Projet SY09 - QSAR

Maxime Emschwiller, Yann Ladeve, Zhishan Tao

12 juin 2020

Résumé

Dans le cadre de l'UV SY09, nous avons analysé un jeu de données portant sur la toxicité de produit ch imiques. Ce rapport présente nos résultats.

1 Introduction

Notre trinôme a travaillé sur le jeu de données *QSAR oral toxicity dataset*. Le jeu de données est un tableau individus-variables composé de 8092 individus et 1024 variables binaires. Chaque individu représente un produit chimique au format *molecular fingerprint*. Les 1023 premières variables sont l'ensemble des atomes et/ou sous-structures chimiques pouvant être présentes chez un individu. La présence ou non d'une variable est représentée par une valeur binaire. La dernière colonne de ce jeu de données informe sur la toxicité de la molécule. La toxicité d'un individu a été déterminée expérimentalement grâce à la *toxicity Lethal Dose* (LD_{50}). Chaque individu peut avoir la valeur *positive* ou *negative*.

L'enjeu de la découverte d'un modèle permettant de déterminer la létalité d'une molécule est double. Il est d'abord éthique puisque aujourd'hui cette létalité est déduite d'expériences sur des animaux. Un groupe de rats est soumis à l'agent et la létalité de cet agent vient de la proportion du groupe qui succombe pondérée par la quantité de produit utilisé. Un individu sera considéré comme *very toxic* (classe positive) si la dose utilisée pour faire succomber la moitié de groupe test (la LD_{50}) est inférieure à $50mg/kg$. Si la dose utilisée pour arriver à cette mortalité est supérieure à ce seuil, l'individu sera dans la classe negative. L'enjeu est ensuite économique puisque de telles expériences coûtent chères et sont très chronophages. Pouvoir se fier à un modèle de prédiction permettrait donc d'économiser de la souffrance animale, du temps, et de l'argent.

Les noms des individus utilisés dans ce jeu de données se trouvent *ici*. Cependant nous n'avons pas trouvé d'information permettant de lier les produits chimiques au format *molecular fingerprint* avec les noms donnés dans ce fichier, ni même de savoir quelles sont les 1023 variables utilisées pour indiquer la présence/absence de celles-ci dans ces individus. Cependant nous pouvons regarder la distribution des classes negative/positive du jeu de données.

1.1 Objectif

À partir de ce jeu de données, établir un modèle permettant de prédire la classe de toxicité d'une molécule. On verra que la problématique se portera sur le déséquilibre important dans la taille des classes. Ce déséquilibre apportera un biais certains dans la précision des modèles et nous forcera à vérifier également leur sensibilité. Un bon modèle sera alors essentiellement un modèle proposant une sensibilité élevée (c'est à dire un modèle prédisant correctement les individus de la classe "positive"), la précision étant bonne dans presque tous les cas.

2 Analyse exploratoire

Ce jeu de données n'étant composé que de variables binaires et que d'une seule variable qualitative, la recherche d'informations semble assez limitée. Cependant plusieurs informations peuvent être récupérées.

2.1 Cohérence

Tout d'abord un contrôle sur la cohérence des données a été fait. Le but est de supprimer les molécules qui n'auraient que des 0 pour les 1023 premières variables (c'est à dire qu'elles ne possèdent aucun de ces atomes) mais qu'elles soient classées comme positive. Ce cas n'est pas possible car l'ajout des variables dans le jeu de données se base sur la structure chimique de chaque individu. Cela implique qu'un individu doit avoir au moins la présence d'une variable. Après vérification, il n'y a aucun individu qui rentre dans ce cas.

2.2 Distribution

Une fois la cohérence du jeu de données vérifiée, nous nous sommes intéressés à la répartition des individus dans les classes "positive" et "negative".

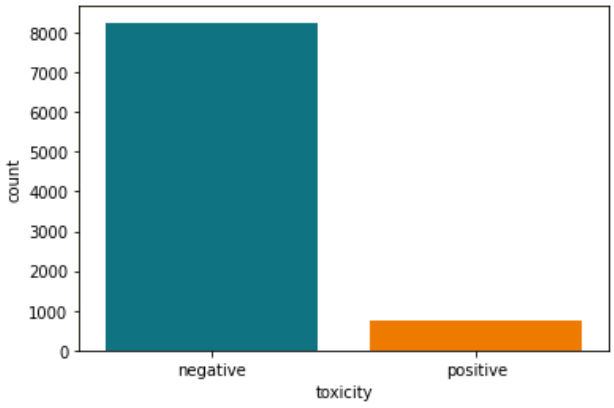

FIGURE 1 – Distribution des classes

Figure 1 montre un déséquilibrage important du jeu de données avec respectivement 91.7% et 8.3% d'individus dans les classes "negative" et "positive".

2.3 Classification Automatique

Il peut être intéressant de voir si naturellement les individus forment des classes. Pour ce faire, on utilise la classification automatique. Ce modèle considère initialement chaque individus comme un cluster puis agrège au fur et à mesure les clusters les plus proches entre eux. L'objectif étant de comparer la classification obtenue par cette méthode avec la classification connue, nous limitons le nombre de clusters à 2. Différentes métriques ont été utilisées pour voir laquelle performe le mieux.

Ensuite l'indice Rand ajusté est utilisé pour comparer la partition obtenue avec celle connue.

Metrics	Rand Score
euclidean	0.003878
manhattan	0.003878
hamming	0.003878
minkowski	0.003878
jaccard	0.008940
dice	0.008940
kulsinski	0.036858

TABLE 1 – Indice de Rand pour différentes métriques

On peut voir que peu importe la métrique utilisée, l'indice de Rand est toujours très proche de 0. Cela veut dire que les classes affectées aux individus par la méthode de classification automatique ont été attribuées de façon complètement aléatoire.

