Optmisation for Machine Learning project report

Student: BENREKIA Mohamed Ali

January 10, 2022

Contents

1 Abalone Dataset presentation

Abalone are a family of reef-dwelling marine snails, Considered as delicacy in many parts of the world. The economic value of abalone is positively correlated with its age. Therefore, to determine the price of Abalone, it is important for both farmers and customers to detect its age accurately. Farmers usually cut the shells and count the rings through microscopes to estimate the abalones age. Telling the age of abalone is therefore difficult and inefficient. Our goal in this report is to leverage machine learning regression to forecast the rings, hence the age of abalones.

Figure 1: A picture of Abalone fish

Data comes from an original (non-machine-learning) study by J Nash et al [1]. The dataset contains a total number of observations of 4176 and a total number of variables of 9, and in the table below we present the description of the attributes:

Name	Data Type	Measurement	Description
Sex	categorical (factor)		M, F, and I (Infant)
Length	continuous	mm	Longest shell measurement
Diameter	continuous	mm	perpendicular to length
Height	continuous	mm	with meat in shell
Whole weight	continuous	grams	whole abalone
Shucked weight	continuous	grams	weight of meat
Viscera weight	continuous	grams	gut weight (after bleeding)
Shell weight	continuous	grams	after being dried
Rings	continuous		+1.5 gives the age in years

Furthermore, we split data into train set (80%) and test set (20%), where the first 8 variables are the features and the variable *Rings* is the target attribute.

1.1 Correlation Matrix

The graph below highlight the correlation between the 9 variables:

[1]Nash, W. J., Sellers, T. L., Talbot, S. R., Cawthorn, A. J., Ford, W. B. (1994). The population biology of abalone (haliotis species) in tasmania. i. blacklip abalone (h. rubra) from the north coast and islands of bass strait. Sea Fisheries Division, Technical Report, 48, p411.

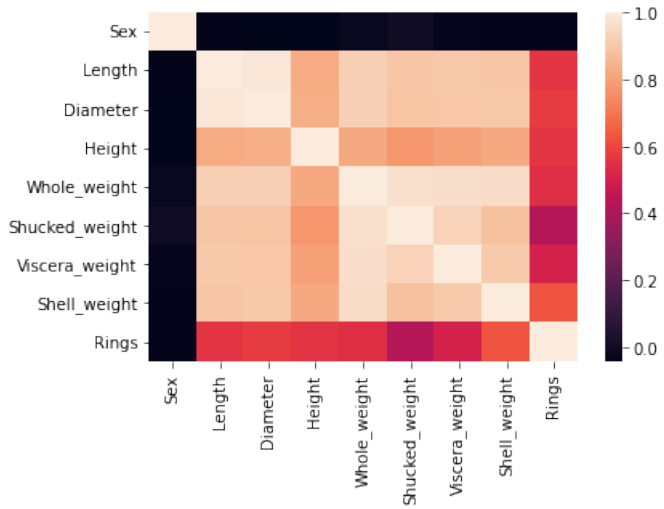

Figure 2: Correlation matrix between the variables

We notice high correlation in data. There seems to be high multicollinearity between the variables. for example correlation between *Diameter* and *Length* is extremely high (about 98.7), similarly Whole *Weight* seems to be highly correlated with other weight predictors as *Shucked weight* and *Viscera weight*.

1.2 Principle components analysis

We perform a PCA method on the train set and we show the *Scree plot* and the *PCA scores plot*:

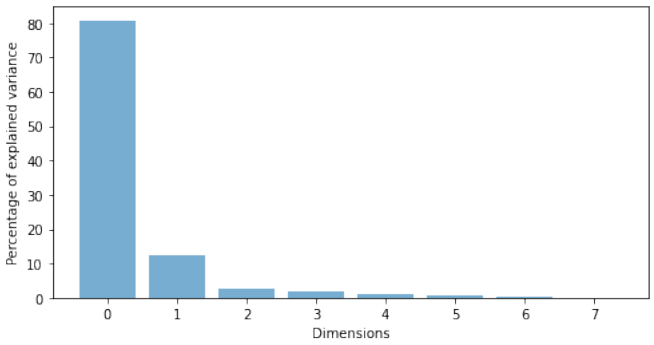

Figure 3: PCA Scree plot

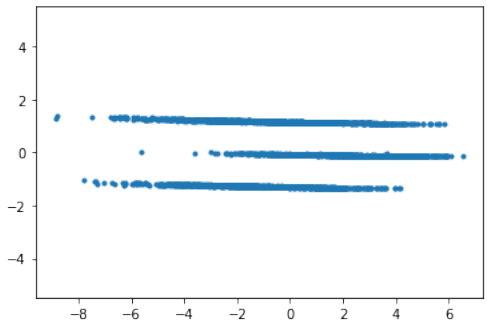

Figure 4: PCA Scores plot

We observe in the *Scree plot* that the first two components of the PCA account for more than (95%) of the variability, and the *PCA scores plot* shows that the samples, when projected into the two important PCA components, are clusted into 3 main groups representing the three type of Sex (Male, Female or Infant).

2 Problem presentation

In this section we present the problem we aim to solve and formulate it as an optimization problem, Given this dataset, we seek a model parameterized by a vector w that explains the data according to a certain loss function, and that the model can predict the target attribute *Rings* from the other 8 variables, hence we opt for a *linear regression* approach to solve the model where the loss function the *Mean square error (MSE)* and we add the l_2 regularization which corresponds to *Ridge regularization*. This results in the following formulation:

$$\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}) := \frac{1}{n}\left(\sum_{i=1}^{n}\left(\mathbf{x}_i^T\mathbf{w} - y_i\right)^2\right) + \lambda\|\mathbf{w}\|^2$$

3 Solution Approaches

In this section we present the different approaches used to solve the Linear Regression problem applied on the training set and at the end we present test and compare the results of notable approaches on the test set. Note that in what follows we denote the loss metric **MSE** as *objective function*, for implementing the problem and some solutions we used the same implementation as the implementations of the optimisation lab sessions (mainly for describing the linear regression, batch and accelerated gradient descent, stochastic gradient descent).

3.1 Test of Batch Gradient Descent

To implement the *Batch Gradient Descent* we need to formulate the gradient of our function to minimize:

$$\nabla f(\mathbf{w}) = \frac{1}{n}\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda\mathbf{w}$$

We test the BGD method with different learning rates/ step sizes:

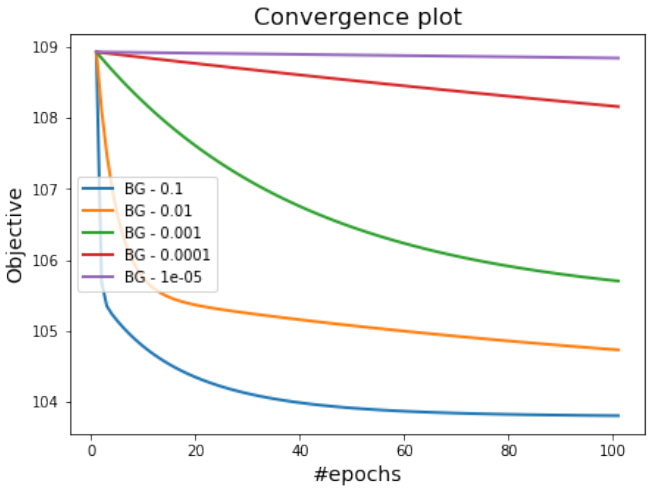

Figure 5: Batch gradient descent with different learning rates

We observe that the *Batch gradient* has a convergence rate of $O(1/K)$ where higher values of the learning rates notably (LR $= 0.1$) gives better convergence rate (the model convergence quickly, while lower values of the learning rate slows the convergence.

