste à considérer des espaces d'hypothèses de capacité d'approximation (ou d'expressivité) croissante et à chercher dans chacun de ces espaces la meilleure hypothèse $\hat{h}$. Normalement, la performance en généralisation de la séquence d'hypothèses correspondante présente un optimum général. On choisit alors l'espace d'hypothèses associé comme espace à explorer pour des apprentissages ultérieurs dans les mêmes conditions.

Une question cruciale devient alors celle de l'évaluation de la performance en généralisation d'une hypothèse.

Estimation de la performance en généralisation. Le cœur du problème est du à ce que par définition, lors de l'apprentissage, la performance en généralisation ne peut être estimée qu'en utilisant les données disponibles, c'est-à-dire l'échantillon d'apprentissage $\mathcal{S}_m$. Pour qu'il y ait bien deux problèmes d'optimisation emboîtés, il est indispensable que le problème d'optimisation lié à l'apprentissage de l'hypothèse $\hat{h}$ (bouche interne) n'utilise pas les mêmes données que le problème d'optimisation des méta-paramètres (boucle externe). Sans entrer dans les détails, on risque sinon d'obtenir des résultats optimistes et biaisés.
Lorsque les données sont abondantes, il est possible de réserver une part significative de l'échantillon pour l'estimation de la performance. On apprend alors $\hat{h}$ sur un sous-échantillon $\mathcal{S}_n \subset \mathcal{S}_m$ et on évalue la performance sur l'échantillon $\mathcal{S}_m \setminus \mathcal{S}_n$, généralement appelé échantillon de validation. Souvent, cependant, les données sont rares et il faut recourir à des procédés tels que la *validation croisée* pour à la fois apprendre l'hypothèse $\hat{h}$ et évaluer aussi précisément que possible la performance en généralisation (voir (Cornuéjols et Miclet, 2010; Japkowicz, 2011) pour des informations plus complètes).
La seule estimation du taux d'erreur en généralisation est parfois insuffisante pour l'objectif que l'on se fixe. Ainsi, par exemple, il peut être utile d'estimer plus précisément les taux de faux positifs et de faux négatifs, ou bien encore de précision et de rappel. Lorsque l'on fait varier certains méta-paramètres de l'algorithme, on peut alors obtenir des courbes montrant l'évolution de ces critères. La courbe ROC (de l'anglais *Receiver Operating Characteristics*, motivé par le développement des radars durant la seconde guerre mondiale) en est l'exemple le plus connu. On cherchera alors typiquement à optimiser l'aire sous la courbe ROC qui caractérise le pouvoir discriminant de la méthode de classification employée.
Il faut souligner que dans le cas de données non étiquetées, en apprentissage non supervisé, il existe des critères spécifiques d'estimation de la performance. Ils sont alors très dépendants d'hypothèses *a priori* sur le type de régularités présentes dans le monde, et peuvent facilement donner des résultats très trompeurs si ces hypothèses ne sont pas vérifiées.

Exemples de méta-paramètres susceptibles d'être optimisés automatiquement. Les méta-paramètres en jeu dans l'apprentissage concernent essentiellement d'une manière ou d'une autre l'espace des hypothèses considéré et la manière de l'explorer. Voici une liste non exhaustive d'exemples.

— *Réglage direct de l'espace des hypothèses*. On citera par exemple les techniques permettant de contrôler automatiquement l'architecture des réseaux connexionnistes, soit en ajoutant des neurones formels et leurs connexions (e.g. *cascade correlation*), soit au contraire en retirant des connexions ou des neurones (e.g. *optimal brain damage*).

— *Réglage indirect de l'espace des hypothèses*. Les techniques de régularisation qui contrôlent un compromis entre l'adéquation aux données et la régularité des hypothèses considérées peuvent également faire l'objet d'une procédure automatique de réglage du compromis. Par ailleurs, la régularisation elle-même peut jouer sur différentes caractéristiques de l'espace des hypothèses. Souvent, on caractérise la régularité d'une hypothèse par la norme ℓ_2 appliquée à ses paramètres, l'idée étant de pénaliser les hypothèses peu « lisses »[4]. Récemment, les recherches se sont tournées vers la norme ℓ_1 qui permet de favoriser les hypothèses parcimonieuses, c'est-à-dire mettant en jeu

4. Une norme ℓ_p appliquée à une fonction paramètrée h s'exprime par : $\ell_p(h) = \left\{\sum_{i=1}^{n} w_i^p\right\}^{1/p}$, où les w_i sont les paramètres, supposés ici en nombre fini, de h.

peu de paramètres. Dans le cas de modèles linéaires, cela se traduit par la préférence pour des modèles s'exprimant à l'aide de peu de variables descriptives. Cela permet une sélection des attributs les plus pertinents ou informatifs pour la tâche considérée. Les méthodes d'élagage d'arbres de décision peuvent être considérées comme appartenant à cette famille de méthodes. De même que les techniques dites de *weight decay* dans les réseaux connexionnistes.

