

**Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej**

Master Thesis

*Development of tools for data quality control for ALICE experiment at CERN, using machine learning methods.*

*Rozwój narzędzi do monitorowania jakości danych, dla eksperymentu ALICE w CERN, z użyciem metod uczenia maszynowego.*

Author: *Bartłomiej Cerek*

Field of Study: Computer Science

Thesis Supervisor: *dr hab. Adrian Horzyk, prof. AGH*

Kraków, 2020

*Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4*

*lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631*

*z późn. zm.): „ Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa*

*całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze*

*ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto*

*rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji*

*oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca*

*taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony*

*o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005*

*r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.) „Za naruszenie*

*przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student*

*ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem*

*koleżeńskim samorządu studenckiego, zwanym dalej „sądem koleżeńskim”, oświadczam, że*

*niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-*

*am) ze źródeł innych niż wymienione w pracy.*

*......................................................*

I would like to express my gratitude to my thesis supervisor for all the guidance, as well as to Professor Jacek Otwinowski and Sebastian Bysiak for valuable advice.

**Table of Content**

[1. Introduction 5](#_Toc37531748)

[2. Quality Control in CERN’s ALICE 7](#_Toc37531749)

[2.1. Introduction 7](#_Toc37531750)

[2.2. Anomaly Detection 9](#_Toc37531751)

[2.3. Anomaly Explanation 10](#_Toc37531752)

[3. Software Tools 11](#_Toc37531753)

[2.1. Python 11](#_Toc37531754)

[2.2. Data Manipulation and Computation Libraries 12](#_Toc37531755)

[2.3. Scikit-learn 12](#_Toc37531756)

[2.4. PyTorch 13](#_Toc37531757)

[4. Data Analysis 15](#_Toc37531758)

[4.1. Exploratory Data Analysis 15](#_Toc37531759)

[4.2. Features Selection 17](#_Toc37531760)

[5. Classic Machine Learning 21](#_Toc37531761)

[5.1. Supervised Outlier Detection 21](#_Toc37531762)

[5.2. Unsupervised Outlier Detection 22](#_Toc37531763)

[Bibliography 24](#_Toc37531764)

[Appendix A 26](#_Toc37531765)

# Introduction

In recent years, **Machine Learning (ML)** and **Deep Learning (DL)** are being frequently listed among the most prominent technologies. “We are now solving problems with machine learning and artificial intelligence that were… in the realm of science fiction for the last several decades. And natural language understanding, machine vision problems, it really is an amazing renaissance.” This is a quote from Jeff Bezos, CEO of Amazon, one of the biggest companies in the world, that succeeded among others, because of the great usage of data. [1] This is not a singular case, in more and more industries data science is being used to create business leverage and outrun the competition.

A natural question to put forward is, can those technologies be used to also help researchers? The development of new machine learning tools and models is a primarily academic field but its findings were mostly applied in business. Now, also scientists are more willing to use techniques of **Artificial Intelligence (AI)** in their work. Some experiments in medicine, physics or chemistry produce lots of data and automated algorithms are a convenient way to find patterns and dependencies that are difficult to be spotted by humans. Research facilities like **CERN** employ ML and DL to push forward the limitations of our knowledge. [2] Organized in 2014 The Higgs Machine Learning Challenge brought many AI enthusiasts to work on solving High Energy Physics problems and encouraged the further application of ML in science. [3] This enthusiasm is generally growing as tools and technologies connected to intelligent algorithms are more available and easier to use. Toolkits and libraries like Scikit-learn for Python are getting a vast number of users across many specializations, thanks to the ease of use and universality of application. [4]

There exist, however, problems and limitations in use of machine learning that should be mentioned. Nowadays many scientists discuss the so-called ‘reproducibility crisis’, a growing problem in metascience which is the inability to replicate or reproduce scientific studies. [5] Because ML algorithms are designed to find patterns in given data, even when there are none. They can fixate on noise and cannot asses their own uncertainty. As most of AI models work as black boxes, they take certain inputs and produce outputs, without explaining their decisions. This can be not only deceptive but sometimes even counterproductive when the ultimate goal is understanding the underlying process. Lack of decision argumentation reduces trust in ML methods and can even make it unusable in fields like medicine, where the confidence in decisions is crucial.

Those issues challenge machine learning and deep learning techniques used for scientific research. Growing movement of ‘explainable artificial intelligence’ propose new algorithms, that not only make a decision, but also try to argument it. Tools for explaining choices of existing black-box models are also researched and developed. [6][7]

The goal of this project is to propose and compare different machine learning and deep learning solutions to quality control task in CERN’s ALICE experiment while addressing the aforementioned issues. The scope of the project includes the analysis of example dataset from ALICE detector, data mining, development of different classification models and proposing pipelines for their usage in unsupervised, supervised and semisupervised cases. Evaluation of the obtained solutions is also provided. An important aspect is the comparison of methods based not only on their accuracy but also on the possibility of practical implementation in the environment of the ALICE experiment.

The thesis consists of 8 chapters, bibliography and appendix. First part is dedicated to introduction of quality control task in CERN’s ALICE and description of used in project software tools. Following this, in depth analysis of sample dataset is presented together with conclusions. Chapters 5 and 6 contain propositions and implementation descriptions of classic machine learning and deep learning models that can be employed as solution to presented problem. Advantages, disadvantages and possible workflow pipelines of each method are presented and discussed. Next entry touches the topic of AI expandability, presenting most suitable applications. Overall results, possible improvements and final results can be found in chapter 8, concluding the thesis.

# Quality Control in CERN’s ALICE

The following chapter describes the quality control problem in the CERN’s ALICE experiment. Purposes of the facility and the detector are summarized before presenting task requirements and evaluation guidelines.

