CS-433 Machine Learning - Class Project 1 Report

Mortiniera Thevie, Petrescu Diana, Wagner Patrik

***Abstract*—The Higgs Boson is an elementary particle in the Standard Model of physics which explains why other particles have mass. Physicists at CERN smash protons into one another at high speed to generate smaller particles. It sometimes produces the mentioned above, Higgs Boson. It is difficult for researchers to measure the Higgs Boson directly (as it decays rapidly into other particles). Therefore, in this project, we want to predict if the event results from a Higgs Boson (signal) or some other process (background) given the observations' (decay signature). We applied several machine learning methods on a labeled data set in order to train the model. We then tested it on a new data set (provided in the corresponding Kaggle competition) and obtained an accuracy of 81.821%.**

I. Introduction

In 2012, Physicists at CERN managed to discover a new particle named Higgs Boson that could change particle physics.

To achieve this, they smash protons into one another at high speed to generate smaller particles, which sometimes produces the Higgs Boson. It is difficult for researchers to measure the Higgs Boson directly (as it decays rapidly into other particles). Therefore, we want to predict if the event results from a Higgs Boson (signal) or some other process (background) given the observations' (decay signature).

In fact, we were given two data sets containing the decay signature. One was labeled and we used it for training a model. It consisted of 250000 rows and 30 attributes. The other one was unlabeled and was meant for testing our model.

Our first step was to do some feature engineering to decide the relevant attributes in order to form the feature set and decide how to get the most out of the data.

We then tried several machine learning methods in order to find the best model parameters. We namely tested linear regression (using gradient descent and stochastic gradient descent), least squares regression, ridge regression, logistic regression and regularized logistic regression methods.

Finally, we chose the best model based on the accuracy obtained from the training data.

II. Exploratory data analysis and feature processing

A. Exploratory data

At first sight, when reading the description of the features [1] and looking at the training set, we noticed that some values were indicated as "undefined" (indicated as -999). We tried several approaches to handle these values.

As these undefined values represented more than 20% of the data set and were spread over various columns, we didn't want to just remove the corresponding lines or columns from the data set.

Hence, we first tried to replace these invalid values with different statistical metrics (calculated from the respective column without taking these undefined values into account). We namely tried to replace them by the mean, the median and the value with the highest frequency of the column. From all these metrics, we obtained the best results with the mean.

However, taking a closer look the the documentation along with the data, we noticed that many variables were depending on a variable named PRI_JET_NUM. Indeed, when this number was equal to 0, then all the jet related features were indicated as undefined. When it was equal to 1, the "leading" jet features were present but not the "sub-leading" ones. Finally, when it was equal to 2 or 3, no variable was undefined. Hence, we decided to split the data set into three different subsets depending on this number. Having these different subset, we could now drop the columns containing only undefined values. We also noticed that the last column was in fact the "sum" of the "jet" columns so we also decided to drop it. We then trained our models independently on each subset and obtain three different regression coefficients to use to make the predictions on the respective subset of the test data set. This significantly improved the accuracy.

B. Feature processing and augmentation

To improve further our accuracy we also did other feature engineering.

1) Feature transformation: We first standardized each column to the normal distribution $N(0, 1)$. We then applied the log transformation on strictly positive columns and stacked them to the original data set. The log transformation can be used to make highly skewed distributions less skewed. This can be valuable both for making patterns in the data more interpretable and for helping to meet the assumptions of inferential statistics [2], [3]. To handle negative values, we also tried to use translation [4]. Namely, if Y is a list of positive and negative values, then :

$$L(Y) = \log(1 + Y - min(Y))$$

We also tried another method and applied the inverse function of the hyperbolic sinus to be able to transform all data (positive and negative values) in the same manner.

2) Feature augmentation: We then stacked our original data set, cleaned and normalized, with the transformation mentioned above.

3) Feature selection: We tried to remove redundancy on the data in order to remove some noise. To do so, we found the columns with a very high correlation coefficient between the feature columns ($c > 0.95$), and we set $c_i = 0.9 * c_i$ and $c_j = 0.1 * c_j$. Moreover, some features were potentially more relevant for the predicted variable. Hence, we also computed the correlation coefficient for each pair $(Y, c_i), \forall c_i \in C$, with C the column list. Then, if $c < 0.8$, we gave a small weight to this column as it was less relevant for the predicted variable. Moreover, another way we tested to reduce the noise was by applying the principal component analysis method and keeping only a minimal dimension.

