# Mini-project 1

Pietro Airaghi, Francesca Luongo, Samuele Mercan

12 November 2018

## Introduction

Event Related Potentials (ERPs) are electrical signals sent by neurons responding to an external stimulus, in our case these stimuli are generated by a cursor moving toward and away from a target. Each movement of the cursor is labelled with '0' if the cursor moved toward the target, and with '1' if it went in an unexpected direction away from the target. The latter would generate an error-related ERPs.

Our dataset was organized in 597 observations each containing a EEG signal recording from sixteen different electrodes at different times, resulting in 2048 features for each event. Each feature thus represents a voltage. This project aimed to train a model, using different techniques, that can separate the two populations of samples containing the cursor movements, the expected ('0') and the unexpected ('1') based on the EEG data.

## Methods

**Part 1: Data exploration, Statistical significance, Feature threshold**

The dataset used to find the best model contains 597 samples each with 2048 features (128 time point for each of the 16 electrodes), also a training label array was provided, which contained the class labels (right cursor movements, around 75% of '0' and 25% of '1').

**Statistical significance:**

Histograms of a single feature were used to differentiate the means of the two classes. As the samples seem to follow a normal distribution, a t-test was performed for features 640 to 780, in order to assess if the difference between the means of the two classes ('0' or '1') in each event is significant. The level of confidence is 5% and the null-hypothesis is that there is no difference between the two means. The MATLAB function *ttest2* was used (command: *[h,p]= ttest2(x,y)*).

**Feature threshold:**

The first approach used to separate the classes was a feature threshold that was set to a certain value of voltage, if one population of samples was under the threshold, it belonged to the expected class ('0'), and if it was above, it belonged to the unexpected class ('1'). Class histograms can help to place the threshold optimally by indicating the mean values belonging to each class, so a mean value of the voltage that the two classes reach for each feature.

To find the threshold that produced minimum class error for this feature, a minimal value of voltage was set (the minimum of standard deviation of the set of samples for this feature) and iteratively 0.001 Volt were added to find the new threshold, until reaching the maximum value of voltage (maximum standard deviation of the set of samples for this feature). For each new threshold, the class error was calculated, and only the smallest error was kept. These steps were performed for features 500 to 1200.

Only class error was used instead of classification accuracy or classification error, because it takes into account that the number of events in each class is not equal by normalizing it, indeed we have 75% of '0' and 25% of '1, whereas classification accuracy or classification error don't. For the class error, both the 1/2 coefficient were kept because precision in classification is not more important for one class than the other (unlike for a HIV test for example). When the class error was over 0.5, we decided to swap the classes in the formula to have a more accurate representation of the classification method. The feature 1093 was chosen to plot the variation of class error and classification error because it had the smallest class error of the range of features.

This first attempt of a threshold, a straight line, is not accurate enough to separate the two classes, the error is too big, localized mostly around 0.5 like a random classifier, so another technique is needed.

**Part 2: LDA and QDA classifiers**

Linear discriminant classifier (LDA) and quadratic discriminant classifier (QDA) are two classifications methods. The first (LDA) can be used when the two classes have the same covariance, it separates the dataset with a linear hyperplane. The latter, the quadratic one, separates the two classes with a quadratic hyperplane. The LDA and QDA classifiers are chosen to separate the data to attribute the features to one class or the other depending on where they are situated with respect to the hyperplane, so it is possible to consider more dimensions, which translate to more features, at the time.

The classifier needs to be trained on the dataset, the function used to train the classifier on the provided dataset is *fitdiscr(features, labels, 'DiscrimType', type, 'Prior', prior)*, giving as output the trained classifier. The discriminant classifiers can be *linear, diaglinear, diagquadratic or quadratic,* the wanted type should be specified when using this function. The prior probability was specified to be uniform, empirical, or defined from the populations frequencies. Uniform prior probability is the same for each class (1/K, K being the number of class), empirical represents the relative frequencies of the class in the dataset and finally the frequencies in the population, which are in this case 30% of '0' and 70% of '1'. The defined prior probability will be used since it is more representative of the population. The MATLAB method *predict(features, labels)* was used to returns the predicted labels of the subset features of the dataset.

