ASSIGNING QUALITY LABELS IN THE HIGH-ENERGY PHYSICS EXPERIMENT ALICE USING MACHINE LEARNING ALGORITHMS*

Tomasz Trzciński, Kamil Deja

for the ALICE Collaboration

Institute of Computer Science, Warsaw University of Technology
Nowowiejska 15/19, 00-665 Warszawa, Poland

(Received July 23, 2018)

Data Quality Assurance plays an important role in many high-energy physics experiments, *e.g.* the ALICE experiment at Large Hadron Collider (LHC) in CERN. Currently used quality assurance methods rely heavily on manual labour and human expert judgements. This refers also to the Time Projection Chamber (TPC), one of the detectors employed by the ALICE experiment. To ease the burden of human quality label assignment, we investigated several state-of-the-art machine learning methods that can automate this process. The evaluated selection of the machine learning methods include artificial neural networks, support vector machines (with linear and non-linear kernels), as well as Random Forests and Naive Bayes Classifier. Our results for the TPC detector indicate, that over 75% of all data points classified by human experts, can be correctly evaluated without any human interaction using Random Forest classifier with over 95% precision.

DOI:10.5506/APhysPolBSupp.11.647

1. Introduction

Data Quality Assurance is an important task to ensure that the data is correctly registered. Therefore, it is essential to have proper Quality Assurance system. In some experiments, such as those carried out in LHC, careful manual data validation is no longer an option. The amount of individual data samples is already tremendous, and with upcoming improvement to "run 3", it will simply exceed human capabilities. Therefore, it is crucial to provide trusted and fast automatic data quality measurement method.

* Presented at the II NICA Days 2017 Conference associated with the II Slow Control Warsaw 2017, Warsaw, Poland, November 6–10, 2017.

The current QA process for Time Projection Chamber [1], which is the main tracking detector in the ALICE experiment [2], relies strongly on human experts. They assign certain quality label, using different tools which create descriptive statistics. Our idea is to use some well-known data classification methods to replace, or at least ease, the work of those experts. That is why the main goal of this work is to build a model which would be capable of predicting the quality of examples with the greatest possible sensitivity. We propose a specific data model, and based on test results, we select the best fitting classification algorithm.

We split the problem into 3 parts in all of which we build a binary classifier. Its objective is to predict if sample belongs to the specific class, or if not sure — leave it for the manual assignment by experts.

We considered several state-of-the-art classification algorithms, commonly used for different classification tasks: Support Vector Machine [3], Random Forest [4], Probabilistic Neural Networks [5], and Naive Bayes Classifier. We performed numerous tests with each algorithm to compare their best results. For parameters optimisation, we used automatic grid and gradient search algorithms. Mixing Random Forest Classifier with Adaptive boosting technique had proven to be the most suitable to this problem, and allowed to fully automatically assign 75% of all data points with over 95% precision.

This work is the first attempt to automatically asses the quality of data sample collection, so-called run, for the TPC detector at the ALICE experiment at the LHC. Results show that with this solution, we can significantly reduce detector experts labour. We are already working on the improvement which will work on the exact raw experiment data.

2. Dataset

Run is a set of experiments performed with the same detector parameters. We considered its descriptive statistics as a single data sample in our dataset. Its total size is around 1500 samples with over 250 numerical attributes, all of them assigned by the detector expert into one of the descriptive classes. For further investigation, we exclude runs, where the detector was offline and did not take part in the experiments. They can be easily distinguished as the ones with no data stored, so in this case, no sophisticated classification method is needed.

In Table I, we can see the high classes distribution imbalance since most of the runs occurred as a good quality ones. However, the class “not set”, refers to the situation where the evaluating expert was uncertain about the quality and, therefore, it can be considered a useful hint. We used it in our experiments as not good quality samples.

TABLE I

Distribution of samples assigned to labels.

Label	No. of samples
Detector was ON and running according to nominal specifications	**795**
Not set	**99**
Detector off	373
Good data but some not full TPC acceptance	**110**
Detector was ON but output cannot be trusted as it is known to be not usable	**68**

Because of the specific goal of our research, we propose to split the problem into 3 individual sub-problems of binary classification. In each case, we want to distinguish some of the samples associated to specific class, or when the precision is not sufficient, leave it to be manually assigned by an expert. Table II presents the division to those three approaches.

TABLE II

Class mapping for 3 created problems.

