L'industrialisation des analyses – Besoins, outils & applications

Françoise Fogelman-Soulié, Erik Marcadé

KXEN, 25 quai Galliéni, 92 158 SURESNES Cedex, France

Francoise@kxen.com, Erik.Marcade@kxen.com

Résumé. Le data mining est aujourd'hui de plus en plus utilisé dans les entreprises les plus compétitives. Ce développement, rendu possible par la disponibilité grandissante de masses de données importantes, pose des contraintes tant théoriques (quels algorithmes utiliser pour produire des modèles d'analyses exploitant des milliers de variables pour des millions d'exemples) qu'opérationnelles (comment mettre en production et contrôler le bon fonctionnement de centaines de modèles). Je présenterai ces contraintes issues des besoins des entreprises ; je montrerai comment exploiter des résultats théoriques (provenant des travaux de Vladimir Vapnik) pour produire des modèles robustes; je donnerai des exemples d'applications réelles en gestion de la relation client. Nous verrons ainsi comment il est possible d'industrialiser le data mining et en faire ainsi un composant facilement exploitable dès qu'on dispose de données.

Abstract. Today data mining is more and more extensively used by very competitive enterprises. This development, brought by the increasing availability of massive datasets, is only possible if challenges, both theoretic and operational, are met : which algorithms should be used to produce models when datasets have thousands of variables and millions of observations; how to run and control the correct execution of hundreds of models. I will present these constraints in industrial contexts; I will show how to exploit theoretical results (coming from Vapnik's work) to produce robust models; I will give examples of real-life applications in customer relationship management. I will thus demonstrate that it is indeed possible to industrialize data mining so as to turn it into an easy-to-use component whenever data is available.

Mots-clés. Data Mining. Robustesse. Passage à l'échelle. Text Mining.

1 Le data mining

1.1 Un peu d'histoire

Le data mining est une discipline qui a émergé progressivement de la convergence de plusieurs domaines : de 1900 à 1990, la *statistique* (Fisher, Cramer, Bayes, Kolmogorov-Smirnoff…) ; de 1940 à 1970, *cybernétique* (Wiener et von Neumann, perceptron de Rosenblatt, Minsky et Papert) ; de 1970 à 1990, *machine learning* : intelligence artificielle, reconnaissance des formes, arbres de décision (Breiman, Friedman) et réseaux de neurones (Hopfield, Kohonen, Rumelhart, LeCun, …), théorie statistique de l'apprentissage (Vapnik) : lors de l'Ecole Modulad de 1996 [5], nous avions montré les liens étroits entre statistique et réseaux de neurones. Depuis, le développement du data mining n'a fait que s'amplifier, grâce aussi à l'informatique qui fournit les moyens matériels indispensables aux traitements de grandes masses de données (ordinateurs rapides, mémoire, disques durs, bases de données).

Nous montrerons ici comment nous avons exploité les travaux théoriques (Statistical Learning Theory, Structural Risk Minimization) de Vladimir Vapnik (arrivé aux Bell Labs en 1991) pour mettre en œuvre une solution industrielle de data mining.

1.2 Le data mining … aujourd'hui

Aujourd'hui, le data mining dispose de nombreuses techniques [8] permettant de traiter des problèmes de grande taille : cependant, en pratique, les entreprises restent souvent incapables d'exploiter l'intégralité de leurs données en produisant tous les modèles nécessaires [9], ce qui s'explique par la faible productivité des outils actuels, utilisables seulement par un expert [10], imposant, selon Gartner, un mode de développement « artisanal » des applications data mining (Figure 1-1).

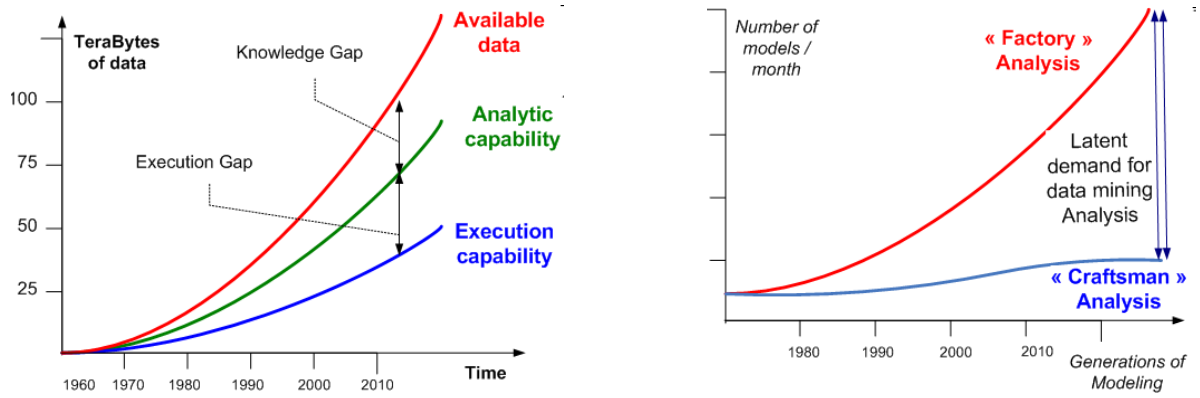


Figure 1-1 – Analyses du Gartner (d'après [9] à gauche et [10] à droite)

2 Les besoins

2.1 Le constat

Les sources de données sont aujourd'hui très nombreuses et les volumes générés croissent exponentiellement : par exemple, le Web génère chez Yahoo! environ 16 milliards de transactions par jour, soit 10 Tera Octets [4] ; les tags RFID, bientôt attachés à tous les produits, génèrent déjà aujourd'hui chez un grand distributeur (3 000 magasins) de l'ordre de 300 millions d'événements par jour [14] ; les réseaux sociaux ont des millions de nœuds : 4,4 sur la communauté blog LiveJournal et 240 sur Microsoft Instant Messenger [15] ; les réseaux de téléphonie mobile génèrent des centaines de millions de données d'appel (CDR) par jour et des millions de données techniques, par exemple 40 millions par jour avec 1000 variables sur une seule grande ville américaine [7] … Cependant, la constitution d'une base de données représente un coût très important : intégrer l'ensemble des données reste un problème difficile, car les sources sont nombreuses, et les formats hétérogènes, non cohérents.

Par ailleurs, les utilisateurs souhaitent de plus en plus maîtriser les outils leur permettant de répondre par eux-mêmes à leurs questions, sans dépendre d'experts (ni être obligés d'en devenir eux-mêmes !) Les besoins d'analyses augmentent, parce que les décisions – opérationnelles ou stratégiques – que doivent prendre les utilisateurs sont de plus en plus nombreuses ; évidemment, la qualité des décisions prises dépend des analyses menées et la vitesse à laquelle on peut produire l'analyse est un facteur clé pour la qualité perçue par l'utilisateur.

Comme on le voit sur la Figure 1-1, le volume des données augmente exponentiellement, le nombre de modèles devrait augmenter de la même manière de façon à aider à fournir la bonne décision pour chaque action entreprise : si ceci n'est pas possible, les gaps indiqués représentent une perte significative pour l'entreprise. La mise en œuvre industrielle du data mining doit donc viser à réduire ces gaps.

2.2 Les données

Comme on l'a vu, les sources de données sont nombreuses dans l'entreprise et l'effort nécessaire pour construire un datawarehouse les intégrant est très significatif. On peut toutefois commencer à travailler les données disponibles sans avoir encore consolidé un datawarehouse global, en exploitant les bases thématiques disponibles, en acquérant éventuellement des données extérieures (données INSEE, géo-marketing, comportementales …) Les analyses permettront alors d'obtenir des résultats exploitables rapidement, d'avoir une première évaluation de la qualité des données disponibles et de la valeur des données externes. On peut ainsi constituer un *business case* étayé par des premiers bénéfices : l'entreprise peut alors valider l'utilité de collecter et stocker les variables utilisées dans les analyses et établir une évaluation des bénéfices apportés par chaque variable.

Bien entendu, les données sont critiques pour toute application de data mining : pas de données, pas de modèle !

2.2.1 Collecte des données

Le processus de collecte de données est complexe : il faut identifier l'ensemble des sources, mettre en place les mécanismes de collecte, définir un référentiel et des règles de gestion pour mettre les données en cohérence, manipuler et transformer les données pour constituer le fichier sur lequel seront réalisées les analyses et qu'on appelle l'*Analytical Data Set* – ADS : la Figure 2-1 illustre ce processus dans le cas d'analyses de connaissance client. Comme on l'a vu, on peut constituer un ADS sans passer par un datawarehouse ; ce sera toutefois indispensable, à terme, si les besoins d'analyses sont très importants.

