1 Objectif

Fractionnement apprentissage/test des données pour l'évaluation des arbres de décision. Introduction des coûts de mauvais classement.

Un des éléments clés du Data Mining est la validation. Elle peut être experte. Un spécialiste du domaine, à partir de son expérience, est en mesure de statuer sur la pertinence des connaissances produites. C'est certainement l'approche qu'il faut privilégier. Mais elle n'est pas toujours possible. Parce qu'on n'a pas toujours un spécialiste sous la main, parce qu'il peut avoir du mal à décrypter clairement les informations fournies par le modèle, etc. On peut alors se tourner vers des techniques mécaniques, fondées exclusivement sur des procédés numériques. Dans la pratique, ces deux approches sont complémentaires. Les ratios numériques sont des indicateurs que le spécialiste utilise pour motiver son expertise.

En apprentissage supervisé, l'évaluation des modèles repose sur une démarche relativement simple. Il s'agit de confronter les prédictions du modèle avec les vraies valeurs prises par la variable d'intérêt (la variable à prédire), puis de comptabiliser les bonnes affectations ou, son complémentaire, les mauvaises affectations. On parle d'erreur (taux d'erreur) théorique. Dans l'exemple que nous traiterons dans ce didacticiel, il s'agit de déterminer si un message électronique arrivant sur un serveur informatique est un message non sollicité (un Spam) ou non. Idéalement, si nous avons accès à la population globale, il suffit de vérifier pour chaque individu s'il est bien classé ou non. En pratique, c'est impossible. Nous n'avons jamais accès à la population globale. Il nous faut donc travailler sur un échantillon de données.

Le plus simple est d'utiliser le même échantillon pour construire et évaluer les performances du modèle. On parle d'erreur empirique ou erreur en resubstitution. Sa mise en œuvre est simple, nous disposons déjà de tous les éléments nécessaires à sa réalisation. Malheureusement, l'erreur mesurée dans ces conditions est fortement biaisée. On parle de « biais d'optimisme ». Elle sous estime l'erreur théorique. D'autant plus fortement que le modèle aura tendance à « coller » aux données. Ce qui est le cas des arbres de décision du type C4.5. De ce point de vue, l'erreur empirique ne convient absolument pas pour comparer les mérites respectifs de différentes techniques d'apprentissage sur le même jeu de données.

La démarche adéquate consiste à subdiviser dans un premier temps les données en 2 parties : la première pour la construction du modèle, la seconde pour son évaluation. L'erreur mesurée dans ces conditions est dite erreur en test. C'est un estimateur non biaisé de l'erreur théorique. A ce titre, elle permet de comparer objectivement les performances de deux techniques d'apprentissage présentant des caractéristiques différentes. Seul bémol à cette approche, si la taille de la base est faible, réserver une fraction importante pour le test pénalise la construction du modèle. A contrario, réserver une part faible pour le test pénalise la fiabilité de l'estimation de l'erreur. Il faut trouver le bon compromis, ou si vraiment l'échantillon est de petite taille (moins du millier d'observations, c'est un ordre d'idée, la complexité du modèle doit être prise en compte aussi), se tourner vers les techniques de ré échantillonnage. Pour en savoir plus, nous conseillons la lecture du support que nous avons mis en ligne à ce sujet http://eric.univ-lyon2.fr/~ricco/cours/slides/resampling_evaluation.pdf

Second point important que nous traiterons dans ce didacticiel, la prise en compte des coûts de mauvais classement dans l'élaboration des modèles. Bien souvent, dans les études réelles, le poids que l'on peut accorder aux erreurs n'est pas symétrique. Reprenons notre exemple du filtrage des messages non sollicités. Laisser passer un message délictueux est tout à fait différent d'éjecter un message licite. Dans le premier cas, le serveur transfère à l'utilisateur, au moins pour la part des messages litigieux non éliminés, la responsabilité de les supprimer manuellement. C'est un peu fastidieux, mais pas rédhibitoire. C'est une opération que la plupart d'entre nous réalisons plusieurs par jour dans notre logiciel de messagerie. Dans le second cas, la gravité de l'erreur est tout autre. Le serveur supprime un message

licite, peut être d'une importance capitale, et dont nous n'aurons jamais connaissance. Imaginez que Monica Bellucci, enfin disponible, se décide à vous inviter à prendre un verre. Hé bien non, non seulement vous ne recevrez pas le message, mais en plus vous n'en saurez jamais rien de toute votre vie. C'est vraiment la catastrophe intégrale. Bref, si l'on comprend bien que toute erreur d'affectation a un coût, la nouveauté est que l'importance du coût peut être différente selon la nature de l'erreur, et que nous devons en tenir compte.

Deux questions viennent alors : comment intégrer ces coûts pour évaluer le modèle et, surtout, comment les prendre en compte dans son élaboration. En effet, il paraît évident qu'un modèle, optimal pour un système de coûts, ne peut l'être pour un autre. Il faut, dans le meilleur des cas, corriger le modèle existant, par exemple en modifiant le seuil d'affectation ; dans le pire des cas, le reconstruire différemment en intégrant explicitement la matrice de coûts de mauvaise affectation lors de la phase d'apprentissage. Dans ce didacticiel, nous mettons en œuvre une technique de construction d'arbres de décision sensible aux coûts, dérivée de C4.5[1], disponible dans le logiciel SIPINA.

2 Données

Notre fichier de données recense 4601 messages. La variable à prédire est la catégorie du message, Spam « oui/non », les variables prédictives sont la fréquence des mots dans le message, la fréquence des caractères, la longueur moyenne des mots en majuscule, etc. La base est subdivisée en 2 parties : 3601 observations réservés pour la construction du modèle, l'échantillon d'apprentissage ; 1000 observations destinées à l'évaluation, l'échantillon test. Une colonne supplémentaire indiquant le statut de chaque observation (learning ou test) est adjointe au fichier[2].

3 Apprentissage et évaluation de l'erreur - Méthode C4.5

3.1 Chargement des données dans SIPINA

Le plus simple est de charger le fichier de données dans le tableur EXCEL. Depuis l'élaboration de la macro complémentaire SIPINA.XLA, il est possible de faire la jonction entre ce tableur, un des outils de Data Mining le plus utilisé au monde, et notre logiciel. Pour savoir comment installer cette macro, nous pouvons consulter la description animée disponible sur notre site web (http://eric.univ-lyon2.fr/~ricco/sipina_download.html, voir « SIPINA ADD-IN FOR EXCEL SPREADSHEET »).

Après avoir sélectionné la plage de données, nous activons le menu SIPINA/EXECUTE SIPINA disponible maintenant dans EXCEL. Une boîte de dialogue apparaît, nous confirmons en cliquant sur OK après avoir vérifié les coordonnées de la sélection. Attention, pour que SIPINA manipule correctement le fichier, il faut que la première ligne soit constituée des noms de variables (Figure 1) et qu'il n'y ait pas de données manquantes.

SIPINA est démarré automatiquement. Les données sont transférées via le presse-papier. Le nombre de variables et le nombre d'observations sont affichés dans la partie basse de l'éditeur (57 variables, dont la variable indicatrice pour le fractionnement apprentissage/test ; 4601 observations). Il est possible d'effectuer des modifications et de nouvelles saisies dans la grille de données SIPINA, mais ce n'est guère conseillé, EXCEL propose des fonctions d'édition autrement plus performantes.

[1] Voir http://www.informatik.uni-freiburg.de/~ml/ecmlpkdd/WS-Proceedings/w10/chauchat_workshop.pdf

[2] http://eric.univ-lyon2.fr/~ricco/dataset/spam.xls ; le fichier a été retravaillé, l'original est accessible à l'URL http://archive.ics.uci.edu/beta/datasets.html (spambase)

