POLITECNICO DI TORINO

Master of Science Program
ICT for Smart Societies

ICT for Health

Final report on the ICT for Health laboratories

Fassio Edoardo 255268

1 Laboratory 1 - Regression on Parkinson's data

1.1 Introduction

In this laboratory we are going to work in a dataset, in which are collected features of patients affected by Parkinson disease. These patients have problems in controlling their movements, in fact they suffer tremor, muscle stiffness, and other symptoms.

The typical treatment prescribed to patients is Levodopa. Levodopa is a drug that allows the dopamine to be transferred in the Substantia Nigra. Substantia Nigra is a part of the brain where are located neurons whose aim is to use dopamine in order to establish a synapse with other neurons.

Parkinson disease is caused by these neurons degeneration. The degeneration causes the lack of dopamine needed to establish the synapse, resulting in slowness of movement and rigidity. By taking Levodopa, it is possible to compensate the lack of dopamine.

However, Levodopa is taken orally and passes through the stomach. As the illness progresses, the muscles of the stomach slow down, therefore the drug stays there for a long time before reaching Substantia Nigra, decreasing the usefulness of the treatment.

Patients need to increase the doses of Levodopa as the illness progresses. In order to optimize the treatment, it is useful to monitor the patients through some parameters. In the laboratory, we try to measure the evolution of the illness. We would like to have a technique to predict UPDRS (the score that the doctors give to the patient) automatically using a simple mechanism, like using voice samples, recorded from the patient.

1.2 The dataset

The dataset used comes from the UC Irvine Machine Learning Repository and it was created by Athanasios Tsanas and Max Little of University of Oxford.

Each patient, which is one row of the dataset, is described by 22 different features, among which it can be found UPDRS. The intersting features of the dataset are the one explained from column 5 to 22. The uninteresting features are: the **number of patient**, because there is no correlation of this feature with the illness gravity. Then, we did not consider the **time**, because we want to predict the patient condition as it comes to the doctor to be visited for the first time. Furthermore, the feature **age** could be useful, but we did not used because it is an integer value. For the same reason, we have also also removed **sex** for the same reason.

The data are collected serially for each patient. For the same medical examination day, it may be possible to have more rows for the same patient, with same UPDRS value but different values of Shimmer, Jitter, etc. The reason is that different attempts have been taken in measuring the values.

The rows of the dataset have been shuffled, so that no two iterations over the entire sequence of training iterations will be performed on the exact same patient's data. Shuffling data serves the purpose of reducing variance and making sure that models remain general and overfit less.

$$z\_train\_norm = \frac{X\_train - \mu\_train}{\sigma\_train}:$$
$$z\_val\_norm = \frac{X\_val - \mu\_train}{\sigma\_train}:$$
$$z\_test\_norm = \frac{X\_test - \mu\_train}{\sigma\_train}:$$

Figure 1: Datasets standardization formula

The starting dataset has been divided into three smaller dataset:

- the **50%** of the total number of patients belongs to the ***training dataset***
- the **25%** of the total number of patients belongs to the ***validation dataset***
- the **25%** of the total number of patients belongs to the ***test dataset***

We then apply the standardization of each dataset's data with respect of the training set data. The mean of each training set's column has to be subtracted to the three dataset values. Moreover, the columns data of the three dataset has to be divided by the standard deviation of each column of the training set. The normalized data resulted are in *Figure 1*, where *z_ train_ norm* is the normalized training set, *z_ val_ norm* is the normalized validation set, and *z_ test_ norm* is the normalized test set, μ_train is the mean row vector of the training set, σ_train is the standard deviation row vector of the training set.

In the standardized datasets, we have mean value $\mu = 0$ and standard deviation $\sigma = 1$ on the training set and a similar result for the validation and test set. Standardizing the features so that they are centered around 0 with a standard deviation of 1 is important if we are comparing measurements that have different units. Furthermore, in iterative learning algorithms, having standardized data, helps certain weights updating faster than not-standardized.

If we would not use the standardization of the data, we should add a column of 1 to the dataset. The hypotesis can be proved in this way: we have to apply a linear regression of feature Total_UPDRS. The regressors are not standardized, so they will have a mean and a standard deviation. In order to find the line that predict the regressand based on the 16 regressors, we will use 16 weights for the features and one weight for the offset, since the mean value of the columns is not zero. However, this is not enough, because we have to consider the weight due to the error on the prediction of the algorithm. The error represents the distance of the real data point from the regression line. In the end, we need 18 weights to find the regression line. Since we have 16 feature, we need to add a new feature, which is a column of 1, in order to consider the error in the model.

If we have applied standardization on the dataset, the mean value of the columns should be zero and the standard deviation equal to 1. If the mean is zero, the line regression's weight corresponding to the offset is equal to zero. Since we have 17 weights, the 16 features are sufficient to regress the variable, so it is not needed to add the column of 1.

1.3 Regressing Total UPDRS

It has been performed regression to predict the Total_UPDRS of the patients using six algorithms:

- Linear Least Square
- Gradient Algorithm
- Steepest Descent Algorithm
- Conjugate Algorithm
- Stochastic Gradient
- Ridge Regression

1.3.1 Linear Least Square estimation

Introduction In Linear Least Square a vector of measurements Y is given. This vector can be written as:
$Y = X^T * w + \nu(n)$
where X is a matrix shaped in this way: a number of rows equal to the number of patients and a number of columns equal to the number of features; w is a column vector of weights, which is unknown; $\nu(n)$ is a column vector of errors, because it is assumed that the measurements are not precise. We want to find an analytical solution to find out w.

