POLITECNICO DI TORINO

ICT for Health

Report Laboratory 1

Professor:
Monica Visintin

4/5

Student:
Iman Ebrahimi Mehr S250190

A.Y.2019-2020

Contents

1 Introduction

1.1 Parkinson's disease (PD)

Parkinson's disease (PD) is the most common age-related motoric neurodegenerative disease initially described in the 1800's by James Parkinson as the 'Shaking Palsy'; Motor symptoms in PD are tightly linked to the degeneration of substantia nigra dopaminergic neurons and their projections into the striatum; also it's characterized by resting tremor, rigidity, akinesia and bradykinesia. The sick neurons project to other structures in the basal ganglia causing the loss of function of the latter that is involved in the coordination of the body movement.

Unified Parkinson's Disease Rating Scale (UPDRS) is a worldwide scale used to evaluate the progression of the disease. The UPDRS is composed of four sub-scales and each of them contain some items (42 in total), which assess impairment related to the PD. The subclass are :

- mentation, behavior and mood (composed of 4 items)
- activities of daily living (composed of 13 items)
- motor (composed of 14 items)
- complication of therapy (composed of 11 item)

More in particular, the speech symptoms related to Parkinson's Disease are: overall loudness level is reduced; rate of speech becomes too slow or too fast; difficulty initiating speech or inappropriate pauses; voice is tremulous and monotonous; hoarse/breathy vocal quality; articulatory effort is reduced or imprecise.

1.2 Goal of laboratory

The goal of laboratory is to find a linear correlation, whether exist or not, between the features of the patients and the grade of the UPDRS, using linear regression. The regression can be realized through different methods or algorithms where each of them starts from a set of observed values (dependent variable) $y(n)\varepsilon R$, with $n = 1, .., N$, also called regressand, and goes back to the independent variable $x(n)\varepsilon R^N$, also called regressor. In linear regression we assume that

$$y(n) = [x(n)]^T w + v(n) = Xw + v(n) \tag{1}$$

where $w = [w_1, .., w_F]$ is a set of weight to be found (they represent the correlation between the regressor and the regressand) and $v(n)$ is the error. Now by defining the **cost function** as $f(w) = ||y - Xw||^2$ which we want to minimize it because by minimizing this, we get $v(n) = 0$ and so $y = Xw$. This means that, knowing the values of w and X , we can predict the values of y. To summarize, we're going to find an optimum set of weights w^* which allow us to find a feature $y(n)$ by a given set of other features stored in a matrix X.

2 The data

2.1 The data set

The dataset which we're going to use in this laboratory was created by the University of Oxford from 42 people recruited in six-month trial of a telemonitoring device for remote symptom progressing monitoring. These data are stored into a table which is composed of 26 columns and 5875 rows. Each column represents a useful feature to describe the severity of the PD, for example there is stored patience number, age, gender, time interval from baseline recruitment date, motor UPDRS, total UPDRS and other 16 voice measurements. Each row corresponds to a single measurement of each feature on a given patient.

2.2 Preprocessing

First of all we want to standardize our data which help us to build features that have similar ranges to each other and make them comparable, to do this procedure we arround our data by Mean of data and divide them to Standard Deviation as follow:

$$z = \frac{x - \mu}{\delta} \tag{2}$$

In the next step we divided our data into 3 part as:

- **training dataset**: used to train algorithms and to evaluate if they are correct or not. The result of executing algorithms is the set of weights w^*;
- **validation dataset**: used in some algorithms to optimize it by changing some given parameters (like the number of iterations in iterative algorithm) and to adjust the w^*;
- **testing dataset**: used finally to predict the y(n) with the w* and calculate error of algorithms.

