# Classifieur naïf de Bayes pondéré pour flux de données

Christophe Salperwyck*, Vincent Lemaire**, Carine Hue**

*Powerspace, 13 rue Turbigo, 75002 Paris
** Orange Labs, 2 avenue Pierre Marzin, 22300 Lannion

**Résumé.** Un classifieur naïf de Bayes est un classifieur probabiliste basé sur l'application du théorème de Bayes avec l'hypothèse naïve, c'est-à-dire que les variables explicatives ($X_i$) sont supposées indépendantes conditionnellement à la variable cible ($C$). Malgré cette hypothèse forte, ce classifieur s'est avéré très efficace sur de nombreuses applications réelles et est souvent utilisé sur les flux de données pour la classification supervisée. Le classifieur naïf de Bayes nécessite simplement en entrée l'estimation des probabilités conditionnelles par variable $P(X_i|C)$ et les probabilités a priori $P(C)$. Pour une utilisation sur les flux de données, cette estimation peut être fournie à l'aide d'un « résumé supervisé en-ligne de quantiles ». L'état de l'art montre que le classifieur naïf de Bayes peut être amélioré en utilisant une méthode de sélection ou de pondération des variables explicatives. La plupart de ces méthodes ne peuvent fonctionner que hors-ligne car elles nécessitent de stocker toutes les données en mémoire et/ou de lire plus d'une fois chaque exemple. Par conséquent, elles ne peuvent être utilisées sur les flux de données. Cet article présente une nouvelle méthode basée sur un modèle graphique qui calcule les poids des variables d'entrée en utilisant une estimation stochastique. La méthode est incrémentale et produit un classifieur Naïf de Bayes Pondéré pour flux de données. Cette méthode est comparée au classique classifieur naïf de Bayes sur les données utilisées lors du challenge « Large Scale Learning ».

## 1 Introduction

Pour les données hors-ligne, des méthodes d'extractions de connaissances performantes et éprouvées depuis plusieurs années existent. Différents types de classifieurs ont été proposés : plus proches voisins, bayésien naïf, SVM, arbre de décision, système à base de règles... Mais avec l'apparition de nouvelles applications comme les réseaux sociaux, la publicité en-ligne, les données du Web... la quantité de données et leurs disponibilités ont changé. Les données auparavant facilement disponibles et pouvant tenir en mémoire (données hors-ligne) sont devenus massives et visibles une seule fois (flux de données). La plupart des classifieurs, prévus pour fonctionner hors-ligne, ne peuvent généralement pas s'appliquer directement sur un flux de données.

Depuis les années 2000, l'extraction de connaissances sur flux de données est devenue un sujet de recherche à part entière. De nombreux travaux traitant cette nouvelle problématique ont été proposés (Salperwyck et Lemaire, 2011; Gama, 2010). Parmi les solutions aux problèmes

de l'apprentissage en-ligne sur flux de données, les algorithmes d'apprentissage incrémentaux sont l'une des techniques les plus utilisées. Ces algorithmes sont capables de mettre à jour leur modèle à partir d'un seul nouvel exemple. Cependant la plupart d'entre eux, bien qu'étant incrémentaux, ne sont pas capables de traiter des flux de données car leur complexité n'est pas linéaire.

Dans cet article, on s'intéresse plus particulièrement à l'un des classifieurs les plus utilisés dans l'état de l'art pour réaliser une classification supervisée en ligne : le classifieur naïf de Bayes. Nous modifions ce classifieur de manière à réaliser un apprentissage en ligne pour flux de données. Ce classifieur ne nécessite en entrée que des probabilités conditionnelles $P(X_i|C)$ (où $X_i$ représente une variable explicative et $C$ une classe du problème de classification) et sa complexité en prédiction est très faible, ce qui le rend adapté aux flux.

Néanmoins, dans le cadre de l'apprentissage hors-ligne, il a été prouvé qu'en sélectionnant les variables (Koller et Sahami, 1996; Langley et Sage, 1994) et/ou en pondérant les variables (Hoeting et al., 1999) on obtient des résultats sensiblement meilleurs. De plus Boullé dans (Boullé, 2006b) a montré le lien entre pondération des variables et moyennage de plusieurs classifieurs naïf de Bayes dans le sens où, à la fin de l'apprentissage, les deux processus produisent des modèles similaires. Ces modèles se différencient du classifieur naïf de Bayes par l'ajout d'une pondération sur chaque variable. Ces poids peuvent être optimisés directement comme cela a déjà été réalisé dans (Guigourès et Boullé, 2011) mais de manière hors-ligne.

Le présent article présente une nouvelle méthode pour estimer incrémentalement les poids d'un classifieur Naïf de Bayes Pondéré (NBP) dans le cadre des flux de données. Cette méthode utilise un modèle graphique proche d'un réseau de neurones artificiels. Le plan de cet article est le suivant : notre modèle graphique ainsi que la méthode permettant d'apprendre les poids à attribuer aux variables explicatives sont présentés au cours de la section 2. La section 3 décrit comment les estimations des probabilités conditionnelles à la classe ($P(X_i|C)$), utilisées en entrée du modèle graphique, sont estimées. La section 4 présente une étude expérimentale de notre classifieur Naïf de Bayes Pondéré (NBP) entraîné incrémentalement sur les bases de données ayant servies au "large scale learning challenge". Enfin, la dernière section conclut cet article.