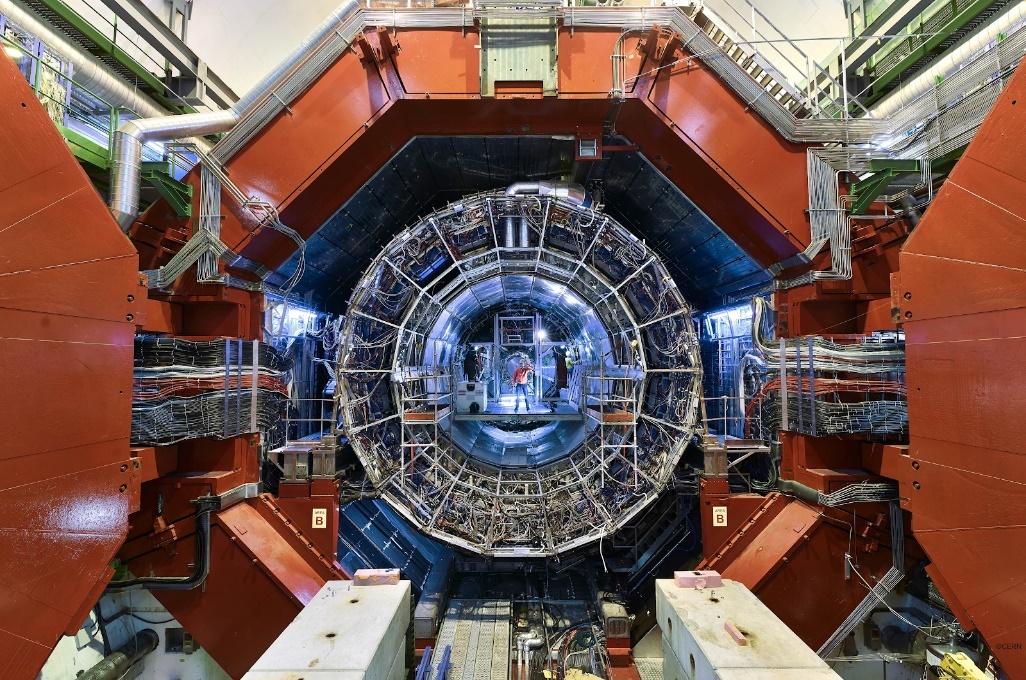
## 2.1. Introduction

**Conseil Européen pour la Recherche Nucléaire (CERN)** is a research organization, focused on particle physics. It was established in 1954 and is placed on the Swiss/French border, near Geneva. CERN is well known for its particle accelerators used for high-energy physics research and also for advancing engineering and computer science. [9] W and Z bosons (1983), as well as Higgs boson (2012), were discovered there. [10] Moreover, it is the birthplace of the World Wide Web (www).

**Large Hadron Collider (LHC)** is the largest particle collider in the world, that was designed and built at CERN and started operating in 2008. It has 27 km circumference and currently is able to reach the energy of 6.5 TeV per beam. LHC consists of 1 232 superconducting, dipole magnets that curve the beam to keep it on the circular path. Additional, quadrupole magnets help focusing the beam. Collider was constructed to answer some of the fundamental, open questions in physics, most notably regarding the Standard Model and mass of elementary particles. [11]

On LHC’s intersection points are located CERN’s experiments. Biggest of them, ATLAS and CMS are general-purpose particle detectors that allowed for the discovery of Higgs boson. Others are dedicated to specialized research. For the purpose of this thesis, **A Large Ion Collider Experiment (ALICE)** has to be presented in more detail. Its goal is to study the collisions of heavy ions (like plomb) to improve the understanding of **quantum chromodynamics (QCD)**, the strong-interaction part of the Standard Model. Beyond that, it can take proton beams data at the highest LHC energies to collect references for the heavy-ion program. ALICE weighs 10 000 tones and is located underground, close to the village of Saint Genis-Pouilly in France. [12] [13]

Operations of LHC are divided into active periods and ‘Long Shutdowns’. During active periods experiments and measurements are being conducted, while Long Shutdowns (LS) are intended for maintenance and introduction of new technologies. As of the period between 2019 and 2021 CERN is in Long Shutdown 2 in which a great upgrade of experiments is performed. ALICE detector is planned to produce a huge amount of data (more than 3 TB/s) and to cope with that, new systems for processing and quality control are being introduced. [14]



*Image 1. Opened ALICE detector. [15]*

The goal of data quality control in LHC’s detectors is to select the correct measurements to be stored and analyzed. All anomalies and acquired outliers should be discarded.  In this thesis, the focus will be put on data from the Time Projection Chamber (TPC). It is the main tracking detector in the ALICE experiment, which also serves particle identification. Measurements are being divided for ‘runs’ which include data from the single fill of the LHC, lasting usually up to 12 hours. Moreover, runs are divided for ‘chunks’, short 8 - 15 minutes periods of data acquisition. Finally, one sample of data to be analyzed is a chunk from a given run, that contains raw and derivated parameters.

Raw, low-level features are related to reconstructed particles’ tracks like a mean number of measured points in the particles' trajectories. Some others are related to global particles’ tracks, like the distance between the vertex and the extrapolated track. There is also group that gathers information from all reconstructed tracks and checks the efficiency and quality of extrapolation of the tracks to the neighboring detectors.

## 2.2. Anomaly Detection

Because of the multidimensionality, data cannot be easily analyzed by humans. Since the parameters of the experiment can change, it is also very difficult to set fix limits on them. The current approach to automated outliers detection was based on gathering data from a certain time frame and calculating mean and standard deviation for all parameters. Then, for all features, a warning or outlier flag was set depending on how many sigmas sample has deviated from the mean. 3 sigmas corresponded with warning and 5 with an outlier. In the end, flags for all parameters got combined, if for at least one parameter warning was set, the sample got a warning label and analogically if at least one parameter had outlier flag then the sample was labeled as an outlier.

This solution has many drawbacks, most notably it is over sensitive for singular deviated features. When data is highly multidimensional, as in this case, the singular skewed feature does not have to indicate that the sample is an anomaly. Comparing just simple statistical parameters does not grasp their underlying relations. On the other side, the approach has some advantages, it is fully automated, quick and does not require any intervention from personnel.

The quality assurance problem seems like a great use case for machine learning, it requires finding patterns in a big amount of multidimensional data. There are, however, few possible ways to apply ML to this task. First, would be supervised learning. In this case, the model would learn to assign labels based on some training set that was already marked. It could, for example, learn to imitate the classic approach, which does not bring much value.  The case would be different if an expert user would label some of the data points. Then, the model could learn to imitate the user. This still sets some limitations, if one model would be used for all types of runs, not regarding varying base parameters of measurement itself, it could generalize too strongly. If many models were used, the user would have to mark each dataset for training separately and that could be very time-consuming.

  The perfect approach would be a completely unsupervised solution, model that can learn to separate data by itself. This model could not only save time that would have to be invested in creating training datasets but also possibly learn new ways to perform the task and outperform classic automated outlier selection and also a manual one.

## 2.3. Anomaly Explanation

Going further, if a machine learning model could learn how to detect outliers in a new, more robust way, it would be greatly informative to explain its choices to humans. Also, it would be easier to build trust of the users. A common problem with ML models is that they are treated as black boxes. The user is specifying inputs and is expecting certain outputs, but inner working, the algorithm, remains unknown. Often, models are intended to work this way but nowadays more and more scientists are striving towards having explainable artificial intelligence. It aims to create solutions that not only can predict answers but also support them with arguments, for example, point out features that mostly influenced prediction.

