POLITECNICO DI TORINO

FIRST REPORT ON THE ICT for HEALTH LABORATORY

A.Y. 2018-2019

Professor:

Visintin Monica

Student:

Bellone Lorenzo

Summary

Introduction

Parkinson is the name attributed to a neurodegenerative disease affecting dopaminergic neurons that are located in a part of the brain called *substantia nigra.*

The dopamine is a neurotransmitter responsible for the coordination of movements and the control of vocal chords and vocal tract that influences the speech. Another role of the dopamine concerns the modulation of pleasure stimulated by several activities such as feeding, sex or the abuse of drugs. Furthermore, the neurotransmitter manages functions related to memory, attention and self-cognition.

Given the importance of dopamine as explained above, people affected by the disease present lack of control of their muscles, showing tremor and problems in walking and, more in general, starting a movement. Another main result of the degeneration of dopaminergic neurons is the difficulty in correctly speaking.

The goal of all the existing treatments is to weaken the occur of the given symptoms, since the cure of the illness has not been discovered yet.

The drugs dispensed to the patients are Levodopa-based, a chemical substance that increases dopamine levels. The beneficial effects of Levodopa last 4 to 5 years, during which the symptoms tend to disappear: this phase is called "Honey Moon". After this period the described chronic symptoms appear again.

A method to evaluate the severity of Parkinson's disease has been named Unified Parkinson's Disease Rating Scale (UPDRS), which is held by a neurologist. The method consists in asking the patient to perform a series of pre-established movements in order to give a score for each single performance. The final grade will be given by adding the results together.

The UPDRS has many controversial aspects, because it requires a considerable amount of time and the results may differ from a neurologist to another.

According to these contras, several articles have shown that more practical methods may be possible, due to the fact that the parameters of voice are potentially correlated to the UPDRS features, even if this solution is not always true, since the illness might not influence all the patients' voices. In order to find these correlations, the method of regression is used.

A set of observed values is used to predict the real measure of the UPDRS: the first ones are called regressors while the data to be predicted are the regressands. In this report the regressors will be identified as **x(n)**, a set of independent variables that causes the regressand, which is the dependent variable y(n). Concerning Parkinson, **x(n)** contains a series of features referring to a specific patient that allows to estimate y(n) in a quicker and less expensive exam. Linear regression will be used to make this estimation, assuming a linear dependency between **x(n)** and y(n). However, this assumption results incorrect since the presence of noise has always to be considered, for this reason y(n) is related to **x(n)** through the equation:

$$y(n) = \boldsymbol{x}(\boldsymbol{n})^T \cdot \boldsymbol{w} + \nu(\mathrm{n}) \tag{1.1}$$

In the relation 1.1, y(n) is the UPDRS of the n^{th} patient, **x(n)** is a column vector containing the features of the n^{th} patient, **w** is a weight vector made of unknown values, which indicates the linear dependency between y(n) and **x(n)**, and ν(n) is a measurement error.

N (number of patients) measurements are performed and the UPDRS values are stored in the vector **y**; the matrix **X** with F rows (number of features) and N columns is defined and, more in general, the equation 1.1 becomes:

$$\boldsymbol{y} = \boldsymbol{X}^T \cdot \boldsymbol{w} + \boldsymbol{\nu}$$

(1.2)

The goal of the linear regression is to find the best weight vector **w** that allows to minimize the square error, assuming that the measurement error ν is small:

$$e(\boldsymbol{w}) = ||\boldsymbol{y} - \boldsymbol{X} \cdot \boldsymbol{w}||^2$$

(1.3)

Once that the best model has been identified using the given **y** and **X**, it could be useful in the prediction of new data. It may happen that the evaluation of a very small error in this training phase leads to a larger one during the prediction of future outputs (running phase), this situation is known as overfitting. Usually the given data are split into a training and a validation sets for the evaluation of the best model, and a test-set, with which the found **w** is used to estimate y(n) although its real value is already known. Furthermore, it is useful to evaluate the mean square error (MSE) for each dataset, defined as:

$$e_{MSE} = ||\boldsymbol{X_s}\boldsymbol{w} - \boldsymbol{y_s}||^2 / Np_s$$

(1.4)

An average of the committed errors and the presence of overfitting can be evaluated through the MSE. In the relation 1.4, $\boldsymbol{X_s}$ and $\boldsymbol{y_s}$ correspond to the subsets of the given **X** and **y** and Np_s is the number of measurements for each dataset.

In the first lab **y** and **X** are already provided in a dataset that stores eighteen important diagnostical features regarding Parkinson's disease for more than five-thousand patients. Among these features there is UPDRS and some measurements of the voice parameters. The aim is to regress UPDRS using the other available data.

Linear regression will be used through six different methods in order to compare the distinct solutions during the final phase. The methods are:

- The Linear Least Square Estimation;
- The ridge regression;
- The gradient algorithm;
- The steepest descent algorithm;
- The stochastic gradient;
- The conjugate gradient.

The Linear Least Square Estimation (LLS)

As already discussed in the introduction, the goal of linear regression is to minimize the square error shown in the equation 1.3. This function can be also rewritten as follows:

$$e(\boldsymbol{w}) = [\boldsymbol{y} - \boldsymbol{X} \cdot \boldsymbol{w}]^{\boldsymbol{T}} [\boldsymbol{y} - \boldsymbol{X} \cdot \boldsymbol{w}]$$

(2.1)

At this point it is necessary to evaluate the gradient of e(**w**) and set it equal to 0 in order to find the minimum error:

$$\nabla e(\boldsymbol{w}) = -2\boldsymbol{X}^T\boldsymbol{y} + 2\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} = \boldsymbol{0}$$

(2.2)

At the end, **w** will be given by the pseudo-inverse of the matrix **X**, defined as $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$, multiplied by the vector **y**:

$$\boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

(2.3)

The training data got from the patients' dataset are used for the evaluation of w (Figure 1) that relies on the Linear Least Square Estimation (LLS).

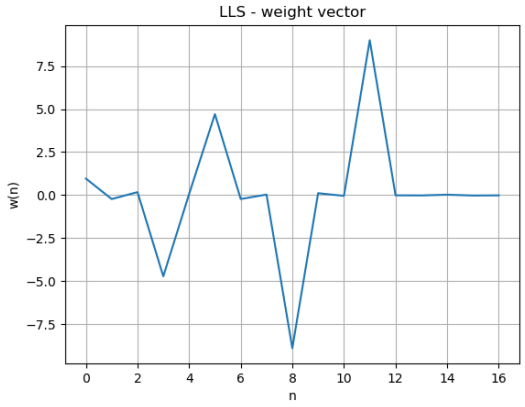

*Figure 1 - **w** obtained with LLS*

It is convenient to verify that the results correspond to the regressand feature multiplying the matrix **X** by the gotten **w**. Given the fact that the predicted UPDRS should be equal to the real one, the graphics of the figures 2 show that these two values match in both the train and test set, with a margin of error not to be underestimated.

The predicted values of UPDRS will be identified as y_hat, while the real values of UPDRS as y.

