Outlier filtering in a sample

Alexander Usatoff
FAMCS of Belarusian State University
Minsk, Belarus
alexander.usatoff@gmail.com
ORCID 0009-0003-6387-1751

Alexander Nedzved
FAMCS of Belarusian State University
Minsk, Belarus
Anedzved@bsu.by
ORCID 0009-0003-6387-1751

Shiping Ye
Zhejiang Shuren University
Hangzhou, China
zjsruysp@163.com
ORCID 0000-0002-9771-716

The problem of filtering outliers in the sample is considered. A genetic algorithm for outlier filtering is proposed, its efficiency is tested on synthetic and real data in the linear regression problem. Synthetic data was generated by applying normally distributed random noise to a linear function. Real data check was performed on The Boston Housing Dataset. Since normally distributed random noise with small variance distorts the original function rather weakly and may, in general, have no outliers, the proposed outlier filtering algorithm showed a noticeably greater efficiency on real data, however, the positive effect of the proposed outlier filtering method was also noticeable on synthetic data.

Keywords—outliers, outliers in data, sample, complex sample, genetic algorithm, classification, regression

I. Introduction

In practical problems of statistics, machine learning and other areas of applied mathematics, it often becomes necessary to build a model based on a sample. However, the sample tends to have outliers – measurement results that stand out from the total sample. This can be due, for example, to incorrect measurement, sensor failure, software bug or human factor. Outliers in classification problem are shown in Figure 1 and outliers in regression problem are shown in Figure 2.

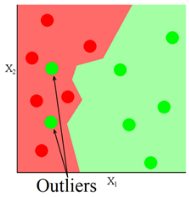

Fig. 1. Outliers in classification problem

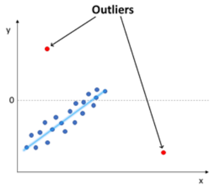

Fig. 2. Outliers in regression problem

Obviously, we want to reduce the influence of such factors on the model. One way to do this is to filter out the outliers from the training sample. The paper describes a genetic algorithm for filtering outliers in a complex sample and checks its effectiveness on model data and real-world data.

II. Existing methods of outliers filtering

A. Outliers problem

To begin with, consider the case of the category of detection problems. In machine learning, the logistic function of loss is often used [1]. The value of this loss function is calculated by the formula (1):

$$L_{\log}(y,p) = -\sum_{i=1}^{n}(y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)). \quad (1)$$

Without detracting from the generality, we consider the case when an object of class 0 was mistakenly assigned a label 1, while the object itself is a typical representative of class 0. In this case, the loss function on this object will turn to infinity, which will lead to incorrect calculations. To avoid turning the value of the loss function to infinity, in practice it is often limited to some sufficiently large number [2]. But even in this case, training will occur in such a way that even one incorrectly marked object can greatly affect the value of the loss function [7].

Consider another example, the AdaBoost algorithm. AdaBoost builds several classifiers, and the weights of objects that have been incorrectly classified increase exponentially with each subsequent classifier that makes mistakes on them [5]. If you do not solve the problem of emissions in the data, the AdaBoost learning process will reach the point that one of the classifiers can only be trained on objects that are emissions.

B. Existing methods of outliers filtering

To solve problems caused by outliers in data, there are the following methods:

1. One of the most basicways to filter emissions is to drop extremes. For example, 1% of the smallest and 1% of the largest values would be considered emissions. However, it is possibleto mislabel an object as an emission. For example, if we are talking about a sample in which people's height is measured, then a person's height of 210 centimeters will clearly be knocked out of the picture and with the described approach can be discarded. like an outlier. However, if we consider the height of the US national basketball team, there a player who is 210 centimeters tall may be more the norm than an ejection. But if a person mistakenly records growth not in centimeters, but in millimeters, this will cause an error of about an order of magnitude, in which case this object will definitely be an outlier. Chozhim is an approach in which outliers are values that lie outside the range of ±3σ.

2. The approach based on the Hampel filter works as follows: all values that do not lie in the range of plus or minus 3 medians of absolute deviations from the median are discarded.

3. The Grubbs test allows you to determine whether the highest or smallest value in a dataset is an outlier. It detects one outlier at a time (maximum or minimum value), so the null and alternative hypothesis of checking the maximum value looks like this:

H_0: The highest value is not an outlier

H_1: The highest value is the outlier

And the minimum one is like this:

H_0: The lowest value is not an outlier

H_1: The lowest value is the outlier

If the value of P is less than the threshold level of statistical significance (usually α = 0.05), then the null hypothesis is rejected, and it is concluded that the smallest/largest value is a deviation. In contrast, if the value of P is greater than or equal to the threshold level of significance, the null hypothesis is not rejected, and it is concluded that the smallest/largest value is not an outlier. T Grabbs is not suitable for a sample of 6 or less and requires a predetermined value of the amount of emissions in advance [6].

