Université Paris Dauphine

Data Challenge

Rythm: Prédiction de l'âge à partir de l'activité cérébrale durant le sommeil.

Élèves:
Guillaume Ausset
Thomas Dopierre
Clément Rebuffel

Encadrants:
Alexandre d'Aspremont
Fajwel Fogel
Fabian Pedregosa

Contents

Dans ce challenge proposé par Rythm dans le cadre de l'option Python et encadré par Fajwel Fogel et Fabian Pedregosa nous avons été amenés à mettre au point plusieurs modèles dans le but de prédire l'âge d'individus à partir des caractéristiques cérébrales de leur sommeil. Nous avons choisi ce challenge pour son intérêt scientifique: répondre à une problématique scientifique bien définie; pour son intérêt technique: traitement de signaux temporels, faible échantillon, grande dimension et enfin pour nous challenger puisque de nombreux participants, y compris le Master MATIS de SupElec utilisent ce challenge. Après avoir développé et testé de très nombreux modèles, notre modèle final (au moment de la rédaction) obtient un score de `17.2888` sur le leaderboard publique ce qui nous place en $\mathbf{1^{ère}}$ position. Le modèle final (provisoire) est:

Algorithme: Ensemble de CNNs
Prétraitements: Filtre passe bande (Butterworth)
Modele 1: `EEGNet v1`, Train pré-traité, Test Pré-traité
Modele 2: `EEGNet v1`, Train pré-traité, Test brut
Modele 3: `EEGNet v1`, Train brut, Test brut
Optimiseur: Adam
Data Augmentation: Patchs aléatoires
Ensembleur: Moyenne arithmétique

1 Présentation du challenge

Ce challenge proposé par Rythm consiste à déterminer l'âge d'un individu à partir d'un EEG de sommeil profond de 5 minutes ainsi qu'un hypnogrammes des différentes phases de sommeil durant une nuit.

1.1 Features

Notre échantillon d'apprentissage est de petite taille et très grande dimension, ce qui rend ce challenge particulièrement intéressant, mais difficile.

Chacun des individus possède en réalité 2 attributs:

Tout d'abord un EEG de 5 minutes de sommeil profond, enregistré à $250Hz$. Cet EEG peut avoir été enregistré soit par des casques de type 0 soit des casques de type 1. En théorie seul le gain du casque diffère, mais aucune stratégie de renormalisation n'a en pratique fonctionné et comme nous le verrons il semble y avoir des différences fondamentales entre `Device 0` et `Device 1`.

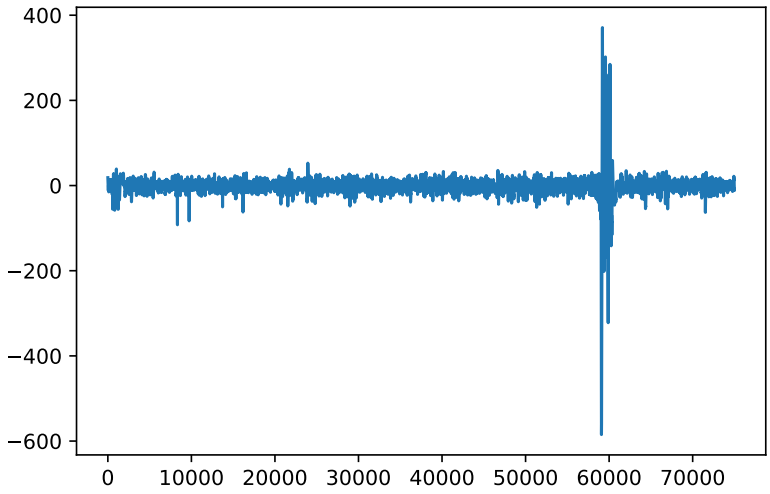


Figure 1.1: EEG de l'individu 0

Les EEGs présentent a priori trois défauts importants:

1. Le signal est de très mauvaise qualité, il y a un grand nombre de valeurs aberrantes.
2. Le casque enregistre 2 canaux, mais n'ont n'avons que la moyenne de disponible. Cela exclut certaines techniques telles que ICA et nous fait perdre une quantité non négligeable d'information.
3. `Device 0` et `Device 1` et ne sont pas comparables.

