

Mémoire présenté devant l'Université de Paris-Dauphine
pour l'obtention du Certificat d'Actuaire de Paris-Dauphine
et l'admission à l'Institut des Actuaires

le

Par : Thomas ROUER

Titre : Estimation de la densité de provisions non-vie ligne à ligne par réseaux de neurones

Confidentialité : Non Oui (Durée : 1 an 2 ans)

Les signataires s'engagent à respecter la confidentialité ci-dessus

*Membres présents du jury de l'Institut
des Actuaires :*

Entreprise :
Nom : KPMG France
Signature :



*Membres présents du Jury du Certificat
d'Actuaire de Paris-Dauphine :*

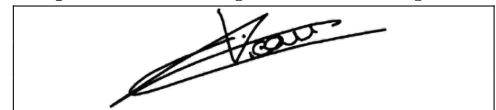
Directeur de Mémoire en entreprise :
Nom : Camille VIEAU
Signature :



*Autorisation de publication et de mise en ligne sur un site de diffusion de documents
actuariels (après expiration de l'éventuel délai de confidentialité)*

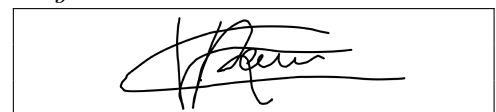
Secrétariat :

Signature du responsable entreprise



Bibliothèque :

Signature du candidat



Résumé

Le cycle de production inversé des produits d'assurance oblige les assureurs à estimer les paiements futurs aux titres de leurs engagements passés. Simples d'utilisation et d'interprétation, les méthodes historiques sur les données agrégées sont encore utilisées par les assureurs. Cependant, la croissance des données disponibles associée à l'amélioration de la capacité de calcul des ordinateurs, donne la possibilité aux assureurs d'utiliser des modèles ligne à ligne, adaptés à la quantité de données recueillies.

Ce mémoire a pour objectif la mise en place d'un modèle de provisionnement ligne à ligne utilisant les réseaux de neurones profonds. D'une part, les données temporelles imposent l'utilisation de modèle adapté (LTSM). D'autre part, les normes nécessitent le calcul d'un quantile et donc la gestion d'un aléa (réseau de neurones bayésien et distribution de sortie).

Un modèle, dérivé de celui de Kuo (KUO, 2020), composé d'un LSTM encoder et décoder associé à un réseau de neurones multi-couche bayésien a été implémenté sur une base de responsabilité civile médicale. La prédiction de la charge ultime a été comparée à celle de Chain-Ladder et aux résultats d'experts provisionnement. Un processus de *Backtesting* vient compléter les tests effectués.

Une étude fine des causes de la différence d'ordre de grandeur entre les résultats prédits et les comparaisons a été effectuée. Plusieurs limites ont ainsi pu être identifiées et des pistes d'améliorations ont été proposées. Ce mémoire, de par son caractère explicatif et ses limites transgressant le cadre des réseaux de neurones, permet d'étayer la discussion sur les réflexions et possibilités des modèles de provisionnement ligne à ligne.

Mots-clés : Assurance non-vie, Provisionnement individuel / ligne à ligne, Réseaux de neurones, LSTM, Réseau bayésien, Distribution/Densité.

Abstract

The inverted production cycle of insurance products requires insurers to estimate future payments for past liabilities. Easy to use and wellknown, historical methods on aggregate data are still used by insurers. However, the growth of available data associated with the improvement of the computing capacity, gives the possibility to use adapted models to the quantity of collected data : the micro-reserving models.

The objective of this paper is to implement a micro-reserving model using deep neural networks. On the one hand, the temporal data impose the use of an adapted model (LSTM). On the other hand, the regulation requires the calculation of a quantile and thus the management of the risk taken with the use of a Bayesian neural network and an output log-normal distribution.

A model, derived from Kuo's (Kuo, 2020), has been implemented on a medical liability database. The model is composed of an LSTM encoder and decoder associated with a Bayesian multi-layer neural network. The prediction of the ultimate has been compared to Chain-Ladder and to the results of reserving experts. A process of *Backtesting* completes the tests performed.

A detailed study of the causes of the difference in order of magnitude between the predicted results and the comparisons was carried out. Several limitations were identified and improvements were proposed. This paper, because of its explanatory character and its limits transgressing the framework of neural networks, contribute to support the discussion on the reflections and possibilities of the micro-reserving models.

Keywords : P&C insurance; Micro-reserving, Neural Network, LSTM, Bayesian neural network, Distribution / density .

Note de Synthèse

Contexte et enjeux

Les contrats d'assurances ont la particularité d'avoir un cycle de production inversé, contraignant les assureurs à déterminer le prix de vente du contrat avant d'en connaître les coûts réels. Cependant, l'assureur se doit d'être en capacité de régler le montant du sinistre s'il survient. Pour garantir sa capacité de règlement des sinistres futurs, l'assureur va constituer des provisions, avec notamment la provision pour sinistre à payer (PSAP).

La PSAP, destinée à couvrir les paiements à venir pour les sinistres déjà survenus, est estimée actuellement par des méthodes se basant sur des données agrégées tel que Chain Ladder et Bornhuetter-Ferguson. Bien maîtrisées et documentées, ces méthodes historiques sont rapides à mettre en place et nécessitent un temps de calcul très faible. Cependant, l'utilisation de données agrégées par année de survenance crée une forte perte d'information, dont notamment les caractéristiques individuelles du sinistre. Dans un monde où une grande puissance de calcul est disponible à présent sur l'ensemble de nos ordinateurs, de nouvelles méthodes dites "ligne à ligne", utilisant l'ensemble des données, sont apparues. L'objectif de recherche de ces nouvelles méthodes vise à challenger les méthodes agrégées afin de gagner en compréhension et en précision.

Ce mémoire a permis l'implémentation d'un modèle de provisionnement ligne à ligne utilisant les réseaux de neurones profonds. Nous concentrerons l'estimation sur la provision pour sinistres reportés insuffisamment provisionnés IBNER (*incurred but not enough reported*).

Une autre contrainte, a été l'obtention d'une densité de la provision. En effet, lors du calcul des réserves en norme française, un quantile des IBNER est utilisé afin d'ajouter une marge de risque prudente. Sous la norme Solvabilité 2, le calcul du risque de réserve par modèle interne peut utiliser le quantile 99,5 de la distribution des réserves. En IFRS 17 aussi, le calcul de la densité des réserves sert à calibrer l'ajustement pour risque de sorte que la somme de la meilleure estimation et de l'ajustement pour risque soit égal à un quantile de la distribution des réserves. Le quantile étant ici choisi en fonction de l'aversion au risque de l'assureur.

Dans la lignée des mémoires de CHARTREL, 2022 et PRIMEL, 2021, ce mémoire innove de par son utilisation des réseaux de neurones profonds pour obtenir une distribution.

Le modèle utilisé

Le modèle retenu est un modèle de provisionnement ligne à ligne. Une fois entraîné, le modèle prend donc en *input* un unique sinistre et doit prédire une distribution pour chaque année de développement inconnue du sinistre.

Pour effectuer la prédiction, la méthode des réseaux de neurones a été employée. Un neurone est une

somme pondérée d'*inputs* passée en argument d'une fonction d'activation (identité, sigmoïde, tanh). Un réseau de neurones consiste à utiliser les *outputs* de neurones comme inputs d'un neurone d'une couche suivante. L'apprentissage est effectué par application de l'algorithme de descente de gradient à une fonction de perte. Cette fonction de perte compare l'*output* attendu et l'*output* prédit par le modèle, avec par exemple la fonction RMSE.

De plus, les données de sinistres ligne à ligne ont 2 variables spéciales : la charge par année de développement et l'ouverture ou la fermeture du sinistre selon chaque année. Ces variables, dites "temporelles", nécessitent l'utilisation d'un réseau de neurones récurrents, permettant de garder en mémoire l'impact des valeurs des pas de temps passé pour la prédiction du pas de temps suivant.

Enfin, afin d'obtenir une densité des réserves, un aléa doit être ajouté au modèle. L'aléa provient essentiellement de la variabilité des données et de l'aléa de l'apprentissage des poids du réseau. Pour ajuster l'aléa dû aux données, une distribution utilisant l'*output* du réseau de neurones a été choisie. Pour prendre en compte l'aléa de l'apprentissage des poids d'un réseau de neurones, le réseau de neurones multi-couches a été remplacé par un réseau de neurones bayésien. En effet, un réseau de neurones bayésien apprend les paramètres d'une loi, généralement une loi normale, à la place de chaque poids du réseau. La variance de cette loi permet ainsi d'avoir une mesure de la qualité de l'apprentissage de ce poids.

La recherche bibliographique a permis de trouver un modèle satisfaisant ces conditions. Le modèle repris est celui de KUO, 2020. Dans son papier, Kuo énonce son modèle et l'applique uniquement sur des données simulées. Dans ce mémoire, une explication plus détaillée des éléments du modèle est fournie. De plus, l'application a été réalisée sur des données réelles et ainsi des limites constructives ont pu être mises en évidence.

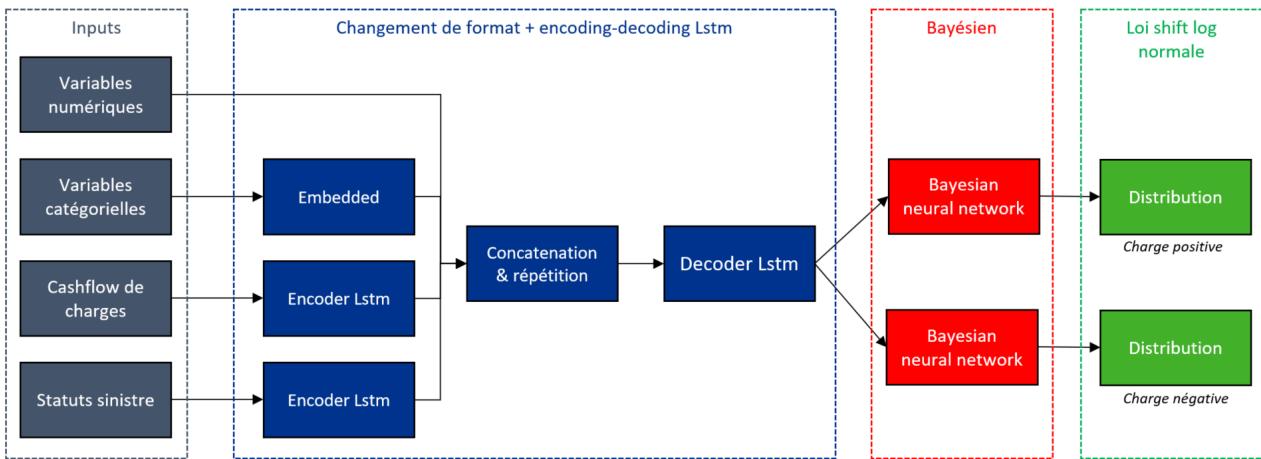


FIGURE 1 : Schéma du modèle de Kuo

Le modèle ci-dessus est composé d'un *lstm* encoder pour condenser les données temporelles. Pour les variables catégorielles, la méthode d'*embedded layer*, associant 2 poids à chaque modalité de chaque variable qualitative, a été utilisée. L'ensemble de l'information est ensuite concaténée. A l'aide d'un *lstm* "one to many", une information par pas de temps est prédite. Un réseau de neurone bayésien prend cette information afin de prédire les paramètres de la loi retenue pour les incréments de charge positifs. Le même procédé est utilisé pour la prédiction des incréments de charge négatifs.

Une particularité du modèle consiste à la séparation des incréments de charge positifs et négatifs, donnant lieu à la prédiction d'une distribution pour chaque incrément positif de charge par année de développement et d'une autre distribution pour chaque incrément négatif de charge par année de développement. Ce choix contesté dans la partie des limites du modèle a été retenu afin de pouvoir appliquer la distribution de sortie voulue

$$P_1 \times (\log \mathcal{N}(\mu, \sigma^2) + \text{shift}) + (1 - P_1) \times 0.$$

Visuellement les incréments de charges positifs ou négatifs ressemblent à une loi log-normale avec des valeurs très extrêmes. La loi log-normale, ne permettant que de prédire des valeurs positives, justifie donc la séparation des incréments de charge.

L'application du modèle et ses limites

Le modèle a été utilisé sur une base de responsabilité civile médicale. La responsabilité médicale désigne l'obligation pesant sur les professionnels de santé de réparer le dommage causé par la mauvaise exécution d'un contrat de soins. C'est une branche longue et la base ne dispose que de 17 années de développement après retraitement. Les variables temporelles disponibles sont la charge, les paiements, le statut (ouvert ou fermé) du sinistre. Les variables qualitatives sont l'année de survenance, d'ouverture, de clôture, le type (matériel/corporel), le canal de distribution du contrat, puis des variables binaires indiquant si le sinistre est litigieux, ré-ouvert ou co-assuré.

Les variations récentes sur les années de survenance les plus anciennes dues à la longueur de la branche font qu'il est plus prudent d'utiliser les montants de charge que les paiements.

Une analyse descendante des variables a permis de réduire le nombre de variable explicative in-temporelles, ne sélectionnant que la variable de litige, de type et de réouverture.

Les résultats ont été comparés aux résultats de Chain-Ladder et au range d'experts (table 1).

	Charge ultime prédite	PSAP
Min	675 504 582	-69 065 953
Max	739 704 342	-4 866 193
Moyenne	714 138 405	-30 432 130
Chain-Ladder	713 897 623	-30 672 912

TABLE 1 : Base de comparaison

L'ensemble de la base de données a été utilisée pour la prédiction. Le modèle prédit les paramètres de 2 distributions pour chaque année de développement de chaque sinistre. A l'aide d'un processus de Monte-Carlo, 1000 simulations de ces lois nous a permis d'obtenir 1000 valeurs de la charge restante prédite pour chaque sinistre. Ces résultats ont ensuite été agrégés par année de survenance afin d'être comparés. L'ordre de l'*input* a été modifié comparé au modèle du papier de Kuo afin de passer d'une prédiction de l'année suivante à une prédiction des 16 premières années de développement. Ce changement provient de la constatation empirique d'une uniformisation des prédictions sur les années de développement lors de l'application initiale. Observons les résultats du modèle modifié en unité monétaire (les montants ont été multipliés par un facteur, proche de 1 pour conserver les ordres de grandeur malgré la confidentialité).

Il y a une différence d'ordre de grandeur entre la charge prédite par notre modèle et nos dires d'experts. Le problème ne provient pas de la variabilité des résultats mais bien de la prédiction

Année de survenance	Charge prédite model	Charge prédite dire d'experts	Charge connue
2 003	42 884 968	42 001 023	42 709 809
2 004	32 487 623	32 450 756	32 365 545
2 005	44 144 601	43 051 899	43 678 279
2 006	47 551 613	43 264 392	46 186 618
2 007	56 667 112	40 330 551	54 150 742
2 008	48 965 375	47 455 923	44 677 318
2 009	56 626 192	47 676 307	50 491 075
2 010	54 197 108	46 696 955	45 553 773
2 011	64 591 443	43 363 023	51 900 585
2 012	56 575 982	37 547 237	40 716 646
2 013	59 983 219	41 278 429	40 636 625
2 014	63 521 558	46 612 004	40 024 901
2 015	75 885 851	35 940 441	38 521 491
2 016	93 805 480	43 266 517	39 662 329
2 017	137 335 346	49 390 300	47 839 085
2 018	155 712 211	51 767 671	46 405 483
2 019	150 036 084	39 741 115	39 050 231
Ultime totale	1 240 971 766	731 834 541	744 570 535

TABLE 2 : Résultat brut du modèle retenu en UM (unité monétaire)

moyenne qui est fautive. En observant les prédictions par année de survenance, une surestimation croissante sur les années de développements récentes est constatée. Un *backtesting* montrant la même conclusion a aussi été réalisé.

Les limites de l'application du modèle

Après les retraitements de la base et l'implémentation du modèle, les travaux de ce mémoire ont principalement consisté à la compréhension de cette sur-estimation et la recherche d'autres limites du modèle. Plusieurs limites ont été trouvées.

La première limite est la différence d'ordre de grandeur des incréments de charges. En effet, contrairement à un provisionnement classique par Chain-Ladder où l'on utilisera une segmentation préalable entre les sinistres attritionnels et graves, le choix retenu ici a été de laisser au modèle la charge d'apprendre la différenciation d'ordre de grandeur à prédire. Cependant, due à l'importance des sinistres de montants très élevés sur le provisionnement final (50% des réserves provient de sinistres de charge connue supérieure à 150 k(UM)), le modèle n'est que très peu sensible aux variations d'incréments de charges faibles.

Une deuxième limite provient de la variable de litige. Cette variable est la plus significative des variables explicatives d'après la sélection effectuée. Cependant, son comportement a fortement changé dans la base. En effet, sur les années de survenance les plus anciennes, un très faible nombre de sinistres passait devant un juge, tandis que sur les années récentes, sans augmentation du nombre de sinistres par année de survenance, la proportion a fortement augmenté et est maintenant de 25%. L'augmentation du nombre de sinistres litigieux associé à un changement de tendance (tendance linéaire haussière pour les années anciennes contrairement à la tendance stable des années récentes), entraîne une surestimation de la prédiction des sinistres litigieux, car son apprentissage repose sur l'historique connu.

Une troisième limite est la difficulté de calibration du modèle. En effet, l'impact de chaque variable et hyper-paramètre a été testé par analyse descendante afin de sélectionner les meilleurs. Cependant la volatilité du modèle bayésien empêche toute comparaison, car il est difficile de savoir si la différence provient du changement testé ou bien de la volatilité du bayésien. En remplaçant le réseau bayésien par un réseau multi-couches, une volatilité persiste. C'est la volatilité due à l'initialisation des poids

initiaux du réseau qui va modifier le minimum local atteint à la fin de l'entraînement. Afin de réduire les volatilités attendues et pour calibrer le réseau, de nombreux entraînements sont nécessaires. La calibration est donc très chronophage, et ce, sur une base de petite taille (63 000 sinistres).

D'autres choix pourraient être remis en question comme la séparation de la charge incrémentale positive et négative et la distribution de sortie. Cependant, ces cas n'ont pas été traités.

Une explication détaillée des différents réseaux de neurones a été réalisée. L'implémentation et l'application de ce modèle sur une base de données réelles permettent d'aborder des limites réellement rencontrées par les actuaires provisionnement dans un contexte de provisionnement ligne à ligne par réseaux de neurones et permettra au lecteur d'identifier plus facilement les points clés et limites de ce type de modèle.

Synthesis note

Insurance contracts have the particularity of having a reversed production cycle, forcing insurers to determine the selling price of the contract before knowing the real costs. However, the insurer must be able to pay the amount of the claim if it occurs. To guarantee its ability to settle future claims, the insurer will set up reserves, in particular the outstanding claims reserve (PSAP).

PSAP, which is intended to cover future payments for claims that have already occurred, is currently estimated by methods based on aggregated data such as Chain Ladder and Borhuetter-Ferguson. These historical methods are well known and documented. They are quick to implement and require very little computation time. However, the use of aggregated data by year of occurrence creates a huge loss of information, including individual loss characteristics. In a world where a large amount of computing power is now available on all our computers, new methods called "line by line", using all the data, have appeared. The research objective of these new methods is to challenge the aggregate methods in order to gain in understanding and accuracy.

This paper has implemented a micro reserving model using deep neural networks. We will focus on the estimation of the IBNER reserve (*incurred but not sufficiently reported*).

Another constraint was to obtain a density of the provision. Indeed, when calculating reserves under the French GAAP, a quantile of the IBNERs is used in order to add a conservative risk margin. Under Solvency 2, the reserving risk with internal model calculation can use the 99.5 quantile of the reserve distribution. Also under IFRS 17, the reserve density is used to calibrate the risk adjustment so that the sum of the best estimate and the risk adjustment is equal to a quantile of the reserve distribution. The quantile is chosen according to the risk aversion of the insurer.

In the line of the thesis of Chartrel, 2022, PRIMEL, 2021 and **FABRE**, this thesis innovates by using deep neural networks to obtain a distribution.

The bayésian neural network

The model chosen is a micro reserving model. Once trained, the model takes a single claim and must predict a distribution for each unknown development year of the claim.

To perform the prediction, the neural network method was used. A neuron is a weighted sum of outputs passed in argument of an activation function (identity, sigmoid, tanh). A neural network consists in using the outputs of a neuron as input of an other neuron of a following layer. The learning is done by applying the gradient descent algorithm to a loss function. This loss function compares the expected output and the output predicted by the model, with the RMSE function for example.

Moreover, the individuals claims data have 2 special variables: the reserve per year of development and the opening or closing year of the claim. These variables, called "temporal", require the use of a

recurrent neural network, allowing to keep in memory the impact of the values of the past timesteps for the prediction of the next timestep.

Finally, in order to obtain a density of reserves, a randomness must be added to the model. The randomness comes mainly from the variability of the data and the randomness of the learning phase of the neural network weights. To adjust the randomness due to the data, a distribution using the output of the neural network was chosen. To take into account the randomness of the learning phase of a neural network, the classical dense neural network has been replaced by a Bayesian neural network. A Bayesian neural network learns the parameters of a distribution, generally a normal distribution, instead of each weight of the network. The variance of this law allows us to have a measure of the quality of the learning of this weight.

The bibliographic research allowed us to find a model satisfying these conditions. The model used is that of Kuo. In his paper, Kuo fixes the structure of the model and applies it only to simulated data. In this thesis, a more detailed explanation of the elements of the model is provided. Moreover, the application was performed on real data and thus constructive limitations could be highlighted.

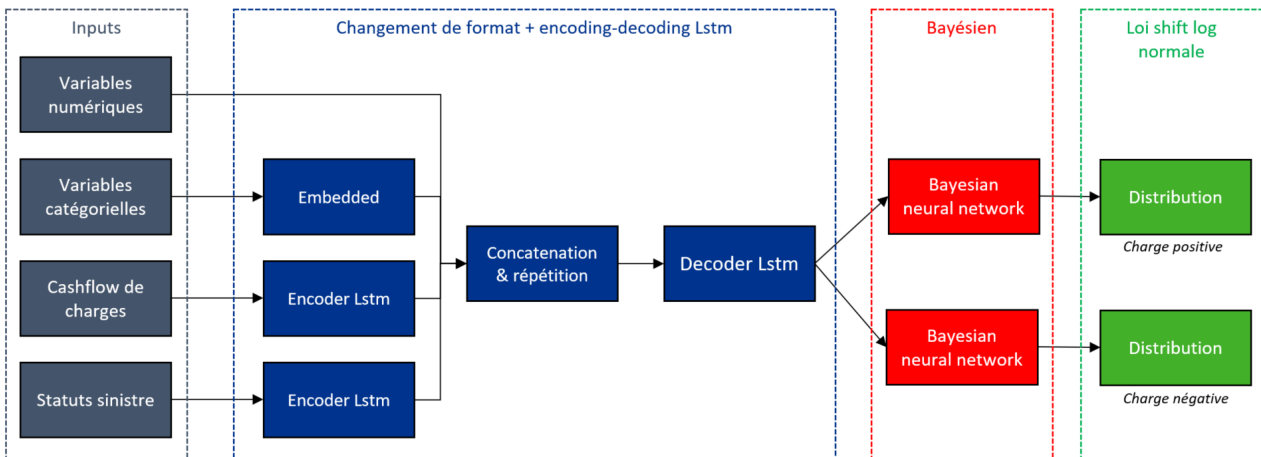


Figure 2: Diagram of Kuo's model

The above model is composed of an encoder to condense the temporal data. For the categorical variables, the embedded layer method, associating 2 weights to each modality of each qualitative variable, has been used. All the information is then concatenated. With the help of a "one to many" *lstm*, an information per time step is predicted. A Bayesian neural network takes this information in order to predict the parameters of the chosen distribution for the positive charge increments. The same process is used for the prediction of negative charge increments.

A special feature of this model is the separation of positive and negative reserve increments, resulting in the prediction of one distribution for each positive reserve increment per development year and another distribution for each negative reserve increment per development year. This contested choice in the limitations section of the model was retained in order to apply the desired output distribution

$$P_1 \times (\log \mathcal{N}(\mu, \sigma^2) + \text{shift}) + (1 - P_1) \times 0.$$

Visually the positive or negative reserve increments looks like a lognormal distribution with extreme values. The lognormal distribution, allowing only positive values to be predicted, justifies the separation of the reserve increments.

The application and limits of the model

The model was used on a medical liability database. Medical liability refers to the obligation of health professionals to repair the damage caused by the improper performance of a contract of care. It is a long branch and the base has only 17 years of development after reprocessing. The temporal variables available are the charge, the payments, the status (open or closed) of the claim. The qualitative variables are year of occurrence, opening, closing, type (material/corporeal), distribution channel of the contract, and binary variables indicating whether the claim is disputed, re-opened or co-insured.

Recent variations in the older years of occurrence due to the length of the industry make it more prudent to use expense amounts than payments.

A top-down analysis of the variables reduced the number of timeless explanatory variables, selecting only the type and reopening dispute variable.

The results were compared to the Chain-Ladder results and the expert range (table 3).

	Charge ultime prédite	PSAP
Min	675 504 582	-69 065 953
Max	739 704 342	-4 866 193
Moyenne	714 138 405	-30 432 130
Chain-Ladder	713 897 623	-30 672 912

Table 3: Comparison database

The entire database was used for prediction. The model predicts the parameters of 2 distributions for each year of development of each claim. Using a Monte-Carlo process, 1000 simulations of these distributions allowed us to obtain 1000 values of the predicted remaining reserve for each claim. To compare with the aggregate result, the prediction is aggregated by year of occurrence. The order of the *input* was changed compared to the model in Kuo's paper to change the next year prediction into an 16 years of development. This change comes from the empirical finding that predictions over the years of development were consistent in the initial application. Let us observe the results of the modified model in monetary units (MU)(the amounts have been multiplied by a factor, close to 1, to preserve the orders of magnitude despite the confidentiality).

There is an order of magnitude difference between the load predicted by our model and our expert opinion. The problem is not the variability of the results, but the average prediction is wrong. Looking at the predictions by year of occurrence, there is an increasing overestimation in recent development years. A *backtesting* showing the same conclusion was also performed.

Les limites de l'application du modèle

After reprocessing the database and implementing the model, the work of this thesis consisted mainly in understanding this overestimation and searching for other limitations of the model. Several limitations were found.

The first limitation is the difference in the order of magnitude of the reserve increments. Indeed, in contrast of a classical Chain-Ladder reserving where a prior segmentation between attritional and severe claims is used, the choice made here was to let the model learn the magnitude of the claim to predict. However, due to the importance of very large claims on the final reserving (50 % of the reserves come from claims with a known expense of more than 150 k(MU)), the model is only slightly

Année de survenance	Charge prédite model	Charge prédite dire d'experts	Charge connue
2 003	42 884 968	42 001 023	42 709 809
2 004	32 487 623	32 450 756	32 365 545
2 005	44 144 601	43 051 899	43 678 279
2 006	47 551 613	43 264 392	46 186 618
2 007	56 667 112	40 330 551	54 150 742
2 008	48 965 375	47 455 923	44 677 318
2 009	56 626 192	47 676 307	50 491 075
2 010	54 197 108	46 696 955	45 553 773
2 011	64 591 443	43 363 023	51 900 585
2 012	56 575 982	37 547 237	40 716 646
2 013	59 983 219	41 278 429	40 636 625
2 014	63 521 558	46 612 004	40 024 901
2 015	75 885 851	35 940 441	38 521 491
2 016	93 805 480	43 266 517	39 662 329
2 017	137 335 346	49 390 300	47 839 085
2 018	155 712 211	51 767 671	46 405 483
2 019	150 036 084	39 741 115	39 050 231
Ultime totale	1 240 971 766	731 834 541	744 570 535

Table 4: Result of the model (MU)

sensitive to variations in low reserving increments.

A second limitation comes from the litigation variable. This variable is the most significant of the explanatory variables based on the selection made. However, its behavior has changed significantly in the database. Indeed, in the oldest years of occurrence, a very small percent of claims went before a judge, whereas in recent years, with no increase in the number of claims per year of occurrence, the proportion has greatly increased and is now 25%. This increase associated with a change in trend (linear upward trend of the cumulative reserve for older years as opposed to the stable trend of recent years), leads to an overestimation of the prediction for litigious claims because its learning is based on known history.

A third limitation is the difficulty of calibrating the model. Indeed, the impact of each variable and hyper-parameter was tested by top-down analysis in order to select the best ones. However, the volatility of the Bayesian model prevents any comparison because it is difficult to know if the difference comes from the tested change or from the volatility of the Bayesian. By replacing the Bayesian network by a multi-layer network, a volatility persists. It is the volatility due to the initialization of the initial weights of the network that will modify the local minimum reached at the end of the training. In order to reduce the expected volatilities and to calibrate the network, many training sessions are necessary. The calibration is therefore very time consuming and this was only a small database (63 000 claims).

Other choices could be questioned such as the separation of the positive and negative incremental reserve and the output distribution. However, these assumptions were not addressed.

A detailed explanation of the different neural networks has been provided. The implementation and application of this model on a real database allows to address the limitations actually encountered by actuaries in the context of micro-reserving and will allow the reader to more easily identify the key points and limitations of these types of models.

Remerciements

Premièrement, je tiens à remercier KPMG FRANCE de m'avoir donné l'opportunité de réaliser mon stage.

Je remercie ensuite Rimen AYOUB pour son accompagnement quotidien, sa patience et sa disponibilité tout au long de ce mémoire.

Je tiens aussi à remercier Charles Erwin ELIACHAR, Camille VIEAU pour leur encadrement et leur support durant l'ensemble de ce mémoire.

Enfin, un sincère remerciement à Axel CHARTREL pour ses relectures et son écoute durant ce stage.

Pour finir, j'adresse un grand merci à l'ensemble de l'équipe Actuariat de KPMG pour l'accueil et le soutien, sans qui ma motivation n'aurait pas été la même.

Table des matières

Résumé	3
Abstract	4
Note de Synthèse	5
Synthesis note	11
Remerciements	15
Table des matières	17
Introduction	19
1 Contexte général	21
1.1 L'assurance non-vie et ses provisions	21
1.2 Enjeux réglementaires	23
1.3 Les différents modèles de provisionnement non-vie	27
1.4 Bilan des modèles de provisionnement et intérêt d'un modèle à densité	36
1.5 Étude d'une base de données de responsabilité médicale	36
2 Le <i>Bayesian Neural Network</i>	39
2.1 Un réseau de neurones	39
2.2 Les réseaux de neurones récurrents	58
2.3 La gestion de l'aléa	62
2.4 Choix des modèles	67
2.5 Processus de prédiction des résultats par Monte-Carlo	69
2.6 Processus de calibration du modèle	70

2.7	Bilan	74
3	Analyse du modèle	75
3.1	Étude de la base de responsabilité civile médicale	75
3.2	Base de comparaison d'un modèle	81
3.3	L'initialisation du modèle de Kuo	82
3.4	Analyse des résultats du modèle de Kuo retenu	93
3.5	Bilan de l'application du modèle	103
	Conclusion	105
	Bibliographie	107
A	Étude du comportement de la variable litige	109
A.1	Étude descriptive détaillée : la charge moyenne au développement 0 selon la variable litige	109
A.2	Étude descriptive détaillée : la charge cumulée des sinistres litigieux par développement	110
A.3	Étude descriptive détaillée : la proportion de sinistre litigieux par survenance	111

Introduction

Une assurance consiste à la vente d'un contrat qui assure qu'en échange d'une prime, une contrepartie sera versée si la réalisation d'un sinistre survient. Contrairement à une entreprise classique dans laquelle on achète un produit, on le transforme puis on le vend, une entreprise d'assurance commence par vendre et ne saura le coût qu'à la clôture définitive du contrat. Pour faire face à cette incertitude, l'assureur va provisionner le coût qu'il estime devoir payer dans le futur. Cette estimation s'appelle la provision technique.

La bonne estimation des provisions techniques est cruciale. En effet, si l'on provisionne peu, on risque de ne pas avoir assez pour faire face aux engagements pris. Dans le cas contraire, si l'on provisionne trop alors on ne peut pas placer l'argent et cela conduit à une perte. Il est donc primordial d'avoir une bonne estimation des coûts futurs liés aux engagements passés.

Pour effectuer le calcul des provisions techniques, les assureurs agrègent leurs contrats par garanties et observent les prédictions sur des données agrégées par année de survenance et année de développement. Des méthodes connues et maîtrisées sur données agrégées permettent à ce jour d'avoir des calculs fiables et compréhensibles. Cependant, dans un monde où l'on dispose de toujours plus de données et de méthodes pour transformer une donnée en information, il serait dommage de ne pas challenger les méthodes passées. Les assureurs disposent des données sinistre par sinistre avant agrégation. Ces données dites ligne à ligne ne sont, pour le moment, pas utilisées dans les calculs mais disposent de plus d'informations que leurs équivalents agrégés.

De plus, les assureurs étant soumis à différentes normes ne peuvent plus se contenter d'avoir seulement une estimation moyenne de ces provisions. Ils ont la nécessité d'obtenir un intervalle de confiance, une distribution de leurs provisions totales.

Dans la continuité d'autres mémoires déjà réalisés à ce sujet par d'autres modèles, ce mémoire a donc pour objectif d'étudier un modèle de *machine learning* composé uniquement de réseaux de neurones afin de calculer la distribution des provisions.

Le premier chapitre visera à mettre en place le contexte de cette étude en rappelant les différentes notions de provisions, les méthodes sur données agrégées et une brève introduction de la base de données.

Le second chapitre développera les notions nécessaires à la compréhension du modèle étudié, de la définition d'un réseau de neurones simple à des problématiques plus complexes permettant de gérer l'aléa dans un modèle. Il se conclura par les méthodes de sélection des paramètres d'un réseau de neurones.

Le troisième chapitre concernera l'application du modèle. Il sera composé d'une étude descriptive de la base de données réelle étudiée, de la sélection des paramètres, des résultats obtenus et d'une explication des limites de celui-ci.

Chapitre 1

Contexte général du provisionnement non-vie

Premièrement, nous allons rappeler les notions essentielles à l'assurance non-vie, avec notamment la notion de provision. Dans un second temps, nous présenterons les enjeux réglementaires en lien avec le provisionnement non-vie. Dans un troisième temps, nous détaillerons les différents modèles de provisionnements sur données agrégées. Enfin, nous conclurons sur un bilan des modèles de provisionnement étudiés.

1.1 L'assurance non-vie et ses provisions

1.1.1 L'assurance non-vie

Une assurance est un contrat entre un assuré, un assureur et un bénéficiaire. L'assuré paye une prime à l'assureur qui l'encaisse et qui va réaliser une prestation dépendant d'un événement aléatoire au bénéficiaire (voir figure 1.1).

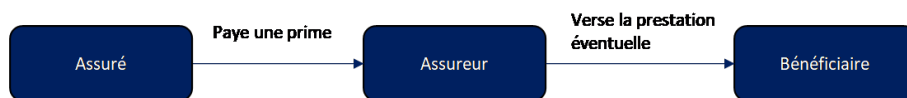


FIGURE 1.1 : Schéma acteurs d'un contrat d'assurance

Le code des assurances a décomposé les différents événements aléatoires en 26 branches. Nous nous concentrerons ici sur l'assurance non-vie pouvant être dénommée par assurance dommage, assurance *P&C* (*Property and Casualty*) ou encore assurance IARD (incendie, accident et risques divers). L'assurance non-vie assure une personne, qui peut être morale ou physique, contre des aléas que peuvent subir ses biens qui ne concernent pas la vie humaine. Elle concerne les branches suivantes :

- Accidents (assurance "individuelle accidents") ;
- Maladie ;
- Corps de véhicules terrestres ;
- Corps de véhicules ferroviaires ;

- Corps de véhicules aériens ;
- Corps de véhicules maritimes, lacustres et fluviaux ;
- Marchandises transportées ;
- Incendie et éléments naturels ;
- Autres dommages aux biens (risques divers) ;
- Responsabilité civile véhicules terrestres automoteurs ;
- Responsabilité civile véhicules aériens ;
- Responsabilité civile véhicules maritimes, lacustres et fluviaux ;
- Responsabilité civile générale ;
- Crédit ;
- Caution ;
- Pertes pécuniaires diverses ;
- Protection juridique ;
- Assistance.

La particularité du domaine de l'assurance est son cycle de production inversé. Le prix de vente du produit d'assurance est connu et fixé tandis que le coût est incertain. Afin de pouvoir faire face à ses engagements, l'assureur doit donc estimer le coût de la prestation future. C'est la définition du provisionnement. L'objectif est de provisionner un montant suffisamment important pour faire face au coût incertain afin de tenir les engagements de l'assureur envers l'assuré. Cette particularité implique que tous les risques ne sont pas assurables. L'assureur peut être dans l'incapacité d'assurer le coût éventuel de la prestation (un avion déplaçant une équipe de foot), peut ne pas disposer de statistiques historiques (assurance d'un satellite) ou peut refuser d'assurer un risque pour des raisons morales (jeux d'argent, drogues). Un autre point important pour l'assureur est la temporalité de la provision. En effet, l'assureur va donc encaisser la prime reçue et la placer le plus longtemps possible afin d'augmenter ses gains. La problématique est d'estimer quand et combien l'assureur va devoir payer pour rembourser les sinistres et les frais liés au contrat. Ceci nous amène aux différentes provisions estimant ces montants.

1.1.2 Les différentes provisions

Par définition, une provision est une somme d'argent mise de côté pour faire face à un événement aléatoire futur. Ce montant va évoluer en fonction des dotations et des reprises. Il existe plusieurs provisions pour faire face aux différents risques rencontrés, les deux principales catégories sont les provisions de primes et les provisions de sinistres.

Il existe différentes provisions de primes telles que :

- les provisions pour primes non acquises (PPNA) qui correspondent à la part de primes versées au cours de l'exercice en cours mais destinée à couvrir les risques des périodes futures

$$PPNA_{Dotation} = Prime_{exercice\ i} \times \frac{Periode\ de\ couverture\ sur\ exercice\ ulterieur}{Periode\ de\ couverture\ totale};$$

- la provision pour risque en cours (PREC) destinée à couvrir le risque de sous-tarifcation par rapport aux constatations réelles des coûts. Elle permet de combler la différence entre un tarif insuffisant et le coût réel du contrat

$$PREC = PPNA \times \max\left(0, \frac{Sinistre\ moyen + Frais\ moyen}{Prime\ moyenne}\right);$$

- Il existe aussi d'autres provisions comme la provision pour primes à émettre (PAE) définie comme l'estimation de la part des primes restant à émettre se rapportant à la période de couverture.