The Rosner test (ESD – extreme Studentized deviate) is used to detect outliers in a sample whose distribution is close to normal and has a size of > = 25. It also allows you to discard the highest and lowest values.

III. Genetic algorithms basics

The described above methods do not guarantee an optimal loss function of outlier filtration in terms of minimization. it is also not possible to consider all possible options for excluding objects from the sample, because this is an NP-hard task. Genetic algorithms are often used to solve NP-hard problems, and this paper describes just such an approach.

A genetic algorithm is a heuristic approximation algorithm that is used to solve optimization and modeling problems by randomly selecting, combining, and varying the desired parameters using mechanisms similar to natural selection in nature. It is a type of evolutionary computation that solves optimization problems using natural evolution techniques such as inheritance, mutations, selection, and cross-linkover. A distinctive feature of the genetic algorithm is the emphasis on the use of the "crossing" operator, which performs a recombination operation of candidate decisions, the role of which is similar to the role of crossing in wildlife.

In the problem under consideration, the population is the vector of inclusion of objects in the filtered sample:

$$\begin{cases} \text{arr}[\text{i}] = 1 \text{ if the i-th object remains in the sample,} \\ \text{arr}[0] = 0 \text{ other.} \end{cases} \tag{2}$$

Then, in some way (often randomly), a population is given: a set of objects that are current solutions to the problem. A fitness function (or fitness function) is introduced, which characterizes how "good" an individual is [4]. For example, if we are talking about a maxim problem and the creation of some function, it is logical as a function Adaptability to take the meaning of this very function. Then, the greater the value of the fitness function for an individual, the better this individual solves the task (maximizing the function).

At each step of evolution (also called epochs), individuals are selected on whom "genetic operations" will be performed, such as interbreeding, mutations, and so on. Often the best individuals have a higher chance of being selected. The value of the fitness function is then calculated and the least adapted individuals "die", that is, they are removed from the population to avoid its unlimited growth. Sometimes an additional restriction is imposed on the "age" of the individual, that is, how many epochs it is viable.

The described evolutionary process is repeated epoch after epoch until one of the conditions for stopping the algorithm is met. These conditions, for example, can be:

- Exhaustion of the time allotted for the solution of the problem

- achieving the maximum number of generations

- Lack of improvement in the population

Thus, the scheme of the genetic algorithm is as follows:

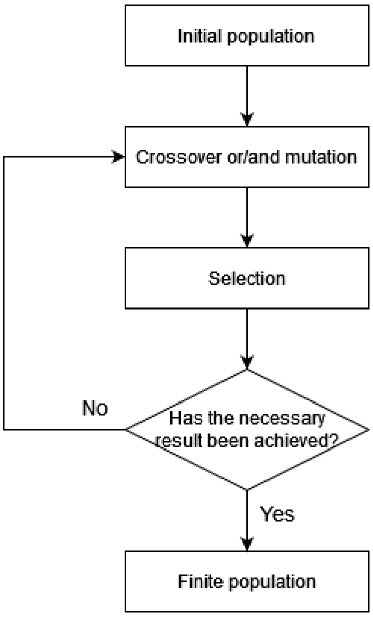
Initial population
Crossover or/and mutation
Selection
No
Has the necessary result been achieved?
Yes
Finite population

Fig 3. Genetic algorithm scheme

It is important to note that genetic algorithms have the following disadvantages.

- globally optimal solution is not guaranteed
- computational complexity

IV. Experiments

For the genetic algorithm in the described problem, possible important parameters were identified:

- save_part is the proportion of objects that will be stored in the sample in the initial population. Since noise is usually only a small part of the sample, let's put save_part = 0.95.
- N is the number of individuals in the initial population. It was set as the length of the vector describing the individual, multiplied by 10.
- fitness_function - a function that will assess the fitness of an individual or how "good" the individual is.

The fitness function is one of the most important parameters of the genetic algorithm, since it is on the basis of the fitness function that the best individuals are selected and the worst die. To solve the problem of filtering emissions, a fitness function has been developed that takes the following parameters:

- first_parent_part is the proportion of genes that a descendant will inherit from the first parent. Intuitively, it can be assumed that good individuals need only minor improvement, so it makes sense to try to leave most of the genes unchanged.
- robust_part - what part of the sample can be excluded without an additional penalty for it.
- mutation_proba is the probability of a gene changing to the opposite. In nature, random mutations play an important role in evolution. With some small probability, the gene will change its value to the opposite. Thanks to this, more possible combinations

will be considered, and the population will not degenerate.

- alpha_for_exclude_penalty is a parameter that affects the importance of the penalty for excluding the proportion of objects from the sample.

Fitness function returns the sum of the average square of an error on a sample and a penalty for excluding objects from the sample. For experiments it was decided to use the quadratic penalty function for excluding objects from sample.