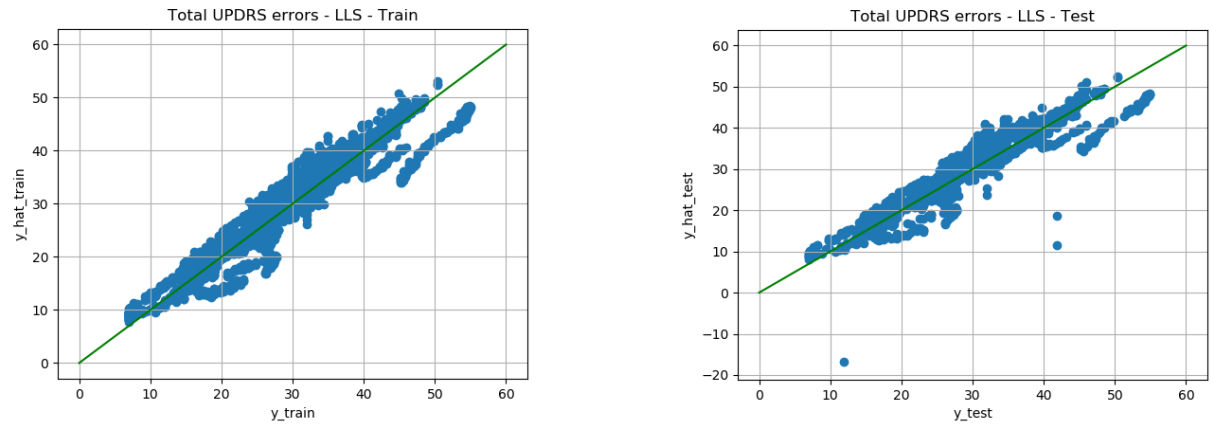

Figure 2 – Linear Least Square Estimation: on the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

Moreover, it is possible to have a better perception of the range and the distribution of the committed error among the patients, as presented in the histograms in figure 3.

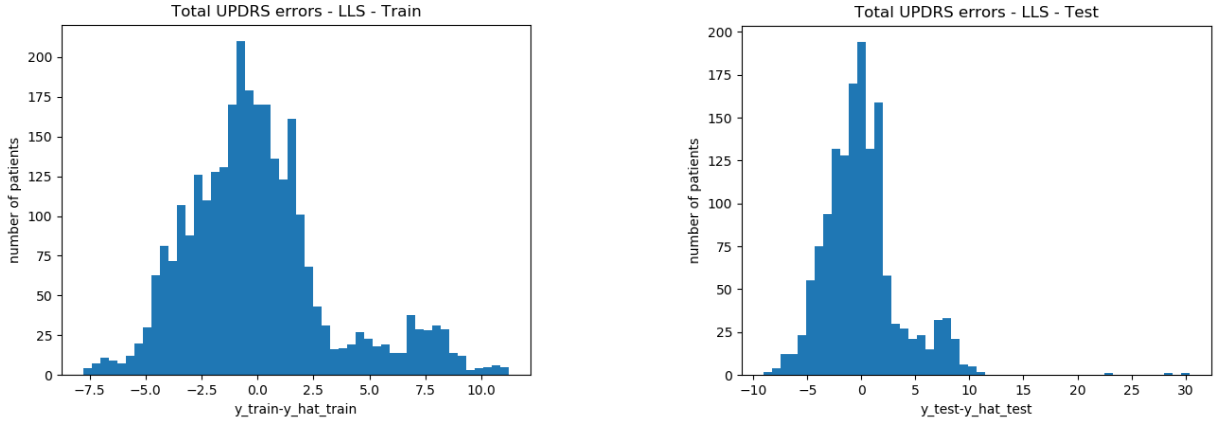

Figure 3 -Linear Least Square Estimation: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

The Ridge Regression (RR)

Sometimes there might be some numerical problems due to the inversion of the matrix $\boldsymbol{X}^T\boldsymbol{X}$ when the LLS is applied. It is convenient to add a constraint to the problem, which means to add a constant λ in the diagonal entries of the matrix $\boldsymbol{X}^T\boldsymbol{X}$. The function to be minimized becomes:

$$e(\boldsymbol{w}) = ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}||^2 + \lambda||\boldsymbol{w}||^2$$

(3.1)

The optimum weight vector **w** can be found through the evaluation of the pseudo-inverse, as already done with the LLS:

$$\boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X} + \lambda\boldsymbol{I})^{-1} \cdot \boldsymbol{X}^T\boldsymbol{y}$$

(3.2)

The resulting matrix $\boldsymbol{X^T X + \lambda I}$ obtained through the regularization just presented in the equation 3.2 is always invertible, without any type of numerical problems.

However, the square error of the original problem (equation 1.3) increases, since that the function of the error has changed, but the model does not pose any issues in the inversion of the matrix.

The problem is to define which is the correct value of λ. It might be convenient to use the training set to get **w** and the validation set to measure the error according to different values of λ. If λ is set equal to zero, the method gives the same performance of the LLS, but when λ increases, the square error also increases for the training dataset and, theoretically, it decreases for the validation dataset.

For the specific provided dataset, the mean square error in the validation set starts to increase when $\lambda = 2$, as shown in figure 4.

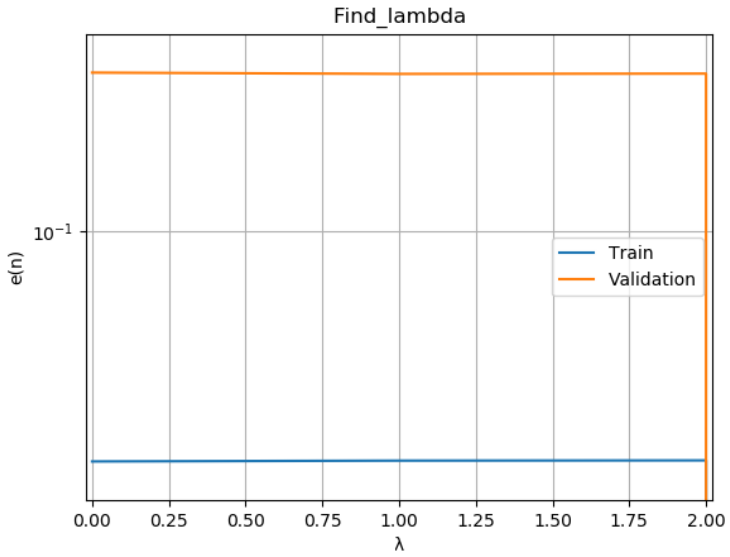

Figure 4 - The mean square error of the training and the validation set as a function of λ, stopped at λ =2

Once λ has been evaluated, the value of the optimum weight vector **w** can be assessed, as already done in the previous method.

It should be noted that the vector **w** resulted in the Ridge Regression (Figure 5) is quite different from the one evaluated in the LLS (Figure 1), though the model derived has a smaller square error in the prediction of future outputs.

