1 Objectif

Comprendre le modèle d'indépendance conditionnelle (Classifieur Bayesien Naïf) lorsque les prédicteurs sont continus.

Le classifieur bayesien naïf est une méthode d'apprentissage supervisé qui repose sur une hypothèse simplificatrice forte : les descripteurs (Xj) sont deux à deux indépendants conditionnellement aux valeurs de la variable à prédire (Y). Pourtant, malgré cela, il se révèle robuste et efficace. Ses performances sont comparables aux autres techniques d'apprentissage. Diverses raisons sont avancées dans la littérature. Nous avions nous même proposé une explication basée sur le biais de représentation dans un précédent tutoriel[1]. Lorsque les prédicteurs sont discrets, on se rend compte aisément que le classifieur bayesien naïf est un séparateur linéaire. Il se pose donc en concurrent direct des autres techniques du même acabit, telles que l'analyse discriminante, la régression logistique, les SVM (Support Vector Machine) linéaires, etc.

Dans ce tutoriel, nous décrivons le modèle d'indépendance conditionnelle dans le cadre des variables prédictives quantitatives. La situation est un peu plus complexe. Nous verrons que, selon les hypothèses simplificatrices utilisées, il peut être considéré comme un séparateur linéaire ou quadratique. **Il est alors possible de produire un classifieur explicite, facilement utilisable pour le déploiement**. Les idées mises en avant dans ce tutoriel ont été implémentées dans **Tanagra 1.4.37** (et ultérieure). Cette représentation du modèle est originale. Je ne l'ai pas retrouvée dans les autres logiciels libres que j'ai l'habitude de suivre (pour l'instant...).

Ce document est organisé comme suit. Tout d'abord (section 2), nous détaillons les aspects théoriques de la méthode. Nous montrons qu'il est possible de parvenir à un modèle explicite que l'on peut exprimer sous la forme d'une combinaison linéaire des variables ou du carré des variables. Dans la section 3, nous décrivons la mise en œuvre de la méthode à l'aide du logiciel Tanagra. Nous confrontons les résultats avec ceux des autres séparateurs linéaires (régression logistique, SVM linéaire, analyse discriminante PLS, analyse discriminante de Fisher). Dans la section 4, nous comparons l'implémentation de la technique dans différents logiciels. Nous mettrons surtout l'accent sur la lecture des résultats. Enfin, section 5, nous montrons l'intérêt de l'approche sur les très grands fichiers. Nous traiterons la base « mutants » comprenant 16592 observations et 5408 variables prédictives avec une rapidité hors de portée des autres techniques.

2 Le classifieur bayesien naïf

Soient $\aleph = (X_1, \ldots, X_J)$ l'ensemble des descripteurs, tous continus ; Y est la variable à prédire discrète (l'attribut classe comportant K modalités). En apprentissage supervisé, pour un individu ω à classer, la règle bayesienne d'affectation optimale revient à maximiser la probabilité a posteriori (on parle de **règle décision MAP**) d'appartenance aux classes c.-à-d.

$$\hat{y}(\omega) = y_{k^*} \Leftrightarrow y_{k^*} = \arg\max_k P\left[Y = y_k / \aleph(\omega)\right]$$

La décision repose donc sur une estimation viable de la probabilité conditionnelle P(Y/X). Cette dernière peut s'écrire d'une manière différente

[1] http://tutoriels-data-mining.blogspot.com/2010/03/le-classifieur-bayesien-naif-revisite.html

$$P[Y = y_k / \aleph(\omega)] = \frac{P(Y = y_k) \times P[\aleph(\omega)/Y = y_k]}{P[\aleph(\omega)]}$$

Comme l'objectif est de détecter le maximum de cette quantité selon y_k, et que le dénominateur n'en dépend pas, nous pouvons ré écrire la règle d'affectation ci-dessus

$$\hat{y}(\omega) = y_{k*} \Leftrightarrow y_{k*} = \arg\max_k P(Y = y_k) \times P[\aleph(\omega)/Y = y_k]$$

2.1 Hypothèse – 1 : l'indépendance conditionnelle des descripteurs

La quantité $P(Y = y_k)$ est facile à estimer à partir d'un échantillon d'observations. Il suffit de calculer les proportions de chaque modalité de la variable cible. Parfois, on utilise souvent la « m probability estimate » pour « lisser » les estimations sur les petits effectifs. Par exemple lorsque nous traitons un problème avec des classes très déséquilibrées.

Si n_k est le nombre d'individu de la modalité y_k dans un échantillon de n observations, nous utilisons

$$\hat{P}(Y = y_k) = p_k = \frac{n_k + \lambda}{n + \lambda \times K}$$

Lorsque λ = 0, nous avons la fréquence relative usuelle. Lorsque nous fixons $\lambda = 1$, nous obtenons l'estimateur laplacien des probabilités.

La véritable difficulté réside finalement dans la production d'une estimation viable de la quantité $P[\aleph(\omega)/Y = y_k]$. Nous sommes souvent obligés d'introduire des hypothèses pour rendre le calcul réalisable. L'analyse discriminante paramétrique stipule que la distribution est gaussienne[2] ; la régression logistique binaire ($Y \in \{+,-\}$) part sur l'idée que le rapport $\frac{P[\aleph(\omega)/Y = +]}{P[\aleph(\omega)/Y = -]}$ appartient à une famille de lois particulières[3].

Dans le cadre du classifieur bayesien naïf, on considère que les descripteurs sont deux à deux indépendants conditionnellement aux valeurs de la variable cible. Par conséquent,

$$P[\aleph(\omega)/Y = y_k] = \prod_{j=1}^{J} P[X_j(\omega)/Y = y_k]$$

Le nombre de paramètres à calculer est réduit de manière drastique. Il nous reste à produire une estimation viable de la quantité $P[X_j / Y = y_k]$ pour chaque variable Xj.

2.2 Hypothèse – 2 : distribution conditionnelle gaussienne

2.2.1 Fonctions de classement

La proposition la plus courante consiste à faire l'hypothèse d'une distribution gaussienne de la probabilité conditionnelle. Pour une variable Xj quelconque, elle s'écrira

[2] http://fr.wikipedia.org/wiki/Analyse_discriminante_linéaire

[3] http://fr.wikipedia.org/wiki/Régression_logistique

$$P\left[X_j / Y = y_k\right] = f_k(X_j) = \frac{1}{\sigma_{k,j}\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x_j - \mu_{k,j}}{\sigma_{k,j}}\right)^2}$$

Où $\mu_{k,j}$ est la moyenne de la variable Xj pour le groupe Y = yk ; $\sigma_{k,j}$ l'écart type. On considère ici que les écarts-type conditionnels sont différents d'un groupe à l'autre. On parle d'hétéroscédasticité.

Ces paramètres sont estimés de la manière suivante :

$$\hat{\mu}_{k,j} = \frac{1}{n_k} \sum_{\omega : Y(\omega) = y_k} x_j(\omega)$$

$$\hat{\sigma}_{k,j} = \frac{1}{n_k - 1} \sum_{\omega : Y(\omega) = y_k} \left[x_j(\omega) - \hat{\mu}_{k,j}\right]^2$$

L'hypothèse est sans aucun doute contraignante. Mais plutôt que d'essayer d'élargir la portée de la méthode en multipliant les hypothèses de distribution (ex. log-normal, gamma, poisson dans le logiciel STATISTICA -- http://www.statsoft.com/textbook/naive-bayes-classifier/), il me paraît plus judicieux de cerner la robustesse de l'hypothèse de normalité. En effet, ce serait là ouvrir la porte à des options à n'en plus finir. A l'extrême, nous en serions à proposer une distribution différente pour chaque variable prédictive.

L'hypothèse de normalité s'avère viable tant que les deux conditions suivantes sont réunies **: les distributions conditionnelles sont plus ou moins symétriques et, surtout, elles sont unimodales**. Lorsqu'elles sont multimodales, avec de plus un chevauchement entre les modalités de la variable cible, toute approximation unimodale, a fortiori gaussienne, n'est absolument pas adaptée et produira un classifieur de mauvaise qualité.

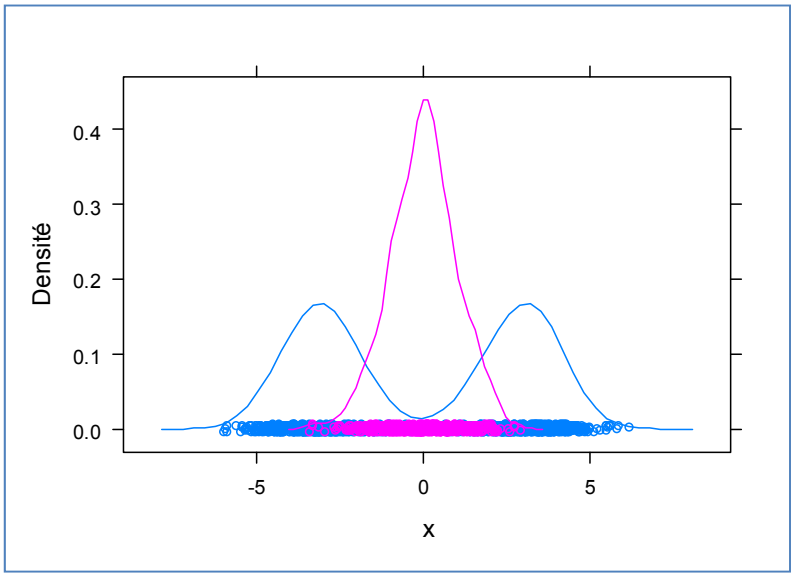


Figure 1 - Distribution bimodale des positifs (bleu), chevauchement avec les négatifs (magenta)

Dans notre exemple (Figure 1)[4], les moyennes conditionnelles sont confondues, laissant à penser qu'il n'y a pas de séparation possible, alors que l'on se rend bien compte que les positifs (bleu) sont parfaitement discernables des négatifs (magenta).

Deux solutions s'offrent à nous lorsque l'hypothèse de normalité n'est pas appropriée. La première consiste à utiliser une estimation non paramétrique. Intéressante en théorie, elle permet de dépasser l'inconvénient ci-dessus, elle s'avère très lourde en pratique. En effet, elle nécessite le calcul la valeur de la densité au voisinage du point à classer lors du déploiement. Il est impossible de produire un modèle explicite. La seconde, la plus simple, consiste à discrétiser préalablement les variables prédictives à l'aide de techniques contextuelles (ex. MDLPC de Fayyad et Irani, 1993 -- http://tutoriels-data-mining.blogspot.com/2010/02/discretisation-comparaison-de-logiciels.html ; http://tutoriels-data-mining.blogspot.com/2008/03/discrtisation-contextuelle-la-mthode.html). De nombreuses références montrent que cette méthode est très efficace. **C'est la solution que je préconise**.

Ceci étant précisé, revenons à l'hypothèse de normalité. En passant aux logarithmes, nous obtenons les fonctions de classement :

$$d(y_k,\aleph) = \ln p_k + \sum_j \left\{ -\left[\frac{1}{2}\ln(2\pi) + \ln(\sigma_{k,j})\right] - \frac{1}{2}\left(\frac{x_j - \mu_{k,j}}{\sigma_{k,j}}\right)^2 \right\}$$

La règle d'affectation bayesienne reste la même, à savoir

$$\hat{y}(\omega) = y_{k^*} \Leftrightarrow y_{k^*} = \arg\max_k d\left[y_k, \aleph(\omega)\right]$$

Puisqu'il s'agit de trouver le maximum par rapport à y_k, tout ce qui ne dépend pas k peut être retiré de l'expression. Nous dirons que la fonction de classement est « proportionnelle à »

$$d(y_k,\aleph) \propto \ln p_k + \sum_j \left\{ -\ln(\sigma_{k,j}) - \frac{1}{2}\left(\frac{x_j - \mu_{k,j}}{\sigma_{k,j}}\right)^2 \right\}$$

$$\propto \ln p_k + \sum_j \left\{ -\ln(\sigma_{k,j}) - \frac{1}{2\times\sigma_{k,j}^2}\left(x_j^2 - 2\times x_j \times \mu_{k,j} + \mu_{k,j}^2\right) \right\}$$

Nous obtenons finalement :

$$d(y_k,\aleph) \propto \ln p_k + \sum_j \left\{ -\frac{1}{2\times\sigma_{k,j}^2}x_j^2 + \frac{\mu_{k,j}}{\sigma_{k,j}^2}x_j - \left(\frac{\mu_{k,j}^2}{2\times\sigma_{k,j}^2} + \ln(\sigma_{k,j})\right) \right\}$$

Équation 1 - Fonction de classement - Hypothèse d'hétéroscédasticité

[4] Voici le code R qui a permis de produire ce graphique.

```
> x <- c(rnorm(1000,-3,1),rnorm(1000,0,1),rnorm(1000,+3,1))
> y <- c(rep(1,1000),rep(2,1000),rep(1,1000))
> library(lattice)
> densityplot(~ x, groups=factor(y))
```

Nous obtenons une équation quadratique, mais sans les termes d'interactions entre les Xj conformément à l'hypothèse d'indépendance conditionnelle. De fait, Tanagra nous fournira les coefficients de cette expression basée sur les variables et leurs carrés pour le classement de nouveaux individus. Il ne sera plus nécessaire de conserver ou afficher les moyennes et écarts-type conditionnels calculés sur les données d'apprentissage.

2.2.2 Un exemple numérique : le fichier IRIS (1)

Prenons l'exemple du fichier IRIS (http://archive.ics.uci.edu/ml/datasets/Iris). Il s'agit de prédire le type de fleurs (3 valeurs possibles : setosa - rouge, versicolor - vert et virginica - bleu) à partir des deux derniers prédicteurs : la longueur (X1) et la largeur (X2) des pétales (Figure 2)[5].