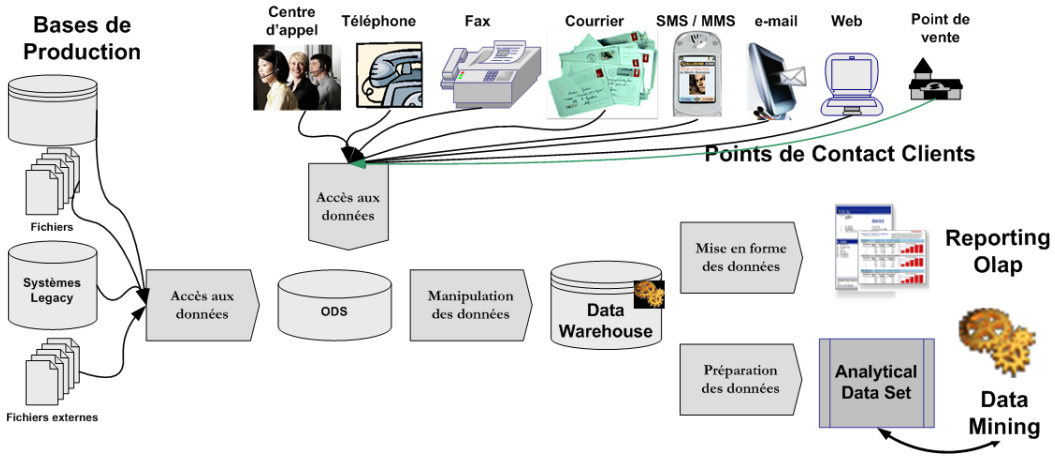


Figure 2-1 – La collecte des données

2.2.2 Préparation des données

Le datawarehouse est la plupart du temps une base multi-domaine, contenant un historique de l'ensemble des données de l'entreprise. Pour réaliser des analyses data mining, on doit d'abord choisir dans le datawarehouse les variables qu'on souhaite intégrer dans l'ADS. On réalise aussi souvent des transformations « métier » : champs calculés (nouvelles variables calculées à partir de variables existantes, comme par exemple des agrégats, le nombre de jours entre l'émission de la facture et le paiement, le profit : prix d'achat - coût de fabrication …). On peut ensuite étudier la qualité des données : leurs distributions, les valeurs manquantes (blancs, espaces, nuls), les outliers, les corrélations … Ensuite, selon les algorithmes et les outils utilisés, on doit coder les variables (recodage des variables continues, catégoriques …).

2.2.3 Qualité des données

La qualité des données doit être aussi bonne que possible : données exactes, non redondantes, complètes … Cependant, en pratique, la qualité des données n'est jamais parfaite : les valeurs remontées ne sont pas correctes (erreurs de saisie), il y a des doublons, des données manquantes … Une bonne technique data mining devra donc être robuste par rapport à ces problèmes de qualité.

Le traitement des données manquantes, en particulier, peut se faire de plusieurs façons : en éliminant toutes les observations non remplies complètement (on risque évidemment d'en éliminer beaucoup !), ou en remplaçant les données manquantes en *imputant* des valeurs estimées (catégorie la plus fréquente, moyenne), ce qui pose le problème des valeurs qui ne sont pas MAR (*missing at random*) ; c'est pourquoi KXEN crée une valeur spéciale KxMissing.

2.2.4 Type de données

On utilise aujourd'hui principalement les *données structurées* disponibles dans les bases de données, les fichiers plats, les fichiers Excel … Cependant, de plus en plus, on disposera de sources de *données non structurées* : texte (pages web, SMS, emails, flux RSS, …), voire multimedia (video, MMS, musique). Le volume des données non structurées représente déjà plus de 50% des données disponibles dans l'entreprise et pourtant ces données restent peu intégrées au datawarehouse [16]. On doit donc envisager des techniques data mining permettant de prendre en compte ces données non structurées au même titre que les données structurées.

Par ailleurs, la plupart des analyses aujourd'hui commence par extraire les données de la base de données en « mettant à plat » variables et observations dans un fichier d'analyse, l'ADS (Figure 2-1). On perd donc la *structure* qui pouvait exister entre les variables. On n'exploite pas non plus la structure des observations : par exemple, le *réseau social* des interactions entre clients est implicite dans les données de communication que recueille l'opérateur téléphonique ; toutefois ces informations ne sont généralement aujourd'hui pas extraites et intégrées dans les analyses data mining : les travaux existants [11] montrent cependant que ces données ont un impact très significatif sur les performances des modèles.

2.3 Le cycle de vie des études

Aujourd'hui le cycle de vie d'une étude data mining (Figure 2-2) permet en moyenne de répondre au besoin en 6 semaines. Pour beaucoup d'applications, cette durée est trop grande, encore plus si on doit produire des centaines de modèles. Il faut donc essayer de raccourcir ce cycle. Pour cela, on peut rapprocher les équipes (en regroupant les compétences IT et études au sein du département marketing) ; industrialiser la mise à disposition des données (datawarehouse d'entreprise, accès par les utilisateurs à des « vues » métier) ; simplifier le processus études (automatisation des analyses simples, utilisation d'outils orientés « utilisateurs » et pas seulement « statisticien »).

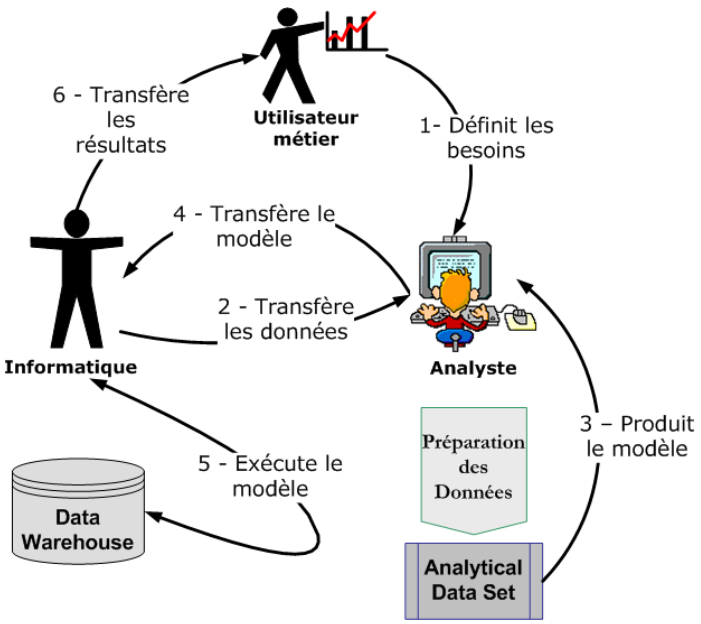


Figure 2-2 – Cycle de vie des études

Dans ce dernier cas, le statisticien devient l'expert de référence (il intervient pour traiter les problèmes complexes, valider les résultats critiques pour l'entreprise) et l'utilisateur métier exécute tout seul les analyses dont il a besoin, quand il en a besoin .

2.4 Qu'apportent les analyses ?

Les analyses data mining permettent de répondre à de très nombreuses questions et ainsi gagner en efficacité. Par exemple, si on veut lancer une campagne marketing, on fera un modèle qui permettra de réduire la taille de la cible (et donc les coûts) par rapport à une campagne non ciblée, tout en augmentant le nombre de réponses (voir un exemple §4). On peut ainsi être amené à devoir développer un grand nombre de modèles. Ainsi, par exemple, dans l'assurance, si on a des clients sur 20 régions géographiques, répartis en 10 segments clients et 20 produits, et qu'on souhaite réaliser 10 types de scores : appétence (nouveaux clients, cross-selling et up-selling), attrition (remboursement anticipé, refinancement, fin de contrat), fraude (déclaration de sinistre, souscription), performances commerciales (prévision des performances), satisfaction client … On aboutit ainsi à 20 x 10 x 20 x 10 = 40 000 modèles *fins*. La capacité de faire des modèles fins permet d'augmenter la performance des modèles (chaque sous-population est homogène), de réaliser des cibles plus réduites (on est donc plus pertinent dans le message, qui est mieux personnalisé), de réduire la pression commerciale, de réduire les coûts (les volumes étant plus petits, les temps de calculs sont réduits, les coûts de l'opération marketing plus faibles et la logistique des opérations simplifiée). Par exemple, Vodafone D2 réalise aujourd'hui, pour ses campagnes marketing, 700 modèles par an (sur une base Teradata) [6].

Aujourd'hui, la plupart des entreprises ont un « gap » dans leur capacité à produire des modèles ou à lancer des actions les exploitant (Figure 1-1). Cependant, la capacité à produire de façon industrielle des analyses data mining est un facteur de compétitivité majeur : ainsi, T. Davenport [3] montre comment les entreprises qui sont des « analytical competitors » sont aussi celles qui sont leaders sur leur marché. Les entreprises doivent donc, pour assurer leur compétitivité, mettre en œuvre un processus de production industrielle d'analyses data mining (la société Netflix citée par Davenport produit par exemple 1 Milliard de prévisions par jour !)