Un hypnogramme correspondant à une série temporelle de taille variable indiquant les états de sommeil de l'individu durant une nuit. Cet hypnogramme à été construit par un algorithme de scoring du sommeil suivant les recommandations standard de la littérature.

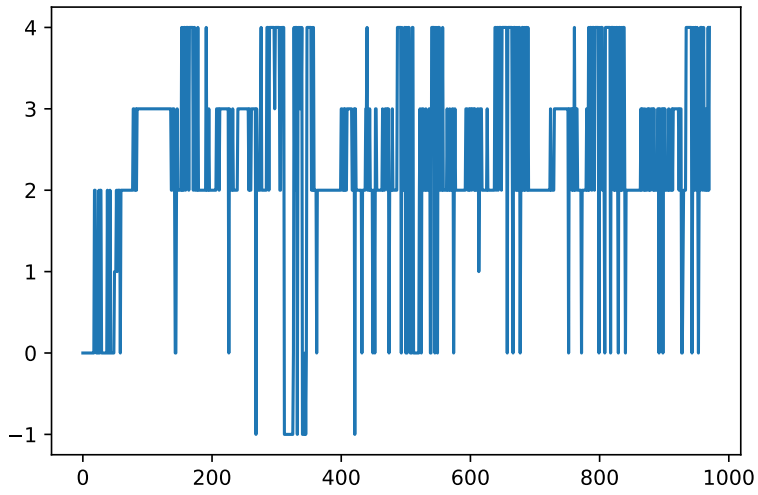


Figure 1.2: Hypnogramme de l'individu 0

1.2 Objectif

Il est également intéressant de visualiser les variables objectifs. Il s'agit ici d'un problème de régression.

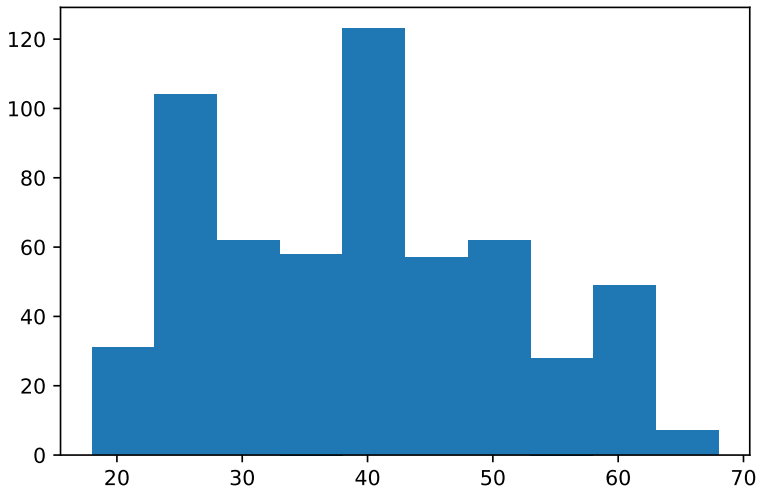


Figure 1.3: Distribution des âges

On voit ainsi qu'il faudra traiter avec grand soin les prédictions au-delà de l'intervalle $[18, 68]$.

1.3 Perte

La perte utilisée pour le challenge est le Mean Absolute Percentage Error (MAPE) défini par:

$$\texttt{MAPE}(\mathcal{D}) = \frac{100}{n}\sum_{i=1}^{n}|\frac{y_i - \hat{y}_i}{y_i}|$$

Le MAPE n'est pas régulier, mais peut être régularisé facilement comme nous le verrons. Dans le cas d'une simple descente de gradient, on peut directement utiliser le MAPE puisque celui-ci admet une sous-différentielle en tout point.