Les principales provisions en assurance non-vie sont celles de la catégorie des provisions de sinistres telles que :

- Les provisions dites dossiers à dossiers qui sont les premières provisions mises en place par l'assureur. En effet, une fois la déclaration d'un sinistre effectuée, l'assureur effectue une dotation de la provisions D/D grâce au contexte du sinistre écrit dans la déclaration.
- Le sinistre peut ensuite évoluer jusqu'à sa clôture. C'est la provision "Incurred but not enough reported" (IBNER) qui est alors dotée afin de réévaluer les provisions D/D.
- Un sinistre peut être déclaré longtemps après sa survenance, c'est un sinistre dit tardif. L'assureur va aussi provisionner le montant estimé actuariellement du coût de ces sinistres. C'est la provision "Incurred but not yet report" (IBNYR).

La provision la plus importante en assurance non-vie est la provision pour sinistres à payer ou PSAP (voir figure 1.2). C'est l'estimation des coûts futurs des sinistres déjà survenus. À une date t fixée, le calcul est le suivant

$$PSAP_t = Provision\ D/D_t + IBNER_t + IBNYR_t.$$

Cette estimation doit se rapprocher le plus possible du montant restant à payer aux titres des sinistres déjà survenus, déclarés ou non.

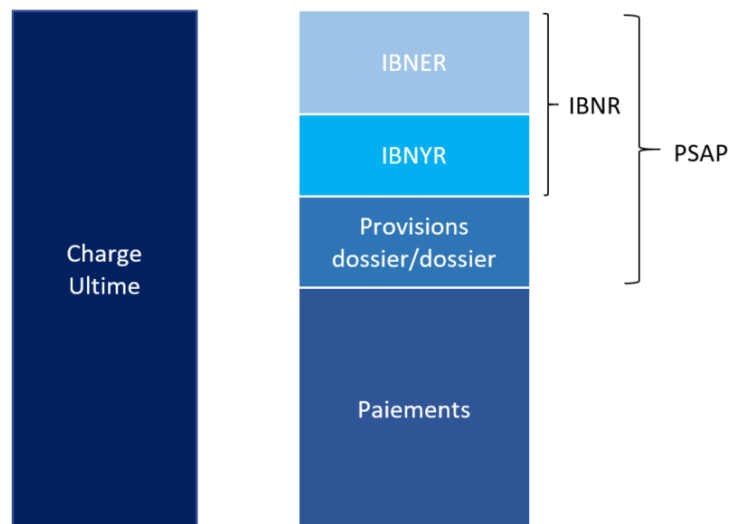


FIGURE 1.2 : Schéma de la provision pour sinistres à payer

1.2 Enjeux réglementaires

Le secteur de l'assurance reposant par nature sur des événements aléatoires est risqué. Pour autant, le secteur de l'assurance joue un rôle central dans l'économie et au sein de la société actuelle. Une faillite d'une société d'assurance impliquerait par effet domino des conséquences catastrophiques aussi bien dans le secteur financier qu'à l'échelle humaine. Afin d'éviter ou plutôt de minimiser les risques de faillite, le secteur est fortement réglementé par différentes normes, notamment en normes comptables françaises, Solvabilité 2 et IFRS 17. L'objectif du provisionnement est de réaliser une estimation des réserves. Ce montant de réserve doit permettre à l'assureur d'honorer ses engagements envers ses

clients. En normes comptables française, l'estimation est "prudente". Cela consiste à l'ajout d'un montant prudent à l'espérance mathématiques des réserves. Cependant, Solvabilité 2 et IFRS 17 imposent d'autres contraintes au calcul des provisions. Nous allons décrire brièvement ces 2 normes.

1.2.1 La norme Solvabilité 2

La directive Solvabilité 2, publiée en 2009 [Parlement Européen et Conseil de l'Union Européenne, 2009] est entrée en application au 1er janvier 2016. Elle s'applique à l'ensemble des assureurs exerçant dans l'union Européenne. Remplaçant et palliant les insuffisances de Solvabilité 1, elle garde l'objectif de protéger les assurés en limitant les risques de faillite de l'assureur. La norme impose des exigences de fonds propres et une marge de solvabilité suffisante. Elle s'articule autour de 3 piliers :

- Les exigences quantitatives ;
- Les exigences qualitatives de gouvernance et de gestion des risques ;
- Les exigences de communication d'informations au public et aux autorités de contrôle.

Le pilier I de Solvabilité 2 regroupe les exigences quantitatives, c'est-à-dire les règles de valorisation des actifs et des passifs, ainsi que les exigences de capital et leur mode de calcul. Solvabilité 2 prévoit deux exigences de capital :

- le minimum de capital requis ou *Minimum Capital Requirement* en anglais (*MCR*) ;
- le capital de solvabilité requis ou *Solvency Capital Requirement* en anglais (*SCR*).

Le *MCR* représente le montant minimum dont doit disposer en permanence l'assureur sous peine de se voir retirer sa licence et donc son droit d'exercer son activité d'assurance. Le *Solvency Capital Requirement*, généralement exprimé grâce au ratio de solvabilité

$$\text{ratio de solvabilité} = \frac{\text{Solvency Capital Requirement}}{\text{Fonds propres}}.$$

Il nous indique que l'entreprise est à 99,5 % solvable à 1 an si son ratio de solvabilité est supérieur à 1. Cela signifie que l'assureur possède des fonds propres permettant de rester solvable à 1 an dans 199 scénarios sur 200. Le premier pilier de Solvabilité 2 change donc la vision du bilan des assureurs (voir figure 1.3).

Détaillant les éléments du schéma, il y a :

- Les actifs doivent être évalués en valeurs de marché à la date du bilan ;
- Les fonds propres sont composés du *SCR* et du capital excédentaire, c'est le montant de capital disponible de la compagnie ;
- Les provisions techniques doivent être calculées avec la vision *Best-Estimate* et calculées par groupe de risques homogènes. Cette estimation ne prend pas en compte la prudence, c'est l'espérance du coût des engagements futurs ;
- La marge pour risque est un montant représentant la prudence prise par rapport au calcul *Best-Estimate*. Elle correspond au montant que demanderait un acheteur pour racheter le portefeuille.

Ce *SCR* peut se calculer avec la formule "Standard" prévue par la directive Solvabilité 2 ou d'un modèle interne complet ou partiel (certains risques étant alors couverts par la formule standard).

La formule "Standard" mise à disposition par l'EIOPA a été adoptée et implémentée par la grande majorité des assureurs français. Cette approche en vision modulaire nécessite de calculer un niveau de *SCR* pour chaque module de la cartographie des risques (voir figure 1.4). Les différents *SCR* obtenus

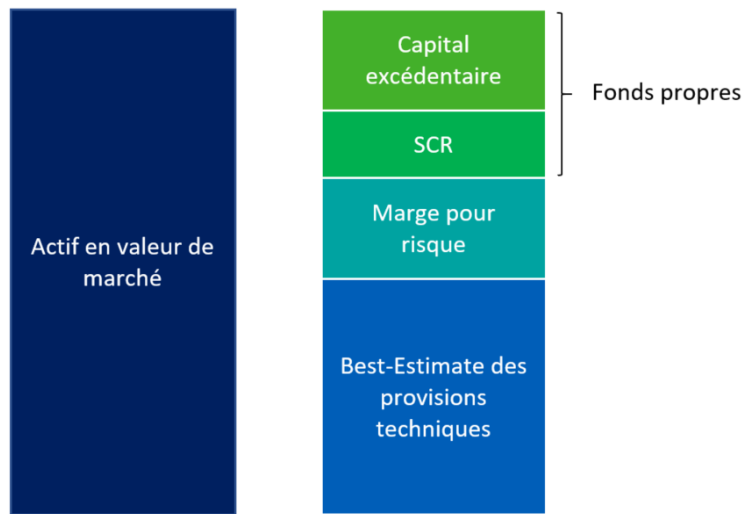


FIGURE 1.3 : Bilan vision Solvabilité 2

pour chacun des modules seront ensuite agrégés en fonction de leur corrélation. Par exemple, le *BSCR* est obtenu par la formule d'agrégation inter-modulaire

$$BSCR = \sqrt{\sum_{(i,j) \in M^2} \rho_{i,j} \times SCR_i \times SCR_j},$$

avec

- *M* la liste des sous modules de niveau n-1, c'est à dire les modules de marché, santé, contrepartie, vie, non-vie et intangible.
- $\rho_{i,j}$ le coefficient de corrélation entre les modules *i* et *j*.

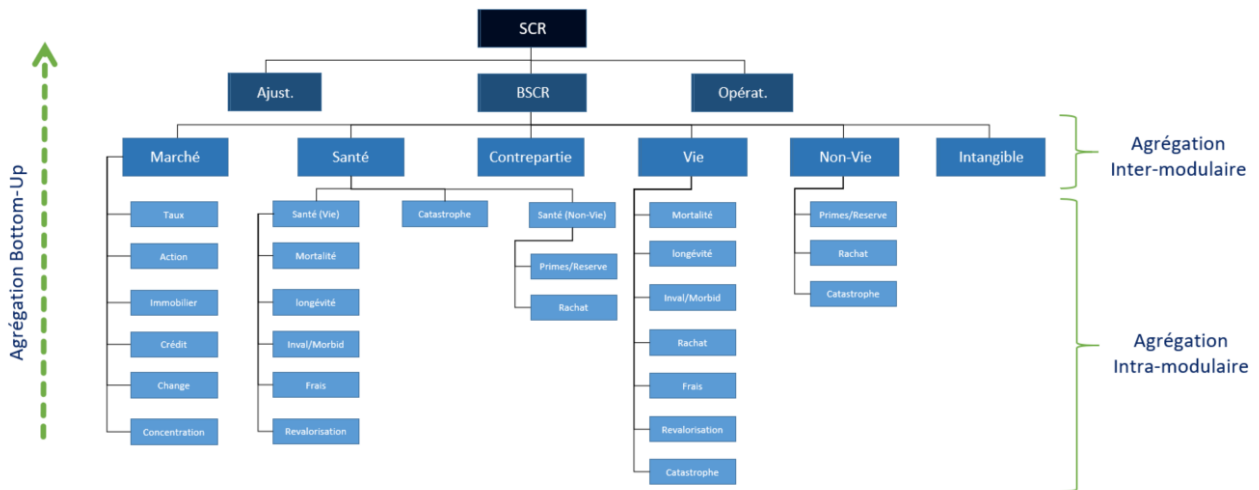


FIGURE 1.4 : Décomposition modulaire du calcul du SCR

Le *SCR* Non-Vie est composé de l'agrégation de plusieurs risques dont le risques de réserves. Solvabilité 2 impose donc de calculer un SCR de réserves et par conséquent de calculer une distribution

permettant notamment d'obtenir le quantile à 99,5 %. En formule standard, ce risque est calculé conjointement avec le risque de primes par une formule fermée. Cette méthode est facilement applicable mais générique. En effet, avec une formule générique, le caractère spécifique du portefeuille de chaque assureur n'est pas pris en compte et la formule se doit d'être plus prudente. Afin d'adapter au mieux le calcul à leur portefeuille, les assureurs peuvent choisir de calculer leurs *SCR* par leurs propres méthodes de calcul rassemblées dans un modèle interne. Malgré l'avantage de précision du modèle interne, la complexité, le coût et la difficulté d'approbation de celui-ci par l'Autorité de Contrôle Prudentiel et de Résolution (ACPR) expliquent que peu d'assureurs s'y intéressent. Hors, un modèle interne permet au *SCR* d'être plus proche du risque réel du portefeuille et, s'il est inférieur à ce que propose la formule standard, permet de conserver le même ratio de couverture pour un capital immobilisé plus faible. En modèle interne, le calcul du *SCR* de réserves nécessite de capturer le risque dû à la mauvaise estimation des provisions en *Best-Estimate*. Cela demande une modélisation de la distribution de la PSAP à un an afin de prendre le quantile à 99,5%. C'est notamment sur cette modélisation que le modèle de ce mémoire pourrait s'appliquer.

1.2.2 La norme IFRS 17

Le 18 mai 2017, la Fondation IFRS a publié la nouvelle norme IFRS 2017 « Contrats d'assurance ». Elle entrera en vigueur le 1er janvier 2023. IFRS 17 remplace la norme IFRS 4 « Contrats d'assurance » publiée en 2004 comme une norme provisoire. IFRS 4 a autorisé les sociétés à continuer à utiliser les règles comptables nationales en matière de contrats d'assurance, ce qui a donné lieu à une multitude d'approches différentes et a rendu difficile pour les investisseurs toute comparaison des performances financières des différentes sociétés. IFRS 17 résout les problèmes de comparaison créés par IFRS 4 en exigeant la comptabilisation de tous les contrats d'assurance de façon homogène. Les sociétés concernées sont les sociétés de l'Union Européenne cotées ou celle émettant de la dette sur le marché Européen. Observons un bilan en vision IFRS 17 (voir figure 1.5).

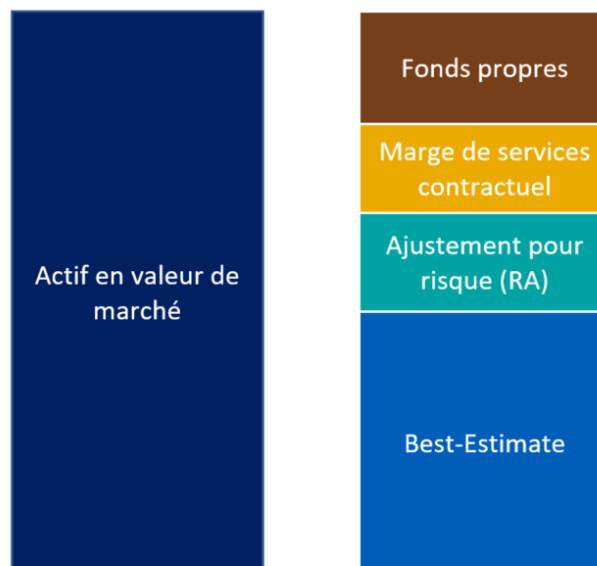


FIGURE 1.5 : Bilan vision IFRS 17

- Les actifs doivent être évalués en valeurs de marché à la date du bilan ;

- Les fonds propres se composent du capital de l'assureur investi dans des actifs non risqués ;
- La marge sur les services contractuels est une provision qui représente le profit futur de l'assureur, c'est un concept nouveau propre à l'IFRS 17. En effet sous Solvabilité 2, les profits futurs sont immédiatement comptabilisés. Sous IFRS17, les profits futurs sont étalés sur la durée de projection ;
- L'ajustement pour risque (*RA*) correspond à l'indemnité attendue par l'assureur pour faire face à l'incertitude des flux futurs de la trésorerie relative aux risques non-financiers. L'ajustement pour risque indique l'aversion au risque d'une entité donc il peut varier d'une entité à une autre ;
- Le *Best-Estimate* est l'estimation des flux de trésorerie futurs en valeur actuelle. C'est une provision médiane des engagements de l'assureur envers les assurés calculé par groupes de contrats homogènes.

L'ajustement pour risque est un montant représentant la prudence prise par rapport au calcul du *Best-Estimate* tout comme la marge pour risques de Solvabilité 2. Il se calcule de sorte que la somme du *Best-Estimate* et du *RA* soit égale à un quantile de la distribution des réserves. Ce quantile est choisi en fonction de l'aversion au risque de l'assureur. C'est un des principaux leviers permettant de manoeuvrer le bilan IFRS 17. Ce mémoire s'inscrit donc aussi dans le cadre du calcul de cette distribution des réserves IFRS 17.

1.3 Les différents modèles de provisionnement non-vie

Dans ce mémoire, notre but sera de calculer la PSAP et d'estimer la distribution des réserves. Afin de calculer cette provision, les assureurs utilisent actuellement des méthodes sur des données agrégées qui ont fait leurs preuves. Dans le but de les challenger, des méthodes sur des données ligne à ligne émergent. Nous présentons ici les principales méthodes de provisionnement sur des données agrégées afin d'en comprendre les différents avantages et inconvénients.

1.3.1 Les différents formats de données

Il existe 2 formats principaux de données, le format agrégé généralement sous forme de triangle et le format ligne à ligne.

Le format agrégé

Un sinistre peut être défini par une date de survenance, date à laquelle le sinistre a eu lieu, une date de clôture à laquelle le sinistre est clos, une date d'ouverture (confondue avec la date de déclaration) à laquelle le sinistre est ouvert chez l'assureur et les différentes années de développement qui symbolisent les différents moments auxquels la PSAP sera réévaluée entre l'ouverture et la clôture.

Les méthodes de provisionnement classiques se basent toutes sur une information sous la forme d'un triangle dans lequel les coûts des sinistres sont agrégés par années de survenance et années de développement. Les montants à l'intérieur du triangle correspondent soit aux paiements effectués soit à la charge (aussi nommé réserve) mais cela ne change pas le format des données.

Considérons le triangle décumulé suivant (voir table 1.1).

Pour un triangle de paiement, le montant $I_{i,j}$ désigne le montant payé par l'assureur durant l'année $i + j$ au titre des sinistres survenus à l'année i . Ce triangle a 3 axes de lecture :

<i>Surv i</i> \ <i>Dev j</i>	0	1	...	$n-1$	n
0	$I_{0,0}$	$I_{0,1}$...	$I_{0,n-1}$	$I_{0,n}$
1	$I_{1,0}$	$I_{1,1}$...	$I_{1,n-1}$	
...		
$n-1$	$I_{n-1,0}$	$I_{n-1,1}$			
n	$I_{n,0}$				

TABLE 1.1 : Triangle incrémental de provisionnement non-vie

- Soit sur une ligne, ce qui permet de lire les paiements d'une année de survenance pour chaque année de développement ;
- Soit sur une colonne, ce qui permet de lire les paiements d'une année de développement pour chaque survenance ;
- Soit sur une diagonale, ce qui permet de lire l'ensemble des paiements payés dans une année calendaire (l'année de survenance la plus récente de la diagonale).

Il peut aussi être intéressant dans certaines méthodes décrites dans la suite de cumuler ce triangle, notamment lorsque qu'un triangle de charge est utilisé. Un triangle est toujours cumulé sur les années de développement afin d'obtenir un total par année de survenance. Pour cumuler un triangle, il faut appliquer la formule cumulative suivante sur le triangle précédent :

$$C_{i,j} = \sum_{k=0}^j I_{i,k}.$$

Un triangle cumulé par année de développement est alors obtenu. Le but des méthodes de provisionnement est de prédire le montant de réserves, sur un triangle cumulé. Cela revient à prédire la dernière colonne (voir table 1.2).

<i>Surv i</i> \ <i>Dev j</i>	0	1	...	$n-1$	n
0	$C_{0,0}$	$C_{0,1}$...	$C_{0,n-1}$	$C_{0,n}$
1	$C_{1,0}$	$C_{1,1}$...	$C_{1,n-1}$	$\widehat{C}_{1,n}$
...
$n-1$	$C_{n-1,0}$	$C_{n-1,1}$...	$\widehat{C}_{n-1,n-1}$	$\widehat{C}_{n-1,n}$
n	$C_{n,0}$	$\widehat{C}_{n,1}$...	$\widehat{C}_{n,n-1}$	$\widehat{C}_{n,n}$

TABLE 1.2 : Triangle Cumulé de provisionnement non-vie

Le format ligne à ligne

Historiquement les assureurs ont utilisé des données agrégés. Chaque base de données avant agrégation comporte donc pour chacun des sinistres les informations suivantes :

- L'année de survenance ou de déclaration du sinistre ;
- L'année de clôture si le sinistre est clos dans la base ;
- Les montants de paiements pour chaque année de développement connue ;

- Les montants de charges pour chaque année de développement connue.

Ces informations essentielles à l'agrégation des données sous forme de triangle sont toujours présentes chez l'assureur. Une base de donnée ligne à ligne va ensuite être composée d'autres variables caractéristiques du sinistres (par exemple le type de véhicule sur une base automobile). Ces informations relatives à chaque sinistre vont varier en fonction de la branche et l'assureur. L'avantage des méthodes ligne à la ligne est premièrement de pouvoir extraire de la valeur de l'information à une granularité plus faible et donc plus détaillée permettant une meilleure compréhension et prédiction des réserves. Deuxièmement, elle permet de prendre en compte les informations relatives à chaque sinistre pour prédire les paiements de chaque sinistres et ainsi avoir une meilleure compréhension du portefeuille. Voyons un exemple de données ligne à ligne (table 1.3) comparées aux données agrégées (table 1.4).

Année de survenance	Année de cloture	Montant de la charge incrémentale		Montant des paiements incrémentale	
		Developpement 0	Developpement 1	Developpement 0	Developpement 1
2002	2002	10	X	2	X
2002	X	23	15	3	8
2002	2003	2	1	7	1
2003	X	4	X	7	X
2003	2003	15	X	3	X
2003	X	8	X	9	X

TABLE 1.3 : Exemple d'un jeu de données ligne à ligne

Triangle incr. de charges				Triangle incr. de paiements			
		Dev.				Dev.	
		0	1			0	1
Surv.	2002	35	16	Surv.	2002	12	9
	2003	27	X		2003	19	X

TABLE 1.4 : Exemple d'un jeu de données agrégées

1.3.2 Méthodes de provisionnement déterministes

À l'aide de ces représentations, nous pouvons maintenant décrire les 3 méthodes usuelles sur données agrégées : Chain-Ladder et Bornhuetter-Ferguson.

Chain-Ladder

La méthode de Chain-Ladder est la méthode d'estimation des réserves la plus connue en provisionnement non-vie, mais aussi la plus utilisée en pratique. Cette méthode multiplicative s'applique uniquement sur un triangle cumulé, par exemple un triangle de charges cumulées. L'idée sous-jacente de cette méthode est de considérer que les coefficients de passages d'une année de développement à une autre sont égaux peu importe l'année de survenance des sinistres. Avec cette hypothèse, il nous suffit d'observer les coefficients passés et nous pourrions donc compléter le triangle cumulé.

Les coefficients de passage entre les années de développement peuvent être formalisés par la notion

de facteurs de développement

$$f_{i,j} = \frac{C_{i,j+1}}{C_{i,j}}.$$

Hypothèse : Les facteurs de développements sont indépendants de l'année de survénance. Formellement,

$$\forall i \in [0, n], \forall j \in [0, n-1], f_j = f_{i,j} = \frac{C_{0,j+1}}{C_{0,j}} = \frac{C_{i,j+1}}{C_{i,j}} = \frac{C_{n,j+1}}{C_{n,j}}.$$

S'ils sont supposés égaux d'après l'hypothèse, en réalité ce cas est rare. Afin d'estimer au mieux le facteur de développement, il est calculé par moyenne des facteurs empiriques liés à l'année de développement j

$$\forall j \in [0, n-1], \hat{f}_j = \frac{\sum_{i=0}^{n-j-1} C_{i,j+1}}{\sum_{i=0}^{n-j-1} C_{i,j}}.$$

Les facteurs de développement pour chaque année sont obtenus. L'estimateur de la charge cumulée des années non connues (telle que $i + j > n$) est

$$\widehat{C}_{i,j} = \prod_{k=n-i}^{j-1} \hat{f}_k \times C_{i,n-i}.$$

Notre triangle cumulé complété de nos prédictions (observable dans la table 1.2) nous permet donc de calculer les réserves estimées par année de survénance

$$\widehat{R}_i = \widehat{C}_{i,n} - C_{i,n-i}.$$

Notre objectif est de calculer une estimation de la PSAP à l'ultime soit la somme des réserves à l'ultime

$$\widehat{PSAP}_{ultime} = \sum_i \widehat{R}_i.$$

Avant tout Chain-Ladder, il convient de vérifier l'hypothèse. Il existe plusieurs moyens : une méthode graphique et une méthode basée sur l'erreur entre les prédictions du triangle supérieur et l'original.

La première méthode consiste à observer le graphique des couples $(C_{i,j}, C_{i,j+1})$. Afin de valider l'hypothèse, il faut observer sur le graphique une droite passant par l'origine ce qui correspond à dire que l'évolution des charges est similaire à l'estimation du facteur de développement.

Une deuxième méthode consiste à calculer les facteurs de développements de Chain-Ladder et à reconstruire à partir de la diagonale (l'information la plus récente) le triangle initial

$$C_{i,j} = \frac{1}{\widehat{f_{n-i-1}}} \frac{1}{\widehat{f_{n-i-2}}} \dots \frac{1}{\widehat{f_j}} C_{i,n-i}.$$

Le triangle obtenu est ensuite comparé avec le triangle original. Cette méthode effectue le *backtesting* de Chain-Ladder.

La méthode de Chain-Ladder est donc limitée par son hypothèse d'indépendance entre les facteurs de développement et l'année de survénance car elle implique une stabilité du portefeuille entre les années de survénance. Cette stabilité peut être remise en cause par un changement de mode de

gestion, une volatilité élevée due à des sinistres graves ou un historique insuffisant. Un faible nombre de sinistres graves (généralement plus volatils que les sinistres attritionnels) n'ont peut-être pas les mêmes facteurs de développement que les facteurs des sinistres attritionnels mais de part leur volume important peuvent fausser ceux du modèle. Il convient de les retraiter au préalable.

Aussi, lors de l'application de cette méthode, les derniers facteurs de développement sont calculés avec peu d'années et perdent en fiabilité. Une anomalie sur ces valeurs aura d'autant plus d'impact qu'elle sera utilisée pour prédire l'ensemble des années suivantes. Un des moyens pour limiter cet impact sera de remplacer les derniers facteurs par un facteur de queue de distribution exponentielle ou de weibull.

Bornhuetter–Ferguson

Une méthode alternative est la méthode de Bornhuetter–Ferguson. Cette méthode s'applique sur un triangle de développement cumulé. Elle repose sur une estimation du montant de charge ultime et une estimation de la cadence de développement de cette charge ultime.

Par définition, la cadence de développement d'une année de développement j est

$$\gamma_{i,j} = \frac{C_{i,j}}{C_{i,n}},$$

avec les $C_{i,j}$ les valeurs d'un triangle cumulé.

Hypothèse : les cadences de développement sont indépendantes de l'année de survenance. Cela peut se réécrire avec la formule suivante

$$\forall i, \gamma_j = \gamma_{i,j} = \frac{C_{i,j}}{C_{i,n}},$$

En pratique, l'estimation des cadences de développement se fait à l'aide des coefficients de Chain-Ladder, formellement :

$$\forall j \in [1, n-1], \widehat{\gamma}_j = \prod_{k=j}^{n-1} \frac{1}{\widehat{f}_k}.$$

Une fois les cadences de développement obtenues, avec les données de la dernière diagonale et la valeur de l'ultime, les montants du triangle cumulé sont complétés .

$$\forall i \in [1, n], \forall j \in [n+1-i, n], \widehat{C}_{i,j} = C_{i,j-1} + \widehat{\alpha}_i \times (\widehat{\gamma}_j - \widehat{\gamma}_{j-1}),$$

Avec

- $\widehat{\alpha}_i$: l'estimation de la charge ultime de l'année i ;
- $\widehat{\gamma}_j$: l'estimation de la cadence de développement de l'année j .

Le point clé de cette méthode est la bonne estimation de la charge ultime pour chaque année de survenance. Elle peut être obtenue par jugement d'expert ou par la méthode des *Loss Ratios* détaillée ci-dessous.

$$\widehat{\alpha}_i = \widehat{C}_{i,n} = LossRatio_i \times Prime_i.$$

La méthode des *Loss Ratios* permet donc d'avoir une estimation de la charge ultime par année de survénance en se basant sur le *loss ratio* de cette même année. Ce *loss ratio* est obtenu par dire d'experts en analysant les *Loss ratios* déjà connus sur les années de survénance clôturées.

Bornhuetter–Ferguson permet, tout comme Chain-Ladder, le calcul des réserves estimées par année de survénance i

$$\widehat{R}_i = \widehat{C}_{i,n} - C_{i,n-i} = (1 - \widehat{\gamma}_{n+1-i}) \times \widehat{\alpha}_i,$$

et le calcul de la PSAP à l'ultime

$$\widehat{PSAP}_{ultime} = \sum_i \widehat{R}_i.$$

Actuellement, Chain-Ladder et Bornhuetter–Ferguson de par leur facilité d'implémentation, d'utilisation et de compréhension des résultats, sont les 2 méthodes de provisionnement les plus utilisées dans le monde. Malgré les fortes hypothèses nécessaires à leur application, les résultats en pratique fonctionnent depuis de nombreuses années. Cependant ces méthodes ne permettent d'obtenir qu'une valeur unique de la PSAP et des réserves. D'autres méthodes existent pour pouvoir obtenir une distribution des réserves, par exemple les méthodes de provisionnement stochastiques.

1.3.3 Méthodes de provisionnement stochastiques

Contrairement aux méthodes déterministes, les méthodes de provisionnement stochastiques ont pour but d'obtenir une distribution des réserves, ou plus généralement de l'ultime par survénance. Ces méthodes, toujours basées sur un triangle, supposent que les $C_{i,j}$ connus sont des variables aléatoires. Les 2 méthodes les plus connues vont être détaillées.

Mack

La méthode de MACK, 1993 proposée en 1993 est une version stochastique du modèle de Chain-Ladder. L'estimation du montant de provision est inchangée. L'avantage de Mack est de proposer un cadre probabiliste permettant d'obtenir une estimation de l'erreur quadratique du modèle. Les 3 hypothèses suivantes remplacent l'hypothèse d'indépendance de Chain-Ladder :

Hypothèse 1 : l'indépendance des années de survénance. Pour deux années de survénance i et k , $[C_{i,0}, \dots, C_{i,n}]$ est indépendant de $[C_{k,0}, \dots, C_{k,n}]$.

Hypothèse 2 : pour une année de survénance donnée, l'espérance du montant de l'année de développement $j+1$, sachant les j années de développement connues, ne dépend que du montant de l'année de développement précédente.

$$\forall i \in [0, n], \forall j \in [0, n-1], \exists f_j \in \mathbb{R}, \mathbb{E}[C_{i,j+1} | C_{i,0}, \dots, C_{i,j}] = f_j C_{i,j}.$$

Hypothèse 3 : pour une année de survénance donnée, la variance du montant de l'année de développement $j+1$, sachant les j années de développement connues, ne dépend que du montant de l'année de développement précédente.

$$\forall i \in [0, n], \forall j \in [0, n-1], \exists \sigma_j \in \mathbb{R}^+, \mathbb{V}[C_{i,j+1} | C_{i,0}, \dots, C_{i,j}] = \sigma_j^2 C_{i,j}.$$

Les 2 premières hypothèses de Mack permettent de réécrire les calculs du modèle de Chain-Ladder. Mack a démontré que les facteurs f_j estimés par les facteurs de Chain-Ladder sont :

- Sans biais. Formellement, $\forall j \in [0, n-1], \mathbb{E}[\widehat{f}_j] = f_j$;
- Non corrélés. Formellement, $\forall (j, k) \in [0, n-1]^2, j \neq k \Rightarrow \text{Cov}(\widehat{f}_j, \widehat{f}_k) = 0$.

Le triangle de charges est complété et le montant de charge à l'ultime non biaisé pour chaque survénance est obtenu en utilisant les formules de Chain-Ladder ; Formellement,

$$\forall i \in [0, n], \forall j \in [n+1-j, n], \widehat{C}_{i,n} = \mathbb{E}[C_{i,n} | C_{i,0}, \dots, C_{i,j}] = \left(\prod_{k=n-i}^{j-1} \widehat{f}_k \right) \times C_{i,n-i}.$$

Sans changement de méthode, le montant de réserve par année de survénance est obtenu par

$$\widehat{R}_i = \widehat{C}_{i,n} - C_{i,n-i}.$$

L'amélioration de Chain-Ladder provient de la 3ème hypothèse qui introduit la variance *a priori* des charges. Considérant l'existence de σ_j^2 , Mack propose un estimateur sans biais de σ_j^2 ,

$$\widehat{\sigma}_j^2 = \begin{cases} \frac{1}{n-j-1} \sum_{i=0}^{n-j} C_{i,j} \left(\frac{C_{i,j+1}}{C_{i,j}} - \widehat{f}_{i,j} \right), & \forall j \in [0, n-2]; \\ \min \left(\frac{\sigma_{n-2}^4}{\sigma_{n-3}^2}, \min \left(\widehat{\sigma}_{n-3}^2, \widehat{\sigma}_{n-2}^2 \right) \right), & \text{si } j = n-1. \end{cases}$$

A l'aide du coefficient σ_j^2 de variance *a priori*, Mack a montré comment obtenir une erreur quadratique moyenne de la prédiction de la charge ultime (*Mean Square Error of prediction*). Sachant l'information connue notée D tel que $D = [C_{i,0}, \dots, C_{i,j}]$, nous avons

$$\text{MSEP}(\widehat{R}_i) = \mathbb{E} \left[\left(\widehat{R}_i - R_i \right)^2 \mid D \right] = \mathbb{E} \left[\left(\widehat{C}_{i,n} - C_{i,n} \right)^2 \mid D \right] = \mathbb{V} \left[\widehat{C}_{i,n} \mid D \right] + \left(\mathbb{E} \left[\widehat{C}_{i,n} \mid D \right] - \widehat{C}_{i,n} \right)^2.$$

Il y a une séparation de l'erreur quadratique en 2 termes. Le premier terme est composé de la partie stochastique et le second représente l'erreur d'estimation entre les \widehat{f}_j et les f_j . Le calcul théorique étant complexe, Mack propose un estimateur de cet estimateur de l'erreur quadratique

$$\forall i \in [1, n], \widehat{\text{MSEP}}(\widehat{R}_i) = \widehat{C}_{i,n}^2 \sum_{j=n-i}^{n-1} \frac{\widehat{\sigma}_j^2}{\widehat{f}_j^2} \left(\frac{1}{\widehat{C}_{i,j}} + \frac{1}{\sum_{k=0}^{n-j-1} C_{k,j}} \right).$$

En allant plus loin, il est possible de retrouver un estimateur de l'erreur quadratique de la PSAP

$$\text{MSEP}(\widehat{PSAP}) = \sum_{i=1}^n \left(\text{MSEP}(\widehat{R}_i) + \widehat{C}_{i,n} \sum_{k=i+1}^n \widehat{C}_{k,n} \sum_{l=n-i}^{n-1} \frac{2\widehat{\sigma}_l^2}{\widehat{f}_l^2 \sum_{m=0}^{n-l-1} C_{m,l}} \right).$$

Le modèle de Mack est donc une adaptation de Chain-Ladder permettant d'avoir un estimateur quadratique des réserves de celui-ci. Malgré la complexité des formules à l'écrit, l'implémentation et l'exécution est rapide et simple d'utilisation. Cependant, elle est difficilement interprétable. En pratique, dans le cas de données irrégulières, cette méthode s'avère assez peu robuste et peut mener à des résultats aberrants, car la vérification et la justification des hypothèses n'est pas toujours possible. De plus, ce modèle ne permet pas d'obtenir une distribution pour les réserves.

Bootstrap

Le modèle de ré-échantillonnage Bootstrap est un modèle permettant d'avoir une distribution des réserves. Le Bootstrap non-paramétrique est plus facilement interprétable et n'impose pas d'hypothèse sur la distribution sous-jacent.

Le modèle se base sur le triangle supérieur incrémental des règlements et le but est d'estimer un intervalle de confiance de l'estimation de la PSAP.

Hypothèse initiale du ré-échantillonnage : comme toute méthode de Bootstrap, elle suppose des variables d'intérêt indépendantes et identiquement distribuées (iid).

Instinctivement, nous souhaiterions utiliser les montants du triangle de règlements. Comme les éléments d'un triangle ne le sont pas, la méthode de ré-échantillonnage Bootstrap est appliquée sur les résidus d'un modèle de provisionnement définis par

$$r_{i,j} = \frac{I_{i,j} - \mathbb{E}[\widehat{I}_{i,j}]}{\sqrt{\mathbb{V}[\widehat{I}_{i,j}]}}.$$

L'hypothèse est ainsi modifiée.

Hypothèse du ré-échantillonnage : les résidus de Pearson du triangle incrémental sont des réalisations de variables aléatoires indépendantes et identiquement distribuées (iid).

La méthode Bootstrap est composée de 3 étapes :

1. Calcul des résidus de Pearson du triangle supérieur :

Avec le triangle de paiements cumulés, les f_j sont calculés d'après la méthode de Chain-Ladder. Puis les montants du triangle supérieur sont recalculés à partir de la dernière diagonale connue $\forall i \in [1, n], \forall j \in [1, n]$ tel que $i + j \leq n + 1$,

$$\widehat{C}_{i,j-1} = \frac{C_{i,j}}{f_{j-1}}.$$

Une fois le triangle cumulé prédit obtenu, il est décumulé pour obtenir un triangle décumulé prédit ($\widehat{I}_{i,j}$). La formule des résidus de Pearson s'applique ensuite

$$r_{i,j} = \frac{I_{i,j} - \widehat{I}_{i,j}}{\sqrt{\widehat{I}_{i,j}}}.$$

Les résidus des deux extrémités de la diagonale des résidus ($r_{1,n-1}$ et $r_{n-1,1}$) sont nuls par construction et non iid. Il faudrait théoriquement ne pas les utiliser dans l'étape de ré-échantillonnage mais en pratique les deux façons de procéder sont généralement conservées ;

2. Effectuer K fois un ré-échantillonnage Bootstrap avec prédiction des montants de réserves sur chaque échantillon.

Une des K étapes consiste à :

- (a) Ré-échantillonner les résidus du triangle supérieur ;

(b) Retrouver un triangle supérieur issu des résidus ré-échantillonnés à partir de la formule

$$\widehat{I}_{i,j}^k = r_{i,j}^k \times \sqrt{\widehat{I}_{i,j}} + \widehat{I}_{i,j}.$$

Avec :

- $k \in [1, K]$: l'étape en cours ;
- $r_{i,j}^k$: le résidus de survénance i et de développement j à l'itération k ;
- $\widehat{I}_{i,j}^k$: un élément du triangle supérieur issu des résidus ré-échantillonné de survénance i et de développement j à l'étape k ;
- $\widehat{I}_{i,j}$: un élément du triangle prédit initialement dans l'étape de calcul des résidus de Pearson.

(c) Calculer l'estimation de la PSAP (\widehat{PSAP}^k) sur ce triangle reconstitué.

La méthode employée est usuellement Chain-Ladder ou Bornhuetter–Ferguson et nécessite donc de re-cumuler le triangle au préalable.

3. Estimation d'une distribution à l'aide des K montants de la PSAP bootstrapée.

Cette étape consiste à l'obtention d'une distribution de la PSAP. Par exemple, une méthode consiste à réaliser un histogramme (voir figure 1.6) afin d'avoir une intuition de sa forme.

Avec cette méthode, il est ainsi possible de calculer la PSAP, les réserves par années de survénance, mais aussi différentes mesures de risque telles que l'écart type, les values at Risk (VAR) et Tail values at risks (TVAR) pour les niveaux de risque souhaités.