If this approach would be employed for quality control in CERN’s ALICE, researchers could not only use machine learning as an automated solution for outliers detection but also have more trust in it and possibly obtain new information regarding data that was not considered before.

# Software Tools

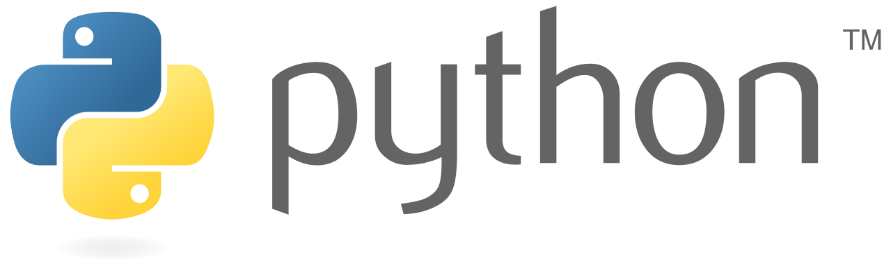
The development of tools for data quality control for the ALICE experiment is a software project focused on data analysis. To adhere to best practices of data science research only open-source and currently maintained tools were used. This chapter consists of short descriptions of the most important of them.

## 2.1. Python

Python is an interpreted, general-purpose programming language that was created by Guido van Rossum and released in 1991. It is multi-paradigm and highly extensible. [16] [17] Thanks to its flexibility, readability, and user-friendliness it is one of the most widespread programming languages in the world and language of choice of data scientists and machine learning engineers. [18] It is delivered with an extensive standard library, which contains the most important functions and structures for basic software development. The great advantage of Python is the fact that it is an open-source project maintained by multiple contributors.

Python can be extended by downloading and installing external packages created by the community. For package management tools like pip are often used, however, a popular choice of data science-related developers is Conda.  It is dedicated and also open-source package manager, that tracks dependencies of installed libraries and allows for creating separate virtual environments to handle conflicts. [19]

Most of the code and documentation for the project were developed in Jupyter Notebooks, web-based interactive computational environments. [20] They allow for transparent and well-annotated software prototyping and data analysis. Each chapter of the thesis corresponds with a set of notebooks, guiding through all steps of the reasoning process.



*Image 2. Python logo. [21]*

## 2.2. Data Manipulation and Computation Libraries

The popularity of Python in the data science field is mostly determined by the availability of powerful, open-source libraries that handle data. As they are structured in Pythonic (following guidelines of PEP-8) style, they are easy to use and intuitive. [22] Most of them are written in C (using Python C API) for low-level memory management and high performance. Crucial packages used in this project are:

* **Numpy,** a basic package for scientific computing. It allows for robust and seamless operations on big matrixes of numeric data and contains functions for linear algebra. Numpy performs much faster calculations than similar implementations written in pure Python. [23]
* **Pandas,** a library for general data manipulation, allowing to group chunks of multi-type vectors as data frames, managing them and executing SQL-like queries. Pandas are a great choice for exploratory analysis as they contain many functions for summarising and visualizing basic parameters of data. They can be used to conveniently load and save data frames from and to files. [24]
* **Matplotlib,** a toolkit for visualization purposes. It contains tools for the easy creation of elegant graphs, plots, and schemas. All graphical elements generated in this project utilize the Matplotlib module, pyplot. [25]

****

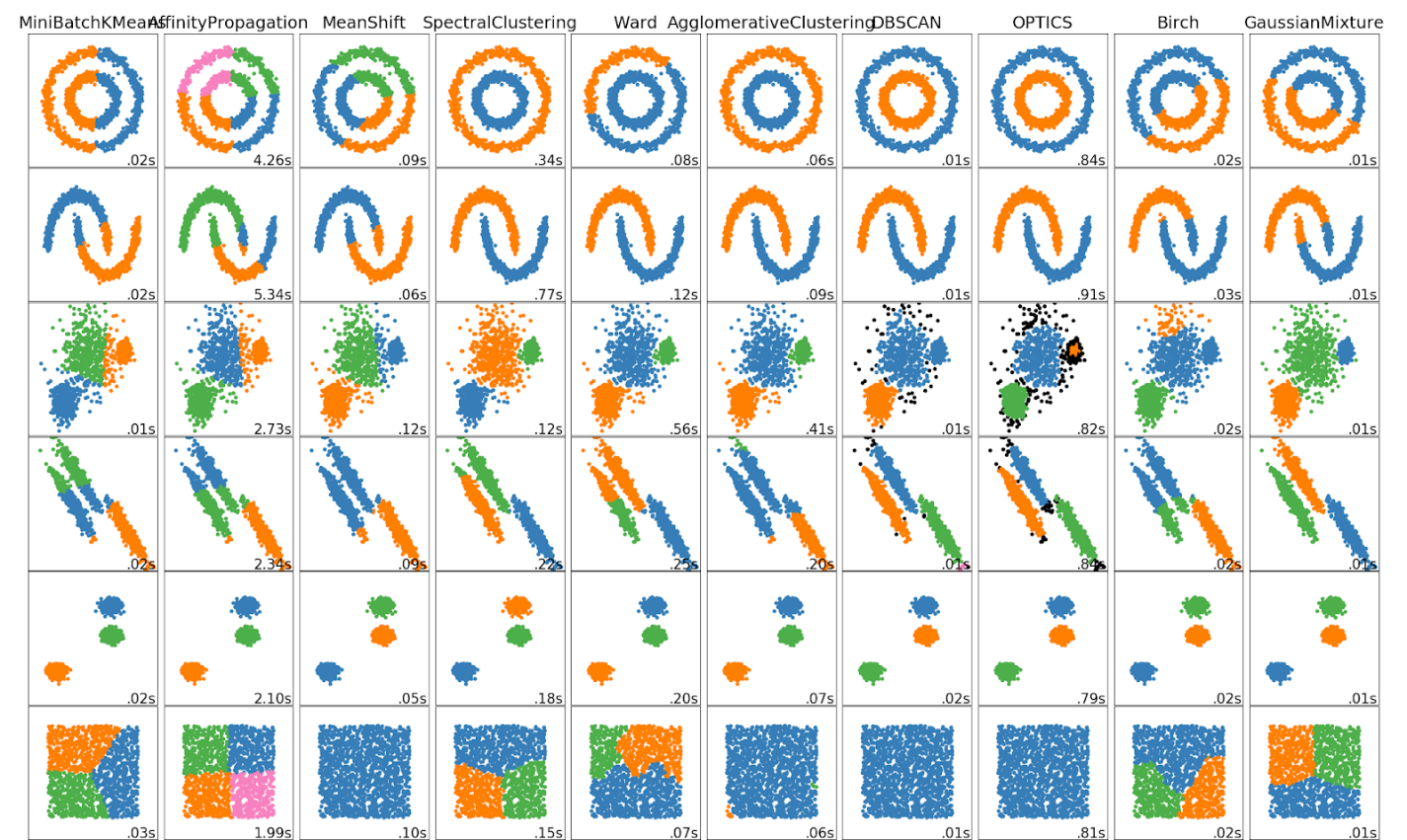
*Image 3. NumPy, Pandas, and Matplotlib logos.[26]*