3.2 Test of Accelerated Gradient Descent

We implemented the *Nestrov's accelerated gradient descent* with a fixed learning rate 0.01 and variable momentum parameter and we compared it to *Batched gradient Descent*, one iteration with *Nestrov's method* can be formulated as follow: The gradient descent stage:

$$w_{t+1} = \theta_t - \varepsilon_t \nabla f\left(\theta_t\right)$$

followed by the momentum-like stage (μ is the momentum):

$$\theta_{t+1} = \phi_{t+1} + \mu_t \left(w_{t+1} - w_t\right)$$

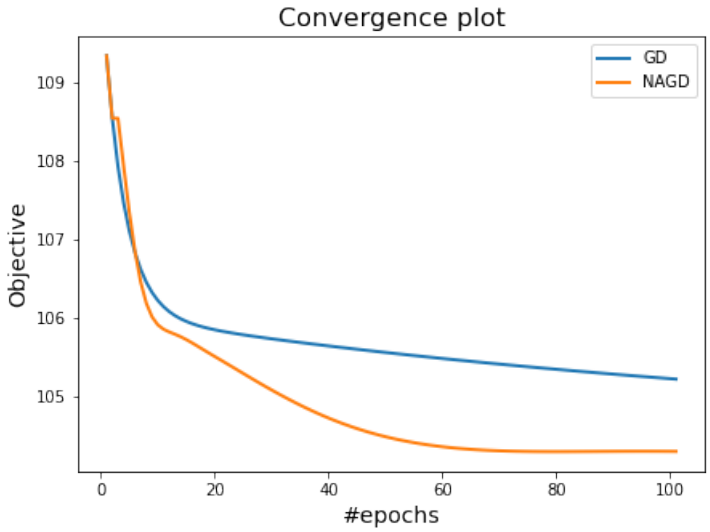

Figure 6: Nestrov's Accelerated gradient descent Vs Batch Gradient Descent

It is clear from the plot that *Nestrov's method* converges at a better rate than *Batch gradient descent* due to utilizing the acceleration with the momentum when calculating the gradients, which helps achieving optimal values quicker.

3.3 Test of Stochastic gradient Descent

In what follows we show different experimentations with the known *Stochastic gradient Descent* on our training set, we formulate the partial gradient as follow:

$$\nabla f_i(\mathbf{w}) = \left(\mathbf{x}_i^T \mathbf{w} - y_i\right) \mathbf{x}_i + \lambda \mathbf{w} \text{ for every } iinsamples$$

3.3.1 Constant Vs Decreasing Learning rate:

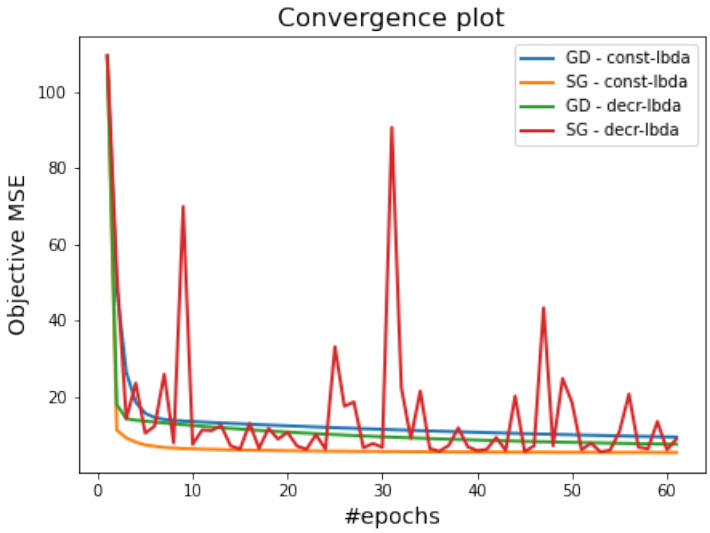

Figure 7: SGD and BGD with Constant Vs Decreasing LR

Unlike *Batch Gradient*, *Stochastic Gradient* is not guaranteed to decrease the function value in a monotone fashion, hence we notice the behavior of SGD when using a decreasing $LR = LR/10$ every 10 epochs, also SGD with a constant LR gives the best convergence rate yet it is important to mention that SGD takes relatively more time on training than classical BGD.

