1-1-2021

How to Gauge Reliability of a Binary Classification Result: A Simple Case

Olga Kosheleva
The University of Texas at El Paso, olgak@utep.edu

Vladik Kreinovich
The University of Texas at El Paso, vladik@utep.edu

Comments:
Technical Report: UTEP-CS-21-11
Published in *Applied Mathematical Sciences* 2021, Vol. 15, No. 2, pp. 95-99.

How to Gauge Reliability of a Binary Classification Result: A Simple Case

Olga Kosheleva[1] and Vladik Kreinovich[2]

[1]Department of Teacher Education
[2]Department of Computer Science
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
olgak@utep.edu, vladik@utep.edu

Abstract

In many practical situations, we need to make a binary decision based on the available data: whether an incoming email is a spam or not, whether to give a bank loan to a company, etc. In many such situations, we can (and do) use machine learning to come up with such a decision. The problem is that while the results of a machine learning model are not 100% reliable, the existing machine learning algorithms do not allow us to decide how reliable is each result. In this paper, for simple examples, we provide a technique for gauging this reliability.

Mathematics Subject Classification: 68T05 62C05

Keywords: binary classification, machine learning, reliability

1 Formulation of the Problem

Need for binary classification. In many practical situations, we need to perform a binary classification, i.e., we need to decide whether a certain property is satisfied or not. For example:

- a computer security system needs to decide whether a given email message is a spam or not,
- a military security system needs to decide whether the image on a radar is an indication of an enemy attack,

- a road management company needs to decide whether the pavement needs repairs or can serve for some more time,
- a bank needs to decide whether to give a company a loan,
- a medical doctor needs to decide whether a patient has a certain disease, etc.

In all these cases, we need to make the corresponding decision based on the relevant measurement results.

Use of machine learning. In rare cases, we know exactly how to make a decision based on the corresponding measurement results. However, in most real-life situations – e.g., the ones mentioned in the previous paragraph – we do not have a clear algorithm. Instead, we need to learn how to classify based on examples in which the classification is known – i.e., by using machine learning; see, e.g., [1, 2].

In each such example k ($k = 1, \ldots, K$), we know the values

$$x^{(k)} = \left(x_1^{(k)}, \ldots, x_n^{(k)}\right)$$

of the corresponding measurement results, and we know to what class this situation belongs. For binary classification, we can describe the two classes by taking $y^{(k)} = 0$ or $y^{(k)} = 1$ depending on which of the two classes this situation belongs to. Based on these example, we want to find a function $f(x_1, \ldots, x_n)$ for which, for all known examples k, we have

$$f\left(x^{(k)}\right) \approx y^{(k)}. \tag{1}$$

In many cases, this closeness is described by minimizing the Euclidean distance between the vectors $\left(f\left(x^{(1)}\right), \ldots, f\left(x^{(K)}\right)\right)$ and $\left(y^{(1)}, \ldots, y^{(K)}\right)$, i.e., equivalently, by minimizing the square of this distance:

$$\sum_{k=1}^{K} \left(f\left(x^{(k)}\right) - y^{(k)}\right)^2; \tag{2}$$

however, other measures of closeness are used as well; see, e.g., [1, 2].

Problem. By applying the resulting function $f(x_1, \ldots, x_n)$ to the measurement results corresponding to the given situation, the system will decide whether the situation has the corresponding property or not (and sometimes, the system is allowed to say "I do not know"). The problem is that this decision is not 100% reliable – as any empirical decision is not 100% reliable. The actual experience of using machine learning shows that sometimes the recommended decision is incorrect. It is therefore desirable not only to produce this decision, but also to indicate to what extent this decision is reliable.

Such information is provided by many statistical methods (see, e.g., [3]), but not by most machine learning techniques.

What we do in this paper. In this paper, we show how, in simple situations, we can gauge reliability of the binary classification result.

2 What We Mean by a Simple Case

What do we mean by simple situations. Let us first describe what we mean by a simple situation. Specifically, we will list possible complications which make the problem of gauging reliability even more difficult to solve, and explain that by a simple situation, we mean a generic situation in which these additional complications are not present.

First assumption: we have sufficiently many examples. In some cases, we do not have enough data to make reliable predictions. In this paper, we assume that there was a sufficient amount of data, so that we can make meaningful predictions – and not just trying to guess based on a single example.

Second assumption: we have been able to find the global minimum. In some cases, we get stuck in a local minimum of the objective function (2). For simplicity, we assume that this is not the case, that we have reached the global minimum of the expression (2) – or at least that we are close to the global minimum.