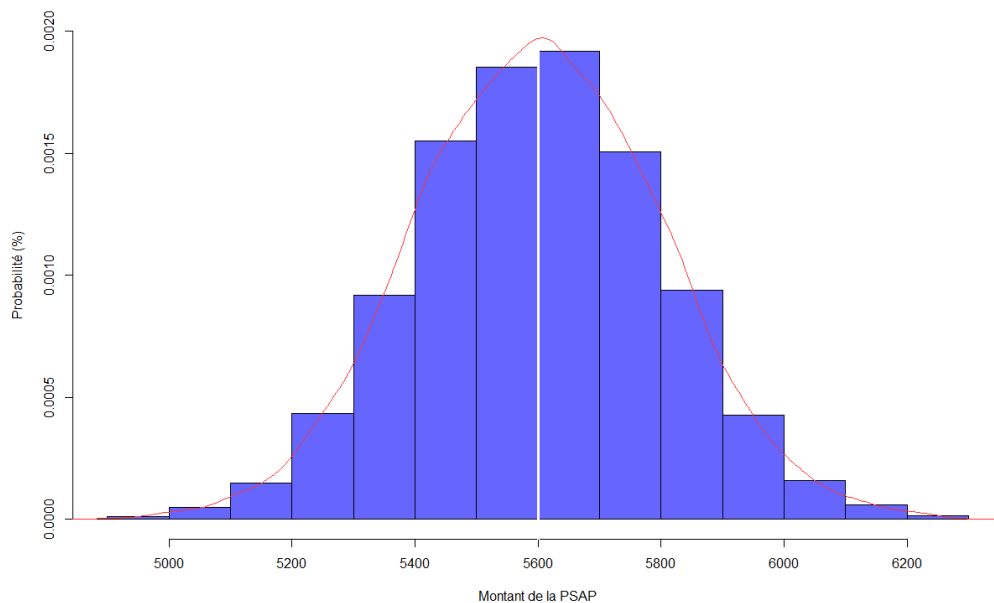


FIGURE 1.6 : Exemple de la densité de la PSAP avec la méthode Bootstrap

La méthode Bootstrap propose une première méthodologie afin d'obtenir une distribution de résultat. Elle reste cependant limitée par les hypothèses de Chain-Ladder. D'autres méthodes sur les données agrégées existent, mais ne seront pas abordées ici. Dans la suite, nous nous concentrerons sur des méthodes lignes à lignes.

1.4 Bilan des modèles de provisionnement et intérêt d'un modèle à densité

Aujourd'hui, les méthodes agrégées sont devenues des classiques de l'actuariat non-vie. Enseignées dans toutes les écoles et utilisées dans l'ensemble des entreprises du marché comme méthodologie de base, ces méthodes sont parfaitement maîtrisées. Simples d'utilisation et ne nécessitant que peu de données pour obtenir des résultats robustes, les nombreuses études à leurs sujets ont permis au secteur d'acquérir une documentation de qualité facilitant l'audit de ces modèles.

Initialement utilisées grâce à leur faible coût en puissance de calcul, ces méthodes pouvaient être appliquées à l'écrit. De nos jours, l'augmentation de la capacité de calculs due à l'arrivée des ordinateurs plus puissants et plus performants nous permettent d'aller chercher une information plus fine et de réduire la perte d'information liée à l'agrégation des données. Déjà utilisées en tarification, les données ligne à ligne sont encore trop peu utilisées en provisionnement. Afin de satisfaire les demandes des normes toujours plus exigeantes en termes de granularité de calcul, des nouvelles méthodes utilisant des données plus fines sont nécessaires. Cela ouvre aussi la porte à une meilleure estimation des provisions comportant l'application de traités de réassurance non-proportionnel. Un traité d'assurance non-proportionnel s'applique à l'échelle d'un sinistre, ce qui complexifie l'application sur des données agrégées.

Avec cette volonté, la recherche actuarielle concernant le provisionnement individuel s'est grandement développée. Cela se reflète par le nombre élevé de mémoires à ce sujet publiés aux cours des dernières années. Par exemple, NORBERG, 1993 revisité en 1999, propose une base souvent réutilisée s'appuyant sur la notion de processus de Poisson. D'autres travaux, comme le mémoire de GOMA, 2022 ou KUO, 2019, utilisent des réseaux de neurones sur données agrégées. Cependant, la quantité de données disponible et la puissance de calcul des nouveaux ordinateurs le permettant, des méthodes de *machine learning* ont été proposées avec notamment une approche liée aux arbres de régression PRIMEL, 2021, à des classifications CHARTREL, 2022, et aux réseaux de neurones profonds KUO, 2020. C'est dans la continuité de ces recherches que s'inscrit ce mémoire.

En majorité, les mémoires d'actuariat s'intéressant à ce sujet évaluent des réserves mais ne permettent pas d'obtenir la densité de celles-ci. La densité, permettant l'obtention de différents quantiles des réserves, est utile pour la norme Solvabilité 2 et IFRS 17. Il est donc nécessaire de l'intégrer aux modèles ligne à ligne.

Dans un premier temps, ce mémoire a pour objectif de proposer une implémentation d'un modèle de provisionnement ligne à ligne utilisant les réseaux de neurones permettant le calcul d'une densité des réserves. Le modèle retenu est celui de KUO, 2020. Dans un second temps, une application du modèle sera réalisée sur des données réelles de sinistres de responsabilité civile médicale.

1.5 Étude d'une base de données de responsabilité médicale

Les données contiennent l'ensemble des sinistres de la branche Responsabilité Civile Médicale en provenance d'un grand assureur français. La responsabilité médicale désigne l'obligation pesant sur les professionnels de santé de réparer le dommage causé par la mauvaise exécution d'un contrat de soins. Cette branche est une branche longue, ce qui peut poser problème aux méthodes de provisionnement classiques.

La base est une version mise à jour de celle utilisée pour les travaux de CHARTREL, 2022 reprise

elle-même des travaux de PRIMEL, 2021. L'étude reprend les traitements déjà implémentés en les mettant à jour. Les traitements comprennent notamment :

- Des vérifications de valeurs manquantes ;
- Contrôle de la cohérence entre les différentes dates d'un sinistre ;
- Vérification de la cohérence des montants de charges, paiements et réserves (Chargement - réserve - règlement = 0) ;
- Le retraitement de l'inflation.

Ces changements entraînent une perte de 0,23% des sinistres qui représentaient 0,29% de la charge totale. Ainsi la base obtenue comporte 75 000 lignes. Une ligne correspond à un sinistre dont la date de survenance est comprise entre 2000 et 2019. Certains sinistres ont eu lieu avant 2003. Or le 30 décembre 2002, la loi About (loi numéro 2000-1577 CODE DES ASSURANCES, 2002) modifie le régime juridique des contrats d'assurance de responsabilité civile médicale. Cette loi change le principe de "fait générateur" par celui de "base réclamation". Cela signifie qu'un assureur n'est plus tenu de réparer un sinistre si le fait générateur survient durant la période de couverture mais si la réclamation survient durant la période de couverture. De plus, la garantie même est changée pour favoriser la continuité de la couverture de l'assuré. Ainsi un médecin à son départ en retraite dispose encore de la garantie pendant 10 ans. La sinistralité des années de survenance précédant 2002 est différente. Le choix d'exclure les années de survenance 2000, 2001 et 2002 a été réalisé afin d'analyser uniquement le comportement des modèles sur une base à jour, ce choix sera détaillé dans l'analyse descriptive de la base.

Un réseau de neurones fonctionne généralement mieux si il a plus d'information disponible. A cet effet, l'ensemble des variables catégorielles sans valeurs non attribuées ont été gardées. D'autres retraitements comme l'arrondi à 0 des très faibles montants de paiements ou de charge (\downarrow). La nouvelle base comporte alors 63 000 lignes.

Chaque ligne (ou sinistre) possède des variables explicatives variant dans le temps (pour chaque année de développement) et des variables explicatives qui ne varient pas dans le temps. Les variables explicatives par années de développement sont :

- La charge (paiements + réserves) ;
- Le paiement des frais ;
- Les paiements hors frais ;
- La réserve estimée par le gestionnaire de sinistres ;
- Le statut du sinistre (ouvert ou fermé).

Les variables explicatives qui ne varient pas dans le temps sont :

- Les variables de dates : l'année d'ouverture, de survenance, de clôture. Ces données permettent aussi de créer la variable indiquant si le sinistre est ouvert ou fermé pour chaque année de développement connue ;
- Les variables de mois : le mois d'ouverture, de survenance ;
- Le type de sinistres selon le type de dommage : dommage de biens ou dommages corporels ;
- Le canal de distribution : courtage, réseau salarié ou agent ;
- Une variable binaire expliquant si le sinistre est tardif (année de survenance du sinistre différente de l'année d'ouverture du dossier) ;
- Le type de résolution du sinistre : à l'amiable ou devant un juge (litige) ;
- La catégorie du sinistre : 33 catégories dont 2 catégories comptent pour plus de 90% des données ;
- Famille : Une variable dont la signification n'est pas connue et qui a pour modalités 44,45 ou 46 ;
- La variable de coassurance : C'est une variable binaire indiquant si le sinistre est en coassurance

ou non ;

- La variable de réouverture : C'est une variable binaire qui indique si le sinistre a été ré-ouvert.

Notre base comporte donc 16 variables qui ne peuvent pas être calculées à partir d'autres variables. Ces 16 variables permettent les créations des variables de charge et de statut du sinistre.

Chapitre 2

Le *Bayesian Neural Network*

L'objectif de ce mémoire est de comprendre, implémenter et analyser une méthode de provisionnement ligne à ligne par réseaux de neurones. De surcroît, cette méthode doit permettre d'obtenir la distribution des réserves par année de survenance. Cependant, la connaissance unique de la distribution de l'ultime ne suffirait pas pour plusieurs raisons :

- Afin de satisfaire les critères de la norme solvabilité 2, le modèle doit permettre de prédire la distribution de la charge à 1 an.
- Afin de satisfaire les critères de la norme IFRS 17, il est nécessaire de disposer des prédictions par année de survenance et développement, car elles permettent de calculer une estimation des *cashflows* à actualiser nécessaire dans la vision économique souhaitée par la directive IFRS 17.

Dans ce mémoire, les IBNYR ne seront pas pris en compte. Tout montant de PSAP, de réserve ou de charge est donc sous-entendu « hors IBNYR ».

Pour débiter ce chapitre, une définition des différents éléments d'un réseau de neurones sera donnée pour un perceptron, un réseau de neurones multicouches et un LSTM. Ensuite, les différents moyens permettant d'ajouter de l'aléa dans notre modèle pour ainsi obtenir une distribution seront expliqués. S'en suivra un état de l'art des modèles étudiés dans ce mémoire. Pour conclure, ce chapitre décrira les processus de prédiction et de calibration du modèle, donnant l'ensemble des concepts permettant l'application d'un modèle de réseau de neurones tel que celui de KUO, 2020.

2.1 Un réseau de neurones

Un réseau de neurones artificiels ou réseau neuronal artificiel est un système dont la conception est à l'origine schématiquement inspirée du fonctionnement des neurones biologiques, et qui par la suite s'est rapproché des méthodes statistiques. Polyvalents, mais plus difficilement explicables, les réseaux de neurones permettent aussi bien d'effectuer une classification qu'une régression. Suivant notre objectif de provisionnement, deux réseaux de neurones paramétriques, supervisés transcrivant une régression seront ici étudiés : le perceptron et le réseau multicouche.

Dans cette section, les notations suivantes seront utilisées :

- Indice de l'année de survenance : $i \in [1 : I]$, avec I le nombre d'années de survenance différentes de la base de données ;
- Indice de l'année de développement : $j \in [1 : J]$, $J \leq I$, avec J le nombre d'années de

développement maximum de la base de données ;

- Indice de la ligne dans base de données : $n \in [1 : N]$, avec N le nombre de lignes maximum ;
- Indice de la variable en *input* du modèle : $k \in [1 : K]$, avec K le nombre de variables ;
- Numéro de chaque modalité unique d'une variable : $u^k \in [1 : U^k]$, avec U^k le numéro de la dernière modalité de la variable K ;
- Numéro de la couche du réseau (*layers* en anglais) : $l \in [1 : L]$, avec L le nombre maximum de couches du réseau.

Le perceptron est un réseau de neurones composé d'un seul neurone. C'est l'exemple le plus simple et il nous servira d'introduction aux différents éléments nécessaires à la compréhension des réseaux de neurones plus complexes. Commençons par observer le schéma d'un perceptron sur la figure 2.1.

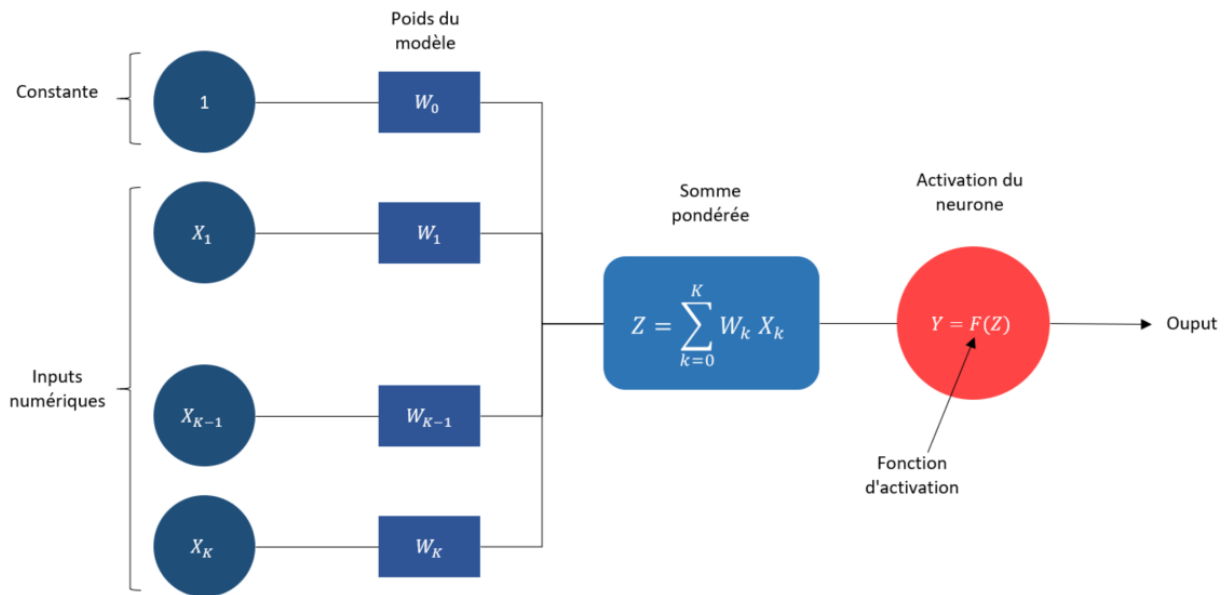


FIGURE 2.1 : Schéma du Perceptron

Un perceptron est composé d'inputs, de poids, d'une fonction d'activation et d'un *output*.

- L'input d'un réseau de neurones : il est composé de variables au format **numérique**. La première étape d'un réseau de neurones consiste à multiplier chaque variable par un poids. Cette multiplication est possible uniquement pour des nombres. Pour des variables qualitatives, il faut appliquer une méthode pour changer le format.
- Les poids d'un perceptron : il y a un poids associé à chaque variable explicative du modèle et un poids associé à une constante (la constante peut être considérée comme une variable fixée à un, elle n'est pas comptabilisée comme une variable supplémentaire, car considérée comme implicite). Il y a donc $K + 1$ poids. L'exemple suivant nous donne une intuition de l'importance de la constante (voir figure 2.2). Un perceptron avec une variable explicative et une fonction d'activation linéaire est l'équivalent à l'application d'une régression linéaire de type $W \times X + B$. Or essayer d'entraîner une droite passant par l'origine à suivre des données ne passant pas par l'origine s'avère peu concluant comme le montre le graphique. Ajouter une constante est donc indispensable au réseau de neurones afin d'obtenir le meilleur résultat.
- La fonction d'activation : son impact est double, elle modifie les bornes de sortie de l'*output*, augmentant le contrôle que nous avons de l'*output* du neurone et permet de dépasser le cadre des problèmes linéaires.

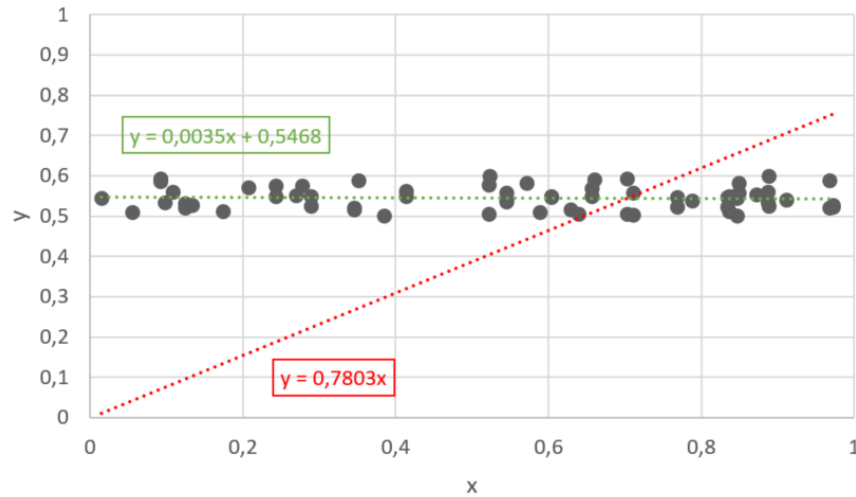


FIGURE 2.2 : Schéma de l'importance du terme constant sur 2 régressions par réseau de neurones linéaires

- L'*output* : sous forme numérique, dans le cadre d'une régression il estime un montant, une valeur dont les bornes dépendent de la fonction d'activation.

Soit $n \in [1 : N]$ une ligne de notre base de données, composée de K variables explicatives X^k avec $k \in [1 : K]$, mise en *input* d'un perceptron. La prédiction d'un *input* est obtenue par la formule

$$\hat{y} = F\left(\sum_{k=0}^K W_k X^k\right).$$

Un perceptron se résume donc à une fonction (2.1) qui à chaque ligne n associe la prédiction

$$\hat{y}_n = \text{Perceptron}(X_n, W, F(\cdot)) = F\left(\sum_{k=0}^K W_k X_n^k\right). \quad (2.1)$$

"Perceptron" est le nom attribué à un réseau de neurones composé d'un unique neurone. Un réseau de neurones est donc une combinaison de perceptron mis les uns à la suite des autres. Ces neurones peuvent être mis en parallèle ou bout à bout. Des neurones mis en parallèle ont les mêmes *inputs*, ils appartiennent à la même couche. Un neurone mis bout à bout d'un autre neurone a pour *input* la sortie du neurone précédent. Il crée une nouvelle couche. Un réseau de neurones peut avoir un grand nombre de couches. Afin de les distinguer, la couche finale du modèle est nommée la couche de sortie et les couches intermédiaires sont nommées couches cachées. Un réseau de neurones est dit dense si chaque *input* d'un neurone d'une couche cachée est composé de l'ensemble des *outputs* des neurones de la couche précédente. La figure 2.3 présente le schéma d'un réseau dense avec 3 couches cachées.

Un réseau dense est composé comme le perceptron d'inputs, de poids et d'une fonction d'activation donnant l'*output*. Il diffère avec l'ajout des couches cachées qui ont chacune leurs poids, et la même fonction d'activation. Le fonctionnement global reste le même que le perceptron. La prédiction de chaque couche cachée d'un *input* d'un réseau dense à 3 couches cachées est obtenue par la formule

$$\begin{cases} Y_c^1 = F_1\left(\sum_{k=0}^K W_{k,c}^0 X_n^k\right) = F_1(z_c^1) \text{ si } c \neq 0 \text{ (couche cachée 1);} \\ Y_b^2 = F_1\left(\sum_{c=0}^3 W_{c,b}^1 Y_c^1\right) = F_1(z_b^2) \text{ si } b \neq 0 \text{ (couche cachée 2);} \\ Y_a^3 = F_1\left(\sum_{b=0}^3 W_{b,a}^2 Y_b^2\right) = F_1(z_a^3) \text{ si } a \neq 0 \text{ (couche cachée 3).} \end{cases}$$

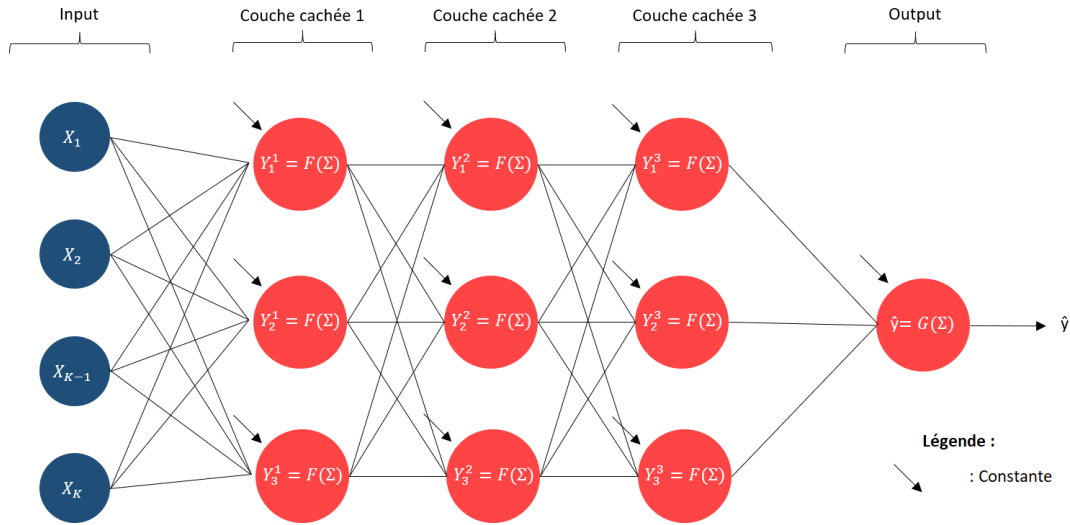


FIGURE 2.3 : Schéma d'un réseau de neurones dense à 3 couches cachées

Et la prédiction de la couche de sortie par la formule 2.2

$$\hat{y} = RN(X_n, W, F_1(\cdot), F_2(\cdot)) = F_2\left(\sum_{a=0}^3 W_{a,1}^3 Y_a^3\right), \quad (2.2)$$

avec

- $W_{k,c}^{l_1}$ est le poids de la variable k de la couche l_1 au neurone c de la couche $l_1 + 1$.
- Y_a^l est la valeur du neurone a de la couche l .
- $X_n^0 = Y_0^1 = Y_0^2 = Y_0^3 = 1$ et correspond à la flèche indiquant une constante comme nous l'avons vu dans le perceptron.
- F_1 correspond à la fonction d'activation des couches cachées et F_2 à la fonction d'activation du neurone de sortie.

Après ce tour d'horizon rapide des éléments du perceptron et d'un réseau de neurones multicouches dense, décrivons ces éléments un à un.

2.1.1 Les méthodes de traitement des *inputs*

Comme énoncé précédemment, les *inputs* d'un réseau de neurones doivent être au format numérique. Par conséquent, il faut retraiter les variables qualitatives (aussi appelées variables catégorielles).

Il faut distinguer 2 types de variables qualitatives : les variables qualitatives ordinales et les variables qualitatives nominales.

Les variables ordinales

Les variables ordinales sont des variables pouvant être ordonnées. Le retraitement doit conserver cette particularité. La méthode la plus connue consiste à attribuer à chaque modalité une valeur numérique

en conservant l'ordre. Les modalités d'une variable qualitative sont les différentes valeurs uniques que la variable peut prendre. Considérons l'exemple des mentions du baccalauréat sur la figure 2.4.

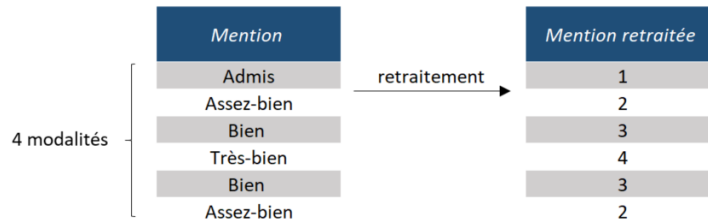


FIGURE 2.4 : Numérisation d'une variable qualitative ordinale

La variable mention au bac a 4 modalités. A chaque modalité est attribuée une valeur différente en gardant l'ordre établi. L'intervalle entre chaque valeur n'est pas forcément unitaire comme sur l'exemple ci-dessus. Une méthode consiste à lier l'intervalle avec la proportion de valeur dans le portefeuille. Soit U modalités distinctes de la variable X ordonnées de mod_1 à mod_U , alors la numérisation de la variable s'effectue grâce à la formule

$$\forall u \in [1, U], \text{ num}(mod_u) = \text{ num}(mod_{u-1}) + \frac{\#(X = mod_u)}{\#X},$$

Avec

- $\text{ num}(mod_u)$: est la transformation numérique de la modalité numérotée u de X . Nous fixons $\text{ num}(mod_1) = 0$ et $\text{ num}(mod_U) = 1$.
- mod_u : la modalité numérotée u de X ,
- $\#(X = mod_u)$ (resp. $\#X$) : le cardinal de la modalité numérotée u de X (resp. le nombre d'éléments de X toutes modalités confondues). L'idée est de conserver les proportions. Dans le cas des mentions au baccalauréat, une autre idée consiste à prendre les pourcentages moyens des 5 dernières années de chacune des mentions afin de conserver la distance entre chacune des variables. La modification des distances est spécifique à chaque variable et ne peut s'effectuer qu'au cas par cas.

Les variables Nominales : méthode *OHE*

Pour une variable catégorielle nominale, une binarisation peut être effectuée (aussi appelé factorisation ou *One hot encoding (OHE)* en anglais). Le *OHE* consiste en la création d'une nouvelle variable binaire pour chacune des modalités de notre variable. Une application concrète est réalisée sur la figure 2.5 pour la variable de couleur des yeux.

Il y a 2 types de binarisations. La première binarisation consiste à transformer chaque modalité en une variable binaire. La couleur marron correspond de manière ensembliste à la modalité "Non(bleu ou vert)". Il y a donc une corrélation négative de 100% entre la couleur marron et l'ensemble de couleur vert et bleu. L'ajout de la variable binaire marron répète l'information. L'impact d'une corrélation parfaite entre les variables peut réduire l'apprentissage d'un réseau de neurones. Il est de bonne pratique d'effectuer la factorisation de U modalités à l'aide de $U - 1$ variables binaires comme sur la factorisation à 2 facteurs du schéma ci-dessus.

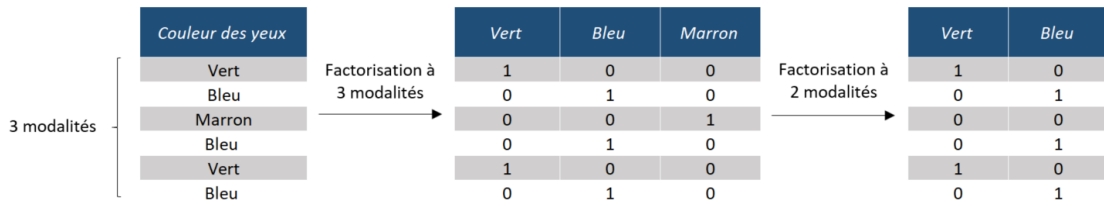


FIGURE 2.5 : Binarisation d'une variable qualitative nominale

Les variables Nominales : méthode *Impact encoding*

Une deuxième méthode de retraitement de variables qualitatives nominales est l'*Impact encoding*. Lorsque le nombre de catégories devient très grand, l'encodage par binarisation peut devenir coûteux, car cela augmente le nombre de variables et par conséquent le nombre de poids du modèle, le temps d'apprentissage et la complexité du réseau de neurones. L'*Impact encoding* consiste à caractériser les catégories par le lien qu'elles entretiennent avec la variable cible. Pour un problème de régression ayant pour variable cible y , une variable catégorique nominale X possédant U catégories mod_1, \dots, mod_U , chaque catégorie mod_u est encodée par sa valeur d'impact

$$impact(mod_u) = \mathbb{E}[y | X = mod_u] - \mathbb{E}[y].$$

Avec :

- $\mathbb{E}[y | X = mod_u]$ correspond à l'espérance de la cible Y sachant que la variable X est fixée à la modalité mod_u . Pour un jeu d'entraînement de taille N contenant des échantillons $\{(x_n, y_n), 1 \leq n \leq N\}$, indépendants et identiquement distribués, l'estimateur de cette espérance est la moyenne des valeurs de y_n pour lesquelles la modalité x_n est égale à m_u définit par

$$\mathbb{E}[y | \widehat{X} = mod_u] = \frac{1}{N_u} \sum_{n \in S_u} y_n.$$

Avec

- S_u : l'ensemble des indices n des observations telles que x_n soit égal à mod_u .
- N_u : le cardinal de l'ensemble S_u .
- $\mathbb{E}[y]$ l'espérance de y s'estime par la formule classique

$$\mathbb{E}[\widehat{y}] = \frac{1}{N} \sum_{n=1}^N y_n.$$

Une application concrète de l'*Impact encoding* est réalisée sur la variable « voie de distribution » sur la figure 2.6.

Cette variable est qualitative, nominale (car elle ne peut pas être ordonnée). Après application, notre nouvelle variable « impact de la voie de distribution » viendra remplacer la variable « voie de distribution » dans le réseau de neurones. Un des défauts principaux de cette méthode est la nécessité d'avoir une variable cible unique numérique ce qui n'est pas toujours le cas. Un autre défaut conséquent a lieu dans le cas où 2 variables ont le même impact final, cette transformation les assimilera ensemble. Une autre méthode existe alors pour pallier ce défaut.

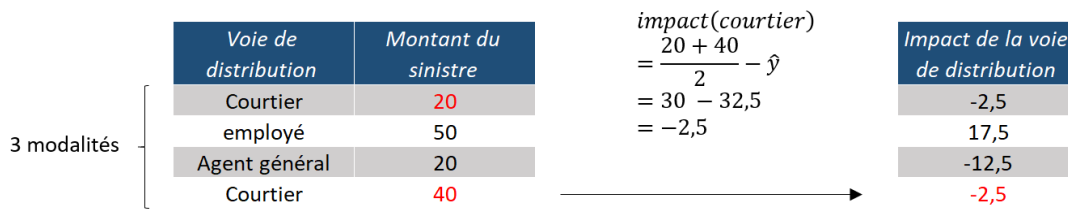


FIGURE 2.6 : *Impact encoding* d'une variable qualitative nominale pour une base de 4 lignes de sinistres

Les variables nominales : méthode de Embedding layer

Une autre solution à l'OHE est l'*embedding layer*. L'OHE présente deux inconvénients majeurs :

- Pour les variables ayant une grande cardinalité avec de nombreuses catégories uniques, le nombre de variables est trop important.
- Si 2 modalités ont des comportements similaires, l'OHE ne prendra pas cette information pour les assimilés ensemble.

L'*embedding layer* remplace chacune des variables catégorielles par un point de \mathbb{R}^2 . Une grande cardinalité ne changera pas la dimension 2 de l'espace de sortie. Sur le schéma 2.7 ci-dessous, la méthode de l'*embedding layer* est représentée avec l'exemple de la voie de distribution précédemment utilisé. Cette méthode se déroule en 2 étapes :

- Une numérisation de la variable en attribuant un numéro à chaque modalité (en pratique commençant à 0).
- L'*embedding layer* va associer 2 poids à chaque modalité. Ces poids seront ensuite entraînés en même temps que les poids du réseau de neurones et par le même processus.

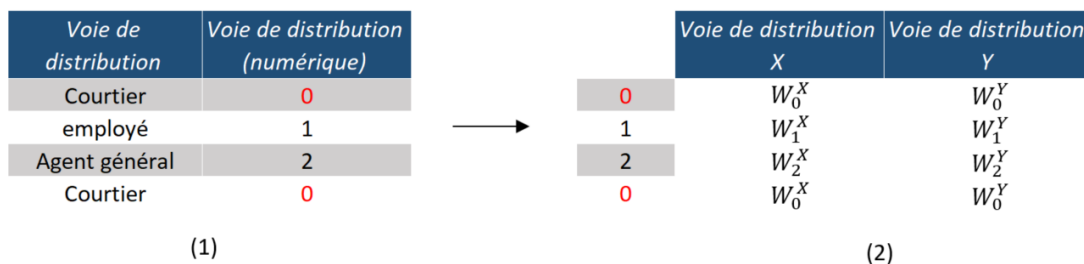


FIGURE 2.7 : *Embedded layer* d'une variable qualitative nominale

Cette méthode fonctionne aussi en dimension supérieure à 2, utile pour des variables ayant de nombreuses modalités. Dans nos modèles, la limite sera la dimension 2. Pour plus de détails, se référer à l'article KOEHRSEN, 2018.

Retraitement final des variables

Après avoir changé le format des variables catégorielles en variables numériques, les inputs (et les outputs observés dans le cadre d'une régression) sont transformés. Il existe 2 transformations couramment appliquées :

- La première transformation consiste en la normalisation 2.3 (centrer réduire) des données

$$\hat{X}_n^i = \frac{X_n^i - \mu(X^i)}{\sigma(X^i)}. \quad (2.3)$$

Avec

- $\mu(X^i)$ est estimé par $\frac{1}{N} \sum_{b=1}^N X_b^i$;
- $\sigma(X^i)$ est estimé par $\sqrt{\frac{1}{N} \sum_{b=1}^N (X_b^i - \mu(X^i))^2}$.

Elle remplace les *inputs* par leur écart à la moyenne en matière d'écart-types.

- La deuxième transformation consiste à appliquer uniquement l'étape de réduction des données 2.4

$$\hat{X}_n^i = \frac{X_n^i}{\sigma(X^i)}. \quad (2.4)$$

Avec

- $\mu(X^i)$ est estimé par $\frac{1}{N} \sum_{b=1}^N X_b^i$;
- $\sigma(X^i)$ est estimé par $\sqrt{\frac{1}{N} \sum_{b=1}^N (X_b^i - \mu(X^i))^2}$.

L'avantage de cette méthode est de conserver des valeurs positives si l'*input* est positif.

Appliquer une transformation est conseillé, car la zone de variation des fonctions d'activation des couches cachées est souvent bornée entre -3 et 3 (2.5). Formellement pour une fonction d'activation bornée,

$$F\left(\sum_{k=0}^K W_k X_n^k\right) = \begin{cases} C_1 \text{ (généralement } -1 \text{ ou } 0) & \text{si } \sum_{k=0}^K W_k X_n^k \leq -3; \\ C_2 \text{ (généralement } 1) & \text{si } \sum_{k=0}^K W_k X_n^k \geq 3. \end{cases} \quad (2.5)$$

Dit plus explicitement, les dérivées des fonctions d'activation possibles s'annulent en dehors de la borne $[-3, 3]$. Si les *inputs* ne sont pas réduits, et par exemple de l'ordre de 10^6 , alors pour avoir des variations subtiles de résultats, il faut des poids de l'ordre de 10^{-6} à 10^{-10} ce qui est lent et sujet à des erreurs d'approximation durant l'entraînement. En effet, les algorithmes de descente de gradient utilisés pour la détermination des poids sont moins performants et demandent plus de temps pour obtenir un tel niveau de précision. Il est beaucoup plus simple de réduire les données initialement.

Il est aussi conseillé d'appliquer la transformation de réduction aux *outputs* réels pour la même raison.

Cas de variables incomplètes

Dans de nombreuses bases de données, les variables ne sont pas complètes. Ces valeurs manquantes posent un problème majeur dans de nombreux modèles. Il existe différents moyens de retraiter ces valeurs

- La suppression de la variable. Dans le cas où la variable possède trop de valeurs manquantes, si elle n'est pas d'une forte importance alors il suffit de ne pas prendre la variable. (L'importance peut se mesurer en entraînant le modèle sur les lignes connues afin d'analyser l'amélioration des résultats avec ou sans celles-ci) ;
- La suppression des lignes inconnues. Dans le cas où la base dispose d'un très grand nombre de lignes, et si la proportion de valeurs manquantes est faible, c'est un choix à privilégier.
- Modifier la variable. Plusieurs modifications peuvent être faites suivant l'information disponible sur ces valeurs inconnues (remplacer les valeurs manquantes par une moyenne ou 0 suivant la variable, voir effectuer une prédiction de cette variable à l'aide d'un modèle de classification)

Dans la base de données, la variable LOB (*Line of business*) est composée de 2 valeurs : ensemble de contrats et autres contrats. Cependant elle a aussi des valeurs manquantes c'est-à-dire de modalité NA (Non attribuée). Ne sachant pas précisément la définition de cette variable, la procédure recommande de la retirer. Cependant, c'est initialement une variable binaire et le traitement des NA d'une variable binaire peut être effectué par un changement de format simple avec l'utilisation d'un réseau de neurones, contrairement aux traitements des NAs lors du traitement d'une variable multicatégorielle.

Dans le cas de variables composées de nombreuses modalités, il faudra se référer à l'article de LITTLE, RODERICK J. A., 2002. Dans le cas binaire, la variable sera retraitée comme sur le schéma 2.8

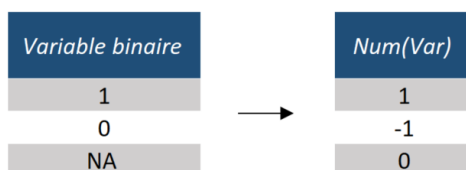


FIGURE 2.8 : Retraitement d'une variable comportant des modalités non attribuées

Sur ce schéma, la modalité NA a été modifiée en 0, car la particularité des réseaux de neurones fait que le poids n'a pas d'impact s'il est multiplié par 0. Les poids uniquement reliés à cette variable ne seront pas actualisés pour des *inputs* ayant la valeur 0. Dans la formule de descente de gradient, le terme dérivé par rapport au poids relatif à la variable LOB vaut 0, l'apprentissage n'a donc pas lieu sur les poids concernés.

Autrement dit, dans le cas d'une variable binaire sans NAs, lorsque la variable vaut 0, les poids uniquement reliés à cette variable ne sont pas appris. Le modèle considère donc ce cas comme le cas par défaut et lorsque la variable vaut 1, le modèle modifie le poids afin que l'impact de la modification de ce poids corresponde à l'impact de la valeur 1 sur cette variable par rapport au cas par défaut.

Dans le cas où la variable binaire comporte des NAs, le cas par défaut devient le cas NAs. Cependant, vu que le poids n'est pas appris pour la variable NAs qui est fixé à 0, le modèle apprend ici les impacts de la variable par rapport à chacune des modalités sans tenir compte de leur complémentarité. Un autre moyen consiste à diviser la variable en 2 variables binaires dont la valeur 0 correspond à une modalité NA.