2.5 La vitesse

La vitesse est un facteur clé de performance. Un délai réduit pour produire un modèle (depuis la conception à la mise en production) permet d'améliorer la productivité des équipes (produire un modèle en 2 jours au lieu de 6 semaines permet de faire plus de modèles) ; améliorer les performances (les données utilisées pour la modélisation sont récentes, le marché n'a pas changé et la performance du modèle en production est celle attendue par le modèle) ; améliorer le « time-to-market » (c'est-à-dire que la réactivité à une offre de la concurrence est plus rapide). C'est en fait souvent cette réduction du time-to-market qui est considéré par les entreprises comme le gain majeur en productivité.

2.6 L'usine à modèles

L'usine à modèles (cf Figure 1-1), c'est *la capacité de traiter des masses de données* (10-100 millions de clients, 5 000 variables) … ce qui demande un algorithme linéaire (ou presque), une manipulation des données minimum, avec seulement quelques passes pour lire les données sans duplication ; *la capacité de produire des masses de projets* (100-1000 projets par an, semaine, jour)… ce qui demande la possibilité d'automatiser la réalisation du modèle, la facilité à exporter / intégrer le modèle en production ; *la capacité de produire les modèles très rapidement* (en quelques jours ou heures) … ce qui demande un outil convivial (utilisable par des utilisateurs métier), avec automatisation des tâches lourdes (codage des données, sélection des algorithmes, exécution du modèle par exemple directement dans la base de données) ; *la capacité de produire des modèles «automatiquement »* (industrialiser la production, l'export et l'exécution du modèle) … ce qui demande d'automatiser le codage des

variables, de disposer d'un langage de script et de pouvoir exporter vers tous formats ; *l'efficacité sur la manipulation des données* (gros volumes, éventuellement dispersés). L'usine à modèles permet alors d'augmenter la productivité (plus de modèles, produits plus vite par moins de personnes moins qualifiées) ; d'augmenter les bénéfices (on peut faire des modèles pour chaque problème … même ceux pour lesquels on n'avait pas le temps) ; d'augmenter la vitesse (« time-to-market » réduit, données plus récentes).

Nous allons maintenant montrer comment nous avons pu, en nous appuyant sur les théories de Vapnik, développer un outil permettant de mettre en œuvre des usines à modèles.

3 La mise en œuvre de KXEN

3.1 Le cadre mathématique

3.1.1 Les notations

On dispose de données d'apprentissage $(x^1, y^1), (x^2, y^2), ..., (x^n, y^n)$ où $x^i = (x_1^i, x_2^i, ..., x_p^i)$ est une observation et y^i est l'étiquette ou la cible associée. y peut être une variable discrète (classification) ou continue (régression). Nous ferons l'hypothèse que, dans la « base d'apprentissage », tous les y^i sont connus. Les (x^i, y^i) sont supposés être un échantillon de tirages i.i.d. issus d'une distribution fixe mais inconnue $P(X,Y)$

On se donne une classe de fonctions $\Phi_\theta = \{f(.,W,\theta), W \in \aleph\}$. Par exemple la classe des polynômes de degré θ, la classe des MLP (réseaux de neurones multi-couches) avec θ neurones cachés. Un modèle issu de cette classe produit donc pour chaque observation x une sortie $y = f(x,W,\theta)$

A partir des données $(x^1, y^1), (x^2, y^2), ..., (x^n, y^n)$, on cherche le « meilleur » modèle $\hat{y} = f(x,\hat{W},\theta)$, produit par un certain algorithme ou principe d'inférence et qui correspond donc au « meilleur » paramètre $\hat{W}$.

3.1.2 Coût et risque

On se donne une fonction de perte $L[y, f(x,W,\theta)]$ qui mesure le coût qu'il y a à remplacer la vraie valeur y par la valeur calculée $f(x,W,\theta)$. Par exemple, on utilise classiquement comme coût l'écart quadratique $L[y, f(x,W,\theta)] = [y - f(x,W,\theta)]^2$ (1)

L'erreur en apprentissage ou *risque empirique* est alors défini comme le coût moyen sur l'ensemble d'apprentissage $R_{emp}(W,\theta) = \frac{1}{n}\sum_{i=1}^{n} L[y^i, f(x^i,W,\theta)]$ (2)

Dans le cas du coût quadratique, le risque empirique est l'écart quadratique moyen MSE (Mean Square Error) $R_{emp}(W,\theta) = \frac{1}{n}\sum_{i=1}^{n} [y^i - f(x^i,W,\theta)]^2$ (3)

L'*erreur en généralisation* est définie par $R_{Gen}(W,\theta) = \int L[y, f(x,W,\theta)] \cdot dP(x,y)$ (4)

C'est le coût moyen théorique sur l'ensemble de la population, c'est à dire l'erreur attendue sur de nouvelles données.

Notre principe d'inférence est la minimisation du risque empirique (*Empirical Risk Minimization* ou ERM)

$$\hat{W}_\theta = \arg \min_W R_{emp}(W,\theta) \quad (5)$$

Dans le cas du risque quadratique, le principe ERM n'est autre que la règle des moindres carrés LMSE (*Least Mean Square Error*)

$$\hat{W}_\theta = arg \min_W \frac{1}{n}\sum_{i=1}^{n}\left[y^i - f\left(x^i,W,\theta\right)\right]^2 \quad (6)$$

Cette étape ERM permet de déterminer le meilleur $\hat{W}$ (*data fit*), elle ne dit pas comment on choisit θ.

3.1.3 Qu'attendons-nous d'un modèle

A partir d'un ensemble d'apprentissage $\left(x^1,y^1\right),\left(x^2,y^2\right),...,\left(x^n,y^n\right)$, on vient de voir que le principe d'inférence ERM consistait à minimiser le risque empirique, c'est-à-dire de maximiser la précision sur l'ensemble d'apprentissage. Il existe évidemment beaucoup de modèles qui minimisent R_{emp} : le lproblème de la détermination du modèle $f(x,W,\theta)$ à partir de l'échantillon fini donné est un problème mal posé [5]. La Figure 3-1 montre plusieurs modèles possibles pour les observations $\left(x^1,y^1\right),\left(x^2,y^2\right),...,\left(x^n,y^n\right)$; le principe ERM nous amènerait à choisir, parmi ceux-là, le dernier, qui a la meilleure précision.

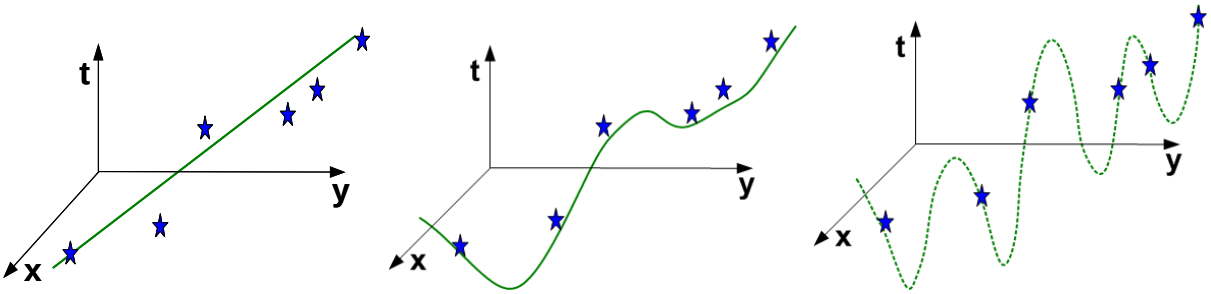


Figure 3-1 – La précision de quelques modèles

On peut alors se demander comment va se comporter le modèle $f(x,W,\theta)$ sur de nouvelles données (ensemble de test) : la Figure 3-2 montre comment les modèles précédents se comportent sur de nouvelles données. On voit que si on veut privilégier la robustesse, c'est-à-dire la qualité du modèle sur de nouvelles données, on sera cette fois amener à choisir, parmi les modèles représentés, le deuxième modèle.

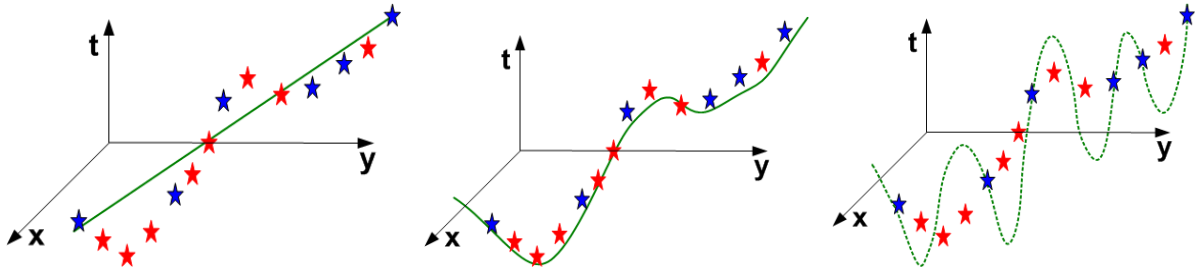


Figure 3-2 – La robustesse des modèles

On voit donc que le principe ERM seul ne peut pas garantir précision et robustesse.