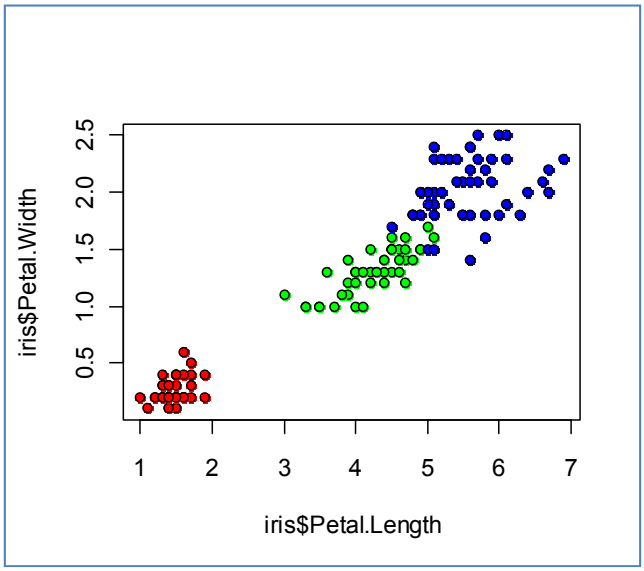


Figure 2 - IRIS de Fisher

a. Construction du modèle prédictif

Nous disposons de 150 observations. Les classes sont équilibrées (λ = 0),

$$p_k = \frac{50}{150} = 0.333,\ \forall k$$

Nous devons maintenant calculer les moyennes et écart-types conditionnels. Nous utilisons les tableaux croisés dynamiques d'Excel. Pour la variable petal.length

	Données	
type	Moyenne de pet_length	Écartype de pet_length
Iris-setosa	1.4640	0.1735
Iris-versicolor	4.2600	0.4699
Iris-virginica	5.5520	0.5519

Pour petal.width

[5] Voici le code R permettant de générer ce graphique.

```
> data(iris)
> plot(iris$Petal.Length,iris$Petal.Width,bg=c("red","green","blue")[unclass(iris$Species)],pch=21)
```

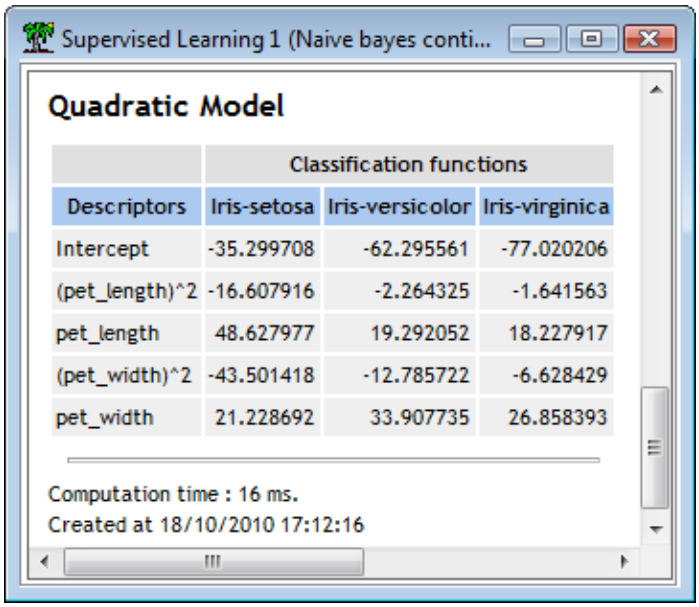
	Données	
type	Moyenne de pet_width	Écartype de pet_width
Iris-setosa	0.2440	0.1072
Iris-versicolor	1.3260	0.1978
Iris-virginica	2.0260	0.2747

Formons à présent les fonctions de classement. Voici le détail des calculs pour la classe « setosa »

$$d(setosa,\aleph) = \ln p_k + \sum_j \left\{ -\frac{1}{2\times\sigma_{k,j}^2} x_j^2 + \frac{\mu_{k,j}}{\sigma_{k,j}^2} x_j - \left(\frac{\mu_{k,j}^2}{2\times\sigma_{k,j}^2} + \ln(\sigma_{k,j}) \right) \right\}$$
$$= -1.099 + \left[\left(-16.608x_1^2 + 48.628x_1 - 33.844\right) + \left(-43.501x_2^2 + 21.229x_2 - 0.357\right) \right]$$
$$= -16.608x_1^2 + 48.628x_1 - 43.501x_2^2 + 21.229x_2 - 35.300$$

Nous faisons de même pour les autres classes, nous obtenons :

$$d(versicolor,\aleph) = -2.264x_1^2 + 19.292x_1 - 12.786x_2^2 + 33.908x_2 - 62.296$$
$$d(virginica,\aleph) = -1.642x_1^2 + 18.228x_1 - 6.628x_2^2 + 26.858x_2 - 77.020$$



b. *Sorties du logiciel Tanagra*

Les sorties du logiciel Tanagra sont en adéquation avec ces calculs. Les coefficients de chaque fonction de classement sont affichés dans une colonne. Les paramètres pour les variables au carré (Xj²) sont intercalés avec ceux pour Xj. Les constantes sont placées dans la première ligne du tableau.

Nous expliciterons le mode opératoire du logiciel plus loin (importation des données, paramétrage de la méthode).

c. *Classement des observations*

Pour le classement, nous appliquons ces fonctions. Nous associons aux individus la modalité de la variable cible correspondant au maximum. Prenons quelques exemples, observons la cohérence entre l'étiquette attribuée et la position du point dans l'espace de représentation (Figure 2).

N°	Coordonnées (X1, X2)	d(setosa)	d(versicolor)	d(virginica)	Prédiction
1	(1.5, 0.5)	**0.013**	-24.695	-41.600	setosa
2	(5.0, 1.5)	-273.393	**-0.350**	-1.546	versicolor
3	(5.0, 2.0)	-338.906	-5.771	**0.283**	virginica

Mis à part le point n°2, les décisions sont assez tranchées. On le comprend aisément à la lumière du graphique nuage de points (Figure 2). Les difficultés surviennent dans la zone située entre les versicolor (en vert) et les virginica (en bleu). Pour le point de coordonnées (X1 = 5, X2 = 1.6), nous

obtiendrions [d(setosa) = -284.755 ; d(versicolor) = -0.923 ; d(virginica) = -0.915]. La décision « iris = virginica »ne tient qu'à un fil.

2.2.3 Cas particulier du modèle binaire

Dans le cas binaire, Y = {+, -}, nous n'avons que deux fonctions de classement, il est facile d'en dériver une fonction de décision unique définie de la manière suivante

$$\begin{aligned} d(\aleph) &= d(+,\aleph) - d(-,\aleph) \\ &= \delta + \sum_j \left\{\alpha_j x_j^2 + \beta_j x_j + \gamma_j\right\} \end{aligned}$$

La règle d'affectation devient

$$\text{Si } d\left[\aleph(\omega)\right] > 0 \text{ Alors } \hat{y}(\omega) = + \text{ Sinon } \hat{y}(\omega) = -$$

2.2.4 Evaluation des prédicteurs

Pour évaluer la pertinence d'une variable prédictive dans le modèle, il est tentant de se baser sur une comparaison des distributions conditionnelles. Comme elles sont supposées gaussiennes, un test de comparaison des K moyennes conditionnelles, bref une analyse de variance, devrait faire l'affaire.

Il y a un premier écueil. Les variances intra-classes ne sont pas supposées égales. Il faudrait donc utiliser des méthodes qui tiennent compte de leur éventuelle hétérogénéité, l'analyse de variance de Welch par exemple[6]. Mais même si cela est faisable, il y a un second écueil autrement plus ennuyeux. A la lumière de la fonction de classement (Équation 1), on se rend compte qu'il y a en réalité une double condition à vérifier pour voir si une variable est inopérante :

$$H_0 : \begin{cases} \dfrac{1}{\sigma_{k,j}^2} = \text{constante}, \forall k \\ \dfrac{\mu_{k,j}}{\sigma_{k,j}^2} = \text{constante}, \ \forall k \end{cases}$$

Si les deux conditions sont effectivement remplies, alors on peut considérer que la variable Xj ne joue aucun rôle dans la discrimination des classes de la variable cible. En effet, elle contribue exactement de la même manière dans toutes les fonctions de classement. Nous pouvons la supprimer du modèle. A l'heure actuelle, j'avoue ne pas connaître de test statistique qui permet de répondre à cette question. En l'état, mieux vaut s'abstenir.

2.3 Hypothèse – 3 : Homoscédasticité – Classifieur linéaire

2.3.1 Fonction de classement

L'affaire devient encore plus intéressante encore lorsque nous introduisons une dernière hypothèse : l'homoscédasticité. Elle stipule que les écarts-type conditionnels sont les mêmes pour chaque variable. Concrètement, pour toute variable Xj, nous avons

$$\sigma_{k,j} = \sigma_j \ , \ \forall k$$

[6] Ricco Rakotomalala, « Comparaisons de populations – Test paramétriques », version 1.2, 2010 ; section 1.3.6, page 21 -- http://eric.univ-lyon2.fr/~ricco/cours/cours/Comp_Pop_Tests_Parametriques.pdf

Nous utilisons la formule de la variance intra-classes pour estimer l'écart-type de Xj,

$$\hat{\sigma}_j^2 = \frac{1}{n-K}\sum_{k=1}^{K}(n_k - 1)\times\hat{\sigma}_{k,j}^2$$

Introduite dans l'expression des fonctions de classement, nous obtenons :

$$d(y_k,\aleph) \propto \ln p_k + \sum_j \left\{ -\frac{1}{2\times\sigma_j^2}x_j^2 + \frac{\mu_{k,j}}{\sigma_j^2}x_j - \left(\frac{\mu_{k,j}^2}{2\times\sigma_j^2} + \ln(\sigma_j) \right) \right\}$$

On se rend compte rapidement que beaucoup d'éléments ne dépendent plus de la modalité k. Nous pouvons les retirer puisque l'objectif est toujours de détecter le maximum. La fonction de classement est simplifiée comme suit :

$$d(y_k,\aleph) \propto \ln p_k + \sum_j \left\{ \frac{\mu_{k,j}}{\sigma_j^2}x_j - \frac{\mu_{k,j}^2}{2\times\sigma_j^2} \right\}$$

Équation 2- Fonction de classement - Hypothèse d'homoscédasticité

Les termes au carré ont disparu. Nous avons une combinaison linéaire des variables prédictives. Autrement dit, nous disposons d'un classifieur linéaire. Sauf circonstances exceptionnelles, il propose des performances en classement similaires aux techniques reposant sur un biais de représentation équivalent (ex. analyse discriminante, régression logistique, etc.).

2.3.2 Un exemple numérique : le fichier IRIS (2)

a. Construction du modèle prédictif

Revenons à notre exemple du fichier IRIS ci-dessus (section 2.3). Il nous faut tout d'abord calculer les écarts-type intra-classes. Nous avons, respectivement pour pet.length (X1) et pet.width (X2) :

$$\hat{\sigma}_1 = \sqrt{\frac{1}{150-3}\left[49\times 0.1735^2 + 49\times 0.4699^2 + 49\times 0.5519^2\right]} = 0.430$$

$$\hat{\sigma}_2 = \sqrt{\frac{1}{150-3}\left[49\times 0.1072^2 + 49\times 0.1978^2 + 49\times 0.2747^2\right]} = 0.205$$

Les trois fonctions de classement s'écrivent :

$$\begin{aligned} d(setosa,\aleph) &= \ln p_k + \sum_j \left\{ \frac{\mu_{k,j}}{\sigma_j^2}x_j - \frac{\mu_{k,j}^2}{2\times\sigma_j^2} \right\} \\ &= -1.099 + (7.906x_1 - 5.787) + (5.808x_2 - 0.709) \\ &= 7.906x_1 + 5.808x_2 - 7.595 \\ d(versicolor,\aleph) &= 23.006x_1 + 31.563x_2 - 71.028 \\ d(virginica,\aleph) &= 29.983x_1 + 48.226x_2 - 133.185 \end{aligned}$$

b. Sorties du logiciel Tanagra

Tanagra fournit ces coefficients. Il y adjoint également des informations sur la pertinence des variables (statistique F et p-value). Nous détaillerons le calcul utilisé plus bas.

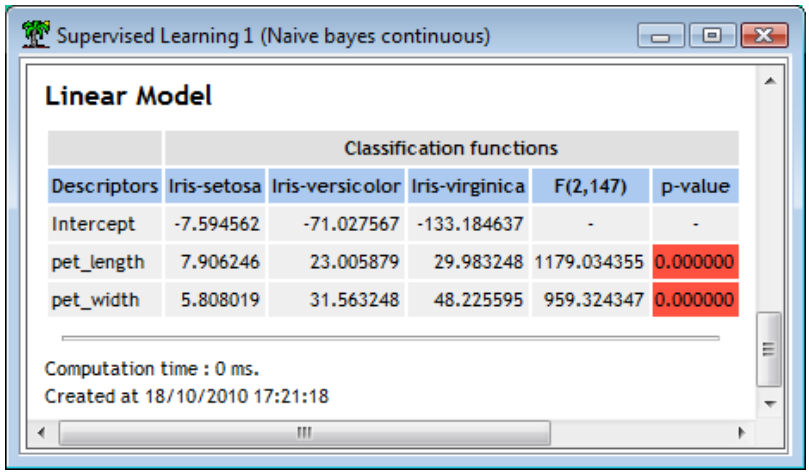


Figure 3 - Modèle linéaire - Fichier IRIS

c. *Classement*

Nous classons les mêmes individus que précédemment. Nous obtenons les affectations suivantes.

