Machine Learning pour le provisionnement des sinistres corporels -

Journées IARD 2018

29 mars 2018

kpmg.fr

maif.fr

Sommaire

1. Provisionnement en assurance Non Vie

1. Provisionnement en assurance Non Vie

A. Problématique du provisionnement

La recherche de la meilleure estimation du coût ultime des sinistres reste une préoccupation forte, notamment pour les garanties corporelles :

- Les garanties à **duration plus longue** constituent à la fois :
  - Une proportion importante des provisions au bilan. Par exemple, chez la MAIF, plus de deux tiers des PSAP sont liées aux garanties corporelles.
  - Un facteur d'incertitude important compte-tenu de la volatilité du coût des différents sinistres

- Les **enjeux** de leur bonne estimation sont multiples :
  - Tarification (en moyenne marché pour les 4 roues, la RC représente environ 1/3 de la prime).
  - Besoin en capital dans les calculs de la formule standard (une hausse de 5% des BE de sinistre se traduit par environ 3% de SCR souscription non vie).
  - Sensibilité des calculs solvabilité 2 aux cadences de liquidation.

- Dans ce contexte, des **méthodes classiques** existent, telles que Chain-Ladder, Bornhuetter-Ferguson (…), ainsi que des méthodes statistiques plus avancées (GLM, bootstrap, …).

Mais la masse des données disponibles est encore très peu exploitée.

- Les caractéristiques de chacun des sinistres ne sont pas (ou très peu) exploitées.
- Dans ce but, nous avons cherché à développer de nouvelles méthodes, basées sur des algorithmes de Machine Learning dont le déploiement est testé par ailleurs sur d'autres problématiques assurancielles.

1. Provisionnement en assurance Non Vie

B. Méthodes de provisionnement

Les méthodes sur données agrégées présentent des avantages, mais également de nombreuses faiblesses :

Avantages

- Robustesse (surtout si historique volumineux)
- Bien connues dans le milieu actuariel
- Relativement faciles à mettre en place et peu coûteuses (en temps et en puissance de calcul)
- Faciles à interpréter et à comprendre par les cellules décisionnaires en entreprise

Inconvénients

- Non exploitation statistique d'un nombre important d'informations
- Prédiction de l'ultime au niveau individuel impossible, alors que des utilisations de ces prédictions seraient envisageables :
  - Détection des situations de sous/sur provisionnement, de cas atypiques, de segments en dérive,
  - Réassurance non proportionnelle.
- Hypothèses parfois non vérifiées, notamment dans la méthode Chain-Ladder
- Conversion d'un référentiel réglementaire à un autre nécessitant généralement des proxys
- Pas de lien entre provisionnement et tarification

2. Données et Modélisation

2. Données et modélisation

A. Présentation des données utilisées

Base de données réelles

Nous utilisons une base de données réelles de sinistres individuels, fournie par la MAIF, ayant les caractéristiques suivantes :

- Garantie Responsabilité Civile Corporelle pour des contrats Auto
- Années de survenances 2004 à 2016
- Dépenses vues à chaque exercice comptable
- Variables relatives à l'assuré (équipement, conducteur principal,…) , au véhicule de l'assuré (classification)
- Variables relatives au sinistre (responsabilité, nombre de victimes, taux AIPP, typologie macro des traumatismes, cours d'appel, …)

Variables et traitements

Divers traitements ont été effectués afin de :

- traiter les valeurs manquantes ou aberrantes (très peu de cas),
- recoder certaines variables (par exemple, réduction du nombre de catégories de véhicules),
- créer de nouvelles variables à partir de celles existantes (pour des besoins de modélisation).

Une première sélection de variables a été effectuée par « jugement d'expert » afin d'éliminer les variables non pertinentes (N° sinistre…).

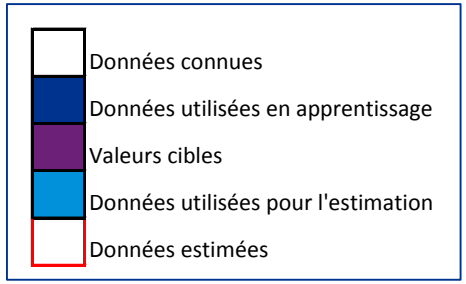
Les variable que nous allons chercher à prédire sont les incréments de dépenses. Nous ne modéliserons que les sinistres RBNS.

Une deuxième sélection de variables sera effectuée par la suite, pour chaque modèle.

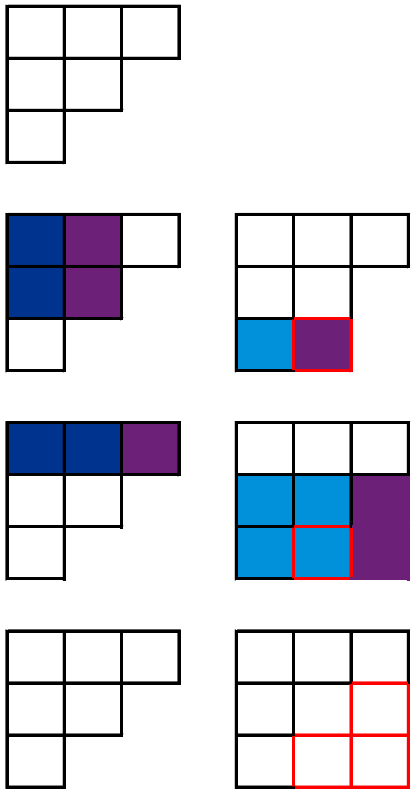
Cette sélection servira à ne conserver que les variables jugées comme « importantes » pour chaque modèle

2. Données et modélisation

B. Modélisation



Idée initiale : nous souhaitons projeter notre triangle de dépenses, afin d'obtenir le triangle de prévisions « complété ».
Nous allons procéder année de développement par année de développement, de façon similaire à un Chain-Ladder.



1. On part des données individuelles : chaque carré représente ici l'ensemble des sinistres survenus une année i, et développés une année j.

2. On suit les étapes suivantes :
   i. On utilise les sinistres ayant au moins deux années de développement ;
   ii. On apprend à notre algorithme à estimer les montants des sinistres à la deuxième année de développement, d'une part à partir des montants à la première année de développement, d'autre part à partir des données individuelles ;
   iii. On estime le montant de la deuxième année de développement pour les sinistres dont nous ne connaissons que la première année de développement.

3. On progresse ainsi de suite, en utilisant les données observées ainsi que les données estimées au pas précédent.

4. On obtient au final le triangle des estimations complété, et donc l'ultime estimé pour chaque sinistre.

Nous aurons ainsi un algorithme différent (au sens des paramètres et de l'apprentissage) pour chaque année de développement.

