Machine Learning (CS-433) – Project 1

The Higgs Boson Machine Learning Challenge

Thomas Benchetrit, Romain Palazzo, Eliott Zemour

October 26, 2020

1 Introduction

The Higgs boson is an elementary particle of the Standard Model which provides an explanation for the origin of mass of other particles. It was theorized in 1964 by Peter Higgs and recently discovered at the Large Hadron Collider at CERN in 2013 by François Englert and Peter Higgs. Since the Higgs boson decays rapidly into other particles, scientists don't observe it directly, but rather measure its "decay signature".
Therefore, assuming that many decay signatures are similar, this machine learning project aims to estimate the likelihood that a given event's signature was the result of a Higgs boson (signal) or some other process/particle (background). In practice, given a vector of features representing the decay signature of a collision, the learning model described in this report aim to predict whether this event is a signal or something else.

2 Learning model

The data of the project is made physical measurements from particle collision made at the CERN. The goal is to be able to tell if one has observed a Higgs boson from available vector of features. Therefore, the result of the model must be binary in the way either one detect or do not detect a Higgs Boson. A logistic model gives a binary result from the data unlike linear regression for instance. That is why the model used in this project is similar to logistic regression. This model is based on the likelihood function which defines the probability of observing a sample, given a set of weights $\mathbf{w}$. According to the course, a good choice for the probability definition is the sigmoid function $\sigma(x)$ which gives the probability law of eq(1) with the possible result y=$\{0,1\}$, $\mathbf{x}$ the data and $\mathbf{w}$ the weights :

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

With this the likelihood function is defined as $L(\mathbf{w}) = \prod_{i=1}^{N} p(y_i|\mathbf{x_i},\mathbf{w})$ for N samples (each samples is supposed independent, thus the probability of observing all is given by the product of the probability of observing each). In practice, the log-likelihood $l(\mathbf{w}) = \ln(L(\mathbf{w}))$ is rather used because its computation is easier. Then the idea is to maximize $L(\mathbf{w})$ (or $l(\mathbf{w})$ by the properties of the log function) for $\mathbf{w}$. This means maximizing the probability of observing all the samples that are indeed observed. Let f be the loss function $f(\mathbf{w}) = -l(\mathbf{w})$. Maximizing $L(\mathbf{w})$ is therefore equivalent to minimizing $f(\mathbf{w})$. Thus after some computation using ln and exp properties and a division by N which prevent big value, the loss function f of the model is defined by Eq(2) :

$$f(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(1-y_i)\mathbf{x}_i^T\mathbf{w} - \log(\sigma(\mathbf{x}_i^T\mathbf{w})) \tag{2}$$

With this an **Optimisation proceed** can be made. The iterative algorithm **Gradient descent** is used for this project to found the best fit. This choice is motivated by the convexity of f which assure descent. However a Ridge regulation is done to limit the size of $\|\mathbf{w}\|$. A step in the descent is defined by $\mathbf{w_{i+1}} = \mathbf{w_i} - \gamma\nabla\mathbf{f}^*(\mathbf{w_i})$ with γ is the desent step length, $f^*(\mathbf{w}) = f(\mathbf{w}) + \lambda\|\mathbf{w}\|$ where λ is the regularisation parameter. $\nabla f^*(\mathbf{w_i})$ is given in eq(3) :

$$\nabla f^*(\mathbf{w}) = 2\lambda w_i + \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\big(\sigma(\mathbf{x}_i^T\mathbf{w}) - y_i\big) \tag{3}$$

3 Data preprocessing

To begin,analyzing and transforming the data is recommanded before applying any learning algorithm. This can prevent a bad fit due to outliers or it can rearrange the data distribution in order to be more meaningful. The first concern about the given data set is how the model should deal with missing datas which are defined by the value '-999' in the csv files. As the features are multiplied by the weights, setting missing data to 0 does not influence the research of the best weights. However setting the values to 0 may have influence on the distribution of a feature. That is why any modification on the distribution should be done before this manipulation. **Standardisation** and **Skewed data** are two methods used in the project to change distributions. The first method is used on normal distributed data in order to not manipulate too high data value and make each data more comparable with each other in a same feature. For each feature, the method subtracts each data by its mean and divides the result by its standard deviation. This way, the mean of the modified data is set at 0 and its variance at 1. In order to standardize not normally distributed features, these lasts are first skewed using the log1p function. Then an empirical analysis of the data has been done (thanks to the information given in the worksheet) and reveals that the feature 22 allows one to split data into different subsets according to the 'jet' (zero,one and multiple jets).The missing features are dropped accordingly, and a classifier is trained on each specific subset. Jet has a physical meaning related to the type of particle studied and explains why there is features with lot of missing values for instance. Finally to be able to fit the data more closely and have a more accurate model a polynomial feature expansion has been done. The maximum degree of the polynomial will be determined by a grid-search.

