Mémoire de projet de recherche

Interprétation des réseaux de neuronnes profonds

- Réalisé par :

  EL ROBRINI Medina

  RAMOUL Rayan Samy

../../2020

Table des matières

Table des figures

Liste des tableaux

Introduction

L'avancée de l'intelligence artificielle a permis de révolutionner le monde de la médecine de précision. Cette dernière également appelée médecine personnalisée, se focalise sur l'analyse des caractéristiques moléculaires et génétiques pour proposer un traitement adapté au patient. Dans le but de mieux comprendre la maladie de la leucémie, des données sous forme d'expression de gènes sont utilisées afin d'essayer de trouver des sous-classes de cette maladie. Pour ce faire, nous utilisons l'apprentissage profond qui continue à faire ses preuves dans tout ce qui est traitement automatique du langage naturel, la bio-informatique, etc. Il serait donc intéressant de voir de plus près l'application du *deep learning* à la médecine de prédiction.

Problématique

Le développement de modèles de *Deep learning* permet d'améliorer grandement la performance des prédictions. Néanmoins, ces modèles soulèvent de nombreuses questions quant à leur interprétabilité. En effet, la précision des modèles ne suffit plus, aujourd'hui, la capacité à expliquer la prise de décision de ces algorithmes est tout aussi importante. C'est même une exigence minimale pour certains processus automatisés. Cette capacité à expliquer les modèles permet d'améliorer la compréhension des résultats mais aussi d'apporter de la crédibilité surtout dans le domaine de la médecine de précision où la rigueur et la précision sont de mises.

Pour répondre à ce besoin d'interprétation, nous utilisons une bibliothèque nommée *iNNvestigate* qui contient plusieurs méthodes d'analyse nous permettant de bien comprendre notre modèle. Cependant, cette bibliothèque est particulièrement adaptée aux entrées sous forme d'images, un second problème est de devoir la convenir aux données vectorielles relatives aux expression de gènes.

Chapitre 1

Etat de l'art

1.1 Apprentissage automatique

L'apprentissage automatique permet à un système d'apprendre à partir des données et non à l'aide d'une programmation explicite. Il en résulte un modèle. Il y a plusieurs types d'apprentissage :

- **L'apprentissage supervisé :** l'apprentissage se fait à l'aide des données d'entrée et de sortie étiquetées. Le but étant de trouver une fonction d'approximation qui lie les entrées aux sorties. On distingue 2 types d'apprentissage supervisé majoritaires :
  — La régression : Ayant pour but de prédire une donnée quantitative continue.
  — La classification : Où le modèle doit prédire une donnée discrète.
- **L'apprentissage non supervisé (Clustering) :** les données sont non étiquetées, de sorte que l'algorithme d'apprentissage puisse déterminer à lui seul des points communs parmis ses données d'entrée.
- **L'apprentissage semi-supervisé :** utilise un ensemble de données étiquetées et non étiquetées. Ce type d'apprentissage est assez intéressant étant donné que l'étiquetage des données peut s'avérer difficile.
- **L'apprentissage par renforcement :** se base sur un cycle d'expérience / récompense et améliore les performances à chaque itération, il a comme particularité de pouvoir être utilisé sans appel à des données humaines.

Il existe plusieurs méthodologies d'apprentissage automatique, on cite : les réseaux de neurones, les arbres de décisions, les machines à support de vecteurs, etc.
Dans le cadre de notre étude, nous sommes dans le cas d'un apprentissage supervisé et nous nous intéressons particulièrement à la méthode des réseaux de neurones.

1.1.1 Les réseaux de neurones

Les réseaux de neurones ont été développés comme un modèle mathématique générique afin de modéliser les neurones biologiques. Ils comportent un certain nombre d'éléments de traitement d'information appelés neurones [5].

Chaque neurone reçoit des entrées et fournit une sortie, grâce à différentes caractéristiques :
— Des poids accordés à chacune des entrées, permettant d'en modifier l'importance de certaines par rapport aux autres. Ce sont des paramètres à estimer lors de la procédure d'apprentissage.

— Une fonction d'agrégation, qui permet de calculer une unique valeur à partir du produit scalaire du vecteur des entrées et des poids correspondants.
— Un seuil (ou biais) θ.
— Une fonction d'activation, qui associe à chaque valeur agrégée une unique valeur de sortie dépendant du seuil [6]. La sortie du neurone aura la formule suivante :

$$y = f(\sum_{i=1}^{n}(x_i * w_i) + \theta)$$

Notons que la fonction f représente la fonction d'activation et la somme en argument représente la fonction d'agrégation décrite plus haut. On distingue plusieurs types de fonction d'activation, parmi elles :

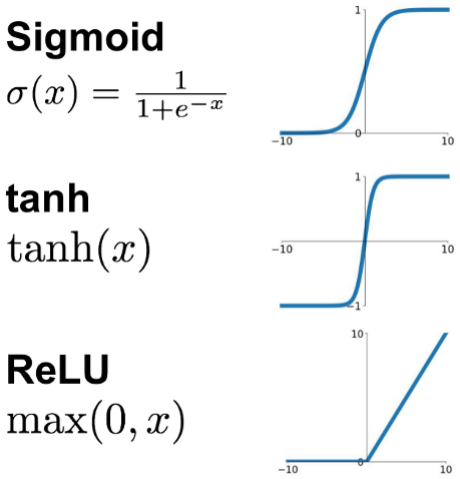


FIGURE 1.1: les fonctions d'activation

Ces différents éléments représentant un neurone sont illustrés dans la figure suivante :

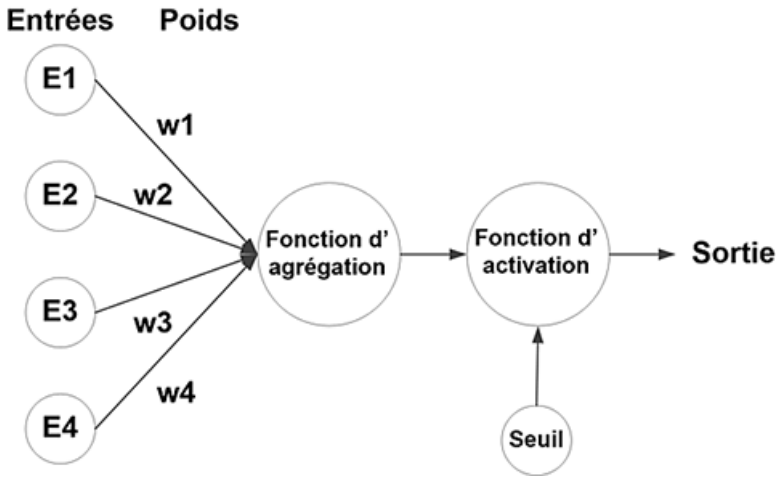


FIGURE 1.2: Neurone artificiel

Cependant, pour apprendre des fonctions plus complexes et pallier au problème des données qui ne sont pas toujours linéairement séparables, le perceptron multicouche ou Multi Layer Percep-

tron (MLP) est apparu. Ce dernier n'est autre qu'un ensemble de couches contenant des neurones artificiels comme le montre la figure suivante :

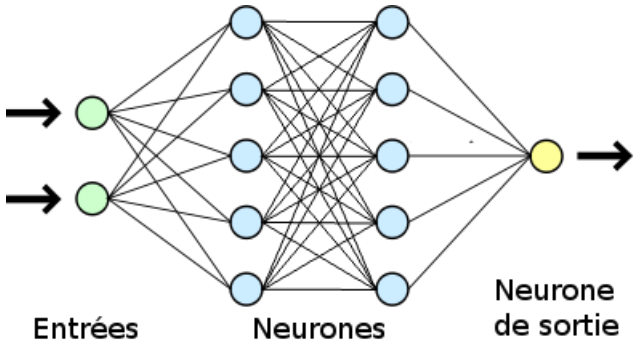


FIGURE 1.3: Perceptron multicouche

Nous remarquons une couche d'entrée contenant autant de neurones qu'il y a de variables dans le problème qu'on veut résoudre, des couches cachées et une couche de sortie qui fournit la réponse du système. Chaque neurone d'une couche cachée est connecté en entrée à chacun des neurones de la couche précédente et en sortie à chaque neurone de la couche suivante. Il est a noter que les fonctions d'activation peuvent être différentes d'une couche à une autre mais les neurones d'une couche doivent tous avoir la même fonction d'activation.

