20 Popular Machine Learning Metrics. Part 1: Classification & Regression Evaluation Metrics

An introduction to the most important metrics for evaluating classification, regression, ranking, vision, NLP, and deep learning models.

Shervin Minaee · Follow

Published in Towards Data Science · 11 min read · Oct 28, 2019

Note: This post has two parts. **In the first part** (current post), I will talk about 10 metrics that are widely used for evaluating classification and regression models. And **in the second part** I will talk about 10 metrics which are used to evaluate ranking, computer vision, NLP, and deep learning models.

Introduction

Choosing the right metric is crucial while evaluating machine learning (ML) models. Various metrics are proposed to evaluate ML models in different applications, and I thought it may be helpful to provide a summary of popular metrics in a here, for better understanding of each metric and the applications they can be used for. In some applications looking at a single metric may not give you the whole picture of the problem you are solving, and you may want to use a subset of the metrics discussed in this post to have a concrete evaluation of your models.

Here, I provide a summary of 20 metrics used for evaluating machine learning models. I group these metrics into different categories based on the ML model/application they are mostly used for, and cover the popular metrics used in the following problems:

- *Classification Metrics (accuracy, precision, recall, F1-score, ROC, AUC, …)*
- *Regression Metrics (MSE, MAE)*
- *Ranking Metrics (MRR, DCG, NDCG)*
- *Statistical Metrics (Correlation)*

- *Computer Vision Metrics (PSNR, SSIM, IoU)*
- *NLP Metrics (Perplexity, BLEU score)*
- *Deep Learning Related Metrics (Inception score, Frechet Inception distance)*

There is no need to mention that there are various other metrics used in some applications (FDR, FOR, hit@k, etc.), which I am skipping here.

As a side note, it is also worth mentioning that **metric is different from loss function**. Loss functions are functions that show a measure of the model performance and are used to train a machine learning model (using some kind of optimization), and are usually differentiable in model's parameters. On the other hand, metrics are used to monitor and measure the performance of a model (during training, and test), and do not need to be differentiable. However if for some tasks the performance metric is differentiable, it can be used both as a loss function (perhaps with some regularizations added to it), and a metric, such as MSE.

> *Some of the metrics discussed here may be very trivial, but I decided to cover them for the sake of completeness of this post.* ***So feel free to skip over the ones you are familiar with. Without any further due, let's begin our journey.***

Classification Related Metrics

Classification is one of the most widely used problems in machine learning with various industrial applications, from face recognition, Youtube video categorization, content moderation, medical diagnosis, to text classification, hate speech detection on Twitter.

Models such as support vector machine (SVM), logistic regression, decision trees, random forest, XGboost, convolutional neural network[1], recurrent neural network are some of the most popular classification models[2].

There are various ways to evaluate a classification model, and I am covering some of the most popular ones below.

1- Confusion Matrix (not a metric, but important to know!)

Let's first make sure we know the basic terminologies used in classification problems before going through the detail of each metric. **You can skip this**

section if you are already familiar with the terminologies.

One of the key concept in classification performance is **confusion matrix** (AKA error matrix), which is a tabular visualization of the model predictions versus the ground-truth labels. Each row of confusion matrix represents the instances in a predicted class and each column represents the instances in an actual class.

Let's go through this with an example. Let's assume we are building a binary classification to classify cat images from non-cat images. And let's assume our test set has 1100 images (1000 non-cat images, and 100 cat images), with the below confusion matrix.

		Actual Class	
		Cat	Non-Cat
Predicted Class	Cat	90	60
	Non-Cat	10	940

Figure 1. A sample confusion matrix

- **Out of 100 cat images** the model has predicted 90 of them correctly and has mis-classified 10 of them. If we refer to the "cat" class as positive and the non-cat class as negative class, then 90 samples predicted as cat are considered as as **true-positive**, and the 10 samples predicted as non-cat are **false negative**.
- **Out of 1000 non-cat images**, the model has classified 940 of them correctly, and mis-classified 60 of them. The 940 correctly classified samples are referred as **true-negative**, and those 60 are referred as **false-positive**.

As we can see diagonal elements of this matrix denote the correct prediction for different classes, while the off-diagonal elements denote the samples which are mis-classified.

Now that we have a better understanding of the confusion matrix, let's get into the actual metrics.