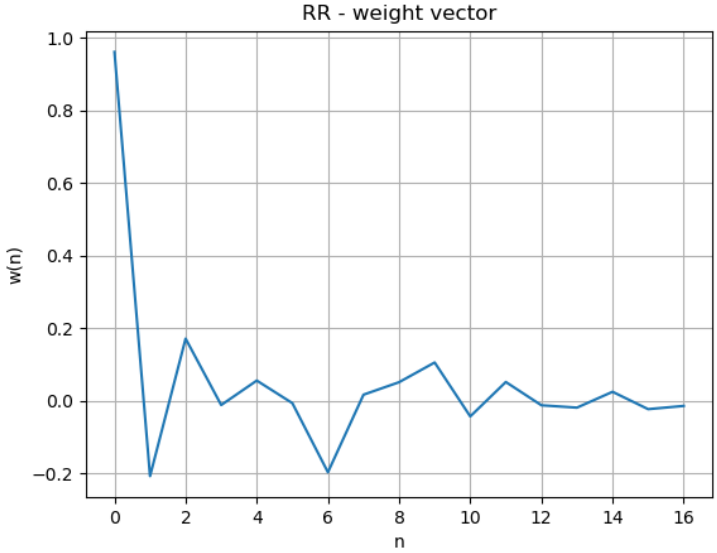

*Figure 5 - **w** obtained in the Ridge Regression*

As in figure 2, the graphs of figure 6 show the existence of a linear correlation between **X** and **y**, but still with a certain margin of error.

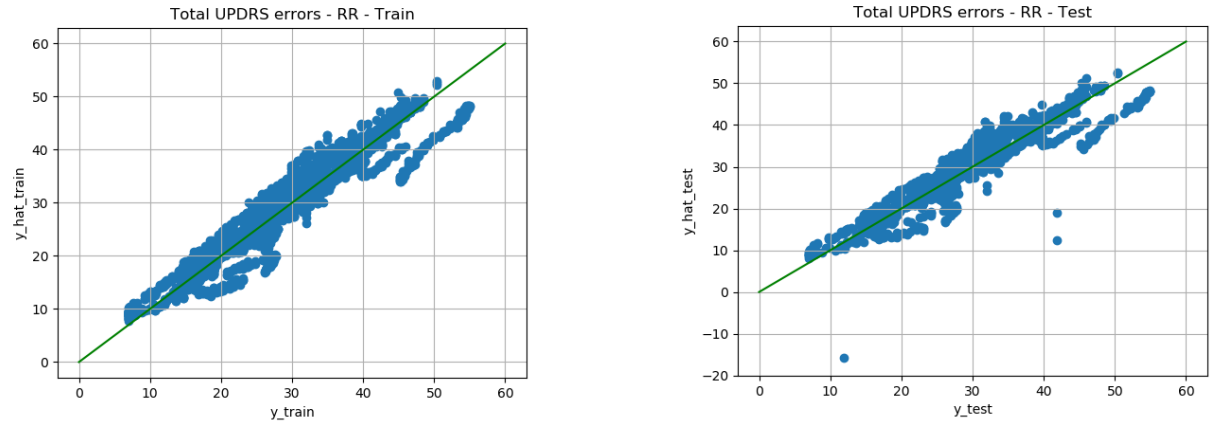

Figure 6 - Ridge Regression: On the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

The distribution of the error among the patients is presented in the histograms in figure 7.

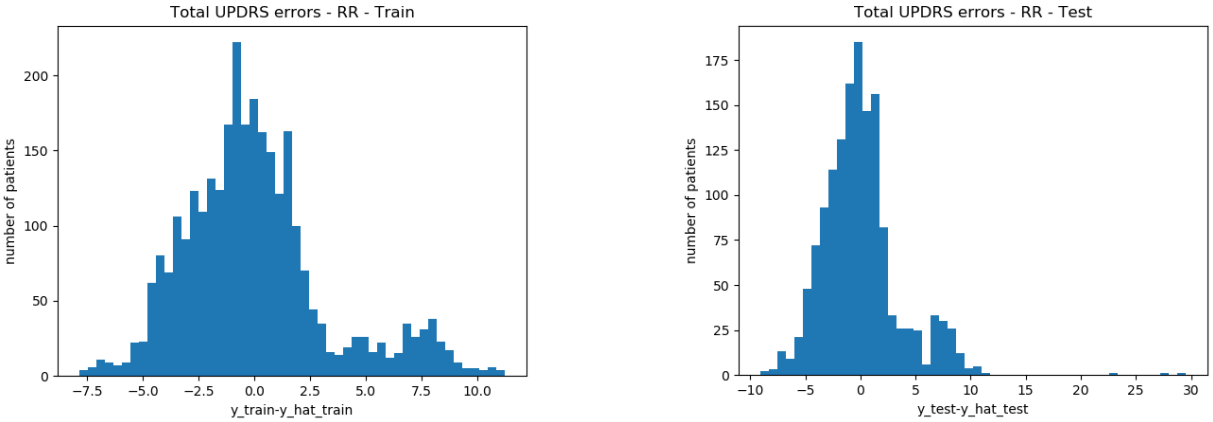

Figure 7 - Ridge Regression: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

The Gradient Algorithm (GA)

The aim of the LLS method is to provide the value of **w** in just one iteration, but there might be some numerical problems as previously described. A possible solution is to use the Ridge Regression adding a constraint to avoid the overfitting, but anyway the evaluation of the pseudo-inverse of the matrix **X** might be expensive from a computational point of view. The other solution is to use some iterative algorithms such as the Gradient Algorithm (GA).

In the report it will be considered that the reader already knows what the gradient of a function is and how it works, furthermore, the complete algorithm can be found on the Web.

In this method the evaluation of **w** is made through many steps until a stop condition is satisfied. For each iteration the gradient of the function 1.3 is computed and the vector **w** is updated moving along the opposite direction of the gradient. The size of the different steps is established by the learning coefficient γ:

$$\boldsymbol{w_k} = \boldsymbol{w_{k-1}} - \gamma \nabla e(\boldsymbol{w_{k-1}}) \tag{4.1}$$

The correct value of γ must be carefully estimated because it might bring many issues in the evaluation of the optimum weight vector.

The advantage of using an iterative algorithm concerns the possibility to stop the iterations earlier to avoid overfitting. The training part is used to evaluate **w** for a certain number of iterations and, during these iterations, the found **w** is also used to estimate the committed error in the validation dataset. The overfitting occurs when the error begins to increase: at this point, the gradient algorithm needs to be stopped.

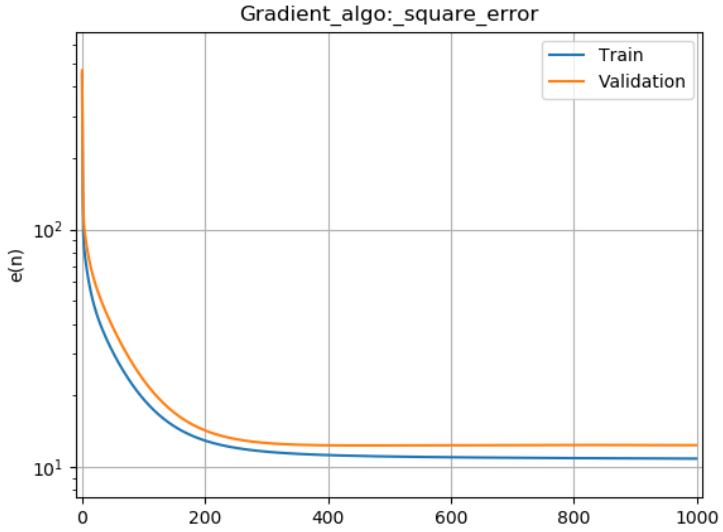

Figure 8 – Gradient Algorithm: the trend of the mean square errors evaluated in each iteration for the training and the validation set.