2.1.2 État de l'art des fonctions d'activations

La fonction d'activation d'un perceptron est la fonction appliquée à la somme pondérée des poids et des inputs pour obtenir l'output du neurone. Dans le cas du perceptron, il n'y en a qu'une, c'est la fonction de sortie. Dans le cas d'un réseau de neurones avec plusieurs couches cachées, les neurones des couches cachées ont tous la même fonction d'activation, cependant la fonction d'activation d'agrégation peut être différente. L'utilisation d'une fonction d'activation est indispensable pour 2 raisons :

- Elle maintient la valeur de la sortie du neurone dans une certaine limite. Ceci est important, car l'entrée dans la fonction d'activation est $\sum_{k=0}^K W_k X_n^k$. Cette valeur, si elle n'est pas restreinte à une certaine limite, peut atteindre une magnitude très élevée, en particulier dans le cas de réseaux de neurones très profonds qui ont des millions de paramètres.
- Elle ajoute de la non-linéarité dans un réseau de neurones. C'est la caractéristique la plus importante d'une fonction d'activation. Si la fonction d'activation est la fonction identité alors le problème du perceptron 2.6 devient

$$\hat{y} = \sum_{k=0}^K W_k X_n^k, \quad (2.6)$$

ce qui est l'expression d'une régression linéaire. Si la fonction d'activation de chaque neurone reste la fonction identité, l'ajout de couches cachées au réseau de neurones augmente uniquement l'ordre de la régression linéaire, ce qui est insuffisant pour la résolution de problèmes complexes.

Une fonction d'activation doit satisfaire les conditions suivantes :

- De dérivée centrée en zéro : la sortie de la fonction d'activation doit être symétrique par rapport à l'origine afin que les gradients ne tendent dans une direction particulière (plus exactement le point de rupture de la dérivée doit se trouver en 0).
- Coût de calcul : les fonctions d'activation sont appliquées après chaque neurone de chaque couche et calculées des millions de fois dans un réseau profond. Elles doivent donc être peu coûteuses en temps et en mémoire. Pour un perceptron ou un réseau multicouche dense, ce point peut être ignoré, mais le choix de la fonction d'activation aura un impact direct sur le temps d'apprentissage pour le modèle présenté après.
- Différentiable : un réseau de neurones est formé à l'aide du processus de descente de gradient, et les couches du modèle doivent donc être différentiables ou du moins différentiables en partie. Il s'agit d'une condition nécessaire pour qu'une fonction puisse fonctionner comme fonction d'activation.

Il existe de nombreuses fonctions d'activations, mais seulement un faible nombre d'entre-elles sont couramment appliquées. Voici les fonctions d'activations non linéaires les plus utilisées :

- La fonction sigmoïde 2.7 (voir figure 2.9). Elle est définie par

$$\text{sigmoïde}(x) = \frac{1}{1 + \exp(-x)}. \quad (2.7)$$

Et sa dérivée 2.8 par

$$\text{sigmoïde}'(x) = \frac{\exp(-x)}{(1 + \exp(-x))^2}. \quad (2.8)$$

C'est une fonction bornée de $\mathbb{R} \rightarrow [0, 1]$. Sa dérivée existe et est symétrique en 0. Si elle est utilisée sur de nombreuses couches cachées, alors l'apprentissage s'arrête à cause du *Vanishing Gradient problem*.

Cette fonction d'activation est peu utilisée comme sur la couche de sortie sauf pour des problèmes de classification binaire. Cependant, c'est un élément clé des couches cachées des réseaux de neurones complexes comme les réseaux de neurones récurrents.

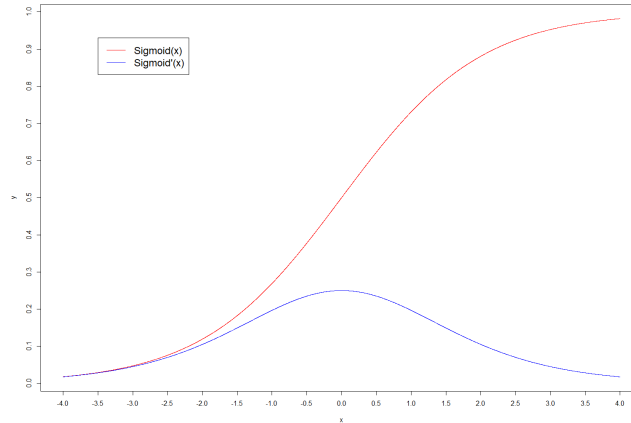


FIGURE 2.9 : Courbe de la fonction d'activation sigmoïde et de sa dérivée

- La fonction softmax : elle permet de probabiliser les résultats de plusieurs sorties de neurones. Elle est généralement appliquée à l'étape finale d'un problème de classification à choix multiples. L'exemple de la figure 2.10 montre que la fonction d'activation softmax permet de transformer la sortie de 5 perceptrons en parallèle en un vecteur de probabilité. C'est une fonction bornée de

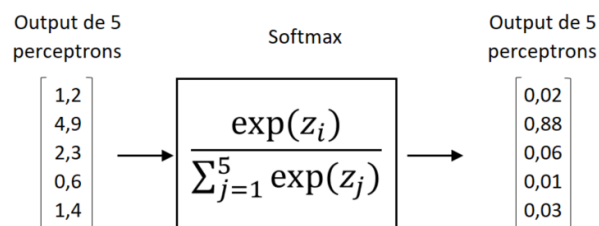


FIGURE 2.10 : Schéma de l'application d'un softmax à la sortie de 5 perceptrons parallèles

$\mathbb{R}^n \rightarrow [0, 1]^n$. Ce type de fonction pourrait être utilisée dans notre cas pour une classification en classes de sinistres fermés ou ouverts pour chaque année de développement.

- La fonction tangente hyperbolique ou \tanh (voir figure 2.11). Elle est définie par

$$\tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)},$$

et sa dérivée par

$$\tanh'(x) = 1 - \tanh^2(x).$$

C'est une fonction bornée de $\mathbb{R} \rightarrow [-1, 1]$. Le point de rupture de sa dérivée est en 0. Elle est cependant coûteuse en temps de calcul et surtout elle ne résout pas le problème de la disparition du gradient en cas de nombreuses couches.

- La fonction *ReLU* (*Rectified Linear Unit*) (voir figure 2.12). Elle est définie par

$$ReLU(x) = \max(0, x) = \begin{cases} x & \text{si } x > 0; \\ 0 & \text{si } x \leq 0. \end{cases}$$

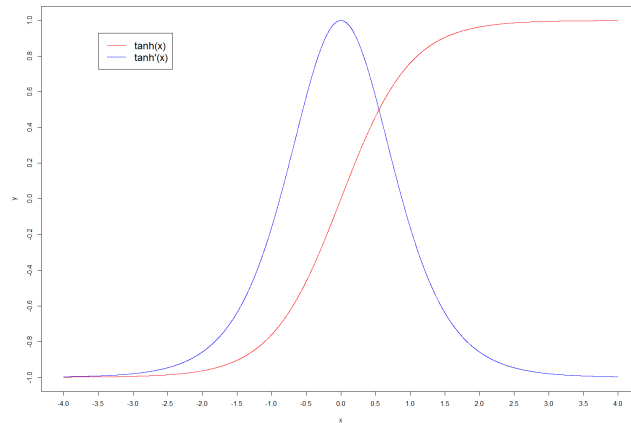


FIGURE 2.11 : Courbe de la fonction d'activation tanh et de sa dérivée

et sa dérivée par

$$ReLU'(x) = \begin{cases} 1 & \text{si } x > 0; \\ 0 & \text{si } x < 0; \\ \text{non défini} & \text{si } x = 0. \end{cases}$$

En effet, la limite des dérivées inférieure et supérieure à 0 n'est pas égale.

C'est une fonction non bornée de $\mathbb{R} \rightarrow [0, \infty]$. Elle est facile à calculer et ne cause pas le problème de la disparition du gradient ce qui en fait une fonction d'activation largement utilisée pour la couche de sortie. Le seul problème est que sa dérivée n'est pas centrée en zéro. Elle souffre aussi du problème du "ReLU mourant". Puisque la sortie est nulle pour toutes les entrées négatives. Cela fait que certains neurones "meurent" complètement (reste toujours à 0) et n'apprennent rien.

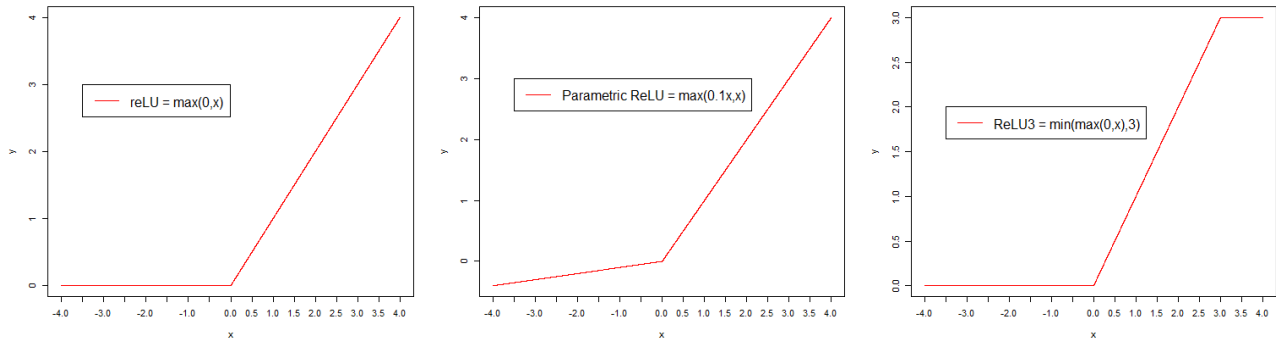
Un autre problème avec *ReLU* est que chaque neurone de chaque couche cachée peut tendre vers l'infini. Cela conduit parfois à des neurones avec des poids trop grands inutilisables, car de faibles variations vont trop changer le résultat.

La version du *Leaky ReLU* ($f(x) = \max(\alpha x, x)$) résout dans une certaine mesure le problème du "ReLU mourant". Cependant si nous fixons α à 1, alors *Leaky ReLU* deviendra une fonction linéaire et ne sera d'aucune utilité. Par conséquent, la valeur de α n'est jamais fixée près de 1. La version du *ReLU3* ($f(x) = \min(\max(0, x), 3)$) (ou *ReLU6*) permet d'éviter de faire exploser la fonction d'activation et donc d'empêcher les gradients d'aller vers l'infini. Une idée naturelle serait de combiner plusieurs formules pour résoudre tous les problèmes en même temps, mais en pratique, cette combinaison n'est pas implémentée.

2.1.3 Les poids et leur initialisation

Les poids sont le fruit de l'entraînement du modèle. Ils représentent l'apprentissage et vont permettre d'extraire l'information de l'input. Ils appartiennent à \mathbb{R} mais sont généralement proches de 0 lorsque les *inputs* et *outputs* sont normalisés. En effet, comme les dérivées des fonctions d'activation bornées s'annulent en dehors de la borne $[-3, 3]$, l'apprentissage ne sera pas plus grand si la somme pondérée est très grande (positivement ou négativement).

Les poids d'un réseau de neurones sont souvent représentés sous la forme d'une matrice, ou pour

FIGURE 2.12 : Courbe de la fonction d'activation *ReLU*, *leaky ReLU* et *ReLU3*

le perceptron d'un vecteur. Reformulons le problème du perceptron

$$Y = [W_0 \ W_1 \ \cdots \ W_{K-1} \ W_K] \times \begin{bmatrix} X_n^0 \\ X_n^1 \\ \cdots \\ X_n^{K-1} \\ X_n^K \end{bmatrix} = [W_0 \cdot X_n^0 + W_1 \cdot X_n^1 + \cdots + W_{K-1} \cdot X_n^{K-1} + W_K \cdot X_n^K].$$

Matriciellement,

$$\hat{y} = F(Z).$$

Pour un réseau multicouche, c'est plus complexe. Matriciellement, le problème peut se reformuler suivant 3 cas :

- Entre l'*input* et les couches cachées ,

$$Y^1 = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = F \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix}, \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} W_{0,1}^{0,1} & W_{0,1}^{1,1} & \cdots & W_{0,1}^{K-1,1} & W_{0,1}^{K,1} \\ W_{0,1}^{0,2} & W_{0,1}^{1,2} & \cdots & W_{0,1}^{K-1,2} & W_{0,1}^{K,2} \\ W_{0,1}^{0,3} & W_{0,1}^{1,3} & \cdots & W_{0,1}^{K-1,3} & W_{0,1}^{K,3} \end{bmatrix} \times \begin{bmatrix} 1 \\ X_n^1 \\ \cdots \\ X_n^{K-1} \\ X_n^K \end{bmatrix};$$

- Entre 2 couches cachées, $\forall l_1, l_2 \in [1, 3]$ sachant $l_1 < l_2$:

$$Y^2 = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = F \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix}, \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} W_{l_1, l_2}^{0,1} & W_{l_1, l_2}^{1,1} & W_{l_1, l_2}^{2,1} & W_{l_1, l_2}^{3,1} \\ W_{l_1, l_2}^{0,2} & W_{l_1, l_2}^{1,2} & W_{l_1, l_2}^{2,2} & W_{l_1, l_2}^{3,2} \\ W_{l_1, l_2}^{0,3} & W_{l_1, l_2}^{1,3} & W_{l_1, l_2}^{2,3} & W_{l_1, l_2}^{3,3} \end{bmatrix} \times \begin{bmatrix} 1 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix};$$

- Entre une couche cachée et l'*output* ,

$$\hat{y} = G([\hat{z}]), [\hat{z}] = \begin{bmatrix} W_{3,4}^{0,1} & W_{3,4}^{1,1} & W_{3,4}^{2,1} & W_{3,4}^{3,1} \end{bmatrix} \times \begin{bmatrix} 1 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix}.$$

Le problème peut être écrit en une seule équation matricielle, mais serait trop conséquent en termes de notations. Les équations imbriquées ci-dessous permettent d'en avoir une meilleure compréhension.

$$\hat{y} = G(W_{3,4} \cdot F(W_{2,3} \cdot F(W_{1,2} \cdot X))) \quad (2.9)$$

Un modèle de réseaux de neurones apprend à l'aide d'un algorithme d'optimisation appelé descente de gradient stochastique qui modifie progressivement les poids du réseau pour minimiser une fonction de perte (la fonction sera définie dans la partie suivante 2.10).

Cet algorithme d'optimisation nécessite un point de départ dans l'espace des valeurs de poids possibles à partir duquel commencer le processus d'optimisation. L'initialisation des poids est une procédure permettant de fixer les poids d'un réseau de neurones à de petites valeurs aléatoires qui définissent le point de départ d'apprentissage du modèle.

Chaque fois qu'un réseau de neurones est initialisé avec un ensemble différent de poids, cela change le point de départ pour le processus d'optimisation, et ainsi change potentiellement l'ensemble final des poids. Le modèle obtenu aura donc des performances différentes selon l'initialisation des poids.

Logiquement, l'idée la plus simple est d'initialiser les poids à 0 et de laisser le modèle les apprendre. Cependant, lorsque tous les poids sont initialisés à 0, la dérivée par rapport à la fonction de perte est la même pour chaque poids dans chaque couche, donc tous les poids auront la même valeur dans les itérations suivantes. Les unités cachées deviennent symétriques et continuent durant l'entièreté de l'apprentissage et le modèle n'obtient pas de meilleure performance qu'un modèle linéaire. De plus, initialiser tous les poids à 0 annule l'impact des biais, le réseau se résume à un modèle linéaire sans biais.

Ne pouvant pas initialiser l'ensemble des poids à 0, une autre idée est d'initialiser les poids aléatoirement. Historiquement cette méthode est celle retenue et utilisée. Par convention, le poids du biais est fixé à 0. Par exemple, les poids étaient initialisés selon diverses lois uniformes telles que :

- Une loi uniforme $\mathcal{U}[-0, 3, 0, 3]$;
- Une loi uniforme $\mathcal{U}[0, 1]$;
- Une loi uniforme $\mathcal{U}[-1, 1]$.

Les poids sont initialisés avec des valeurs très faibles, car sinon le terme z_n de 2.11 devient significativement plus élevé et si une fonction d'activation bornée est appliquée, la pente du gradient change très lentement. La conséquence est que l'apprentissage prend beaucoup de temps. Cependant, si les poids sont initialisés avec des valeurs trop faibles, nous retombons dans le cas d'initialisation avec 0.

De plus, le problème de la disparition du gradient se pose toujours. *ReLU* et *leaky ReLU* résolvent le problème mais il persiste pour les fonctions d'activation sigmoïde et tanh.

Il existe d'autres méthodes d'initialisation des poids. Leurs usages dépendent de la fonction d'activation choisie dans les couches cachées et elles ont pour objectif d'améliorer le temps et la vitesse de convergence du réseau par rapport à une uniformisation aléatoire :

- Pour les fonctions d'activation sigmoïde et tanh, la fonction d'initialisation des poids utilisée est celle de "glorot et Xavier". Il en existe 4 : "Xavier uniforme", "Xavier normale", "Glorot Uniforme" et "Glorot normale" similaires deux à deux.
 - "Xavier uniforme" initialise les poids aléatoirement selon la fonction

$$Weight \sim \mathcal{U} \left[-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}} \right],$$

avec n le nombre d'entrées du neurone.

- "Xavier normale" initialise les poids aléatoirement selon la fonction

$$Weight \sim \mathcal{N}\left(0, \frac{1}{\sqrt{n}}\right),$$

avec n le nombre d'entrées du neurone. Avec l'exemple de la fonction d'activation tangente, une condition sur les difficultés de *vanishing gradient* ou d'*exploding gradient* peut être obtenue

$$n_l Var(W_l) = \begin{cases} < 1 \Rightarrow \text{Vanishing Gradient}; \\ = 1 \Rightarrow \text{OK}; \\ > 1 \Rightarrow \text{Exploding Gradient}. \end{cases}$$

Le critère de variance de l'initialisation de "Xavier normal" est ainsi retrouvé;

- "Glorot Uniforme" : Provenant de GLOROT et BENGIO, 2010, l'initialisation des poids est effectuée par une distribution uniforme

$$Weight \sim \mathcal{U}[-limit, limit],$$

avec : $limit = \sqrt{\frac{6}{(n+o)}}$ n étant le nombre d'entrée dans le neurone et o le nombre de valeurs ressorties ;

- "Glorot normale" : Provenant de GLOROT et BENGIO, 2010, l'initialisation des poids est effectuée par une distribution uniforme

$$Weight \sim \mathcal{N}(0, limit),$$

avec : $limit = \sqrt{\frac{2}{(n+o)}}$ n étant le nombre d'entrées dans le neurone et o le nombre de valeurs ressorties.

- La fonction d'activation ReLU utilise la fonction d'initialisation des poids nommée "he normale". Formellement,

$$Weight \sim \mathcal{N}\left(0, \frac{\sqrt{2}}{\sqrt{n}}\right),$$

avec n le nombre d'entrées du neurone. La justification étant plus complexe, le lecteur peut s'informer sur HE et al., 2015.

L'initialisation d'un réseau de neurones est une étape cruciale de son apprentissage à ne pas négliger. En pratique, de nombreux *packages* tels que *Keras* dans le logiciel R) intègrent déjà par défaut des fonctions d'activations complexes comme celle de "glorot et Xavier" dans leur algorithme. L'utilisateur gardant la possibilité de les modifier.

Maintenant que nous avons passé en revue les *inputs*, la fonction d'activation et l'initialisation des poids d'un réseau de neurones, décrivons son mécanisme d'apprentissage en détail.

2.1.4 L'apprentissage d'un réseau de neurones

Un réseau de neurones, ou le perceptron dans notre exemple, fonctionne en 2 phases. La phase d'entraînement et la phase de prédiction. L'apprentissage du réseau s'effectue durant la phase d'entraînement. Entraîner un réseau de neurones supervisé signifie modifier les poids de celui-ci afin que l'*output* du réseau soit le plus proche de l'*output* théorique au sens de la minimisation d'une fonction de perte. De par ce processus, nous souhaitons que le réseau de neurones capte le caractère général d'un *input* pour effectuer une prédiction.

En pratique pour l'apprentissage d'un algorithme de machine learning supervisé, une séparation de la base de données en 3 sous-bases de données est effectuée :

- La base d'apprentissage (ou *train dataset*). Elle représente entre 60 et 80 % de notre base initiale. C'est sur cette base que nous allons entraîner les poids du modèle.
- La base de validation (ou *validation dataset*). Elle représente environ 10 % de notre base initiale. Cet ensemble d'échantillons est utilisé uniquement pour évaluer la performance intermédiaire du réseau de neurones et permettre de sélectionner les meilleurs hyper paramètres. Par exemple, lors de l'entraînement d'un réseau de neurones dense, il faut déterminer le nombre de couches cachées optimal. Le réseau est entraîné avec 1 ou 3 couches cachées. La comparaison des résultats de chaque réseau s'effectuera sur la base de validation.
- La base de test (ou *test dataset*). Elle représente environ 10 % de notre base initiale. Cet ensemble d'échantillons est utilisé uniquement pour évaluer la performance finale du réseau de neurones. Il ne doit pas y avoir de modifications du modèle dépendant de la performance sur cette base.

La distinction entre la base test et la base de validation n'est pas toujours effectuée en fonction du nombre de données disponibles dans la base initiale. La sélection des données allant dans chaque base peut s'effectuer aléatoirement, mais cela peut entraîner de mauvaises prédictions d'une classe sous-représentée. L'idée la plus courante est de tirer aléatoirement sans remise en gardant les proportions initiales de diverses classes d'inputs préétablies. Les classes retenues sont généralement celles qui vont poser des problèmes par la suite. Un exemple classique en provisionnement est de garder la proportion de sinistres graves et attritionnels dans chaque base. Pour aller plus loin, des méthodes de *clustering* non supervisées pourraient être utilisées.

À l'aide de la base d'apprentissage, la phase d'entraînement se déroule en 2 étapes : une phase *Forward* dans laquelle un ou plusieurs *inputs* sont passés dans le modèle et une phase *Backward-propagation* dans laquelle le résultat précédent est utilisé pour calculer les poids. La phase *Forward* est similaire à la prédiction d'un output vu précédemment 2.1 et 2.2.

L'algorithme 1 ci-dessous, correspond à l'algorithme de *gradient descente* appliqué au cas d'un réseau de neurones. Cet algorithme a pour but d'actualiser les poids pour trouver le minimum d'une fonction de coût dépendant de ces mêmes poids. La fonction de coût est aussi nommée fonction de perte ou *loss* et sera notée J . Dans une régression, elle peut se calculer par la formule de l'erreur quadratique moyenne (*Mean Square Error*) et notre but est de la minimiser. Formellement,

$$\operatorname{argmin}_W J(\hat{y}, y) = \operatorname{argmin}_W \frac{1}{T} \sum_{n=1}^T (\hat{y}_n - y_n)^2, \quad (2.10)$$

Avec

- \hat{y} : le vecteur des prédictions par le réseau de neurones de chaque *input* de la base de données.
- y : le vecteur des observations des montants de provisions de chaque *input* de la base de données.
- T : la taille d'un *batch*. $T \in [1, N]$ le nombre de lignes de notre base d'entraînement. Un *batch* est un lot d'inputs provenant de notre base d'entraînement. L'algorithme de descente de gradient est toujours réalisé *batch* par *batch*. Un *batch* est donc une sous-base d'entraînement sur laquelle la fonction de *loss* est calculée permettant une actualisation de poids.

Avec

- i : le nombre d'actualisations de chaque poids.
- $(W)_i$: l'ensemble des poids du modèle actualisé 0 fois.
- $(X)_n$: les variables de la ligne n de notre base d'apprentissage.
- \hat{y}_n : l'output du réseau de neurones lorsque la ligne n est passée en *input*.
- α : le *learning rate* ou taux d'apprentissage. Il modère la vitesse d'apprentissage des poids. ($\alpha \in [0, 1]$)

Algorithm 1 Algorithme de la descente du gradient d'un RN

Require: Initialisation des poids du modèle $(W)_0$ (donc 0 fois actualisés)Soit une base de données à N lignes et une ligne input du RN dénommée $(X)_n$ avec $n \in [1; N]$ $i \leftarrow 0$ $epoch = 1$ **repeat**Découpage de la [Base d'entraînement] en $(\lfloor \frac{N}{T} \rfloor + 1)$ *Batch***for each** batch in [Base d'entraînement] **do**1) **Phase Forward****for** $(X)_n \in batch$ **do** $\hat{y}_n \leftarrow \text{Perceptron}((X)_n, (W)_i)$ **end for** $J \leftarrow J((\hat{y}), (y))$ avec (\hat{y}) le vecteur des \hat{y}_n et (y) le vecteur des y sorties théoriques associées aux $(X)_n$ 2) **Phase Backward** $(W)_{i+1} \leftarrow (W)_i + \alpha \times \nabla J$ $i \leftarrow i + 1$ **end for** $epoch = epoch + 1$ **until** Convergence

- ∇J : le gradient de la loss J par rapport au vecteur de poids $(W)_i$
- *batch* : petit lot d'inputs de taille T . Par exemple, si la base d'entraînement comporte $N = 100$ lignes et que la taille d'un *batch* est $T = 30$ alors, la séparation la base aléatoirement entre 3 *batches* de 30 observations et 1 *batch* de 10 observations. Cette notion est fortement liée à la notion d'*epoch*.
- Une *epoch* est une unité de mesure, elle symbolise une utilisation unique de chaque ligne de l'ensemble de la base d'apprentissage. Les *batches* étant une segmentation de la base d'entraînement, l'utilisation de l'ensemble des *batches* correspond à une seule utilisation de chaque ligne de la base d'entraînement et donc d'une *epoch*.

L'application de l'algorithme de descente de gradient sur la base d'apprentissage peut se présenter sous différentes formes. Nous allons ici en présenter 3 :

- *Batch Gradient Descent (BGD)* : cette méthode est l'application de l'algorithme précédent avec la taille de *batch* fixée à $T = N$ la taille de la base d'apprentissage. À chaque étape, l'ensemble de la base est utilisée pour calculer la fonction de loss et actualiser les poids (une *epoch* correspond ici à une actualisation des poids). Le temps d'entraînement peut être prolongé pour de grandes quantités de données. Néanmoins, cette méthode est stable, car nous ne modifions pas les paramètres de notre modèle aussi fréquemment que d'autres variantes de descente de gradient. Comme nous avons besoin de l'ensemble des données à chaque actualisation des poids, la *BGD* n'est pas efficace sur le critère de la mémoire utilisée. Un autre problème à cette méthode est l'obtention de minimums locaux. En effet, passer toujours les mêmes données pour chaque actualisation peut conclure à un minimum local dans la descente de gradient.
- *Stochastic Gradient Descent (SGD)* : cette méthode est l'application de l'algorithme précédent avec la taille de *batch* fixée à $T = 1$, soit une seule ligne de la base d'apprentissage. À chaque étape, une ligne de la base est utilisée pour calculer la fonction de loss et actualiser les poids (une *epoch* correspond ici à N actualisations des poids). L'algorithme est dit stochastique, car l'ordre dans lequel est prise chaque ligne est aléatoire. Grâce à l'actualisation des poids à chaque

observation de la base d'apprentissage, le SGD peut éviter un minimum local, mais a peu de chance d'obtenir une solution stable au final. La méthode se fait donc battre après un grand nombre d'itérations par le *BGD*. La méthode *SGD* a un apprentissage beaucoup plus rapide, l'actualisation des poids a lieu N fois par *epoch* comparé à 1 fois avec le *BGD*. La sélection aléatoire de l'ordre des inputs des *batch* induit un bruit, favorisé par la volatilité de chaque input.

- *Mini-Batch Gradient Descent (MBGD)* : cette méthode a pour but de combiner les points forts de la *SGD* et de la *BGD*. C'est un mélange des deux méthodes. L'algorithme est appliqué avec $1 < T < N$ pour la taille de chaque *batch*. Plus la taille de chaque *batch* est proche de 1 plus l'algorithme possède les qualités du *SGD* (rapide et esquive le problème des minimums locaux, mais instable), plus elle est proche de N plus l'algorithme possède les qualités du *BGD* (lent et stable, mais a un problème des minimums locaux). Un hyper paramètre du modèle important du modèle est donc la taille du *batch* T .

Commençons par expliciter le terme du gradient de la loss pour le perceptron et pour un réseau de neurones afin d'améliorer notre compréhension de l'actualisation de nos poids.

Le gradient d'un perceptron

Le gradient de la loss d'un perceptron est défini pour chacun des poids par

$$\nabla J = -\frac{\delta J}{\delta W} = \begin{bmatrix} -\frac{\delta J}{\delta W^0} \\ \dots \\ -\frac{\delta J}{\delta W^K} \end{bmatrix}.$$

En utilisant le théorème de dérivation des fonctions composées, le gradient se décompose comme le produit de dérivées partielles

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial \hat{y}_n} \frac{\partial \hat{y}_n}{\partial z_n} \frac{\partial z_n}{\partial W}.$$

Pour un unique *input* (*SGD*), le premier terme est obtenu par dérivation de l'erreur quadratique moyenne

$$\frac{\partial J}{\partial \hat{y}_n} = \frac{\partial (y_n - \hat{y}_n)^2}{\partial \hat{y}_n} = -2(y_n - \hat{y}_n) = 2(\hat{y}_n - y_n).$$

Le second terme dépend de la fonction d'activation

$$\frac{\partial \hat{y}_n}{\partial z_n} = \frac{\partial F(z_n)}{\partial z_n} = \begin{cases} 1 & \text{si F est la fonction identité;} \\ \frac{\exp(-z_n)}{(1+\exp(-z_n))^2} & \text{si F est la fonction sigmoid;} \\ 1 - \tanh^2(x) & \text{si F est la fonction tanh;} \\ F'(z_n) & \text{pour toute fonction d'activation.} \end{cases} \quad (2.11)$$

Avec $z_n = \sum_{k=1}^K X_n^k W_k$ dans le cas du perceptron. La dérivée de la fonction d'activation doit être rapide d'évaluation, car elle sera évaluée un grand nombre de fois dans la descente de gradient.

Le troisième terme est la simple dérivée de la somme pondérée des *inputs*

$$\frac{\partial z_n}{\partial W} = \frac{\partial X_n \cdot W}{\partial W} = X_n.$$

Enfin pour un perceptron de fonction d'activation identité, la formule globale (2.12) permet d'actualiser les poids avec la méthode (*SGD*)

$$-\frac{\delta J}{\delta W_k} = -2X_n^k(\hat{y}_n - y_n). \tag{2.12}$$

Pour les méthodes (*BGD ou MBGD*) utilisant la moyenne de T *inputs*, une moyenne des gradients (2.13) est obtenue pour chacun des T *inputs*.

$$-\frac{\delta J}{\delta W_k} = \frac{-2}{T} \sum_{n=1}^T X_n^k(\hat{y}_n - y_n). \tag{2.13}$$

Le gradient d'un réseau de neurones multicouches

Le gradient de la fonction de *loss* d'un réseau de neurones multicouches fonctionne selon les mêmes concepts que celui d'un perceptron. Le changement majeur provient du fait qu'une séquence de neurones ajoute des termes dans la dérivée composée. L'idée du calcul est de sommer les gradients de chaque chemin partant du poids et allant jusqu'à l'*output* donc appartenant à la fonction de *loss*. Un exemple simplifié (2.13) d'un réseau multicouche à 2 couches cachées et 2 neurones par couches cachées montre effectivement cette notion.

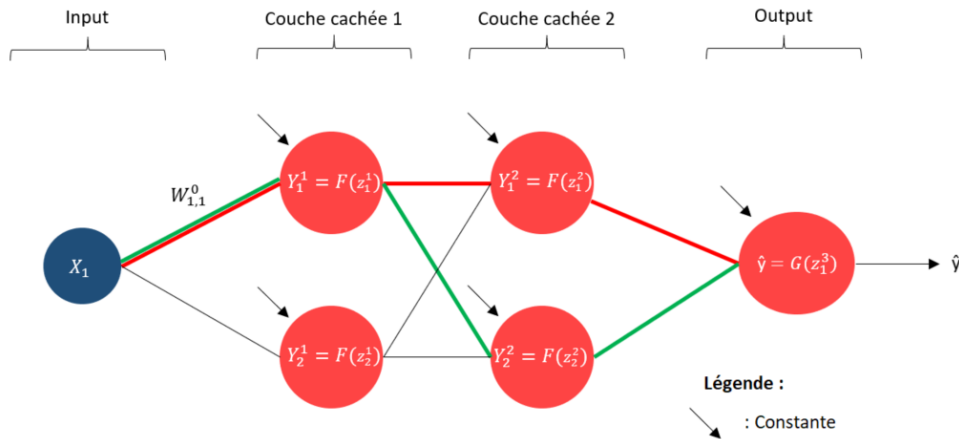


FIGURE 2.13 : Schéma d'un réseau de neurones à 2 couches cachées

Pour calculer le gradient de la *loss* par rapport au poids $W_{1,1}^0$ (2.14), il faut donc sommer les gradients des deux chemins possibles (en vert et en rouge dans le schéma ci-dessous). Formellement, la formule devient

$$\frac{\delta J}{\delta W_{1,1}^0} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_1^3} \frac{\partial z_1^3}{\partial Y_1^2} \frac{\partial Y_1^2}{\partial z_1^2} \frac{\partial z_1^2}{\partial Y_1^1} \frac{\partial Y_1^1}{\partial z_1^1} \frac{\partial z_1^1}{\partial W_{1,1}^0} + \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_1^3} \frac{\partial z_1^3}{\partial Y_2^2} \frac{\partial Y_2^2}{\partial z_1^2} \frac{\partial z_1^2}{\partial Y_1^1} \frac{\partial Y_1^1}{\partial z_1^1} \frac{\partial z_1^1}{\partial W_{1,1}^0}. \tag{2.14}$$

Pour plus de détails, le lecteur peut se référer au lien suivant :

L'étude de chaque élément d'un réseau de neurones basique se conclut ici. Nous avons effectué une description du fonctionnement, de l'initialisation, de l'apprentissage et de la prédiction pour un perceptron et un réseau de neurones multicouches dense. Ces concepts sont d'une grande importance, car ils sont la base de la théorie des réseaux de neurones. Le but de ce mémoire est d'appliquer un réseau de neurones à un problème de provisionnement. L'objectif n'est donc pas la prédiction d'une valeur unique comme proposée par les réseaux de neurones présentés précédemment, mais une prédiction de plusieurs pas de temps. Une méthode pour résoudre cette difficulté est d'utiliser un réseau de neurones récurrents.

2.2 Les réseaux de neurones récurrents

L'objectif du provisionnement est de prédire les paiements que l'assureur devra effectuer annuellement. Dans ce but, l'assureur dispose de l'historique des paiements antérieurs pour chaque sinistre. Des données temporelles sont donc présentes dans la base de données. Il serait moins efficace d'estimer les paiements futurs avec uniquement le dernier paiement connu alors que nous avons des montants de paiements sur plusieurs années.

Les réseaux de neurones comme le perceptron et le réseau multicouches sont conçus pour des points de données indépendants les uns des autres. Cependant, avec des données se présentant sous la forme de séquences temporelles, une modification du réseau de neurones pour intégrer les dépendances entre ces points de données est nécessaire.

Un réseau de neurones récurrents ou *Recurrent neural network (RNN)* prend comme input une variable temporelle et les prédictions des pas de temps passés afin d'obtenir les prédictions des pas de temps futurs. Il existe différentes architectures de réseaux de neurones récurrents (voir figure 2.14).

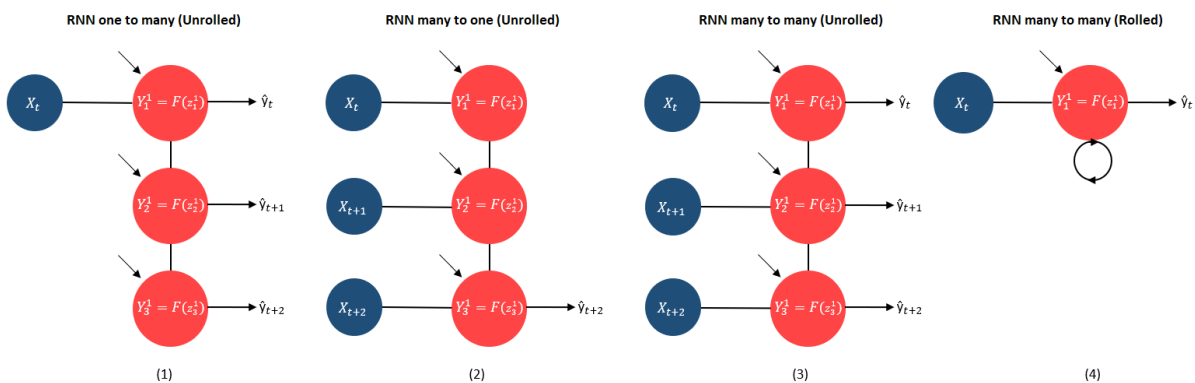


FIGURE 2.14 : Schéma de 3 architectures de *RNN* avec une représentation condensée

Un réseau de neurones récurrents est composé de plusieurs cellules liées séquentiellement. Une cellule est une structure de neurones avec un *input* et un *output*. Dans le cas d'un réseau de neurones récurrents, une cellule a pour *input* un pas de temps d'une variable temporelle et la prédiction de la cellule précédente pour le pas de temps. Son *output* servira également d'input à la prochaine cellule.

Sur le schéma, chaque cellule est un neurone classique comme vu précédemment, mais ce n'est pas forcément le cas.

Dans le cas (1), l'*input* est une information unique, la prédication est composée de plusieurs temporalités. C'est l'exemple d'un sinistre de l'année en cours. Avec une seule année de paiement connue, le réseau de neurones prédit les paiements année après année (sur le schéma les 3 années suivantes).

Dans le cas (2), l'*input* est composé de plusieurs temporalités et l'*output* est unique. Cela correspond à la prédiction du prochain paiement d'un sinistre avec les 3 derniers paiements connus.

Dans le cas (3), nous avons plusieurs *inputs* et *outputs*. Ce dernier cas plus général est la combinaison des 2 premiers et permet de prédire les 3 prochaines années de paiements en utilisant 3 années de paiements connus. La version enroulée (4) est une représentation plus compacte du cas (3) souvent observée dans la littérature.

L'avantage des réseaux de neurones récurrents réside dans la capacité du réseau à conserver l'information. Cependant ce type de structure comporte 2 défauts importants : le temps d'exécution peut être assez long, car chacun des poids à actualiser est relié et possède de nombreux chemins vers la sortie surtout en cas de couches cachées rajoutées et le problème de la disparition du gradient. En effet, un RNN standard ne parvient pas à apprendre en présence de décalages temporels supérieurs à 10 pas de temps entre les événements d'*inputs* pertinents et les *outputs* voulus.

Afin de pallier ce problème, différentes architectures de réseaux de neurones récurrents sont apparues. Parmi les 2 architectures les plus connues, le *Long short-term memory (LSTM)* a été choisi, car présent dans le modèle et plus précisément nous expliquerons le fonctionnement d'une structure composée de *LSTM* : le *Recurrent neural network encoder-decoder*.

2.2.1 LSTM

Un *Long short-term memory (LSTM)* est un réseau de neurones récurrents suivant le schéma précédent sauf que chaque cellule n'est pas un neurone classique, mais une cellule ayant une structure plus complexe. Cette cellule est composée de trois inputs, deux sorties et quatre neurones appelés « portes ». Les portes sont des neurones qui régulent le flot d'informations. Une cellule se présente sous la forme suivante : 2.15.

