1 Objectif

Manipulation du format de fichier « sparse » dans Tanagra.

Le traitement statistique des documents non-structurés est un vrai challenge. « Non-structuré » est a priori difficile à cerner[1]. Dans le cadre du data mining, on l'utilise *généralement* pour toute donnée qui s'écarte du classique attribut-valeur où l'observation est décrite par un vecteur de longueur constante de caractéristiques (ex. un client est décrit par son âge, son revenu, …).

Ainsi, même si intrinsèquement le texte et l'image, pour ne citer qu'eux, ont une structure interne cohérente, un pré-traitement est nécessaire pour parvenir à une description « attribut-valeur ». Cette transformation engendre des tableaux de données volumineux avec certaines spécificités. Il est possible de les exploiter pour proposer un format de description parcimonieux, et aboutir à des fichiers de taille réduite. Il s'agit en quelque sorte d'une forme de compression sans perte. Mais le fichier reste lisible avec un simple éditeur de texte. Comme nous le constaterons pour l'exemple que nous traiterons, la réduction de la taille du fichier n'est absolument pas négligeable.

Dans ce tutoriel, nous décrivons le format de fichier « sparse » (format « creux ») pour la régression et le classement reconnu par **Tanagra** (depuis la version **1.4.44**), il est directement inspiré du format traité par les bibliothèques de calcul « svmlight », « libsvm » et « libcvm »[2]. Nous illustrons son exploitation dans un processus de catégorisation de texte appliquée à la base Reuters, bien connue en data mining[3].

2 Le format « sparse »

2.1 Du tableau attribut valeur à la description sparse

Le point de départ est toujours une collection de documents. Les algorithmes d'apprentissage ne savent pas traiter directement du texte brut. Il nous faut donc le transformer en tableau de valeurs où chaque observation (ligne) correspond à un texte, et les colonnes sont des descripteurs extraits automatiquement. La solution la plus simple est de d'utiliser le mot (le « terme ») comme descripteur.

Voyons un exemple simple pour cerner l'idée. Nous avons 3 documents :

A. « Le soleil brille dans le ciel ».
B. « Le ciel est bleu ».
C. « Voilà le produit pour faire briller ».

Nous utilisons la simple occurrence des termes[4] c.-à-d. nous comptons l'apparition de chaque mot dans chaque document. En attribut-valeur, nous obtenons un tableau à 3 lignes et 12 colonnes.

[1] http://en.wikipedia.org/wiki/Unstructured_data

[2] http://svmlight.joachims.org/ ; http://www.csie.ntu.edu.tw/~cjlin/libsvm/ ; http://c2inet.sce.ntu.edu.sg/ivor/cvm.html

[3] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

[4] Les subtilités sont très nombreuses pour réduire le nombre de colonnes. Il faudrait plus qu'un tutoriel pour détailler les différentes stratégies (ex. voir http://www.statsoft.com/textbook/text-mining/).

Document	1 le	2 soleil	3 brille	4 dans	5 ciel	6 est	7 bleu	8 voilà	9 produit	10 pour	11 faire	12 briller
A	**2**	**1**	**1**	**1**	**1**	0	0	0	0	0	0	0
B	**1**	0	0	0	**1**	**1**	**1**	0	0	0	0	0
C	**1**	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**

Deux caractéristiques sautent aux yeux. (1) Le tableau de données peut être rapidement très volumineux. En effet, nous avons la liste des mots susceptibles d'apparaître dans l'ensemble des documents en colonne. Leur nombre sera d'autant plus élevé que la taille et la quantité des textes augmente. (2) Le tableau est rempli de zéro. En effet, peu de mots, dans l'ensemble des termes possibles, apparaissent dans chaque document. Stocker explicitement ces informations dans le fichier de données n'est pas très judicieux. Il faut mettre en place une stratégie de compression.

Le format « sparse » consiste à ne recenser que les termes qui apparaissent réellement dans chaque document. Pour notre exemple, en considérant qu'à chaque mot est associé un numéro, nous stockerons les données sous la forme suivante :

Document	Description
A	`1:2 2:1 3:1 4:1 5:1`
B	`1:1 5:1 6:1 7:1`
C	`1:1 8:1 9:1 10:1 11:1 12:1`

Chaque valeur non nulle est préfixée par son numéro de colonne. En contrepartie, les colonnes qui n'apparaissent pas dans la ligne correspondent implicitement à la valeur 0. A la sortie, nous espérons obtenir une forte réduction de l'espace nécessaire au stockage. Elle sera d'autant plus spectaculaire que la proportion des cases nulles est élevée dans notre tableau de données.

2.2 Le format « sparse »pour la régression et le classement

Dans l'analyse prédictive, chaque observation est étiquetée : soit par un nombre indiquant son groupe d'appartenance si l'on est dans le cadre de l'apprentissage supervisé ; soit par la valeur de la variable dépendante si l'on est dans le cadre de la régression. A l'instar de ce qui est proposé par les librairies Svmlight et Libsvm, l'étiquette est placée en première position dans la ligne.

Reprenons notre exemple ci-dessus. Mettons que les documents A et B appartiennent à la première catégorie « 1 », et que C appartient à « -1 ». Notre fichier prendra la forme suivante :

```
1 1:2 2:1 3:1 4:1 5:1
1 1:1 5:1 6:1 7:1
-1 1:1 8:1 9:1 10:1 11:1 12:1
```

Que ce soit en classement ou en régression, l'étiquette correspond toujours à une valeur numérique dans le fichier. Un recodage sera nécessaire dans Tanagra pour qu'il la considère comme une indicatrice de la catégorie d'appartenance.

2.3 La base « reuters (money fx) »

Nous traitons la base Reuters. C'est un classique de la catégorisation de texte. Nous disposons d'une collection de nouvelles. Chacune d'entre elle est indexée par une ou plusieurs catégories (TOPICS). Le document suivant par exemple est associé aux thèmes « money-fx » et « interest ».

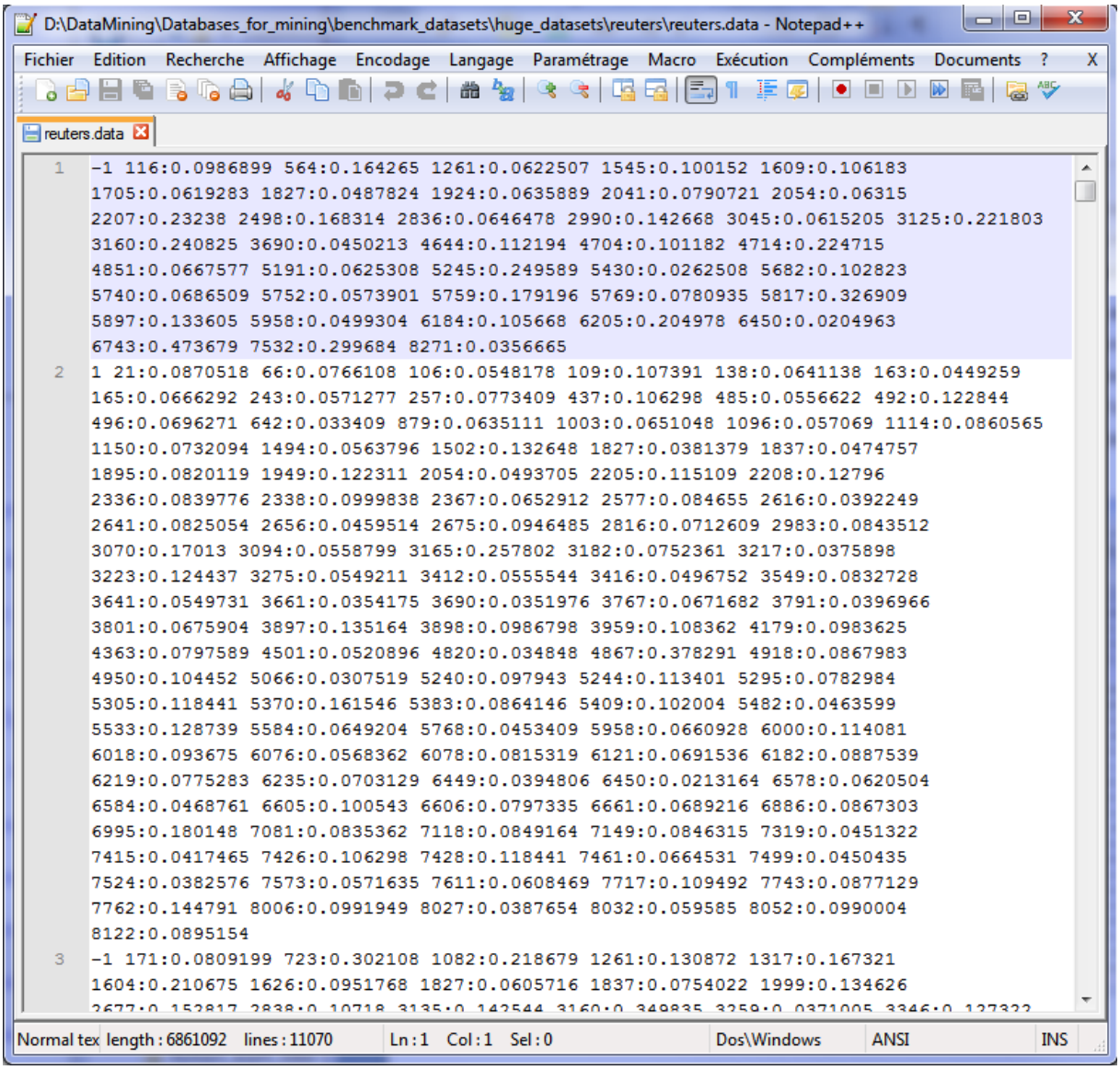
```
<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="5764"
NEWID="221">
<DATE>26-FEB-1987 21:05:51.60</DATE>
<TOPICS><D>money-fx</D><D>interest</D></TOPICS>
<PLACES><D>japan</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
&#5;&#5;&#5;RM
&#22;&#22;&#1;f0438&#31;reute
u f BC-AVERAGE-YEN-CD-RATES   02-26 0096</UNKNOWN>
<TEXT>&#2;
<TITLE>AVERAGE YEN CD RATES FALL IN LATEST WEEK</TITLE>
<DATELINE>      TOKYO, Feb 27 - </DATELINE><BODY>Average interest rates on yen
certificates
of deposit, CD, fell to 4.27 pct in the week ended February 25
from 4.32 pct the previous week, the Bank of Japan said.
    New rates (previous in brackets), were -
    Average CD rates all banks 4.27 pct (4.32)
    Money Market Certificate, MMC, ceiling rates for the week
starting from March 2          3.52 pct (3.57)
    Average CD rates of city, trust and long-term banks
    Less than 60 days          4.33 pct (4.32)
    60-90 days                 4.13 pct (4.37)
    Average CD rates of city, trust and long-term banks
    90-120 days             4.35 pct (4.30)
    120-150 days            4.38 pct (4.29)
    150-180 days            unquoted (unquoted)
    180-270 days            3.67 pct (unquoted)
    Over 270 days           4.01 pct (unquoted)
    Average yen bankers' acceptance rates of city, trust and
long-term banks
    30 to less than 60 days unquoted (4.13)
    60-90 days              unquoted (unquoted)
    90-120 days             unquoted (unquoted)
 REUTER
&#3;</BODY></TEXT>
</REUTERS>
```

