



**GROUPE
BPCE**



ENSAE 3ème année
Stage de fin d'études
Année scolaire 2016-2017

Prévision du défaut sur le périmètre de TPE

XU KAI

Maître de stage : Smaïl IBBOU
du 15 mai au 14 novembre

Remerciements

En premier lieu, j'adresse particulièrement mon remerciement à M.Samîl IBBOU, mon maître de stage au sein de BPCE, pour m'avoir permis d'effectuer mon stage de fin d'étude dans de bonnes conditions, et pour le sujet passionnant sur lequel j'ai pu travailler.

Je tiens également à remercier M.Quentin JAMMES pour le temps qu'il m'a consacré et son suivi tout au long de mon stage. Merci pour les connaissances qu'il m'a transmises ainsi que ses remarques précieuses qui m'ont permis d'approfondir ma compréhension du sujet.

Enfin je remercie tous les collègues de mon équipe pour leur patience et leur soutien tout au long de mon stage.

Présentation du groupe BPCE

Le Groupe BPCE s'appuie sur ses deux principaux réseaux, Banque Populaire et Caisse d'épargne, ainsi que sur ses filiales pour mener à bien toutes les métiers de la banque et de l'assurance. Il est au service de 32 millions de clients avec ses 108 000 collaborateurs. Grâce à ses filiales, le groupe offre à ses clients un service complet : solutions d'épargne et d'investissement, service de placement, de trésorerie, de financement, d'assurance et gestion d'actifs.

Il est le deuxième groupe bancaire en France issu de la fusion en 2009 qui est mise en œuvre en réponse à la crise des subprimes. Les deux réseaux coopératifs gardent leur enseignes et leur indépendance mais coordonne et mettre en commun leur politique commune. En ce cas, des services de back-office s'intègrent tels que la direction des risques qui comprend l'unité Validation du pôle Analyses consolidées et Modèles dans laquelle j'ai effectué mon stage de fin d'études.

Table des matières

1	Introduction	1
1.1	Contexte	1
1.2	Présentation du sujet	1
2	Traitemen t des données	3
2.1	Alimentation de la base des données	3
2.2	Traitemen t des variables	3
3	Apprentissage semi-supervisé	4
3.1	Carte Kohonen (SOM)	4
3.2	Détection de structure topologique	7
3.2.1	Enveloppe convexe	7
3.2.2	Algorithm e t-SNE	8
3.3	Carte Kohonen supervisée (SSOM)	9
3.3.1	Sous-échantillonnage	9
3.3.2	Modèle de carte Kohonen supervisée	9
3.4	Évaluation des performances	12
3.4.1	Indice de performance	12
3.4.2	Comparaison des performances	14
4	Apprentissage supervisé	15
4.1	Évaluation des performances	15
4.2	Calibrage des paramètres	16
4.3	Erreur de généralisation	17
4.4	Modèle d'apprentissage	17
4.4.1	Modèle de référence : Arbre de décision	17
4.4.2	Régression logistique pénalisée	19
4.4.3	Forêts aléatoires	20
4.4.4	Gradient Boosting	21
4.5	Sélection de variables	22
4.5.1	Filter	22

4.5.2	Embedded	23
4.5.3	Variables plus importantes	24
4.6	Résultat des modèles modifiés	25
5	Apprentissage hybride	28
5.1	Construction des variables synthétiques	28
5.1.1	Hétérogénéité	28
5.1.2	Classifieur local et distance signée	29
5.2	Modèle de prévision	30
5.2.1	Procédure de modélisation	30
5.2.2	Résultat empirique	30
6	Validation croisée	32
6.1	Validation croisée Monte Carlos	32
6.2	Résultat empirique	32
7	Conclusion et perspectives	35

Liste des tableaux

3.1	Définition : Matrice de confusion	12
3.2	Base d'apprentissage	14
3.3	Base de test	14
4.1	Sélection des variables	25

Table des figures

3.1	Carte Kohonen	4
3.2	Base d'apprentissage	6
3.3	Base de test	6
3.4	Test de séparabilité linéaire	8
3.5	Base de Iris	9
3.6	Base d'apprentissage TPE	9
3.7	Carte Kohonen supervisé	10
3.8	Base d'apprentissage	11
3.9	Base de test	11
3.10	Matrice de confusion : Base de test	12
4.1	Courbe ROC	18
4.2	Courbe Gini	18
4.3	Courbe ROC	19
4.4	Courbe Gini	19
4.5	Courbe ROC	20
4.6	Courbe Gini	20
4.7	Courbe ROC	22
4.8	Courbe Gini	22
4.9	Base d'apprentissage	26
4.10	Base de test	26
4.11	Base d'apprentissage	27
4.12	Base de test	27
5.1	Hétérogénéité géométrique	29
5.2	Courbe ROC	30
5.3	Courbe Gini	30
5.4	Base d'apprentissage	31
5.5	Base de test	31
6.1	Base d'apprentissage	33
6.2	Base de test	33

6.3	Base d'apprentissage	34
6.4	Base de test	34

Chapitre 1

Introduction

1.1 Contexte

Les TPEs (très petites entreprises) constituent un pilier important de l'économie française en occupant environ 20% de l'emploi dans le secteur concurrentiel, et 20% de la valeur ajoutée créée par l'ensemble des entreprises. Actuellement, elles représentent un contingent important du Corporate. Cependant, les modèles actuels de prévision de défaut sur la classe d'actifs Corporate ne peuvent pas bien capter ses comportements spécifiques.

L'apprentissage statistique a été élaboré pour décrire au mieux un phénomène à partir des données volumineuses. Pourvu que la population modélisée soit suffisamment grande pour être représentative et stable, l'apprentissage statistique sera un moyen idéal à mettre en place afin de décrire au mieux les caractéristiques du défaut de TPE.

En effectuant une refonte des données des réseaux Banques populaires, Caisse d'épargne et Natixis, le groupe BPCE a accumulé des données volumineuses de TPE à la fois comportementales et financières. Ainsi, les données de TPE du groupe possèdent les caractéristiques de quantité volumineuse et de grande dimension, ce qui permet d'en faire l'objet potentiel de l'apprentissage statistique.

1.2 Présentation du sujet

La prévision du défaut des clients est l'un des problèmes fondamentaux au cœur de l'attention des banques. L'approche traditionnel consiste à chercher les variables, principalement comptables, qui différencient au mieux les clients défaillants et les clients sains à partir de l'analyse financière. Cependant, un gap de performances est observé sur la population de TPE sous les modèles existants.

L'objet de notre travail est donc d'appliquer la technique d'apprentissage statistique dans les

TPEs afin d'améliorer la prévision du défaut associée à leur comportement spécifique. En ce cas, une meilleure allocation des crédits sera réalisée par une meilleure identification des risques.

Il est possible que les TPEs susceptibles de passer en défaut se distinguent de leurs homologues sains par la structure topologique de leurs caractéristiques. Nous souhaitons pouvoir représenter nos contrats en conservant la topologie d'entrée, et sans être handicapés par la présence de nombreuses données manquantes. c'est pourquoi des méthodes neuronales de carte Kohonen qui permet de faire la classification en préservant la topologie des individus sont implémentées. Basé sur les résultats obtenus de la carte Kohonen, les algorithmes d'apprentissages supervisés sont proposés dans le chapitre 4 pour ce type de tâches. Ensuite, une méthode hybride est mise en place dans le chapitre 5 en fusionnant nos cartes de Kohonen et les méthodes supervisés afin de construire un cadre assez performant et raffiné. Enfin, les validations croisées sont effectuées dans le chapitre 6 sur les modèles modifiés en vue de tester la robustesse de nos modèles. Il s'agira d'un côté d'explorer l'espace des entrées et de l'autre d'en développer un cadre prédictif sur les clients TPEs.

Chapitre 2

Traitement des données

2.1 Alimentation de la base des données

Les TPE correspondent à un sous-ensemble des portefeuilles actuels de la Banque de Détail (Retail) Professionnel et Entreprise (Corporate). La critère d'être TPE est que la chiffre d'affaires compris entre 1,5 et 5M€ ou montant total des engagements accordés plus que 1 M€. L'objet sur lequel nous travaillons est les comptes à vue avec bilan dans la population TPE.

L'organisation du groupe BPCE rend l'alimentation de la base TPE. La base totale est un échantillon occupant environ 1/6 des sociétés qualifiées dans 4 cohortes de Juin 2012 à Décembre 2013 à pas semestriel. La population atypique (associations, entreprises étrangères, SCI, ...) est supprimée de la base.

Cette base des 18902 effectifs dont environ 3% de clients sont en défaut est divisée en 2 parties : une base d'apprentissage comportant 70% des effectifs qui sert à construire et estimer les modèles différents ; une base de test comportant 30% des effectifs utilisée pour tester la stabilité des modèles.

2.2 Traitement des variables

Pour tenter de faire la prévision, nous disposons de variables de 4 catégories : compte, donnée externe, ratios financiers et signalétique. Au total, nous disposons de 202 variables explicatives parmi lesquelles 290 variables quantitatives et 12 variables qualitatives.

Les variables qualitatives sont mises sous forme de tableau disjonctif pour se transformer en nouvelles variables binaires, ce qui produit environ 56 nouvelle variables. Et les variables quantitatives se normalisent pour être centrées réduites. Du coup, nous travaillons sur 246 variables explicatives générées. En outre, les données manquantes d'une variable seront remplacées par la valeur médiane des données réellement observées si besoin.

Chapitre 3

Apprentissage semi-supervisé

3.1 Carte Kohonen (SOM)

La carte Kohonen est un modèle de réseau neurone artificiel qui nous permet d'apprendre la structure topologique des entrées de manière non supervisée. Elle est formée d'une couche de neurones disposé en grille ou ficelle pour regrouper les entrées. Chaque neurone est caractérisée par un vecteur poids de même dimension que les entrées qui détermine sa position dans l'espace d'entrée. L'apprentissage se déroule d'une manière de la compétition parmi les neurones pour la représentation des entrées qui lui sont présentées. Une partition de Voronoi sera implémenté en vue de découper l'espace d'entrée à partir des vecteurs poids appartenant aux neurones.

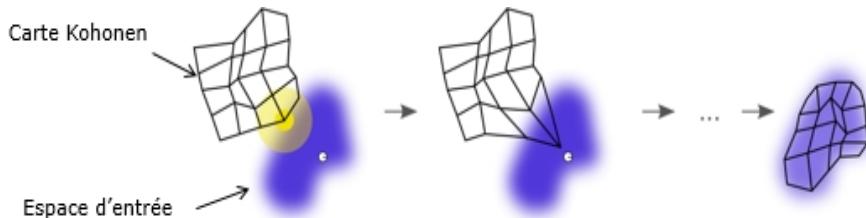


FIGURE 3.1 – Carte Kohonen

Contrairement aux méthodes factorielles telles que l'analyse en composantes principales ou l'analyse de correspondances multiples qui réalisent des projections sur des sous-espaces linéaires de l'espace des entrées, la carte Kohonen capture plutôt la structures topologique des sous-groupes au lieu de l'emplacement précis des individus. D'ailleurs, elle est plus robustes face aux données manquantes [1].

L'algorithme de carte Kohonen [1] :

- 1) Initialisation aléatoire des vecteurs poids \vec{w}
- 2) A instant t , une observation $x(t + 1) \subset R^d$ est choisie aléatoirement et présentée au réseau
- 3) La neurone gagnante est définie par :

$$i_0 = \operatorname{arg\inf}_i \|x(t+1) - w_i(t)\|^2$$

pour la distance euclidienne 4) Les vecteurs poids de neurone gagnante $w_{i_0}^t$ et ses voisins sont mis à jour par :

$$\begin{cases} w_i^{t+1} = w_i^t + \epsilon(t)(x(t+1) - w_i^t) & \forall i \in V_{r_t}(i_0) \\ w_i^{t+1} = w_i^t & \forall i \notin V_{r_t}(i_0) \end{cases}$$

où $V_{r_t}(i_0)$ est le voisinage de rayon $r(t)$ autour de la neurone gagnante i_0 mesuré par la distance de Manhattan $d(\cdot, \cdot)$

$$1_{i \in V_{r_t}(i_0)} = \begin{cases} 1 & \text{if } d(i, i_0) \leq r_t \\ 0 & \text{if } d(i, i_0) > r_t \end{cases}$$

et où $\epsilon(t)$ est le taux d'apprentissage qui satisfait les conditions de Robbins–Monro :

$$\sum_{t=0}^{\infty} \epsilon_t = \infty \quad \text{and} \quad \sum_{t=0}^{\infty} \epsilon_t^2 < \infty$$

Cependant, il se rencontrera deux faiblesses pour cet algorithme [5] : 1) la convergence prématuée ; 2) la mauvaise initialisation. La convergence prématuée est un phénomène dans lequel le réseau de neurones converge si tôt que le résultat se trouve dans un optimum local. Et une mauvaise initialisation pourrait conduire à un réseau moins performant. Pour éviter les deux faiblesses, la quatrième étape du algorithme est modifiée :

$$w_i^{t+1} = w_i^t + \epsilon'(t) e^{-\frac{-\|(w_{(i_0)} - w_i^t)\|^2}{r(t)^2}} (x(t+1) - w_i^t)$$

De plus, une décroissance exponentielle aux hyperparamètres est effectuée en fonction du temps t et du nombre d'itérations T borné :

$$\epsilon'(t) = \epsilon(0) \left(\frac{\epsilon(T)}{\epsilon(0)} \right)^{\frac{T}{t}} \quad r(t) = r(0) \left(\frac{r(T)}{r(0)} \right)^{\frac{T}{t}}$$

Il est évident que la nouvelle mise à jour satisfait aussi la condition de Robbins–Monro :

$$\begin{aligned} \sum_{t=0}^{\infty} \epsilon'(t) e^{-\frac{-\|(w_{(i_0)} - w_i^t)\|^2}{r(t)^2}} &= \sum_{t=0}^T \epsilon'(t) e^{-\frac{-\|(w_{(i_0)} - w_i^t)\|^2}{r(t)^2}} + \sum_{t=T+1}^{\infty} \epsilon(t) = \infty \\ \sum_{t=0}^{\infty} \epsilon'(t)^2 e^{-\frac{-2\|(w_{(i_0)} - w_i^t)\|^2}{r(t)^2}} &< T\epsilon(0)^2 + \sum_{t=T+1}^{\infty} \epsilon(t)^2 < \infty \end{aligned}$$

Après le calibrage de paramètre, nous choisissons une 5×5 carte munie des hyperparamètres : $\epsilon(T) = 0.04, \epsilon(0) = 0.01, r(0) = 2.99, r(T) = 0.65$ pour un nombre d'itérations $T=13231$. Nous entendons faire une discrimination en utilisant le regroupement généré par notre carte de Kohonen entraînée. Chaque neurone est marqué d'une classe dont sont muni la plupart des observations représentées par cette neurone.