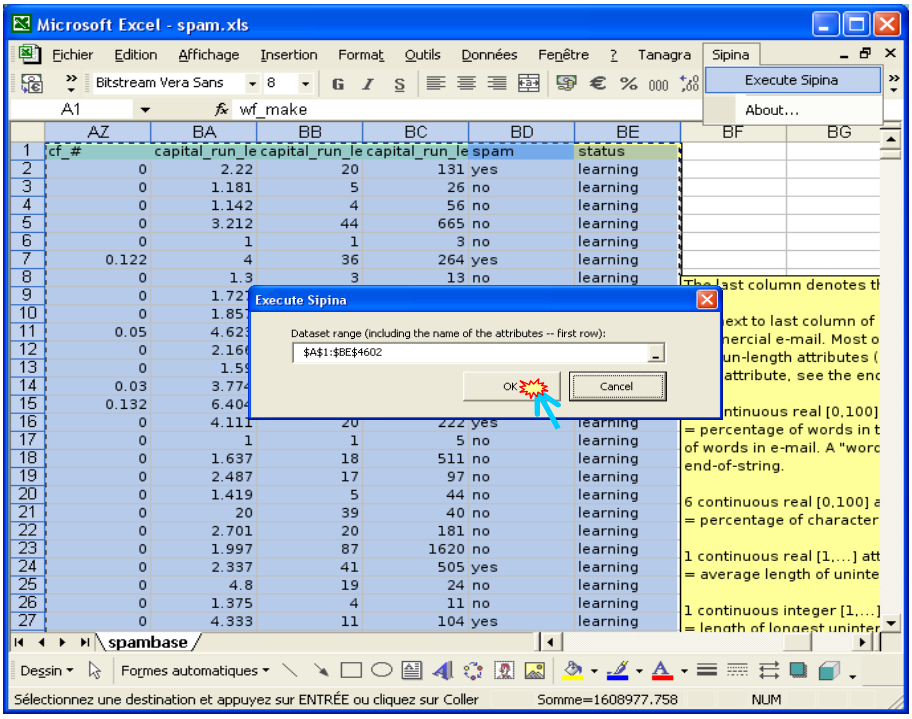
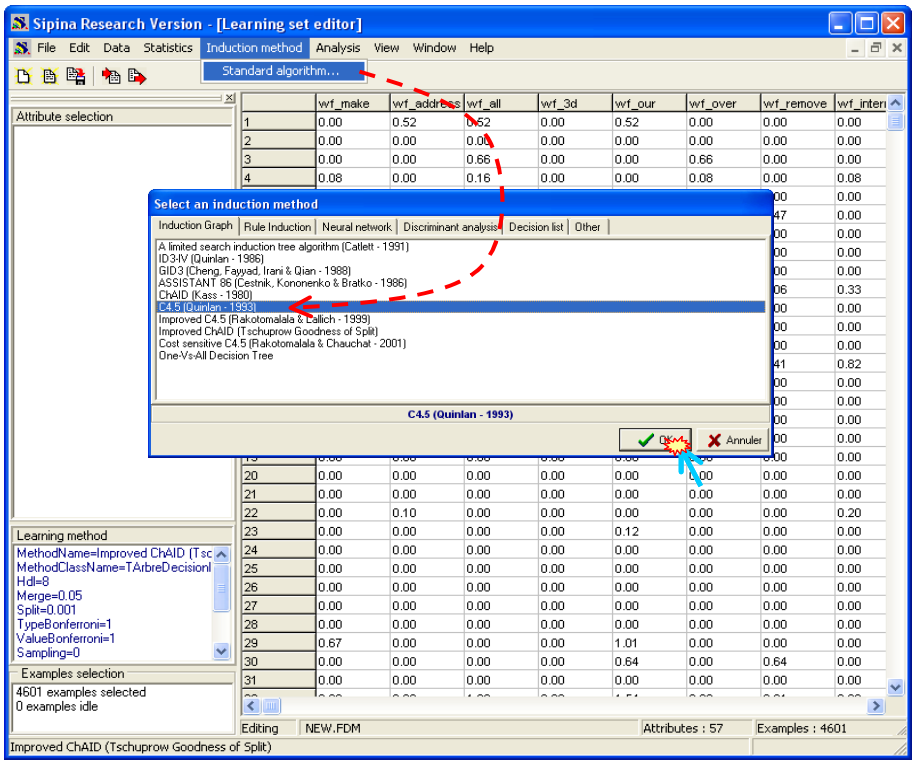


Figure 1 - Transfert des données d'Excel vers SIPINA

3.2 Choisir une méthode d'apprentissage – C4.5

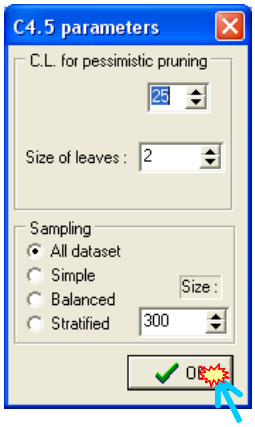
Première opération dans tout logiciel de traitement exploratoire des données : choisir la technique d'analyse. Pour ce faire, nous activons le menu INDUCTION METHOD / STANDARD ALGORITHM, une boîte de dialogue décrivant les algorithmes disponibles apparaît.



Comme nous pouvons le constater, SIPINA intègre plusieurs techniques d'apprentissage supervisées, autres que les arbres de décision. Leur intérêt dans ce logiciel est très relatif depuis que nous avons mis en ligne **TANAGRA** (http://eric.univ-lyon2.fr/~ricco/tanagra/) qui reprend dans un cadre harmonisé l'ensemble de ces méthodes, et en y adjoignant les techniques non supervisées (classification) et descriptives (analyse factorielle). *SIPINA n'est vraiment intéressant que pour les arbres de décision, il recense un grand nombre de techniques référencées fondées sur les arbres, certaines ne sont disponibles nulle part ailleurs (GID3 par exemple).*

Nous sélectionnons la méthode C4.5. Parce que, d'une part, c'est une référence incontournable dans la communauté de l'apprentissage automatique, d'autre part, parce que nous avons développé une méthode dérivée sensible aux coûts de mauvais classement. Nous pourrons la confronter directement avec la version originelle de Quinlan (1993).

Une boîte de dialogue de paramétrage apparaît. Nous la validons telle quelle[3].



Les spécifications de la méthode sont retranscrites dans la section médiane de la partie gauche de la fenêtre principale de SIPINA (Figure 2).

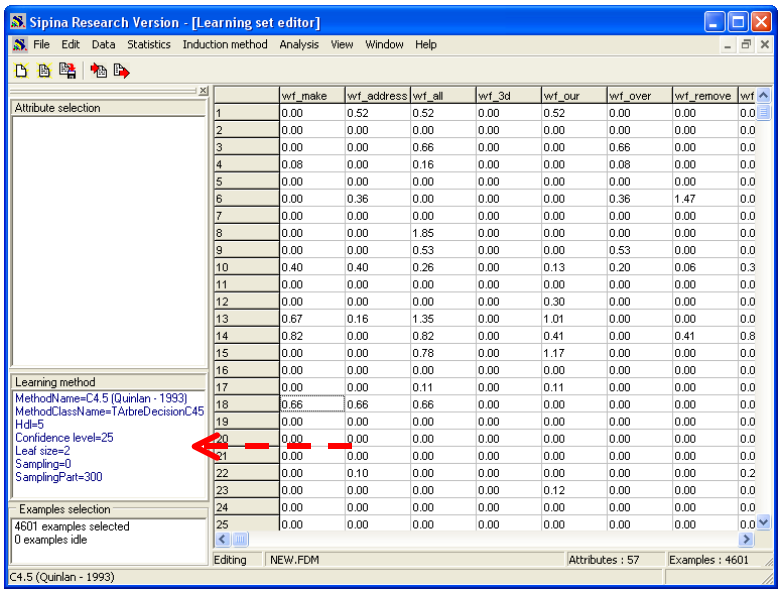
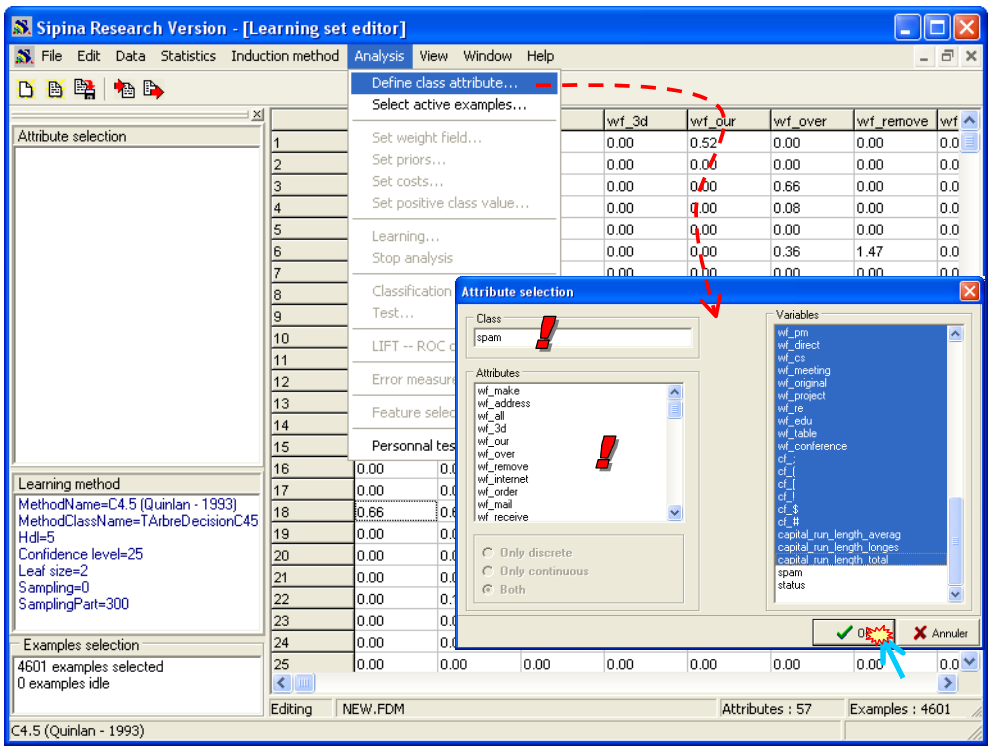


