Statistique en Grande Dimension et Apprentissage

Compte-rendu de travail

Professeur : Fabien Panloup

Introduction

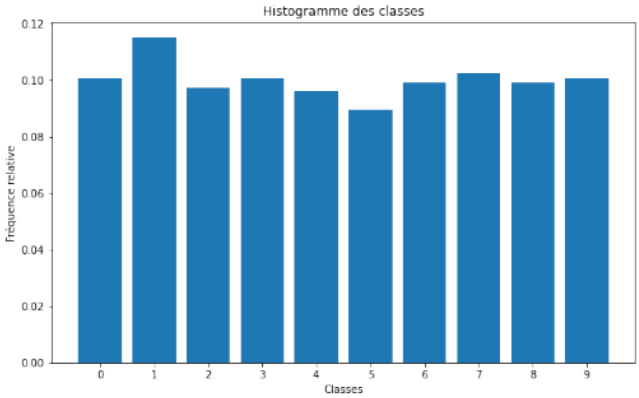
Ce rapport offre un résumé des travaux abordés durant le premier semestre de notre formation. Le master de Data Science d'Angers accorde une importance particulière à la statistique en grande dimension et aux modèles d'apprentissage. L'apprentissage automatique (Machine Learning) constitue un élément central de ce cursus. Dans ce rapport, nous présenterons une sélection de modèles classiques, en mettant l'accent sur leurs principes plutôt que sur les détails techniques du code. Pour approfondir, nous recommandons de consulter les Jupyter Notebooks associés à chaque partie.

Sommaire :

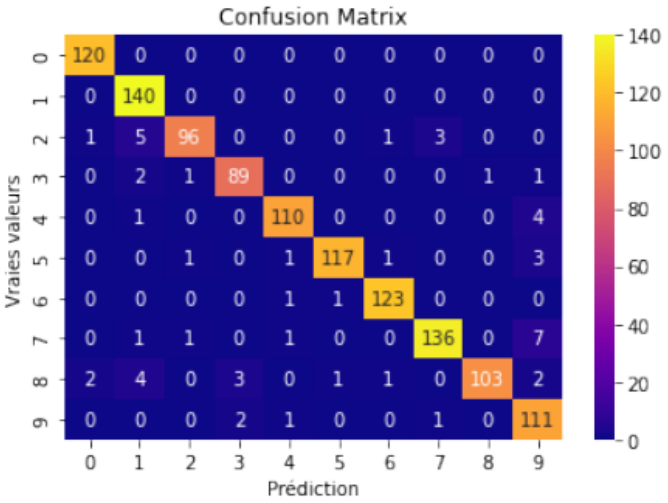
I KNN avec la base de données MNIST

Dans cet exercice, en lien avec les données MNIST, nous avons commencé par charger les modules importants tels que Pandas, NumPy, Matplotlib. Ensuite, nous avons stocké les données à l'aide de Pandas et nous avons fait les premières visualisations de chaque ligne qui correspondent chacune à une photo de nombre écrit à la main. Ainsi, nous avons dû redimensionner les lignes pour les mettre sous forme de matrices de taille 28x28 afin de pouvoir les afficher avec Matplotlib.

Nous avons examiné l'équilibre des classes cibles, afin de vérifier la répartition uniforme des différents chiffres. Cette répartition étant relativement uniforme, nous avons choisi l'*accuracy* comme métrique d'évaluation de nos futurs modèles. L'objet de cet exercice était de mettre en avant les modèles du type KNN (K-plus proche voisins).



