

Projet d'évaluation : Biostatistiques approfondies (Stat3)

Hanae Chouali


January 2022

1 Introduction

Le projet a pour but de construire un modèle permettant de prédire le type de cancer sur la base d'un ensemble de *features*. Plusieurs modèles de classification seront construits et entraînés à cet effet à partir des données dont nous disposons. Nous effectuons par la suite une comparaison entre ces différents modèles en termes de précision et du temps d'exécution, ce qui nous permettra de restreindre notre choix de modèles que nous jugerons les plus pertinents.

2 Données

L'ensemble des données contient deux tableaux. Un premier tableau contient l'ensemble des exemples décrits par des *features*. Il s'agit d'un tableau de dimension 801×20531 où 801 est le nombre d'exemples et 20531 est le nombre de *features* (variables explicatives). Toutes des données du tableau sont numériques et il n'existe pas de données manquantes. Ces données sont donc déjà prêtes à être utilisées dans un modèle d'apprentissage. Le deuxième tableau contient les labels associés à chacun des 801 exemples. Chaque label correspond à un type de cancer : PRAD, LUAD, BRCA, KIRC et COAD. Les données figurant dans le tableau des labels ne sont pas numériques et nécessitent d'être transformées en données numériques avant d'être utilisées dans un modèle d'apprentissage. On associe respectivement les labels numériques 1, 2, 3, 4 et 5 aux types PRAD, LUAD, BRCA, KIRC et COAD.



	Class
sample_0	PRAD
sample_1	LUAD
sample_2	PRAD
sample_3	PRAD
sample_4	BRCA

	Class
sample_0	1
sample_1	2
sample_2	1
sample_3	1
sample_4	3

FIGURE 1 – Tableau des labels

Des statistiques descriptives sur l'ensemble des exemples considérés figurent dans le fichier notebook et ne sont pas rapportées ici en raison du grand nombre des *features*. La figure ci-dessous décrit la distribution des labels.

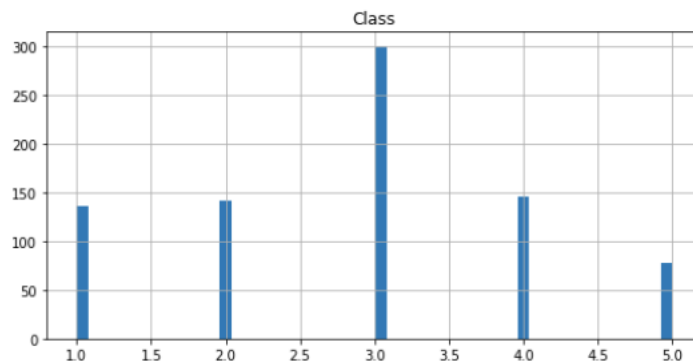


FIGURE 2 – Distribution des labels

On remarque que c'est le type BRCA qui est le plus dominant avec environ 300 cas parmi les 801 exemples considérés tandis que le type COAD est le moins fréquent avec moins de 100 cas.

En utilisant la méthode `TRAIN_TEST_SPLIT` de `SKLEARN.MODEL_SELECTION`, nous effectuons une répartition de notre jeu de données en données d'entraînement (70% de l'ensemble des données) et données de test (30% de l'ensemble des données). Nous avons ainsi la répartition suivante :

	Nombre d'exemples
Données d'entraînement	640
Données de test	161
Ensemble des données	801

La taille de l'échantillon utilisé pour l'entraînement est très petite devant le nombre de features. Cela risque de causer un sur-apprentissage (overfitting) à travers lequel la performance du modèle sur l'échantillon sera très bonne mais sera beaucoup moins bonne dès qu'on l'applique sur un échantillon autre que celui utilisé pour l'entraînement. Dans ce cas, plusieurs solutions peuvent être considérées pour résoudre ce problème. On pourra considérer une pénalisation (de type Ridge ou Lasso) qui permet de pénaliser les grandes dimensions, ou effectuer une analyse en composantes principales qui permet également de réduire la dimension de l'échantillon. La première approche est incluse implicitement dans les différents modèles considérés par la suite, tandis que la deuxième approche sera explicitement abordée dans les prochaines sections.

