CS433 - Machine Learning Project 1

Luca Bracone Omid Karimi Gianni Lodetti

***Abstract*—Given a dataset of simulated collision events in a large hadron collider, we utilise various classification and regression methods to estimate whether an event has produced a Higgs boson or not. Our best estimate came from using logistic regression and gradient descent on an augmented feature space. After about 15'000 iterations we obtain approximately 83% accuracy on `aicrowd.com`.**

I. Introduction

The Higgs boson is an elementary particle in the Standard Model of physics which explains why other particles have mass. Its discovery at the Large Hadron Collider at CERN was announced in March 2013. In this project, we apply machine learning techniques to actual CERN particle accelerator data to recreate the process of "discovering" the Higgs particle. For some background, physicists at CERN smash protons into one another at high speeds to generate even smaller particles as by-products of the collisions. Rarely, these collisions can produce a Higgs boson. The goal of this project is to use the provided dataset of decay signatures, and machine learning techniques to build a binary classification model that can determine wether a given signature is signal (a Higgs boson) or background (something else). We will explore and clean the provided dataset, look at the features, attempt to expand the features and apply different classification techniques.

II. Data pre-processing

A. Preliminary processing

We start by adding an intercept term (a column of ones) to the design matrix, this is common practice, and it allows the predictor to have a default value in case all the features are at zero. It also makes the predictor much more accurate in the case of linear regression since the fitted hyper-plane does not necessarily have to go through zero now. We also choose to standardize the observations column-wise. This makes the observations easier to compare to each other.

B. Dealing with missing values

The training set contains 250000 events. Each row has an ID, a label, and finally 30 feature columns. We noticed that eleven feature columns in particular contained many times the value "-999", this value represents values that are meaningless or not computable. We have tried two methods to solve this problem. The first one was to replace each missing value by the median of the column it found itself in. Though this method may seem like an arbitrary heuristic, it is the one that provided the best results in our tests, and is the one we use in the final submission. The second method was motivated by observing that whether a column had missing value or not, was determined by a categorical feature. Namely, it was the column `PRI_jet_num` who determined if many other features had a missing value or not. Given that, we tried to separate the dataset in three sub-datasets based on the value of `PRI_jet_num`, and within each sub-dataset we removed the columns that had missing values. This generates three sets of weights and during the testing phase we would use the appropriate set to provide a prediction.

C. Adding new features

To augment the feature space of our predictor we include new variables in the design matrix. To take into account the observation that some features had exponential behaviors aswell as the relationship between variables, we add new features containing the pairwise products of all current features aswell as 30 new colummns containing the $\log$ of the original 30 features.

III. Model selection and results

To test our models we simply split the data into a test and train set with 20% and 80% of the samples respectively.

TABLE I
Accuracy results with different models before feature expansion

Model	**Accuracy[%]**
Least squares	74.5
Least Squares GD	74.5
Least Squares SGD	73.2
Ridge Regression	74.50
Logistic Regression	74.77
Reg_Logistic Regression	75.07

TABLE II
Accuracy results with different models after the feature expansion, with 100 iterations(for the models that use iterations)

Model	**Accuracy[%]**
Least Squares	–
Least Squares GD	66.17
Least Sqaures SGD	66.39
Ridge Regression	82.12
Logistic Regression	80.60
Reg_Logistic Regression	80.60

The fact that ordinary least squares estimator after feature expansion does not compute at all may be surprising. A possible explanation is that when trying to compute the ordinary least squares estimator $\hat{\beta} = X(X^TX)^{-1}y$, having a large number of features (about 550) causes some columns to be approximately collinear. Sometimes it is a design problem, but in this case we believe it is due by sheer chance. This causes the determinant of X^TX to be extremely small and the inversion becomes numerically unstable and even if it gets computed at all the resulting estimator has extremely high variance, enough even to flip the sign of some coordinates $\hat{\beta}_i$. Ridge regression was invented in part to address this issue, and we see it perform significantly better than least squares gradient descent methods. The fact that these methods perform worse after feature expansion may be explained by the fact that adding so many features causes the gradient descent to get stuck on local minimas that are worse than before. This could be caused by the loss space becoming more "chaotic" after adding so many features. Ridge regression perfoms well even after the feature expansion it perfoms even better than logistic regression with 100 iterations, however logistic regression outperforms ridge regression if we increase the number of iterations, given this we have decided that our final model will use logistic regression with many iterations. In our final model we ran regularized logistic regression for 15000 iterations and obtains an accuracy of 0.831 in our local test aswell as on AICrowd submission.

IV. Discussion

Although logistic regression did perform better in the end, we were surprised by how well ridge regression performed for the binary classification, since linear regression is not known for being the best options for classification given that it uses MSE loss and that its predicted values are continuous and not probabilistic. We also observed that feature expansion makes a huge difference in the perfomance of our model, this indicates that some features are closely related and a better interpreted together, however we did not take the time to look into these relationships more closely and perhaps that would allow us to build a better model by allowing us to remove some of the many columns we extended our data with. One final observation is the fact that the regularized logistic regression and standard logistc regression perfom just as well. We believe this is because we have a large dataset that helps us avoid overfitting, this is backed by the fact that our training accuracy is near identical to our accuracy error and the feedback accuracy from our submission on AICrowd.

V. Conclusion

In this project we did not have to look much into the details of the dataset. Taking the median over columns to replace nan values and extending our features to include the cross product of features aswell as their log's was enough to obtain a large dataset, that we could apply one of the proposed models to. After testing out several models, logistic regression with 15000 iterations proved to be the clear winner, which is not surprising given linear regression is not known to be the best choice for classification, however ridge regression was much faster to compute and not far behind in its predictions accuracy. In the end this approach allowed us to achieve a relatively good result(compared to the other results in the AIcrowd project leaderboard), although we believe there remains space for improvement. We did not spend too much time trying to understand exactly what each feature represents, and it might of been interresting to further explore these relationships to determine which pairs of features or other combinations of features are best evaluated together. Understanding the features more intimately, might also of helped us clean up the nan values in the dataset with values more relevant then just taking the median. Finally since we did not have any overfitting problems with our current model, perhaps this means there is space to improve by selecting a more complex model, like for a example a neural network with several layers.