In this part, the training was done for all the discriminant classifier on 90% of one tenth of the dataset (randomly chosen features) and tested on the remaining 10%. The different classifiers were then compared with their respective class error.

**Training and testing error:**

The data were split in two subsets of equal size, the aim here was to see if the classifiers trained on the first subset, the training set, can be used on the test set, which is the unseen data and give a satisfactory prediction. The training and test error were then compared for each classifier. The inconvenient with this method is that the train set is too small, and this can lead to high variability.

**Cross-Validation for performance estimation:**

A k-fold cross-validation was performed, to increase the number of samples for the training and improve its accuracy. The *cvpartition* function was used to randomly separate the dataset but considering their labels, so spreading them equally in every subset. A standard seed (45) was used in the whole project for reproducibility. If a k-fold partition is chosen, there will be k-1 subsets for the training of the classifier and one to test. After that, the class error is calculated. This method is then used iteratively and at every step the test set is switched with a subset previously used for training. The cross-validation error is the mean of all the class errors calculated for each iteration.

For each discriminant classifier parameter (*linear, diaglinear, diagquadratic* or *quadratic*) a cross-validation was performed in order to optimize this hyperparameter. However, the feature size doesn't allow to run a *quadratic* training since the number of parameters to estimate was too high compared to those of samples. For this reason, a smaller subset of features (1/50 of the original dataset) was taken to estimate the performance of this classifier type, but when the other classifiers were trained with this smaller number of features, the results were excluding possible relevant features, so the results were not promising.

The 10-fold cross-validation enabled us to choose the optimal classifier. Even if, a *repartition* of the folds can potentially change each time the result because of the random choice of the partitions, this is why a more solid method is needed and thus the nested cross-validation was performed to assess the stability of our chosen method of classification.

## Part 3: Nested Cross-Validation, Hyperparameter Optimization and features selection:

To further optimise the model, a selection of the features, that as classifiers are hyperparameters, was performed with three different techniques: *Fisher's method, PCA* and *Forward Feature Selection.*

**Fisher's method, *rankfeat()*** was used in order to rank the features bringing more information about the classification labels (informs about the "discriminative power" contained in the feature) for each single feature. The function works as a filter and, for each iteration of the cross validation, the new best feature is stocked with the previous one.

**Principal Component Analysis (PCA)** is a filter used to reduce the dimension of our dataset. It will set a basis of linearly independent vectors that can define the whole dataset of features with a linear combination. This is possible because some information brought by some features is redundant or void. MATLAB *pca(x)* function was used to perform the PCA ([*coeff, score, variance] = pca(X)* were score is the matrix of needed features).

**Forward Feature Selection (FFS)** is a wrapper that takes into account how features interact with each other contrary to Fisher's method. This feature reduction method tries to assemble the best features and aim to minimize the test class error. It consider the previously selected features in order to avoid adding highly correlated ones, as could happen with Fisher's *rankfeat()*, so redundant information is eliminated. MATLAB function *sequentialfs()* is used on the whole set of unranked features, it performs an inner loop cross-validation that stops adding features when the criterion is not improving.

A 10-fold cross-validation was performed using *linear, diaglinear, diagquadratic* classifier, in the loop of the cross-validation, the reduction of dimensionality was performed to find the optimal number of features selected in function of the test set smallest class error. The nested cross-validation was used to assess the performance of the selected model.

Nested cross validation is used to estimate the unbiased performance of the chosen model, as well as its stability. For every type of classifier, it consists on k-outer folds (**estimation of the generalization error**) nested with a k-inner folds cross validation (**hyperparameter selection**).

After performing the dimension reduction, we decided to take the fisher method for reduction linked with a 10-fold cross-validation, a box-plot was then generated in order to illustrate the test class error for each classifier type (*linear, diaglinear, diagquadratic*) and the hyperparameter variation during the process. An ANOVA table was used to check if the difference between the mean is significant.