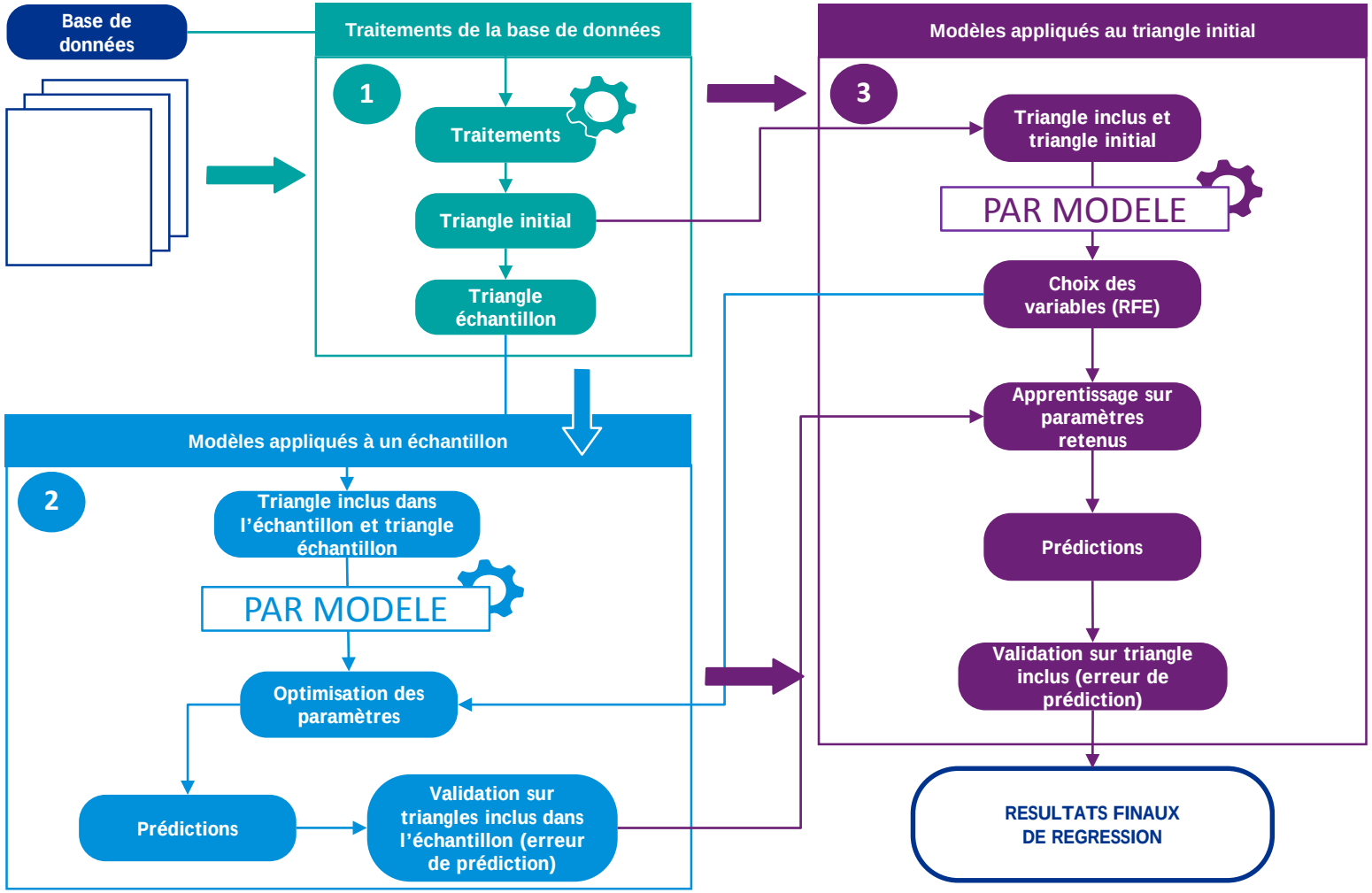
2. Données et modélisation

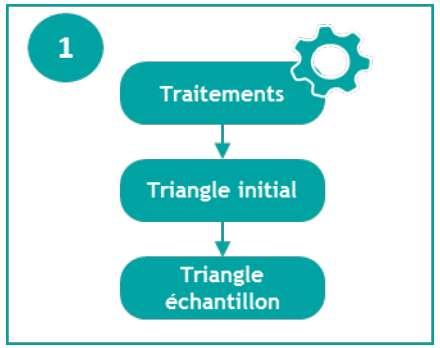
Le détail des différents blocs est présenté dans les slides suivantes

B. Modélisation

Nous allons séquencer la modélisation en 3 étapes :

1. Traitements de la base de données
2. Modèles appliqués à un échantillon
3. Modèles appliqués au triangle initial





2. Données et modélisation

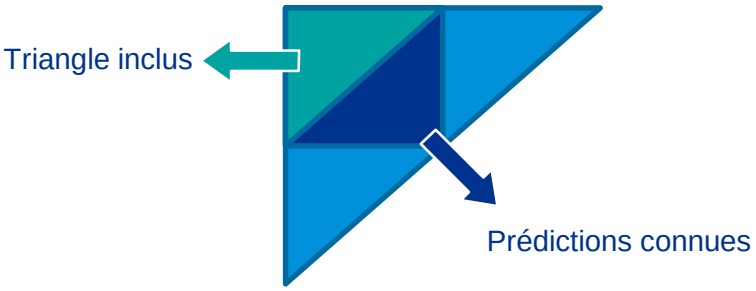
B. Modélisation

1. Traitements de la base de données

Nous séparons nos données en « sous triangles » comme suit :

- Séparation entre « triangle initial » et « triangle échantillon » :
  - Le triangle initial correspond au triangle sans aucune modification
  - Le triangle échantillon correspond à un triangle obtenu par une technique d'échantillonnage stratifié sur la variable « grave », qui représentera 20% du triangle initial.
  - L'intérêt de ce triangle est que les calculs s'y feront plus rapidement, vu qu'il est moins volumineux que le triangle initial.

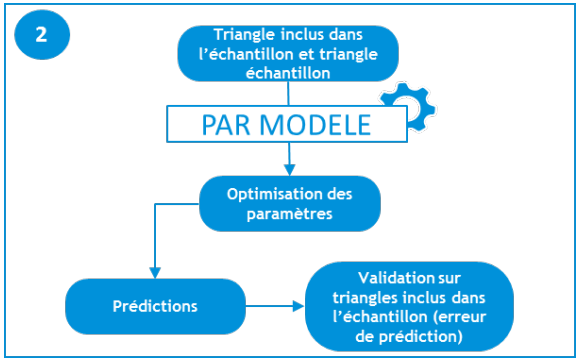
- Séparation entre « triangle total » et « triangle inclus »
  - Le triangle inclus correspond au « rectangle » inclus dans notre triangle. Il permettra de comparer les prédictions à des données observées pour valider les modèles.



Triangle	Nombre de lignes	Périmètre	Construction	Utilité
Initial	255 900	2004 - 2016	Ensemble des données initiales	-
Echantillon (20%)	51 180	2004 - 2016	Echantillonage stratifié	Test de paramètres
Inclus	145 671	2004 - 2010	Périmètre 2004 - 2010	Validation des modèles Comparaison à des données réelles
Inclus dans l'échantillon (20%)	29 286	2004 - 2010	Périmètre 2004 - 2010 de l'échantillon stratifié	Validation des modèles Comparaison à des données réelles

2. Données et modélisation

B. Modélisation



2. Modèles appliqués à un échantillon du triangle

(*) Ces éléments seront décrits par la suite

Les étapes suivantes seront réalisées par modèle (i.e. par algorithme)

- Nous allons utiliser les algorithmes de Machine Learning suivants :
  - Arbre de régression et forêts aléatoires (package « ranger » de R)
  - Réseaux de neurones : perceptron multi-couches (MLP) à plusieurs couches cachées (package « RSNNS » de R)

- Les modèles seront tout d'abord appliqués sur le triangle inclus pour validation grâce à la comparaison aux données réelles et à un Chain-Ladder, avant d'être étendues au triangle initial.

a. Choix des variables par Recursive Feature Elimination (RFE) (*)

- Une première étape consiste à choisir qu'elles seront les variables explicatives utilisées **pour chaque modèle**, à partir du triangle « Initial ».