Une cellule *LSTM* composée de plusieurs éléments :

- X_t : l'ensemble des variables connues au pas temporel t .
- h_t : la prédiction de la cellule au pas temporel t . Elle est obtenue par la formule

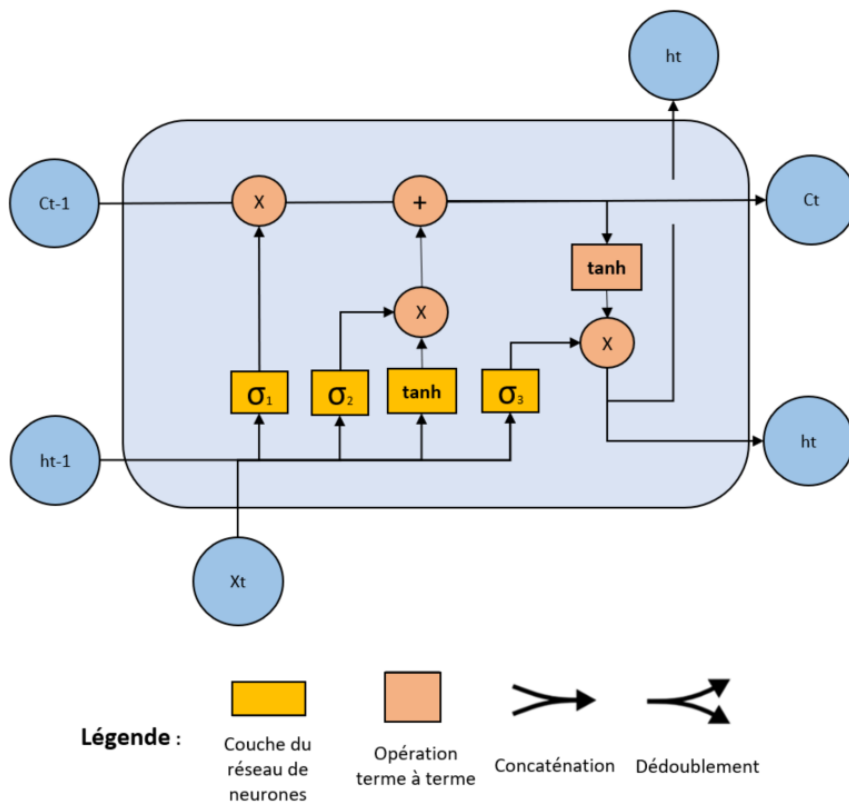
$$h_t = \tanh(C_t) \times \sigma_3.$$

- h_{t-1} : la prédiction de la cellule au pas temporel $t - 1$, initialisé à 0 pour h_0 (mémoire à court terme).
- C_t : État de la cellule au pas temporel t , obtenue par la formule :

$$C_t = C_{t-1} \times \sigma_1 + \tanh_1 \times \sigma_2.$$

- C_{t-1} : État de la cellule au pas temporel $t - 1$ initialisé à 0 pour C_0 (mémoire à long terme).
- σ_1 : la porte d'oubli (*forget gate*). Cette porte contrôle quelle information doit être oubliée.

$$\sigma_1 = \text{Sigmoid}(\text{Concatenation}(X_t; h_{t-1}) \times W_a).$$

FIGURE 2.15 : Schéma d'une cellule *LSTM*

Comme la fonction sigmoïde est comprise entre 0 et 1, elle détermine quelle valeur de l'état de la cellule doit être rejetée (multipliée par 0), mémorisée (multipliée par 1) ou partiellement mémorisée (multipliée par une valeur comprise entre 0 et 1).

- σ_2 : la porte d'entrée (*input gate*). Cette porte aide à identifier les éléments importants qui doivent être ajoutés à l'état de la cellule.

$$\sigma_2 = \text{Sigmoïde}(\text{Concatenation}(X_t; h_{t-1}) \times W_b).$$

Les résultats de la porte d'entrée sont multipliés par le candidat à l'état de cellule, donc seules les informations jugées importantes par la porte d'entrée sont ajoutées à l'état de cellule.

- \tanh_1 : la nouvelle information qui doit être transmise à l'état de la cellule donnée par la formule suivante

$$\tanh = \tanh(\text{Concatenation}(X_t; h_{t-1}) \times W_c).$$

est une fonction de l'état caché à l'instant t-1 précédent et de l'entrée x à l'instant t. La fonction d'activation est ici tanh. Grâce à la fonction tanh, la valeur de la nouvelle information se situe entre -1 et 1. Si la valeur de N_t est négative, l'information est soustraite de l'état de la cellule et si la valeur est positive, l'information est ajoutée à l'état de la cellule à l'instant présent.

- σ_3 : la porte de sortie (*output gate*). Son objectif est de participer à la prédiction de h_t . Elle est obtenue par la formule

$$\sigma_3 = \text{Sigmoïde}(\text{Concatenation}(X_t; h_{t-1}) \times W_d).$$

L'idée sous-jacente à la cellule *LSTM* est de conserver l'information importante au fil des pas de temps en ajoutant ou retirant de l'information à l'aide des différentes portes. L'apprentissage et la prédiction sont similaires à un réseau de neurones multicouches. La fonction de *loss* est dérivée par rapport à l'ensemble des chemins utilisant le poids.

Il est important de comprendre que les poids d'une cellule à un pas de temps donné sont les mêmes que ceux à un autre pas de temps. Cela augmente le nombre de chemins utilisant le poids, mais diminue le nombre de poids global. L'*overfitting* en est réduit, mais l'apprentissage reste long lorsqu'il y a de nombreux pas de temps.

Un *LSTM* peut aussi être composé de couches cachées. En effet, la cellule d'un *LSTM* est composée de 4 neurones précédemment explicités, mais chacun de ces neurones ne permet de prédire qu'une seule information. Si le but est d'obtenir un vecteur en *output*, comme pour les réseaux denses, il faut démultiplier chaque neurone. Un *LSTM 3* est donc un *LSTM* dans lequel chaque neurone est remplacé par 3 neurones en parallèle. Par exemple, la porte d'oubli σ_1 deviendra $\sigma_{1.1.3}$.

Les réseaux de neurones récurrents ont pour but principal de garder la mémoire. Ils peuvent cependant être aussi associés au processus de sélection de l'information principale afin d'extraire une généralisation permettant une meilleure prédiction. C'est le cas de la structure d'*encoding-decoding*.

2.2.2 La méthode *encoder-decoder* ?

La méthode *encoder-decoder* ou *Seq2seq LSTMs* consiste à utiliser un *LSTM many to one* pour lire la séquence d'entrée, un pas de temps à la fois, afin d'obtenir une grande représentation vectorielle à dimension fixe, puis d'utiliser un autre *LSTM one to many* pour extraire la séquence de sortie de ce vecteur. Ce mix des 2 permet notamment de réaliser un *LSTM many to many* où l'entrée est connue sur les premières années temporelles et où l'objectif est de prédire la sortie sur les autres années.

La capacité du *LSTM* à apprendre avec succès sur des données présentant des dépendances temporelles à long terme en fait un choix naturel pour cette application en raison du décalage temporel considérable entre les entrées et leurs sorties correspondantes.

L'application d'un réseau de neurones récurrents permet d'utiliser des données temporelles et de prédire des données temporelles. Les éléments explicités précédemment permettent de créer un réseau de neurones, de l'initialiser, de l'entraîner, mais ce mémoire consiste en l'obtention d'une distribution. Dans ce but, nous allons voir 2 méthodes permettant d'ajouter de l'aléa dans le modèle.

2.3 La gestion de l'aléa

L'aléa au sein du modèle de machine learning peut se diviser en 2 catégories : l'aléa induit par les données et l'aléa induit par l'apprentissage du modèle.

2.3.1 L'aléa des données

L'objectif d'un modèle est de comprendre une généralisation du lien entre l'input et l'output. En effet, en utilisant trop de poids et peu d'*inputs* dans notre base d'apprentissage, le modèle obtenu va trop apprendre sur chaque donnée y compris le bruit des données (voir figure 2.16). C'est la définition de l'*overfitting*. L'*overfitting* est une erreur qui se produit dans la modélisation des données, du fait qu'un modèle s'approche trop étroitement de chaque donnée individuellement généralement par manque de données.

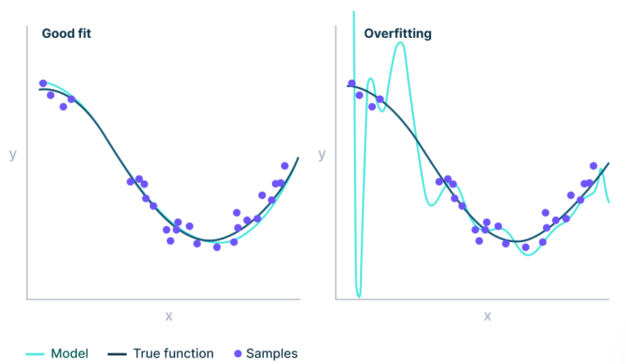


FIGURE 2.16 : Exemple d'*Overfitting*

Cependant l'*overfitting* nous indique qu'un modèle est capable d'apprendre plus d'informations qu'une moyenne simple. Pour avoir un apprentissage de la variance de nos données, une idée consiste à ajouter une hypothèse à notre modèle : l'hypothèse d'une distribution de sortie.

En supposant une loi de sortie pour le bruit induit dans les données, nous pouvons désormais attribuer une loi comme *output* de notre modèle. Observons sur un schéma comment faire dans le cas d'une loi normale 2.17.

Sur ce schéma, avant l'utilisation d'une loi en *output* du réseau de neurones, le modèle prédit seulement la valeur \hat{y}_1 , une moyenne générale. Appliquer la distribution au bruit de nos données ne change pas le résultat de la moyenne, \hat{y}_1 reste le même, cependant le modèle apprend un autre critère : la variance autour de cette moyenne pour chaque donnée. C'est la représentation du bruit. En

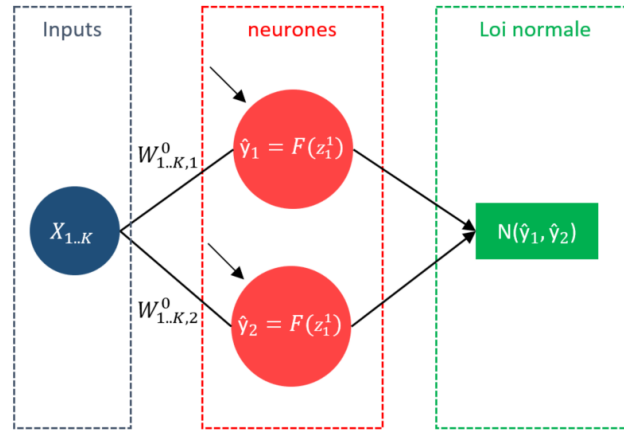


FIGURE 2.17 : Exemple d'un réseau de neurones avec une distribution comme output

pratique, pour appliquer une distribution à un réseau de neurones, il faut autant de sorties (neurones en parallèle) que de paramètres nécessaires pour la loi. Ainsi, chaque *input* va passer dans le réseau et prédire les paramètres de sa loi. Sur le schéma, il n'y a pas de poids entre les neurones et le bloc de loi normale. En effet, une distribution de sortie n'est pas une étape d'apprentissage, elle ne possède pas de poids. L'entièreté de l'apprentissage est réalisée en amont dans les neurones.

Le fonctionnement d'un bloc de distribution se différencie sur 2 points clés du modèle : la prédiction du modèle et la fonction de *loss*.

Lorsqu'une ligne de la base de données est passée en *input*, le modèle retourne une loi. Suivant l'objectif il peut être intéressant d'utiliser la loi pour obtenir une moyenne, sa variance, ou seulement des échantillons.

Dans la partie précédente, la fonction de *loss* était l'erreur quadratique moyenne. Le modèle retournant à présent une distribution, il manque le terme \hat{y} . $\hat{y} = \mu$ (la moyenne de la distribution) ne fonctionne pas, car le modèle n'apprend pas le terme de variance si ce terme n'a pas d'impact sur la fonction de *loss*. Une idée serait de prendre un échantillon de la loi, mais cela donnerait un terme de *loss* très volatile. La meilleure solution est de changer la formule de la *loss* entre 2 valeurs par la formule de la *loss* entre 2 distributions.

La divergence de Kullback-Leibler

La divergence de Kullback-Leibler (appelée *KL divergence*) est une mesure de la différence entre deux distributions de probabilité. Classiquement, dans la théorie bayésienne, il existe une certaine distribution vraie $P(X)$ et le but est de l'estimer par une distribution approximative $Q(X)$. Dans ce contexte, la divergence de KL mesure la distance de la distribution approximative Q à la distribution réelle P .

Mathématiquement, considérons deux distributions de probabilité P et Q définies sur le même espace de probabilité \mathcal{X} alors la *KL divergence* de Q à P s'écrit

$$D_{KL}(P\|Q) = \mathbb{E}_{x \sim P} \left[\log \frac{P(X)}{Q(X)} \right] = \begin{cases} \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right) & \text{pour des distributions discrètes;} \\ \int_{\mathcal{X}} \log \left(\frac{P(dx)}{Q(dx)} \right) P(dx) & \text{pour des distributions continues.} \end{cases}$$

La *KL divergence* peut se réécrire en fonction de la *cross entropy* entre P et Q , notée $\mathcal{H}(p, q)$ et $\mathcal{H}(P(X))$ l'*entropy* de P ,

$$D_{KL}(P\|Q) = \mathbb{E}_{x \sim P}[-\log Q(X)] - \mathcal{H}(P(X)) = \mathcal{H}(p, q) - \mathbb{E}_{x \sim p}[-\log p(x)].$$

Elle possède les propriétés suivantes :

- La divergence KL n'est pas symétrique : c'est-à-dire que

$$D_{KL}(P\|Q) \neq D_{KL}(Q\|P).$$

Par conséquent, il ne s'agit pas non plus d'une métrique de distance.

- La divergence KL peut prendre des valeurs dans $[0, \infty]$. En particulier, $(P \stackrel{\mathcal{D}}{=} Q) \Leftrightarrow D_{KL}(P\|Q) = 0$,

Afin de calculer la *loss* d'un réseau de neurones supervisé, 2 distributions sont disponibles : la distribution des observations réelles (P) et la distribution obtenue par le modèle dépendant des poids du modèle (Q_W). La *KL divergence* n'étant pas symétrique, lors de l'approximation de la distribution réelle P , il faut choisir entre deux fonctions objectifs potentielles à optimiser :

- La divergence de KL. Formellement, le problème d'optimisation devient

$$\operatorname{argmin}_W D_{KL}(P\|Q_W).$$

Cela revient à échantillonner des points de P et essayer de maximiser la probabilité de ces points sous Q_W . Une bonne approximation signifie que partout où P a une probabilité élevée, Q_W doit également avoir une probabilité élevée ce qui entraîne une recherche de la moyenne. Afin de pouvoir évaluer cette fonction objectif, nous avons besoin soit d'un ensemble d'échantillons du vrai modèle P soit d'un mécanisme d'échantillonnage du vrai modèle. C'est le cas pour un réseau de neurones par apprentissage supervisé.

- La divergence inverse de KL. Formellement le problème d'optimisation devient

$$\operatorname{argmin}_W D_{KL}(Q_W\|P).$$

Cela revient à échantillonner des points de Q_W et essayer de maximiser la probabilité de ces points sous P . Une bonne approximation signifie que partout où Q_W a une probabilité élevée, P doit également avoir une probabilité élevée. Rien n'oblige donc la distribution estimée à essayer de couvrir tous les "pics" de la distribution réelle. Cependant le "pic" de la distribution estimée sera mieux estimé qu'avec la divergence de KL. Afin de pouvoir évaluer cette fonction de *loss*, nous devons être en mesure d'évaluer les probabilités des points des données sous le vrai modèle P .

Malgré ses inconvénients, l'apprentissage supervisé indique fortement l'utilisation de la *KL divergence*. De plus, minimiser la *KL divergence* revient à maximiser la log-vraisemblance. La démonstration est disponible à PATACCHIOLA, 2021. La fonction de *loss* devient donc

$$loss = \sum_{i=1}^T \sum_{t=1}^J \log \widehat{f_{Y_{i_t}}}(Y_{i_t}).$$

En fixant la distribution du bruit de l'input, l'utilisation du maximum de vraisemblance comme fonction de *loss* permet d'améliorer nos poids et ainsi capter la moyenne et le bruit dû aux données. Il existe plusieurs distributions possibles.

Les lois de probabilités

Pratiquement, la loi sera choisie dans la liste suivante, chaque loi impliquant un certain nombre de paramètres de sorties du modèle :

- Une loi normale. C'est une loi basique qui permettra de faire un témoin pour voir l'amélioration des autres lois.

$$\mathcal{N}(\mu, \sigma^2).$$

Avec cette loi, la sortie du modèle est composée 2 valeurs numériques faciles d'interprétation : μ et σ ;

- Un mélange d'une loi *shift* log-normale et d'une loi déterministe valant 0.

$$P_1 \times (\log \mathcal{N}(\mu, \sigma^2) + \text{shift}) + P_2 \times 0. \quad (2.15)$$

Avec cette loi, la sortie du modèle est composée 4 valeurs numériques X_1 à X_4 :

- $X_1 = \mu$ et $X_2 = \sigma$: la moyenne et variance de la loi normale sous-jacente à la loi log-normale. La moyenne et la variance de la loi log-normale peuvent se retrouver à l'aide de ces paramètres par

$$\begin{aligned} \mathbb{E}(\log \mathcal{N}) &= \exp\left(\mu + \frac{\sigma^2}{2}\right); \\ \mathbb{V}(\log \mathcal{N}) &= (\exp(\sigma^2) - 1) \exp(2\mu + \sigma^2). \end{aligned} \quad (2.16)$$

- Les termes X_3 à X_4 correspondent aux probabilités d'avoir un sinistre ouvert ou fermé à ce pas de temps obtenu à l'aide de la fonction softmax. Formellement,

$$\begin{aligned} P_1 &= \frac{\exp^{X_3}}{\exp^{X_3} + \exp^{X_4}}; \\ P_2 &= \frac{\exp^{X_4}}{\exp^{X_3} + \exp^{X_4}}. \end{aligned} \quad (2.17)$$

Cette distribution est la distribution de KUO, 2020.

C'est une loi un peu plus complexe, le terme est décalé vers la gauche afin de prendre en compte les zéros de la base. Sans un décalage suffisamment important (de l'ordre de 2), les erreurs d'approximations de la fonction de perte en 0 vont créer un apprentissage faux qui va donner lieu à une fonction de loss négative ;

- Un mélange de lois gaussiennes.

$$f_{\hat{X}_{i,j}}(x) = \sum_{k=1}^K \alpha_{i,j,k} \phi(x | \mu_{i,j,k}, \sigma_{i,j,k}).$$

Afin de réduire l'impact de l'hypothèse de loi, une possibilité est d'utiliser un mélange de lois gaussiennes. Le nombre de lois mélangées est soit un hyper paramètre fixe, soit un paramètre appris du modèle borné. En effet, comme BISHOP, 1994 le note, un mélange de lois gaussiennes créé par un réseau de neurones, avec suffisamment de lois mélangées et de couches cachées, est capable d'approximer n'importe quelle distribution. Cependant, ce cas ne sera pas abordé en profondeur dans ce mémoire.

Cette liste n'est pas exhaustive, mais suffira pour l'implémentation numérique. L'aléa dû aux données est pris en compte, il faut ensuite prendre en compte l'aléa dû au modèle.

2.3.2 L'aléa du modèle : un bayésien

Lors de l'utilisation d'un réseau de neurones, même dans le cas linéaire où il est possible de créer un exemple dont la solution optimale est connue, obtenue cette solution optimale par l'apprentissage des

pois est très difficile. L'objectif d'un réseau de neurones est de se rapprocher de cette solution, c'est une approximation, car les poids ont un nombre de décimales fini. Les poids du réseau de neurones ne sont donc pas "parfaits" à la fin de l'apprentissage. Dans le but de compenser l'aléa dû à l'apprentissage des poids du modèle, une couche composée d'un *Bayesian neural network dense* peut être introduite (voir figure 2.18).

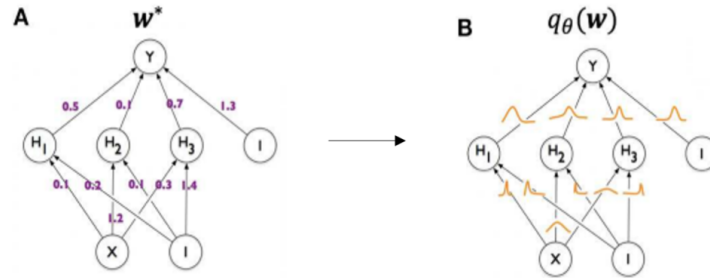


FIGURE 2.18 : Comparaison du schéma d'un réseau de neurones à 1 couche cachée (A) et d'un réseau de neurones bayésien à 1 couche cachée (B).

Un réseau de neurones bayésien (BNN) dense a une forme similaire au réseau de neurones multicouches dense vu précédemment. La différence réside dans les poids qui ne sont pas des valeurs fixées, mais des réalisations de distributions. Ainsi un BNN n'apprend pas des poids d'un réseau de neurones dense, mais apprend les paramètres d'une distribution choisie pour chaque poids.

Il fonctionne en minimisant la log-vraisemblance (souvent par l'utilisation de l'*Evidence Lower bound (ELBO)* une borne inférieure de la log-vraisemblance), s'efforçant ainsi de trouver une distribution des poids a posteriori qui doit :

- bien s'adapter aux données réelles (autrement dit, obtenir un maximum de log-vraisemblance élevé) ;
- rester proche de la distribution a priori des poids (grâce à la *KL divergence*).

Notons $P(W/D)$ avec $D = (x_i, y_i)$ la distribution théorique de poids sachant les données et les observations. Le modèle est initialisé avec :

- La distribution des poids a priori : $P(W)$, usuellement une loi normale ;
- La forme de la distribution estimée des poids a posteriori : $q(w/\theta)$ avec θ les paramètres à apprendre.

L'output du modèle est le vecteur de paramètre θ tel que la distribution estimée des poids a posteriori $q(w/\theta)$ approxime bien la distribution a posteriori des poids $P(W/D)$.

L'apprentissage des poids est obtenu par la divergence de Kullback-Leibler entre la distribution estimée des poids a posteriori et la distribution des poids a priori. Cela nous donne la fonction de *loss* générale du modèle

$$loss = \sum_{i=1}^T \sum_{t=1}^J \log f_{Y_{it}}(Y_{it}) + \frac{1}{|\mathcal{D}|} D_{KL}(q(w|\theta) || P(w)).$$

Une réalisation de la distribution des poids est réalisée à chaque actualisation de poids dans

l'algorithme de descente de gradient. Pour la phase de prédiction, une nouvelle réalisation des poids du modèle est effectuée à chaque changement de *batch* dans la base de test.

La prochaine partie consiste à établir l'ensemble des modèles composés des éléments expliqués dans les parties précédentes que nous étudierons.

2.4 Choix des modèles

Précédemment, nous avons expliqué le fonctionnement d'un réseau de neurones tel que le perceptron ou le réseau de neurones multicouches. Puis nous avons détaillé une méthode afin d'utiliser efficacement une information temporelle dans un réseau de neurones (avec un RNN). Enfin, nous avons explicité comment gérer l'aléa des données (avec une distribution) et l'aléa dû au modèle (avec un BNN). À l'aide des éléments expliqués précédemment, nous pouvons désormais les assembler et obtenir le modèle final de ce mémoire.

2.4.1 Le modèle de Kuo

Le modèle final (sur la figure 2.19) provient de l'article de KUO, 2020.

L'application du modèle passe par une séparation initiale des données en 4 groupes de variables :

- Les variables quantitatives par pas de temps : les *cash-flows* de charges ou de paiements ;
- La variable qualitative par pas de temps : le statut du sinistre (ouvert ou fermé) connu pour chaque année ;
- Les variables explicatives numériques : par exemple l'année de développement ;
- Les variables explicatives catégorielles : par exemple le moyen de distribution du produit d'assurance.

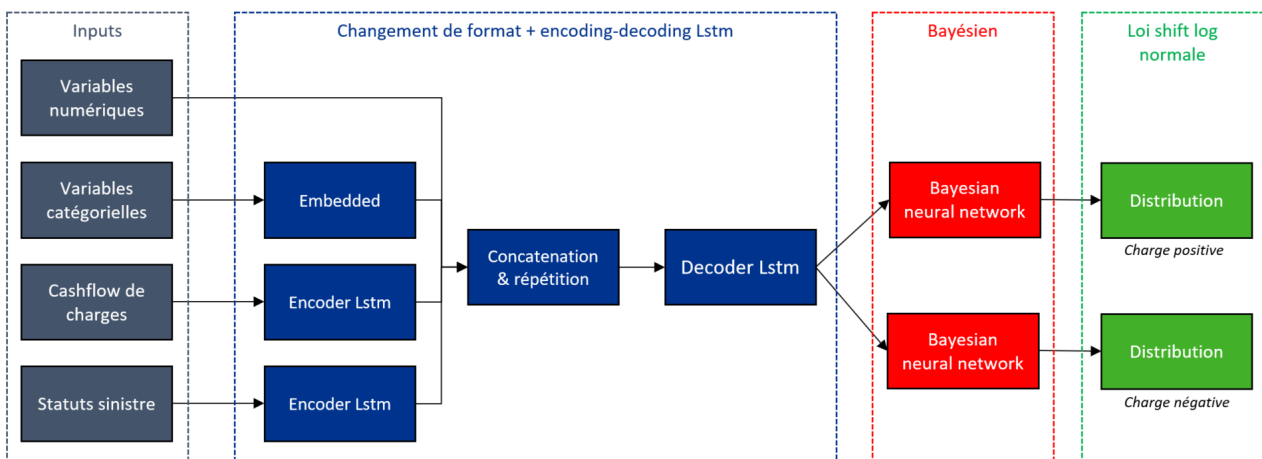


FIGURE 2.19 : Schéma du modèle de Kuo 2020

Le modèle de Kuo est composé d'une couche d'*embedding* pour retraiter les données catégorielles et d'une méthode *encoder-decoder* utilisant les *LSTM* afin de traiter l'information par pas de temps. L'*output* pour chaque pas de temps en sortie du décodeur est passé en *input* d'un réseau de neurones

bayésien permettant de capter l'aléa dû aux poids du modèle et les valeurs d'output du BNN sont les paramètres de la distribution.

Le modèle de Kuo va séparer l'*output* de son modèle entre les charges positives et les charges négatives. Cette opération permet de toujours prédire des montants supérieurs ou égaux à 0 et permet ainsi l'utilisation d'une loi log-normale. Cependant, cette hypothèse peu justifiée a pour défaut qu'elle induit la possibilité pour le modèle de prédire une charge positive et une charge négative non nulle ce qui est impossible en pratique.

Pour récapituler, ce modèle prend en entrée 1 sinistre avec son information connue sur les charges, son statut et d'autres variables explicatives invariantes dans le temps. En sortie, ce modèle prédit une distribution pour chaque montant de charge par pas de temps y compris pour des pas de temps déjà connus. En pratique, les distributions prédites des valeurs connues sont masquées afin de ne prendre en compte que les distributions prédites des valeurs non connues. Cela se retranscrit aussi dans la fonction de *loss* qui par le même procédé ne va pas calculer de maximum de vraisemblance ni de divergence de KL sur les pas de temps connus. Le modèle n'apprend donc pas à prédire la charge d'une année de développement à l'aide du montant de cette charge en *input*, ce qui est logique.

Afin de valider l'impact positif de chaque élément complexifiant le modèle, nous allons comparer le modèle de Kuo à d'autres modèles plus simples tels que le modèle minimal permettant de répondre à notre problème.

2.4.2 Le modèle invariant

Un modèle plus simple utilisant un réseau de neurones et sortant une distribution existe. Il permettra d'être le témoin de base pour l'autre modèle. Son schéma est le suivant (voir figure 2.20).

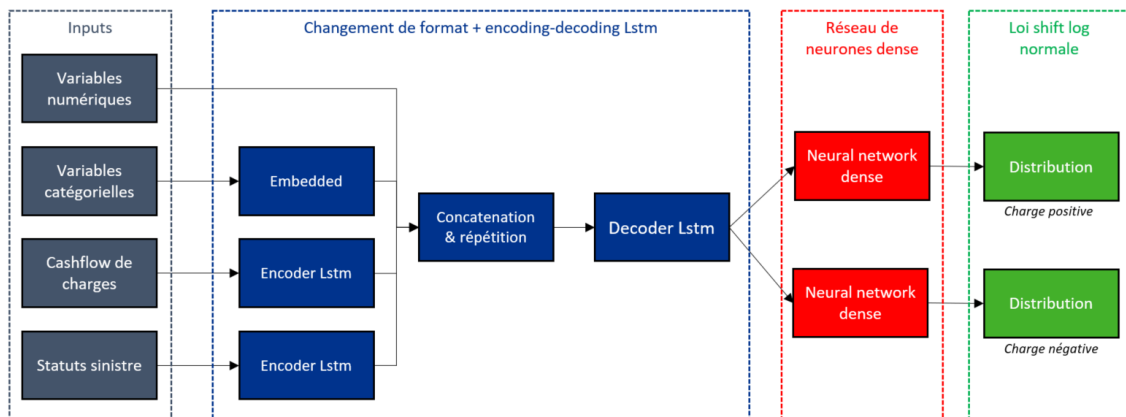


FIGURE 2.20 : Schéma d'un modèle simplifié

La séparation entre les charges positives et négatives a été conservée afin de faciliter la comparaison des distributions finales. Le modèle invariant et le modèle de Kuo pourront ainsi être comparés. Un *BNN* peut être considéré comme une complexification d'un réseau de neurones dense classique. En effet, le cas simple ou la distribution des poids n'a pas de variance revient à trouver directement le poids et donc est équivalent à un réseau dense. Ce second modèle permettra de calibrer les hyperparamètres et d'effectuer la sélection des variables explicatives les plus pertinentes sans se préoccuper du caractère aléatoire des poids grâce à son caractère déterministe une fois le modèle entraîné. La

calibration des paramètres du *BNN* sera effectuée dans un second temps.

Une fois les 2 modèles implémentés et entraînés sur notre base d'apprentissage, il va falloir prédire leurs résultats sur la base test.

2.5 Processus de prédiction des résultats par Monte-Carlo

Le but de cette partie est d'expliquer comment obtenir une distribution de la charge non connue d'un sinistre à partir d'un *input* (l'information d'un sinistre pour une année donnée) et d'un modèle entraîné. Dans un réseau de neurones classique, il suffit de passer l'input dans le modèle afin d'obtenir la valeur de l'output. Dans un réseau de neurones plus complexe, il faut prendre en compte qu'il y a 2 facteurs d'aléa dans notre modèle : la distribution finale et le fait que la distribution change à chaque fois que le modèle est "compilé" grâce à sa partie bayésienne.

Avec une distribution de sortie, le processus suivant permet l'obtention de la distribution (voir figure 2.21).

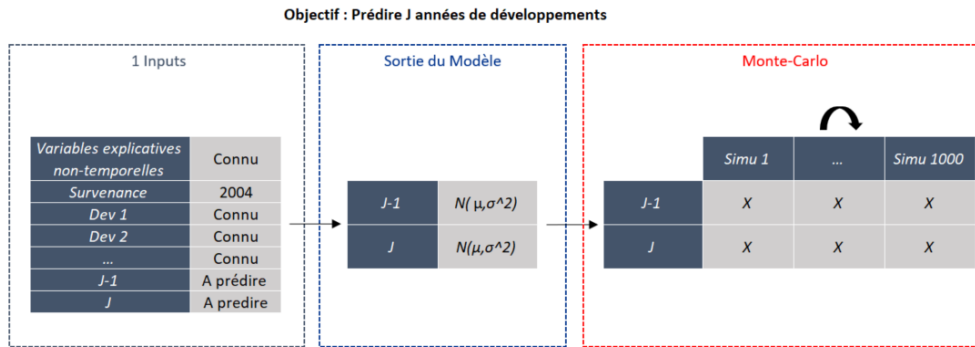


FIGURE 2.21 : Schéma de la prédiction d'un sinistre par Monte-Carlo

Une procédure classique de type Monte-Carlo est utilisée afin d'obtenir, par exemple, 1 000 simulations de Monte-Carlo de la valeur de la charge pour chaque année de développement inconnue d'un sinistre.

Afin d'avoir un résultat plus interprétable et comparable aux autres méthodes, le choix d'agrèger (en sommant) les résultats de chaque sinistre par rapport à la date de survenance et de développement a été retenu. Au final, pour chaque année de survenance, pour chaque année de développement inconnue, le modèle prédit donc 1 000 simulations de notre charge incrémentale. Une deuxième agrégation par sommation sur les différentes années de développement et en rajoutant la charge connue pour obtenir une distribution de la charge ultime prédite par année de survenance sera aussi réalisée.

De prime abord le moment où la partie bayésienne du réseau est appliquée n'est pas visible. Les poids du modèle sont fixés avant d'obtenir la sortie du modèle lors du changement de *batch* dans la prédiction. Le nombre de *batch* utilisé pour la prédiction est un hyper paramètre, mais il va être borné pour des raisons pratiques de mémoire. En effet, outre le fait de changer les poids du modèle, prédire par petits lots permet de ne garder en mémoire qu'un faible nombre d'informations.

La prédiction en une fois de 100 000 sinistres pour 17 années de développement, en supposant une

répartition uniforme des sinistres sur les années de survenance nécessite alors une capacité de stockage de

$$100\,000 \times \frac{17}{2} \times MC_{simu} \sim 1\,000\,000\,000 \text{ valeurs numériques,}$$

en notant MC_{simu} le nombre de simulations de Monte-Carlo choisit de préférence supérieur à 1000. Afin de réduire la mémoire nécessaire au stockage, les résultats sont agrégés comme expliqués précédemment par année de survenance. La capacité de stockage nécessaire devient de $17 \times 1000 = 17\,000$ valeurs numériques pour le stockage des résultats finaux, auquel il faut ajouter la mémoire nécessaire pour le calcul. Avec une taille de *batch* de 1000, le besoin est d'environ un million de valeurs numériques stockées, un nombre beaucoup plus raisonnable à gérer pour un ordinateur de bureau classique. Le changement des poids du BNN est ainsi naturellement effectué dans le processus.

Dans les parties précédentes, de nombreux hyper paramètres ont été cités. Afin d'obtenir le modèle ayant les meilleures prédictions, une calibration du modèle est nécessaire.

2.6 Processus de calibration du modèle

La calibration de chaque modèle est essentielle afin d'obtenir les meilleurs résultats possibles. En supposant que la base de données a déjà été retraitée au préalable, il faut ensuite sélectionner les variables explicatives et calibrer les hyper paramètres du modèle.

2.6.1 Sélection des variables explicatives

Pour sélectionner les variables explicatives les plus pertinentes pour chaque situation, plusieurs méthodes peuvent être utilisées.

Une première idée est de regarder la corrélation entre les variables et la variable à expliquer. Cependant, contrairement à d'autres modèles, il n'est pas prouvé que les réseaux de neurones sont négativement impactés par de fortes corrélations surtout dans le cadre d'un réseau de neurones profonds où l'objectif est de conserver l'ensemble des variables et laisser le modèle décider. Ce critère n'est donc pas un bon critère pour discriminer des variables.

Une deuxième idée serait de regarder une valeur signifiant l'importance de la variable comme la *p-values* dans une régression linéaire ou logistique. Malheureusement, dans les réseaux neuronaux, ce n'est pas si facile, car les *p-values* ne sont pas facilement disponibles.

Même si les méthodes classiques ne fonctionnent pas, d'autres méthodes existent telles que la sélection par le diagramme de Hinton ou la sélection par analyse descendante. Il faut cependant garder à l'esprit qu'un réseau de neurones suffisamment profond n'a théoriquement pas besoin d'une sélection de variables pour obtenir les meilleurs résultats, car il suffit de mettre les poids de la variable à 0 pour nullifier son effet. La sélection permet surtout une simplification du modèle afin d'améliorer la compréhension de l'impact de chaque variable sur le résultat. Une réduction du nombre de variables implique aussi une réduction du nombre de poids dans le modèle et donc une diminution du temps d'entraînement et de prédiction.

Sélection par diagramme de Hinton

Reprenant ce qui a été dit au-dessus, si le poids d'une variable est à 0 alors son impact est annulé. Une façon simple de trouver ces variables est de visualiser les poids dans un diagramme de Hinton. Un diagramme de Hinton visualise les poids entre les entrées et les neurones cachés sous forme de carrés, la taille du carré étant proportionnelle à la taille du poids et la couleur du carré représentant le signe du poids. Lorsque l'ensemble des poids reliant une variable aux neurones de la première couche cachée sont proches de zéro, celle-ci ne contribue pas et peut être abandonnée sans impacter les résultats. La figure 2.22 ci-dessous présente un exemple de diagramme de Hinton pour un réseau neuronal comportant 4 neurones dans la première couche cachée et 5 variables. Le schéma indique que la variable "revenu" a de faibles poids négatifs ou positifs par rapport aux autres variables. Elle n'impacte que peu le résultat du réseau, la supprimer est possible. La procédure de sélection des variables est très simple :

- Inspecter le diagramme de Hinton et supprimer la variable dont les poids sont les plus proches de zéro.
- Réestimer le réseau neuronal avec la variable supprimée ;
- Poursuivre l'étape 1 jusqu'à ce qu'un critère d'arrêt soit satisfait. Le critère d'arrêt peut être une diminution de la performance prédictive ou un nombre fixe d'étapes.

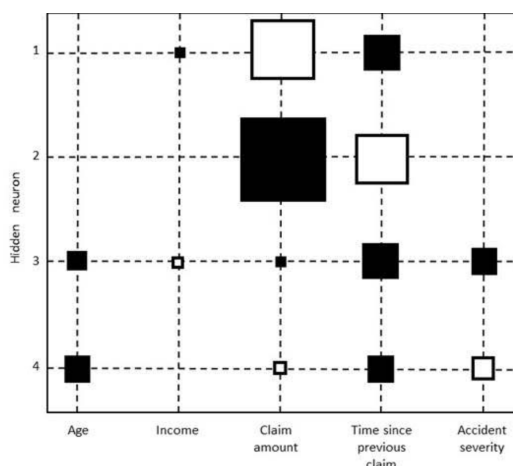


FIGURE 2.22 : Schéma d'un diagramme de Hinton d'un réseau de neurones multicouches

Sélection par analyse descendante

Une autre façon de procéder à la sélection de variables consiste à utiliser la procédure de sélection de variables par analyse descendante. L'idée est la même que la précédente, sauf que le critère d'avoir des poids faibles est remplacé par le fait que le réseau ne perd pas en performance sans la variable.