L'objectif des réseaux de neurones est d'ajuster les poids de telle sorte à minimiser la fonction d'erreur (*Cost function*). Cette dernière a plusieurs types par exemple l'erreur quadratique a la formule suivante :

$$\frac{1}{n}\sum_{i=1}^{n}(prediction_i - y_i)^2$$

- $prediction_i$: la prédiction du modèle
- y_i : la véritable sortie du modèle

Ainsi, le problème revient donc à trouver le minimum de cette fonction d'erreur. Au départ, les poids sont générés de manière aléatoire. Ils sont ensuite ajustés au fur et à mesure grâce à la descente du gradient. La modification des poids est propagée de la couche de sortie jusqu'à la couche d'entrée et c'est ce qu'on appelle la rétro-propagation du gradient. Il en résulte *un modèle* autrement dit une matrice de poids et de biais.

1.1.1.1 Les limitations des réseaux de neurones

Bien que l'apprentissage automatique a permis d'apprendre plusieurs concepts il reste limité quand il s'agit des problèmes plus complexes tel que la reconnaissance des images. Le passage à l'apprentissage profond a eu lieu suite aux limitations suivantes :

- L'aprentissage automatique nécessite des données structurées tandis que le système du deep learning est capable d'identifier lui-même les caractéristiques discriminantes sans avoir besoin d'une catégorisation préalable.
- L'apprentissage automatique utilise peu de couches contrairement à l'apprentissage profond qui à travers ses multiples couches cachées a l'avantage de réduire le taux d'erreur surtout pour des applications qui nécessitent plusieurs filtres tels que de la reconnaissance d'images par exemple.

- les algorithmes de type Machine Learning produisent toujours un résultat numérique, comme une classification ou un score. Les résultats du Deep Learning, eux, peuvent être n'importe quoi, y compris du texte en langage naturel pour sous-titrer une image par exemple.

  L'apprentissage profond a besoin d'une grande masse de données et donc nécessite un temps de calcul important, c'est pourquoi la vraie révolution est liée aux technologies actuelles notamment la parallélisation des calculs comme le montre l'exemple de Nvidia Cuda et la quantité de données accessibles.

1.1.2 Les réseaux de neurones profonds

Les réseaux de neurones profonds sont des réseaux de neurones qui contiennent au moins deux couches cachées, leur architecture est donc plus complexe que le perceptron multicouche vu plus haut. Ils sont chargés d'approximer des fonctions qui permettront de faire de la prédiction.

Le passage aux réseaux de neurones profonds s'est révelé bien utile pour la résolution des problèmes complexes nécessitant un grand volume de données. L'avancée technologiques des ordinateurs niveau hardware notamment les GPU (Graphics Processing Units) ont fourni une meilleure puissance de calcul conséquente et suffisante à l'apprentissage de ce type de modèle. Des processeurs spécifiques ont alors été développés, adaptés aux différentes phases de calcul des algorithmes.

Trois grandes familles de réseaux d'apprentissage profond ont été développés on distingue les réseaux de neurones convolutifs (CNN) pour l'analyse d'images, les modèles séquentiels (LSTM par exemple) utiles lorsqu'il y a une dimension temporelle ainsi que les auto-encodeur.

1.1.3 Intérpretation des réseaux de neurones profonds

1.1.3.1 Défintion

L'interprétabilité est la capacité d'expliquer ou de présenter des informations dans des termes humainement compréhensibles. Par exemple, cela peut se faire avec une représentation visuelle de l'importance des variables.

Aujourd'hui la tendance est que l'explicabilité d'un modèle est une métrique primordiale au même titre que la performance. On peut distinguer trois niveaux d'interprétabilité dans les algorithmes d'apprentissage automatique [9] :

- **Haute interprétabilité** : ce niveau inclut les algorithmes de régression, les arbres de décision et les règles de classification traditionnelle.
- **Interprétabilité moyenne** : ce niveau inclut des algorithmes plus avancés tels que les modèles graphiques.
- **Faible interprétabilité** : ce niveau inclut des techniques avancées d'apprentissage automatique telles que les SVM, les méthodes d'apprentissage ensembliste et d'apprentissage profond. Au mieux, ils fournissent des informations sur l'importance des variables pour l'explicabilité du modèle.

Ces différents algorithmes d'apprentissage automatique selon leur degré d'interprétabilité sont présentés dans la figure ci-dessous

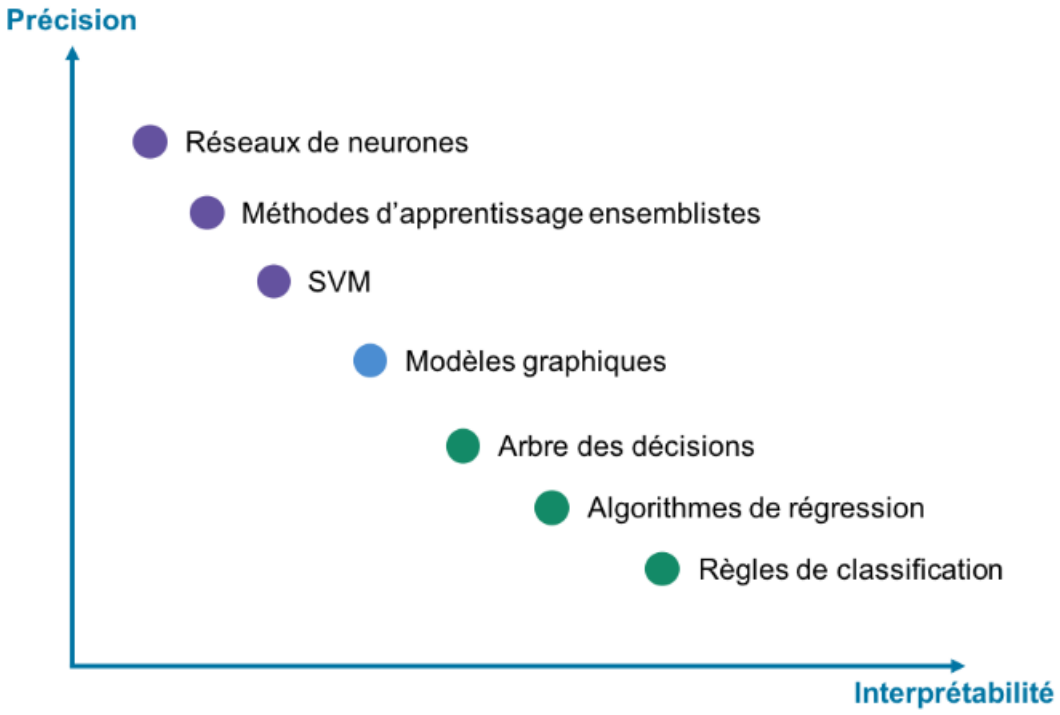


Image adaptée à partir de Dam et al. 2018 [2]

Nous pouvons remarquer que les réseaux de neurones, utilisés dans notre travail, ont une très faible interprétabilité par rapport aux autres algorithmes.

Selon CHAKRABORTY et all [10], On a plusieurs volé d'interprétabilité :

1.1.3.2 Interprétation des modèles par leur transparence (*Model transparency*)

L'étude de la transparence des modèles s'avère si importante qu'elle ne se limite pas qu'à expliquer les modèles seulement mais peut aussi nous aider à les améliorer. Ce type d'interprétation est axé sur les détails techniques, elle est définie par ces trois éléments :

- Similarité : (Simultability) : la capacité de l'être humain à reproduire les mêmes étapes utilisées par le modèle et prédire la même sortie. Cela prouve que les changements effectués au niveau du modèle ont bien été compris par l'humain ce qui le rend interprétable.
- Decomposition (Decomposability) : Le faite de pouvoir justifier le choix des paramètres du model.
- Transparence de l'algorithme (Algorithmic transparency) : Le faite de pouvoir expliquer de manière simple et précise le process de l'algorithme d'apprentissage.

1.1.3.3 Interprétation des modèles par leur fonctionnement (*Model functionality*)

peut se faire grace à :

- Description textuelle : La description du modèle se fait avec du texte et c'est un tout autre modèle qui se charge de cette tache. Par exemple Krening et al [11] ont entraîné un modèle basé sur l'apprentissage par renforcements. Ils ont ensuite construit un autre modèle qui sert à faire le lien entre l'état du modèle précèdent et une explication textuelle.

- Visualisation : un autre moyen assez parlant pour expliquer un modèle est de visualiser ses paramètres. l'une des approches pour ce faire s'appelle T-SNE (t-Distributed Stochastic Neighbourhood Embedding) [12]. Il s'agit d'une méthode permettant de représenter un ensemble de points d'un espace à grande dimension dans un espace de deux ou trois dimensions, le principe est similaire à l'ACP (analyse en composante principale).
- Explication local : au lieu d'expliquer tout le modèle globalement, on explique les changements locaux causés par une entrée bien spécifique. Parmi les méthodes d'interprétation les plus connues on retrouve la méthode LIME [13]. Cette dernière est indépendante du modèle, ce qui signifie qu'elle peut être appliquée à n'importe quel algorithme d'apprentissage automatique. La technique LIME tente de comprendre le modèle en perturbant l'entrée des échantillons de données et voir comment les prédictions changent.
  La sortie de LIME reflète la contribution de chaque caractéristique à la prédiction d'un échantillon de données.