3 Methods and Algorithms

3.1 LLS pseudoinverse

Starting from the equation of the linear regression $y(n) = Xw + v(n)$, the Linear Least Square (LLS) algorithm try to find the w that minimizes the cost function $f(w) = ||y - Xw||^2$ that can be written as

$$f(w) = [y - Xw]^T[y - Xw] = y^T y - y^T Xw - w^T X^T y + w^T X^T Xw \tag{3}$$

Now, we evaluate the gradient (multi-variable generalization of the derivative) and set it equal to 0.

$$\nabla(w) = -2X^T y + 2X^T Xw = 0 \tag{4}$$

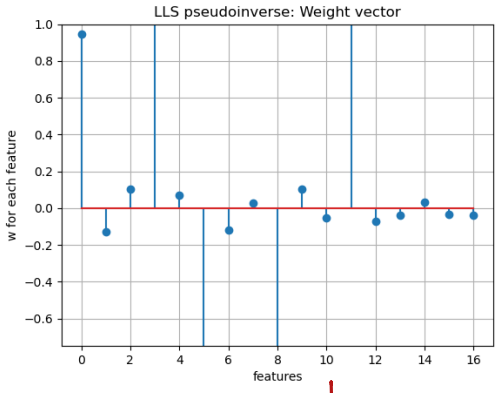
And finally by simplifying we get the optimum weight vector as:

$$w^* = [X^T X]^{-1} X^T y \tag{5}$$

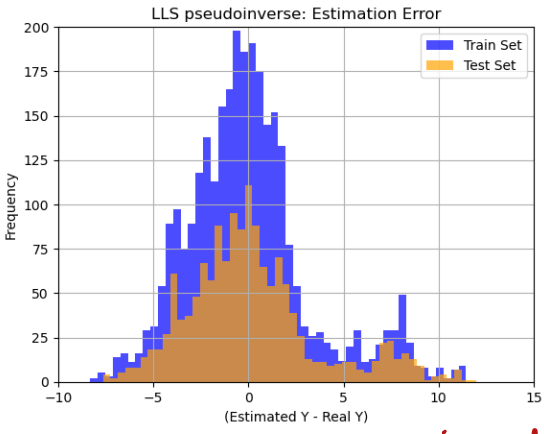
The term: $[X^T X]^{-1} X^T$ is the pseudoinverse of the rectangular matrix X (training data).

3.1.1 Result

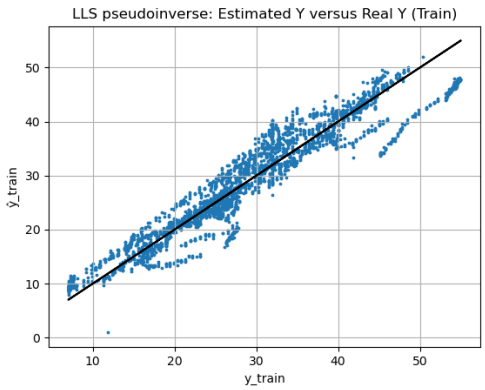
The optimum values of W^* are shown in figure 1a. To evaluate the algorithm we compare the real vector y , extracted from the training and testing dataset, with the predicted vector $\hat{y} = Xw^*$, calculated by executing the algorithm, as we can see in figures 1c and 1d.
The figure 1b represent the relation between the error, which is the difference $y - \hat{y}$, and number of times it occurs. The histogram has been ~~realized~~ computed by using training dataset(blue) as well as testing dataset(orange).



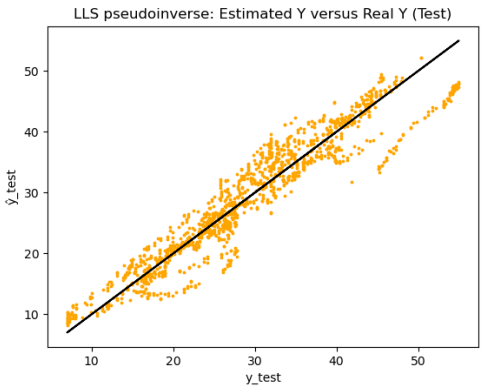
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train ing datasets*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$