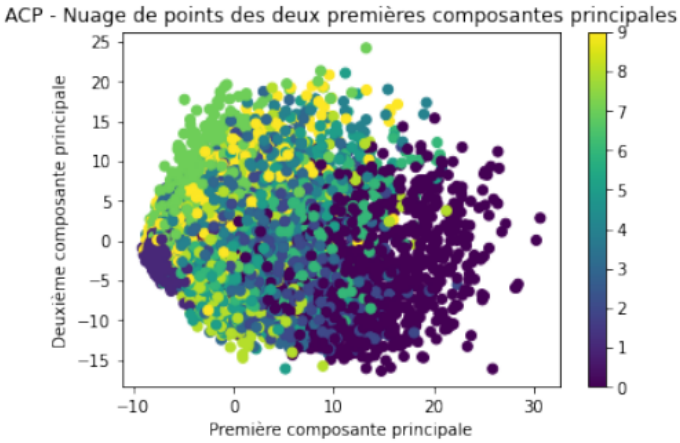
Nous avons utilisé la méthode "*train_ test_ split*" de Scikit-learn pour diviser notre dataset en un ensemble d'entraînement et un ensemble de test. Dans une première partie, nous avons entraîné un modèle naïf avec 10 voisins sur les données d'entraînement et nous l'avons évalué sur les données test. Nous avons obtenu un score de 95 % de réussite autant pour la partie train que pour la partie test. Nous avons par la suite affiché ces résultats dans une matrice de confusion pour mieux comprendre d'où venait les erreurs. Nous avons aussi afficher les images des nombres qui étaient mal prédit par le modèle.



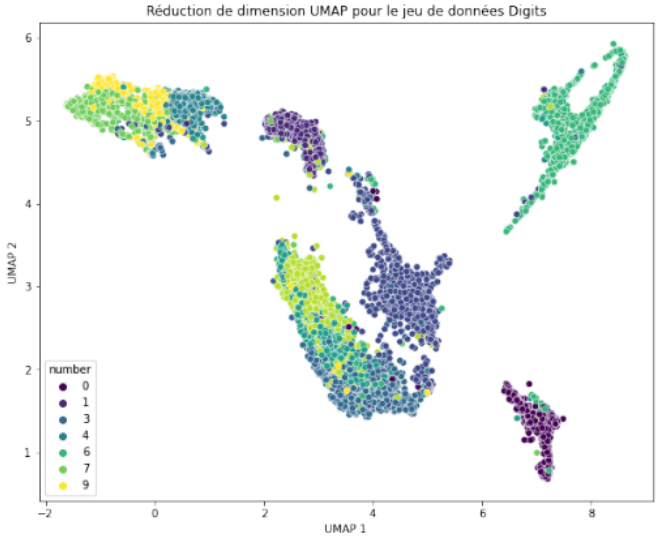
Par la suite, dans le but d'améliorer le score de prédictions, nous avons fait appel à la méthode *'GridSearchCV'* de *Sklearn* qui nous permet de tester plusieurs paramètres pour le nombre de voisins et pour des distances différentes avec le modèle KNN. Nous n'avons pas réussi à augmenter notre score sur la partie test et nous avons obtenu un score de 100 % sur la partie train. Ceci reflète de l'*Overfitting* sur les données d'entraînement. Nous avons donc gardé le modèle initiale à 10 voisins qui nous donne un score de 95% de réussite autant sur les données train que test.

Partie personnelle

- Une approche personnelle au cours de cet exercice a été de faire une ACP pour réduire le nombre de dimensions de ce jeu de données. Nous avons ensuite fait passer les données réduites en dimension dans la méthode *'GridSearchCV'* de *Sklearn* pour trouver un modèle qui colle au mieux aux données d'entraînement. Le modèle proposé a été un KNN à 12 voisins avec la distance euclidienne. L'intérêt de cette réduction de dimension est la rapidité d'exécution, cependant nous obtenons un score de 90 % de réussite. La méthode à ACP ne sépare pas assez les classes de nombre car il s'agit de données non linéairement séparables.



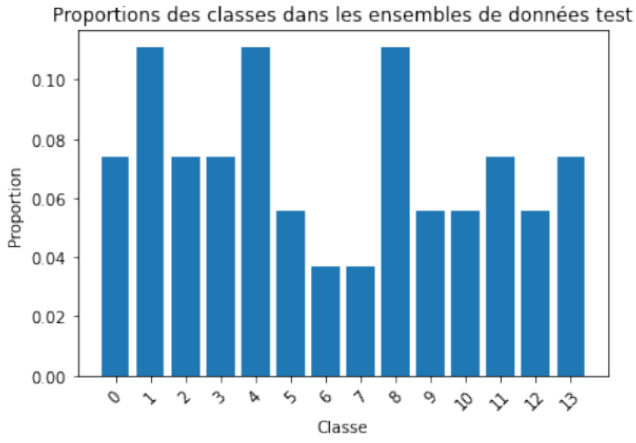
- Enfin, nous avons utilisé une autre méthode de réduction de dimension appelée UMAP. Cette méthode a la particularité de projeter dans un espace de dimension inférieure, des données présentes dans un espace de grande dimension tout en respectant la structure topologique des données initiales. Nous avons présenté cette méthode durant le cours de data analyse, c'est pourquoi nous avons pensé à l'utiliser dans ce projet. Nous obtenons un score de 91 % de réussite avec cette méthode, le temps de calcul est très rapide et nous arrivons tout de même à un score de 92% de réussite sur les données test.