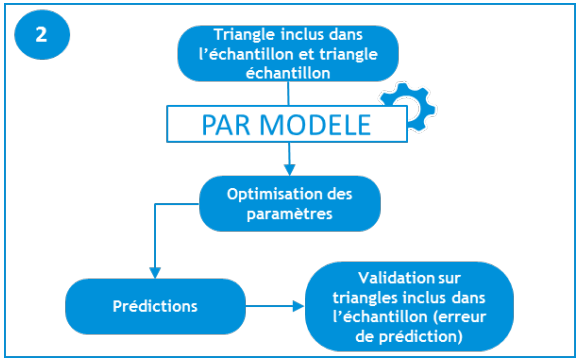
b. Optimisation des paramètres

- Cette étape consiste à trouver, pour chaque modèle, les paramètres optimaux. Nous utilisons une méthode dite « **recherche par quadrillage** », combinée à une **validation croisée(*)** à 10 échantillons.
- La combinaison de paramètres ayant obtenu le plus petit RMSE(*) sera celle dite « optimale », et conservée pour la prédiction.
- Nous procédons de la façon suivante, **pour chaque année de développement** :
  1. Choix des paramètres (dépendant de l'année de développement) à optimiser
  2. Subdivision des données en 10 échantillons pour l'utilisation de la validation croisée
  3. Prédiction avec validation croisée pour chaque combinaison de paramètres
  4. Comparaison des erreurs individuelles (RMSE) obtenues pour chaque combinaison

Certains paramètres seront communs à toutes les années de développement, pour ceux-ci nous comparerons les RMSE obtenus lors du dernier développement.

2. Données et modélisation

B. Modélisation



2. Modèles appliqués à un échantillon du triangle

c. Prédictions

(*) Ces éléments seront décrits par la suite

- Une fois les variables explicatives et les paramètres optimaux obtenus, nous pouvons réaliser les prédictions.

- Les variables cibles sont :
  - Les incréments de dépenses (pour l'année N : dépense cumulée[N] – dépense cumulée[N-1])

- Seuls les sinistres non-clos durant leur premier développement sont projetés. Les dépenses des sinistres clos sont gardées constantes dans les prédictions.

d. Validation et comparaison

- Nous cherchons à valider les modèles sur le « triangle inclus » du « triangle échantillon ».

- Nous comparons les prédictions obtenues, par modèle, aux données réelles et à un Chain-Ladder. Les métriques d'erreur retenues sont :
  - L'erreur relative globale (*)
  - Root-Mean Squared Error (RMSE) (*)
  - Mean Absolute Error (MAE) (*)

- Une fois les modèles validés, nous répétons les étapes B) à D) sur le « triangle échantillon » total.

A noter : les paramètres optimaux obtenus sur le « triangle échantillon » seront réutilisés sur le « triangle initial ».

2. Données et modélisation

Un point sur : les métriques d'erreurs

Nous allons utiliser les métriques d'erreurs suivantes pour comparer les résultats obtenus aux données réelles.

Erreur de prédiction globale : Erreur relative absolue

- Correspond à l'erreur de prédiction : |(prédit-réel)/réel|
- C'est une métrique d'erreur « agrégée », calculée directement sur la valeur agrégée par développement et survenance

Les deux métriques suivantes sont des métriques d'erreurs individuelles :

RMSE (Root Mean Square Error)

- Pour $X = (x_1, \dots, x_n), Y = (y_1, \dots, y_n)$, on a : $\text{RMSE}(X,Y) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - y_i)^2}$

MAE (Mean Average Error)

- Pour $X = (x_1, \dots, x_n), Y = (y_1, \dots, y_n)$, on a : $\text{MAE}(X,Y) = \frac{1}{n}\sum_{i=1}^{n}|x_i - y_i|$

Comparaison des métriques individuelles

- Similarités :
  - Le MAE et le RMSE expriment chacune une erreur moyenne de prédiction dans l'unité de la variable d'intérêt.
  - Les deux métriques sont comprises entre 0 et l'infini, 0 représentant une erreur nulle
  - Elles sont toutes les deux indifférentes à la direction des erreurs
- Différences :
  - Le fait d'élever les différences au carré avant d'en faire la moyenne entraîne que le RMSE donne un poids relativement plus important aux grandes erreurs : ainsi, le RMSE sera plus utile si on veut pénaliser fortement les grands écarts.
  - Si nous ne voulons pas pénaliser plus fortement les grandes erreurs, le MAE sera plus intéressant.

2. Données et modélisation

Un point sur : la validation croisée

Intérêt

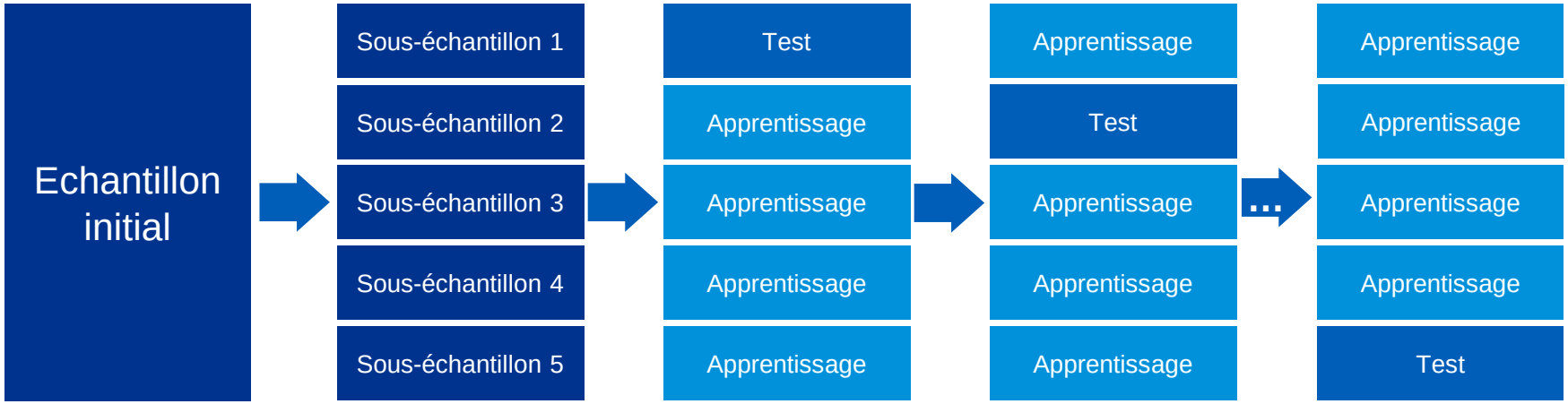
- La validation croisée permet d'estimer la fiabilité d'un modèle par une technique d'échantillonnage, et réduit :
  - le biais « d'échantillonnage » (biais causé par la séparation aléatoire en échantillons d'apprentissage et de test)
  - Le risque de sur-apprentissage.

