CS-433 Machine Learning
Project 1

Sara Pagnamenta, Etienne Salimbeni, Luka Chinchaladze

***Abstract*—In this paper we propose a Machine Learning approach to predict the presence of Higgs boson after a collision using CERN's Higgs Boson data set (Kaggle Higgs Boson Challenge). The main focus is on data pre-processing, which includes replacing missing values and outliers, data augmentation and, more importantly, feature selection and expansion. Both least squares gradient descent and logistic regression, together with a combination of the aforementioned techniques, have been tested all along and eventually compared. In the end, the probabilistic nature of the logistic regression resulted more appropriate for this binary classification's problem. By combining missing values replacement, feature expansion and logistic regression, we show that the model can be trained to reach an accuracy of $\sim 82\%$.**

I. Introduction

CERN's Higgs Boson dataset has been built from official ATLAS full-detector simulation. It is a collection of measurements on Higgs boson events as well as other collisions considered as 'background'. The aim of this paper is to find a binary classification on those measurement in order to separate the Higgs Boson detections from backgrounds events.
In the following work we will provide reasoning for choosing two distinct models, as well as compare their performances with different preprocessing methods. Finally we will conclude with the ultimate procedure on how we managed to achieve our best accuracy.

II. Data and methods

A. Model

Whilst for the information purposes all the basic functions are provided in the file attached, for the model testing we mostly focused on using the least squares GD and logistic regression. Even though the least squares approach is theoretically the best linear model one could use, it suffers when the data matrix is invertible. While this is not a problem for the original data matrix provided, when dealing with the feature expansion a lot of times the data matrix became singular and the least squares approach was not able to compute the weight vector. Therefore, for the complete comparison of two methods we opted out for GD and logistic regression.

B. Preprocessing

1) Data analysis: Before we had done any preprocessing we visually inspected the distribution of each feature vector to assess their usefulness during training as well as to help us decide which data augmentation techniques to use. It was evident that in many feature vectors (11 to be precise) the missing values accounted for substantial part of the entire vector, thus, drastically changing the distribution. We counted the number of missing values in each column and compared them to each other. We discovered that the columns contained 3 distinct counts of missing values and these were: 35279, 74421, 124255. Upon further inspection we have noticed that the smaller ones were subsets of the larger ones and the large counts were equivalent sets, thus, if we had decided to remove all the data points from one of the columns with large number of missing values it would mean that all of them would be removed from all the other columns as well. However, we decided to keep all the data points and simply replace the missing values (see the following section). From the Fig. 1 it is clear that some feature columns were more useful than the others. It is visually evident that the $Col\ 25$ (lower right quadrant) does not provide good distinction between detections and the background noise, on the other hand $log(Col\ 13)$ and $Col\ 0$ (upper and lower left quadrants) clearly differentiate between the two.

Henceforth, assume that the data was always appropriately normalised with the mean 0 and $\sigma = 1$.

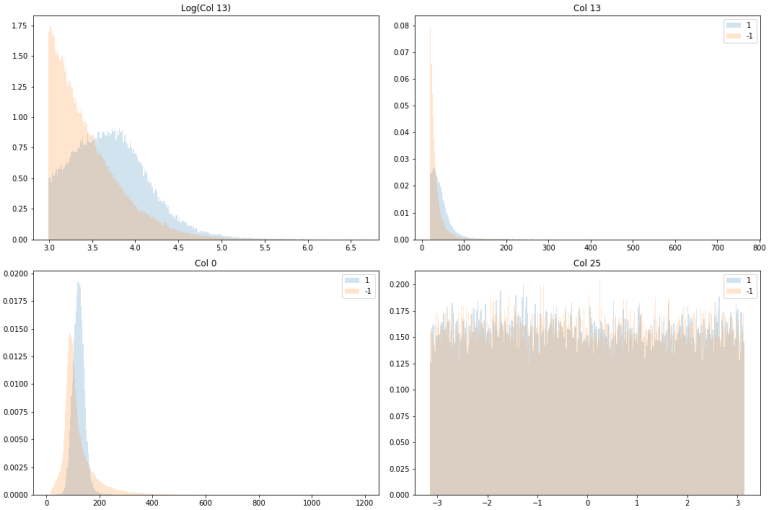

Fig. 1: The normalised histograms for selected features.