# 2 Classifieur Naïf Bayésien Pondéré incrémental

## 2.1 Introduction : le classifieur Naïf de Bayes (NB) et le classifieur Naïf de Bayes Moyenné (NBM)

Le classifieur Bayésien naïf est une méthode d'apprentissage supervisé qui repose sur une hypothèse simplificatrice forte : les variables $X_i$ sont indépendantes conditionnellement à la classe à prédire. Cette hypothèse naïve ne permet pas de modéliser les interactions entre différentes variables. Cependant, sur de nombreux problèmes réels, cette limitation n'a que peu d'impact (Hand et Yu, 2001; Langley et al., 1992). L'idée de départ de ce classifieur vient de la formule de Bayes : $P(C|X) = \frac{P(C)P(X|C)}{P(X)}$. La probabilité conditionnelle jointe $P(X|C)$ étant difficilement estimable on utilise la version naïve (appelée par la suite NB) de ce classi-

fieur. La probabilité de la classe devient dans ce cas :

$$P(C_k|X) = \frac{P(C_k)\prod_i P(X_i|C_k)}{\sum_{j=1}^{K}(P(C_j)\prod_i P(X_i|C_j))} \quad (1)$$

où $j$ est l'indice de la classe ($j \in \{1, ..., K\}$), $i$ l'indice de la variable explicative et $k$ une classe d'intérêt.

La classe prédite est celle qui maximise la probabilité conditionnelle $P(C_k|X)$. Les probabilités $P(X_i|C_k)$ peuvent être estimées par intervalle à l'aide d'une discrétisation pour les variables numériques. Pour les variables catégorielles, cette estimation peut se faire directement si la variable prend peu de valeurs différentes ou après un groupage dans le cas contraire. Le dénominateur de l'équation 1 normalise le résultat tel que $\sum_k P(C_k|X) = 1$. Un des avantages de ce classifieur dans le contexte des flux de données réside en sa faible complexité en déploiement, complexité qui ne dépend que du nombre de variables utilisées.

L'état de l'art montre toutefois que le classifieur bayésien naïf peut être amélioré de deux manières : (i) en sélectionnant les variables (Koller et Sahami, 1996; Langley et Sage, 1994) ; (ii) en pondérant les variables (Hoeting et al., 1999) ce qui est proche d'un moyennage de modèles bayésiens (BMA = Bayesian Model Averaging) (Hoeting et al., 1999) ; ces deux processus, sélection-pondération, pouvant être mélangés de manière itérative. Le classifieur bayésien naïf moyenné résultant est similaire au classifieur bayésien naïf mais ajoute une pondération par variable tel que :

$$P(C_k|X) = \frac{P(C_k)\prod_i P(X_i|C_k)^{w_i}}{\sum_{j=1}^{K}(P(C_j)\prod_i P(X_i|C_j)^{w_i})} \quad (2)$$

où chaque variable explicative $i$ est pondérée par un poids $w_i$ dans l'intervalle $[0, 1]$).

L'approche revient à un moyennage de modèles et en possède les qualités. Le moyennage de modèles vise à combiner la prédiction d'un ensemble de classifieurs de façon à améliorer les performances prédictives. Ce principe a été appliqué avec succès dans le cas du bagging (Breiman, 1996) qui exploite un ensemble de classifieurs appris sur une partie des exemples. Dans ces approches, le classifieur moyenné résultant procède par vote des classifieurs élémentaires pour effectuer sa prédiction. A l'opposé des approches de type bagging, où chaque classifieur élémentaire se voit attribuer le même poids, le moyennage de modèles bayésiens (BMA = Bayesian Model Averaging) (Hoeting et al., 1999) pondère les classifieurs selon leur probabilité *a posteriori*.

## 2.2 L'approche proposée : Naïf Bayésien Pondéré incrémental (NBP)

Lorsque l'on se place dans le cadre de l'apprentissage hors-ligne, les poids du classifieur Naïf de Bayes Moyenné peuvent être estimés de différentes manières : (i) par moyennage de modèles (Hoeting et al., 1999) ; (ii) par moyennage de modèles avec optimisation des poids basée sur un critère MDL (Minimum Description Length) (Boullé, 2006b) ; (iii) par optimisation directe des poids par une descente de gradient (Guigourès et Boullé, 2011). Toutefois toutes ces méthodes fonctionnent en chargeant toutes les données en mémoire et nécessitent de les relire plusieurs fois. L'approche proposée dans cet article optimise directement les poids du classifieur et est capable de fonctionner sur flux de données.