II Classification de cancers avec des arbres de décisions

Dans cet exercice, nous avons également chargé les modules Python essentiels pour le Machine Learning en lien avec les arbres de décision, tels que Pandas et Sklearn. La problématique de cet exercice était de classifier le cancer d'un individu en fonction de ses caractéristiques génétiques, couvrant un grand nombre de gènes. La difficulté réside dans le fait qu'il y a peu d'individus (143) pour un nombre très élevé de colonnes (16 063).

Nous avons d'abord vérifié la répartition des classes dans cet échantillon et constaté qu'elle n'était pas uniforme. Il est apparu que des métriques telles que l'accuracy ne seraient pas suffisantes pour caractériser la performance d'un modèle. Par conséquent, nous avons envisagé d'autres métriques, telles que la précision, le recall et le F1-score.



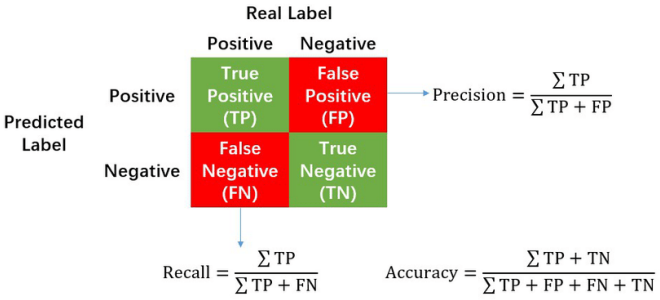
Explications :

Precision : $\mathrm{P} = \dfrac{TP}{TP+FP}$

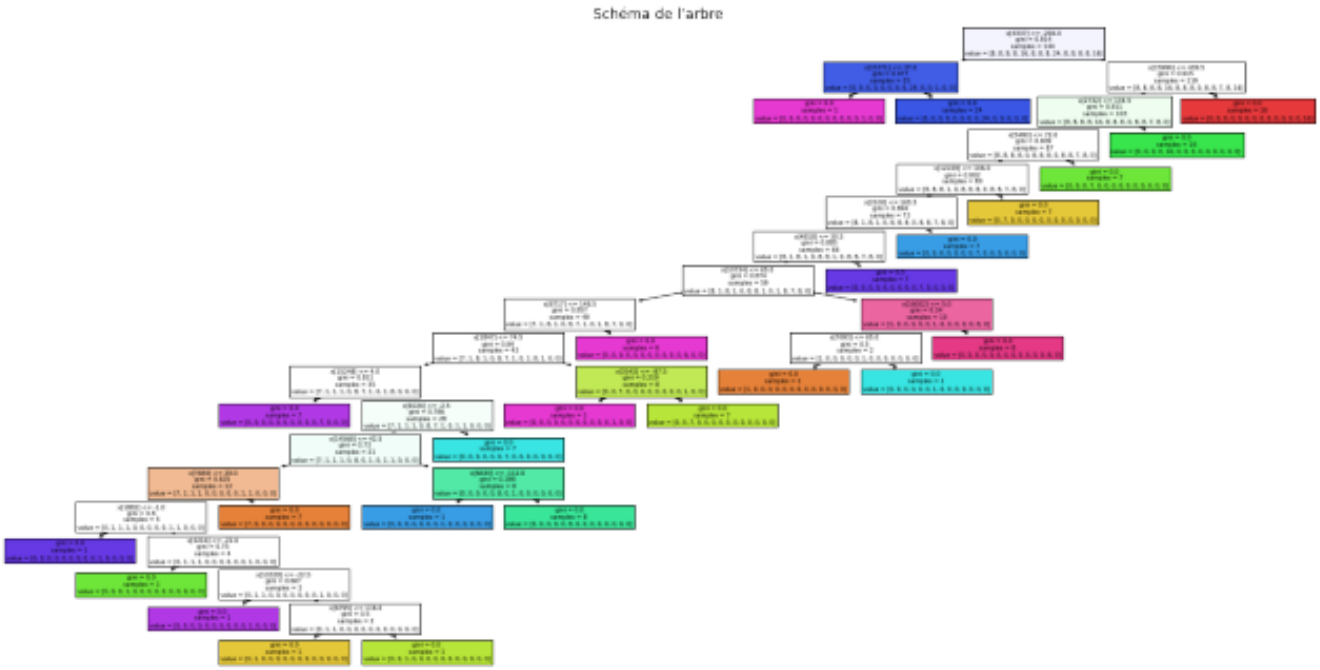
Recall : $\mathrm{R} = \dfrac{TP}{TP+FN}$