Le déséquilibre de la répartition du nombre d'observations et de la répartition du défaut est observé dans les résultat de deux bases de données. Conformément à notre attente, la neurone avec des observations le plus nombreuses et ses voisins possèdent les taux de défaut plus bas que ceux des neurone avec des observations moins nombreuses. Cependant, la différence n'est pas si évident que toutes les neurones sont marquées de la classe saine, ce qui est contre notre attente.

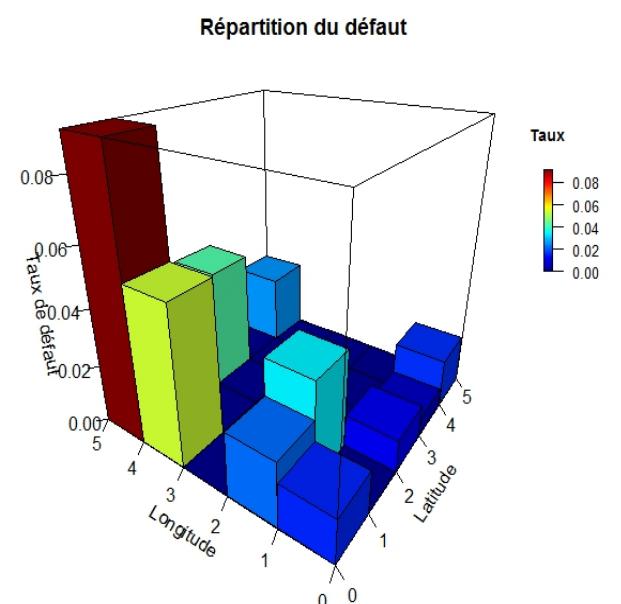
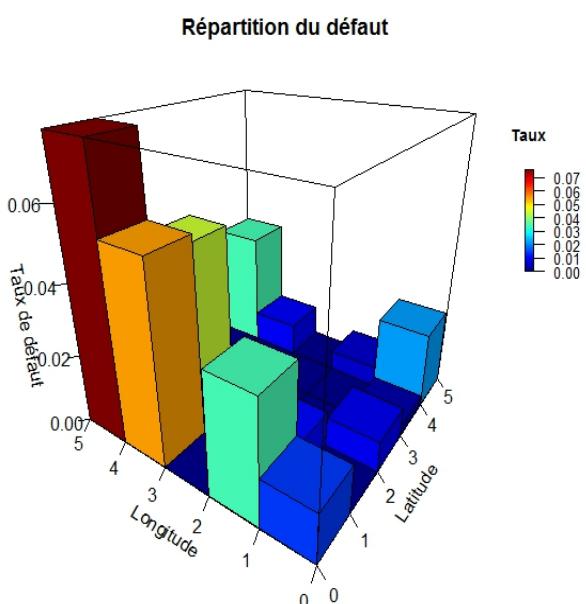
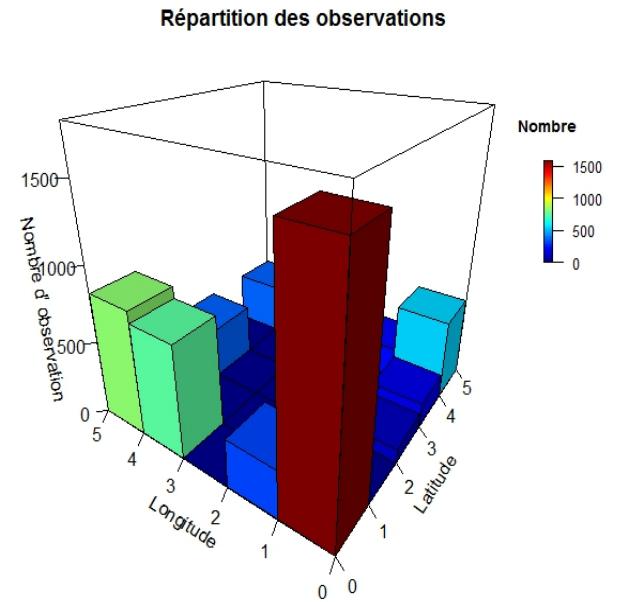
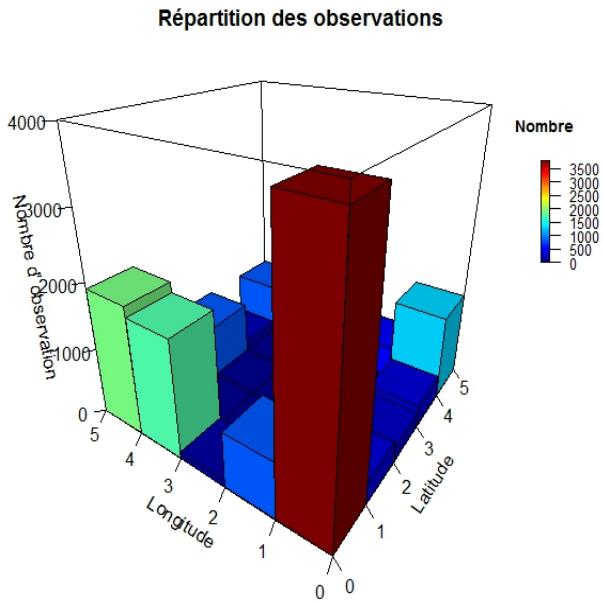


FIGURE 3.2 – Base d'apprentissage

FIGURE 3.3 – Base de test

En fait, ce que nous avons implémenté provient d'une technique de la propagation d'étiquette qui appartient à l'apprentissage semi-supervisé. Selon Chapelle et al [3], la réussite de l'apprentissage semi-supervisé réside dans trois hypothèse : 1) les entrées regroupées dans le même groupe tendent à être de même classe ; 2) l'hyperplan qui sépare les entrées de classes différentes devrait traverser une zone des entrées à faible densité ; 3) les entrées de haute dimension se tombent approximativement sur un objet géométrique de basse dimension.

En ce cas, notre découverte surprenante est possible d'être expliquée par les écarts entre les hypothèses et la structure topologique de nos entrées réelles. Nous explorons donc la structure topologique de nos entrées.

3.2 Détection de structure topologique

3.2.1 Enveloppe convexe

Pour un problème de classification à deux classes, il faut examiner tout d'abord la séparation linéaire pour vérifier la validité des discriminants linéaires. La séparation linéaire est une caractéristique selon laquelle les données de deux classes peuvent être classifiés par un hyperplan. C'est à dire, pour une population \vec{x} de deux classes, il existe un vecteur \vec{w} et un seuil c qui permettent le hyperplan $\vec{w} \cdot \vec{x} = c$ sépare les deux classes.

L'enveloppe convexe d'une classe est l'ensemble convexe le plus petit parmi ceux qui la contiennent. Si les enveloppes convexes se croisent, les deux classes se voient comme inséparables linéairement. Ici, l'analyse en composantes principales est effectuée pour visualiser les enveloppes convexes en 2 dimensions.

La figure 3.1 révèle que les classes ne sont pas séparables linéairement parce que les enveloppes se chevauchent. Cette découverte nous montre que l'analyse discriminante linéaire ne peut pas être appliquée à nos données. De plus, les méthodes plus avancées permettant la séparation non-linéaire seront implémentées.

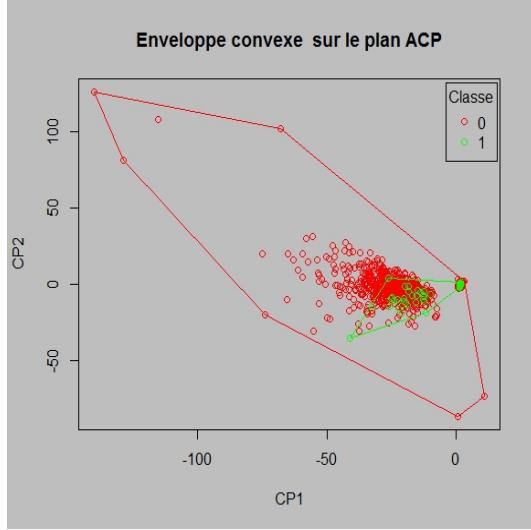


FIGURE 3.4 – Test de séparabilité linéaire

3.2.2 Algorithme t-SNE

Pour démontrer la faiblesse de la carte Kohonen non-supervisée qui ne profite que la structure topologique, une visualisation des données est effectué par l'algorithme t-SNE (t-distributed stochastic neighbor embedding). Pour les points à grande dimension $\mathbf{x}_1, \dots, \mathbf{x}_N$, une distribution de probabilité p_{ij} est définie à partir de la probabilité conditionnelle $p_{j|i}$ s'agissant de la similarité des paires des individus :

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

Une distribution de probabilité q_{ij} est également définie de la même manière pour les points projetés $\mathbf{y}_1, \dots, \mathbf{y}_N$:

$$q_{j|i} = \frac{\exp(-\|\mathbf{y}_i - \mathbf{y}_j\|^2)}{\sum_{k \neq i} \exp(-\|\mathbf{y}_i - \mathbf{y}_k\|^2)}$$

$$q_{ij} = \frac{q_{j|i} + q_{i|j}}{2N}$$

L'algorithme t-SNE est destiné à trouver une projection optimale des points à grande dimension dans un espace de 2 dimension en minimisant la divergence de Kullback-Leibler entre les deux distributions. De cette manière, la proximité des individus est bien gardée. Les points projetés $\mathbf{y}_1, \dots, \mathbf{y}_N$ sont déterminés de manière suivante :

$$\inf_{y_1, \dots, y_N} KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \quad (3.1)$$

Le succès de la classification en utilisant la structure topologique réside dans la séparation topologique des individus de classes différentes. Beaucoup de bases de données de l'apprentissage machine

profite de cette propriété, par exemple, la Base de Iris. Cependant la population de notre base ne manifeste la séparation provenant de la structure topologique, ce qui devient une faible de la carte Kohonen non-supervisée. Les figures suivantes illustre cette faiblesse.

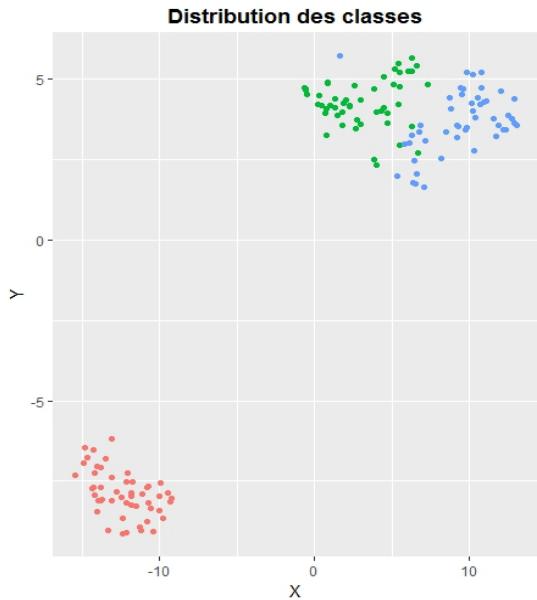


FIGURE 3.5 – Base de Iris

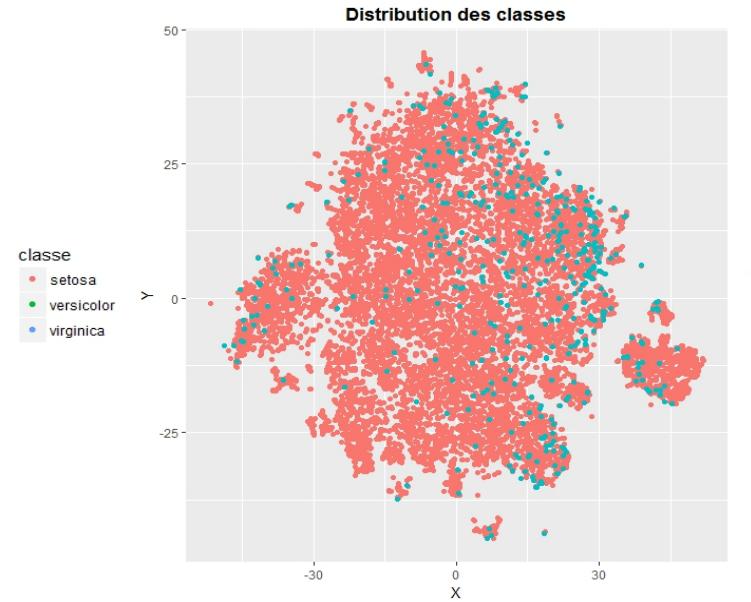


FIGURE 3.6 – Base d'apprentissage TPE

3.3 Carte Kohonen supervisée (SSOM)

La section précédente nous montre que seulement la structure topologique n'est pas suffisant pour faire la classification. Le déséquilibre des classes nous pose aussi un problème. Notre regroupement nécessite des informations supplémentaires et des traitements particuliers. C'est pourquoi le sous échantillonnage et la carte Kohonen supervisée sont effectués.

3.3.1 Sous-échantillonnage

Le sous échantillonnage est une technique qui lutte contre le déséquilibre des classes en diminuant des observations de classe majoritaire dans la base d'apprentissage.[16][?, CN] Il est préférable d'utiliser cette méthode lorsque la base de données est énorme et la réduction du nombre des entrées aide à améliorer le temps d'exécution et les problèmes de stockage. Ici, un sous-échantillonnage des observations de classe majoritaire est réalisé par tirage aléatoire selon une loi uniforme.

3.3.2 Modèle de carte Kohonen supervisée

D'après Melssen et al [5], la carte KOhonen supervisé est un réseau fusionné qui profite à la fois des entrées et des classes à l'étape d'apprentissage. Ce réseau se décomposé en deux types de couches : couches d'entrée et couches de sortie. Ici, nous pouvons décomposer de plus les couches

d'entrée en deux nouveaux types : couches d'entrée quantitative et couches d'entrée qualitative. De cette manière, on obtiendra trois types de couches au lieu de deux types de couches en modèle traditionnel. Le réseau s'entraîne de même manière que celui non supervisé sauf que une pondération des couches est possible.