3 Visualisation

Le jeu de données possédant de nombreuses variables, il est difficile de savoir si certaines possèdent des corrélations fortes entre elles. C'est pourquoi nous utilisons des méthodes factorielles pour essayer d'extraire des tendances représentatives.

3.1 Analyse en Composantes Principales

La méthode d'analyse en composantes principales est un algorithme permettant de réduire le nombre de dimensions tout en préservant le maximum d'information sur des axes qui la résume. Des axes appelés **composantes principales** sont générées. Ces composantes principales sont combinaisons linéaires des axes initiaux. La qualité de la compression est mesurée grâce à la quantité d'inertie expliquée par chaque composantes.

Lors de la compression de $QSAR$ en deux axes factoriels, nous obtenons une inertie cumulée faible de 16.9%.

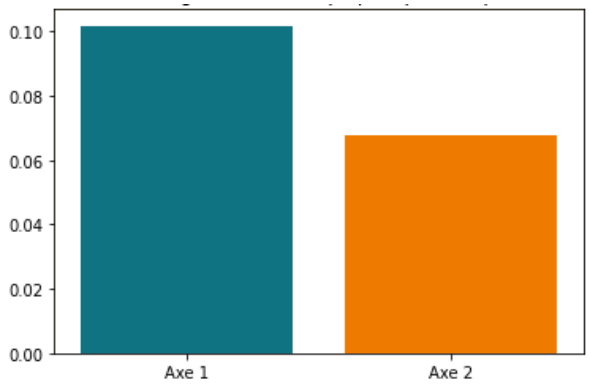

FIGURE 2 – Inertie expliquée par chaque axe

Ce résultat n'étant pas convaincant, nous ne pourrons pas nous baser dessus pour obtenir une visualisation interprétable des données dans le plan. Nous avons regardé le nombre d'axes factoriels qu'ils nous faudraient pour avoir une inertie cumulée de 95%. 610 axes sont nécessaires pour obtenir ce résultat. La différence de dimensions est notable, nous essaierons donc de l'utiliser pour améliorer certains modèles sensibles à la dimension.

4 Jeu de données déséquilibré

Comme expliqué dans l'introduction, la distribution des individus de *QSAR* n'est pas équilibrée. Ce déséquilibre peut amener à des modèles prédictifs trompeurs. En effet lorsque l'on applique la méthode des *K-Nearest Neighbours*, nous obtenons à première vue une précision très satisfaisante.

	precision	recall	f1-score
negative	0.95	0.99	0.97
positive	0.72	0.38	0.50
accuracy			0.94

TABLE 2 – Résultat apprentissage KNN

En effet si l'on considère la métrique *précision* comme variable expliquant la qualité du modèle alors 0.94 est un très bon résultat. Cependant nous sommes intéressés par la possibilité de pouvoir prédire les molécules qui sont toxiques. Cette métrique est appelée la *sensibilité* qui représente le pourcentage d'individus réellement toxiques qui ont été classés comme tel. Ici ce résultat est de 0.38, ce qui n'est pas satisfaisant.

Plusieurs techniques sont efficaces pour faire face à un jeu de données déséquilibré. Pour cette analyse, nous avons procédé à un rééquilibrage du jeu de données par réduction de l'échantillon de la classe majoritaire, par augmentation de la classe minoritaire, ou encore par pondération.

Il est à noter que ces méthodes peuvent amener à des biais induis par des pertes de données importantes ou l'exagération de données contradictoires. Elles sont donc à manier avec prudence et à prendre en compte dans l'interprétation des résultats.

4.1 Méthodes d'augmentation de la classe minoritaire

Ces méthodes de sur-échantillonage (*oversampling* consistent à générer des instances de la classe minoritaire. Ici, on générera des individus positifs. Nous avons utilisé la méthode appelée SMOTE (pour Synthetic Minority Over-sampling TEchnique) pour générer des individus positifs en respectant la tendance formée par les individus déjà présents dans le jeu de données. On considère qu'il est possible de définir un espace délimité par des individus positifs dans lequel ces derniers sont majoritaire. Alors, si les classes étaient de tailles égales, cet espace serait un cluster pour notre classe minoritaire. On va donc artificiellement remplir cet espace

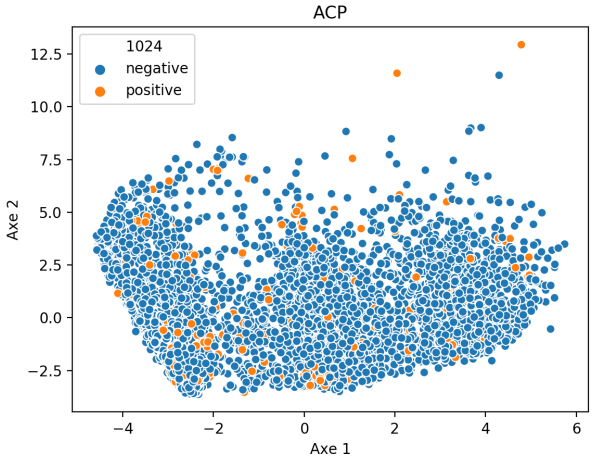

FIGURE 3 – avant SMOTE

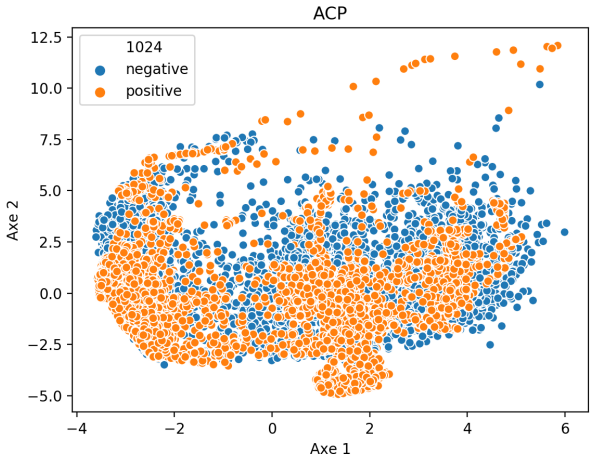

FIGURE 4 – après SMOTE

d'individus minoritaires pour équilibrer la taille des jeu de données.