According to the given dataset, the graph of the figure 8 shows that overfitting does not occur since the MSE of the validation set never increases. In this case, setting a reasonable number of iterations might be an acceptable stop condition. The value of the optimum **w** (figure 9) is taken as the one evaluated in the last iteration.

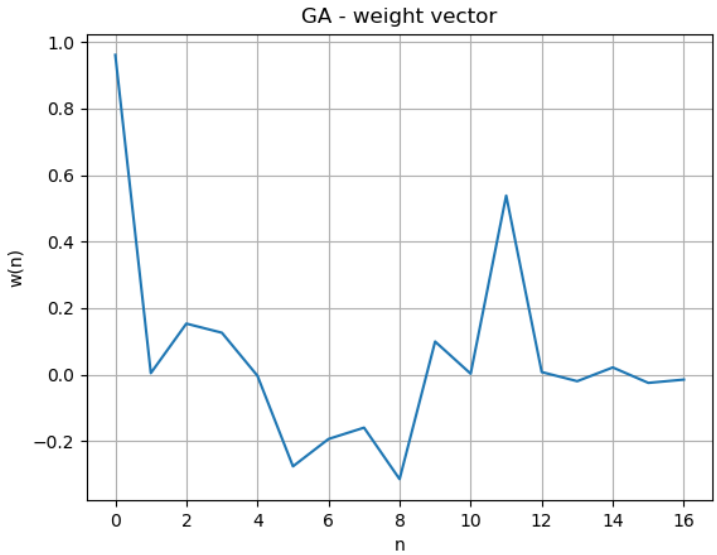

*Figure 9 - **w** obtained in the Gradient Algorithm*

As already done before, it is necessary to verify if the predicted values match with the real UPDRS (Figure 10) plotting the real versus the predicted regressands.

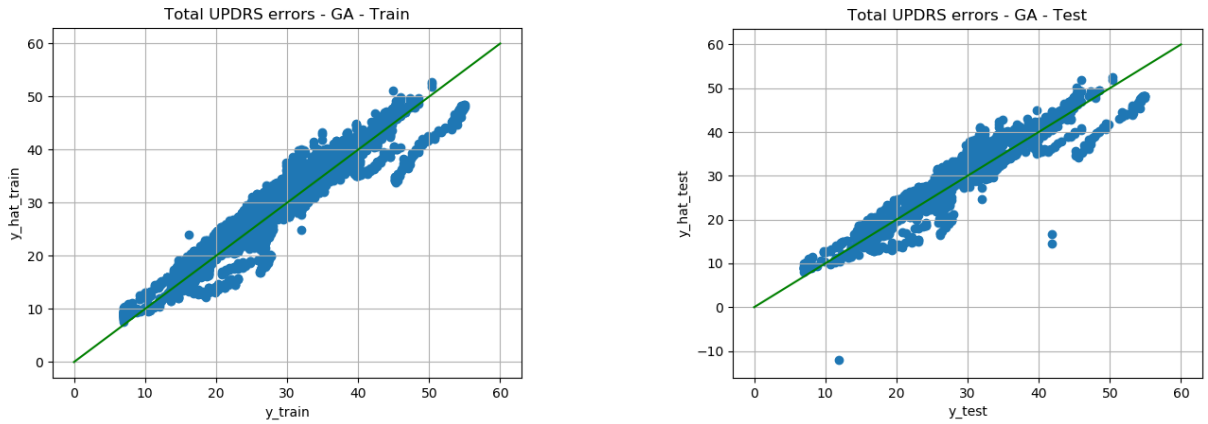

Figure 10 - Gradient Algorithm: On the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

In figure 11 is represented the distribution of the error among the patients using the Gradient Algorithm.

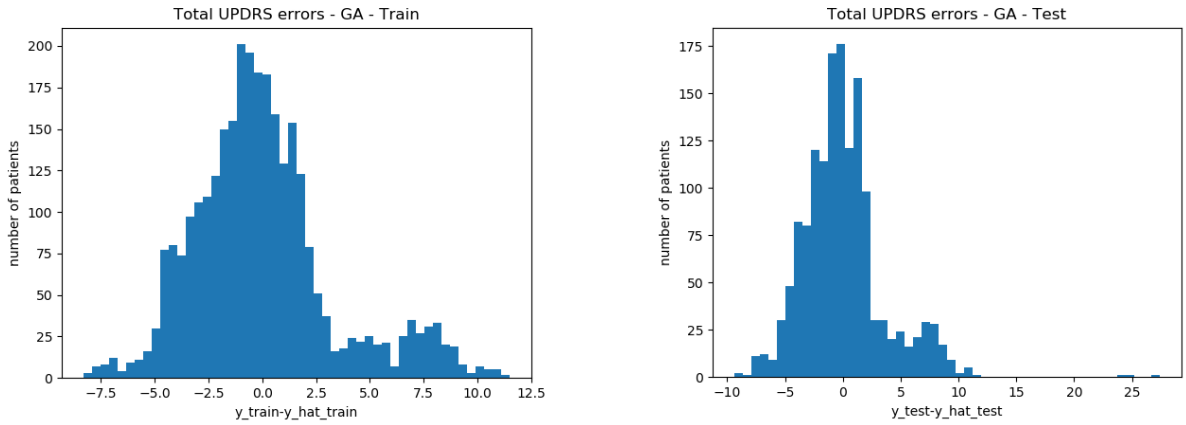

Figure 11 - Gradient Algorithm: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

The Steepest Descent Algorithm (SDA)

The Steepest Descent Algorithm is quite similar to the Gradient Algorithm. This method uses the same iterations described in the equation 4.1 but with optimal values of the learning coefficient γ.

The function to be minimized is approximated through the Tailor series stopped at the second term for each iteration. Consequently, a function of γ is obtained:

$$e(\boldsymbol{w_{k+1}}) \simeq e(\boldsymbol{w_k}) - \gamma \nabla e(\boldsymbol{w_k})^T \nabla e(\boldsymbol{w_k}) + \frac{1}{2}\gamma^2 \nabla e(\boldsymbol{w_k})^T \boldsymbol{H}(\boldsymbol{w}_k) \nabla e(\boldsymbol{w_k}) = g(\gamma)$$

(5.1)

In the equation 5.1, $\boldsymbol{H}(\boldsymbol{w}_k)$ represents the Hessian matrix evaluated at $\boldsymbol{w_k}$ and $-\gamma \nabla e(\boldsymbol{w_k})$ represents the vector $\boldsymbol{w_{k+1}} - \boldsymbol{w_k}$, as shown in the Gradient Algorithm (equation 4.1).

The optimum γ is the one which minimizes $e(\boldsymbol{w_{i+1}})$ that is not a function of **w** anymore, but it depends on γ. In order to find the minimum, it is necessary to calculate the derivative of the function 5.1 with respect to γ and to set it equal to zero.