4 Avoiding overflows in logistic loss and gradient computation

The implementation of the logistic regression loss involves the so-called sigmoid function $\sigma(x)$ defined by:

$$\sigma(z) := \frac{1}{1+\exp(-z)} \qquad (4)$$

- The terms in $1+\exp(-z)$ are on very different scales for sufficiently large $z > 0$. As a result, round off errors will cause the denominator to get truncated to 1.
- The `numpy.exp()` function overflows for values above 710 (float64), and it constitutes a problem when it comes to apply the `numpy.log()` in Eq.(2). Indeed, we obtain `numpy.log(numpy.exp(750))` $=$ `inf` $\neq 750$.

A stable implementation of the logistic loss (Eq(2))is proposed. The last term in Eq.(2) requires special attention in order to avoid any overflows or round off errors. We make use of an accurate implementation of the log-sigmoid function inspired by Mächler Martin[1] (2012). Indeed, the so-called `logsig()` method used to compute $\log(\sigma(x))$ is defined as following:

$$\log\big(\sigma(x)\big) = \begin{cases} x & x < -33 \\ x - \exp(x) & -33 \leq x \leq -18 \\ -\log(1+\exp(-x)) & -18 < x \leq 37 \\ -\exp(-x) & 37 < x \end{cases}$$

Finally, a robust implementation of the logistic loss (Eq.(3)) gradient is necessary in the framework of the gradient descent method. The $(\sigma(\mathbf{x_i^T w}) - y_i)$ term is computed as follow in order to avoid overflows resulting from the sign of x (note that both expressions are mathematically equivalent):

$$\sigma(x) - y = \begin{cases} \frac{1}{1+\exp(x)}\big((1-y)\exp(x) - y\big) & x \leq 0 \\ \frac{1}{1+\exp(-x)}\big((1-y) - y\exp(-x)\big) & x > 0 \end{cases}$$

5 Results

After having implemented the learning model, and verified the functioning of the gradient-descent algorithm, an hyperparameter optimization is processed. Indeed, a grid search over feature polynomial expansion's degree and λ have been made. As usual, separate models are trained according to jet number, and the results obtained are the following:

	jetnum = 0	jetnum = 1	jetnum = 2
degree	3	3	3
λ	7.3×10^{-4}	7.9×10^{-4}	4.5×10^{-4}

Table 1: Optimal degrees and λ obtained from the grid search, max iter $= 700$ and $\gamma = 0.085$.

As an example, these optimal parameters are used in a learning process over half of the train dataset, (ratio $= 0.5$, `max_iter` $= 700$ and $\gamma = 0.085$), and the scores (percentage of good predictions) obtained are 83.8%, 78.9% and 80.3% for jetnum $= 0, 1, 2$ respectively.

6 Discussion

First, after comparison with and without the data preprocessing, the results were well better with the preprocessing. According to the result, splitting the data for different jetnum seems good because the value of λ changes depending on the jetnum. Moreover the quality of the fit varies with the jet num, jet num 0 is better fitted than jet num 1 for instance. However the three fits gives approximately the same result, near to 80% which is close to the result obtain in the AiCrowd submission (81.1%), which indicates that the model is consistent. Moreover, other method was tested as K-nearest neighbours or Newton descent but the results were not better or the time complexity too high.

7 Conclusion

During this project, the trained model was able to determine with an 81% accuarcy if a given set of data was a Higgs boson or not. To go further, a more extensive data analysis could have been pursued. Moreover, other training models could have been use, such as Support Vector Machine, or Deep Learning.

[1]Mächler, Martin (2012) "Accurately Computing log(1- exp(-|a|)) Assessed by the Rmpfr package". *The Comprehensive R Archive Network.*