Un point important à noter: le MAPE a tendance à surestimer les erreurs hautes par rapport aux erreurs basses. On a donc plutôt intérêt à sous-estimer nos prédictions (par exemple en prédisant $\lfloor \hat{y} \rfloor$, ou en arrondissant de manière intelligente par exemple en utilisant une fonction du type puisque l'on sait que les âges seront toujours des entiers:

```
def round_(x, c = 0.5):
    xi = np.floor(x)
    return xi+1 if x-xi > c else xi
```

En effet bien qu'il faille sous-estimer les âges on peut considérer qu'une prédiction de 21.9 signifie que 22 est l'âge prédit avec une grande certitude, si l'on trace le MAPE en fonction du point de coupure choisi on trouve a chaque une courbe du type:

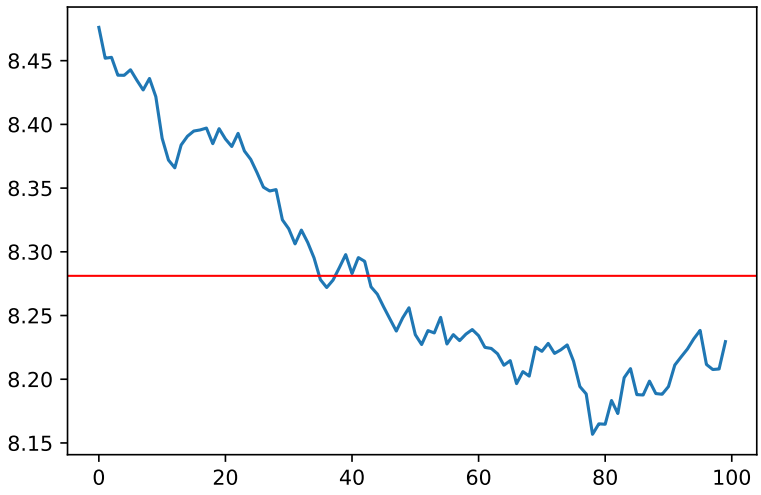


Figure 1.4: MAPE en fonction de la coupure

On voit donc qu'il est intéressant d'arrondir a $x.9$ (la droite rouge représentant le score si l'on prédisait les âges bruts)

2 Littérature

Nous avons commencé par effectuer une rapide revue de la littérature sur d'éventuels liens entre les EEGs durant le sommeil et l'âge.

Ainsi, suivant les recommandations de Amin et al., 2015 nous avons tenté d'extraire les composantes wavelet des EEGs, sans aucun succès. Une autre approche qui nous paraissait prometteuse pour créer des features puisque nous possédions 2 séries temporelles était l'utilisation de la signature de chemins rugueux tels que dans Chevyrev and Kormilitzin, 2016 et Lyons, 2014. Bien que l'on ait réussi à obtenir un faible signal sur les hypnogrammes avec cette méthode le résultat n'a pas été concluant et nous avons abandonné l'idée.

Nous avons également trouvé dans la littérature plusieurs papiers explorant le lien entre âge et complexité du signal cérébral. La définition de complexité changeant entre les différents papiers nous avons néanmoins décidé d'implémenter les différentes méthodes présentées dans Krakovská et al., 2015 et Zappasodi et al., 2015. La encore, bien que faisant mieux que le hasard nous n'avons néanmoins pas retenu ces features, les performances n'étant pas entièrement satisfaisantes.

Bien que les différentes méthodes présentent dans la littérature n'aient pas portées leurs fruits cela nous a permis de s'assurer qu'il n'était pas vain d'utiliser l'EEG, feature de très mauvaise qualité dans nos données, pour essayer de réussir ce challenge. C'est pour cette raison que nous avons dans notre premier "bon" modèle que nous présentons ensuite choisi de construire manuellement des features inspirées des papiers précédents.

Un des aspects importants de la littérature que nous avons néanmoins retenus est l'importance des basses fréquences pour la prédiction de l'âge. On peut le vérifier aisément en analysant les coefficients retenus par un `LASSO` entraîné sur le spectre des EEGs:

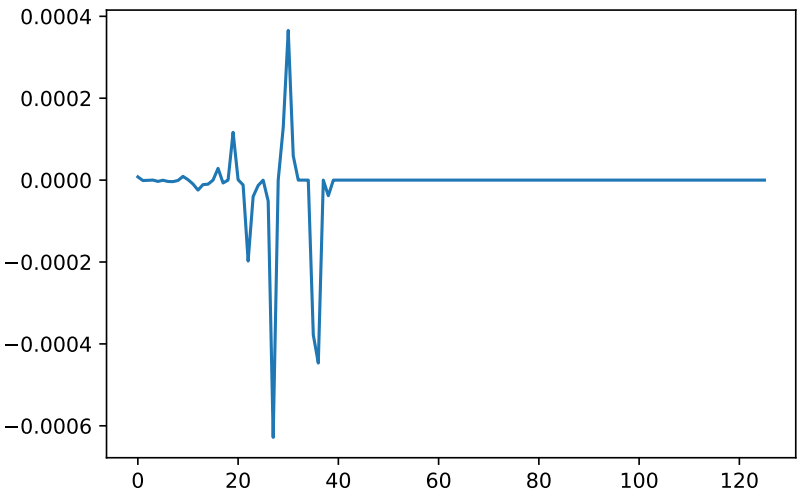


Figure 2.1: Coefficients dans le `LASSO`

3 Premier modèle

3.1 Hypnogrammes

Cette section a pour but d'explorer le traitement des hypnogrammes.

Un hypnogramme est une série temporelle de taille variable, prenant ses valeurs dans $\{-1, 0, 1, 2, 3, 4\}$. Les entiers positifs correspondent à l'état du sommeil du patient (0 étant éveillé jusqu'à 4 pour sommeil profond) et -1 sert de repère pour quand il était impossible d'identifier l'état de sommeil.

3.1.1 Difficultées rencontrées et solutions apportées

À la vue des différents hypnogrammes, plusieurs difficultés nous sont apparues:

- **La longueur des hypnogrammes est différente pour toutes les observations.** Cette difficulté est majeure, car elle ne nous permet pas d'appliquer directement des modèles sur l'hypnogramme. De plus, il est difficile de raisonner en numéraire (par exemple, faire une feature qui donne le nombre de 0 dans l'hypnogramme), car un hypnogramme plus long aura plus d'occurrences de chaque état (en moyenne).

  SOLUTION
  Chaque feature numéraire sera calculée en % de la longueur de l'hypnogramme

- **Les hypnogrammes ont beaucoup de discontinuités et sont très impropres.** Nous avons observé que les hypnogrammes avaient énormément de sauts qui nous semblaient peu justifiés (par exemple, un 0 au milieu des 4). En effet, il semble étrange qu'un individu soit en sommeil profond puis se réveille brusquemment et repasse en sommeil profond juste après.

  SOLUTION Nous avons "lissé" l'hypnogramme grâce à une méthode des K plus proches voisins (en pratique, $K = 5$), nous changeons la valeur uniquement lorsque $p > 0.5$ c'est a dire lorsque l'on est sur avec une haute probabilité que la valeur doit être changée.

- **Des débuts à 0 et des fins à 0.** Nous avons observé que les hypnogrammes commencent et se terminent souvent par des séries 0 (état éveillé). Or, si nous ajoutons des suites de zéros au début et à la fin cela risque de fausser nos features.

  SOLUTION
  Cette étape est effectuée une fois la correction précédente appliquée. Nous retirons simplement les 0 de tête et de queue. Nous sauvegardons néanmoins la taille de ces périodes pour nous en servir comme feature.

3.1.2 Features extraites

Grâce à l'hypnogramme, nous avons pu extraire quelques features qui nous paraissaient pertinentes. Les features calculées en nombre sont annotées #, et celles en pourcentage de la taille de l'hypnogramme, %:

- **Chaine de Markov.**[#] Une manière intéressante d'approcher l'hypnogramme est de le considérer comme une chaîne de Markov : les états de sommeil sont sans doute liés, et la probabilité de passer de l'un à l'autre peut refléter l'âge du sujet. Nous avons donc établi une matrice de taille 5x5 ou l'élément (i, j) est la fréquence de passer de i à j :

$$K(i,j) = \frac{\#\{t, x_t = i, x_{t+1} = j\}}{\#\{t, x_t = i\}}$$

- **Plus longue séquence et position.**[#,%] Pour chaque état possible i, nous avons calculé la taille de la plus longue séquence de cet état dans l'hypnogramme (en pourcentage).

$$k^* = \frac{1}{T} \max\{k, \exists t, [x_t, ..., x_{t+k-1}] = [i, ..., i]\}$$

  Nous avons également gardé la position où démarre cette séquence.

- **Nombre de sauts.**[#,%] Nous avons calculé le nombre de fois où le signal saute, c'est-à-dire le nombre de changements d'état du sommeil.