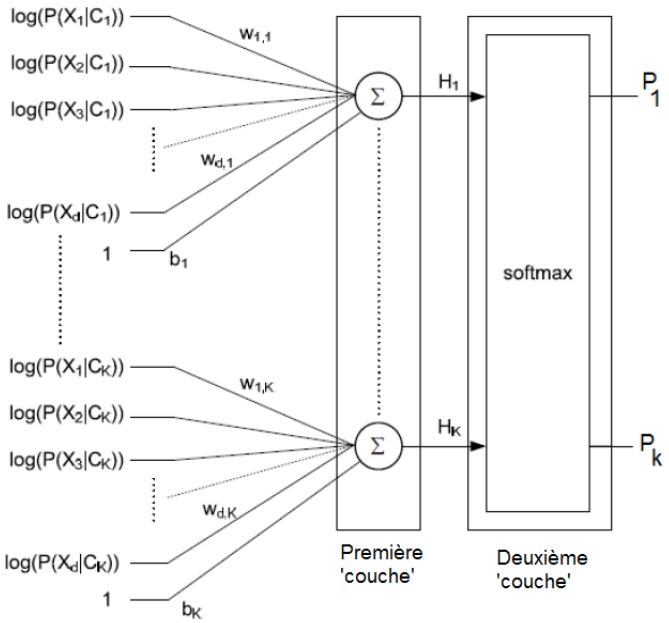


FIG. 1 – *Modèle graphique pour l'optimisation des poids du classifieur bayésien moyenné.*

La première étape consiste à créer un modèle graphique (voir Figure 1) (Whittaker, 1990) dédié à l'optimisation des poids. Il permet de réécrire l'équation 2 sous la forme d'un modèle graphique où le classifieur bayésien naïf pondéré reçoit un poids par variable et par classe tel que présenté dans l'équation 3. Les poids que nous cherchons à optimiser sont donc plus nombreux dans ce modèle graphique. En effet le poids n'est plus seulement associé à la variable, mais à la variable conditionnellement à la classe, soit : (i) $w_{ik}$ le poids associé à la variable $i$ et à la classe $k$ et (ii) $b_k$ le biais lié à la classe $k$. Ce biais correspond à l'estimation de la probabilité $P(C)$ et peut donc varier au cours du temps.

La première couche du modèle graphique est une couche linéaire réalisant une somme pondérée $H_k$ pour chaque classe $k$ , tel que $H_k = \sum_{i=1}^{d} w_{ik} log(P(X_i|C_k)) + b_k$. La seconde couche du modèle graphique est un $Softmax$ tel que : $P_k = \frac{e^{H_k}}{\sum_{j=1}^{K} e^{H_j}}$. Finalement le modèle graphique proposé, dans le cas où les entrées sont les logs des estimation conditionnelles aux classes ($log(p(X_i|C_k), \forall i, k)$, donne en sortie les valeurs $P_k$ ($\forall k$) telles que :

$$P_k = \frac{e^{b_k + \sum_{i=1}^{d} w_{i,k} log(p(X_i|C_k))}}{\sum_{j=1}^{K} e^{b_j + \sum_{i=1}^{d} w_{i,j} log(p(X_i|C_j))}} \tag{3}$$

c'est à dire des $P_k = P(C_k|X)$.

Les variables positionnées en entrée de ce modèle sont issues des résumés univariés construits sur le flux. Ceux-ci seront présentés dans la section suivante (section 3).

L'optimisation des poids est réalisée à l'aide d'une descente de gradient stochastique pour une fonction de coût donnée. Pour un exemple donné $X$, la règle de modification des poids

est :

$$w_{ij}^{t+1} = w_{ij}^{t} - \eta \frac{\partial co\hat{u}t^X}{\partial w_{ij}} \tag{4}$$

où $co\hat{u}t^X$ est la fonction de coût appliquée à l'exemple $X$ et $\frac{\partial co\hat{u}t^X}{\partial w_{ij}}$ la dérivée de la fonction de coût vis à vis des paramètres du modèle, ici les poids $w_{ij}$. Le calcul de cette dérivée (détaillé dans l'annexe de cet article) aboutit à :

$$\frac{\partial C}{\partial H_k} = P_k - T_k, \forall k \tag{5}$$

où $T_k$ désigne les valeurs de probabilité désirées (target) et $P_k$ les sorties obtenues. Il ne reste ensuite plus qu'à inclure la partie couche linéaire de notre modèle graphique pour avoir les dérivées partielles $\frac{\partial co\hat{u}t}{\partial w_{ik}}$. La modification des poids a donc une complexité calculatoire très faible.

La méthode de descente de gradient en-ligne utilisée dans cet article est celle utilisée habituellement pour réaliser une rétropropagation et 3 principaux paramètres (Lecun et al., 1998) sont à prendre en compte : (i) la fonction de coût ; (ii) le nombre d'itérations ; (iii) le pas d'apprentissage.