Algorithmiquement, il faut :

- Construire un réseau avec toutes les N variables ;
- Retirer chaque variable à tour de rôle et réestimer le réseau. N réseaux comportant chacun $N - 1$ variables sont obtenus ;
- Supprimer la variable dont l'absence donne le réseau le plus performant ;
- Répéter cette procédure jusqu'à ce que les performances diminuent de manière significative.

Tracer la performance en fonction du nombre de variables est équivalent à tracer une courbe de la performance à chaque itération de l'algorithme. Au début, la performance stagne ou peut même augmenter quelque peu. Lorsque des variables importantes sont supprimées, la performance commence à diminuer. Le nombre optimal de variables se situe alors autour du coude de la courbe.

Pour évaluer la mesure de la performance des modèles, la valeur de la *loss* sera utilisée, mais aussi la comparaison par rapport à un chain-ladder ou aux dires d'experts. Le calcul de cette valeur s'effectue soit sur la base test, soit durant l'entraînement par exemple. Le modèle invariant (sans BNN) sera utilisé pour l'ensemble de la sélection de variables et les variables retenues seront aussi considérées comme les plus pertinentes pour le réseau bayésien.

2.6.2 Calibration des hyper paramètres

Une fois les variables explicatives choisies, il va falloir calibrer les hyper paramètres du modèle. Les hyper paramètres d'un réseau de neurones comme celui de Kuo se divisent en 5 parties : les paramètres de l'apprentissage, de la loi de sortie, du RNN, du BNN et ceux de la prédiction.

L'algorithme consiste à tester sur le même jeu de données, le même réseau de neurones avec pour unique variation les différentes valeurs d'un hyper paramètre et ainsi choisir la valeur donnant le réseau le plus performant. Comme avec l'analyse descendante, si l'hyper paramètre est une variable numérique alors plusieurs valeurs de l'hyper paramètre seront testées. Une courbe des performances de chaque modèle pourra être tracée en espérant avoir une courbe linéaire permettant d'identifier un optimal visuellement.

L'objectif de cette partie est de déterminer l'ordre de test des hyper paramètres et pour chaque hyper paramètre, les valeurs à tester.

Les hyper paramètres de l'apprentissage d'un réseau de neurones

Les paramètres liés à la phase d'apprentissage sont les paramètres d'initialisation des poids et les paramètres d'apprentissage classique :

1. Le coefficient d'apprentissage α . Dans l'algorithme de descente de gradient, il limite l'actualisation des poids. La documentation du *package Keras* considère une valeur par défaut valant 0,01. Sachant que 1 correspond à l'apprentissage sans coefficient limitant, $\alpha \in [0.001, 1]$;
2. Si la structure comporte un réseau bayésien, le coefficient dans le terme de la divergence de KL entre les distributions de poids est modifiable. De base fixé à $\frac{1}{N}$, si l'apprentissage des poids varient trop entre les modèles alors ce paramètre peut être augmenté afin d'améliorer l'apprentissage de la distribution des poids ;
3. Le nombre d'*epoch* et la taille de *batch*. La valeur dépendra fortement de la structure du modèle. Une courbe sera utilisée pour déterminer visuellement l'arrêt de l'apprentissage et le début de l'*overfitting*. Cette courbe permet aussi de voir si l'apprentissage s'arrête et donc si un minimum local est atteint ou que le gradient disparaît à cause du grand nombre de pas de temps ;
4. L'initialisation des poids. Elle sera effectuée selon les différentes initialisations possibles précédemment énoncées (glorot uniforme, xavier normal, ...);

Les hyper paramètres d'une loi de sortie

L'ordre de sélection des hyper paramètres de la loi sera toujours le coefficient impactant la moyenne puis celui de la variance et dépendra de la loi choisie.

Les hyper paramètres d'un RNN

Les hyper paramètres du RNN, du fait de la structure déjà très figée d'un LSTM, ne concernent que le nombre de valeurs retournées :

5. La quantité d'information ressortie de l'encoder. Fixée à 3 dans le papier de recherche de KUO, 2020, diverses valeurs seront essayées (entre 1 et 5) sachant que le nombre de données en *input* limite la dimension maximale.
6. La quantité d'information ressortie par le décodeur. Les valeurs testées varieront entre 1 et 5 pour les mêmes raisons ;

Les hyper paramètres d'un *BNN* ou *RN dense*

Le choix des lois a priori et a posteriori pour les poids ne sera pas remis en cause (loi normale), mais certains paramètres peuvent être ajustés :

7. La variance de la loi a priori ;
8. Le facteur dans l'échelle de la loi a posteriori ;
9. Le nombre de couches cachées du *BNN* ou du *RN dense* (par défaut une seule). Un modèle avec 2 ou 3 couches cachées sera testé afin de s'assurer que le modèle n'est pas limité par la dernière couche ;

Les hyper paramètres de l'étape de prédiction

10. Le nombre de simulations effectuées de l'ensemble de la base de prédiction ;
11. La taille de chaque *batch* de simulations (changement des poids du modèle si *BNN*) ;
12. Le nombre de simulations de Monte-Carlo effectué. Les valeurs testées varieront entre 500 et 2000 suivant la volatilité des résultats obtenus.

L'ordre présenté ici n'est que théorique. En pratique, les paramètres d'entraînement varient en fonction de la complexité du modèle choisi et il faudra vérifier entre chaque changement majeur que l'entraînement du modèle reste optimal. Les hyper paramètres de la prédiction peuvent être calibrés bien avant, étant un peu à part car n'utilisant pas la valeur de la *loss* afin d'être acceptés.

2.7 Bilan

Ce chapitre a l'objectif d'ancrer toutes les informations nécessaires à la conception, la compréhension et les processus d'application de notre modèle à des données réelles.

La théorie des réseaux de neurones a premièrement été introduite à partir d'un réseau de neurones simple comme le réseau multicouche dense ce qui a permis d'expliquer l'ensemble des caractéristiques d'un réseau. En complexifiant le réseau de neurones à cause des besoins de la problématique (utilisation des données temporelles et obtention d'une distribution par une compréhension de l'aléa du modèle et des données), nous avons introduit les réseaux de neurones récurrents, les distributions de sorties et les réseaux de neurones bayésiens. L'ensemble de ces éléments a permis d'appliquer le modèle de Kuo. Cependant, appliquer aveuglément un modèle n'est pas un processus scientifique, il faut valider l'impact de chaque élément non indispensable tel que le *BNN*, à l'aide du modèle invariant par exemple. Afin d'appliquer le modèle sur nos données, le processus de calibration et de prédiction par Monte-Carlo a été développé. Cependant, comme le lecteur a pu le comprendre au cours de ce chapitre, les réseaux de neurones sont des objets complexes et ayant une allure de boîte noire. Une combinaison d'un modèle plus simple et interprétable comme un modèle linéaire généralisé avec un réseau de neurones appliqué aux résidus serait un test intéressant pour améliorer la compréhension des résultats, mais dépasse le champ d'application de cette étude.

Chapitre 3

Analyse du modèle

Ce chapitre a pour objectif de développer la partie pratique de ce mémoire. Dans un premier temps, une étude descriptive utilisant les statistiques descriptives permet d'analyser la base de données de responsabilité civile médicale.

Dans un second temps, nous effectuerons une description de la base de comparaison du modèle. La comparaison s'effectuera avec une méthode classique sur données agrégées comme Chain-Ladder, mais aussi avec les résultats d'experts provisionnement.

Dans un troisième temps, l'étude se portera sur l'initialisation du modèle comportant le format atypique des données en *input*, la sélection de la base d'apprentissage, des variables explicatives et des hyper paramètres.

Dans un quatrième temps, nous analyserons les résultats de ces modèles selon la qualité de la prédiction de la charge ultime, la qualité de la prédiction de la densité et l'analyse rétrospective du modèle par une méthode de *backtesting* à 2 ans. L'objectif est de comprendre les avantages et les limites de la prédiction. Dans un dernier temps, un essai d'amélioration du modèle sera détaillé suivi de conclusions et des améliorations possibles.

3.1 Étude de la base de responsabilité civile médicale

La base de responsabilité civile médicale a été brièvement présentée dans le chapitre 1 (cf. 1.5) en décrivant les variables explicatives temporelles et invariantes dans le temps. L'objectif d'un modèle de provisionnement dans notre étude est la prédiction de la distribution de la charge ou des paiements, à l'ultime.

Cependant, dans le cas d'une branche longue où l'information n'est pas suffisante même sur les années de survenance les plus anciennes, la charge ultime est plus facilement prédite, car elle contient l'information ajoutée du gestionnaire de sinistres. Une prédiction de la distribution de la charge à 1 an lors de la phase de *backtesting* sera aussi réalisée. Afin d'analyser ces résultats, il est important de comprendre en détail les caractéristiques et tendances de la base de données. (Pour préserver la confidentialité des données, les montants ont été multipliés par un facteur, proche de 1 pour conserver les ordres de grandeur et faciliter l'interprétation. Les montants seront donc exprimés en unités monétaires (UM)).

3.1.1 Les principales caractéristiques de la base

Un changement de loi impactant

Comme décrit précédemment, la base initiale possède les années de survenance 2000 à 2019. Cependant, la loi *About* vient changer le coeur de la garantie à partir de l'année 2003.

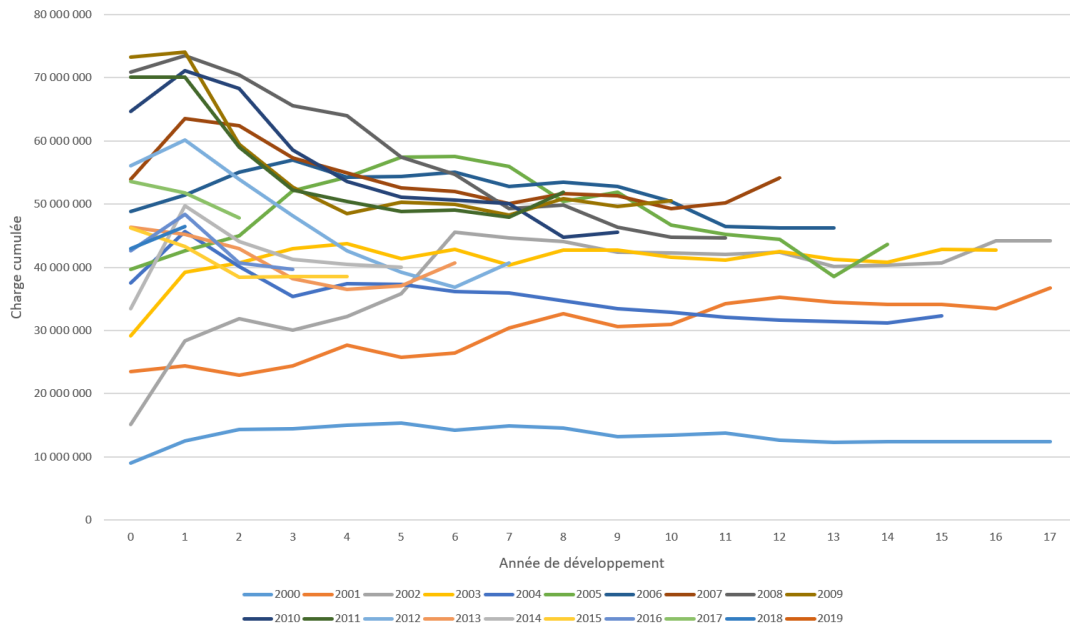


FIGURE 3.1 : Charge cumulée par année de développement pour chaque année de survenance

Sur le graphique 3.1, les charges cumulées des 3 premières années (2000 à 2002) se distinguent clairement de par leur tendance haussière contrairement aux autres années. De plus, la charge de l'année de développement 0 est clairement plus faible que les autres années. En prenant en compte le changement de loi *About* fin 2002, une exclusion des années 2000 à 2002 a été effectuée.

Les années les plus récentes n'apportent que peu d'informations à cause du faible nombre d'années de développement connues. Cependant, elles se démarquent par une charge plus faible après 2 ou 3 années de développement comparé aux années de survenance 2005 à 2011. Sur ce graphique, 3 comportements différents peuvent donc être observés suivant les années de survenance 2000 à 2002, 2003 à 2011 ou 2011 à 2019. Le graphique précédent montre aussi que la charge varie encore après de nombreuses années, c'est un signe de branche longue.

Une branche longue

Une des premières caractéristiques à connaître lors d'un exercice de provisionnement non-vie est la longueur de la branche. La longueur de la branche se caractérise par le nombre d'années nécessaires pour que l'ensemble des sinistres soit fermé. Dans notre base, elle n'est pas connue, car il reste encore des sinistres ouverts sur les dernières années.

Le graphique 3.2 montre que le nombre de sinistres ne tombe pas à 0, car le nombre d'années de développement est insuffisant ce qui indique que la branche responsabilité civile médicale est une

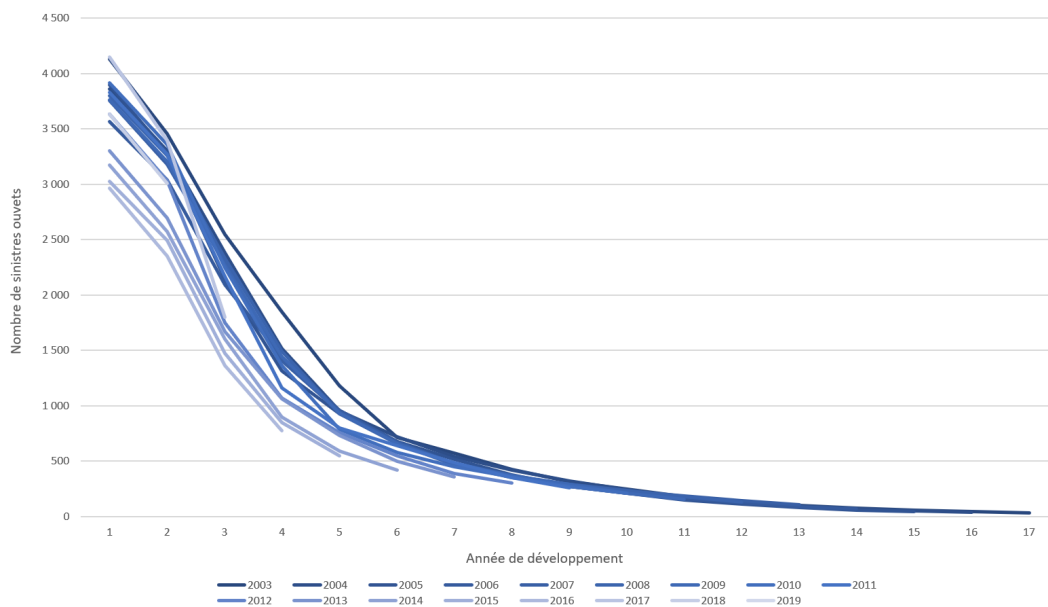


FIGURE 3.2 : Nombre de sinistres ouverts par année de développement pour chacune des survenances

branche longue. Ce comportement est prévisible, car pour avoir le coût définitif d'un sinistre de la branche, il faut établir 2 facteurs : la *quantum* et la part de responsabilité.

- Le *quantum* correspond au coût total du sinistre, en considérant une part de responsabilité de 100 %. Pour être déterminé, le *quantum* nécessite l'état stable de la victime. Or ce principe, dit "principe de consolidation" prolonge de plusieurs années la déclaration du montant définitif. Dans le cas de victimes mineures, le juge établit une indemnisation provisoire, mais attend la majorité de la personne afin de déclarer le *quantum* final.
- La part de responsabilité est aussi sujette à la création d'un délai, mais généralement elle est moins longue que le *quantum*. Il peut être compliqué de déterminer la part de responsabilité de chacun lors d'un accident médical, surtout lorsque l'élément déclencheur est multiple (par exemple, la perte d'utilisation d'un bras avec 3 opérations n'implique pas forcément 100 % de responsabilité pour la dernière opération et peut avoir été causée par la première).

Les données ne sont clairement pas complètes même pour les années les plus anciennes. Un facteur de queue pourra être considéré pour obtenir la charge ultime après application des méthodes de provisionnements classiques. Le modèle ne prédira pas la charge ultime, mais la charge cumulée à l'année de développement 16 pour être exact. Un autre point visible sur le graphique est la baisse du nombre de sinistres ouverts sur les dernières années de développement par rapport aux années plus antérieures. Cependant, la tendance reste très similaire. Ce comportement stable est une caractéristique inattendue sur une branche aussi instable.

Une homogénéité du nombre de sinistres

Une autre caractéristique importante de cette base de données est l'homogénéité des années de survenance. En effet, comme le nombre de sinistres par année de survenance du graphique (cf figure 3.2) le

montre, les sinistres sont relativement bien repartis entre les années.

Aucun changement de politique de souscription ni de changement du profil de risque recherché n'a été détecté au cours des dernières années. Ce critère semble cohérent, car les assurés du secteur sont limités (médecins, corps de métier médical et centres médicaux). Le nombre de sinistres étant stable sur chaque année de survenance, il ne faussera pas l'apprentissage du modèle en sur apprenant un comportement caractéristique d'une année surreprésentée.

Des sinistres graves

La base de données comporte un grand nombre de sinistres important. Observons le tableau des quantiles.

Quantile	0%	30%	35%	50%	75%	80%	85%	90%	95%	99%	100%
charge cumulée connue du sinistre	0	0	26	509	3 328	4 873	7 600	15 294	34 775	150 288	8 250 900
charge cumulée connue	0	0	0	2 131 553	25 305 654	37 736 559	56 083 607	88 775 664	167 108 628	329 198 079	744 570 535

TABLE 3.1 : Quantile de la charge cumulée connue

Sur la table 3.1, le quantile de la charge cumulée connue nous montre des statistiques intéressantes :

- 30% des sinistres ont une charge nulle et plus de 50% ont une charge très faible. Des sinistres ayant une charge nulle peuvent encore être modifiés, cependant la quantité importante de sinistres attritionnels est à remarquer ;
- Les 1% de sinistres les plus graves ont des montants de charges supérieurs à 100 000. Palier généralement utilisé pour séparer les sinistres entre eux ;
- Les 1% des sinistres les plus graves représentent 50% de la charge totale et les 5% des sinistres les plus graves sur le critère de la charge connue en vision 2019 représentent 75% de la charge connue.

L'objectif de prévision de la charge ultime est clairement influencé par les sinistres les plus graves. Cependant il est difficile de prédire si le sinistre sera un sinistre grave à l'aide des caractéristiques de l'année de développement 0 de celui-ci.

3.1.2 Analyse des variables explicatives

À l'aide de la base initiale explicitée précédemment, nous avons pu créer 16 variables explicatives qui potentiellement peuvent avoir un impact fort sur le résultat. Outre les 3 variables variant à chaque pas de temps (la charge incrémentale positive, la charge incrémentale négative et le statut du sinistre), 13 autres variables peuvent être analysées. Nous allons discuter les statistiques caractéristiques de ces variables afin d'émettre un avis d'expert sur leur implémentation dans le modèle.

Des sinistres tardifs

La longueur de la branche n'est pas la seule subtilité de ces données. En effet, certains sinistres sont déclarés lors d'un exercice comptable postérieur à celui de la survenance (cf. table 3.2). Ce cas est

fréquent sur la branche responsabilité civile médicale, car une erreur médicale non critique met souvent du temps avant de devenir visible. Il peut alors exister un délai entre l'erreur réalisée lors de l'acte couvert par l'assureur et la déclaration de l'assuré à la constatation des conséquences de l'erreur. L'exemple fréquent est celui d'un morceau de compresse oublié dans le corps d'un patient qui créera des problèmes plusieurs mois ou années après l'opération.

Année attendues av. déclaration	0	1	2	3	4	[5;10[[10;18]
Nombres de sinistres	51 743	8 479	556	165	66	77	18
% de la base totale	84,68%	13,88%	0,91%	0,27%	0,11%	0,13%	0,03%
Montant de charge concernée	628 620 013	94 184 423	7 982 865	7 687 356	624 896	5 148 292	322 690
% de la charge totale	84,43%	12,65%	1,07%	1,03%	0,08%	0,69%	0,04%
Charge moyenne	12 149	11 108	14 358	46 590	9 468	66 861	17 927

TABLE 3.2 : Statistiques descriptives des sinistres tardifs

Ces sinistres correspondent à une grande partie de la base de données (environ 15%). De plus, 18 sinistres ont été déclarés plus de 10 années après la survenance, ce qui confirme le caractère long de la branche. Initialement, une variable a été créée afin d'aider le modèle à comprendre le comportement différent de ces sinistres. Cependant comme les statistiques le montrent, la charge moyenne de ces sinistres n'est pas différente sauf pour les sinistres les plus tardifs. Même sur ces sinistres, le faible nombre fait qu'un seul sinistre très coûteux peut fausser la moyenne. Ces sinistres ne représentent qu'une faible proportion de la charge totale, il a été retenu de ne pas considérer les sinistres tardifs et donc la variable associée dans le modèle.

Des sinistres réouverts

En plus du caractère tardif des sinistres, certains sinistres sont sujets à des réouvertures. Une réouverture consiste à une seconde ouverture d'un sinistre pourtant clos. Un exemple classique est celui d'une opération qui laisse une trace visible à cause d'une erreur. Le sinistre est ouvert puis clos après compensation. Cependant, lors d'une opération future, un médecin découvre que l'erreur a aussi eu des conséquences internes. Le sinistre est ainsi réouvert.

Année attendues av. déclaration	Non Réouvert	Réouvert
Nombres de sinistres	57 338	3 766
% de la base totale	93,84%	6,16%
Montant de charge concernée	668 082 057	76 488 478
% de la charge totale	89,73%	10,27%
Charge moyenne	11 652	20 310

TABLE 3.3 : Statistiques descriptives des sinistres réouverts)

Les statistiques des sinistres réouverts (cf. table 3.3) montrent une séparation tranchée et non négligeable. Le nombre de sinistres concernés est faible, mais assez important pour avoir un impact significatif si le coût moyen global est considéré à la place du coût moyen des sinistres réouverts. Cette variable sera donc implémentée dans le modèle.

Les autres caractéristiques d'un sinistre

La base de données comporte d'autres variables. Maintenant que le modèle d'étude statistique a été présenté sur l'exemple des tardifs et des réouverts, nous allons effectuer le même raisonnement sur l'ensemble des autres variables afin de garder les variables significatives au sens du coût moyen. Il faut cependant garder à l'esprit que sur de faibles montants, l'impact d'un sinistre grave peut totalement fausser les conclusions.

Année attendues av. déclaration	Base entière	Litige	Hors litige	Matériel	Corporel	Représentant	Courtiers	Employés
Nombres de sinistres	61 104	9 099	52 005	21 610	39 494	17 996	42 934	174
% de la base totale	100,00%	14,89%	85,11%	35,37%	64,63%	29,45%	70,26%	0,28%
Montant de charge concernée	744 570 535	509 958 544	234 611 991	24 034 048	720 536 487	165 762 985	577 714 560	1 092 990
% de la charge totale	100,00%	68,49%	31,51%	3,23%	96,77%	22,26%	77,59%	0,15%
Charge moyenne	12 185	56 046	4 511	1 112	18 244	9 211	13 456	6 282

TABLE 3.4 : Statistiques descriptives des différentes variables

Dans cette première analyse (cf table 3.4), la variable de litige, du type du sinistre et de voie de distribution sont observées. D'après le critère de la charge moyenne, les variables de litige et du type de sinistre sont très impactantes. De plus, la proportion de la charge totale impactée est significative. Ces deux variables sont donc potentiellement pertinentes pour le modèle. Quant à elle, la variable de voie de distribution possède des variations entre ces différentes modalités. L'écart entre la charge moyenne des sinistres du réseau employé et les autres est significatif. Cependant, la faible proportion de la charge ultime en jeu en fait une variable faiblement intéressante à première vue pour notre modèle. Elle ne sera pas implémentée.

Année attendues av. déclaration	Base entière	Avec coassurance	Sans coassurance	Catégorie RCCE	Catégorie RCME
Nombres de sinistres	61 104	2 305	58 799	38 881	17 073
% de la base totale	100,00%	3,77%	96,23%	63,63%	27,94%
Montant de charge concernée	744 570 535	9 850 008	734 720 527	711 518 708	13 437 334
% de la charge totale	100,00%	1,32%	98,68%	95,56%	1,80%
Charge moyenne	12 185	4 273	12 495	18 300	787

TABLE 3.5 : Statistiques descriptives des différentes variables

Sur la deuxième analyse (cf table 3.5), les variables de co-assurance et la catégorie des sinistres sont représentées. La variable de catégorie comporte 34 modalités différentes. Cependant plus de 90 % des sinistres sont décrits par 2 modalités (RCCE/RCME pour responsabilité civile exploitation des dommages corporels/matériels). La différence de charge moyenne étant significative entre ces 2 catégories, cette variable est intéressante. Cependant, elle est grandement corrélée avec la variable du type de sinistre. La variable de catégorie est plus complexe de compréhension du fait de son grand nombre de modalités (36 contre 2 pour la variable du type de sinistre). La variable de catégorie du sinistre ne sera pas implémentée dans la liste des variables principales. La variable de co-assurance apparaît aussi comme significative, mais avec un faible nombre et montant de sinistres. Elle ne sera pas prioritaire dans l'implémentation, mais pourra être considérée comme une piste d'amélioration.

D'autres variables créées à partir des variables initiales ont été essayées, cependant aucune segmentation intéressante n'a été relevée sur le critère du coût moyen sauf la variable du mois d'ouverture du sinistre dans la base de données de l'assureur (cf table 3.6).

Année attendues av. déclaration	Base entière	Mois d'ouverture du sinistre			
		Janvier	Juin	Septembre	Décembre
Nombres de sinistres	61 104	4 558	5 870	4 789	5 012
% de la base totale	100,00%	7,46%	9,61%	7,84%	8,20%
Montant de charge concernée	744 570 535	75 475 906	50 023 443	47 801 676	80 667 452
% de la charge totale	100,00%	10,14%	6,72%	6,42%	10,83%
Charge moyenne	12 185	16 559	8 522	9 982	16 095

TABLE 3.6 : Statistiques descriptives du mois d'ouverture

Les sinistres ont une répartition globalement homogène entre les différents mois d'ouverture. Cependant, une différence de coût moyen marquante peut être observée, notamment aux extrêmes (sur les mois de janvier et décembre pour un fort coût moyen et sur les mois de juin et septembre pour les plus faibles coûts moyens). La charge concernée est de l'ordre de 6 à 11 % de la charge totale. L'impact n'est pas négligeable, ce qui valide une implémentation de la variable du mois d'ouverture malgré la faible compréhension de la dynamique causant ces variations.

L'analyse descriptive nous indique une première liste de variables explicatives :

- Litige ;
- le type de sinistre ;
- Le mois d'ouverture ;
- la réouverture.

La variable de catégorie ne sera pas dans la liste réduite, car trop corrélée avec le type de sinistre. Cette liste pourra évoluer au cours de la sélection de variables et des choix de modèles.

Afin de sélectionner les meilleures variables et les meilleurs hyper paramètres, il va falloir comparer les différents résultats des modèles obtenus avec une base de comparaison.

3.2 Base de comparaison d'un modèle

Notre modèle vise initialement à calculer la charge ultime. Cependant, comme l'analyse descriptive l'a démontré, la longueur de la branche responsabilité médicale fait que la profondeur de l'historique disponible n'est pas suffisante pour obtenir la charge ultime. Il faut donc rajouter un facteur de queue. Cependant, cela n'empêche pas un modèle d'être entraîné ni comparé sur la prédiction avant l'ajout d'un facteur de queue. Plusieurs méthodes sont possibles. Celles présentées ici sont la méthode de Chain-Ladder et le résultat des travaux de provisionnement d'actuaire non-vie expérimentés (cf figure 3.7).

Les actuaires provisionnement de « KPMG » ont effectué en interne des travaux de provisionnement sur les données. Leur objectif était de créer un intervalle de meilleure estimation (*BE*) afin de pouvoir comparer leurs estimations au modèle. Les résultats sont les suivants :

Année de survenance	charge connue	Chain Ladder	Moyenne expert provisionnement
2003	42 709 809	42 709 809	42 597 156
2004	32 365 545	32 214 782	32 732 258
2005	43 678 279	44 614 775	45 050 488
2006	46 186 618	50 123 420	47 302 533
2007	54 150 742	57 614 482	53 235 918
2008	44 677 318	46 911 777	46 654 219
2009	50 491 075	52 420 210	50 808 995
2010	45 553 773	46 012 289	45 263 017
2011	51 900 585	51 666 561	49 265 046
2012	40 716 646	39 768 526	39 689 550
2013	40 636 625	38 877 820	39 226 591
2014	40 024 901	37 816 522	39 431 168
2015	38 521 491	34 913 771	35 742 052
2016	39 662 329	35 096 082	36 379 440
2017	47 839 085	40 031 025	40 930 290
2018	46 405 483	35 845 197	37 256 695
2019	39 050 231	27 260 576	32 572 988
Ultime totale	744 570 535	713 897 623	714 138 405

TABLE 3.7 : Résultat sur données agrégées par Chain-Ladder en millions d'UM

	Charge ultime prédite	PSAP
Min	675 504 582	-69 065 953
Max	739 704 342	-4 866 193
Moyenne	714 138 405	-30 432 130

TABLE 3.8 : Résultats des experts provisionnement

Les dires d'experts (cf table 3.8) s'accordent sur une charge ultime inférieure à la charge connue (environ 745 millions). Cela suggère une certaine prudence de la part des gestionnaires de sinistres, ce qui n'est pas impossible.

3.3 L'initialisation du modèle de Kuo

Précédemment, nous avons expliqué l'importance de l'initialisation d'un réseau de neurones et de la mise en place de son architecture. Nous allons décrire ici en détail la séparation de notre base de données, la sélection de nos variables explicatives et les différents hyper paramètres testés en incluant l'initialisation.

3.3.1 La séparation de la base de données

L'objectif de prédiction du modèle couplé à un faible nombre de données a conduit à une séparation simple : une base d'apprentissage composée de 95 % de nos données et une base de validation de 5 % pour s'assurer que le modèle n'apprend pas trop des données ou le choix des hyper paramètres. La

base de test est composée de l'ensemble des données, l'objectif étant une prédiction des montants des années de développements inconnus. Pour le cas du *backtesting*, le même processus sera retenu.

3.3.2 Le format des données et les changements du modèle de Kuo

La reprise du modèle de Kuo consiste à reprendre l'architecture théorique de son modèle, sa loi de sortie et les différents *inputs* préconisés. Cependant après plusieurs essais pratiques, la variabilité des résultats et le manque d'apprentissage ont donné lieu à des modifications du format de l'*input* et de l'*output* de comparaison dans la phase d'apprentissage. Pour mieux comprendre cela, nous expliquerons dans un premier temps le format initial puis les modifications choisies.

Les *inputs*

Ce modèle prend en entrée un sinistre composé de variables explicatives non temporelles dont le traitement (OHE, embedding, etc.) a déjà été abordé et des variables explicatives temporelles telles que la charge incrémentale positive et négative ou le statut du sinistre.

Un modèle de provisionnement utilise généralement la charge ou les paiements. Le modèle de Kuo prend l'hypothèse de l'utilisation de la charge. La charge peut être vue sous 2 formats différents : la charge incrémentale ou la charge cumulée. Un réseau de neurones va uniquement prédire des montants de charges incrémentaux. En effet, dans une prédiction incrément par incrément, un poids mal calibré est beaucoup moins susceptible de fausser la prédiction globale qu'un poids mal calibré dans un modèle cumulé, la prédiction incrémentale pouvant toujours capitaliser sur la bonne prédiction des incréments précédents.

La charge se présente dans le modèle comme un vecteur de valeurs numériques sur laquelle la transformation de réduction a été appliquée. Le modèle de Kuo sépare la charge incrémentale positive de la charge incrémentale négative. Cette séparation a été reprise dans cette section, car le but est d'appliquer le modèle le plus proche possible de celui de Kuo sur des données réelles. De plus, changer cette hypothèse remet en cause l'utilisation d'une loi log-normale.

En effet, la densité des montants incrémentaux est regroupée autour de 0. Cependant des montants de charge importants aussi bien positivement que négativement avec de faibles probabilités existent. Une loi à queue épaisse telle que la loi log-normale peut tout à fait s'ajuster à des données positives de ce type sous réserve d'un décalage pour prendre en compte le 0, mais la loi log-normale ne peut pas prédire de larges montants négatifs. Ainsi la séparation entre la charge incrémentale positive et négative permet d'ajuster 2 lois log-normales aux données.

L'information d'un sinistre peut être représentée sous le format visible sur la table 3.9. Il est

Variable explicative invariante			Variable explicative variantes								
Var 1	...	Var K	Charge Dev 0	...	Charge Dev 16	Recours Dev 0	...	Recours Dev 16	Statut Dev 0	...	Statut Dev 16

TABLE 3.9 : Format d'un sinistre dans la base initiale

composé de l'ensemble des variables explicatives décrites dans l'analyse de la base, des variables de charges incrémentales positives et négatives, du statut de sinistre en vision en fin 2019 et de la date d'observation des données de la base.

L'incrément temporel des variables temporelles est les années de développement ("Dev" sur la table 3.10). Contrairement à l'intuition d'un actuare provisionnement travaillant sur des données agrégées, les montants temporels commencent à partir de l'ouverture du sinistre et non de la survenance du sinistre.

	Survenance	Ouverture	Dev 0	Dev 1	Dev 2	Dev 15	Dev 16
début à la survenance	2003	2004	0	10	12	...	25
début à l'ouverture	2003	2004	10	12	...	25	X

TABLE 3.10 : Format d'un sinistre dans la base initiale

Sur le schéma 3.10, la différence d'incrémentation des années de développement sur données agrégées ou sur données ligne à ligne apparaît. Sur des données agrégées, la norme est d'avoir des années de développement par année de survenance afin de pouvoir visualiser les montants d'une année comptable donnée sur chaque diagonale. Sur une base de données ligne à ligne, les développements sont comptabilisés à partir de l'ouverture. La différence est donc uniquement sur les sinistres tardifs, qui représentent 15 % de la base. Cette distinction est importante, car il est nécessaire de fixer si le modèle apprendra et prédira le comportement des sinistres tardifs avec des 0 initialement ou non. La logique est de conserver des années de développements commençant à la date d'ouverture du sinistre, car la cadence de développements d'un sinistre tardif à partir de l'année d'ouverture est similaire à celle d'un sinistre non tardif. En effet, la charge incrémentale est majoritairement définie à l'ouverture donc sur l'année de développement 0. Prendre en compte un 0 sur cette année de développement fausserait les prévisions des modèles. Le format retenu sera celui des développements débutant à l'année d'ouverture du sinistre.

À l'aide de l'information d'un sinistre au format explicité ci-dessus, il faut à présent transformer l'information en l'*input* pour le modèle de KUO. L'*input* a un format différent suivant son utilisation : l'entraînement du modèle (sur le triangle supérieur en format agrégé) ou la prédiction de la charge incrémentale inconnue (sur le triangle inférieur en format agrégé).

Pour l'entraînement du modèle, l'information de chaque sinistre en vision fin 2019 est donc déclinée en plusieurs lignes d'entraînement correspondant à la vision de chaque fin d'année connue (sous réserve d'information disponible).

	Variable explicative invariante			Variable explicative temporelle			A Predire		
Vision 2003	Var 1	...	Var K	Dev 0			Charge Dev 1	...	Charge Dev 16
...	Var 1	...	Var K	Charge Dev 0	...	Charge Dev ...	Charge Dev	Charge Dev 16
Vision 2017	Var 1	...	Var K	Charge Dev 0	...	Charge Dev 14	Charge Dev 15	...	Charge Dev 16
Vision 2018	Var 1	...	Var K	Charge Dev 0	...	Charge Dev 15	Charge Dev 16		

TABLE 3.11 : Décomposition d'un sinistre d'ouverture en 2003 en plusieurs lignes pour l'entraînement du modèle

Un sinistre datant de 2003 (comme sur le schéma 3.11) possède 17 années de développement connues. Ce sinistre se sépare donc en 16 lignes composées chacune de la vision à une année donnée de ce sinistre. L'année de développement 0 est nécessaire à la prédiction du modèle et l'année de développement 16 est toujours nécessaire à la comparaison. Elle ne peut pas être utilisée dans l'input de l'entraînement, car au moins une valeur connue est nécessaire lors de l'entraînement.

Sur la première ligne, l'information est celle disponible la première année, soit fin 2003 et la prédiction vise les développements survenus les années 2004 à 2019. Sur la seconde ligne, l'information est celle disponible la deuxième année, soit fin 2004 et les développements survenus les années 2005 à 2019. Ainsi, une démultiplication de l'information disponible est possible et permet l'entraînement du modèle. Par généralisation, tout sinistre possédant $j \in [0, 16]$ années de développement connues (y compris celle où le sinistre est fermé) se transforme en j lignes d'apprentissages du modèle (où n'est pas prise en compte dans le cas de la dernière année de survenance, car $j = 0$).

La base de données initiale après retraitement est composée d'environ 62 000 lignes à vision fin 2019 se transforme en une base de 498 000 lignes, ce qui est cohérent, car en remarquant l'équipartition du nombre de sinistres sur les années de survenance, un sinistre dispose d'environ 8 années de développement connues en moyenne. Ainsi, la base comporte environ

$$62000 \times 8 = 496\,000 \text{ lignes.}$$

La base d'apprentissage comporte ainsi 95% des données soit 473 200 lignes et la base de validation comporte 24 900 lignes.

Observons la répartition de ces lignes suivant leur dernière année de développement connue (ou plutôt sélectionnée comme dernière année de développement connue s'il y a eu une décomposition en plusieurs lignes).

	Dev 1	Dev 2	Dev 3	Dev 4	Dev 5	Dev 6	Dev 7	Dev 8	Dev 9	Dev 10	Dev 11	Dev 12	Dev 13	Dev 14	Dev 15	Dev 16
Toute la base	57 663	54 034	49 889	46 926	43 901	40 729	37 429	33 798	29 972	26 056	22 299	18 502	14 758	11 209	7 351	3 570
5 dernières années connues	16 934	16 605	16 091	16 954	17 845	18 430	18 927	19 040	18 763	18 705	18 729	18 502	14 758	11 209	7 351	3 570

TABLE 3.12 : Nombre de lignes dans la base d'entraînement selon la dernière année de développement connue

Sur la table 3.12, la première ligne correspond à la méthode classique explicitée précédemment. Une disproportion logique due à la démultiplication des lignes les plus anciennes apparaît. Afin de réduire cet impact, une autre solution peut être proposée : limiter la démultiplication des lignes de la base d'entraînement à la vision des 5 dernières années. Avec ce changement un sinistre de 2003 (le plus ancien) ne correspondra plus qu'à 5 lignes au maximum et la prédiction sera effectuée sur 5 lignes maximum dans la phase d'apprentissage.

