Rapport Projet - *Machine Learning*

INTRODUCTION

Le Machine Learning est un procédé consistant à entraîner un modèle à prédire une variable à partir d'un certain nombre de variables prédictives. Un modèle est défini par une fonction (régression linéaire, logistique…) et par un certain nombre de paramètres ou poids. La fonction choisie dépend de la nature des données à analyser (catégoriques, numériques, continues…). Le choix du nombre de paramètres détermine la complexité du modèle et donc sa capacité prédictive par rapport à sa capacité d'apprentissage. Une fonction de coût est également associée au modèle. Elle permet de mesurer à quel point la prédiction de notre modèle est éloignée de la réalité, et prend en compte le nombre de paramètres utilisés (régularisation). L'objectif est donc de définir un modèle tel que cette fonction soit minimisée, traduisant une capacité prédictive optimale notamment afin d'éviter tout risque de sur/sous ajustement (over/underfitting).
Dans le cadre de ce projet, on utilisera un réseau neuronal. Ce sont des régressions logistiques assemblées les unes à la suite des autres ce qui imite le principe de fonctionnement des cerveaux humains. Chaque neurone prend un certain nombre d'entrées (inputs), avec un certain poids, leur applique une fonction d'activation (sigmoïde, relu…) et renvoie une sortie (output).
Notre objectif est de construire un modèle capable de prédire la classe de cancers (LUAD, COAD, PRAD, KIRC, BRCA) en fonction du niveau d'expression de 20 530 gènes chez 801 individus (RNA-Seq).

MATERIELS ET METHODES

Analyse du jeu de données

Cet ensemble de données fait partie du jeu de données « RNA-Seq (HiSeq) PANCAN ». Il s'agit d'une extraction aléatoire de l'expression de 20 530 gènes pour 801 échantillons venant de patients présentant différentes tumeurs : BRCA, KIRC, COAD, LUAD et PRAD.
Le jeu de données présente les échantillons en colonne. Le niveau d'expression de chaque gène pour les 801 patients se trouve dans les 800 premières lignes. Les 800 suivantes contiennent le type de tumeur qui est associé à chaque patient. Le niveau d'expression de chaque gène est compris entre 0 et 15, 15 pour le gène le plus différentiellement exprimé par rapport à un individu témoin (ne présentant aucune tumeur). Il s'agit donc de variables numériques continues. Les différentes classes de cancers ou variables à prédire sont, quant à elles, catégoriques.

Manipulation du jeu de données

La variable *data_fusion* est un dataframe obtenu après reformatage des données d'origine. Elle contient le numéro des échantillons de 0 à 800 en ligne et les 20 530 gènes en colonne (variables prédictives X). La dernière colonne contient les tumeurs associées à chaque patient, ce qui correspond à la variable y, celle que l'on souhaite prédire.

Tri des données

Les variables prédictives étant trop nombreuses par rapport au nombre d'échantillons, il est nécessaire de trier nos données afin d'éviter un cas de surajustement. L'objectif est de sélectionner les gènes dont le niveau d'expression serait caractéristique d'une classe de cancer et seraient donc de bonnes variables

prédictives et inversement, éliminer les gènes ne présentant pas de différentiel d'expression et donc ne nous apportant aucune information.
Nous avons donc généré l'écart-type ou « standard deviation » du niveau d'expression de chacun des gènes toutes classes confondues. Ce dernier est compris entre 0 et 6,7. Nous avons alors généré deux jeux de données différents : le premier, X_t1, composé des 499 gènes ayant un écart-type supérieur à 3 et le second, X_t2, composé des 109 gènes ayant un écart-type supérieur à 4.

Préparation des données

Pour qu'un modèle soit efficace (précision et rappel), il faut que la machine ait été suffisamment entrainée afin d'éviter le phénomène de sous-ajustement (underfitting). Il ne faut pas non plus qu'elle ait appris les données qu'on lui a fournies par cœur, sinon on assiste au phénomène de surajustement (overfitting). Dans les deux cas, la capacité de prédiction du modèle sera très faible. On divise donc nos données, 70% sont des données d'entraînement (x_train, y_train), 30% sont les données de test (x_test, y_test). La variable y est encodée one-hot.