L'objectif de la modélisation est de produire une fonction d'affectation qui permet d'associer automatiquement une nouvelle à sa catégorie d'appartenance.

En pratique, on préfère considérer le traitement sous un angle binaire pour simplifier. On cherche à savoir si un texte appartient ou non à une catégorie cible. Dans la base, les observations « positives » sont les nouvelles qui lui sont associées ; les « négatives » sont les autres, non étiquetées ou appartenant à d'autres catégories.

Concernant le problème traité dans ce tutoriel[5], la catégorie cible est « **money-fx** ». On cherche à savoir si les nouvelles sont relatives ou non à ce thème. La transformation du texte en tableau de valeurs numériques (on parle de « **pondération** ») a déjà été réalisée. La base comporte ainsi 8 315 descripteurs. Initialement, elle est subdivisée en deux parties : l'échantillon d'apprentissage comporte 7 770 lignes, l'échantillon test 3 299. Nous les avons concaténés dans un seul fichier « **reuters.data** ». Voici un aperçu des premières lignes.

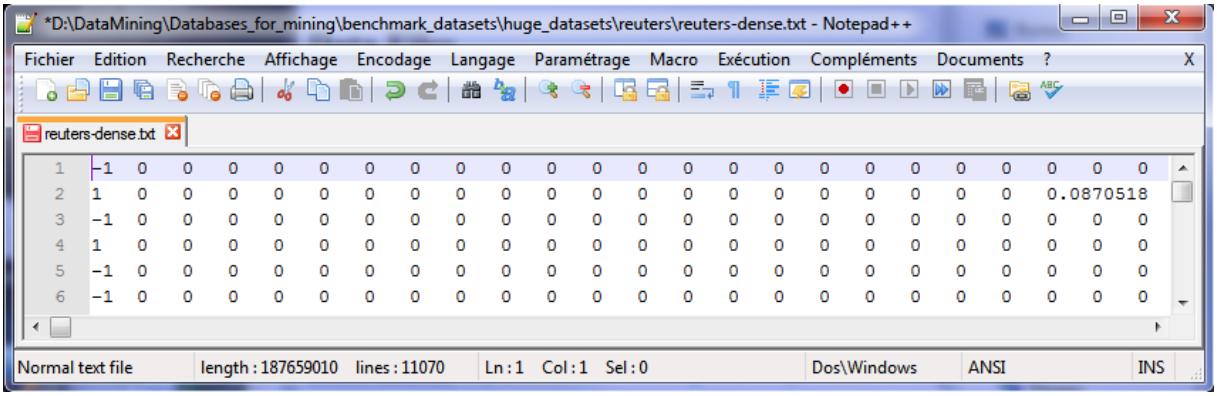
Note : le point décimal est toujours le « . » quel que soit le système d'exploitation.



La première nouvelle n'est pas associé au thème « money-fx ». La variable n°1 (V1) prend la valeur 0, V2 = 0, …, V116 = 0.0986899, etc. La seconde est associée au thème cible, avec V1 = 0, V2 = 0, …, V21 = 0.0870518, V13 = 0, etc.

La taille du fichier sur le disque est de 6 701 Ko. Nous avons tenté de le recoder en un format « dense » (attribut-valeur). Voici les premières lignes :

[5] Reuters (money-fx/non money-fx) data set, elle accompagne la librairie LIBCVM - http://c2inet.sce.ntu.edu.sg/ivor/cvm.html



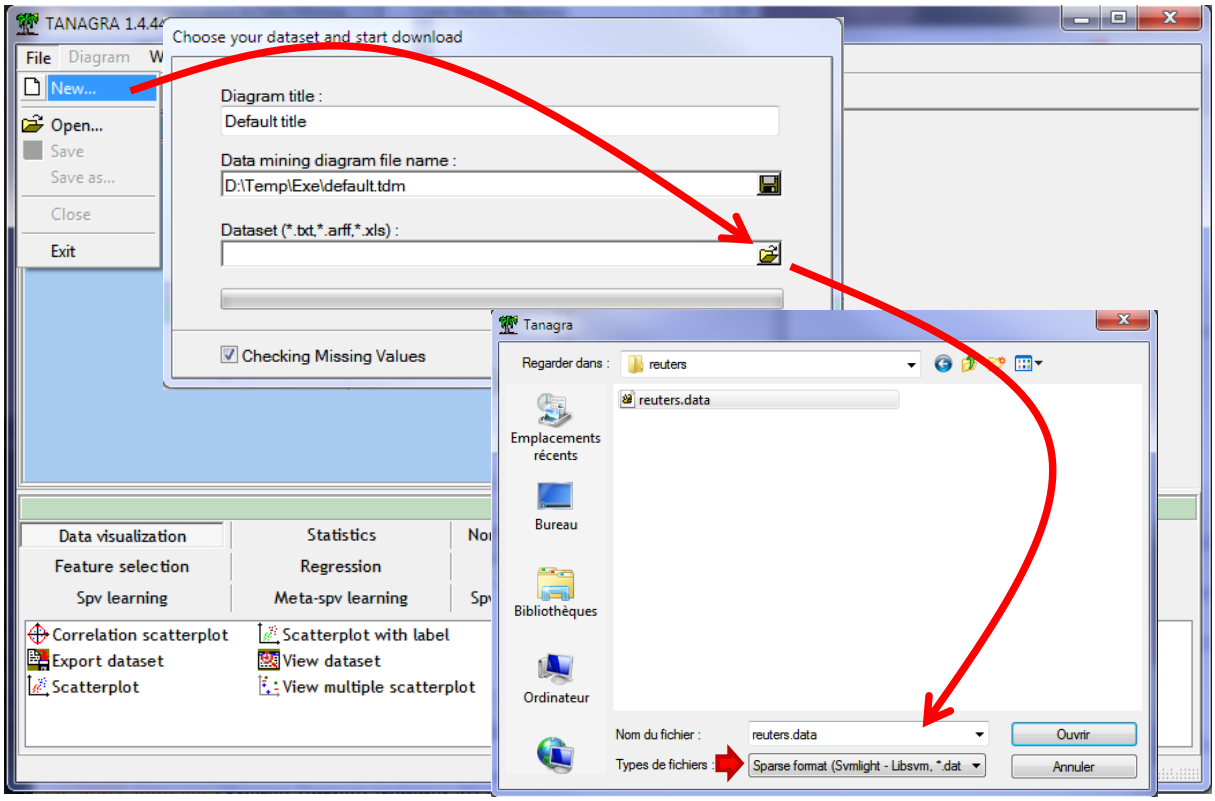
Nous pouvons passer d'une représentation à l'autre sans perte d'information. A la différence que sous cette nouvelle forme, le fichier occupe 183 309 Ko sur le disque. **Il est 27 fois plus volumineux** ! Voilà tout l'intérêt du format « sparse ».

3 Traitement avec Tanagra

A partir de la version 1.4.44, Tanagra sait gérer le format « sparse ». Il est très largement inspiré de la description fournie pour les librairies Svmlight, Libsvm, Libcvm. **L'extension de fichier doit toujours être « .dat » ou « .data » pour que Tanagra sache l'identifier**.