3.3.2 Different constant Learning rate:

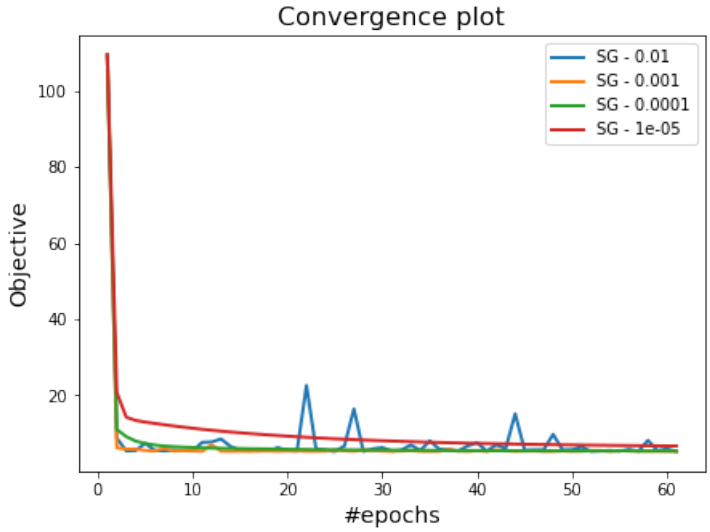

Figure 8: SGD with multiples Constant LR

We notice two important things, first SGD converges to better values of the objective function than BGD and second, when using lower values of LR, the SGD has a stable behavior (LR= 1e-5).

3.3.3 Different decreasing Learning rate:

In this experimentation we change how we decrease the learning rate during the training with the factor α such that $LR = LR/\alpha$ every 10 epochs:

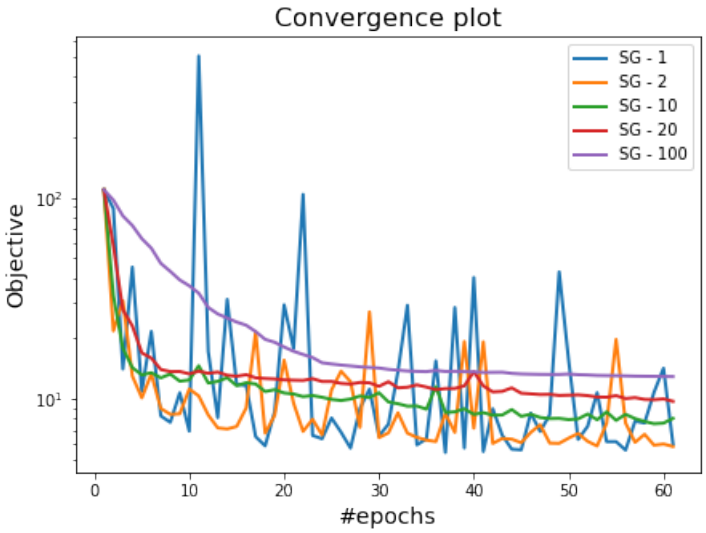

Figure 9: SGD with multiples Decreasing LR

We plot the objective function in a log scale due to the oscillation, we observe that the higher the α factor the less variation (oscillation) by the SGD model yet the model converges to non optimal values in contrast to higher values of α where the model can reach lower values of the objective function.

3.3.4 Different Batch size:

We experiment with batch sizes $n_b \in \left\{1, \frac{n}{100}, \frac{n}{10}, \frac{n}{2}, n\right\}$:

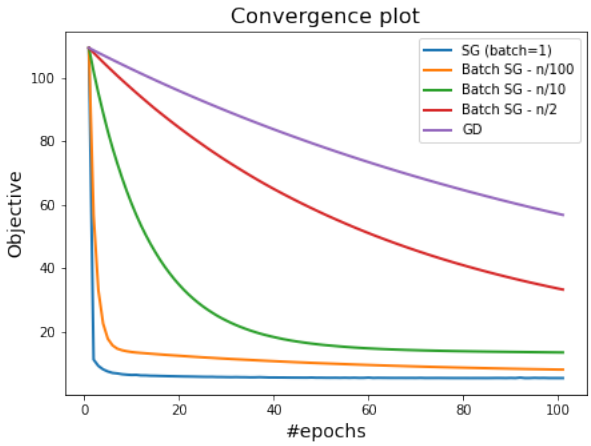

Figure 10: SGD with different batch sizes

We notice in this plot that, the closer we are to a BGD ($n_b = n$) the convergence rate is slow yet the training is fast, in contrast to, when we lower the batch size, the closer we are to SGD ($n_b = 1$), the convergence rate is quick yet the training is slow, this agrees with the theoretical guarantees established during the lectures.

3.3.5 Stochastic Gradient with Averaging:

Averaging helps maintaining an average of all SGD iterates and outputs the last average, as well as the function values, yet it is known that it smooths the trajectory of the SGD and doesn't perform differently as we can see in the plot below:

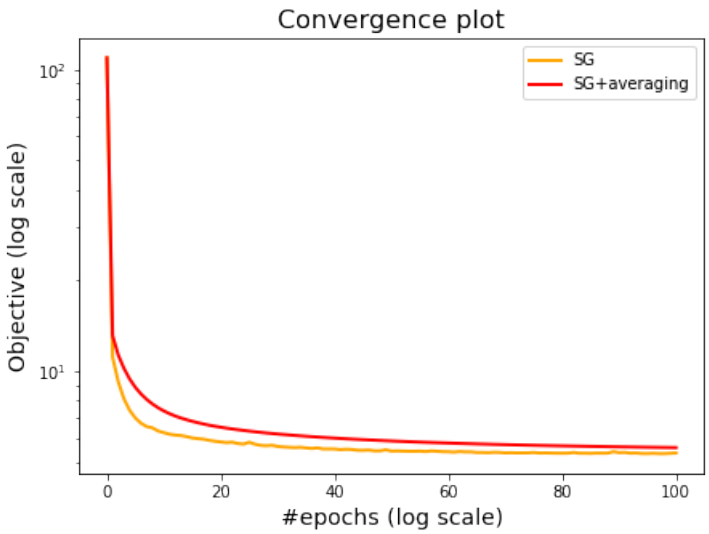

Figure 11: SGD with Averaging

3.3.6 Stochastic Gradient with Diagonal Scaling:

This variant of SGD relies on a diagonal scaling of the gradient. This corresponds to rescaling the stochastic gradient step component-wise as follows:

$$[\mathbf{w}_{k+1}]_i = [\mathbf{w}_k]_i - \frac{LR}{\sqrt{[\mathbf{v}_k]_i + \mu}} [\nabla f_{i_k}(\mathbf{w}_k)]_i$$

where $\mu > 0$ is a regularization parameter, and $v_k \in \mathbb{R}^d$is defined recursively by $v_{-1} = 0_{\mathbb{R}^d}$ and

$$\forall k \geq 0, \forall i = 1, \ldots, d, \quad [\mathbf{v}_k]_i = \begin{cases} \beta [\mathbf{v}_{k-1}]_i + (1-\beta) [\nabla f_{i_k}(\mathbf{w}_k)]_i^2 & \text{for RMSProp} \\ [\mathbf{v}_{k-1}]_i + [\nabla f_{i_k}(\mathbf{w}_k)]_i^2 & \text{for Adagrad} \end{cases}$$

