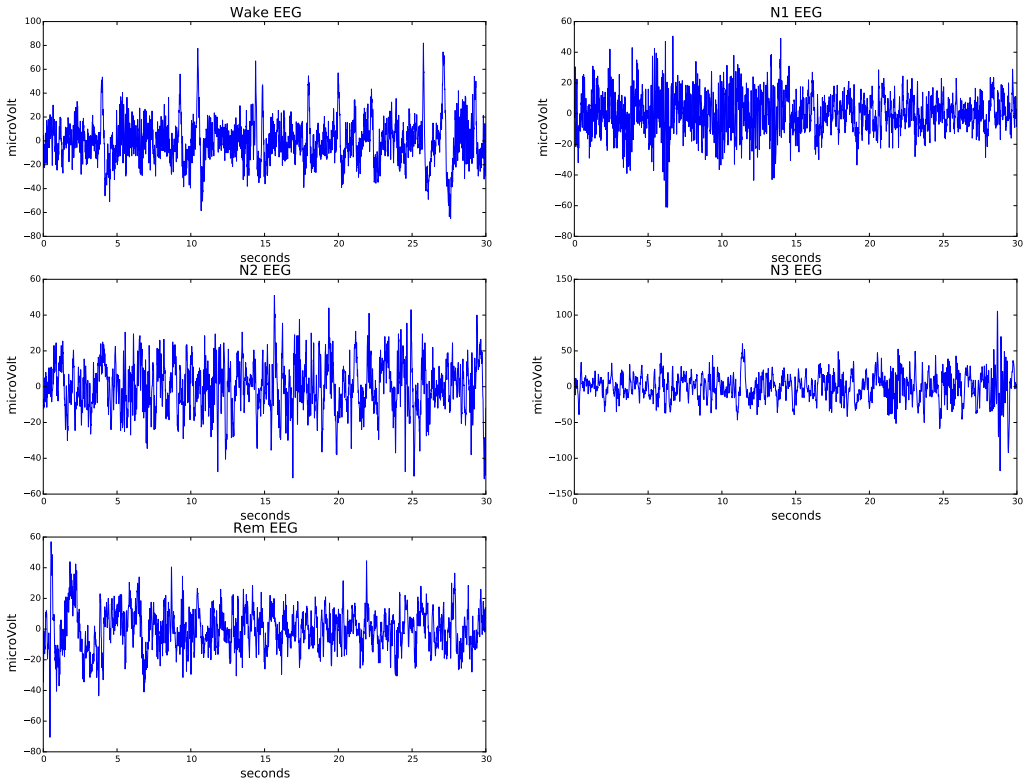
EEG Challenge MDI343

Jonathan Ohayon

23 avril 2015

1 Rappel du Projet

L'objectif est de prédire l'état de sommeil d'un individu en fonction des ondes électriques générées par son cerveau. Plus précisément il s'agit de labelliser automatiquement des échantillons de 30s de données d'électroencéphalographie (EEG) en fonction du stade de sommeil durant lequel elles ont été enregistrées. L'envoi contient 2 versions, celle qui a obtenue le meilleur score (avec une légère difference due a un écrasement des configurations -0.2%) qui est la version avec le script pour le calcul et l'execution. La deuxième version contient les nouvelles features mais je n'ai pas eu le temps de recherche une meilleur configuration pour battre le score précèdent.



2 Création des features

L'extraction des features se base sur plusieurs articles.

2.1 Transformée de Fourier

La transformée de Fourier va nous permettre d'extraire une information sur l'amplitude des fréquences présentes dans le signal. Pour cela nous utiliserons la fonction FFT de numpy. Nous définirons des bandes de fréquences significatives dans l'étude des EEG. Le but est de trouver les ratios de puissance ainsi que l'amplitude max ainsi que la variance dans les bandes de fréquence $[1, 4]$ delta, $[4, 8]$ theta, $[8, 12]$ alpha, $[12, 16]$ sigma, $[16, 32]$ beta, $[32, 45]$ beta2. Il est aussi possible de calculer à ce niveau d'autre features, notamment la fréquence moyenne, la fréquence médiane, la fréquence pic ... [PNPL12] Dans la version "FinalExtracted", la transformée de Fourier a été remplacer par sa version short calculée à l'aide de MNE.

