MSc in Computer Engineering
Intelligent Systems
2018/2019

Emotion Recognition

Lorenzo Simi
Matteo Suffredini

Contents

1 Introduction

1.1 The problem

The purpose of the experiment was to determine the emotions experienced by various people through the use of signals produced by various biometric sensors. The sensors used are 40: 32 are EEG, 2 are EOG, 2 are EMG and the others are GSR, respiration belt, plethysmograph and temperature.
There were created 40 videos, each of them associated with a different emotion and with a duration of one minute. 32 people were asked to describe them in terms of arousal and valence, using the Self-Assessment Manikin (SAM), that has a range of value from 1 to 9.
The aim of the project is to design and develop two multi-layer perceptron (MLP) artificial neural networks that accurately estimate a person's valence and arousal levels.

1.2 Provided data

The collected data were included in a dataset consisting of 32 parts, each containing the experiments conducted on a single person, in particular the data collected by the sensors and the values of arousal and valence. For each video attempt, 8064 samples were collected for each sensor, corresponding to 63 seconds sampled at 128 hz.

2 Data preprocessing

First of all the dataset samples were cut in the initial part, as each test conducted is influenced by the previous test.
The second step was to choose how many and which windows to use to get data ready for feature extraction. Initially we decided to use a single window including all the samples (excluding those previously cut), after that we decided to use 3 windows (the first including the first part of the samples, the second including the second part and the third including the central part).

2.1 Feature extraction

The feautures considered for feature extraction are 6 in the time domain (mean min max kurtosis skewness median) and 2 in the frequency domain (meanfreq and medianfreq). All the extracted features are normalized by means of the function *normalize*, which returns the vectorwise z-score of the data in the extracted features matrix with center 0 and standard deviation 1.

2.2 Feature selection

From the set of extracted features a subset has been selected exploiting the *sequentialfs* function and using as support network a fitnet with a hidden layer of 10 neurons

2.3 Training phase

For the training phase of the net the objective is to achieve the lowest MSE value on the test set, because it represents the real performance.

3 Testing phase arousal

All the following tests are made with one hidden layer, except for experiment 3.3.

3.1 One window

In Table 1 are shown the features selected by the *sequentialfs* and their sensors taken into account.

Features	Sensors
mean	-
min	22 - 29
max	-
kurtosis	7 -33
skewness	27
median	-
mean freq	2 - 8
median freq	-

Table 1: Arousal one window - selected features

3.1.1 Levenberg-Marquardt (trainlm)

HN	MSE	Regression	Training
21	3.1241	0.3683	trainlm

Table 2: Arousal one window lm - performance

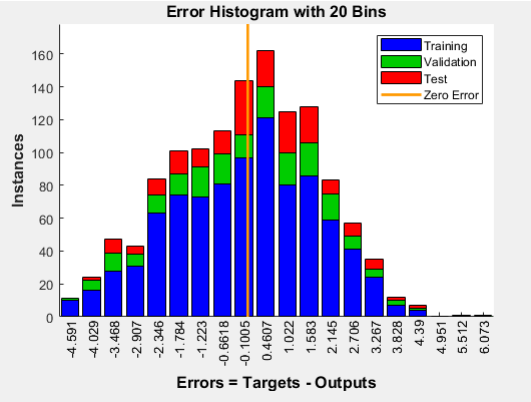

Figure 1: Arousal one window lm - error histogram

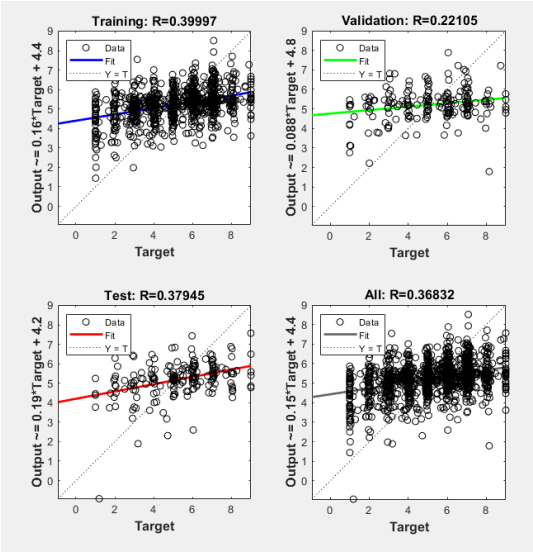

Figure 2: Arousal one window lm - regression

As can be seen from the graphs and table, the error histogram has the typical bell shape with a peak at the centre which corresponds to the zero error. This is good because it confirms that the errors on the test set (indicated in red) are concentrated around zero and we do not have very large errors in points away from zero. The MSE is about 3.12 and the regression in this case is not very high, as you get a value of about 37%.

