Finding Higgs boson with Machine Learning Models

Zeming Chen, Simin Fan, Soyoung Oh

***Abstract*—This project aims to use machine learning models to detect a true *Higgs boson* event signal. We preprocessed the raw signal data for training the machine learning models. Then, validated the trained models by cross-validation with grid search for hyperparameter tuning. For the performance improvement, we further conducted polynomial feature augmentation. We achieved a test accuracy of 78.4% and F1-score of 68.1% with the improved ridge regression.**

I. Introduction

In this project, we analyzed the given training data to identify potential problems which could deteriorate the model performance with binary classification task. Based on the findings from the analysis, we preprocessed the training data and trained the six different machine learning regression models to validate the trained models by cross-validation with hyperparameter tuning using grid search. Based on the baseline results, we selected the best performance model, regularized logistic regression, to analyze the effect of outlier filtering. For the performance improvement, we conducted polynomial feature augmentation with varying degree. For the final evaluation, we tested the baseline models and improved models on the AICrowd platform.

II. Dataset analysis

The training data consists of N = 250,000 events and d = 30 features of 1 categorical variable and 29 continuous variables. The test data consists of N = 568,238 events. Each event is labeled by a binary variable indicating either the background (label: b) or Higgs boson (label: s) signals. We pre-processed the training data in the following steps.

III. Feature Preprocessing

Our feature engineering includes the following steps: data imputation, one-hot encoding of the categorical feature (`PRI_jet_num`), and data normalization. For improving the performance of the best baseline model, we further conducted outlier filtering and polynomial feature augmentation.

Data imputation. The missing values (i.e., -999.0) were filled with four different values: feature's mean or median, each label feature's mean or median. We finalized to use median of features for the replacement since it is robust to outliers.

One-hot encoding. Each of the integer values in the categorical variable, `PRI_jet_num`, was replaced by binary representations for the four categories. That is, the value of {0,1,2,3} is encoded as {[0,0,0,1], [0,0,1,0], [0,1,0,0], [1,0,0,0]}, respectively. As the sum of the encoded features across all the data points yields a vector of ones, the offset (i.e., bias term) is not used.

Data normalization. For categorical value, we conducted data normalization by using standardization scaling where all the features are subtracted by the mean $\mu_d = \frac{1}{N}\sum_{i=1}^{N} x_{i,d}$, and then divided by the standard deviation $\sigma_d = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{i,d} - \mu_d)^2}$.

Outlier filtering. For the statistical robustness, we filtered out outliers whose feature value $x_{i,d}$ falls outside the range of $[\mu_d - \eta\sigma_d, \mu_d + \eta\sigma_d]$, where we set $\eta = 10$ as an outlier factor. After the filtering process, the number of 1,089 samples are deleted as outliers from training data which is 0.43% of the total samples.

IV. Experiments and Analysis

A. Baseline Models

For baseline models, we implemented 6 different machine learning algorithms, including Mean Square Error with gradient descent (`mse_gd`), Mean Square Error with stochastic gradient descent (`mse_sgd`), Least Square (`least_square`), Ridge Regression (`ridge`), Logistic Regression with stochastic gradient descent (`logistic`), and Regularized Logistic Regression with stochastic gradient descent (`reg_logistic`). For each model, we conducted hyperparameter tuning using grid search. The optimal hyperparameters are listed in Table I.

Model	Hyperparameter	Value
`mse_gd`	Learning rate (γ)	0.1
`mse_sgd`	Learning rate (γ)	5e-3
`ridge`	Regularization weight (λ)	1e-6
`logistic`	Learning rate (γ)	0.01
`reg_logistic`	Learning rate (γ)	0.01
	Regularization weight (λ)	1e-4

Table I
Optimal hyperparameters from grid search

Model	Accuracy ↑	Precision ↑	Recall ↑	F1 ↑
`mse_gd`	73.0	63.0	51.6	56.7
`mse_sgd`	74.3	65.8	51.9	58.0
`least_square`	**74.8**	**67.9**	50.1	57.6
`ridge`	74.6	67.8	49.5	57.2
`logistic`	**74.8**	66.0	54.5	**59.7**
`reg_logistic`	**74.8**	66.0	**54.6**	**59.7**

Table II
Validation Performance of Baseline Models

We performed 5-fold cross-validation on the training data and recorded model checkpoints in each fold by monitoring the model's validation accuracy. For (stochastic) gradient descent, we ran training epochs until validation accuracy does not significantly improve which results in 200 for `gd` and 10 for `sgd`. We report the baseline performances of the models from cross-validation in Table II. As the table shows, regularized logistic model outperforms the other baseline models with accuracy, recall, and F1-score (74.8%, 54.6%, and 59.7%). While the least square shows the best performance with a precision metric of 67.9%. Therefore, we decided to use regularized logistic regression model for further improvement.