2- Classification Accuracy

Classification accuracy is perhaps the simplest metrics one can imagine, and is defined as the **number of correct predictions divided by the total number of predictions,** multiplied by 100. So in the above example, out of 1100 samples 1030 are predicted correctly, resulting in a classification accuracy of:

Classification accuracy= (90+940)/(1000+100)= 1030/1100= 93.6%

3- Precision

There are many cases in which classification accuracy is not a good indicator of your model performance. One of these scenarios is when your class distribution is imbalanced (one class is more frequent than others). In this case, even if you predict all samples as the most frequent class you would get a high accuracy rate, which does not make sense at all (because your model is not learning anything, and is just predicting everything as the top class). For example in our cat vs non-cat classification above, if the model predicts all samples as non-cat, it would result in a 1000/1100= 90.9%.

Therefore we need to look at class specific performance metrics too. Precision is one of such metrics, which is defined as:

Precision= True_Positive/ (True_Positive+ False_Positive)

The precision of Cat and Non-Cat class in above example can be calculated as:

Precision_cat= #samples correctly predicted cat/#samples predicted as cat = **90/(90+60) = 60%**

Precision_NonCat= 940/950= 98.9%

As we can see the model has much higher precision in predicting non-cat samples, versus cats. This is not surprising, as model has seen more examples of non-cat images during training, making it better in classifying that class.

4- Recall

Recall is another important metric, which is defined as the fraction of samples from a class which are correctly predicted by the model. More formally:

Recall= True_Positive/ (True_Positive+ False_Negative)

Therefore, for our example above, the recall rate of cat and non-cat classes can be found as:

Recall_cat= 90/100= 90%

Recall_NonCat= 940/1000= 94%

5- F1 Score

Depending on application, you may want to give higher priority to recall or precision. But there are many applications in which both recall and precision are important. Therefore, it is natural to think of a way to combine these two into a single metric. **One popular metric which combines precision and recall is called F1-score**, which is the harmonic mean of precision and recall defined as:

F1-score= 2*Precision*Recall/(Precision+Recall)

So for our classification example with the confusion matrix in Figure 1, the F1-score can be calculated as:

F1_cat= 2*0.6*0.9/(0.6+0.9)= 72%

The generalized version of F-score is defined as below. As we can see F1-score is special case of F_𝛽 when 𝛽= 1.

$$F_\beta = (1+\beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

Top highlight

It is good to mention that there is always a trade-off between precision and recall of a model, if you want to make the precision too high, you would end up seeing a drop in the recall rate, and vice versa.

6- Sensitivity and Specificity

Sensitivity and specificity are two other popular metrics mostly used in medical and biology related fields, and are defined as:

Sensitivity= Recall= TP/(TP+FN)

Specificity= True Negative Rate= TN/(TN+FP)

7- ROC Curve

The **receiver operating characteristic curve** is plot which shows the performance of a binary classifier as function of its cut-off threshold. **It essentially shows the true positive rate (TPR) against the false positive rate (FPR) for various threshold values. Let's explain more.**

Many of the classification models are probabilistic, i.e. they predict the probability of a sample being a cat. They then compare that output probability with some cut-off threshold and if it is larger than the threshold they predict its label as cat, otherwise as non-cat. As an example your model may predict the below probabilities for 4 sample images: **[0.45, 0.6, 0.7, 0.3]**. Then depending on the threshold values below, you will get different labels:

cut-off= 0.5: predicted-labels= [0,1,1,0] (default threshold)
cut-off= 0.2: predicted-labels= [1,1,1,1]
cut-off= 0.8: predicted-labels= [0,0,0,0]

As you can see by varying the threshold values, we will get completely different labels. And as you can imagine each of these scenarios would result in a different precision and recall (as well as TPR, FPR) rates.

ROC curve essentially finds out the TPR and FPR for various threshold values and plots TPR against the FPR. A sample ROC curve is shown in Figure 2.