(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

Figure 1: Linear Least Square

3.2 The gradient algorithm

It is a first-order optimization algorithm. This means it only takes into account the first derivative when performing the updates on the parameters. On each iteration, we update the parameters in the opposite direction of the gradient of the objective function $f(w)$. The size of the step we take on each iteration to reach the local minimum is determined by the learning rate γ. After having evaluated the gradient as:

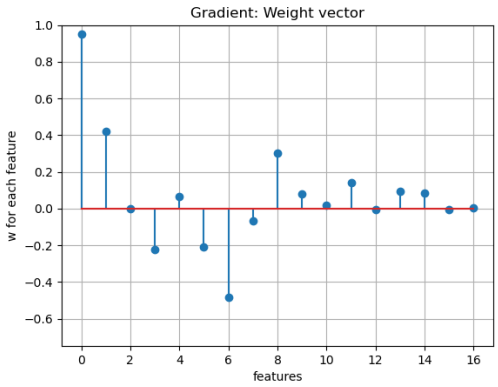
$$\nabla f(\mathbf{w}) = 2\mathbf{X}^\top(\mathbf{X}\mathbf{w} - \mathbf{y}) \tag{6}$$

Starting from an initial guess w_i with $i = 0$ and being γ a positive constant which we call it step size, we find the new point as:

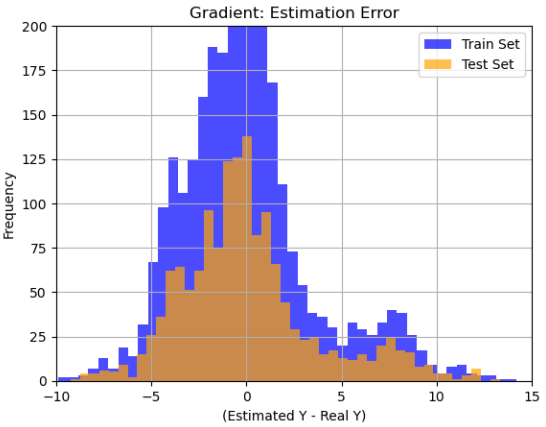
$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma \nabla f(\mathbf{w}_i) \tag{7}$$

3.2.1 Result

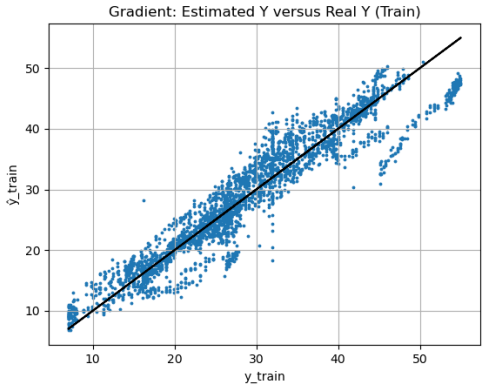
Figure 2a shows the optimum values of W^*. Compassion between this figure and figure 1a, represents less correlation in some specific values in Gradient algorithm. this correlation results in having high number of times which we have less error that is shown in figure 2b. In order to evaluate the algorithm we compare the real vector y , extracted from the training and testing dataset, with the predicted vector $\hat{y} = Xw^*$, calculated by executing the algorithm, as we can see in figures 2c and 2d.



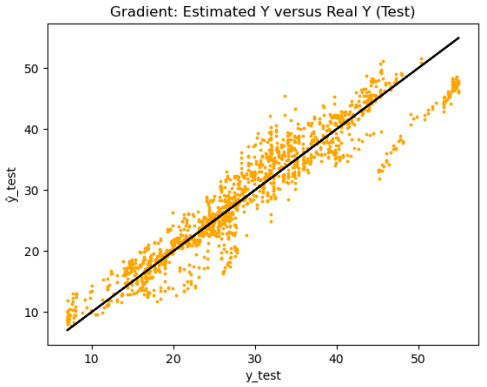
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$