Third assumption: there are sufficiently many parameters in our model. The usual way to find the desired function is to start with some general model $f(x_1, \ldots, x_n, c_1, \ldots, c_m)$ and to find the values of the parameters $c_1, \ldots, c_m$ that minimize the objective function (2).

If we use, e.g., linear models

$$y = c_0 + \sum_{i=1}^{n} c_i \cdot x_i,$$

with only $n + 1$ parameters, then it is difficult to achieve closeness (1) for all the examples. For simplicity, we assume that we use a model with sufficiently many parameters, so that it is possible to achieve closeness (1).

3 Analysis of the Problem and the Resulting Recommendation

Case when all examples indicate the same thing. In general;

- we have a situation $x = (x_1, \ldots, x_n)$, and
- we want to find out whether the corresponding property is satisfied in this situation.

Crudely speaking, what machine learning does is looks for similar past situations $x^{(k)} \approx x$. In some cases, in all such close-to-x situations, we had the same classification $y^{(k)}$ – for example, the same value $y^{(k)} = 0$. In this case, under our simplifying assumptions, the function $f(x)$ will be close to 0 for all these values and thus, close to 0 for the new situation x as well. So, we will get $f(x) \approx 0$.

Similarly, if for all similar-to-x situations k, we had $y^{(k)} = 1$, then our model will return $f(x) \approx 1$.

Often, the situation is more complicated. In reality, the situation is usually more complicated – simply because the available information x is not completely sufficient to uniquely determine whether the situation has the corresponding property or not. For example, whether a company will be able to repay the loan depends not only on the company itself, it also depends on the overall difficult-to-predict economic situation.

Analysis. In such realistic settings, out of all C close-to-x situations – i.e., situations for which $x^{(k)} \approx x$:

- for some of them, we have $y^{(k)} = 0$ and
- for others, we have $y^{(k)} = 1$.

Let us denote the number of such situations, correspondingly, by C_0 and C_1, so that $C_0 + C_1 = C$. In this case, for these situations, minimizing the corresponding part of the criterion (2) means minimizing the sum

$$C_0 \cdot (f(x) - 0)^2 + C_1 \cdot (f(x) - 1)^2. \tag{3}$$

Differentiating this expression with respect to the unknown $f(x)$ and equating the resulting derivative to 0, we get

$$2C_0 \cdot f(x) + 2C_1 \cdot (f(x) - 1) = 0.$$

If we divide both sides of this equality by 2 and then move terms not depending on $f(x)$ to the right0hand side, we get $(C_0 + C_1) \cdot f(x) = C_1$, i.e., $C \cdot f(x) = C_1$ and

$$f(x) = \frac{C_1}{C}. \tag{4}$$

In other words, in this case, the value $f(x)$ generated by our trained model will be different from 0 and 1 – and will be equal to the proportion of the close-to-x examples for which $y^{(k)} = 1$.

Towards a recommendation. If in half of the previous close-to-x cases, we have $y^{(k)} = 1$, then a natural conclusion is that in our case:

- we will have $y = 1$ with probability 0.5, and
- we will have $y = 0$ with the remaining probability $1 - 0.5 = 0.5$.

Similarly, if in 70% of the previous close-to-x cases, we have $y^{(k)} = 1$, then a natural conclusion is that in this case:

- we will have $y = 1$ with probability 0.7, and
- we will have $y = 0$ with the remaining probability $1 - 0.7 = 0.3$.

In general, we arrive at the following recommendation.

Resulting recommendation. For any new situation $x = (x_1, \ldots, x_n)$, we plug in the measurement results $x_1, \ldots, x_n$ into the trained model, and get some value $f(x)$. In our simplified situation, this value always belongs to the interval $[0, 1]$.

Our recommendation is to view the value $f(x)$ as the probability that for the situation x, we have $y = 1$. With the remaining probability $1 - f(x)$, we have $y = 0$.

Acknowledgments

This work was supported in part by the National Science Foundation grants 1623190 (A Model of Change for Preparing a New Generation for Professional Practice in Computer Science), and HRD-1834620 and HRD-2034030 (CAHSI Includes). It was also supported by the program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478.

References

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, New York, 2006.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, Massachusetts, 2016.

[3] D. J. Sheskin, *Handbook of Parametric and Non-Parametric Statistical Procedures*, Chapman & Hall/CRC, London, UK, 2011.

Received: January 16, 2021