3.1.2 Bayesian Regularization (trainbr)

HN	MSE	Regression	Training
19	3.7187	0.37778	trainbr

Table 3: Arousal one window br - performance

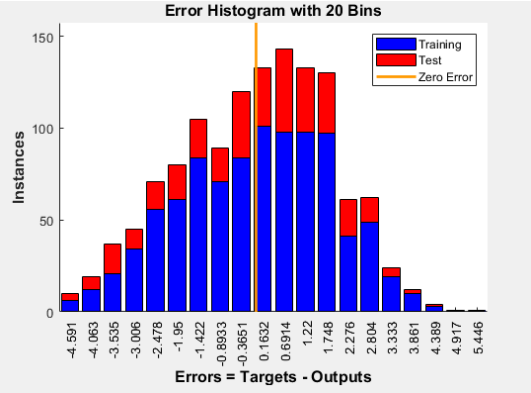

Figure 3: Arousal one window br - error histogram

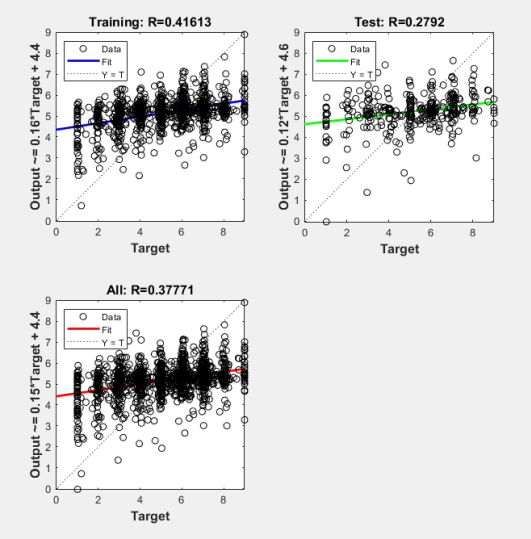

Figure 4: Arousal one window br - regression

In the case of *Bayesian Regularization* algorithm, the validation set is not present because the algorithm itself validates the data internally.

The MSE is about 3.72 and the regression in this case is not very high, as you get a value of about 38% on the test set.

The bell shape of the error histogram remains basically the same as in the previous case.

3.2 Three windows

Seeing the results obtained in previous tests, we tried to find a way to increase performance and so we moved on to consider three time windows for extracting features from samples.

First of all, we performed a test using *Levenberg-Marquardt* as a training algorithm, but we got worse results than with *Bayesian Regularization*, so from now on we always used *Bayesian Regularization* for training.

Features	Sensors
mean	2 - 19 -35
min	25 - 29
max	15
kurtosis	2
skewness	17
median	20
mean freq	4 - 12 -18
median freq	21

Table 4: Arousal three windows - selected features

HN	MSE	Regression	Training
8	3.327	0.4328	trainbr

Table 5: Arousal three windows - performance

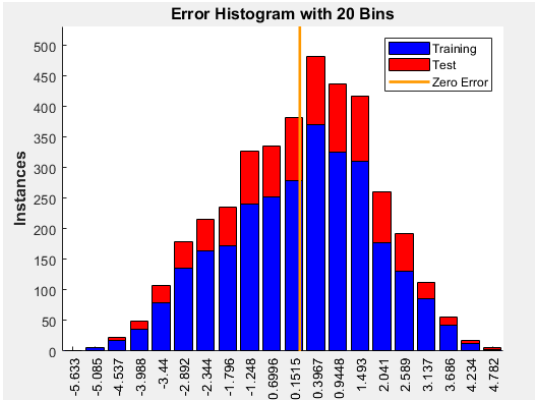

Figure 5: Arousal three windows - error histogram

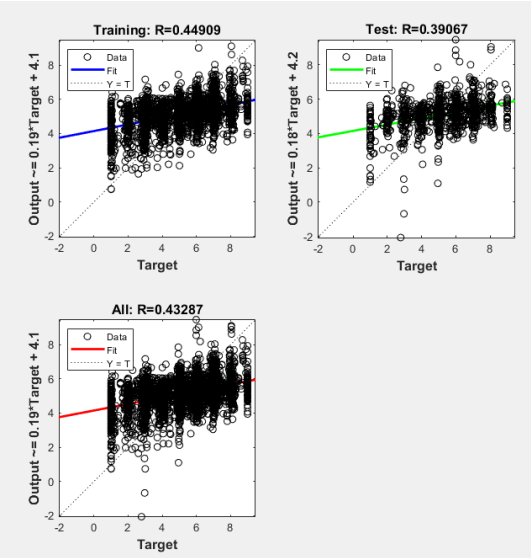

Figure 6: Arousal three windows - regression

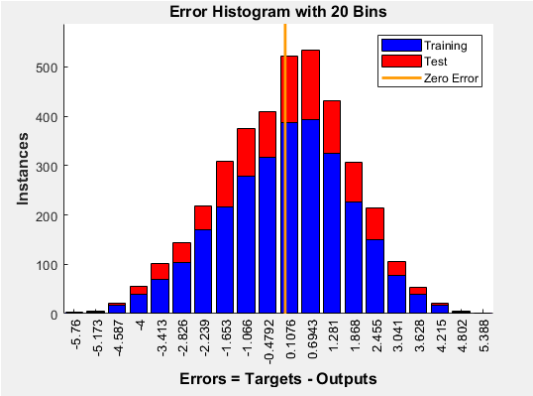

Figure 7: Arousal three windows 2hl - error histogram

The MSE is about 3.33 and the regression in this case is about 43%.