Principe

1. On divise l'échantillon initial en k sous-échantillons.
2. On sélectionne un échantillon parmi les k, qui servira d'échantillon test. Les (k-1) autres échantillons sont utilisés pour l'apprentissage. On calcule l'erreur d'apprentissage sur l'échantillon test.
3. On répète le processus pour utiliser chaque sous-échantillons exactement une fois.
4. On obtient l'erreur de prédiction finale comme moyenne des erreurs de prédiction sur chaque sous-échantillon

- Nous utiliserons une validation croisée à 10 sous-échantillons.

- Exemple de la validation croisée avec 5 sous-échantillons :



2. Données et modélisation

Un point sur : le RFE

Choix des variables explicatives

- Nous souhaitons réaliser une sélection des variables, pour ne conserver que celles considérées comme les plus « **importantes** » pour les modèles.

- L'intérêt de cette sélection de variables est double :
  - Réduction du temps de calcul pour la phase d'apprentissage et pour la phase de test
  - Eliminer le « bruit » dans les prédictions pouvant être causé par des variables ayant peu d'impact

Importance des variables

- Nous allons utiliser des fonctions pour évaluer l'importance des variables. Ces fonctions se divisent en deux groupes :
  - Celles qui utilisent l'information issue des modèles
  - Celles indépendantes des modèles

- L'avantage d'une approche basée sur un modèle est qu'elle est liée à la performance obtenue par le modèle, et peut inclure la structure des corrélations entre les prédicteurs pour le calcul des importances.

- Pour les modèles que nous utilisons, nous avons les algorithmes suivants d'importances de variables :
  - Pour les forêts aléatoires : « décroissance moyenne en erreur de prédiction »
  - Pour les réseaux de neurones : algorithme de Garson

RFE

- L'algorithme va procéder de manière itérative, par modèle :
  - Avec des paramètres « par défaut », en utilisant toutes les variables, calculer un RMSE avec une validation croisée à 10 sous-échantillons
  - Calculer l'importance des variables par une méthode liée au modèle, et enlever la variable la moins importante
  - Recommencer jusqu'à ne conserver que la variable la plus importante
  - Finalement, retenir les variables liées au modèle avec le RMSE le plus faible

3. Algorithmes d'apprentissage statistique

3.A Forêts aléatoires (FA)

3. Algorithmes d'apprentissage statistique

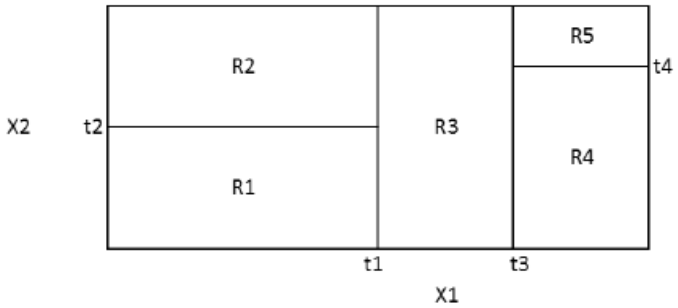
A. Forêts aléatoires

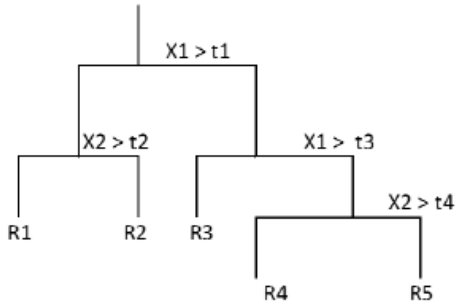
Qu'est-ce qu'un arbre de décision ?

- Un arbre de décision est un algorithme de Machine Learning, utilisé pour des tâches de **classification** ou de **régression**
- Il existe plusieurs type d'algorithmes d'arbre de décision, nous allons étudier l'algorithme **CART**

De quoi est composé un arbre de décision (CART) ?

- Objectif : création de parties binaires récursives de l'espace, de façon successives
- Règle de segmentation, qui permet de choisir les variables contributives à la prédiction
- Nœuds, Feuilles, Profondeur
- Critère d'arrêt : l'algorithme continue jusqu'à un critère d'arrêt (par exemple, nombre minimum d'individus dans les feuilles terminales)





Représentation d'un partitionnement d'un espace bidimensionnel, et arbre de régression associé. C'est un arbre à 4 nœuds, 5 feuille, de profondeur 4.

Un arbre de décision seul est souvent très instable, nous allons utiliser une méthode d'aggrégation afin d'obtenir des résultats plus robustes.

3. Algorithmes d'apprentissage statistique

A. Forêts aléatoires (FA)

Bagging et Forêts aléatoires (FA)

- Le **bagging** (*Bootstrap aggregation*) est un algorithme d'aggrégation de modèles.

- Cet algorithme a pour objectif de moyenner les prévisions de plusieurs modèles indépendants, de façon à réduire :
  - **la variance de prévision**
  - **le risque de surapprentissage**
  - **le risque lié aux optima locaux**

- De façon schématique, le *bagging* identifie le modèle « moyen » approchant le mieux les données d'intérêt. On mesure la stabilité du modèle grâce à l'erreur *out-of-bag*, à partir d'échantillons bootstraps.

- Une forêt aléatoire est une agrégation d'arbres de décisions par une méthode de bagging, qui comporte en plus une étape de tirage aléatoire.

Calibrage des paramètres sur le triangle échantillon

- Le **nombre d'arbres** par forêt sera fixé à 500, ce qui est suffisant pour assurer la stabilité des résultats

- Le principal paramètre calibré sera : **le nombre de variables testées à chaque segmentation** (paramètre *mtry*), ce paramètre dépendra de l'année de développement.

- Nous effectuerons des tests de sensibilité sur les paramètres :
  - Critère de segmentation (*variance, extratrees, maxstat*)
  - Nombre d'individus dans les feuilles terminales (*min.node.size*), qui est assimilable à un critère d'arrêt (nous aurons des arbres plus profond pour un *min.node.size* petit)

3. Algorithmes d'apprentissage statistique

A. Forêts aléatoires (FA)

Nombre de variables testées à chaque segmentation (*mtry*)

- Nous obtenons les résultats suivants :

Dépenses

mtry\dev	2	3	4	5	6	7
1	0,04106	0,01692	**0,00911**	0,00466	**0,00442**	**0,01253**
2	0,03800	0,01605	0,00920	**0,00465**	0,00442	0,01314
3	0,03674	**0,01593**	0,00931	0,00470	0,00450	0,01402
4	0,03650	0,01603	0,00939	0,00480	0,00461	0,01423
5	**0,03645**	0,01609	0,00957	0,00484	0,00471	0,01472
6	0,03667	0,01626	0,00957	0,00490	0,00477	0,01496
7	0,03680	0,01634	0,00988	0,00493	0,00486	0,01511
8	0,03695	0,01643	0,00969	0,00498	0,00490	0,01561
9	0,03705	0,01655	0,00964	0,00510	0,00494	0,01528
10	0,03702	0,01658	0,00964	0,00514	0,00505	0,01507

mtry retenu	5	3	1	2	1	1

RMSE en phase d'apprentissage, pour chaque développement des dépenses.