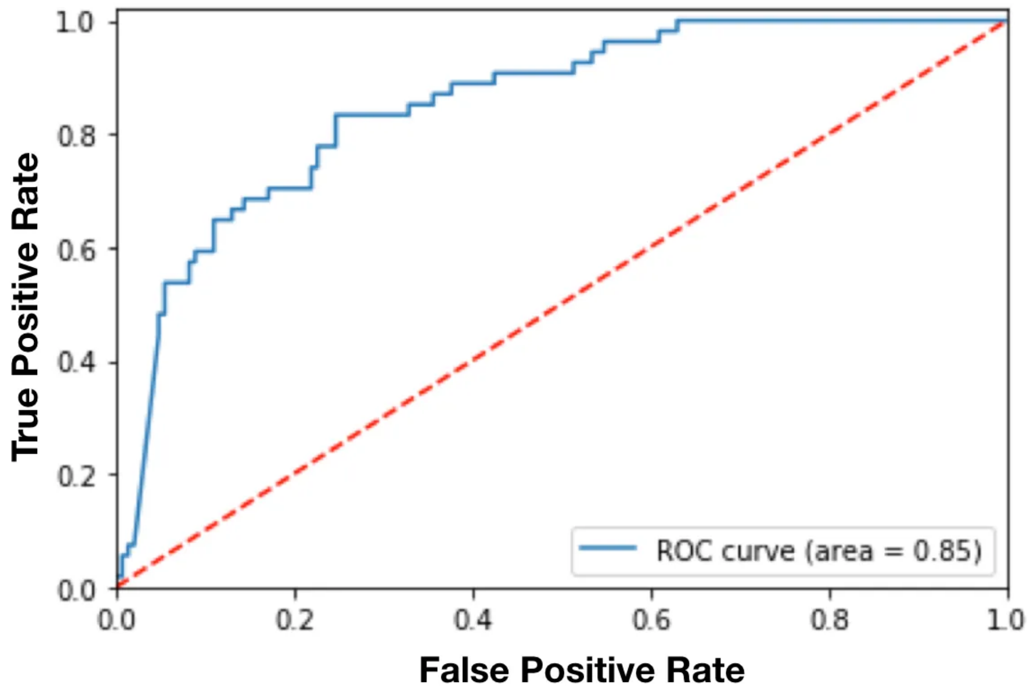

Figure 2. A sample ROC curve.

As we can see from this example, the lower the cut-off threshold on positive class, the more samples predicted as positive class, i.e. higher true positive rate (recall) and also higher false positive rate (corresponding to the right side of this curve). Therefore, there is a trade-off between how high the recall could be versus how much we want to bound the error (FPR).

ROC curve is a popular curve to look at overall model performance and pick a good cut-off threshold for the model.

8- AUC

The **area under the curve** (AUC), is an aggregated measure of performance of a binary classifier on all possible threshold values (and therefore it is threshold invariant).

AUC calculates the area under the ROC curve, and therefore it is between 0 and 1. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.

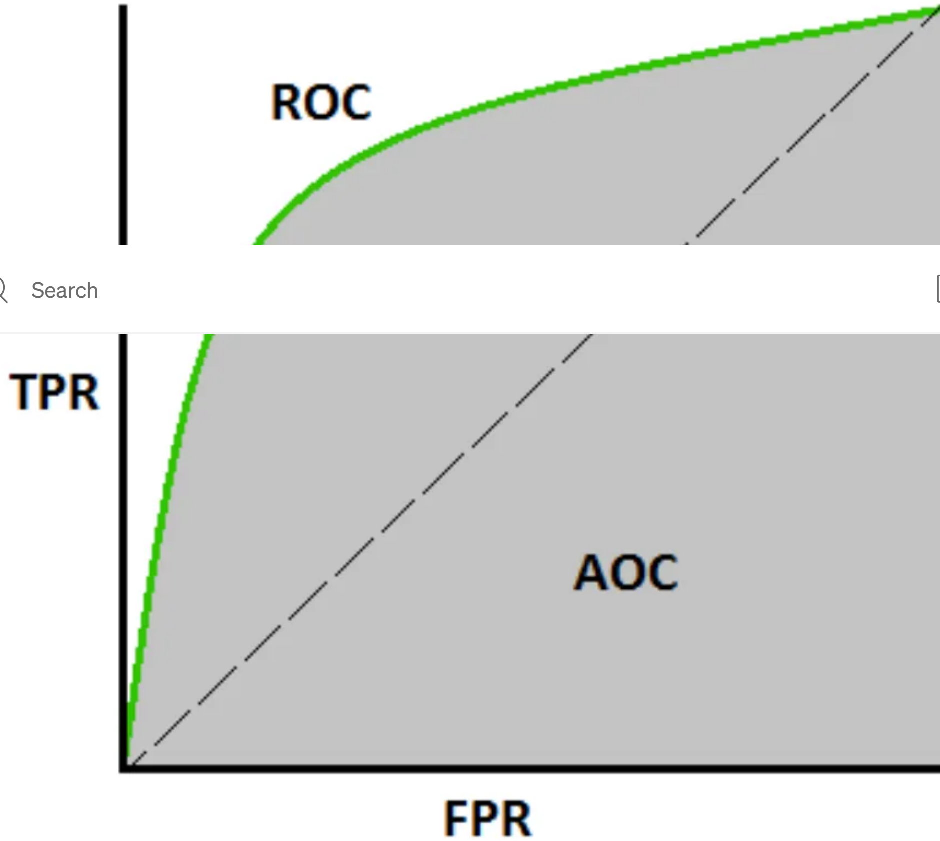

Figure 3. The gray area in this ROC curve denotes the AUC.

