Machine learning project 1

Nikravech David, Rodriguez Mateos Borja, Salaün Antoine

***Abstract*—**

I. Introduction

The Higgs Boson is an elementary particle which explains why other particles have mass. Its discovery occurred during collision experiments, generating big amount of data. One example is the Higgs Boson data-set taken from the Atlas experience at CERN, that was analyzed during this project. Data-sets like these were used to distinguish Higgs Boson's signals from background noise, therefore performing a binary classification.
The aim of this project is to use machine learning algorithms in order to find the best approach to complete this task. One will first implement six basic machine learning algorithms. One of these algorithms will then be further developed into a methodology, which can be used to predict the presence of the Higgs Boson.

II. Models and Methods

In order to study extract information out of output variables or classify, one can proceed in using different types of regression over the input data and predicts outputs for new inputs. One would like to approximate the output data y_n by using regression $f_{\vec{w}}(\vec{x_n})$ over the input variables $\vec{x}_n$, organised in a matrix X, weighted by a weight vector $\vec{w}$, i.e. $y_n \approx f_{\vec{w}}(\vec{x}_n)$.

A. Linear regression

When using the linear regression model, one assumes that the input and output data follow a linear relation, i.e. :

$$y_n \approx f_{\vec{w}}(\vec{x}_n) = w_0 + \sum_{d=1}^{D} x_{nd} w_d = X^T \vec{w} \tag{1}$$

The error made by the model is quantified using a loss function. More precisely, the loss function used throughout this project is the Mean Square Error (MSE).

$$L_{MSE}(\vec{w}) := \frac{1}{2N} \sum_{n=1}^{N} [y_n - f_{\vec{w}}(\vec{x}_n)]^2 \tag{2}$$

The goal is to minimize the loss function to find the best weights associated with the linear model through different means.

a) Using gradient descent (GD): To do so, one can apply the gradient descent method (GD). The gradient of the loss function is defined by 3.

$$\vec{\nabla} L(\vec{w}) = -\frac{1}{N} X^T \vec{e} \tag{3}$$

where $\vec{e} = \vec{y} - X\vec{w}$ is the error vector.

One looks for the minimum of the loss function, with respect to the weights, by iteratively updating the weights with a step in the opposite direction of the gradient. This procedure is repeated the number of times needed in order to minimize the loss 4 with respect to $\vec{w}$. γ is the step size.

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \gamma \vec{\nabla} L(\vec{w}^{(t)}) \tag{4}$$

b) Using stochastic gradient descent (SGD): The loss minimization can also be done following the SGD method. One rewrites the loss $L_{MSE}(\vec{w})$ as :

$$L_{MSE}(\vec{w}) = \frac{1}{N} \sum_{n=1}^{N} L_n(\vec{w}) \tag{5}$$

The gradient of the loss caused by the training of the nth sample is

$$\vec{\nabla} L_n(\vec{w}) = -\vec{x}_n^T \cdot (y_n - \vec{x}_n^T \cdot \vec{w}) \tag{6}$$

The weights are then updated the following way :

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \gamma \vec{\nabla} L_n(\vec{w}^{(t)}) \tag{7}$$

It is worth noting that at every step the loss of a different sample is taken to compute the weight's update and the SGD has computational cost in $\mathcal{O}(D)$ while for GD it is $\mathcal{O}(N \cdot D)$, N being the number of data and D the number of features. In conclusion, SGD method is computationally N times cheaper than GD method.

B. Least squares regression using normal equations

Another method to compute the weights minimizing the loss is the least square regression, using normal equations. Assuming the loss function is convex, the minimum can be found by setting the gradient of the loss 4 to zero and solve, for $\vec{w}$ the system of equations 8.

$$X^T(\vec{y} - X\vec{w}) = 0 \Longleftrightarrow X^T\vec{y} = X^T X \vec{w} \tag{8}$$

C. Ridge regression using normal equations

When one wants to have a more powerful linear model to fit of the data, one can augment the feature vector. But this leads to an over-fitting of the data. To counter this, one can use the regularization, that is adding a penalty term $\Omega(\vec{w})$ to the loss $L_{MSE}(\vec{w})$, that will limit the complexity of the model. In general, one uses the L$_2$-norm to regulate, namely $\Omega(\vec{w}) = \lambda ||\vec{w}||_2^2$. It will minimize the effect of large weights. Since in this project the MSE is used, with the penalty term, the model is named ridge regression and consists in minimizing :

$$\frac{1}{2N} \sum_{n=1}^{N} [y_n - f_{\vec{w}}(\vec{x}_n)]^2 + \lambda ||\vec{w}||_2^2 \tag{9}$$