	data_train	positive	negative
qsar	6744	556	6188
SMOTE	12376	6188	6188

TABLE 3 – Nombre d'individus total et par classe (sampling_strategy= 1)

4.2 Méthodes de diminution de la classe majoritaire

Ces méthode de sous-échantillonnage (*undersampling* consistent à supprimer des individus de la classe ma-

joritaire aléatoirement. On garde en revanche tous les individus de la classe minoritaire.

4.3 Méthode de pondération

Les méthodes précédentes ont l'inconvénient altérer les individus présent dans le jeu de données. Un autre moyen d'équilibrage est de donner une valeur inégale à chaque individu. Ainsi, en donnant une valeur supérieure aux individus de la classe minoritaire, nous leurs donnons une plus grande importance pour les modèles afin de contrebalancer leur sous-nombre. Les deux classes seront alors également considérées par les modèles de prédiction.

5 Modèles prédictifs

5.1 K plus proches voisins

K-Nearest Neighbours est une méthode d'apprentissage supervisée. Dans notre cas, il est utilisé comme une méthode de classification. La règle de décision est la suivante ; pour un nouveau point k, KNN regarde les n plus proches voisins de celui-ci (le nombre n de voisins est défini par l'utilisateur). Le point k est ensuite affecté à la classe majoritairement représentée parmi ces n plus proches voisins. Ce modèle prédictif a l'avantage d'être simple à comprendre et a implémenter, de plus il n'y pas de phase d'apprentissage. Cependant plus la taille du jeu de données est importante, plus la durée de compilation s'allonge.

Pour déterminer les paramètres optimaux pour ce modèle, nous avons utilisé la fonction "GridSearchCV" de la librairie python sklearn. Les paramètres à déterminer sont le nombre de voisins et la métrique de distance. Le score qui a servi de référence pour choisir les meilleurs paramètres est la sensibilité. Pour les métriques de distance nous avons proposé plusieurs types : *euclidean, manhattan, hamming, minkowski, jaccard, dice et kulsinski.* Les paramètres optimaux obtenus après validation croisée sont 1 pour le nombre de voisins et kulsinski pour la métrique.

k = 1	précision	sensibilité	spécificité
qsar	0.92	0.57	0.95
over_sampling	0.92	0.54	0.95
under_sampling	0.70	0.80	0.69

TABLE 4 – Résultat apprentissage KNN

Le réequilibrage du jeu de données avec la méthode d'under sampling permet d'avoir une amélioration de la sensibilité mais cela un impact sur la spécificité qui de ce fait réduit la précision générale.

5.2 Binned Nearest Neighbours

Binned Nearest Neighbours (BNN[1]) prédit la réponse cible au moyen d'un nombre variable k de voisins selon le critère du vote majoritaire. L'idée principale est de considérer pour la prévision tous les voisins qui sont le plus similaires et comparables à la cible. Pour sélectionner les voisins les plus similaires, des groupes de similitude (bins) sont prédéfinis et les voisins sont répartis dans ces groupes en fonction de leur similitude avec la cible. Les voisins qui sont dans le groupe avec la plus grande similitude sont considérés pour la prédiction.

La méthode BNN implique d'abord une procédure de regroupement (binning) de la mesure de similitude telle que définie ci-dessous.

Les groupes de similarité sont déterminés par l'optimisation d'un paramètre α, qui définit la largeur de l'intervalle :

$$\text{bin}_m^\alpha = \left[S_m^\alpha, S_{m+1}^\alpha\right] \quad m = 1, 2, 3, \ldots$$

bin no.	similarity vector	exponent α						
(m)	(S_m)	0.1	0.3	0.5	0.7	0.9	1.2	1.5
bin 1	1.0	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
bin 2	0.9	0.9895	0.9689	0.9487	0.9289	0.9095	0.8812	0.8538
bin 3	0.8	0.9779	0.9352	0.8944	0.8554	0.8181	0.7651	0.7155
bin 4	0.7	0.9650	0.8985	0.8367	0.7791	0.7254	0.6518	0.5857
bin 5	0.6	0.9502	0.8579	0.7746	0.6994	0.6314	0.5417	0.4648
bin 6	0.5	0.9330	0.8123	0.7071	0.6156	0.5359	0.4353	0.3536
bin 7	0.4	0.9124	0.7597	0.6325	0.5266	0.4384	0.3330	0.2530
bin 8	0.3	0.8866	0.6968	0.5477	0.4305	0.3384	0.2358	0.1643
bin 9	0.2	0.8513	0.6170	0.4472	0.3241	0.2349	0.1450	0.0894
bin 10	0.1	0.7943	0.5012	0.3162	0.1995	0.1259	0.0631	0.0316
bin 11	10^{-2}	0.6310	0.2512	0.1000	0.0398	0.0158	0.0040	0.0010
bin 12	10^{-3}	0.5012	0.1259	0.0316	0.0079	0.0020	0.0003	0.0000
bin 13	10^{-4}	0.3981	0.0631	0.0100	0.0016	0.0003	0.0000	0.0000
bin 14	10^{-5}	0.3162	0.0316	0.0032	0.0003	0.0000	0.0000	0.0000
bin 15	10^{-6}	0.2512	0.0158	0.0010	0.0001	0.0000	0.0000	0.0000
bin 16	10^{-7}	0.1995	0.0079	0.0003	0.0000	0.0000	0.0000	0.0000
bin 17	10^{-8}	0.1585	0.0040	0.0001	0.0000	0.0000	0.0000	0.0000