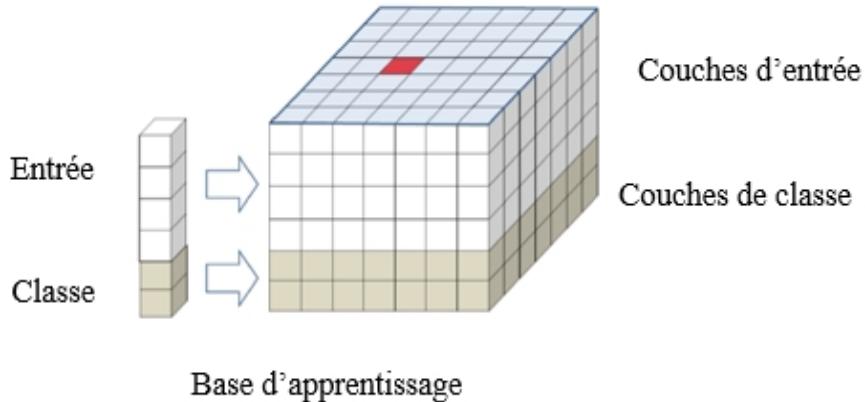
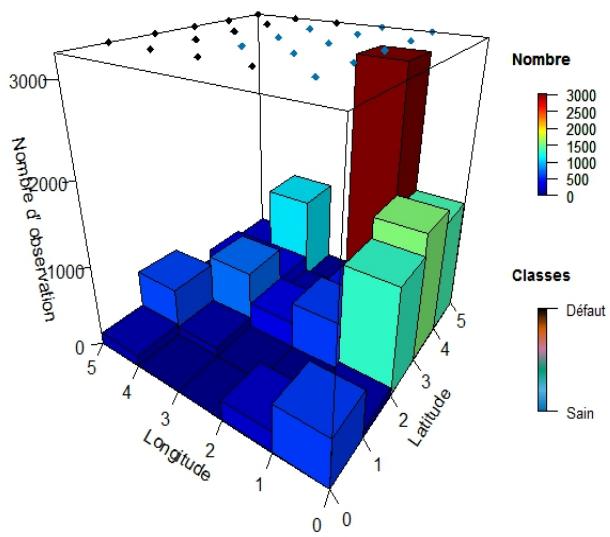


FIGURE 3.7 – Carte Kohonen supervisé

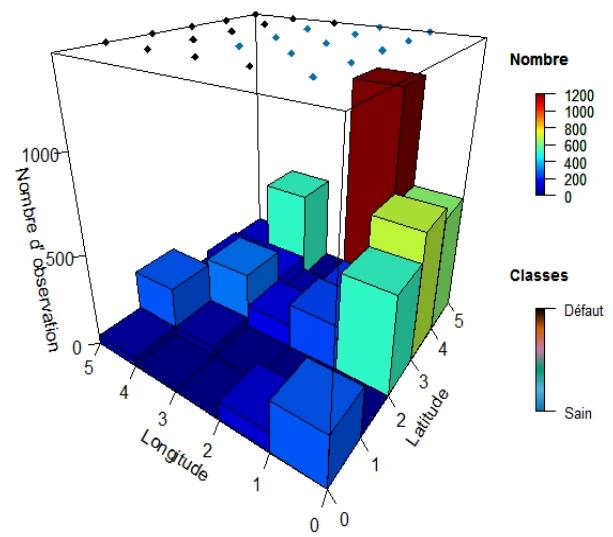
Nous effectuons tout d'abord une sous échantillonnage en diminuant le nombre des clients sains dans la base d'apprentissage à environ 1.1 fois celui de clients défaillants. Ensuite, un calibrage de paramètre est réalisé. Pour la simplicité, nous supposons les poids parmi les couches de même type sont pareils. De ce fait, on ne fait que le calibrage de paramètre sur les poids de différent type. Enfin, un réseau dont un poids de 1/2 est affecté aux couches d'entrée et un poids de 1/2 est affecté aux couches de sortie se trouve le plus raisonnable. En plus, le poids affecté aux couches d'entrée se décompose en deux parties : 3/8 pour les couches d'entrée quantitative et 1/8 pour celles d'entrée qualitative.

La carte Kohonen non supervisée ne peut pas produire les neurone marqué de classe défaillante. Par contre, la carte Kohonen supervisée combinée avec le sous échantillonnage est capable de produire des neurones défaillantes, ce qui est démontré dans les plafonds des figures de boîte suivantes. Le reste des observations sont classifiées selon la classe de neurone qui les représente. Le résultat de la classification sur des deux bases complètes se démontre dans les figures suivantes. Cette fois, la relation inverse entre le nombre des observations de neurone et le taux de défaut local est plus évident. Le taux de défaut augmente sur les neurones défaillantes malgré un manque de la dominance du côté de nombre des clients défaillants.

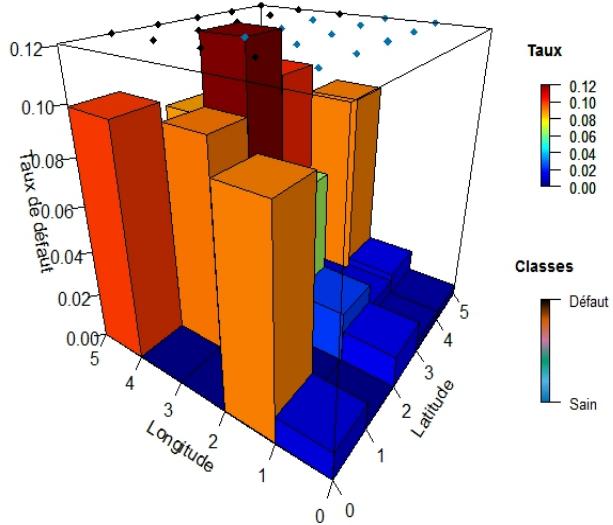
Répartition des observations



Répartition des observations



Répartition du défaut



Répartition du défaut

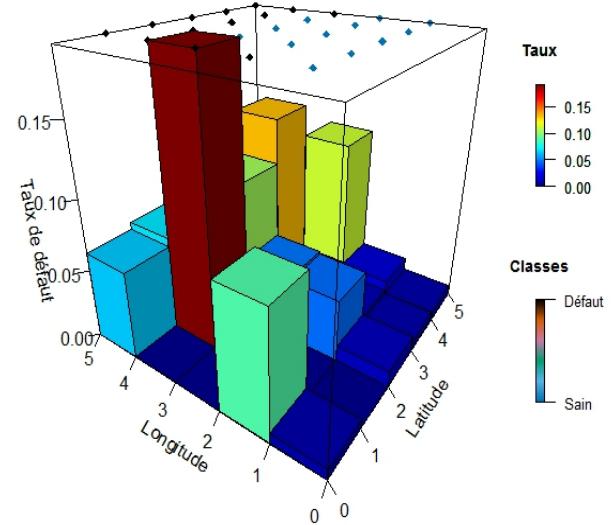


FIGURE 3.8 – Base d'apprentissage

FIGURE 3.9 – Base de test

D'ailleurs, une matrice de confusion est construit sur la base de test pour manifester la qualité de classification de la carte Kohonen supervisée. Chaque colonne de la matrice représente le nombre d'occurrences d'une classe réelle lorsque chaque ligne représente le nombre d'occurrences d'une classe estimée.

	Classe réelle 0	Classe réelle 1
Classe estimée 1	FP (faux positifs)	VP (vrais positifs)
Classe estimée 0	VN (vrais négatifs)	FN (faux négatifs)

Tableau 3.1 – Définition : Matrice de confusion

La matrice nous donne les informations suivantes : sur les 5480 clients sains, 3887 seront estimés comme sains, soit 71% des clients sains sont correctement prédits ; et sur les 189 clients défaillants, 133 seront estimés comme défaillants, soit 70.4% des clients sains sont correctement prédits. Ce résultat est de bonne performance, mais il nous reste à voir s'il existe des modèles traiter mieux des informations fournies.

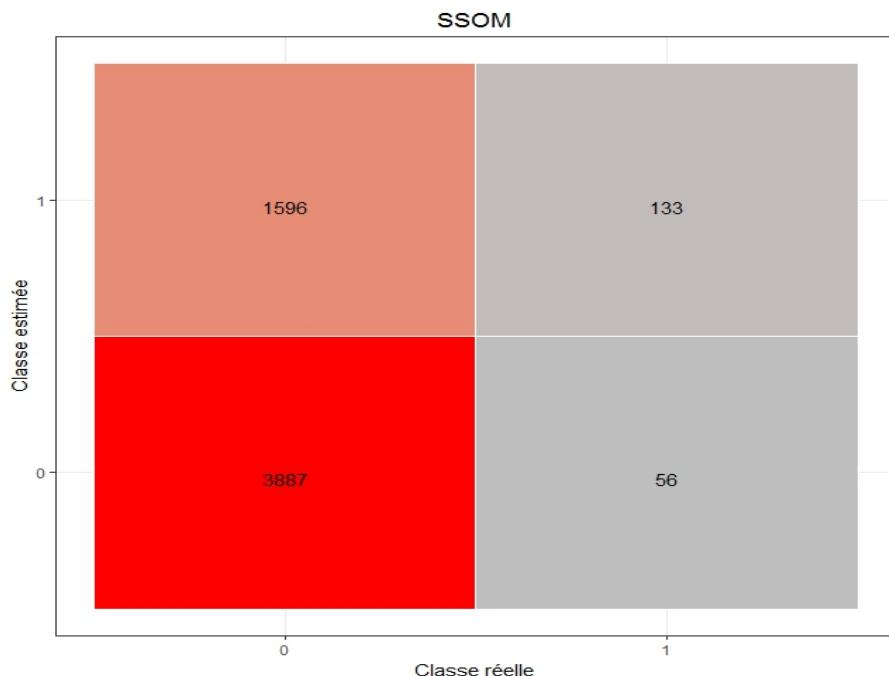


FIGURE 3.10 – Matrice de confusion : Base de test

3.4 Évaluation des performances

3.4.1 Indice de performance

Au cours du regroupement, il nous faut évaluer la qualité de regroupement pour la comparaison et le calibrage de paramètre. Les méthodes d'évaluation doivent être compétent pour mesurer la performance des réseaux générés et tolérer le cas où les neurones marquées de classe défaillante manquent comme on l'a vu avec l'échec de la carte Kohonen non supervisé. Du coup, trois critères sont choisis pour faire l'évaluation : pureté, information mutuelle normalisée et indice de Rand. Sauf l'explication spécifique, les partitions ci-dessous sont deux à deux disjoints par défaut.

Pureté

La pureté est un critère qui évalue la mesure dans laquelle les groupes contiennent une classe unique. Compte tenu d'un ensemble de groupes M et d'un ensemble de classes D , tous les deux divisent N individus. La pureté se calcule de manière suivante :

$$\frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d|$$

Information mutuelle normalisée (NMI)

Contrairement à la pureté qui préfère le grand nombre de groupes, l'information mutuelle normalisée pénalise le grand nombre de groupes. En ce cas, la taille de la carte Kohonen se contrôle. Elle mesure la similarité entre deux partitions d'un ensemble. Compte tenu d'un ensemble de groupes U de R groupes et d'un ensemble de classes V de C classes, les entropies des deux partition se définissent par :

$$H(U) = - \sum_{i=1}^R P(i) \log P(i); \quad H(V) = - \sum_{j=1}^C P'(j) \log P'(j)$$

Et l'information mutuelle de deux partitions se définit par :

$$MI(U, V) = \sum_{i=1}^R \sum_{j=1}^C P(i, j) \log \frac{P(i, j)}{P(i)P'(j)}$$

$$NMI(U, V) = \frac{MI(U; V)}{[H(X) + H(Y)]/2}$$

Indice de Rand (RI)

L'indice de Rand mesure aussi la similarité entre deux partitions d'un ensemble. L'enjeu de ce critère est de mesurer la consistance (le taux d'accord) entre deux partitions. Nous notons deux partitions U et V :

$U \setminus V$	Co-groupée	Non co-groupée
Co-groupée	a	b
Non co-groupée	c	d

L'indice de Rand se calcule par :

$$RI(\pi_1, \pi_2) = (a + d) / \binom{n}{2}$$

3.4.2 Comparaison des performances

Indice \ Méthode	SOM	SSOM	<i>SSOM*</i>
Pureté	0.967	0.967	1
NMI	0.00179	0.0126	0.421
RI	0	0.00647	0.185

Tableau 3.2 – Base d'apprentissage

Indice \ Méthode	SOM	SSOM
Pureté	0.967	0.967
NMI	0.00283	0.0119
RI	0.00145	0.00647

Tableau 3.3 – Base de test

*SSOM** : Carte Kohonen supervisée sur la base d'apprentissage sous échantillonnée

Selon les tableau ci-dessus, l'effet de la combinaison du sous-échantillonnage et l'apprentissage supervisé de SSOM est vraiment évident. Les trois mesure de performance de *SSOM** sont toutes supérieures que celles de carte Kohonen. Néanmoins, il nous faut remettre les individus exclus par le sous échantillonnage pour faire la prévision. En ce cas, l'effet de sous-échantillonnage s'affaiblit et une baisse de la performance est observée. Cependant, l'effet du apprentissage supervisé de SSOM reste dans toutes les deux bases de données, ce qui permet une meilleure performance de SSOM sur la base complète que celle de SOM classique.

Dans le chapitre suivant, nous explorerons l'apprentissage supervisé pour étudier comment profiter mieux de l'interaction entre des entrées et les classes. Dans le chapitre 5, nous introduirons une méthode hybride qui essaie de garder l'effet de sous échantillonnage en utilisant la carte Kohonen lorsque l'apprentissage se déroule de manière supervisée.

Chapitre 4

Apprentissage supervisé

Ce chapitre présente les algorithmes d'apprentissage supervisé qui cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage. Pour la classification, le but des algorithmes d'apprentissage supervisé est d'estimer le lien entre des entrées x et des classes y grâce à la fonction de prédiction (le classifieur) apprise des données connus $f(\cdot)$. En général, cette fonction de prédiction se détermine dans un espace de fonctions, soit espace d'hypothèses, en minimisant un risque moyen $E[l(f(x), y)]$ défini par la fonction de perte $l(f(x), y)$.

D'après le chapitre précédent, il existe deux enjeux pour la construction de classifieur : l'information de classe et la séparation non linéaire. En effectuant l'apprentissage supervisé, nous profitons bien de l'information de classe des entrées. Quant à la séparation non linéaire, quatre classifieurs non linéaires sont choisis : Arbre de décision, Régression logistique pénalisée, Forêts aléatoires et Gradient Boosting. Nous rappelons dans les sections suivantes les principaux des algorithmes utilisés et définitions usuelles.

4.1 Évaluation des performances

Courbe ROC

La courbe ROC est un outil d'évaluation et de comparaison des modèles qui est associé à la matrice de confusion introduite dans le chapitre précédent. La définition de la courbe de taux nécessite du taux de vrais positif (TVP) et du taux de vrais négatifs (TVN).

$$\text{TVP} = \text{Sensibilité (sensitivity)} = \text{VP/Positifs}$$

$$\text{TVN} = \text{Spécificité (specificity)} = \text{VN/Négatifs}$$

La construction de la courbe ROC consiste à faire varier le « seuil » de classification de 1 à 0 et, pour chaque cas, calculer le TVP et le TVN que l'on reporte dans un graphique : en abscisse le TVN, en

ordonnée le TVP.

