MVA: fMRI and Brain computer interfaces (2020/2021)

BCI challenge: Detecting error-related potentials

Autor: *Charbel-Raphaël Segerie* *(April 25, 2021)*

Abstract In this document, I present my solution to the Kaggle challenge: "BCI Challenge @ NER 2015". The goal of the challenge is to train a statistical model to spot erroneous BCI predictions, by looking at brain signals right after the prediction. Using EEG data, the BCI attempts to determine which letter the user is looking at and presents that letter to the user. If the letter coincides with the same letter the user was thinking about, the feedback is considered positive, otherwise the feedback is negative. The challenge is to predict, based on the user's response to the feedback, whether the feedback is positive or negative.
By extracting some classical features, I built a model able to reach an AUC of 0.6841. Combining the prediction of this model with the prediction of a model based on Riemannian geometry, we go up to an AUC of 0.77, which corresponds to be in the top 4 of the competitors in the original competition.[1]

0.1 Description of data

0.2 Small literature review of FRN

Feedback-related negativity (FRN) is a brain wave that is triggered when the individual observes an event that is opposite to the expected event. The paper Huang and Yu [2014] is a good introduction to FRN:

Positive and negative FRN During the process of learning, individuals need to quickly evaluate feedback to determine whether the action/behavior was appropriate (positive feedback) or inappropriate (negative feedback).[2]

Continuum or discretization of FRNs? We can then make an analogy with the concept of *reward* in reinforcement learning: The reward would be here the dopamine with a phasic increase or decrease of the *basal ganglia*. Research is underway to determine if the amplitude of the dopaminergic response is proportional to the error, which could translate for our kaggle challenge by FRNs more easily detectable when the machine chooses very different/very distant letters on the screen, and therefore possibly by the need to classify FRNs according to their amplitudes (for example differentiate Small FRN from Big FRN ?)

Scarcity Hypothesis: Rare negative feedbacks induce more salient FRN? Another interesting fact about FRNs is that they are more easily detected for rare events (Holroyd Nieuwenhuis et al. in a study focused on gambling). This is rather good news because it indicates that the rare negative feedbacks from individuals such as individual 6 2 (individual 6 is understood 9 times out of 10 by the BCI, but which in return provides unbalanced training data since they then provide 9 times more data with positive feedbacks than negative ones), will be more easily detected by our system despite the unbalanced data. We have done this study in the section 4.5.

[1] The Google Colab on which I worked is available here.

[2] Technically, we are only trying to detect negative feedback in this Kaggle, but in order to detect all error related potentials, including positive ones, we would have had to ask the participant each time if they thought they were "focused enough" during the task for the BCI to do its job, and record events where the individual felt their attention waning, but where the BCI was still able to correctly determine the letter.

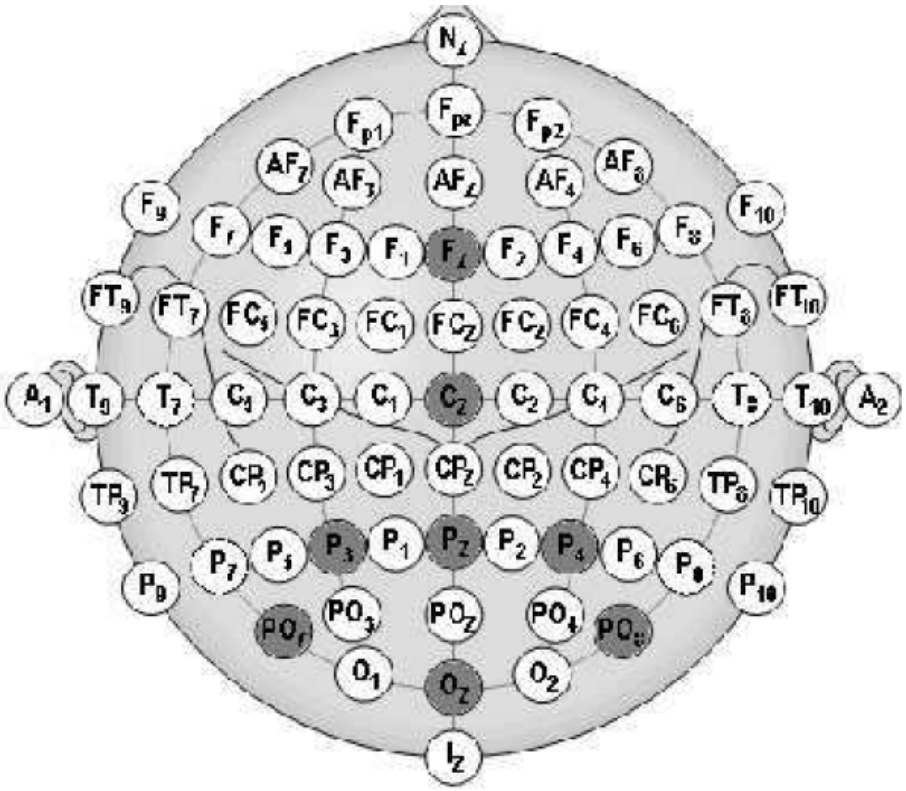

Figure 1: Recommended electrodes for P300-based BCI design, according to (Krusienski et al, 2006). Figure taken from Lotte [2014].