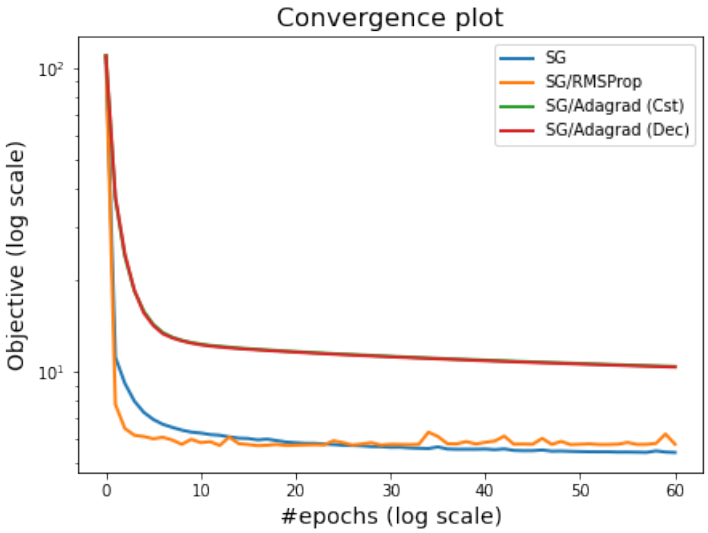

Figure 12: SGD with Diagonal Scaling

We observe that the diagonal scaling with RMSProp has the best convergence rate, although it suffers some oscillation, and the classical SGD converges to relatively better values of the objective function. It is important to mention that the use of $\mu > 0$ prevents the scaling of each component of the stochastic gradient from going to zero.

3.4 Lasso regression with Iterative Soft Thresholding (ISTA)

In this experimentation we slightly change our function minimize where we replace the *Ridge* regularization with *Lasso* regularization (l_1 norm), and we use the so-called iterative soft thresholding (ISTA), aka proximal gradient aka forward-backward. It performs first a gradient step (forward) of the smooth part of the function and then a proximal step (backward) step which corresponds to the soft-thresholding operator :

$$\mathcal{S}_s(z) \equiv \max(|z| - \lambda, 0)\,\mathrm{sign}(z)$$

One iteration of this algorithm is as follow :

$$w_{k+1} = \mathcal{S}_{\lambda\tau}\left(w_k - \tau X^\top (X w_k - y)\right)$$

We plot the graph corresponding to the objective values in function of different λ:

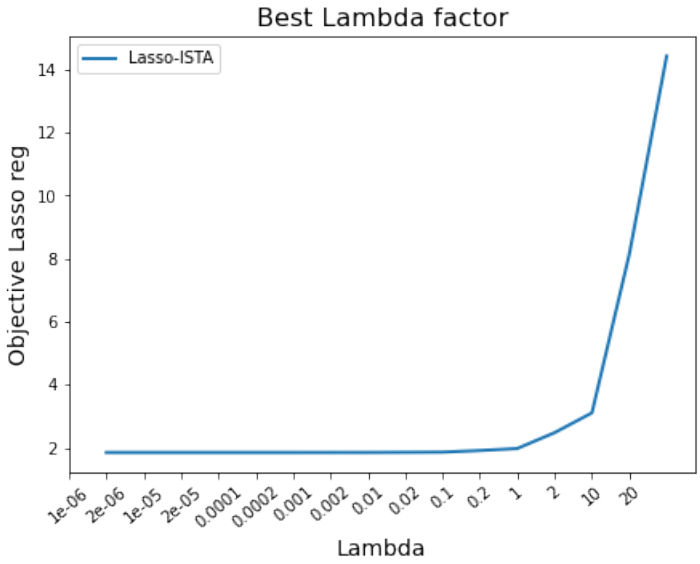

Figure 13: Lasso regression with ISTA

We notice that lower values of λ gives better results in term of objective value (i.e: MSE loss).

4 Performance on Test set

We compare the result we obtained when applying the SGD model and ISTA on the test set and we measure the MSE loss:

- **Classical SGD** : MSE Loss = 5.750
- **ISTA** : MSE Loss = 1.843

5 Conclusion

Throughout this project we had the opportunity to look more in-depth and implement well known optimization algorithms to solve machine learning problems and with real-world datasets. We presented the Abalone dataset and formulated the problem as a regression problem where we aim to predict the age of Abalone fish, then we experimented on several Gradient Descent algorithms notably (Batch Gradient, Accelerated Gradient and Stochastic Gradient), also, we implemented the Lasso regression using iterative soft thresholding which held the best results on the test set.