3 Analyse en composantes principales

L'analyse en composantes principales (ACP) est une méthode qui permet de transformer des variables corrélées entre elles en nouvelles variables, appelées composantes principales, qui sont décorréliées entre elles. Cette méthode permet également de réduire considérablement la dimension des données en sélectionnant les caractéristiques qui capturent le maximum de variance des données. La notion de décorrélation peut être illustrée à travers la figure 4 dans laquelle nous calculons la corrélation entre les composantes principales issues d'une ACP.

Dans un premier temps, nous effectuons une ACP linéaire. Nous considérons par la suite une ACP avec un noyau polynomial qui permet d'améliorer les résultats de l'apprentissage comme nous le verrons par la suite.

3.1 ACP linéaire

Compte tenu de la taille de l'échantillon d'entraînement considéré, le nombre maximal des composantes principales est limité à 640 composantes. Le tableau suivant ainsi que la figure 3 dans l'annexe décrivent le pourcentage de variance expliquée par les nouvelles composantes principales définies en fonction du nombre de composantes choisies.

Nombre de composantes	1	2	3	10	100	640
Pourcentage de variance expliquée	10.85%	19.60%	27.38%	46.91%	73.72	100%

3.2 ACP avec noyau

Nous retenons cette technique car elle permet par la suite d'obtenir une précision meilleure dans la phase de test à la suite de l'apprentissage de quelques modèle, et ce, par rapport à l'ACP linéaire. Plus d'informations sur la méthode peuvent être retrouvées dans le lien suivant : https://en.wikipedia.org/wiki/Kernel_principal_component_analysis.

4 Apprentissage basé sur des méthodes de Machine Learning

Dans cette section, nous construisons différents modèles de Machine Learning que nous entraînons sur la base des données introduites dans la section 2 pour la prédiction de la classe du cancer. Pour chaque modèle, nous effectuons une étude de performances en variant quelques hyperparamètres. Ensuite, nous comparons les performances des différents modèles étudiés selon deux critères : la précision et le temps d'exécution.

4.1 Régression logistique

Nous effectuons dans un premier temps une régression logistique en utilisant les données récupérées sans recours à l'ACP. Nous choisissons de varier l'hyperparamètre C utilisé pour la régularisation, afin de mettre en application le concept de la validation croisée et d'obtenir une meilleure performance.

Le graphe de la figure 5 dans l'annexe mettent en évidence la performance du modèle en fonction de différentes valeurs prises pour le paramètre C .

Le tableau suivant analyse la performance de 3 modèles de régression logistique dans lesquels nous utilisons soit les données brutes, les données obtenues grâce à une ACP linéaire ou bien grâce à une ACP avec noyau polynomial. La performance de chaque modèle est utilisée suivant deux principaux critères : Précision et temps mis par l'algorithme pendant l'entraînement.

	Régression logistique	ACP linéaire + Régression logistique	ACP avec noyau polynomial + Régression logistique
Précision sur données d'entraînement	99.69 %	90.47 %	92.18 %
Précision sur données de test	100%	44.10 %	35.50 %
Temps d'exécution pendant l'entraînement (en s)	114.40	0.59	0.65
Valeur du paramètre C retenu	0.25	1.0	4.0

Le tableau précédent montre que l'utilisation d'une ACP permet de réduire considérablement le temps d'exécution mis pour l'entraînement du modèle. En revanche, l'utilisation de l'ACP réduit largement la précision du modèle sur l'échantillon du test. La même remarque reste valable même en considérant un nombre plus grand de composantes principales. Nous ne validons donc pas l'utilisation de l'ACP dans ce cas.

4.2 Support Vector Machine

Nous considérons dans un premier temps un modèle SVM en utilisant les données récupérées sans recours à l'ACP. Nous choisissons toujours de varier l'hyperparamètre C utilisé pour la régularisation.