F1_score : $\mathrm{F1} = 2 \times \dfrac{P \times R}{P+R}$

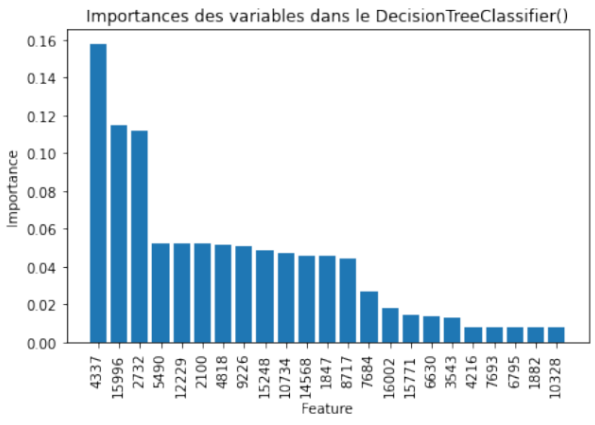
Accuracy_score : $\mathrm{Acc} = \dfrac{TP+TN}{TP+TN+FP+FN}$

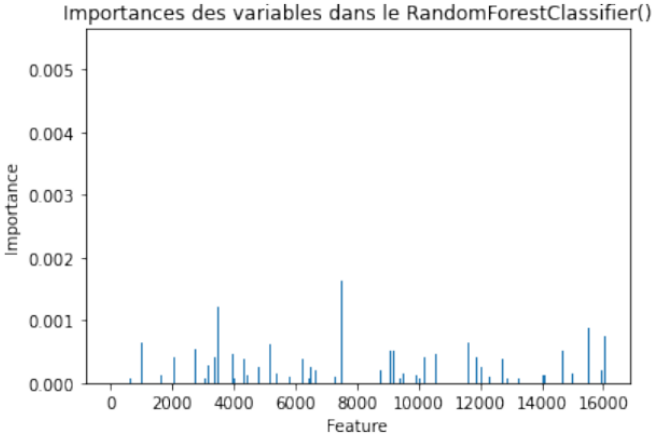


Nous avons créé des fonctions utiles pour l'évaluation de modèles basés sur des arbres de décision. La première fonction, nommée *evaluation*, permet pour un modèle préalablement entraîné, de fournir des prédictions sur les datasets d'entraînement et de test. Elle offre également une visualisation des scores, de la matrice de confusion, et de l'arbre lui-même, si cela est possible.



La deuxième fonction importante, *feature_ importance*, nous permet de connaître, lorsque cela est possible, l'importance des variables dans les décisions prises par le modèle. Selon le nombre de variables explicatives, deux cas se présentent : si le nombre est inférieur à 60, nous pouvons toutes les afficher avec une hauteur proportionnelle à leur importance ; si le nombre dépasse 60, l'affichage devient illisible, et nous avons donc choisi de ne pas montrer ces variables.





Nous avons entraîné des modèles classiques d'arbres de décision, qui nous ont donné des scores d'environ 40%. Ensuite, en utilisant la méthode GridSearchCV, nous avons tenté d'affiner nos prédictions en recherchant les meilleurs paramètres. Nous avons réussi à améliorer légèrement le score de prédiction sur l'ensemble de test, atteignant toutefois 100 % de réussite sur l'ensemble d'entraînement. Cette performance suggère un surajustement (overfitting) sur les données d'entraînement, indiquant un manque de généralisation des modèles

que nous avions préalablement entraînés. En conséquence, nous avons essayé d'implémenter une méthode de Bagging, notamment avec le classifieur Bagging de Sklearn. Nous avons ainsi obtenu des scores légèrement meilleurs, de l'ordre de 62 % de réussite. Cependant il y avait toujours un surajustement car nous atteignions toujours 100 % de réussite sur les données d'entraînement.

