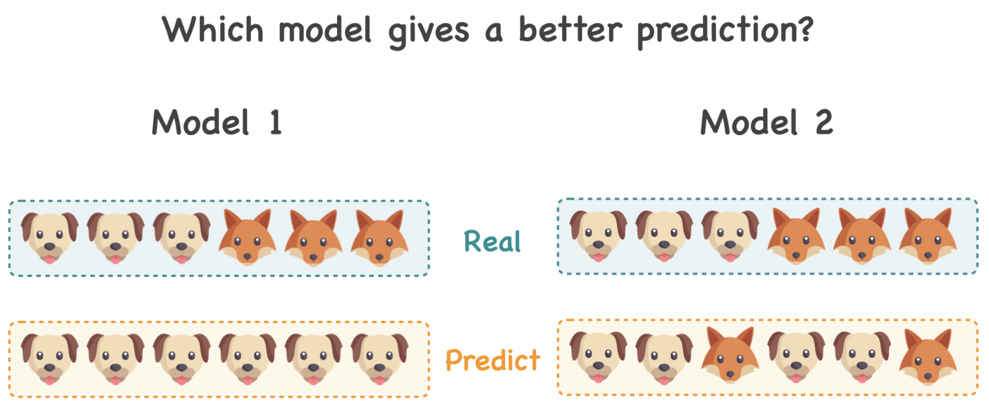
Theory: Classification performance metrics

93 users solved this topic. Latest completion was 1 day ago.

46 minutes 0 / 5 problems solved

Skip this topic Start practicing

You have already done a lot of work at this stage — you prepared the data, you figured out the libraries, you built a couple of models. But how do you determine which model is better? Compare the following two model outputs before answering this question: "Which model gives a better prediction?"



Any ideas? As you can see, the question of the model's quality is not as trivial as it might seem at first glance. In this topic, we are going to analyze various widespread metrics used in binary classification problems and look at cases where each metric performs best.

§1. The basics

A **metric** reflects the relationship between two lists — the real data and the prediction. It is important to note that none of the metrics is related to the model. A metric is introduced to our research after fitting, so it is a useful tool for performance evaluation.

Let's think about errors and their profoundness. Some classification problems infer two classes that are equal in importance and quantity. For example, what is in the picture, a fox, or a dog? But most often, in real life, we meet with complicated problems, where errors can be multiple in their nature. For example, finding a fraud banking transaction. It's okay if a normal transaction was predicted as a fraud, you can call the bank user and make sure everything is fine. However, if the model predicted a fraud transaction as a sound one — you and your client could lose a large amount of money.

So, it is essential to select the correct model estimator based on the data and the context of the problem.

§2. Accuracy

The easiest way to determine a model performance is to count the number of correct answers out of a total number. In other words, we can check the **accuracy**. As we've mentioned above, this works well with a balanced dataset, where errors are more or less equally distributed per class.

$$Accuracy = \frac{\text{Correct answers}}{\text{All answers}}$$

Current topic:

Classification performance metrics

Topic depends on:

Introduction to classification

Topic is required for:

Naive Bayes classifier

Decision Trees

Table of contents:

Remember, algorithms are lazy! Suppose you have 100 emails. 98 of them are legit, so there are just two spam emails. If your model predicted that, then the model's accuracy is 98%. Is that a good model though?

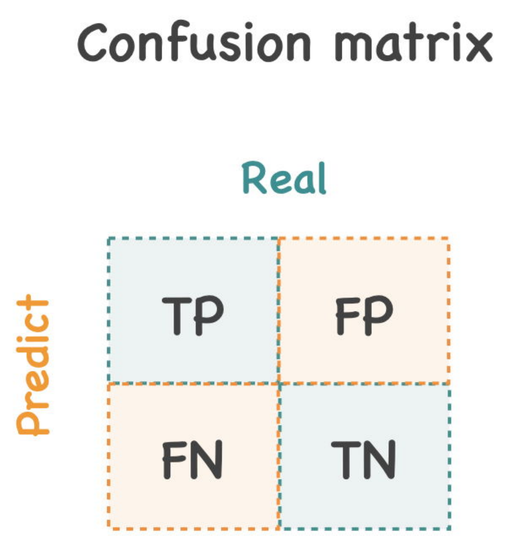
This example shows that accuracy is not a very relevant metric with unbalanced problems or when each class has a different error weight. But if your problem has a balanced sample, then it is obvious that your goal is to achieve accuracy equal to 1. And don't forget about overfitting.