**Part 4: Statistical Significance and Final Model**

The output of a nested cross-validation is an array of *k* errors, one for each fold. The mean test error across the *outer* folds is therefore an estimator of the test error of the model selected after the first round of *inner* folds. If we compute the *standard deviation* on the *k* errors we find the robustness of the model; how much variability this model should have when confronted with the test set after being trained on the whole training set comprised the validation set.
In order to test if the *estimated error* differs from a random distribution we are going to use the function *ttest* on Matlab comparing our test errors distribution with a normal distribution centered at 0.5, that correspond to a random distribution. We test the *null hypothesis* that the means don't have a significative difference and we use a 5% significance level for the *alternative hypothesis.*

Since Student's t-test assumes features with a normal distribution normality is tested with a one-sample Kolmogorov-Smirnov test with the command *kstest,* which returns a test decision for the null hypothesis that the data in vector x comes from a standard normal distribution, against the alternative that it does not come from such a distribution.

## Results

The results described in this section show our learning path during the discovery of the various classification and features selection methods. The final classifier should be the fruit of the analysis of each of these methods.

**Part 1: Data exploration, Statistical significance, Feature threshold**

**Statistical significance:**

The figure 1 shows different p-values for a range going between feature n°640 and n°780, the level of confidence chosen is 5%. There are many features that seems like good discriminant, as they have a really small p-value. Two histograms were also displayed below to show two features and their distribution, for feature n°711 the p-value is smaller than the confidence level, so with the t-test we reject the null hypothesis, so there is a significant difference between the means of the expected and unexpected events, whereas for feature n°729 we can't reject it as the p-value is bigger than 5%.

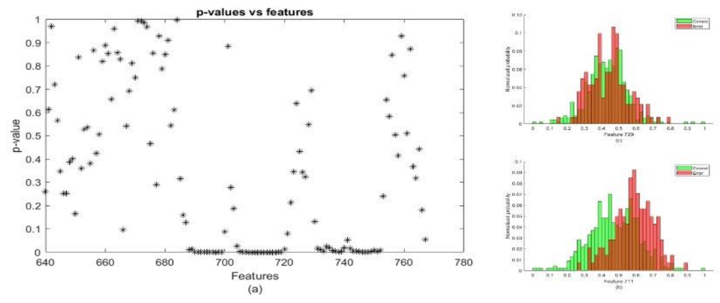


**Figure 1. P-values of the t-test performed to show if the difference between the means of the correct and incorrect classes for each feature (between 640 and 780) is significant and histograms representing two features and their normalized distribution**, in green are the correct cursor movements and in red the erroneous one. (a) plot of the different p-values of the t-test for each feature between 640 and 780, (b) distribution of feature n°711 its p-value is 3.8e-16 (<< α=0.05, Ho is rejected). (c) distribution of feature n°729, its p-value is 0.695 (> α=0.05, Ho is not rejected).

**Feature Threshold:**

Different feature thresholds were set to attribute events to each class and the class error was calculated. The class error is really big for many of the features shown, it is close to 0.5, so the classification is not much relevant for our dataset, it's almost random. Also, the class error is a more effective error assessment than classification error.

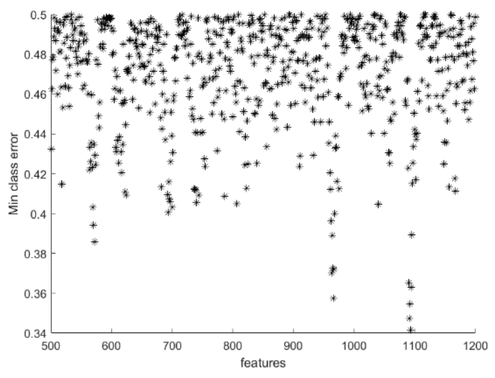


**Figure 2. Smallest Calculated Class Error with a varying threshold.** After setting different thresholds for each feature from n°500 to n°1200, the smallest class error found for the different threshold was plotted. The class errors don't exceed 0.5 because we chose to invert the classes when it became bigger.