N°	Coordonnées (X1, X2)	d(setosa)	d(versicolor)	d(virginica)	Prédiction
1	(1.5, 0.5)	**7.169**	-20.737	-64.097	setosa
2	(5.0, 1.5)	40.649	**91.347**	89.070	versicolor
3	(5.0, 2.0)	4.553	107.128	**113.183**	virginica

Les conclusions sont les mêmes qu'avec le modèle quadratique. La situation est un peu différente vers la zone frontière, pour le point de coordonnées (X1 = 5, X2 = 1.6), la décision bascule vers « versicolor » cette fois-ci [d(setosa) = 41.229 ; d(versicolor) = 94.503 ; d(virginica) = 93.893].

2.3.3 Cas particulier du problème binaire

Dans le cas de la discrimination binaire, à partir d'une différence termes à termes des coefficients de deux fonctions de classement, nous pouvons produire une fonction de décision unique, à l'instar de l'analyse discriminante ou de la régression logistique

$$d(\aleph) = a_0 + a_1 x_1 + a_2 x_2 + \cdots$$

Avec toujours la règle d'affectation,

$$\text{Si } d[\aleph(\omega)] > 0 \text{ Alors } \hat{y}(\omega) = + \text{ Sinon } \hat{y}(\omega) = -$$

2.3.4 Evaluation des prédicteurs sous hypothèse d'homoscédasticité

Contrairement au modèle sous hypothèse d'hétéroscédasticité, il est possible de produire un test simple pour évaluer la pertinence d'une variable. En effet, à partir de la fonction de classement (Équation 1), nous constatons qu'une variable Xj ne contribue pas à la discrimination si

$$H_0 : \mu_{k,j} = \text{constante}, \forall k$$

Ce n'est, ni plus ni moins, que l'hypothèse nulle de l'analyse de variance. Un test on ne peut plus classique, d'autant plus que l'on a stipulé l'égalité des écarts-type conditionnels (homoscédasticité). La statistique du test pour la variable Xj s'écrit

$$F_j = \frac{\frac{\sum_k n_k \left(\hat{\mu}_{k,j} - \hat{\mu}_j\right)^2}{K-1}}{\frac{\sum_k (n_k - 1)\hat{\sigma}^2_{k,j}}{n-K}}$$

Sous l'hypothèse nulle, elle suit une loi de Fisher à (K-1, n-K) degrés de libertés. La région critique du test au risque α s'écrit $F_j \geq F_{1-\alpha}(K-1, n-K)$; $F_{1-\alpha}$ est le quantile d'ordre (1-α) de la loi de Fisher.

Pour le fichier IRIS, nous constatons que les deux variables sont très pertinentes. On s'en serait douté à la lumière des nuages de points conditionnels dans le plan (X1, X2) (Figure 2). La statistique F prend la valeur 1179.03 (resp. 959.32) pour la variable pet.length (resp. pet width) (Figure 3). Nous retrouvons les mêmes valeurs en effectuant une analyse de variance (Figure 4).

Remarque : Attention, ce test évalue la pertinence des variables de manière individuelle, sans tenir compte des interactions. En cela, il est conforme à l'hypothèse d'indépendance conditionnelle des prédicteurs. Mais cela veut dire aussi qu'il ne tient pas compte du tout de la redondance. Si une variable est dupliquée 20 fois, elle sera 20 fois significative.

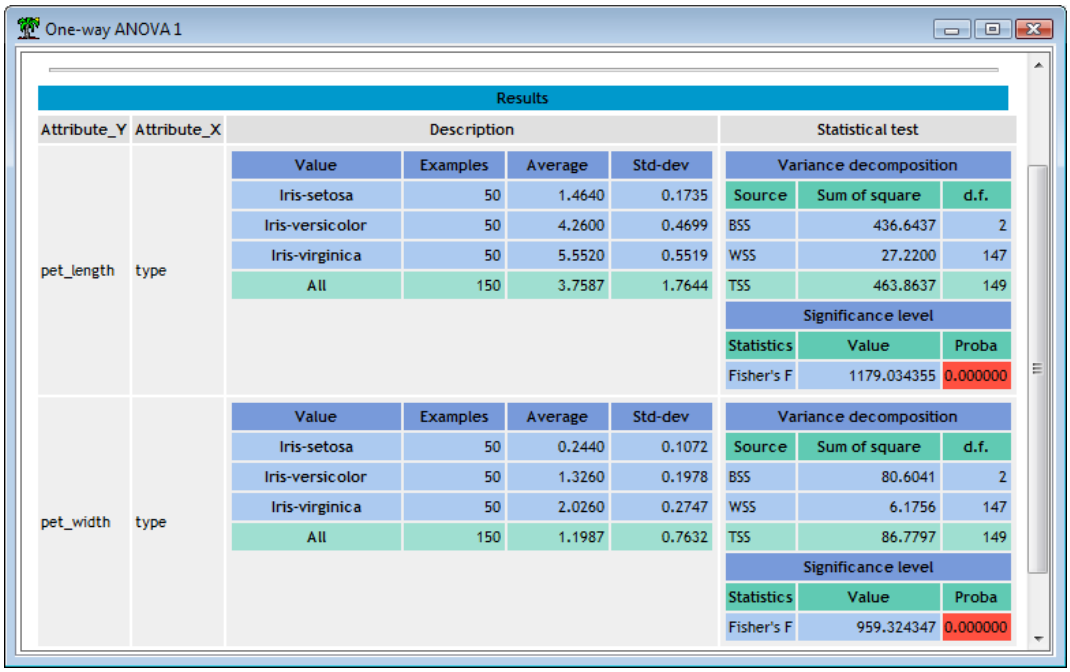


Figure 4 - Analyse de variance - Fichier Iris

3 Classifieur bayesien naïf avec Tanagra

3.1 Données

Pour illustrer la méthode, nous utilisons la base « breast cancer » (breast.txt), très connue dans notre communauté (http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29). L'objectif est de diagnostiquer le caractère malin ou bénin d'une tumeur à partir des caractéristiques des cellules prélevées. Ce fichier est d'autant plus intéressant que l'hypothèse de normalité conditionnelle des distributions est sérieusement mise à mal. Les variables sont certes numériques, mais elles correspondent à des indicatrices d'intervalles après discrétisation des variables originelles

auxquelles nous n'avons pas accès. En traçant les fonctions de densité conditionnelles, nous notons que l'hypothèse d'homoscédasticité n'est pas très crédible non plus. Néanmoins, et c'est ce qui nous rend plutôt optimiste quant à la construction d'un modèle prédictif efficace, nous constatons que les classes sont plutôt décalées entres elles quelle que soit la variable.

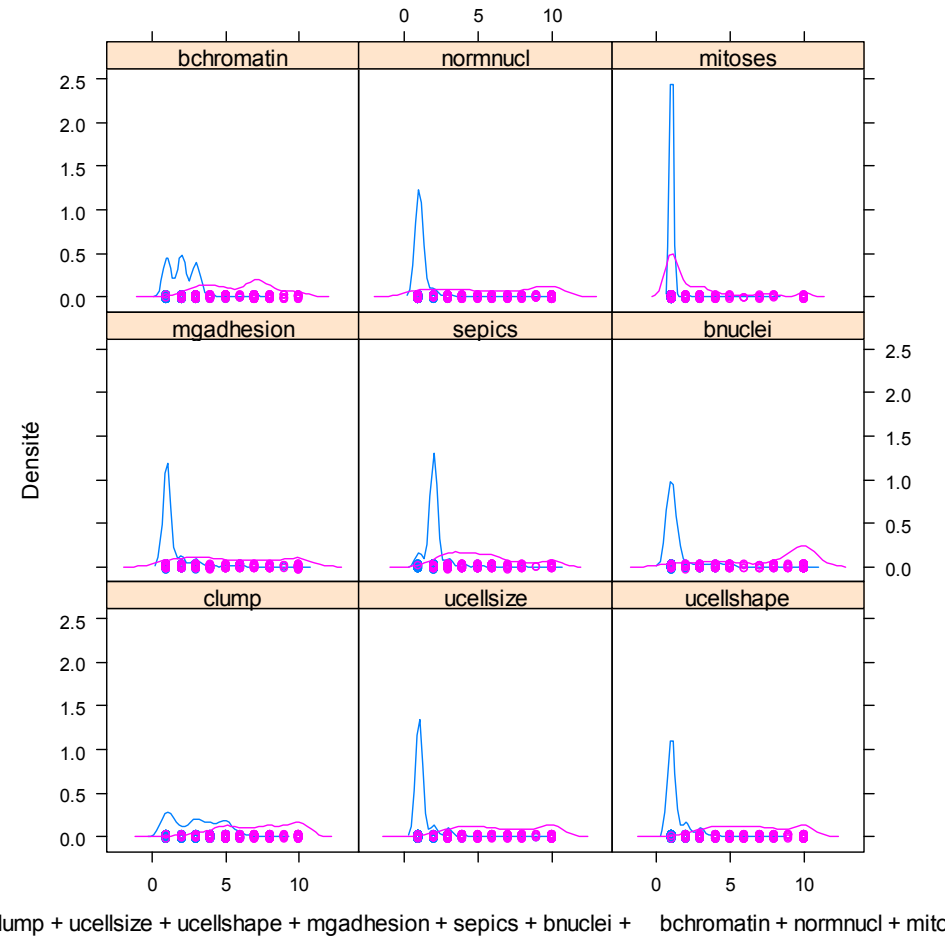
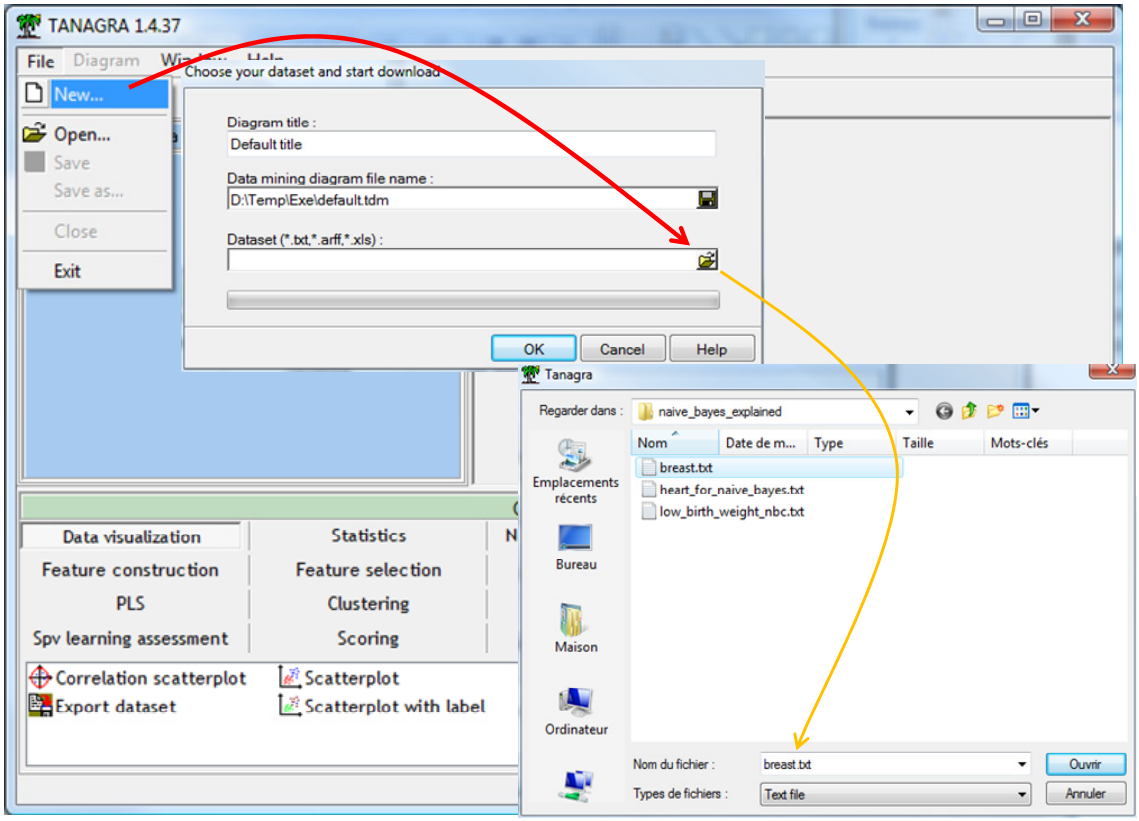


Figure 5 - Fichier "breast" - Fonctions de densité des prédictives conditionnellement aux classes

Dans ce qui suit, nous allons construire le modèle bayesien naïf sur les données « breast ». Nous mettrons un accent particulier sur la lecture du résultat. Concernant le modèle sous hypothèse d'homoscédasticité, puisqu'il est linéaire, nous comparerons les coefficients de l'hyper plan séparateur avec ceux fournis par d'autres techniques linéaires. Nous constaterons qu'ils sont cohérents.