Figure 2 – Méthode choisie : C4.5

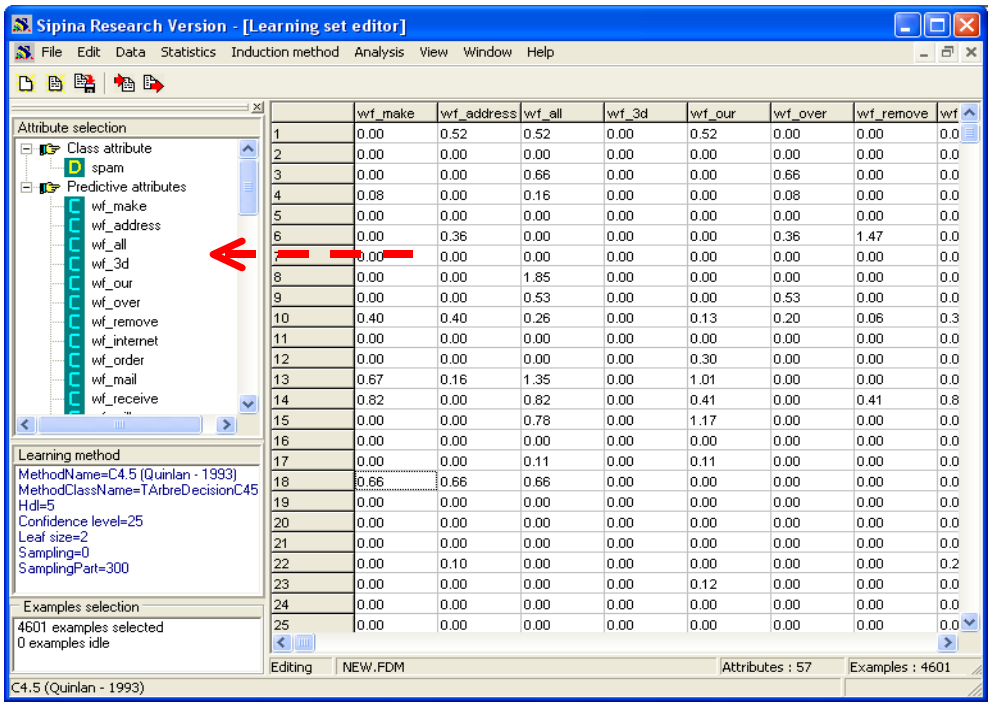
[3] Se référer au document suivant pour plus d'informations sur la méthode C4.5 : http://eric.univ-lyon2.fr/~ricco/cours/slides/arbres_decision_cart_chaid_c45.pdf

3.3 Définition du problème

L'étape suivante consiste à spécifier la variable à prédire (SPAM) et les variables prédictives (les autres, à l'exception de STATUS). Pour ce faire, nous activons le menu ANALYSIS / DEFINE CLASS ATTRIBUTE, une boîte de dialogue apparaît. Par glisser déposer, nous positionnons les variables dans les emplacements adéquats, la sélection multiple est possible.



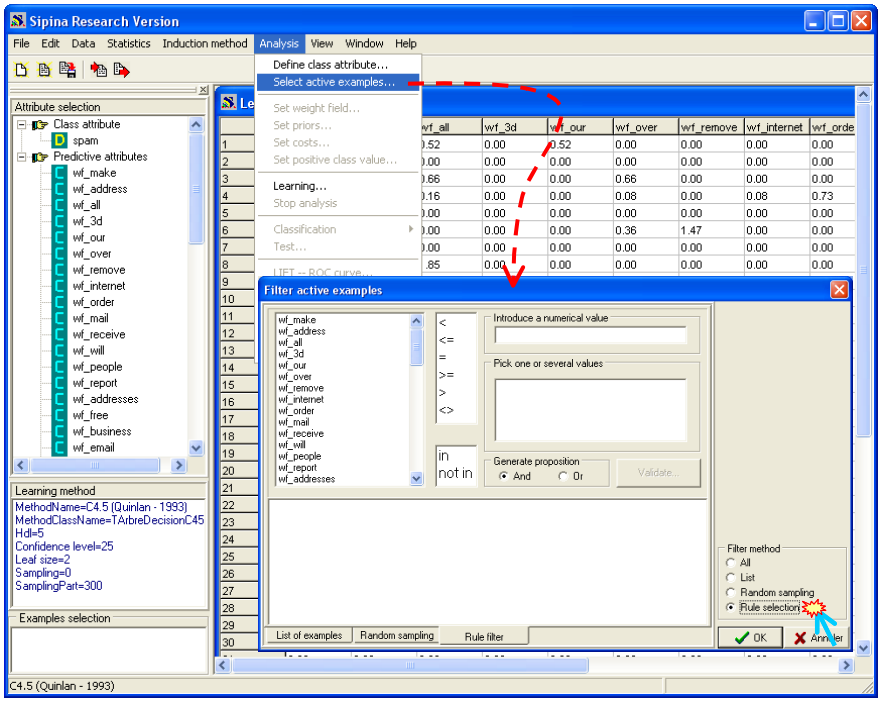
Après validation, les variables sont recensées dans la section haute de la partie gauche de la fenêtre principale.



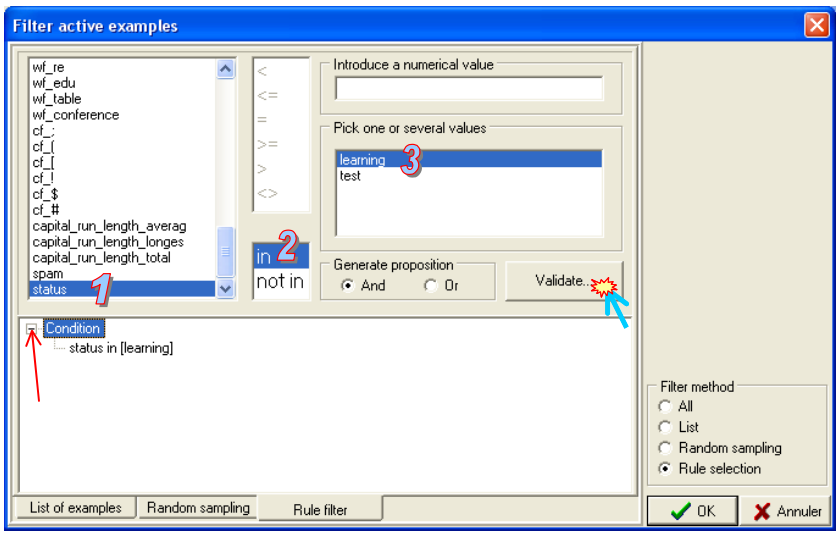
Le sigle « D » devant la variable indique qu'elle est discrète (nominale), « C » indique qu'elle est continue (quantitative).

3.4 Fractionnement des données en apprentissage et test

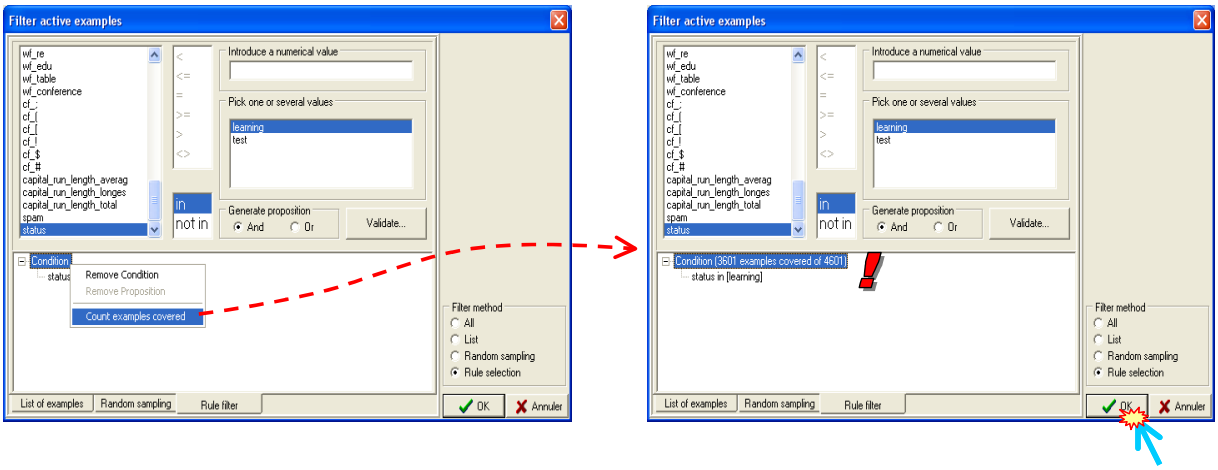
L'idée est de subdiviser l'échantillon en 2 parties : la première pour la construction du modèle, l'échantillon d'apprentissage ; la seconde pour son évaluation, l'échantillon test. La variable STATUS indique le rôle de chaque observation. Pour sélectionner les individus en apprentissage, nous activons le menu ANALYSIS / SELECT ACTIVE EXAMPLES. Une boîte de dialogue apparaît. Nous sélectionnons l'option RULE SELECTION.