Une autre problématique est d'obtenir un vecteur de prédiction de taille uniforme alors que le nombre d'années à prédire dépend de la dernière année connue. Cette problématique se résout dans l'application en masquant les années sur lesquelles la fonction de *loss* ne doit pas être calculée (masquer signifie que ces valeurs ne seront pas impliquées dans le calcul de la *loss*, ni du gradient lié à celle-ci et n'influenceront donc pas sur les poids).

Le modèle de KUO décrit ce format de donnée, cependant l'ordre dans lequel la base de comparaison est présentée au modèle a été modifié.

Prenons l'exemple de la charge positive. Ce schéma représente donc le format de présentation de la variable de charge positive utilisée comme *output* de comparaison dans le calcul de la *loss*. Les croix signifient une absence d'information et sont remplacées en pratique par une valeur "9999" qui sert de masque et ne sera pas comptabilisée dans le modèle. Peu importe la méthode, la même information est requise. Ce qui change est la disposition de cette information. Le modèle de Kuo considère que le premier élément du vecteur de comparaison est la valeur de la première année de développement inconnue du sinistre. La modification considère que le vecteur de comparaison est fixe et que la première année de développement à prédire n'est pas forcément à la même place.

Vision 2003					
output de comparaison KUO	Dev 1	Dev 2	...	Dev 15	Dev 16
output de comparaison modifié	Dev 1	Dev 2	...	Dev 15	Dev 16
Vision 2004					
output de comparaison KUO	Dev 2	Dev 3	...	Dev 16	X
output de comparaison modifié	X	Dev 2	...	Dev 15	Dev 16
Vision 2017					
output de comparaison KUO	Dev 15	Dev 16	...	X	X
output de comparaison modifié	X	X	...	Dev 15	Dev 16

FIGURE 3.3 : Schéma du format de l'output de comparaison nécessaire à l'entraînement du modèle

Ce changement d'ordre provient d'une constatation. En effet, après quelques essais sur les données, l'ordre du modèle de KUO, 2020 crée une uniformisation des montants sur les différentes années de développement. En effet, dans l'étape d'apprentissage, l'*output* de comparaison fait que le modèle doit apprendre des poids qui vont servir à prédire l'année de développement 1 pour un sinistre de vision 2003, mais ces mêmes poids serviront à prédire l'année de développement 2 pour un sinistre de vision 2004 et ainsi de suite. C'est donc une prédiction du terme suivant et non d'une année de développement fixée. La seule variation sera l'*input* qui contiendra l'information de l'année supplémentaire. En pratique cela n'est pas suffisant pour effectuer une réelle variation du résultat entre les années de développement sur la base de données. Pour pallier ce problème, la solution trouvée a été de fixer le vecteur de prédiction et de masquer les valeurs déjà connues. Par souci de cohérence, l'*input* aussi a été changé par fixation des années de développement. Le modèle a ainsi des poids attribués pour chaque année de développement.

La base d'entraînement, la structure du modèle et les variables ont été décrites. Afin d'obtenir un modèle pertinent, il faut ensuite effectuer une sélection des variables et des hyper paramètres.

3.3.3 La sélection des variables explicatives

À l'aide des différentes analyses statistiques précédentes, une liste de variable à implémenter a été créée. Outre les variables temporelles qui ne seront pas rediscutées pour le moment, les variables de litige, du type du sinistre, de réouverture et du mois d'ouverture auront leur impact sur le modèle testé.

Cependant dans l'objectif de minimiser l'ajout d'expertise humaine, une analyse descendante a été testée sur l'ensemble des variables explicatives (y compris celle non recommandée par l'analyse statistique). Après plusieurs essais, la volatilité de prédiction du modèle étant beaucoup trop grande, l'impact du retrait d'une variable est invisible. Malgré les diverses solutions utilisées pour réduire la volatilité, cette méthode s'est conclue par un échec dû aux grands nombres de variables explicatives et l'impossibilité de fixer la reproductibilité du modèle à chaque entraînement.

Afin de faciliter le processus, le périmètre des variables a été restreint à la liste composée des variables sélectionnées par l'analyse statistique. De même, le réseau de neurones bayésien a été remplacé par un réseau de neurones dense plus facile à manipuler. La sélection de variables s'effectue donc sur le modèle invariant.

La subtilité du calibrage d'un modèle repose sur l'ordre de sélection entre la sélection de variables

et la sélection des hyper paramètres. En effet, si les hyper paramètres sont trop peu calibrés, les résultats seront anormaux et il sera difficile de valider chaque variable sachant que l'apprentissage est catastrophique. Dans le cas où les hyper paramètres sont calibrés en premier, il ne faut pas que le choix de ceux-ci implique un biais dans la sélection de variables. La solution retenue a été d'effectuer une sélection d'hyper paramètres rapide avec l'ensemble des variables puis d'effectuer la sélection de variables par analyse descendante.

La variabilité de l'apprentissage du modèle, problématique centrale qui augmente drastiquement la difficulté de comparaison entre 2 modèles, dépend de 3 éléments : la base d'entraînement (ici fixée), le mélange à chaque *epoch* (fixé par une graine en pratique) et l'initialisation des poids de ce modèle. Les poids d'un modèle sont aléatoires, mais déterminants suivant que l'apprentissage finira dans un minimum local ou global. Ils existent 2 solutions pour pouvoir tirer des conclusions malgré ces variations. La première est de fixer les poids (en fixant une graine ou par initialisation manuelle). La seconde est de faire plusieurs entraînements afin d'obtenir des résultats comparables et de prendre le minimum par exemple de 3 à 5 modèles. C'est la seconde méthode qui a été retenue, car les poids peuvent être fixés qu'une fois l'architecture et les variables définissent. Observons les variations dues à chaque variable sur le modèle invariant 3.13.

Listes des variables :	Litige	Mat/Corpo	Mois_ouv	Ré-ouverture	toutes variables	Aucune variables
Loss obtenue sans la variable	0,701	0,618	0,607	0,629	0,622	0,713
Validation Loss obtenue sans la variable	0,728	0,647	0,639	0,655	0,652	0,741
Variable supprimée ?	Non	Non	Oui	Non		
Loss obtenue sans la variable	0,660	0,620	X	0,626		
Validation Loss obtenue sans la variable	0,697	0,652	X	0,652		
Variable supprimée ?	Non	Oui	X	Non		
Loss obtenue sans la variable	0,674	X	X	0,638		
Validation Loss obtenue sans la variable	0,720	X	X	0,657		
Variable supprimée ?	Non	X	X	Oui		

TABLE 3.13 : Résultat de l'analyse descendante sur le modèle invariant

Chaque colonne correspond à la suppression de cette variable dans le modèle. Les 3 premières lignes nous indiquent le montant minimum de la *loss* (sur la base d'apprentissage et de validation) pour 5 entraînements du modèle à l'aide des mêmes variables (l'initialisation en aléatoire). Le modèle, avec l'ensemble des variables de la liste réduite de l'analyse descriptive, atteint une *loss* de 0.65 et une *loss* de 0.74 sans aucune des variables. L'impact des variables reste significatif, cependant le modèle peut tout à fait être entraîné sans.

La première sélection est claire, la variable de litige ne doit pas être retirée. Cependant, la proximité des valeurs de *loss* sur les 3 autres modèles implique qu'ils n'ont qu'un faible impact, voire un impact négatif. Un impact négatif est purement impossible, car en prenant des poids à 0, il est possible de supprimer l'impact d'une variable. Un impact négatif d'une variable signifie donc par abus de langage qu'elle ralentit l'apprentissage et que ses poids sont faibles.

Après analyse descendante, la variable de litige apparaît clairement comme la variable la plus impactante pour le modèle. Dans l'objectif d'obtenir le meilleur modèle, la variable de réouverture a aussi un impact significatif même s'il est plus faible. Les 2 variables sont ainsi gardées.

Afin de ne pas oublier les autres variables de notre modèle, l'impact de chaque variable a été étudié. Tour à tour un modèle composé de la variable litige, réouverture et de la variable ajoutée a été entraîné

(cf. table 3.14). Cependant, sur le critère de *loss*, aucune des variables n'a amélioré significativement la *loss* du modèle.

Listes des variables :	Année de développement (num)	Année d'ouverture (num)	Année d'occurrence (num)	Année de développement (cat)	Année d'ouverture (cat)	Année d'occurrence (cat)	Line of business (cat)	Tardif (cat)	Catégorie (cat)	Moyen de distribution (cat)	Co-assurance (cat)	Le mois de survenance (cat)
Loss obtenue avec le modèle avec variable	0,643	0,632	0,637	0,678	0,636	0,660	0,639	0,684	0,634	0,684	0,662	0,821
Validation Loss obtenue sans la variable	0,671	0,659	0,665	0,695	0,643	0,677	0,665	0,711	0,660	0,710	0,686	0,848
Variable retenue ?	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non	Non

TABLE 3.14 : *Loss* des modèles avec une variable ajoutée

La variable de l'année d'ouverture et la variable de l'année de survenance peuvent être considérées soit comme des variables numériques, soit comme des variables catégorielles. La distinction a été faite ici afin de couvrir l'ensemble des possibilités, cependant aucune de ces variations d'*inputs* n'a pu améliorer le modèle significativement. Afin de simplifier l'interprétabilité du modèle, ces variables ne seront pas conservées.

La liste des variables explicatives invariantes par pas de temps définitives du modèle invariant est donc uniquement composée de la variable de litige et de réouverture.

Le modèle invariant a une structure composée d'un *embedding layer*. Pour récapituler, le modèle invariant utilise :

- Un *embedding layer* pour les variables catégorielles ;
- Une transformation de réduction des variables numériques.

Cependant, d'autres méthodes ont été introduites dans la partie théorique. Afin de challenger ce choix de structure, la sélection de variables a été réalisée avec un *OHE* pour les variables binaires.

Le changement de retraitement des *inputs* catégoriels n'a pas permis d'améliorer les performances du modèle, les valeurs de *loss* étant globalement plus élevées. L'*embedding layer* n'amointrit pas les résultats du modèle, mais l'améliore. Seul l'impact du type de sinistre se dégage comme étant plus important avec la méthode *OHE* qu'avec un *embedding layer*.

Listes des variables :	Méthode de traitement
Litige (cat)	Embedding
Type M/C (cat)	OHE
Ré-ouverture	Embedding
Loss obtenue	0,618
Validation Loss	0,652

TABLE 3.15 : Modèle retenu après la sélection de variable et résultats

La table 3.15 montre le modèle obtenu à l'issue de la sélection de variables. La valeur de sa *loss* est la valeur minimale trouvée pour les paramètres fixés choisis. Les variables suivantes seront donc conservées avec leur traitement pour la suite de ce chapitre : la variable de litige et la variable de réouverture avec un traitement dans le modèle par un *embedding layer* et la variable du type de sinistre avec un simple *OHE*. La prochaine étape est la calibration des hyper paramètres du modèle.

3.3.4 La sélection des hyper paramètres

À l'aide de la méthode de l'analyse descendante, nous avons fixé les 4 variables du modèle. L'étape suivante est de continuer à minimiser la fonction de loss en modifiant les hyper paramètres sur le modèle invariant.

Il est complexe de bien fixer totalement un modèle. Un modèle, suivant la qualité des poids initiaux (qui est aléatoire), va apprendre plus ou moins vite. Afin d'éviter ce problème, maintenant que le modèle a une structure fixée, les mêmes poids seront utilisés pour l'ensemble de la sélection des hyper paramètres (en dehors des variations de méthodes d'initialisation des poids logiquement). Les poids retenus sont des poids initiaux permettant à un modèle invariant de converger rapidement sur 15 epochs pour obtenir une *loss* minimale avoisinant les 0.6. Plus précisément, ce sont les poids initiaux du modèle de la figure 3.15 qui sont repris. Une reproductibilité de l'entraînement du modèle est ainsi obtenue. Chaque variation de résultat correspond alors à l'impact de chaque hyper paramètre.

Les hyper paramètres de l'apprentissage

Les premiers paramètres à être appris sont le nombre d'*epochs* et le nombre de *batches*. Ils n'ont pas besoin d'être parfaits, mais l'apprentissage d'un réseau de neurones nécessite de fixer ces paramètres dans le bon ordre de grandeur évitant tout sous-apprentissage (*underfitting*) ou surapprentissage (*overfitting*).

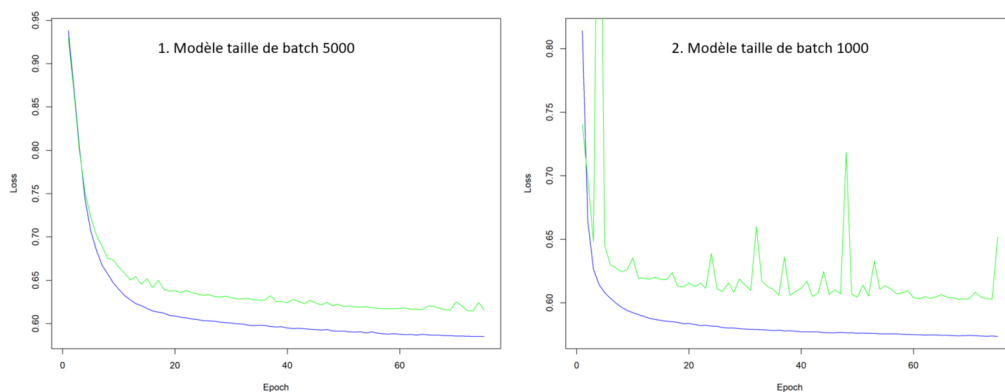


FIGURE 3.4 : Courbe de la *loss* en fonction du nombre d'*epochs* pour 2 modèles différents de par la taille du *batch* pris : 5000 ou 1000.

Sur la figure 3.4, pour le premier modèle, la *loss* de l'échantillon de validation stagne entre 40 et 60 epochs malgré la pente légèrement décroissante de la *loss* de la base d'apprentissage. L'apprentissage s'arrête ici, si l'apprentissage continue, il y a un fort risque d'*overfitting*. Un palier est donc atteint. Le nombre de 50 *epochs* est choisi sous réserve de garder la même taille de *batch* et le même coefficient d'apprentissage.

Pour le second modèle ayant une plus faible taille de *batch*, il y a un plus grand nombre d'actualisations de poids pour chaque *epoch* ce qui entraîne une augmentation logique de la volatilité de la *loss*. Le résultat final est meilleur aussi bien sur l'échantillon d'apprentissage que celui de validation, mais la volatilité des résultats peut indiquer une plus grande sensibilité du modèle. Un nombre de 50 *epochs* est risqué, car très proche d'un grand pic de variation. Le nombre de 60 *epochs* semble cohérent.

Il est nécessaire d'observer ce graphique et de sélectionner le bon nombre d'*epoch* à chaque chan-

gement d'un paramètre afin de s'assurer que l'apprentissage reste faiblement volatile.

Epochs	75	75	75	75	75	75
Taille de batch	5 000	20 000	1 000	200	5 000	1 000
Coefficient d'apprentissage	0,1	0,1	0,1	0,1	0,01	0,01
Epoch de l'arrêt de l'apprentissage	50	75	60	50	Insuffisant	Insuffisant
Loss finale obtenue	0,585	0,609	0,573	0,567	0,649	0,626
Validation Loss finale obtenue	0,616	0,638	0,605	0,600	0,677	0,666

TABLE 3.16 : Résultat des différentes *loss* suivant le nombre d'*epochs*, la taille de *batch* et le coefficient d'apprentissage

Sur la table 3.16, plusieurs valeurs pour la taille de *batch* et le coefficient d'apprentissage ont été testées. L'augmentation de la taille de *batch* permet une plus grande stabilité de la courbe de la *loss* au détriment de la vitesse d'apprentissage. La taille de batch de 5000 permettant aussi l'obtention de résultats stables, la recherche a été concentrée sur des tailles plus petites. Pour une taille de 1000 ou 200, les résultats sont bons, mais plus volatils comme le graphique précédent a pu le montrer. Contre la prédiction naturelle, la taille de *batch* de 200 présente une courbe plus stable jusqu'à 70 *epochs* que celle de 1000. Les résultats avec un faible coefficient d'apprentissage ne sont pas concluants et nécessitent plus de temps d'apprentissage pour être performants, ce qui les écarte, car un temps d'apprentissage court est préférable.

En se basant sur le critère de la *loss* et de la volatilité, les hyper paramètres retenus sont 50 *epochs* avec 5000 lignes par *batch* et 0.1 de taux d'apprentissage.

Les hyper paramètres de la loi de sortie

Une fois les hyper paramètres de l'apprentissage retenus, les prochains paramètres à calibrer sont ceux de la loi de sortie.

La loi de sortie est fixée à une loi log-normale. La loi initiale (cf. équation 2.15) se traduit avec ses hyper paramètres par

$$P_1 \times (\log \mathcal{N}(X_1, \text{Min var} + \text{Scale}_1 \times S(\text{Scale}_2 \times X_2)) + \text{Shift}) + P_2 \times 0. \quad (3.1)$$

Avec

- $S(X) = \frac{1}{1+\exp(-X)}$: la fonction sigmoïde vue précédemment dans la partie associée aux fonctions d'activation. Cette fonction permet de réduire la valeur de la variance entre 0 et 1. Cette réduction a été reprise du modèle de Kuo, mais doit être vérifiée ;
- $\text{Min var} = 0,5$ par défaut pour éviter les problèmes de *loss* négative ;
- $\text{Scale}_1 \in [0.01, 1]$ est un terme permettant de limiter ou non la variance de la loi normale sous-jacente à celle de la loi log-normale après la sigmoïde. Ainsi, si les variations sont limitées à 0.01, la variance de la loi appartient à $[0.50, 0.51]$. C'est une hypothèse non négligeable stabilisant le modèle ;

- $Scale_2 \in [0.01, 1]$ est un autre terme limitant le terme de variance, mais avant l'application de la sigmoïde. Ce terme permet de réguler le montant de X_2 afin de faciliter l'apprentissage. L'impact de ce terme est moindre logiquement ;
- $Shift \in [-0.3, 3]$ est un terme permettant d'inclure le 0 dans les valeurs possibles de la loi log-normale.

Le premier terme à tester est celui de la variance en ajoutant de la liberté au réseau. Ainsi le premier test vise à valider l'utilisation d'un coefficient réducteur et de la fonction sigmoïde dans la variance de la loi. Une taille de *batch* de 1000 (à la place de 5000) a été utilisée pour accroître le nombre d'actualisation, compensant ainsi le ralentissement de l'apprentissage causé par la non-réduction de la variance avec une fonction sigmoïde.

Reduction de variance par sigmoïde	Aucune	Présente
Scale 1	0,10	0,01
Scale 2	1	1
Shift	-0,75	-0,75
Meilleure loss obtenue	0,534	0,576
Meilleure validation Loss obtenue	0,558	0,605

TABLE 3.17 : Hyper paramètres des meilleurs modèles avec et sans réduction par fonction sigmoïde obtenue

Sur la table 3.17, la sélection des hyper paramètres influant sur la variance de la loi du modèle a été testée. Une première séparation a été effectuée entre un modèle avec une réduction sigmoïde du paramètre de variance et sans cette réduction. L'ordre des hyper paramètres testés est ensuite composé du coefficient de variance hors sigmoïde et de celui à l'intérieur de la sigmoïde (fixé à 1 sans fonction sigmoïde). Pour finir, le décalage de la loi permettant d'inclure le 0 a été fixé.

Le modèle a plus de facilités à apprendre 10 fois la variance et le meilleur décalage est d'environ 0.75. Ces choix de paramètres se comprennent, car la variance de la loi log-normale est petite. L'ordre de grandeur des inputs et des poids suggère des résultats assez grands. Une division facilite alors l'apprentissage. Ce décalage est celui utilisé comme paramètre initial et avait été calibré après quelques essais sur un modèle sans aucune variable ajoutée. Un biais provenant de la sélection des autres hyper paramètres en utilisant cette valeur fixe de décalage a probablement été créé. Malgré tout, cette valeur est une approximation de l'optimum dans notre démarche.

Pour le modèle avec réduction sigmoïdale, le terme de décalage et de *scale* est similaire. La réduction de la variance dans la fonction sigmoïde n'a que peu d'impact sur la *loss* (de l'ordre 10^{-4}). L'hyper paramètre n'ayant pas d'impact, il est supprimé (mis à 1).

Sur les 2 meilleurs modèles, le modèle retenu d'après le critère de la *loss* est celui sans la fonction de réduction de variance sigmoïdale.

Le changement d'initialisation

Un modèle de réseau de neurones est toujours très dépendant de l'initialisation de ces poids. 2 initialisations différentes impliquent des apprentissages différents et l'apprentissage ne se finit pas toujours

dans le même minimum local. Il est beaucoup plus difficile de faire apprendre au modèle à sortir d'un minimum local durant l'apprentissage que de relancer avec différentes initialisations le même modèle afin de l'éviter. Les initialisations possibles sont celles de *Glorot* et de *Xavier*, uniforme ou normale. Par défaut, le package *keras* inclus dans le package *tensorflow* ALLAIRE et TANG, 2022, utilise une initialisation glorot uniforme (hors biais initialisés à 0). Depuis le début de cette phase de calibrage des hyper paramètres, l'ensemble des travaux a été effectué avec des poids initiaux favorisant les résultats. Quelques essais non concluants ont été réalisés avec d'autres initialisations. Le modèle retenu continuera d'utiliser des poids initialisés par une méthode de glorot uniforme.

Les hyper paramètres de la prédiction

Les hyper paramètres de prédiction sont simples à définir dans un modèle sans partie bayésienne, car la taille de *batch* de prédiction dépend seulement des problèmes de mémoires abordés dans l'explication du processus de Monte-Carlo. Elle est fixée à 5000. De même, il n'est pas nécessaire de simuler plusieurs fois le modèle. La simulation multiple du modèle est très coûteuse en temps et a pour avantage dans le cas d'un réseau bayésien de pouvoir appliquer plusieurs distributions de poids pour le même échantillon. Cependant, sans partie bayésienne il est inutile de simuler plusieurs fois l'entraînement du même modèle. Le nombre de simulations est ainsi fixé à 1.

Pour trouver le nombre de simulations de Monte-Carlo nécessaire à une convergence et une stabilité des résultats du modèle, il suffit d'entraîner un modèle et de le prédire plusieurs fois avec différents nombres de simulations de Monte-Carlo.

Le nombre de 1000 simulations pour chacune des 16 distributions a été retenu. Malgré la faible variation restante, le temps de calcul conséquent de 30 minutes pour une prédiction de résultats avec 1000 simulations est raisonnable comparé aux 5 heures pour 10 000 simulations. Un moyen pour accélérer la prédiction des résultats serait d'améliorer la mémoire RAM qui limite la parallélisation du calcul en empêchant l'ordinateur de charger en mémoire beaucoup de données simultanément, ou une plus grande capacité de parallélisation du code.

Le modèle retenu

Le modèle final utilisera les variables de charge incrémentale positive et négative, le statut du sinistre à chaque année de développement, les variables explicatives de litige, du type de sinistre et de réouverture. Le modèle final retenu aura comme hyper paramètres : 50 *epochs*, 5000 de taille de *batch*, pas de fonction sigmoïde, une *scale* de 0,1 dans la variance et 1000 simulations de Monte-Carlo par *batch* de 5000 lignes.

L'ensemble des paramètres ont été testés pour une structure fixe. Cependant, challenger cette structure est un moyen de confirmer les choix de modèle comme cela a été le cas avec le traitement des variables catégorielles, mais cela ne sera pas fait ici.

Bilan de la sélection du modèle

La sélection de variables et des hyper paramètres d'un modèle de réseaux de neurones profonds est une étape complexe et très chronophage. Cette étape est cruciale, car de la sélection découle l'entièreté des résultats que nous présenterons par la suite. L'idéal serait d'effectuer une sélection à l'aide de

la combinaison de l'ensemble des possibilités de chaque hyper paramètre et variables, afin de ne pas introduire de biais en fixant par défaut les hyper paramètres avant la sélection.

Pour la sélection de variables, le fait que les poids ne puissent être fixés, car le nombre de poids diffère suivant le nombre de variable, crée une volatilité dans les résultats obligeant à reproduire plusieurs fois le même modèle avant de sélectionner le meilleur. Avec les 19 couples de variables et traitements différents possibles, et un minimum de 5 prédictions par couple, l'analyse descendante effectuée considère un minimum de $25 \times 5 = 125$ modèles à entraîner.

Pour la sélection d'hyper paramètres, en considérant des poids fixés, les 10 hyper paramètres du modèle impliquent avec notre sélection un minimum de 22 modèles entraînés et l'ajout des variations de l'initialisation des poids et de la structure augmente ce chiffre à 37. Le temps cumulé nécessaire à la reproduction de cette sélection se compte en jours et avoisine les 3,5 à 5 jours en cumulé. Le principal défaut de la méthode est donc son temps d'application.

Concernant les améliorations, différentes structures ont été testées. Cependant l'architecture basique du modèle n'a pas été modifiée. Le modèle final comporte encore une grande volatilité de la *loss* sur l'échantillon de validation. La limite de ce type de méthodologie est que le modèle apprend de manière autonome sur les données sans jamais prendre en compte l'objectif de prédiction de la charge ultime. Ce type d'approche ne garantit donc en rien la bonne prédiction sur des années inconnues. D'autres approches consistent à choisir le meilleur modèle basé sur la valeur de la *loss* et sur sa volatilité, mais aussi en prenant en compte une distance tolérable par rapport à la prédiction (dans notre cas par rapport aux dires d'experts) malgré le biais que cela peut induire. L'approche réalisée limite le biais induit, mais favorise la possibilité d'une prédiction anormale. Observons à présent les résultats du modèle invariant retenu.

3.4 Analyse des résultats du modèle de Kuo retenu

Après la sélection de variables et la sélection des hyperparamètres uniquement basés sur la valeur de la *loss*, la prochaine étape est la prédiction des résultats du modèle sur la base test. Les résultats obtenus vont être mis en perspective avec la charge connue actuelle (après réagrégation de la charge positive et négative prédite du modèle).

3.4.1 Les résultats

Le modèle obtenu prédit les résultats suivants :

La table (3.18) montre les résultats du modèle calibré. L'observation immédiate est la distance de la prédiction de la charge ultime du modèle avec les dires d'experts qui est d'environ 700 à 800 millions. Cette prédiction est totalement incohérente avec la réalité. La prédiction est très loin en dehors de la range des dires d'experts. Sur les années de survenance les plus anciennes (2003 - 2006), le modèle n'est pas trop déraisonnable. Il prédit une hausse de la charge, ce qui est un comportement prudent. Cependant sur les années de survenance les plus récentes la prédiction explose. La charge contient les paiements et l'estimation du gestionnaire de sinistre. Des variations peuvent survenir, mais l'ordre de grandeur est incohérent.

Année de survénance	Charge prédite model	Charge prédite dire d'experts	Charge connue
2 003	42 884 968	42 001 023	42 709 809
2 004	32 487 623	32 450 756	32 365 545
2 005	44 144 601	43 051 899	43 678 279
2 006	47 551 613	43 264 392	46 186 618
2 007	56 667 112	40 330 551	54 150 742
2 008	48 965 375	47 455 923	44 677 318
2 009	56 626 192	47 676 307	50 491 075
2 010	54 197 108	46 696 955	45 553 773
2 011	64 591 443	43 363 023	51 900 585
2 012	56 575 982	37 547 237	40 716 646
2 013	59 983 219	41 278 429	40 636 625
2 014	63 521 558	46 612 004	40 024 901
2 015	75 885 851	35 940 441	38 521 491
2 016	93 805 480	43 266 517	39 662 329
2 017	137 335 346	49 390 300	47 839 085
2 018	155 712 211	51 767 671	46 405 483
2 019	150 036 084	39 741 115	39 050 231
Ultime totale	1 240 971 766	731 834 541	744 570 535

TABLE 3.18 : Résultat brut du modèle retenu

La prédiction d'une densité

Observons la densité des prédictions

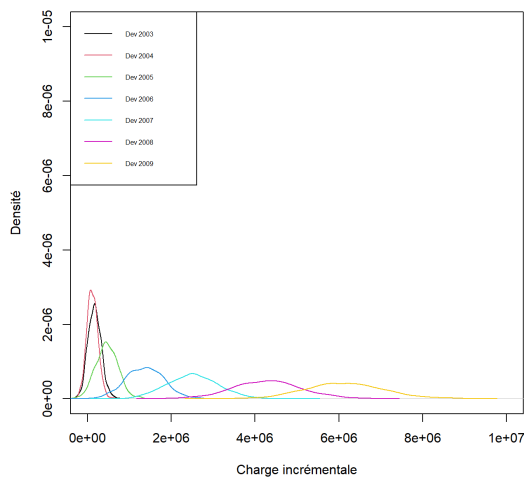


FIGURE 3.5 : Densité de la charge ultime prédite par année de survénance (1 à 7)

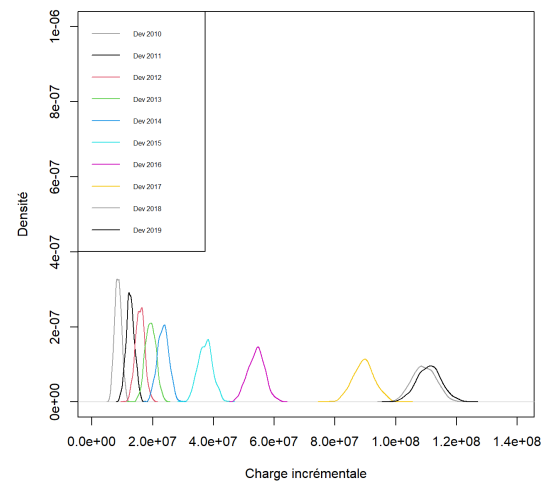


FIGURE 3.6 : Densité de la charge ultime prédite par année de survénance (8 à 16)

La moyenne de chaque prédiction (utilisée afin d'obtenir les résultats précédents) est faussée à partir de l'année 2007. Il est donc plus difficile d'interpréter les courbes. Néanmoins, l'ordre des prédictions est conservé. De plus l'étendue de la densité varie et s'agrandit sur les prédictions des années récentes. Ce comportement logique se justifie par la hausse du nombre d'années de développement à prédire. Le modèle prédit en moyenne une hausse de la charge sur chaque développement. L'apprentissage d'une

charge incrémentale négative (en vision agrégée) n'est pas validé.

Le *backtesting* du modèle

Afin de compléter notre analyse du modèle, un *backtesting* à 2 ans a été réalisé. L'objectif étant aussi de pouvoir tester la qualité de la densité trouvée. La base d'entraînement correspond à la vision fin 2017. La prédiction visera les charges incrémentales de l'année calendaire 2018 et 2019.

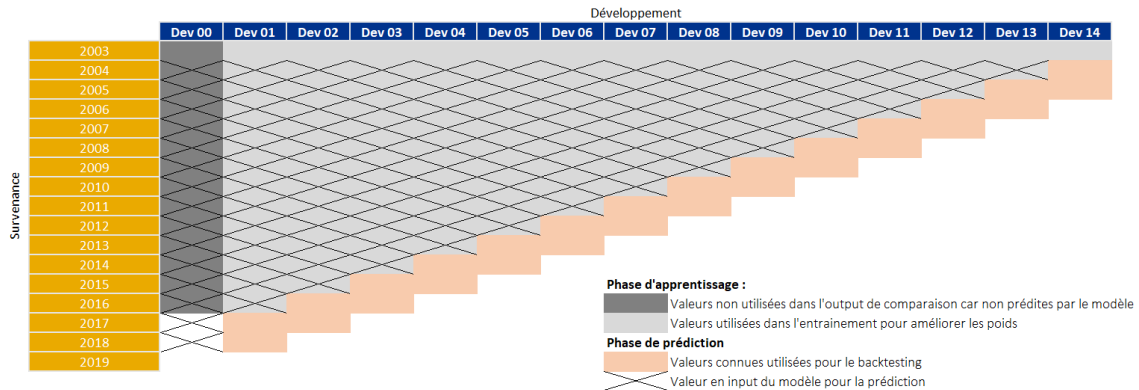


FIGURE 3.7 : Schéma des bases de données nécessaires au *backtesting* sous la forme d'un triangle en vision agrégée

La figure (3.7) indique les différents développements conservés pour chaque phase du *backtesting* selon l'année de survenance du sinistre. La vision agrégée est utilisée pour faciliter la compréhension, mais les sinistres de la base sont bien au format ligne à ligne.

À l'aide de la structure, des variables et des hyper paramètres choisis précédemment, nous entraînons le modèle sur les bases définies dans le schéma. Comparons les résultats obtenus.

	Backtesting de l'année calendaire 2018	Backtesting de l'année calendaire 2019	Backtesting de l'année calendaire 2018 et 2019
réel	-26 676 796	16 522 455	-10 154 341
Moyenne de la distribution prédite	62 806 188	56 254 733	119 060 921
Quantile 5% de la distribution prédite	56 131 893	49 734 545	105 866 437
Quantile 95% de la distribution prédite	69 527 608	62 929 515	132 457 124

TABLE 3.19 : Résultat du *backtesting*

En observant, les chiffres de la table 3.19, la conclusion naturelle est que le modèle est loin d'obtenir de bons résultats. La valeur réelle n'est même pas dans l'intervalle entre les quantiles 5% et 95%. Le montant réel des années calendaires est très volatile. Cependant, les prédictions du modèle sont uniquement positives et l'ordre de grandeur n'est pas respecté. Le modèle ne s'applique pas bien aux données.

Dans la prochaine section, nous allons détailler les limites rencontrées avec ce modèle.

3.4.2 Les limites de l'application du modèle

Lorsqu'un modèle ne permet pas d'obtenir une bonne prédiction, cela peut venir de trois causes distinctes.

La première non négligeable est une erreur humaine dans le code. C'est le cas lorsqu'il y a une différence entre le code actuel et le code théorique voulu par la personne. Ces erreurs sont souvent des erreurs non bloquantes pour l'exécution, ce qui les rend difficiles à repérer.

Une attention particulière doit être portée sur les données utilisées et sur leurs retraitements. Une vérification détaillée de la base d'*input*, d'*output* de comparaison et d'*input* de prédiction est nécessaire.

La deuxième cause provient des dynamiques sous-jacentes à la base de données. Ces erreurs se repèrent à l'aide de statistiques descriptives et d'études de sensibilité des variables.

La troisième cause provient de la structure et des hypothèses du modèle qui ne sont pas vérifiées ou de l'entraînement qui n'est pas optimal.

Dans ce mémoire, l'erreur humaine existe, mais a été réduite au maximum par plusieurs relectures. L'entraînement a été challengé à de nombreuses reprises durant la phase de sélection des hyper paramètres. Nous allons donc nous concentrer sur les dynamiques sous-jacentes à la base de données, la structure et les hypothèses du modèle.

Plusieurs intuitions ressortent du lot : la variable litige semble problématique, le modèle semble peiner à apprendre les informations temporelles. À l'aide de la prédiction d'un seul sinistre dont nous ferons varier les caractéristiques, nous allons essayer de comprendre les limites du modèle appris.

Un changement de comportement de la variable litige

La sélection de variables a fait ressortir la variable de litige comme une des variables les plus importantes. Observons son comportement par année de survenance.

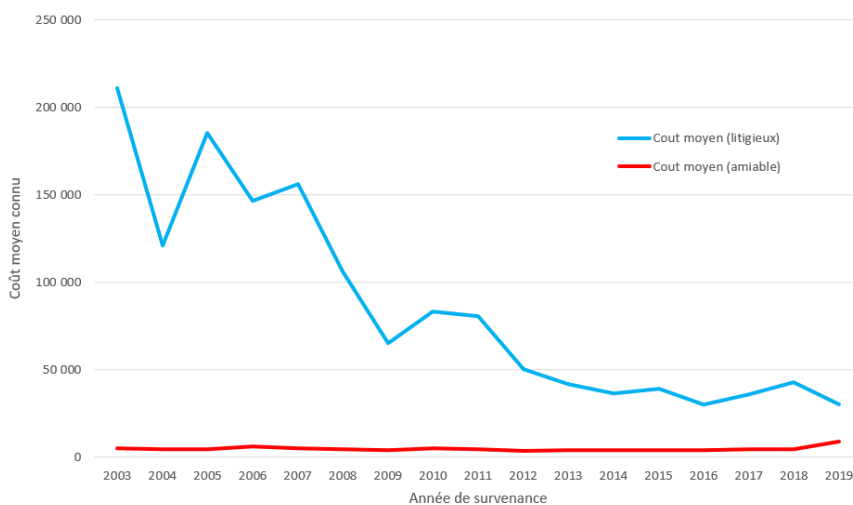


FIGURE 3.8 : Courbe de l'évolution temporelle de la charge moyenne connue des sinistres litigieux ou à l'amiable

Sur la figure 3.8, nous observons que la variable de litige a un comportement atypique. Sur les années de survenance les plus antérieures, un sinistre litigieux a un très fort coût moyen et il y a très peu de sinistres litigieux (environ 2%). Sur les années les plus récentes au contraire, les sinistres déclarés comme litigieux sont beaucoup plus nombreux (environ 25 % des sinistres totaux). Avec ces informations, il semble que le comportement des sinistres litigieux a fortement changé entre 2003 et 2019. Cependant, la comparaison est effectuée entre la charge au développement 16 pour l'année 2003 et la charge au développement 1 pour l'année 2019. Cependant la conclusion reste la même. Pour une explication plus détaillée, se référer à l'annexe (A).

Les 2 autres variables du modèle, la variable de réouverture et du type de sinistre sont relativement stable entre les différentes années de survenance.

Afin de valider la théorie de sur prédiction du montant des sinistres litigieux des années récentes, nous allons étudier la prédiction de la charge ultime de la survenance 2019 d'un sinistre. Le modèle va prédire la charge incrémentale des années de développement 1 à 16 qui, agrégée, permettra d'obtenir la prédiction de la charge ultime du sinistre. Un sinistre en 2019 dispose en *input* du modèle de 3 variables explicatives (la variable de litige, du type et de réouverture) qui seront toutes testées. Il est forcément ouvert au développement 0 et l'ouverture ou la fermeture des autres années ne sont pas connues. La charge incrémentale positive en 0 sera choisie, la charge incrémentale négative vaut toujours 0 au développement 0 et les autres années ne sont pas connues. Afin d'obtenir une meilleure visualisation de l'ensemble des prédictions possibles, dans le sous-ensemble des sinistres de 2019 ayant les mêmes variables explicatives, le minimum, la médiane, la moyenne et le maximum de la charge incrémentale positive au développement 0 seront testés.