Création du modèle

Les variables prédictives étant continues et la variable à prédire catégorique, une régression logistique pourrait être utilisée. En revanche, la variable à prédire est multi-classes et non à deux classes, une « simple » régression logistique n'est donc pas adaptée à cette analyse.
Nous avons choisi d'utiliser un réseau de neurones à 3 couches (une couche d'entrée, une couche de sortie et une couche intermédiaire). Pour la couche intermédiaire composée de 10 neurones, la fonction Relu est utilisée. Elle permet de résoudre les problèmes de saturation des courbes sigmoïde et tangente hyperbolique. On utilise pour la couche de sortie composée de 5 neurones (un par classe de tumeur) la fonction softmax. Cette fonction est spécialisée dans la classification en multi-classes et est donc adaptée à notre analyse. Cette fonction convertit ses entrées en vecteur de probabilités dont la somme est égale à 1.
Comme fonction de coût à optimiser, nous avons choisi d'utiliser le fonction categorical_crossentropy. Elle est adaptée aux cas où la variable à prédire ne peut appartenir qu'à une seule classe et où elle est encodée one-hot. Nous avons réalisé l'analyse deux fois, la première avec le jeu de données X_t1 (model1) et la seconde avec X_t2 (model2) afin de comparer les résultats.

Evaluation du modèle

Nous avons ensuite mesuré la capacité prédictive de notre modèle. Pour cela nous avons créé la fonction precision(). Elle prend en paramètre la table de contingence établie à partir des données test (x_test) auxquelles on a appliqué le modèle, z_t, et des véritables variables à prédire, y_test.
Enfin, nous avons généré la courbe d'apprentissage associée à chacune des deux analyses grâce à la fonction learning_curve() qui prend en paramètre x_train/test, y_train/test ainsi que le modèle utilisé.

RESULTATS

Une fois notre modèle défini et entraîné, nous avons évalué son efficacité au travers d'une table de contingence générée à partir des données prédites, z_t, et de la réalité, y_test. Avant de réaliser l'analyse avec l'architecture définitive (e.i. 10 neurones composant la couche intermédiaire), nous avions fait l'analyse avec seulement 3 neurones et les résultats obtenus étaient plus cohérents (précision proche mais différente de 1). En revanche, aucune tumeur appartenant à la classe 2 (COAD) n'avait été identifiée par le modèle. A l'issu de ce résultat, nous avons donc décidé d'augmenter la complexité du modèle (passage de 3 à 10 neurones dans la couche intermédiaire) et avons obtenu la table de contingence présentée dans la **figure 1**. Cette fois, le résultat suggère que notre prédiction serait

« parfaite », nous n'avons que des vrais positifs et donc une précision de 1 ce qui est impossible. Le modèle réagit comme s'il avait été entraîné avec les données test et connaissait donc le résultat à l'avance. Malgré cette supposition, nous n'avons pas pu identifier l'origine de l'erreur.

col_0	0	1	2	3	4
row_0					
0	101	0	0	0	0
1	0	47	0	0	0
2	0	0	29	0	0
3	0	0	0	43	0
4	0	0	0	0	45

Figure 1 : Table de contingence de la prédiction des différents types de tumeurs (en colonne) par rapport à la réalité (en ligne) obtenue à l'issue de l'analyse des deux jeux de données, t1 et t2, par les deux modèles model1 et model2.

A la suite de ce résultat, nous avons réalisé la courbe d'apprentissage avec chacun des deux jeux de données. Dans les deux cas, nous avons obtenu le graphe présenté dans la **figure 2**. Ce résultat suggère d'autant plus qu'il y a un problème avec l'apprentissage du modèle. En effet, au vu de cette courbe, les deux modèles sont capables de prédire de manière exacte l'ensemble des 265 tumeurs test avec seulement 5 données d'entrainement, ce qui est impossible.