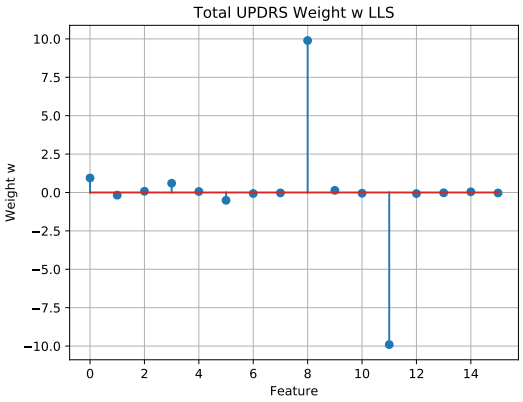
The typical way it is chosen to find out w is to minimize the square error. The square error is function of w:
$f(w) = \|y - X * w\|^2$
The gradient of $f(w)$ is evaluated and then we set it equal to zero to find the minimum of the function. It is possible to show that certainly we find a minimum, because $f(w)$ is a quadratic function, so the function will be positive or zero. By solving the equation, we find out w:

$$\nabla f(w) = -2 * X^T * y + 2 * X^T * X * w = 0$$
$$w = (X^T * X)^{-1} * X^T * y$$

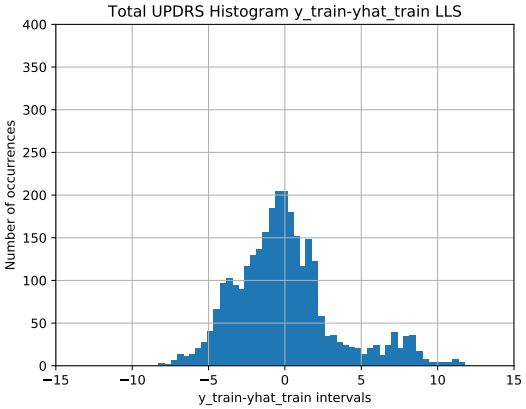
Once found the optimum value of vector w, the unknowns, then this estimation can be substituted inside the formula of $f(w)$ and the minimum is evaluated. It is called linear because we assume that y linearly depends on w, through a matrix X. It is called least square because the square norm of the error is is taken. The drawback of the close form is that we have to compute the inverse of a matrix: this might be computationally complex using a huge number of data, but it might not exist, because the eigenvalues might be close to zero.

Results In Figure 2 are shown the result got by applying Linear Least Estimation. From this solution, we can observe that Total_UPDRS linearly depends most on **feature 8**(**Shimmer:APQ3**) and **feature 11**(**Shimmer:DDA**), that give the most relevant contribution in estimating the regressand.

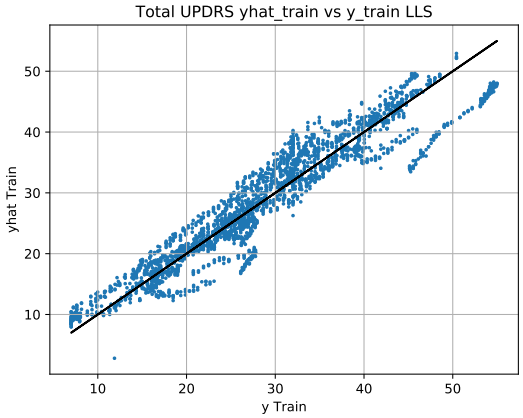
(a) *Weights w coefficients estimation*



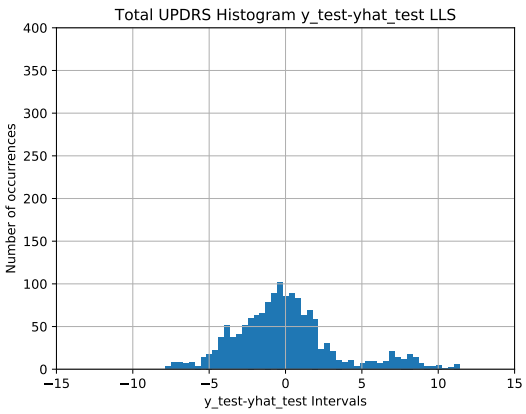
(b) *Histogram distribution of y_train-ŷ_train*



(c) *Estimation of ŷ_train vs y_train*



(d) *Histogram distribution of y_test-ŷ_test*



(e) *Estimation of ŷ_test vs y_test*

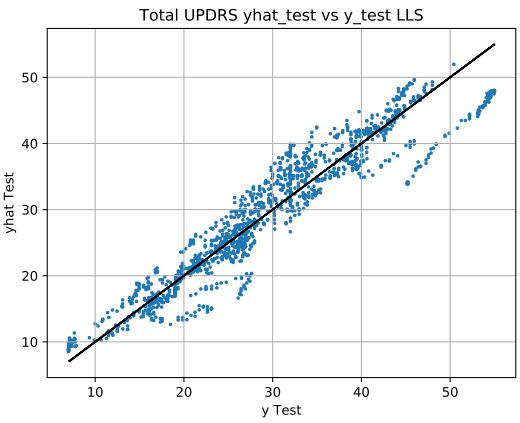


Figure 2: Results of Total_UPDRS regression using Linear Least Estimation

It has been observed that the solution gives **better** estimation on the **training set** than the test set, in fact the training set's mean square error is smaller than the one found for the test set. From the histograms, it seems that the **distribution** of the error is **similar** in both cases (even if the test set's number of occurrences are smaller because the test set is smaller than the training) and the range of the error values is identical in both cases.

However, from the two scatter plots, it seems that the intermediate values (in the range 30-40) of Total_UPDRS are better predicted than the higher and the smaller ones, where some points have been predicted very far away from the linear trend of estimation. Even if the performances are not perfect, it seems to predict correctly the real values of Total_UPDRS.