Note that when $\lambda = 0$, it is the least square regression.
Taking the gradient, with respect to the weights, of 9 and setting it to zero leads to :

$$-\frac{1}{N} X^T (\vec{y} - X\vec{w}) + 2\lambda\vec{w} = 0 \tag{10}$$

After algebraic transformations, one needs to solve, for $\vec{w}$, the following system of equations :

$$X^T\vec{y} = (X^T X + 2N\lambda I)\vec{w} \tag{11}$$

D. Logistic regression

When one performs binary classification on a set of data, and considers the problem as a regression one and also interprets the predicted outputs as probability, it is necessary to perform a transformation on the values, shifting and restricting them to a [0,1] interval. To do so, one may use the logistic function :

$$\sigma(t) = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}} \tag{12}$$

For a given weight vector $\vec{w}$ and a given feature vector $\vec{x}$, one can predict the probability of the class label, 0 and 1, for a given $\vec{x}$ to be :

$$\begin{aligned} p(1|\vec{x},\vec{w}) &= \sigma(\vec{x}^T\vec{w} + w_0) \\ p(0|\vec{x},\vec{w}) &= 1 - \sigma(\vec{x}^T\vec{w} + w_0) \end{aligned} \tag{13}$$

In order to determine the best weights to classify the best the data, one needs to minimize the following loss function :

$$L_{log}(\vec{w}) = \sum_{n=1}^{N} \log\left(1 + exp(\vec{x}_n^T\vec{w})\right) - y_n\vec{x}_n^T\vec{w} \tag{14}$$

The gradient of such a loss function is :

$$\vec{\nabla} L_{log}(\vec{w}) = X^T(\sigma(X\vec{w}) - \vec{y}) \tag{15}$$

it is important to note that the notation $\sigma(X\vec{w})$ means it is the application of the logistic function to each component of the column vector $X\vec{w}$ and for one data point :

$$\vec{\nabla} L_{log,n}(\vec{w}) = \vec{x}(\sigma(\vec{x_n}^T\vec{w}) - y_n) \tag{16}$$

One just needs to apply the GD or SGD method to update the weights with the previously given gradients.

E. Regularized logistic regression

In the case where the data are linearly separable, there is no finite weight vector minimizing the cost function L_{log}. Therefore, one must add a penalty term to it. Thus the problems transforms into the minimization of :

$$L_{log,reg}(\vec{w}) = \sum_{n=1}^{N} \log\left(1 + exp(\vec{x}_n^T\vec{w})\right) - y_n\vec{x}_n^T\vec{w} + \frac{\lambda}{2}||\vec{w}||_2^2 \tag{17}$$

	Parameters			
Methods	λ	γ	max$_{iters}$	Pred [%]
Regularized Logistic Regression				
Logistic Regression				
Ridge Regression				
Least Squares				
Gradient Descent				
Stochastic Gradient Descent				

TABLE I
MANDATORY ALGORITHMS COMPARISONS TABLE.

F. Algorithms' implementation

The implementation of the six methods was following the guideline presented in the lecture notes. They are listed in the table I, with the relevant parameters used and their prediction/performance rate on the training data set. One must note that the penalty term λ and the step-size γ were chosen for each method after a parameter scan **??** over the training data-set. The maximum number of iteration was chosen randomly. The methods were trained over % of the training data set and the remaining % were tested on.

The prediction accuracy are ranging from % to %. One can see that the INSERT FUNCTION works the best on the data-set, while the others work less well. The INSERT FUNCTION will be used for the rest of the project.

G. Data Analysis and pre-processing

The data on which the regression will be made is composed of 250'000 training data points and 570'000 test points. These data points have 30 different parameters. First of all it is possible to notice that there are some '-999' values in the data-set, these correspond to a 'Not a Number' value. If these values are not pre-processed it would create outliers in the data-set hurting the accuracy of the prediction of our model.

To treat these values they are replaced by the mean or the median value of the corresponding feature, thus setting the value to zero or close to zero when the normalization of the data is done.

Then the data is normalized following the standard relation:

$$X' = \frac{X - \bar{X}}{\sigma} \tag{18}$$

with $\bar{X}$ the mean value of the feature without counting the 'NaN' and σ its standard deviation.

Finally by visually analyzing the data it is possible to see that the feature 'PRI_jet_num' takes finite values 0,1,2,3, also there seems to be a correlation on the number of 'NaN' of some features and the 'jet' value. Knowing this, the data-set will be divided in sub data-sets for each value of the feature 'PRI_jet_num'.

H. Cross-validation

As mentionned in II-F, the INSERT FUNCTION Regression was used for the training and the prediction of Higgs' bosons. In particular, one used the cross-validation in order to train the model on a wider range a data and to minimize the variance. One can also plot the losses with respect to the number of folds used during the cross-validation METTRE PLOT K-FOLDS.

Fig. 1. Signal compression and denoising using the Fourier basis.

I. Parameter Scan

As mentioned in II one implemented a parameter scan over the penalty term λ and the step-size γ in order to minimize the loss of all the methods with respect to these two hyper-parameters. The parameter scan was performed on % of the training data-set, namely 3 of the four sub-set defined by the feature 'PRI_jet_Num'. The optimum parameters for each function is reported in Table I, while the graphs **??** **??** display the averaged loss over the training data-set used as a function of λ and γ respectively.

III. Results and interpretation

A. Linear regression using gradient descent

B. Linear regression using stochastic gradient descent

C. Least squares regression using normal equations

D. Ridge regression using normal equations

E. Logistic regression using gradient descent or SGD

F. Regularized logistic regression using gradient descent or SGD

IV. Conclusion

Fig. 2. Signal compression and denoising using the Daubechies wavelet basis.