- **Variation Totale.**[#,%] Nous avons calculé la variation totale d'un signal, c'est-à-dire la somme des variations. Par exemple, la variation totale de $[1, 2, 5, 1]$ vaut $|2 - 1| + |5 - 2| + |1 - 5| = 8$.

- **Longueur de l'hypnogramme**[#]
- **Temps passé dans les différentes phases**[#,%]
- **Nombre de reveils**[#,%]

3.2 Traitement des EEGs

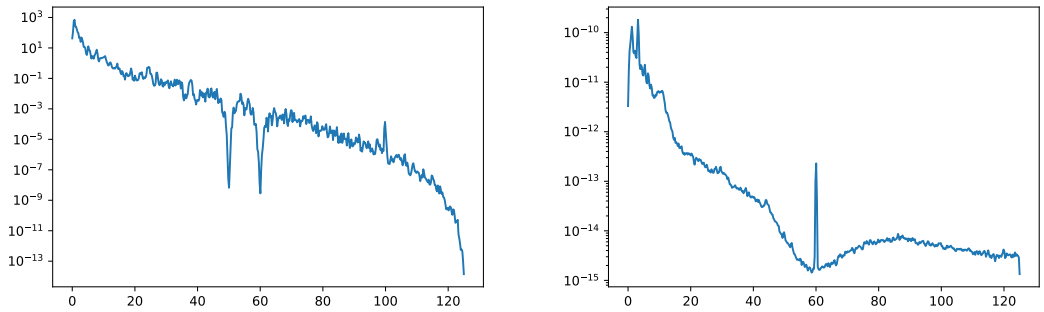
Nous avons vu dans la section littérature qu'il y avait sûrement de l'information à extraire du signal EEGs. Notamment, il devrait y avoir une diminution de la puissance spectrale avec l'âge (il y a néanmoins un bémol: la puissance spectrale est également plus faible chez les hommes à cause d'un crane plus épais, nos données ne contiennent pas la variable `Sexe` qui pourtant paraît primordiale dans le cas d'une étude médicale), nous avons donc décidé de passer dans le domaine fréquentiel et de calculer la puissance spectrale des différentes bandes éventuellement présentes durant le sommeil:

- **Delta:** $<$ 4Hz, sommeil à ondes lentes.

- **Theta:** $4 - 7$Hz, endormissement.
- **Alpha:** $8 - 15$Hz, relaxation, fermeture des yeux.
- **Beta:** $16 - 31$Hz, pensée active, anxiété.

Nous avons donc construit le spectre des EEGs des différents individus par méthode de Welch (une simple FFT avait une variance bien trop élevée, le spectre résultant étant trop bruité pour être exploité).

La encore nous notons une différence significative entre les `Device 0` et `Device 1`:



(a) Spectre d'un individu 0

(b) Spectre d'un individu 1

Nous observons la même chose sur tous les individus 0 ou 1, le spectre semblant presque inversé chez les individus 1.

Puisque selon Rythm seul le gain différe entre `Device 0` et `Device 1` nous avons tenté plusieurs stratégies de renormalisation: renormalisation dans le domaine temporel, renormalisation dans le domaine fréquentielle, renormalisation par un CNN... Aucune technique n'a fonctionné et garder les signaux d'origines nous a donné les mêmes résultats, probablement parceque garder les `Device 1`, faibles en nombres et au comportement étrange, à un effet de régularisation très fort qui empêche nos modèles de sur-apprendre.

3.3 Création d'une forêt boostée

Après avoir testé un grand nombre de modèles simples nous nous sommes concentrés sur le boosting avec comme booster de base des arbres de décisions pour leur très bonnes performances et leur capacité a peu overfitter.