— *Contrôle du processus d'exploration de l'espace des hypothèses*. Toujours dans le but de contrôler le risque de sur-apprentissage, certaines techniques limitent le processus de recherche de la meilleure hypothèse $\hat{h}$, par exemple en arrêtant l'optimisation avant que ne soit atteint un optimum. L'idée sous-jacente est qu'ainsi on empêche l'algorithme de capturer des régularités accidentelles, propres à l'échantillon de données disponibles mais non représentatives des vraies régularités du monde. On citera la *règle arrêt prématuré* utilisée dans les réseaux connexionnistes.

— Dans les méthodes à base de comparaison à des exemples d'apprentissage (voir section 10.2.2), ce sont les fonctions noyau ou les distances utilisées qui constituent les principaux méta-paramètres à régler. Il existe désormais des approches pour apprendre automatiquement les fonctions noyau prises dans un espace de fonctions disponibles. Ces techniques doivent alors compenser l'absence de connaissances expertes par une taille d'échantillon d'apprentissage plus importante.

Le méta-apprentissage peut également consister à sélectionner automatiquement le meilleur algorithme d'apprentissage pour un type d'application donné. Le méta-apprentissage s'attaque alors directement au « no-free lunch theorem » en cherchant à trouver pour chaque classe de problèmes les algorithmes les mieux adaptés. Inversement, on peut aussi chercher à déterminer la « carte de compétences » de chaque algorithme. Étant donné alors un problème, un algorithme maître décide quelle est la meilleure méthode pour le résoudre. Si cette approche est séduisante, il faut reconnaître que les efforts passés, parfois au niveau de programmes européens, n'ont pas sensiblement modifié l'activité des praticiens de l'apprentissage artificiel qui continuent à faire appel à leur expérience (Giroud-Carrier *et al.*, 2004).

10.5 Conclusion

Nous avons décrit jusqu'ici le cadre classique de l'apprentissage artificiel. Il se caractérise par le fait qu'il est davantage orienté vers la capacité de prédiction que par celle de compréhension. L'accent est donc mis sur la production d'hypothèses ou de modèles du monde qui sont souvent des fonctions, d'un espace d'observations à un espace d'étiquettes, et qui sont assez peu intégrées à une théorie du domaine. Par ailleurs, dans ce cadre, les données appartiennent généralement à un espace vectoriel, c'est-à-dire qu'elles sont décrites par un nombre limité et fixe d'attributs. Les sorties ou étiquettes quant à elles sont souvent unidimensionnelles : une classe ou un réel. Finalement, le cadre classique repose fondamentalement sur l'hypothèse que les données sont tirées aléatoirement à partir d'une distribution stationnaire. C'est ce qui permet en effet de donner une expression à la performance visée sous la forme du risque réel, qui est une espérance. C'est aussi ce qui permet de recourir à des théorèmes centraux limite pour déterminer les performances attendues des algorithmes mis en jeu.

Depuis une dizaine d'années, un nombre croissant de champs d'applications (génomique, analyse de texte, recherche d'information sur le web, étude de réseaux sociaux, ...) ne rentrent pas

dans le cadre classique rendant nécessaire l'émergence de nouvelles directions de recherche. On peut les catégoriser en fonction des caractéristiques dépassant le cadre classique :

