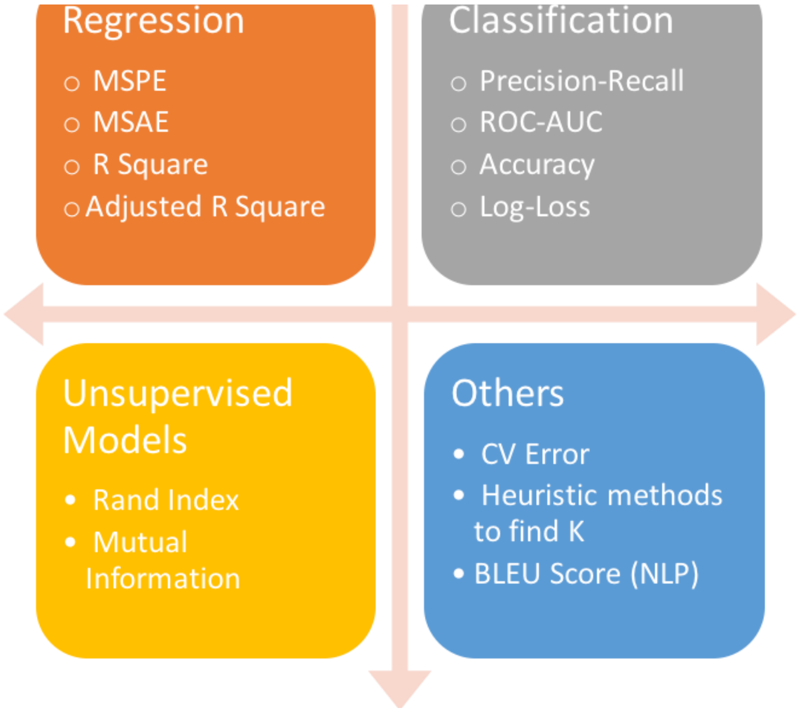
Choosing the Right Metric for Evaluating Machine Learning Models — Part 2

Alvira Swalin Follow
May 2, 2018 · 8 min read ★

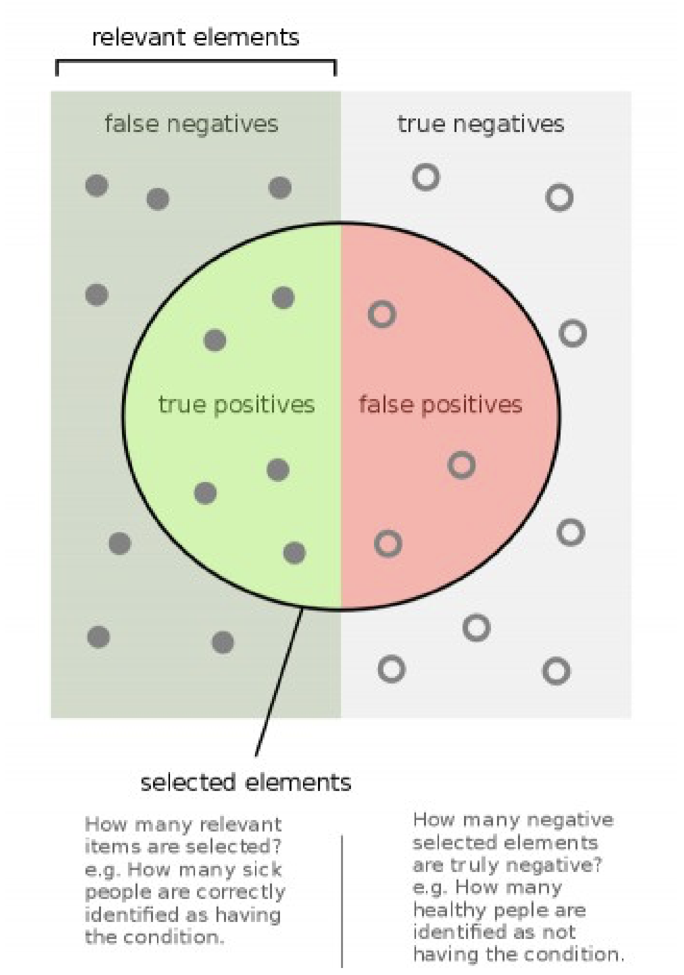
Second part of the series focussing on classification metrics