TABLE 5 – Valeurs de similarité des 17 groupes pour certaines valeurs α sélectionnées

Les similarités S sont différentes valeurs de similitude définies dans l'ordre décroissant avec une raison de 0.1 dans la plage [0.1, 1] et 10^{-1} dans la plage [0, 0.1] (par exemple, 0.01, 0.001, etc.). Le dernier groupe a été divisé en sept pour éviter qu'il ne soit trop grand, lorsqu'il s'agit de petites valeurs de α (par exemple, $\alpha = 0.1$, bac $10 = 0.7943$; 5), et ainsi éviter le risque que tous les n - 1 les objets tombent ensemble dans le dernier groupe. L'exposant optimal α utilisé pour définir les seuils de bin est recherché dans la plage [0.1, 1.5] avec un pas de 0.05 et est sélectionné comme valeur donnant l'erreur de classification la plus faible par un protocole de validation. 5 montre les valeurs de similitude pour les

différents groupes et les valeurs α sélectionnées. On peut déduire que lorsque α est compris entre 0 et 1, une distribution concave des groupes est obtenue et les groupes ont une largeur croissante (cést-à-dire que les premiers groupes correspondant aux valeurs de similitude les plus grandes sont plus étroits), tandis que lorsque α est supérieur à 1, une distribution convexe est obtenue et les groupes ont une largeur décroissante (cést à dire , les premiers groupes sont plus larges). Dans la figure 5, deux exemples sont présentés pour $\alpha = 0.5$ et $\alpha = 1.4$.

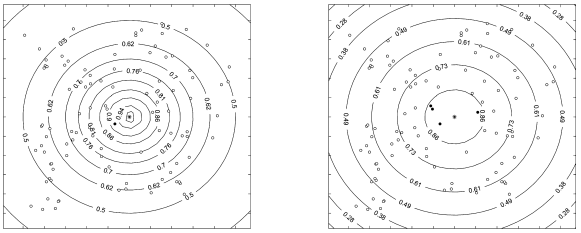

FIGURE 5 – Séquence de regroupement pour $\alpha = 0.5$ (côté gauche) et $\alpha = 1.4$ (côté droit). Dans le premier cas, le premier groupe de similitude non vide est (0.94-0.90) avec une largeur de 0.04 contenant un objet ; dans le second cas, le premier groupe de similitude non vide est (1.00-0.86) avec une largeur de 0.14 contenant 4 objets.

Une fois les groupes définis, l'algorithme BNN classe les objets sur la base du schéma suivant :

1. la similitude avec la cible est calculée pour chaque objet ;
2. les objets sont répartis dans les groupes de similitude en fonction de leur similitude avec la cible ;
3. seuls les objets du premier groupe non vide sont sélectionnés comme voisins les plus proches à utiliser pour la prédiction ;
4. la prédiction est prise comme vote majoritaire ; lorsque plusieurs classes partagent le même nombre de voisins, l'objet cible est affecté à la classe ayant la somme maximale des contributions de similarité.

La première ligne de la TABLE 6 est le taux de précision lorsque $\alpha = 0.2$. En raison du déséquilibre des données, nous avons utilisé under sampling[2] et over sampling[3] pour équilibrer l'ensemble d'apprentissage.

$\alpha = 0.2$	précision	sensibilité	spécificité
qsar	0.94	0.55	0.97
over_sampling	0.94	0.57	0.97
under_sampling	0.89	0.75	0.91

TABLE 6 – Résultat apprentissage BNN

FIGURE 6 est la matrice de confusion après avoir utilisé under sampling. On peut voir que la sensibilité augmente considérablement

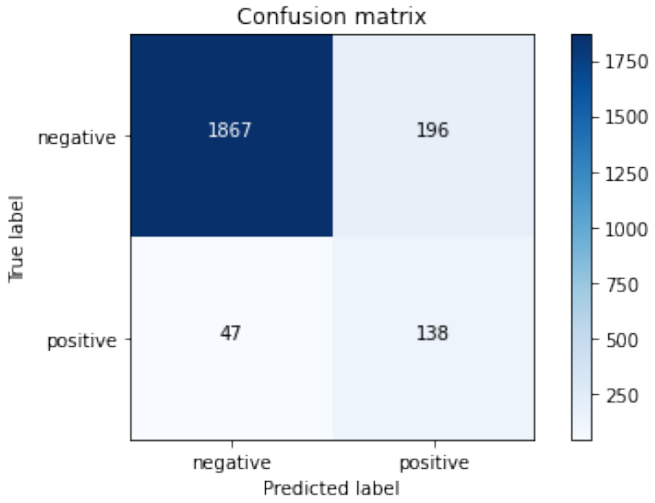

FIGURE 6 – Matrice de confusion de BNN(α = 0,2).

5.3 Analyse discriminante

5.3.1 Motivations

Comme vu dans l'introduction, l'une des caractéristiques problématiques de notre jeu de données est le déséquilibrage entre les classes. Nous avons essayé d'y faire face en utilisant des méthodes dédiées (i.e la méthode SMOTE), nous voudrions néanmoins un modèle prenant en compte et gérant cette disparité. Les modèles d'analyse discriminante de Fisher répondent parfaitement à ce besoin. Ceux-ci incluent dans leur décision les probabilités *a priori*, c'est à dire dans notre cas les tailles relatives des classes.

Nous allons ainsi vérifier la pertinence des modèles d'analyse discriminantes qui, s'ils amènent de bon résultats, nous permettraient d'exploiter le jeu de données brut.