## 2.3. Scikit-learn

Scikit-learn, often also referred to as sklearn, is an open-source machine learning library. It is built on top of NumPy, SciPy, and Matplotlib, so it can be seamlessly integrated with most data science projects developed in Python. As it is distributed on a BSD license, it can be freely used commercially. [27]

Sklearn contains a variety of working out-of-box models for regression, classification, clustering, and other standard ML tasks. Linear Regression, SVM, Gaussian Mixture or Naive Bayes are just to name a few popular examples. The user is also equipped with preprocessing and utility functions that cover most of the repeatable parts of projects. An important aspect of the library are standardized work pipelines. The majority of classes in Sklearn implement similar methods and properties. After understanding the workflow with one model, the user can perform the same steps with most of the other models and expect analogous behaviors. This intuitiveness, together with an approachable learning curve and great functionality, make Scikit-learn the first choice for many machine learning projects.

The library comes with detailed documentation and many examples that can be starting points for development. It also provides comparisons of different models and lists of criteria for the selection of the right tool. Most of classic machine learning models used in the project were trained and evaluated using sklearn functions.



*Image 4. A comparison of the clustering algorithms from scikit-learn documentation. [27]*

## 2.4. PyTorch

PyTorch is an open-source Python library based on Torch written in Lua language. It is mainly developed by Facebook AI Research Lab for deep learning tasks. Many of the most commonly used DL models, being part of libraries like Uber’s Pyro or HuggingFace’s Transformers, are created in PyTorch. [28]

Library provides functions for advanced tensor computations, including Graphics Processing Units (GPU) acceleration with Compute Unified Device Architecture (CUDA).  It also implements crucial elements necessary for the creation and training of artificial neural networks like optimizers, loss functions, and datasets handlers. A big advantage of PyTorch is the automatic differentiation package (autograd). It tracks operations performed on tensors and independently calculates required gradients, making backpropagation steps very easy to implement.

In general, PyTorch is more flexible than its major competitor Keras (being part of the TensorFlow package as of version 2.0) but requires a more low-level approach. Bigger code overhead is necessary for simple neural networks, however, just as Keras, package comes with ready to use standard layers like dense, convolutional or recurrent. To make easier use of PyTorch, packages are being developed to provide wrappers with sklearn-like interfaces. One of most notable examples is Skorch [29]



*Image 5. PyTorch logo .[28]*

# Data Analysis

Before applying any machine learning models, in depth data analysis and preprocessing needs to be performed. This chapter elaborates on dataset used in project and describes its properties. It also guides through the process of obtaining lightweight version of the dataset.

## 4.1. Exploratory Data Analysis

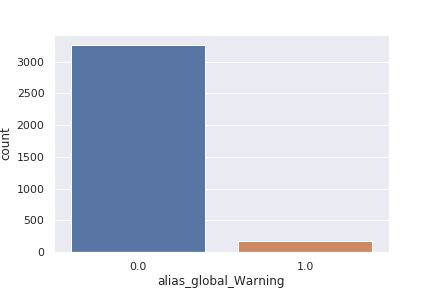
Usually, the first step when dealing with the new dataset is **Exploratory Data Analysis (EDA).** It is an approach to data, that aims in the summarization of its main characteristic, using basic statistical tools and visualization. [30] It was popularized by John Tukey, recognized American mathematician and creator of Fast Fourier Transform (FFT). EDA allows the user to quickly get a general orientation in the dataset and formulate the first hypothesis, even if not very precise, then being a good starting point. Basic parameters of concatenated f, o and p data frames were collected in *Table 1.* Then, descriptive parameters of all features were calculated.

*Table 1. Basic parameters of data from the ALICE detector.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Samples** | **Incomplete Samples** | **All Cols.** | **Metadata Cols.** | **No Variation Cols.** | **Labels** | **Features** |
| 3429 | 0 | 257 | 18 | 6 | 2 | 231 |

A simple conclusion that can be drawn from shapes of the obtained data frame, is that we are dealing with strongly multidimensional data while having not that many samples. Around **3500** is enough to train most of the simple machine learning binary classifiers but it can be not enough for more complex models and deep learning.

After dropping metadata and columns with no variation (all the same values) we are being left with 231 features that can be used for classification. There are also two binary labels, precisely: **alias\_global\_Warning** and alias**\_global\_Outlier**. They were both placed by a simple automatic outlier detection algorithm and can be used as a baseline for accuracy tests of created models. They can be also used to simulate labels placed by an expert. In this kind of anomaly detection problem, we prefer to have high **recall** (probability of detection) as we want to be sure to catch all outliers even for the price of a few false positives. This is why in all algorithms, more sensitive, **alias\_global\_Warning** will be used as a label. Only **4,9%** of all samples are marked with this label.



*Image 6. Bar plot comparing the number of inliers to the number of outliers in the data.*

As it was mentioned in the chapter 2 **Quality Control in CERN’s ALICE** some features are derivated from other ones, so there is suspicion of high correlations between them. Moreover, as the correlation of each feature to the label was examined, it turned out that many of them show close to no correlation.

To obtain a visual representation of the problem matrix of correlations was generated for all relevant features and labels. It can be found in **Appendix A**. As it is not easy to see through a matrix of 233 x 233 elements, all the irrelevant elements below diagonal and elements with absolute correlations below the arbitral value of **0.75** were left blanc. Even after filtering those elements there are **163** correlation pairs left. This is a high number taking into consideration that there are 231 features, however, it is not that informative. All of those features could be for example correlated to only one other feature. Unfortunately, looking at the generated matrix, we see that pairs are spread quite coherently. we can judge that in general there is a big **redundancy** in the dataset and dimensionality reduction can be a good idea, however, it has to be performed carefully.

The two most important pieces of the information that can be taken out from this exploratory analysis, are that we are dealing with the relatively small but highly multidimensional and imbalanced dataset. And that there is a big suspicion of data **redundancy**. Taken into consideration all the aforementioned arguments, **dimensionality reduction** is a natural next step in the preprocessing phase.