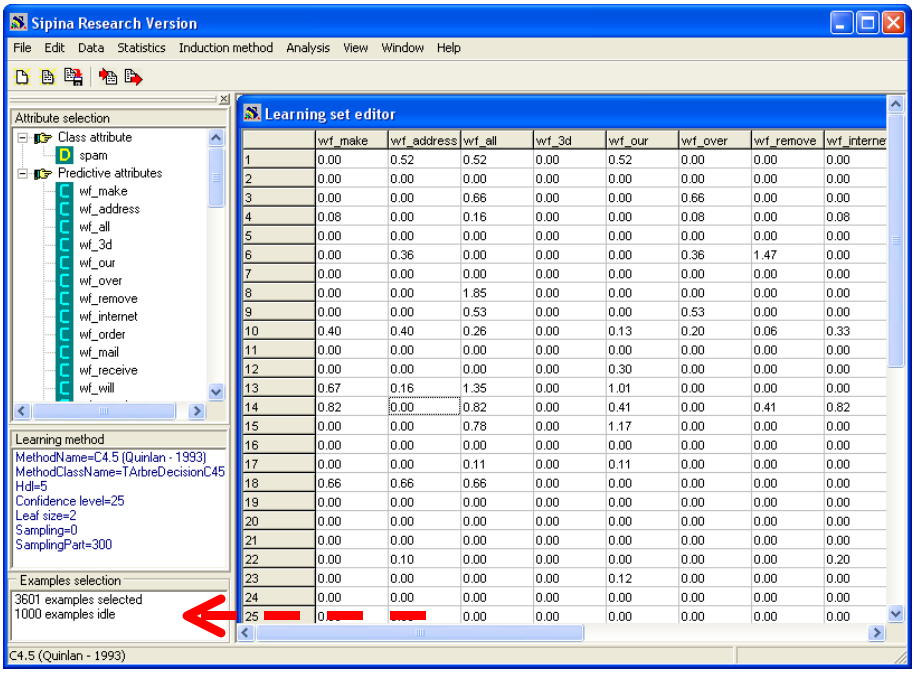
Nous accédons à la fenêtre de définition de la règle de filtrage des observations. Nous devons spécifier que les individus correspondant à « STATUS = learning » doivent faire partie de la fraction apprentissage des données. Nous devons opérer dans un premier temps dans la partie haute de la fenêtre : (1) Sélectionner la variable qui va définir le fractionnement, STATUS ; (2) Sélectionner l'opérateur de comparaison, IN ; (3) Sélectionner la modalité qui apparaît dans la partie « Pick one or several attributes ». Il nous reste à valider l'opération en cliquant sur le bouton VALIDATE. La condition définition le sous ensemble « apprentissage » apparaît dans la partie basse de la fenêtre. Nous visualisons le détail de la règle en cliquant sur le bouton « + ».



Pour vérifier la conformité de la sélection, nous pouvons compter les observations respectant la condition. Il faut pour cela activer le menu contextuel sur la règle (clic avec le bouton droit), et activer l'option « Count Examples Covered », nous voyons apparaître le décompte.



3601 observations sont maintenant réservées pour l'apprentissage. Nous entérinons ce choix en cliquant sur le bouton OK. Le mode de subdivision des données est affiché dans la section basse de la partie gauche de la fenêtre principale.

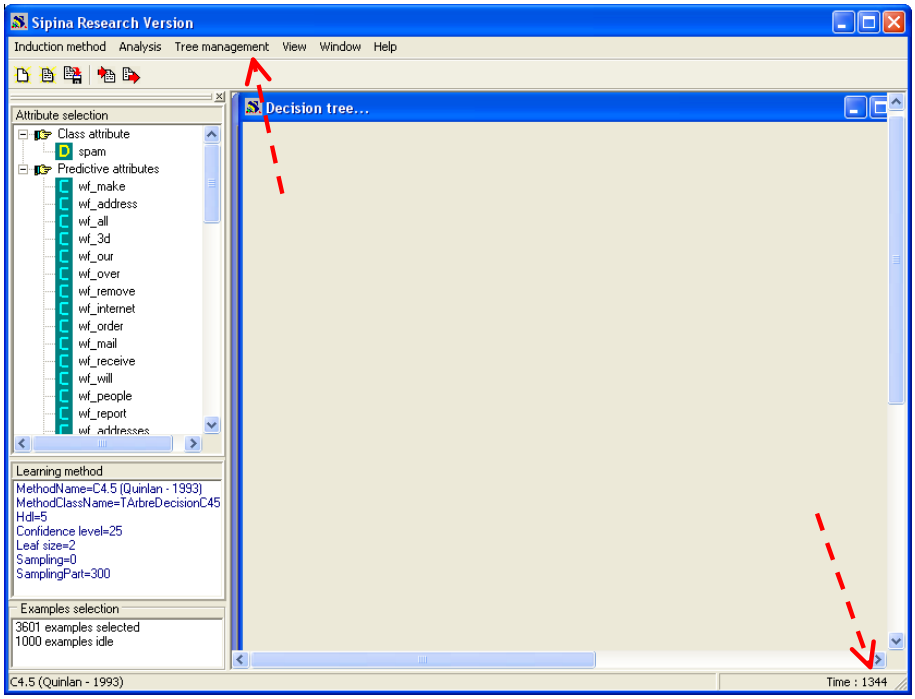


3.5 Apprentissage – Construction du modèle de prédiction

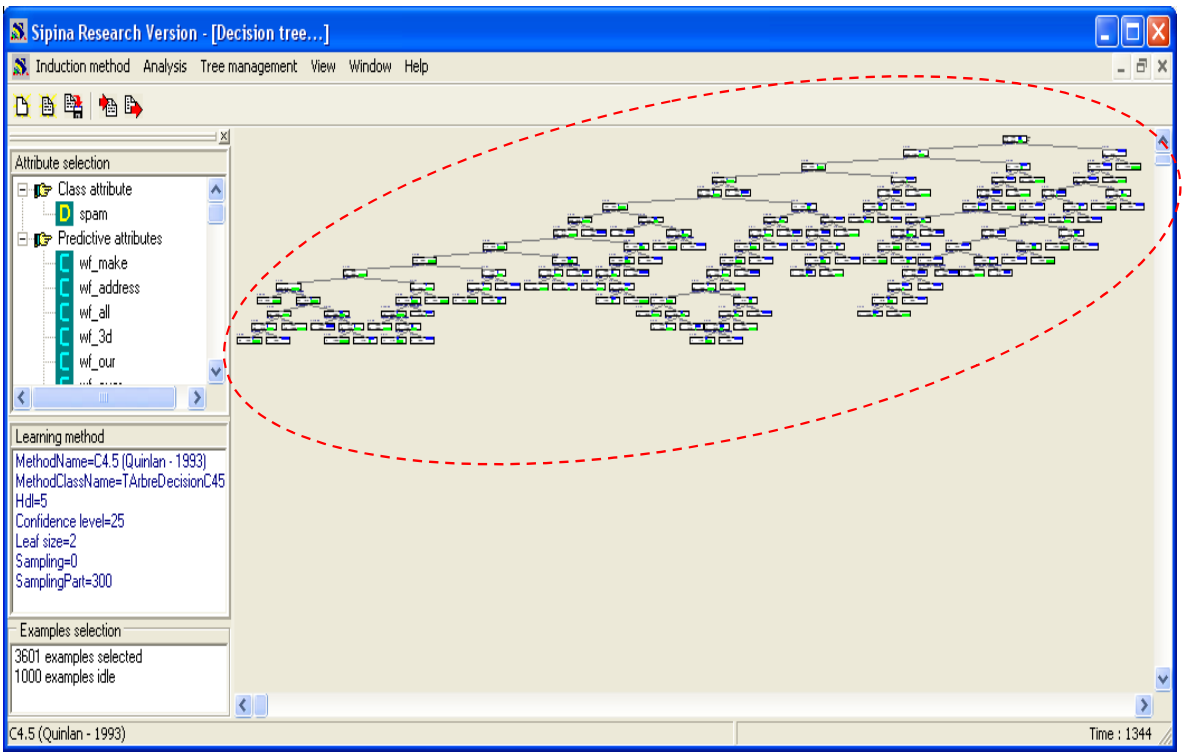
Il nous reste à lancer l'apprentissage. Pour cela, nous activons le menu ANALYSIS / LEARNING. Curieusement, la fenêtre DECISION TREE semble vide, c'est tout simplement parce que l'arbre est très grand, il est situé dans la partir droite de la fenêtre d'affichage.

Un premier indicateur situé en bas à droite de la fenêtre principale nous fournit la durée d'exécution de l'opération[4]. Un menu TREE MANAGEMENT apparaît également dans la barre de menus.

[4] 1 seconde 344 millièmes sur ma machine.



Pour visualiser l'arbre lui même, le plus simple est d'effectuer un zoom arrière à l'aide du menu TREE MANAGEMENT / ZOOM OUT ou d'utiliser directement le raccourci clavier CTL + Q.



L'arbre est de très grande taille, nous ne pouvons pas raisonnablement en tirer une interprétation des résultats qui tienne la route. Il paraît plus approprié dans ce cas de transformer l'arbre en règles pour pouvoir les étudier individuellement.