(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

Figure 2: The gradient algorithm

3.3 The stochastic gradient

In the Stochastic Gradient we write the gradient of the cost function as

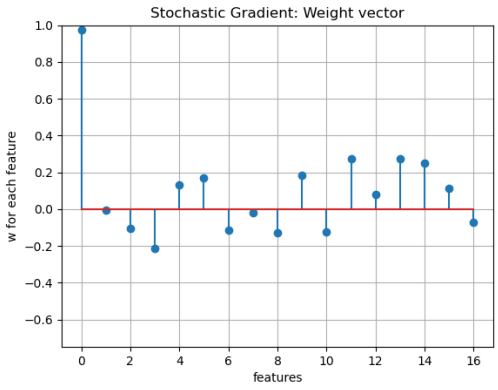
$$\nabla f(\mathbf{w}) = \sum_{n=0}^{N} \nabla f_n(\mathbf{w}) = \sum_{n=0}^{N} [[\mathbf{x}(n)]^T \mathbf{w} - y(n)]\mathbf{x}(n) \tag{8}$$

Starting from a random initial vector w_i, we find the next value of the vector as:

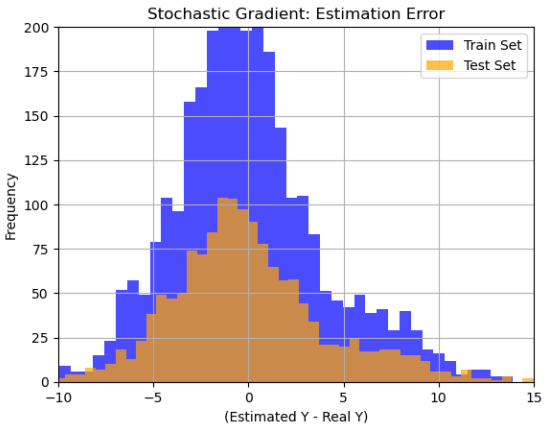
$$\mathbf{w}_{i+1} = \mathbf{w}_i - \gamma \nabla f_i(\mathbf{w}_i) \tag{9}$$

3.3.1 Result

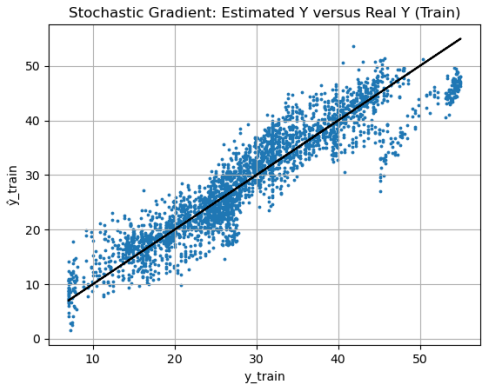
Like previous sections, the figures 3a,3b,3c and 3d are representative of the good results given by the algorithm. By looking at figure 3a it can be observed that also in this case feature 0 has most important weight over the other features (value of about 1). Points scattered in figures 3c and 3d this time seem a bit disperse.



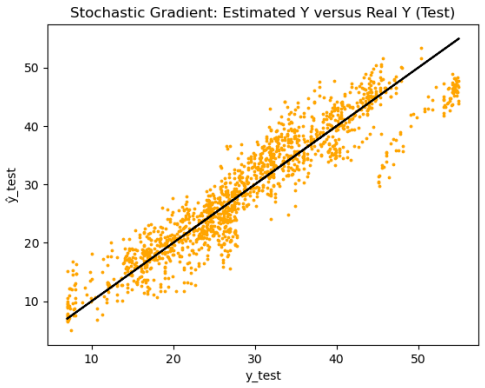
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$



(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

Figure 3: The stochastic gradient

3.4 The conjugate gradient

Considering the Linear Least Square problem

$$f(w) = ||y - Xw||^2 = \frac{1}{2}[y^T y + w^T X^T X w - 2y^T X w] \tag{10}$$

in the **conjugate gradient** we assume

$$f(w) = \frac{1}{2} w^T X^T X w - y^T X w + \frac{1}{2} y^T y = \frac{1}{2} Q w - b^T w + c \tag{11}$$

where $Q = X^T X$ and $b = X^T y$ and c is constant that doesn't influence the value W^*. The gradient of $f(w^*)$ is:

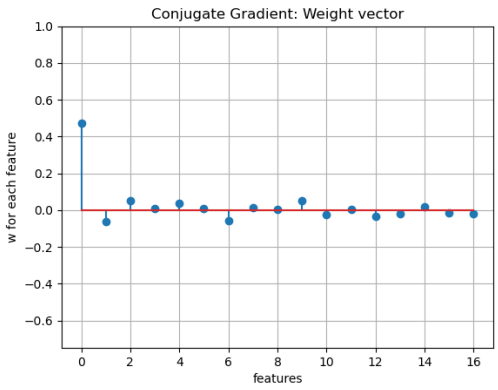
$$\nabla f(w^*) = Q w^* - b = 0 \tag{12}$$

A solution can be found with the help of *conjugate vectors* which are vectors orthogonal with respect to a matrix $\vec{Q}$. It means that the vectors $\vec{d_i}$ and $\vec{d_k}$ are Q-orthogonal if $\vec{d_i}^T \vec{Q} \vec{d_k} = 0$.

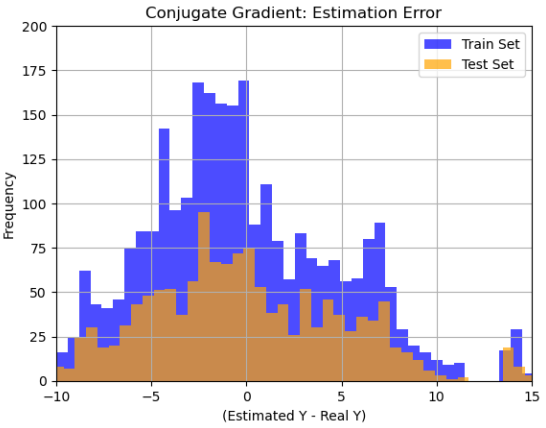
At first we set $\vec{d_0} = -\vec{g_0} = \vec{b}$ and $\vec{w_0} = 0$ as the initial solution, where $\vec{g}$ is the direction of the gradient and $\vec{d}$ is the actual direction taken by the algorithm. The next directions are computed as $\vec{d_{k+1}} = -\vec{g_{k+1}} + \beta_k \vec{d_k}$ where β is a coefficient that depends on the other parameters.

3.4.1 Result

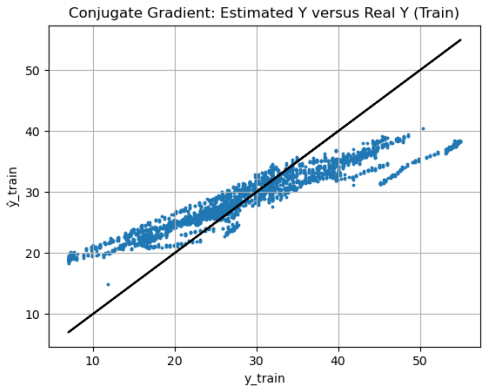
Figure 3a represents the feature with highest weight is again the first one, but the value is lower (0.5 instead of 0.95 as with the Gradient Algorithm and Steepest Descent). The other correlation of features are almost near 0. It can be noticed from figure 3b there is a wider distribution of error. The most of occurrences this time is almost around -0.3 and not exactly 0.



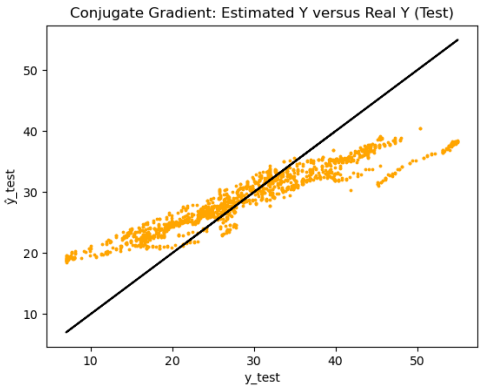
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$