## 4.2. Features Selection

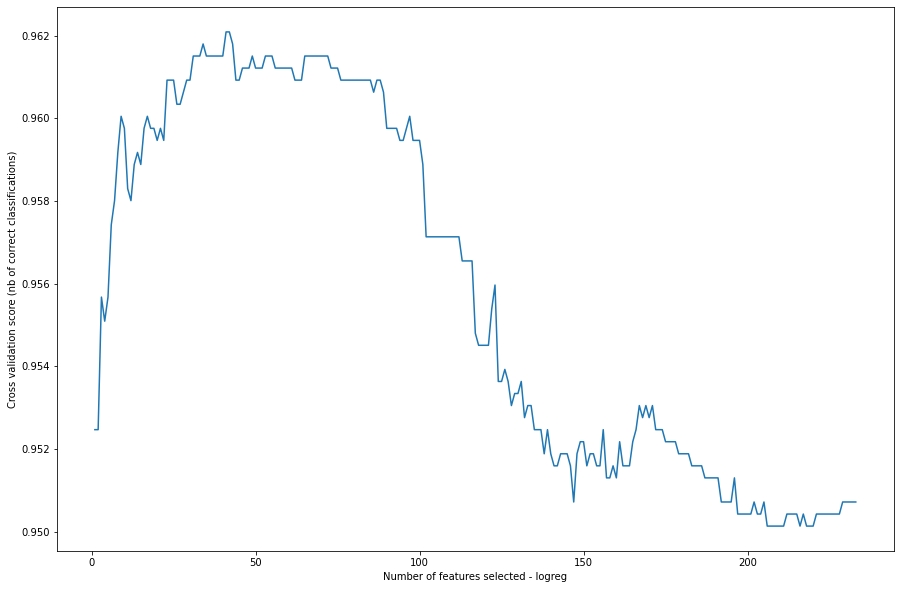
Dimensionality reduction is a process that aims to limit the number of random variables under consideration and obtain a set of principal variables. [31] It is usually applied when the amount of data to analyze is overwhelming or parts of it are redundant. Additional, useless information not only makes it difficult to operate on data but also slows down training and using of machine learning models. It can also introduce confusion to algorithms and lower their accuracy. [32]

Common approaches to dimensionality reduction are either removal of some variables or transforming data to contain the crucial information, from original data, in a smaller set of variables. Later method got big coverage in recent years thanks to very efficient techniques like Principal Component Analysis (PCA) or Autoencoders. [33] [34] Problem with those is that the obtained set of parameters is difficult to be analyzed by humans. Information like [height, weight, clothing size] can be quite successfully compressed by PCA to a two-element vector [V1, V2]. This new data can be used for classification tasks, but new features are generally meaningless without further processing.

One of the main goals of this project is to have the ability to explain why a certain sample was classified as outlier or inliner, by pointing out features that mostly influenced the model’s decision. Because of that, dimensionality reduction based on creating a new set of features is not applicable, even though it could result in better accuracy.

**Feature selection** is a simpler technique that aims to select the best parameters for further classification using some criterium, for example, correlation with the label. Usually, good criterium can be how strongly the accuracy of selected model decrease after removing the feature from the training set. **Recursive Feature Elimination (RFE)** is a method that applies this approach recursively and allows for ranking all features from most useful to most redundant. [35] Scikit learn contains, working out of the box, implementation of RFE that is compatible with most of the standard models. Selecting, which model to use is however not that trivial. Since for this project, many different classifiers are used, performing multiple refittings would take a lot of time. Moreover, some of them will be trained in a not supervised manner, so specifying the accuracy metric that RFE could use can be complicated. Rather than this, it would be more convenient to specify one model that more or less universally will select the best parameters.

As the main goal is still to decide if the new sample is outlier or inlier, simple binary classifiers may be a good choice. As the most appropriate option, **logistic regression** was selected for the task with 3-fold cross-validation, to predict **alias\_global\_Warning** label.



*Image 7. Cross-Validation Score for different numbers of used features.*

After performing RFE with classifier of a choice, features were ranked and saved in order in a file for further use. The number of features to use can be selected by finding the point where the accuracy is highest or arbitrarily, by balancing accuracy and resources. *Image 7.* shows how the cross-validation score was changing when more features were added. First, it is rising quickly as crucial data is added to the model. Best accuracies are achieved around 40 and 90 used features and then score is dropping as more redundant parameters are being used.

It is worth to notice two things. The range in which values are changing is very small, the lowest recorded score is above 0.95 while the highest is above 0.96. This means that logistic regression had no problems learning how to imitate a classic, automated detection algorithm and that was expected. What is interesting, a very small number of features was necessary to reproduce this effect.

Recursive Feature Elimination allowed us to determine which variables are important from the classification point of view. The remaining features should not, however, be blindly rejected. The goal is to surpass the classic algorithm for outliers detection, not to imitate it. When new models will be introduced, their performance may be improved by not using some of the redundant features but this has to be carefully validated.

Physicists working on ALICE data made their own list of **sensitive features**. Those 47 features are crucial from the point of view of scientists interpreting raw data, they can usually hint that the sample is an outlier.

The natural following step is a comparison, between features, selected by RFE and sensitive features. As it turned out in the top **100** parameters (out of 231) ranked by REF using logistic regression there were 19 sensitive features. It seems like not many, however, because data contains derivated features, that are sometimes non-linear transformation of other features, they can be actually more informative for the models and allow for easier classifications.

In conclusion when pure machine learning classification is performed, most appropriate, are parameters selected by the RFE method. When explainability starts to play a role, it may be worthy to append missing, sensitive features to see how the model will evaluate their impact on the decision.

KEEEP EMPTY!

# Classic Machine Learning

The following chapter is focused on applying machine learning methods, not involving deep learning, to solve the problem of data quality control in CERN’s ALICE experiment. This approach, nowadays often referred to as classic machine learning, makes use of tools like regressions and decision trees. Supervised, unsupervised and novelty detection techniques will be described and compared. [37]

## 5.1. Supervised Outlier Detection

In the chapter on Feature Selection, a machine learning model has been used to asses the usefulness of features. It was **linear regression**, one of the simplest binary classifiers. Those models utilize labeled training set to learn to predict which of two categories a new sample will belong. In the quality assurance problem, this technique can be used to mimic either classic automatic algorithm, by using for training labels generated by it, or more interestingly, the expert user.