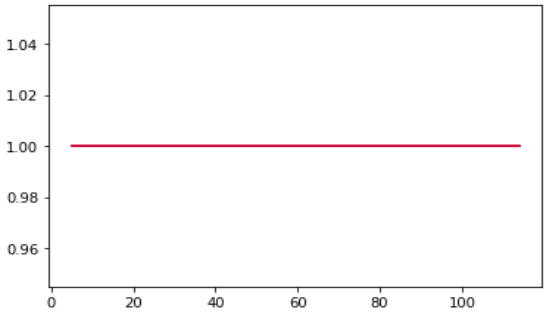

Figure 2 : « Learning Curve » représentant la qualité des prédictions. Abscisses : Nombre d'échantillons utilisés pour l'entraînement. En ordonnées : Précision ou Accuracy. La courbe représentant l'entraînement (training) est en rouge, celle représentant la prédiction (testing) est en bleu. Les deux droites sont superposées.

Quel que soit le jeu de données ou le modèle utilisé on obtient les mêmes résultats incohérents. On peut donc supposer que la source du problème se trouve dans l'architecture des modèles ou dans le traitement des données en amont. Néanmoins nous n'avons pas réussi à l'identifier tout en rendant le script fonctionnel. En théorie, nous aurions dû obtenir une table de contingence similaire à celle présentée par la **figure 3**, avec une prédiction proche de 1.

	0	1	2	3	4
0	**80**	8	5	5	3
1	1	**40**	0	5	1
2	6	0	**20**	2	1
3	0	9	3	**31**	0
4	2	3	2	3	**35**

Figure 3 : Table de contingence théorique représentant une possibilité de résultats cohérents. En ligne : tumeurs observées dans la réalité (y_test). En colonne : tumeurs prédites (z_t).

La courbe d'apprentissage obtenue aurait dû être telle que la capacité de prédiction du modèle (précision) augmente avec le nombre d'échantillons utilisés pour l'apprentissage.

DISCUSSION

En admettant que les résultats obtenus soient cohérents, différentes approches auraient pu être utilisées pour améliorer notre analyse.
Tout d'abord, la réduction de dimensions de la variable X (nombre de gènes) aurait pu être faite de manière différente. Par exemple, nous aurions pu calculer l'écart-type (standard deviation) de l'expression de chaque gène mais pour les individus présentant une même classe de tumeurs. Ainsi, nous aurions eu la variance de l'expression de chaque gène à l'intérieur d'une même classe et aurions pu sélectionner seulement les gènes présentant la plus grande variation d'expression entre les différentes classes. En effet, un gène dont l'expression varie beaucoup au sein de toutes les classes n'est donc pas caractéristique d'une classe particulière et ne nous est pas utile pour la prédiction. Cela aurait aussi pu être réalisé par un test ANOVA (multiple) (Analyse of Variance) qu'il aurait fallu coupler à un test des hypothèses de normalité au sein de chaque groupe (Test Shapiro-Wilkinson, histogramme et Q-Q plot sur les résidus), d'homogénéité des variances (Test de Levene) et vérifier que les observations soient indépendantes. Ce test permet en plus de ne sélectionner que les gènes dont la variation de l'expression est statistiquement significative (calcul de la p-valeur).
Nous aurions également pu diminuer le nombre de gènes par une analyse de la covariance entre les différents gènes. En effet, l'information que l'on extrait de deux gènes dont l'expression est parfaitement corrélée est la même que si nous ne considérions que l'un des deux gènes mais le nombre de gènes utilisés est plus important (dimensions). Il serait donc intéressant de réaliser une analyse ANCOVA (Analyse of Covariance) afin d'éliminer les gènes qui n'augmentent pas la capacité prédictive du modèle.
Dans une approche similaire, nous aurions pu réaliser une analyse en composantes principales (PCA) pour réduire le nombre de variables prédictives en regroupant les variables corrélées (covariance) en composantes principales (dimensions). L'objectif est de ne sélectionner qu'un nombre restreint de composantes principales telles que la distribution des individus en fonction de ces composantes reste similaire à la distribution des individus en fonction de l'ensemble des variables prédictives.

En conclusion, le « machine learning » peut être un outil efficace afin de prédire une variable à partir d'un nombre important de données, de même nature ou multi-omiques. Ainsi, ce procédé pourrait par la suite être développé et utilisé dans le cadre du projet de médecine personnalisée pour orienter les traitements de personnes atteintes de cancers ou de maladies génétiques et organiser des dépistages.