Le graphe de la figure 6 dans l'annexe mettent en évidence la performance du modèle en fonction de différentes valeurs prises pour le paramètre C .

Le tableau suivant analyse la performance de 3 modèles SVM de manière similaire que pour la régression logistique.

	SVM	ACP linéaire + SVM	ACP avec noyau polynomial + SVM
Précision sur données d'entraînement	99.84%	84.22%	80.94%
Précision sur données de test	100%	47.83%	41.61%
Temps d'exécution pendant l'entraînement (en s)	154.05	0.72	1.31
Valeur du paramètre C retenu	0.25	4.0	2.0

Les mêmes remarques écrites dans le cas de la regression logistique restent toujours valable dans le cadre du modèle SVM.

4.3 Random Forest

Nous considérons dans un premier temps un modèle Random Forest en utilisant les données récupérées sans recours à l'ACP. Nous choisissons dans ce cas de varier l'hyperparamètre *max_depth* correspondant à la profondeur maximale de l'arbre.

Le graphe de la figure 7 dans l'annexe mettent en évidence la performance du modèle en fonction de différentes valeurs prises pour le paramètre *max_depth*.

Le tableau suivant analyse la performance de 3 modèle Random Forest de manière similaire que pour la régression logistique et le SVM.

	Random Forest	ACP linéaire + Random Forest	ACP avec noyau polynomial + Random Forest
Précision sur données d'entraînement	99.69%	90.62%	92.81%
Précision sur données de test	100%	35.40%	63.35%
Temps d'exécution pendant l'entraînement (en s)	287.60	20.64	31.86
Valeur du paramètre <i>max_depth</i> retenu	11	71	71

Les mêmes remarques écrites dans le cas de la regression logistique et du SVM restent toujours valable dans le cadre du modèle Random Forest.

5 Apprentissage basé sur les réseaux de neurones

Dans cette partie, nous considérons l'architecture suivante pour le réseau de neurones :

- une couche d'entrée ayant 20531 unités,
- une couche dense comportant 1024 unités,
- une activation avec la fonction ReLu,

- une couche dense comportant 256 unités,
- une activation avec la fonction ReLu,
- une couche de sortie comportant 5 unités.

Le réseau de neurones considéré dans cette section permet également d’obtenir une précision de 100% sur les données de test, tout comme les différents modèle de Machine Learning abordés précédemment. D’autre part, l’entraînement du modèle a été effectué sur une durée largement plus petite que celle prise par les autres modèles, ce qui le rend le plus adapté à ce problème, et plus précisément à ce jeu de données.

6 Conclusion

A travers l’étude conduite ci-dessus, l’analyse en composante principale (ACP), malgré avoir réduit considérablement la durée de l’entraînement, a permis d’obtenir une performance largement réduite par rapport à celle obtenue par les modèles initialement sans recours à l’ACP. Il faut noter que ces modèles utilisent implication une pénalisation qui permet d’éviter le risque de l’overfitting lié au fait que la taille de l’échantillon d’entraînement considéré est inférieure au nombre des features. Finalement, l’utilisation d’un réseau de neurones nous a permis d’aboutir à une très bonne performance tout en assurant une durée d’entraînement très petite par rapport aux autres modèles de Machine Learning, et ce, sans recours à l’ACP.