To check the accuracy of the aforementioned solution three popular classifiers were trained and compared. Because of the inaccessibility of the expert data, labels assigned by the automatic solution were used to prove the concept. Classifiers were run on two data sets. Full, containing all 233 features and lightweight, containing 133 features. The lightweight dataset is a union between 100 most important features selected by RFE and sensitive features selected by ALICE physicists. 80% of data were used for training and 20% for testing, stratification was used to ensure a similar distribution of outliers in both sets.  As the dataset is highly imbalanced and contains way more inliers than outliers, apart from accuracy, **balanced accuracy** will be used as a metric. Classifiers used for tasks are:

* **Logistic Regression** - the most common approach to model the probability of binary events such as win/lose, pass/fail. It is based on estimating the parameters of the logistic function. It is possible to apply on its regularization methods such as l1 and l2, to improve its ability to generalize. [38]
* **Random Forest -** ensemble method, using bagging of multiple decision trees created from partial sets of features with bootstrapping. It is one of the most robust and well generalizing methods based on trees. [39]
* **Support Vector Machine** - an algorithm that places hyperplane in between two sets of samples from different classes. Kernels can be used to find separation where the linear split is not possible. [40]

*Table 2. Comparison of the accuracy of different supervised classifiers.*

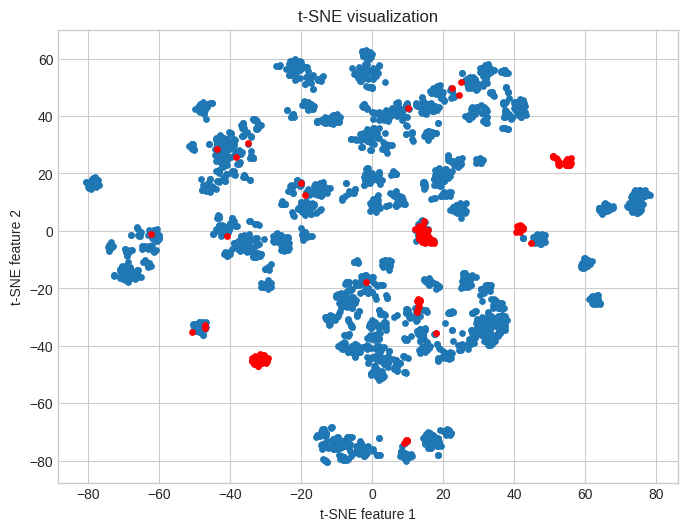
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Classifier | Full Dataset | | Lightweight Dataset | |
| Accuracy | Balanced Acc. | Accuracy | Balanced Acc. |
| Logistic Regression | 90.96% | 90.83% | 98.83% | 87.80% |
| Random  Forest | 99.42% | 95.38% | 98.83% | 90.76% |
| Support Vector Machine | 98.32% | 94.24% | 99.13% | 92.35% |

The first arising conclusion, similar to one from 4.2 subchapter **Features Selection**, is that simple machine learning algorithms were able to reach balanced accuracy around 90% without any finetuning, so imitating standard algorithm is relatively easy. Second, is that the lightweight dataset, even though almost half smaller, allows for obtaining quite similar results.

## 5.2. Unsupervised Outlier Detection

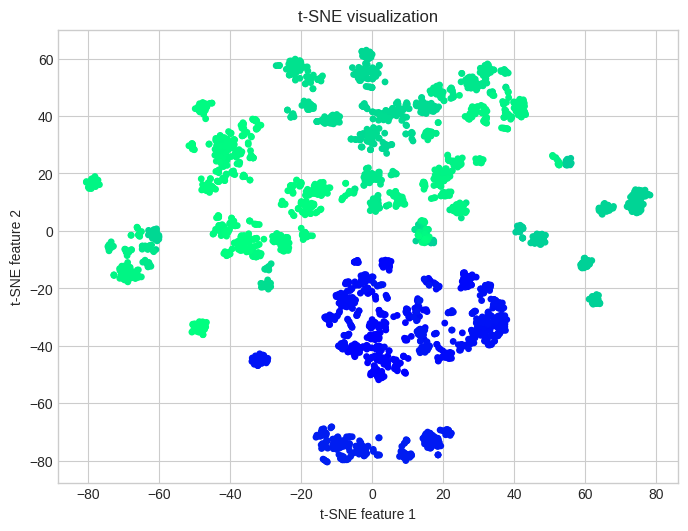
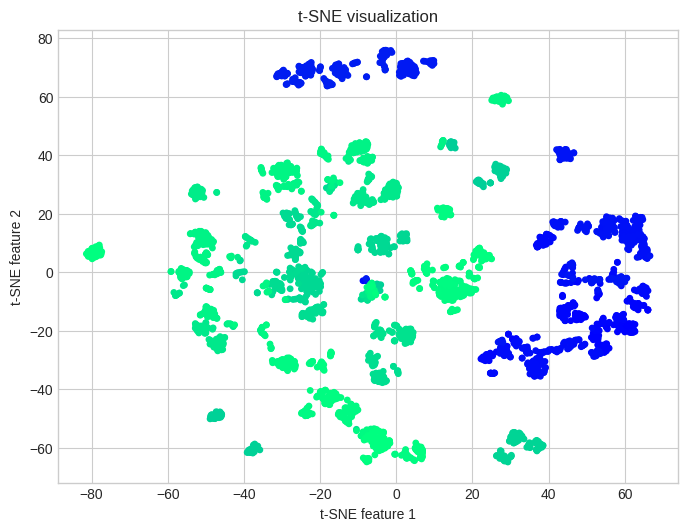
Unsupervised outlier detection is far more practical as it does not require a labeled training set, however, using it introduces new challenges. The most important of them is the evaluation of the results. We can compare groups obtained by unsupervised ML algorithms with those from the classic automatic approach, but as it is not ground truth and new techniques are expected to outperform the former approach, this is not the most suitable method.

Quality control data is highly multidimensional so even for an expert user, it is not easy to decide whether the sample is an outlier. It would be convenient to reduce the dimensionality of the data for visualization purposes, to get at least some intuition about localization of anomalies. For this task, **T-distributed Stochastic Neighbor Embedding (t-SNE)** can be employed. It allows for nonlinear dimensionality reduction by creating probability distribution over samples in multidimensional space, then a similar distribution in lower-dimensional space and reducing [Kullback–Leibler divergence](https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence) between them. Those distributions are created to give a high probability of pick to pairs of similar objects and low to dissimilar objects. t-SNE iteratively decreases differences between them and as a result, similar samples are close to each other in a low dimensional space and visual inspection becomes possible. Unfortunately, just as in the case of PCA, obtained features cannot be easily interpreted.  Another drawback of t-SNE is relatively slow execution, which however, is not an issue in case of rarely performed visualization. [41]

****

*Image 8. t-SNE visualization of normal (left) and lightweight (right) datasets with samples marked as outliers by the former automated method in red.*

Having the ability to visualize anomaly detection can be the first step of the result evaluation. t-SNE method for both versions of the dataset created many sparse clusters with few outlying regions rather than one big blob and surrounding outliers. It means that data is not coherent and probably the correct samples very strongly. Analyzed set contains measurements from around 200 runs of LHC. Samples from each run can be marked with a different color in the manner that runs that have a longer time frame between them will have a stronger difference in color.