However, not all of these methods improved the score. Therefore, after testing them all independently and looking at the resulting increase/decrease of accuracy, we only kept the combination that gave the best results. Namely, we applied the log transformation only on positive columns and stacked the original subset to the subset containing log transformed values. This way we increased considerably the accuracy for all the machine learning methods.

III. Models and choice of implementations

Let X denote the predictor variables, the feature matrix created from the given data set, Y denote the dependant variable, the variable we want to predict, and w denote the vector containing the regression coefficients. We want to minimize ϵ, the noise or error, from the equation $Y = Xw + \epsilon$. In order to build our predictions, we tested six different machine learning techniques. We namely tested linear regression (using gradient descent (GD) and stochastic gradient descent (SGD), least squares regression, ridge regression, logistic regression and regularized logistic regression methods (using SGD).

For all the methods mentioned above, we had to find the best parameters. We begin the parameters search by running a grid search on them, and choosing the ones for which the root mean squared error was minimal. The parameters to find were the following ones: k_fold (used for the cross validation), degree (degree of the basis polynomial to augment X), max_iterations (number of step in the gradient descent for example), gamma: learning rate, lambda: (regularization factor for ridge regression and regularized logistic regression).

To test our model's ability to predict new data, to flag problems like over-fitting or selection bias, and to give an insight on how the model will generalize to an unknown data set, we used a k-fold cross validation method [5]. We split the training set into k subsets of same size. The first one was used as the test set and the remaining k-1 were used as training set and we repeated this process with different partitions of training and test subsets. To compute the regression coefficients (weights), we took the mean of the weights obtained for the different partitions during cross validation (except for the ridge regression for which we obtained the best parameters with the cross validation and then found the weights by running again the method on all the data).

IV. Results

The results found on each model were the following ones: Linear regression using GD (72.9196%), linear regression with SGD (72.8048%), least squares (73.6788%), ridge regression (81.6708%), logistic regression (80.7)%, regularized logistic regression (81.94)%. To obtain these results, we used a 5 / 6-fold cross validation and we tested them on the training data set as if we didn't have the predictions. We noticed that the results on Kaggle were similar to the ones found on the training data set.

As we can see with results above, the method which gave the best accuracy **(81.821%)** was regularized logistic regression using stochastic gradient and feature augmentation. The two principal parameters to test for this method were lambda (we tested 15 values between 10^{-6} and 1, and degree (we tested $2, 3, 4, 5$). The best parameters found were $(lambda, degree) = (0.0072, 2), (0.1389, 2), (0.1389, 2)$ (for respectively the subset were PRI_JET_NUM equaled $0, 1, [2, 3]$). Moreover, we used an adaptive gamma rate $gamma = 1/n\_iter$ and we noticed that $max\_iterations = 500$ was a very good parameter because increasing the number of iterations didn't result in a tangible change as the steps became very small. We tested k_fold 4,5,6,8,10 and 6 was the best parameter.

V. Conclusion

We managed to find an accuracy of **81.821%** on Kaggle using regularized logistic regression. Concerning the run time of this method, the parameter search was the part that took the most time due to grid search over all combination of parameters. The actual regression only took around 5 minutes, which is a good time.

When looking at our final results on the different Machine Learning techniques, we noticed that there was a gap on the accuracy performance between linear methods (Linear regression with GD or SGD and Least squares) in comparison to the others. However, if we had to choose one of this method, the least squares method took less run time than the two others and gave us a better accuracy.

Moreover ridge regression, logistic regression and regularized logistic regression gave us better results as we used linear basis function model to fit non linear data (as the data wasn't necessarily linear). Finally, the two best methods (which gave similar results) were ridge regression and regularized logistic regression. This could be explained by the fact that we avoid over-fitting by adding the regularizer.

REFERENCES

[1] A.-B. et al. Learning to discover: the higgs boson machine learning challenge. [Online]. Available: https://higgsml.lal.in2p3.fr/files/2014/04/documentation_v1.8.pdf

[2] D. M. Lane. Log transformations. [Online]. Available: http://onlinestatbook.com/2/transformations/log.html

[3] Wikipedia. Data transformation (statistics). [Online]. Available: https://en.wikipedia.org/wiki/Data_transformation_(statistics)#Common_case

[4] R. Wicklin. Log transformations: How to handle negative data values? [Online]. Available: https://blogs.sas.com/content/iml/2011/04/27/log-transformations-how-to-handle-negative-data-values.html

[5] Wikipedia. Cross-validation (statistics). [Online]. Available: https://en.wikipedia.org/wiki/Cross-validation_(statistics)