(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

Figure 4: The conjugate gradient

3.5 The steepest descent algorithm

The steepest decent algorithm is used to optimize the step of the gradient algorithm. If γ is very small, it would take long time to converge and become computationally expensive. If γ is large, it may fail to converge and overshoot the minimum. Our goal is to find the optimum vector $\mathbf{w}$ and with this iterative

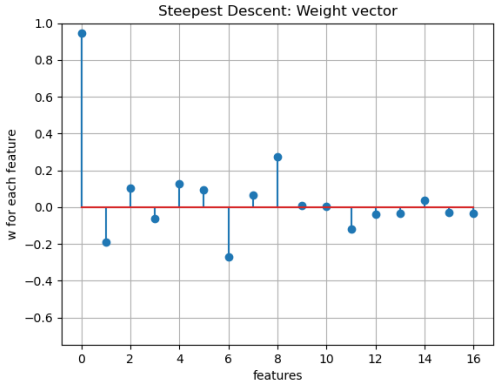
method, we will improve step by step our γ by evaluating, for each $\mathbf{x_i}$ the $\nabla f(\mathbf{x_i})$ and the Hessian Matrix in that point: $\mathbf{H}(\mathbf{x}_i) = 2\mathbf{X}^T\mathbf{X}$.

$$\gamma_i = \frac{\|\nabla f(\mathbf{x}_i)\|^2}{\nabla f(\mathbf{x}_i)^T \mathbf{H}(\mathbf{x}_i) \nabla f(\mathbf{x}_i)} \tag{13}$$

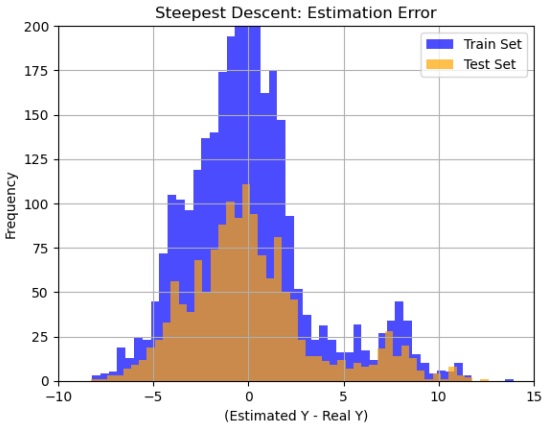
$$\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma_i \nabla f(\mathbf{x}_i) \tag{14}$$

3.5.1 Result

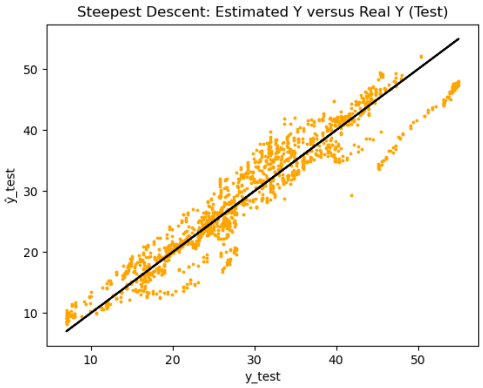
As in the previous algorithms, in the figures 5c and 5d we see how well the algorithm performs and the method fits well the dataset. in figure 5a it can seen correlation of features are almost like correlation of stochastic gradient algorithm. In histogram of figure 5b we have again the highest value in feature 0.



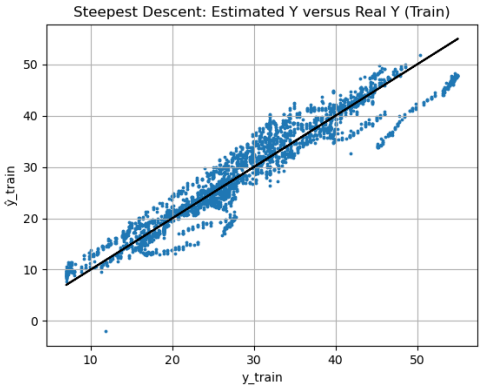
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$