1.1.4 Outils d'investigation INNVESTIGATE

Dans le cadre de notre étude, nous utilisons un outils d'interprétation de réseaux de neurones appelé iNNvestigate réalisé par Maximilian Alber et al [7]. Cette bibliothèque regroupe l'implémentation de plusieurs méthodes d'analyse.

La performance des méthodes d'interprétation dépend du modèle que l'on souhaite interpréter,c'est pourquoi une comparaison empirique est faite pour pouvoir évaluer ces méthodes et déterminer celle dont l'analyse du réseau de neurones profond est la plus pertinente.

La comparaison qualitative et quantitative des méthodes d'analyse

La bibliothèque reçoit en entrée un réseau de neurones préalablement entrainé et procède à l'analyse, notons qu'elle est particulièrement adaptée aux entrées sous forme d'image. Il en résulte une comparaison qualitative des méthodes d'interprétation. Malheureusement, cette comparaison qualitative peut laisser place à une certaine subjectivité, par exemple sur un classifieur d'image l'analyse a donné les résultats suivant :

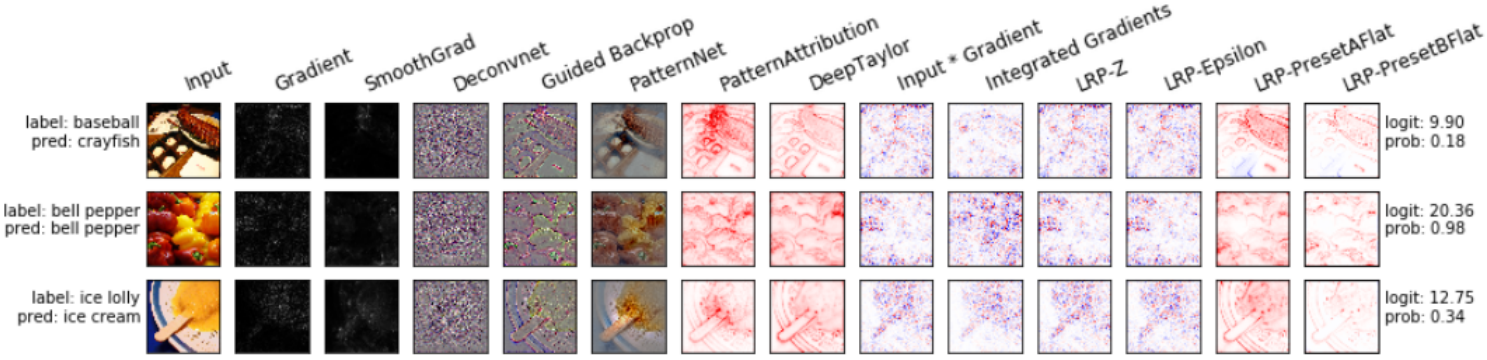


FIGURE 1.4: Comparaison qualitative des différentes méthodes d'analyse

A travers la figure 1.4, nous remarquons que la la comparaison des méthodes d'analyse reste très approximatif surtout pour des données vectorielles tel que les expressions de gènes. Pour pallier à ce problème il serait interessant d'établir une comparaison quantitave plus représentative, c'est

pourquoi une implémentation d'une méthode de perturbation est fourni [8] avec la bibliothèque.

L'intuition derrière cette méthode est qu'en perturbant les attributs identifiés comme importants par la méthode d'analyse, nous aurons une toute autre classification. Cela prouve qu'en effet, la méthode d'analyse a eu raison de choisir ces attributs parmi les plus pertinents. Il en résulte une bien meilleure compréhension des réseaux de neurones profonds.

Chapitre 2

Approche Layer-Wise Relevance Propagation (LRP)

Chapitre 3

Expérimentation

3.1 Description des données

Le problème étudié pour ce projet se situe dans le contexte de la détection de la Leucémie, nous utilisons pour cela des données disponibles en ligne : Des investigations executées sur 11 laboratoires différents, ou des mesures d'expressions de gènes ont été prise sur des patients distincts, et ce associé à leurs profils de Leucémie (leurs Types).

Le jeu de données fournit est encodé sous format GSE ; une sérialisation d'ensemble de fichiers GSM ; qui eux sont des expériences individuelles. Ces derniers contenant les données pertinentes à l'apprentissage.

Ainsi dans chaque expérience ou mesure prise sur un patient, nous possédons un vecteur de caractéristiques associé à chaque expression de géne, et contenant les informations suivantes :

— Son identifiant dans la série.
— Une mesure de signal du génome prise selon le protocole DS.
— Une mesure faite avec le protocole DQN3.
— La valeur P associée à chaque signal mesuré.

Enfin pour une même expérience nous pouvons extraire des métadonnées dont les plus utiles pour ce projet sont :

— Le titre de l'expérience associée.
— La numérotation de l'experience dans sa série.
— Nom de l'organisme qui a effectué l'experience.
— Le protocole utilisé.
— Le type de Leucémie associé au patient ce qui servira ainsi de label pour cette entrée.

La lecture de ce type de fichier se fait à l'aide de la librairie GEOparse de Python.

3.2 Protocole expérimental

3.2.1 Préparation des données

Pour optimiser le processus d'apprentissage certaines précautions sont à prendre, celles-ci amélioreront son efficacité et ses résultats :

— Les etiquettes du jeu de données sont sous forme de 18 catégories, qui peuvent être réduites à 6 catégories principales. Effectuer ce type de réduction (diminution de nombre de classes)

Classes	Sous-Classes	Nombre de patients
ALL	Mature B-ALL with t(8 ;14)	750
	Pro-B-ALL with t(11q23)/MLL	
	c-ALL/pre-B-ALL with t(9 ;22)	
	T-ALL	
	ALL with t(12 ;21)	
	ALL with t(1 ;19)	
	ALL with hyperdiploid karyotype	
	c-ALL/pre-B-ALL without t(9 ;22)	
AML	AML & AML with t(8 ;21)	542
	AML with t(15 ;17)	
	AML with inv(16)/t(16 ;16)	
	AML with t(11q23)/MLL	
	AML with normal karyotype + other abnomalities	
	AML complex aberrant karyotype	
CLL		448
CML		76
MDS		206
Non-leukemia		74

TABLE 3.1

permet d'obtenir un modèle bien plus efficace en y perdant cependant la spécification de so us-catégorie. Ces catégories sont representéés dans le tableau suivant :

— Encoder les classes sous un format de "One-Hot encoding" signifiant ainsi qu'une etiquette sera sous format 6 bits, un bit associé à chaque classe. Ainsi pour un patient donné seulement un seul de ces bits sera mis à un. Ce type de transformation d'etiquettage permet non pas d'avoir un seul neurone en sortie du modèle mais plusieurs, améliorant ainsi la précision de la prédiction. Cette méthode a fait ses preuves dans l'amélioration de la précision dans le contexte de la classification [2].

3.2.1.1 Gestion des Données Désquilibrées

Les données du jeu de données étant desiquilibrées d'une classe à une autre, certaines mesures sont à prendre afin d'éviter un surapprentissage sur certaines classe, un modèle pouvant avoir tendance à maximiser son taux $\frac{Nombred'élémentsprédits correctement}{Nombred'élémentstotal}$ en prédisant le plus souvent la classe majoritaire.

— Après calcul du nombre de patients par classe, on peut observer une distribution deséquilibrée, ce qui en pratique signifie que notre modèle pourra avoir tendance à effectuer un surapprentissage et choisir de prédire le plus souvent la classe dominante. Afin de palier à ce problème nous effectuons le calcul du poids de chaque classe par rapport à l'ensemble du jeu de données et envoyons ce paramètre au modèle afin qu'il l'utilise pour equilibrer son apprentissage.

— Pour que le choix d'un modèle se fasse de manière pertinente il faut utiliser une fonction d'evaluation adaptée au jeu de données et ses caractéristiques, dans ce contexte de classes desequilibrées nous avons opté pour la mesure de précision qui est une fonction prenant en compte les faux positifs ce qui permet de détecter si une classe se retrouve prédite

majoritairement. De plus, il est important dans un but analytique d'effectuer le calcul d'une matrice de confusion qui dénote pour les données : leurs véritables classes parallélement à leurs classes prédites.

$$PRECISION = \frac{\sum_{i=0}^{N} VP}{\sum_{i=0}^{N} VP + \sum_{i=0}^{N} FP}$$

- VP : Vrais Positifs.
- FP : Faux positifs.
- N = Nombre de patients (de données).