— *Données non indépendantes et identiquement distribuées*. C'est le cas de l'analyse de sources séquentielles de données provenant par exemple de mesures qui ont une corrélation temporelle, voire spatiale dans les nouvelles applications environnementales. De plus, ces séquences peuvent être issues de *processus non stationnaires*. On parle alors de dérive de concept. L'apprentissage ne peut alors plus se faire en une fois, à partir d'une base d'exemples constituée. Il faut avoir recours à des algorithmes d'apprentissage en-ligne. L'une des questions majeures est de déterminer ce qui doit être mémorisé et ce qui peut être oublié tandis que le flux de données est traité, parfois en temps réel. La complexité des algorithmes devient alors un souci majeur. Tandis que la communauté de l'apprentissage artificiel parle d'apprentissage en-ligne et de ces variantes (apprentissage incrémental, apprentissage et dérive de concept, ...), la communauté des bases de données et de la fouille de données étudie la fouille à partir de « flux de données ». (Voir (Quinonero-Candela *et al.*, 2009; Sugiyama et Kawanabe, 2012; Gama et Gaber, 2007; Gama, 2010))
— *Données semi-supervisées*. L'association d'une étiquette à chaque observation est souvent une opération coûteuse, requérant par exemple un expert. Un cas trivial est celui de la détection de spam ou pourriels. L'étiquetage des courriels nécessite l'attention de l'utilisateur. Sachant qu'à côté des exemples ainsi étiquetés il existe naturellement une masse de données non étiquetées (e.g. les courriels reçus mais non étiquetés), est-il possible d'en profiter pour aider ou affiner l'apprentissage supervisé ? C'est à ce problème que s'attaque l'apprentissage semi-supervisé. On peut dire grossièrement que si de bons a priori existent sur la forme de la distribution des données, la présence de données non étiquetées peut être bénéfique. En revanche, des apriori erronés peuvent conduire à une dégradation des résultats de l'apprentissage (Cornuéjols et Miclet, 2010).
— L'apprentissage peut avoir comme objectif de *calculer en sortie une structure*, comme par exemple une molécule, un arbre de dérivation grammaticale ou encore la structure typique d'un réseau. Il devient alors intéressant, voire nécessaire, de tirer parti d'une notion de distance ou de similarité dans cet espace de sortie. Dans certains cas, on utilise aussi une distance définie sur l'espace combiné des entrées et des sorties. Si les algorithmes ne sont pas forcément différents, l'apprentissage de sorties structurées introduit cependant tout un ensemble de nouveaux problèmes, mais aussi de nouvelles possibilités.
— Un nombre croissant d'applications implique de calculer en sortie non pas une valeur, par exemple l'étiquette associée à une entrée, mais de *produire un classement* (aussi appelé ordonnancement) de l'ensemble des données fournies en entrée. Ainsi, étant donné le profil d'un utilisateur on cherchera à classer les films ou les ouvrages les plus susceptibles de l'intéresser. Les notions classiques de fonctions de perte et de risque empirique ne peuvent alors plus être utilisées directement. Toute une nouvelle classe d'algorithmes est ainsi développée pour répondre à ces nouveaux objectifs.
— L'un des secteurs de recherche les plus actifs récemment concerne le *graph mining*, c'est-à-dire l'exploitation de données relatives à des graphes. Certaines applications visent à caractériser les nœuds d'un graphe, par exemple à apprendre le concept d'auteur influant ou d'amplificateur de rumeur. D'autres applications se focalisent sur

la comparaison de graphes, par exemple des graphes d'interaction cellulaires. Dans tous les cas, de nouvelles techniques de comparaison de données et de données avec les hypothèses demandent à être mises au point.

— L'une des limites de la logique classique est son incapacité à représenter l'incertitude et à permettre le raisonnement incertain. Le calcul des probabilités offre de son côté un cadre rigoureux pour le raisonnement incertain, mais il est limité aux concepts propositionnels. Depuis plusieurs années, en particulier dans le cadre de l'apprentissage relationnel statistique (*Statistical Relational Learning*), des efforts de recherche s'efforcent de marier au mieux les possibilités des deux approches, et de dépasser leurs limites. L'une des premières questions est de définir la notion de « couverture » d'un exemple par une hypothèse. Dans le cadre de la logique, cette notion dépend d'une procédure de preuve qui renvoie un booléen `vrai` ou `faux`. Dans le cadre du calcul probabiliste, la notion de couverture devient probabiliste, renvoyant une valeur de probabilité.