2.2 Wavelet

Les Wavelets sont utilisées pour décomposer le signal, soit en utilisant la version continue CWT avec la famille d'onde de Morlet, soit avec la version discrete DWT. La DWT permet de séparer les différentes bandes de fréquences des EEG comme la transformée de Fourier mais permet d'avoir une evolution des fréquences du signal en fonction du temps. J'ai aussi calculé quelques features provenant des EMG, notamment le zero-crossing, ainsi que la Willison amplitude WAMP, la Myopulse percentage rate (MYOP), Maximum Fractal Length (MFL), Waveform length (WL). [KSO+11], [BP13]

2.3 Transformé de Hilbert (Harmonique)

La transformée de Hilbert codée dans scipy permet d'accéder a un set de features Harmonique, notamment amplitude instantanée, la phase instantanée et la fréquence instantanée. En me basant sur l'article : http ://www.hindawi.com/journals/tswj/2014/420561/

2.4 Features sur le signal brut et filtré

J'ai calculé quelques features sur le signal brut, notamment la kurtosis, la skewness ainsi que la variation d'amplitude moyenne. Ainsi qu'un petit calcul de compaction du signal avec l'algorithme de kolmogoroff qui permet de distinguer si un signal est répétitif ou pas. [PNPL12]

2.5 Features provenant de Pyrem

Le calcul de ces features provient de pyrem, un projet en ligne sur Github qui reprend des fonctions calculée dans Pyeeg mais en améliorant certain des calculs. Nous avons retenu notamment l'entropie du signal, hurst, fisher_info, svd_entropy, spectral_entropy, Higuchi fractal dimension, Petrosian fractal dimension,

2.6 Creation de fenêtre de 15 secondes

Finalement pour augmenter le nombre de données, j'ai aussi essayé de partitionner mes données en fenêtres temporels de 15 secondes et de recalculer l'ensemble de mes features dessus.

3 Selection de features

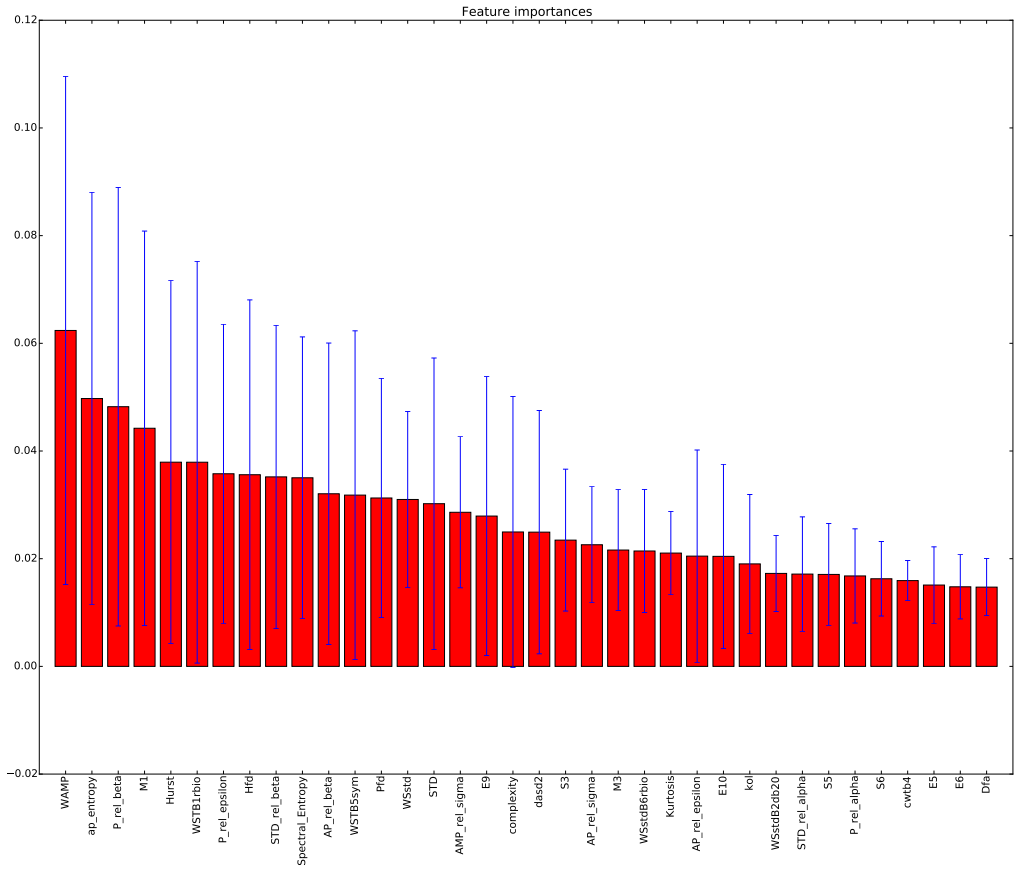
J'ai principalement essayé trois types de selection de features.

3.1 Greedy

Il s'agit ici de rajouter les features une a une en vérifiant d'augmenter le score au fur et a mesure en utilisant une cross-validation. Cette algorithme est assez lent quand le nombre de features augmentent et depend aussi de la vitesse d'execution du classifieur choisi.