Class	Definitely good	Good	Definitely bad
Detector was ON and running according to nominal specifications	**v. good**	**good**	to check
Good data but some not full TPC acceptance	to check	**good**	to check
Not set	to check	to check	to check
Detector was ON but output cannot be trusted as it is known to be not usable	to check	to check	**bad**

First one is the retrieval of the 100% correct samples. This means that we combined runs assigned by experts to all classes other than "Detector was ON and running according to nominal specifications". The second approach was the extension of the first one, with the samples which were almost 100% correct — those assigned to "Good data but some not full TPC acceptance" class. The last task was determining 100% bad quality samples.

We propose this class mapping, because of its association to the potential usage, since each of the approaches addresses one of the main use cases in terms of data quality assurance. From further processing, we excluded samples which exceeded 8σ variation from mean value on each attribute. In total, it was around 30 runs (3%). We performed a PCA transformation to

reduce number of dimensions to 26, with preserving 100% of information in data. New parameters showed no significant correlation with date, so we were able to perform the k-fold validation method, without being concerned of biasing the results.

3. Methods

There are plenty of different, equally good classification algorithms. Selecting the best one is always a challenging task, and requires both great understanding of data specificity and accurate testing. From the numerous options we tested, we will describe only the ones which gave the best results. The ultimate goal of all classification algorithms is defining on the basis of p-dimensional data distribution in training dataset, the $(p-1)$-dimensional hyperplane, which will separate samples from different classes in the whole probable data domain. That definition poses a great difficulty in how to achieve this generalisation. Particular algorithms are used to approach the task in different way.

3.1. Naive Bayes

Naive Bayes Classifier is probably the simplest solution, which, however, sometimes can produce surprisingly good results. It is based directly on the Bayesian theorem of conditional probability. First, the classifier is trained which means evaluation of probability for each possible decision class in terms of all attribute values. Then, for a given sample x, with set of attributes A, the predicted class c is chosen as the one which has the highest probability $p(c|A)$.

What makes Naive Bayes Classifier truly naive is an assumption that all of the attributes are independent. This is obviously usually false, and often leads to pure results. Naive Bayes Classifier generates decision class boundary exactly in the middle between most probable attribute values for a different decision class. This is not always a good solution, especially for biased distribution of samples in classes. In our test, we tried to overcome this problem by applying weights to different decision classes.

3.2. Random Forest

Random Forest algorithm [4] is motivated by the idea that a bigger set of weak classifiers can produce better results than the single strong one. For the large number of trees, it follows the Strong Law of Large Numbers theorem. Together with the tree structure, it can lead to the conclusion that a big set of combined decision trees will not overfit to the dataset, but produce a limiting value of the generalization error.

The model building depends on creating multiple simple decision trees, each one trained on limited range of attributes. The final prediction for unseen samples x can be made by taking the majority vote from individual decision trees. In that way, the samples are divided by complex but linear decision boundary. Sometimes it is based only on limited number of dimensions, if they provide significant separation. This behaviour is one of the strong points of the algorithm since it usually leads to better generalisation.

When applying ensemble learning methods, it is known that the more different and the smaller simple classifier is, the better the outcoming results. We obtained that difference by assigning only five attributes to each tree. To keep them small, we used the maximal tree depth, which prevents overfitting.

3.3. Support vector machine

The support vector machine algorithm [3] is a linear classifier which constructs a decision boundary hyperplane in the best possible way. It is achieved by placing it so that it ideally separates data samples of different classes and, simultaneously, has the largest distance to the nearest training-data point of any class (so-called functional margin). This approach minimises possible generalisation error of the classifier.

There are numerous variants of SVM, however, the most important improvement is the application of non-linear kernel, which allows creation of non-linear decision boundary. It is resolved by mapping original finite-dimensional space into a much higher-dimensional space, where linear separation is possible [6].

In our tests, we used different kernels with polynomial and sigmoid functions. We also allowed not ideal separation, hoping to achieve better generalisation. The same as with Naive Bayes Classifier, we weighted decision classes to adjust to their distribution dissimilarity.

3.4. Probabilistic neural networks

Probabilistic neural network is a non-linear classifier following the basic idea from the Rossenblatt perceptron [7]. It is a simple neural network with only one hidden layer and a softmax function as an output. The training used is a simple back error propagation algorithm, which tries to minimise the loss function defined as a difference between returned probability of class, and the real one.

Probabilistic neural network creates a non-linear decision boundary which is often very complex. It can lead to the so-called over-fitting which means that the classifier is too well adjusted to the training set, so it looses its generalisation ability.