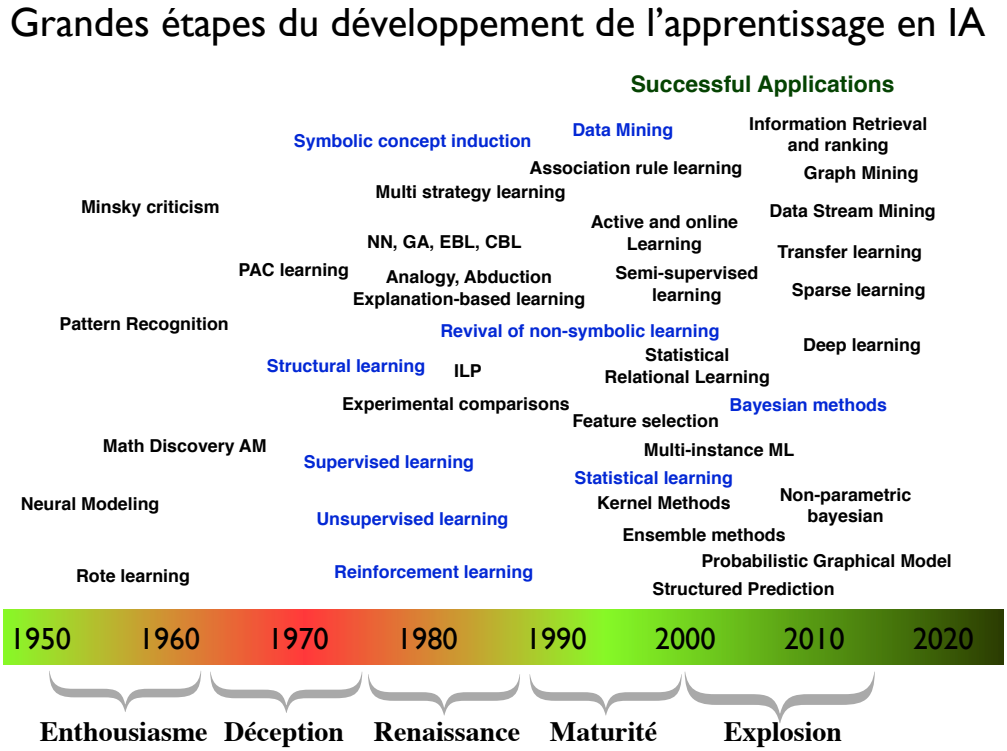

FIGURE 8: *Esquisse d'une fresque historique présentant diverses techniques d'apprentissage artificiel avec leur date approximative d'apparition.*

Finalement, il faut noter que le cadre classique de l'apprentissage artificiel a presque toujours considéré que les problèmes d'apprentissage étaient isolés. Pour chaque problème, on développait un nouveau système d'apprentissage et on réalisait un nouvel apprentissage indépendamment de ce qui avait été réalisé ailleurs. Mais pour apprendre à jouer au Go, peut-on

profiter d'avoir appris à jouer aux dames, ou, au contraire, faut-il l'éviter ? Les travaux sur le raisonnement par analogie portaient en germe ce genre de questions. Le nouveau courant sur l'*apprentissage par transfert* les explore dans le cadre de l'induction. Cela va modifier les algorithmes d'apprentissage par l'intégration de connaissances passées et le problème de leur utilisation et de leur traduction pour faire face à un nouveau contexte.

Pour ceux qui souhaitent tester ou utiliser des algorithmes d'apprentissage, il existe de nombreux logiciels spécialisés, dont certains dans le domaine public. La référence (Harrington, 2012) décrit plusieurs algorithmes classiques d'apprentissage artificiel en détaillant leur code en Python. Un ouvrage d'approche similaire avec le code en R est (Conway et White, 2012). La « boîte à outils » Weka est décrite dans (Witten *et al.*, 2011).

Ce chapitre a décrit les grandes classes d'approches algorithmiques existantes. Il ne pouvait être question d'énumérer et de présenter l'ensemble des algorithmes développés au cours des six dernières décennies. Durant cette période, l'accent s'est progressivement déplacé d'approches algorithmiques et symboliques très inspirées par l'étude de la cognition naturelle à des approches de nature bien davantage statistique lorsque la pression des applications a mis en exergue l'exploitation de bases de données décrites en attribut-valeur et supposées indépendantes et identiquement distribuées. Ne pouvant rendre compte en détail de cette évolution, nous terminons cependant par la fresque de la figure 8 évoquant un certain nombre de techniques et leurs dates d'apparition approximatives.

Références

AGRAWAL, R. et SRIKANT, R. (1994). Fast algorithms for mining association rules in large databases. *In Very Large Data Bases (VLDB-94)*, pages 487–499, Santiage, Chile.

ANDERSON, J. (1995). *The architecture of cognition*, volume 5. Lawrence Erlbaum.