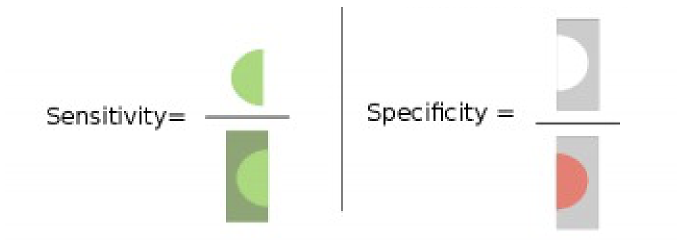
In the first blog, we discussed some important metrics used in regression, their pros and cons, and use cases. This part will focus on commonly used metrics in classification, why should we prefer some over others with context.



Definitions

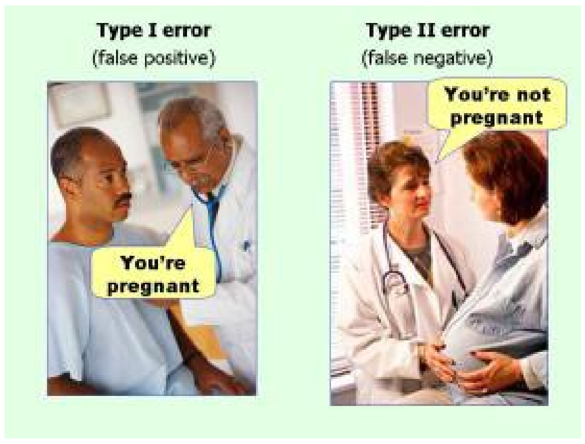
Let's first understand the basic terminology used in classification problems before going through the pros and cons of each method. You can skip this section if you are already familiar with the terminology.


relevant elements
false negatives
true negatives
true positives
false positives
selected elements
How many relevant items are selected? e.g. How many sick people are correctly identified as having the condition.
How many negative selected elements are truly negative? e.g. How many healthy peple are identified as not having the condition.



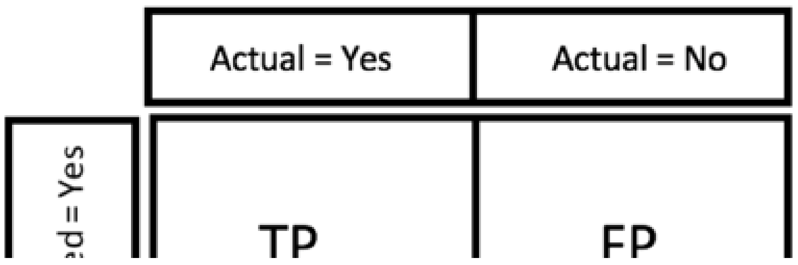
Source of Image: Wikipedia

- **Recall or Sensitivity or TPR (True Positive Rate)**: Number of items correctly identified as positive out of total true positives- TP/(TP+FN)
- **Specificity or TNR (True Negative Rate):** Number of items correctly identified as negative out of total negatives- TN/(TN+FP)
- **Precision:** Number of items correctly identified as positive out of total items identified as positive- TP/(TP+FP)
- **False Positive Rate or Type I Error:** Number of items wrongly identified as positive out of total true negatives- FP/(FP+TN)
- **False Negative Rate or Type II Error:** Number of items wrongly identified as negative out of total true positives- FN/(FN+TP)

