CLASSIFICATION VIA REGRESSION TECHNIQUES: A STUDY

Tilak Purohit (343083) *Xintong Kuang (328784)* *Manon Boissat (272022)*

Project–1, CS433- Machine Learning 2021, EPFL

ABSTRACT

Machine learning Algorithms can deal with complex, high dimensional data set and has become significant in various fields. In this project, we implement various regression based techniques to classify signal and noise in the Higgs Boson data set. We evaluated the data via regression based machine learning algorithms. We tested the performance of different models and the best performing models and its corresponding parameters were chosen for predicting the labels for the test data, we achieved 79% categorical accuracy in the aicrowd platform.

1. INTRODUCTION

The Higgs Boson is an elementary particle in the Standard model of physics which decay rapidly. We want to explore the potential of some machine learning algorithms which would be able to make a good recognition of the preferred particle. The main aim for our work is to process and clean the raw data and then to apply regression models and to construct accurate classifier to generate prediction. we attempt feature engineering that could improve prediction results. A 4-fold cross validation was used to optimize the hyper-parameter of our best model.

2. METHODOLOGY

In this sections we elaborate on our pre-processing techniques and the observations we made from the data. Secondly, we discuss on the regression techniques we implemented and summarize our results. Third, we talk about the feature augmentation and its necessity.

2.1. Data pre-processing

The measurements provided in the Higgs-Boson corpus have some prominent errors. The erroneous data could tamper with the experiments and can potentially lead to a lousy machine learning model. We came up with some observations and method to clean the dataset. We provide our observation and pre-processing methods below:

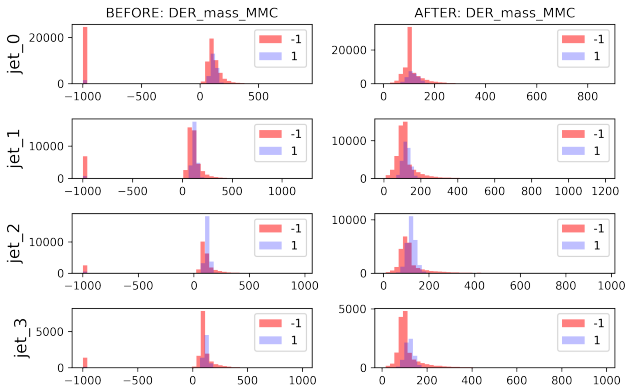
1. Categorical feature `PRI_jet_num`: On observing the histogram plots for each of the columns (feature), we noted a discrete features with values limited to 0,1,2 & 3. Further investigation revealed, jet's to be the pseudo particles which originates from high energy quark or gluon. While the other features are some form of measurements which are related to the jet's, but `PRI_jet_num` denotes the number of particles which appeared in the detector. We hypothesise a direct relation of these discrete features to the measurements provided in the form of remaining features, therefore we split the raw data into four sets corresponding to the jet number i.e, 0,1,2,&3. The distribution of the data could be noted from table 1.

Table 1: Distribution of the raw data, and the 4 subsets jet_(0,1,2,3) derived from the raw data.

	# Samples Label = 1	# Samples Label = -1
raw-data (tX)	85667	164333
jet_0	25492	74421
jet_1	27710	49834
jet_2	25734	24645
jet_3	6731	15433

2. The zero-variance features: The zero variance feature have almost no predictive capability and in some cases lead to numerical crashes. After dividing the data into four set it was observed that set jet_0 has 11 such columns and set jet_1 has 7 such columns where the variance is zero, where as set jet_2 and set jet_3 have no zero variance columns. To avoid any computational crash these features were removed from the respective sets, anyway zero variance features have no predictive power.
3. The -999 values: After the zero-variance features were removed from the four sets, it was observed that feature `DER_mass_MMC` had multiple -999 values which can be observed from figure 1, these could be the error values. We replace these values with the median of the remaining values to capture a less erroneous distribution. After step (2) & (3) no other column had -999 value. Since we did not had a prior knowledge of the data we did not attempt to remove any other outlier other than -999 from the data.

Fig. 1: Distribution of `DER_mass_MMC` with the -999 values (left) and after pre-processing (right)



Click here for supplementary material

4. Mean and variance normalization: This is an efficient statistical method to reduce the noise from the measurement features. We normalize all the features from the four sets as a final step before training the ML models. The normalization expression is as following:

$$z = (f - \mu_f)/(\sigma_f + \epsilon)$$

where, μ_f is the mean, and σ_f is the standard deviation for the feature (f).