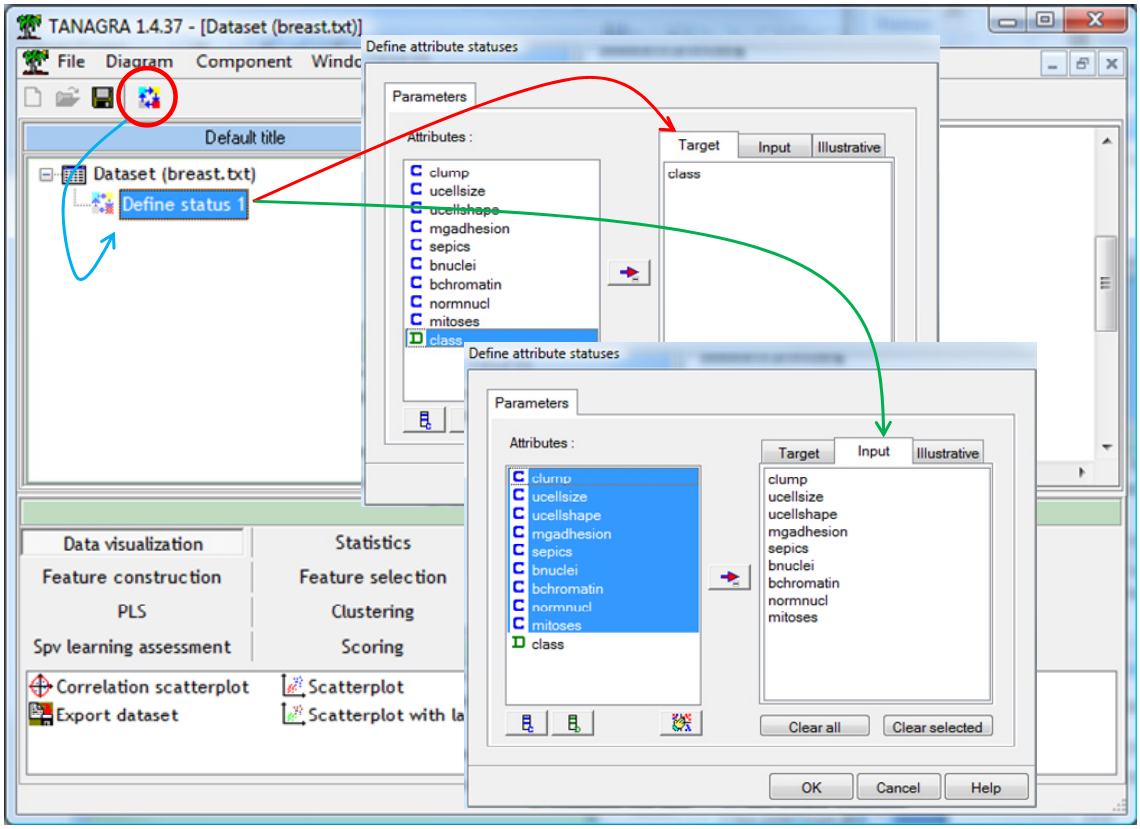
3.2 Classifieur bayesien naïf avec Tanagra

Après avoir démarré Tanagra, nous actionnons le menu FILE / NEW pour créer un nouveau diagramme. Nous sélectionnons le fichier « breast.txt » (format texte avec séparateur tabulation).



699 individus et 10 variables sont importés.

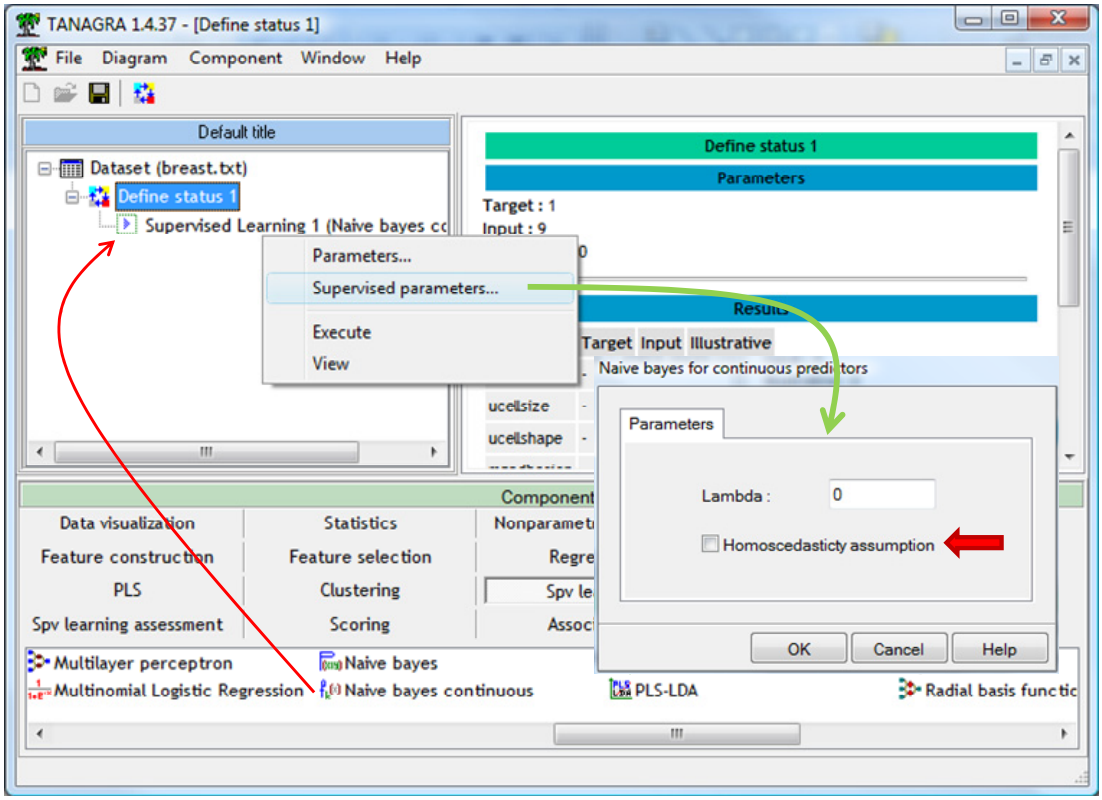
Nous insérons le composant DEFINE STATUS pour définir le rôle des variables. CLASS est la variable cible en TARGET, les autres (CLUMP...MITOSES) sont les prédictives en INPUT.



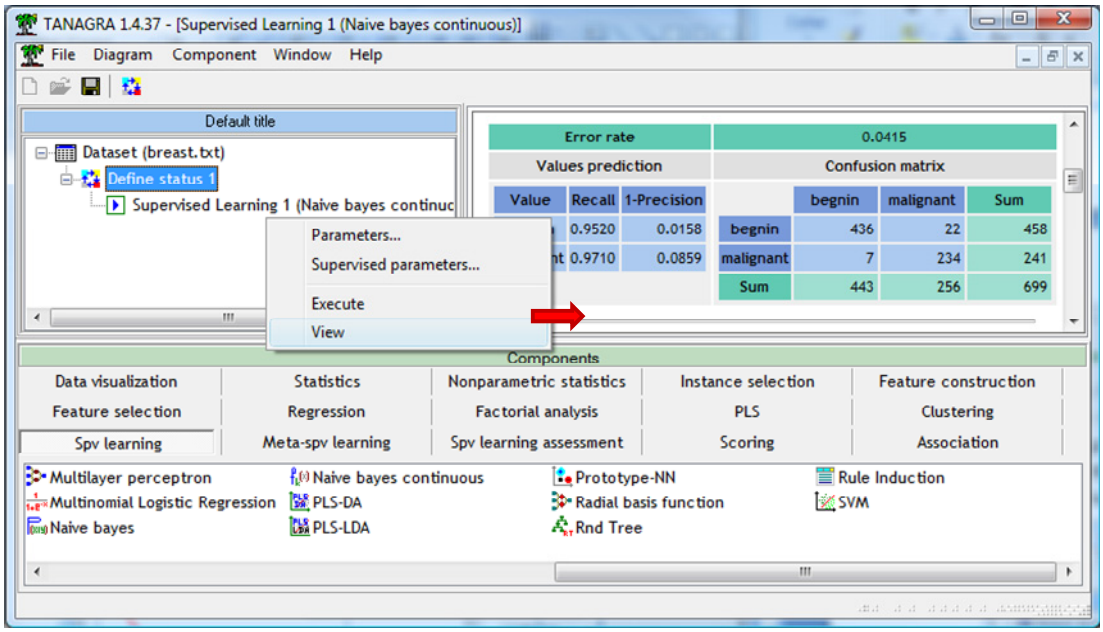
Nous cliquons sur le menu contextuel VIEW pour valider l'opération.

3.2.1 Modèle quadratique sous hypothèse d'hétéroscédasticité

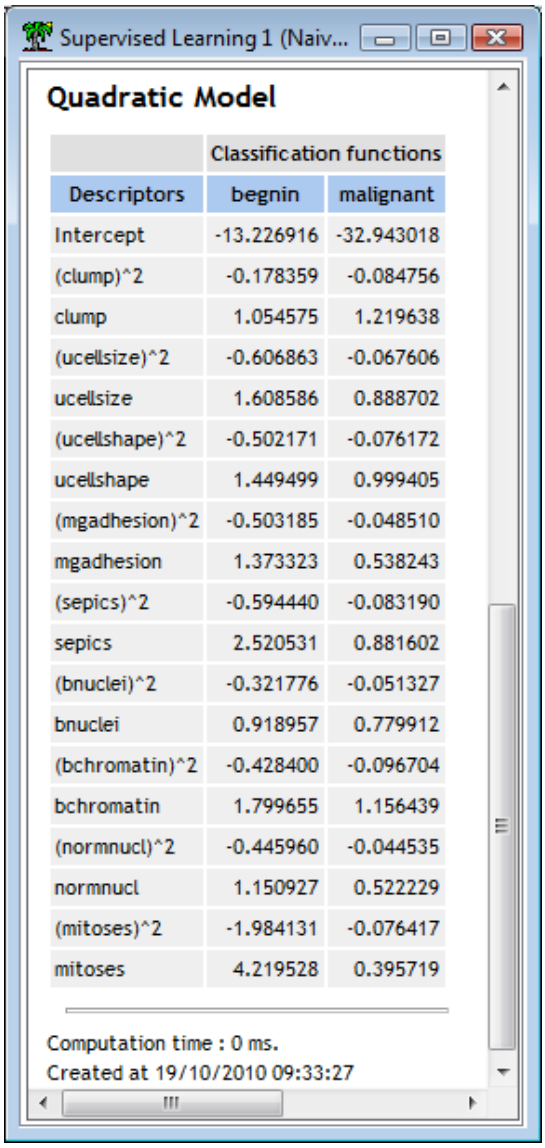
Nous introduisons le composant NAIVE BAYES CONTINUOUS (onglet SPV LEARNING) dans le diagramme. Nous cliquons sur le menu contextuel SUPERVISED PARAMETERS pour définir les paramètres de l'algorithme. Dans un premier temps, nous souhaitons fonctionner sous l'hypothèse d'hétéroscédasticité pour obtenir un modèle quadratique. Le paramètre lambda correspond à la correction introduite lors de l'estimation des probabilités a priori des classes.



Nous actionnons le menu VIEW. Nous obtenons la matrice de confusion et le taux d'erreur en resubstitution (4.15%). Ils sont donnés à titre indicatif. On sait que le taux d'erreur est souvent optimiste dans ces conditions.



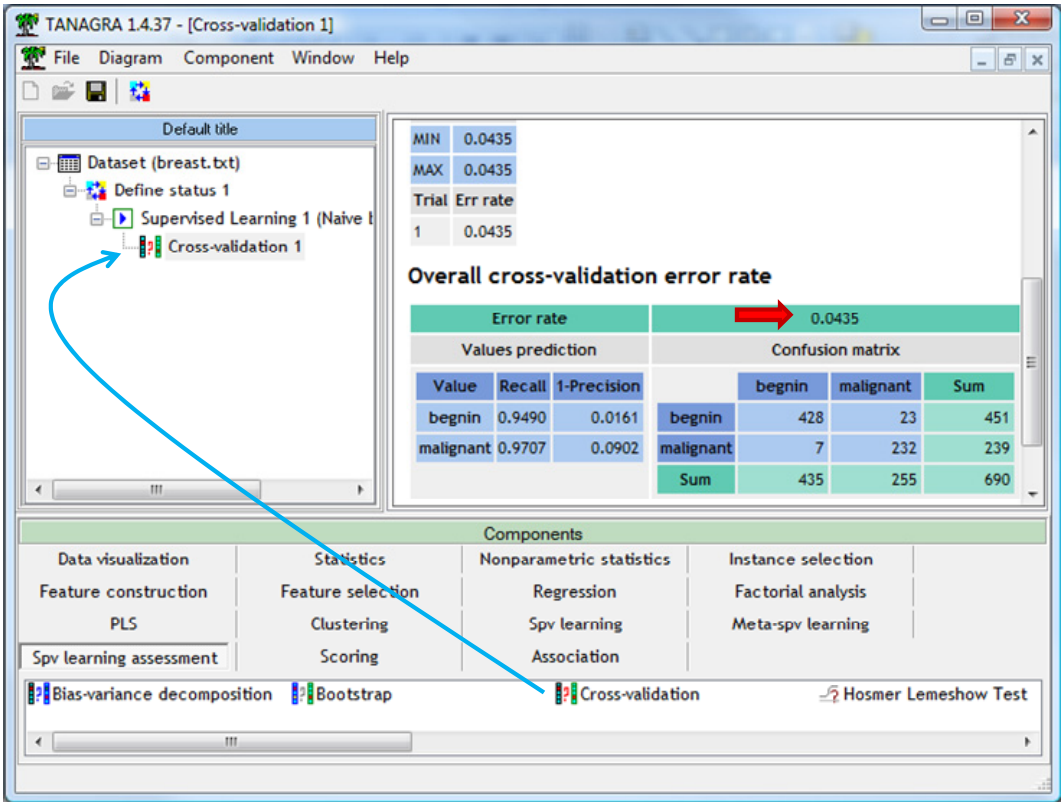
Plus bas, nous avons les coefficients des fonctions de classement.