- Nous remarquons notamment que le modèle retient plus de variables testées à chaque segmentation pour les premiers développements que pour les développements ultérieurs.

- Forêts utilisées : 500 arbres, *splitrule = variance, min.node.size = 1*

3. Algorithmes d'apprentissage statistique

A. Forêts aléatoires (FA)

Règle de segmentation (*variance, extratrees, maxstat*)

- Nous utilisons le paramètre *mtry* optimisé, nous allons maintenant tester de façon simultanée la règle de segmentation ainsi que le nombre d'individus dans les feuilles terminales :

Dépenses

Règle\mns	1	5	10	20
Variance	22 809	22 926	**22 701**	23 094
Extratrees	23 733	23 710	23 758	23 736
Maxstat	25 426	25 417	25 424	25 408

RMSE en comparaison des données réelles, année de développement 7

- La règle de segmentation *variance* donne les meilleurs résultats.

- Ne pas prendre la profondeur maximale pour les dépenses semble donner de meilleurs résultats.

- (*) Les règles de segmentations correspondent respectivement à :
  - Variance : réduction de la variance dans les sous-arbres
  - Extratrees : algorithme extremely randomized trees (Geurts et al. 2006)
  - Maxstat : sélection par maximisation de statistiques de rang (Wright et al. 2016)

3.B Réseaux de neurones (PMC)

3. Algorithmes d'apprentissage statistique

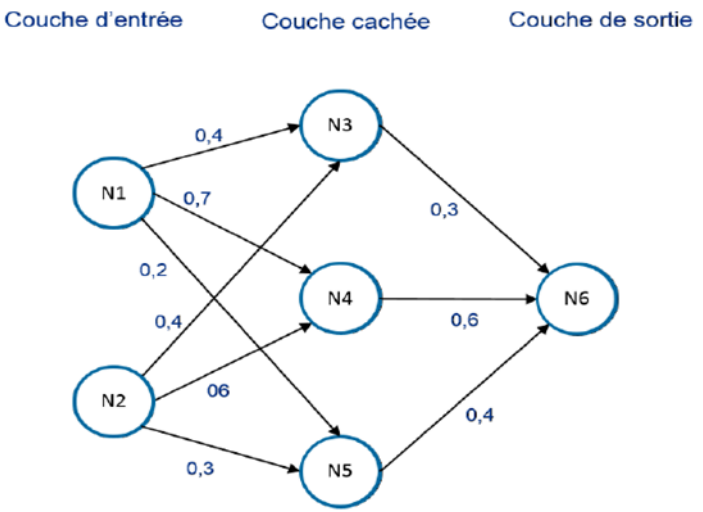
B. Réseaux de neurones (PMC)

Qu'est-ce qu'un réseau de neurones PMC ?

- Un réseau de neurones est un **graphique orienté pondéré**.
- Il est composé de nœuds, appelés **neurones**. Les neurones sont connectés entre eux, et sont organisés en plusieurs niveaux appelés **couches**.
- Une **matrice de poids synaptiques** pondère la connectivité entre les neurones, et représente la transmission d'information entre ceux-ci.

De quoi est constitué un réseau de neurones PMC ?

- Un réseau de neurones est composé des éléments suivants :
  - Plusieurs **nœuds**, répartis en une couche d'entrée, une couche de sortie et d'une couche cachée minimum.
  - Une **matrice de poids synaptiques**, représentant la connectivité du réseau.
  - Une **fonction d'entrée**, généralement commune à tous les neurones d'une même couche, et qui lie les poids aux nœuds suivants.
  - Une **fonction d'activation**, qui estime l'activité du nœud.
  - Généralement, d'un **algorithme d'apprentissage**, qui permet de modifier les poids synaptiques en fonction des données présentées.
  - Des **seuils d'activation** peuvent aussi être présents.



Représentation d'un réseau de neurones à une couche cachée, à 6 neurones.

Chaque neurone de la couche d'entrée représente soit une variable continue, soit une modalité d'une variable catégorielle.

De façon schématique, les flèches indiquent dans quel sens évolue l'information, et les poids associés la quantité d'information passant entre les neurones.

3. Algorithmes d'apprentissage statistique

B. Réseaux de neurones (PMC)

Calibrage des paramètres sur le triangle échantillon

- Le nombre d'itérations sera fixé à 500, afin de réduire le risque de surapprentissage. Une étape d'aggrégation par **bagging** permettra par la suite de rendre les modèles plus robustes.
- Le principal paramètre calibré sera : le nombre de neurones par couche cachée (pour deux couches cachées). Ce paramètre dépendra de l'année de développement.
- Nous effectuerons des tests de sensibilité sur les paramètres :
  - Algorithme d'apprentissage (*Rétropropagation du Gradient de l'erreur, Resilient Propagation, Quickprop, SCG*)
  - Fonction d'activation (Logistic, TanH, Linear, Exponential)
  - Présence ou non d'une troisième couche cachée
  - Taux d'apprentissage
  - Momentum

Réseaux de neurones utilisé :

- Itérations maximum = 500
- Fonction d'activation = Logistic
- Algorithme d'apprentissage = Rétropropagation du gradient de l'erreur

Dépenses

Nombre de neurones dans la 1ère couche cachée	Nombre de neurones dans la 2ème couche cachée	Développement					
		2	3	4	5	6	7
1	1	0,05168	0,01478	0,01194	0,00519	0,00553	0,02023
1	2	0,04929	0,01407	0,01138	0,00523	0,00617	0,01901
1	3	0,04898	0,01407	0,01126	0,00519	0,00560	0,01936
2	1	0,04870	0,01457	0,01211	0,00525	0,00553	0,01996
2	2	0,04784	0,01352	0,01116	0,00520	0,00569	0,01895
2	3	0,04777	0,01398	0,01205	0,00520	0,00626	0,01834
3	1	0,04829	0,01334	0,01171	**0,00513**	0,00559	0,01796
3	2	0,04570	0,01330	0,01143	0,00548	0,00555	0,01705
3	3	0,04600	0,01343	0,01148	0,00606	0,00555	**0,01703**
4	1	0,04193	0,01336	0,01146	0,00519	0,00558	0,01929
4	2	0,04695	0,01324	0,01170	0,00519	**0,00551**	0,01785
4	3	0,04199	0,01327	0,01135	0,00527	0,00555	0,01762
5	1	**0,04179**	0,01330	0,01155	0,00521	0,00571	0,01967
5	2	0,04374	**0,01244**	**0,01105**	0,00521	0,00574	0,01779
5	3	0,04461	0,01296	0,01109	0,00526	0,00575	0,01801
Architecture retenue		**(5,1)**	**(5,2)**	**(5,2)**	**(3,1)**	**(4,2)**	**(3,3)**

RMSE en phase d'apprentissage, pour chaque développement.