After all the steps are performed, the optimum value of γ results:

$$\gamma_i = \frac{||\nabla e(\boldsymbol{w_i})||^2}{\nabla e(\boldsymbol{w_i})^T \boldsymbol{H}(\boldsymbol{w}_i) \nabla e(\boldsymbol{w_i})} \tag{5.2}$$

The Steepest Descent Algorithm can be stopped earlier in order to avoid the overfitting, as in the case of the Gradient Algorithm. On the one hand, this method generally requires less iterations but, on the other hand, the computation of the Hessian matrix and hence the evaluation of the learning coefficient is quite expensive from a computational point of view.

The MSE of the validation set does not increase in this case as well, as figure 12 shows. It is necessary to set another stop condition. Moreover, figure 12 reveals that the optimum value of **w** is reached in a smaller number of iterations compared to the Gradient Algorithm.

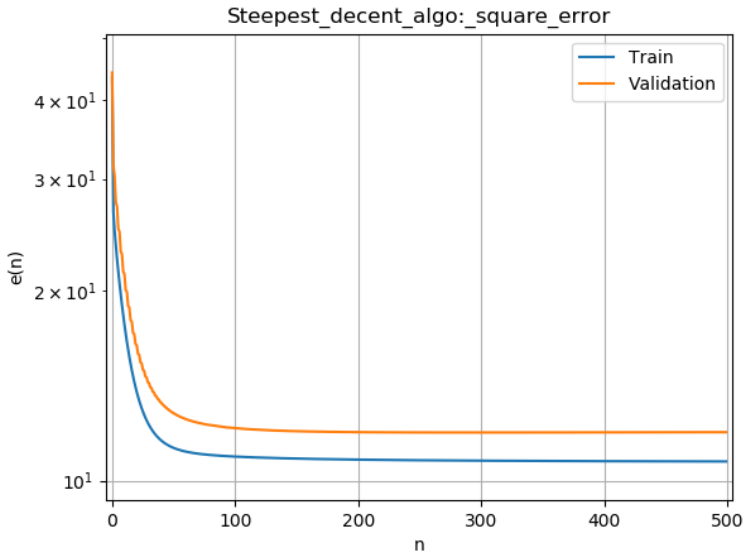

Figure 12 - Steepest Descent: the iterative mean square errors evaluated for the training and the validation set

According to figure 12, a reasonable number of iterations might be two-hundreds in this case. At this point, **w** is evaluated (figure 13).

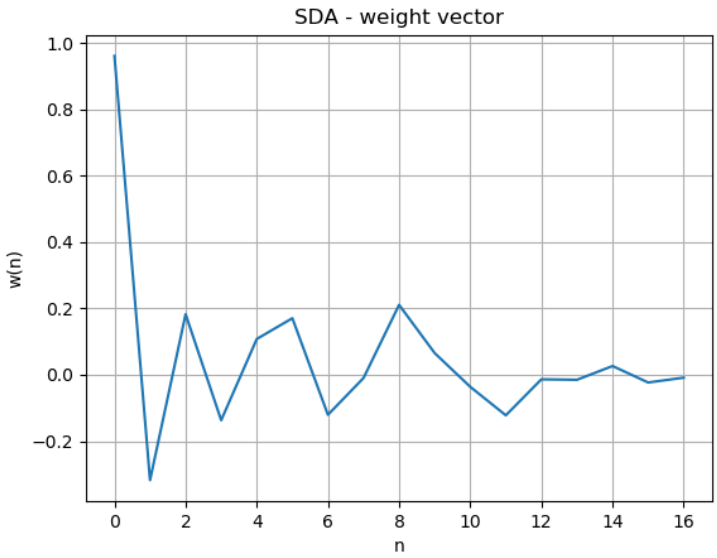

*Figure 13 - **w** obtained in the Steepest Descent*

Through the optimum **w** and the matrix of regressors **X**, the regressands can be predicted. The correlation evaluated through the Steepest Descent between the predicted and the real values of UPDRS is shown in figure 14.

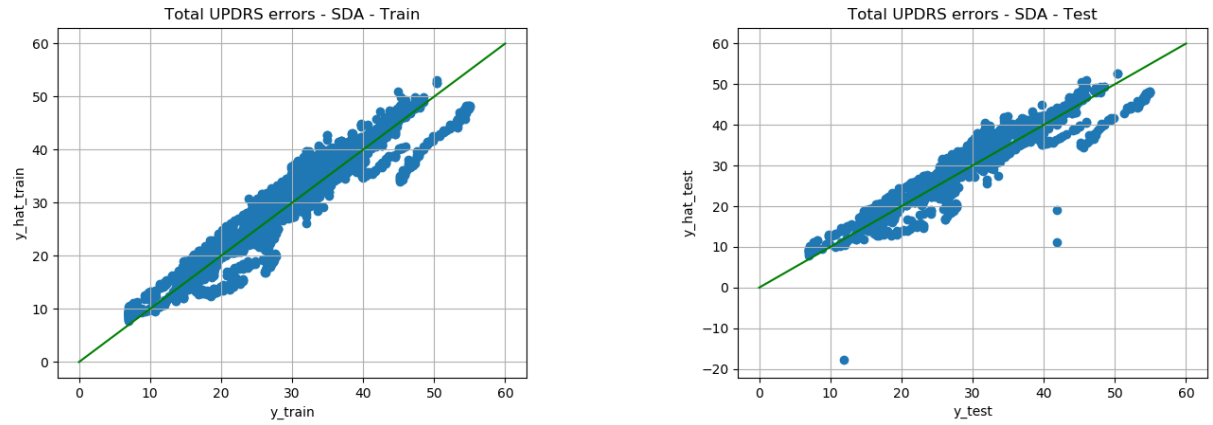

Figure 14 - Steepest Descent Algorithm: on the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

In figure 15 is represented the distribution of the error among the patients using the Steepest Descent Algorithm.

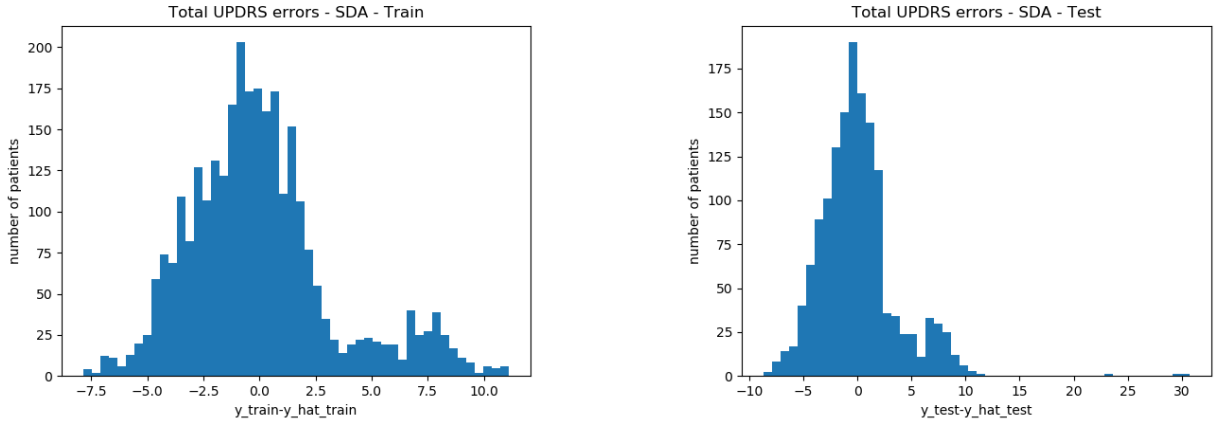

Figure 15 - Steepest Descent Algorithm: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