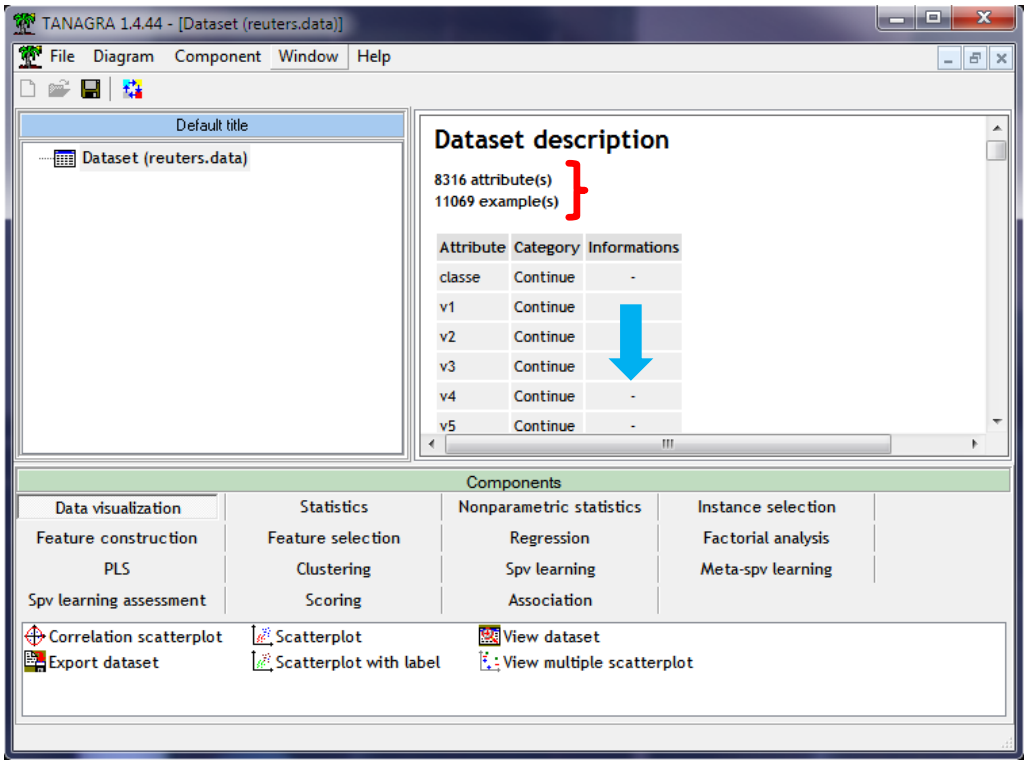
Dans ce qui suit, nous montrons comment importer ce type de fichier, opérer les transformations nécessaires pour qu'un apprentissage supervisé soit possible, construire le modèle prédictif sur l'échantillon d'apprentissage, et évaluer la qualité de la prédiction à l'aide d'une courbe ROC calculée sur l'échantillon test.

3.1 Importation des données



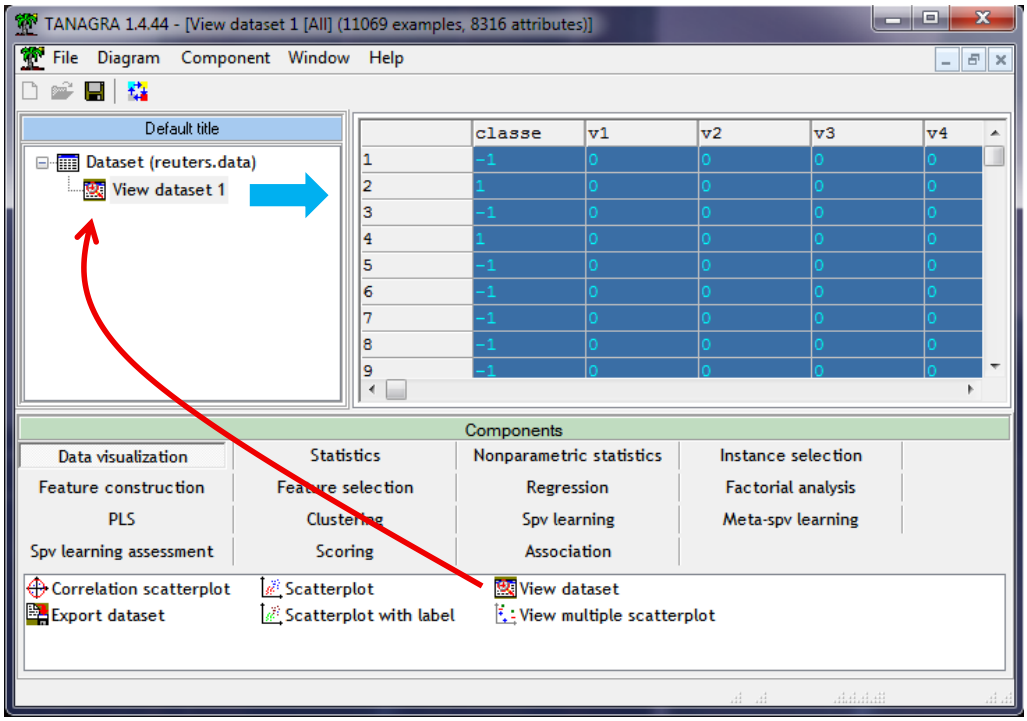
Après avoir démarré Tanagra, nous actionnons le menu FILE / NEW pour créer un nouveau projet et importer le fichier de données. Nous sélectionnons le fichier « **reuters.data** ».

Nous obtenons 11 069 observations et 8 316 attributs. Les noms de variables sont attribués automatiquement : « classe » pour la première colonne, « V1, V2, … » pour les suivantes.



3.2 Visualisation

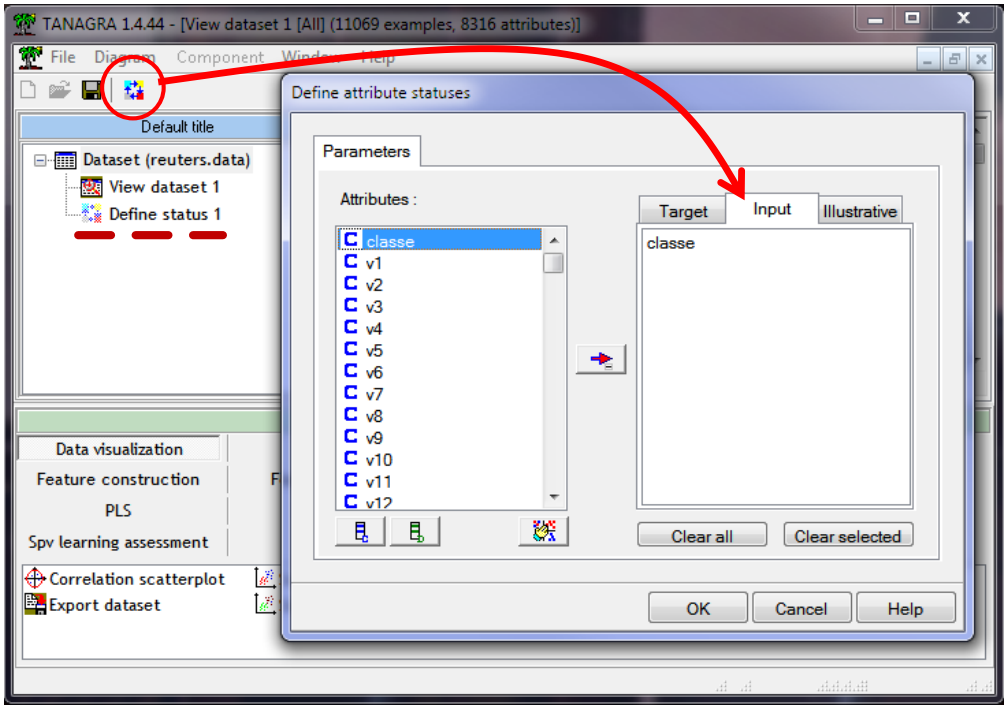
Vérifions tout de suite si les données ont été correctement chargées. Nous utilisons le composant de visualisation VIEW DATASET (onglet DATA VISUALIZATION).



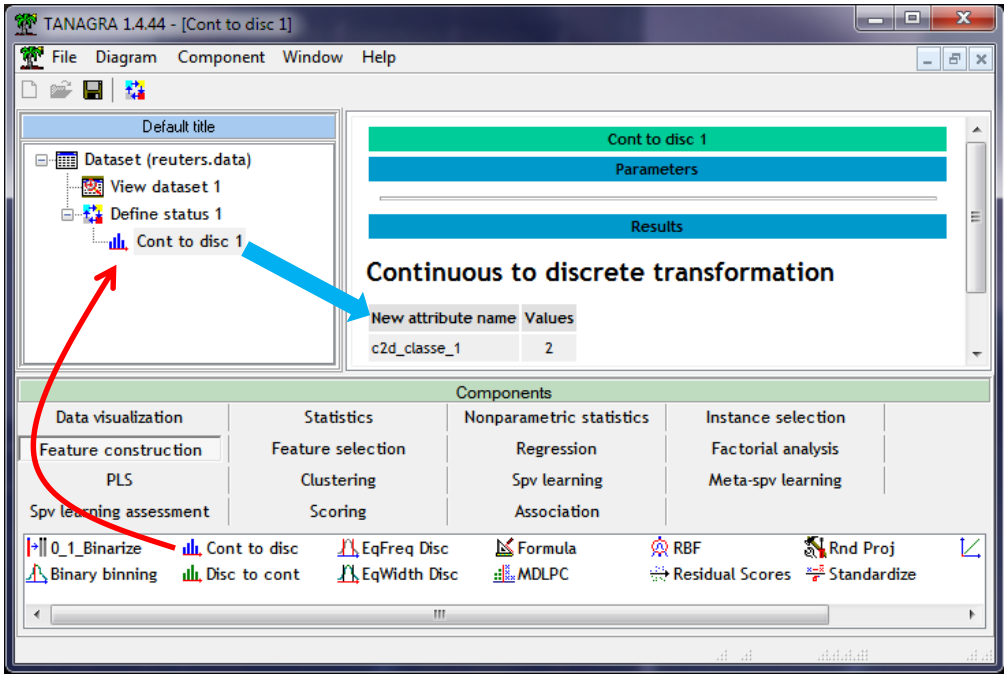
Le fichier est au format « sparse ». Mais les données sont codées au format classique « attribut-valeur » en interne. L'occupation mémoire peut rapidement devenir importante. Il est de 594 204 Ko à ce stade. Cette solution est un peu bancale, elle permet néanmoins de traiter de manière indifférenciée les données creuses et denses dans Tanagra.