2) Missing values: Missing values had to be relocated within a meaningful range and we decided to do so after a standardisation step that did not consider those missing values. Replacing them with zero was a good start, but choosing a constant value with a statistical relevance adapted to each feature seemed more promising. Therefore, we tried to change the -999.0 with the mean, the median or the mode (computed by dividing the samples into small bins) of the feature they belong to. Without any further preprocessing, replacing missing values with any of the aforementioned constants was not very effective. However, once we combined this step with features selection and expansion (see point 5 and Table III), the mode gave us a slightly better accuracy. We expected this outcome: since distributions' patterns vary between features (see Fig. 1), the mode seemed to us best suited to maintain a consistent partition.

3) Outliers: Due to the different types of distribution, outliers where defined depending on the interquartile range of the feature. They were then replaced with the limit of the lower ($Q1 - 1.5 * iqr$) or upper bound ($Q3 + 1.5 * iqr$). As we expected, this step marginally improved the results of the linear regression but not the ones of the logistic regression, since the latter is by definition more robust to outliers. Given this minimal/nonexistent improvement, we decided not to use it when we proceed to a complete training of the dataset.

4) Under/Over sampling: When tackling the problem of classification and especially in the binary case like this one, it is crucial to ensure that the data in the training set is represented equally. The usual rule of thumb in ML is that the more data the better, to have complete and rich data set to avoid overfitting etc. However, this is not always the case when one particular event (in our case background noise) dominates all the others. In simpler terms if the data is not balanced then the model will only learn to guess one specific label, this will give false sense of training by outputting high accuracy. In our data set the ratio of background:detection events was $\sim 2:1$. We tried to balance the data set, such that the ratio was $\sim 1:1$ and used the following methods: under represent the background events by randomly removing them, over represent detections by adding Gaussian noise of form $\sim \mathcal{N}(0, 0.01)$ to randomly selected samples and by taking the middle value between 2 successive data points ($\frac{x_i + x_{i+1}}{2}$).

While simply looking at accuracies in Table I might suggest that after data augmentation the model performed worse than before, it might be worth looking at the columns labelled TPR and PPV. TPR tells us how many of total detections have we predicted, whereas,

Method	Acc.	TPR	TNR	PPV	F1
Without	75.42%	53.00%	87.17%	68.40%	59.72%
Under	73.99%	73.82%	74.07%	59.69%	66.01%
Gaussian	73.88%	77.28%	72.11%	59.01%	66.92%
Middle	73.94%	75.56%	73.10%	59.27%	66.43%

TABLE I: This table gives the summary statistics of different balancing techniques used. Here TPR is sensitivity, TNR is specificity, PPV is precision and F1 is a composite measure of a test's accuracy. These statistics were computed after standardisation only with the missing values replaced by mode of the column and trained on the logistic regression.

PPV is the measure of how precise were those predictions. It is left up to the reader's interpretation to decide which of them is the more useful measure of model's performance. While accuracy gives us a simple comparison, maybe in the particular case of event detection it is more useful to have more complete but less clean set of sample predictions. From the data augmentation techniques used above we argue that adding Gaussian noise marginally outperformed the others and thus, was further tested with other preprocessing techniques (see Table III).

5) Features selection/expansion: When it comes to feature expansion , we tested the following 3 transformations:

- $\mathbf{log(abs(x_1))}$: because as you can see in Fig. 1 some feature distributions looks exponential and normalising them helps the model
- $\mathbf{x_1 \times x_2}$: to take into consideration features interactions , we also tried combinations of the 3 but no improvement was noticed.
- $\mathbf{x_1^2}$: as higher powers of did not improve our model

All 3 transformations had a major impact on our model accuracy (see Results). However, they complicate our model by adding around 524 new features.

This was not an issue for our logistic regression but our linear model was too simple to handle these transformations. To counter this we performed feature selection.

To remove irrelevant and redundant features, we selected the initial 30 features in 3 steps :

- Removing the features that had the same distribution for both classes (as col 25 in Fig. 1)
- Removing the features that, when removed from training dataset, the accuracy did not change.
- Removing the features that had their weights very close to 0.

After this selection, we redo the same feature expansion, re-select all the new features and end up with 19 features. Instead of selecting twice, we could have just selected all the 524 features after the expansion but due to a lack of time and computational power we opted for this method. To sum up, our best accuracy was reached by the logistic regression with expansion and no selection.