L'aire sous la courbe (AUC) est la surface de l'aire située sous le tracé de la courbe ROC dessinée dans un repère. Elle indique la probabilité pour que le classifieur $f(\cdot)$ place un positif devant un négatif (AUC = 1 pour le meilleur classifieur). Cette mesure sera aussi inclue dans notre évaluation.

Indice de Gini

L'indice de Gini développé par le statisticien Corrado Gini mesure la qualité de la classification binaire. Il est le rapport des deux aires suivantes : A : aire entre la courbe du modèle construit et la droite de l'aléatoire B : aire entre la courbe du modèle idéal et la droite de l'aléatoire :

$$0 \leq Gini = \frac{A}{B} \leq 1$$

Ce indice s'approche vers 1 quand la performance du modèle est meilleur.

4.2 Calibrage des paramètres

Le calibrage des paramètres est le processus de sélection des valeurs pour les paramètres d'un modèle qui maximisent la précision du modèle. Deux méthodes sont implémentées pour que les paramètres soient déterminés au coût acceptable. Elles s'appliquent aux modèles différents : l'AICc pour les modèles de régression et la recherche par quadrillage pour le reste.

AICc

Le critère d'information d'Akaike (AIC) mesure la qualité d'un modèle statistique en prenant compte à la fois de la vraisemblablement et du nombre de paramètres. Elle se définit par :

$$AIC = 2k - 2 \ln(L)$$

où k est le nombre de paramètres à estimer du modèle et L est le maximum de la fonction de vraisemblance du modèle. Elle est efficace pour traiter les modèles imbriqués.

L'AICc est une correction de l'AIC prenant compte de plus de la taille des observations n :

$$AICc = AIC + \frac{2k(k+1)}{n-k-1}$$

Recherche par quadrillage

Puisque la plupart des classificateurs non linéaire ne sont pas modèle imbriqué, l'AICc ne s'adapte plus aux tels modèles. Prenant compte du coût d'apprentissage, nous décidons de faire la recherche par quadrillage combiné avec «holdout method » pour le calibrage des modèles autre que la régression. C'est à dire, une recherche exhaustive est effectuée à travers un sous-ensemble spécifique de l'espace de paramètre. La mesure de performance est mise en place sur la base de test. Les paramètres

qui possèdent la meilleure performance sur la base de test seront choisis.

Par exemple, un modèle de deux paramètres C et γ s'applique à la classification. Les deux paramètres sont continus, afin d'effectuer une recherche par quadrillage, on sélectionne un ensemble fini de valeurs "raisonnables" pour chacun d'eux : $C \in \{10, 100, 1000\}$; $\gamma \in \{0.1, 0.2, 0.5, 1.0\}$. La recherche aura lieu dans l'ensemble des paires de $\{10, 100, 1000\}$ et $\{0.1, 0.2, 0.5, 1.0\}$. La paire de paramètre de la performance la plus meilleure sera choisie.

4.3 Erreur de généralisation

L'objectif d'apprentissage est de trouver une fonction $f(x)$ qui permet de prédire les valeurs de sortie y basées sur des entrées x . Le risque moyen $E[l(f_n)]$ d'une telle fonction f_n se définit comme suit :

$$E[l(f_n)] = \int_{X \times Y} l(f_n(x, y)) \rho(x, y) dx dy$$

où $\rho(x, y)$ est la loi conjointe de probabilité inconnue pour x et y .

Puisque la loi conjointe de probabilité $\rho(x, y)$ est inconnue, nous ne peut que calculer le risque moyen empirique sur la base d'apprentissage S :

$$E[l_S(f_n)] = \frac{1}{n} \sum_{i=1}^n l(f_n(x_i), y_i)$$

L'erreur de généralisation G est donc la différence entre le risque moyen sur la loi de probabilité conjointe sous-jacente de x et y le risque moyen empirique sur la base d'apprentissage. Elle est définie par :

$$G = E[l(f_n)] - E[l_S(f_n)]$$

Les algorithmes d'apprentissage sont toujours utilisés pour minimiser $E[l_S(f_n)]$. Cependant, il faut faire l'attention à l'erreur de généralisation qui nous permet d'éviter le sur-apprentissage sensible au bruit de la base de l'apprentissage. C'est pourquoi les validations croisées et l'évaluation de performance sur tous les deux bases de données sont effectuées du côté de la validation. D'un autre côté de la construction des modèles, il est raisonnable d'éviter les problèmes de surapprentissage et sousapprentissage entraînés par l'erreur de généralisation. Le sousapprentissage est possible d'être amélioré par le biais du calibrage de paramètres. Le surapprentissage se résoudra dans le cadre de la régularisation de modèles.

4.4 Modèle d'apprentissage

4.4.1 Modèle de référence : Arbre de décision

L'arbre de décision [6] est un classifieur classique pour la séparation non-linéaire des entrées, ce qui lui fait un bon candidat pour le modèle de référence. Elles se basent sur un découpage de l'espace

engendrée par les variables en utilisant les hyperplans parallèles aux axe. Chaque nœud intérieur de l'arbre représente un découpage, et les nœud terminaux représente une valeur des classes pour la classification. Le découpage s'effectue en divisant l'arbre du sommet vers les les nœud terminaux en choisissant à chaque étape une variable d'entrée qui réalise le meilleur partage de l'ensemble des observations selon certain critère.

La fonction de perte de l'arbre de décision est /

$$l(f(x), y) = \begin{cases} 1 & \text{if } f(x) = y \\ 0 & \text{if } f(x) \neq y \end{cases}$$

Le critère de l'entropie de Shannon est choisi pour le découpage :

$$MI(X, Y) = \sum_{x \in X} \sum_{y \in Y} P(x, y) \log \frac{P(x, y)}{P(x)P'(y)}$$

Pour lutte contre les classes déséquilibre, un critère modifié sont construit à partir du critère original en imposant les poids de classes :

$$MI(X, Y) = \sum_{x \in X} \sum_{y \in Y} w(y)^{-1} P(x, y) \log \frac{P(x, y)}{P(x)P'(y)}$$

Après le calibrage de paramètres, les paramètres $w(y) = \{0.7, 0.3\}$ se trouve le plus raisonnable. Le résultat de l'arbre généré est comme suit :

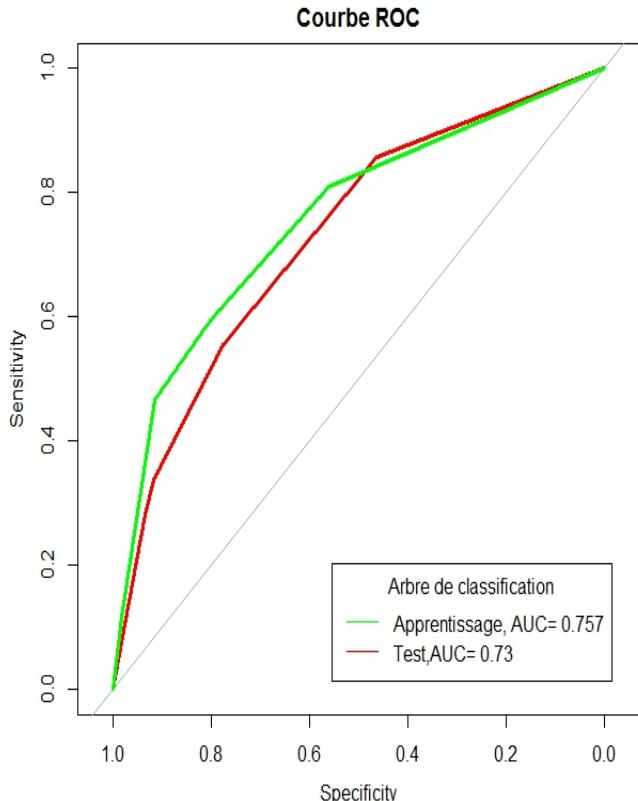


FIGURE 4.1 – Courbe ROC

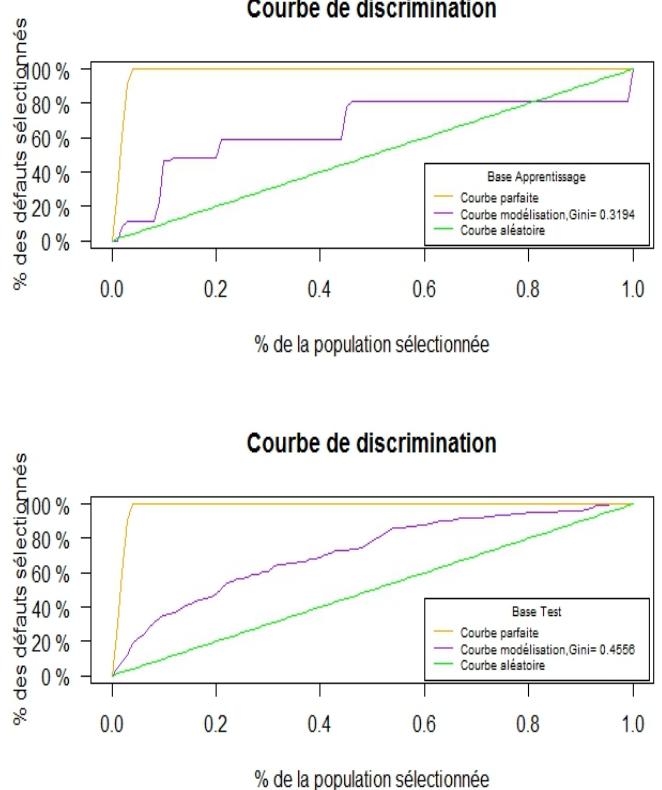


FIGURE 4.2 – Courbe Gini

4.4.2 Régression logistique pénalisée

La régression logistique est un modèle largement utilisé lorsque la réponse est catégorique. Cependant, les modèles de régression se trouvent inappropriés pour des entrées de haute dimensions à cause des problèmes inverses. C'est pourquoi une régression logistique pénalisée est introduite. Le but de la régression logistique est de minimiser la perte logistique moyenne empirique :

$$\inf_{\beta_0, \beta} \quad \frac{1}{N} \sum_{i=1}^N w_i l(y_i, \beta_0 + \beta^T x_i)$$

En ajoutant une terme de pénalisation de type Lasso dans l'équation, un modèle de la régression logistique pénalisée est élaboré pour minimiser le risque moyen empirique pénalisé :

$$\inf_{\beta_0, \beta} \quad \frac{1}{N} \sum_{i=1}^N w_i l(y_i, \beta_0 + \beta^T x_i) + \lambda \|\beta\|_1$$

Un algorithme particulier est appliqué à la minimisation du risque moyen empirique.[7]. Le modèle avec le meilleur AICc est choisi pour la classification et le résultat est comme suit :

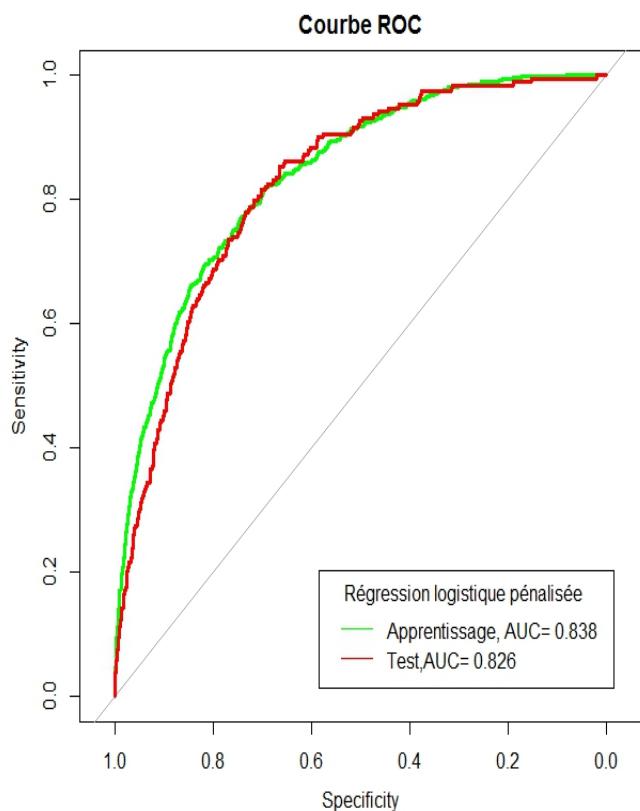


FIGURE 4.3 – Courbe ROC

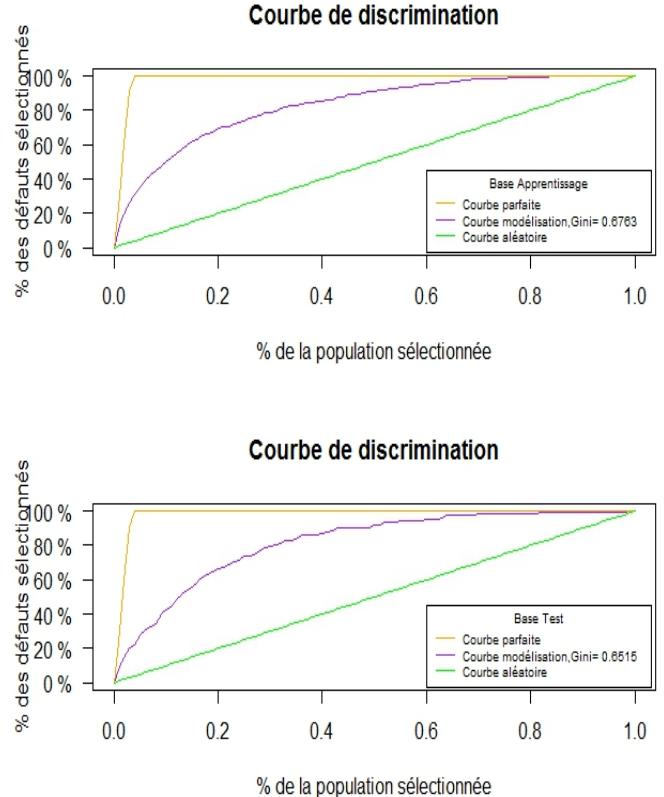


FIGURE 4.4 – Courbe Gini

Conformément à notre attente, la régression pénalisée améliore la performance de la classification.

La performance d'arbre de décision est plutôt modeste avec un AUC de 0.73 sur la base de test. Par contre, la performance de la régression pénalisée atteint un AUC de 0.826 sur la base de test qui est proche de celui de 0.838 sur la base d'apprentissage, ce qui montre une petite erreur de généralisation.

4.4.3 Forêts aléatoires

La section précédente a introduit un modèle de référence qui utilise l'arbre de décision. À partir de ce modèle, le modèle de des forêts aléatoires est construit en appliquant une multitude d'arbres de décision aux sous ensembles des variables explicatives dans un cadre de l'agrégation de modèles. La fonction de perte de l'arbre de décision est donc :

$$l(f(x), y) = \sum_{n=1}^N l_n(f(x), y)$$

où $l_i(f(x), y)$ est la perte de n ième arbre dans les forêts.