Data Engineering The information of interest to the Machine Learning practitioner is contained in the following paragraph: "The FRN which peaks at around 250 ms after feedback onset, is maximal at frontal-central scalp electrode sites and is most likely generated in the anterior cingulate cortex (ACC)" Huang and Yu [2014]: this paragraph indicates that merely analyzing 1.3 seconds after feedback should be sufficient to observe the FRN.
The information on the feature selection of the channels seemed to me sometimes contradictory for example between the figure 1, which advises to use the electrodes at the back of the head and the fact that the FRN is maximal on the frontal-central scalp region, thus rather on the frontal part of the head. Therefore, I don't think that going further in the neurological or psychological expertise is useful in the competition. The winning team's solution is not at all based on a literature review of the FRN and just does feature engineering based on classical data science methods.

0.3 Description of the participants

Data from sixteen subjects are provided as a training set. Ten other subjects constitute the validation set. Two subjects constitute the score used for the public leaderboard and eight for the private leaderboard. The latter is used for the final ranking. Most of the data for this competition come from Margaux et al. [2012].
The different individuals in the study are more or less well understood by the BCI: for some, the BCI makes a near perfect score, but for others the BCI is wrong one time out of 2 (see figure 2). This heterogeneity between the different participants makes the study very difficult.
It is noticeable that the participants are all quite different from each other as in figure 3. It may be possible to create clusters of subjects in a self-supervised manner, and then create a model for each type of subject, but we have not explored this approach.

subject	
13	0.523529
26	0.532353
12	0.555882
16	0.620588
14	0.632353
2	0.647059
23	0.650000
11	0.661765
17	0.664706
20	0.694118
24	0.702941
18	0.767647
7	0.902941
21	0.917647
22	0.920588
6	0.929412

Figure 2: BCI success rate. For individual 6, the BCI manages to predict the right letter 9 times out of 10. For individual 13 only one time out of 2. But even a 52% success rate is still much better than guessing at random ($52\% \gg 1/36$ letters and numbers), so the system used by the BCI is already working admirably well.