Dans le cadre de la classification supervisée le meilleur choix pour la fonction de coût, du fait que les sorties du modèle graphique à apprendre prennent uniquement deux valeurs $\{0, 1\}$, est le log vraisemblance (Bishop, 1995), qui optimise $log(P(C_k|X))$. Le nombre d'itérations est dans notre cas égal à 1 du fait que le modèle est mis à jour après chaque exemple et seulement une fois. Etant donné que l'apprentissage est réalisé sur un flux de données, nous choisissons de n'effectuer qu'une itération par exemple du flux, de ne pas utiliser ni d'early stopping (Prechelt, 1997) ni d'ensembles de validation (Amari et al., 1997). Finalement le seul paramètre à ajuster est le pas d'apprentissage. Une valeur trop faible aboutit à une convergence longue pour atteindre le minimum de la fonction de coût, alors qu'un pas trop grand ne permet pas d'atteindre ce minimum. Dans le cas d'un apprentissage hors-ligne il est possible de régler sa valeur par une méthode de validation croisée mais dans le cas de l'apprentissage sur flux, ou en une passe, ceci n'est pas envisageable. Pour les expérimentations de cet article un pas fixe ($\eta = 10^{-4}$) a été choisi. Cependant si la présence de dérive de concept est suspectée il peut être intéressant d'avoir un pas d'apprentissage adaptatif (Kuncheva et Plumpton, 2008).

# 3 Estimation des densités conditionnelles

Cette section présente comment sont estimées les probabilités conditionnelles $P(X_i|C_k)$ qui doivent être placées à l'entrée du modèle graphique présenté au cours de la section précédente. Les méthodes d'estimation sont présentées ci-dessous brièvement n'étant pas la contribution majeure de cet article.

Pour nos expérimentations trois estimations sont utilisées (voir Figure 2) pour calculer $P(X_i|C_k)$ pour chaque variable numérique explicative $i$ et pour chaque classe $k$ : (i) une méthode de discrétisation à deux niveaux basée sur des statistiques d'ordre tel que décrite dans (Salperwyck et Lemaire, 2013) (ii) une méthode de discrétisation à deux niveaux « cPiD » qui est une version modifiée de la méthode PiD (Gama et Pinto, 2006) (iii) une approximation gaussienne. L'approche peut être la même dans le cas des variables catégorielles (non détaillée

dans cet article) en mettant dans le premier niveau l'approche count-min sketch (Cormode et Muthukrishnan, 2005) et dans le deuxième niveau la méthode de groupage MODL. Le lecteur intéressé pourra trouver un état de l'art et davantage de détails sur les techniques d'estimation de densités conditionnelles dans le chapitre 3 de (Salperwyck, 2012).

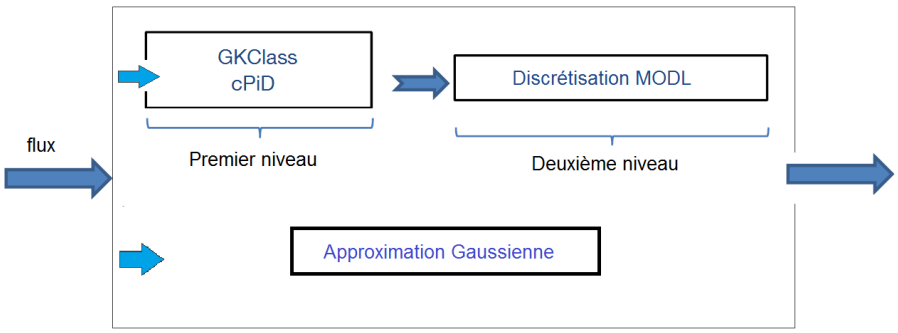


FIG. 2 – *Méthode à deux niveaux.*

L'estimation de notre méthode à deux niveaux est basée sur un premier niveau qui produit les quantiles de chaque variable explicative $i$ du flux de données. Chaque quantile, $q$, est un tuple qui contient : $< v_q^i, g_q^i, (g_{qj}^i)_{j=1,...K} >$ où pour chaque variable explicative (i) $v_q^i$ est une valeur observée (ii) $g_q^i$ correspond au nombre de valeurs entre $v_{q-1}^i$ et $v_q^i$ (iii) $(g_{qj}^i)$ est, pour les $K$ classes du problème de classification supervisé, le nombre d'éléments dans $q$ appartenant à la classe $j$. Le second niveau est un algorithme hors-ligne qui est appliqué sur les quantiles. Dans cet article le nombre de tuples utilisé est égal à 100 correspondant donc à une estimation des centiles. Le réglage de la valeur du nombre de tuples est discuté dans (Salperwyck, 2012). cPid et GkClass, décrits ci-dessous, sont deux méthodes permettant d'obtenir les quantiles.

## 3.1 cPid

(Gama et Pinto, 2006) ont proposé une méthode de discrétisation à deux niveaux pour une variable numérique. Le premier niveau est un mélange entre les méthodes « Equal Width » et « Equal Frequency » (détaillé dans (Gama et Pinto, 2006) p. 663). Le premier niveau est actualisé de manière incrémentale et nécessite d'avoir plus d'intervalles que le second niveau. Le second niveau utilise l'information contenue dans le premier niveau pour construire une deuxième discrétisation. De nombreuses méthodes peuvent être utilisées pour le second niveau : Equal Width, Equal Frequency, Entropy, Kmoyenne... L'avantage de PiD est d'avoir un premier niveau très rapide et purement incrémental. Dans cPiD nous apportons une modification afin d'avoir une mémoire constante. L'augmentation de la consommation mémoire de la méthode PiD est due à la création de nouveaux intervalles. En effet si un intervalle devient trop peuplé alors il est divisé en deux intervalles contenant chacun la moitié des individus. Notre modification consiste, suite à la division d'un intervalle, à fusionner les deux intervalles

consécutifs dont la somme des comptes est la plus faible. Ainsi le nombre d'intervalles stockés reste toujours le même. Aucune comparaison n'est réalisée dans cet article entre PiD et cPiD car notre intérêt se porte sur une utilisation à mémoire constante des méthodes.