Mean Square Error evaluation		
Training Set	Validation Set	Test Set
11.1494	11.1859	11.5255

1.3.2 Gradient Algorithm

Introduction In Gradient Algorithm the function needed to be minimized is the same described in the previous section. However, this solution will use a numerical solution, not analytical. We want to evaluate the gradient of the function $f(w)$, now called **objective function**, written before:
$\nabla f(w) = -2 * X^T * y + 2 * X^T * X * w = 0$

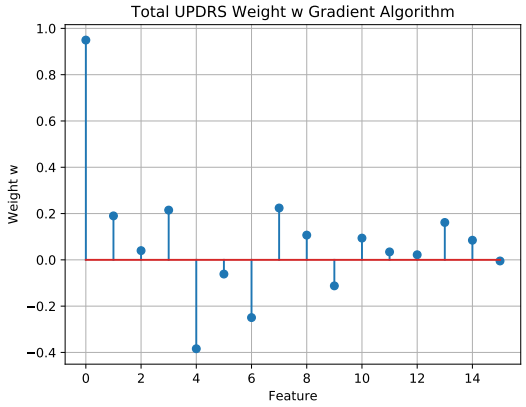
where X is the training data matrix. The algorithm starts with an initial guess $X0$. Then the gradient is evaluated at point $X0$. The result is multiplied by the coefficient γ, which is called the **learning coefficient**.

The γ is a small positive constant, which is chosen a priori. If γ is kept too big, the solution moves around the minimum, without converging to the solution, while if it is kept too small, the algorithm will converge to the solution after a lot of iterations. A reasonable value of γ has to be chosen. Finally, we sum the result with the previous point to get the next point. In brief:
$x_{i+1} = x_i - \gamma * \nabla f(x_i)$
where we use $+$ in case it is needed to find the maximum, otherwise a $-$ it is used to find the minimum. In this case, from the point x_i, we are moving against the direction of the gradient to find the minimum, so we use $-$.

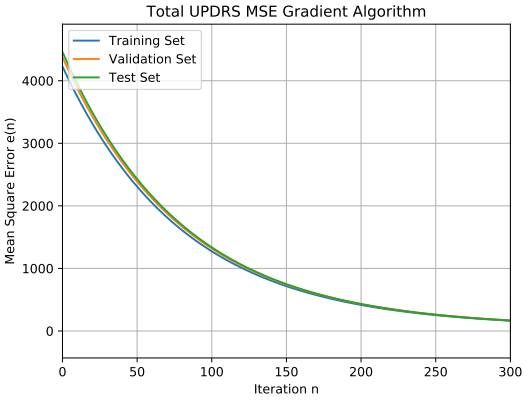
This procedure has to be iterated for a number of times that depends on singular function. There are three stopping conditions:

- The number of iterations (e.g 10^3)
- $|f(x_{i+1}) - f(x_i)| < \epsilon$
- $\dfrac{|f(x_{i+1}) - f(x_i)|}{\max(1, f(x_i)} < \epsilon$

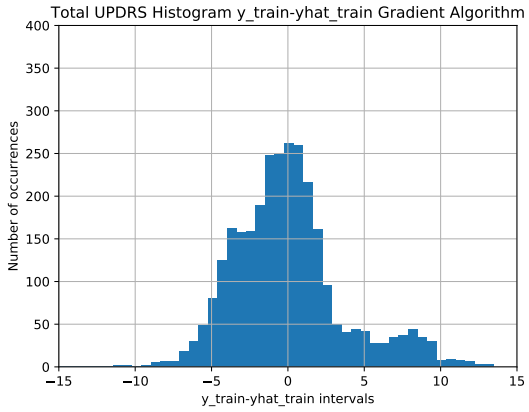
(a) *Weights w coefficients estimation*



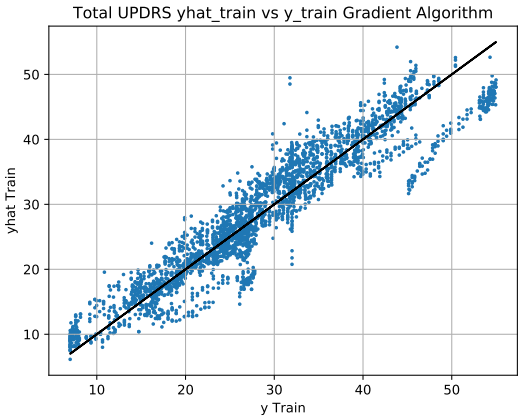
(b) *Mean Square Error estimation*



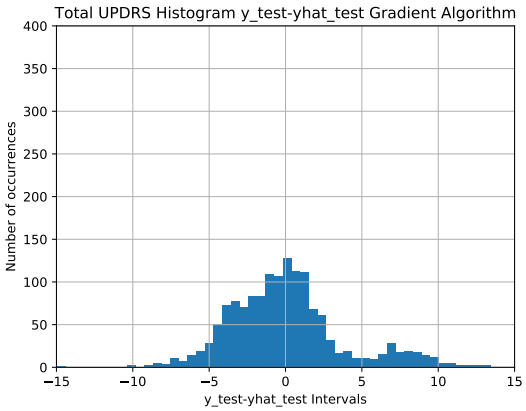
(c) *Histogram distribution of y_ train-ŷ_ train*