The values in the error histogram appear to be approaching zero.

3.3 Three windows with 2 hidden layers

HN 1	HN 2	MSE	Reg	Training
10	4	3.2371	0.4949	trainbr

Table 6: Arousal three windows 2hl - performance

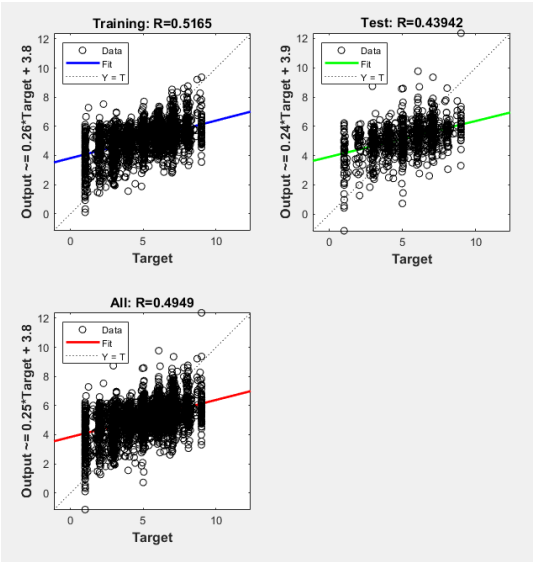

Figure 8: Arousal three windows 2hl - regression

The MSE is about 3.24 and the regression in this case is about 49%.

The shape of the histogram is slightly better than in the previous case because of the greater symmetry.

3.4 Three windows with balanced dataset

We have divided the arousal values, saving the respective original indexes, into 0.5 amplitude intervals starting from 1 and ending at 9. Then we counted the number of instances for each interval and randomly undersampled the intervals that had values in excess of the mean, calculated by dividing the total number of samples by the number of intervals.

We have cut the values in greater quantity from the original dataset, after which we have used 3 time windows as in the previous case.

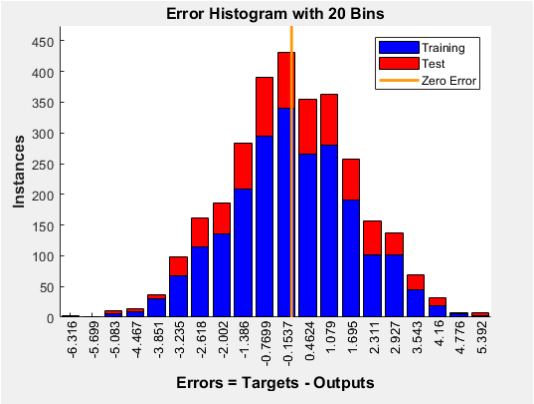

Figure 9: Arousal three windows balanced -error histogram

Features	Sensors
mean	15
min	8 - 21
max	18
kurtosis	22
skewness	-
median	16 - 30 - 39
mean freq	1 - 12 - 37
median freq	-

Table 7: Arousal three windows balanced - selected features

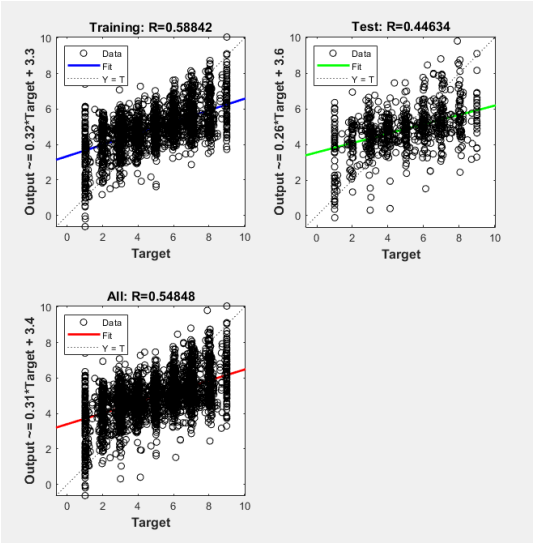

Figure 10: Arousal three windows balanced - error regression

HN	MSE	Regression	Training
21	3.8982	0.5485	trainbr

Table 8: Arousal three windows balanced - performance

The MSE is about 3.90 and the regression in this case is about 55%.