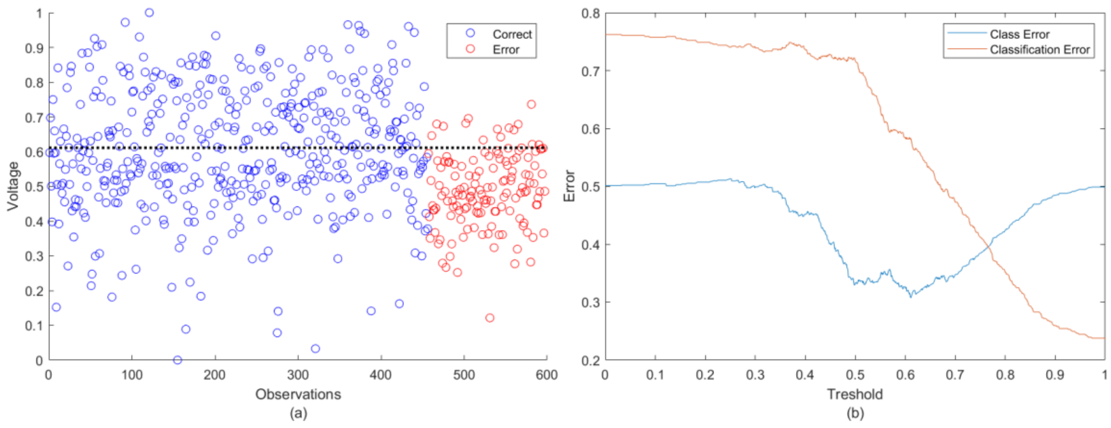


**Figure 3. Threshold for feature 1093. which has the smallest class error and class error and classification error evolution for this feature when the threshold varies.** (a) Representation of feature 1093, in blue are the correct cursor movement and in red the incorrect one. The black dashed line represent the value of the best founded threshold corresponding to the minimal class error (b) Evolution of the class error and classification error when the threshold varies for feature 1093. As the number of observations in the correct and incorrect classes are imbalanced, the minimum class error should be taken into account rather than the classification error, therefore the optimal threshold is 0.611 [V].

## Part 2: LDA and QDA classifiers

Only one tenth of the features were taken into account for this part, leading to a subset of 597 events with around 200 features to separate with the LDA or QDA hyperplanes and, as the dimension of selected features was so restricted, the linear hyperplane training error was always really small, since it is possible to draw a hyperplane when the dimension is too big for the number of events. For example, in a 3D space it is always possible to separate 4 points with a hyperplane. This is the reason behind the choice of reducing the dimensionality of our dataset by removing some random features. Anyway, we still have a huge overfitting for the linear hyperplane and the fact that diagquadratic, which has only a minimal overfit, and seems to perform a bit better diaglinear, suggests us that this hyperplane could be the best one to classify our events.

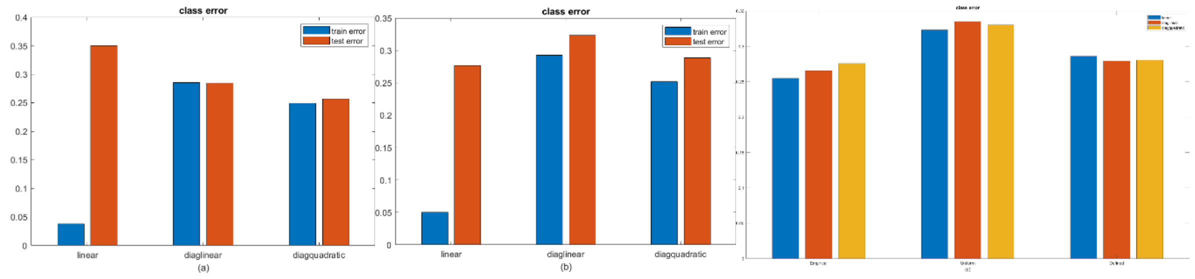
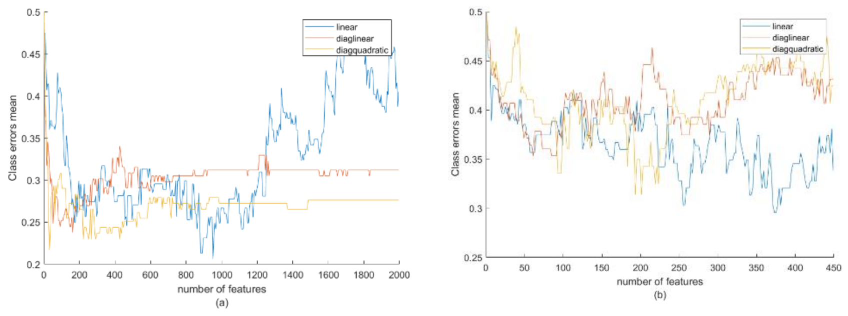