5.3.2 Modèles testés

Nous avons testé les modèles classiques de l'analyse discriminante, à savoir le modèle naïf bayesien de Gauss, l'analyse discriminante linéaire, et l'analyse discriminante quadratique. De plus, notre jeu de données étant composé de variables binaires, nous avons jugé bon d'inclure dans nos essais le modèle naïf bayésien de Bernoulli.

Ce dernier modèle exploite des variables binaires

comme des discrétisations de variables quantitatives suivant une loi normale. Dans ce cas, la variable binaire prendrait la valeur 1 quand la valeur de la variable gaussienne sous-jacente dépasse un certain seuil. Cette hypothèse est d'ores et déjà discutable puisque nos variables indique une présence ou une absence d'agent chimique. Cette interprétation semble affirmer la binarité des variables au détriment de variables gaussienne sous-jacentes. Voyons néanmoins comment se comporte ce test face à notre jeu de données.

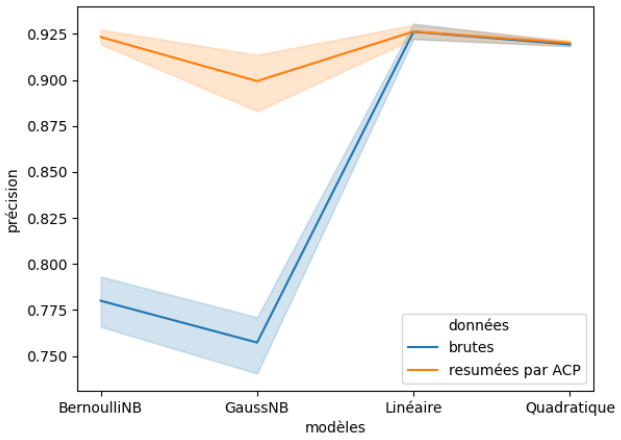

FIGURE 7 – Valeurs de précisions pour les différents modèles sur le jeu de données brut et résumé par ACP

5.3.3 Mise en forme préalable du jeu de données

L'une des limites connues de l'analyse discriminantes et sa sensibilité au nombre de paramètres à estimer. En particulier, les analyses discriminantes linéaires et quadratiques, qui limites les hypothèses sur le jeu de données, y sont très sensibles. Il parait donc avantageux de réduire ce nombre de paramètres en diminuant par exemple le nombres de variables. Nous avons vu dans la partie 3.1 que l'ACP nous permettait de diviser quasiment par deux le nombre de variables tout en gardant 95% de l'information. Toutefois, si cette diminution du nombre de variables peut être avantageux pour les méthodes citées, la perte de précision induite pourrait desservir les autres modèles. Nous allons donc tester ces modèles en comparant les résultat depuis le jeu de données brut et après application de l'ACP. Pour comparer ces modèles, nous visualiserons bien sûr leur précision mais également leur sensibilité puisqu'on a vu que ce score en particulier était important. En effet, la classe "negative" étant largement sur représentée, un modèle pourrait atteindre une précision respectable en classant tous les individus dans cette classe.

5.3.4 Résultats

En faisant tourner les modèles présentés implémentés dans la librairie *sklearn*, et après validations croisées sur 10 itérations, nous obtenons pour la précision le graphe suivant :

On remarque que la précision est globalement bonne puisque elle est dans tous les cas supérieure à 75%. Il est notable que sur le jeu de données brut, les modèles naïfs bayesiens aient tout de même une précision bien moindre. Plus étonnant, l'ACP comble cette différence en augmentant la précision pour ces modèles, la précision des modèles d'analyse discriminantes restent, elles, inchangées. Comme dis dans la section précédente, nous escomptions plutôt l'inverse puisque l'ACP était destiné à améliorer les modèles d'analyse discriminantes.

En fait, les changements apportés par l'ACP sont bien sensibles mai se trouvent ailleurs. Voyons les résultats pour la sensibilité :

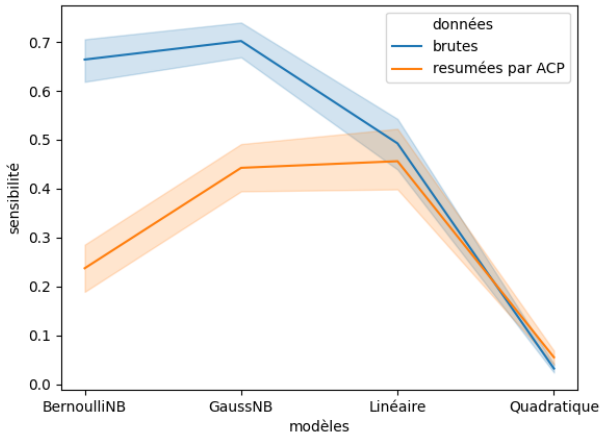

FIGURE 8 – Valeurs de sensibilité pour les différents modèles sur le jeu de données brut et résumé par ACP

On remarque ici que l'ACP a toujours peu d'effet sur les modèles d'analyses discriminantes. On remarque en revanche que la méthode diminue grandement la sensibilité des modèles naïfs bayesien. On en conclut que la perte d'information dûe à l'ACP a pour effet d'empêcher ces modèles de reconnaître les individus positifs et les amènes à tout classer en "negative". Alors leur précision augmente, mais leur pertinence diminue. De plus, l'ACP n'est pas utile pour les modèles d'analyse discriminantes, on peut donc conclure que l'utilisation de l'ACP n'est pas pertinente ici, on s'intéressera aux résultats issus du jeu de données brut.

5.3.5 Interprétation des résultats

Les seuls modèles qui nous permettent d'identifier les individus positifs sont les modèles bayesiens naïfs. On a alors une sensibilité de environ 0.7, c'est à dire environ égale à celle trouvée par d'autres modèles, sans modification préalable du jeu de données. On peut donc conclure que les modèles bayésiens naïfs sont des modèles adaptés au vu du jeu de données.