(d) *Estimation of ŷ_ train vs y_ train*



(e) *Histogram distribution of y_ test-ŷ_ test*



(f) *Estimation of ŷ_ test vs y_ test*

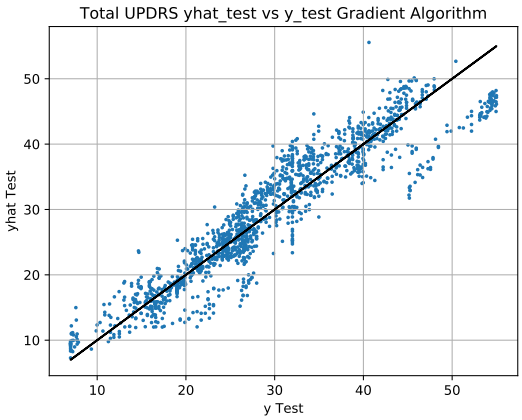


Figure 3: Results of Total_UPDRS regression using Gradient Algorithm

Result In Figure 3 are shown the results got by applying the Gradient Algorithm. The solution has been applied using a number of iterations $Nit = 10000$ and the learning coefficient $\gamma = 10^{-7}$. From this solution, it is possible to observe that Total_UPDRS depends mostly on feature 0 (**Motor_UPDRS**), on feature 4 (**Jitter: PPQ5**) and on feature 6 (**Shimmer**) and 7 (**Shimmer (dB)**). The error of the training and test dataset are distributed around 0. Most of the occurrences of the error happen on intervals of the error next to zero, so most of the prediction seems to be correct. From the error plot it is possible to observe that the error decreases with the number of iterations, either for the training set, for the validation and test set. If after a certain number of iterations, the error of the validation set follows the one of the training set, this means that there is not overfitting of data.

From Figure 2(d) and Figure 2(e), it can be observed that the general of the data follows the axis bisector, this conferms what we have said in the histogram.

Mean Square Error evaluation		
Training Set	Validation Set	Test Set
13.4021	14.2076	14.1289

1.3.3 Steepest Descent Algorithm

Introduction The main task in Steepest Descent algorithm is to find the optimum γ, which minimizes the function:
$x_{i+1} = x_i - \gamma * \nabla f(x_i)$

It is possible to approximate at point x_i the objective error function that had to be minimized through the Taylor series:
$f(x_{i+1}) \approx f(x_i) - \gamma_i * \nabla f(x_i)^T * \nabla f(x_i) + \frac{1}{2} * \gamma_i^2 * \nabla f(x_i)^T * H(x_i) * \nabla f(x_i)$

Let call this expression $h(\gamma_i)$, which is function of γ. The gradient of $h(\gamma_i)$ taken with respect to γ is set to zero and the optimum γ is evaluated in equation (1):

$$\gamma_i = \frac{||\nabla f(x_i)||^2}{f(x_i)^T * H(x_i) * \nabla f(x_i)} \tag{1}$$

In general, steepest descent algorithm requires less step to converge to the solution.

Result In Figure 4 are shown the results got by applying the Steepest Descent Algorithm. The weight calculated by the algorithm shows that the most relevant features used to predict Total_UPDRS are feature 0 (**Motor UPDRS**), feature 3(**Jitter:RAP**) and feature 5 (**Jitter:DDP**). From the histogram of training and test data is possible to see that the error is distributed like the Gradient Algorithm and the maximum of occurrences is not exactly on the zero. It is not as precise as the Gradient Algorithm.

It is difficult to evaluate the optimum γ because we need to evaluate the Hessian matrix, which requires the calculation of the transpose of the matrix X. However, the algorithm uses less iterations to find the solution once calculated the optimum value of γ.

Mean Square Error evaluation		
Training Set	Validation Set	Test Set
11.1499	11.1802	11.5114

1.3.4 Conjugate Algorithm

Introduction The Conjugate Algorithm uses the concept of conjugate vectors. The conjugate vectors are orthogonal with respect to a Q matrix. It means that the vectors d_i and d_k are Q-orthogonal if:
$d_i^T * Q * d_k = 0$.
The problem needed to be solved is $Q * w^* - b = 0$. The aim is to find w^*. The orthogonal vectors are used for the same Q matrix. The w^* can be written as a linear combination of the orthogonal vector:

$$w_* = \alpha_0 * d_0 + \alpha_1 * d_1 + ... + \alpha_{N-1} * d_{N-1} \tag{2}$$

where the d_k are the Q-orthogonal vectors. The α_k coefficient are found through the following equation:

$$\alpha_k = \frac{d_k^T * Q * w^*}{d_k^T * Q * d_k} \tag{3}$$

By starting from an original vector (a vector of zeros is suggested), moving along the Q-orthogonal vectors that gives a new direction each time, the algorithm keeps on finding a better solution. By starting from a solution $w_* = 0$, you evaluate the gradient of the function called g. The first step is taken in the opposite direction with respect of the gradient, because the minimum has to be evaluated. Then once arrived at point w_{k+1}, the direction of the movement is not that of the gradient, but:

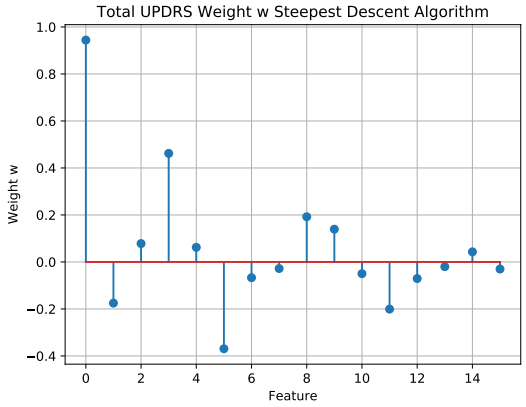
$$\beta_{k+1} = -g_{k+1} + \beta_k * d_{k-1} \tag{4}$$

The algorithm converges in N steps.