ANDERSON, J., KLINE, P. et BEASLEY, C. (1979). A general learning theory and its application to schema abstraction. *Psychology of learning and motivation*, 13:277–318.

BARBER, D. (2012). *Bayesian Reasoning and Machine Learning*. Cambridge University Press.

BASTIDE, Y., PASQUIER, N., TAOUIL, R., STUMME, G. et LAKHAL, L. (2000). Mining minimal non-redundant association rules using frequent closed itemsets. *In Computational Logic*, pages 972–986.

BENGIO, Y. et CUN, Y. L. (2007). Scaling learning algorithms towards ai. *In* BOTTOU, L., CHAPELLE, O., DECOSTE, D. et WESTON, J., éditeurs : *Large-Scale Kernel Machines*, page 416. MIT Press.

BISHOP, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer-Verlag, Secaucus, NJ, USA.

BLUM, A. et MITCHELL, T. (1998). Combining labeled and unlabeled data with co-training. *In 11th Computational Learning Theory (COLT-98)*, pages 92–100. Morgan Kaufmann.

BREIMAN, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

BREIMAN, L., FRIEDMAN, J., OLSHEN, R. et STONE, C. J. (1984). *Classification and regression trees*. Wadsworth and Brooks/Cole Advanced Books and Software.

CONWAY, D. et WHITE, J. M. (2012). *Machine Learning for Hackers*. O'Reilly.

CORMEN, T., LEISERSON, C., RIVEST, R. et STEIN, C. (2001). *Introduction to algorithms*. MIT press.

CORNUÉJOLS, A. et MICLET, L. (2010). *Apprentissage artificiel. Concepts et algorithmes. (2ème édition)*. Eyrolles.

de la HIGUERA, C. (2010). *Grammatical inference: learning automata and grammars*. Cambridge University Press.

DREYFUS, G., MARTINEZ, J.-M., SAMUELIDES, M., GORDON, M., BADRAN, F. et THIRIA, S., éditeurs (2008). *Apprentissage statistique.* Eyrolles.

DZEROSKI, S. et LAVRAC, N., éditeurs (2001). *Relational data miniing*. Springer.

FREUND, Y. et SCHAPIRE, R. (1996). Experiments with a new boosting algorithm. *In Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156.

FÜRNKRANZ, J., GAMBERGER, D. et LAVRAC, N. (2012). *Foundations of Rule Learning*. Springer.

GAMA, J. (2010). *Knowledge Discovery from Data Streams*. Chapman & Hall.

GAMA, J. et GABER, M. M., éditeurs (2007). *Learning from Data Streams. Processing Techniques in Sensor Networks*. Springer.

GANTER, B., STUMME, G. et WILLE, R., éditeurs (2005). *Formal Concept Analysis: Foundations and Applications*. Springer.

GANTER, B., WILLE, R. et FRANKE, C. (1998). *Formal Concept Analysis: Mathematical Foundations*. Springer.

GETOOR, L. et TASKAR, B., éditeurs (2007). *An introduction to statistical relational learning*. MIT Press.

GIROUD-CARRIER, C., VILALTA, R. et BRAZDIL, P. (2004). Special issue on meta-learning. *Machine Learning journal*, 54.

HAN, J. et KAMBER, M. (2006). *Data mining. Concepts and techniques (2nd Ed.)*. Morgan Kaufmann.

HAN, J., PEI, J., YIN, Y. et MAO, R. (2004). Mining frequent patterns without candidate generation : A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87.

HARRINGTON, P. (2012). *Machine Learning in Action*. Manning.

HASTIE, T., TIBSHIRANI, R. et FRIEDMAN, J. (2009). *The elements of statistical learning. Data mining, inference, and prediction*. Springer.

JAPKOWICZ, N. (2011). *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press.

JEBARA, T. (2003). *Machine Learning: Discriminative and Generative*. Springer.

KELLEY, C. (1987). *Iterative methods for optimization*, volume 18. Society for Industrial Mathematics.

KODRATOFF, Y. et MICHALSKI, R. S. (1990). *Machine learning: an artificial intelligence approach*, volume 3. Morgan Kaufmann Publishers.

KOLLER, D. et FRIEDMAN, N. (2009). *Probabilistic Graphical Models. Principles and Techniques*. MIP Press.

KOTSIANTIS, S. B. (2007). Supervised machine learning: A review of classification tech-

niques. *Informatica*, 31:249–268.