3.2.2 Processus d'Apprentissage du Réseau de Neurones Profond

Le processus d'apprentissage consiste en plusieurs étapes d'entrainement durant lesquels les hyper paramètres sont ajustés compte tenu des résultats observés progressivement. Des choix ont été pris concernant certains de ces hyper paramètres ; ainsi pour les couches intérmediaires la fonctions d'activation retenue fût ReLU car étant très utilisée dans plusieurs architectures populaires et étant simple et fiabilité [1] et Sigmoid pour les 6 neurones de sortie car sa plage de valeur correspond à celle d'un *One-Hot Encoding*.

Pour le reste, durant les experimentations le nombre de couches cachées a été varié de 3 à 10, ainsi que le nombre de neurones par couche en avançant par palliers : 50, 100, 1000 et enfin 1500. De plus, afin d'éviter le problème des minimums locaux une même architecture passait par 3 reinitialisations et entrainements afin d'avoir éviter ainsi le problème des minimum locaux. Au fur et à mesure de l'entrainement, il a fallut ajuster les hyperparamètres tel que le pas d'apprentissage, quand il fut trop grand 10 -3 induisant que l'erreur ne diminuait plus (et que donc la précision n'augmentait pas non plus) afin d'obtenir une convergence, mais sans non plus le faire au point que celle-ci se fasse trop lentement, le meilleur pas pour cela égalait 10 et en conséquence il fallût augmenter le nombre d'epochs afin d'atteindre une erreur suffisamment minime. Différents algorithmes d'optimisations ont été testés : ADAM, Stochastic Gradient Descent ainsi que la Descente du Gradient classique. cependant ADAM malgrés son efficacité réputée [3] n'a pu atteindre que le maximum de 0.85 en précision, le Stochastic Gradient Descent quant à lui a atteint les 0.93. Le jeu de données a la particularité que pour chaque expression de géne deux types de mesures ont été prise : DS et DQN3, l'experimentation en effectuant l'apprentissage en utilisant les mesures DS n'a atteint que le score maximal de 60/100 de précision, et par conséquent nous avons opté sur l'utilisation de la seconde mesure. Enfin l'ajout de Dropout a permis l'amélioration de la précision, plus particulièrement pour la dernière classe qui est très peu présente dans le jeu de données et dont ses résultats dans la matrice de confusion s'en sont améliorés.

3.2.3 Interpretation du Réseau de Neurones

Après cela, on passera à l'étape d'interpretation consistant à déterminer grâce à un algorithme choisi, quelles sont les variables les plus importantes et pertinentes pour le meilleur modèle dans sa tache de prédiction.

L'analyse avec chacun des algorithmes : gradient, smoothgrad, deconvnet, guided backprop, input t gradient, integrated gradients, lrp.z, lrp.epsilon est effectuée et nous pouvons dés lors observer les notes attribuées à chacune des variables par ces analyseurs.

Ce procédé nous retournera donc un vecteur de la même taille que l'entrée du modèle (donc 53031 ou le nombre d'expressions de génes par individu) et ou chaque expression de géne a

un score d'importance attribué. Nous ajoutons aussi à cela une version triée de ce vecteur nous permettant ainsi de visualiser les proportions de variables que chaque méthode considère comme importantes.

3.2.4 Calcul de Corrélation

Dans le but d'évaluer de manière plus représentative les résultats obtenus par l'analyse de chacun des algorithmes nous appliquerons un calcul de matrice corrélation, celui-ci nous permettra de déterminer parmis eux lesquels ont une relation linéaires (considérant ainsi les même variables comme importantes), ou inversement (ainsi les variables qu'une méthode considérerai comme importante, la deuxième méthode les considérerai comme banales), ou encore si il n'y a aucune relation entre les deux analyses.

3.2.5 Perturbation des Entrées pour la Comparaison des Méthodes

La perturbation est un procédé qui consiste à prendre les variables détérminées comme pertinentes par un algorithme d'analyse (ou interprétation) et en perturber la valeur afin d'observer l'effet résultant sur la précision des prédictions en moyenne. Cependant cette fonction sous iNNvestigate est dirigée et développée pour les images, il a donc fallût l'adapter au contexte des données vectorielles sachant que le concept de voisinage n'a plus de réel consistance ou signification contrairement à l'image ou vouloir perturber une région signifierai affecter une zone ou carré de pixels précis. Il existe 3 types de perturbations :

- Perturbation par zéros : on remplace les valeurs de variables en entrées par des zéros.
- Perturbation par valeur aléatoire : on remplace les valeurs par une valeur aléatoire.
- Perturbation par moyenne : on calcule la moyenne de valeur sur la ligne, et c'est elle qu'on utilise pour remplacer ces variables importantes.

Cette perturbation nous l'avons effectué de manière incrémentale sur les 1000 variables que chaque modèle considèrait comme les plus importantes afin d'en observer l'impact sur la précision de celui-ci. Celle-ci nous permettra dans un premier temps de comparer les méthodes d'interprétations entre elles, et enfin la qualité des types de perturbation.

La qualité d'un algorithme d'analyse se définit par la vitesse à laquelle décroit la précision du modèle lorsque l'on perturbe les valeurs que cet analyseur juge pertinentes [8].

3.2.6 Hyper paramètrage de LRP

3.2.6.1 Interpretation et Analyse

Afin d'approfondir notre analyse nous nous concentrerons sur la méthode Layer-Wise Relevance Propagation (LRP) et nous ajusterons ses hyperparamètres afin d'observer l'effet que cela aura sur la qualité de l'analyse.

3.2.6.2 Perturbation des expressions de génes

Pour ce faire nous appliquerons la même méthodologie que cité précedemment, dans un premier temps en effectuant une analyse en observant les proportions de variables considérées comme importantes, puis en les perturbant et observant les courbes résultantes.

3.2.7 Analyse par classe de Leucémie

3.2.7.1 Expressions de génes pertinentes

Après ces étapes nous nous attarderons sur l'observation de l'analyse LRP dépendemment des classes, afin de déterminer ou l'existence ou non d'une homogénéité dans les pertinences de variables d'une catégorie à un autre ; exemple : les variables permettant de prédir une Leucémie de type ALL sont-elles les même que celles pour prédire une de type CLL ?

3.2.7.2 Analyse en composantes principales (ACP)

L'ACP ou Analyse en composantes principales est une un méthode d'analyse de données permettant la projection de données ayant un grand nombre de variables (et faisant partie du même ensemble) sur un nombre réduit d'axes décorrelés que l'on appel "Axes Factoriels" nous allons utiliser cette méthode afin de pouvoir mieux visualiser nos résultats graphiquement en représentant sur les deux axes factoriels résultants les analyses faites sur chacun des types de Leucémie. Ainsi nous réduirons les 53031 représentant l'importance de chacune des expressions de génes à seulement 2 dimensions.

3.2.7.3 Partitionnement des Données

Le partitionnement de données (ou "Clustering" en Anglais) est une autre méthode d'analyse de données permettant de créer des groupes homogénes, l'objectif étant de regrouper les individus les plus similaires entre eux. Nous appliquerons cette méthode et grâce à elle nous pourrons déterminer combien de groupes de variables importantes existent par catégories, et donc quelles types de Leucémie utilisent un nombre assez similaire entre elles de variables pour être prédites et associer ainsi un groupement d'expression de génes à plusieurs classes en même temps. Pour ce faire nous allons utilisé l'algorithme k-means se basant sur l'utilisation de centre de gravités initialisés aléatoirement puis se décalant dépendemment des éléments les contenant. Cet algorithme demandant de définir au préalable le nombre de clusters, afin de définir ce nombre on utilise la mesure "Silhouette" qui permet de maximiser la distance inter-clusters et minimiser celles intra-clusters.

3.3 Résultats

3.3.1 Apprentissage du Réseau de Neurones Profond

Les résultats de précision obtenus par le processus d'apprentissage sont représentés dans le tableau ci-dessous.