**Figure 4. Comparison of class error of train and test set using the linear, diaglinear, diagquadratic classifiers. (a) Using defined prior probability and without cross-validation. (b) Using defined prior probability and 10-fold cross validation. (c) Comparing the various prior probabilities and with a 10-fold cross validation.** Different discriminant classifiers were used to segregate the features and attribute them to a class, the class error of the test set was then calculated. The classifiers were used specifying the prior probability: empiric, uniform and defined (frequencies in the population, 30% of '0', 70% of '1'). The quadratic classifier was not used because it needed less features as mentioned in the method section. (a) class error of the test set representing 10% of the subset of features (subset is 1/10 of the total features) and the class error of the train set representing 90% of the subset of features, the classifiers trained are linear, diaglinear, diagquadratic, the prior probability is defined by the population frequencies (b) class error calculated for the train and test set after a 10-fold cross-validation, the classifiers trained are linear, diaglinear, diagquadratic, the prior probability is defined by the population frequencies (c) class error calculated on the test set for every classifier and with different prior probabilities, uniform probability gives the worst error whereas the empirical prior probability seems to minimize the class error compared to the defined population frequencies, we think that it is because the empirical prior probability is more representative of the dataset that we have (25% of '1' and 75% of '0') but not representative of the population frequencies.

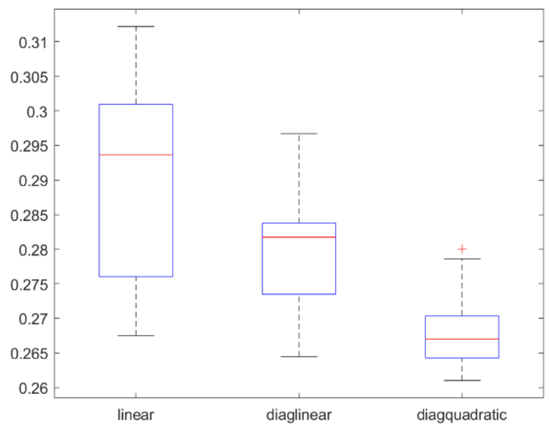
## Part 3 : Nested Cross-Validation with Hyperparameter Optimization:

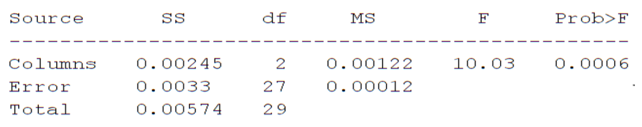


| | Linear | Diaglinear | Diagquadratic |
|---|---|---|---|
| Stop criterion | 0.4564 | 0.448 | 0.4111 |
| Number of selected features | 15 | 7 | 30 |

