Machine Learning Project 1 report

Charles Thiébaut, Mohammed Amine Tourari, Benjamin Raymond
Ecole Polytechnique Fédérale de Lausanne

***Abstract*—"On the fourth of July 2012, researchers at CERN announced the discovery of a new particle in the mass region of 126GeV. This observation is consistent with Higgs boson by the Standard Model. This particle is the simplest manifestation of the Brout-Englert-Higgs mechanism". [1]**

I. INTRODUCTION

The goal of this project is to predict if a given sample corresponds to a boson or to background noise using the machine learning techniques and algorithms learned over 5 weeks.
The data doesn't come from real observation but was generated through a simulation base on knowledge on particle physics (official ATLAS full detector simulator). Three type of events are generated: one signal and two background samples. All events have 30 features. Feature description can be found in the challenge description [2].

II. PRE-PROCESSING

Numerous techniques have been tried for both features expansion and data cleaning.

A. Data cleaning

Most of the data cleaning had to do with a lot of missing values (-999) , described below are some methods that we have tried and used depending on the models described later.

Mean replacement
: Replace invalid value by the mean computed on valid values (without the -999).

Drop
: Drop column if more than 50% of the values are invalid.

First method was used for all model before data split was put in place.

B. Data Expansion and splitting

Polynomial basis
: Simplest method of feature expansion used and first seen in class, polynomial bases were added to the original data using the following method :

$$\phi_i(x) = x^i \tag{1}$$

$$\forall i, 1 < i < N$$

Other basis were tried: Cosine and polynomial roots but as their didn't provide satisfying results on baseline, we chose not to use them in our final model.

Gaussian distance
: A lot of features looked to have either one of three categorical names, therefore we thought of adding a similarity metric by category as follow :

$$K(x, y) = \exp\left(-\frac{\|x-y\|_2^2}{2\sigma}\right) \tag{2}$$

$$\forall x, y \text{ with } x, y \text{ in same category.}$$

$$\sigma = 1 \text{ standard deviation,}$$

chose to be one as if data was standardized, worked better than with other value in experiments.
One could see that the above function is a non linear form of euclidean distance with a negative sign. Therefore it measures similarity rather then distances between two given points. It's also bounded to zero and one which make sense for a measure of similarity.

Angles expansion
: Similarly to what is described above, some features had names containing phi, which was confirmed by reading the documentation of the project to be angles. Similarly to what was done before, a metric was added to angles. As cosine base were tried on all data but didn't provide convenient results (cosine bases could be seen as "Fourier basis"), we limited those bases to column that contained angles. We also added pairwise cosine distance following similar categorical argumentation described above.

Standardize
: Subtract mean and divide by standard deviation. However as this didn't provide convenient result on baseline model, we chose not to use it. Magnitude of a given feature might be more important our cases.

Jet Split
: As described in the project documentation, samples results depends a lot on the value of `PRI jet num` (column 22 of the original data). This value is the only categorical value present in the original data. Three model were therefore constructed depending on the value of this column (one when it was 0, one when 1, one when more than 1).

III. METHODS

For this python project, the use of external library apart from `numpy` was forbidden. Find below the algorithm suggested by the project outline.

A. Least Squares

Applying the formulas seen in class and using, as advised during the exercise session, `numpy.linalgsolve` rather then inverting a matrix using `numpy.inverse` to make the operation more reliable.
The goal of this method is to minimize the following loss :

$$L(w) = \frac{1}{2N}(y - Xw)^T(y - Xw) \tag{3}$$

which has a minimum with the close form:

$$w^* = (X^TX)^{-1}X^Ty \tag{4}$$

Least squares was also implemented using Gradient Descent and Stochastic Gradient Descent using the following formula :

$$w^{(t+1)} = w^{(t)} - \gamma^{(t)}\nabla L(w^{(t)}) \tag{5}$$

where L is the loss with respect to w and is given by:

$$\nabla L(w^{(t)}) = \frac{-1}{N}(X^T(y - Xw)) \tag{6}$$

B. Ridge Regression

This was the second method seen in class, it was described as a method to counter the negative effect of over fitting through penalizing too complex models.