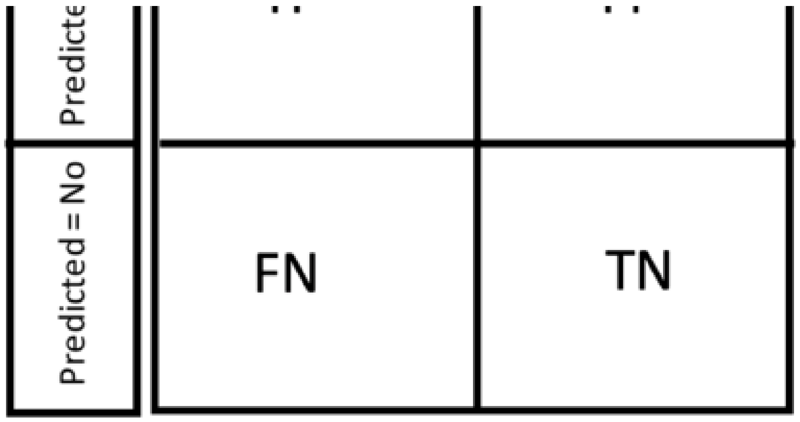


Source of Image: Effect Size FAQs by Paul Ellis

- **Confusion Matrix**



	Actual = Yes	Actual = No
ed = Yes	TP	FP

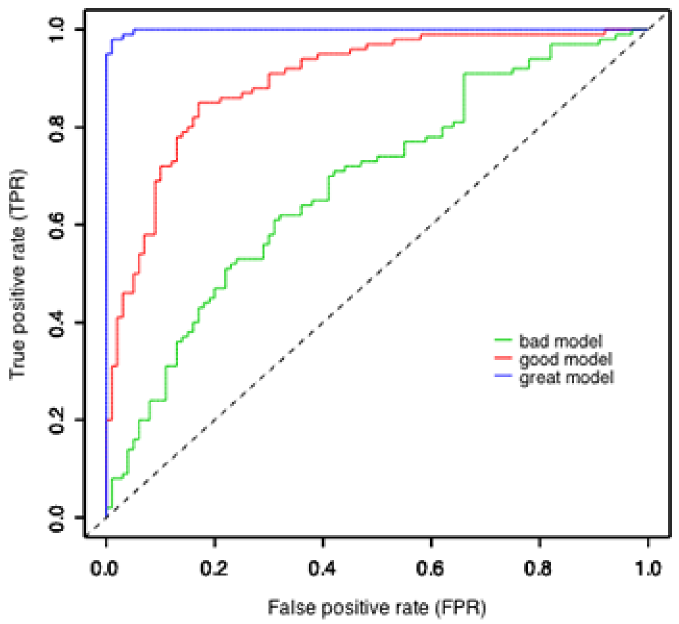

Predicte
Predicted = No
FN
TN

- **F1 Score**: It is a harmonic mean of precision and recall given by-
  F1 = 2*Precision*Recall/(Precision + Recall)

- **Accuracy:** Percentage of total items classified correctly- (TP+TN)/(N+P)

ROC-AUC Score

The probabilistic interpretation of ROC-AUC score is that if you randomly choose a positive case and a negative case, the probability that the positive case outranks the negative case according to the classifier is given by the AUC. Here, rank is determined according to order by predicted values.



Source of Image: UNC Lecture

> *Mathematically, it is calculated by area under curve of sensitivity (TPR) vs. FPR(1-specificity). Ideally, we would like to have high sensitivity & high specificity, but in real-world scenarios, there is always a tradeoff between sensitivity & specificity.*

Some important characteristics of ROC-AUC are-

- The value can range from 0 to 1. However auc score of a random classifier for balanced data is 0.5
- ROC-AUC score is independent of the threshold set for classification because it only considers the rank of each prediction and not its absolute value. The same is not true for F1 score which needs a threshold value in case of probabilities output

Log-Loss

Log-loss is a measurement of accuracy that incorporates the idea of probabilistic confidence given by following expression for binary class:

$$-(y\log(p) + (1-y)\log(1-p))$$

It takes into account the uncertainty of your prediction based on how much it varies from the actual label. In the worst case, let's say you predicted 0.5 for all the observations. So log-loss will become -log(0.5) = 0.69. Hence, we can say that anything above 0.6 is a very poor model considering the actual probabilities.