§3. Confusion matrix

We suggest the following notation to define more relevant metrics. For binary classification problems, with classes marked as 0 and 1, we can define the following notions:

- True Positive (TP) — the real "ones" that were correctly predicted as "ones"
- True Negative (TN) — the real "zeros" that were correctly predicted as "zeros"
- False Positive (FP) — the real "zeros" that were predicted as "ones"
- False Negative (FN) — the real "ones" that were predicted as "zeros"

This can be represented as a **confusion matrix**:



As an example, consider the confusion matrix based on the example above:

[redacted]

Accuracy can now be calculated as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Obviously, for multi-classification, a confusion matrix is generalized to any number of classes. It is important to note here that when you get a confusion matrix, your goal is to get large values on the main diagonal, and small values for all other elements.

§4. Precision and recall

These metrics are usually considered together since they are two sides of the same thing. Precision is the proportion of the predicted "ones" that were the real "ones". Recall is the fraction of all "ones" which were detected by a model.

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

In other words, recall demonstrates the algorithm's ability to detect some class in general, and precision displays the ability to distinguish this class from other classes.

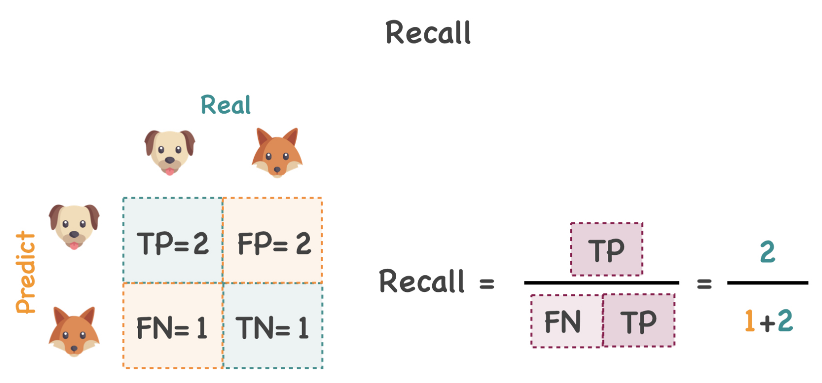
As an appropriate illustration of the above, take a look at the following picture:

[redacted]

Finally, fox-dog wise:

Precision

[redacted]



In contrast to accuracy, it is essential to note that precision and recall do not depend on the ratio of classes, so they apply to unbalanced samples. In a similar manner to accuracy, your goal is to get as close to 1 as possible.

§5. F-score

Precision and recall are two convenient metrics, and there is a way to optimize them. To do this, we can combine them into one, and the result will be the so-called F-score or F-measure.

$$F = \frac{2\,\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

F-measure

[redacted]

The F-score reaches its maximum when the precision and recall results are equal to one. It is close to zero if at least one of the two is close to zero. In our example, the F-score has a value of 0.57, which is somewhere in between (and not very good).

If in your problem, one metric (either precision or recall) is more important than another, we can further complicate the formula. To establish the importance of a particular metric, we can introduce an additional beta parameter and recast the formula in the following form:

$$F_\beta = \frac{\left(1+\beta^2\right)\text{Precision} \times \text{Recall}}{\beta^2\text{Precision} + \text{Recall}}$$

Regardless of the selected beta value, a good model should calculate a result close to 1.

§6. AUC-ROC

The previous metrics, rigorously speaking, referred to a situation where the model predicts a result in zeros or ones (integers, if there are multiple classes). But many models do not output the class itself, but the probabilities of classes — real numbers from zero to one. In this case, the data scientist sets a threshold up to which all results are zeroed, and after which they become ones. However, in real practice, the threshold is often not so obvious. There is a way to evaluate the model without determining a specific threshold — to draw the Receiver Operating Characteristic Curve (ROC curve) and estimate the Area Under Curve (AUC).