Pour la modalité « begnin » par exemple, nous avons :

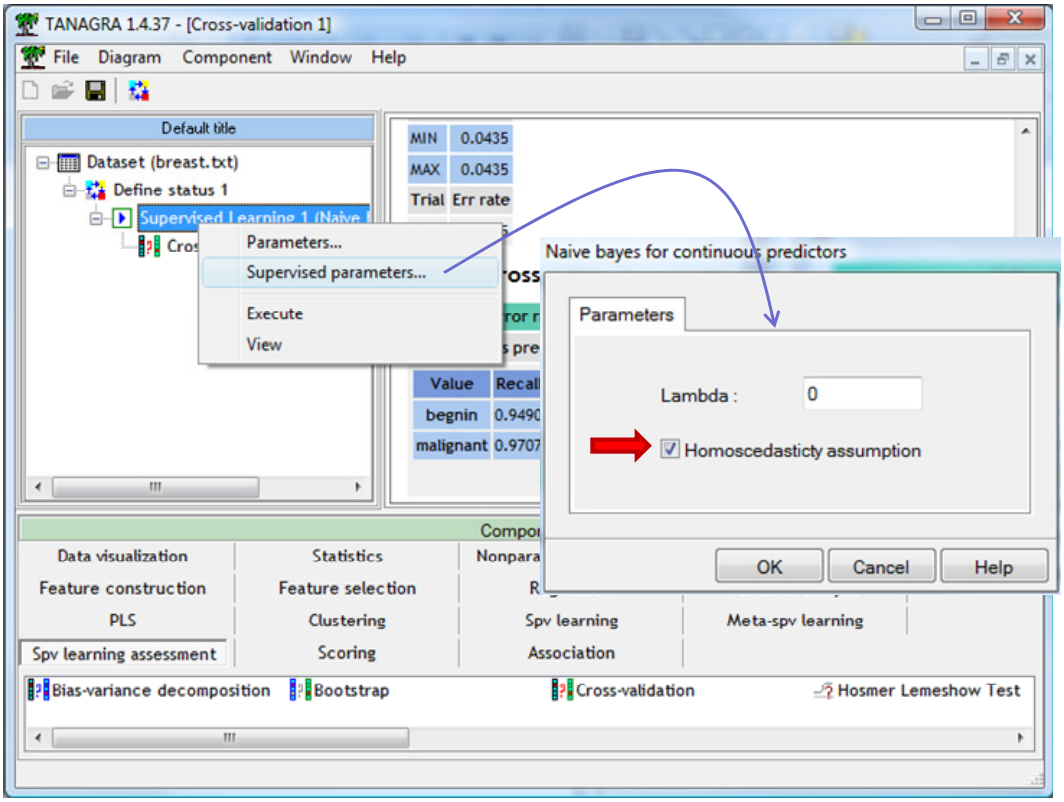
$$d(begnin, \aleph) = -13.226916 - 0.178359 \times (clump)^2 + 1.054575 \times clump + \cdots$$

Pour obtenir une évaluation moins biaisée des performances en généralisation du classifieur, nous introduisons la validation croisée (CROSS-VALIDATION, onglet SPV LEARNING ASSESSMENT). Nous cliquons sur VIEW. Tanagra nous indique que le taux d'erreur en généralisation serait de 4.35%.



3.2.2 Modèle linéaire sous hypothèse d'homoscédasticité

Voyons maintenant ce qu'il se passe si nous passons au modèle linéaire. Nous activons le menu SUPERVISED PARAMETERS du classifieur bayesien naïf. Nous sélectionnons l'option « Homoscedasticity Assumption ».



Nous cliquons à nouveau sur VIEW. Nous obtenons le modèle suivant.

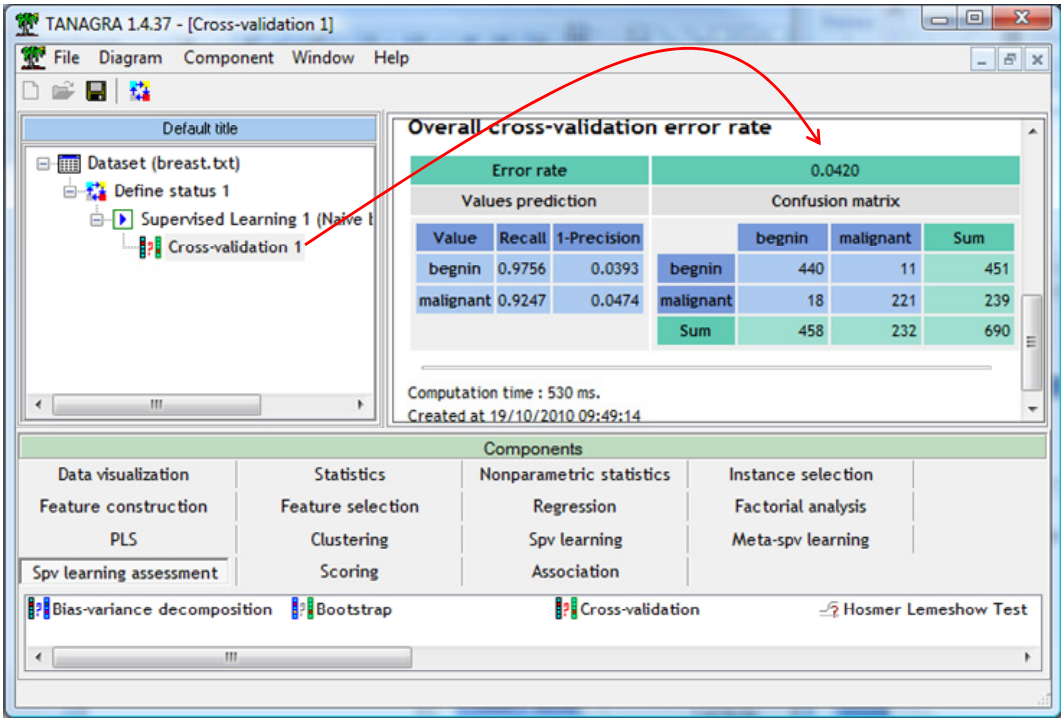
Linear Model

	Classification functions			
Descriptors	begnin	malignant	F(1,697)	p-value
Intercept	-4.787695	-49.901570	-	-
clump	0.764032	1.859474	733.206978	0.000000
ucellsize	0.429352	2.129259	1408.527213	0.000000
ucellshape	0.495435	2.251986	1419.305530	0.000000
mgadhesion	0.324866	1.320701	657.793700	0.000000
sepics	0.808864	2.021601	608.719555	0.000000
bnuclei	0.326527	1.737311	1374.423037	0.000000
bchromatin	0.825128	2.348867	933.287297	0.000000
normnucl	0.280463	1.274319	717.628041	0.000000
mitoses	0.439712	1.070712	152.040239	0.000000

Computation time : 0 ms.
Created at 19/10/2010 09:41:45

Principal différence par rapport au modèle quadratique, nous disposons d'une indication sur l'impact de chaque descripteur sur l'explication de la variable cible. Ils semblent tous très significatifs. Ce qui n'est guère étonnant étant donné les décalages entre les distributions conditionnelles, constatés lors de la description des données (Figure 5).

Pour avoir une idée des performances en généralisation, nous actionnons le menu contextuel VIEW de la validation croisée. Le taux d'erreur est de 4.20%. Le modèle linéaire est au moins aussi bon que le modèle quadratique.



3.2.3 Déduire une fonction de décision unique dans le cas binaire

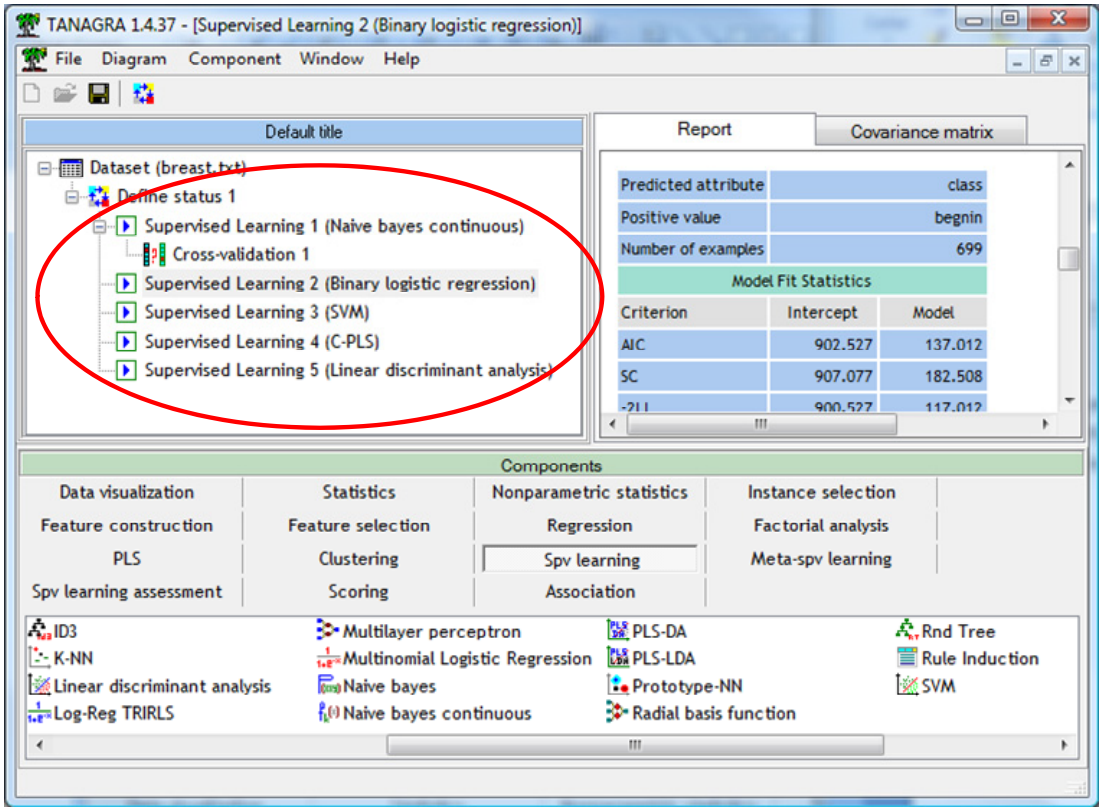
Puisque nous sommes dans un problème à deux classes, nous pouvons déduire l'équation de l'hyperplan séparateur. Il suffit de faire une différence termes à termes entre les coefficients des fonctions de classement. Nous obtenons pour notre fichier :

Descriptors	d(X)
Intercept	45.1139
clump	-1.0954
ucellsize	-1.6999
ucellshape	-1.7566
mgadhesion	-0.9958
sepics	-1.2127
bnuclei	-1.4108
bchromatin	-1.5237
normnucl	-0.9939
mitoses	-0.6310

3.3 Comparaison avec d'autres techniques linéaires

3.3.1 Comparaison des coefficients de la fonction de décision

Puisqu'il existe d'autres approches pour induire des séparateurs linéaires, voyons comment elles se positionnent par rapport au classifieur bayesien naïf (NBC). Nous avons testé : la régression logistique (BINARY LOGISTIC REGRESSION), les SVM linéaires (SVM), l'analyse discriminante PLS (C-PLS) et l'analyse discriminante prédictive (LINEAR DISCRIMINANT ANALYSIS - LDA). Nous utilisons le diagramme suivant dans Tanagra :



Nous recensons dans les fonctions de décision selon les méthodes.

Descriptors	NBC (linear)	Logistic.Reg	SVM	C-PLS	LDA
Intercept	45.1139	9.6710	3.6357	0.6053	20.2485
clump	-1.0954	-0.5312	-0.1831	-0.0350	-0.8867
ucellsize	-1.6999	-0.0058	-0.0290	-0.0195	-0.6081
ucellshape	-1.7566	-0.3326	-0.1275	-0.0219	-0.4381
mgadhesion	-0.9958	-0.2403	-0.0590	-0.0133	-0.1749
sepics	-1.2127	-0.0694	-0.0777	-0.0103	-0.2153
bnuclei	-1.4108	-0.4001	-0.1675	-0.0363	-1.2181
bchromatin	-1.5237	-0.4107	-0.1610	-0.0263	-0.5589
normnucl	-0.9939	-0.1447	-0.0650	-0.0151	-0.4619
mitoses	-0.6310	-0.5507	-0.1437	0.0087	-0.0773

Nous constatons une très forte cohérence des résultats, au moins en ce qui concerne les signes des coefficients. Il n'y a que C-PLS qui soit un peu différent concernant la variable « mitoses ».

3.3.2 Performances en classement

Le fichier « breast » est réputé facile à apprendre. La comparaison des performances en généralisation est anecdotique dans ce contexte. Nous donnons les taux d'erreur en validation croisée à titre indicatif, sans en tirer des conclusions tranchées sur l'efficacité des méthodes.