5.4 Régression logistique

La régression linéaire permet de caractériser les liens entre une variable à expliquer (Y) quantitative et des variables explicatives (X1, X2, X3, ... Xn) au moyen du modèle présenté par la formule en (1). À l'évidence, ce modèle ne s'applique pas aux variables qualitatives et notamment binaires où Y s'exprime en termes de oui/non. Il est donc nécessaire d'utiliser un modèle adapté permettant de relier les variables explicatives à la variable qualitative (Y) à prédire. L'astuce de la régression logistique consiste non pas à modéliser la variable qualitative Y mais la probabilité que celle-ci se réalise. Le modèle logistique (formule en (2)) permet une expression non linéaire, variant de façon monotone entre 0 et 1, de cette probabilité en fonction des variables explicatives (Xi).

$$Y = \beta + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon \qquad (1)$$

$$\begin{aligned} Ln(\frac{P}{1-P}) &= logit(P) \\ &= \beta + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon \end{aligned} \qquad (2)$$

Si la frontière de décision est un polynôme, nous utilisons **PolynomialFeatures (degree = D)**, alors nous avons deux paramètres qui ne sont pas sûrs (D et C). Nous utilisons **GridSearch** pour trouver les meilleurs paramètres.

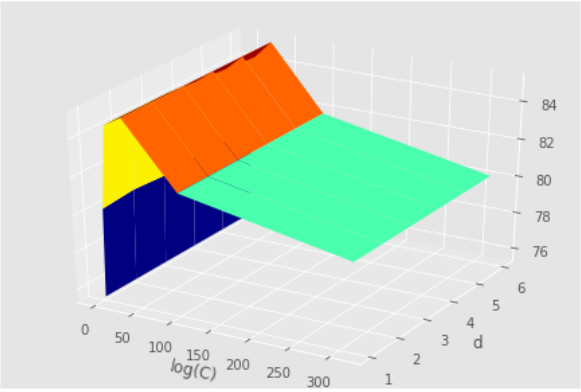

FIGURE 9 – La relation entre les parametres(C,D) et la précision.

La figure 9 suivante est la relation entre les paramètres et la précision. Nous avons constaté que D n'affecte pas la précision, de sorte que la frontière de décision peut être une fonction linéaire. De ce fait nous pouvons réduire considérablement le temps de calcul.
En raison du déséquilibre des données, nous définissons la valeur de class weight[4].

$$weighting = \frac{n\_samples}{n\_classes * n\_samples\_with\_class}$$

Nous obtenons une pondération de classes = 6.08 :0.54.
Lorsque ces pondérations ne sont pas définies, la régression logistique est sur-ajustée, entraînant peu de sensibilité. Une fois ces pondérations définies, bien que la specificité soit réduite, la sensibilité est considérablement améliorée.

	précision	sensibilité	spécificité
class weight= None	0.93	0.39	0.97
class weight= 6.08 :0.54	0.85	0.75	0.86

TABLE 7 – Résultat de Logistic regression

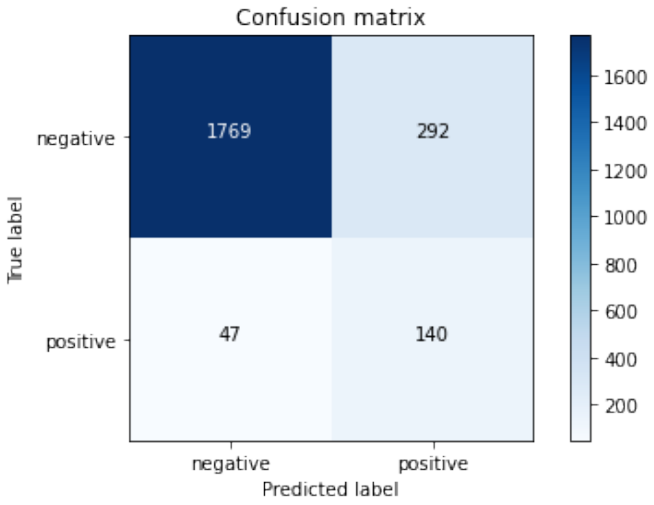

FIGURE 10 – Matrice de confusion de Logistic Regression($class\_weight = 6.08 : 0.54$).

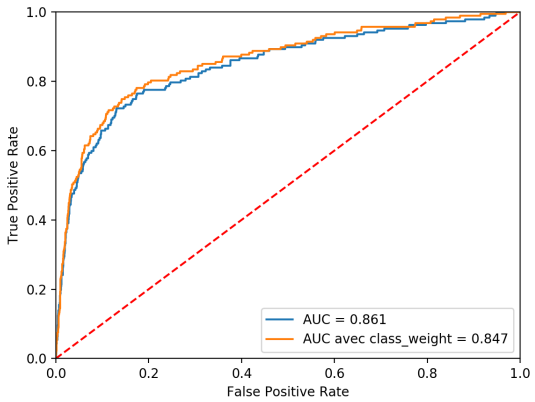

FIGURE 11 – ROC-AUC

Si l'on considère la métrique sensibilité comme seul élément de qualité d'un modèle, la régression logistique avec pondération présente de meilleurs résultats. Si l'on regarde la capacité du modèle à distinguer les deux classes, on peut voir grâce à la 11 que AUC est à peu près identique.