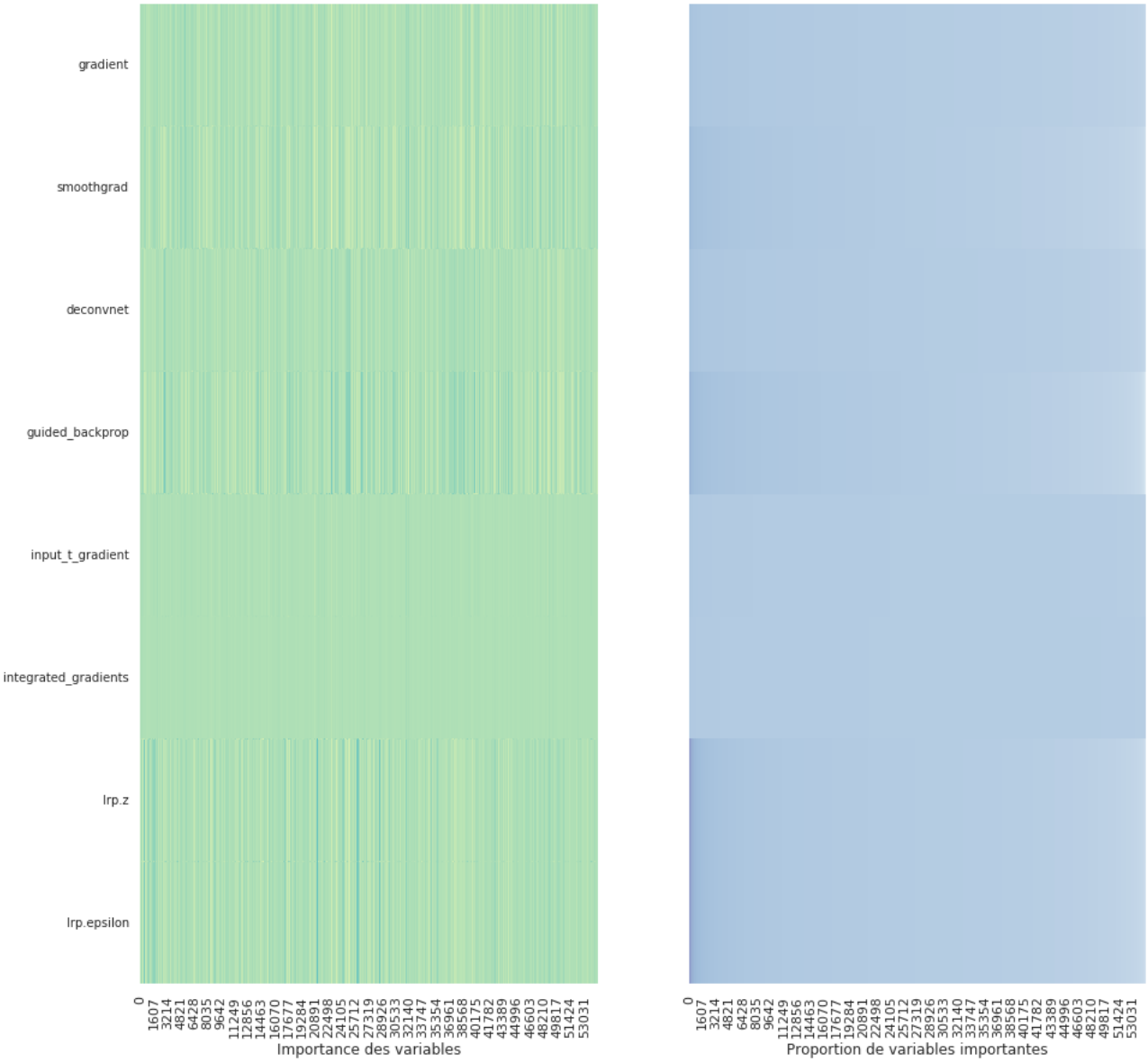
	Nombre de neurones				
Nombre de couches	50	100	1000	1500	2000
3	0.887	0.890	0.880	0.883	0.888
4	0.892	0.8808	0.879	0.8875	0.888
5	0.8891	0.9303	0.8933	0.8813	0.8811
6	0.90452	0.8921	0.8748	0.90120	0.86747
7	0.8970	0.864	0.89224	0.875	0.896

TABLE 3.2: Résultats

On peut constater que les résultats sont autours des 90/100 pour la plus part, cependant l'architecture performant le mieux est celle avec 5 couches cachées et 100 neurones par couche et par conséquent elle sera retenue pour la suite du projet.

3.3.2 Interpretation avec les différents algorithmes

Après l'application des différents algorithmes d'analyse et interpretation sur notre réseau de neurones nous obtenons les graphiques suivants :

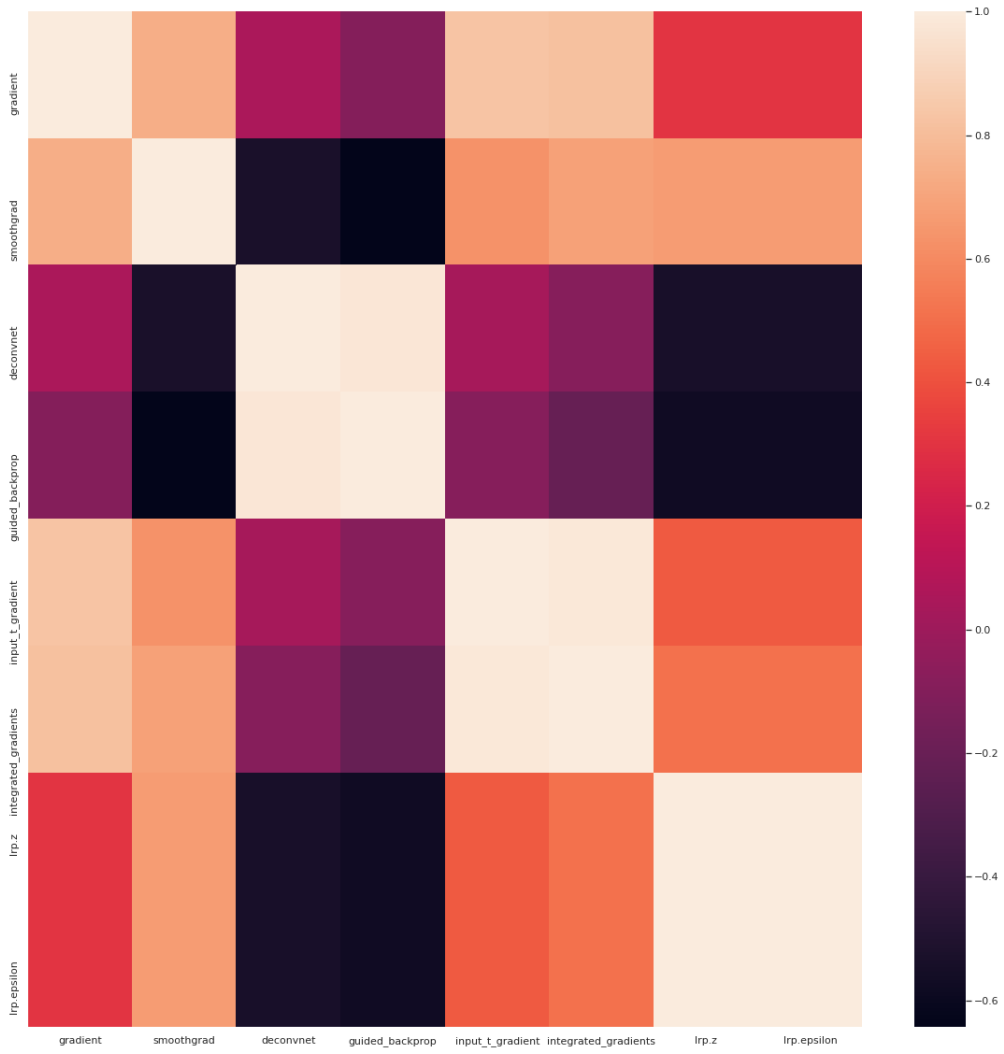


Dans le graphe de gauche on représente pour chacun des algorithmes d'analyse l'importance qu'il consacré à l'ensemble des 53031 variables, plus la couleur est prononcée plus élevée est sa valeur d'importance.
Celui de droite est une version triée du premier permettant ainsi de visualiser les proportions de variables importantes par chacun des analyseurs.

On peut dés lors observer que chacun de ces algorithmes détermine une proportion différente de variables importantes pour la prédiction plus particulièrement sur les deux dernières que sont LRP.Z et LRP.Epsilon qui se démarquent de part le fait qu'ils accordent à leurs premières variables une valeur bien plus elevés que ne le font les autres algorithmes pour les leurs.