3.1.4 Dimension de Vapnik Chervonenkis

La dimension de Vapnik Chervonenkis – ou VC dimension – mesure la capacité de modélisation de la classe de fonctions Φ_θ. Nous présentons ce concept dans le cas d'une classification en 2 classes. La généralisation au cadre général de la classification ou de la régression est similaire [17].

Etant donné un échantillon de n observations $\left(x^1, x^2, ..., x^n\right)$ en p variables : $x^i = \left(x_1^i, x_2^i, ..., x_p^i\right)$. Il y a 2^n façons de séparer ces n observations en 2 classes.

On dit que la famille de fonctions $\Phi_\theta = \left\{f\left(., W, \theta\right), W \in \aleph\right\}$ *pulvérise* l'échantillon si toutes les 2^n séparations sont réalisables (avec un $\hat{W}_\theta$ bien choisi). On dit que la famille Φ_θ est de *VC dimension* $h_\theta \in$ si h_θ est le nombre maximum de points qui peut être pulvérisé par Φ_θ :

- Il existe au moins un échantillon de h_θ observations qui peut être pulvérisé par Φ_θ
- Aucun échantillon de $h_\theta + 1$ observations ne peut être pulvérisé par Φ_θ

Par exemple, si on utilise la famille des droites de $\Re^2$, la Figure 3-3 montre que la VC dimension de cette famille est 3 :

- Il y a au moins un échantillon de 3 points qui peut être pulvérisé par les droites
- Aucun échantillon de 4 points ne peut être pulvérisé par les droites (ni 4 points en position générale, ni 4 points en position particulière)

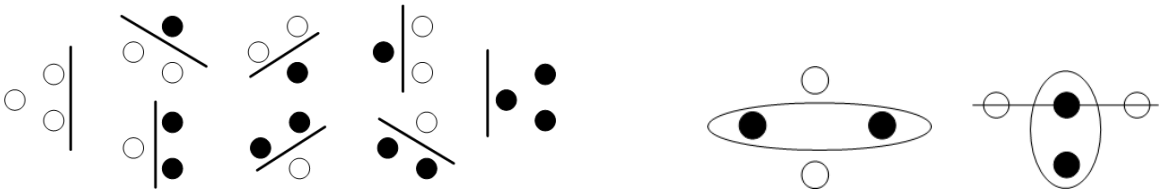


Figure 3-3 –Echantillons de 3 et 4 points de $\Re^2$

3.1.5 Statistical Learning Theory

La « Statistical Learning Theory » de Vapnik [17, 18] est une théorie générale qui utilise la VC dimension et repose sur 4 principes :

- Consistence (robustesse) : capacité à généraliser correctement sur de nouvelles données ;
- Vitesse de convergence : capacité à généraliser de mieux en mieux quand le nombre de données d'apprentissage augmente ;
- Contrôle de la capacité de généralisation : stratégie qui permet de contrôler la capacité de généralisation à partir des seules données d'apprentissage ;
- Stratégie pour obtenir de bons algorithmes : stratégie qui permet de garantir et mesurer la capacité de généralisation du modèle que l'algorithme produit

3.1.5.1 Consistence

On dit que le principe d'inférence ERM est consistent pour la classe de fonctions $\Phi_\theta = \left\{f\left(., W, \theta\right), W \in \aleph\right\}$ si et seulement si $R_{emp}(\theta)$ et $R_{Gen}(\theta)$ convergent vers la même limite quand la taille de l'échantillon n tend vers l'infini.

Vapnik a démontré [17] que c'est le cas si et seulement si la famille Φ_θ est de VC dimension h_θ finie.

3.1.5.2 Vitesse de convergence

V. Vapnik a démontré [17] le théorème suivant :

Quel que soit $\eta \in [0,1]$, alors, avec probabilité $1-\eta$ $\quad R_{Gen}(\theta) \le R_{emp}(\theta) + \varepsilon(n,h)$ (7)

avec $\quad \varepsilon(n,h) = \sqrt{\dfrac{1+ln\left(2n/h\right)}{n/h} - \dfrac{ln\,\eta}{n}}$ (8)

Ce résultat est indépendant de la distribution $P(X,Y)$ de (X,Y) : il démontre que, si n est assez grand, $\varepsilon \approx 0$ et donc l'erreur en généralisation est du même ordre que l'erreur en apprentissage, c'est-à-dire que le modèle est robuste (Figure 3-4 à gauche)

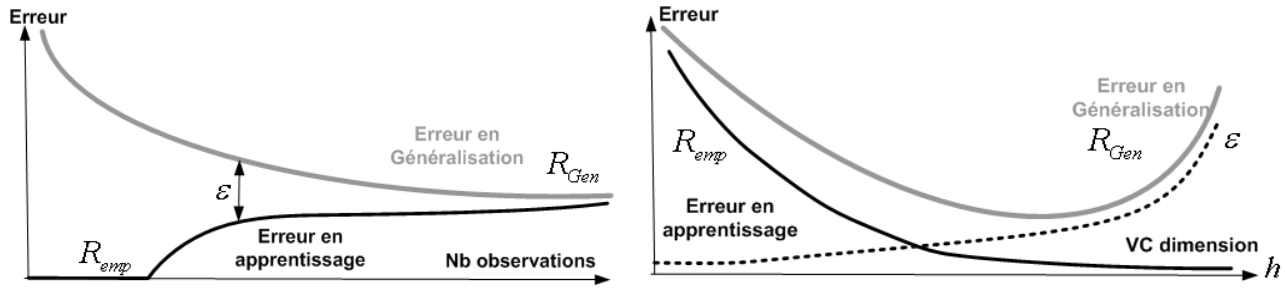


Figure 3-4 – Consistence du principe d'inférence ERM (à gauche) & Contrôle de la capacité de généralisation (à droite)

3.1.5.3 Contrôle de la capacité de généralisation

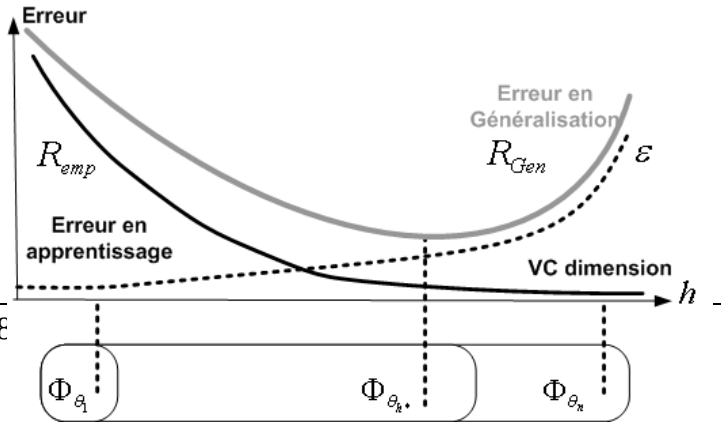
En pratique, on ne peut pas rendre n aussi grand que nécessaire : n est la taille de la base dont on dispose. On voit donc, dans l'équation (7) ci-dessus qu'on a deux cas possibles :

- Quand n/h est grand, on minimise le risque empirique R_{emp} et on est assuré que R_{Gen} est du même ordre ;
- Quand n/h est petit, on doit minimiser les deux termes : R_{emp} et $\varepsilon(n,h)$. La Figure 3-4 (à droite) montre que, à n fixé, quand h augmente, l'écart $\varepsilon \to +\infty$ et donc, à partir d'une VC dimension h^* l'erreur en généralisation R_{Gen} se met à croître et devient de plus en plus différente de R_{emp}.

Le point h^* du minimum de R_{Gen} correspond au meilleur compromis entre précision (R_{emp} petit) et robustesse (R_{Gen} petit).

3.1.5.4 Stratégie pour obtenir de bons algorithmes

Nous venons de voir qu'il existait une valeur optimale h^* qui réalise le meilleur compromis entre précision et robustesse : il nous faut une stratégie qui nous permette de l'obtenir. Vapnik [17] introduit pour cela la SRM (*Structural Risk Minimization*) : on utilise des familles de fonctions emboîtées



$\Phi_{\theta_1} \subset \Phi_{\theta_2} \subset ... \subset \Phi_{\theta_k} \subset ...$ de VC dimension croissante : $h_1 < h_2 < ... < h_k < ...$ (Figure 3-5).