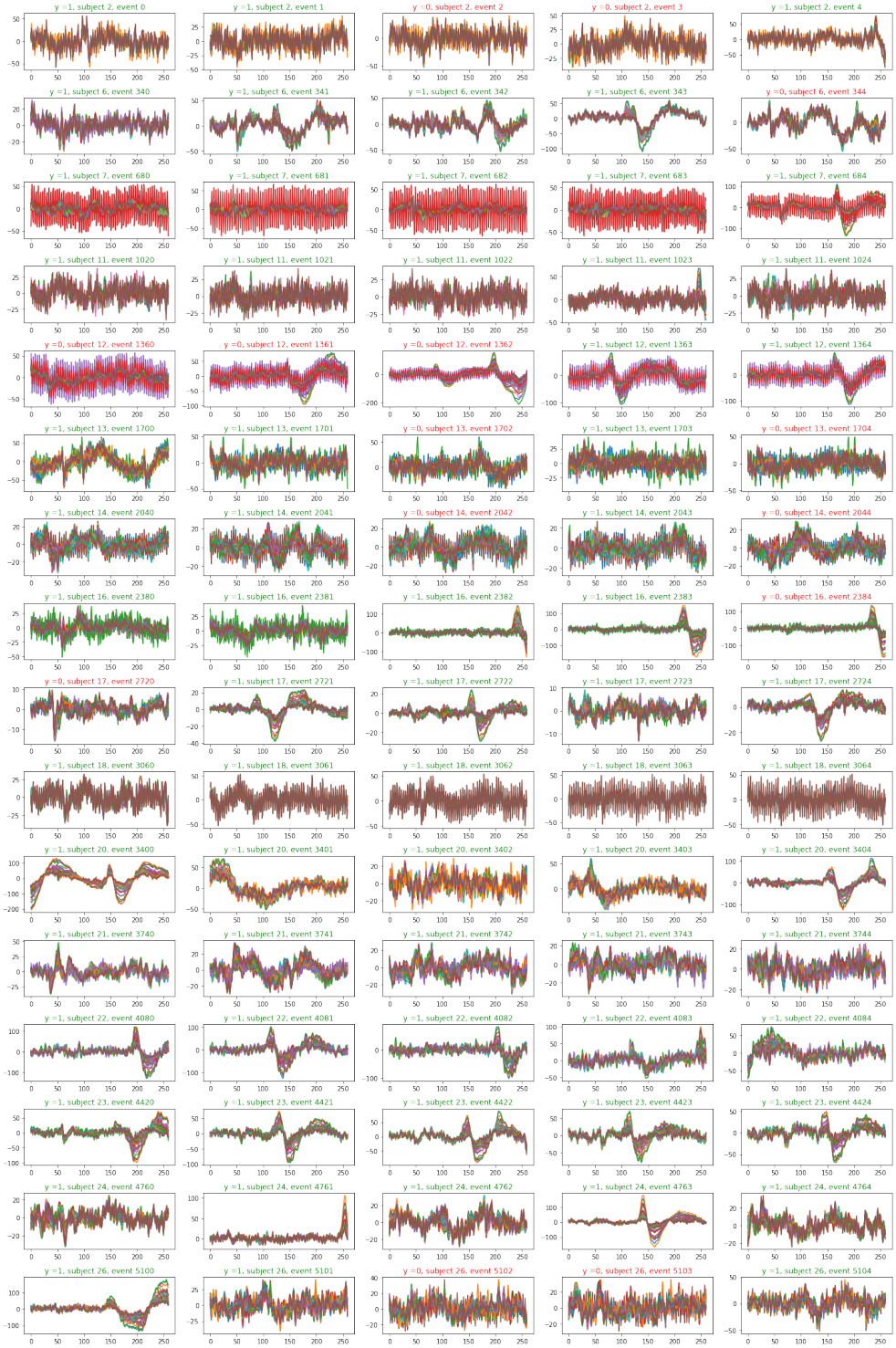

Figure 3: You can see on this image 5 epochs per subject. The green and red titles indicate respectively positive and negative feedback. We can see that individuals often have very different characteristics. On the other hand we can see that some channels for some users are very "noisy" such as the channel in red for individual 7. But this kind of artifact disappears when using the low pass filter (12 Hz) and we can see that the response of the channels on the following figure is very clean, and remains coherent in an inter-channel way.

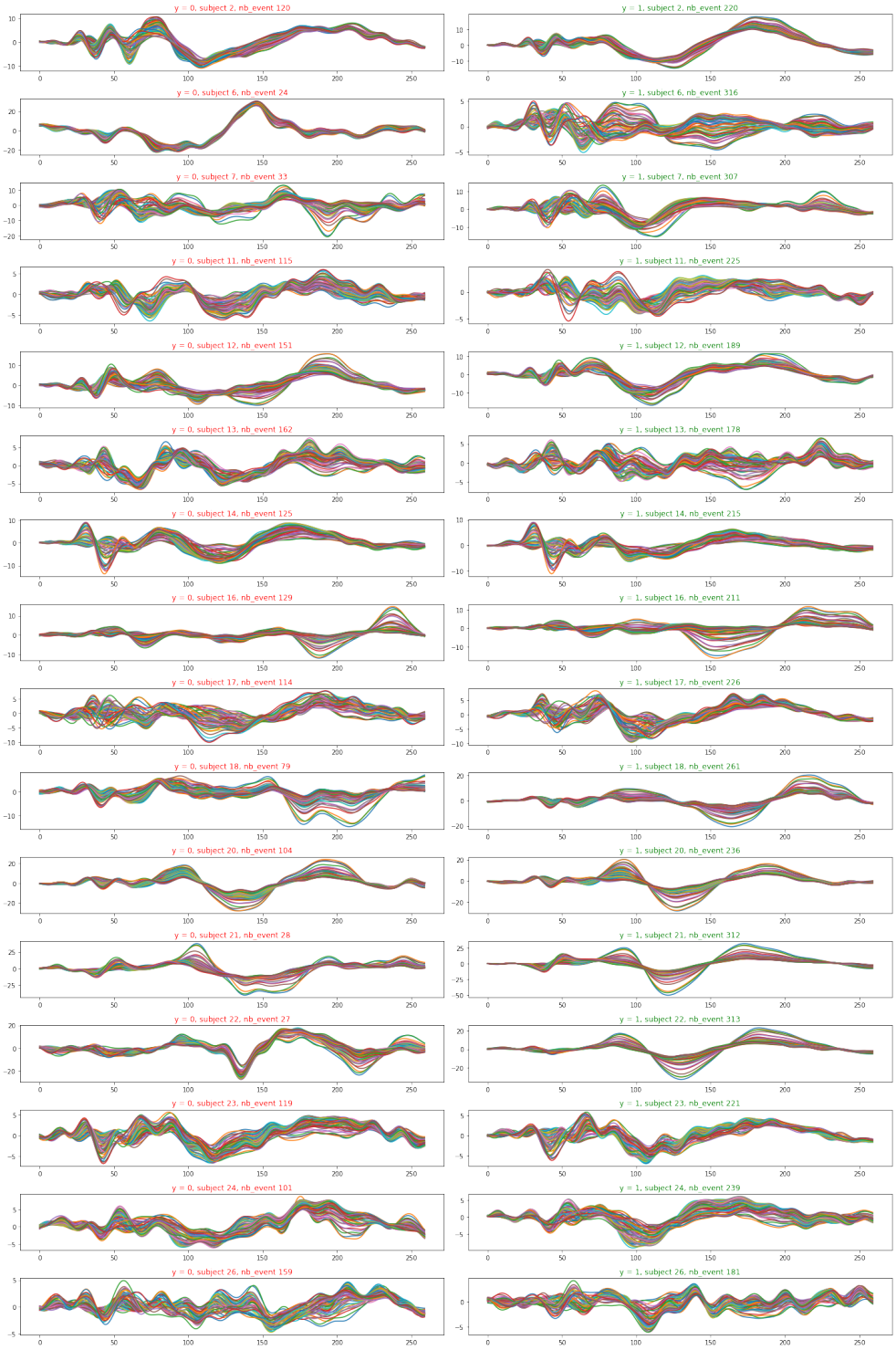

Figure 4: We can see on this image the evoked response for each subject and for the positive and negative feedback. This kind of image is very nice, but if we consider the impressive scores of the Riemannian geometry model, the real information is not really in the shape of these signals, but in the relationship encoded by the covariance matrix between the different channels. Moreover, we can see in this picture that the differences between the subjects are much more marked than the differences between the positivity/negativity of the feedback.