3.3 Recodage de la variable cible

La variable « classe » est numérique, Tanagra pense qu'elle est quantitative. Il faut la recoder. Pour ce faire, nous introduisons le composant DEFINE STATUS et nous la plaçons en INPUT.



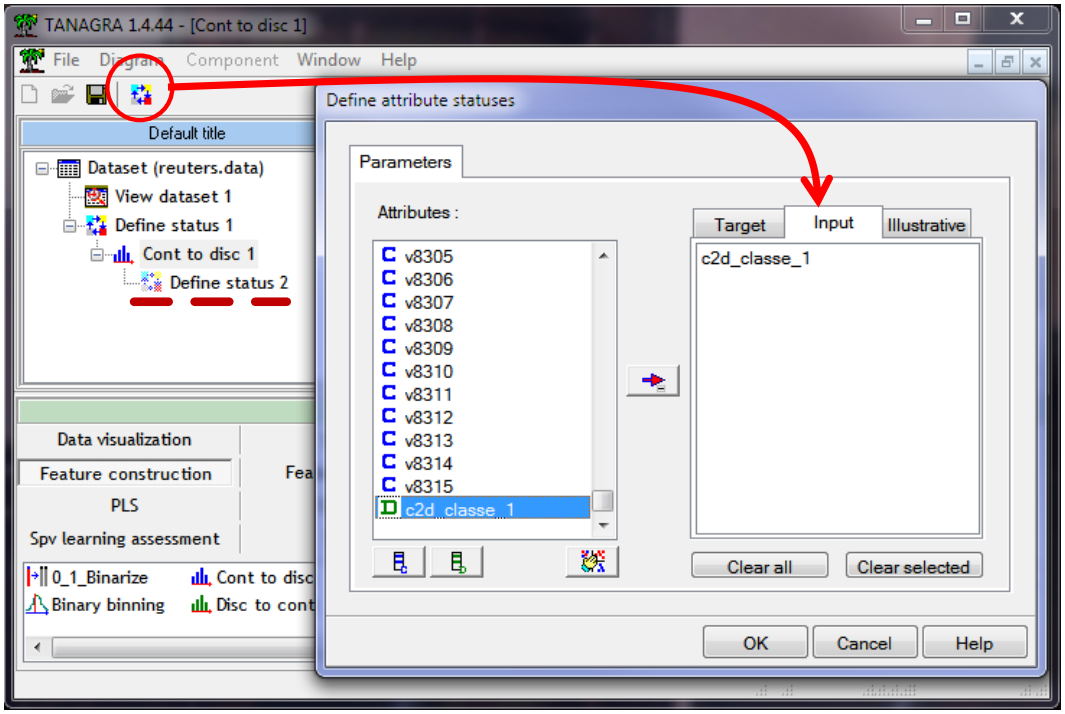
Nous insérons le composant CONT TO DISC (onglet FEATURE CONSTRUCTION). Il recode la colonne de valeurs numériques en variable catégorielle en associant les modalités aux valeurs distinctes.



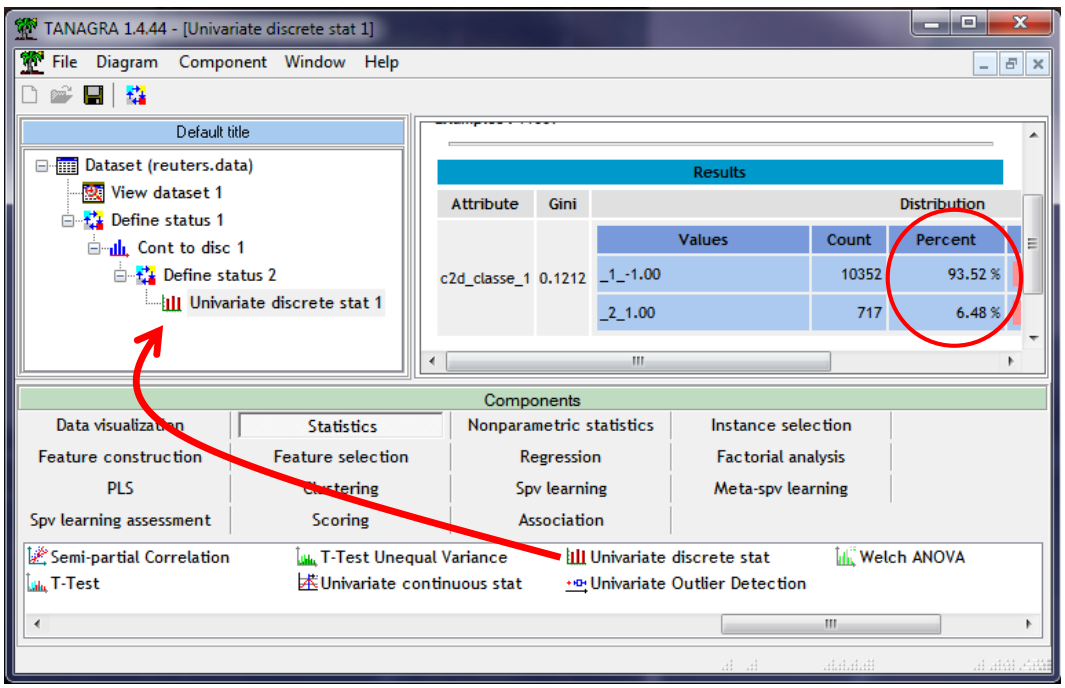
Après avoir cliqué sur le menu contextuel VIEW, nous constatons que la nouvelle variable C2D_CLASSE_1 est binaire. Les modalités correspondent aux valeurs « -1 » et « 1 » de classe.

3.4 Distribution des classes

Nous pouvons calculer la répartition des valeurs en insérant un autre DEFINE STATUS. Nous plaçons la nouvelle variable en INPUT.

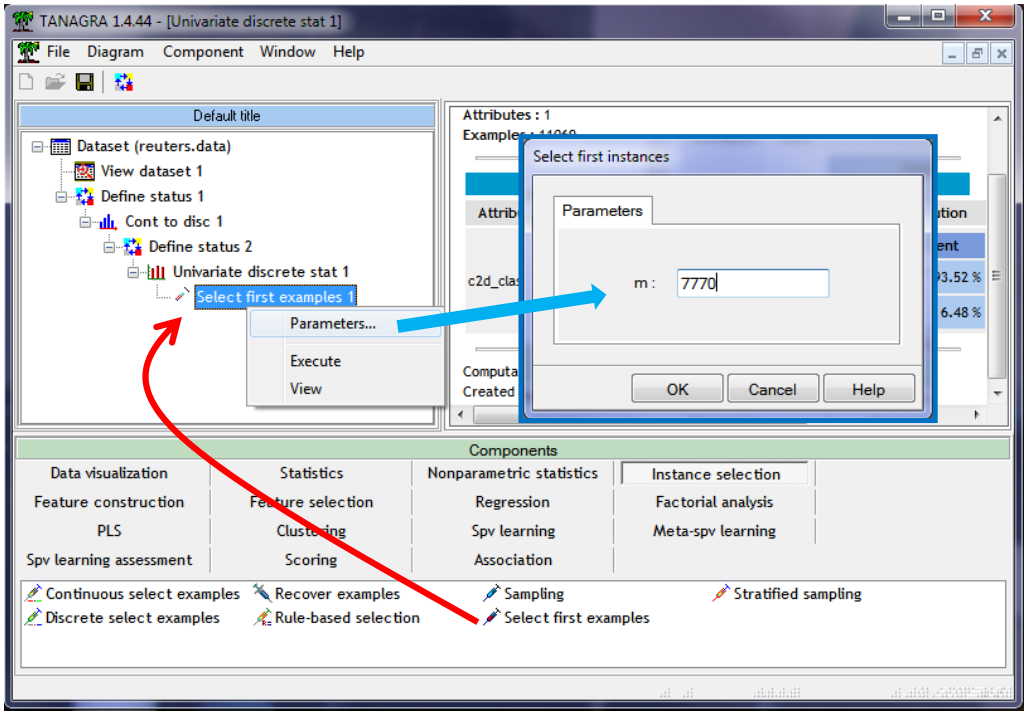


Puis, nous ajoutons le composant UNIVARIATE DISCRETE STAT (onglet STATISTICS). Nous constatons que les classes sont assez déséquilibrées. Les nouvelles correspondant à la modalité cible représentent 6.48% des observations disponibles.



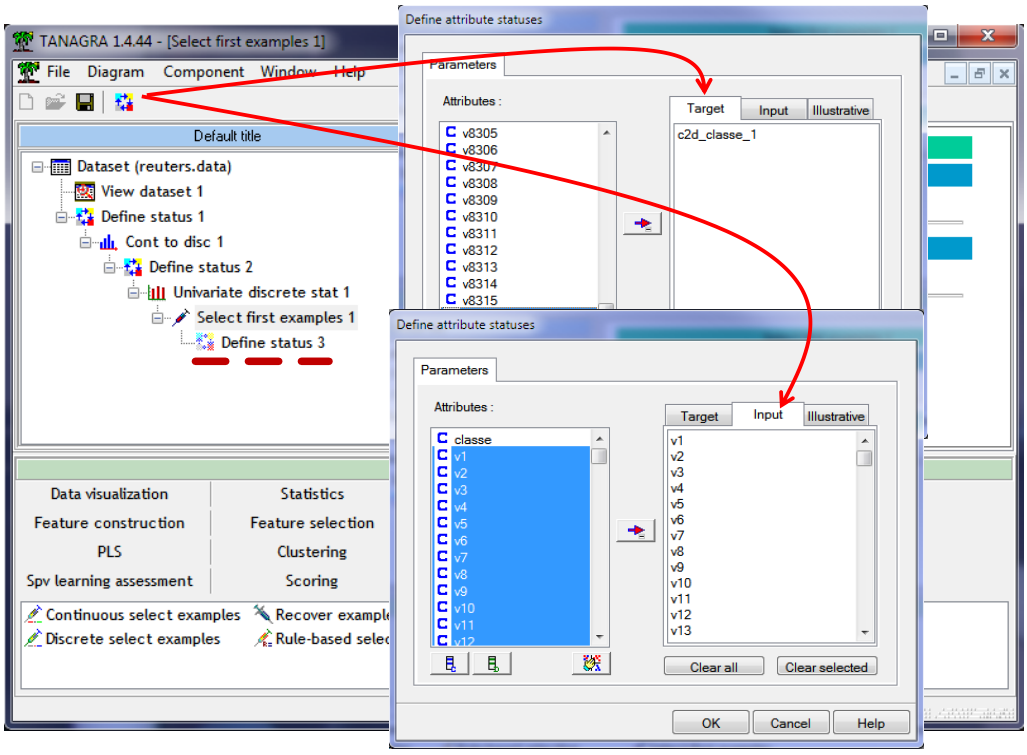
3.5 Subdivision apprentissage – test

Les échantillons d'apprentissage et de test ont été concaténés dans un seul fichier. Nous devons indiquer à Tanagra la partie dévolue à la modélisation. Nous insérons le composant SELECT FIRST EXAMPLES (onglet INSTANCE SELECTION). Nous le paramétrons de la manière à choisir les 7 770 premières observations pour l'apprentissage :