Figure 3-5 – Structural Risk Minimization

L'algorithme pour déterminer le modèle est alors le suivant : on découpe l'ensemble de données en deux parties, l'une est dite *ensemble d'estimation* et l'autre l'*ensemble de validation* (on découpe quelquefois en trois parties, en ajoutant de plus un *ensemble de test*, qu'on n'utilisera pas pour produire le modèle, mais uniquement pour mesurer finalement les performances du modèle produit). On va utiliser l'erreur en validation comme estimateur de l'erreur en généralisation :

1. Commencer avec Φ_{θ_1}

2. Fit des données : pour chaque Φ_{θ_k}, faire :

– Sur l'ensemble d'estimation (ou échantillon d'apprentissage), produire le « meilleur » modèle de Φ_{θ_k}, c'est à dire choisir $\hat{W}_{\theta_k} = arg\min_{W} R_{emp}(W, \theta_k)$ (9)

– Mesurer l'erreur sur l'ensemble de validation (ou échantillon de généralisation)

$$R_{Val}(\hat{W}_{\theta_k}) = \frac{1}{n_{val}} \sum_{i=1}^{n_{val}} L\left[y^i, f(x^i, \hat{W}_{\theta_k}, \theta_k)\right] \quad (10)$$

– Si $R_{Val}(\hat{W}_{\theta_{k-1}}) \succ R_{Val}(\hat{W}_{\theta_k})$ alors faire $k = k+1$ et aller à 2;

sinon stop et faire $\theta_{k^*} = \theta_k$

3. Choix du modèle

– Le meilleur modèle est celui qui correspond à θ_{k^*}

La SRM est une façon d'implémenter le principe du rasoir d'Occam. Plus précisément, alors que le rasoir d'Occam utilise comme mesure de la complexité la complexité du modèle, SRM utilise la complexité de la famille de fonctions.

3.1.6 Conclusion

La Statistical Learning Theory de Vapnik apporte un ensemble de résultats permettant de contrôler la classe de modèles où on recherche la solution et la VC dimension h de la classe retenue ; la SRM est une méthode de contrôle qui garantit le meilleur compromis précision / robustesse du modèle obtenu. Les résultats étant indépendants de la distribution des données, on s'affranchit de la nécessité de connaître cette distribution, de l'estimer : en effet, estimer la distribution nécessite de résoudre un problème plus complexe que de simplement déterminer le modèle. Notons que la SRM ne donne aucune indication sur la « bonne » classe de modèles, sauf que sa VC dimension doit être finie.

La SRM est utilisée dans de nombreux techniques de data mining. Par exemple, pour les réseaux multi-couches, on peut définir une structure emboîtée de SRM [2] :

– Par l'architecture : en augmentant progressivement le nombre de neurones de la couche cachée (Figure 3-6)

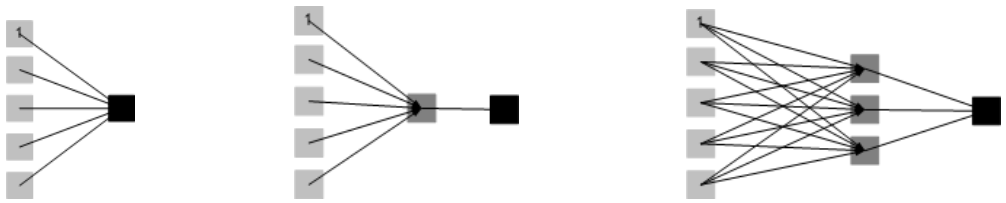


Figure 3-6 – Famille emboîtée de MLP

– Par l'algorithme d'apprentissage : prenons la classe $\Phi_{\lambda_i} = \{f(x; W, \lambda_i), \|W\| \leq \lambda_i\}$ des réseaux multi-couches dont les poids W sont bornés, avec $\lambda_1 < \lambda_2 < ... < \lambda_h < \cdots < \lambda_k < ...$ La solution optimale dans Φ_{λ_i} est celle qui minimise :

$$\Re(W, \lambda_i) = \frac{1}{n}\sum_{k=1}^{n}\left[y^k - f(x^k; W, \lambda_i)\right]^2 + C_i \sum_j W_j^2 \qquad (11)$$

où C_i dépend de λ_i, le paramètre de contrôle de la pulvérisation. On retrouve le *weight decay.*

3.2 La SRM en pratique dans KXEN

KXEN utilise la méthode de Structural Risk Minimization de Vapnik. On découpe l'ADS en trois sous-ensembles (Figure 3-7) pour l'estimation (fit des données), la validation (choix du modèle) et le test (si on veut mesurer les performances du modèle final; cet ensemble est optionnel et n'est bien sûr pas utilisé pour déterminer le meilleur modèle)..

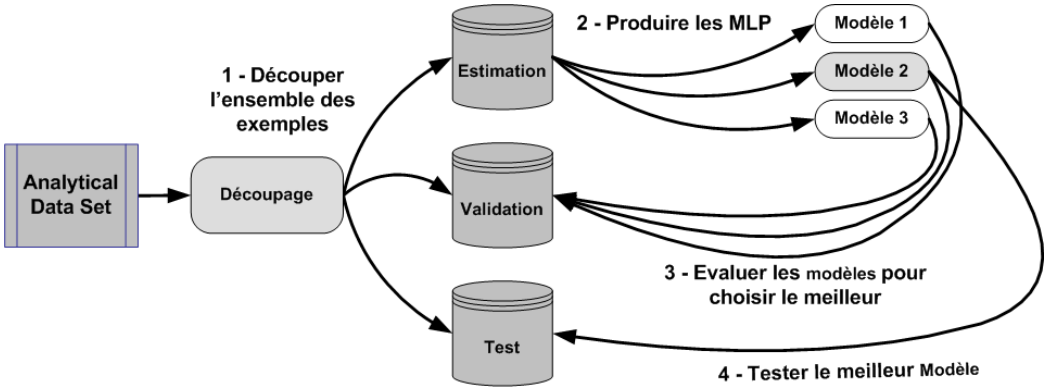


Figure 3-7 – Mise en oeuvre de la SRM dans KXEN

On utilise une famille emboîtée de polynômes de degré donné d :

$$\Phi_{\theta_i}^d = \{f(x; W, \theta_i), \|W\| \leq \theta_i\} \qquad (12)$$

où f est un polynôme de degré d et coefficients W bornés avec $\theta_1 < \theta_2 < \theta_h < \cdots < \theta_k < ...$

On détermine le vecteur optimal de coefficients W dans $\Phi_{\theta_i}^d$ en utilisant l'ensemble d'estimation $(x^1, y^1), (x^2, y^2), ..., (x^n, y^n)$ comme :

$$W_i^* = arg\,min_W \frac{1}{n}\sum_{k=1}^{n}\left[y^k - f(x^k; W, \lambda_i)\right]^2 \qquad (13)$$

sous la contrainte $\|W\| \leq \lambda_i$.

Ce qui est équivalent (en utilisant le Lagrangien) à minimiser :

$$\Re(W,\lambda_i)=\frac{1}{n}\sum_{k=1}^{n}\left[y^k-f\left(x^k;W,\lambda_i\right)\right]^2+C_i\sum_j W_j^2 \qquad (14)$$

où C_i est le coefficient de Lagrange (ou *ridge*) qui dépend de λ_i. On voit donc ici que cette approche est équivalente à une méthode de régularisation et $\Re(W,\lambda_i)$ est le *risque régularisé*. KXEN peut donc être considéré comme une méthode de régression polynomiale régularisée ridge.

Le paramètre θ_{i*} optimal est obtenu par la méthode décrite au §3.1.5.4 avec l'ensemble de validation.

Enfin, KXEN utilise comme fonction de perte non pas l'écart quadratique (1) ou le nombre d'erreurs de classification, mais *KI* – KXEN Information Indicator – le rapport entre l'aire sous la courbe de lift et l'aire sous la courbe du modèle parfait.

A partir d'un modèle $f(x;W,\lambda)$, notons C_s le classifieur au seuil s :

$$\begin{aligned} C_s(x) &= 1 \quad \text{si} \quad f(x,W,\lambda)\geq s \\ &= 0 \quad \text{sinon} \end{aligned} \qquad (15)$$

Si on note $G(s)$ la proportion des observations dont le score est supérieur à s :

$$G(s)=\int_s^{+\infty} f(x,W,\lambda).dP(x) \qquad (16)$$

et $\alpha(s)$ la *sensibilité* du classifieur C_s : $\alpha(s)=VP/nbVP$ (17)

où *VP* est le nombre d'exemples positifs (vraie classe 1) correctement classifiés et *nbVP* le nombre d'exemples réellement positifs ; alors la courbe de lift (Figure 3-8) représente α en fonction de G.