1 Description of my solution

1.1 Difficulties

The creation of a generic model for classification of EEG signals is very complicated. Indeed, the signals are very specific of the participants as we can see in the following section. The signal-to-noise ratio is very small, we record all the brain activity while only a small part of the brain activity is relevant for the considered task.

1.2 Strategy for resolution

The Kaggle took place in 2015. It is now 2021. So it's now much easier than it used to be to get good results for many reasons:

- The winners have published their method.
- Numerous python packages have been published that didn't exist at the time or that were only in their infancy (MNE, PyTs, Pyriemann, Auto-Sklearn, Pandas...) and the interface of libraries such as Sklearn has never been so clean.

So I started by getting familiar with Riemannian geometry, which is the main tool used by Alexandre Barachant who was the winner of the competition. But Riemannian geometry only explores the spatial structure of the data using the covariance matrices of the different channels. Honestly, getting such results simply by using spatial information is impressive, both in terms of engineering and the beauty of mathematics as well as the powerful computational resources allocated to the task. But the other top-participants in the competition often exploited a whole range of features far from those used in Riemannian geometry. I used Duncan Barrack's solution [Dun 2015] and the manual Lotte [2014] to build a model based on complementary information to those used by Alexandre Barachant.
If I summarize my strategy in one sentence, I would say that there is no need to be original, and that one should not hesitate to take advantage of the countless resources contained in the Python ecosystem. So I was constantly gathering resources from left and right to build my colab.

1.3 Combining Two Models to Win

In accordance with my strategy, I constructed two sets of features: one globally orthogonal to the spatial information used in Riemannian geometry and the other using the covariance matrices between the different channels.
In both models, I focused on the time windows of 1.3s after the feedback, which contain the *Event related potentials* associated with the user's reaction to the feedback, which according to the paper Huang and Yu [2014] should be more than enough since the FRN is maximal 250 ms after the event.
For our model inspired by the 4th of the competition we extracted:

- Temporal characteristics
- Frequency characteristics
- Metadata
- Template matching
- Spatial characteristics

The details of the two models are given in the sections 2 and 3 respectively.

2 First Model: Classic Machine Learning

2.1 Feature engineering

Our feature engineering is based on the advices given in the article Lotte [2014]. Template matching is an idea I had spontaneously while doing a lot of visualization. I realized later that some top competitors of the Kaggle Challenge had also done the same.
Another point, which seems essential in my learning process, I absolutely wanted to avoid importing PyTorch during this competition. I already did PyTorch extensively during this year, especially during the sleep apnea challenge Dre, which consisted in segmenting the sleep phases, I could very well have reused a convolution model. For the sake of learning new things, I didn't use PyTorch, but it's worth noting that adding a CNN in the final bagging is certainly a good idea.

2.1.1 temporal features

The important thing is to reduce the dimension. We have a dataset with about 5000 labels. So we need at most 500 features unless we use techniques such as SVD which allow a number of features equal to the number of samples, but as a first approach we have to reduce the dimension considerably.

- We applied a first bandpass filter between 1-12 Hz which is a value advised by Lotte [2014]. To be more exact, we use a 5th order Butterworth bandpass filter.
- We extracted epochs of 1.3 seconds with the help of the MNE library Gramfort et al. [2014] (it is better to apply the low-pass filter before epoching otherwise one risks to make appear artifacts).
- We visualize the epochs in order to check if some information is redundant.
- With the help of the visualizations, we notice that the different channels are often correlated. We therefore choose to make a spatial subsampling by selecting only one channel out of 10. Indeed, one could criticize the choice of one channel out of 10 which may not seem very optimized but after visualization, but it seems to me that the essential information has not been lost by this procedure.
- After having done spatial subsampling, we can do temporal subsampling and keep only one point out of three, which is approximately a sampling of 200/3 Hz. We can see the result of DownSampling on the figure 5.

2.1.2 frequency features

Before filtering with the 12 Hz low pass filter we extract the frequency characteristics using the scipy function *welch* through the DEM interface. We compute the power spectral density (PSD) using Welch's method for each Hz between 0 and 50 HZ. Then we compute the average over all channels.

2.1.3 Template matching

After calculating an average template (evoked response) for positive feedbacks as well as negative feedbacks represented in figure 6, we extracted for all epochs:

- The Euclidean distance with the two templates as well as the ratio of the two distances.
- The correlation with the two templates for the average of the different channels.
- The difference between the correlation with the positive and negative template as well as the binarization of this difference (boolDiff).
- The standard deviation between the different channels.