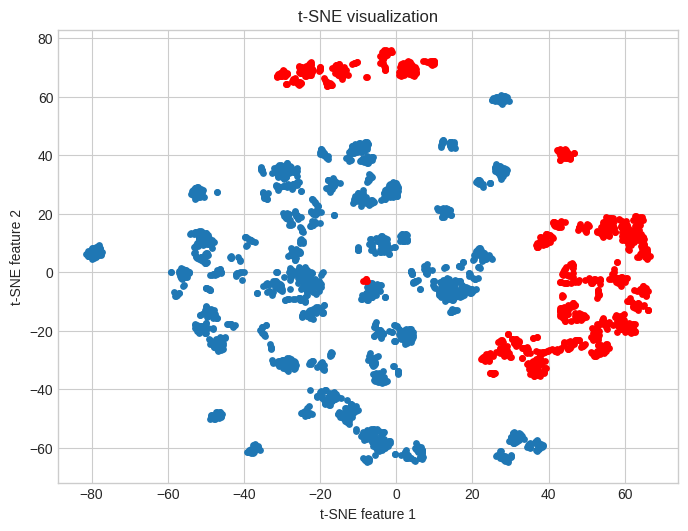
****

*Image 9. t-SNE visualization of normal (left) and lightweight (right) datasets with different runs marked with different colors.*

This way it is clearly visible that there are considerable differences between runs and the longer timeframe between them, the more visible they are. This is another proof that finding anomalies for quality control is not a trivial task. It is worth noticing that the orientation of samples does not play an important role as t-SNE only tries to preserve similarity by distance, while the exact location does not play a role.

General methods for initial evaluation, namely t-SNE visualization and comparison with the former algorithm, are established so unsupervised training can be performed. As there are no impactful differences between normal and lightweight datasets since now in this chapter only results from lightweight will be presented.

One of the most popular classic machine learning algorithms for cauterization is **k-means** algorithm. It is a simple technique to group data by minimizing the sum of squares within clusters as a criterion. [42] After setting the number of required clusters to 2 (inliers and outliers) algorithm was run on data and results were marked on t-SNE visualization that can be seen on *Image10*.

****

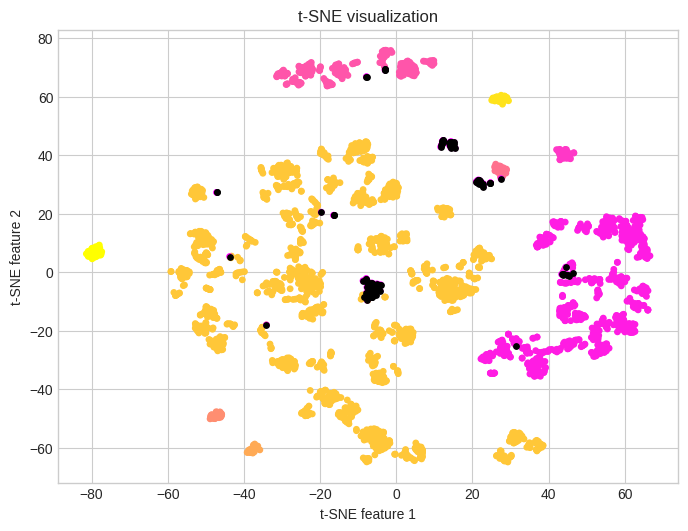
*Image 10. t-SNE visualization of lightweight datasets with k-means clusters marked.*

k-means did not bring expected results. Trying to create clusters with similar variations, the algorithm failed to select outlying samples. Valuable information that can be retrieved from the comparison of *Image9* and *Image 10* is that k-means separated the two biggest groups of runs. Strong difference between them has to be taken into account when applying further methods.

More sophisticated algorithms for clustering are **Density-based spatial clustering of applications with noise (DBSCAN)** and **Ordering points to identify the clustering structure (OPTICS)**. They are based on the exploration of low and high-density regions in data and do not require specifying the expected number of clusters. Moreover, they are able to point out outliers. [43][44][45] With this approach not only unsupervised detection is obtained, but also information about differences between samples groups (as differences between runs in case of this dataset) can be found and presented to the user. DBSCAN is generally less sensitive to irregularities in data, so it was used for quality control task and results were marked on the same t-SNE visualization as previous methods. The behavior of the algorithm can be modified by two key parameters, **eps** which is the maximum distance that two samples can have between each other to be considered as in the neighborhood of the other and a **minimum number of samples,** that have to be in the neighborhood of point to consider it as the core point. Changing those will adjust the DBSCAN for specific goals. In the case of ALICE quality control, relatively strong generalization is required, as detecting multiple correct clusters does not bring much new information. From the other side, sensitivity is still important to detect anomalies. For the selection process two arbitrary requirements were set:

* Number of correct detected clusters should be below **10**
* Percent of found anomalies should be **between 2 - 5%**

 Using the **grid search** method parameters eps=0.7, min\_samples=30 were selected.

****

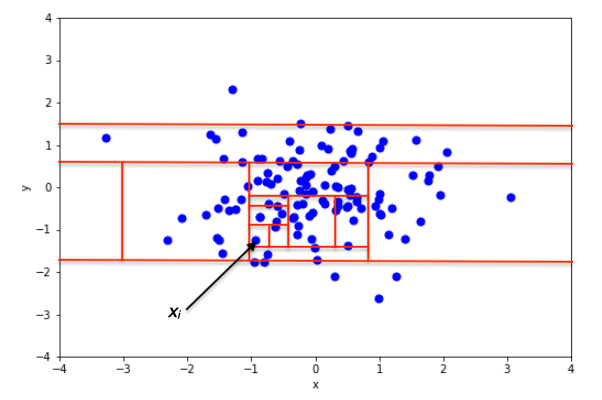
*Image 11. t-SNE visualization of lightweight datasets with DBSCAN clusters marked, outliers are black.*

 DBSCAN parameterized accordingly found 8 clusters from which 3 biggest correspond with the biggest groups of runs as seen in Image 2. It also located 134 (3,9%) of anomalies from which 104 were also flagged by the former automatic algorithm. This seems like a very promising result, DBSCAN turned out to be less restricting than comparing deviation from the mean, leaving out samples that had only a few outlying features.

