# Predicting Higgs Boson Signals from Collision Events with Linear Models

Rahul Rajesh, Zhuang Xinjie, Chitrangna Bhatt
*School of Computer and Communication Sciences, EPFL, Switzerland*

***Abstract*—A critical part of scientific discovery is the communication of research findings to peers or the general public. Mastery of the process of scientific communication improves the visibility and impact of research. While this guide is a necessary tool for learning how to write in a manner suitable for publication at a scientific venue, it is by no means sufficient, on its own, to make its reader an accomplished writer. This guide should be a starting point for further development of writing skills.**

## I. Introduction

The Higgs Boson is an elementary particle in the Standard Model of Physics which explains why other particles have mass. In this paper, the aim is to predict whether a particle collision event was one from the Higg's boson or from some other process/particle given a vector of features representing the decay signatures of the event. These predictions will be done using linear models and some preprocessing of the input data. The next few sections outline the implementation details.

## II. Models and Methods

This is a binary classification problem where a label has to outputted to indicate that the collision event is from the Higg's Boson or not. In order to achieve optimal results using the models, some work needs to be done to process the feature set that is given.

### A. Data Exploration

The dataset provided consists of 250,000 events with 30 feature columns. The features consisted of raw quantities and derived data computer from those raw quantities. All variables are floating point integers except for one variable *PRI_jet_num* which is an integer with value of 0,1,2,3. There are undefined values in the dataset indicated as -999.

### B. Data Processing

1) *Label Encoding*: In order for the model to predict the output, the "Higgs Boson" or "Backgroud" events have to be encoded as a number. For logistic regression, the encoding was chosen to be [0, 1] while for linear regression the encoding was [-1, 1].

2) *Imputation*: The -999 values among the variables would skew the predictions the model can produce. A good strategy is to replace the -999 with either the mean/median/mode of the column. All three were tested and gave similar results. Mode was used as replacement in the final model.

3) *Log Transformation*: Some of the variables had some right skew. These variables may cause the model to assume a relationship between the spread of the variable and the label produced. In order to reduce this from affecting the final results, an inverse log transformation was applied to all variables that had positive values. Experimentation showed better results after this transformation was performed.

$$x_{*,j} = log(1/1 + x_{*,j})$$

4) *Polynomial Basis*: A simple linear model can only draw a straight line between the points. This would not give good results when the data has a non-linear decision boundary. In order to work around this, feature expansion was done using a polynomial basis function. Powers of the feature matrix was taken and concatenated together.

5) *Standardization*: Since polynomial terms are used in the feature matrix, standardization needs to be done to reduce the multicollinearity in the data. This would help our linear models to not produce misleading results. Standardization was done on the feature matrix by subtracting the mean and dividing by the standard deviation.

### C. Choice of Model / Strategy

The linear models chosen to tackle this problem were logistic regression and ridge regression with L2 regularization. For Ridge Regression, the closed form normal equation was employed while gradient descent was used for logistic regression. The models were run separately and the one that had the best accuracy was taken to be the final model.

Additional methods like grouping the features based on *PRI_jet_num* were considered and running individual models on them were considered. However, they did not yield better results due to the reduction in the feature space when training each model.

*Tuning Hyperparameters*: The Models used had a number of hyperparameters, namely the *lambda* value for regularization, the degree used for feature expansion and the *step-size/max-iterations* for gradient descent. A grid search was carried out to find out the best value to be used for each of the hyperparameters. A 10-fold Cross Validation was used to verify the train/test accuracy and choose the optimal hyperparameter value.

Table I below summarizes the optimal values that were found:

| | Ridge Regression | Logistic Regression |
|---|---|---|
| Lambda | cell5 | cell6 |
| Degree | cell8 | cell9 |
| Step-Size | cell8 | cell9 |
| Max Iterations | cell8 | cell9 |

Table I: Results of hyperparameter tuning

## III. Results and Discussion

The 2 tables below showcase the train or test accuracy (again using a 10-fold cross-validation) before and after feature processing. Note, a cutoff value of 0 was used for linear regression (y values in [-1,1]) while a cutoff of 0.5 was used for logistic regression (y values in [0, 1]).

| | Train Accuracy | Test Accuracy |
|---|---|---|
| Ridge Regression | cell5 | cell6 |
| Logistic Regression | cell8 | cell9 |

Table II: Before Feature Processing

| | Train Accuracy | Test Accuracy |
|---|---|---|
| Ridge Regression | cell5 | cell6 |
| Logistic Regression | cell8 | cell9 |

Table III: After Feature Processing

It is clear that feature processing has a tremendous effect on the performance of the models. The justification for this is also outline in section II above. Both models were submitted to the online platform AICrowd to validate on the test set and the accuracies/F1-Score obtained are show below:

| | Accuracy | F1-Score |
|---|---|---|
| Ridge Regression | cell5 | cell6 |
| Logistic Regression | cell8 | cell9 |

Table IV: Leaderboard Submission

Between Ridge Regression and Logistic Regression, Logistic Regression performs better overall. This is expected behaviour considering that this is a classification problem with y values [0, 1]. A ridge regression linear model would exaggerate the cost and unreliably predict the results based on that. A really high degree was required to get a reliable prediction from the ridge regression model (possibly leading to overfitting). Logistic Regression wraps the predictions in a sigmoid function to get it between the required range of [0, 1]. This provides a more accurate measure from the cost function.

However, the results obtained are not perfect and there more improvements that could be done to improve the accuracy obtained.

- *Grid Search for hyperparameters*: Due to resource constraints, the grid search done for hyperparameter tuning did not cover all the possible values. There is a possiblity that there is a better combination of parameters that yield better results.
- *Correlation Matrix*: It is also good to plot a correlation matrix between features and drop features that are highly correlated to each other
- *Ensemble Techniques*: Even though this paper focuses on Linear Models, the accuracy for classification problems like this could be improved by using a combination of different models (e.g. Support Vector Machines / Decision Trees) and using voting/averaging the results [1]

## IV. Summary

The paper proves that simple Linear Models can be used to solve rather complex problems like the Higgs Boson Prediction problem. A logistic regression model achieved a 81% accuracy after some processing of the feature matrix was done. The paper highlights the importance of feature processing and hyperparameter tuning in the final performance of a Linear Model. Further strategies to improve the accuracy of our model such as ensemble techniques were also considered.

## Acknowledgements

The authors thank our instructors Martin Jaggi and Rüdiger Urbanke for their guidance on the project.

## References

[1] A. Hunt and D. Thomas, *The Pragmatic Programmer*. Addison Wesley, 1999.