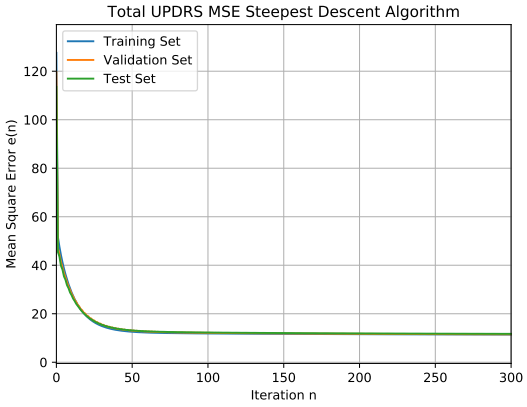
Result In Figure 5 are shown the results got by applying the Conjugate Algorithm. The weight calculated by the algorithm shows that the most relevant features used to predict Total_UPDRS are feature 0 (**Motor UPDRS**), feature 1(**Jitter %**) and feature 9 (**Shimmer:APQ5**). The distribution of the error shows the same dynamic seen in the previous algorithms. Most of the errors are distributed around the intervals next zero. The Mean Square Error of the validation and test set follows the training set trend, confirming the fact that there is no overfitting.

Mean Square Error evaluation		
Training Set	Validation Set	Test Set
11.1499	11.1802	11.5093

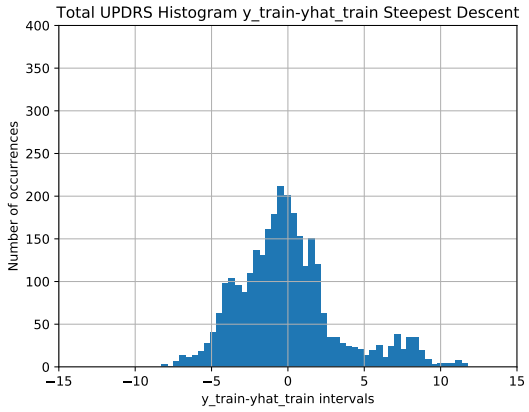
(a) *Weights w coefficients estimation*



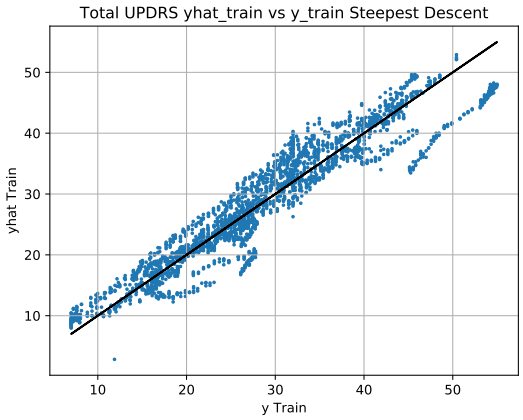
(b) *Mean Square Error estimation*



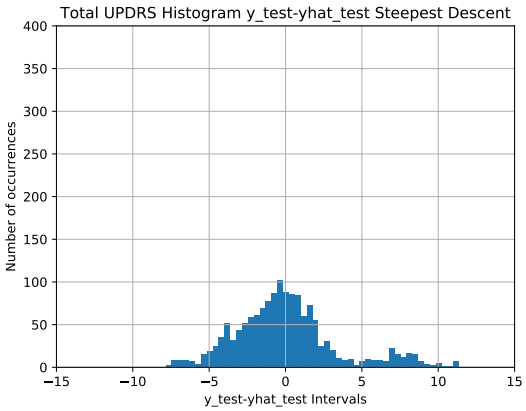
(c) *Histogram distribution of y_train-ŷ_train*



(d) *Estimation of ŷ_train vs y_train*



(e) *Histogram distribution of y_test-ŷ_test*



(f) *Estimation of ŷ_test vs y_test*

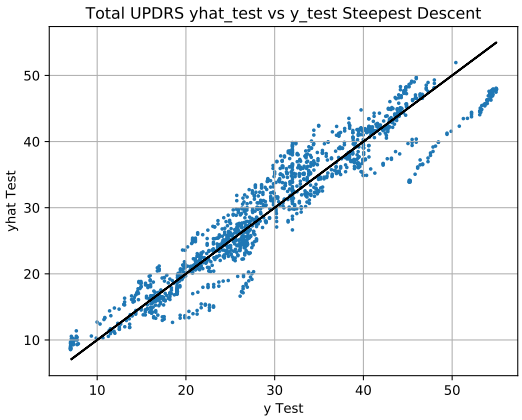
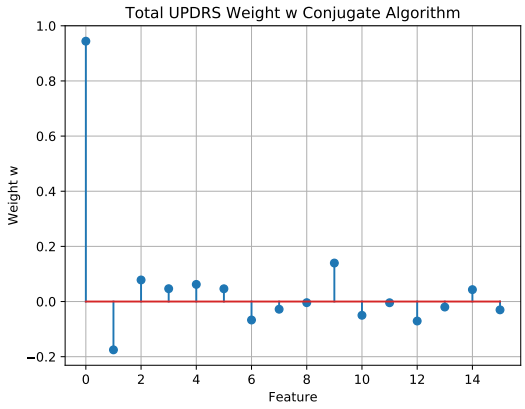
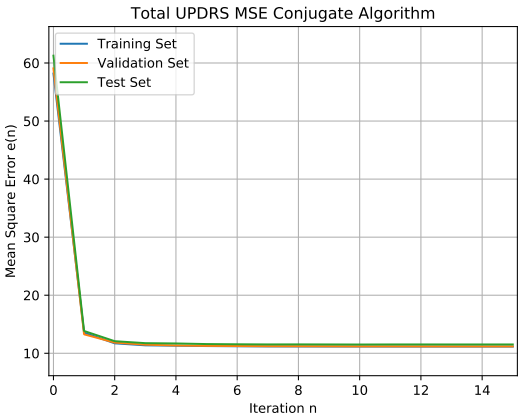