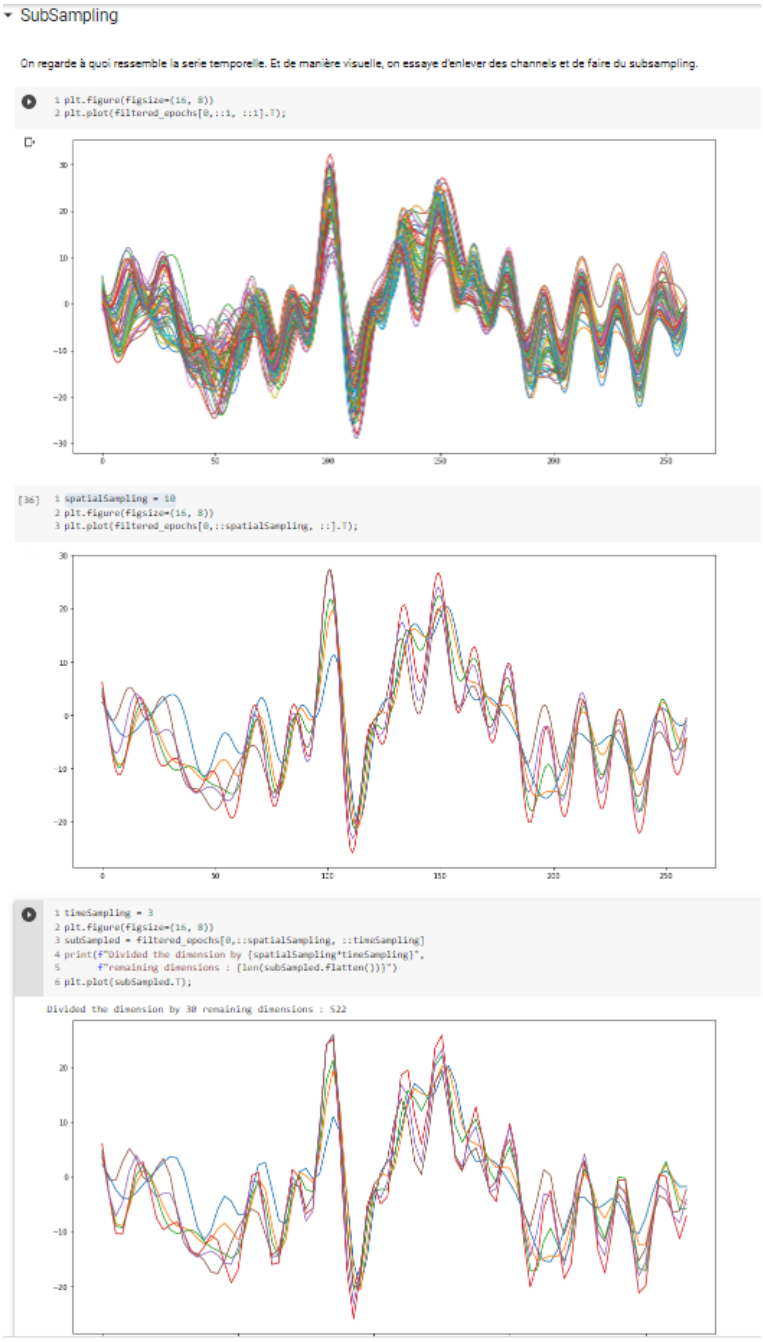

Figure 5: Visualization of the spatial subsampling followed by the temporal subsampling. This last image contains 10 times fewer channels and 3 times fewer points than the first image. But we can see that it contains in essence the information of the first image.

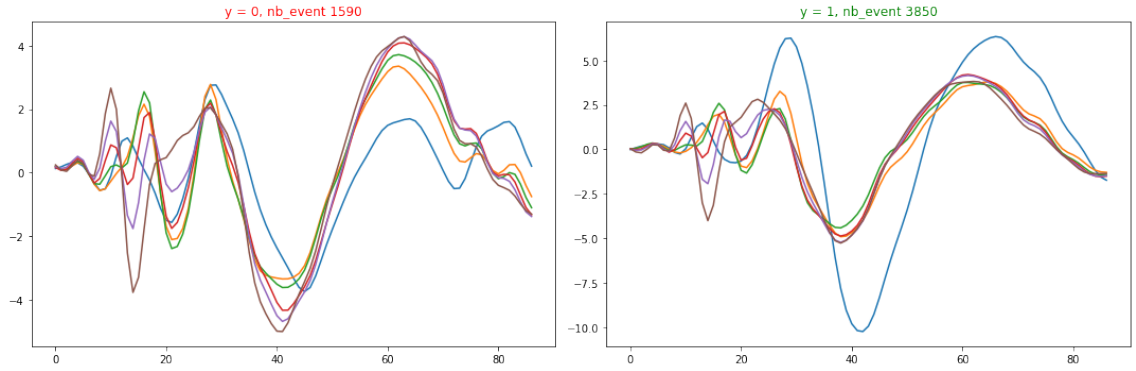

Figure 6: After calculating the low-pass filter (¡12 Hz), we calculate the average template (evoked response) for the positive and negative feedback.

2.1.4 Meta features

We converted the winners code from python 2 to python 3. And we did the preprocessing with their code that create the epochs as well as create a metadata file containing:

- Session id : the session number of the current epoch.
- FeedBack : the Feedback count since the beginning of the session.
- Letter : the Letter position in the current word.
- Word : the word count since the beginning of the session.
- FeedBackTot : the feedback count since the first session.
- WordTot : the word count since the first session.
- isLong : was the current word flashed 8 times, i.e. a long sequence of flashes.