V. Performance improvement

A. *Synthetic data*

Proposed approach was assessed in a linear regression problem. For the basic check, synthetic data was generated, which is a function of the form kx + b with added normally distributed noise. Below is a histogram of quality improvements in 100 cases. The quality of the constructed model was improved in 58% of cases. In the same 2%, the quality has not changed.

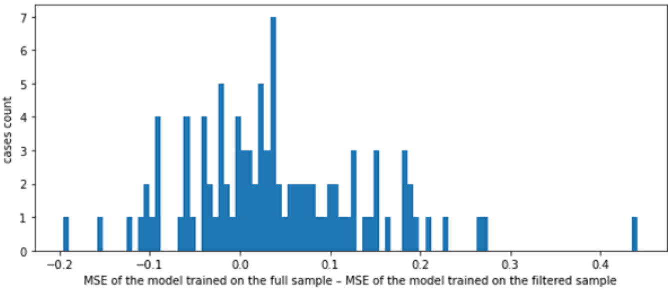

Fig. 4. Performance improvement histogram on synthetic data

B. *Real-world data*

To test the proposed approach on real data, we chose the regression problem on the Boston dataset. This dataset is one of the most famous examples for regression problem. Since in the case under consideration it is very important to assess the quality of the model as accurately as possible, most of the sample was used as a test. In addition, with a smaller training sample size, the effect of outliers can have a greater impact on the quality of the model.

Below is the distribution of the improvement in the quality of the model trained on the filtered sample relative to the quality of the model built on the entire sample. In 81 cases out of 100, outlier filtering by the proposed method improved the quality of the model. Below is a histogram of quality improvements in 100 cases.

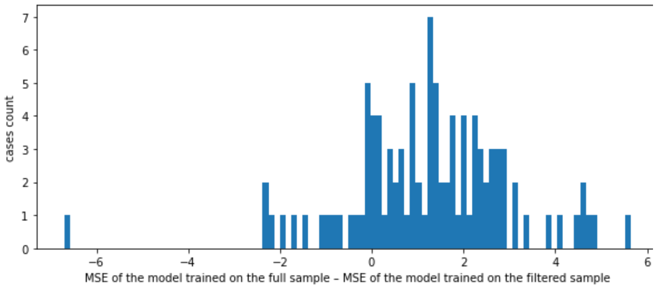

Fig. 5. Performance improvement histogram on real-world data

VI. Possible future improvements

The following possible improvements to the proposed approach can be noted:

1. Caching the results of a fitness function calculation. Training takes a lot of time. And for many individuals, the value of a fitness function is repeatedly recalculated. The recalculation takes a long time: you need to select a subsample from the training sample, then train the model, and then check the quality of the constructed model on the validation sample. Caching can be done, for example, by creating a class for an individual object in which the result of calculating a fitness function will be stored.

2. Change the generation of the initial population. In the current implementation, the initial population is generated completely randomly. As a more meaningful initial approximation, you can take a sample filtered by some naïve method. For example, discard the objects on which the average square of the error is the largest or make the exclusion of the object of their sample during generation probabilistic, so that the probability is higher the greater the square of the error on it.

VII. Conclusion

In this paper, we developed and implemented the filtering of emissions in the sample and improved the representativeness of the data using a genetic algorithm, the work of which is focused on the transformation of a sample with the presence of emissions and noise. The proposed algorithm showed its effectiveness on model and real-world data on a large number of tests conducted. The genetic outliers filtering algorithm is designed to be included in the data preprocessing process before or in conjunction with the use of machine or deep learning algorithms and can be used in various software systems for data analysis and processing.

Acknowledgment

The authors wish to thank Polina Tishkovskaya for paper design.

References

[1] Vyugin V.V. Mathematical Foundations of the Theory of Machine Learning and Forecasting V.V. Vyugin: 2013. – 387 p

[2] Faleychik, B.V. Mathematical Modeling and Optimization complex systems. / B. V. Faleychik. – Minsk, 2019. – 81 p.

[3] Vitaliy Feoktistov. Differential Evolution: In Search of Solutions (Springer Optimization and its Applications). 2006. Springer-Verlag, Berlin, Heidelberg.

[4] Vapnik V. Principles of Risk Minimization for Learning Theory // Advances in neural information processing systems. — 1992.

[5] NIST/SEMATECH e-Handbook of Statistical Methods. – Mode of access: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35h3.htm. – Date of access: 30.03.2022.

[6] Genetic Algorithm: Reviews, Implementations, and Applications. – Mode of access: https://arxiv.org/ftp/arxiv/papers/2007/2007.12673.pdf. – Date of access: 30.03.2022.

[7] Burakov M. Genetic algorithm: theory and practice. – St. Petersburg, 2008. – 164 p.