Trois paramètres sont choisis pour situer le modèle : nombre d'arbre N , nombre minimal de nœuds terminaux d'arbre na et nombre maximal de nœuds terminaux des forêts nf . Après la recherche par quadrillage sur $N = \{100, 200, \dots, 800\}$, $na = \{5, 6, \dots, 15\}$ et $nf = \{10, 15, \dots, 30\}$, des forêts aléatoires du triplet (800, 10, 25) sont choisi pour la classification.

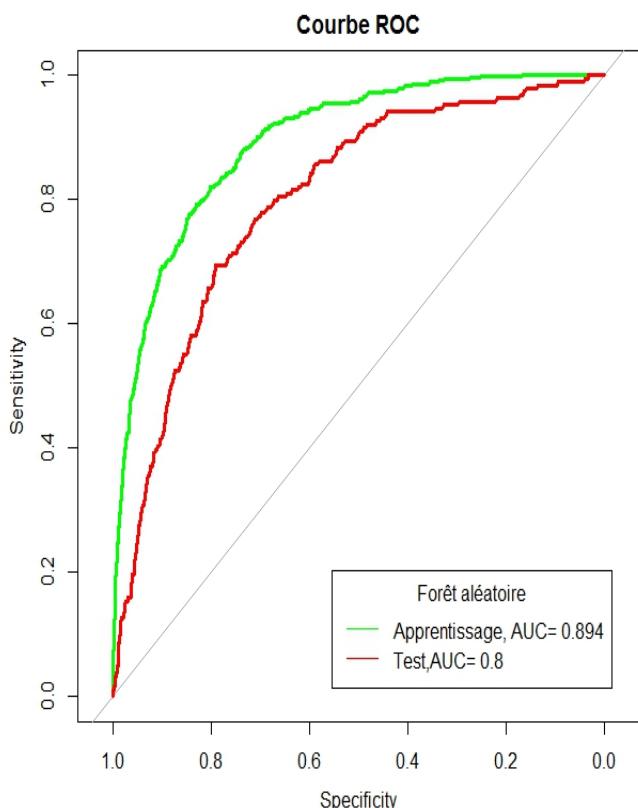


FIGURE 4.5 – Courbe ROC

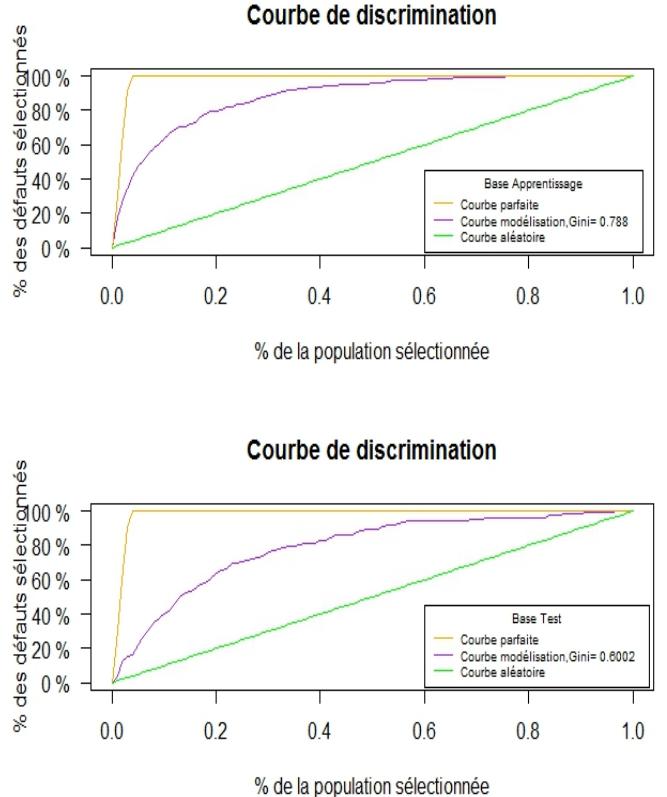


FIGURE 4.6 – Courbe Gini

L'amélioration de la performance des forêts aléatoires est évident par rapport au modèle de référence. Cependant, l'erreur de généralisation augmente au fur et à mesure que une augmentation de la performance dans la base d'apprentissage, ce qui nous fait un souci d'apprentissage.

4.4.4 Gradient Boosting

Lorsque le vote majoritaire de forêts aléatoires est balancé, le Boosting effectuera un vote majoritaire pondéré. La fonction de perte du Boosting est donc :

$$l(y, w) = \sum_{n=1}^N w_n l_n(f(x_n), y_n)$$

Il s'agit d'une itération pour déterminer w et f :

$$\begin{aligned} F_0(x) &= \arg \inf_w \sum_{n=1}^N L(y_n, w), \\ F_m(x) &= F_{m-1}(x) + \arg \inf_{f \in \mathcal{H}} \sum_{n=1}^N L(y_n, F_{m-1}(x_n) + f(x_n)) \end{aligned}$$

Cependant, La recherche de la solution sera coûteux et ennuyant. En ce cas, nous recourons à une simplification dont la fonction de perte se limite dans les fonctions différentiables au lieu de l'espace hypothèses entier. Ici, la perte logistique sera utilisée pour remplacer la perte de 0-1. Une étape de descente le plus abrupte est appliquée à ce problème de minimisation comme ce que fait dans la recherche linéaire. En ce cas, l'entraînement se déroulera de la manière suivante :

$$\begin{aligned} F_m(x) &= F_{m-1}(x) - w_m \sum_{n=1}^N \nabla_{F_{m-1}} L(y_n, F_{m-1}(x_n)), \\ w_m &= \arg \min_w \sum_{n=1}^N L(y_n, F_{m-1}(x_n) - w \nabla_{F_{m-1}} L(y_n, F_{m-1}(x_n))) \end{aligned}$$

Pour bien situer le modèle, trois paramètres sont choisis : taux d'apprentissage η , réduction minimale de perte pour la partition γ et la taille maximal d'arbre individu md . Après la recherche par quadrillage sur $\eta = \{0, 0.05, \dots, 0.3\}$, $\gamma = \{0, 0.04, \dots, 0.2\}$ et $md = \{4, 6, \dots, 10\}$, le modèle des paramètre du triplet $(0.05, 0.08, 6)$ est choisi pour la classification.

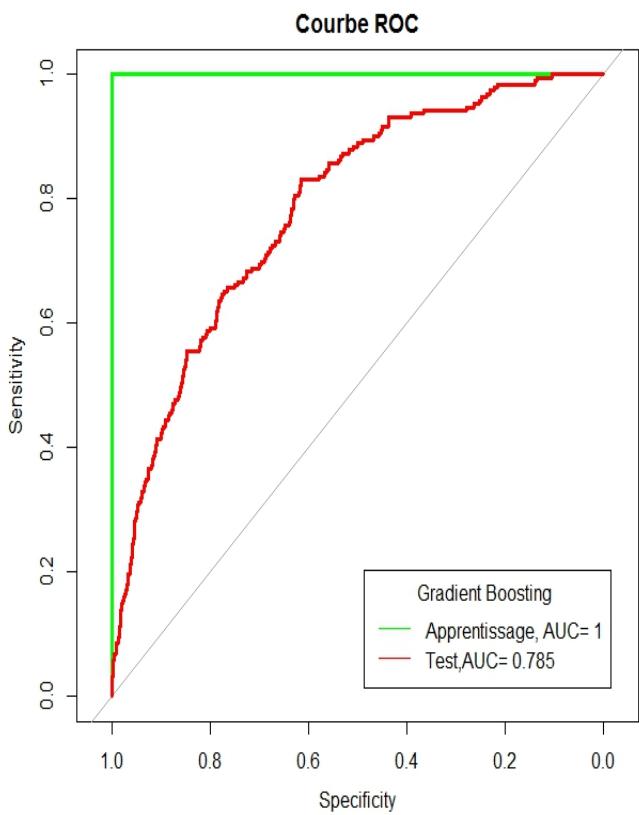


FIGURE 4.7 – Courbe ROC

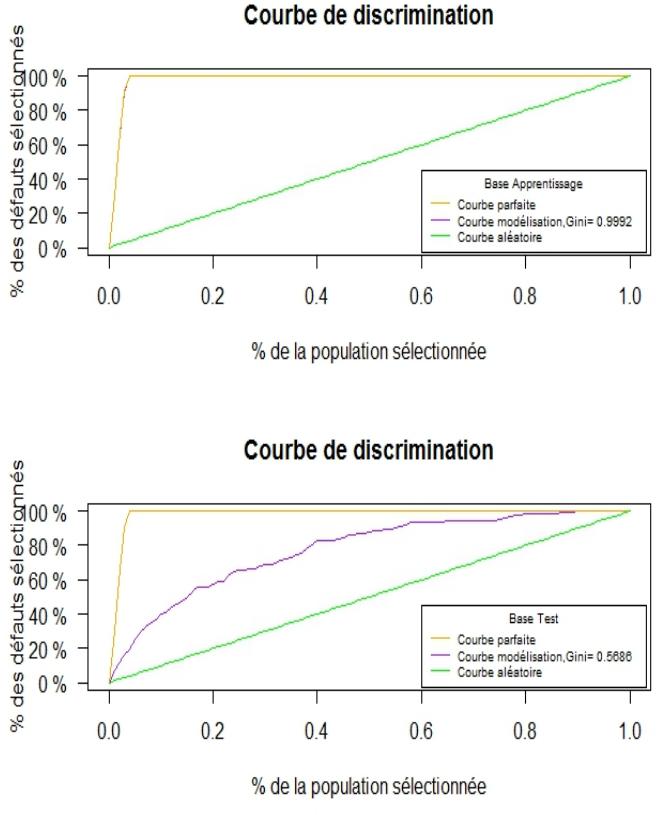


FIGURE 4.8 – Courbe Gini

Le surapprentissage est observé selon la courbe ROC et la courbe Gini, ce qui nous montre que une sélection de variables est nécessaire pour éliminer du bruit dans la base d'apprentissage, soit les variables non discriminantes.

4.5 Sélection de variables

La sélection de variables consiste à trouver un sous-ensemble "pertinent" de variables parmi celles de l'ensemble de départ. Il existe principalement trois catégories des algorithmes de sélection : "wrapper", "filter", et "embedded". À cause du temps coûteux pour effectuer les méthodes "wrapper", nous ne prenons compte que six méthodes de deux dernières parmi lesquelles deux sont de catégorie "filter" et les autres appartiennent à la catégorie "embedded".

4.5.1 Filter

Filter de variance

L'élimination des variables de basse variance est justifiée par Kuhn et al [10]. Elle est une méthode naïve mais robuste pour les modèles variés. Au lieu de l'élimination, nous donnons une note à chaque

variable en faisant une normalisation des variances par ses somme.

Critère de χ^2

Pour utiliser le critère de χ^2 , nous devons transformer des variables quantitatives en variables catégorielles. Ici, chaque variable quantitative est regroupée en 5 catégories selon sa grandeur. Le test de χ^2 [8] sert à déterminer l'existence d'une relation entre deux variables catégorielles : variable explicative x et variable des classes y .

Un tableau de contingence est construit pour les deux variables. Notons O_{ij} l'effectif observé de données pour lesquelles x prend la valeur i et y la valeur j . Une valeur d'espérance d'occurrence E_{ij} sous l'hypothèse d'indépendance est définie par :

$$E_{ij} = \frac{O_{i+} \times O_{+j}}{N}$$

où O_{ij} est nombre d'observations pour lesquelles $X = i$ et O_{+j} est nombre d'observations pour lesquelles $Y = j$. En utilisant les valeurs observées O_{ij} et E_{ij} , la statistique de χ sera construite :

$$T = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

La valeur-p du test servira la mesure de l'importance des variables. Comme les traitement précédents, la sommes des valeurs-p sera normalisée à 100.

4.5.2 Embedded

Sélection de type lasso

Dans la section précédente, on a déjà effectué une régression logistique pénalisée du terme de Lasso.[11] L'avantage de cet régression réside dans le fait que la pénalisation impose une loi priori de Laplace aux coefficients de Laplace. De cette manière, les coefficients de variables non significatives tendent à 0 à cause d'une loi de priori plus pointue. Nous ferons une normalisation selon les coefficients en transformant ses somme en 100. Les coefficients normalisés serviront à la mesure de l'importance de variables.

Sélection de type forêts aléatoires

Les forêts aléatoires peuvent aussi faire une sélection des variables.[9] La réduction de Gini moyenne est la moyenne de la contribution d'une variable dans la réduction de Gini-gain. La réduction de Gini-gain aura lieu quand un variable utile est enlevé du modèle. Donc les variables avec les grande réduction de Gini moyenne seront les variables les plus importantes. De même manière que la méthode précédente, la somme des réductions de Gini moyennes se transforme en 100.

Sélection de type gradient boosting

L'importance des variables est évaluée en calculant la contribution des variables à l'amélioration de précision.[12] En ajoutant un nœud à une variable discriminant, la précision du classifieur s'augmente. La proportion de la contribution des variables à la précision augmentée nous indique l'importance des variables. Cette proposition est donc proposée pour distinguer les variables discriminantes.

Sélection stabilisante

La sélection stabilisante est une nouvelle approche[13] basée sur le bootstrap et le modèle de base. Le modèle de base peut être un classifieur arbitraire. Elle consiste à appliquer le modèle de base aux sous-ensembles des entrées et variables, ce qui est différent que les forêts aléatoire effectué sur seulement des sous-ensembles des entrées. Le gradient boosting de perte logistique est choisi comme le modèle de base ici. La fréquence que une variable est considérée importante dans des modèles de base sert à la mesure de l'importance. Pour la comparaison, la somme de la fréquence se normalise à 100. Idéalement, la note des variables inutiles sera proche de zéro.

Sélection de moyenne

Nous entendons faire une nouvelle évaluation robuste depuis les six méthodes précédentes. Donc, la sélection de moyenne est de calculer une moyenne des notes précédentes pour évaluer l'importance des variables. En ce cas, l'importance des variables évaluée sera un vote des méthodes différentes.

4.5.3 Variables plus importantes

Après le calcul de sept mesure, nous montrons ici les 20 variables les plus important selon le critère de la sélection moyenne. La plupart des variables concerne les donnée de compte, ce qui implique l'importance des informations internes. D'ailleurs, les variables externes nous aident aussi à la classification mais en jouant un rôle supplémentaire. L'exploration des données externes sera un développement potentiel.