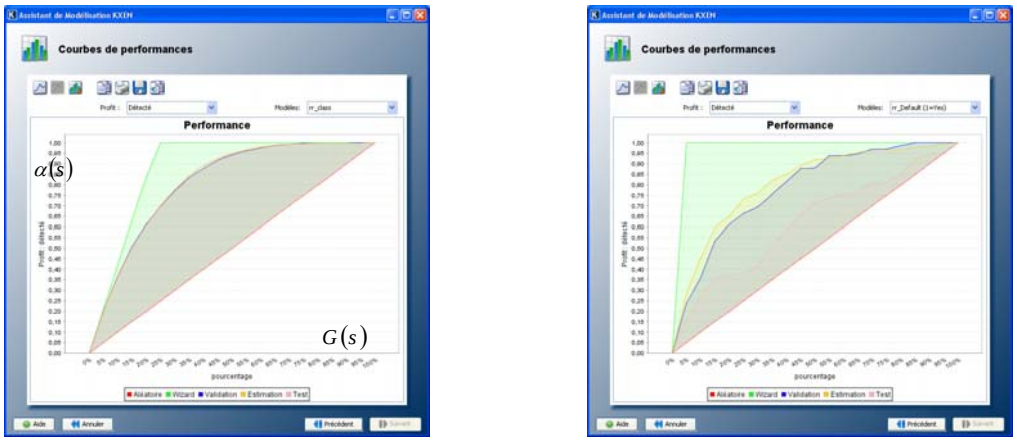


Figure 3-8 – Courbe de lift

Si on note $\beta(s)=VN/nbVN$ la *spécificité*, où *VN* est le nombre d'exemples négatifs correctement classifiés et *nbVN* le nombre d'exemples réellement négatifs, alors *l'aire sous la courbe ROC*, *AUC* est défini par : $AUC=\int_{+\infty}^{-\infty}\alpha(s).d[1-\beta(s)]$ (18)

Et on voit facilement que *KI* n'est autre que l'*index de Gini*, relié à *AUC* par :

$$KI=2AUC-1 \qquad (19)$$

L'écart entre les erreurs sur l'ensemble d'apprentissage et l'ensemble de validation est alors mesurée par : $\varepsilon = KI_{Valid} - KI_{Estim} = 1 - KR$ (20)

où KI_{Valid}, KI_{Estim} représentent respectivement les indices KI pour les ensembles d'estimation et de validation. Si KR est proche de 1, on est donc assuré que le modèle sera robuste, c'est-à-dire qu'il généralisera correctement sur de nouvelles données. Par contre si KR est faible (KR varie entre 0 et 1), le modèle peut produire de mauvaises performances sur de nouvelles données : la Figure 3-8 (à droite) montre, par exemple, les courbes de lift pour les ensembles d'estimation, de validation et de test dans le cas d'un échantillon où KR est petit et donc la qualité du modèle est mauvaise (comme l'atteste la dégradation des performances observée sur l'ensemble test).

KXEN utilise également la SRM pour coder automatiquement les variables.

Pour réaliser une régression (une classification ou une segmentation supervisée), on code les variables une par une de la façon suivante (Figure 3-9) :

Pour chaque variable x_j, $j = 1,..., p$,

- On construit une famille emboîtée de codages :
  - Si x_j est une variable continue, on la découpe en P intervalles (par défaut $P = 20$) et on regroupe progressivement les intervalles ;
  - Si x_j est une variable nominale ou ordinale, on regroupe progressivement les valeurs

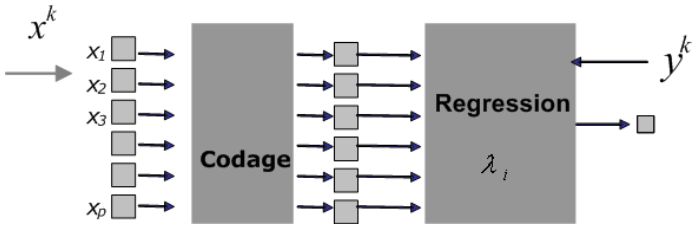


Figure 3-9 – Codage des variables

- On produit, pour chaque codage le modèle expliquant la cible par cette seule variable, et on choisit, par la méthode SRM habituelle, le meilleur modèle.
- Chaque variable a donc ses coefficients KI et KR individuels, qui représentent sa capacité à expliquer, seule, la variable cible.

Le codage produit est non linéaire et dépend du modèle (la cible) : par exemple, la Figure 3-10 montre le codage de 2 variables continues pour un modèle prédictif (« gagner plus de 50 k$ par an ») pour la base *Adult* introduite dans [13]. On voit que, pour la variable *Age*, les chances de gagner plus de 50 k$ augmentent progressivement jusque vers 55 ans, pour décroître ensuite ; pour la variable *Capital-Gain*, on a un effet de seuil : il faut que les gains en bourse soient assez grands pour avoir des chances de gagner plus de 50 k$. Ce codage dépend de la cible : ainsi, si on construit, avec les mêmes variables, un modèle pour prévoir *Age*, on voit que la variable *Capital-Gain* est codée différemment (Figure 3-10 à droite) : le gain en bourse croit avec l'âge.

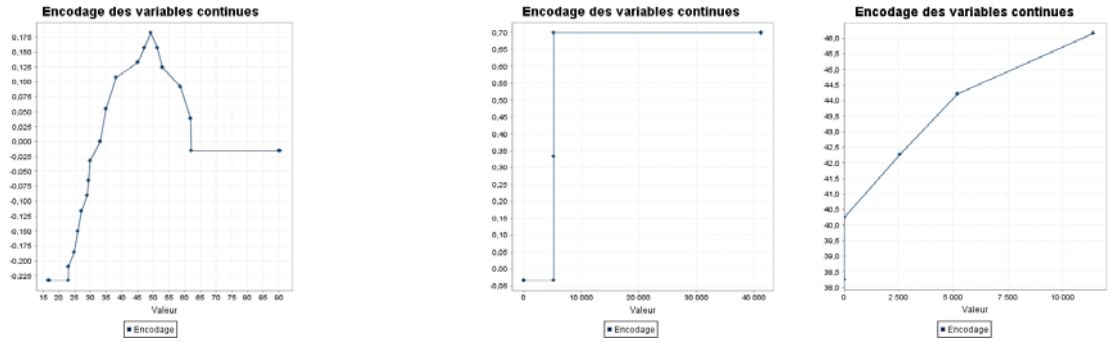


Figure 3-10 – Codage des variables continues *age* et *capital-gain* dans la base *Adult*

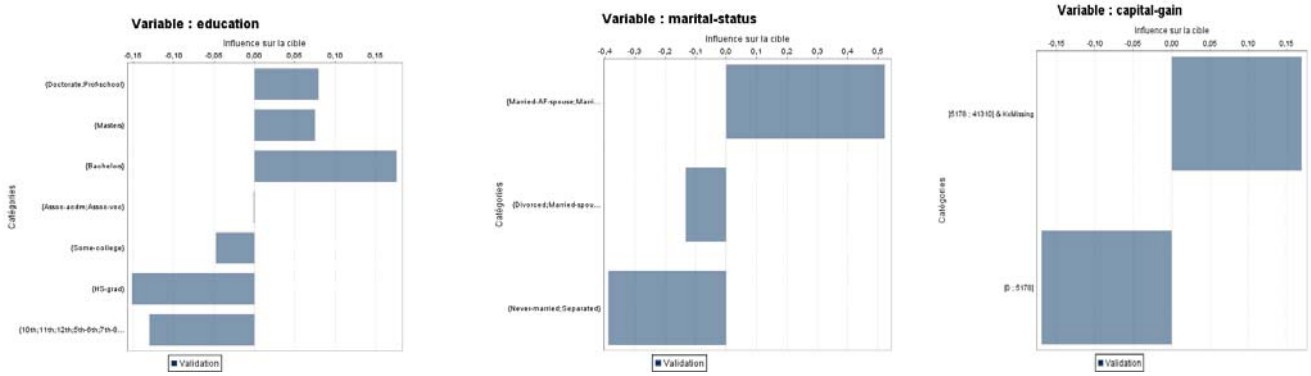


Figure 3-11 – Codage des variables *education, marital-status* et *capital-gain* (base *Adult*)

Les catégories des variables nominales sont regroupées en fonction de la cible : par exemple, la Figure 3-11 montre le codage de 2 variables nominales pour le même modèle que précédemment. Les 14 catégories de la variable *education* ont été regroupées en 7 groupes, et les 7 catégories de *marital-status* en 3. La Figure 3-11 (à droite) montre aussi les 2 catégories de la variable *capital-gain* avec le seuil calculé par le codage (5 178), on y voit également que le codage a créé une catégorie *KxMissing* pour coder les données manquantes de cette variable (c'est la stratégie qu'a choisie KXEN) : cette valeur est regroupée avec les fortes valeurs de *capital-gain*, une indication que la donnée n'est certainement pas MAR –Missing At Random– (une des raisons pour lesquelles KXEN n'utilise pas de méthode d'imputation pour traiter les données manquantes)

Les indicateurs de performance sont $KI = 0{,}807$ et $KR = 0{,}987$: le modèle est donc de bonne qualité et robuste. La courbe de lift du modèle construit est celle de la Figure 3-8 à gauche.