Method	Error rate (%)
NBC (quadratic)	4.35
NBC (linear)	4.20
Logistic regression	3.77
SVM (linear)	3.04
C-PLS	3.33
LDA	4.20

4 Le classifieur bayesien naïf dans les autres logiciels

4.1 Données

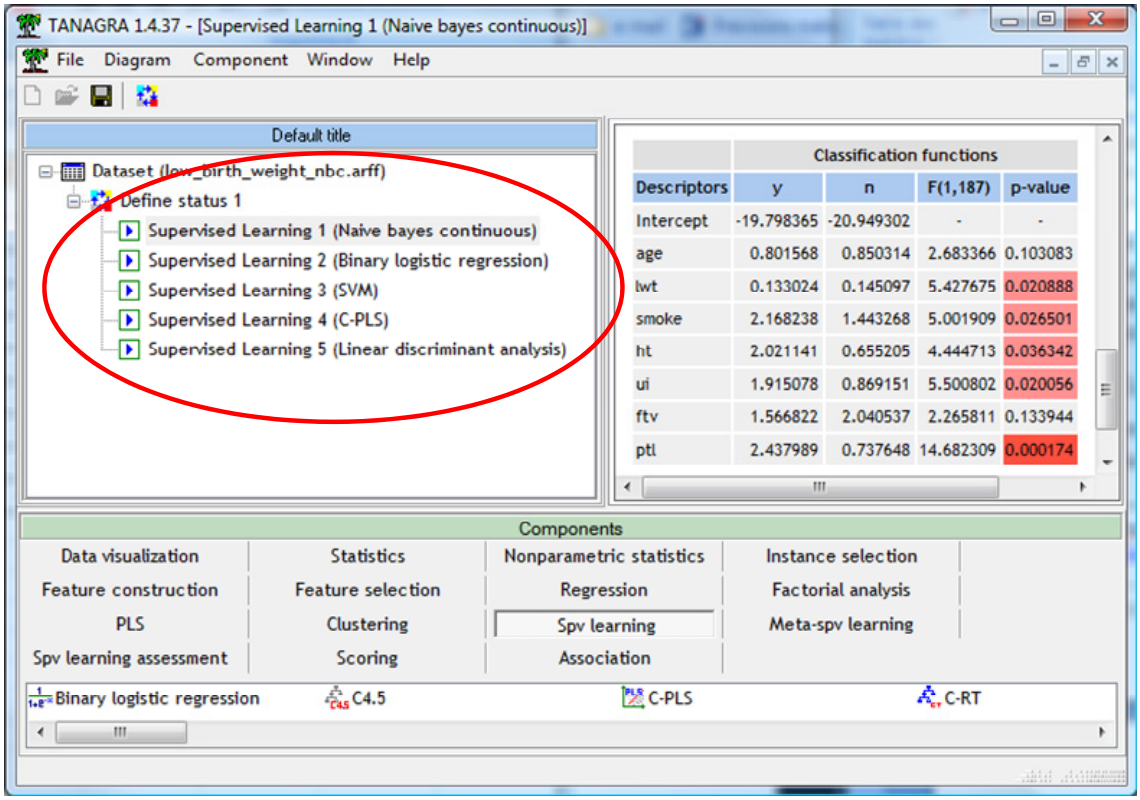
Nous utilisons le fichier « low_birth_weight_nbc.arff » dans cette section[7]. L'objectif est d'expliquer le faible poids des bébés à la naissance à partir des caractéristiques (poids, etc.) et du comportement de la mère (tabagisme, etc.).

Ici, plus que précédemment, les conditions d'application du classifieur bayesien naïf ne sont manifestement pas réunies. La majorité des variables sont binaires[8]. L'hypothèse de normalité ne tient pas la route du tout. Pourtant, nous constaterons que la technique est robuste. Elle donne des résultats comparables aux autres approches linéaires.

4.2 Comparaison des méthodes

Nous avons mené une étude comparable à la précédente. Nous avons lancé tour à tour les techniques.

[7] Hosmer and Lemeshow (2000) Applied Logistic Regression: Second Edition.
[8] http://www.statlab.uni-heidelberg.de/data/linmod/birthweight.html



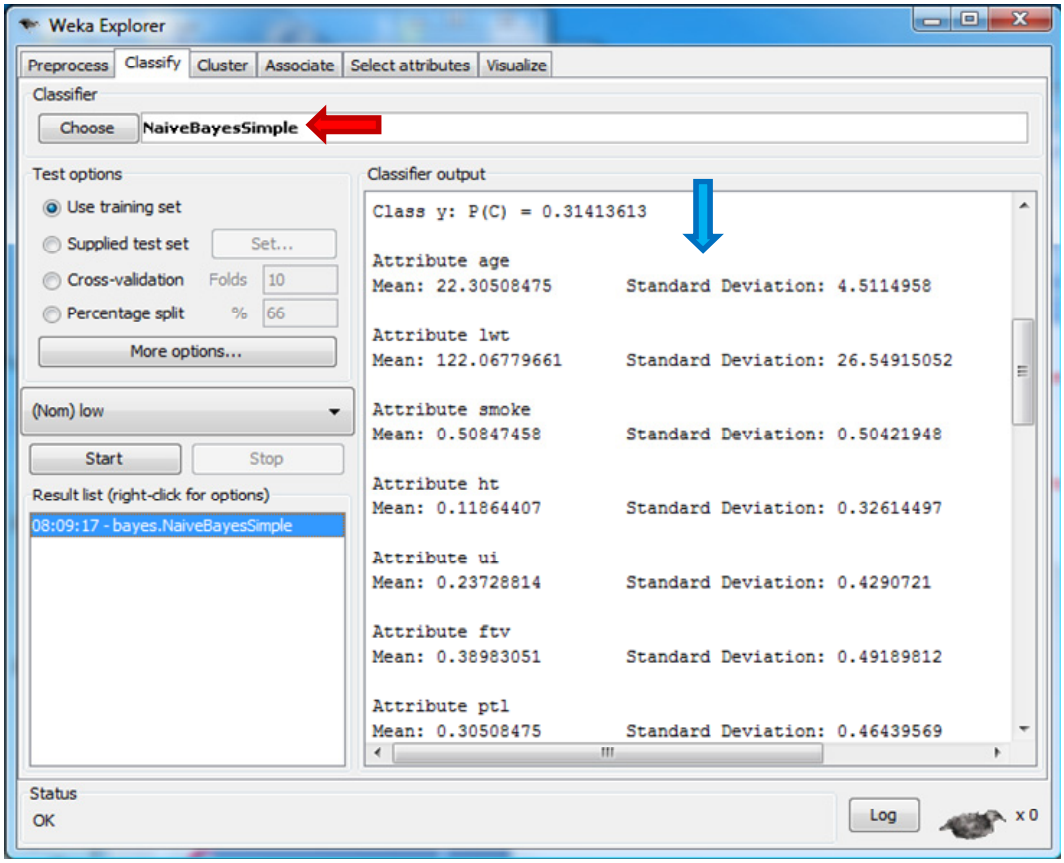
Nous obtenons les coefficients suivants à la sortie.

Descriptors	NBC (linear)	Logistic.Reg	SVM	C-PLS	LDA
Intercept	1.1509	1.4915	-0.9994	0.3284	1.0666
age	-0.0487	-0.0467	0.0000	-0.0073	-0.0415
lwt	-0.0121	-0.0140	0.0000	-0.0020	-0.0125
smoke	0.7250	0.4460	0.0012	0.0952	0.4921
ht	1.3659	1.8330	0.0043	0.3622	2.0134
ui	1.0459	0.6680	0.0014	0.1447	0.7927
ftv	-0.4737	-0.2551	-0.0006	-0.0529	-0.2684
ptl	1.7003	1.3368	1.9973	0.2893	1.5611

Les résultats sont sacrément cohérents, ne serait-ce qu'au niveau des signes des coefficients (mis à part SVM). Leurs valeurs sont étonnamment proches également, tout du moins en ce qui concerne le classifieur bayesien naïf, la régression logistique et l'analyse discriminante linéaire. Les interprétations du rôle des prédictives sur l'explication de la variable cible sont les mêmes. C'est plutôt rassurant.

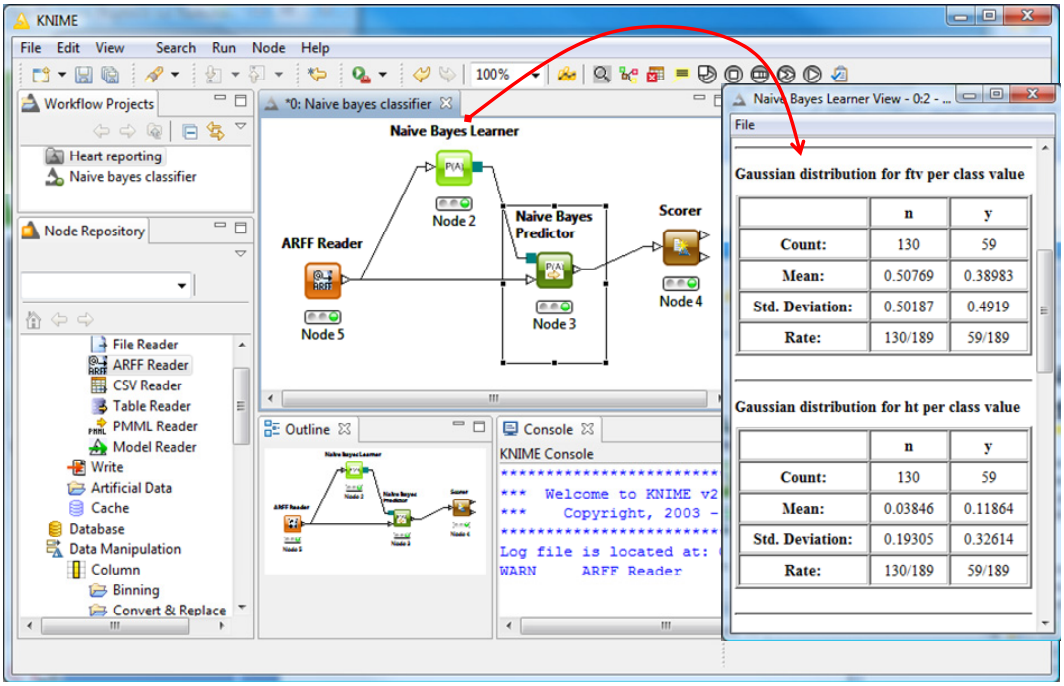
4.3 Implémentation avec Weka

Nous utilisons Weka en mode EXPLORER. Après avoir chargé les données, nous sélectionnons l'onglet CLASSIFY. Parmi la multitude de méthodes disponibles, nous choisissons NAIVE BAYES SIMPLE qui correspond au classifieur bayesien naïf gaussien et hétéroscédastique présenté dans ce document. Weka ne fournit pas de modèle explicite facile à déployer. Il se contente d'afficher les moyennes et écarts-type conditionnels pour chaque variable prédictive. A paramètre égal (modèle hétéroscédastique), la matrice de confusion en resubstitution) concorde avec celle proposée dans Tanagra.



4.4 Implémentation avec Knime

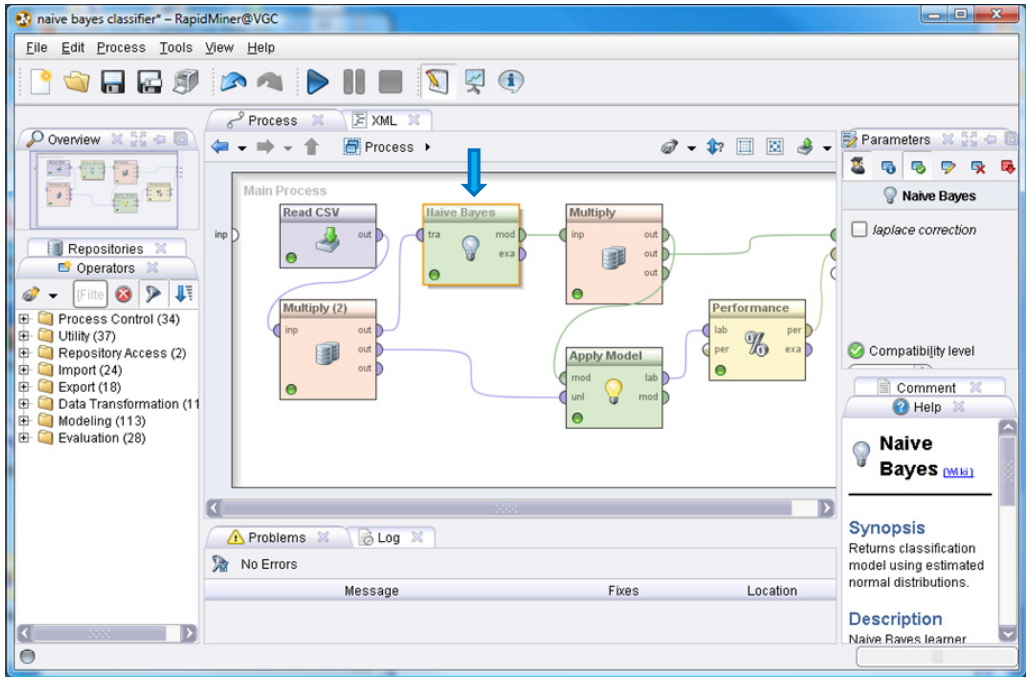
Nous construisons le diagramme suivant dans Knime 2.2.2.



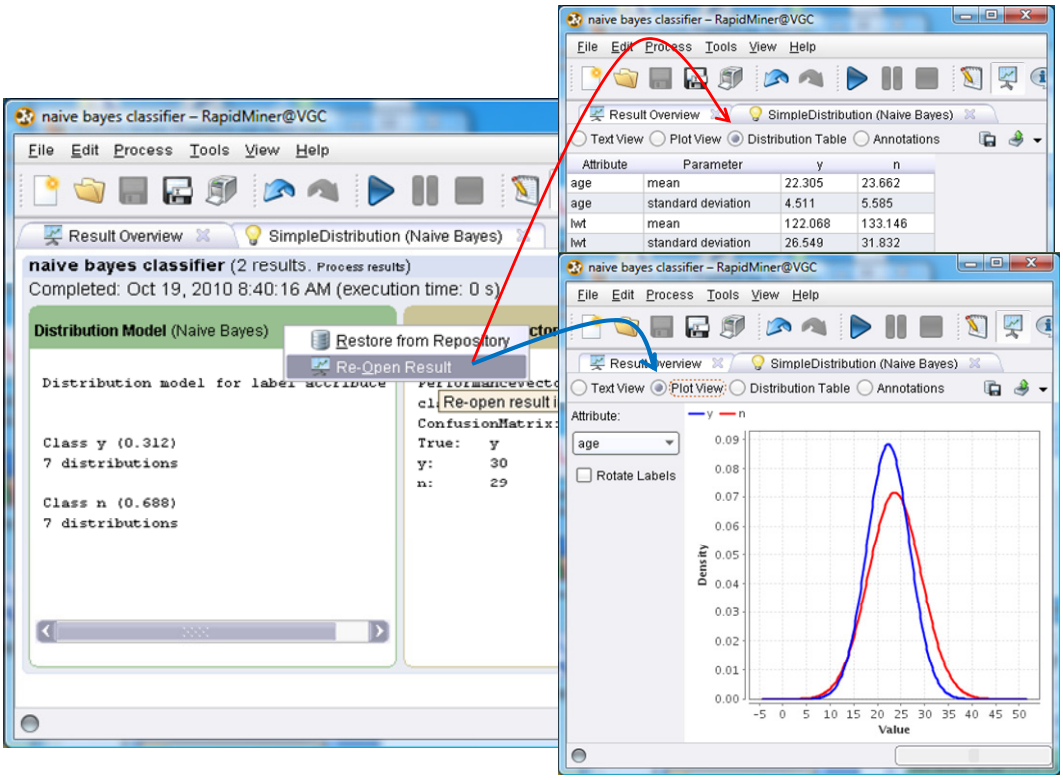
Tout comme Weka, Knime transcrit les paramètres des gaussiennes conditionnelles (moyennes et écarts-type). Nous avons les mêmes valeurs. Pourtant, curieusement, la matrice de confusion et le taux d'erreur en resubstitution ne sont pas les mêmes. Les classifieurs ne fonctionnent pas de la même manière. J'avoue ne pas très bien comprendre la source de ces disparités.