Appendices

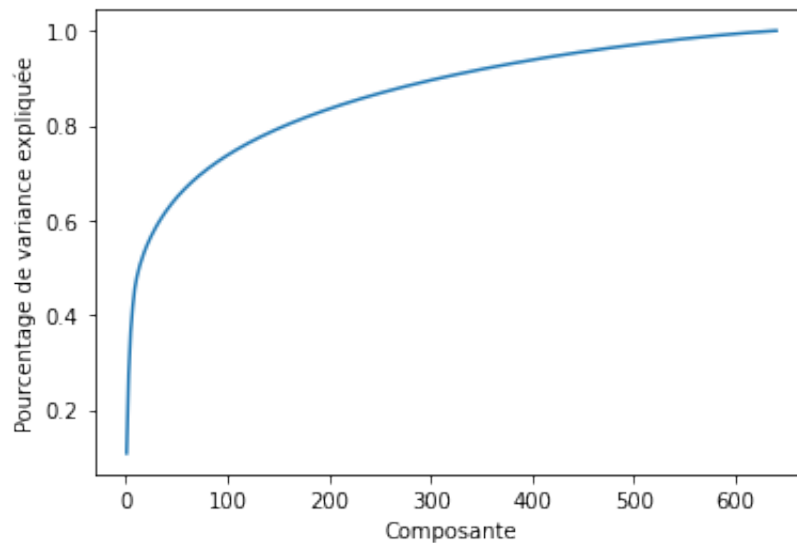


FIGURE 3 – Pourcentage cumulé de variance expliquée en fonction du nombre de composantes retenues dans l’ACP effectuée en section 3.1

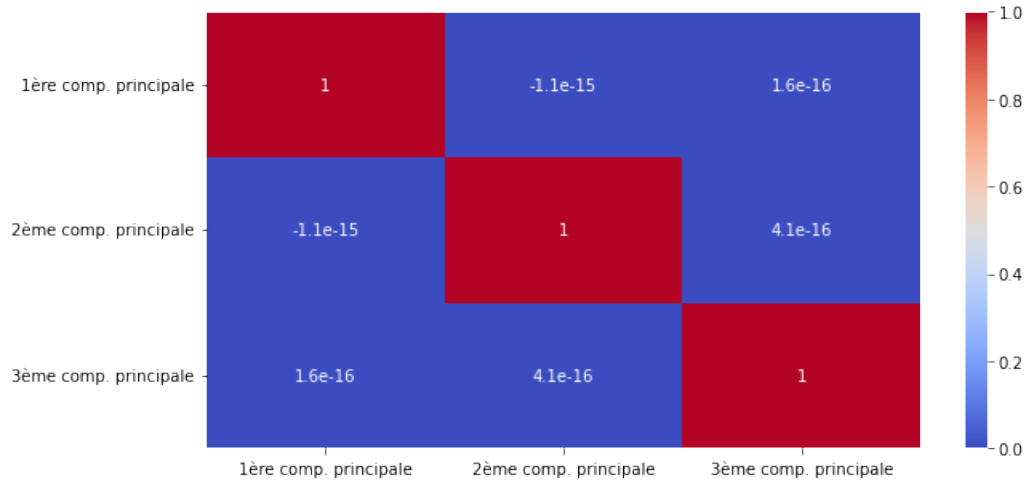
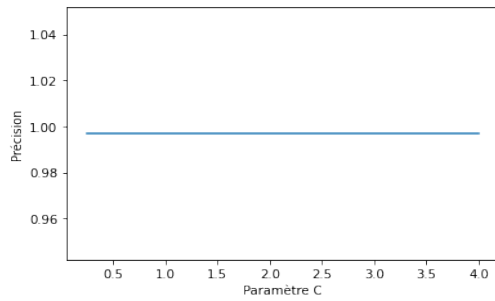
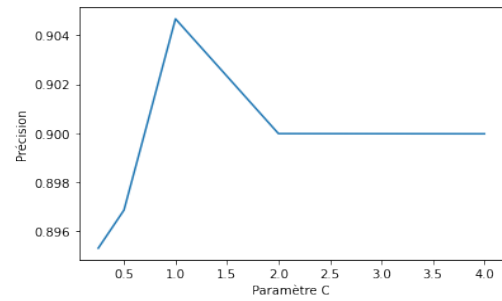