3.5. AdaBoost — Adaptive boosting

Adaptive boosting [8] technique relies on the simple idea that the classifier should consider which examples are harder to determine, and try to optimise the prediction with respect to this knowledge. The implementation is based on iterative approach with weighted samples. First, the model is created on the whole training set, with initial sample weights equal to 1. Then, in every iteration, the model is validated and the weights of the samples are reassigned so that the ones of the wrongly predicted values are being increased. In the end of the iteration, the new model is created, with respect to the newly assigned sample weights. It means that every sample is duplicated in the training dataset the number of times equal to its weight. The whole algorithm stops after specified number of iterations.

4. Results

Figures 1, 2 and 3 present the results of the performed experiments. We discovered that combination of Random Forest algorithm and Adaptive boosting technique is the best choice for this task. In total, we were able to predict around 75% of all samples, with over 95% precision.

For the first model, where we predicted only the good quality run, we obtained satisfying results 0,878 AUC for the best Random Forest classifier. The results are presented in Fig. 1. We aim at reducing the false positive rate. Precisely, in our dataset, we managed to predict 6,63% good quality samples with no error, 17,56% with 99% precision or up to 62,55% with 5% of errors.

When we combined 100% good quality runs with those without full acceptance, we improved the result of our classifiers up to 0,9 AUC. We built a classifier which was able to predict 7,16% of samples with full correctness, or up to 52,50% with 99% precision. As shown in Fig. 2, in this approach, Random Forest classifier outperformed other algorithms. We discuss this issue more carefully in Section 4.1.

For the last problem, we had probably too few samples for exact validation, however, the results we obtained were promising. We were able to predict even 20% of all definitely bad runs with no error.

4.1. Random Forest evaluation

We estimate a few reasons why Random Forest algorithm showed the best results among the others. First of all, with respect to the experts knowledge, not all of over 250 attributes seem to be essential in terms of measuring the quality of the given sample. It is known that some of the statistics used for the run description are oversensitive and can lead to false conclusions. Even when we deleted the known outliers, we can still expect that some of the given samples are not fully correct on all attributes.

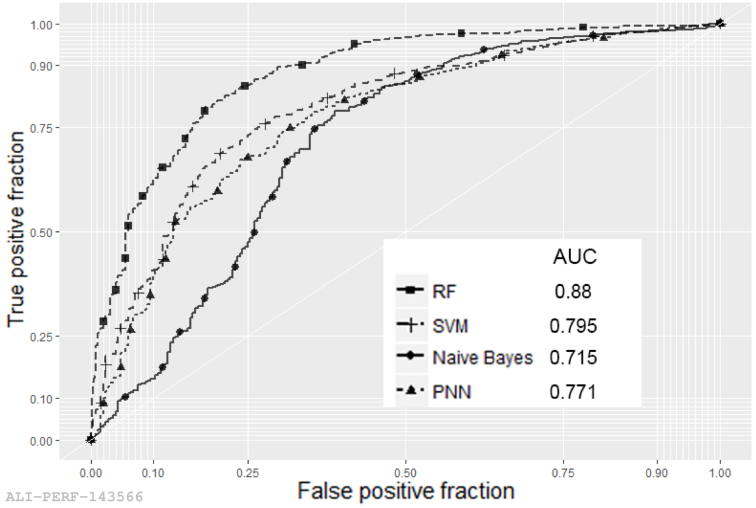

Fig. 1. ROC curves of different classification algorithms used to predict good quality runs. It is plainly visible that Random Forest algorithm gives the best results in this approach.

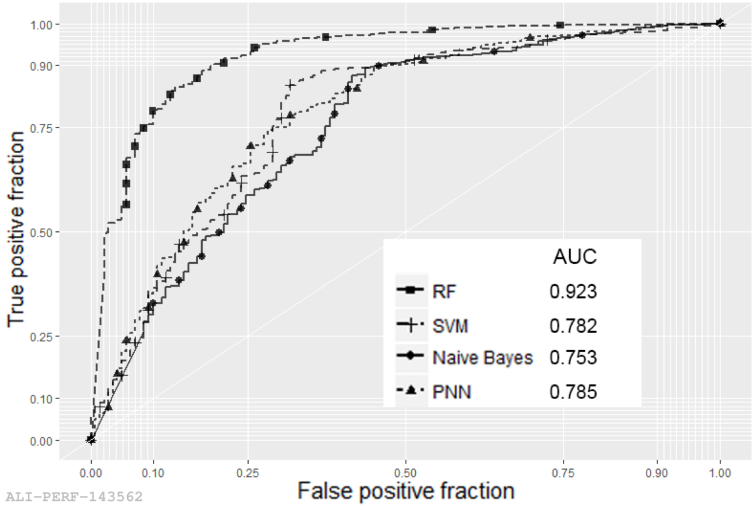

Fig. 2. ROC curves of different classification algorithms used to predict good and almost good quality runs. Random Forest clearly outperforms other algorithms. The results for this approach are also much better than for the others. That is thanks to combining good and almost good classes, which in reality are very similar.