3. Algorithmes d'apprentissage statistique

B. Réseaux de neurones (PMC)

Algorithme d'apprentissage et fonction d'activation (2/2)

- Nous testons maintenant simultanément deux autres algorithmes d'apprentissage et trois autres fonctions d'activation :

  - Algorithmes testés : *Standard Backpropagation, Resilient backpropagation, Scaled Conjugate Gradient*
  - Fonctions d'activations testées : Logistique, Tangente Hyperbolique, Identité, Exponentielle

Dépenses

Apprentissage\Activation	Logistic	TanH	Linear	Exponential
Std_Backpropagation	26 397	**24 295**	51 375	24 619
Rprop	35 368	24 317	35 116	30 173
SCG	25 699	40 338	140 829	25 678

- Nous obtenons les meilleurs résultats avec l'algorithme de rétropropagation de l'erreur, et comme fonction d'activation la tangente hyperbolique.
- Les très forts RMSE obtenus avec la fonction d'activation « Identité », ainsi qu'avec l'algorithme *SCG,* indiquent un sous-apprentissage : le réseau de neurones n'a pas convergé avec le nombre d'itérations donné.
- En regardant en détail les prévisions, nous remarquons que l'algorithme « SCG » à tendance à sous-estimer de façon systématique les dépenses, et ce même si nous augmentons le nombre d'itérations. Nous décidons donc de conserver l'algorithme de rétropropagation de l'erreur, et la fonction d'activation TanH.

Troisième couche cachée

- Nous testons maintenant l'effet d'une troisième couche cachée à notre réseau : dans la littérature, il est souvent décrit qu'une troisième couche cachée n'est généralement pas nécessaire et augmente le risque de surapprentissage

Dépenses

Nombre de couches cachées	RMSE
2	**24 295**
3	28 435

	Développement					
	2	3	4	5	6	7
Neurones 3è couche	1	2	1	1	1	2

- L'ajout d'une troisième couche cachée semble effectivement affaiblir le pouvoir prédictif de nos réseaux.

3. Algorithmes d'apprentissage statistique

B. Réseaux de neurones (PMC)

Taux d'apprentissage et momentum

- Nous testons maintenant simultanément le taux d'apprentissage et le momentum.

- Le taux d'apprentissage représente la vitesse à laquelle le réseau parcourt les données fournies. Un taux d'apprentissage trop fort pourrait empêcher le réseau de converger convenablement, un taux trop faible expose le réseau à un risque de surapprentissage.
- Le momentum représente une fraction de l'ancienne variation de poids, qui est introduite dans le calcul de la nouvelle variation. Cela permet d'éviter des problèmes d'oscillations, notamment quand la surface d'erreur a une zone minimale très réduite.

Dépenses

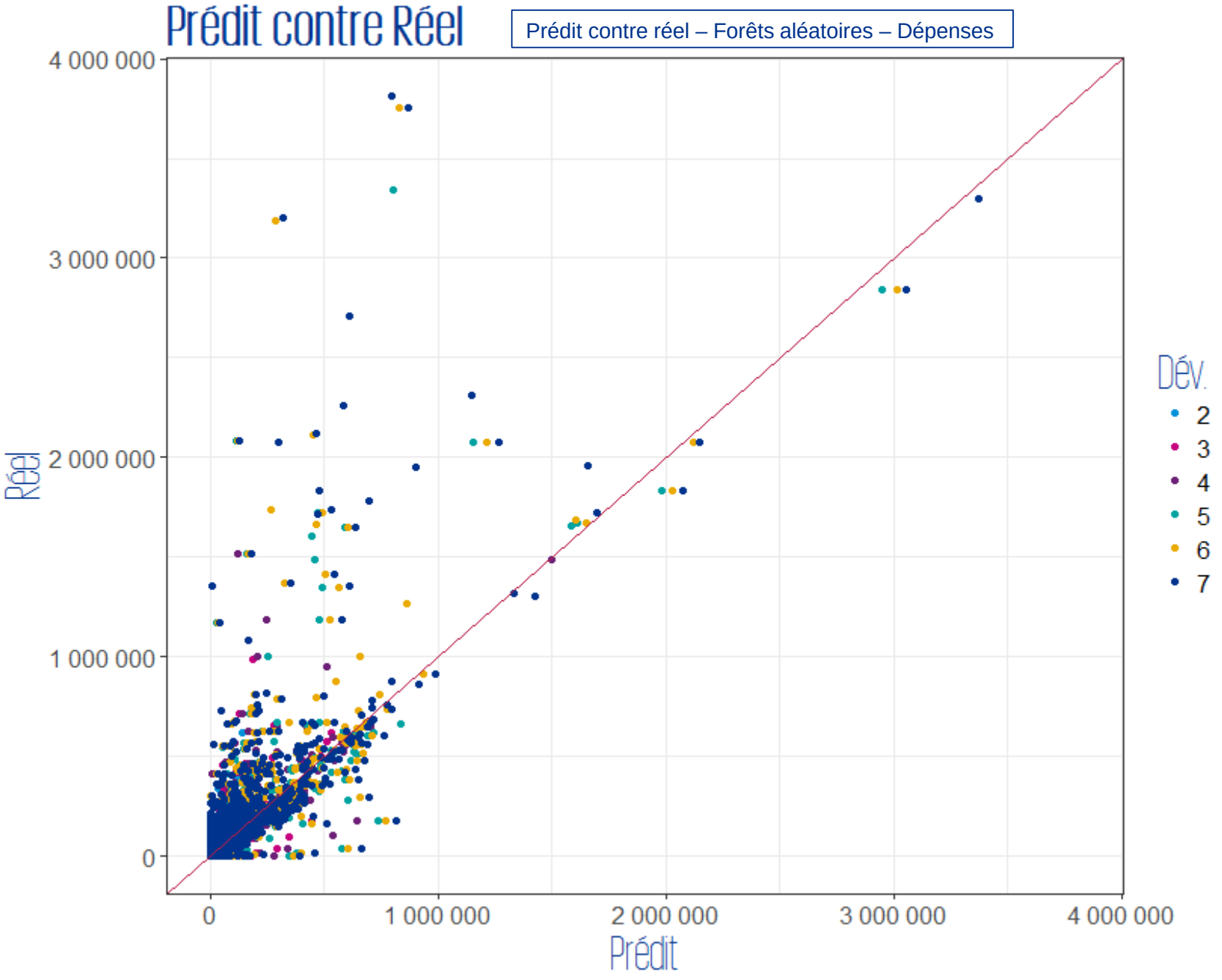
Tx d'apprentissage\Momentum	Moment 0	Moment 0.1	Moment 0.25	Moment 0.5
Taux d'apprentissage 0.1	**25 746**	26 837	29 821	33 941
Taux d'apprentissage 0.25	26 855	35 197	31 678	41 454
Taux d'apprentissage 0.5	33 118	42 741	32 355	30 851

- Nous obtenons un taux d'apprentissage relativement faible, à 0.1. Il apparait que le terme de momentum présente peu d'utilité pour la prédiction des dépenses.

Malgré tous nos tests de paramètres, les résultats des réseaux de neurones sont encore relativement instables. Ils seront rendus plus consistants grâce à une étape d'aggrégation par bagging, de 100 réseaux de neurones.