		Litige	Litige	Litige	Litige	amiable	amiable	amiable	amiable
		Corporel	Corporel	Matériel	Matériel	Corporel	Corporel	Matériel	Matériel
		Reouvert	Non-reouvert	Reouvert	Non-reouvert	Reouvert	Non-reouvert	Reouvert	Non-reouvert
Nombre de sinistres		12	709	0	16	15	901	12	1 272
Pred x Nombre	Prediction avec min	1 297 973	67 643 088	0	1 467 618	483 855	-494 550	211 785	-8 966 328
	Prediction avec médiane	1 429 524	82 572 834	0	1 538 979	606 001	21 005 301	274 141	-10 546 152
	Prediction avec moyenne	1 454 194	82 151 901	0	1 591 093	717 530	22 841 053	233 338	-10 814 024
	prediction avec max	1 422 188	41 929 388	0	1 767 618	717 670	10 566 063	290 780	-1 006 999
PSAP predite	Prediction avec min	61 643 441							
	Prediction avec médiane	96 880 629							
	Prediction avec moyenne	98 175 084							
	prediction avec max	55 686 708							

TABLE 3.20 : Tableau de PSAP obtenue sur la survenance 2019 selon les variables et la valeur de la charge connue au développement 0

Sur la table 3.20, les sinistres de la survenance 2019 ont été séparés selon leurs variables explicatives. En utilisant une méthodologie "Nombre x Coût Prédit d'un sinistre", le but ici est d'observer les causes de la sur prédiction du modèle.

Le résultat est assez transparent, ce sont les sinistres litigieux corporels non réouverts qui sont à l'origine de ces montants de charges trop importants et ce peu importe la charge initiale choisie. Afin d'améliorer le modèle, il est important de déterminer les années de développement sur lesquels la charge explose.

Sur la figure 3.9, nous pouvons observer 2 comportements distincts entre les sinistres litigieux ou non. Les sinistres litigieux ont une prédiction haute pour chacune des années de développement. Les sinistres litigieux corporels non réouverts ne sont donc pas les seuls à avoir une charge prédite trop importante, c'est le cas de tous les sinistres litigieux. Le montant en jeu est simplement plus important du fait du grand nombre de sinistres litigieux corporels non réouverts.

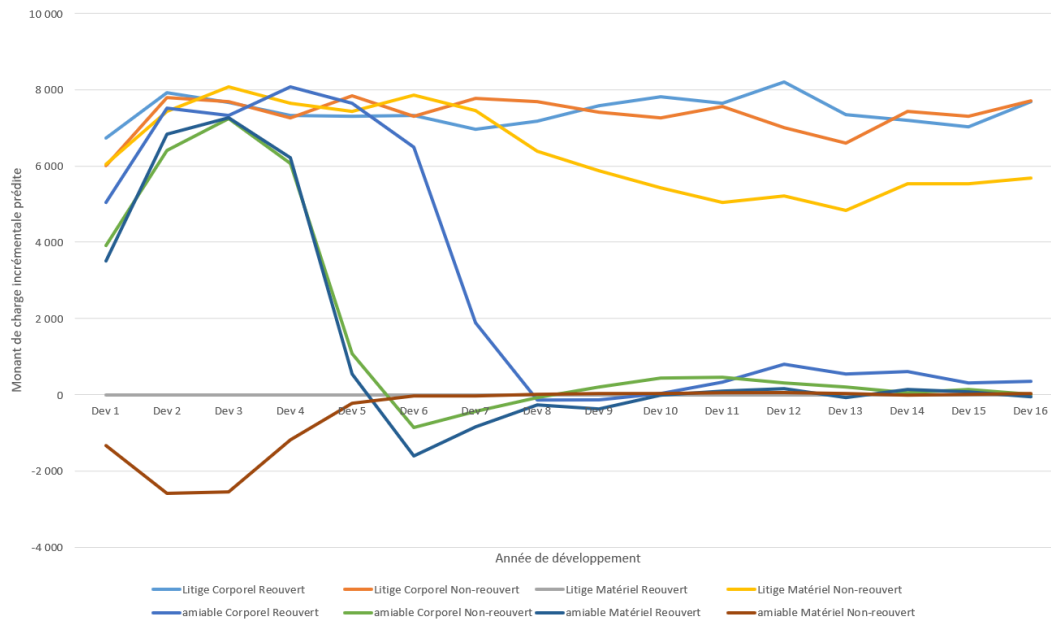


FIGURE 3.9 : Courbe de PSAP obtenue sur la survénance 2019 par année de développement pour la prédiction d'un sinistre suivant ces variables explicatives

Comme expliqué dans l'annexe A, le comportement haussier dans les 5 premières années de développement d'un sinistre est un comportement révolu sur les dernières survénances. Cependant cette hausse étant grandement présente dans la base de données, il est logique que le modèle l'apprenne. Cette forte hausse des premières années fausse grandement les résultats.

En se référant au graphique 3.1, la variation de charge entre l'année de développement 10 et 16 ne dépasse pas les 5 millions pour chaque année de survénance. Or ici, sur la survénance 2019, nous avons environ 35 millions entre le développement 10 et 16. Ce montant s'explique par la variation de comportement (montant et nombre) des sinistres litigieux corporels au cours du temps.

En plus de la surestimation de la charge pour les sinistres litigieux, le montant prédit est régulier sur les années de développement et notamment sur les dernières années ce qui est tout à fait anormal. Les sinistres à l'amiable ont eux un comportement dit "normal" avec de très faibles variations prédites sur les dernières années.

Une des idées précédemment abordées avec la figure 3.12, est d'utiliser une démultiplication d'un maximum des 5 dernières années connues de chaque sinistre. Cette modification de la base d'apprentissage a pour but de ne pas apprendre à prédire les premiers développements des années de survénance les plus anciennes. Dans la pratique, les résultats obtenus avec cette nouvelle base sont légèrement moindres. La prédiction s'améliore de 100 millions, ce qui est insuffisant pour conclure à une réelle avancée des résultats.

Une deuxième limite provient de l'hypothèse initiale de séparation des incréments de charges positifs et négatifs.

La séparation entre les charges incrémentales positives et négatives

Cette séparation n'est pas anodine. Ce choix de modèle permet l'utilisation d'une loi log-normale. Plus exactement, lors du calcul de la log-vraisemblance, la valeur $f_X(\text{output de comparaison})$ est calculée. Ce calcul n'est pas possible pour l'ensemble des valeurs négatives de l'*output* de comparaison, lors de l'utilisation de la densité de probabilité d'une loi log-normale. La prédiction de montants négatifs étant nécessaire au modèle, une séparation a été créée.

Cependant, ce choix comporte des inconvénients. En observant la base de données, de nombreux sinistres rencontrent la même situation : le gestionnaire de sinistre estime une hausse de la charge une année puis se ravise l'année suivante. Dans le but d'obtenir la meilleure estimation de la charge ultime, ce comportement difficile à prédire pour le modèle actuel induit une hausse des montants prédits. Théoriquement, cela ne change pas le résultat agrégé sur l'année de survenance. Mais, au niveau individuel, la prédiction de la charge ultime d'un sinistre a de plus grandes chances d'être fautive, car cela augmente les montants de charge incrémentale positive et négative prédite.

Un autre inconvénient majeur est le fait que la prédiction de la charge cumulée totale puisse être négative. La base ne prend pas en compte les recours. Il ne devrait pas y avoir de charge cumulée négative pour un sinistre. Or, la déconnexion des montants positifs et négatifs prédits rend possible cette situation. Si un sinistre à une charge initiale nulle (environ 25% de la base actuelle), et la prédiction est une prédiction négative comme c'est le cas pour la prédiction de sinistre de 2019 à l'amiable, matériel et non réouvert, le modèle prédit une situation impossible en pratique : une charge cumulée négative.

Une idée d'amélioration, en gardant la séparation de la prédiction, serait d'agréger les résultats dans le modèle à l'aide d'une seule distribution composée des 2 précédentes. Il serait alors possible de prédire un mélange des 2 distributions prédites actuellement. Cette distribution ne permettrait qu'un seul montant prédit (positif ou négatif). Afin de résoudre le second problème, une borne doit aussi être ajoutée afin de ne pas rendre la charge cumulée négative. Ces possibilités n'ont pas été implémentées dans ce mémoire.

Une autre solution serait un changement de loi de sortie, permettant l'utilisation de la charge incrémentale négative et positive (par exemple une loi normale). Cependant, cette séparation peut avoir une utilité sur une base de données ayant des comportements de charge incrémentale positive et négative différents, mais stables, afin de pouvoir dissocier les prédictions.

Le faible impact des variables temporelles

Contrairement aux autres variables explicatives, les variables temporelles ont été considérées comme appartenant à la structure du modèle initial de Kuo. Cependant, cela ne garantit pas leur utilité. L'impact significatif du développement 0 de la charge incrémentale positive a été montré dans la partie précédente sur l'année 2019 en observant les différences de prédictions du tableau 3.20 entre une charge minimale et une charge maximale sur l'année de développement 0.

Analyser l'influence de chaque temporalité de la charge incrémentale positive, négative et du statut du sinistre peut être effectué soit par test de la prédiction d'un seul sinistre, soit en observant les poids du réseau. Cependant, la structure composée d'un LSTM encoder et décodeur rend l'analyse des poids trop complexe.

Prenons alors l'exemple d'un sinistre de survenance 2011 (au milieu des années de survenance). En modifiant légèrement chaque temporalité, l'impact sera mis en évidence. Formellement, ce processus

est appelé étude de sensibilité. Afin de valider l'hypothèse "certaines temporalités sont inutiles dans le modèle" de grandes modifications seront effectuées à la place : un ajout de 40 k(UM). Ce montant correspond à l'écart-type des charges incrémentales de la base.

Charge incrémentale positive augmentée de 40k€								
	Dev 8	Dev 7	Dev 6	Dev 5	Dev 4	Dev 3	Dev 2	Dev 1
Charge totale prédite	113 986	115 614	114 812	115 367	112 982	114 420	114 234	116 706

Charge incrémentale négative augmentée de 40k€								
	Dev 8	Dev 7	Dev 6	Dev 5	Dev 4	Dev 3	Dev 2	Dev 1
Charge totale prédite	86 504	109 902	110 558	104 906	106 541	107 835	105 025	105 406

TABLE 3.21 : Sensibilité de la prédiction de la charge d'un sinistre en fonction d'une modification d'un de ces *inputs*

Sur un sinistre à l'amiable, une à une les charges incrémentales positives ou négatives a été augmentée de 40 kUM. Le montant de charge ultime positive ou négative est ensuite observé sur la table ci-dessus (3.21). Les résultats nous montrent une légère variation du montant de charge finale prédite. Cette variation n'est pas significative pour la plupart des variations. Certaines valeurs sont cependant impactantes comme l'année de développement 8 des charges incrémentales négatives. Cependant, ce montant ne change pas la constatation générale : les montants de charges incrémentales positives ou négatives (hors développement 0) n'ont que peu d'impact sur la prédiction d'un sinistre.

La variable de statut du sinistre (ouvert ou fermé à chaque développement connu), qui subit un traitement similaire à la charge incrémentale, a une particularité. Son format double est composé d'un vecteur de 0 (resp. 1) pour signifier que le sinistre est fermé (resp. ouvert) pour chaque développement connu du sinistre. Il est combiné à son vecteur complémentaire par inversion des 0 et des 1. Ce format double combiné à des valeurs simples (0 ou 1) permet au LSTM puis au modèle de bien capter l'information. Cette affirmation se retrouve dans l'étude de sensibilité de la prédiction d'un sinistre en fonction du statut, qui indique un impact clair.

Précédemment, une distinction de comportement a été observée entre les sinistres litigieux et les sinistres à l'amiable. Cependant, la faible sensibilité aux changements de montants de charges incrémentales est similaire, peu importe la catégorie.

La cause potentielle provient des données. En effet, les montants de charges incrémentales sont en majorités nuls, car à chaque développement une des charges incrémentales positives ou négatives est nulle par construction. Même sans cela, la majorité des charges incrémentales sont nulles car il existe de nombreux 0 dans la charge incrémentale de chaque sinistre de la base.

Un réseau de neurones n'apprend pas de poids liés aux *inputs* ayant pour valeur 0, mais apprend les autres poids, permettant ainsi d'obtenir une moyenne des sinistres sans cette variable. La sur-représentation des valeurs 0 dans les montants de charges implique un plus faible apprentissage des poids liés aux valeurs ayant un grand nombre de modalités à 0. De plus, la *loss* étant calculée sur de nombreux développements, le modèle aura tendance à prédire une moyenne sans prendre en compte les variables de charges. Une observation des poids finaux du réseau de neurones multicouches confirme ce fait. Le biais de ce réseau est très haut tandis que les variations de poids sont plus faibles causant la non-influence des variations de montants de la charge incrémentale.

Une idée d'amélioration par création de variables serait de garder la charge à l'année de développement

0 (significative) et de ne prendre en compte que la somme des charges incrémentales connues (hors développement 0). Cela permettrait de réduire la taille de l'*input*, et notamment le nombre de 0 en *input* en supprimant les 2 vecteurs de charges incrémentales. Ces changements de variables permettraient aussi de simplifier le modèle et donc sa compréhension et son explicabilité en supprimant le LSTM encoder de l'information de la charge.

De même, malgré son fonctionnement actuel, une simplification similaire du vecteur de statut du sinistre pourrait être introduite. Une idée serait d'utiliser une variable pour le nombre d'années que le sinistre est fermé (0 pour un sinistre encore ouvert) et une variable pour le nombre d'années que le sinistre est ouvert (0 pour un sinistre fermé), afin de garder cette complémentarité efficace. Ce changement théorique aurait pour conséquence la suppression totale du LSTM encoder facilitant ainsi l'interprétabilité des résultats du modèle.

Une autre problématique majeure dans cet exercice de provisionnement est d'arriver à faire apprendre (et prédire) au modèle des montants de charges d'ordre de grandeur totalement différents. En effet, comme l'a montré ce paragraphe, l'impact de la charge est trop souvent nul. Cela provient aussi de l'ordre de grandeur de la prédiction de cette charge. En effet, dans le tableau de la distribution de la charge connue (3.1), de nombreux sinistres ont une charge connue très importante. Les poids du modèle sont donc appris aussi bien avec des sinistres inférieurs à 1 k(UM) qu'avec des sinistres à 1 M(UM). Ce qui crée un effet d'uniformisation de la prédiction pour les nombreux sinistres attritionnels.

Deux améliorations peuvent permettre de contourner l'ordre de grandeur de la charge : l'ajout d'une variable explicative des sinistres dit "graves" à l'aide d'une classification, ou une classification au préalable suivie d'un modèle pour chaque classe de sinistres, dans le cas où la première méthode échoue.

La volatilité du modèle due à l'initialisation

Une fois la base de données, les variables et les hyper paramètres fixés, un réseau de neurones ont plusieurs facteurs de volatilité autre que la distribution de sortie du modèle. Le premier, par ordre d'exécution, est la séparation aléatoire entre la base d'entraînement et la base de validation. Cette séparation est aléatoire selon une loi uniforme. Le logiciel d'implémentation R (R CORE TEAM, 2022) et python (VAN ROSSUM et DRAKE, 2009) permettent de fixer la graine à la base de cet aléa.

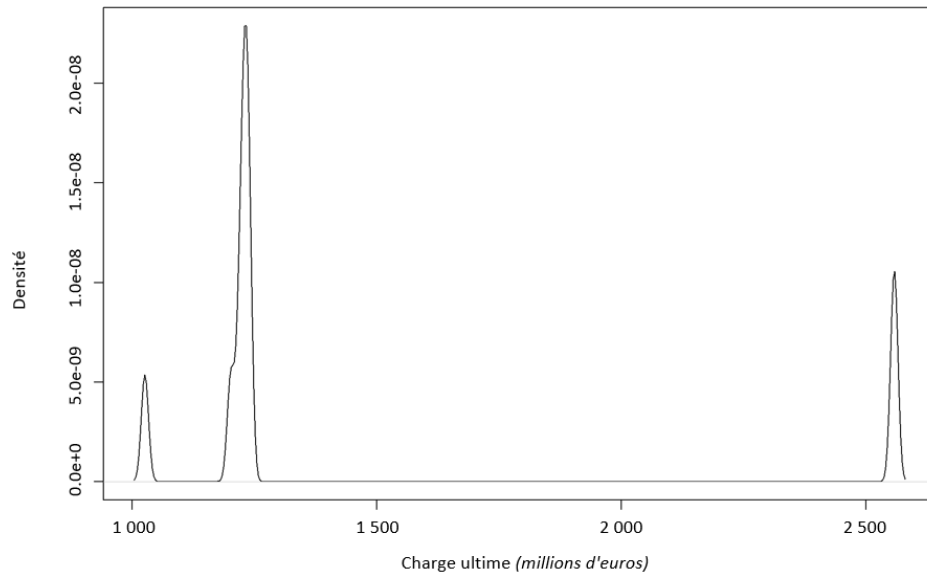
Le second est l'initialisation des poids. Les poids sont fixés suivant la méthode aléatoire déclarée dans le modèle (glorot uniforme). Cependant, afin d'obtenir une reproductibilité totale, il est aussi possible de les fixer.

Le troisième facteur de volatilité est dû au mélange dans le modèle. En effet, lors de la phase d'entraînement, avant chaque *epoch*, le modèle mélange l'ensemble des lignes de l'échantillon d'apprentissage. Ce mélange est également fixable afin d'obtenir une reproductibilité totale.

Parmi ces 3 facteurs, le premier et le troisième ont un très faible impact sur le résultat final quitte à augmenter le nombre d'*epochs* (l'impact est inférieur à 10^6 sur les PSAPs). Cependant, l'initialisation a un impact fort. Observons ceci dans un tableau.

Sur le graphique 3.10, plusieurs valeurs de la charge ultimes prédites très différentes existent. Cela indique que plusieurs minima locaux sont atteints à la fin de l'entraînement (afin d'arriver à une *loss* stable synonyme de l'obtention d'un minimum local, le nombre d'*epochs* a pu être augmenté de 50 à 80 *epochs*). En changeant les poids initiaux, aucun modèle n'améliore concrètement le modèle étudié en termes de *loss*. En comparant les résultats, un modèle a une *loss* plus forte et obtient un résultat

plus réaliste (1 026 millions). Ce cas est purement aléatoire, car l'apprentissage n'était toujours pas stable au bout de 50 *epoch*. Les autres modèles bloquant à une *loss* non minimal sont imprévisibles. Ils peuvent obtenir un résultat similaire au modèle ou un résultat totalement anormal (2 558 millions d'UM de charge ultime prédite par exemple).



Loss	0,92	0,61	0,57	0,52	0,73	0,74	0,57	0,54	0,54	0,73
Validation Loss	0,94	0,64	0,59	0,55	0,76	0,76	0,60	0,57	0,57	0,75
Charge ultime prédite (M€)	2 558	1 026	1 233	1 227	2 559	1 201	1 236	1 227	1 238	1 219

FIGURE 3.10 : Distribution des résultats selon différentes initialisations des poids du modèle et valeurs de *loss* (base d'entraînement et de validation) pour chaque prédiction

L'apprentissage est donc bénéfique à la prédiction, ce qui rassure. Un critère pour valider l'apprentissage du modèle dans ce cas est l'obtention d'une *loss* sur l'échantillon de validation inférieure à 0.6. Une fois cette *loss* minimale atteinte le modèle prédit des charges ultimes, très similaires de l'ordre de 1 200 millions d'UM. Cette prédiction n'est pas du bon ordre de grandeur, mais elle est stable par initialisation.

Le LSTM décoder

Contrairement au modèle de Kuo, en fixant un ordre de prédiction, cela crée un biais dans le modèle. En effet, lors de la phase d'apprentissage, le modèle n'apprend pas des informations connues grâce à un masque comme expliqué précédemment. Cependant, lors de la prédiction, l'entrée d'une cellule LSTM va utiliser la sortie de la cellule précédente et donc potentiellement la prédiction d'une année connue, pour prédire l'année suivante. Ce n'est pas un problème créant une grande erreur, sinon la prédiction de l'année 2003 (la plus ancienne) en serait grandement impactée. Cependant, ce choix de modèle pourrait être contesté. Une solution simple est la démultiplication du modèle, un pour chaque année de développement à prédire. La taille de l'*input* étant fixée, l'information en sortie de LSTM sera toujours inconnue et l'utilisation d'un masque dans la phase d'entraînement n'est plus nécessaire. Ce changement n'a pas été implémenté.

3.5 Bilan de l'application du modèle

L'objectif de ce chapitre est d'analyser l'application du modèle de Kuo (modifié) sur une base de données réelles. Après une étude descriptive de la base et les modifications expliquées, la sélection de variables et d'hyper paramètres n'ont pas suffi à obtenir un modèle entraîné prédisant une charge ultime crédible. Cependant, cette différence a permis de questionner en profondeur les différentes causes de cet écart.

Plusieurs limites du modèle sur cette base de données ont été mises en avant. La variable de litige d'un sinistre est une variable très significative, mais son comportement (en nombre et coût) a changé durant l'historique ce qui augmente la prédiction finale. La différence d'ordre de grandeur des montants de charges ne permet pas au modèle d'apprendre les faibles variations notamment à cause des fonctions bornées dans les LSTM. L'initialisation des poids peut faire varier le minimum local obtenu. Cependant, l'observation de la *loss* permet de déterminer si le minimum global est obtenu (*loss* > 0.6). Sur les *loss* inférieures à 0.6, le modèle apprend et prédit toujours une charge ultime proche de 1200 M(UM). L'apprentissage est donc stable, peu importe l'initialisation.

Certains choix de modèles comme la séparation entre les charges incrémentales positives et négatives ou la loi de sorties ont aussi été challengés. Des idées d'amélioration pour la création d'un modèle plus performant ont été proposées pour chaque limite. Cependant, la volatilité de la base utilisée et la différence d'ordre de grandeur restent les deux facteurs les plus limitants du modèle, une bonne classification des sinistres n'ayant pas été trouvée. L'application de ce modèle sur une base plus stable telle qu'une base Auto ou une base ayant des données plus précises et pertinentes sur chaque sinistre permettrait d'exploiter ce modèle.

Un autre facteur limitant est le temps de sélection des hyper paramètres. Afin d'obtenir un bon modèle, sans intuition au préalable, ce temps est de quelques jours. Cependant, une base de données beaucoup plus massive nécessiterait d'effectuer la sélection d'hyper paramètres sur un échantillon bien choisi de la base, ce qui n'a pas été nécessaire dans le cas de la base de responsabilité civile médicale.

Conclusion

Le cycle de production inversé d'un produit d'assurance impose aux assureurs d'analyser et de prédire ses paiements futurs. Les provisions servent à garantir que l'assureur est en capacité de payer les sinistres pour lesquels il s'est engagé. Diverses méthodes de provisionnement sont utilisées par les assureurs (Chain Ladder, Bornhuetter-Ferguson, Mack Bootstrap,...). Historiquement, ces méthodes sont uniquement basées sur des données agrégées. De nos jours, avec l'augmentation de la quantité de données disponibles et l'amélioration des capacités de traitements, de nouvelles méthodes de provisionnement ont été créées à partir des techniques de *Deep Learning*. Ces méthodes permettent l'utilisation de variables disponibles à la granularité d'un sinistre. La prédiction de la provision peut ainsi être effectuée sur chaque classe de sinistre ayant un comportement similaire.

Dans la suite d'autres travaux réalisés sur ces nouvelles méthodes dite ligne à ligne, ce mémoire a eu pour objectif d'expliquer et d'implémenter un modèle de provisionnement ligne à ligne utilisant les réseaux de neurones profonds. Les données ligne à ligne dispose de montants de charges années après années. Afin de traiter ces *inputs* temporels, la méthodologie utilisant un LSTM encoder et décoder a été retenue.

Une autre contrainte était nécessaire au modèle : l'obtention de la distribution de la charge ultime prédite. En effet, plusieurs normes encadrent les travaux de provisionnement des assureurs (Solvabilité 2, IFRS 17). Ces normes imposent à l'assureur d'être en mesure de calculer des quantiles de la charge ultime. Les modèles ligne à ligne se doivent d'être en capacité de prédire une distribution, permettant le calcul des quantiles désirés. Les recherches d'un tel modèle ont menées à l'utilisation d'un réseau de neurones bayésien et d'un bloc de distribution final, correspondant à la structure du modèle de KUO, 2020. La distribution du modèle de Kuo est un mélange d'une distribution log-normale décalée en 0 et d'une distribution fixe à 0. Afin d'utiliser la loi log-normale, le modèle effectue une séparation de la charge incrémentale entre la charge incrémentale positive et négative.

Une implémentation de ce modèle, légèrement modifié pour éviter une prédiction constante sur les années de développement, a été effectuée. Les résultats obtenus sont encore loin de l'ordre de grandeur des prévisions réalisées par Chain-Ladder ou par des experts provisionnement. L'apport de ce mémoire réside donc dans les limites et les pistes d'améliorations trouvées.

Les premières limites proviennent de la base de données. La base de responsabilité civile médicale est une base ayant une charge incrémentale très volatile avec des sinistres très graves. La prédiction des faibles montants de charges a été confondue par le modèle devant les montants des sinistres graves. De plus, le comportement d'un sinistre litigieux (variable la plus pertinente de notre modèle), a des changements de comportements importants entre les années récentes et les années les plus anciennes. Or le nombre de sinistres litigieux a été multiplié par 10 entre 2003 et 2019, ce qui augmente la prédiction de manière très significative. Contrairement à d'autres bases, cette base ne disposait pas de caractéristiques très fines comme le type de médecin en faute, la partie du corps touchée, l'âge de la personne subissant le sinistre.

D'autres limites proviennent du modèle. La séparation de la charge incrémentale positive et négative crée des prédictions incohérentes à l'échelle d'un sinistre (charge cumulée négative et prédiction d'une charge incrémentale positive et négative simultanément). Le choix d'un LSTM utilisant des fonctions d'activations bornées n'est pas adapté à des variations de charge incrémentale si importante et comportant de nombreuses valeurs à 0. Les montants de charges incrémentales, autre que l'année de développement 0, n'ont que peu d'impact sur la prédiction.

Plusieurs pistes d'améliorations ont été évoquées. La création d'un modèle pour chaque année de développement permettrait de réduire les problèmes liés à la taille variable de *l'inputs*. La séparation des sinistres graves et attritionnels par une classification supervisée est nécessaire afin d'uniformiser l'ordre de grandeur des prédictions. Les variables incrémentales de charge n'étant pas significatives actuellement, l'ajout des variables de la charge connue et du dernier montant de charge incrémentale connu simplifierait considérablement le modèle, mais nécessite d'être comparée sur une base de sinistres de même ordre de grandeur.

Ce mémoire, à travers son explication détaillée des différents modules composant un réseau de neurones et ses pistes d'améliorations, vise à accompagner un éventuel successeur en l'aidant à identifier les points clés pour son propre modèle.

Bibliographie

- ALLAIRE, J. et TANG, Y. (2022). tensorflow: R Interface to 'TensorFlow'. R package version 2.9.0. URL : <https://CRAN.R-project.org/package=tensorflow>.
- BISHOP, C. M. (1994). Mixture Density Networks. (Unpublished) Aston University, Birmingham. URL : https://publications.aston.ac.uk/id/eprint/373/1/NCRG_94_004.pdf.
- CHARTREL, A. (2022). Classification a priori de sinistres pour le provisionnement individuel. Mém. de mast. ENSAE, IP Paris.
- CODE DES ASSURANCES (2002). Loi About. URL : <https://www.legifrance.gouv.fr/jorf/id/JORFTEXT000000234122>.
- GLOROT, X. et BENGIO, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Sous la dir. de TEH, Y. W. et TITTERINGTON, M. T. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy : PMLR.
- GOMA, T.-S. (2022). Apport des réseaux neuronaux au provisionnement IARD : Application d'architectures récurrentes à la prédiction d'évolution temporelle de montants. Mém. de mast. ENSAE, IP Paris.
- HE, K., ZHANG, X., REN, S. et SUN, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. URL : <https://arxiv.org/abs/1502.01852>.
- KOEHRSEN, W. (2018). Neural Network Embeddings Explained. URL : <https://towardsdatascience.com/neural-network-embeddings-explained-4d028e6f0526>.
- KUO, K. (2019). DeepTriangle: A Deep Learning Approach to Loss Reserving. *Risks* 7.3.
- KUO, K. (2020). Individual Claims Forecasting with Bayesian Mixture Density Networks. URL : <https://arxiv.org/pdf/2003.02453.pdf>.
- LITTLE, RODERICK J. A., R. A. (2002). Statistical analysis with missing data (2nd ed.)
- MACK, T. (1993). Distribution-free Calculation of the Standard Error of Chain Ladder Reserve Estimates. *ASTIN Bulletin* 23.2, 213–225.
- NORBERG, R. (1993). Prediction of Outstanding Liabilities in Non-Life Insurance1. *ASTIN Bulletin* 23.1, p. 95-115.
- PATACCHIOLA, M. (2021). Evidence, KL-divergence, and ELBO. URL : <https://mpatocchiola.github.io/blog/2021/01/25/intro-variational-inference.html>.
- PRIMEL, D. (2021). Comparaison de différents modèles de provisionnement ligne à ligne et discussions dans le cadre d'une application en non-vie. Mém. de mast. Univ. Paris-Dauphine, Univ. PSL.
- R CORE TEAM (2022). R: A Language and Environment for Statistical Computing. Vienna, Austria : R Foundation for Statistical Computing. URL : <https://www.R-project.org/>.
- VAN ROSSUM, G. et DRAKE, F. L. (2009). Python 3 Reference Manual. Scotts Valley, CA : CreateSpace.

Annexe A

Étude du comportement de la variable litige

A.1 Étude descriptive détaillée : la charge moyenne au développement 0 selon la variable litige

Observons la charge moyenne des sinistres litigieux ou non à l'année de développement 0.

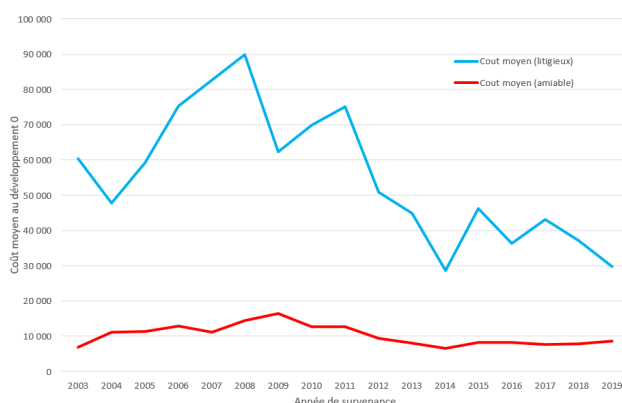


FIGURE A.1 : Courbes des évolutions temporelles de la charge moyenne connue des sinistres litigieux ou à l'amiable au développement 0

Sur la figure (A.1), la charge moyenne des sinistres litigieux et la charge moyenne des sinistres à l'amiable au développement 0 de chaque survenance sont observées. La charge moyenne au développement 0 des sinistres à l'amiable n'a que peu évolué au cours du temps contrairement à la charge moyenne des sinistres litigieux au développement 0.

Décrivons la charge moyenne des sinistres litigieux au développement 0.

Sur les années 2003 et 2004, la charge moyenne reste faible (environ 60 M€). Sachant que la charge moyenne connue en vision 2019 sur ces sinistre est de 150 m€, une forte hausse sera apprise par le modèle.

La charge moyenne augmente ensuite sur les années 2005 à 2011 (environ 75M€). Cela peut être dû à une plus grande prudence de la part des gestionnaires de sinistres ou à des paiements plus importants. La procédure de délibération des sinistres litigieux est lente ce qui rend peu probable des paiements importants sur l'année de survenance du sinistre. Sur les années 2005 à 2008, la charge moyenne connue est plus grande (de l'ordre de 100m€). Le modèle devrait apprendre une hausse. Sur les années 2009 à 2011, la charge moyenne connue est similaire à celle prédite au développement 0. Pour les années 2012 à 2019, la charge moyenne au développement 0 est plus faible (environ 40 m€). La charge moyenne connue est similaire, ce qui est cohérent car il y a peu d'années connues. Le modèle ne devrait pas prédire de grande variation.

La conclusion de ce graphique est que la charge moyenne d'un sinistre litigieux au développement 0 à grandement varié entre 30 et 90 m€. Cette information permet de valider un changement de comportement de ces sinistres au cours du temps. En associant ce graphique aux montants de charge moyenne connue, de grands écarts indiquant une prédiction haussière du modèle sont observés. Il faut à présent vérifier à quel développement ces écarts se creusent. Si les écarts interviennent dans les dernières années de développements inconnues sur les branches courtes, alors le modèle doit bien prédire une grande hausse sur les derniers développements. Dans le cas où les écarts sont présents sur les premiers développements déjà connus des années récentes, la prédiction sera a priori erronée.

A.2 Étude descriptive détaillée : la charge cumulée des sinistres litigieux par développement

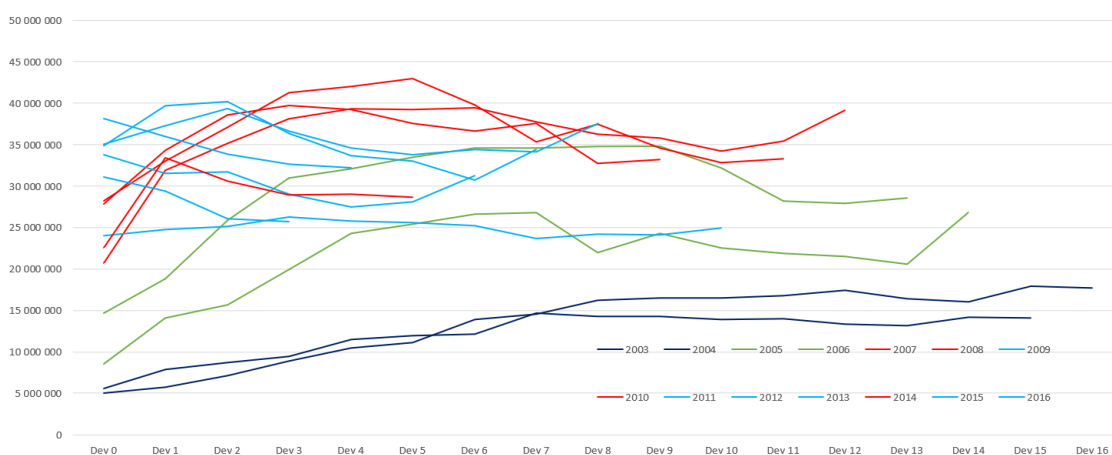


FIGURE A.2 : Courbe de l'évolution la charge cumulée connue des sinistres litigieux pour chaque année de développement

Sur le graphique (A.2), quatre comportements distincts de la charge cumulée des sinistres litigieux sont affichés. En bleu foncé, les années 2003 et 2004. La charge des sinistres litigieux a un comportement linéaire à la hausse.

En vert, les années de transition 2005 et 2006. La charge cumulée subit une forte hausse dans les 5 premières années, puis elle stagne ou décroît légèrement.

En rouge, les années 2007, 2008, 2010 et 2014 ont un comportement similaire aux années de transition. La différence réside dans le point de départ qui a significativement augmenté (10M€ à 25M€). Les

A.3. ÉTUDE DESCRIPTIVE DÉTAILLÉE : LA PROPORTION DE SINISTRE LITIGIEUX PAR SURVENANCE

gestionnaires de sinistres ayant plus de visibilité ont probablement ajusté la charge des nouveaux sinistres.

En bleu clair, les années les plus récentes (2009, 2011 à 2013, 2015 et 2016) sont globalement stable. Leur point de départ est encore plus haut (environ 30 à 35 M€).

Cette distinction permet de voir l'amélioration de l'estimation des gestionnaires de sinistres. En reprenant la conclusion du graphique (A.1), l'écart entre la charge moyenne au développement 0 et la charge moyenne connue se creuse dans les 5 premières années. La prédiction doit ainsi être relativement stable au cours du temps.

L'étude précédente se base sur une vision agrégée qui n'est pas en accord avec le modèle qui se base sur des données ligne à ligne. En observant les données ligne à ligne, il faut prendre un troisième critère en compte : la proportion des sinistres litigieux.

A.3 Étude descriptive détaillée : la proportion de sinistre litigieux par survenance

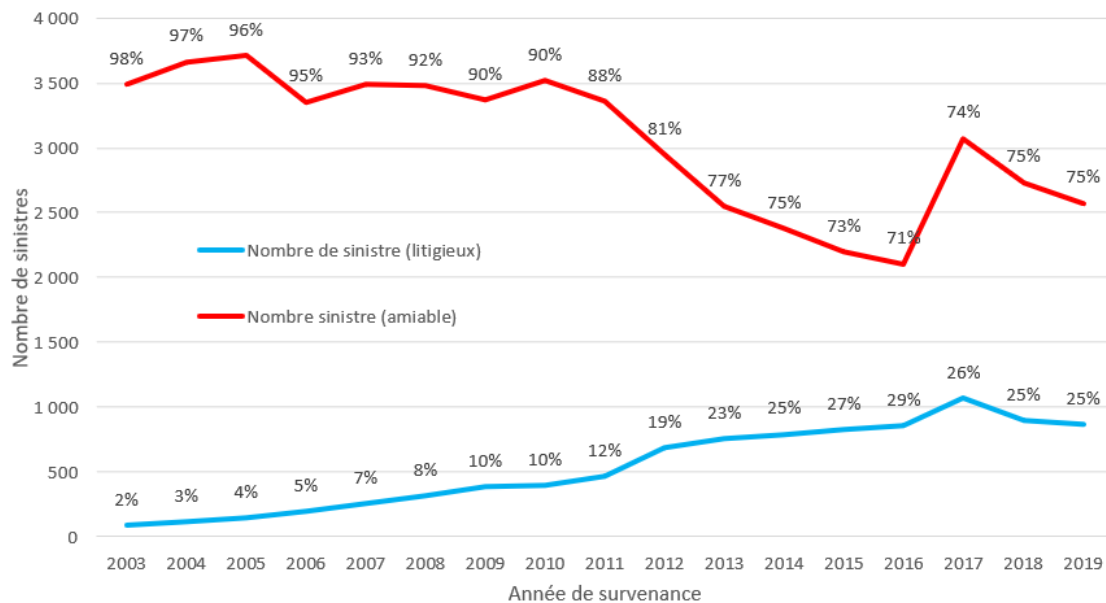


FIGURE A.3 : Courbes des évolutions du nombre de sinistres selon la variable litige par année de survenance

La figure (A.3) montre l'évolution du nombre de sinistres suivant la variable de litige. Le nombre de sinistres reste constant, cependant la proportion de litigieux change. Sur les années les plus anciennes, cette proportion est très faible (de l'ordre de 2%) et sur les années les plus récentes la proportion est très forte de l'ordre de (25%). Un augmentation de cette échelle signifie soit un changement de souscription prenant en compte des acteurs plus susceptibles d'aller contester les faits devant un juge, soit un changement de comportement de l'assuré lié à une démocratisation du recours à la justice. Même si le coût moyen a baissé, l'augmentation du nombre de sinistre ajouté au fait que les sinistres litigieux anciens ont un comportement haussier sur les 5 premières années de développement fait

apprendre une hausse au modèle. Cette hausse est démultipliée par le nombre de sinistre litigieux, ce qui fait exploser la prédiction.