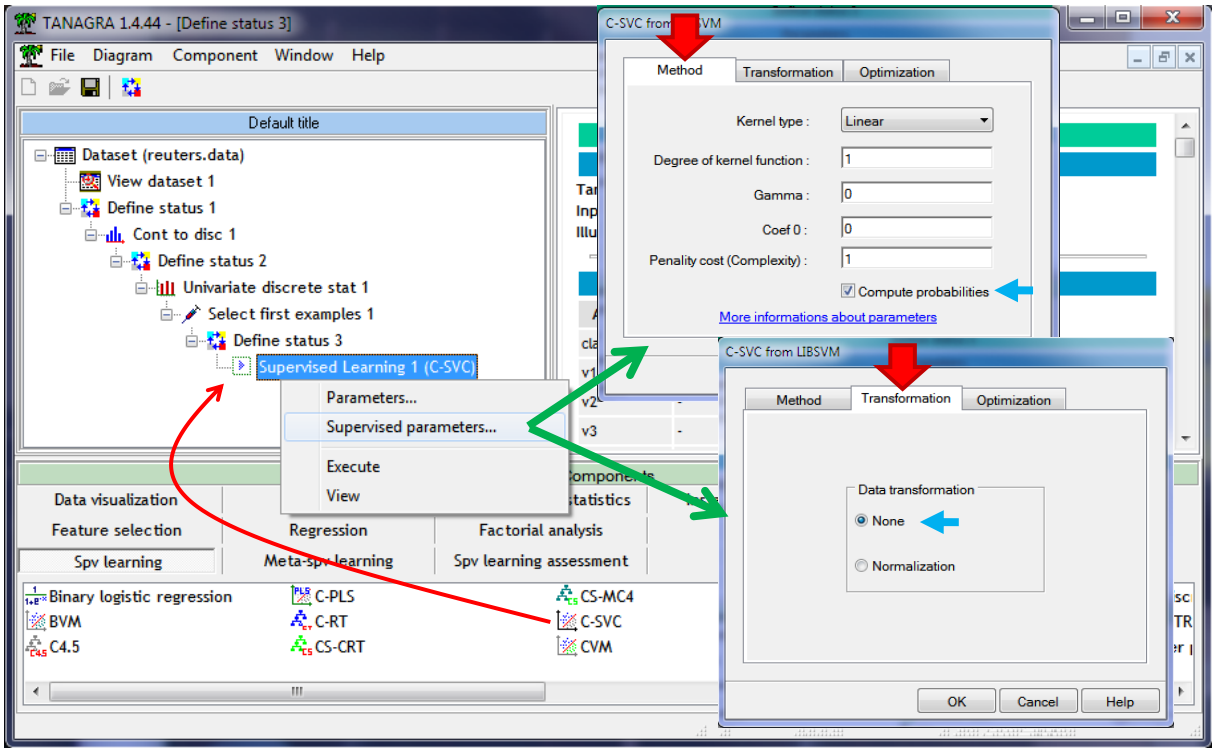
Nous cliquons sur VIEW pour acter la subdivision.

3.6 Apprentissage supervisé – C-SVC de Libsvm



Nous souhaitons modéliser la relation entre le thème des nouvelles et leur contenu. Pour préciser le rôle des variables, nous insérons le composant le composant DEFINE STATUS : nous plaçons en TARGET la variable cible recodée C2D_CLASSE_1, et en INPUT les descripteurs V1 à V8315. Attention, la colonne CLASSE ne doit plus être utilisée à ce stade de notre étude.

Nous plaçons ensuite le composant C-SVC (onglet SPV LEARNING). Il implémente un SVM (support vector machine). Nous spécifions les paramètres suivants :



Deux éléments sont à noter : nous demandons explicitement le calcul des probabilités d'affectation afin de pouvoir calculer la courbe ROC par la suite ; il n'est pas nécessaire de normaliser les données.

Nous cliquons sur VIEW pour lancer les traitements. Nous obtenons les résultats au bout de 43 secondes. Le taux d'erreur en resubstitution est de 0.59 %. Il ne faut pas trop s'y attarder : parce qu'il est calculé sur l'échantillon d'apprentissage ; parce que la dimensionnalité est très élevée, il y a donc un réel danger de sur apprentissage ; enfin parce que les classes sont très déséquilibrées.

Classifier performances

Error rate			0.0059			
Values prediction			Confusion matrix			
Value	Recall	1-Precision		_1_-1.00	_2_1.00	Sum
_1_-1.00	0.9965	0.0029	_1_-1.00	7207	25	7232
_2_1.00	0.9610	0.0461	_2_1.00	21	517	538
			Sum	7228	542	7770

Classifier characteristics

Data description

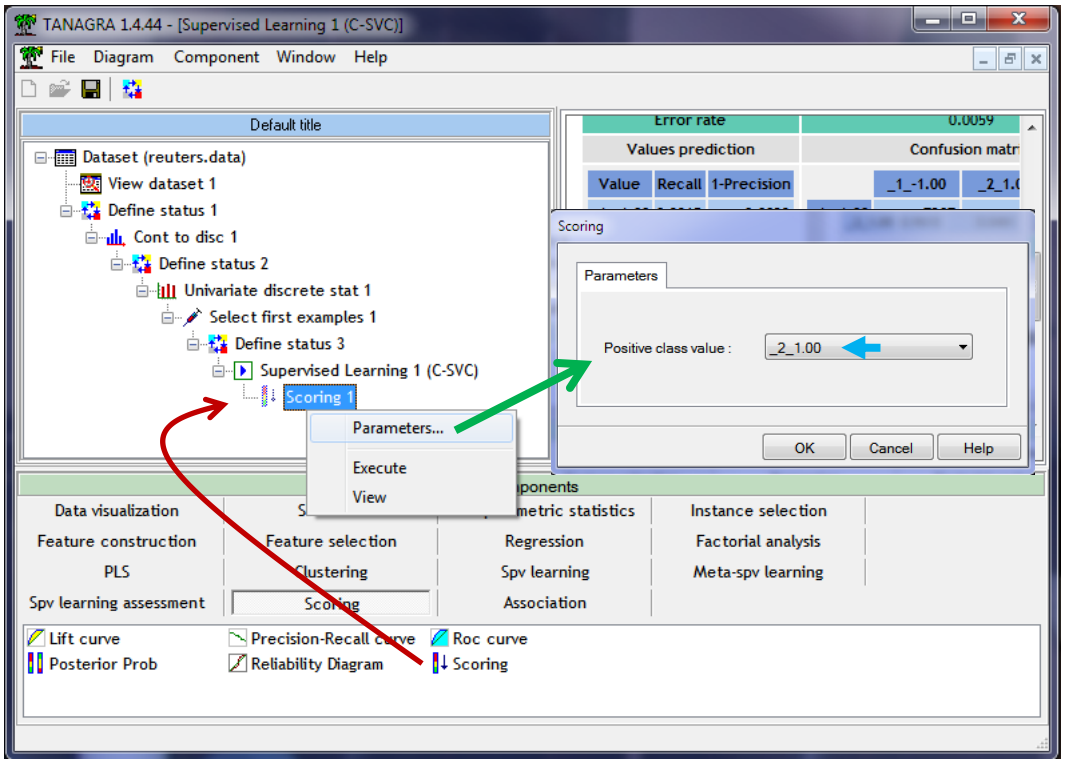
Target attribute	c2d_classe_1 (2 values)
# descriptors	8315