5.5 Arbres

5.5.1 Arbres de décision

Les arbres de décision sont des méthodes d'apprentissages permettant la classification et/ou la régression d'individus. La représentation du modèle est celui d'un arbre binaire. Un arbre binaire est un arbre dont chaque noeud possède un unique prédécesseur et 0 ou 2 successeurs. Le fonctionnement est le suivant. La classification part du jeu de données en entier puis le sépare récursivement dans le but d'avoir les sous arbres les plus homogènes possibles. Ces divisions sont faites jusqu'à ce que chaque feuille soit complètement homogène (pur). La séparation optimale est obtenue en testant toutes les possibilités de division de variables. Pour chaque test un indice est calculé. Cet indice s'appelle l' *indice de Gini* :

$$G(p) = \sum_{k=1}^{g} p_k(1 - pk)$$

La séparation optimale est celle ayant l'indice de Gini minimale. Cette méthode d'apprentissage à l'avantage d'être très intuitive et compréhensible auprès des utilisateurs. Elle nécessite très peu de transformation du jeu de données et s'utilise aussi bien pour de la classification que pour de la régression. Cependant comme tout modèle, elle fait face à des problèmes comme le sûr-apprentissage. Pour rappel le sûr-apprentissage a lieu quand modèle s'adapte trop précisément au jeu de données avec lequel il s'entraîne. Cela implique donc un très bon résultat pour ce jeu de données mais peu aboutir à des résultats médiocres lorsqu'il est appliqué à d'autres individus. Deux méthodes peuvent être utilisées pour gérer ce problème. Tout d'abord le *pré-élagage*, qui permet de limiter la profondeur de l'arbre et le nombre d'individus minimum requis dans un noeud pour y effectuer une séparation. Il existe également le *post-élagage* qui vise à effectuer l'arbre complet puis supprimer les branches qui participent au sur-apprentissage.

Un apprentissage avec pré-élagage a été effectué. L'utilisation de *GridSearchCV* a permis d'obtenir les paramètres optimaux pour appliquer cette méthode. Le modèle est optimale avec une profondeur d'arbre maximal (max_depth) de 11 et un nombre minimum d'individus (min_samples_split) de 49.

	précision	sensibilité	spécificité
qsar	0.92	0.31	0.97
qsar balanced	0.81	0.70	0.82

TABLE 8 – Résultat de DecisionTreeClassifier avec méthode de pré-élagage

On peut voir que ce modèle d'apprentissage donne un meilleur résultat au niveau de la sensibilité quand le jeu est équilibré. Pour rappel la règle de décision pour séparer un noeud se fait à partir de l'indice Gini qui se base sur la proportion de chaque classe dans ce noeud. L'indice de Gini va donc favoriser la pureté de la classe majoritaire au dépend de la classe minoritaire, qui dans le cas des contraintes de pré-élagage (empêchant une par-

tition complète), ne va avoir que peu ou aucune feuille pure.

Une autre faiblesse de l'arbre de décision est sa sensibilité au jeu d'apprentissage reçu. En effet de légères variations peuvent amener à des résultats très différents.

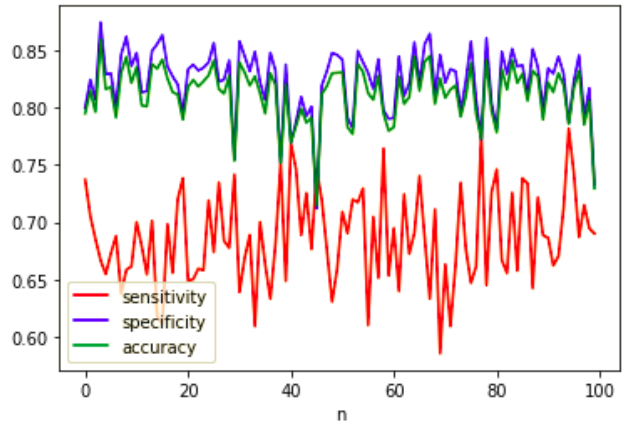

FIGURE 12 – Variation des métriques pour 100 essais

Des modèles d'ensemble comme Bagging ou les Forêts aléatoires permettent d'éviter ce problème.

5.5.2 Bagging

Bagging (Bootstrap aggregation) évite cette dépendance qu'a un arbre de décision seul, en divisant en n sous jeux d'apprentissage le jeu initial. De cette façon, n modèles sont obtenus. Lors de la prédiction du jeu de test, n résultats sont obtenus et la moyenne entre tous ces résultats est faite.

	précision	sensibilité	spécificité
qsar balanced	0.91	0.66	0.93

TABLE 9 – Résultat de Boostrap Aggregation

5.5.3 Forêt aléatoire

Le modèle de Forêt aléatoire va plus loin que le modèle précédent. En effet Bagging permet d'éviter la sensibilité au jeu d'apprentissage mais certaines variables peuvent être corrélées et donc même avec des jeu différents les structures apprises peuvent être similiaire. Ce problème est palié en choisissant, en plus d'un sous jeu d'apprentissage, seulement certains attributs de manière aléatoire. Par défaut la valeur est $\sqrt{n\_features}$ avec n_features le nombre de variables.

	précision	sensibilité	spécificité
qsar balanced	0.92	0.62	0.95

TABLE 10 – Résultat de Random forest

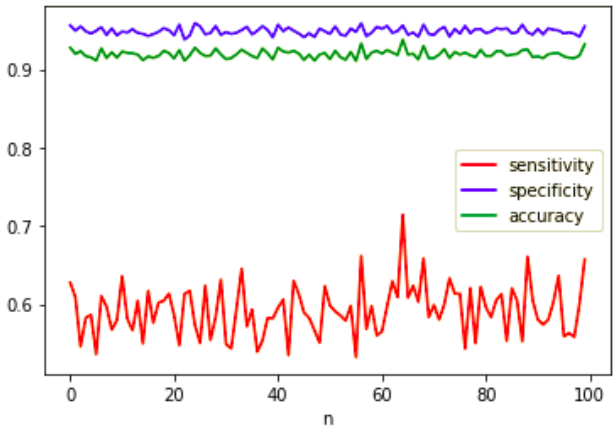

FIGURE 13 – Variation des métriques pour 100 essai

Comme on peut le voir sur 13, les valeurs de spécificité et de précision sont beaucoup plus constantes que 12. La variation de la sensibilité est aussi moins importante mais toujours présente. On constate également que cette valeur est en moyenne inférieure aux valeurs établies par la méthode Arbre de décision.