3.2 Features_importances

le but est ici de discriminer les features en utilisant la fonction features_importances des arbres, soit en utilisant le classifieur RandomForest ou ExtraTrees. Le code est une reprise d'un exemple de sklearn.



3.3 PCA

Finalement, j'ai aussi essaye de faire une pca sur les features obtenues pour essayer de diminuer la dimension pour l'apprentissage.

4 Choix de Modele

J'ai essayé de multiple modèle au cours du challenge. j'ai commencé avec une foret aléatoire, puis avec un vote entre plusieurs classifieurs notamment un svn, une foret aléatoire et un knn. J'ai aussi tenté d'intégrer un MLP mais les calculs étaient long et ne me donnait pas de résultats satisfaisant.

J'ai aussi essayer d'implementer trois kerne pour mon SVM, un premier qui est une combinaison d'un kernel polynomial et gaussian, un deuxième qui est un version pondérée du kernel gaussian. Malheureusement je n'ai pas réussi à trouver comment bien calculer les meta-parametres de ces deux kernels. Et de plus les certains calculs matriciels ne passaient pas sur mon ordinateur.

Une derniere tentative a aussi été effectuée avec un arbre de SVM, en essayant de classifier dans un premier temps le réveil contre le reste, puis le N1 et Rem contre N2 et N3. puis N1 contre Rem et N2 contre N3 mais cela ne m'a pas donné de résultat satisfaisant. [LCR+15]

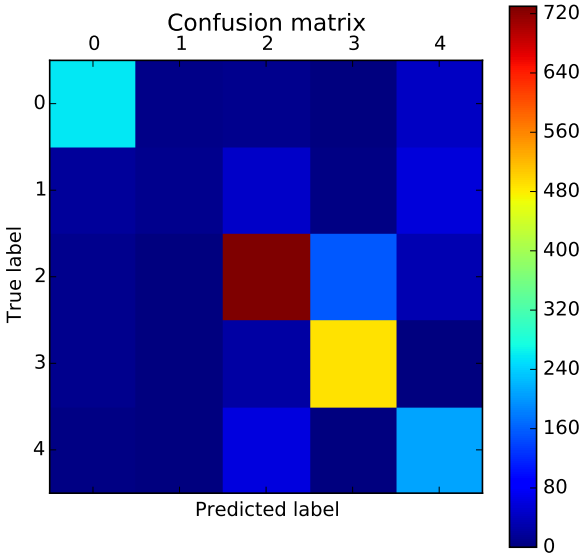
Finalement le modèle que j'ai selectionmé est le suivant :

- Un premier SVM sur les features selectionnées par un algorithme Greedy sur le signal entier.
- Un deuxième SVM sur les eatures selectionnées par un algorithme Greedy sur le signal decoupé en portion de 15s.

Le vote est alors fait en prenant le vote du premier SVM sur le signal entier et les deux votes des découpes sur le deuxième SVM en les pondérant de moitié. J'ai déterminé le gamma et le C des SVM à l'aide d'un Gridsearch.

5 Conclusion

J'ai au final un algorithme qui obtient 79.25% (j'ai malheureusement ecrasé les configuration du modèle qui obtenait 79.5%). J'aurais bien aimé essayer d'intégrer des traitements en amont sur le signal pour essayer d'enlever du bruit. Les méthodes que j'ai essayée n'ont jamais réussi a faire augmenter le score. J'avais aussi commencé à essayer de detecter les outliers notamment les signaux comportant du bruit ou ayant une forme bizarre (il y a certain signaux qui oscille en 0 et 1 qui ne sont pas des EEG normaux). Il y a aussi une grosse difficulté pour determiner la classe N1 comme nous pouvons le voir sur la matrice de confusion.



Références

[BP13] Varun Bajaj and Ram Bilas Pachori. Automatic classification of sleep stages based on the time-frequency image of eeg signals. *Computer methods and programs in biomedicine*, 112(3) :320–328, 2013.

[KSO+11] Sirvan Khalighi, Teresa Sousa, Dulce Oliveira, Gabriel Pires, and Urbano Nunes. Efficient feature selection for sleep staging based on maximal overlap discrete wavelet transform and svm. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 3306–3309. IEEE, 2011.

[LCR+15] Tarek Lajnef, Sahbi Chaibi, Perrine Ruby, Pierre-Emmanuel Aguera, Jean-Baptiste Eichenlaub, Mounir Samet, Abdennaceur Kachouri, and Karim Jerbi. Learning machines and sleeping brains : Automatic sleep stage classification using decision-tree multi-class support vector machines. *Journal of neuroscience methods*, 2015.

[PNPL12] A Phinyomark, A Nuidod, P Phukpattaranont, and C Limsakul. Feature extraction and reduction of wavelet transform coefficients for emg pattern classification. *Elektronika ir Elektrotechnika*, 122(6) :27–32, 2012.