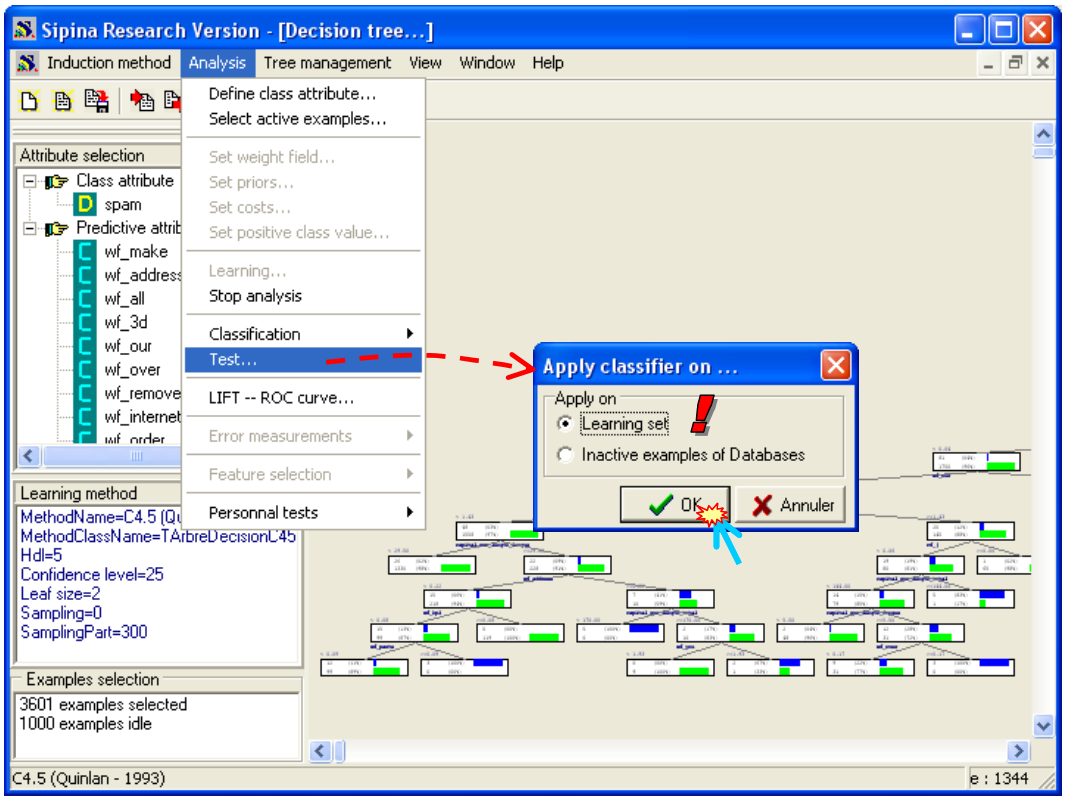
Ce n'est pas notre propos dans ce didacticiel. Nous voulons avant tout produire un modèle prédictif performant. Voyons justement ce qu'il en est de la qualité globale de l'arbre.

3.6 Evaluation sur les échantillons apprentissage et test

3.6.1 Matrice de confusion et erreur en resubstitution

Première démarche, confronter les prédictions du modèle avec les vraies valeurs de la variable dépendante. Les résultats sont recensés dans un tableau croisé dit « matrice de confusion » : en ligne, les modalités vraies, en colonne, les modalités prédites.

Pour élaborer cette matrice, nous activons le menu ANALYSIS /TEST, une boîte de dialogue apparaît. Elle nous demande de spécifier la fraction du fichier sur lequel nous voulons réaliser la confrontation. Pour l'instant, nous travaillons exclusivement sur l'échantillon d'apprentissage (LEARNING SET).



Une nouvelle fenêtre contenant la matrice de confusion apparaît, accompagnée du taux d'erreur inscrit dans la barre de statut (Figure 3). Nous lisons :

- 80 Spams ont été classés comme messages licites ;
- 54 messages licites ont été classés comme Spams ;
- Les autres messages ont été classés correctement (1342 spams en spams, 2125 messages licites à juste titre).

Dans une matrice de confusion de la forme suivante :

Ligne : valeurs observées Colonne : valeurs prédites	Prédiction positive	Prédiction négative
+ (modalité « positive » : spams)	a	b
-	c	d

Nous déduisons le taux d'erreur : $e = \frac{b+c}{a+b+c+d}$

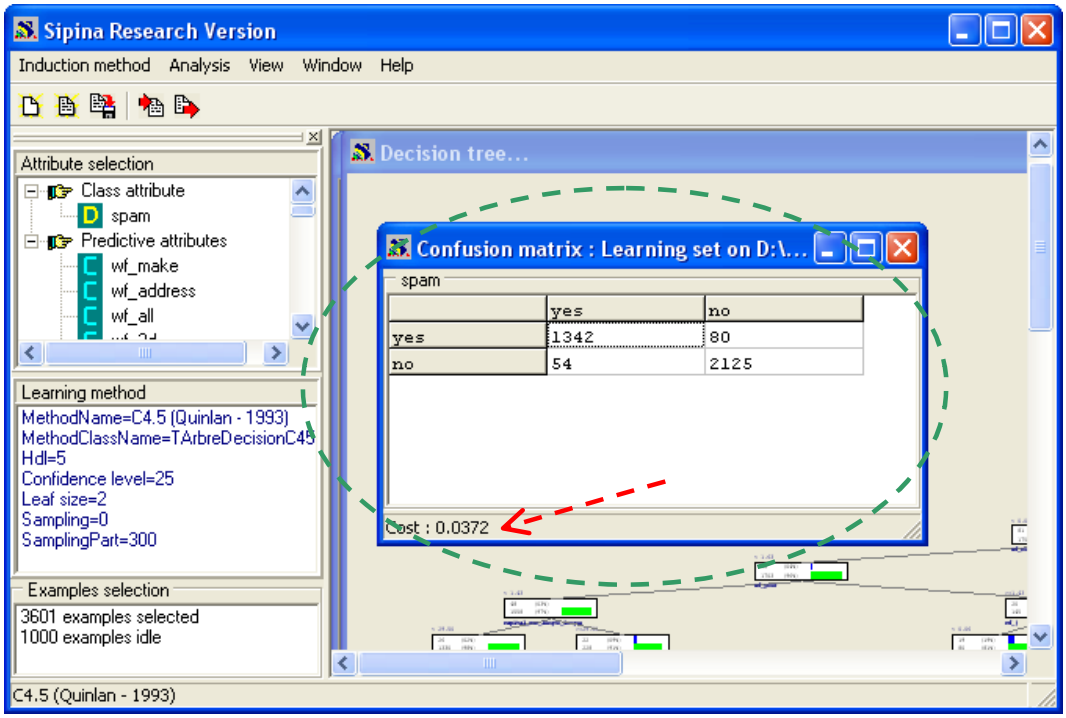


Figure 3 – Matrice de confusion et erreur en resubstitution

Dans notre cas, l'erreur en resubstitution est

$$e_r = \frac{80+54}{3601} = 3.7\%$$

Deux points de comparaison pour apprécier au mieux cette valeur : si nous classons tous les e-mails comme Spams, nous aurions un taux d'erreur de 2179/3601 = 60.5% ; si nous classons tous les messages comme licites, le taux d'erreur serait de 1422 / 3601 = 39.5%.

Il semble que le modèle soit très performant. En réalité, on n'en est pas très sûr car, surtout s'agissant des arbres de décision, le taux d'erreur en resubstitution est (presque) toujours trop optimisme. L'écart par rapport au taux théorique est d'autant généreux que l'arbre de décision est grand. Ce qui est notre cas. Nous obtenons les informations sur les caractéristiques de l'arbre en activant (après avoir activé la fenêtre de l'arbre de décision) le menu TREE MANAGEMENT / GENERAL INFORMATIONS. Une boîte de dialogue résume les principales informations (Figure 4).

Notre arbre comporte 74 feuilles, il propose donc 74 règles. Le nombre moyen d'observations sur une feuille est 3601 / 74 ~ 48. Plus faible sera cette valeur, plus instable sera l'arbre, et moins l'erreur empirique sera crédible pour donner une idée des performances de l'arbre. Dans notre cas, chaque règle est portée en moyenne par 48 observations c.-à-d. avec un support de 48 / 3601 ~ 1.4%. C'est assez problématique. Il paraît plus qu'opportun de mesurer l'erreur sur le fichier test.