(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

Figure 5: The steepest descent algorithm

3.6 The ridge regression (optimize λ)

It is possible that vector w^* takes very large values and over-fitting occurs, due to this reason, it might be convenient to solve the new problem:

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|^2 = \min_{\mathbf{w}} f(\mathbf{w}) \tag{15}$$

where λ has to be set conveniently (trial and error). The solution of this problem can then be obtained using the pseudo-inverse

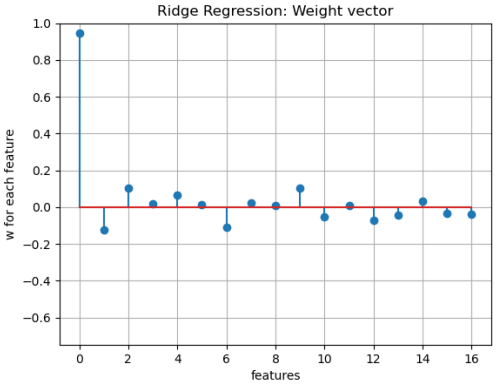
$$\nabla f(\mathbf{w}) = 2\mathbf{X}^\top\mathbf{X}\mathbf{w} - 2\mathbf{X}^\top\mathbf{y} + 2\lambda\mathbf{w} = 0 \tag{16}$$

so we get

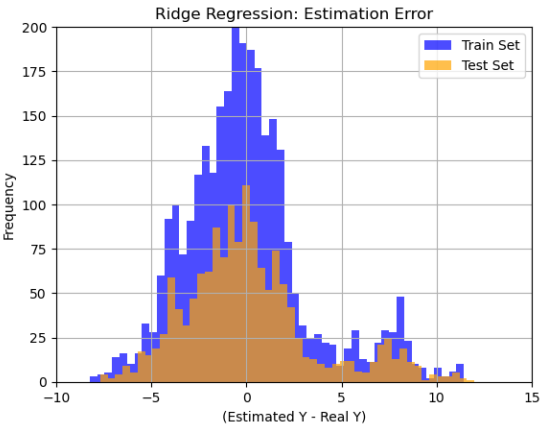
$$\hat{\mathbf{w}} = \left(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^\top\mathbf{y} \tag{17}$$

3.6.1 Result

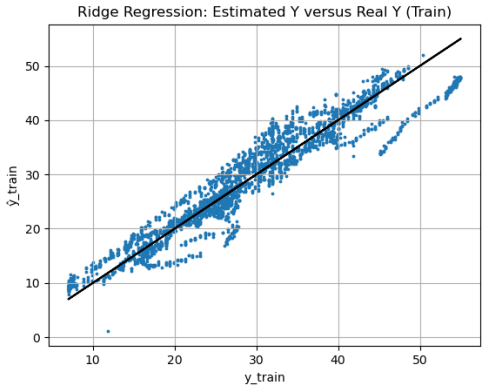
Looking at Figures 6c and 6e it can be easily noticed that those figures are very similar to the ones of Gradient and Steepest Descent Algorithms: large bandwidth (from about -0.7 to 1.1) and not symmetric shapes; of course, also the distributions of points in figure 6b are pretty much the same to those in the same algorithm named before.