En général, les non-linéarités sont toutes prises en compte par le codage, si bien qu'ensuite on peut se contenter d'un polynôme d'ordre 1 (voir équation $\Phi^d_{\theta_i} = \{ f(x; W, \theta_i), \|W\| \leq \theta_i \}$ **(12)**, c'est-à-dire que KXEN réalise une régression linéaire ridge dans l'espace des variables codées.

L'exploitation de la SRM permet ainsi de coder automatiquement les variables (en produisant un codage robuste), d'automatiser la production du modèle (l'utilisateur n'a besoin de fixer aucun hyper-paramètre) et d'obtenir un modèle robuste, avec un indicateur *KR* mesurant la confiance sur les performances en généralisation du modèle. La classe des modèles proposés est limitée aux polynômes : l'utilisateur n'a donc pas besoin de comparer plusieurs algorithmes. Cette automatisation permet de réduire les temps de production des modèles d'un facteur 10 en moyenne, notamment en réduisant massivement la phase d'exploration et recodage des données.

4 Quelques exemples

De très nombreux clients utilisent aujourd'hui KXEN (voir le site http://www.kxen.com/). Nous allons ici décrire deux exemples illustrant les points introduits au § 2 ; tous les résultats décrits ici ont été développés avec le logiciel KXEN Analytic Framework. Nous terminerons par un exemple intégrant du text mining.

4.1 Le logiciel KXEN Analytic Framework

Le logiciel intègre les théories de Vapnik comme nous l'avons décrit au § 3 et comprend les modules suivants (Figure 4-1) :

- Des modules de préparation de données : KEL, KSC et KTC ;
- Un module de codage automatique : K2C ;
- Des modules de modélisation : K2R, K2S, KTS et KAR
- Un module d'export du modèle : KMX

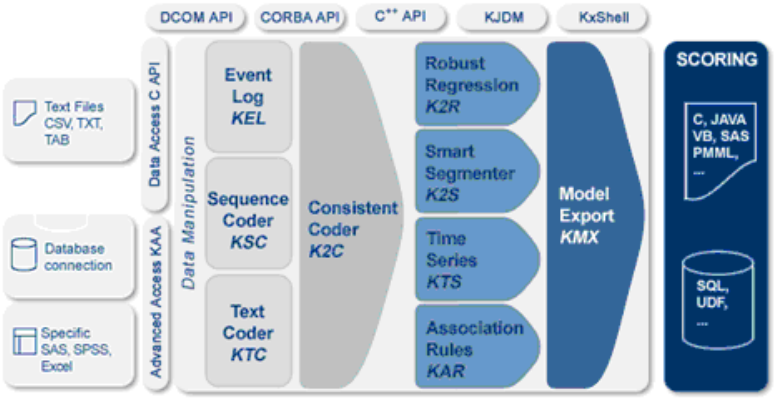


Figure 4-1 – KXEN Analytic Framework

Comme on le voit, KXEN ne propose pas une bibliothèque d'algorithmes, mais des « fonctions d'analyse » conformément aux standards JDM (Java Data Mining) [12].

4.2 Crédit Lyonnais

L'offre bancaire de LCL – Le Crédit Lyonnais – englobe toute la gamme de produits et services bancaires, les produits de gestion d'actifs et d'assurance, et la gestion de patrimoine, soit quelques 400 produits. LCL lance chaque année plus de 130 actions de marketing direct, sur toute la France, par le biais d'emailings, de mailings ou d'envois par SMS, pour un total d'environ 10 millions de contacts sur des clients ou des prospects.

Avant le déploiement de KXEN, les équipes de marketing direct de LCL réalisaient leurs campagnes marketing à partir d'une dizaine de scores généralistes, parmi lesquels « Faire fructifier son capital », « Percevoir des Revenus », « S'assurer au quotidien ». Le département marketing opérationnel voulait disposer de scores plus précis, en fonction des spécificités propres aux offres intégrées dans les grandes familles de produits, et facilements évolutifs.

Avec les outils existants, entre deux et cinq jours étaient nécessaires pour construire de tels scores, délais jugés bien trop longs par le responsable ciblages et analyses de résultat chez LCL. De plus, cette méthode ne permettait pas d'affiner les scores.

Un projet pilote sur une opération grandeur réelle visant à promouvoir une assurance multirisques habitation a été effectué pour évaluer les principales offres du marché, parmi lesquelles KXEN. Pour cela, deux groupes ont été constitués, le premier utilisant les scores KXEN, l'autre utilisant le score d'affinité spécifique aux produits de la famille « S'assurer au quotidien ». Le taux de retour sur le projet pilote utilisant les scores KXEN a été 2,5 fois supérieur au groupe de référence, sur une cible de 250 000 clients. Remarquons que, statistiquement, on devait bien s'attendre à ce qu'un score spécifique – celui réalisé ici avec KXEN – produise un meilleur résultat qu'un score générique regroupant plusieurs produits. De plus, la solution KXEN Analytic Framework a permis d'élaborer un score d'appétence opérationnel en une demi-journée au lieu de plusieurs jours habituellement.

Aujourd'hui, 160 modèles sont créés par an – au lieu d'une dizaine auparavant avec les techniques traditionnelles – pour 130 campagnes de marketing direct sur l'année. Utilisée au quotidien, la solution KXEN s'est notamment démarquée par sa capacité à traiter les volumes importants de données et à s'intégrer de façon transparente dans le système d'information décisionnel de LCL. Cette caractéristique technique permet aux utilisateurs qui ont été formés à la solution KXEN de préparer les données directement dans l'environnement UNIX existant sans avoir à effectuer de fastidieux transferts de données vers les postes de travail. KXEN facilite la lisibilité et la compréhension des ciblages de campagnes : les fiches de scores obtenus sont transmises aux chefs de projets marketing qui s'occupent de chaque campagne, ainsi qu'aux chefs de produits qui participent aux réunions de ciblage. Les résultats, quant à eux, sont diffusés de façon plus large via des rapports plus graphiques et moins techniques. Les intervenants dans les réunions de ciblage ont plébiscité l'outil, ils sont maintenant souvent à l'initiative de la création de nouveaux scores.

4.3 Sears

Comme tous les distributeurs aujourd'hui, Sears, le troisième groupe de distribution américain, fait face à une concurrence féroce et une évolution rapide de son marché. Pour maintenir sa position, Sears s'est concentré sur la réduction des coûts et l'amélioration de la productivité. En particulier, dans son métier de vente à distance (sur catalogue), Sears a développé depuis longtemps une expertise de modélisation qui lui permet d'optimiser ses promotions et ses offres en ciblant les clients les plus susceptibles d'y répondre favorablement.

Cependant, l'environnement analytique, largement orienté grands systèmes, devenait de plus en plus lourd à exploiter et à connecter aux bases de données, les ressources informatiques spécialisées nécessaires devenant de plus très difficiles à trouver.

Sears lance alors un projet [1] visant à rendre son système de marketing direct plus productif, plus réactif, avec moins de ressources. Le projet s'est déroulé en trois phases :

1. Intégration des sources de données multi-canal (grands magasins, vente en ligne, catalogue), multi-marques (Sears, Orchard Supply Hardware, Lands' End), donénes crédit, démographiques, … Le data mart incluant ces données comprend plus de 900 attributs et Sears l'intègre alors au datawarehouse Teradata de l'entreprise, permettant à l'équipe catalogue de réduire ses coûts opérationnels.
2. Intégration de KXEN pour automatiser le processus d'analyse. Sears a ainsi automatisé la préparation des données, y compris le codage des variables nominales et la description des importances relatives des attributs.
3. Utilisation de KXEN pour le déploiement : on génère automatiquement le code SQL et UDF du modèle dans Teradata et on peut ainsi scorer dans la base, sans avoir à extraire les données. Alors que dans l'ancien environnement, il fallait des semaines pour créer un modèle, et plusieurs heures

pour scorer la base, Sears crée aujourd'hui des modèles robustes en quelques heures et score 75 millions de clients en 30 minutes.

4.4 Text mining

En 2006, la compétition Data Mining Cup http://www.data-mining-cup.com a attiré 580 participants : eBay Allemagne a fourni les données de 8000 enchères d'une partie de son site (celles concernant les ventes d'iPod Apple). La tâche assignée est de construire un modèle permettant, à un vendeur, de prévoir si le prix de vente final de son produit est supérieur au prix moyen dans la catégorie du produit. Ce modèle doit fournir au vendeur les éléments lui permettant de déposer son enchère pour réussir à vendre au plus fort prix. Ces données comprennent deux champs *listing_title* et *listing_subtitle*, qui sont des champs que le vendeur peut remplir en texte libre pour décrire son produit.