## 5.3. Dedicated Anomaly and Novelty Detection Algorithms

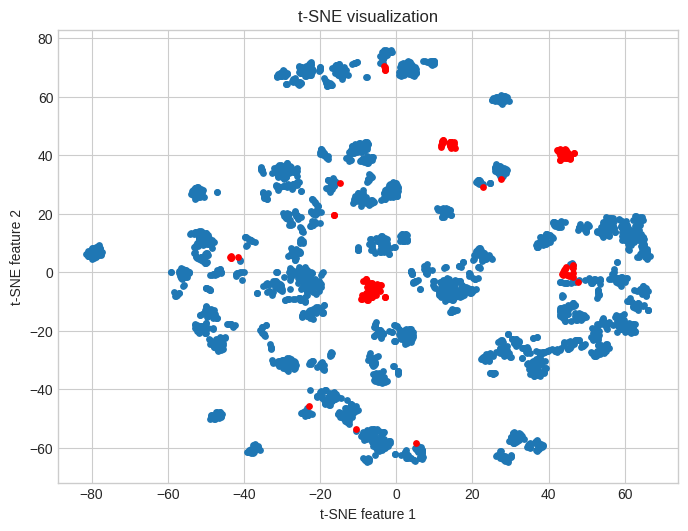
Problems of novelty and outlier detection were often tackled as supervised binary classification problems. Usual unavailability of training sets and their lack of balance were forcing researches to tilt to unsupervised methods. Simple techniques dedicated to anomalies issues, like computing robust covariance, proved to be useful only for coherent datasets but at the beginning of the 21th century, new, more specialized methods were developed. One of the most popular ones was the **Isolation Forest**.

Isolation Forest utilizes the fact that outliers are easier to separate from the core of the dataset than normal samples. Just like Random Forest, it is an ensemble method using trees, in this case, called iTrees. Multiple iTrees are being created by randomly splitting the dataset in a manner to isolate each sample. Fewer separations are required to isolate the sample, higher is its probability of being an anomaly. In the end, all created trees vote and, in this way, final decisions are being made. This straightforward algorithm is highly universal, it can be used with most datasets without any preprocessing and requires only specifying the number of trees to be used. As it has linear time complexity it is very fast and on top of that has low memory requirement. [46]



*Image 12. Example of random separations made by Isolation Forest to find outliers.[47]*

All those characteristics make Isolation Forest a perfect candidate for quality assurance task. Classifier, with a default number of iTrees (100), was fit to the lightweight dataset and found outliers were marked on t-SNE visualization.

****

*Image 13. t-SNE visualization of lightweight datasets with Isolation Forest outliers marked.*

**176** anomalies were found by Isolation Forest which is slightly more than the former algorithm identified. From those, **131** were matching which is, again, a very good score. From visual comparison with *Image 1.* it is clear that the same big concentration zones of anomalies were marked by both methods. There are, however, differences – Isolation Forest marked more samples around contracted zones and omitted many singular chunks labeled as outliers by a simple method, which were probably false positives.

As a conclusion, it seems that DBSCAN and Isolation Forest are the most suitable classic machine learning algorithms for the ALICE quality control task. The former method provides greater sensitivity for outliers and fast execution, while DBSCAN can bring additional information about differences in correct data. A full comparison between methods and juxtaposition with different approaches can be found in the chapter 8 **Summary and Conclusions.**

# Bibliography

TO BE REWRITTEN BY STANDARDS

[1]<https://www.cnbc.com/2017/05/08/amazon-jeff-bezos-artificial-intelligence-ai-golden-age.html>

[2] <https://journals.aps.org/prl/pdf/10.1103/PhysRevLett.121.241803>

[3] <https://iopscience.iop.org/article/10.1088/1742-6596/664/7/072015/pdf>

[4] <https://scikit-learn.org/stable/>

[5] <https://en.wikipedia.org/wiki/Replication_crisis>

[7] <https://arxiv.org/pdf/1708.08296.pdf).>

[8] <https://arxiv.org/abs/1602.04938>

[9] <https://home.cern/>

[10] <https://arxiv.org/pdf/hep-ph/9504378.pdf>

[11] <https://home.cern/science/accelerators/large-hadron-collider>

[12] <https://home.cern/science/experiments/alice>

[13] <https://iopscience.iop.org/article/10.1088/1748-0221/3/08/S08002/meta>

[14]  <https://indico.cern.ch/event/505613/contributions/2227515/attachments/1342855/2026384/Poster-v4-120.pdf>

[15] <https://www.indissoluble.com/wp-content/uploads/2018/04/ALICE1-1.jpg>

[16] <https://www.python.org/about/>

[17] <https://en.wikipedia.org/wiki/Python_(programming_language)>

[18] <https://www.tiobe.com/tiobe-index/>

[19] <https://en.wikipedia.org/wiki/Conda_(package_manager)>

[20] <https://jupyter.org/>

[21] <https://upload.wikimedia.org/wikipedia/commons/thumb/f/f8/Python_logo_and_wordmark.svg/1280px-Python_logo_and_wordmark.svg.png>

[22] <https://www.python.org/dev/peps/pep-0008/>

[23] <https://numpy.org/>

[24] <https://pandas.pydata.org/>

[25] <https://matplotlib.org/>

[26] <https://www.scipy.org/>

[27] <https://scikit-learn.org/stable/>

[28] <https://pytorch.org/>

[29] <https://skorch.readthedocs.io/en/stable/>

[30] <https://en.wikipedia.org/wiki/Exploratory_data_analysis>

[31] <https://en.wikipedia.org/wiki/Dimensionality_reduction>

[32] <https://members.loria.fr/moberger/Enseignement/AVR/Exposes/TR_Dimensiereductie.pdf>

[33] <http://pzs.dstu.dp.ua/DataMining/pca/bibl/Principal%20components%20analysis.pdf>

[34] <https://www.cv-foundation.org/openaccess/content_cvpr_workshops_2014/W15/papers/Wang_Generalized_Autoencoder_A_2014_CVPR_paper.pdf>

[35]

<https://www.researchgate.net/profile/Ke_Yan4/publication/272892870_Feature_Selection_and_Analysis_on_Correlated_Gas_Sensor_Data_with_Recursive_Feature_Elimination/links/5aa84dd7aca2726f41b16bf0/Feature-Selection-and-Analysis-on-Correlated-Gas-Sensor-Data-with-Recursive-Feature-Elimination.pdf>

[36] <https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html>

[37] <https://towardsdatascience.com/deep-learning-vs-classical-machine-learning-9a42c6d48aa>

[38] <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html>

[39] <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

[40] <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

[41] <http://www.jmlr.org/papers/v9/vandermaaten08a.html>

[42] <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>

[43] <https://dl.acm.org/doi/abs/10.1145/3068335>

[44] <https://dl.acm.org/doi/abs/10.1145/304181.304187>

[45] <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>

[46] <https://ieeexplore.ieee.org/abstract/document/4781136>

[47] <https://en.wikipedia.org/wiki/File:Isolating_a_Non-Anomalous_Point.png>

# Appendix A