### 3.2 GKClass

Cet algorithme proposé dans (Greenwald et Khanna, 2001) est un algorithme destiné à calculer les quantiles en utilisant une mémoire de $O(\frac{1}{\epsilon}log(\epsilon N))$ dans le pire cas, avec $N$ le nombre d'éléments observés et $\epsilon$ l'erreur souhaitée. Cette méthode ne requiert pas de connaitre au préalable la taille $N$ du flux et est insensible à l'ordre d'arrivée des exemples. Un des avantages de cette méthode est que, selon le besoin, on peut soit définir l'erreur maximale souhaitée, soit la mémoire maximale à utiliser. Dans le première cas l'erreur maximale est fixée et le résumé consomme autant de mémoire que nécessaire pour que l'erreur maximale ne soit pas dépassée. Dans le deuxième cas on fixe une quantité de mémoire maximale et on l'utilise au mieux pour minimiser l'erreur. Nous avons adapté cet algorithme pour qu'il stocke directement les comptes par classe. Le second niveau utilise la méthode de discrétisation supervisée MODL (Boullé, 2006a).

### 3.3 Approximation Gaussienne (AG)

Cette méthode suppose que la distribution des données se rapproche d'une loi normale que l'on va chercher à approximer. Pour cela il suffit de ne conserver que les trois valeurs par classe : la moyenne $\mu$, l'écart type (ou la variance $\sigma$) et le nombre $n$ d'éléments qui définissent cette gaussienne. Le maintien de ces trois valeurs peut se faire de manière incrémentale et donc cette méthode est parfaitement adaptée à une utilisation en-ligne ou sur les flux et ne comporte qu'un seul niveau. Elle sera utilisée comme référence dans le cadre de nos expérimentations du fait que les bases de données du « Large Scale Learning » ont été générées à l'aide de générateurs gaussiens. L'indicateur d'évaluation retenu est la précision (« accuracy ») c'est à dire $\frac{TP+TN}{TP+TN+FP+FN}$ où $TP, TN, FP$ et $FN$ sont respectivement le nombre de vrai-positifs, vrai-négatifs, faux-positifs et faux-négatifs.

## 4 Expérimentations

### 4.1 Protocole

Les expérimentations pour ce classifieur sont réalisées sur les bases du challenge Large Scale Learning [1] [2], proposé par le réseau d'excellence PASCAL. Toutes ces bases contiennent 500 000 exemples étiquetés, ce qui peut est considéré comme suffisant pour évaluer un algorithme en-ligne. Nous utilisons les bases alpha, beta, delta et gamma, qui possèdent 500 variables numériques et les bases epsilon et zeta, qui en contiennent 2000. Les jeux de données sont séparés en ensemble de test/apprentissage. Les 100 000 premiers exemples sont pris comme ensemble de test et les autres comme ensemble d'apprentissage.

1. http://largescale.ml.tu-berlin.de/about/
2. http://jmlr.csail.mit.edu/papers/topic/large_scale_learning.html

## 4.2 Résultats

Pour la première partie des expérimentations, un classifieur Naïf de Bayes (NB) sans pondération est utilisé. Il utilise les estimations de densités conditionnelles aux classes issues des méthodes décrites dans la section précédente. Les résultats sont présentés dans le tableau 1 et montrent que l'estimation des probabilités conditionnelles est précise pour les trois méthodes du fait que le classifieur naïf de Bayes obtient de bons résultats avec chacune d'entre elles. Malgré le fait que les données du challenge aient été générées à l'aide d'un générateur gaussien les deux autres méthodes, cPid et GKClass, obtiennent des résultats similaires à la méthode basée sur l'approximation Gaussienne (AG). Entre les deux méthodes cPid et GKClass, le résumé GKClass, apporte des garanties précision / mémoire utilisée, a des résultats comparable à cPid et ne fait pas d'hypothèse sur la nature de la distribution des données. De ce fait il a été choisi pour la suite des expérimentations.