Of course, we did not use the leak data.

2.2 Classifiers used

After building the temporal frequency, template matching and meta features, we used a Random Forest Classifier. We tested numerous meta parameters. With or without features selection. It turns out that the best model is a Random Forest using 1000 trees. We could also be influenced by Dunkan, and pick one type of model per type of features and then combine the different models.

3 Second Model: Riemannian Geometry

We became familiar with the geometry framework. It is an interesting approach that allowed the team that came first in the competition to distinguish itself Barachant et al. [2013].

3.1 Essence of Riemannian geometry

In this paragraph we quickly summarize the paper Barachant et al. [2013]:
The idea is the following: One cannot just sum coefficient by coefficient two covariance matrices since the sum of two SPD matrices is not necessarily an SPD matrix (Think of identity matrices and minus Identity matrices). So in order to create a meaningful average of a set of covariance matrices, we have to project all the covariance matrices in a linear space (take the LOG of all the covariance matrices),

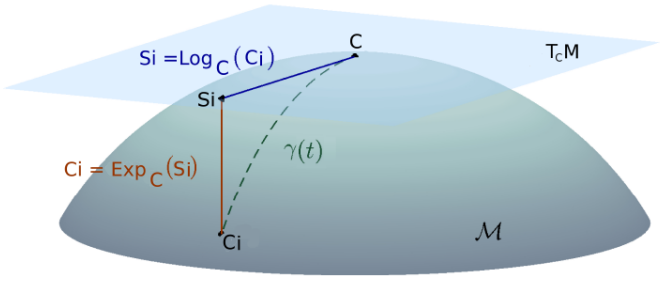

Figure 1: Manifold $\mathcal{M}$ and the corresponding local tangent space $\mathcal{T}_{\mathbf{C}}\mathcal{M}$ at $\mathbf{C}$. The Logarithmic map $\text{Log}_{\mathbf{C}}(.)$ projects the matrix $\mathbf{C}_i \in \mathcal{M}$ into the tangent space. The Exponential map $\text{Exp}_{\mathbf{C}}(.)$ projects the element of the tangent space $\mathbf{S}_i$ back to the manifold.

Figure 7: Figure from Barachant et al. [2013].

then average the LOG of the covariance matrices. We then obtain the final average matrix by taking the EXP of the average of the LOGs as shown in the figure 7.
We can then *half-vectorize* the matrices by transforming them into a vector keeping only the upper triangular part of the matrices and then use this vector in a final classifier like an SVD.

3.2 Implementation

We used the *pyriemann* library in order to compute the average covariance matrix of the positive feedbacks as well as the one of the negative feedbacks using XdawnCovariances estimators and the ElectrodeSelection estimator. We then used a tangentSpace classifier, which project data in the tangent space and apply a classifier (LogisticRegression) on the projected data.
We used the recommended meta parameters after checking them by GroupKFold cross validation. But we did not do any bagging which is very expensive computationally. Thus, the final model used is trained in just 30 seconds.

4 Results

4.1 Contribution of each model

- The model using pyriemann has an AUC of 0.75.
- The model using feature engineering achieves an AUC of 0.69.

4.2 Bagging

The average of the two predictions gives an AUC of 0.78, which honestly is a bit disappointing considering the work done in the construction of the different features, and the fact that the two models are built on ideas carefully chosen as different... So it seems to us that the main information is to be found in the spatial configuration of the signal.

4.3 Univariate Analysis

We can study the influence of the main features on the predictions of the model with the SHAP values as in figure 8.

4.4 Feature importance

It is obvious that most of the information is in the geometry of the signal. However, it is still interesting to make an analysis of the main contributions of the interpretable features done in figure 9.