Le module de codage textuel de KXEN, KTC, fonctionne de la façon suivante (Figure 4-2) :

– L'ensemble des mots du texte sont extraits et les indicateurs habituels calculés (comptage, tf, tf-idf)

– On applique ensuite des stop lists qui permettent d'éliminer les « mots vides » ;

– On applique ensuite des règles de lemmatisation pour extraire les racines des mots ;

– Ces racines constituent des colonnes qui sont ajoutées à l'ADS

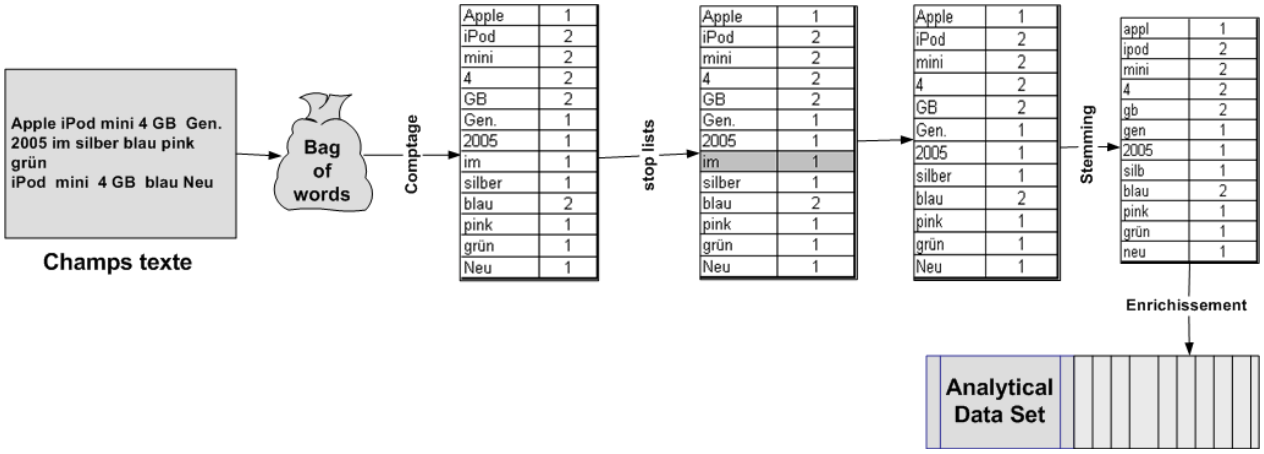


Figure 4-2 – Codage de texte

Nous avons utilisé KTC pour extraire les racines des deux champs textuels, puis avons produit divers modèles de prévision de la variable *gms_greater_avg*, qui vaut 1 si le prix final *gms* est plus grand que le prix moyen de la catégorie *category_avg_gms*. Nous avons réalisé les modèles suivants :

– K2R : on exécute un codage automatique K2C, suivi d'une régression K2R ;

– K2R + données supplémentaires. On peut calculer des variables supplémentaires à partir des données fournies. Par exemple, nous avons ajouté des variables indiquant le mois de l'année, le jour du mois ; le jour de la semaine où l'enchère commence ou finit (*listing_end_monthofyear*, *listing_start_monthofyear*, *listing_end_dayofmonth*, *listing_start_Monday*…) ;

– KTC German : on exécute KTC, suivi de K2C et K2R sur les variables initiales complétées des variables textuelles extraites :

– KTC avec langage DMC : on peut particulariser le langage en prenant en compte des mots spécifiques du domaine ;

– KTC avec DMC et un modèle polynomial d'ordre 2

La Figure 4-3 montre les résultats de ces modèles en mesurant le score obtenu sur la base test fournie par la compétition Data Mining Cup : comme on le voit, les données textuelles apportent beaucoup d'information, et, avec un modèle polynomial de degré 2, on obtient finalement un score supérieur à celui du gagnant de la coupe. On voit bien que les variables textuelles apportent beaucoup d'information, d'autant plus si on utilise un langage « métier » qui incorpore quelques éléments spécifiques du contexte de la compétition DMC.

Expérience	Score	Rang
K2R	2320	139
K2R + données complémentaires	2852	123
KTC German	4232	68
KTC avec langage dédié DMC	4408	44
DMC (le gagnant)	5020	1
KTC – DMC – K2R order 2	5356	

Figure 4-3 – Résultats sur la Data Mining Cup 2006

5 Conclusion

Nous avons montré comment les entreprises, qui collectent volumes croissants de données, sont confrontées, pour la réalisation d'un nombre croissant d'analyses data mining, à la nécessité de mettre en place des *usines à modèles* capables d'industrialiser le processus de modélisation. Nous avons ensuite présenté le cadre théorique de la théorie de l'apprentissage statistique de Vladimir Vapnik, en montrant que la minimisation structurelle du risque apporte une méthode contrôlée pour produire des modèles robustes. Nous avons ensuite décrit comment KXEN a mis en œuvre ces résultats théoriques pour produire un outil logiciel KXEN Analytical Framework qui répond aux besoins de production industrielle des utilisateurs. Nous avons enfin donné quelques exemples de réalisations pratiques illustrant les apports de l'approche *usines à modèles* : capacité à traiter de grands volumes de données, à produire de nombreux modèles, très rapidement, presqu'automatiquement, avec des utilisateurs métier, … ce qui permet à l'entreprise d'augmenter sa productivité.

Nous pensons que de telles approches se développeront de plus en plus, les besoins des entreprises ne faisant que croître alors que leurs ressources restent limitées. Enfin, l'utilisation de nouvelles sources de données (textes, réseaux sociaux, …) sera de plus en plus répandue.

6 Références

1- Bibler, Paul and Bryan, Doug Sears: A Lesson in Doing More With Less. TM Tipline. (sept. 2005) http://ga1.org/tmgroup/notice-description.tcl?newsletter_id=1960075&r=#6

2- Bottou, L. La mise en oeuvre des idées de VladimirN. Vapnik. In Statistiques et méthodes neuronales. Ecole Modulad, Montpellier. S. Thiria, O. Gascuel, Y. lechevallier, S. Canu eds. Dunod, Paris, 262-274, (1997)

3- Davenport, T. H., Harris, J. G. : Competing on Analytics: The New Science of Winning. Harvard Business School Press. (2007)

4- Fayyad, U. A Data Miner's Story – Getting to Know the Grand Challenges, Invited Talk, KDD'07. (2007). http://videolectures.net/kdd07_fayyad_dms/

5- Fogelman Soulié, F. Réseaux de neurones et Statistiques, une introduction. In Statistiques et méthodes neuronales. Ecole Modulad, Montpellier. S. Thiria, O. Gascuel, Y. lechevallier, S. Canu eds. Dunod, Paris, 1-19, (1997)

6- Fogelman Soulié, F. Data Mining in the real world. What do we need and what do we have ? KDD'06, Philadelphia, August 20, 2006. Workshop on Data Mining for Business Applications. 49-53, (2006). http://labs.accenture.com/kdd2006_workshop/dmba_proceedings.pdf

7- Fogelman Soulié, F., Bryan, D. Data mining for quality improvement. KDD'07, Second Workshop on Data Mining Case Studies and Success Stories, (2007)

8- Hand, D., Mannila, H., Smyth, P. Principles of Data Mining. MIT Press. (2001)

9- Herschel, G. : CRM Analytics Scenario : The Emergence of Integrated Insight. Gartner Customer Relationship Management Summit (2006)

10- Herschel, G. Customer Data Mining: Golden Nuggets, Not Silver Bullets. Gartner Customer Relationship Management Summit (2006)

11- Hill, S., Provost, F. and Volinsky, C. Network-Based Marketing: Identifying Likely Adopters via Consumer Networks. Statistical Science, Vol. 21, No. 2, 256–276. (2006)
http://pages.stern.nyu.edu/~fprovost/

12- Hornick, M.F., Marcade, E., Venkayala, S. Java Data Mining. Strategy, Standard, and Practice. A practical guide for architecture, design, and implementation. Morgan Kaufmann series in data management systems. Elsevier. (2007)

13- Kohavi, R. Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid, In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, (1996)
ftp://ftp.ics.uci.edu/pub/machine-learning-databases/adult/

14- Jiawei Han, Warehousing and Mining Massive RFID Data Sets, adma'06 (2006)
http://www.itee.uq.edu.au/~adma06/Jiawei_adma06_rfid.pdf

15- Kleinberg, J. Challenges in Social Network Data: Processes, Privacy and Paradoxes, KDD'07. (2007). http://videolectures.net/kdd07_kleinberg_cisnd/

16- Russom, P (2007) BI Search & Text Analytics. TDWI Best Practices Report. (2007).
http://www.tdwi.org/Publications/WhatWorks/display.aspx?id=8449

17- Vapnik, V.N. : The Nature of Statistical Learning Theory. Springer Verlag. (1995).

18- Vapnik, V.N. : Universal learning Technology : Support Vector Machines. NEC Journal of Advanced Technology, Vol.2, No.2 Spring 2005. 137-144. (2005)
http://www.nec.co.jp/techrep/en/r_and_d/a05/a05-no2/a137.pdf