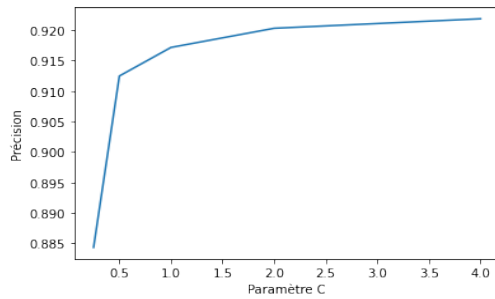
FIGURE 4 – Corrélation entre les trois première composantes principales issues de l'ACP effectuée dans la section 3.1



(a) sans PCA

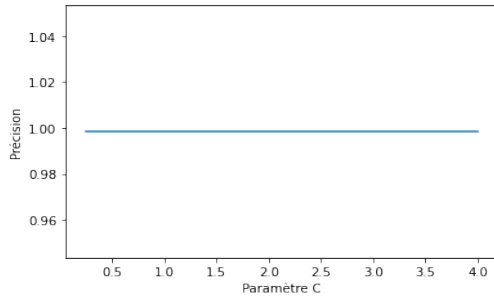


(b) Avec PCA

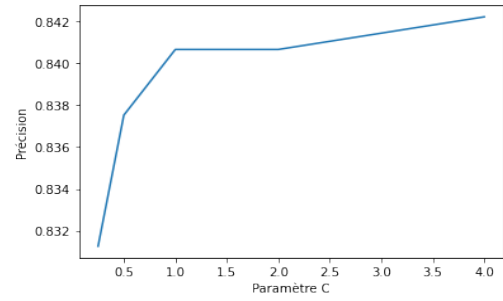


(c) Avec *kernel* PCA

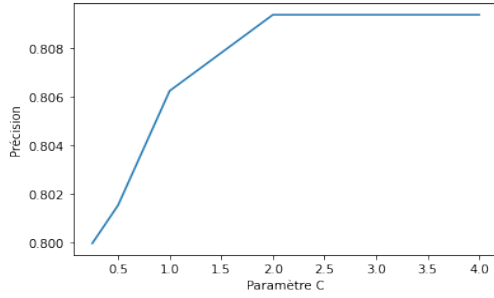
FIGURE 5 – Précision du modèle *Logistic Regression* en fonction du paramètre de régularisation C



(a) sans PCA

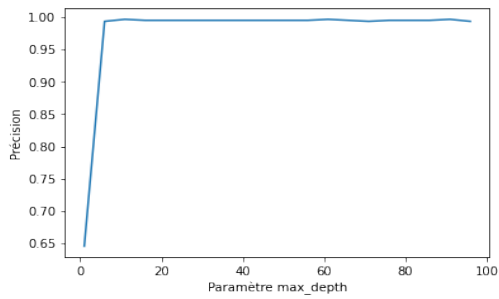


(b) Avec PCA

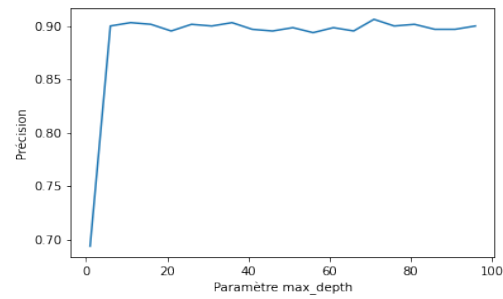


(c) Avec *kernel* PCA

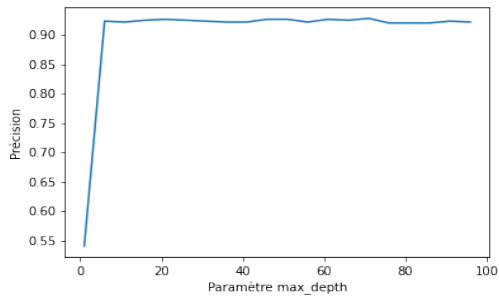
FIGURE 6 – Précision du modèle *Support Vector Machine* en fonction du paramètre de régularisation C



(a) sans PCA



(b) Avec PCA



(c) Avec *kernel* PCA

FIGURE 7 – Précision du modèle *Random Forest* en fonction de l'hyperparamètre max_depth