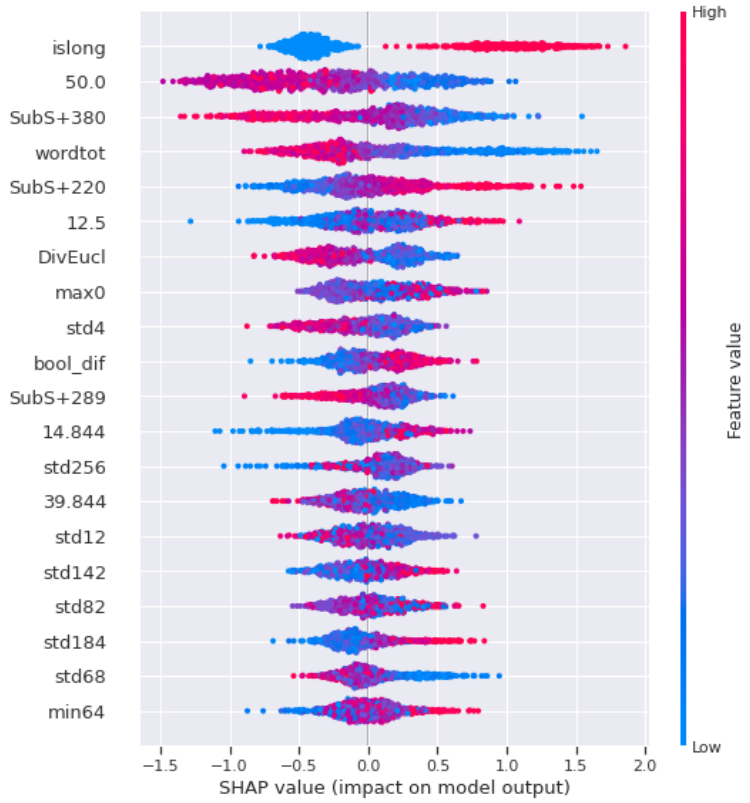

Figure 8: SHAP value: average impact on the model prediction. Interpretation: (1) islong is 1 when the sequence is long and 0 otherwise. Thus, a long sequence is correlated to a better prediction because the feedback is more often 1 (the feedback is more often positive). (2) Interestingly, the higher the wordtot is, the more the subject loses focus and the quality of the feedback decreases, which means that it is better to prefer short sessions. (3) DivEucl is the division of the Euclidean distance of the filtered signal with the positive template divided by the distance to the negative template, so DivEucl is naturally also anti-correlated to the positive feedback. (4) The other features are not interpretable as is (Numbers such as 50.0, or 12.5 represent the value taken by the PSD taken for the 50Hz, 12.5HZ frequency. Subs380 represents the 380th component of the subsampling. Max0 represents the highest value of over all channels taken at time 0).

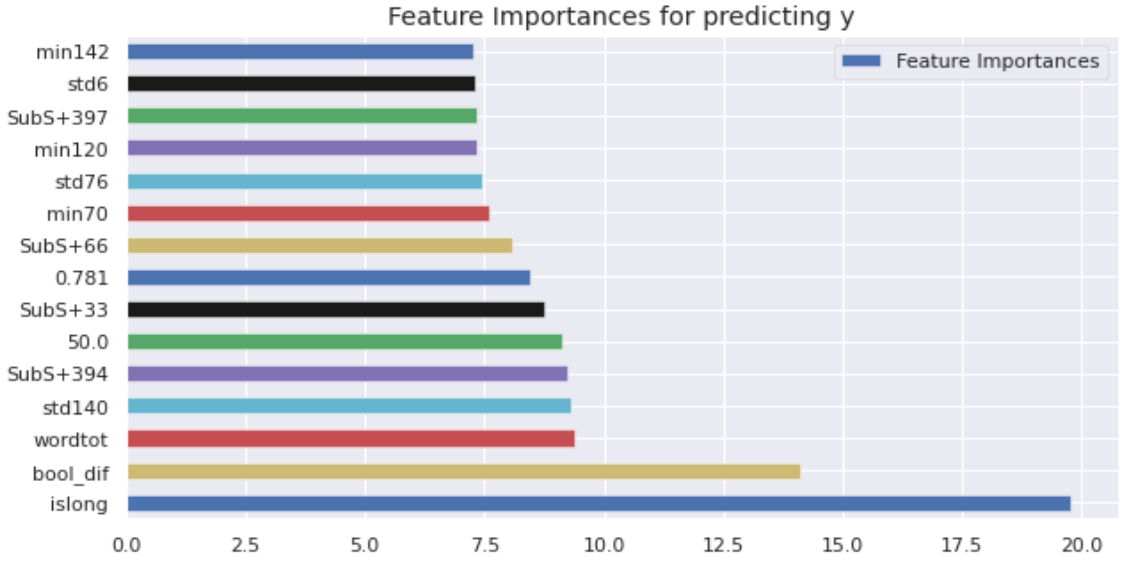

Figure 9: Feature importance computed on our first classical model by RandomForest. We can see that the length of the sequence is the most important feature. booldif is the second most important feature, whose construction is explained in the section 2.1.3. We can see that after these two features, the contribution of the other features is not negligible because it has been empirically shown that not putting a feature selection improves the performances, but these other features are difficult to interpret.

4.5 Analysis of the Scarcity Hypothesis

We formulated a hypothesis in section 0.2 about the link between the rarity of negative feedback and the magnitude of the FRN: theoretically, in the reinforcement learning analogy, the greater the surprise the greater the FRN, and therefore the more detectable. We therefore compared the AUC of different subjects in the validation set as a function of the ratio of positive feedback to the total number of feedbacks for this subject.

Results We can see from the figure 10 that as we increase the ratio of the number of positive to negative feedbacks, the better AUC our predictive model has.

Possible interpretation This means that people already correctly interpreted by the BCI will then also be correctly interpreted by our error correction system, while people misinterpreted by the primary letter recognition system will also be misinterpreted by our secondary error correction system. It is obvious that people who see the BCI being wrong every other time cannot really be surprised by the succession of errors repeated by the machine and get used to the repeated errors of the machine by lassitude, and so the FRN would only appear at the beginning of the experiment for these people.

Critics But it may also be that these people are inherently more difficult to interpret for both the primary or secondary system. The fact that *wordtot* is also anti-correlated with negative feedbacks (fig. 8) is a sign that shows the influence of either lassitude or deconcentration or failure habit and seems to go in the direction of the scarcity hypothesis, but further experiments with a protocol set before the interpretation of the data are necessary to decide if they are intrinsically more difficult to interpret or if these results derive from the scarcity hypothesis.