The bell shape of the error histogram remains basically the same as in the previous case.

3.5 Three windows with 3-fold cross-validation

The selected features used are from Table 4.

We used 3-fold cross validation, because using more than 3 folds we do not have enough samples available.

HN	MSE	Regression	Training
7	2.8282	0.5622	trainbr

Table 9: Arousal three windows CV - performance

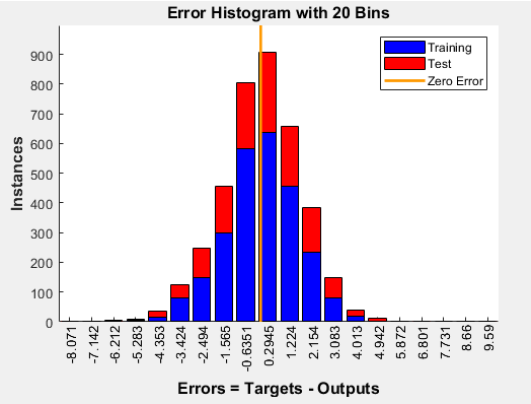

Figure 11: Arousal three windows CV - error histogram

The MSE is about 2.82 and the regression in this case is about 56%.

In this case we have a great improvement on the shape of the histogram, which narrows very much around zero.

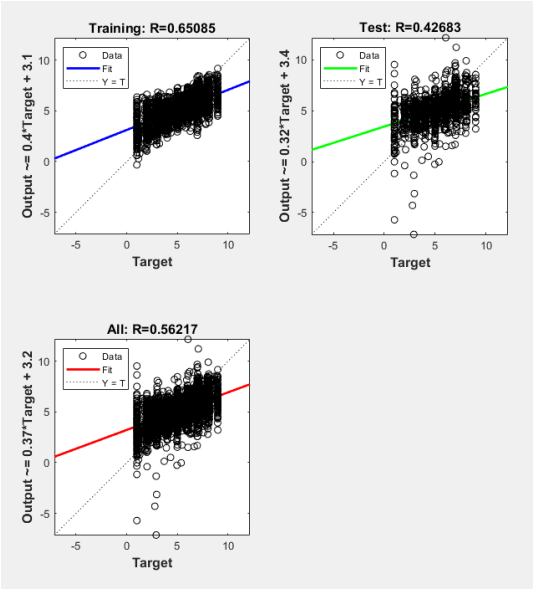

Figure 12: Arousal three windows CV - regression

3.6 RBF

The selected features used are from Table 4.

HN	MSE	Regression
324	2.8281	0.5538

Table 10: Arousal three windows RBF - performance

At the end of the testing phase carried out on the fitnet network, we used the best result obtained in terms of MSE to create a similar network based on *RBF*. We used the newrb function and obtained a network with 324 neurons in the hidden layer and a regression of about 55%.

3.7 Classifier

In order to better understand the values that cause a larger error in the network, we have built a *classifier* using the MATLAB *patternnet* function and dividing the samples into three classes: the first with values between 1 and 3, the second with values between 3 and 6 and the third with values between 6 and 9.

The selected features used are from Table 4.

HN	MSE	Training
20	0.2070	trainbr

Table 11: Arousal three windows classifier - performance

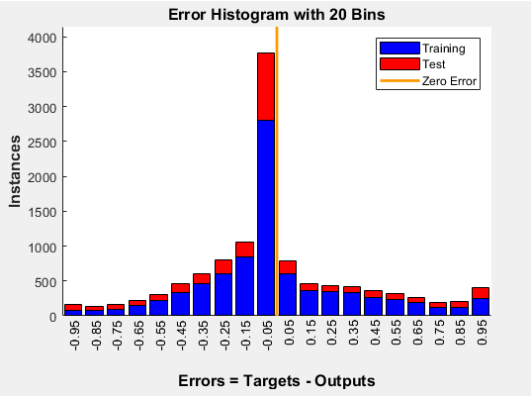

Figure 13: Arousal three windows classifier - error histogram

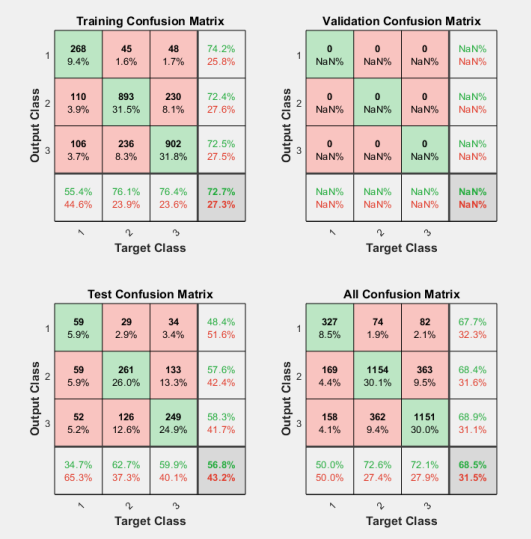

Figure 14: Arousal three windows classifier - confusion matrix

As can be seen from the confusion matrix, especially from the test set one, the best results are obtained in the medium-high part of the values. This is not surprising, because by analyzing the dataset you can see that it is unbalanced in the middle. Overall the network has an accuracy of 56.8% on the test set.
In this case the shape of the histogram is almost perfect since there is only one high column near the zero error.