Figure 4: Results of Total_UPDRS regression using Steepest Descent

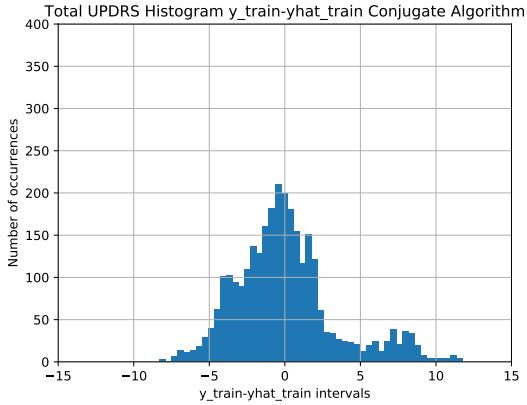
(a) *Weights w coefficients estimation*



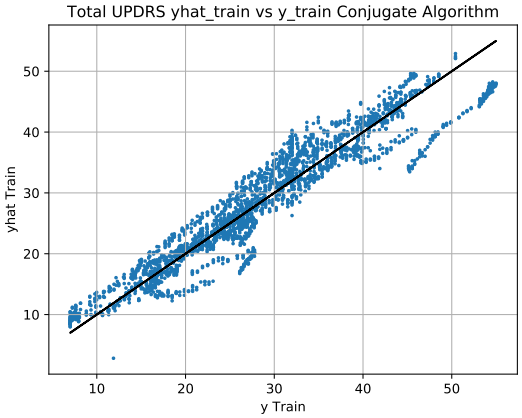
(b) *Mean Square Error estimation*



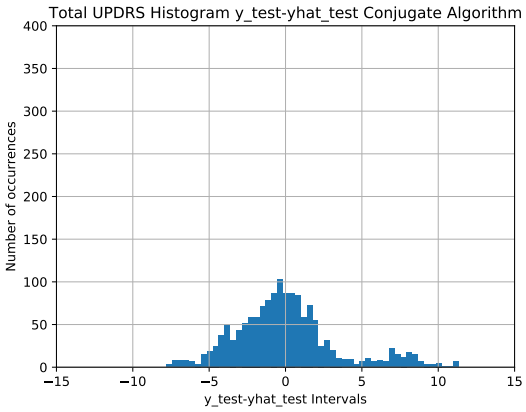
(c) *Histogram distribution of y_ train-ŷ_ train*



(d) *Estimation of ŷ_ train vs y_ train*



(e) *Histogram distribution of y_ test-ŷ_ test*



(f) *Estimation of ŷ_ test vs y_ test*

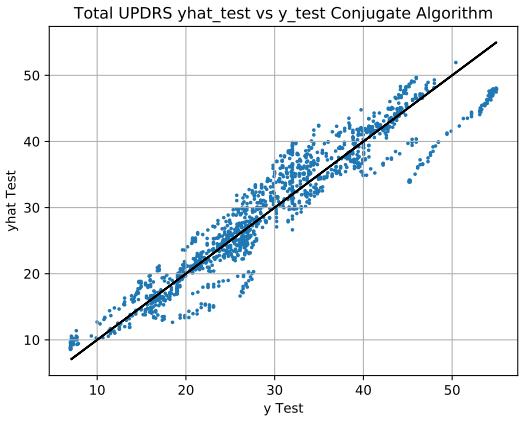


Figure 5: Results of Total_UPDRS regression using Conjugate Algorithm

1.3.5 Stochastic Gradient Algorithm

Introduction It is possible to apply the Stochastic Gradient algorithm whenever an objective function, which has to be minimized or maximized, can be written as a sum of smaller functions. It can be observed that the gradient of a sum, is a sum of smaller gradients

$$\nabla f(w) = \sum_{n=0}^{N} \nabla f_n(w) = 2 * \sum_{n=0}^{N} [(x(n))^T * w - y(n)] * x(n) \tag{5}$$

$$w_{i+1} = w_i - \gamma * \nabla f_i(w) \tag{6}$$

Result In Figure 6 are shown the results got by applying the Stochastic Gradient Algorithm. By observing the weight w, the Total_UPDRS depends mostly on feature 0 (**Motor UPDRS**), on feature 5 (**Shimmer:DDP**) and feature 8 (**Shimmer:APQ3**). Starting from the evaluation of the error, it is expected that the Mean Square Error falls down with the number of iterations. However, the Mean Square Error decreases slower than the other algorithms. Since the validation error is following the training set error, there is no overfitting.

The histogram shows that the error is distributed around the intervals next to zero. In both case most of the error occurrences are distributed next to zero, so most of the patient's Total_UPDRS prediction is similar to the real one. However, the error occurrences range is larger than the other algorithm.

Mean Square Error evaluation		
Training Set	Validation Set	Test Set
14.1932	14.2898	14.3141

1.3.6 Ridge Regression Algorithm

Introduction In Ridge Regression, we add a condition on the square norm of w. Instead of minimizing the classic error objective function, it is minimized the objective function minus the constant times the constraint. The constant is λ, called tuning coefficient, the constraint is $||w||^2$. The $\lambda * ||w||^2$ should not increase too much, otherwise numerical problems might occur.