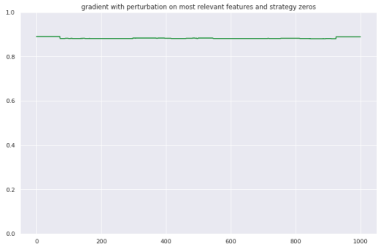
3.3.3 Calcul de Corrélation

Afin de mieux visualiser les variables que chaque algorithme définit comme pertinentes nous avons appliqué un calcul des coefficients de corrélations



Ci-dessus on peut observer une forte corrélation entre les deux méthodes de LRP signifiant qu'elles considérent globalement les même variables importantes, pareil pour Guided Backprop et input t gradient. D'une autre part on remarque que les méthodes Deconvnet et Guided Backprop ont un coefficient se rapprochant du -1 ce qui veut dire qu'elles ont une relation inverse et que donc les variables que l'un définit comme importante, l'autre méthode aura tendance à la définir négligeable.

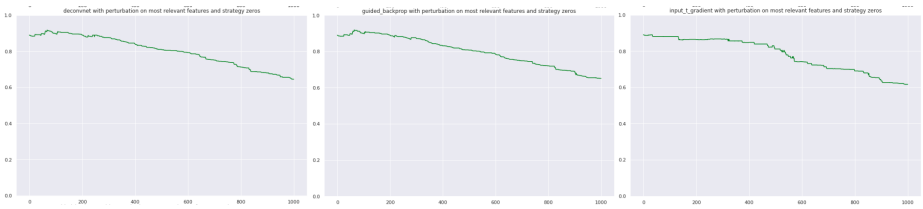
3.3.4 Perturbation des données



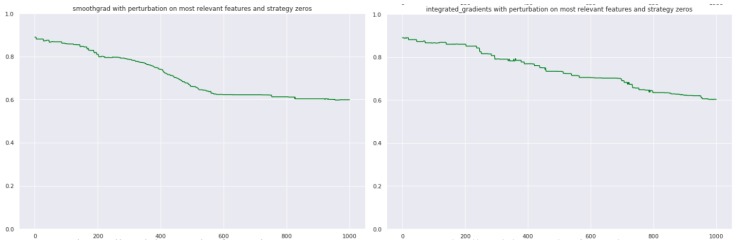
(a) Gradient

D'abord on observe certaines méthodes dont la perturbation des entrées qu'elles considérées importantes n'affecte point la précision de la prédiction c'est le cas pour la méthode de gradient, et par conséquent on en déduit que l'analyse est erronée.

De plus certaines méthodes donnent décroissance constante au fur et à mesure des perturbations, résultat que l'on peut interpreter par une équivalence d'importance de ces variables selectionnées.



(a) Deconvnet (b) Guided BackProp (c) Input t gradient

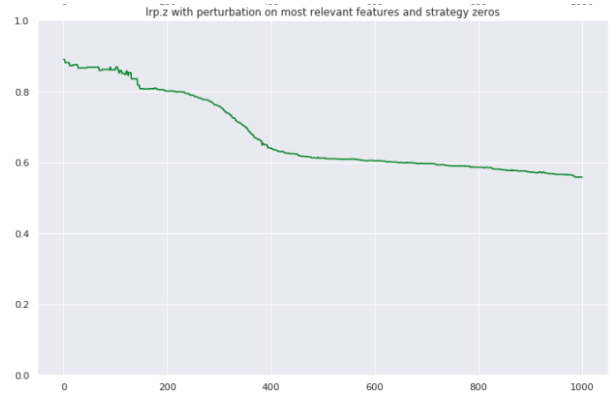


(a) Smoothgrad (b) Integrated Gradients

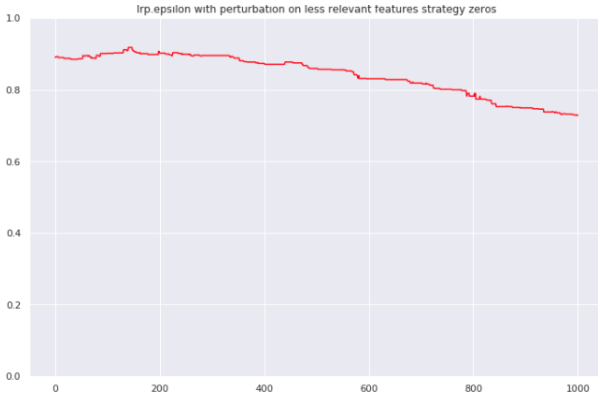
Enfin, et ce sont les cas qui nous intéréssent sont ceux dont la courbe initiale est la plus grande, signifiant que les variables déterminées ont une importance toute particulière dans la réalisation de la prédiction.

(a) LRP.Z (b) LRP.EPSILON

On remarque que les méthode dont les courbes sont les plus prononcées de toutes sont LRP.Epsilon et LRP.z, on peut donc en conclure que cette méthode a une efficacité plus prononcée sur ce modèle déterminant ainsi les variables les plus décisives ce qui va nous pousser à nous intéresser plus précisement à cette analyseur.
Afin d'appuyer ce propos comparons le processus de perturbation dans un premier temps sur les variables qu'il juge les plus importantes et dans un second celles qu'il considére comme les moins importantes.

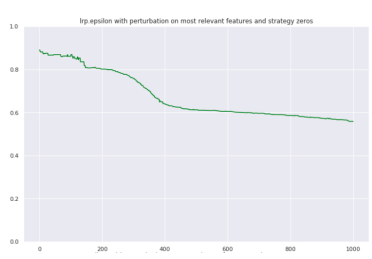


(a) LRP.Z plus importantes variables

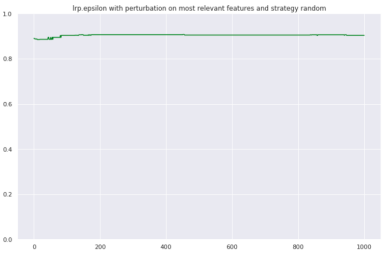


(b) LRP.Z moins importantes variables

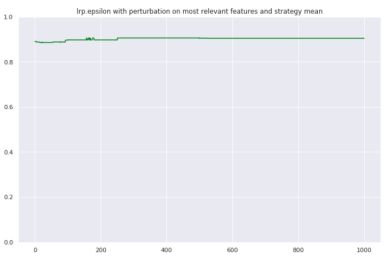
On remarque une claire différence entre cette perturbation et celle initiale, ici le modèle stagne dans sa perturbation confirmant ainsi que l'analyseur a été efficace dans son evaluation.
Concernant les différentes méthodes de perturbation implémentés analysons leurs impacts sur la précision du modèle.



(a) Perturbation par zéros



(b) Perturbation par valeurs aléatoires



(c) Perturbation avec la moyenne

Avec les 2 autres méthodes que sont la perturbation par moyenne et par valeur aléatoire on constate une différence d'impact qui se retrouve réduit réduit de manière plus que conséquente comparativement à la perturbation avec des zéros.

3.3.5 Hyper paramètrage de LRP

3.3.5.1 Interpretation et Analyse

Tout comme pour les parties précédentes, le coté gauche des graphes ci-dessous représente les expressions de gênes dans leurs ordres initials et leurs couleurs déterminent leur importance et pertinence dans la prédiction, tandis que le coté droit lui est une version triée du coté gauche nous permettant ainsi de visualiser les proportions de variables importantes. Trois types de résultats sont ainsi observés par les analyses :

— Premier cas (LRP.Z IB / LRP Epsilon) : ces méthodes ont considéré certaines variables comme extrémement importantes pour la prédiction générale.

(a) Premier cas

— Second cas (LRP Alpha=1 Beta=0 / LRP epsilon IB / LRP W Square / LRP sequential preset b) : considérent des variables importantes différentes que celles considérées par les méthodes du premier cas, et n'en considére aucune comme extrémement pertinente.

(a) Second cas

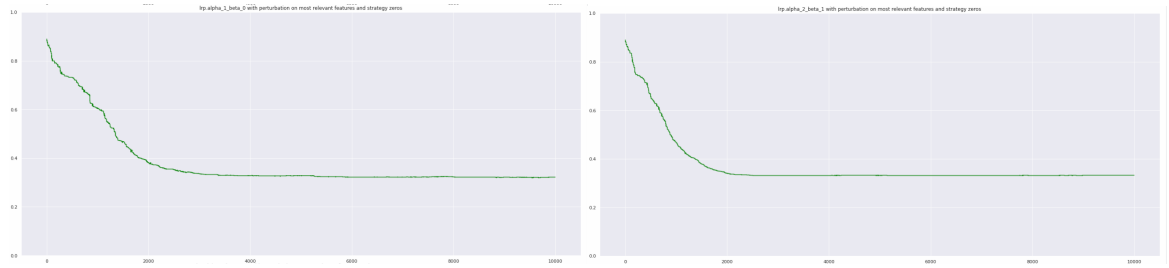
— Troisième cas (LRP.Z / LRP Epsilon / LRP.Z plus / LRP.Z Plus Fast / LRP sequential preset a) : considérent casiment toutes les variables comme négligeables mise à part un petit nombre considéré comme très légérement pertinent.