4 Testing phase valence

All the following tests are made with one hidden layer, except for experiment 4.3.

4.1 One window

In Table 12 are shown the features selected by the *sequentialfs* and their sensors taken into account.

Features	Sensors
mean	-
min	20 - 24
max	-
kurtosis	-
skewness	15 - 27 - 36
median	-
mean freq	22
median freq	-

Table 12: Valence one window - selected features

4.1.1 Levenberg-Marquardt (trainlm)

HN	MSE	Regression	Training
20	3.9528	0.2916	trainlm

Table 13: Valence one window lm - performance

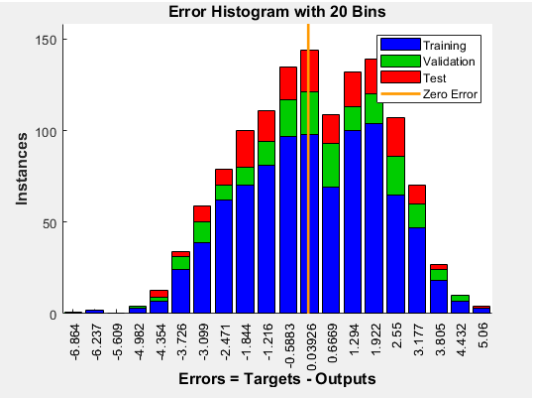

Figure 15: Valence one window lm - error histogram

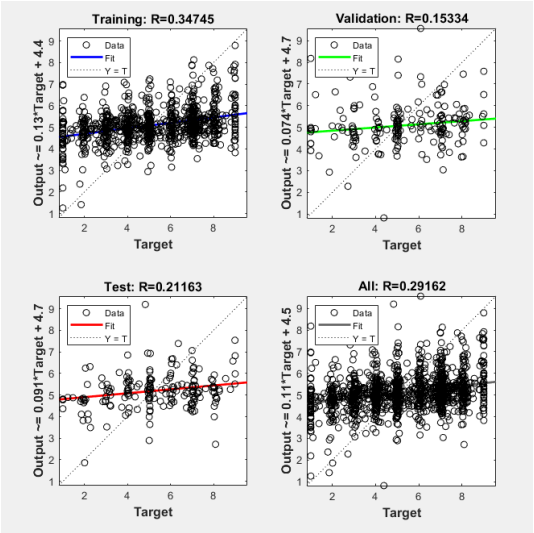

Figure 16: Valence one window lm - regression

As can be seen from the graphs and table, the error histogram has the typical bell shape with a peak at the centre which corresponds to the zero error. This is good because it confirms that the errors on the test set (indicated in red) are concentrated around zero and we do not have very large errors in points away from zero.

The MSE is about 3.95 and the regression in this case is not very high, as you get a value of about 29%.

4.1.2 Bayesian Regularization (trainbr)

HN	MSE	Regression	Training
10	4.0448	0.1115	trainbr

Table 14: Valence one window br - performance

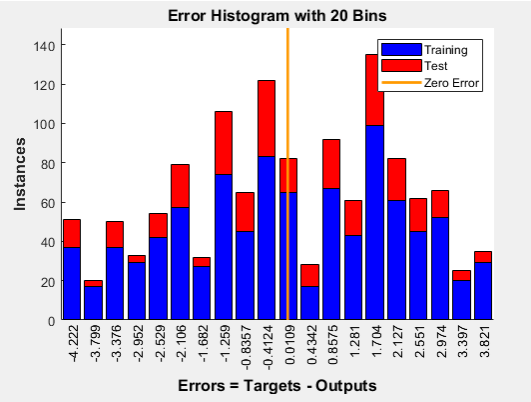

Figure 17: Valence one window br - error histogram

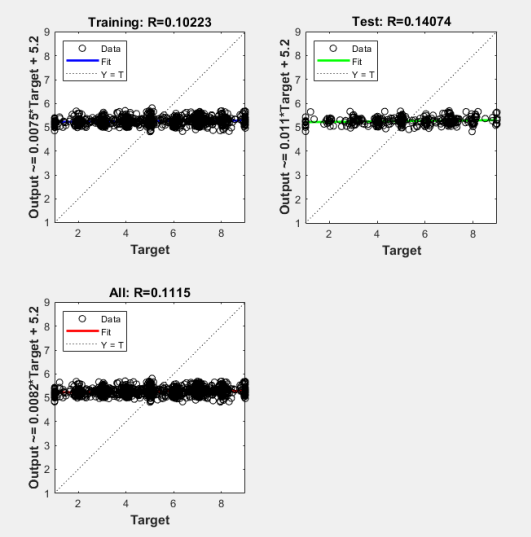

Figure 18: Valence one window br - regression

As can be seen from the graphs and table, the error histogram has not a bell shape. This is bad, because we have a high dispersion of error values and they are not concentrated in the middle corresponding to zero error.