Case 1

Comparison of Log-loss with ROC & F1

S.No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Actual (Balanced)	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
Predicted (Model 1)	0.1	0.1	0.1	0.1	0.1	0.1	0.6	0.6	0.5	0.5	0.9	0.9	0.9	0.9	0.9	0.9
Predicted (Model 2)	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.7	0.7	0.7	0.8	0.8	0.8	0.8

Consider Case 1 (Balanced Data), it looks like model 1 is doing a better job in predicting the absolute probabilities whereas model 2 is working best in ranking observations according to their true labels. Let's verify with the actual score:

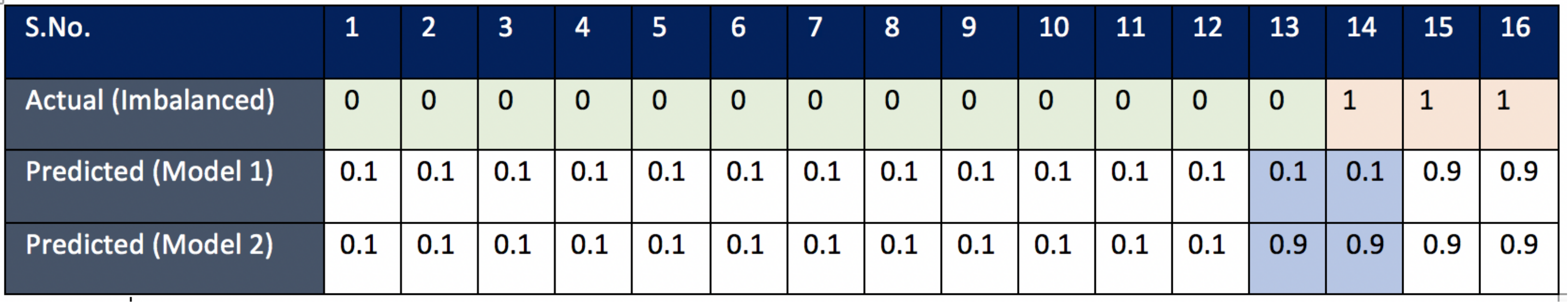
	F1 (threshold=0.5)	F1 (Threshold which maximize score)	ROC-AUC	Log-Loss
Model 1	0.88	0.88	0.94	0.28
Model 2	0.67	1	1	0.6

If you consider log-loss, Model 2 is worst giving a high value of log-loss because the absolute probabilities have big difference from actual labels. But this is in complete disagreement with F1 & AUC score, according to which Model 2 has 100% accuracy. Also, you would like to note that with different thresholds, F1 score is changing, and preferring model 1 over model 2 for default threshold of 0.5.

> *Inferences drawn from the above example (balanced):*
> *- If you care for absolute probabilistic difference, go with log-loss*
> *- If you care only for the final class prediction and you don't want to tune threshold, go with AUC score*
> *-F1 score is sensitive to threshold and you would want to tune it first before comparing the models*

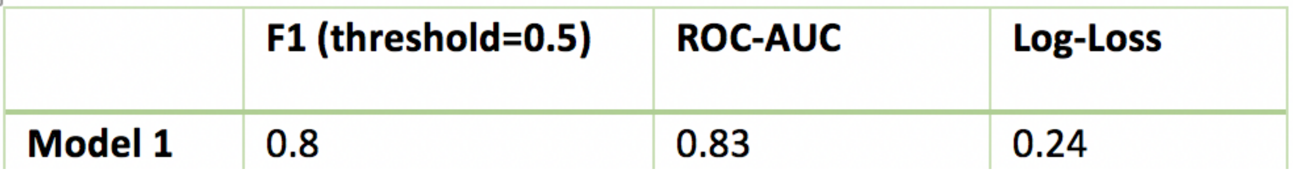
Case 2

How each of them deals with class imbalance?