Notre idée suivante a été d'expérimenter avec un modèle de type Random Forest, en variant les paramètres habituels. La Random Forest présente des similitudes avec le Bagging, mais introduit un élément aléatoire dans le choix des variables, aussi bien au début qu'à chaque noeud pour la discrimination. En utilisant la méthode GridSearchCV, nous avons testé de nombreux paramètres et avons obtenu un score d'accuracy de 61 % et un F1-score de 57 % sur l'ensemble de test. Nous gardions cependant le problème de sur-apprentissage.

Comme le demandait l'exercice, nous avons également utilisé des méthodes de Gradient Boosting avec des bibliothèques dédiées via Sklearn. Malheureusement, nous n'avons pas réussi à dépasser un score de 62 % d'accuracy pour l'ensemble test.

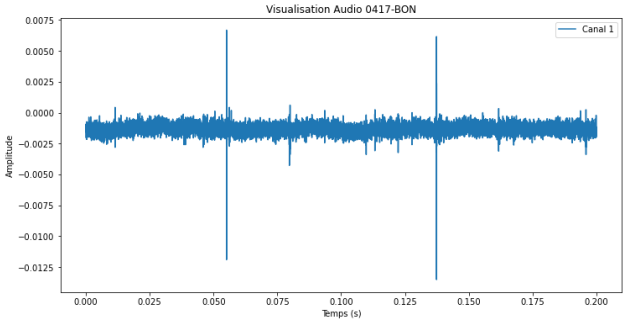
Bilan

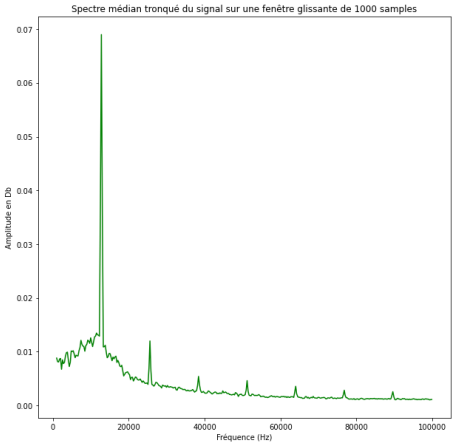
- Le nombre limité d'individus ne nous permet pas de construire un modèle robuste et rend celui-ci très sensible au surajustement, faute de généralisation suffisante.
- La présence de certains cancers spécifiques, tels que le cancer de la prostate ou des ovaires, complique la tâche. Une présélection basée sur des critères tels que le sexe de l'individu pourrait être envisagée.
- Nous avons également tenté une réduction de dimension via l'ACP (Analyse en Composantes Principales), mais le manque d'individus a conduit à de mauvaises prédictions dans les modèles entraînés avec des données réduites en dimension.

III Data challenge : Biosonar - Détection de clics d'Odontocètes

*C*e dernier projet, un data challenge en collaboration avec l'université de Toulon, s'est révélé être l'aspect le plus captivant de ce semestre. Le défi consistait à détecter la présence de biosonars dans des enregistrements audio marins, sur un vaste ensemble de plus de 30 000 échantillons. Notre objectif était d'identifier la présence de dauphins ou de cachalots en utilisant des modèles de Machine Learning et de Deep Learning.