| | Alpha | | | Beta | | | Delta | | |
|---|---|---|---|---|---|---|---|---|---|
| # examples | 40 000 | 100 000 | 380 000 | 40 000 | 100 000 | 380 000 | 40 000 | 100 000 | 380 000 |
| GKClass | 54,40 | 54,49 | 54,56 | 49,79 | 51,05 | 51,23 | 80,56 | 82,22 | 83,47 |
| cPiD | 54,36 | 54,52 | 54,57 | 49,79 | 51,09 | 51,12 | 80,66 | 82,39 | 83,77 |
| AG | 54,62 | 54,67 | 54,67 | 51,21 | 51,50 | 51,31 | 84,58 | 85,10 | 85,08 |
| | Gamma | | | Epsilon | | | Zeta | | |
| # examples | 40 000 | 100 000 | 380 000 | 40 000 | 100 000 | 380 000 | 40 000 | 100 000 | 380 000 |
| GKClass | 92,63 | 93,51 | 94,23 | 70,48 | 70,37 | 70,43 | 78,35 | 78,63 | 78,48 |
| cPiD | 92,93 | 93,83 | 94,41 | 70,52 | 70,38 | 70,36 | 78,50 | 78,48 | 78,42 |
| AG | 95,09 | 95,10 | 95,16 | 70,66 | 70,57 | 70,43 | 78,96 | 78,77 | 78,52 |

TAB. 1 – *Précision du classifieur naïf de Bayes sans pondération utilisant GKClass, cPiD et l'approximation gaussienne pour calculer les probabilités conditionnelles.*

Grâce aux résultats présentés dans le tableau 1 nous savons que l'estimation des densités conditionnelles est précise. Par conséquent nous pouvons à présent évaluer le comportement de notre classifieur Naïf de Bayes Pondéré (NBP). Les résultats comparent quatre classifieurs :

1. un classifieur Naïf de Bayes (NB) entraîné hors-ligne et utilisant la discrétisation MODL (Boullé, 2006a) et toutes les données chargées en mémoire ;
2. un classifieur Naïf de Bayes Moyenné (NBM) entraîné hors-ligne et utilisant la discrétisation MODL (Boullé, 2006a) et l'algorithme décrit dans (Boullé, 2006b) pour calculer les poids des variables explicatives - cette méthode est l'une des meilleures de l'état de l'art (Guyon et al., 2009) ;
3. un classifieur Naïf de Bayes (NB) entraîné en-ligne et utilisant la méthode de discrétisation à deux niveaux qui utilise GKClass au niveau 1 et la discrétisation MODL au niveau 2 ;
4. notre classifieur Naïf de Bayes Pondéré (NBP) entraîné en-ligne et dont les poids sont estimés à l'aide de notre méthode basée sur un modèle graphique. La méthode de discrétisation utilise GKClass comme niveau 1 et la discrétisation MODL comme niveau 2.

La table 2 montre que les résultats obtenus par notre NBP en-ligne sont encourageants : il est meilleur que le NB en-ligne sauf pour les jeux de données Gamma et Delta. Sa performance est proche du NBM hors-ligne qui est sans doute l'un des meilleurs classifieurs bayésien qui peut être obtenu sur ce jeu de données. Mais la version hors-ligne nécessite d'avoir toutes les données en mémoire et de les lire plusieurs fois, ce qui n'est pas possible dans le cadre d'un apprentissage sur flux de données. Notre approche pour construire le classifieur NBP n'utilise que très peu de mémoire grâce à notre résumé à deux niveaux pour estimer les probabilités conditionnelles. De plus elle est entièrement incrémentale grâce au modèle graphique et à la descente de gradient pour estimer les poids. Ces premiers résultats sont encourageants et semble indiquer que l'on pourrait avoir les mêmes résultats que la version hors-ligne NBM avec notre classifieur en-ligne NBP si plus d'exemples étaient disponibles. Des travaux futurs permettront de confirmer ou d'infirmer cette hypothèse.

| | Alpha | | | Beta | | | Delta | | |
|---|---|---|---|---|---|---|---|---|---|
| #examples | 40000 | 100000 | 380000 | 40000 | 100000 | 380000 | 40000 | 100000 | 380000 |
| hors-ligne NB (1) | 54,60 | 54,61 | 54,61 | 49,79 | 51,36 | 51,19 | 80,78 | 82,44 | 83,91 |
| hors-ligne NBM (2) | 66,13 | 67,30 | 68,77 | 49,79 | 51,39 | 53,24 | 80,80 | 82,46 | 83,91 |
| en-ligne NB (3) | 54,40 | 54,49 | 54,56 | 49,79 | 51,05 | 51,23 | 80,56 | 82,22 | 83,47 |
| en-ligne NBP (4 ) | 64,03 | 66,40 | 67,61 | 49,79 | 49,79 | 52,20 | 75,35 | 77,74 | 79,53 |
| | Gamma | | | Epsilon | | | Zeta | | |
| #examples | 40000 | 100000 | 380000 | 40000 | 100000 | 380000 | 40000 | 100000 | 380000 |
| hors-ligne NB (1) | 92,95 | 93,99 | 94,63 | 70,35 | 71,04 | 70,58 | 78,39 | 78,34 | 78,26 |
| hors-ligne NBM (2) | 92,95 | 94,00 | 94,64 | 84,36 | 85,34 | 86,01 | 88,67 | 89,51 | 90,43 |
| en-ligne NB (3) | 92,63 | 93,51 | 94,23 | 70,48 | 70,37 | 70,43 | 78,35 | 78,63 | 78,48 |
| en-ligne NBP (4) | 90,25 | 91,05 | 91,76 | 69,25 | 74,76 | 79,95 | 73,82 | 79,91 | 84,52 |