SVM characteristics

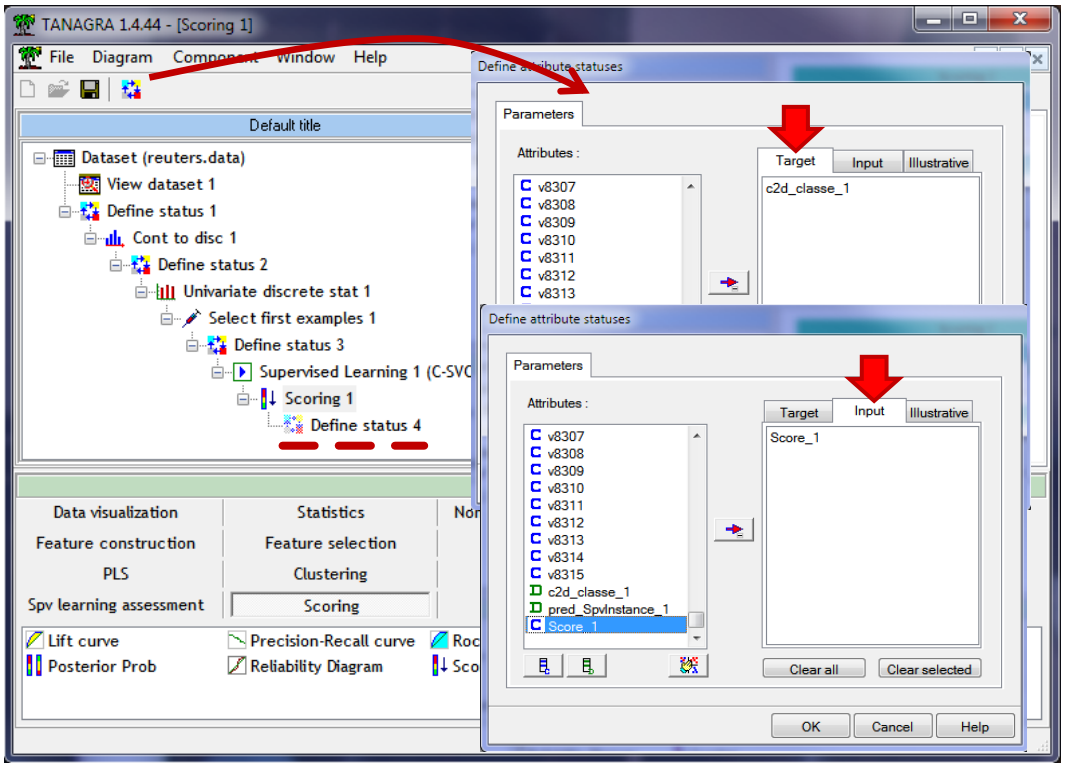
Characteristic	Value
# classes	2
# support vectors	789
# support vectors for each class	
# sv. for _1_-1.00	519
# sv. for _2_1.00	270

3.7 Scoring et construction de la courbe ROC sur l'échantillon test

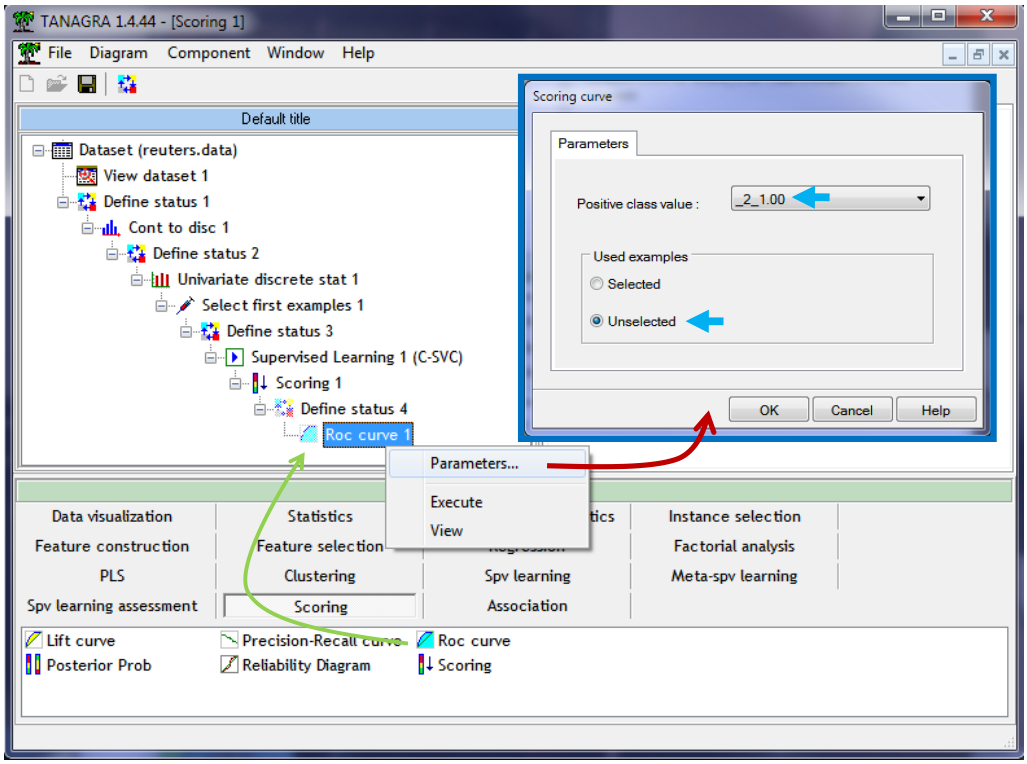
Pour construire la courbe ROC, nous devons calculer la probabilité d'appartenir à la modalité positive de la classe sur la totalité de la base. Nous insérons le composant SCORING (onglet SCORING). Nous le paramétrons pour calculer le score de la modalité « 1 » de la classe.



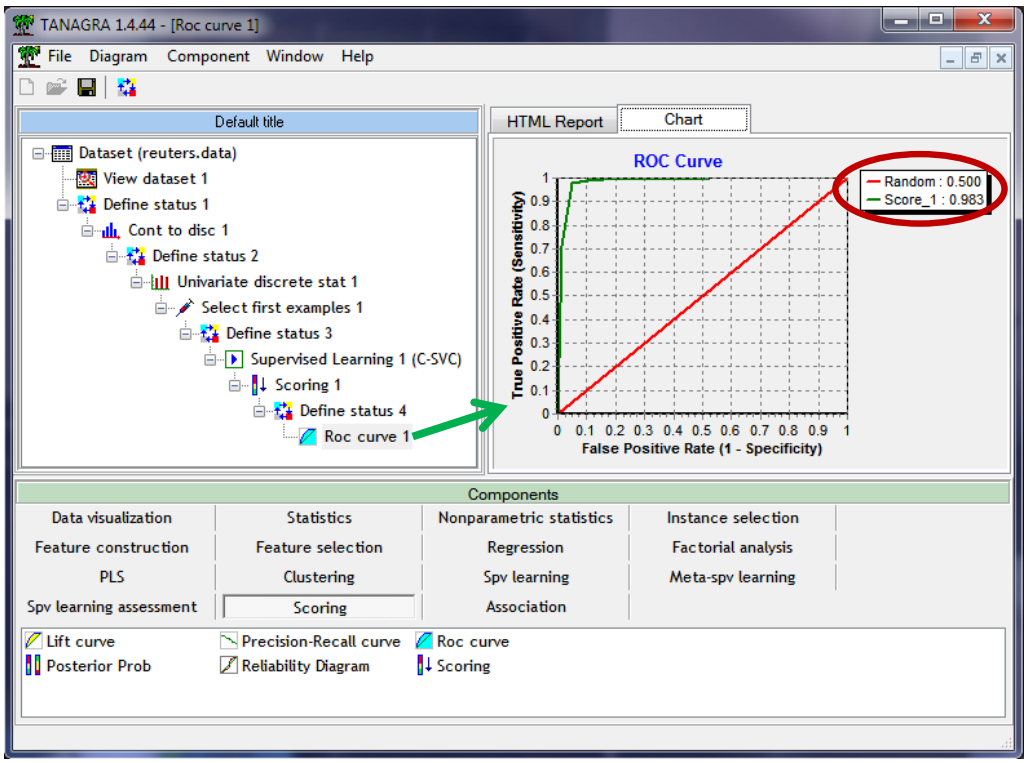
Ensuite, nous ajoutons le composant DEFINE STATUS. La cible est la classe d'appartenance C2D_CLASSE_1, en INPUT nous plaçons le score précédemment calculé SCORE_1.



Nous ajoutons le composant ROC CURVE (onglet SCORING). Nous le paramétrons de manière à ce que la courbe soit établie pour la modalité positive de la classe (CLASSE = 1), à partir de l'échantillon test n'ayant pas participé à l'élaboration du modèle (USED EXAMPLES = UNSELECTED).



