Evaluation Metrics for Machine Learning - Accuracy, Precision, Recall, and F1 Defined

After a data scientist has chosen a target variable - e.g. the “column” in a spreadsheet they wish to predict - and completed the prerequisites of transforming data and building a model, one of the final steps is evaluating the model’s performance.

**Confusion Matrix**

Choosing a performance metric often depends on the business problem being solved. Let’s say you have 100 examples in your dataset, and you’ve fed each one to your model and received a classification. The predicted vs. actual classification can be charted in a table called a confusion matrix.[0](https://pathmind.com/wiki/accuracy-precision-recall-f1#zero)

|  | **Negative (predicted)** | **Positive (predicted)** |
| --- | --- | --- |
| **Negative (actual)** | 98 | 0 |
| **Positive (actual)** | 1 | 1 |

The table above describes an output of negative vs. positive. These two outcomes are the “classes” of each examples. Because there are only two classes, the model used to generate the confusion matrix can be described as a *binary classifier*. (Example of a binary classifier: spam detection. All emails are spam or not spam, just as all food is [a hot dog or not a hot dog](https://itunes.apple.com/us/app/not-hotdog/id1212457521?mt=8).)

To better interpret the table, you can also see it in terms of true positives, true negatives, false positives and false negatives.

|  | **Negative (predicted)** | **Positive (predicted)** |
| --- | --- | --- |
| **Negative (actual)** | true negative | false positive |
| **Positive (actual)** | false negative | true positive |

**Accuracy**

Overall, how often is our model correct?

accuracy formula[1](https://pathmind.com/wiki/accuracy-precision-recall-f1#one)

As a heuristic, or rule of thumb, accuracy can tell us immediately whether a model is being trained correctly and how it may perform generally. However, it does not give detailed information regarding its application to the problem.

The problem with using accuracy as your main performance metric is that it does not do well when you have a severe class imbalance. Let’s use the dataset in the confusion matrix above. Let’s say the negatives are normal transactions and the positives are fraudulent transactions. Accuracy will tell you that you’re right 99% of the time across all classes.

But we can see that for the fraud class (positive), you’re only right 50% of the time, which means you’re going to be losing money. Hell, if you created a hard rule predicting that all transactions were normal, you’d be right 98% of the time. But that wouldn’t be a very smart model, or a very smart evaluation metric. That’s why, when your boss asks you to tell them “how accurate is that model?”, your answer might be: “It’s complicated.”

To give a better answer, we need to know about *precision*, *recall* and *f1 scores*.

[Learn How to Apply AI to Simulations »](https://pathmind.com/)

**Precision**

When the model predicts positive, how often is it correct?

precision formula

Precision helps when the costs of false positives are high. So let’s assume the problem involves the detection of skin cancer. If we have a model that has very low precision, then many patients will be told that they have melanoma, and that will include some misdiagnoses. Lots of extra tests and stress are at stake. When false positives are too high, those who monitor the results will learn to ignore them after being bombarded with false alarms.

**Recall**

recall formula

Recall helps when the cost of false negatives is high. What if we need to detect incoming nuclear missiles? A false negative has devastating consequences. Get it wrong and we all die. When false negatives are frequent, you get hit by the thing you want to avoid. A false negative is when you decide to ignore the sound of a twig breaking in a dark forest, and you get eaten by a bear. (A false positive is staying up all night sleepless in your tent in a cold sweat listening to every shuffle in the forest, only to realize the next morning that those sounds were made by a chipmunk. Not fun.) If you had a model that let in nuclear missiles by mistake, you would want to throw it out. If you had a model that kept you awake all night because *chipmunks*, you would want to throw it out, too. If, like most people, you prefer to not get eaten by the bear, and also not stay up all night worried about chipmunk alarms, then you need to optimize for an evaluation metric that’s a combined measure of precision and recall. Enter the F1 score…

**F1 Score**

f1 formula

F1 is an overall measure of a model’s accuracy that combines precision and recall, in that weird way that addition and multiplication just mix two ingredients to make a separate dish altogether. That is, a good F1 score means that you have low false positives and low false negatives, so you’re correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it’s 1, while the model is a total failure when it’s 0.

Remember: All models are wrong, but some are useful. That is, all models will generate some false negatives, some false positives, and possibly both. While you can tune a model to minimize one or the other, you often face a tradeoff, where a decrease in false negatives leads to an increase in false positives, or vice versa. You’ll need to optimize for the performance metrics that are most useful for your specific problem.