The function to minimize in a sum of least squares and a penalization term :

$$L(w) = \frac{1}{2N}(y - Xw)^T(y - Xw) + \lambda\|w\|_2^2 \tag{7}$$

The above has an explicit solution of the form :

$$w^* = (X^T X + \lambda^{'} I)X^T y \tag{8}$$

with $\lambda = \frac{\lambda^{'}}{2N}$.

C. Logistic Regression

The last method that was implemented were logistic regression and regularized logistic regression : Logistic regression loss has the form :

$$L(w) = \sum_{n=1}^{N} ln[1 + exp(x_n^T w)] - y_n x_n^T w \tag{9}$$

As the above formula doesn't have a close form, we used gradient descent as describe in (3). With the gradient given by

$$\nabla L(w) = X^T[\sigma(Xw) - y] \tag{10}$$

where $\sigma(x)$ is given by $\frac{\partial ln[1+exp(x)]}{\partial x}$ or $\frac{exp(x)}{1+exp(x)}$. The previous representation of $\sigma(x)$ isn't computationally stable and therefore we used $\frac{1}{1+exp(-x)}$.

IV. Model performance measurement and comparison

For each of the three implemented methods above, a cross validation was implemented in order for us to test locally for the best models. This was done using K fold on the training data and computing the average of each round validation accuracy. This metric was put in place in addition to $RMSE$ (Square root of Mean square error) to have the same metric as the online submission and for ease of local comparison.

The search for the best model hyper parameters was done using grid search. Hyper parameters research wasn't pushed to its finest has the project was meant as a first experience with Machine Learning and not a competition over best scores.

A. Least Squares

As Least Squares is the simplest model of the three, was therefore the first one tried. It gave reasonable results, around 80% online using only polynomial. Our setup described below meant that we didn't had to simplify computing complexity by using gradient descent and could use the close form directly.

B. Ridge Regression

Second method tried, and also the one which gave the best results. As Ridge regression can be equal to the previously discussed method, we chose to do most of our hyper parameter tuning and feature expansion research on this method. The training and testing parts were done on split data depending on the value of `PRI jet num` as explained above. The combination of angles expansion, drop of invalid values, polynomial basis and Gaussian distance was then applied to the data. These expansion allowed us to gain accuracy, 1% to 1.5% more than what is described above, depending if the data set is split or not. Without splitting the data set, mean replacement rather then dropping invalid values column gave better results.

C. Logistic Regression

The task of this project being a classification problem, Logistic Regression should be a good candidate and should provide better result then previous models. However, it turned out not to be computationally efficient and hard to tune in order to have results matching our least square baseline.

V. Setup

Most of the project was done using `Google Colab` as it allowed us to share the code instantly and without versioning issues (although it also meant that local version had to be made). It also allowed us to have access to powerful machine that let us try more complex model and close form version of the algorithm.

Then, we merged our work into a github repository to match the expected final code submission form. Unit tests are located in the $test$ folder, source code is in src, and utility scripts (for running, checking environment, downloading datasets) are in $scripts$. Details about how to run these scripts are in the root $README$ file.

VI. Result and Conclusion

This project allowed us to have a first experience on real world problem solving. Results found with relatively basic model were quite impressive (around 80%) and more complex model provided only marginal improvement on these.
We could only implement a limited number of methods given the five weeks duration allocated for this project. Some more feature selection could have been implemented through principal component analysis or similar methods. More kernel types (in the same way as what was done with Gaussian and angles) could have been implemented too, although again as the subject will be covered in later weeks (SVM), we'll have the opportunity to experience with them in the second project.

References

[1] CERN, *The Higgs boson*, Jul. 2012. [Online]. Available: https://home.cern/topics/higgs-boson

[2] *Higgs challenge*. [Online]. Available: https://higgsml.lal.in2p3.fr/files/2014/04/documentation_v1.8.pdf