4. Résultats et comparaisons

4. Résultats et comparaisons



4. Résultats et comparaisons

Erreurs individuelles

- Nous comparons maintenant les modèles entre eux grâce aux erreurs individuelles, en vision « triangle » (en regroupant par année de survenance et par développement). Un code couleur permet d'afficher le modèle qui comme l'erreur la plus faible, et l'erreur correspondante :

FA vs PMC — FA PMC

RMSE

2004					
2005					9 322
2006				10 936	14 736
2007			11 945	24 668	35 117
2008		6 138	14 524	18 192	20 549
2009	6 886	14 328	24 220	26 351	32 171
2010	8 191	13 127	17 346	22 504	25 192	27 764

FA vs PMC — FA PMC

MAE

2004					
2005					482
2006				552	1 048
2007			845	1 391	2 046
2008		916	1 713	2 018	2 523
2009	1 587	2 302	2 926	3 220	3 770
2010	1 787	2 903	3 537	3 959	4 220	4 724

- Les forêts aléatoires semblent généralement commettre moins d'erreurs individuelles que les réseaux de neurones. Le MAE indique que sur les 3 dernières survenance, les réseaux de neurones commettent des erreurs moyennement plus faibles, mais commettent des erreurs plus importantes sur les grandes valeurs.

Coefficient de détermination (R^2)

- Le coefficient de détermination (R^2) est une mesure de la qualité de prédiction d'une régression linéaire. C'est le carré du coefficient de corrélation linéaire R.

FA vs PMC — FA PMC

R^2

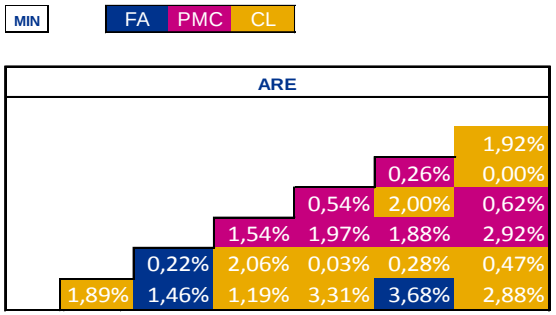
2004					
2005					0,94
2006				0,88	0,81
2007			0,89	0,71	0,60
2008		0,93	0,76	0,71	0,70
2009	0,80	0,57	0,67	0,68	0,63
2010	0,50	0,43	0,33	0,33	0,34	0,33

- Le meilleur ajustement semble être fait par les forêts aléatoires.

4. Résultats et comparaisons

Comparaison avec Chain Ladder (1/3)

- Nous comparons nos modèles avec un Chain Ladder grâce à l'erreur relative absolue :



- Globalement, sur le dernier développement, nous obtenons les résultats suivants :
  - Dépenses :
    - CL : +0,29%
    - Forêts aléatoires : +4,14%
    - Réseaux de neurones : +0,26%

4. Résultats et comparaisons

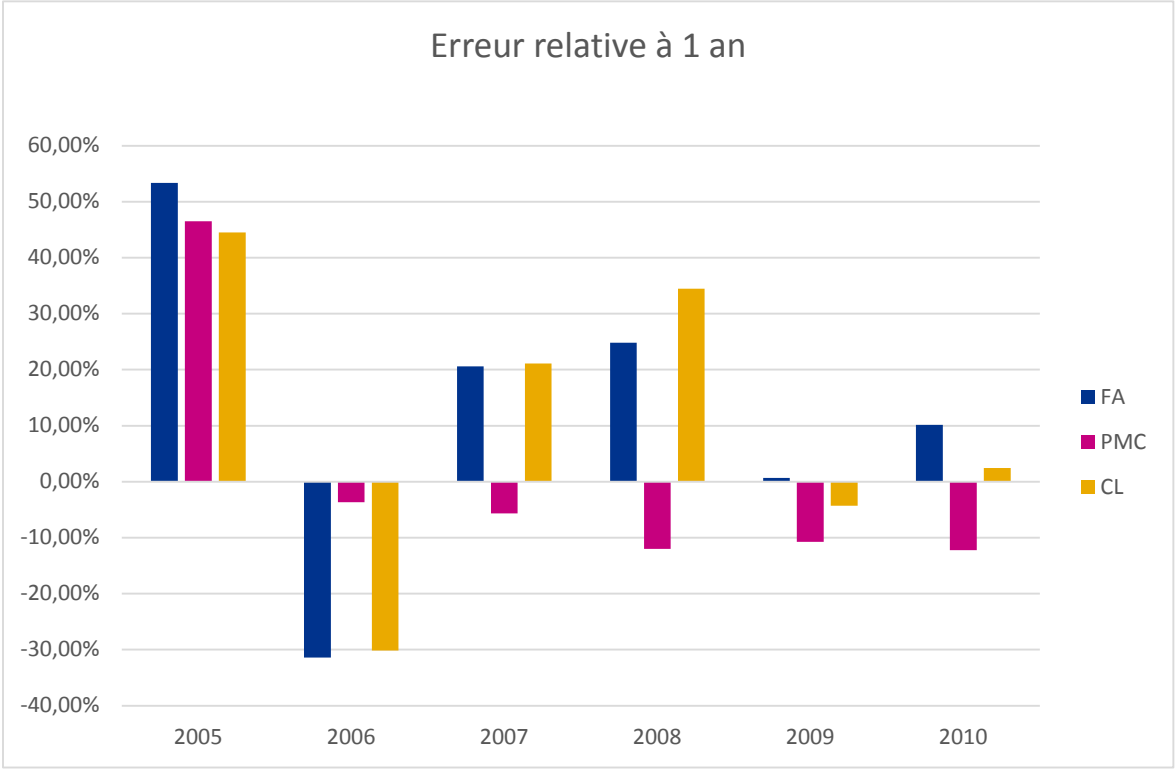
Comparaison avec Chain Ladder (2/3)

- Nous pouvons aussi analyser l'erreur relative à un an (première diagonale prédite moins dernière diagonale connue) et l'erreur relative à l'ultime (dernier développement moins dernière diagonale connue).

Dépenses

	RESERVE 1 AN (M€)						
	Réel	FA		PMC		CL	
		M€	Ecart %	M€	Ecart %	M€	Ecart %
2004	-	-	-	-	-	-	-
2005	4,76	7,30	53,3%	6,97	46,5%	6,88	44,5%
2006	7,21	4,94	-31,4%	6,94	-3,7%	5,04	-30,1%
2007	9,63	11,61	20,6%	9,09	-5,6%	11,66	21,1%
2008	10,69	13,34	24,8%	9,41	-12,0%	14,37	34,5%
2009	21,65	21,80	0,7%	19,32	-10,7%	20,73	-4,3%
2010	30,79	33,92	10,2%	27,03	-12,2%	31,55	2,5%
TOTAL	84,73	92,92	9,7%	78,77	-7,0%	90,22	6,5%

- Bien que le Chain-Ladder ait l'erreur relative à 1 an absolue la plus petite, nous remarquons que les erreurs sont globalement plus stables pour les réseaux de neurones; il y a aussi moins d'effets de compensation.



