Classification Metrics

Sunday 14 April 2024 7:10 AM

① Accuracy

Actual	Predicted	
1	0	✗
0	1	✗
1	1	✓
1	0	✗
0	1	✗
1	0	✗
0	1	✗
0	0	✓
0	0	✓

same logic for binary & multiclass classification problem

Accuracy score = $\frac{\text{correctly predicted}}{\text{total \# of points}}$ [0,1] ↑ better

$= \frac{3}{9} = \frac{1}{3}$

= 33%

* Same logic for multi-classification

* How much accuracy is good?

→ Depends on the problem we are dealing with & the implications / consequences the inaccuracy will lead to.

90% accuracy might sound good while predicting if a customer will renew subscription based on the current month's activity but the same accuracy when predicting the presence of cancer by looking at a x-ray is very bad as it means for every 100 patients there a chance 10 people may die.

* Accuracy doesn't tell us the nature of the mistake.

	actual	predicted	
i.e.	1	0	→ How many
	0	1	→ How many

* Always ask 3 questions b4 selection of measure in classification problems.

1) Dataset — Balanced / Imbalanced

2) Problem — Binary / Multi-class

3) Which error are we more concerned about?

② Confusion matrix

		Predicted 1 (+)	Predicted 0 (−)
Actual	(+) 1	TP ✓	FN [Type II]
	(−) 0	FP [Type I]	TN ✓

* We can extract the accuracy score from the confusion matrix

accuracy score = $\frac{TP+TN}{TP+TN+FP+FN}$

ICR = 1 − accuracy score

```
from sklearn.metrics import confusion_matrix, classification_report
```

confusion_matrix(true, observed)

Note: Not choosing the right evaluation metrics can mislead your judgement

Accuracy vs Imbalanced dataset

→ Assume we are building a terrorist classification system

		Predicted 1	Predicted 0
Actual	1	0	1
	0	0	99999

accuracy = $\frac{99999}{99999+1}$ = 99.99%

* Accuracy can be misleading when dealing with imbalanced dataset.

It is obvious if a model is trained on Balanced Dataset, most chances predictions will also be balanced.

Actual # of Positives ≈ Predicted # of Positives

Actual # of Negatives ≈ Predicted # of Negatives

TP ≈ TN

FP ≈ FN

③ Precision [What proportion of predicted positives is truly positive]

		Predicted 1	Predicted 0
Actual	1	TP	FN
	0	FP	TN

precision = $\frac{TP}{TP+FP}$ [0,1] ↑ better

```
from sklearn.metrics import recall_score, precision_score, f1_score
```

average = None

* There is a tradeoff b/w Precision & Recall.

④ Recall [What proportion of actual positives is correctly classified] [True Positive Rate] [sensitivity]

		Predicted 1	Predicted 0
Actual	1	TP	FN
	0	FP	TN

~~1000~~/~~1200~~ ~~1000~~/~~1500~~

Recall = $\frac{TP}{TP+FN}$ [0,1] ↑ better

* eg. cancer

high threshold	low threshold
TP↓ FP↓↓	TP↑ FP↑↑
TN↑ FN↑↑	TN↓ FN↓↓
P↑	P↓
R↓	R↑

Choosing threshold
- w.r.t Precision → Precision & Recall vs Threshold (y) (x)
- w.r.t. recall → Precision vs Recall curve (y) (x)
- w.r.t. f1-score → F1 score vs threshold (y) (x)

we can use AUPRC for model comparison also. ↑ better

* Whether to use precision [FP] [Type I] or Recall [FN] [Type II] depends on if the Type I or Type II error has more implications / consequences.

⑤ F1 score [not sure if Type I or Type II error is more problematic? use F1 score]

F1 score = $\frac{2PR}{P+R}$ → Penalises the lower value [Harmonic mean]

* In Binary classification for Precision & Recall, we discuss in terms of 1/positive

Multi-class Precision & Recall

Actual \ Predicted	setosa	versicolor	virginica	Actual Total
setosa	25	5	10	40
versicolor	0	30	4	34
virginica	4	10	20	34
Predicted total	29	45	34	108

$F1_s = 2P_sR_s / (P_s + R_s)$

$F1_{ver} = 2P_{ver}R_{ver} / (P_{ver} + R_{ver})$

$F1_{vir} = 2P_{vir}R_{vir} / (P_{vir} + R_{vir})$

'macro'

'weighted'

$P_s = 25/29$ $P_{ver} = 30/45$ $P_{vir} = 20/34$

[balanced] 'macro' precision: $\frac{P_s + P_{ver} + P_{vir}}{3}$

[imbalanced] 'weighted' precision: $\frac{40}{108}P_s + \frac{34}{108}P_{ver} + \frac{34}{108}P_{vir}$

$R_s = 25/40$ $R_{ver} = 30/34$ $R_{vir} = 20/34$

'macro' recall [balanced]: $\frac{R_s + R_{ver} + R_{vir}}{3}$

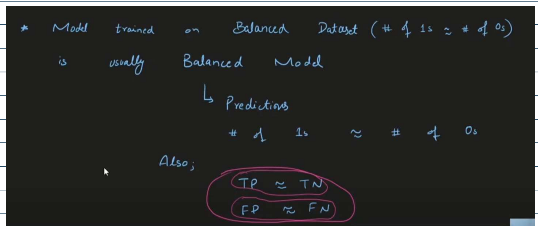
'weighted' recall [imbalanced]: $\frac{40}{108}R_s + \frac{34}{108}R_{ver} + \frac{34}{108}R_{vir}$

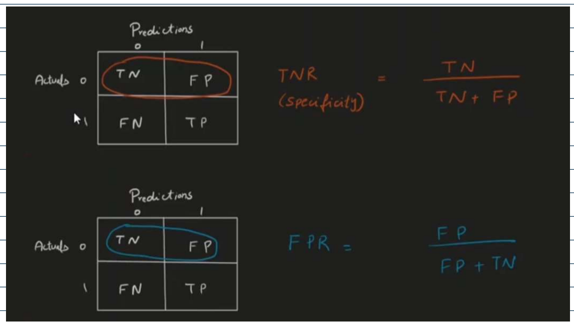
Recommendation :- Always label ⇒ majority_class → 0_class ✓

minority_class → 1-class ✓

↓

So that, will not confuse in selecting the performance measure



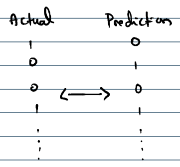


Matthews correlation coefficient

→ Symmetric metric for imbalanced dataset

→ measure of the quality of the classification

→ MCC is a measure of association for two binary variables



Range [-1, +1]

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

→ caters all counts i.e. TP, TN, FP, FN
more symmetric than f1-score

→ Just to make output within [-1, +1]
Normalization constant, always +ve

FP = 0, FN = 0 → perfect +1 — Perfect prediction

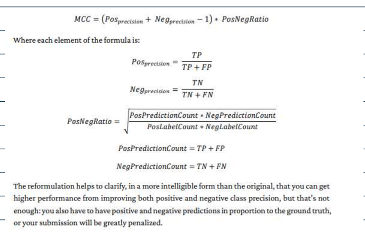
(TP×TN) = (FP×FN) → very bad 0

TP = 0, TN = 0 → fault in model -1 — inverse prediction

→ use cases
- Imbalance Dataset
- all classes are equally important {like in f1-score}

→ MCC is more informative than f1-score in evaluating binary classification problems.

→ B/C it takes into account the balance ratios of 4 confusion matrix cells.



$$MCC = (Pos_{precision} + Neg_{precision} - 1) * PosNegRatio$$

Where each element of the formula is:

$$Pos_{precision} = \frac{TP}{TP + FP}$$

$$Neg_{precision} = \frac{TN}{TN + FN}$$

$$PosNegRatio = \sqrt{\frac{PosPredictionCount * NegPredictionCount}{PosLabelCount * NegLabelCount}}$$

$$PosPredictionCount = TP + FP$$

$$NegPredictionCount = TN + FN$$

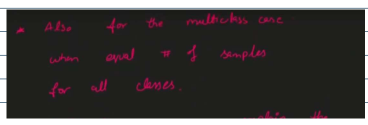
The reformulation helps to clarify, in a more intelligible form than the original, that you can get higher performance from improving both positive and negative class precision, but that's not enough: you also have to have positive and negative predictions in proportion to the ground truth, or your submission will be greatly penalized.

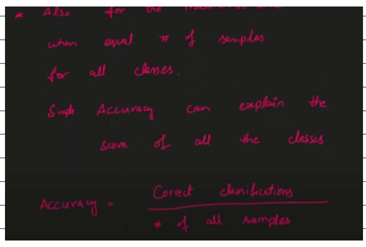
* MCC doesn't depend on which class is the positive one

Multi-class classification Performance measures

1
- 0 as 1
- 1 as 0

In Multi-class: (class) Choose which class we are more concerned about → (error) Choose which error we are more concerned about
- Type I — Precision
- Type II — Recall
- Both — f1 score





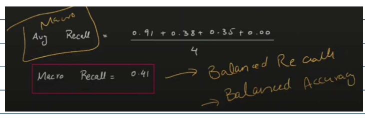
→ In multi-class problems, we extend the formulas of Binary class.

→ We are dealing with multiple classes, so calc concerned score for each class & then take the Average.

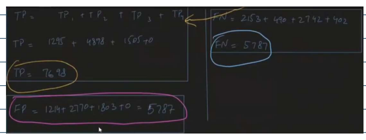
→ Average
- Micro
- Macro
- Weighted

} Averaging Techniques

Macro → Compute performance for each class & then average



Micro [not suitable for imbalanced dataset]



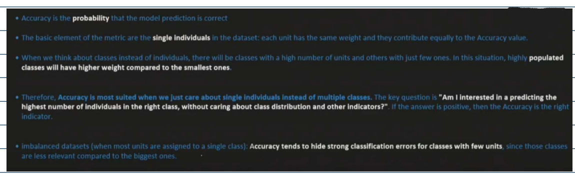
$\boxed{\text{Micro Recall = Micro Precision = Micro F1 score = Accuracy}}$

- → Micro Averaging holds same properties as Accuracy
- → Global Metric
- → Not good measure when classes are not balanced

→ Micro avg gives equal weight to each sample

→ Macro avg gives equal weight to each class

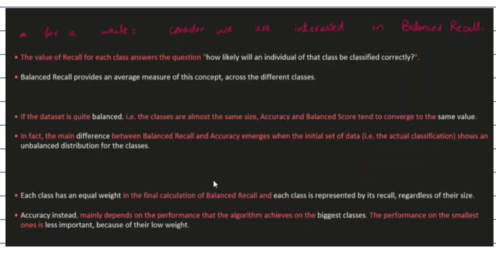
→ If we have same # of samples for each class, both macro & micro will provide the same score.



- Accuracy is the **probability** that the model prediction is correct
- The basic element of the metric are the **single individuals** in the dataset: each unit has the same weight and they contribute equally to the Accuracy value.
- When we think about classes instead of individuals, there will be classes with a high number of units and others with just few ones. In this situation, highly **populated classes will have higher weight compared to the smallest ones**.
- Therefore, **Accuracy is most suited when we just care about single individuals instead of multiple classes.** The key question is **"Am I interested in a predicting the highest number of individuals in the right class, without caring about class distribution and other indicators?"**. If the answer is positive, then the Accuracy is the right indicator.
- imbalanced datasets (when most units are assigned to a single class): **Accuracy tends to hide strong classification errors for classes with few units**, since those classes are less relevant compared to the biggest ones.

- Using this metric, **it is not possible to identify the classes where the algorithm is working worse.**

→ The metric is very intuitive & easy to understand.
Both in binary cases & multi-class cases the Accuracy assumes values b/w 0 & 1, While the quantity missing to reach 1 is called misclassification rate.



- The value of Recall for each class answers the question "how likely will an individual of that class be classified correctly?".
- Balanced Recall provides an average measure of this concept, across the different classes.
- If the dataset is quite balanced, i.e. the classes are almost the same size, Accuracy and Balanced Score tend to converge to the same value.
- In fact, the main difference between Balanced Recall and Accuracy emerges when the initial set of data (i.e. the actual classification) shows an unbalanced distribution for the classes.
- Each class has an equal weight in the final calculation of Balanced Recall and each class is represented by its recall, regardless of their size.
- Accuracy instead, mainly depends on the performance that the algorithm achieves on the biggest classes. The performance on the smallest ones is less important, because of their low weight.

- Summarizing the two main steps of Balanced Recall, first we compute a measure of recall for the algorithm on each class, then we apply the arithmetic mean of these values to find the final Balanced Recall score.
- All in all, Balanced Recall consists in the arithmetic mean of the recall of each class, so it is "balanced" because every class has the same weight and the same importance.
- A consequence is that smaller classes eventually have a more than proportional influence on the formula, although their size is reduced in terms of number of units. This also means that Balanced Score is insensitive to imbalanced class distribution and it gives more weight to the instances coming from minority classes. On the other hand, Accuracy treats all instances alike and usually favours the majority class.
- Balanced Score: This may be a perk if interested in having good prediction also for under-represented classes, or a drawback if we care more about good prediction on the entire dataset.
- The smallest classes when misclassified, are able to drop down the value of Balanced score, since they have the same importance as largest classes have in the equation.
- The class presents a high number of individuals, its bad performance is caught up also by the Accuracy.
- Instead, the class has just few individuals, the model's bad performance on this class cannot be caught up by Accuracy.

- If we are interested in achieving good predictions also for rare classes, the information of Balanced Score guarantees to spot possible predictive problems also for the under-represented classes.

Weighted : compute performance for each class & then take weighted average

```
y_test.value_counts()

Ideal      5388
Good       4247
Premium    3448
Fair        402
Name: cut, dtype: int64
```

$$\text{Weighted\_avg Recall} = \frac{5388(0.91) + 3448(0.38) + 4247(0.35) + 402(0.00)}{5388 + 3448 + 4247 + 402}$$

Weighted_avg Recall = 0.57

- The Balanced Score Weighted takes advantage of the Balanced Score formula multiplying each performance(i.e., recall) by the weight of its class, namely the frequency of the class on the entire dataset. We add also the sum of the weights at the denominator, with respect to the Balanced Score.
- Balanced score weighted holds the goodness of both accuracy and balanced score metrics.
- Once recalls have been weighted by the frequency of each class, the average of recall is no longer dirtied by low frequency classes(unlike balanced score): large classes will have a proportional weight to their size, and small ones will have a resized effect, compared with the Balanced Score formula.

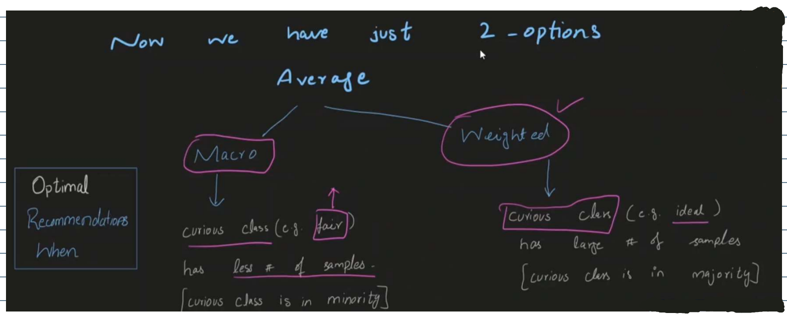
- Since every recall is weighted by the class frequency of the initial dataset, Balanced Recall Weighted could be a good performance indicator when the aim is to train a classification algorithm on a wide number of classes.
- In fact, this metric allows to keep separate algorithm performances on the different classes, so that we may track down which class causes poor performance(like balanced score). At the same time, it keeps track of the importance of each class thanks to the frequency(unlike balanced score).
- This ensures to obtain a reliable value of the overall performance on the dataset: we may interpret this metric as the probability to correctly predict a given unit, even if the formula is slightly different from the Accuracy.

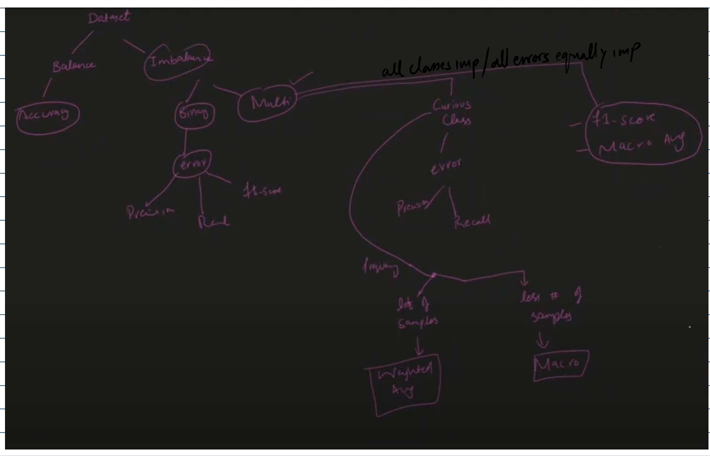
freq   class

- Weighted average Recall = Accuracy
- Interpretation is different
- Accuracy → w.r.t samples
- Weighted average Recall → w.r.t classes
  - denominator already same
  - numerator

sum of correct classifications w.r.t each class = all correct classifications

(Weighted Recall)   (accuracy)






```
x_train, x_test, y_train, y_test = train_test_split(X, y, random_state=2, stratify=y)
```