TAB. 2 – *Précision des différents classifieurs naïf de Bayes étudiés.*

# 5 Conclusion

Les résultats de notre version en-ligne du classifieur naïf de Bayes sont prometteurs. Ses performances sont meilleures que celles de la version en-ligne non pondérées et proche de la version pondérée hors-ligne. Cependant nos résultats pourraient encore être améliorés dans de prochains travaux. Notre première piste d'amélioration serait d'utiliser les résumés GKClass comme des « mini-batch » (Cotter et al., 2011) et de réaliser plusieurs itérations pour accélérer la descente de gradient. Notre seconde proposition serait d'avoir un pas adaptatif pour la descente de gradient : rapide au début de l'apprentissage puis plus lent par la suite, ou de prendre en compte le taux d'erreur comme dans (Kuncheva et Plumpton, 2008).

Le pas d'apprentissage pourrait aussi être contrôlé par une méthode de détection de changement de concept afin de ré-augmenter le pas dès qu'une détection a lieu et donc de ré-apprendre plus rapidement. Il faudrait de plus mettre à jour les résumés suite à la détection afin d'avoir des estimations correspondant au nouveau concept. De nombreuses méthodes de détection existent mais afin de rester cohérent avec notre approche la grille MODL (Salperwyck et al., 2013) pourrait être utilisée pour détecter les changements de distribution.

# Références

Amari, S., N. Murata, K. R. Muller, M. Finke, et H. H. Yang (1997). Asymptotic statistical theory of overtraining and cross-validation. *IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council 8*(5), 985–96.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. New York, USA : Oxford University Press, Inc.

Boullé, M. (2006a). MODL : A Bayes optimal discretization method for continuous attributes. *Machine Learning 65*(1), 131–165.

Boullé, M. (2006b). Regularization and Averaging of the Selective Naive Bayes classifier. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 1680–1688.

Breiman, L. (1996). Bagging predictors. *Machine learning 24*(2), 123–140.

Cormode, G. et S. Muthukrishnan (2005). An improved data stream summary : the count-min sketch and its applications. *Journal of Algorithms 55*(1), 58–75.

Cotter, A., O. Shamir, N. Srebro, et K. Sridharan (2011). Better mini-batch algorithms via accelerated gradient methods. *CoRR*.

Gama, J. (2010). *Knowledge Discovery from Data Streams*. Chapman and Hall/CRC Press.

Gama, J. et C. Pinto (2006). Discretization from data streams : applications to histograms and data mining. In *Proceedings of the 2006 ACM symposium on Applied computing*, pp. 662–667.

Greenwald, M. et S. Khanna (2001). Space-efficient online computation of quantile summaries. *ACM SIGMOD Record 30*(2), 58–66.

Guigourès, R. et M. Boullé (2011). Optimisation directe des poids de modèles dans un prédicteur Bayésien naif moyenné. In *Extraction et gestion des connaissances EGC'2011*, pp. 77–82.

Guyon, I., V. Lemaire, M. Boullé, G. Dror, et D. Vogel (2009). Analysis of the KDD Cup 2009 : Fast Scoring on a Large Orange Customer Database. *JMLR : Workshop and Conference Proceedings 7*, 1–22.

Hand, D. J. et K. Yu (2001). Idiot's Bayes ? Not So Stupid After All ? *International Statistical Review 69*(3), 385–398.

Hoeting, J., D. Madigan, et A. Raftery (1999). Bayesian model averaging : a tutorial. *Statistical science 14*(4), 382–417.

Koller, D. et M. Sahami (1996). Toward Optimal Feature Selection. *International Conference on Machine Learning 1996*(May), 284–292.

Kuncheva, L. I. et C. O. Plumpton (2008). Adaptive Learning Rate for Online Linear Discriminant Classifiers. In *Proceedings of the 2008 Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pp. 510–519. Springer-Verlag.

Langley, P., W. Iba, et K. Thompson (1992). An analysis of Bayesian classifiers. In *Proceedings of the National Conference on Artificial Intelligence*, Number 415, pp. 223–228.

Langley, P. et S. Sage (1994). Induction of Selective Bayesian Classifiers. In R. L. D. M. Poole et D (Eds.), *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*,

pp. 399–406. Morgan Kaufmann.

Lecun, Y., L. Bottou, G. B. Orr, et K. R. Müller (1998). Efficient BackProp. In G. Orr et K. Müller (Eds.), *Lecture Notes in Computer Science*, Volume 1524 of *Lecture Notes in Computer Science*, pp. 5–50. Springer Verlag.

Prechelt, L. (1997). Early Stopping - but when ? In *Neural Networks : Tricks of the Trade, volume 1524 of LNCS, chapter 2*, pp. 55–69. Springer-Verlag.

Salperwyck, C. (2012). *Apprentissage incrémental en ligne sur flux de données*. Ph. D. thesis, University of Lille.

Salperwyck, C., M. Boullé, et V. Lemaire (2013). Grille bivariée pour la détection de changement dans un flux étiqueté. In *EGC*, pp. 389–400.