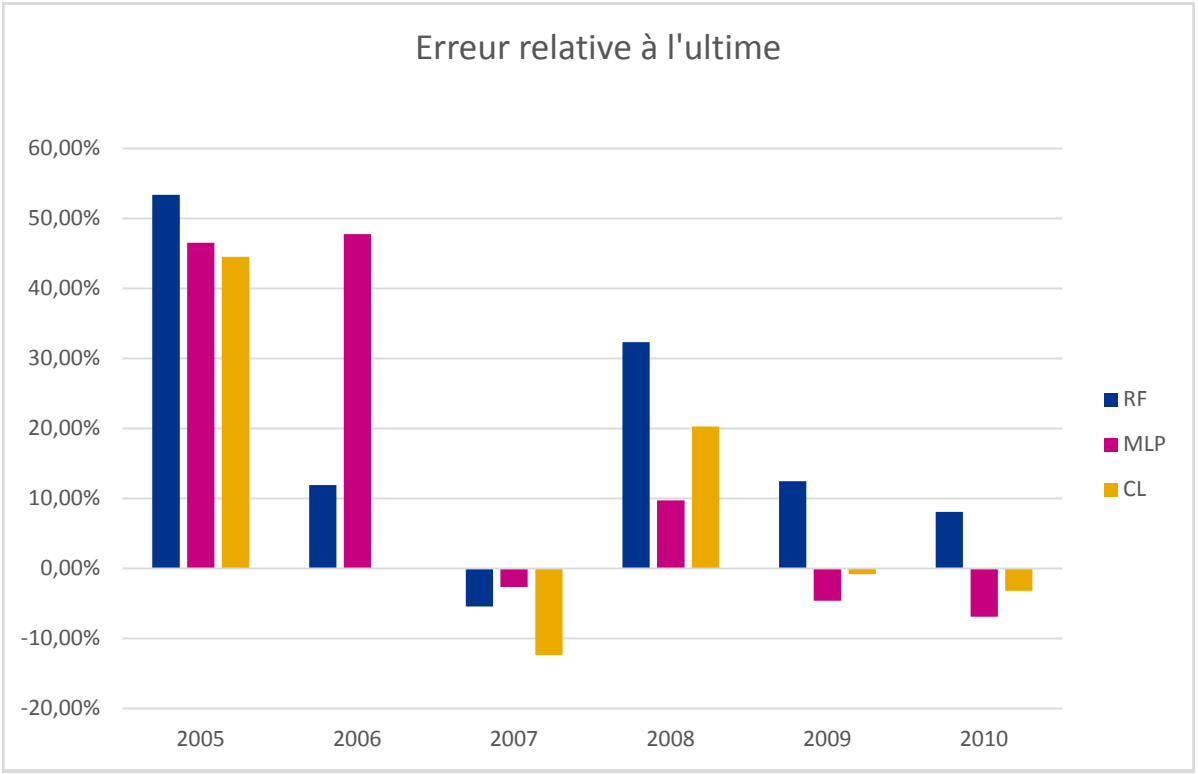
4. Résultats et comparaisons

Comparaison avec Chain Ladder (3/3)

- Nous pouvons aussi analyser l'erreur relative à un an (première diagonale prédite moins dernière diagonale connue) et l'erreur relative à l'ultime (dernier développement moins dernière diagonale connue).

Dépenses

	RESERVE ULTIME (M€)						
	Réel	FA		PMC		CL	
		M€	Ecart %	M€	Ecart %	M€	Ecart %
2004	-	-	-	-	-	-	-
2005	4,76	7,30	53,3%	6,97	46,5%	6,88	44,5%
2006	11,59	12,97	11,9%	17,12	47,7%	11,59	0,0%
2007	27,48	25,99	-5,4%	26,75	-2,7%	24,08	-12,4%
2008	31,02	41,04	32,3%	34,04	9,7%	37,30	20,2%
2009	55,61	62,52	12,4%	53,05	-4,6%	55,13	-0,9%
2010	84,51	91,34	8,1%	78,66	-6,9%	81,81	-3,2%
TOTAL	214,96	241,15	12,2%	216,58	0,8%	216,79	0,8%



- Le Chain-Ladder est comparable aux réseaux de neurones sur l'erreur relative à l'ultime.

5. Ouverture

Ouverture

Et maintenant ?

- Nous avons obtenu des prédictions individuelles concernant les dépenses automobiles RCC.
- Les résultats sur le triangle total sont en cours de consolidation. Les analyses des résultats au niveau individuels sont à approfondir.

A ce stade des travaux, les résultats ne sont pas significativement meilleurs au niveau global que ceux obtenus par Chain Ladder. Pour tenter d'améliorer ces résultats, deux options possibles :

- Poursuivre l'approche actuelle (prévision pas à pas des visions de fin d'exercice) en testant d'autres algorithmes (XGBoost, GLM avec pénalisation Ridge/Lasso…), en agrégeant différents modèles entre eux, par exemple avec l'algorithme d'agrégation COBRA, combiner des modèles provision / dépenses.
- Modifier l'approche et chercher par exemple à appréhender le provisionnement via la prédiction des durées de vies des sinistres et des instants où sont réalisées les dépenses.

Mais, ces résultats sont issus de provisions construites au niveau individuel.

- Il nous reste à bien nous approprier les résultats obtenus à ce stade (résultats à confirmer dans le temps, segmentation des sinistres les mieux prévus, travaux à enrichir pour les sinistres les moins biens estimés).
- Pour ensuite envisager une meilleure maîtrise du provisionnement, à travers :
  - Le renforcement du niveau de confiance dans le niveau global de charge ultime
  - La capacité à suivre plus finement le provisionnement.

Facilité de mise en œuvre de ces techniques ?

- Caractéristiques de l'ordinateur utilisé : CPU i7-6600U 2.70 GHz, RAM 8 GO
- Langage utilisé : R, packages : ranger, RSNNS, caret, ggplot…
- Temps de calcul :
  - Forêts aléatoires : paramètres ~ 1h, prédiction sur le triangle initial ~ 30mn
  - Réseaux de neurones : paramètres ~ 3h, prédictions sur le triangle initial (bagging) ~ 1 journée

Merci de votre attention

Questions ?

Annexes

Bibliographie

Sur le provisionnement individuel :

- Ragnar Norberg. Predictoin of outstanding liabilities in non-life insurance. Astin Bulletin, 29(01):5-25, 1999
- PD England and RJ Verral. Stochastic claims reserving in general insurance, 2007
- Katrien Antonio & Richard Plat. Micro-level stochastic loss reserving for general insurance, Scandinavian Actuarial Journal, 2014(7°:649-669, 2014

Sur le Machine Learning :

- The Elements of Statistical Learning. T. Hastie, R. Tibshirani, and J. Friedman. Springer Series in Statistics Springer New York Inc., New York, NY, USA, (2001)
- Data Analytics for Non-Life Insurance Pricing, Mario Wüthrich, SSRN Manuscript 2870308

Sur le Machine Learning pour le provisionnement individuel :

- Machine learning in individual claims reserving, Mario Wüthrich, to appear in Scandinavian Actuarial Journal.
- Non parametric individual claim reserving in insurance, C. Robert & M. Baudry, Chaire DAMI.
- Individual Claim Development with Machine Learning, B. Harej, R. Gächter, S. Jamal

Contacts

Damien Fabre Rudelle
KPMG Paris
Consultant, Actuariat Assurance
Tel: +33 1 55 68 63 77
Mob : +33 7 76 23 04 86
dfabrerudelle@kpmg.fr

Fabrice Oger
MAIF
Responsable Actuariat & Provisionnement
Tel : +33 5 49 73 71 94

fabrice.oger@maif.fr

kpmg.fr

maif.fr