**Figure 5. 10-fold Cross-validation with linear, diaglinear, diagquadratic classifiers with reduction of dimensionality.** Inside of the 10-fold cross-validation loop, the reduction of features was performed, using different methods (Fisher method, PCA and FFS) (a) Fisher Method using *rankfeat()* MATLAB function, the less correlated features ordered by discriminative power, the minimal class error reached is stabilized around 0.3, and the diagquadratic classifier seems

to perform the best since the error is smaller than with the other classifiers. The optimal number of features in this case would be around 200, since when it increases the error doesn't vary much. (b) evolution of the class error in function of the number of features selected by PCA, the estimated class error for the classifiers doesn't give the same outcome as *rankfeat()*, so this way to use PCA may not be suited for the dataset. (table) class error variation in function of the number of features selected with FFS method and number of features selected at "stop criterion", the class errors are bigger than with fisher method or PCA so we decided not to use this future reduction method.





```
Source        SS        df      MS          F      Prob>F
-------------------------------------------------------
Columns    0.00245     2   0.00122    10.03   0.0006
Error      0.0033     27   0.00012
Total      0.00574    29
```

(a) (b)

**Figure 6. Boxplot representing the mean and variance of the class error after 10-fold cross-validation was performed combined with Fisher method and ANOVA table.** (a) The boxplots illustrate the variation of the class error during a nested cross-validation, the diagquadratic classifier has the most stable error evolution, contrary to the linear and diaglinear one. Therefore diagquadratic classifier was used on the dataset recovered after performing the dimensionality reduction with Fisher method *rankfeat()* (see Fig.5). Quadratic classifier won't be used in our final model, as mentioned in the method section. (b) With the ANOVA test, the difference between the means was assessed to be significant (p-value = 0.0006 << α=0.05, Ho is rejected).

## Part 4 statistical significance and Final Model

The errors values suggest that there is a difference with respect to the 50% random level. Kolmogorov test for normality does not refuse the null hypothesis that the errors are distributed normally.

| | Linear | Diaglinaer | Diagquadratic |
|---|---|---|---|
| Decision | Reject | Reject | Reject |
| p-value | 0.0020 | 0.0041 | 2.0125e-05 |

**Figure 8. t-test to compare if errors mean differs from a random 0.5 distribution. We reject the null-hypothesis that the means are not statistically different with at the 5% significance level.** We can reject the null-hypothesis at a 5% significance level. A good practice would be testing if the error follows a normal distribution before applying a Student's t-test.

## Conclusion

Identifying the correct classification algorithm, in order to correctly assign to which group some specific observations belong, requires an extreme understanding of the available data. The most remarkable peculiarity of the dataset is the fact that the data repartition causes a biased error estimation if a correct prior probability is not considered. Thus, we used a defined prior probability representing the real frequencies of the population rather than the empiric or uniform prior probabilities. Even if empiric gives better results (smallest error) for the used dataset, it is overfitted to the dataset and would not be effective if the data provided are different than our initial repartition of labels, since the proportions are not representative of the population.

Although an implemented classification method based on a single feature threshold can provide a class error comparable to the other methods, an important aspect is to consider the ratio between features and observations. In the case of our dataset, the big number of events and the impossibility to find a single descriptive feature suggest that feature reduction methods should be applied.

The best strategy to train a final classifier is to combine multiple methods learned so far. We suggest to choose a diagquadratic classifier as it was the one with the best trade off between class errors mean and the difference between test error and train error on a 10-fold cross validation computed using all the features (figure 4c). Next, in order to determine the hyperparameters one should perform a Nested Cross Validation using the Fisher-score ranking method for the features inside the inner loop. Then, we think that it would be a good choice to eliminate the redundant information due to correlated features by selecting only the first 16 (figure 5b) and project them using principal component analysis.

Another consideration is that it seems to be useful to exploit the combination of supervised and unsupervised learning, such as a PCA and a fisher ranking for example, because the first one allows to reduce the size of the data by looking for a larger clustering, and the other one takes into account the labels and thus can find the real “division”.

This guidelines for the final classifier are given by what we could learn from the analysis done in this mini project. Unfortunately, due to complications with the MATLAB code and a lack of time planning, we could not give life to our idea of a final model.

However the best results was achieved with the combination of diagquadratic classifier and Fisher ranking method reaching a class error of 0.2165 using 31 features. In fact, diagquadratic seem to be the classifier that segregate best our features, linear was excluded because the difference between the train and test class error was huge (around 0 for the train and 0.3 for the test), whereas diagquadratic was more stable.