B. *Influence of outliers*

As logistic regression models are based on all of the data points, removing the outliers will change the decision boundary around the test points of the logistic classifiers. To analyze the effect of outliers in the training process with the regularized logistic regression, we conducted ablation experiments by varying the outlier factor (η). As in Table III, the outlier filtering drops the precision of the model 1.1–2.3% point. However, when it comes to the other metrics, loss (-0.4), accuracy (+2.4), recall (+1.4), and F1-score (+0.2), the trained model with the outlier filtering shows improvements. The results imply the importance of the outlier filtering process using the regularized logistic regression model with classifying high-dimensional feature data.

η	**loss ↓**	**Accuracy ↑**	**Precision ↑**	**Recall ↑**	**F1 ↑**
-	0.50	75.3	**67.1**	55.0	60.3
3	**0.46**	**77.7**	64.8	53.7	58.7
7	0.50	74.9	65.3	**56.4**	**60.5**
10	0.50	74.8	66.0	54.6	59.7

Table III
RESULTS OF OUTLIER FILTERING WITH VARYING FACTOR

C. *Polynomial Feature Augmentation*

Considering non-linearity relations between features in given data, such as mass, angles, and kinetic energy, we expanded each feature with a polynomial transformation. In polynomial logistic regression, the polynomial order has a certain influence on the regression performance [1]. That is, if the polynomial frequency is too high, the over-fitting will occur. To find the optimal polynomial degree, we conducted experiments with regularized logistic regression by varying the degree of polynomial features (p = 1 to 6). We used the same hyperparameters as in Table I, and the outlier filtering factor as η = 10. As in Table IV, feature augmentation with a polynomial degree of 3 significantly improves the baseline (p = 1) performance. The result implies that adding higher order of features to the original linear features could increase the complexity of the model.

In addition to the varying performances across different polynomial factors, the regularized logistic regression is also sensitive to hyperparameter (λ) as in Figure 1. Therefore, it is important to consider polynomial factor with hyperparameter of the model at the same time.

p	**loss ↓**	**Accuracy ↑**	**Precision ↑**	**Recall ↑**	**F1 ↑**
1	0.50	74.2	65.6	52.3	58.2
2	0.48	76.5	67.8	59.0	63.1
3	**0.47**	**77.4**	**70.3**	57.6	63.3
4	0.49	76.5	67.3	57.9	62.2
5	0.48	76.8	66.5	**61.1**	**63.7**
6	0.53	74.8	63.2	56.1	59.4

Table IV
RESULTS OF POLYNOMIAL FEATURE AUGMENTATION

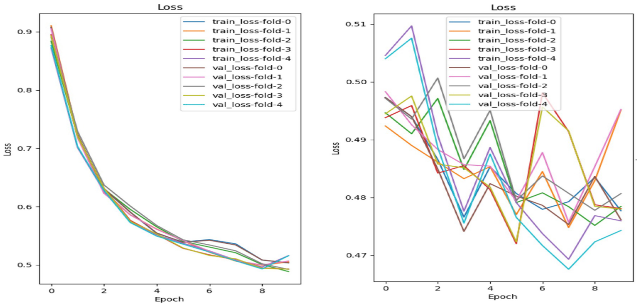

Figure 1. Training and validation loss across 5-fold cross validation with regularized logistic regression of $p = 4$, (left): $\lambda = 1e{-5}$ (right): $\lambda = 1e{-6}$

V. Test Set Evaluation

We trained each model using 5-fold cross-validation and evaluated them on the test set. We implemented horizontal voting to decide the final prediction based on the majority across the five model predictions from cross-validation. The final results from AICrowd are described in Table V. The polynomial feature augmentation improves the performances across different models. The ridge regression with polynomial degree 6 scored the highest test accuracy of 78.4%, while the logistic regression with a polynomial degree of 4 achieved the highest F1-score with 68.9%.

Model	**Accuracy ↑**	**F1 ↑**
`mse_gd`	73.1	56.8
`mse_sgd`	74.2	58.0
`least_square`	75.1	59.4
`ridge`	74.8	57.7
`logistic`	75.0	59.5
`reg_logistic`	75.0	59.5
`ridge_improved` (p=6)	**78.4**	68.1
`logistic_improved` (p=4)	77.5	**68.9**
`reg_logistic_improved` (p=3)	76.7	61.2

Table V
TEST PERFORMANCE OF THE MODELS

VI. Conclusion

Our work shows the importance of feature engineering and hyperparameter tuning process in regression models for a classification task. In detail, filtering outliers and augmenting high-order features significantly affect the performance of logistic models. Therefore, it is important to not only tuning the model hyperparameters, but also the parameters of preprocessing.

References

[1] X. Wan, “The influence of polynomial order in logistic regression on decision boundary,” in *IOP Conference Series: Earth and Environmental Science*, vol. 267, no. 4. IOP Publishing, 2019, p. 042077.