S.No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Actual (Imbalanced)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Predicted (Model 1)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9
Predicted (Model 2)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9

The only difference in the two models is their prediction for observation 13 & 14. Model 1 is doing a better job in classifying observation 13 (label 0) whereas Model 2 is doing better in classifying observation 14 (label 1). The goal is to see which model actually captures the difference in classifying the imbalanced class better (class with few observations, here it is label 1). In problems like fraud detection/spam mail detection, where positive labels are few, we would like our model to predict positive classes correctly and hence we will sometime prefer those model who are able to classify these positive labels



	F1 (threshold=0.5)	ROC-AUC	Log-Loss
Model 1	0.8	0.83	0.24

Model 2	0.86	0.96	0.24

> *Clearly log-loss is failing in this case because according to log-loss both the models are performing equally. This is because log-loss function is symmetric and does not differentiate between classes .*

Both F1 score and ROC-AUC score is doing better in preferring model 2 over model 1. So we can use both these methods for class imbalance. But we will have to dig further to see how differently they treat class imbalance.

S.No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Actual (Imbalanced – few positive)	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Predicted (Model 1)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9
Predicted (Model 2)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9
Actual (Imbalanced – few negative)	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
Predicted (Model 3)	0.1	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
Predicted (Model 4)	0.1	0.1	0.1	0.1	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9

In the previous example, we saw that there were few positive labels. In the second example, there were few negative labels. Let's see how F1 score & ROC-AUC differentiate between these two cases.

	F1 (threshold=0.5)	**ROC-AUC**	**Log-Loss**
Model 1	0.8	0.83	0.24
Model 2	0.86	0.96	0.24
Model 3	0.963	0.83	0.24

Model 4	0. 96	0.96	0.24

ROC-AUC score handled the case of few negative labels in the same way as it handled the case of few positive labels. An interesting thing to note here is that F1 score is pretty much same for both Model 3 & Model 4 because positive labels are large in number and it cares only for the misclassification of positive labels.

> *Inferences drawn from above example:*
> *- If you care for a class which is smaller in number independent of the fact whether it is positive or negative, go for ROC-AUC score.*

When will you prefer F1 over ROC-AUC?

When you have a small positive class, then F1 score makes more sense. This is the common problem in fraud detection where positive labels are few. We can understand this statement with the following example.

```python
from sklearn import metrics
import numpy as np

y_true = np.concatenate((np.ones(100), np.zeros(900)))

a = np.random.uniform(0.5,1, 5)
b = np.random.uniform(0,0.5, 995)
y_pred1 = np.concatenate((a,b))

a = np.random.uniform(0.5,1, 90)
b = np.random.uniform(0,0.5, 910)
y_pred2 = np.concatenate((a,b))

print(metrics.f1_score(y_true, y_pred1>0.5))
print(metrics.f1_score(y_true, y_pred2>0.5))

print(metrics.roc_auc_score(y_true, y_pred1))
print(metrics.roc_auc_score(y_true, y_pred2))
```

ROC-AUC vs F1 hosted with ❤️ by **GitHub** view raw

We can see that model (1) predicts 5 positives out of 100 true positives in a dataset of size 10K observations, while another model (2) predicts 90 positives out of 100 true positives. Clearly, model (2) is doing a much better job than model (1) in this case. Let's see if both F1 score & ROC-AUC score are able to capture that difference

F1 score for model (1) = 2*(1)*(0.1)/1.1 = 0.095

F1 score for model (2) = 2*(1)*(0.9)/1.9 = 0.947

Yes, the difference in F1 score reflects the model performance.

ROC-AUC for model (1) = 0.5

ROC-AUC for model (2) = 0.93

ROC-AUC gives a decent score to model 1 as well which is nota good indicator of its performance. Hence we should be careful while picking roc-auc for imbalanced datasets.

Which metric should you use for multi-classification?