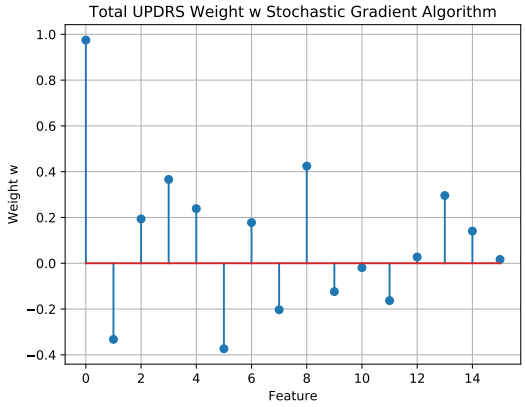
$$\min ||y - X * w||^2 + \lambda * ||w||^2 \tag{7}$$

$$\nabla f(w) = -2 * X^T * y + 2 * X^T * X * w + 2 * \lambda * w \tag{8}$$

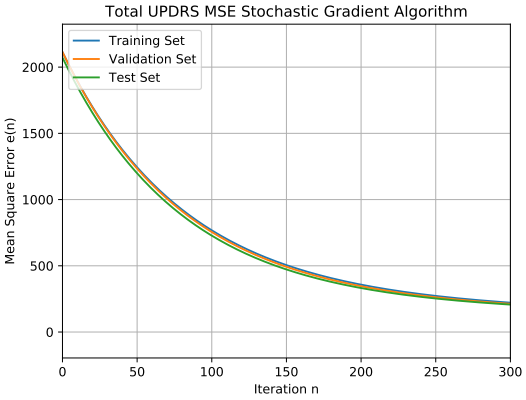
$$w = (X^T * X + \lambda * I)^{-1} * X^T * y \tag{9}$$

where I is the identity matrix, λ is the tuning coefficient. The presence of the term $\lambda * I$ makes the matrix invertible. The square error will be higher because the objective function to be minimized is different. λ is important because allows to shrink the weight coefficients. If $\lambda = 0$, we have the same solution of Linear Least Estimation. If we increase λ we are reducing the feature coefficients weight. It is important to choose the proper value of λ in order to avoid overfitting.

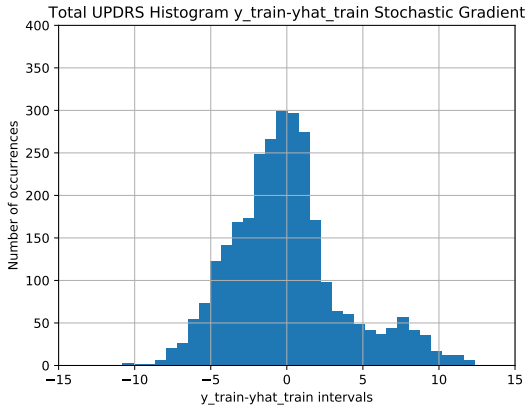
(a) *Weights w coefficients estimation*



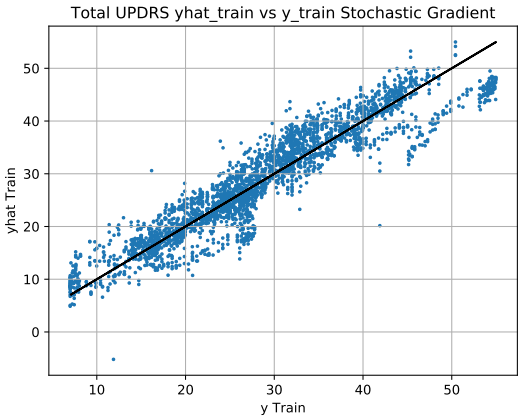
(b) *Mean Square Error estimation*



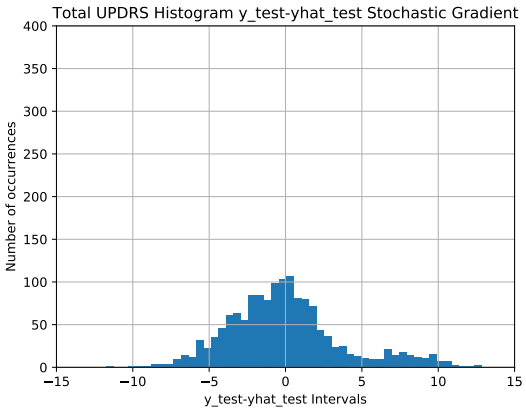
(c) *Histogram distribution of y_ train-ŷ_ train*



(d) *Estimation of ŷ_ train vs y_ train*



(e) *Histogram distribution of y_ test-ŷ_ test*



(f) *Estimation of ŷ_ test vs y_ test*

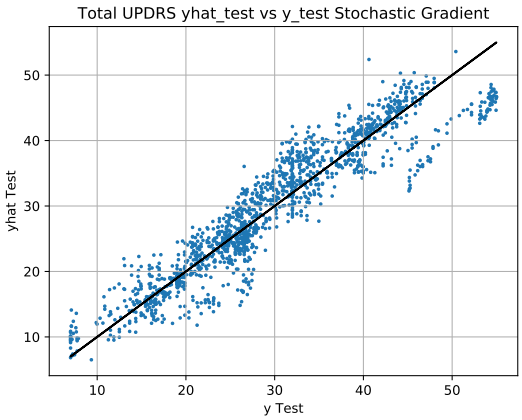


Figure 6: Results of Total_UPDRS regression using Stochastic Gradient

Result In Figure 7 are shown the results got by applying the Ridge Regression. By observing the weight w, the Total_UPDRS depends mostly on feature 0 (**Motor UPDRS**). In the graph of the error, the tuning coefficient λ has been iterated from 0 to 200 in order to find the minimum error given by the validation set. From the graph, it is observed that while the training set error keep increasing against λ, the validation set error first decrease with λ, then it starts increasing again due to the overfitting problem. It has been found a minimum in $\lambda = 1$. Setting that value of λ, we should avoid overfitting. The results are obtained by setting that value of λ.