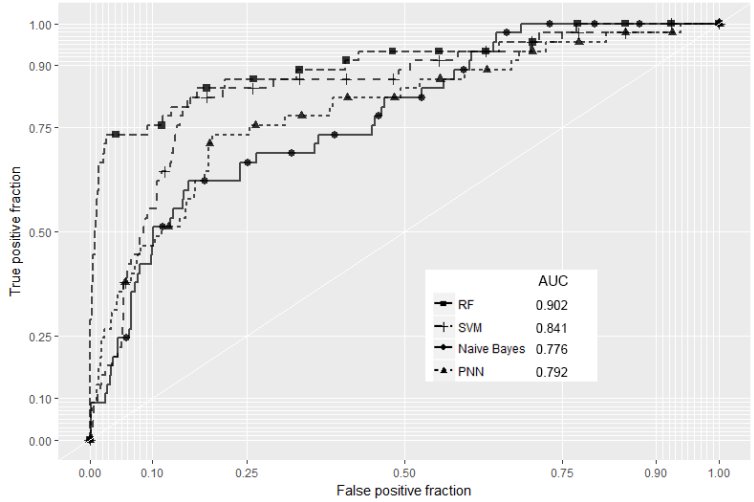

Fig. 3. ROC curves of different classification algorithms used to predict bad quality runs. Results in this approach show that we can correctly predict most of the samples, but because of small number of bad runs, those results cannot be fully trusted.

While using Random Forest algorithm such attributes are automatically detected and simply excluded from further investigation. All of the solutions using algorithms based on measuring the distance between samples for example SVM, are more affected by that problem. This is due to the nature of measuring the distance, which is calculated equally on all of the dimensions.

Secondly, the Adaptive boosting method is a very useful technique when building the prediction model on considerably small dataset. Its impact is the higher, the more unstable is the basic classification algorithm. In particular, it should not affect Naive Bayes Classifier and its application for SVM is of little use.

5. Future works

Knowing the importance of the problem, we are currently working on the new solution for fully automated quality assurance tool based on anomaly detection idea with the Generative Adversarial Network [9] model. Using conditional generative adversarial network [10], we will generate a series of probable particle showers based on known particles momentum. Then, when a new sample arrives from a detector, we will compare it to the results generated by our generator, and highlight possible anomalies. This idea is already successfully applied for medical purposes in [11].

6. Conclusion

We present a solution for automated quality assessment of runs for the TPC detector in the ALICE experiment at the LHC. We carried out detailed analysis with different data models and classification algorithms. It led to the solution with 3 binary classifiers based on the Random Forest algorithm combined with Adaptive boosting technique. Results show that with this concept, we can reduce detector experts labour by around 75%. We are currently working on the improvement of this solution which will work on the exact raw experiment data using the Generative Adversarial Network technique.

This work has been supported by the research project UMO-2016/21/D/ST6/01946: "Development of machine learning methods for monitoring the quality of large volume data and interactive methods for their visualization on the example of the ALICE experiment on the Large Hadron Collider at CERN", funded by the National Science Centre, Poland (NCN).

REFERENCES

[1] ALICE Collaboration, *Nucl. Instrum. Methods Phys. Res. A* **622**, 316 (2010).

[2] ALICE Collaboration, *Int. J. Mod. Phys. A* **29**, 1430044 (2014).

[3] C. Cortes, V. Vapnik, *Mach. Learn.* **20**, 273 (1995).

[4] L. Breiman, *Mach. Learn.* **45**, 5 (2001).

[5] D.F. Specht, *Neural Networks* **3**, 109 (1990).

[6] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed., Cambridge University Press, New York 2007, p. 883, Section 16.5: *Support Vector Machines*.

[7] F. Rosenblatt, *Psychol. Rev.* **65**, 386 (1958).

[8] Y. Freund, R.E. Schapire, *J. Comput. Syst. Sci.* **55**, 119 (1997).

[9] I. Goodfellow *et al.*, *Generative Adversarial Nets*, in: *Advances in Neural Information Processing Systems* 27 (NIPS Proceedings 2014), pp. 2672–2680.

[10] M. Mirza, S. Osindero, `arXiv:1411.1784 [cs.LG]`.

[11] T. Schlegl *et al.*, *Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery*, in: Proc. of International Conference on Information Processing in Medical Imaging, Springer, 2017, pages 146–157.