Project 1:Higgs Boson Machine Learning Challenge

Cho Hyun Jii, Poopalasingam Kirusanth, Rodriguez Natalia

***Abstract*—In the following project our goal was to build a machine learning classifer to detect the Higgs Boson, as a part of the Higgs Boson Machine Learning Challenge. In the Standard Model of particle physics, the Higgs Boson is responsible for the mass of other elementary particles. We provide our process of feature engineering and experiment with different models to achieve the best classification of the Higgs Boson signal and background (uninteresting events). The best results were obtained using ridge regression method and this was evaluated using the *F1 score*.**

I. Introduction

The data set provided by CERN consists of a training set of 250,000 labeled events, either signal or background, and a test set of 568,238 unlabeled events. The events are described by 30 features which can be engineered and used to train models for particle prediction. The features consist of the physical properties of the events. In the following sections we will explain the manipulation we did to the data, and the different methods we used to analyze the data.

II. Feature Engineering

On first glance of the data, we noticed that many features were not on the same scale (Table 1) and there were many -999 values which represent undefined values. These values are outside the normal range and given the fact that this could represent a case in which you did not have a signal associating a value to this variables would create an even noisier data set. Advanced classifiers may be able to determine that these values should be ignored; however, simple models, such as least squares linear regression would be highly influenced, as they are sensitive to outliers. The first challenge was to determine how to deal with these undefined values. A few ways to handles these values would be:

1) Simply remove the events with any undefined values
2) Remove the features with any undefined values
3) Replace the undefined values with a specific value.

There were 181,886 events with -999 values and 11 sparse features, so it was unreasonable to remove any events or features because our data set would be drastically reduced. Thus, we decided to manipulate the features and then replace any remaining undefined values with 0.

Upon careful inspection of the data set, we noticed that the *PRI_jet_num* feature was the only categorical feature in our data. Depending on the value of this feature, certain other features are undefined, as signified by the value -999.

For our feature engineering, we partitioned the dataset based on the *PRI_jet_num* feature and dropped the corresponding undefined features (Table 2). There were only a handful of undefined values in *DER_mass_MMC*; however, we did not drop this feature because mass is an important property to consider when classifying particles. Finally, in both cases, we scaled the features by standardizing the values of each feature in the data to have zero mean and a standard deviation of 1. By standardizing the data set we remove possible outliers. We standardized the features with respect to the training set as following:

$$x_{tr} = \frac{x_{tr} - \mu_{x_{tr}}}{\sigma_{x_{tr}}} \tag{1}$$

	5	6
count	72543	72543
mean	2.403735	371.783360
std	1.742226	397.699325
min	0.00000	13.602000
max	8.503004	974.979000
25%	0.882500	111.977000
50%	2.107000	225.885000
75%	3.69000	478.226000

Table 1. Statistical analysis of few features.

PRI_jet_num	*Undefined features + PRI_jet_num (feature indices)*
0	[4, 5, 6, 12, 22, 23, 24, 25, 26, 27, 28, 29]
1	[4, 5, 6, 12, 22, 26, 27, 28]
2	[22]
3	[22]

Table 2. Features dropped based on *PRI_jet_num* value.

III. Prediction methods

We performed an exploratory study using the following models:

- *Least Squares*: This method aims to minimize the square of the residuals, i.e. the difference between the observations and the model predictions.
- *Least Squares using GD and SGD*: These methods also aim to minimize the square residuals but they incorporate a corrections of the weights used in the predictive model using the gradient descent and the stochastic gradient descent respectively.
- *Logistic Regression and Regularized Logistic Regression*: Logistic regression with gradient descent models a dependent binary variable using a logistic function that transforms predictions in $\mathbb{R}$ into a true probability. Regularized logistic regression incorporates a term proportional to the λ parameter.
- *Ridge Regression*: Ridge regression is a regularization method that allows to correct the error introduced by collinearity of the data set that would be incurred by only using Least Squares. We used the standard Euclidean

norm and the λ parameter to introduce the regularizer term.

IV. Evaluation Metric

In order to evaluate the performance of our model on our data and to fine-tune our parameters, we use 10-fold cross validation. In our cross validation, we split the data into train and test subsets. These train and test subsets are partitioned by the *PRI_jet_num* feature, as described above. Briefly, after partitioning by the value of *PRI_jet_num*, we drop any corresponding features (Table 2). Because we partition the data, we then train the model and compute a different set of weights for each partition. Finally, we make the predictions on the partitions of the test set with corresponding weights. The predictions are then compared to the actual labels. We determine the performance by comparing F1-scores.