2.2. Regression Models

Six regression methods were implemented from scratch and were evaluated using the given corpus. All the four sets derived from the given data were split into train and the test set in the ratio of 8:2 respectively, with the $seed = 1$. All the six models were separately trained using the train set and the weights learned by the respective models were used for predicting the classes in the test set and evaluating the models. The classification results from the four sets were combined and reported for the performance check. Hyper-parameters were optimized by repeated experiments. The model performance with the fine-tuned hyper-parameters are summarized in table 2.

Table 2: Model Performance for the test set in the provided dataset. γ denotes the step size, λ denotes the regularization-term, $iter.$ shows the maximum number iterations set for the parameter learning

Method	γ	λ	iter. (#)	Accuracy (%)
Gradient Descent (GD)	0.07	-	100	**70.22**
Stochastic GD (bs.=1)	0.002	-	200	66.96
Least Squares	-	-	-	**70.23**
Ridge Regression	-	10^{-7}	-	**70.36**
Logistic Regression (LR)	10^{-6}	-	100	69.18
Regularized LR	10^{-6}	0.1	100	68.98

From table 2, it can be observed that methods like GD, Least squares and ridge-regression gave above 70% of accuracy. Models like LR and Regularized-LR were not too far behind from the 70% accuracy mark, this may be due to the fact that hyper-parameters could be better optimized. Stochastic GD gave the least accuracy among all the six models.

2.3. Feature augmentation and optimization

Since, ridge regression gave the best accuracy in our case - 70.36%, we choose this model for further evaluation.

Ridge regression model can be simply defined as:

$$\min_w L(w) + \Omega(w)$$

where, L is a loss function and $\Omega(w)$ is the penalizing term, defined as below:

$$L(w) = \frac{1}{2N}\sum_{n=1}^{N}[y_n - y']$$

$$\Omega(w) = \lambda||w||_2^2$$

To further evaluate and enhance the representation power of our model, we augmented our features using the polynomial basis of degree d, this could be expressed in the following manner:

$$\phi(x_n) = [1, x_n, x_n^2, ..., x_n^d]$$

and now the prediction could be expressed as, $y' = \phi(x_n)^T w$. To optimize the hyper-parameter (λ) we resorted to a grid search and for each value of the search we ran a 4 fold cross validation to prevent overfitting. The optimal (λ) was picked on the iteration were the average of the root mean squared error (rmse) for the test set is minimum. The weights for 4 folds, obtained from the optimal iteration were averaged and used for the final classification. We saw a boost in prediction accuracy for ridge regression on the local test set, without data-augmentation overall accuracy was 70.36% and with augmentation + parameter tuning we got overall accuracy of 79.44%. Jet wise results are summarized in table 3.

Table 3: Optimal hyper-parameters for ridge regression and the prediction results corresponding to `PRI_jet_num` for local test set.

	jet_0	jet_1	jet_2	jet_3
degree (d)	6	6	6	6
λ	$1.0e+00$	$1.0e-10$	$6.2e-07$	$1.9e-02$
Accu. (%)	81.48	78.39	81.09	70.31

3. RESULTS AND DISCUSSION

For the final test data evaluation, the data was pre-processed and augmented $(d = 6)$ as defined in the above sections. The optimal weights obtained from ridge regression were considered for the prediction of the test data. On our local test data the model gave an accuracy of 79.44%. For the given test data we obtained an accuracy of 79% with an f1-score of 68% on the aicrowd platform. It can be observed that our trained model did not deviated much from results it delivered on the local test-train set. This shows the models were trained in a robust manner but, the feature processing & engineering part could have been better optimized to come closer to the optimal data distribution for a better performance.

4. CONCLUSION

In this work we implemented and studied five different regression techniques and least square method. We performed data augmentation and observed its importance. We also learnt the gravity of hyper-parameter tuning. The study make us realize the importance of domain knowledge, specially for feature engineering/pre-processing which is one of the main aspect of the field. The main emphasis of our study was to better understand the model behaviour with different settings. Our approach may not be very optimal but helped us realize the techniques better.

The complete work was done in python using only two libraries namely - numpy and matplotlib.