This curve is plotted in two coordinates — True Positive Rate (TPR) and False Positive Rate (FPR), which are defined in the above terminology as:

$$TPR = \frac{TP}{TP+FN}$$

$$FPR = \frac{FP}{FP+TN}$$

[redacted]

In our fox-dog terms we would have the following:

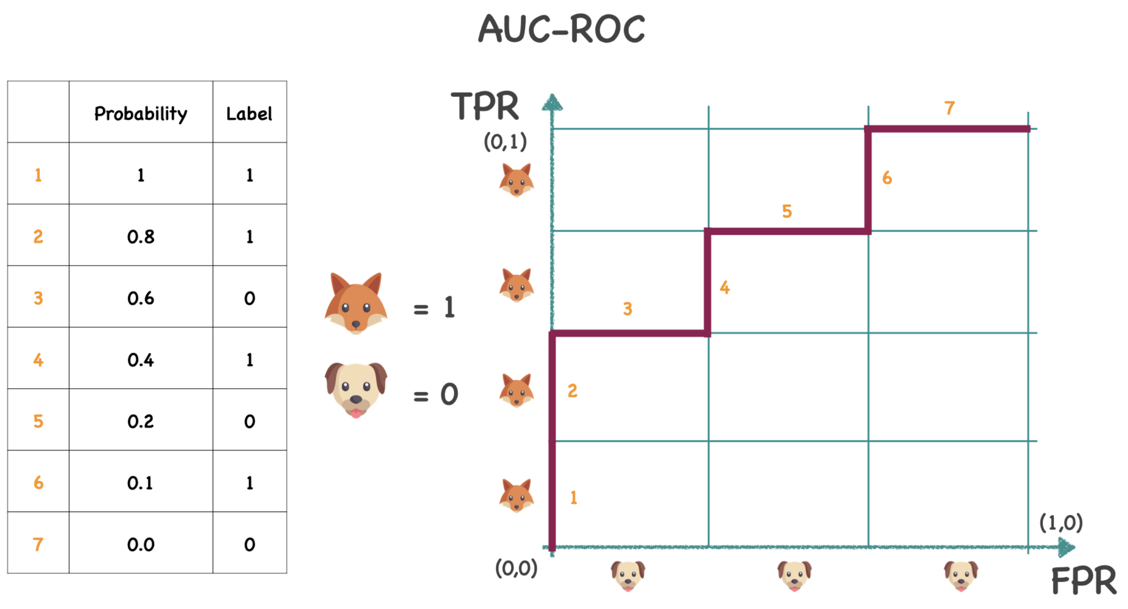
[redacted]

[redacted]

The True Positive Rate, also known as the sensitivity, shows which percentage of class "1" objects we classified correctly. The False Positive Rate, also known as the specificity, shows the proportion of class "0" that were mistakenly classified as "1". If you look closely, you will notice that Sensitivity (TPR) and recall are the same, but they are usually called different terms depending on the task.

Our ideal curve describes the case when the TPR is maximum, and the FPR is minimum, which means that the curve should tend to the point (0,1). Moreover, each point on the graph corresponds to the choice of a certain threshold.

Let's try to build a ROC-Curve for a simple model. Let's say we have some model that predicts not the labels, but the probability that the selected object is a fox. Then let's order the objects in descending order of this probability and give the objects numbers, as shown in the table:



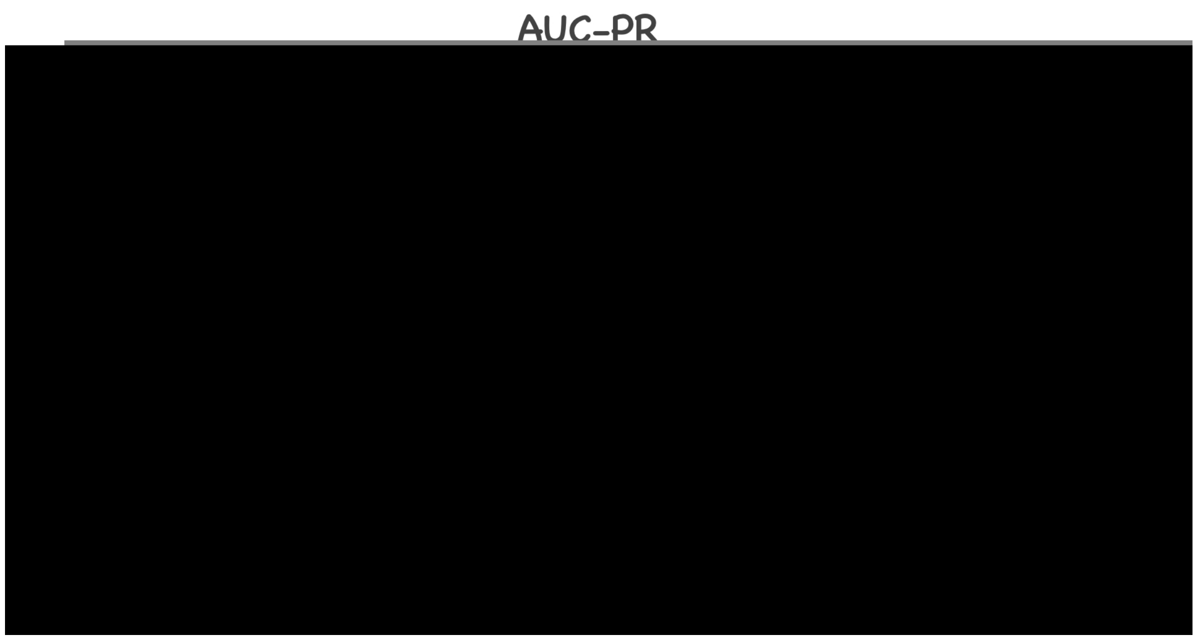
	Probability	Label
1	1	1
2	0.8	1
3	0.6	0
4	0.4	1
5	0.2	0
6	0.1	1
7	0.0	0