We have further three types of non-binary classification:

- **Multi-Class:** classification task with more than two classes such that the input is to be classified into one, and only one of these classes. Example: classify a set of images of fruits into any one of these categories — apples, bananas, and oranges.
- **Multi-labels:** classifying a sample into a set of target labels. Example: tagging a blog into one or more topics like technology, religion, politics etc. Labels are isolated and their relations are not considered important.
- **Hierarchical:** each category can be grouped together with similar categories, creating meta-classes, which in turn can be grouped again until we reach the root level (set containing all data). Examples include text classification & species classification. For more details, refer this blog.

In this blog, we will cover only the first category.

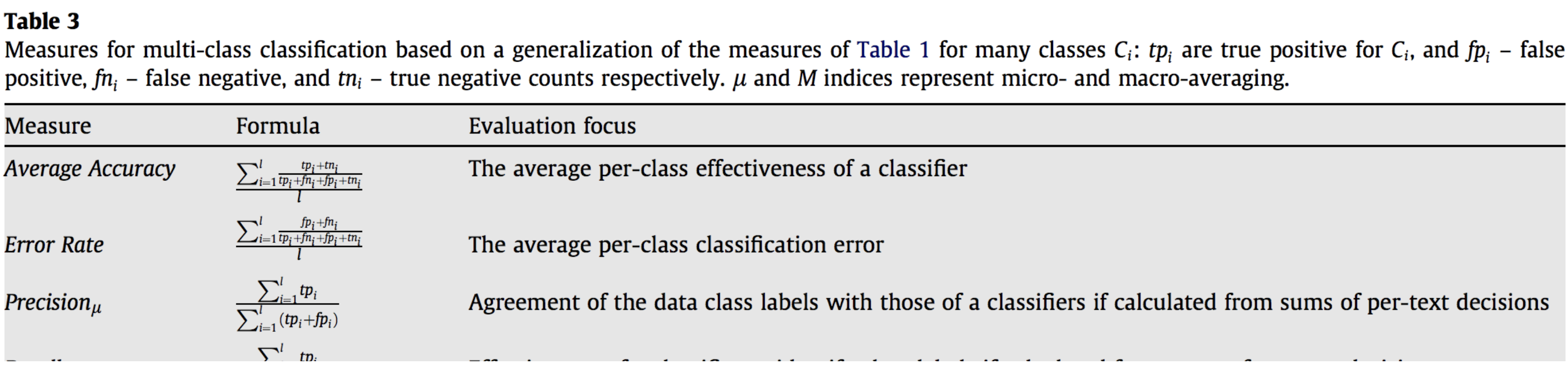
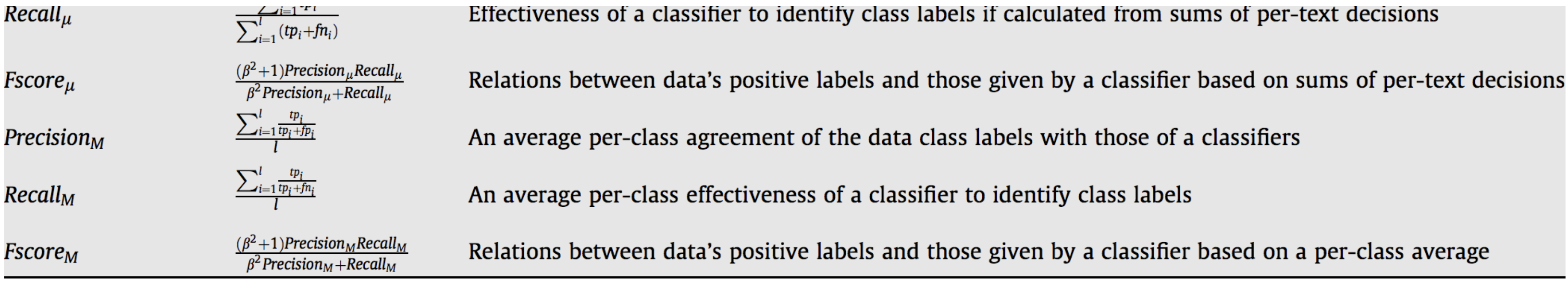


Table 3
Measures for multi-class classification based on a generalization of the measures of Table 1 for many classes C_i: tp_i are true positive for C_i, and fp_i – false positive, fn_i – false negative, and tn_i – true negative counts respectively. μ and M indices represent micro- and macro-averaging.