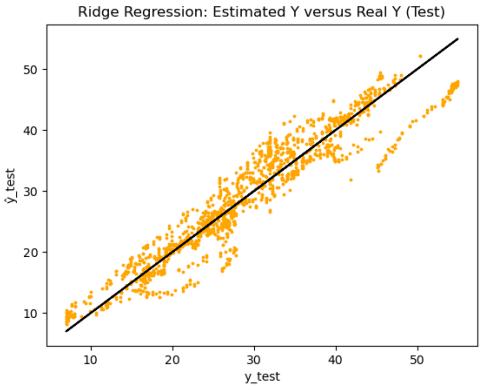
(a) w^* *optimize vector*



(b) $\hat{y} - y$ *for Test and train*



(c) $\hat{\mathbf{y}}_{\text{train}}$ *VS* $\mathbf{y}_{\text{train}}$

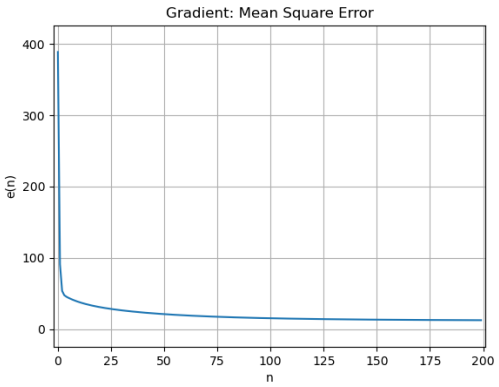


(d) $\hat{\mathbf{y}}_{\text{test}}$ *VS* $\mathbf{y}_{\text{test}}$

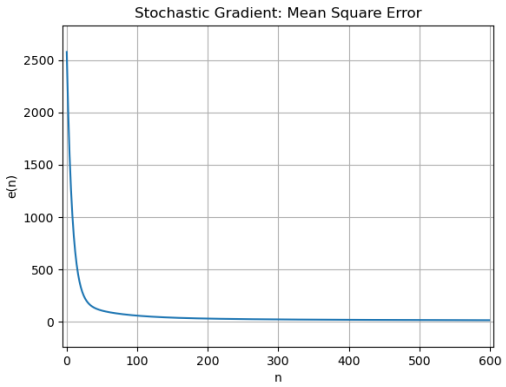
Figure 6: The ridge regression (optimize λ)

4 Conclusions

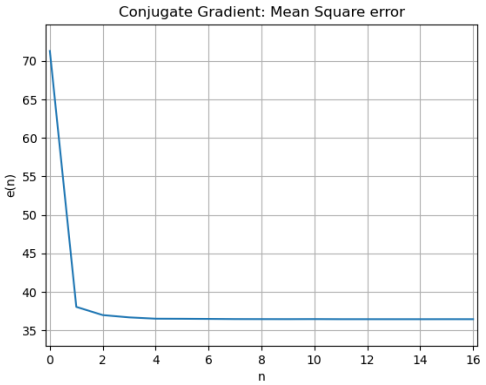
Comparing the results from all the algorithms in terms of Mean Square Error which shown in figure 7, more or less all the algorithms return similar Mean Square Error, this means that we can use each of them in the same way.



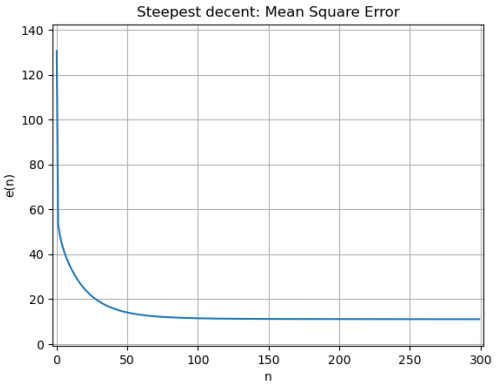
(a) *the gradient algorithm*



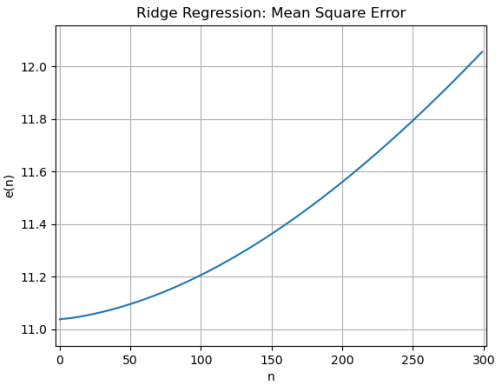
(b) *the stochastic gradient*



(c) *the conjugate gradient*



(d) *the steepest descent algorithm*



(e) *the ridge regression*

Figure 7: **Mean square error of algorithms**

Missing table with the MSE values
Is linear regression adequate from the medical point of view?