(a) Troisième cas

3.3.5.2 Perturbation des expressions de génes

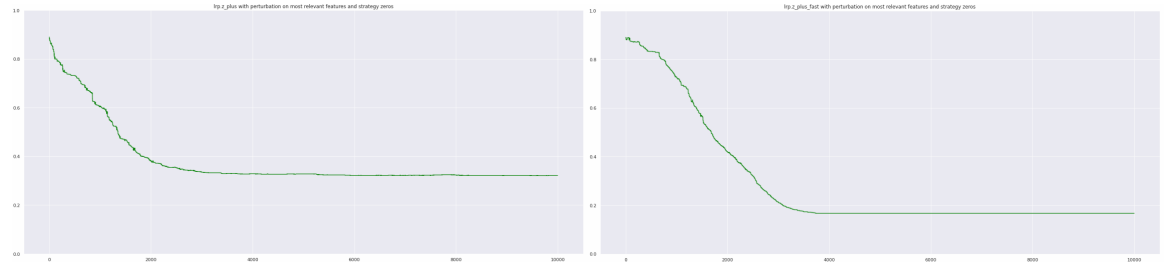
En appliquant une perturbation des données d'expressions de génes considérées comme importantes par ces variantes de LRP nous avons pu observer les résultats ci-dessous :

En faisant varier les valeurs de α et β on obtient cependant des résultats plus ou moins similaires



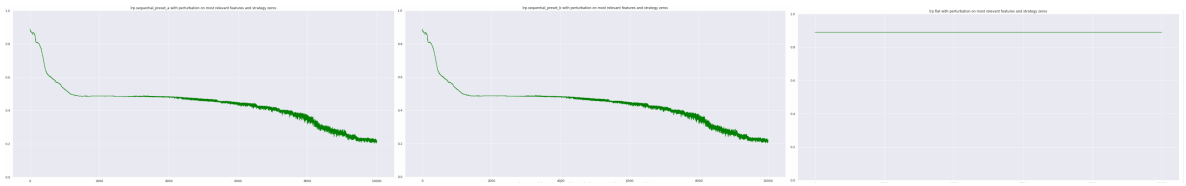
(a) alpha=1 beta=0 (b) alpha=2 beta=1

Les versions ”plus” et ”plus fast” quant à elles de LRP n’ont quant à eux obtenu que des résultats peu satisfaisants compartiviement au reste



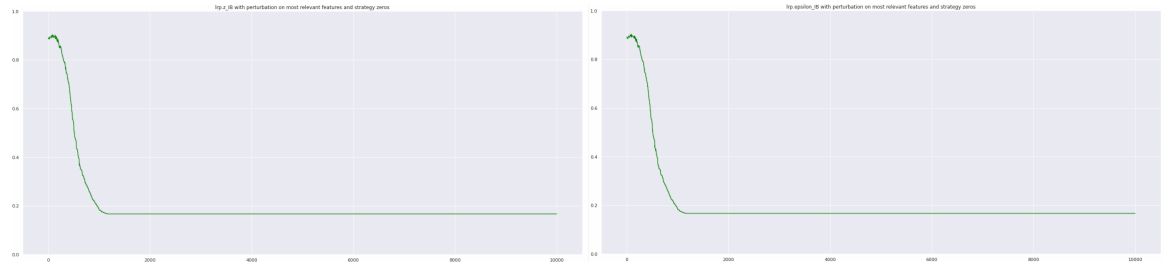
(a) LRP Z Plus (b) LRP Z Plus Fast

Les pires résultats observés ont été sur les méthodes LRP Sequential avec preset sur a et sur b, ainsi que sur la version Flat de LRP :



(a) LRP Sequential Preset A (b) LRP Sequential Present B (c) LRP Flat

Les meilleurs résultats ont été observés sur les méthodes lrp.z.ib et lrp.epsilon.ib et sont plutôt similaires :

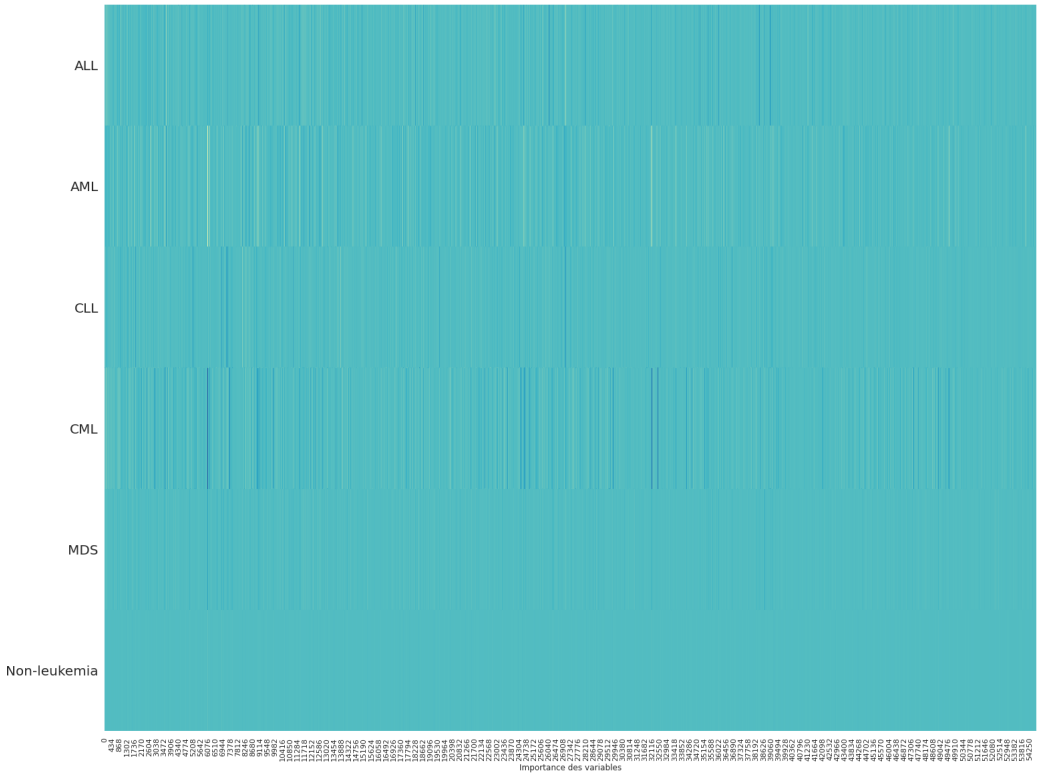


(a) LRP Z IB (b) LRP Epsilon IB

3.3.6 Analyse par classe de Leucémie

3.3.6.1 Expressions de génes pertinentes

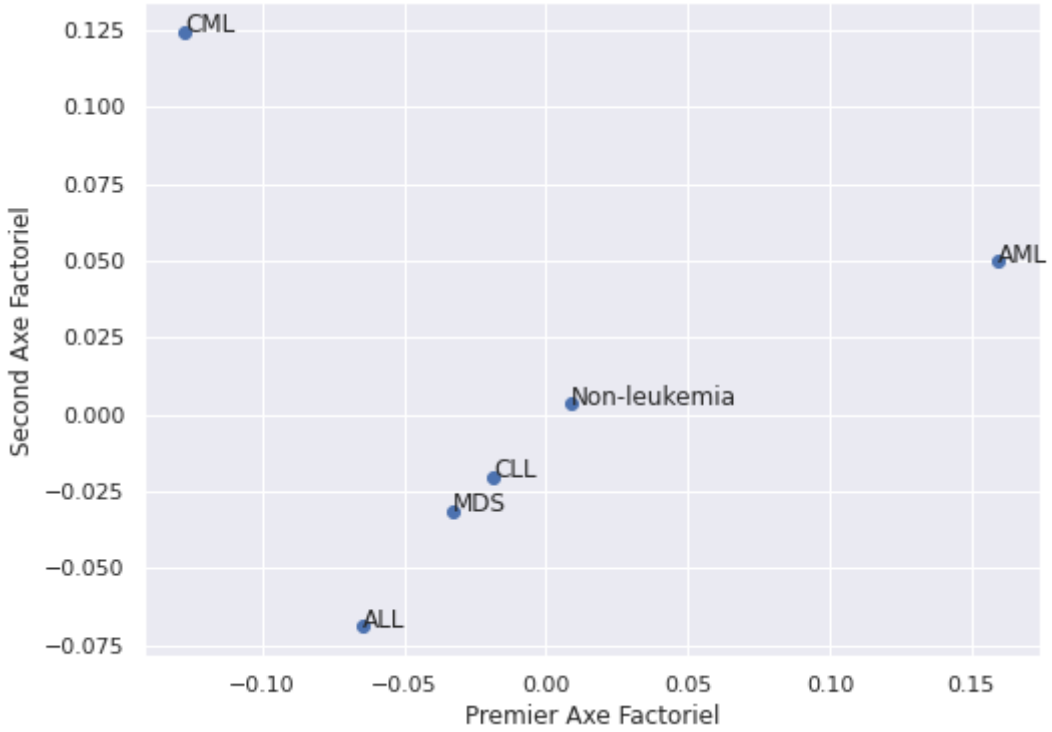
Après application de LRP sur chaque classe séparemment on peut observer des résultats très variés d’une catégorie de Leucémie à une autre, on en déduit que les variables pertinentes à la prédiction peuvent varier d’un type de Leucémie à un autre.