The *f1 score* is a binary accuracy test and it is given by[1]:

$$f1\ score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2}$$

Considering correctly predicted positives and negative values (*tp*) and (*tn*) and incorrectly predicted positive and negative values (*fp*) and (*fn*), precision and recall are given by:

$$precision = \frac{tp}{tp + fp} \tag{3}$$

$$recall = \frac{tp}{tp + fn} \tag{4}$$

V. Results

To obtain good accuracy of our predictions, we consider both the feature engineering and prediction method. As mentioned above, we partitioned the data based on the *PRI_jet_num*. We evaluated the different prediction methods and show the performance of logistic regression, least squares, and ridge regression in Figure 1. As we can see in figure 1 the best results were obtained with ridge regression, and therefore, we optimized the hyperparameters for this model and continued with this method for our final predictions of the unlabeled test set. Following this initial study we determined the degree of the polynomial expansion based on the *f1 score*.

We studied different configurations for the polynomial base representation, we studied representations from degree 1 to 20 and obtained the best results for degree 14 as can be seen in figure 2.

To determine the most appropriate value for lambda we fixed the degree of the polynomial expansion and ran a grid search. We found the highest *f1 score* for lambda equal to $1\text{x}10^{-15}$, the obtained result are presented in figure 3.

VI. Discussion

Although we obtained a good result in the Kaggle competition leader board (82.9% accuracy), we could explore more ways to improve the result. One way is to try if we can reduce the feature set using Principal component analysis (PCA) or try to get a better classification result with Support Vector Machines.

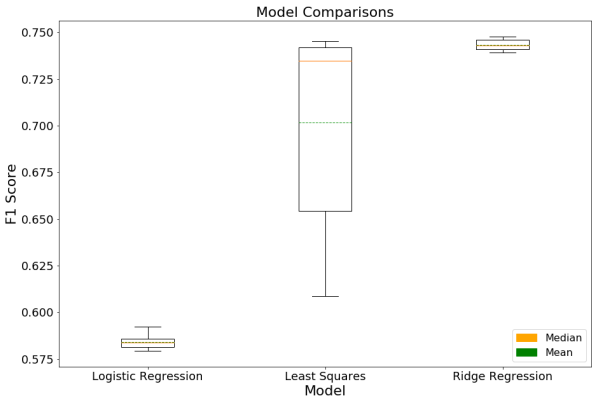

Fig. 1: Model comparisons

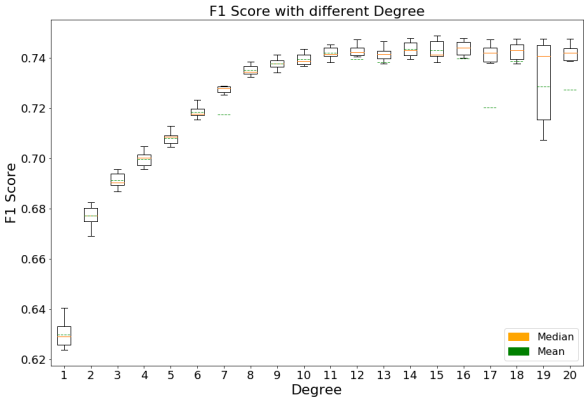

Fig. 2: Model predictions for different degrees

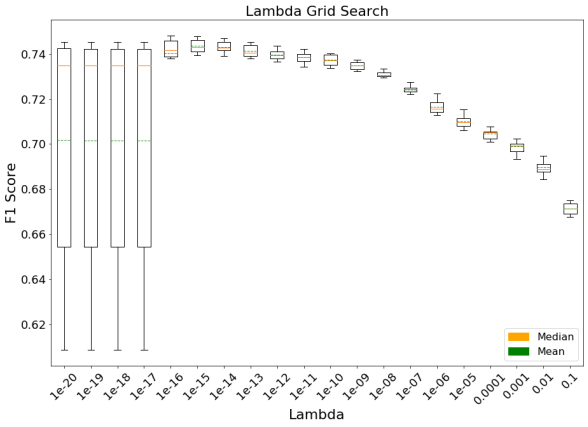

Fig. 3: Model predictions for different lambda

References

[1] Y. Sasaki, "The truth of the f-measure," 2007. [Online]. Available: https://www.researchgate.net/publication/268185911_The_truth_of_the_F-measure