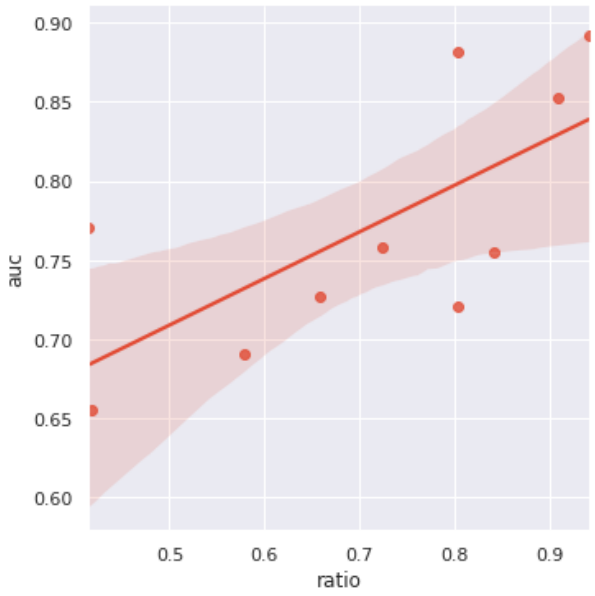

Figure 10: AUC for different individuals on the validation set as a function of the share of positive feedbacks among all feedbacks. The trend is clearly upward: Either the scarcity hypothesis is true or some people are inherently more difficult to interpret for both the primary P300 BCI and our secondary correcting system.

5 Complements : Rejected approaches

5.1 Classifiers

We have methodically explored the PyTimes series library (PyTs), but none of the proposed methods allowed us to obtain good results. (Nearest Neighbors with Time Warping did not work (Indeed, on the 3-second scale, the time warping is negligible), and the other methods (BOSS VS, SAX-VSM) based on dictionary creation are extremely slow during prediction.

5.2 Data Augmentation

We have thought about a smart way to do data augmentation for example with the TSaug library. But doing data augmentation did not work. It was impossible to find semantic invariants of the data. Neither time warping, temporal subsampling, nor Gaussian noise were relevant, and systematically reduced the performance on the validation set.

5.3 Feature selection

The feature selection step avoids model overfitting on the training set, but reduces the performance on the validation set. This suggests that I could have added many more features to my classical model, and it explains why I do not quite reach the performances of [Dun 2015].

5.4 MNE library

Artifacts Initially, my first guess was that using the MNE library would not have improved my performance in practice: I know MNE well. I am currently doing an internship with Alexandre Gramfort and I use MNE on a daily basis. But I don't think that using MNE for anything other than creating epochs is relevant here: none of the winners of the competition seem to have done such processing. When we look at the data, we can see that the ocular artifacts are clearly visible as in figure 11. But cardiac artifacts have been removed by the organizers of the competition. *It seems that the model benefits from*

Figure 11: We can see that ocular artifacts have clearly not been removed in those 50 seconds.

those ocular artifacts: the movement of the subject's eyes is not random and contains information that is interpreted by the model.

Going further? But it turns out that since the Riemannian geometry model performs very well, the information is certainly contained in the geometry of the signal. So I thought that an interesting approach would be to compute the source of the signal in the brain using for example the MNE library and then work directly on the reconstructed data using MNE to create a classifier. But after trying to use MNE, it turns out that the P300 is not in the list of available montages. So I did not insist.

References

Detecting sleep apnea from raw physiological signals by dreem. `https://challengedata.ens.fr/challenges/45`. Accessed: 17-04-2021.

Duncan barrack's solution. bci challenge ner 2015. `https://github.com/duncan-barrack/kaggle_BCI_challenge/blob/master/report.pdf`. Accessed: 17-04-2021.

Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Classification of covariance matrices using a riemannian-based kernel for bci applications. *Neurocomputing*, 112:172–178, 2013.

Alexandre Gramfort, Martin Luessi, Eric Larson, Denis A Engemann, Daniel Strohmeier, Christian Brodbeck, Lauri Parkkonen, and Matti S Hämäläinen. Mne software for processing meg and eeg data. *Neuroimage*, 86:446–460, 2014.

S Holroyd Nieuwenhuis, N Yeung, and JD Cohen. Cb (2003). errors in reward prediction are reflected in the event-related brain potential. cognitive neuroscience and neuropsychology, 14 (18), 4.

Yi Huang and Rongjun Yu. The feedback-related negativity reflects "more or less" prediction error in appetitive and aversive conditions. *Frontiers in neuroscience*, 8:108, 2014.

Fabien Lotte. A tutorial on eeg signal-processing techniques for mental-state recognition in brain–computer interfaces. *Guide to brain-computer music interfacing*, pages 133–161, 2014.

Perrin Margaux, Maby Emmanuel, Daligault Sébastien, Bertrand Olivier, and Mattout Jérémie. Objective and subjective evaluation of online error correction during p300-based spelling. *Advances in Human-Computer Interaction*, 2012, 2012.