The Stochastic Gradient Algorithm (SGA)

The function to be minimized seen so far (1.3) can be written also as a sum of smaller functions:

$$e(\boldsymbol{w}) = \sum_{n=0}^{N} [[\boldsymbol{x}(n)]^T \boldsymbol{w} - y(n)]^2 = \sum_{n=1}^{N} e_n(\boldsymbol{w})$$

(6.1)

In the equation 6.1, **x**(n) is a row vector that indicates all the features of the n^{th} measurement and y(n) is the corresponding regressand feature. Consequently, the gradient of the error function is:

$$\nabla e(\boldsymbol{w}) = \sum_{n=1}^{N} [[\boldsymbol{x}(n)]^T \boldsymbol{w} - y(n)]\, \boldsymbol{x}(n) = \sum_{n=1}^{N} \nabla e_n(\boldsymbol{w})$$

(6.2)

This algorithm is based on the same concept of the Gradient Algorithm, where the evaluation of the optimum weight vector is made through many steps moving in the opposite direction of the gradient. The gradient considered is every time just one term of the summation, and it changes for each iteration.

The initial vector is randomly chosen, and it is updated through the equation 6.3.

$$\boldsymbol{w}_1 = \boldsymbol{w_0} - \gamma \nabla e_0(\boldsymbol{w_0})$$

(6.3)

Then, the updating continues until a stop condition is verified. More in general, **w** is evaluated as represented in the relation 6.4:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w_k} - \gamma \nabla e_k(\boldsymbol{w_k})$$

(6.4)

The learning coefficient γ is arbitrarily chosen, therefore, all the issues concerning a wrong estimation of γ are still present. Anyway, the evaluation of the gradient $\nabla e_k(\boldsymbol{w_k})$ involves just a scalar product of two vectors without any matrix inversion nor multiplication between matrix and vector; this leads to easier calculations.

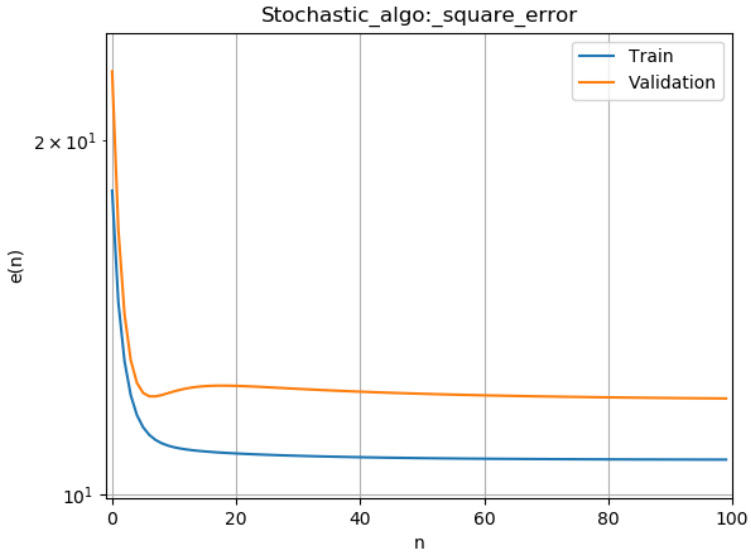

Figure 16 - Stochastic Gradient: the trend of the mean square errors evaluated in each iteration for the training and the validation set.

In the evaluation of the MSE concerning the SGA, during the first iterations overfitting seems to occur (Figure 16), but after an initial rising the MSE gets back to decrease. Therefore, a reasonable stop condition concerns the setting of a number of iterations. In this specific situation the Stochastic Gradient Algorithm can be stopped at one-hundred iterations.

The found vector **w** through the Stochastic Gradient Algorithm is shown in figure 17.

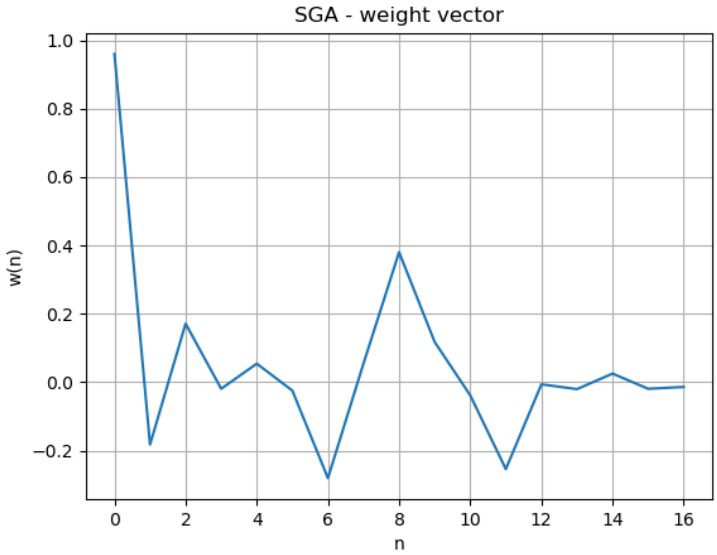

*Figure 17 - **w** obtained in the Stochastic Gradient Algorithm*

Through the optimum **w** and the matrix of regressors **X**, the regressands can be predicted. The correlation evaluated through the Stochastic Gradient Algorithm between the predicted and the real values of UPDRS is shown in figure 18.

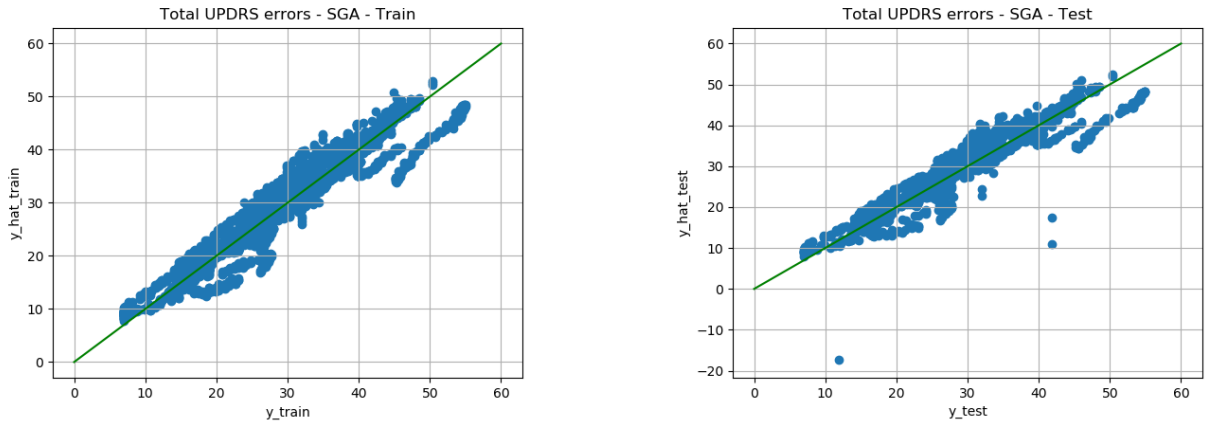

Figure 18 - Stochastic Gradient Algorithm: on the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

In figure 19 is represented the distribution of the error among the patients using the Stochastic Gradient Algorithm.