Pour cela nous avons utilisé `XGBoost` et puisque notre critère a optimiser est le MAPE nous avons décidé d'écrire notre propre fonction de perte pour nous adapter au problème. En effet `XGBoost` a besoin de la hessienne et le MAPE n'est pas 2 fois dérivable, nous avons donc décider d'utiliser une approximation régulière du MAPE en notant que:

$$|x| = \max(x, -x) \simeq \log\left(e^{x} + e^{-x}\right) - \log 2$$

Notre perte est donc:

```
def mape_obj(y_pred, dtrain):
    y_true = dtrain.get_label()
    rerr = np.divide(y_true - y_pred, y_true)
    grad = 100*np.divide(np.tanh(-rerr), y_true)
```

```
    hess = 100*np.divide(np.divide(4 * np.square(np.cosh(rerr)),
    np.square(np.cosh(2 * rerr) + 1)), np.square(y_true))
    return grad, hess
```

4 Modèle Final

Ne parvenant que péniblement à améliorer notre modèle précédent nous nous sommes alors tournés vers la création d'autres modèles afin de réaliser un ensemble de modèles. Afin d'obtenir un ensemble de qualité, il est nécessaire d'avoir des modèles très différents.

Puisque la majorité de nos performances venait jusqu'à présent de l'exploitation de l'hypnogramme et d'un modèle de forêt aléatoire boostée, il nous a semblé judicieux de nous concentrer sur la création d'un modèle portant sur les EEGs en particulier.

La littérature ainsi que les résultats précédents semblaient indiquer qu'une majorité de l'information se trouve dans le domaine fréquentiel, or la plupart des opérations que l'on a effectuées jusqu'à présent (multiplication dans le domaine fréquentiel principalement) peuvent s'exprimer comme des convolutions dans le domaine temporel. Nous avons donc décidé de porter nos efforts sur la construction à base d'un modèle de réseau de neurones convolutionels.

Le problème principal dans notre cas qui aurait dû en théorie rendre impossible ou presque l'utilisation d'un tel modèle est la très faible taille de notre échantillon, des échantillons plusieurs milliers de fois plus grandes au moins étant généralement requis. Pour pallier ce problème nous avons donc décidé d'avoir recours à de l'enrichissement artificiel des données: nous sélectionnons au hasard des patchs de taille 16384, ce qui donne en théorie un échantillon virtuel de taille 581x58616 $\simeq$ 34M (avec évidemment un très fort recoupement entre certains échantillons).

Il nous a ensuite fallu concevoir une architecture assez riche pour apprendre, mais assez parcimonieuse pour ne pas sur apprendre, il est facile de vérifier par exemple qu'en prenant notre modèle final de base `EEGNet` et en rajoutant une couche ou des filtres on parvient très rapidement à descendre vers 0 d'erreur d'apprentissage, mais 28 d'erreur de test (soit juste prédire la moyenne) alors qu'en simplifiant même très peu le modèle l'erreur d'apprentissage ne descend jamais sous 27. Il semble il y avoir une fenêtre de modèles répondant au compromis très mince.

Enfin afin de limiter le sur-apprentissage nous avons largement fait usage de régularisation comme le `DropOut` ou la `Batch Normalization` (qui par ailleurs a pour effet d'accélérer grandement l'apprentissage).

L'architecture que nous avant retenue est:

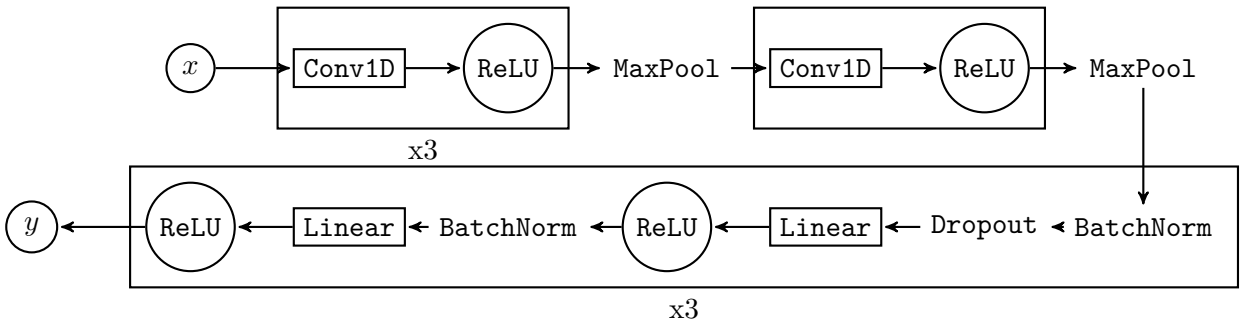


Figure 4.1: `EEGNet`

Ou plus precisement avec les parametres:

```
eegnet (
  (conv1): Conv1d(1, 3, kernel_size=(5,), stride=(1,))
  (conv2): Conv1d(3, 3, kernel_size=(16,), stride=(16,))
  (conv3): Conv1d(3, 9, kernel_size=(7,), stride=(1,))
  (maxpool1): MaxPool1d (size=3, stride=2, padding=0, dilation=1, ceil_mode=False)
  (conv4): Conv1d(9, 9, kernel_size=(5,), stride=(1,))
  (maxpool2): MaxPool1d (size=12, stride=8, padding=0, dilation=1, ceil_mode=False)
  (batchnorm4): BatchNorm1d(558, eps=1e-05, momentum=0.1, affine=True)
  (dropout1): Dropout (p = 0.5)
  (fc1): Linear (558 -> 512)
  (batchnorm5): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True)
  (fc2): Linear (512 -> 512)
  (dropout3): Dropout (p = 0.5)
  (batchnorm6): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True)
  (fc3): Linear (512 -> 512)
  (batchnorm7): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True)
  (fc4): Linear (512 -> 512)
  (dropout5): Dropout (p = 0.5)
  (batchnorm8): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True)
  (fc5): Linear (512 -> 512)
  (batchnorm9): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True)
  (fc6): Linear (512 -> 1)
)
```

Nous avons ensuite entrainé plusieurs variantes avec ou sans pré traitement des données puis créé un ensemble pour parvenir à notre résultat actuel.

5 Conclusion

Après de nombreux changements et améliorations de modèle, notre modèle final s'avère être un réseau de neurones convolutionnel. Nous n'avions au début pas tenté cette approche de peur que la très faible quantité de données combinée au très grand nombre de paramètres des réseaux de neurones ne soit synonyme de sur apprentissage. Il s'avère au final qu'en combinant une procédure de data augmentation efficace avec plusieurs méthodes de régularisation nos CNN sont nos modèles qui sur apprennent le moins! Alors que notre but principal étant d'ajouter des modèles variés a notre forêt pour réaliser un ensemble, les performances des CNN sont telles que l'ensemble final n'inclue pas la forêt. De plus l'utilisation de CNNs simplifie énormément le *pipeline* de création de modèles et prédiction en n'utilisant plus que les données brutes, sans feature engineering. L'inconvénient étant bien sur la colossale augmentation de la puissance de calcul nécessaire à la création du modèle. Néanmoins la petite taille des réseaux garantit toujours une prédiction extrêmement rapide. Étant donné plus de temps, nous aurions aimé tester plus d'architectures de réseau et de moyens de prétraitement afin de réaliser un ensemble de meilleure qualité.

Bibliography

Amin, Hafeez Ullah, Aamir Saeed Malik, Rana Fayyaz Ahmad, Nasreen Badruddin, Nidal Kamel, Muhammad Hussain, and Weng-Tink Chooi

2015 "Feature extraction and classification for EEG signals using wavelet transform and machine learning techniques", *Australasian Physical & Engineering Sciences in Medicine*, 38, 1, pp. 139-149, ISSN: 0158-9938, DOI: 10.1007/s13246-015-0333-x, http://link.springer.com/10.1007/s13246-015-0333-x. (Cited on p. 7.)

Chevyrev, Ilya and Andrey Kormilitzin

2016 "A Primer on the Signature Method in Machine Learning", p. 45, arXiv: 1603.03788, http://arxiv.org/abs/1603.03788. (Cited on p. 7.)

Krakovská, A., R. Škoviera, G. Dorffner, and R. Rosipal

2015 "Does the Complexity of Sleep EEG Increase or Decrease with Age ?", *Measurement*, pp. 77-80. (Cited on p. 7.)

Lyons, Terry

2014 "Rough paths , Signatures and the modelling of functions on streams", 291244, pp. 1-24, arXiv: arXiv:1405.4537v1. (Cited on p. 7.)

Zappasodi, Filippo, Laura Marzetti, Elzbieta Olejarczyk, Franca Tecchio, and Vittorio Pizzella

2015 "Age-related changes in electroencephalographic signal complexity", *PLoS ONE*, 10, 11, pp. 1-13, ISSN: 19326203, DOI: 10.1371/journal.pone.0141995. (Cited on p. 7.)