(a) Variables Pertinentes à la Prediction pour chaque type de Leucémie

3.3.6.2 Application de l'Analyse en composantes principales

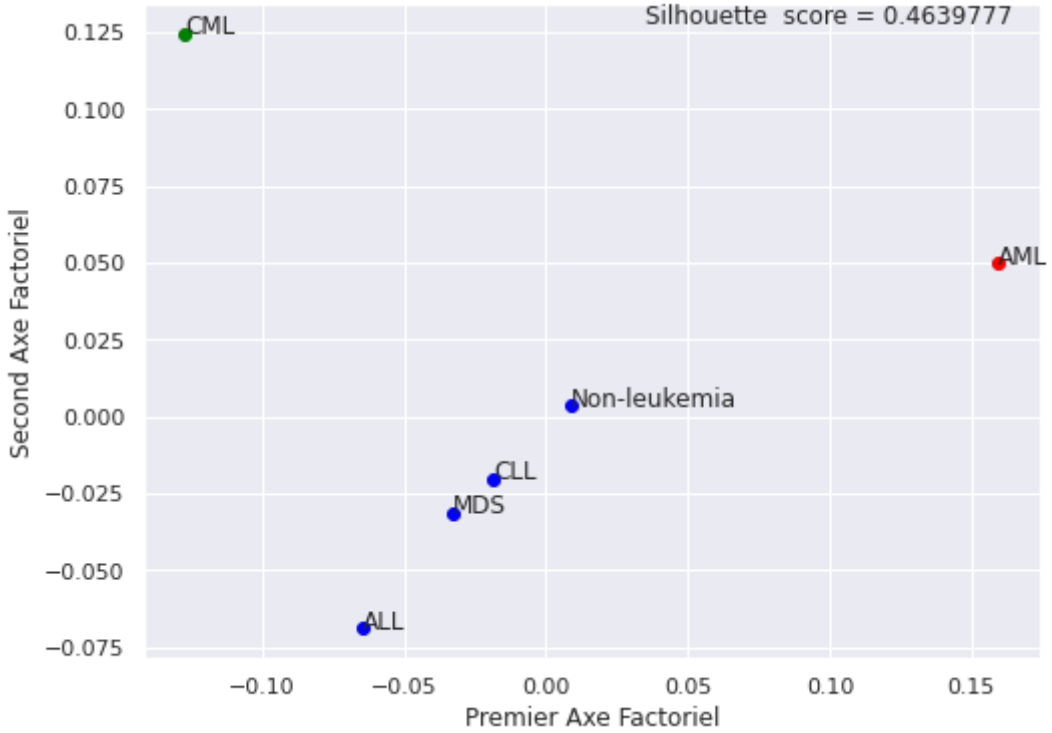
Par l'application de l'Analyse en composantes principales nous obtenons deux axes factoriels (qui représentent les 53031 dimensions initiales pour chaque expression de géne) et grâce à cela nous pouvons afficher ce graphe qui permet de représenter spatialement les différentes catégories de Leucémie selon l'ensemble des variables considérées comme importantes pour la prédiction de chacune d'entre elles.



(a) Représentation des types de Leucémie sur les axes factoriels résultants

3.3.6.3 Partitionnement des Données

Après avoir obtenu une représentation conscise nous effectuons la procédure de clustering sur ces classes et la mesure Silouhette nous définit le nombre optimal de 3 clusters.



(a) Types de Leucémie divisés en familles selon les expressions de génes liées

Nous pouvons déduire de ce clustering et graphe que dans la prédiction du type de Leucémie d'un patient la méthode d'analyse et d'interprétation LRP détermine (et ce de manière plus efficace que les autres méthodologies du genre) que 3 grandes familles d'expressions de gênes sont importantes ; une première nous permettra de prédire la catégorie CML, une deuxième la catégorie AML et enfin une dernière pour les catégories : CLL, MDS, ALL et pas de Leucémie.

Conclusion

Le but de ce projet était de répondre à plusieurs questions : quel est le potentiel d'efficacité des méthodes d'interprétation sur des réseaux de neurones traitant des données vectorielles ? Peut-on le représenter de manière visuelle ? Comment évaluer et perturber des variables dans ce contexte ? Peut-on développer de la connaissance de par ce procédé ?
Au travers de ce travail de recherche nous avons pu effectué plusieurs conclusions ; tout d'abord l'efficacité de la méthode LRP sur les données vectorielles d'expressions de génes et ce de manière supérieure aux autres algorithmes d'interprétation existants.
D'une autre part, malgrés les différences existantes entre les différents principes des méthodologies d'interpretation de réseaux de neurones, on peut retrouver une corrélation entre leurs résultats, qu'elle soit positive (accord commun sur les variables pertinentes) ou négative (contradiction sur les variables pertinentes). Nous avons aussi pu observer que les variables pertinentes à une prédiction dans un même jeu de données peuvent varier d'une classe à une autre ainsi la prédiction d'une Leucémie de type ALL dépendait de variables différentes que la prédiction du type AML par exemple.
Ces conclusions pourraient ouvrir de nouvelles pistes dans les domaines scientifiques dont sont issus les jeux de données, notre objectif dans ce projet ayant été d'observer le comportement des algorithmes et techniques d'inteprétations (habituellement utilisées sur des images) sur des données vectorielles. Ainsi les expressions de génes ayant fait preuve d'importance dans la catégorisation de Leucémie (au travers du processus d'inteprétation puis de perturbation) pourraient nous en apprendre plus sur la maladie. Les regroupements de familles (ou clusters) de types de cette maladie pourraient ouvrir des pistes vers la recherche des similarité effective entre celles-ci.
En conclusion, l'apprentissage profond peut s'avérer bien utile dans la création d'une association ou mapping d'une donnée x vers une prédiction $y = f(x)$, mais il est possible d'aller plus loin de par l'inteprétation (et validation de cette inteprétation par perturbation) de son modèle et ses prédictions afin de mieux comprendre celles-ci ; comprenant ainsi mieux ce qui a été décisif à cela, on peut alors compléter ce procédé par des méthodes d'analyse de données créant ainsi des observations qui elles seront sources de pistes de recherche futures pour tout domaine ayant fourni ces données.

Bibliographie

[1] Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” Haifa, 2010, pp. 807–814. [Online]. https ://dl.acm.org/citation.cfm

[2] Kedar Potdar, Taher S. Pardawala, Chinmay D. Pai, ”A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers”, International Journal of Computer Applications (0975–8887)Volume 175 –No.4, October 2017

[3] Diederik P. Kingma, Jimmy Lei Ba, ”ADAM : A METHOD FOR STOCHASTIC OPTIMIZATION”, ICLR 2015

[4] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin , Klaus-Robert Müller, ”Evaluating the Visualization of What a Deep Neural Network Has Learned”, IEEE Transactions on Neural Networks and Learning Systems, vol. PP, no. 99, pp. 1-14, 2016.

[5] Gurney Kevin : *an introduction to neural networks*, UCL Press, 1997.

[6] Mathivet Virginie : *L'intelligence artificielle pour les développeurs*, édition ENI, 2015

[7] Maximilian Alber et al : *iNNvestigate neural networks !*, 2018.

[8] Samek et al., perturbation analysis, 2017.

[9] H. Dam et al., *Explainable Software Analytics*, 2018.

[10] Supriyo Chakraborty et all, *Interpretability of Deep Learning Models : A Survey of Results*, 2017

[11] Samantha Krening et all, *Learning from explanations using sentiment and advice in rl. IEEE Transactions on Cognitive and Developmental Systems*, 2017.

[12] Laurens van der Maaten and Geoffrey Hinton. *Visualizing data using t-sne*. Journal of machine learning research, 2008.

[13] Marco Tulio Ribeiro, et all. *Why should i trust you ? : Explaining the predictions of any classifier*. International conference on knowledge discovery and data mining, 2016.