Variable \ Méthode	glmImp	rfIMP	xgbImp	stabImp	varImp	chiImp	moyImp
COT_BDF_ENTREP.Sup	15.77	0.08	0.01	10.31	0.00	4.60	5.13
TOP_INCID_SIREN.0	17.54	0.08	0.02	9.79	0.00	3.02	5.08
NOTE_MOTEUR_T	3.55	2.66	4.38	10.50	0.51	6.92	4.75
TOP_INCD_12M_SIREN_C.0	6.53	1.87	0.01	10.61	0.04	5.99	4.17
NBJ_DEPASS_MOIS_SIREN_12M_2	1.97	4.79	0.10	10.39	0.51	5.06	3.80
NB_JOUR_DEBT_SIREN_12M	1.14	3.51	1.68	7.67	0.51	4.62	3.19
ALLUR_CAV	0.97	1.24	0.08	7.53	0.51	4.00	2.39
NOTCOF	2.52	0.37	0.10	6.92	0.51	2.32	2.12
COT_BDF_ENTREP.5	4.01	0.06	0.01	3.31	0.03	1.36	1.46
COT_BDF_ENTREP.3	7.81	0.10	0.02	0.00	0.04	0.75	1.45
TOP_INCD_12M_SIREN_C.0.1	0.00	2.53	0.00	0.00	0.04	5.99	1.43
TOP_INCD_12M_SIREN_C.1.1	0.74	1.66	0.00	0.00	0.04	5.99	1.40
SLDE_MOY_SIREN_3M	0.17	5.57	1.47	0.16	0.51	0.08	1.33
TOP_INCID_SIREN.1	0.00	0.07	0.00	4.34	0.00	3.02	1.24
MNT_EE_CLI_12M_SIREN	0.91	2.16	0.75	2.29	0.51	0.33	1.16
SLDE_MOY_SIREN_1M	0.00	4.37	1.50	0.00	0.51	0.17	1.09
NOTMOT	0.00	1.14	0.09	0.09	0.51	4.45	1.05
COD_ACT_REF.6	5.85	0.00	0.05	0.08	0.01	0.14	1.02
PD_COFACE	0.00	0.41	0.09	2.80	0.51	1.98	0.96
RSS1	0	1.29	0.64	3.03	0	0	0.83

Tableau 4.1 – Sélection des variables

4.6 Résultat des modèles modifiés

Nous construisons sept ensembles de variables pour vérifier l'effet de la sélection des variables. Le nombre maximale des éléments des ensemble est 172 qui est exactement celui du "filter" de gradient boosting. Nous profitons donc de la troncation provenant du modèle. En ce cas, le nombre 172 est donc modèle dépendant. Mais il se varie peu quand les paramètre qui situent le classifieur de Gradient Boosting se changent selon notre observation.

Les notations des méthodes s'écrivent comme suit :

varImp : Filter de variance, chiImp : Critère de χ^2 , glmImp : Sélection de type lasso, rfIMP : Sélection de type forêts aléatoires, xgbImp : Sélection de type gradient boosting, stabImp : Sélection stabilisante, moyImp : Sélection de moyenne.

Les modèles originaux sont modifiés en se contraignant aux variables dans l'ensemble construit par un des 7 méthodes. En fonction des résultats, l'erreur de généralisation se diminue et la performance de la base de test s'améliore pour la plupart des cas. Parmi les modèles modifiés, la régression logistique pénalisée se trouve le plus performant. Nous la choisissons pour la validation croisée dans le but de la vérification de robustesse et de la comparaison avec notre nouveau modèle dans le chapitre

5. Puisque la classification est refait sur les modèles de même paramètres, l'effet de la sélection des variables est validé. Cependant, la performance des méthodes de sélection face aux modèles différents se varie. Les méthodes dont la performance reste robuste et supérieur que celle des modèles originaux sont le "filter" de type lasso et le "filter" de la sélection moyenne. Donc nous utiliserons les modèles combinant avec des deux méthodes pour la validation croisée de suite.

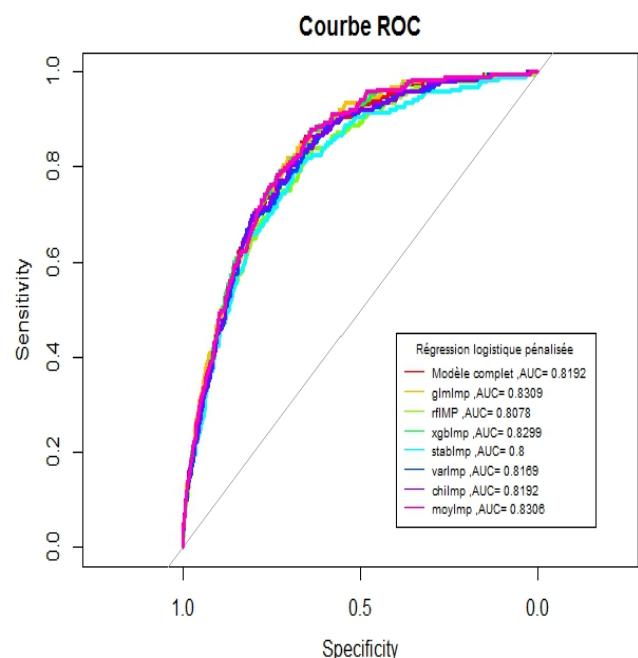
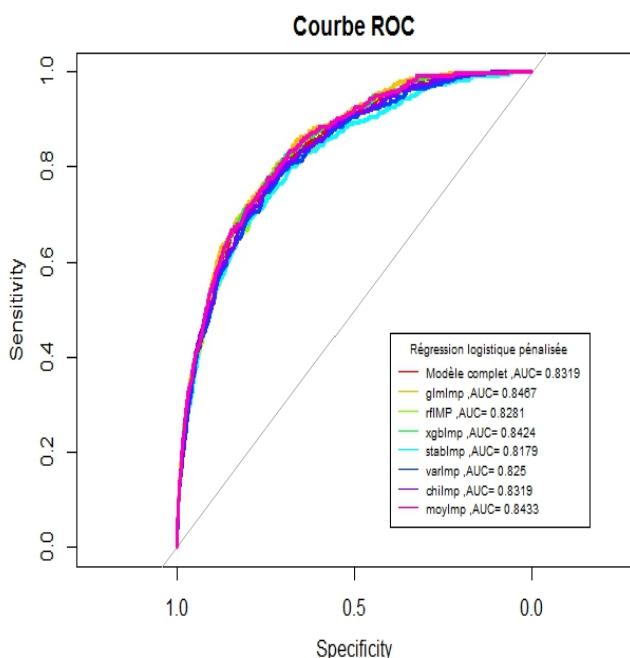
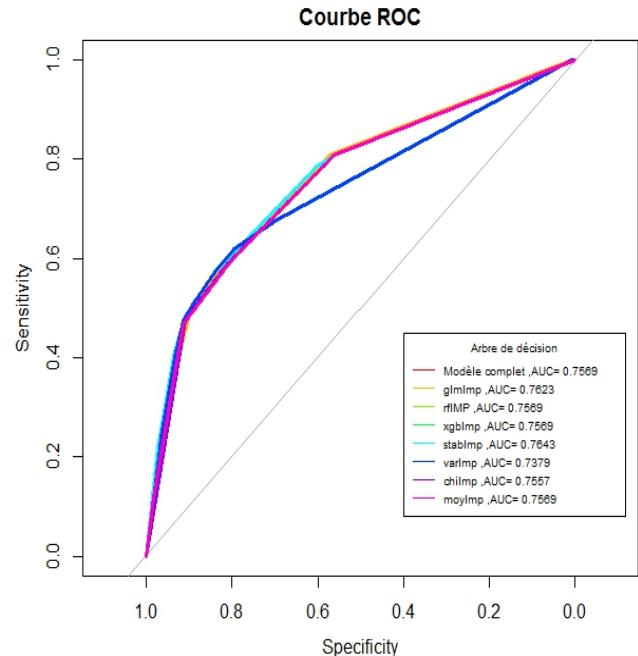
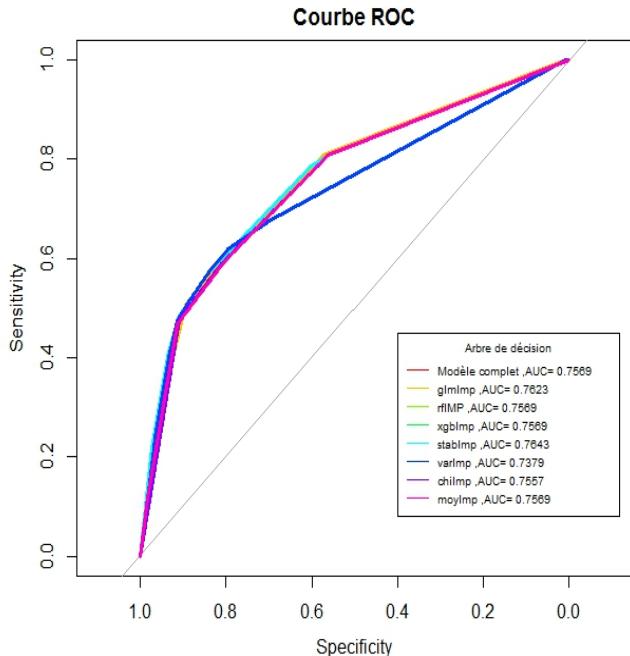


FIGURE 4.9 – Base d'apprentissage

FIGURE 4.10 – Base de test

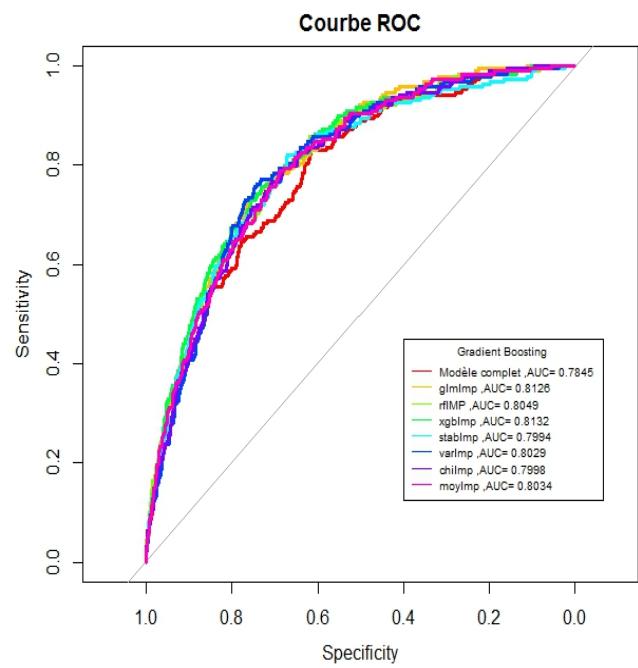
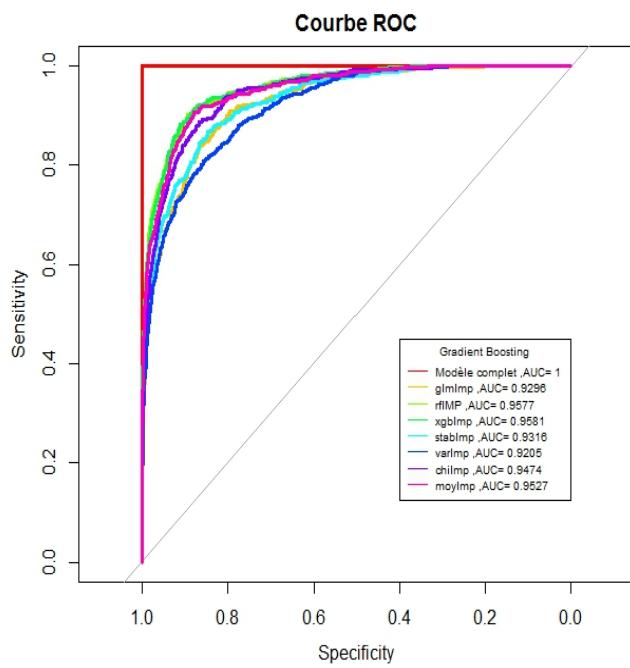
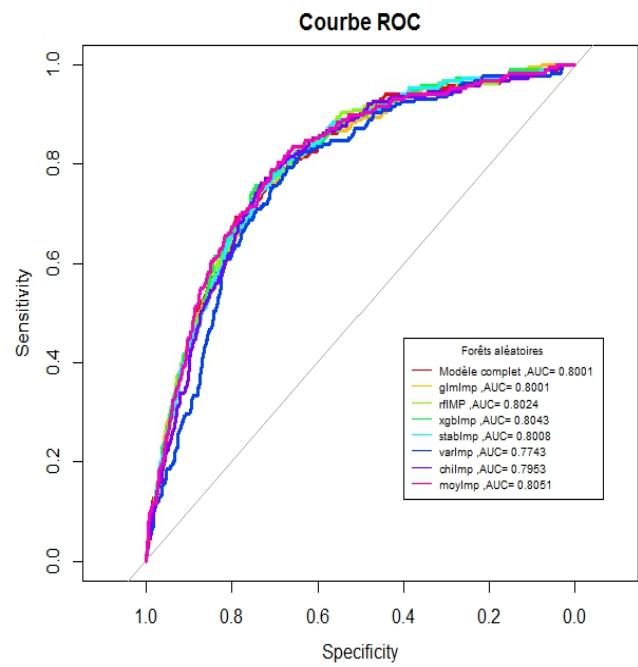
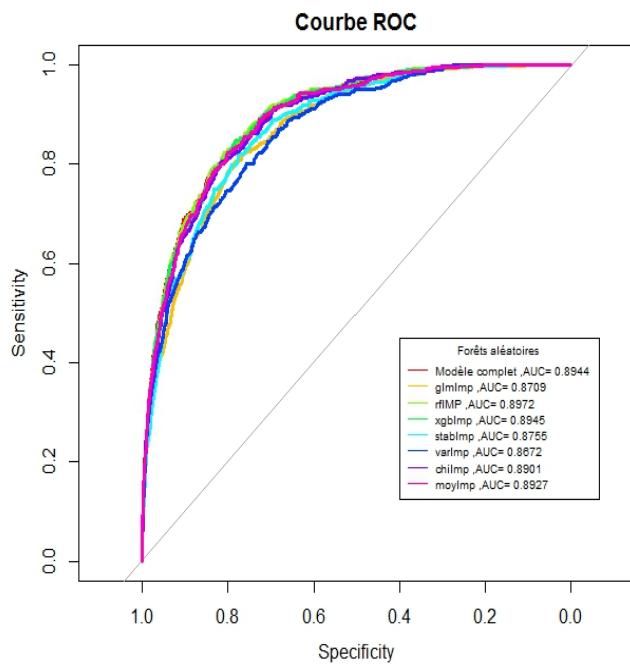


FIGURE 4.11 – Base d'apprentissage

FIGURE 4.12 – Base de test

Chapitre 5

Apprentissage hybride

Dans ce chapitre, nous présentons un cadre d'apprentissage hybride qui profite à la fois l'apprentissage non supervisé et l'apprentissage supervisé. Selon le chapitre 3, le sous-échantillonnage améliore la performance d'apprentissage. Cependant, cet avantage se diminue quand nous remettons les observation de la base originale exclues par le sous-échantillonnage. Nous voulons donc garder l'avantage de l'échantillonnage d'une manière moins directe pour éviter la baisse entraînée par la remise de les observations exclues.