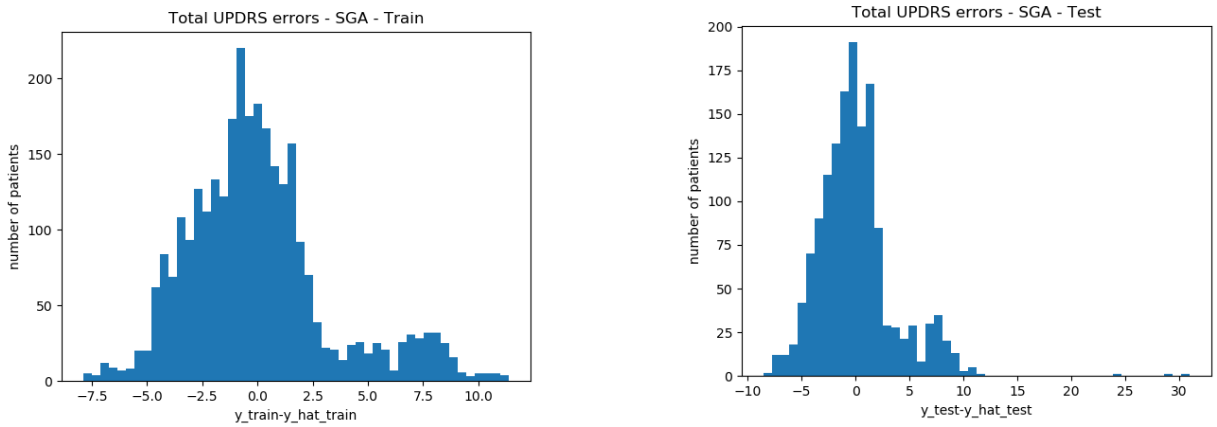

Figure 19 - Stochastic Gradient Algorithm: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

The Conjugate Gradient Algorithm (CGA)

The Conjugate Gradient Algorithm is an efficient method that allows to get the best weight vector in less than F steps, where F is the number of columns of the matrix **X**.

In order to explain this method, the equation 2.2 is rewritten as:

$$\nabla e(\boldsymbol{w}) = 2\boldsymbol{Q}\boldsymbol{w} - 2\boldsymbol{b} = \mathbf{0} \tag{7.1}$$

In 7.1 $\boldsymbol{Q} = \boldsymbol{X}^T\boldsymbol{X}$ and $\boldsymbol{b} = \boldsymbol{X}^T\boldsymbol{y}$.

The solution is based on the conjugate vectors of the matrix **Q,** since that the optimum **w** can be considered as a linear combination of **Q**-orthogonal vectors.

$$\boldsymbol{w} = \alpha_0\boldsymbol{d}_0 + \alpha_1\boldsymbol{d}_1 + \cdots + \alpha_{F-1}\boldsymbol{d}_{F-1} \tag{7.2}$$

In the relation 7.2, $\boldsymbol{d_k}$ is one of the conjugate vectors of matrix **Q** and α_k are the coefficients to be estimated. Both $\boldsymbol{d_k}$ and α_k are unknown, but they can be evaluated through the iterations given by the Conjugate Gradient Algorithm.

Each iteration concerns a movement along the $\boldsymbol{d_k}$ direction, the size of the step is given by α_k, so the vector **w** is updated through 7.3:

$$\boldsymbol{w}_{k+1} = \alpha_k\boldsymbol{d}_k + \boldsymbol{w}_k \tag{7.3}$$

The first iteration of the CGA concerns $\boldsymbol{w}_0 = 0$ and implies that the first direction of movement is equal to the opposite direction of the gradient evaluated in $\boldsymbol{w}_0$, $-\nabla e(\boldsymbol{w}_0)$, also equal to **b**. The total number of steps is F at most since the number of conjugate vectors of a square matrix is equal to the dimension of the matrix itself.

The values of α_k and $\boldsymbol{d}_k$ are given by relations that, for practical reasons, have not been incorporated in the report. These relations, as the complete algorithm, are available on many other sources.

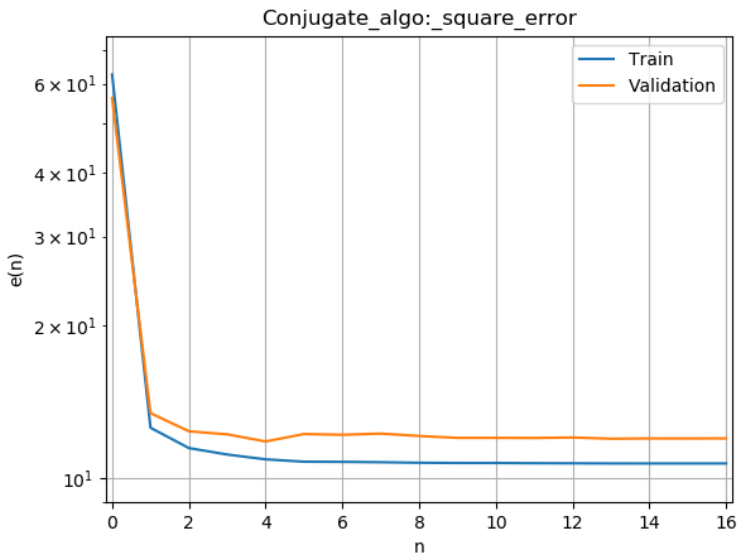

Figure 20 - Conjugate Gradient: the trend of the mean square errors evaluated in each iteration for the training and the validation set.

As previously explained, the iterations of the Conjugate Gradient Algorithm are F at most. In the given dataset the regressors are seventeen for each patient and, as shown in figure 20, the optimum weight vector is found in less than seventeen steps. Figure 21 represents the values of the gotten **w**.

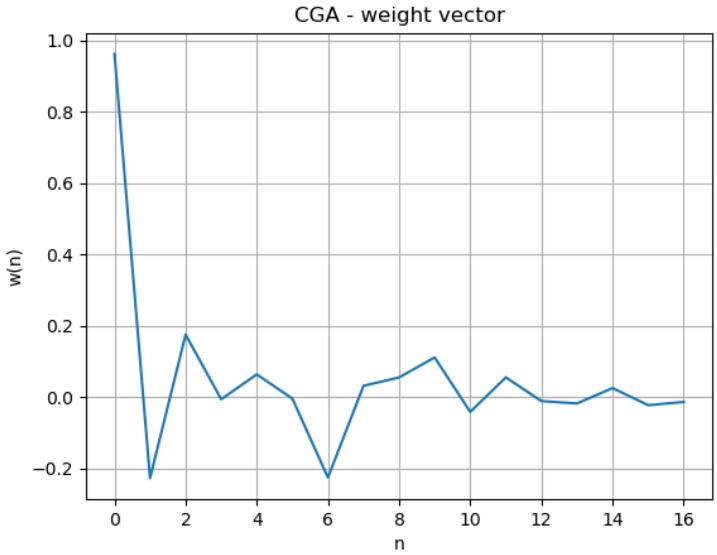

*Figure 21 - **w** obtained in the Conjugate Gradient Algorithm*

The correlation evaluated through the Conjugate Gradient Algorithm between the predicted and the real values of UPDRS is shown in figure 22.