The distribution of the error seems to be similar to the ones seen in the other algorithm results. Most of the values are predicted close to the real value of Total_UPDRS.

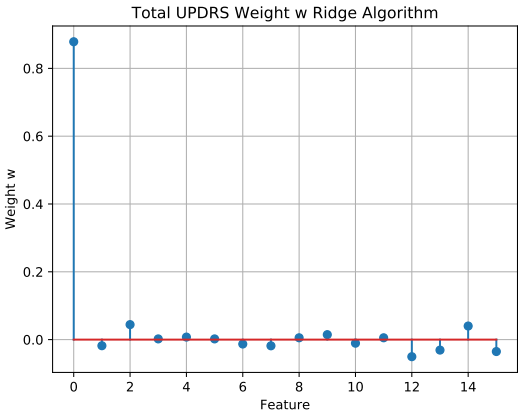
Mean Square Error evaluation		
Training Set	Validation Set	Test Set
11.1494	11.1811	11.5107

1.4 Conclusion

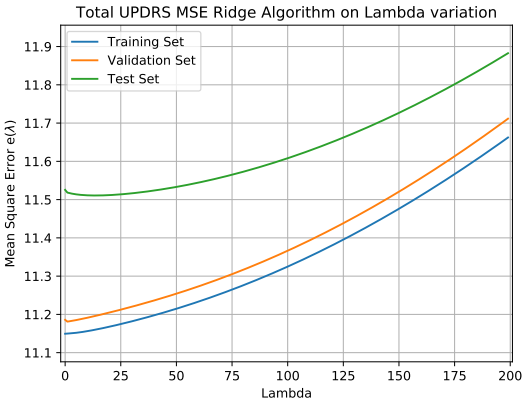
From the results obtained, the most precise algorithm for the training set is **Ridge Regression**, since it achieved the least Mean Square Error. However, the results on the test set has to be observed since they picture a more general model for the prediction. The **Conjugate Algorithm** offers the best results for the test set.

Generally, only two algorithms offers very poor perfomances: the **Gradient Algorithm** and the **Stochastic Gradient Algorithm**, since their predictions on average is pretty far from the real Total_UPDRS condition of the patients, compared to the other algorithms. Moreover, they offer also very low performances in terms of computational time, which is very high among the other algorithms. This is also due to the high number of iterations chosen to get the most accurate results.

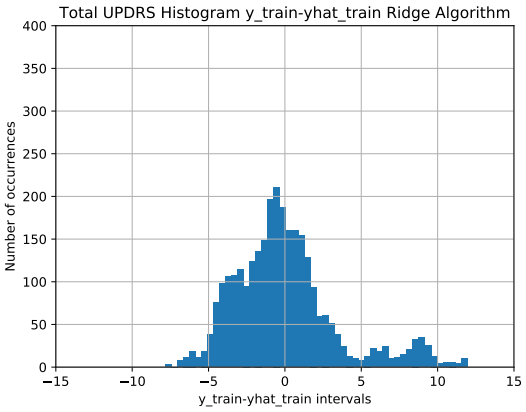
(a) *Weights w coefficients estimation*



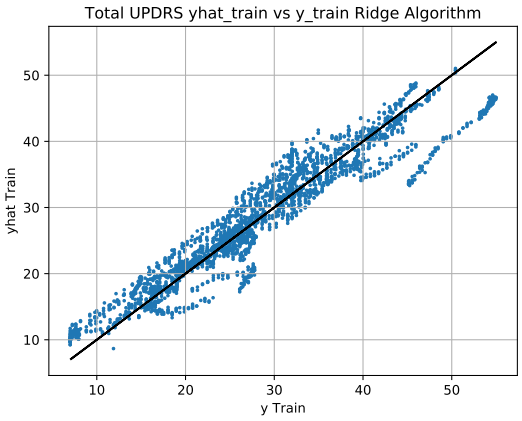
(b) *Mean Square Error estimation*



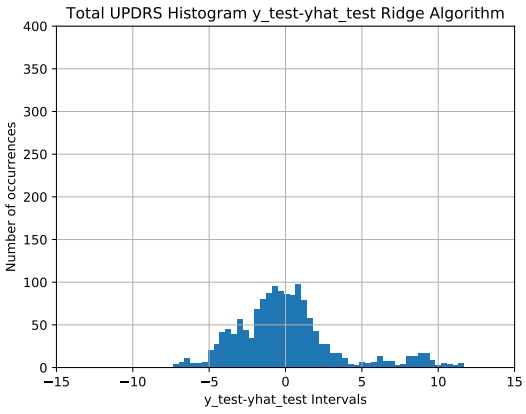
(c) *Histogram distribution of y_ train-ŷ_ train*



(d) *Estimation of ŷ_ train vs y_ train*



(e) *Histogram distribution of y_ test-ŷ_ test*



(f) *Estimation of ŷ_ test vs y_ test*

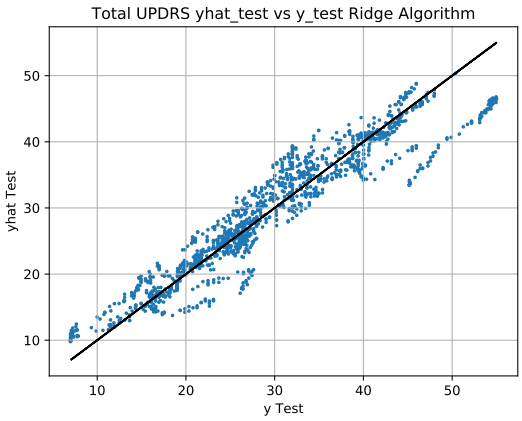


Figure 7: Results of Total_UPDRS regression using Ridge Algorithm