D'après l'avis de chang[15], nous trouvons que un sous-échantillonnage peut être remplacé par une division des observations de classe majoritaire. La carte Kohonen sera un outil idéal pour la division grâce à sa compétence pour la préservation de la structure topologique. Basé sur la division, les ensembles locaux d'apprentissage qui comporte chacun un sous-ensemble des observations de classe majoritaire et le total des observations de classe minoritaire serons construits et ils sont plus équilibre en classe. Les variables synthétique s'apprend à partir des ensembles en utilisant des classifieurs locaux. Un classifieur final sera proposé pour apprendre le lien entre les classes et des variables synthétique.

Un avantage de ce cadre est que le nombre des variables mise dans le classifieur final sont modèle dépendant au lieu de entrée dépendant, ce qui nous aide à contrôler la dimension des entrées. D'un autre côté, l'apprentissage local coûtera moins de piles au meure d'apprentissage.

5.1 Construction des variables synthétiques

5.1.1 Hétérogénéité

L'hétérogénéité géométrique est le cauchemar pour la classification des classes. Il est difficile d'avoir une bonne précision de prévision s'il existe une certaine hétérogénéité dans la base d'apprentissage. Il est idéalement de limiter les clients dans une sous-population le plus susceptible d'être défaillant sur laquelle nous faisons l'apprentissage. Cependant, nous disposons peu de informations à

priori sur les clients défaillants. Du coup, les clients sains des autre sous-population sont prise dans la base d'apprentissage. En ce cas, un seul classifieur n'est pas suffisant pour la classification comme montré dans la figure suivante. C'est pourquoi la division des clients sains est effectuée.

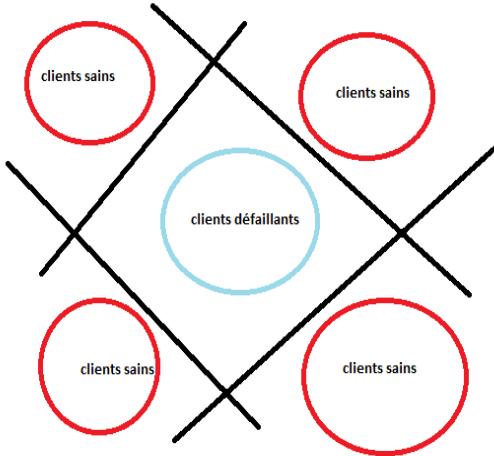


FIGURE 5.1 – Hétérogénéité géométrique

5.1.2 Classifieur local et distance signée

Un classifieur local est imposé sur chaque ensemble local. D'après la détection de la structure topologique, la séparation d'un ensemble local est susceptible d'être non-linéaire. Selon Boczko et al[14], un classifieur local de SVM (Machine à vecteur supports)muni de RBF noyau sera utilisé. De plus, la distance signée $d(x)$ générée du classifieur local sert de variables synthétiques à entrer dans le classifieur final.

Le RBF noyau de paramètre σ entre deux entrées x et x' s'écrit comme suuit :

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

Le SVM muni de RBF noyau recherche à ramener un problème de classification à celui de trouver un hyperplan optimal $\langle w, \varphi(x) \rangle + b = 0$ où $\varphi(x)$ est une transformation de x dont le produit scalaire est le RBF noyau. Le problème de l'optimisation de SVM se définit comme suit :

$$\begin{cases} \inf_w & \frac{1}{2} \|w\|^2 \\ sc & \forall i, y_i (\langle w, x_i \rangle + b) \geq 1 \end{cases}$$

La distance signée $d(x)$ est donc :

$$d(x) = \frac{\langle w, x_i \rangle + b}{\|w\|}$$

5.2 Modèle de prévision

5.2.1 Procédure de modélisation

La régression logistique pénalisée est trouvé à la fois robuste et performante dans la classification. Nous la prenons pour le classifieur final. Les trois éléments principaux de modélisation sont donc déterminés : la division des observations de classe majoritaire, les classificateurs locaux et le classifieur final. L'algorithme d'apprentissage hybride est comme suit :

- 1) Faire un regroupement basé sur la carte Kohonen de 9×9 sur les entrées de classe majoritaire \vec{w} . La taille de la carte est calibrée sur un quadrillage $\{5, 6, \dots, 10\} \times \{5, 6, \dots, 10\}$.
- 2) Construire les ensembles locaux qui se compose d'un sous ensemble des entrées de classe majoritaire et le total des entrées de classe minoritaire.
- 3) Dans chaque groupe, le SVM muni de RBF noyau est entraînée pour construire un hyperplan de séparation locale
- 4) À partir des hyperplans locaux, calculer des distances signées pour tous les entrées
- 5) En utilisant des distances signées comme variables explicatives synthétiques, effectuer une régression logistique pénalisée pour la prévision de probabilité d'être défaillant.

5.2.2 Résultat empirique

Une petite erreur de généralisation est observée dans le modèle hybride. Comparé aux modèles d'apprentissage supervisé, la performance est plutôt satisfaisant avec les variables explicatives maximale de 81 au lieu de l'entier de variables explicatives.

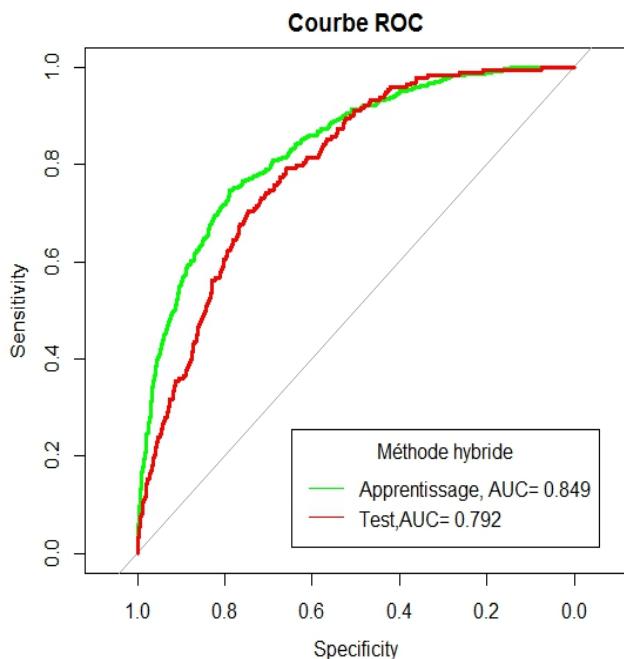


FIGURE 5.2 – Courbe ROC

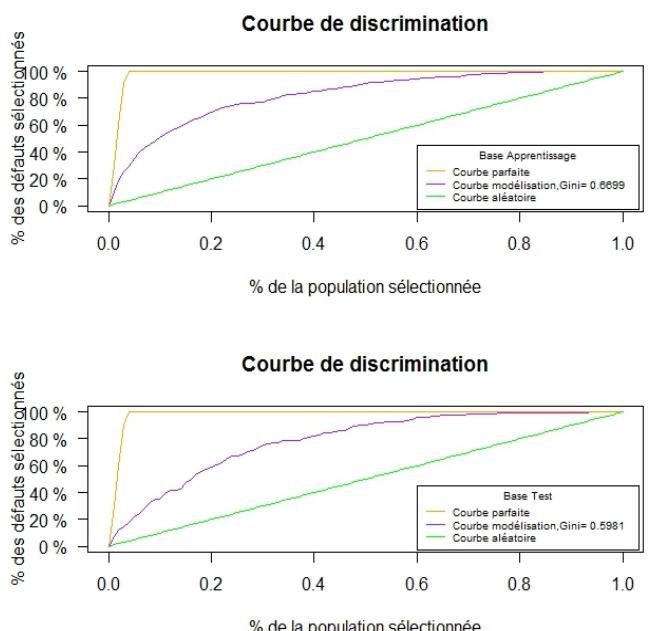


FIGURE 5.3 – Courbe Gini

Pour voir l'effet de la sélection des variables, le modèle est refait sur les variables les plus importantes. Les performances des modèles des "filter" de lasso et "filter" de moyenne sont meilleur à la fois dans la base d'apprentissage et la base de test par rapport au modèle original. Donc nous choisissons les deux modèles modifiés pour la validation croisée suivante.

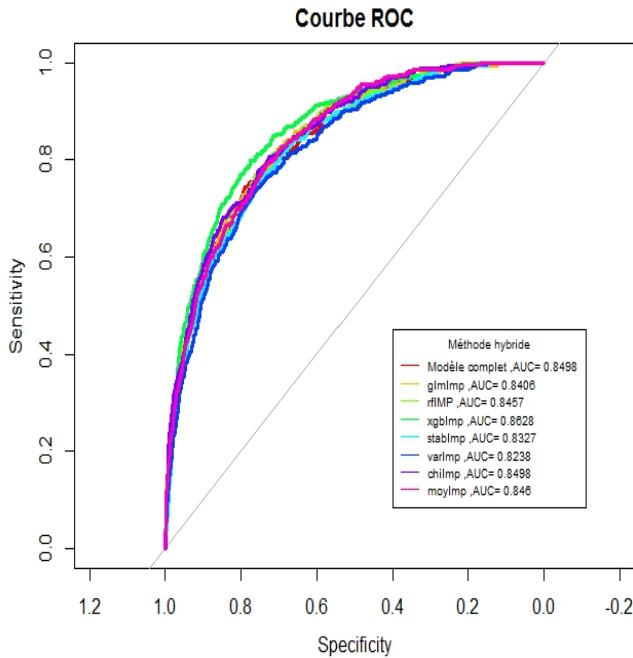


FIGURE 5.4 – Base d'apprentissage

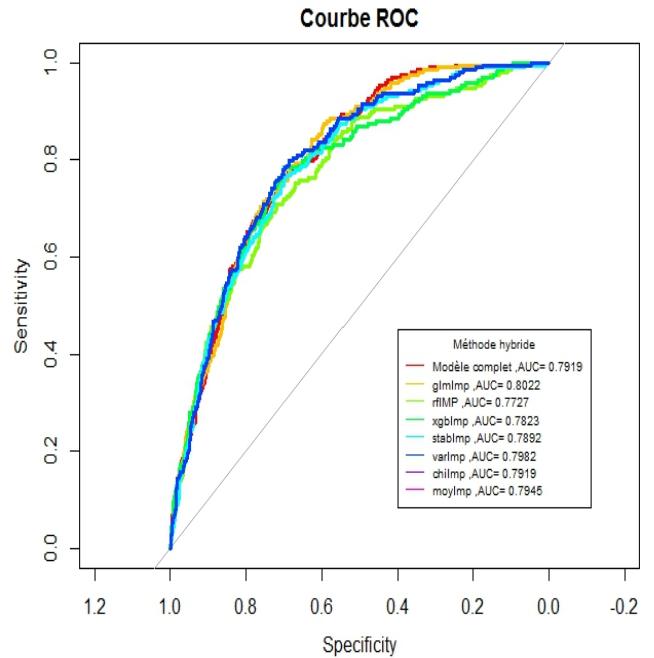


FIGURE 5.5 – Base de test

Chapitre 6

Validation croisée

6.1 Validation croisée Monte Carlos

La validation croisée consiste à vérifier la robustesse des modèle d'apprentissage statistique. La méthode le plus usuelle est validation croisée de k-fold. On divise la base d'apprentissage en k parties, et puis on sélectionne une partie comme base de validation et les (k-1) autres parties comme nouvelle base d'apprentissage. En répétant k fois cette opération, on obtiendra une suite de la mesure de performance comme AUC ou erreur quadratique pour la base de validation. La moyenne de la suite sera utilisée pour évaluer la performance. Normalement, il nous faut un grand k pour diminuer le biais. Cependant, les clients défaillants n'occupent que 3% du total et un grand k comme 10 rend la base de validation non représentatif à cause de la dispersion de répartition topologique des clients défaillants.

Par contre, la validation croisée Monte Carlo [16] stratifiée n'a pas de souci de sous-représentation. Du point de vue de Chen et al[17], la validation croisée Monte Carlo rend le modèle plus raisonnable. En effectuant un échantillonnage stratifié selon les classes, nous pouvons toujours construire une base d'apprentissage de 70% et une base de validation de 30% du total. De cette manière, le représentatif de la base de validation est mieux assuré.

6.2 Résultat empirique

Les validations sont mise en place sur deux modèles modifiés : régression logistique pénalisé s'appuyant sur les variables les plus importantes et modèle hybride s'appuyant sur les variables les plus importantes. Les variables les plus importantes sont choisies selon la critère de "filter" de type lasso et "filter" de moyenne. Les performance de la base de test est plus variant que celle de la base d'apprentissage. Les petites écart de AUC moyen sont observée sur tous les deux modèle, ce qui montre la robustesse des modèle. D'ailleurs, la performance de notre modèle hybride est proche de celle du modèle supervisé. C'est une preuve qui montre les variables synthétiques sont informatives

en classification. Les variables modélisées dépendantes à l'air d'être une cure potentielle du fléau de dimension.