Measure	Formula	Evaluation focus
Average Accuracy	$\frac{\sum_{i=1}^{l} \frac{tp_i+tn_i}{tp_i+fn_i+fp_i+tn_i}}{l}$	The average per-class effectiveness of a classifier
Error Rate	$\frac{\sum_{i=1}^{l} \frac{fp_i+fn_i}{tp_i+fn_i+fp_i+tn_i}}{l}$	The average per-class classification error
Precision$_\mu$	$\frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l} (tp_i+fp_i)}$	Agreement of the data class labels with those of a classifiers if calculated from sums of per-text decisions



$Recall_{\mu}$	$\sum_{i=1}^{l}(tp_i+fn_i)$	Effectiveness of a classifier to identify class labels if calculated from sums of per-text decisions
$Fscore_{\mu}$	$\frac{(\beta^2+1)Precision_{\mu}Recall_{\mu}}{\beta^2 Precision_{\mu}+Recall_{\mu}}$	Relations between data's positive labels and those given by a classifier based on sums of per-text decisions
$Precision_{M}$	$\frac{\sum_{i=1}^{l}\frac{tp_i}{tp_i+fp_i}}{l}$	An average per-class agreement of the data class labels with those of a classifiers
$Recall_{M}$	$\frac{\sum_{i=1}^{l}\frac{tp_i}{tp_i+fn_i}}{l}$	An average per-class effectiveness of a classifier to identify class labels
$Fscore_{M}$	$\frac{(\beta^2+1)Precision_{M}Recall_{M}}{\beta^2 Precision_{M}+Recall_{M}}$	Relations between data's positive labels and those given by a classifier based on a per-class average

https://www.sciencedirect.com/science/article/pii/S0306457309000259

As you can see in the above table, we have broadly two types of metrics- micro-average & macro-average, we will discuss the pros and cons of each. Most commonly used metrics for multi-classes are F1 score, Average Accuracy, Log-loss. There is yet no well-developed ROC-AUC score for multi-class.

Log-loss for multi-class is defined as:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij})$$

Where,

N	*No of Rows in Test set*
M	*No of Fault Delivery Classes*
Y_{ij}	*1 if observation belongs to Class j; else 0*
P_{ij}	*Predicted Probability that observation belong to Class j*

> *- In Micro-average method, you sum up the individual true positives, false positives, and false negatives of the system for different sets and then apply them to get the statistics.*
> *- In Macro-average, you take the average of the precision and recall of the system on different sets*

Micro-average is preferable if there is a class imbalance problem.

. . .

In the third part, I will be focussing on metrics used in unsupervised learning problems where it is even harder to quantify the correctness of a model without the presence of target variables. Stay tuned! In the meantime, check out my other blogs here!

. . .

About Me: Masters Candidate in Data Science at USF. Interested in working with cross-functional groups to derive insights from data, and apply Machine Learning knowledge to solve complicated data science problems.
LinkedIn: www.linkedin.com/in/alvira-swalin

References

1. https://classeval.wordpress.com/simulation-analysis/roc-and-precision-recall-with-imbalanced-datasets/
2. https://en.wikipedia.org/wiki/Precision_and_recall
3. https://www.sciencedirect.com/science/article/pii/S0306457309000259
4. https://stats.stackexchange.com/questions/11859/what-is-the-difference-between-multiclass-and-multilabel-problem
5. https://datascience.stackexchange.com/questions/15989/micro-average-vs-macro-average-performance-in-a-multiclass-classification-settin/16001

Machine Learning Metrics Classification Towards Data Science