The MSE is about 4.04 and the regression in this case is not very high, as you get a value of about 11%.

4.2 Three windows

Seeing the results obtained in previous tests, we tried to find a way to increase performance and so we moved on to consider three time windows for extracting features from samples.

First of all, we performed a test using *Levenberg-Marquardt* as a training algorithm, but we got worse results than with *Bayesian Regularization*, so from now on we always used *Bayesian Regularization* for training.

Features	Sensors
mean	2 - 8
min	-
max	2 - 7 - 11 - 23 - 33
kurtosis	-
skewness	19
median	23 - 24 - 34
mean freq	18 - 31
median freq	24 - 28

Table 15: Valence three windows - selected features

HN	MSE	Regression	Training
9	3.8413	0.4417	trainbr

Table 16: Valence three windows - performance

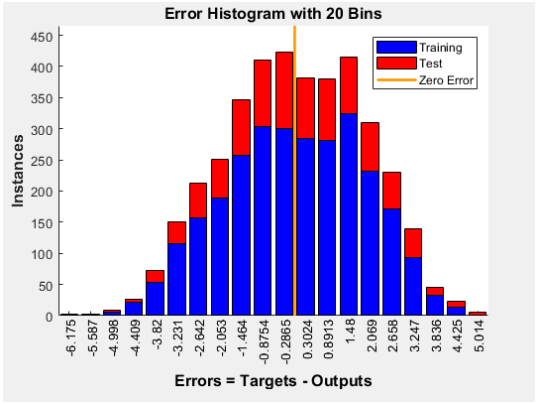

Figure 19: Valence three windows - error histogram

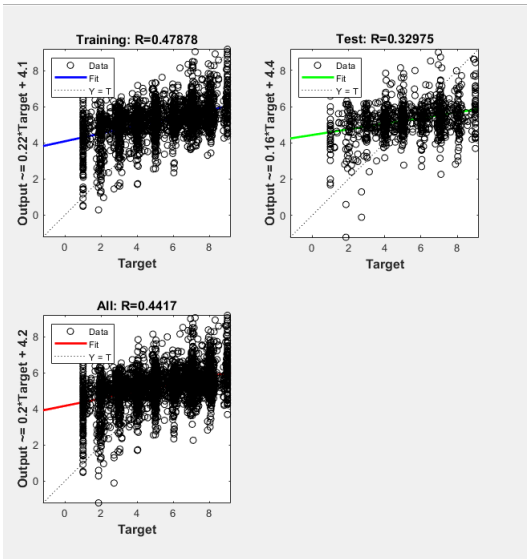

Figure 20: Valence three windows - regression

The MSE is about 3.84 and the regression in this case is about 44%.

The values in the error histogram appear to be approaching zero.

4.3 Three windows with 2 hidden layers

HN 1	HN 2	MSE	Reg	Training
7	5	3.7968	0.4667	trainbr

Table 17: Valence three windows 2hl - performance

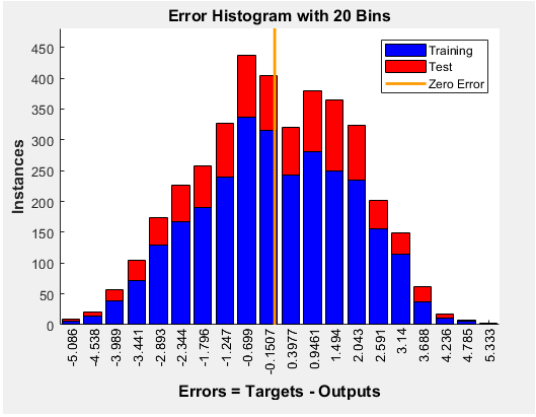

Figure 21: Valence three windows 2hl - error histogram

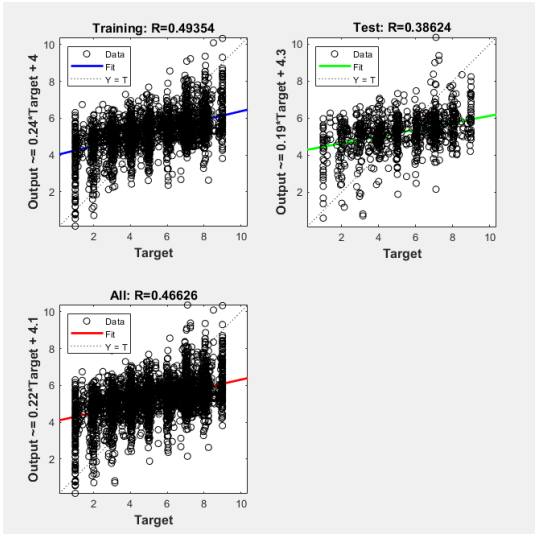

Figure 22: Valence three windows 2hl - regression

The MSE is about 3.80 and the regression in this case is about 47%.