On high-level, the higher the AUC of a model the better it is. But sometimes threshold independent measure is not what you want, e.g. you may care about your model recall and require that to be higher than 99% (while it has a reasonable precision or FPR). In that case, you may want to tune your model threshold such that it meets your minimum requirement on those metrics (and you may not care even if you model AUC is not too high).

Therefore in order to decide how to evaluate your classification model performance, perhaps you want to have a good understanding of the business/problem requirement and the impact of low recall vs. low precision, and decide what metric to optimize for.

From a practical standpoint, a classification model which outputs probabilities is preferred over a single label output, as it provides the flexibility of tuning the threshold such that it meets your minimum recall/precision requirements. Not all models provide this nice probabilistic outputs though, e.g. SVM does not provide a simple probability as an output (although it provides margin which can be used to tune the decision, but it is not as straightforward and interpretable as having output probabilities).

Regression Related Metrics

Regression models are another family of machine learning and statistical models, which are used to predict a continuous target values[3]. They have a wide range of applications, from house price prediction, E-commerce pricing systems, weather forecasting, stock market prediction, to image super resolution, feature learning via auto-encoders, and image compression.

Models such as linear regression, random forest, XGboost, convolutional neural network, recurrent neural network are some of the most popular regression models.

Metrics used to evaluate these models should be able to work on a set of continuous values (with infinite cardinality), and are therefore slightly different from classification metrics.

9- MSE

"Mean squared error" is perhaps the most popular metric used for regression problems. It essentially finds the average squared error between the predicted and actual values.

Let's assume we have a regression model which predicts the price of houses in Seattle area (show them with $\hat{y}_i$), and let's say for each house we also have the actual price the house was sold for (denoted with y_i). Then the MSE can be calculated as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Sometimes people use RMSE to have a metric with scale as the target values, which is essentially the square root of MSE.

Looking at house pricing prediction, RMSE essentially shows what is the average deviation in your model predicted house prices from the target values (the prices the houses are sold for).

10- MAE

Mean absolute error (or mean absolute deviation) is another metric which finds the average absolute distance between the predicted and target values. MAE is define as below:

MAE is known to be more robust to the outliers than MSE. The main reason being that in MSE by squaring the errors, the outliers (which usually have higher errors than other samples) get more attention and dominance in the final error and impacting the model parameters.

It is also worth mentioning that there is a nice maximum likelihood (MLE) interpretation behind MSE and MAE metrics. If we assume a linear dependence between features and targets, then MSE and MAE correspond to the MLE on the model parameters by assuming Gaussian and Laplace priors on the model errors respectively.

Inlier Ratio Metric:

There is also another metric for evaluating regression models, called inlier ratio, which is essentially the percentage of data points which are predicted with an error less than a margin. This metric is mainly used in RANSAC[4] model and its extensions (a family of robust estimation models).

Summary

In this post, we provided an introduction to some of the 10 popular ML metrics used for evaluating the performance of classification and regression models. In the next part of this post, we are going to provide an introduction to **10 more advanced metrics used for assessing the performance of Ranking, Statistical, Computer Vision, NLP, and Deep Learning Models.**

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. "**Deep learning**",

MIT press, 2016.

[2] Christopher M. Bishop, “**Pattern recognition and machine learning”,** springer, 2006.

[3] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. “**The elements of statistical learning”,** Springer series in statistics, 2001.

[4] Tilo Strutz, “**Data fitting and uncertainty: A practical introduction to weighted least squares and beyond”,** Vieweg and Teubner, 2010.

Machine Learning · Data Science · AI · Metrics · Tutorial

Written by Shervin Minaee

Follow

597 Followers · Writer for Towards Data Science

ML Team Lead at Snapchat — PhD from NYU. Personal page: https://sites.google.com/site/shervinminaee/home Google-Scholar: https://bit.ly/36ubOCf

More from Shervin Minaee and Towards Data Science