Si l'on considère la métrique sensibilité comme seule élément de qualité d'un modèle, alors l'arbre de décision présente un meilleur résultat que le modèle Bagging ou Forêt aléatoire. Cependant si l'on considère les modèles en fonction de celui qui arrive le mieux à distinguer les deux classes (*negative* et *positive*) alors les méthodes d'ensemble performent mieux. On peut le voir notamment grâce à la métrique AUC (Area Under the Curve).

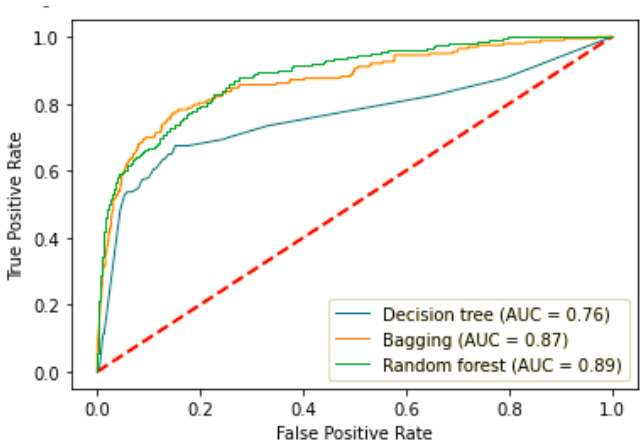

FIGURE 14 – ROC-AUC

6 Conclusion

En terme de prédiction nous pouvons conclure que l'objectif est atteint puisque plusieurs modèles ont démontré une bonne capacité de prédiction. Suivant les usages, la métrique à considérer ne sera pas la même. Pour autant, même la métrique qui nous a paru la plus problématique, la sensibilité, a finalement atteint de bons scores avec les modèles naïfs bayésiens ou suite à des rééquilibrages. Bien sûr, ce score est à mettre en perspective avec l'utilisation que l'on fait des résultats. En fonction de cette utilisation, une précision plus ou moins grande sera attendue. Nous pensons toutefois que le score des différentes métriques est améliorable avec une augmentation de la taille du jeu de données, ce travail a simplement validé que ce jeu se prêtait effectivement à une prédiction par modèles d'apprentissage.

	précision	Sn	Sp
KNN			
qsar	0.92	0.57	0.95
over_sampling	0.92	0.54	0.95
under_sampling	0.70	0.80	0.69
BNN			
qsar	0.94	0.55	0.97
over_sampling	0.94	0.57	0.97
under_sampling	0.89	0.75	0.91
Analyse discriminante			
GaussNB	0.75	0.70	0.75
BerloulliNB	0.78	0.64	0.80
LDA	0.92	0.5	0.97
QDA	0.91	0.003	0.99
Regréssion Logistique			
qsar	0.93	0.39	0.97
qsar balanced	0.85	0.75	0.86
Arbres de décision			
qsar	0.92	0.31	0.97
qsar balanced	0.81	0.70	0.82
Bagging			
qsar balanced	0.91	0.66	0.93
Fôret aléatoire			
qsar balanced	0.92	0.62	0.95

TABLE 11 – Résumé et comparaison des résultats des modèles d'apprentissage

D'un point de vue plus personnel et académique, ce projet nous a permis d'utiliser plusieurs méthodes vues en cours : KNN, Analyse discriminante, Régression logistique... Nous nous sommes également essayés à une méthode que nous n'avions pas étudiée mais qui avait été utilisée par l'équipe de recherche à l'origine de l'étude, BNN. Nous avons pu constaté la pertinence de ces différents modèles dans un problème réel. Nous avons également été amenés à réfléchir au meilleur moyen de valider, ou non, nos résultats comme satisfaisants. Ainsi, si nous avons d'abord été heureusement étonnés de voir l'excellente précision de tous nos modèles, nous avons ensuite remarqué que la problématique se situait ailleurs. Etudier un jeu de données déséquilibré nous a fait prendre conscience de la particularité que peut représenter un problème réel, ici en nous obligeant à analyser la sensibilité plutôt que la précision absolue.

Références

[1] Roberto Todeschini, Davide Ballabio, Matteo Cassotti, and Viviana Consonni. N3 and bnn : Two new similarity based classification methods in comparison with other classifiers. *Journal of chemical information and modeling*, 55(11) :2365–2374, 2015.

[2] imblearn.under_sampling.RandomUnderSampler — imbalanced-learn 0.5.0 documentation. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.under_sampling.RandomUnderSampler.html.

[3] imblearn.over_sampling.smote — imbalanced-learn 0.5.0 documentation. https://imbalanced-learn.readthedocs.io/en/stable/generated/imblearn.over_sampling.SMOTE.html.

[4] sklearn.linear_model.logisticregression — scikit-learn 0.23.1 documentation. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

[5] Davide Ballabio, Francesca Grisoni, Viviana Consonni, and Roberto Todeschini. Integrated qsar models to predict acute oral systemic toxicity. *Molecular informatics*, 38(8-9) :1800124, 2019.

[6] Mohamed El Sanharawi and F Naudet. Comprendre la régression logistique. *Journal français d'ophtalmologie*, 36(8) :710–715, 2013.

[7] Pierre Traissac, Yves Martin-Prével, Francis Delpeuch, and Bernard Maire. Régression logistique vs autres modèles linéaires généralisés pour l'estimation de rapports de prévalences. *Revue d'épidémiologie et de santé publique*, 47 :593–604, 1999.

[8] Toward Data Sciences. https://towardsdatascience.com.