Divide the TPR axis by the number of ones (4 foxes), and the FPR axis by the number of zeros (3 dogs). Now we need to build a broken line from point (0, 0) to (1, 1). We would take a step up if the real label is "fox" (1) and step to the right if the real label is "dog" (0). This is how we get a ROC curve.

This metric is entirely different from the previous ones by the numbers you would expect from a good model. If the model predicts good results, then the ROC-AUC will be about 1 (a square with sides of 1), but a bad model will give results close to 0.5, which is equivalent to a random prediction.

§7. AUC-PR

The ROC-AUC metric also has a significant drawback in its performance on unbalanced data. It can give an inadequate evaluation of the algorithm's performance in unbalanced sets, as is the case with accuracy. For these reasons, similarly to the logic above, we can come back to the precision and recall terms and build a curve with these coordinates.



Similar to the example above, we have a set of precision and recall metrics for each sample. We sort it by precision. Next, we start our curve from the upper left corner and set the Precision and Recall points according to the table. As you can see from the reasoning, this approach does not involve each class's quantity, so this approach is more relevant to unbalanced data. Likewise, try to avoid an area equal to 0.5.

§8. Logistic Loss

This feature is also called cross-entropy. This is a prevalent and essential metric derived from the maximum likelihood method, which we will talk about in another topic.

$$logloss = -\frac{1}{l} \cdot \sum_{i=1}^{l} \left(y_i \cdot \log\left(\hat{y}_i\right) + (1 - y_i) \cdot \log\left(1 - \hat{y}_i\right)\right)$$

In the formula above, we use the notation: y_i — the real label, $\hat{y}_i$ — the predicted label.

Let's look at the formula more closely. We can understand that we are looking for maximum accuracy, taking into account the penalty for incorrect predictions. This metric is interesting in the way that we do not want to have high results. A good result will be close to 0.

§9. Summary

In this topic, we have covered several vital metrics for binary classification. Note that many of these can be generalized to multi-classification cases.

Let's once again recall all the main points, pros, and cons of each of the metrics:

- Accuracy is a simple metric, easy to interpret, but unfortunately does not work well with unbalanced samples.
- The confusion matrix is a convenient presentation of the results in TP, FP, TN, and FN terms, which allows us to look more broadly at the predictions, also see the mistakes for each class. Cons — it is not easy to compare two matrices to measure the quality of the models.
- Precision is a measure that shows how many of all predicted samples of a certain class were predicted correctly. It can be used to show how the algorithm works on a specific class but it is not exhaustive.
- Recall shows how many of the real values of a specific class the model could predict. It shows well the quality of the model with small classes, but it is not exhaustive.
- F-measure is a combination of Precision and Recall metrics since tasks often need to be optimized concurrently. Not very intuitive to interpret, difficult to set an additional β—parameter.
- ROC-AUC is the area under the ROC curve. The most popular metric for analyzing model performance as it predicts probabilities rather than classes. It helps to determine the relevant *threshold* for binary classification tasks but doesn't work well with unbalanced samples.
- AUC-PR is the area under the Precision-Recall curve. It is a quality metric that is not related to the number of classes and therefore works well on unbalanced classes.

- LogLoss is a quality metric, which is based on the addition of a penalty for the model's confidence if a prediction is wrong. Easy to optimize and positively to reveal.

Further, in practice, you will have the opportunity to solve problems and remember all the information of this topic.

Report a typo

9 users liked this theory. 6 didn't like it. **What about you?**

Start practicing

Comments (1) Hints (0) Useful links (0) Show discussion