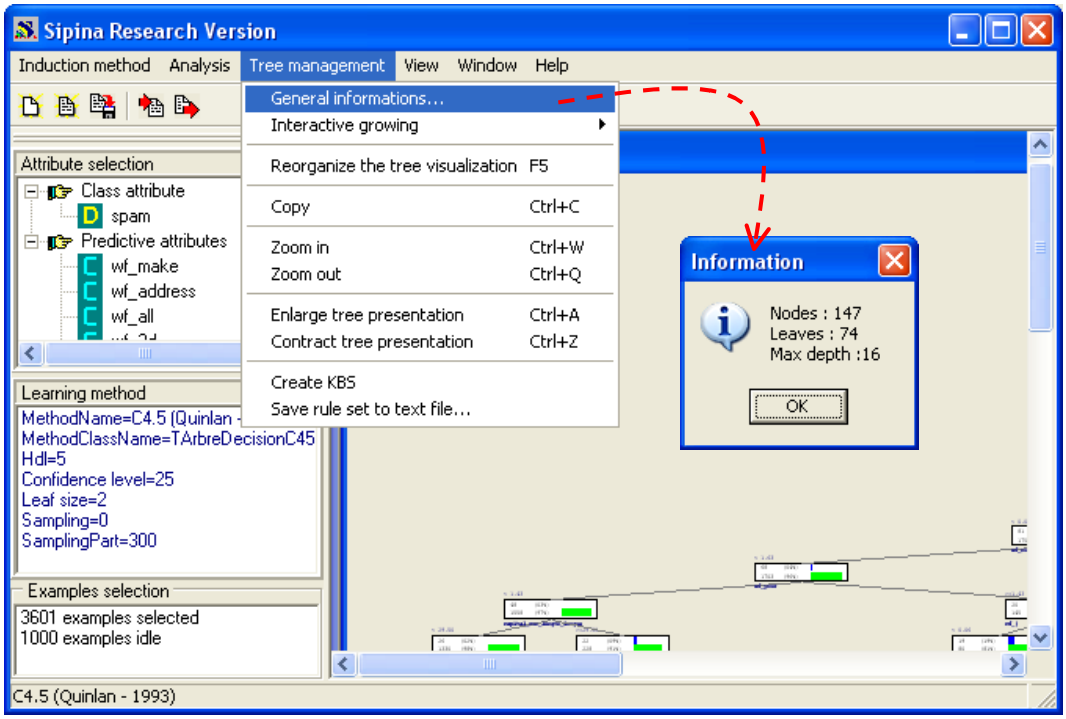
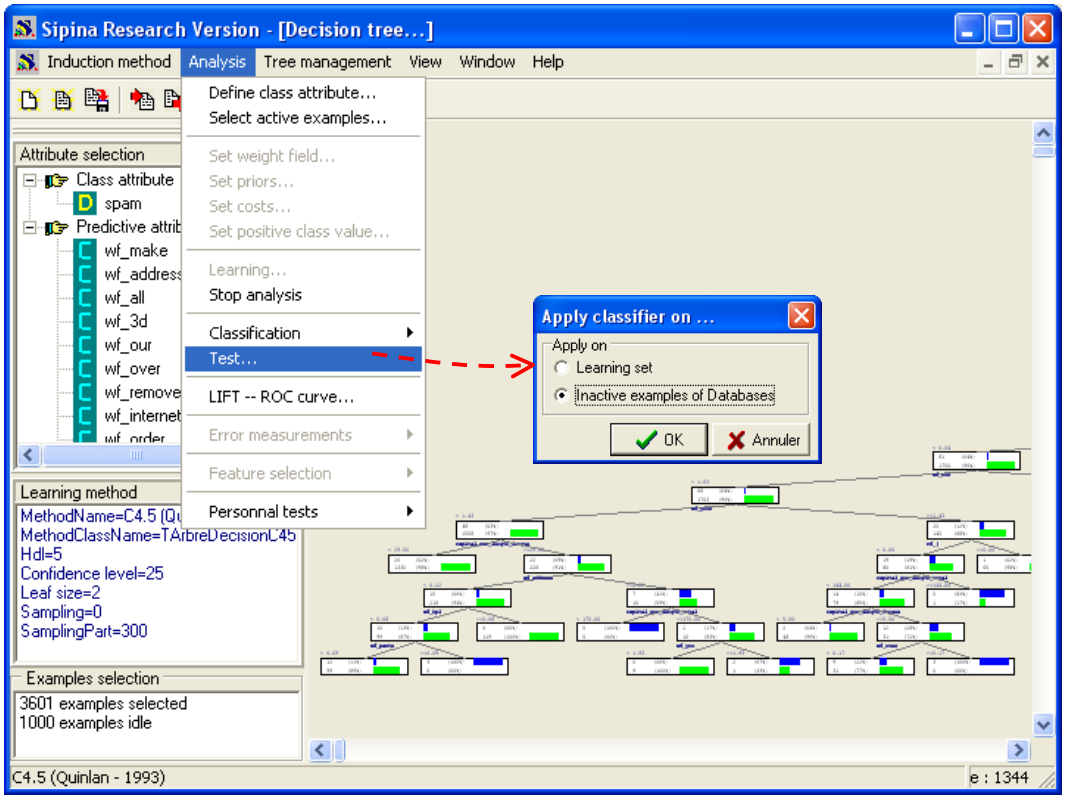


Figure 4 – Informations sur les caractéristiques de l'arbre

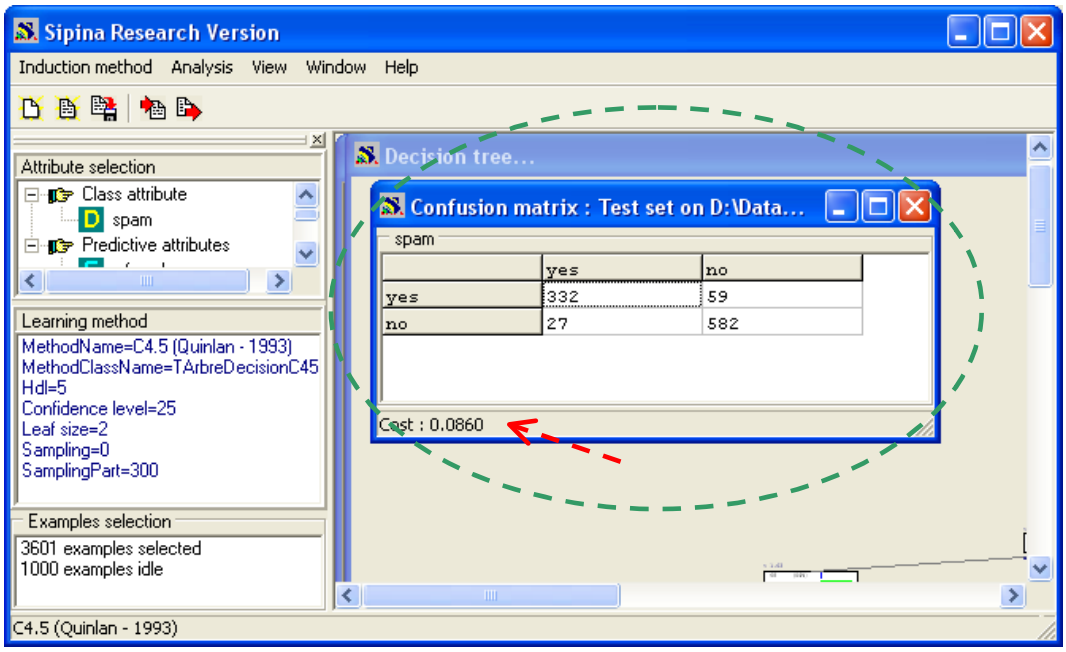
3.6.2 Matrice de confusion et erreur en test

Nous voulons cette fois-ci évaluer les performances de l'arbre sur la fraction des données que nous avons initialement mis de côté : l'échantillon test.

De nouveau, nous activons le menu ANALYSIS / TEST, mais dans la boîte de dialogue qui apparaît, nous choisissons l'option INACTIVE EXAMPLES OF DATABASES.



La matrice de confusion est affichée. La somme totale des valeurs du tableau est bien égal à 1000, l'effectif des observations réservées pour le test.



L'erreur en test est

$$e_t = \frac{59+27}{1000} = 8.6\%$$

C'est un estimateur non biaisé des véritables performances de l'arbre de décision. Pour un message électronique à jauger, la probabilité de le classer de manière erronée est de 8.6%.

3.6.3 Autre indicateurs : rappel, précision et F-Mesure

Dans notre étude, et c'est le cas très souvent, les deux modalités de la variable à prédire ne revêtent pas la même importance, ni la même signification. La modalité positive, les messages non sollicités, est définie clairement ; la modalité négative, les messages licites, regroupe en réalité toute une série de messages hétéroclites, qui va du message purement professionnel entre deux collègues jusqu'aux commentaires sur les matches de foot du week-end entre 2 copains. Le taux d'erreur ne tient pas compte de cela pour évaluer les performances du classement.

Pour cette raison, dans la recherche d'informations et la catégorisation de texte[5], il est souvent conseillé d'utiliser 2 autres critères qui reflètent mieux les préoccupations du domaine :

- Le rappel $r = \frac{a}{a+b}$ indique la proportion de la modalité positive que l'on a réussit à recouvrer. Dans notre cas, il est égal à r = 332/(332 + 59) = 84.9% c.-à-d. notre système a détecté 84.9% des messages délictueux qui sont passés sur le serveur.
- La précision $p = \frac{a}{a+c}$ indique la proportion des prédictions positives qui sont réalisées à juste titre. Dans notre exemple, p = 332/(332 + 27) = 92.5% c.-à-d. pour chaque message stoppé par le filtre anti-spam, nous avons une probabilité de 92.5% qu'il soit réellement illicite.

Un modèle sera bon s'il combine un rappel élevé, il ne laisse passer aucun spam, et précis, à chaque fois qu'il stoppe un message, il le fait à juste titre. On comprend aisément que ces deux critères sont antinomiques, à force de vouloir retrouver des messages illicites, nous augmentons le risque de stopper à tort un bon message ; si nous voulons stopper qu'à coup sûr, nous prenons le risque de laisser passer un certain nombre de messages illicites.

Manipuler simultanément deux valeurs n'est jamais bien facile, surtout lorsque nous voulons comparer les performances de plusieurs modèles. On propose dans la littérature un critère synthétique, dit F-Mesure, qui est en fait une moyenne harmonique entre le rappel et la précision. Elle est définie de la manière suivante :

$$F_\alpha = \frac{(1+\alpha^2)\times r \times p}{\alpha^2 \times p + r}$$

Plus grand sera la F-Mesure, meilleur sera le modèle. Idéalement : r = 1, p = 1, alors F = 1.

α est le poids que l'on accorde à la précision ou au rappel. Si nous leur accordons le même poids, nous choisirons $\alpha = 1$ et la F-Mesure serait égale à $F_1 = \frac{2\times 0.925 \times 0.849}{1\times 0.925 + 0.849} = 0.885$ pour notre problème de spams.

Les autres valeurs de α couramment utilisées dans la littérature sont : $\alpha = 0.5$, on accorde deux fois plus d'importance à la précision qu'au rappel ; et $\alpha = 2$, on accorde deux fois plus d'importance au rappel par rapport à la précision.

La bonne valeur de α dépend des contraintes du domaine et des objectifs de l'étude. Nous pouvons aussi définir des plages de valeurs pour étudier le comportement des modèles sur différents scénarios.