4.5 Implémentation avec RapidMiner

Le diagramme sous RapidMiner 5.0 est très similaire à celui de Knime. Nous appliquons le modèle sur les données d'apprentissage pour obtenir le taux d'erreur en resubstitution.



Après exécution, les résultats sont affichés dans une nouvelle fenêtre.



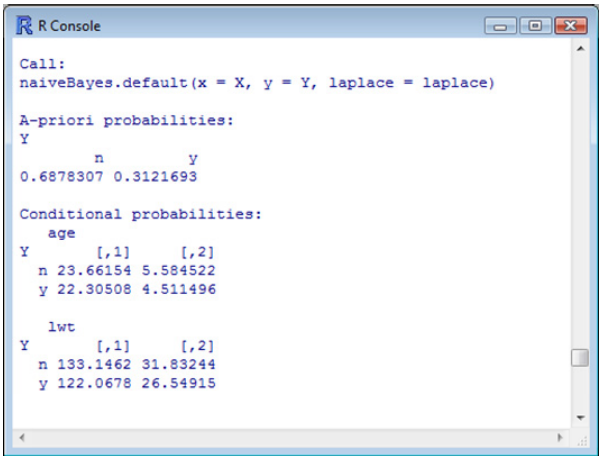
RapidMiner semble très peu loquace à première vue (Fenêtre OVERVIEW). Il faut sélectionner les en-têtes des sous fenêtres pour obtenir le détail des distributions conditionnelles, sous forme graphiques ou via l'affichage des paramètres estimés (moyenne et écart-type). Les résultats correspondent en tous points à ceux de Weka et de Tanagra. Aucun modèle explicite n'est fourni.

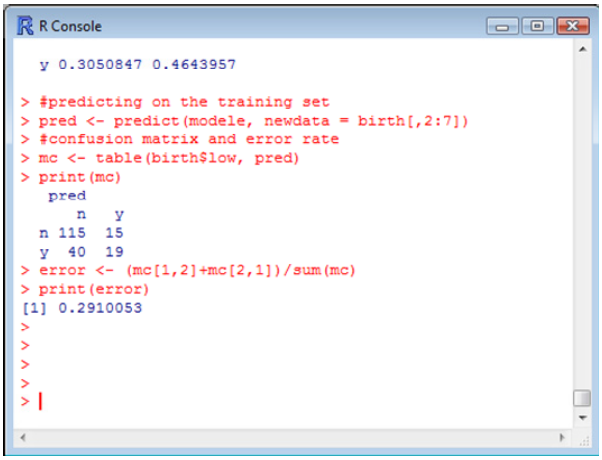
4.6 Implémentation avec R

Nous utilisons le package **e1071** pour R. Voici le code source pour : (1) produire le modèle prédictif ; (2) le visualiser ; (3) l'appliquer sur les données d'apprentissage ; (4) en déduire la matrice de confusion et le taux d'erreur.

```
#loading the dataset
birth <- read.table(file="low_birth_weight_nbc.txt",header=T,sep="\t")
summary(birth)
#loading the package
library(e1071)
#learning process
modele <- naiveBayes(low ~ ., data = birth)
print(modele)
#predicting on the training set
pred <- predict(modele, newdata = birth[,2:7])
#confusion matrix and error rate
mc <- table(birth$low, pred)
print(mc)
error <- (mc[1,2]+mc[2,1])/sum(mc)
print(error)
```

Nous obtenons les moyennes et écarts-type conditionnels. Les paramètres calculés sont cohérents avec ceux de Weka, de RapidMiner et de Tanagra (modèle hétéroscédastique).





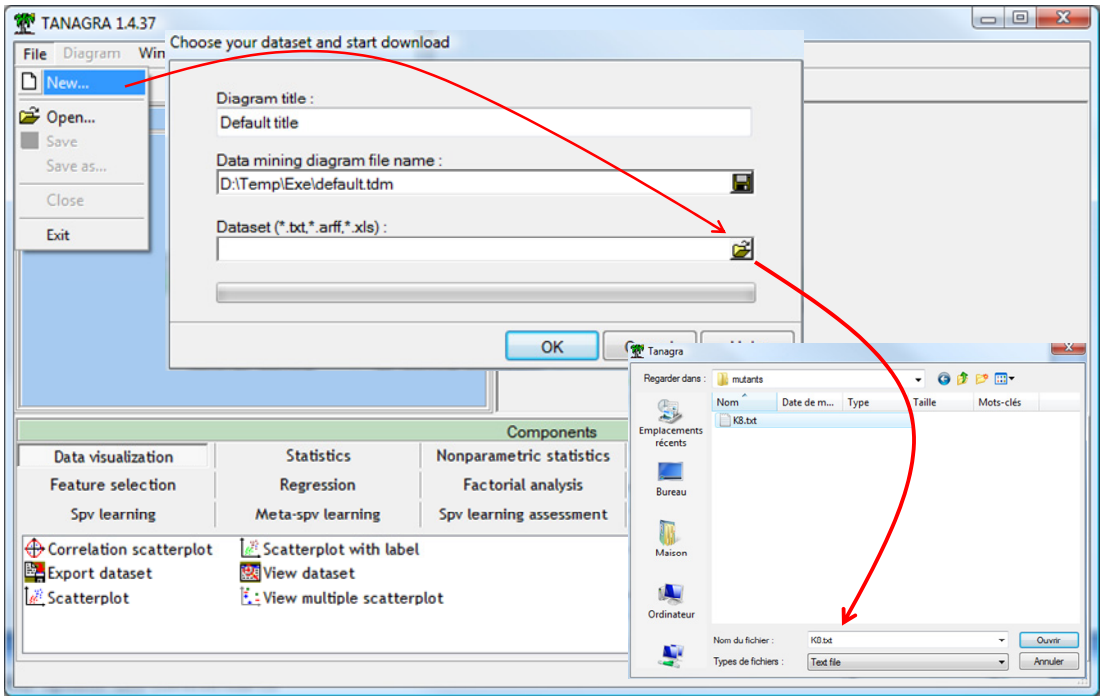
5 Traitement de très gros fichiers

Pourquoi s'embarrasser d'une nouvelle méthode linéaire alors qu'il en existe d'autres par ailleurs, très performantes, largement connues et reconnues ? La régression logistique par exemple est bien installée dans les études marketing. Les SVM, de par leur faible variance, semblent incontournables dès que le ratio nombre de variables – nombre d'observations s'inverse.

La raison est simple. **Le classifieur bayesien naïf est imbattable en termes de temps de traitement**. Cela peut être décisif dans l'appréhension des grandes bases de données. Dans cette section, nous analysons une base contenant 16592 observations et 5409 variables (5408 prédictives). Chargée en mémoire, elle occupe déjà une place considérable. Si la construction du modèle est gourmande en ressources, la question même de la faisabilité des calculs est posée.

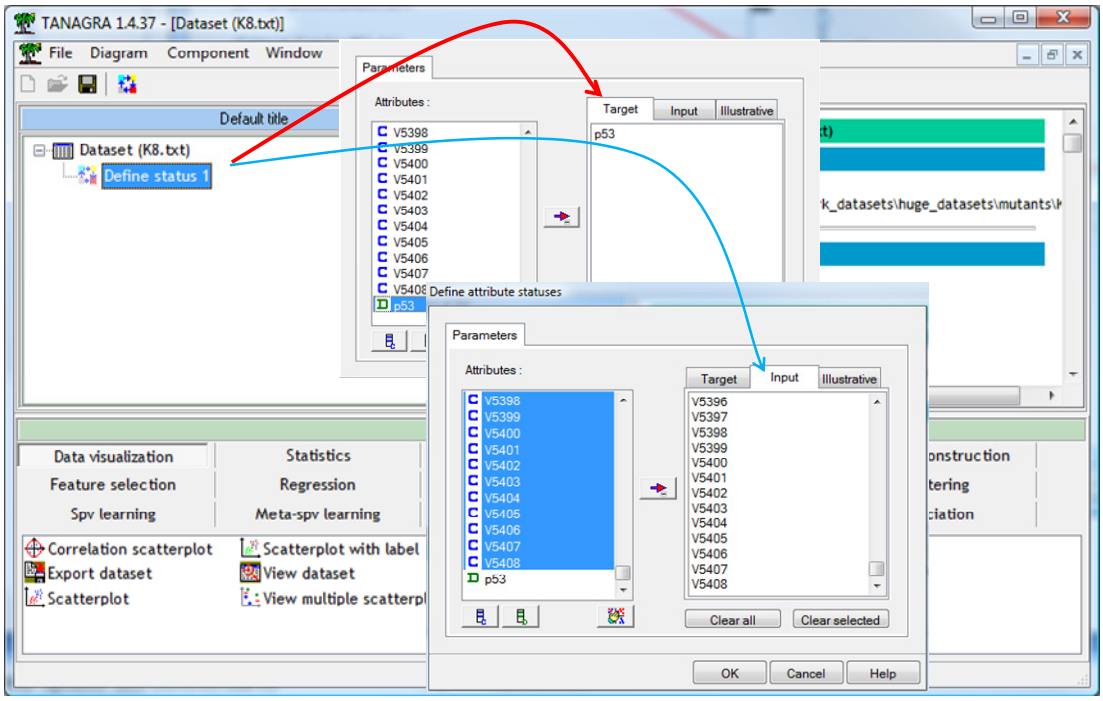
5.1 Importation des données

Nous utilisons la base « mutants » dans cette section (*la taille de la base est considérable, nous ne l'avons pas copiée sur notre serveur* -- http://archive.ics.uci.edu/ml/datasets/p53+Mutants). Nous devons prédire la variable « p53 » (active : positif vs. inactive : négatif). Après avoir démarré Tanagra, nous créons un nouveau diagramme en actionnant le menu FILE / NEW. Nous sélectionnons le fichier « K8.txt ».

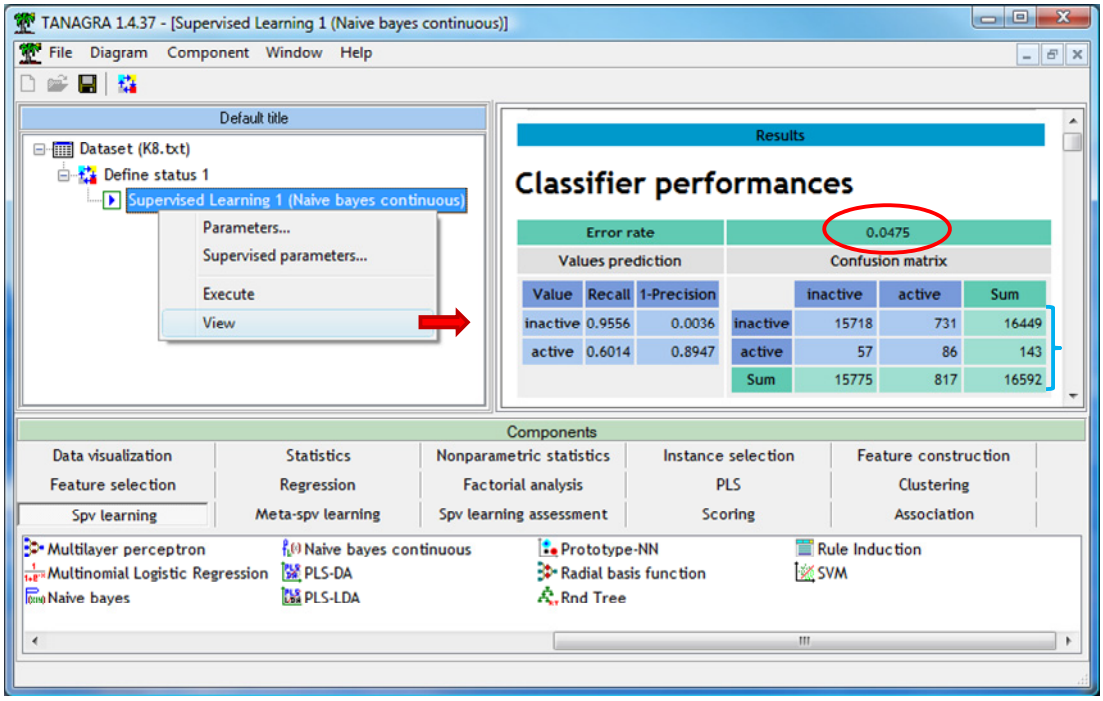


5.2 Apprentissage

Nous utilisons le composant DEFINE STATUS pour définir le rôle des variables. Nous plaçons p53 en TARGET, les autres variables en INPUT.



Nous insérons la méthode NAIVE BAYES CONTINUOUS. Nous actionnons le menu VIEW pour lancer les calculs et visualiser les résultats. Par défaut, Tanagra construit le modèle linéaire.