Method	Linear	Logistic
Expansion	65%	82%
Expansion + Selection	79%	78%

TABLE II: Accuracy results replacing missing values before expansion

III. Results

The following table summarises our results of both linear and logisitc regression. We have split the dataset into train (80%) and test (20%) sets, trained and tested our model on the train and test sets respectively.

	Linear		Logistic	
Method	Accuracy	F1	Accuracy	F1
1) Without	74.35%	56.60%	74.40%	58.00%
2) Replace missing values	74.57%	57.10 %	75.00%	58.75%
3) Replace miss. & Outliers	74.47 %	58.15%	74.1%	58.10%
4) Data Augm. (Gauss)	72.93%	66.41%	72.92%	66.41%
5) Data Augm. (Gauss) + 2)	72.45%	64.69%	73.83%	66.56%
6) Feature Expansion +2)	79.71%	67.20%	82.48%	73.15%
7) Feature Expansion +2) +4)	78.19%	71.14%	81.52%	73.01%

TABLE III: The summary of different pipelines used in prepossessing. Note that for point 1 and 4 we simply standardised the whole matrix with -999. and we computed the training on that whereas for the others we standardised without taking into account -999, which were replaced after.

We also computed our model accuracy with the AIcrowd dataset. The pipeline used on the competition platform, which also gave us the best results (categorical accuracy: 82.2% , F1 : 74.3%), is as follows: replace missing values with the mode of the vector, expand features and proceed with logistic regression gradient descent (point 6 in Table III).

IV. Discussion

As one can see from Table III, selecting and expanding the features is the most crucial step in our procedure and the others become relevant only when combined. Once we obtained $\sim 82\%$ of accuracy, we computed the cross-validation by dividing the dataset into five groups. However, we noticed that there was almost no difference in the losses between the group we were testing and the groups we were training. This was an indicator that our model was not encountering over-fitting, therefore we decided not to use cross-validation. A possible explanation for not overfitting is a rather simple model that we have used that have low feature dimensionality and very large number of data points to train on, therefore, we argue that the use of regularised models that penalise over-fitting was not necessary.

This work has provided the comparison of two distinct models that are linear and logistic. It was evident from the beginning that when dealing with classification problem one should stick with the logistic regression as it provides a more meaningful output which is the probability of the label, whereas the linear regression tries to predict the labels and it has no meaning once the prediction value goes beyond -1 and 1. Furthermore linear regression suffers greatly from the presence of outliers, whereas the logistic regression does not. The results in Table III only support this hypothesis and therefore, in the final submission we used the logistic regression over the linear one.

V. Conclusion

To conclude we argue that for the specific classification problem logistic regression is a superior option compared to the linear one such as least squares GD. The standard preprocessing techniques, such as missing value replacement, outlier removal (for linear GD) and data augmentation only marginally improved the accuracy and only proved to be useful once combined with the feature expansion method, which we deem to be the biggest difference maker.

This improved the accuracy from $\sim 74\%$ without any preprocessing to $\sim 82\%$ with the combination of missing values replacement followed by our extensive feature expansion. Even though with our 82.2% accuracy we remained around the middle of the classification, our F1 score of 74.3% was quite outstanding compared to the neighbouring teams, which was a satisfactory accomplishment for us.

Due to time and computational constraints , we limited ourselves to this pipeline. But more feature expansions could be tested and a deeper analysis on the meaning of our features could make a difference.

Closer to the project submission deadline we noticed the peculiar nature of the feature column #22, which was labelled as the "jet number". Unlike any other column its entries were discrete and only contained values $\{0, 1, 2, 3\}$. We noticed that the the discrete values could partially explain the pattern of missing values. For example all the data points that had a value of 0 would have values missing from column 23 whilst all the other points with different jet number would not (in the same column). At the same time data point with jet number 1 would not have any missing values in column 23 but would be missing from column 12. This explains our finding mentioned in the Data analysis section (B1) of the paper that the large sets of missing values were equivalent, hinting on the fact that their distribution along the column was not random and must be associated with the type of experiment ran in Cern. We suggest splitting the data into 4 parts based on jet number and train 4 distinct models. However, this and the meaningful combination of 4 separate models into one is left for future andeavours.