The bell shape of the error histogram remains basically the same as in the previous case.

4.4 Three windows with balanced dataset

We have divided the valence values, saving the respective original indexes, into 0.5 amplitude intervals starting from 1 and ending at 9. Then we counted the number of instances for each interval and randomly undersampled the intervals that had values in excess of the mean, calculated by dividing the total number of samples by the number of intervals.

We have cut the values in greater quantity from the original dataset, after which we have used 3 time windows as in the previous case.

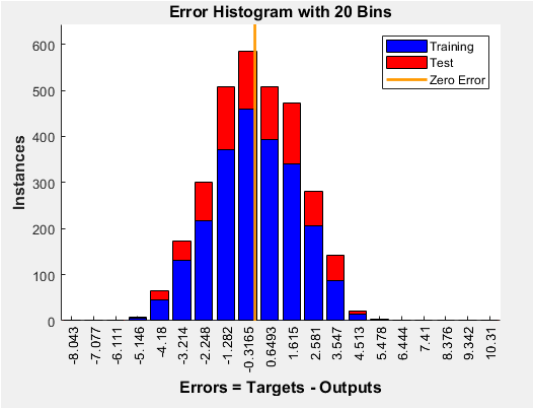

Figure 23: Valence three windows balanced - error histogram

Features	Sensors
mean	14 - 19 - 26
min	26
max	12 - 18 - 29
kurtosis	-
skewness	38
median	2 - 34
mean freq	22
median freq	-

Table 18: Valence three windows balanced - selected features

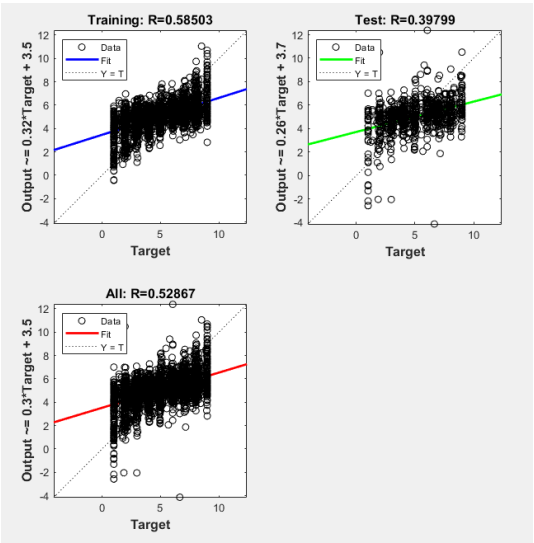

Figure 24: Valence three windows balanced - regression

HN	MSE	Regression	Training
25	4.4968	0.5287	trainbr

Table 19: Valence three windows balanced - performance

The MSE is about 4.50 and the regression in this case is about 53%.

In this case we have a great improvement on the shape of the histogram, which narrows very much around zero.

4.5 Three windows with 3-fold cross-validation

The selected features used are from Table 15.
We used 3-fold cross validation, because using more than 3 folds we do not have enough samples available.

HN	MSE	Regression	Training
18	3.3190	0.5207	trainbr

Table 20: Valence three windows CV - performance

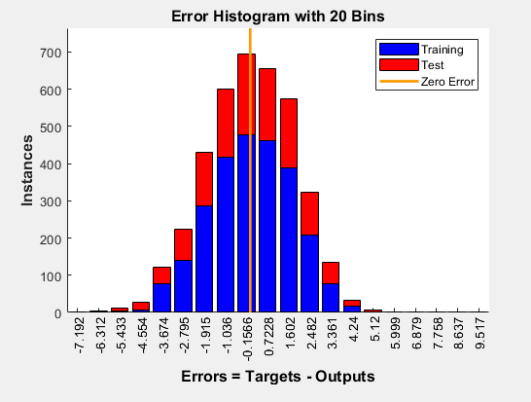

Figure 25: Valence three windows CV - error histogram

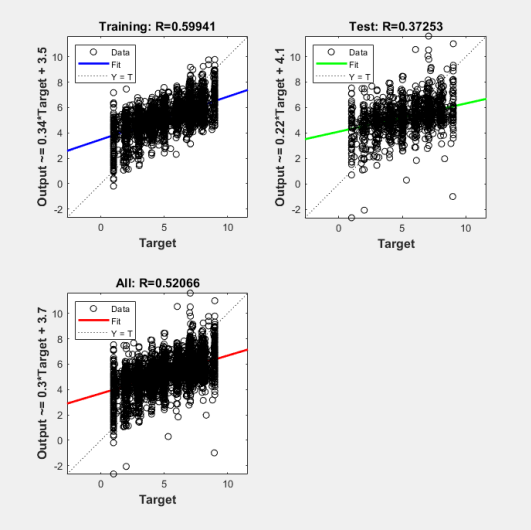

Figure 26: Valence three windows CV - regression

The MSE is about 3.32 and the regression in this case is about 52%.
The bell shape of the error histogram remains basically the same as in the previous case, but performance increases considerably.