"Filter" de type lasso :

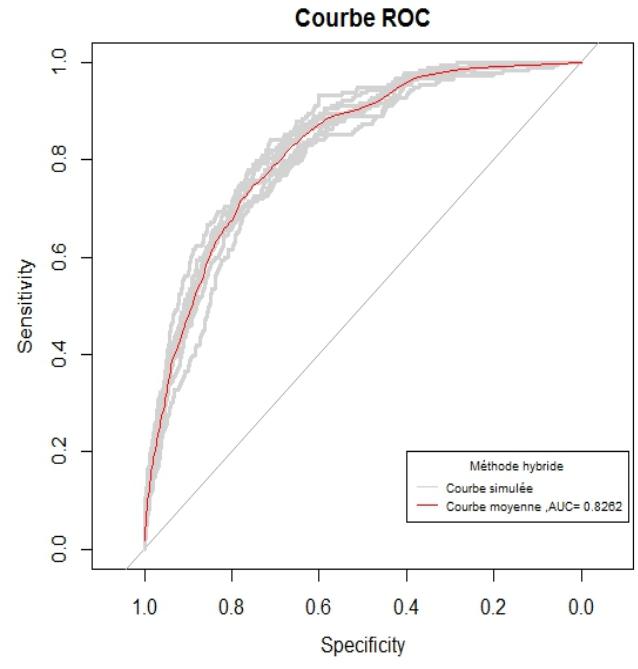
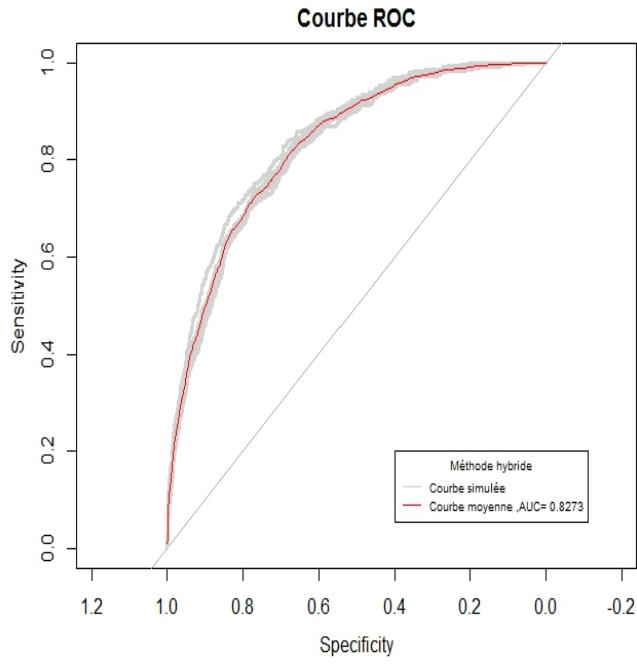
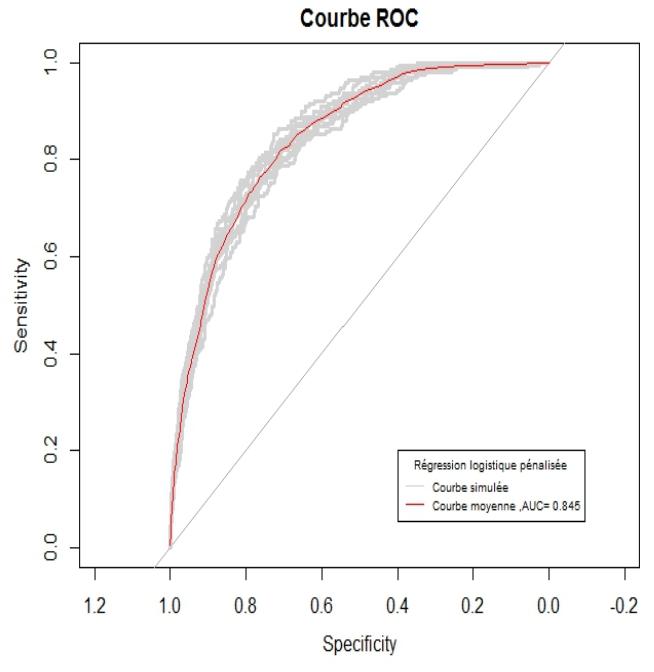
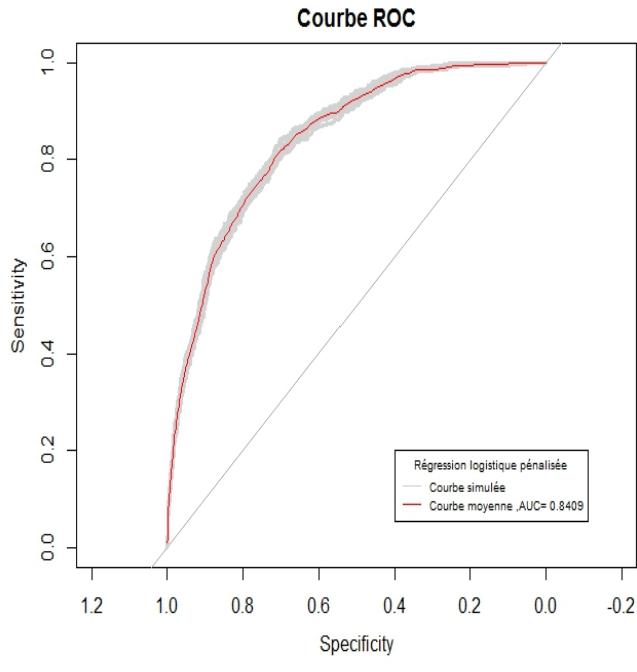


FIGURE 6.1 – Base d'apprentissage

"Filter" de moyenne :

FIGURE 6.2 – Base de test

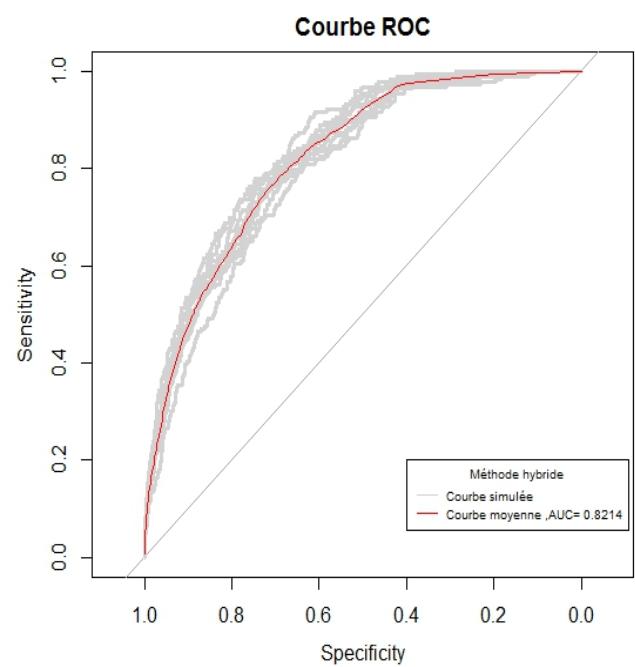
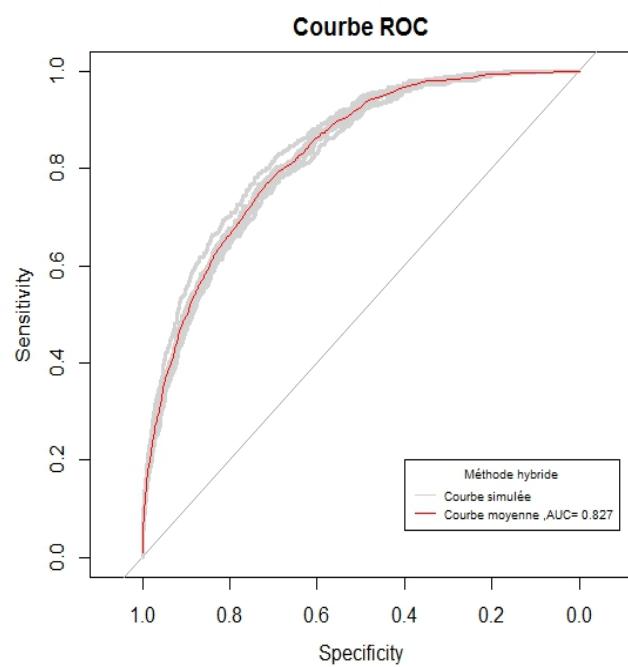
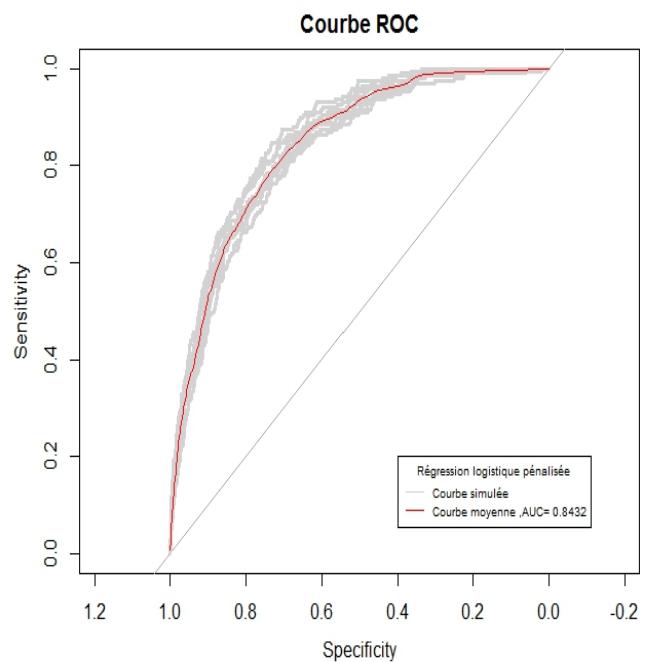
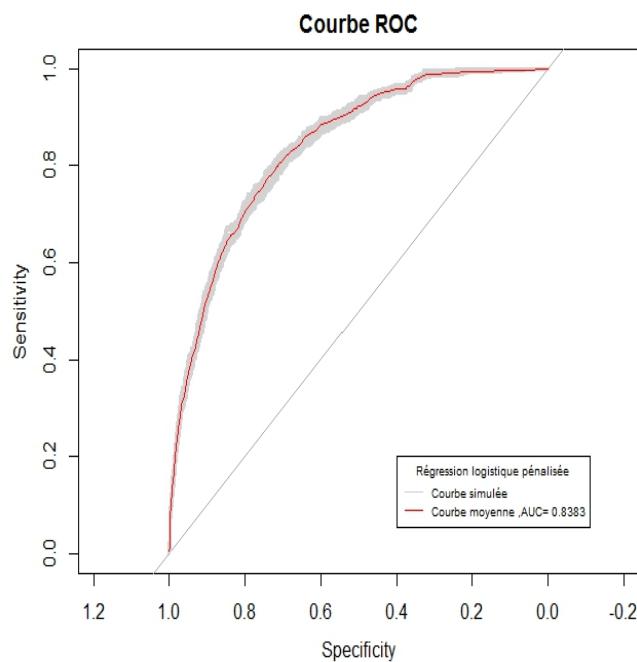


FIGURE 6.3 – Base d'apprentissage

FIGURE 6.4 – Base de test



Chapitre 7

Conclusion et perspectives

Durant l'étude du chapitre 3, on a vu que la classification s'appuyant sur l'apprentissage de la structure topologique dépend de la séparation naturelle. Au cas où la situation est plus complexe, il nous faut l'information supplémentaire de classe et le reconstruction de base d'apprentissage par sous-échantillonnage. Deux essais sont effectués : soit utiliser l'information de classe comme nouveau entrée combinant avec sous-échantillonnage pour entraîner une carte Kohonen supervisé, soit prendre l'information de classe comme sortie et employer les modèles d'apprentissage supervisé pour étudier le lien entre les entrées et les sorties. D'ailleurs une sélection des variables est effectuée pour trouver un sous espace des entrées le plus discriminant.

Basée sur les deux essais, une méthode d'apprentissage hybride est proposé pour profiter de l'avantage de deux essais. La carte Kohonen joue le rôle de sous-échantillonnage d'une manière plus résistant. Les classificateurs locaux sont mises en place pour construire les variables synthétiques. La performance de cette méthode est satisfaisant comparée au modèle de référence. Selon le résultat de validation croisée, elle est compétitive comme une méthode des variables modèle dépendantes face aux modèles d'apprentissage supervisé des variables entrée dépendantes.

Il y a encore plusieurs côté à faire attention. D'abord, du côté de la qualité de donnée, il nous faut bien traiter le fusionnement des bases de données de cohortes. La base d'apprentissage se construit des bases de cohortes différentes en faire une échantillonnage stratifié. Cependant, un simple fusionnement des bases risque de sous-estimer le taux des clients défaillants. Les entrées des clients tiré des cohortes avant de la dernière sont préférable d'être renouvelées dans le fusionnement des bases de données.

Ensuite, une pondération d'entrées est possible d'être implémentés en ajustant la fonction de perte selon les cohortes. En ce cas, nous prendrons compte de l'effet du temps.

Enfin, les modèles plus avancés et les nouvelles méthodes de modifier l'espace des entrées peuvent être proposés pour améliorer la prévision et approfondir notre compréhension des données.

Bibliographie

- [1] IBBOU Smail *Classification, analyse des correspondances et methodes neuronales*. . These, Universite Paris 1, 1992.
- [2] Melssen, Willem, Ron Wehrens, and Lutgarde Buydens. *Supervised Kohonen networks for classification problems*. Chemometrics and Intelligent Laboratory Systems 83.2 (2006) : 99-113.
- [3] Chapelle, Olivier, Bernhard Scholkopf, and Alexander Zien. *Semi-Supervised Learning*. MIT Press, 2006
- [4] Chawla, Nitesh V. *Data mining for imbalanced datasets : An overview*. Data mining and knowledge discovery handbook. Springer US, 2009. 875-886.
- [5] Melssen, W., Wehrens, R., Buydens, L. *Supervised Kohonen networks for classification problems*. Chemometrics and Intelligent Laboratory Systems, 83(2), 99-113 (2006).
- [6] Breiman L., Friedman J. H., Olshen R. A., and Stone, C. J. *Classification and Regression Trees*. (1984) Wadsworth.
- [7] Jerome Friedman, Trevor Hastie, Robert Tibshirani *Regularization Paths for Generalized Linear Models via Coordinate Descent*. Journal of Statistical Software, 33(1), 1-22. (2010).
- [8] Schütze, H. *Introduction to information retrieval* . Proceedings of the international communication of association for computing machinery conference.(2008, June).
- [9] Louppe, G., Wehenkel, L., Sutera, A., Geurts, P. *Understanding variable importances in forests of randomized trees*. . Advances in neural information processing systems (pp. 431-439), (2013).
- [10] Kuhn, M., Johnson, K. . *Applied predictive modeling* New York, NY : Springer (2013)
- [11] Richard G. Baraniuk *Compressive Sensing*. IEEE Signal Processing Magazine
- [12] Xu, Z., Huang, G., Weinberger, K. Q., Zheng, A. X. *Gradient boosted feature selection*. Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 522-531). ACM. (2014, August).
- [13] Meinshausen, N., Bühlmann, P. *Stability selection*. Journal of the Royal Statistical Society : Series B (Statistical Methodology), 72(4), 417-473.(2010).
- [14] Boczko, E. M., Xie, M., Wu, D., and Young, T. *Comparison of binary classification based on signed distance functions with support vector machines*. OCCBIO'09. Ohio Collaborative Conference on (pp. 139-143). IEEE.2009

- [15] Yuan-chin Ivan Chang *Boosting SVM classifiers with logistic regression.*
www.stat.sinica.edu.tw/library/c tec rep/2003 - 03.pdf,2003
- [16] Dubitzky,, Werner ; Granzow, Martin ; Berrar, Daniel *Fundamentals of data mining in genomics and proteomics*. Springer Science & Business Media. p. 178. (2007)
- [17] W Chen, Y Du, F Zhang, R Zhang, B Ding *Sampling error profile analysis (SEPA) for model optimization and model evaluation in multivariate calibration*. Journal of Chemometrics(2017)