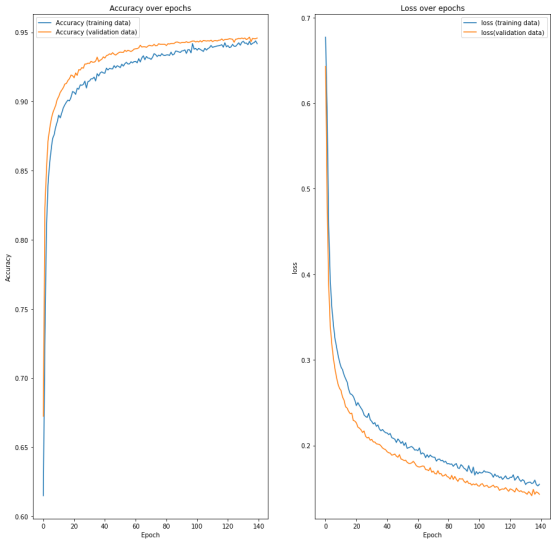
*L*a première étape consistait à maîtriser le jeu de données imposant. Nous avons utilisé la bibliothèque Librosa pour l'analyse préliminaire des données audio. Notre approche s'est divisée en quatre parties distinctes, chacune explorée dans un Jupyter Notebook. Nous avons commencé par examiner les fréquences communes de chaque signal et leur amplitude moyenne en utilisant une transformée de Fourier roulante et d'autres caractéristiques spectrales telles que le centre spectral et la platitude spectrale, extraites avec Librosa.

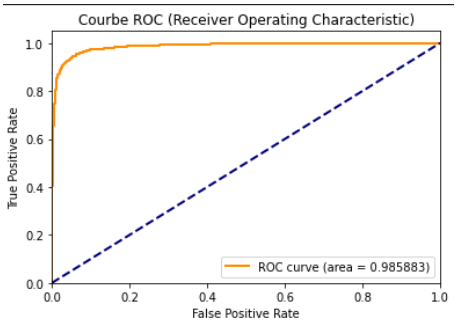




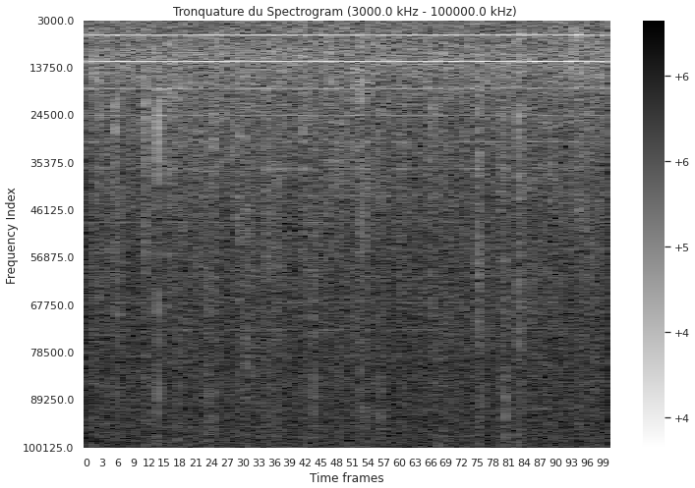
*F*ace à l'ampleur des données, nous ne pouvions pas charger toutes les caractéristiques d'un signal dans un tableau pour le traitement. Nous avons donc synthétisé l'information de chaque enregistrement audio en un nombre restreint de variables explicatives, stockées dans un DataFrame Pandas après s'être assurés de leur homogénéité.

*N*ous avons d'abord utilisé un modèle simple de Random Forest, qui a surpassé le benchmark mais sans atteindre un niveau de performance satisfaisant. Nous avons ensuite élaboré un modèle de Deep Learning utilisant des réseaux de neurones multicouches avec TensorFlow et Keras, parvenant à un score proche des 90 % sur la plateforme.





Nous avons également travaillé avec les spectrogrammes des enregistrements audio, une tâche complexe en raison de leur volume en mémoire. Après avoir ajusté les spectrogrammes pour se concentrer sur l'amplitude des fréquences pertinentes (3KHz - 150KHz) et les avoir transformées en valeurs logarithmiques, nous avons élaboré un modèle de réseau de neurones convolutif (CNN) pour traiter ces images de spectrogrammes. Malgré les longues durées d'entraînement (5h à 6h parfois), nous avons atteint seulement un score de 91 %.



Dans la troisième partie du challenge, nous avons expérimenté avec des modèles hybrides, combinant des données de Fourier et de spectrogrammes, mais ces approches se sont avérées moins performantes, avec un taux de réussite de seulement 76 % et un manque flagrant de capacité à généraliser.

En conclusion, ce data challenge a été extrêmement stimulant. Bien que de nombreux modèles n'aient pas atteint les performances escomptées, l'expérience acquise dans l'implémentation de réseaux de neurones a été très enrichissante. La progression du score de validation a révélé l'importance d'explorer différentes architectures et approches dans le traitement des données complexes.