Salperwyck, C. et V. Lemaire (2011). Classification incrémentale supervisée : un panel introductif. *Revue des Nouvelles Technologies de l'Information, Numéro spécial sur l'apprentissage et la fouille de données A5*, 121–148.

Salperwyck, C. et V. Lemaire (2013). A two layers incremental discretization based on order statistics. *Statistical Models for Data Analysis*, 315–323.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics*. Wiley.

# Annexe - Calcul de la dérivée de la fonction de coût

Cette annexe explicite le calcul de la dérivée de la fonction de coût pour le classifieur bayésien moyenné avec optimisation des poids par descente de gradient.

Le modèle graphique permet d'avoir directement en sortie la valeur des $P(C_k|X)$. Le but étant de maximiser la vraisemblance il suffit alors de minimiser le log vraisemblance. On décompose tout d'abord la partie softmax en considérant que chaque sortie de la fonction $softmax$, avant la phase de normalisation, peut être vue comme étant la succession de deux étapes : une phase d'activation suivi d'une fonction recevant la valeur de l'activation. Ici la fonction d'activation peut être vue comme étant $O_k = f(H_k) = exp(H_k)$ et la sortie de la partie softmax de notre modèle graphique est : $P_k = \frac{O_k}{\sum_{j=1}^{K} O_j}$. La dérivée de la fonction d'activation est :

$$\frac{\partial O_k}{\partial H_k} = f'(H_k) = exp(H_k) = O_k \tag{6}$$

La fonction de coût étant le log vraisemblance, il faut considérer deux cas : (i) soit on désire apprendre pour la valeur 1 ; (ii) soit on désire apprendre la valeur 0. On pose pour la suite :

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{\partial C}{\partial P_k}\frac{\partial P_k}{\partial O_k}\frac{\partial O_k}{\partial H_k} \tag{7}$$

**Dans le cas où l'on désire obtenir la valeur 1** en remplaçant (6) dans (7) :

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{\partial C}{\partial P_k}\frac{\partial P_k}{\partial O_k}\frac{\partial O_k}{\partial H_k} = \frac{-1}{P_k}\frac{\partial P_k}{\partial O_k}O_k \tag{8}$$

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{-1}{P_k}\left[\sum_{l=1,l\neq k}^{K}\left(\frac{O_l}{(\sum_{j=1}^{K} O_j)^2}\right)\right]O_k = \frac{-1}{P_k}\left[\frac{(\sum_{j=1}^{K} O_j) - O_k}{(\sum_{j=1}^{K} O_j)^2}\right]O_k \tag{9}$$

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{-1}{P_k}\left[\frac{(\sum_{j=1}^{K} O_j) - O_k}{(\sum_{j=1}^{K} O_j)}\right]\frac{O_k}{(\sum_{j=1}^{K} O_j)} \tag{10}$$

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{-1}{P_k}\left[1 - \frac{O_k}{(\sum_{j=1}^{K} O_j)}\right]\frac{O_k}{(\sum_{j=1}^{K} O_j)} \tag{11}$$

D'où on obtient finalement :

$$\frac{\partial \text{Coût}}{\partial H_k} = \frac{-1}{P_k}[1 - P_k]P_k = P_k - 1 \tag{12}$$

**Dans le cas où l'on désire obtenir la valeur 0** l'effet de l'erreur est uniquement transmis par la normalisation issue de la fonction $softmax$ (la dérivée de la fonction d'erreur vis-à-vis d'une unité de sortie pour laquelle la sortie désirée est 0 est nulle). On obtient à l'aide de calculs similaires :

$$\frac{\partial \text{Coût}}{\partial H_k} = P_k \tag{13}$$

**On conclut alors que :**

$$\frac{\partial \text{Coût}}{\partial H_k} = P_k - T_k, \forall k \tag{14}$$

où $T_k$ désigne les valeurs de probabilité désirées (target) et $P_k$ la probabilité estimée par le modèle graphique.

Il ne reste ensuite plus qu'à inclure la partie couche linéaire de notre modèle graphique pour avoir les dérivées partielles $\frac{\partial \text{Coût}}{\partial w_{ik}}$.

# Summary

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with naive independence assumption. The explanatory variables ($X_i$) are assumed to be independent from the target variable ($C$). Despite this strong assumption this classifier has proved to be very effective on many real applications and is often used on data stream for supervised classification. The naive Bayes classifier simply relies on the estimation of the univariate conditional probabilities $P(X_i|C)$. This estimation can be provided on a data stream using a "supervised quantiles summary". The literature shows that the naive Bayes classifier can be improved (i) using a variable selection method (ii) weighting the explanatory variables. Most of these methods are related to off-line learning and need to store all the data in memory and/or require reading more than once each example. Therefore they cannot be used on data stream. This paper presents a new method based on a graphical model which computes the weights on the input variables using a stochastic estimation. The method is incremental and produces a Weighted Naive Bayes classifier for data stream. This method will be compared to classical naive Bayes classifier on the Large Scale Learning challenge datasets.