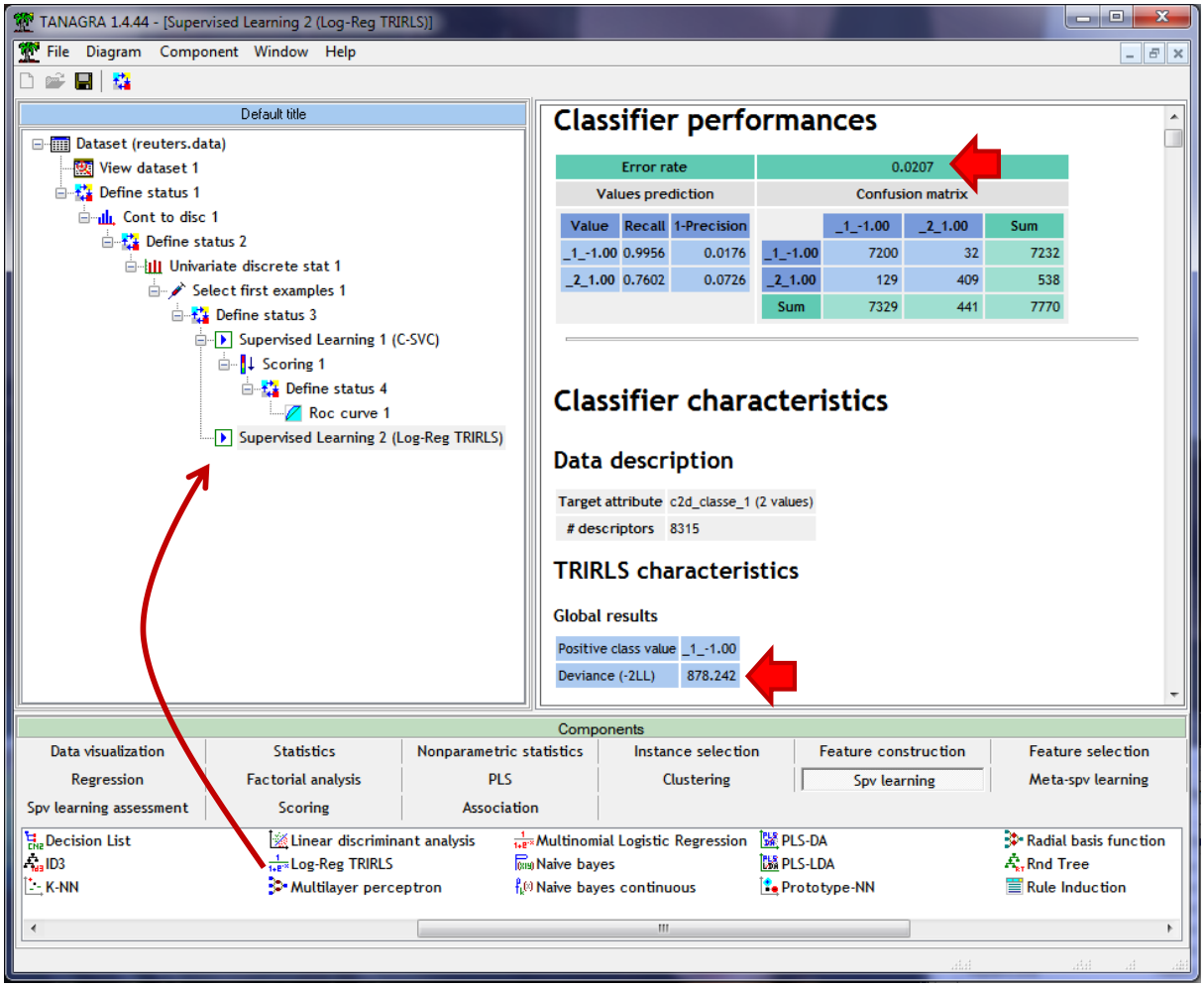
Nous constatons que notre modèle est d'excellente qualité. L'aire sous courbe AUC (Are Under Curve) est égale à 0.983. Pour deux individus pris au hasard, nous avons 98.3% de chances de placer un positif (nouvelle « money-fx ») devant un négatif (tout autre nouvelle).



3.8 Comparaison avec la régression logistique

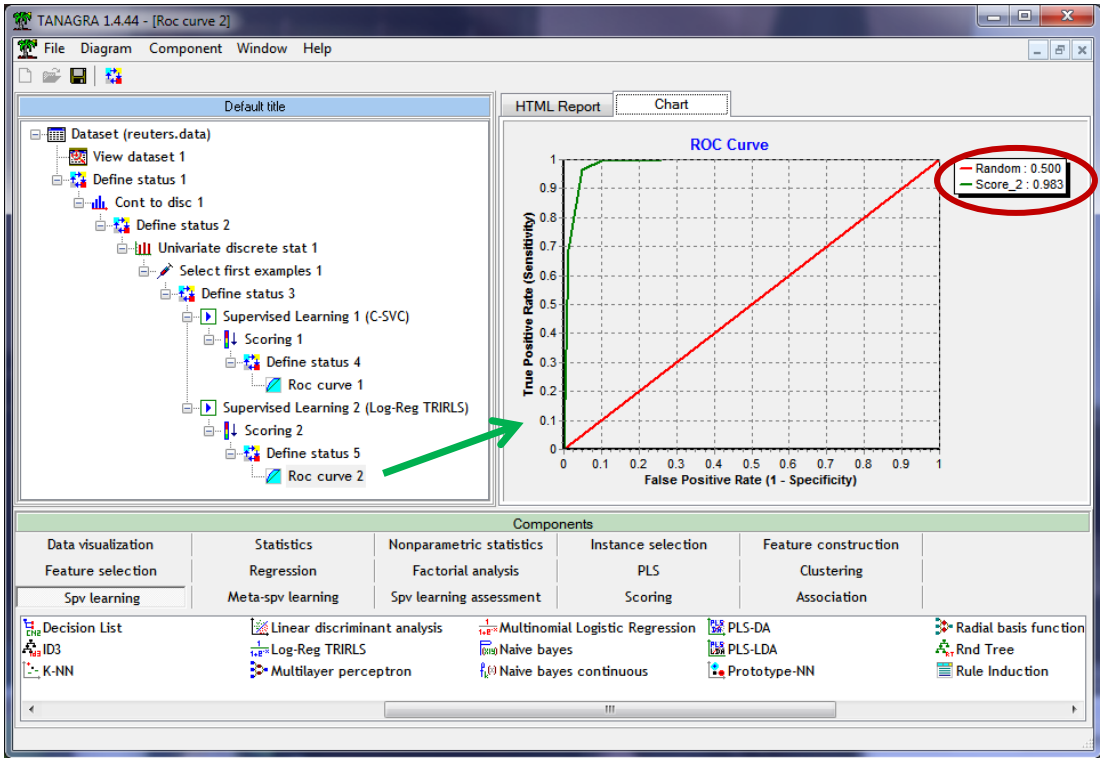
Dans un deuxième temps, nous souhaitons mener la même étude mais à l'aide de la régression logistique. Lancer ce type de méthode sur nos données (8315 variables explicatives) paraît excessivement optimiste… sauf pour la méthode TRIRLS (http://autonlab.org/autonweb/10538). Autant elle paraît peu décisive comparativement à une implémentation usuelle sur les bases classiques avec un grand nombre d'observations sur un nombre de variables limité[6] ((l'optimisation de la déviance est très approximative avec les paramètres par défaut), autant elle s'avère particulièrement efficace lorsque le rapport s'inverse. Nous avons intégré la bibliothèque sous forme de DLL dans Tanagra. Elle a été mise à jour récemment (Tanagra version 1.4.44).

Nous ajoutons le composant LOG-REG TRIRLS (onglet SPV LEARNING) dans le diagramme. Nous lançons les calculs. Les résultats arrivent au bout de 94 secondes sur notre machine. Nous obtenons les coefficients de la régression, la déviance du modèle est -2LL = 878.242.



Le taux d'erreur en resubstitution de 2.07%. Mais nous savons maintenant que cet indicateur, particulièrement dans notre contexte, est particulièrement suspect. Nous préférons nous en tenir à la courbe ROC calculée sur l'échantillon test.

[6] Quelques centaines, cf. http://tutoriels-data-mining.blogspot.fr/2012/01/regression-logistique-sur-les-grandes.html



Finalement, la régression logistique est aussi performante que le SVM linéaire avec un AUC = 0.983.

4 Importation avec d'autres logiciels

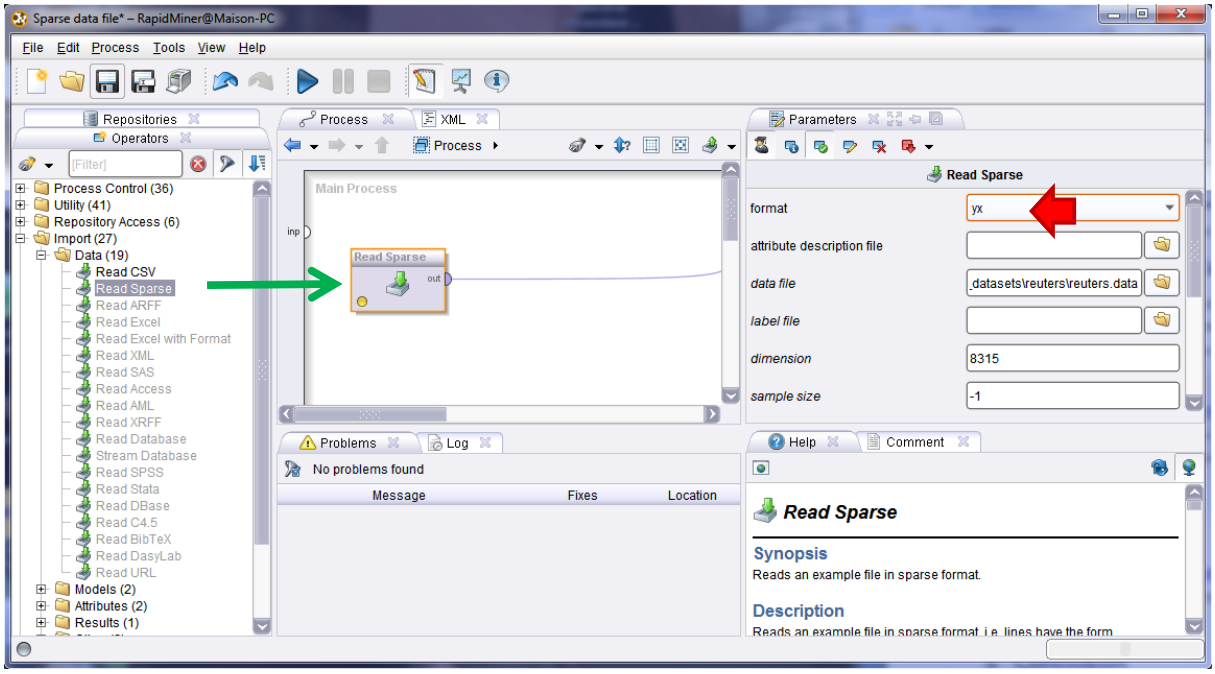
Bien évidemment, Tanagra n'est pas le seul à traiter les fichiers au format « sparse » de Svmlight et Libsvm. La fonctionnalité est présente dans plusieurs logiciels, nous décrivons très succinctement les outils disponibles sous (entres autres) RapidMiner et Weka.