[5] http://en.wikipedia.org/wiki/Information_retrieval

4 Introduction des coûts non symétriques

Pour un administrateur système, les deux types d'erreurs que peut faire un filtre anti-spam ne sont absolument pas symétriques. Laisser passer un spam entraîne au pire la grogne des utilisateurs qui se lassent d'avoir à supprimer manuellement les messages vantant tel ou tel produit pharmaceutique augmentant notre Q.I., ou nous promettant de recevoir des millions si nous laissons nos coordonnées bancaires. Supprimer un message licite l'expose à des plaintes d'utilisateurs qu'il aura privé de messages peut être primordiaux. Bien que dans ce cas, vu que ces messages sont perdus pour tout le monde, le destinataire n'est pas censé savoir qu'il est passé à côté de la chance de sa vie.

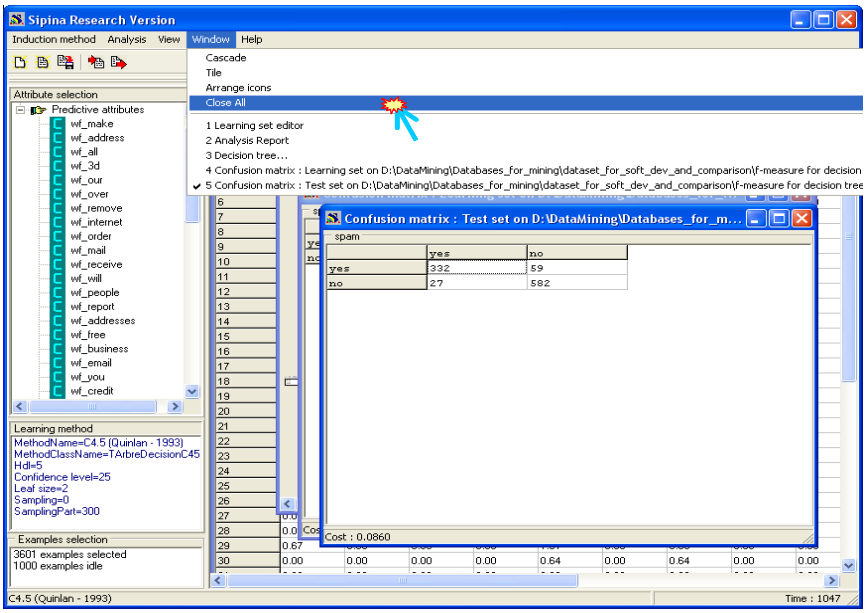
En tous les cas, dans ce contexte, la précision est autrement plus importante que le rappel, nous devons d'une part en tenir compte pour l'évaluation, d'autre part l'intégrer dans le système d'apprentissage de manière à ce que l'on produise un modèle de prédiction adaptée à la configuration choisie. Mettons que dans notre cas du filtrage des e-mails, **la précision est 10 fois plus importante que le rappel**. En ce qui concerne l'évaluation, nous pouvons modifier le paramètre de la F-Mesure en adoptant la pondération $\alpha = 0.1$.

La question est plus délicate pour la construction du modèle. L'approche le plus simple est d'intégrer cette nouvelle contrainte sous forme de matrice coûts de mauvais classement et d'utiliser une méthode qui en tient compte explicitement.

Dans SIPINA, nous avons développé une méthode « sensible aux coûts de mauvais classement » dérivée de C4.5 (Chauchat et al., 2001), inspirée des travaux de Bradford et al. (1998). Il faut de nouveaux spécifier la méthode et introduire la matrice de coût avant de procéder à l'apprentissage.

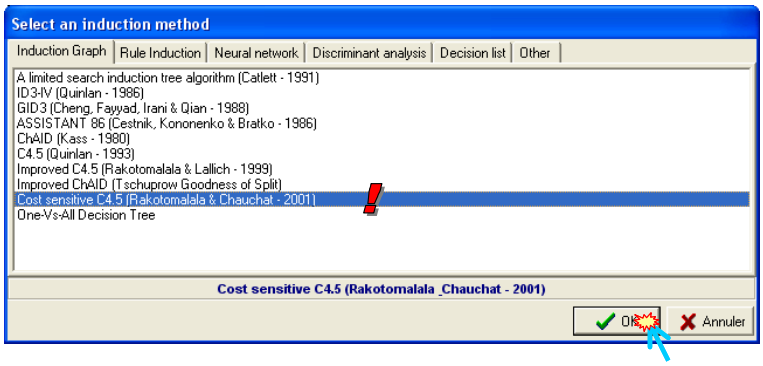
4.1 Stopper l'analyse précédente

Tout d'abord, il faut clôturer l'analyse précédente. Pour cela, le plus simple est d'activer le menu WINDOW / CLOSE ALL. Nous clôturons ainsi l'analyse et fermons par la même occasion toutes les fenêtres relatives à cette session de travail.

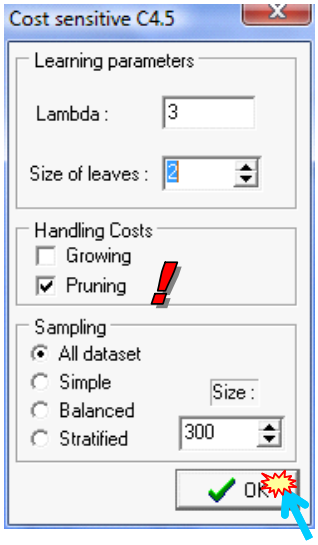


4.2 Choisir une méthode sensible aux coûts

Etape suivante, nous devons sélectionner une méthode adaptée. Nous activons de nouveau le menu INDUCTION METHOD / STANDARD ALGORITHM, et dans la boîte de dialogue qui apparaît, nous choisissons la méthode COST SENSITIVE C4.5 (CHAUCHAT & RAKOTOMALALA, 2001).



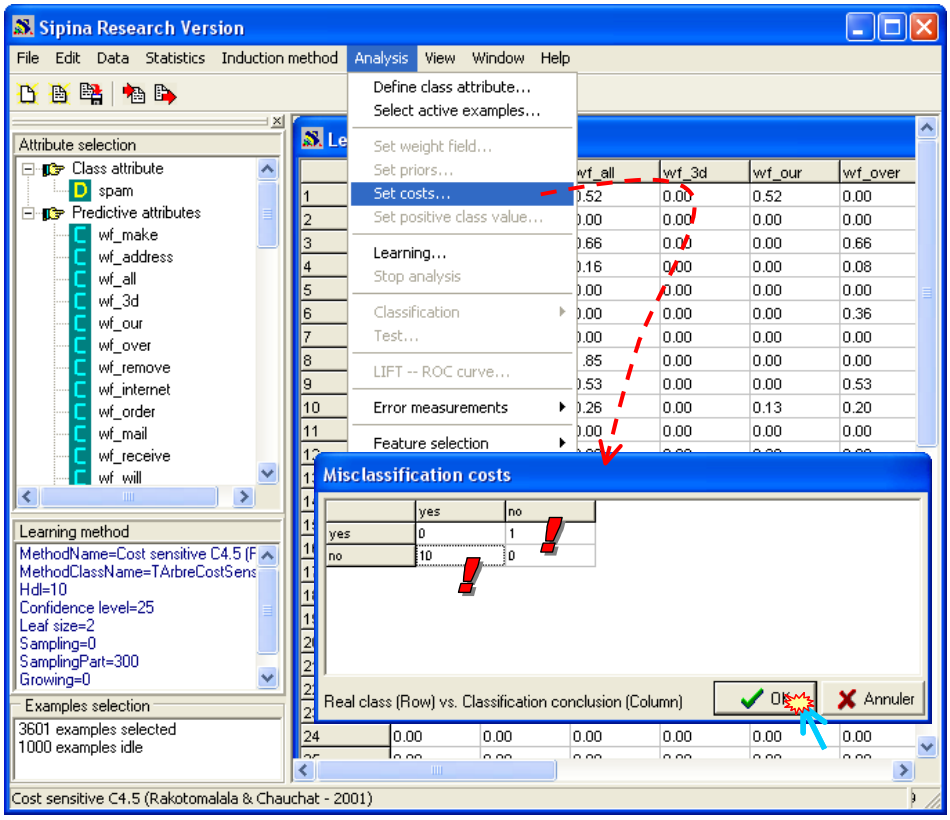
Après avoir validé, la fenêtre de paramétrage de la méthode apparaît. Nous retrouvons les paramètres de C4.5, mais avec la possibilité de prendre en compte les coûts lors de la phase d'expansion de l'arbre (rarement efficace) et lors de la phase d'élagage (indispensable). Nous validons les options par défaut.



Le paramètre LAMBDA est utilisé pour l'estimation « laplacienne » des probabilités. Il faut surtout le comprendre comme un paramètre de lissage qui permet de « lisser » l'estimation des probabilités sur les petits effectifs. S'il est égal à 0, les probabilités sur les feuilles seront estimées à l'aide des fréquences brutes.