LAVRAC, N. et DZEROSKI, S. (1994). *Inductive logic programming - techniques and applications*. Ellis Horwood series in artificial intelligence. Ellis Horwood.

MICHALSKI, R., CARBONEL, J., MITCHELL, T. et KODRATOFF, Y. (1993). Apprentissage symbolique, une approche de l'intelligence artificielle. *Volumes I and II, Cepadues Editions*.

MICHALSKI, R., CARBONELL, J. et MITCHELL, T. (1986). *Machine learning: An artificial intelligence approach (vol. 1 and 2)*, volume 1 and 2. Morgan Kaufmann.

MICLET, L. (1990). Grammatical inference. *In* BUNKE, H. et SANFELIU, A., éditeurs : *Syntactic and structural pattern recognition theory and applications*. World Scientific.

MINSKY, M. et PAPERT, S. (1988). *Perceptrons (2nd ed.)*. MIT Press.

MITCHELL, T. (1979). *Version spaces: An approach to concept learning*. Ph.d. thesis, Stanford University.

MITCHELL, T. (1982). Generalization as search. *Artificial Intelligence journal*, 18:203–226.

MITCHELL, T. (1997). *Machine Learning*. McGraw-Hill.

MUGGLETON, S. (1995). Inverse entailment and progol. *New Generation Comput.*, 13(3&4): 245–286.

NILSSON, N. (2010). *The quest for artificial intelligence. A history of ideas and achievements*. Cambridge University Press.

PEARL, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

PLOTKIN, G. (1970). A note on inductive generalization. *In Machine Intelligence*, volume 5, pages 153–163. Edinburgh University Press.

QUINLAN, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kauffman.

QUINLAN, J. R. (1996). Learning first-order definitions of functions. *CoRR*, cs.AI/9610102.

QUINONERO-CANDELA, J., SUGIYAMA, M., SCHWAIGHOFER, A. et LAWRENCE, N., éditeurs (2009). *Dataset shift in machine learning*. MIT Press.

RAEDT, L. D. (2008). *Logical and Relational Learning*. Springer.

RAEDT, L. D., FRASCONI, P., KERSTING, K. et MUGGLETON, S., éditeurs (2008). *Probabilistic Inductive Logic Programming - Theory and Applications*, volume 4911 de *Lecture Notes in Computer Science*. Springer.

ROSENBLATT, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

SAITTA, L., GIORDANA, A. et CORNUÉJOLS, A. (2011). *Phase Transitions in Machine Learning*. Cambridge University Press.

SCHAPIRE, R. (2003). The boosting approach to machine learning: An overview. *In* DENISON, D. D., HANSEN, M. H., HOLMES, C., MALLICK, B. et YU, B., éditeurs : *Nonlinear Estimation and Classification*. Springer.

SCHÖLKHOPF, B. et SMOLA, A. (2002). *Learning with kernels*. MIT Press.

SHAPIRE, R. et FREUND, Y. (2012). *Boosting: Foundations and Algorithms*. MIT Press.

SHAWE-TAYLOR, J. et CRISTIANINI, N. (2004). *Kernel methods for pattern analysis*. Cambridge University Press.

SNYMAN, J. (2005). *Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms*, volume 97. Springer.

SUGIYAMA, M. et KAWANABE, M. (2012). *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. MIT Press.

SURAJ, Z. (2004). An introduction to rough sets theory and its applications: A tutorial. *In ICENCO'2004*, Cairo, Egypt.

VAPNIK, V. (1995). *The nature of statistical learning theory*. Springer.

WINSTON, P. (1970). Learning structural descriptions from examples. Rapport technique, DTIC Document.

WITTEN, I., FRANK, E. et HALL, M. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann.

WOLPERT, D. (1996a). The existence of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1391–1420.

WOLPERT, D. (1996b). The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390.

WOOLFOLK, A. (2001). *Educational Psychology*. Allyn and Bacon.

XU, R. et WUNSCH, D. (2008). *Clustering*. Wiley-IEEE Press.

ZAKI, M. J. (2000). Generating non-redundant association rules. *In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, August 20-23, KDD*, pages 34–43.

ZELEZNÝ, F. et LAVRAC, N. (2006). Propositionalization-based relational subgroup discovery with rsd. *Machine Learning*, 62(1-2):33–63.

ZOU, Z.-H. (2012). *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall.