Classification metrics

Carl McBride Ellis

carl.mcbride@u-tad.com

Performance metrics

Regression

- RMSE if using the L2 loss
- MAE if using the L1 loss

classification (probabilistic) (strictly proper scoring rules)

- log-loss
- Brier score

classification (dichotomized)

- confusion matrix
- accuracy ✖
- F_1 score
- Matthews correlation coefficient* (MCC) ✔

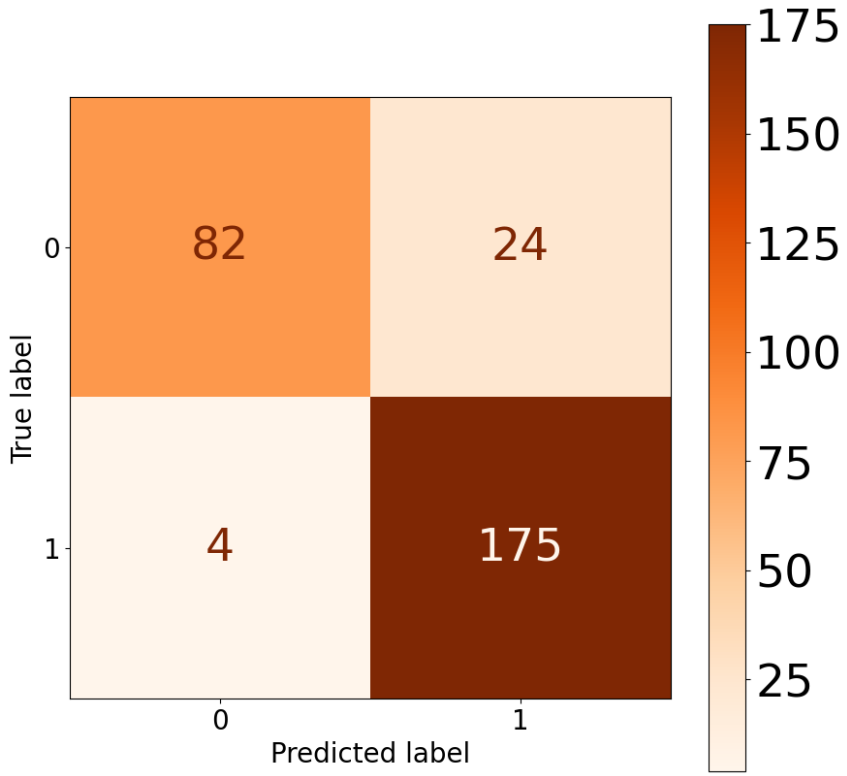
(* unlike scikit-learn and pyTorch Keras does not implement the MCC metric)

Binary classification: there are 4 possible outcomes:

false positive	(FP)	$y = 0, \hat{y} = 1$
false negative	(FN)	$y = 1, \hat{y} = 0$
true positive	(TP)	$y = 1, \hat{y} = 1$
true negative	(TN)	$y = 0, \hat{y} = 0$

Remember that the true value is y and our predicted value is $\hat{y}$ and that the positive class is 1, thus the 'negative' class is 0

A confusion matrix or "contingency table":



Notebook: "Titanic: In all the confusion..."

Accuracy score

- the most common classification metric
- the easiest to understand
- the WORST classification metric to use (why?)

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{\text{correct classifications}}{\text{all classifications}}$$

`sklearn.metrics.accuracy_score`

Precision and *recall*

- $\text{precision} = \frac{TP}{TP+FP}$ (when we are interested in few false positives)
- $\text{recall} = \frac{TP}{TP+FN}$ (when we are interested in few false negatives)

F_1 score (when the precision is as important as the recall)
is the harmonic mean of the precision and recall:

- $F_1 = 2\frac{\text{precision} \times \text{recall}}{\text{precision}+\text{recall}}$
- `from sklearn.metrics import f1_score`

(Paper: *"Evaluation Methods for Ordinal Classification"*)

Matthews correlation coefficient

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

`sklearn.metrics.matthews_corrcoef`

(paper: "The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification")

Baseline scores:
Dichotomized

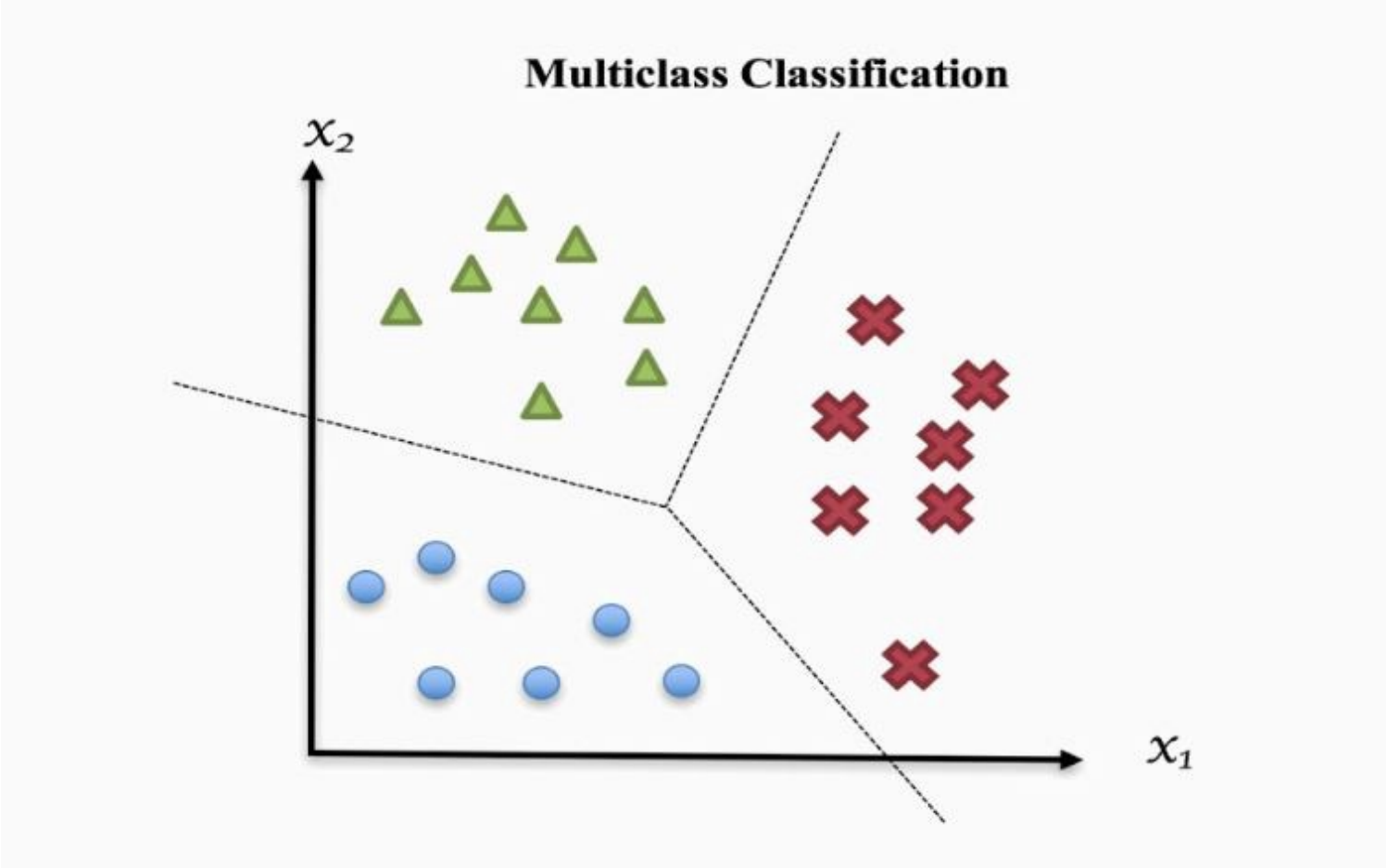
```
from sklearn.metrics import accuracy_score as metric

y_baseline = np.zeros( len(y_validation) )
metric(y_validation, y_baseline)
```

Probabilistic

```
from sklearn.metrics import log_loss as metric

y_train_mean = np.full(len(y_validation),np.mean(y_train))
metric(y_validation, y_train_mean)
```



One Vs Rest (OVR)

Multiclass metrics

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
$Recall_\mu$	$\frac{\sum_{i=1}^{l} tp_i}{\sum_{i=1}^{l} (tp_i+fn_i)}$	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

(Paper: *"A systematic analysis of performance measures for classification tasks"*)