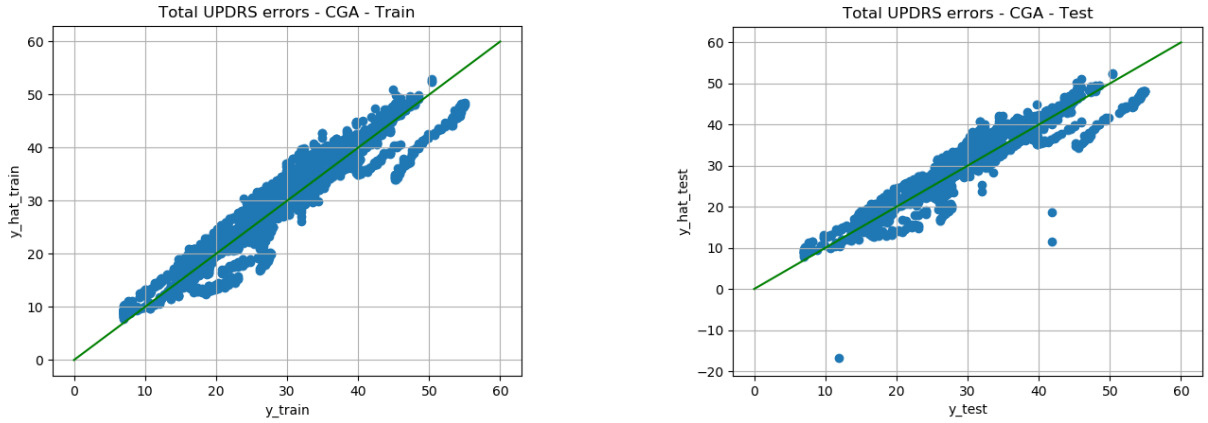

Figure 22 - Conjugate Gradient Algorithm: on the left, the real versus the predicted values (y_train vs y_hat_train) of UPDRS in the training set. On the right, the real versus the predicted values (y_test vs y_hat_test) of UPDRS in the test set

In figure 23 is represented the distribution of the error among the patients using the Conjugate Gradient Algorithm.

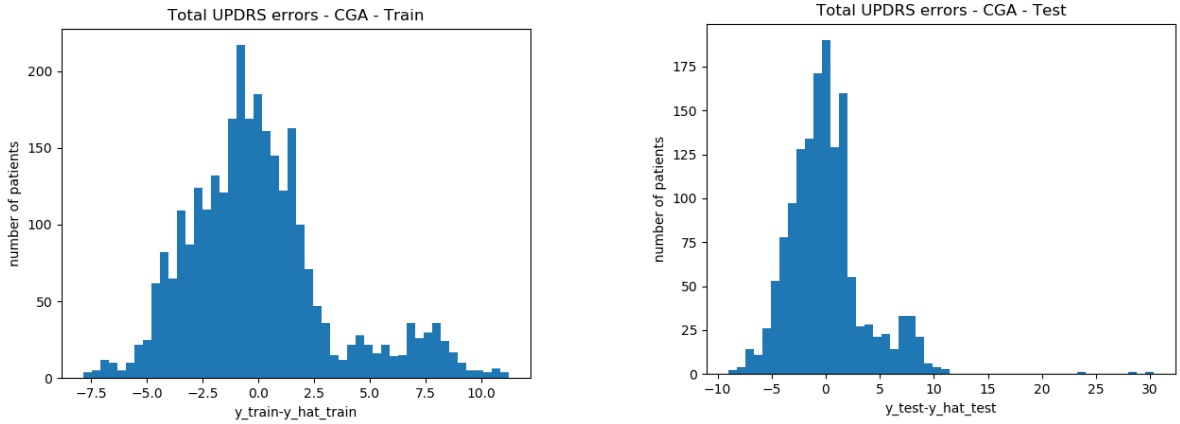

Figure 23 - Conjugate Gradient Algorithm: on the left, histogram that shows the distribution of the error among the patients of the training set. On the right, histogram that shows the distribution of the error among the patients of the test set.

Conclusions

The results obtained from the specific dataset have proved to be very similar, regardless of the method applied (Figure 24). Since overfitting never occurs, the iterative algorithms can be run for a large number of iterations and the gotten results are very close to the LLS and RR pseudoinverse. The weight vector obtained through the LLS is quite different from the others, as shown in figure 24. The problem in the inversion of the matrix $\boldsymbol{X}^T\boldsymbol{X}$ could be a reasonable explanation.

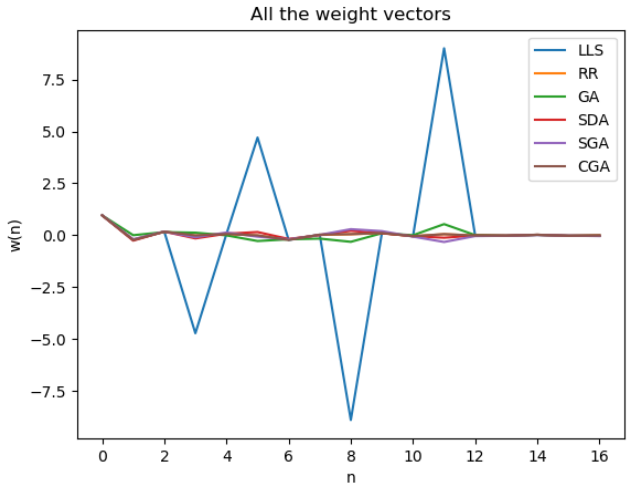

Figure 24 - Comparison among the weight vectors obtained through different methods

The mean square errors evaluated for the training set are smaller than the other ones evaluated for the validation and the test set, but the gap between these quantities is quite small, as resumed in the table 8.1.

	LLS MSE	RR MSE	GA MSE	SDA MSE	SGA MSE	CGA MSE
Training Set	10.698	10.701	10.820	10.721	10.714	10.701
Validation Set	11.996	11.992	12.340	11.927	12.076	11.990
Test Set	12.798	12.706	12.713	12.821	12.885	12.799

Table 8.1: *The MSE evaluated in the six described methods for the training, the validation and the test dataset*

In all the figures representing the real versus the predicted values of UPDRS in the test set, the predicted values are included in an interval of ten points at most from the best prediction. Since the range of total UPDRS goes from 0 to 199 points, an error of ten points is totally reasonable. Moreover, the real results of UPDRS may differ from one neurologist to another, since it is a subjective exam. The human error may be bigger than the error just evaluated through the linear regression.

Another consideration to be done is that sometimes an offset might be present between the regressors and the regressands. Plotting this situation in a one-dimensional problem would imply a graph showing a straight line that does not pass through the zero. The problem can be solved by adding a column of ones to the matrix of the regressors; thus, the dimension of the optimum vector **w** increases by one. It has been convenient to normalize the given data before starting the regression to avoid this situation. In fact, normalizing the data means to get rid of the ones column since they turn into zero, without affecting the result of **w**. However, the results shown have been un-normalized in order to simplify the understanding of the output data.