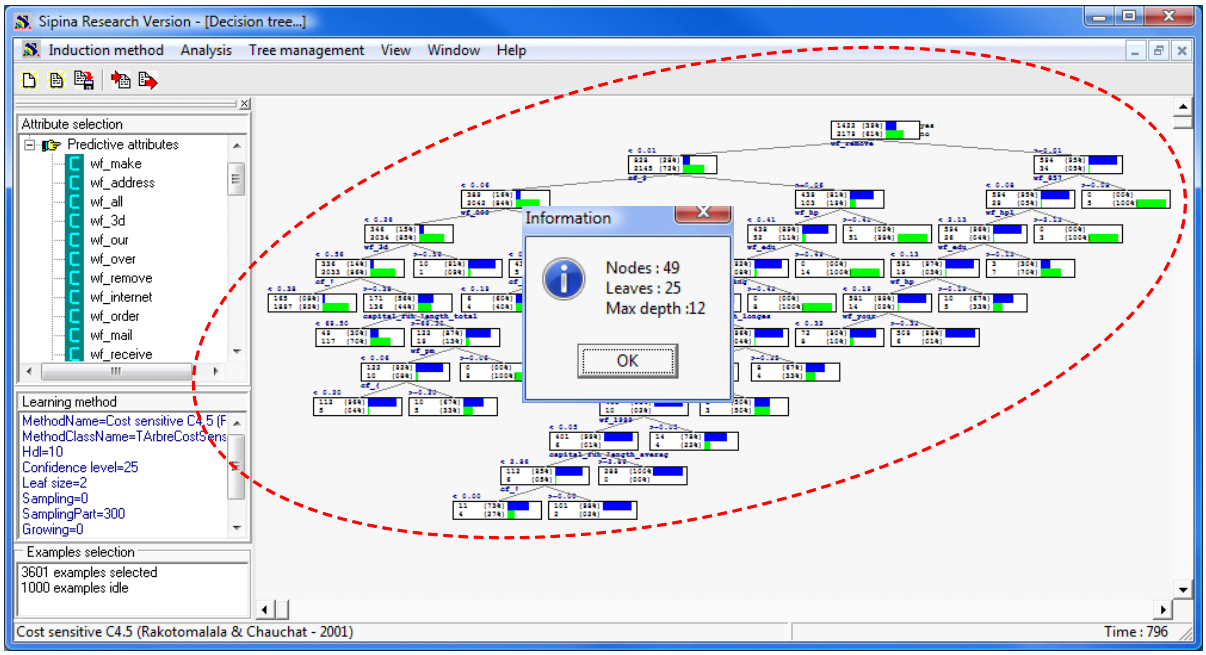
4.3 Définir les coûts de mauvais classement

Choisir une méthode sensible aux coûts n'est réellement intéressant que si nous pouvons lui spécifier justement quel système de coûts utiliser. Nous activons le menu ANALYSIS / SET COSTS. Dans la boîte de dialogue qui apparaît, nous spécifions bien que l'affectation d'un message licite en spam coûte 10, et que l'affectation d'un spam en message licite coûte 1. En d'autres termes, la précision est 10 fois plus importante que le rappel. Nous validons ce choix en cliquant sur le bouton OK.

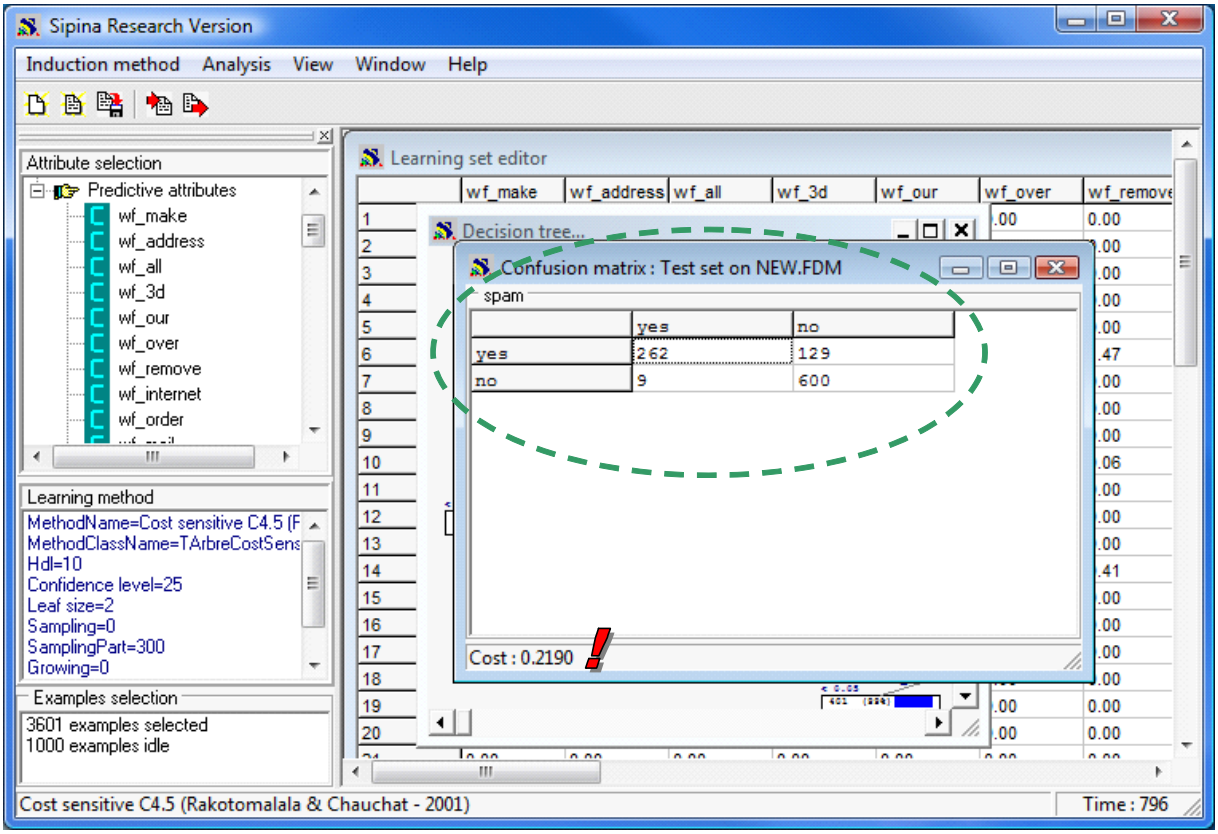


4.4 Apprentissage et test

Il ne nous reste plus qu'à lancer le traitement en activant le menu ANALYSIS / LEARNING. L'arbre est plus petit que précédemment, avec 36 feuilles quand même.



C'est maintenant l'heure de vérité. Nous appliquons ce modèle sur les mêmes données test (ANALYSIS / TEST / INACTIVE EXAMPLES OF DATABASES) que précédemment. Nous obtenons une nouvelle matrice de confusion, différente bien entendu.



Pour apprécier pleinement les apports de cette méthode sensible aux coûts, nous recensons les principaux indicateurs de performance dans un tableau récapitulatif, en prenant comme repère les résultats fournis par la méthode C4.5 « standard » (Tableau 1).

Indicateur	C4.5 « standard » (M1)	C4.5 « sensible aux coûts » (M2)
Vrai positifs[6]	332	262
Faux positifs[7]	27	9
Taux d'erreur	8.6%	13.8%
Rappel	84.9%	67.0%
Précision	92.5%	96.7%
F-Mesure (0.1)	0.924	0.963

Tableau 1 – Indicateurs calculés à partir de la matrice de confusion en test

Si l'on se réfère au taux d'erreur en test, notre modèle sensible au coût (M2) est moins bon que C4.5 « standard » (M1). Mais ce n'est pas vraiment un problème. En effet, il est totalement inapproprié pour évaluer notre nouvelle configuration. Penchons nous sur les autres indicateurs :

- M2 classe moins d'observations en positif (262 + 9 = 271) que M1 (332 +27 = 359). C'est normal, une mauvaise affectation « classer en spam un message licite » est fortement pénalisée.
- Cela se retrouve dans le nombre de faux positifs. Il y en a 3 fois moins avec M2 (9 contre 27).
- La précision est ainsi nettement meilleure (96.7% pour M2 contre 92.5% pour M1).
- Cela se paie au niveau du rappel, il est fortement dégradé pour M2 (67.0% contre 84.9%).

[6] Les observations que l'on a classées « positifs » et qui le sont réellement.

[7] Les observations que l'on a classées « positifs » et qui se révèlent être des « négatifs »

- Enfin globalement, si l'on tient compte à la fois du rappel et de la précision avec la pondération choisie (poids 10 fois plus élévé pour la précision), M2 se révèle être légèrement meilleur (0.963 contre 0.924). C'est ce que l'on cherchait. Il est heureux que l'apprentissage ait pris en compte les spécifications que nous lui avons fournies.

Moralité de tout ceci : si Monica nous écrit vraiment, avec notre nouveau filtre anti-spam, il n'y a que 3.3% (3.3% = 100% - 96.7%) de chances que sa missive passe indûment à la trappe. Nous pouvons garder espoir… mais en attendant il faudra éliminer manuellement le surcroît de spams qui inondent notre boîte e-mail.

5 Conclusion

Ce didacticiel avait un double objectif. Le premier était de montrer qu'il est possible de réaliser le schéma apprentissage – test sur une subdivision du fichier prédéfini par l'utilisateur dans SIPINA. Cela permet de comparer objectivement les performances de plusieurs algorithmes implémentés dans SIPINA, mais aussi implémentés dans d'autres logiciels qui savent manipuler les fichiers apprentissage et test. Quasiment la totalité des logiciels de Data Mining en fait.

Le second sujet important abordé dans ce didacticiel est l'intégration des coûts de mauvais classement dans la construction des modèles. Cette fonctionnalité est trop rare dans les logiciels de Data Mining. Elle est implémentée dans SIPINA mais malheureusement très peu connue. Il me paraissait intéressant de montrer comment le mettre en œuvre, quoi en penser, et comment lire les résultats dans cette configuration.