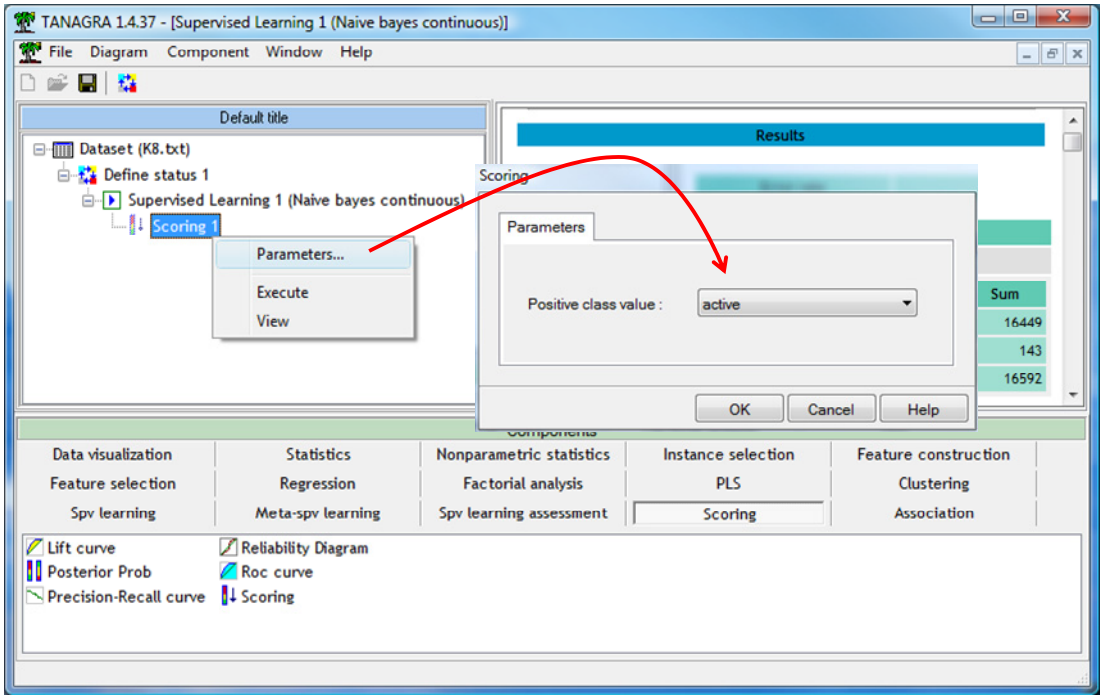


Le taux d'erreur en resubstitution est 4.75%. Mais ça ne veut absolument rien dire. D'une part, parce qu'il est calculé sur les données d'apprentissage. D'autre part, parce que les classes sont très déséquilibrées : 0.86% des observations sont positifs (active). Si on prédit systématiquement négatif (inactive), le taux d'erreur serait de 0.86%, bien meilleur que le chiffre annoncé ci-dessus.

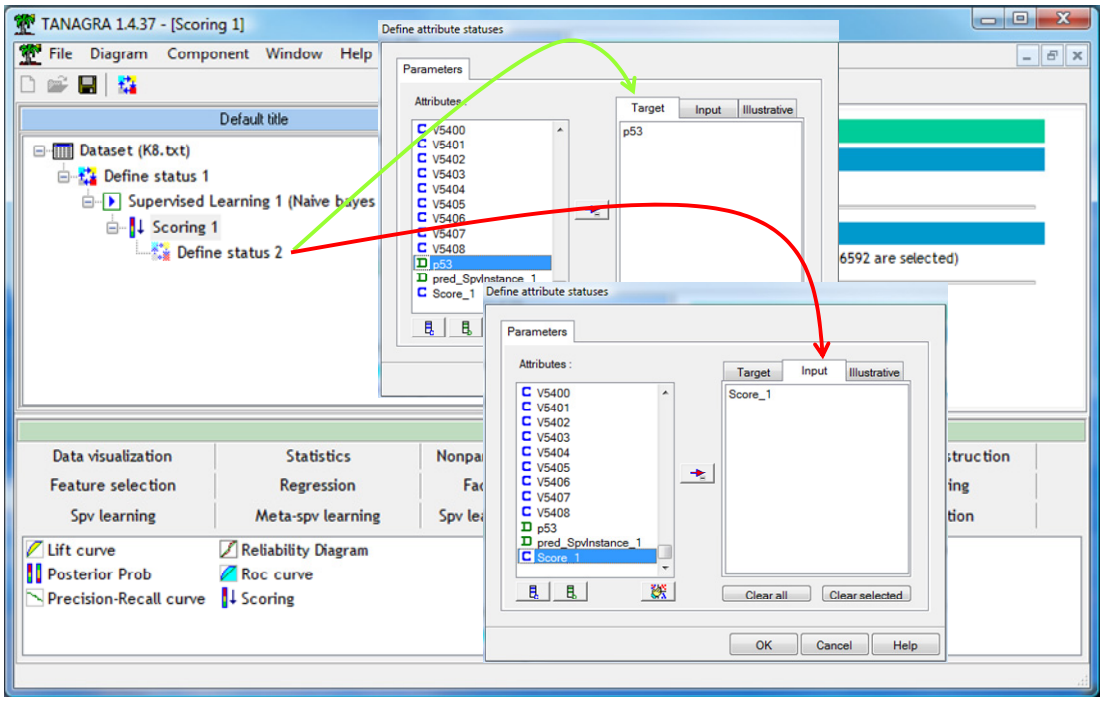
La durée de l'apprentissage a été de 10.7 secondes. L'occupation mémoire du modèle est négligeable par rapport à celle des données. Si nous avons à implémenter la régression logistique par exemple, rien que la matrice hessienne occuperait (5408 x 5408 x 8) / 2 ≈ 120 Mo en mémoire (x 8 parce que codée en double précision ; /2 en tenant compte du fait qu'elle est symétrique). Comme il va falloir l'inverser, il faut compter 120 Mo supplémentaire en mémoire. Sans compter le temps de construction de ces matrices. L'affaire est vite ingérable.

5.3 Construction de la courbe ROC

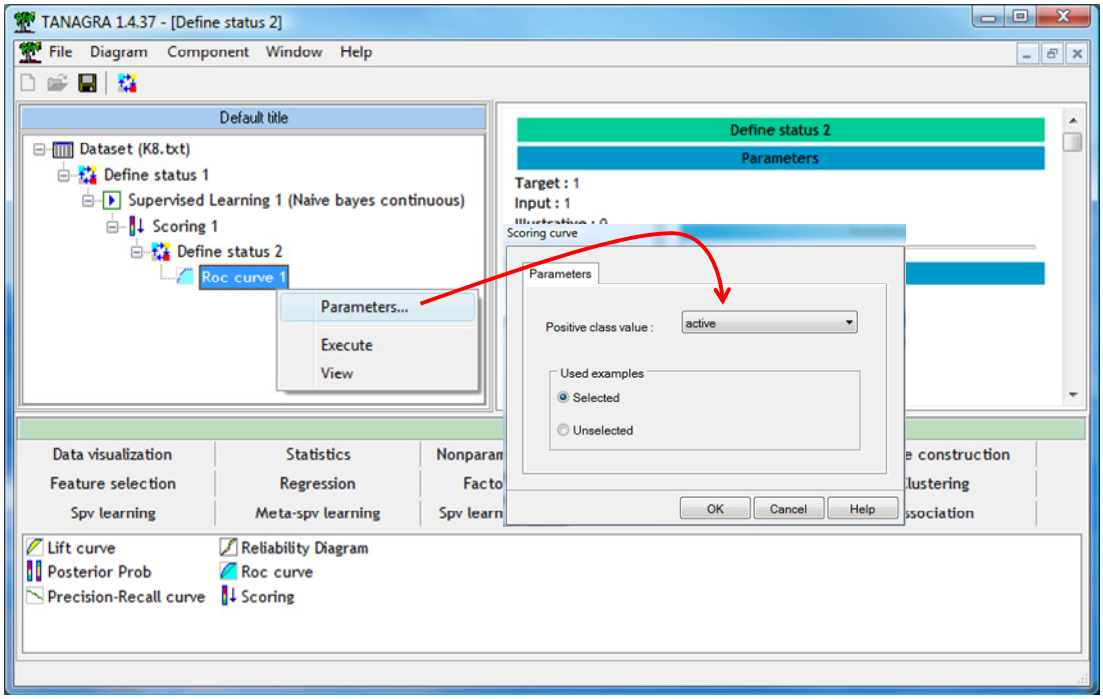
Nous souhaitons construire la courbe ROC pour évaluer le classifieur : est-ce qu'il est capable d'attribuer un score plus élevé aux individus positifs (qu'aux individus négatifs). Pour cela, nous devons « scorer » les individus à l'aide du composant SCORING (onglet SCORING). Nous désignons la classe positive (p53 = active) via le menu contextuel PARAMETERS.



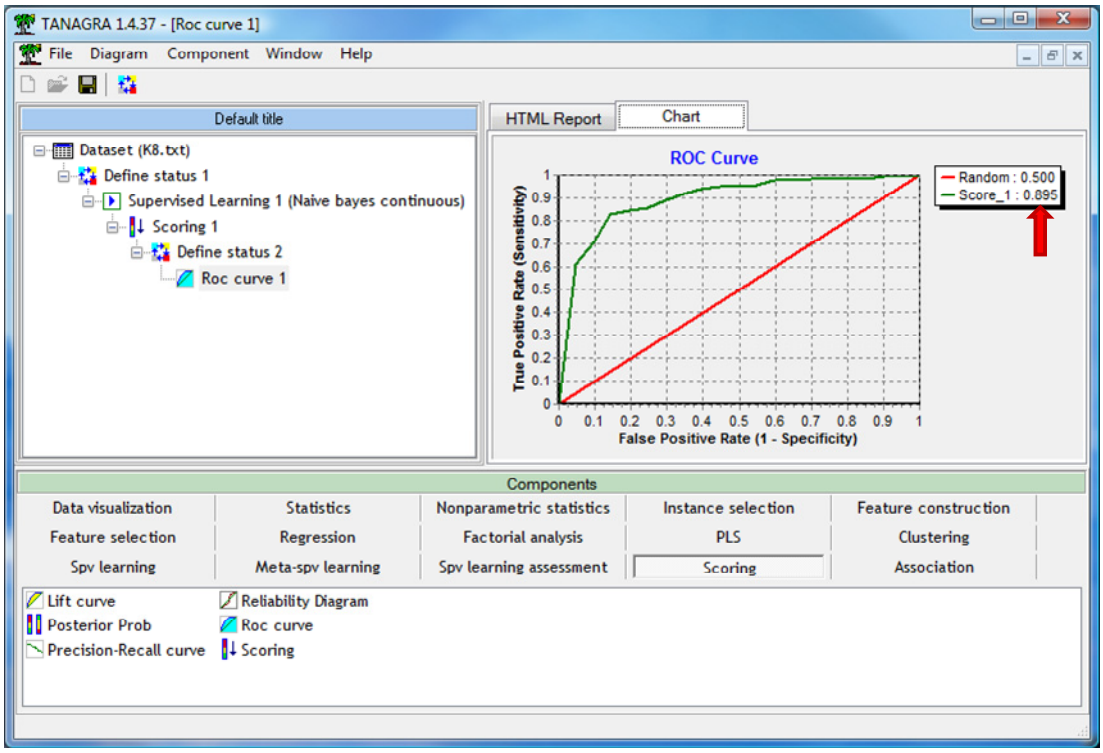
Via le menu DEFINE STATUS, nous précisons la variable cible (TARGET) et la colonne score (INPUT).



Il ne reste plus qu'à placer le composant ROC CURVE. Nous le paramétrons de manière à désigner « active » comme modalité positive de la variable cible. Nous construisons la courbe sur les données d'apprentissage, l'idée étant avant tout de montrer la faisabilité des calculs.



Nous cliquons sur VIEW. Le graphique apparaît avec une aire sous la courbe de AUC = 0.895.



5.4 Bilan – Temps de traitement et occupation mémoire

Voici un bilan du temps de traitement et de l'occupation mémoire[9] du logiciel à chaque étape.

[9] Mesurée à l'aide du gestionnaire de tâches Windows.

Etape	Temps de calcul (sec.)	Occupation mémoire (Ko)
Démarrage du logiciel	-	3 620
Chargement des données	25.7	489 252
Construction du classifieur	10.7	490 928
Scoring des observations	5.9	491 008
Courbe ROC	0.6	508 088

A niveau de performances en classement égal, je doute fort que l'on puisse faire mieux – plus rapide, moins gourmand en mémoire – avec une autre méthode produisant un classifieur linéaire. Dans Tanagra, elles ont toutes échoué sur notre jeu de données (régression logistique, SVM, analyse discriminante PLS, analyse discriminante de Fisher – *soit le logiciel a affiché « mémoire insuffisante », soit au bout d'un très long moment, je n'ai pas eu la patience d'attendre la fin des calculs*). Nous sommes parvenus aux mêmes conclusions avec les autres logiciels cités dans ce document.

6 Conclusion

Dans ce tutoriel, nous montrons qu'il est possible de produire un modèle explicite, sous la forme de combinaison linéaire des variables, et éventuellement de leurs carrés, pour le classifieur bayesien naïf avec des variables prédictives continues. Il vient en contrepoint du document consacré à la même technique, mais pour les prédicteurs catégoriels (http://tutoriels-data-mining.blogspot.com/2010/03/le-classifieur-bayesien-naif-revisite.html).

L'affaire n'est pas anodine car, ici également, nous cumulons tous les avantages. D'une part, si l'on s'y prendre bien, une seule passe sur les données permet de calculer toutes les moyennes et les écarts-type des variables prédictives, et d'en déduire les coefficients des fonctions de classement. L'apprentissage est très rapide, on peut appréhender de très grandes bases de données. D'autre part, la distribution du modèle est facilitée. Un simple jeu de coefficients [K fois (nombre de variables + 1) dans le pire des cas] est nécessaire à son déploiement.