4.6 RBF

The selected features used are from Table 15.

HN	MSE	Regression
270	3.3099	0.5200

Table 21: Valence three windows RBF - performance

At the end of the testing phase carried out on the fitnet network, we used the best result obtained in terms of MSE to create a similar network based on *RBF*.
We used the newrb function and obtained a network with 270 neurons in the hidden layer and a regression of about 52%.

4.7 Classifier

In order to better understand the values that cause a larger error in the network, we have built a *classifier* using the MATLAB *patternnet* function and dividing the samples into three classes: the first with values between 1 and 3, the second with values between 3 and 6 and the third with values between 6 and 9.
The selected features used are from Table 15.

HN	MSE	Training
20	0.2174	trainbr

Table 22: Valence three windows classifier - performance

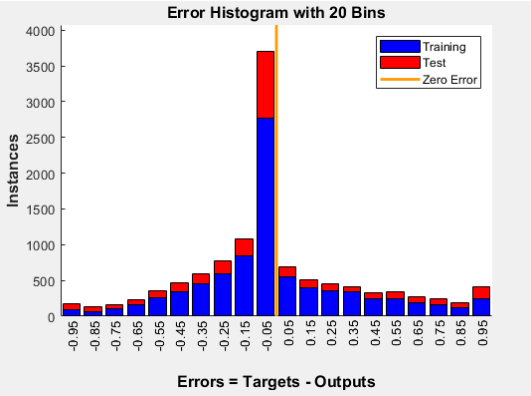

Figure 27: Valence three windows classifier - error histogram

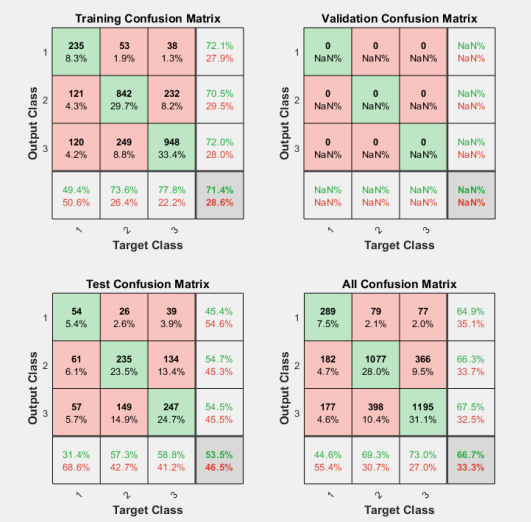

Figure 28: Valence three windows classifier - confusion matrix

As can be seen from the confusion matrix, especially from the test set one, the best results are obtained in the medium-high part of the values. This is not surprising, because by analyzing the dataset you can see that it is unbalanced in the middle. Overall the network has an accuracy of 53.5% on the test set.

5 Conclusions

In both cases it is easy to see that using only one time window the performances are not acceptable. It is therefore necessary to use 3 windows to obtain acceptable results that allow us to increase the performances, respect to the best case of one window, by 16% for arousal and 51% for valence in regression, but we have a worsening in terms of MSE on test set, which is the most important performance.

For this reason we have decided to make a test with a network with two layers, but obtaining similar results that do not justify the greater complexity of the new network.

Trying to manually balance the dataset leads to an improvement in terms of regression, but a worsening in MSE on test set.

The best performance is obtained with 3-fold cross validation, since we have an improvement of 15% for arousal and 14% for valence in terms of MSE and an improvement respectively of 30% and 18% in terms of regression, w.r.t. the best results in previous cases.

By training an *RBF* network that has an MSE requirement equal to the best value previously obtained by the fitnet network, the complexity of the network is significantly greater, by one order of magnitude, for the number of neurons in the hidden layer. For this reason, using an *RBF* network is not convenient.

Using a *classifier* instead of a *fitnet* is advantageous in terms of performance, as there is an improvement in MSE of 93% for both arousal and valence.

6 Possible improvements

Since the *classifier* provides the best results, it may be appropriate to try to train a classifier using a balanced dataset in order to further improve performance.

Given that in all the experiments conducted were taken into account in most cases sensors related to EEG, it might therefore be interesting to proceed to new experiments considering only the data of those sensors.