4.1 RapidMiner

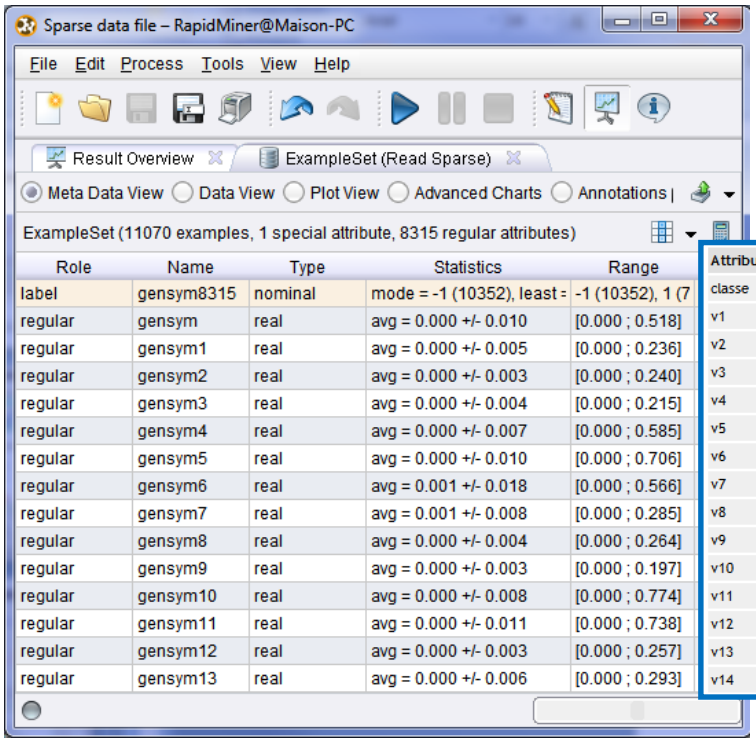
RapidMiner 5.2.006 (http://rapid-i.com/content/view/181/190/) sait lire les fichiers « sparse » tels que nous les décrivons dans ce tutoriel. En y regardant de plus près, on se rend compte que son champ d'application est plus large. A l'aide des différents paramètres, nous pouvons spécifier précisément l'organisation des données.

Le rôle du paramètre FORMAT est important. Il permet de spécifier la présence de l'étiquette dans le fichier (dans le cadre de la classification automatique, elle n'est pas nécessaire) et, le cas échéant, sa position. En ce qui nous concerne, nous avons choisi FORMAT = YX parce que l'étiquette est en première position dans la ligne.

Les autres paramètres sont largement décrits dans l'aide contextuelle.



RapidMimer fournit un résumé statistique des variables après le chargement (min, max, moyenne, écart type). Notons également qu'elles sont nommées automatiquement.



Role	Name	Type	Statistics	Range
label	gensym8315	nominal	mode = -1 (10352), least =	-1 (10352), 1 (7
regular	gensym	real	avg = 0.000 +/- 0.010	[0.000 ; 0.518]
regular	gensym1	real	avg = 0.000 +/- 0.005	[0.000 ; 0.236]
regular	gensym2	real	avg = 0.000 +/- 0.003	[0.000 ; 0.240]
regular	gensym3	real	avg = 0.000 +/- 0.004	[0.000 ; 0.215]
regular	gensym4	real	avg = 0.000 +/- 0.007	[0.000 ; 0.585]
regular	gensym5	real	avg = 0.000 +/- 0.010	[0.000 ; 0.706]
regular	gensym6	real	avg = 0.001 +/- 0.018	[0.000 ; 0.566]
regular	gensym7	real	avg = 0.001 +/- 0.008	[0.000 ; 0.285]
regular	gensym8	real	avg = 0.000 +/- 0.004	[0.000 ; 0.264]
regular	gensym9	real	avg = 0.000 +/- 0.003	[0.000 ; 0.197]
regular	gensym10	real	avg = 0.000 +/- 0.008	[0.000 ; 0.774]
regular	gensym11	real	avg = 0.000 +/- 0.011	[0.000 ; 0.738]
regular	gensym12	real	avg = 0.000 +/- 0.003	[0.000 ; 0.257]
regular	gensym13	real	avg = 0.000 +/- 0.006	[0.000 ; 0.293]

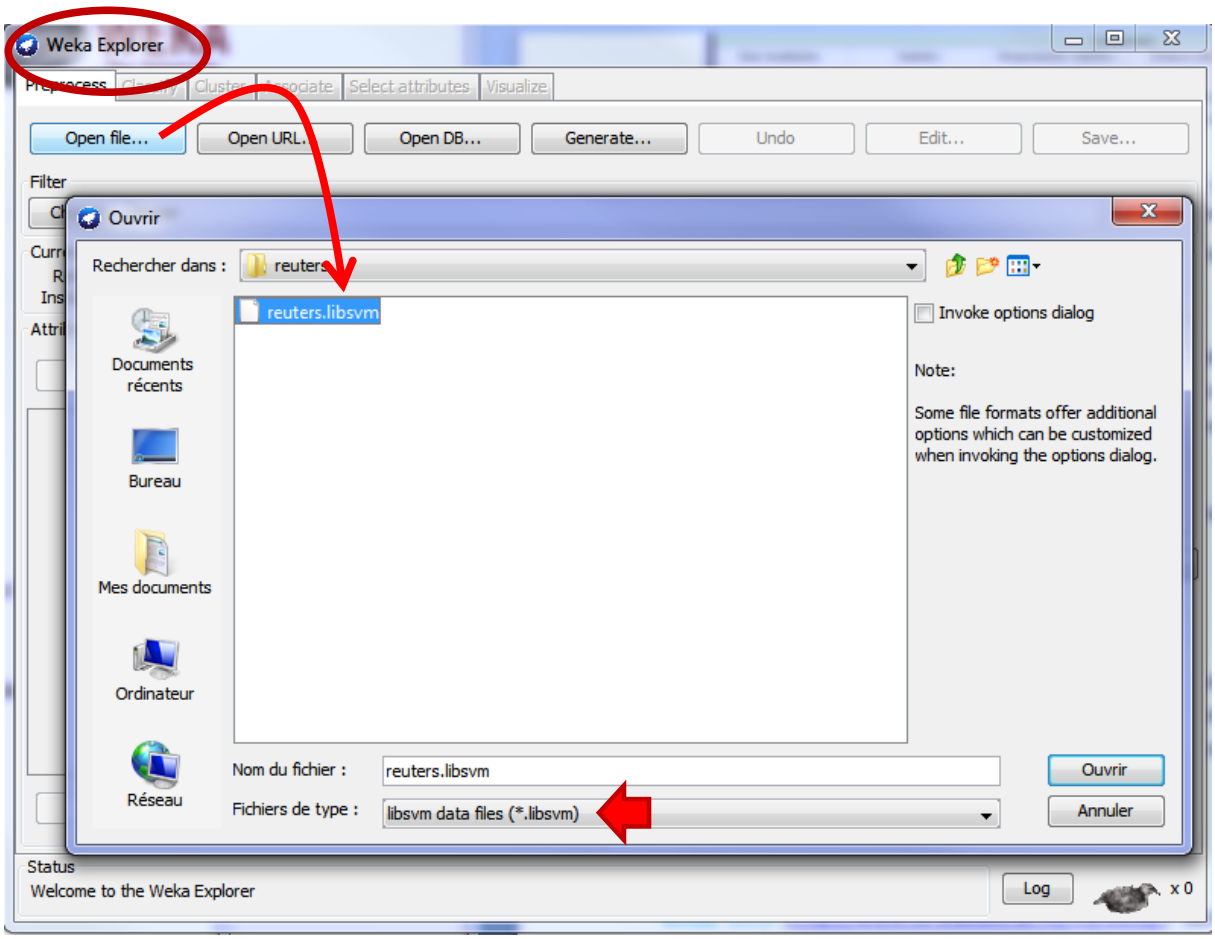
TANAGRA
Univariate Continuous Stat

Attribute	Min	Max	Average	Std-dev	Std-dev/avg
classe	-1	1	-0.8704	0.4923	-0.5655
v1	0	0.518098	0.0003	0.0100	33.0697
v2	0	0.236209	0.0001	0.0050	38.9161
v3	0	0.240174	0.0001	0.0034	61.8230
v4	0	0.214651	0.0001	0.0035	60.7377
v5	0	0.584598	0.0001	0.0074	55.7416
v6	0	0.705593	0.0002	0.0097	61.4414
v7	0	0.566478	0.0012	0.0176	14.6031
v8	0	0.285344	0.0005	0.0084	15.9816
v9	0	0.263867	0.0001	0.0044	42.7770
v10	0	0.197253	0.0000	0.0026	62.4435
v11	0	0.773622	0.0001	0.0082	72.6606
v12	0	0.737673	0.0002	0.0108	61.3924
v13	0	0.257434	0.0001	0.0033	63.6812
v14	0	0.29251	0.0002	0.0063	28.3777

A titre de comparaison, j'ai mis les indicateurs calculés avec le composant UNIVARIATE CONTINUOUS STAT (onglet STATISTICS) de Tanagra. Nous retrouvons bien les mêmes valeurs.

4.2 Weka

Weka 3.7.5 (http://www.cs.waikato.ac.nz/ml/weka/) sait lire les fichiers « sparse » au format Svmlight ou Libsvm. Il s'attend à ce que l'extension du nom de fichier soit « .libsvm ».



5 Conclusion

Le format « sparse » permet de réduire la taille des fichiers en adoptant une représentation particulière des valeurs. Comme tout algorithme de compression, il existe des contextes où il engendre un gain nul voire négatif. Lorsque nous sommes confrontés à une base classique, avec une très faible proportion de valeurs nulles, il n'est pas avantageux du tout. A contrario, il prend toute son importance lorsque nous traitons des données issues d'un prétraitement sur une collection de données non structurées. Nous avons pris l'exemple du text mining – plus précisément la catégorisation de texte - dans ce tutoriel.