METRICS FOR MULTI-CLASS CLASSIFICATION: AN OVERVIEW

A WHITE PAPER

Margherita Grandini
CRIF S.p.A.*
m.grandini@crif.com

Enrico Bagli
CRIF S.p.A.*

Giorgio Visani
CRIF S.p.A.*
Department of Computer Science†
University of Bologna

August 14, 2020

ABSTRACT

Classification tasks in machine learning involving more than two classes are known by the name of "multi-class classification". Performance indicators are very useful when the aim is to evaluate and compare different classification models or machine learning techniques. Many metrics come in handy to test the ability of a multi-class classifier. Those metrics turn out to be useful at different stage of the development process, e.g. comparing the performance of two different models or analysing the behaviour of the same model by tuning different parameters. In this white paper we review a list of the most promising multi-class metrics, we highlight their advantages and disadvantages and show their possible usages during the development of a classification model.

1 Introduction

In the vast field of Machine Learning, the general focus is to predict an outcome using the available data. The prediction task is also called "classification problem" when the outcome represents different classes, otherwise is called "regression problem" when the outcome is a numeric measurement.
As regards to classification, the most common setting involves only two classes, although there may be more than two. In this last case the issue changes his name and is called "multi-class classification".

From an algorithmic standpoint, the prediction task is addressed using the state of the art mathematical techniques. There are many different solutions, however each one shares a common factor: they use available data ($\mathbf{X}$ variables) to obtain the best prediction $\hat{Y}$ of the outcome variable Y.
In Multi-class classification, we may regard the response variable Y and the prediction $\hat{Y}$ as two discrete random variables: they assume values in $\{1, \cdots, K\}$ and each number represents a different class.
The algorithm comes up with the probability that a specific unit belongs to one possible class, then a classification rule is employed to assign a single class to each individual. The rule is generally very simple, the most common rule assigns a unit to the class with the highest probability.
A classification model gives us the probability of belonging to a specific class for each possible units. Starting from the probability assigned by the model, in the two-class classification problem a threshold is usually applied to decide which class has to be predicted for each unit. While in the multi-class case, there are various possibilities; among them, the highest probability value and the softmax are the most employed techniques.

Performance indicators are very useful when the aim is to evaluate and compare different classification models or machine learning techniques.

*CRIF S.p.A., via Mario Fantin 1-3, 40131 Bologna (BO), Italy
†Università degli Studi di Bologna, Dipartimento di Ingegneria e Scienze Informatiche, viale Risorgimento 2, 40136 Bologna (BO), Italy

There are many metrics that come in handy to test the ability of any multi-class classifier and they turn out to be useful for: i) comparing the performance of two different models, ii) analysing the behaviour of the same model by tuning different parameters.

Many metrics are based on the Confusion Matrix, since it encloses all the relevant information about the algorithm and classification rule performance.

1.1 Confusion Matrix

The confusion matrix is a cross table that records the number of occurrences between two raters, the true/actual classification and the predicted classification, as shown in Figure 1. For consistency reasons throughout the paper, the columns stand for model prediction whereas the rows display the true classification.

The classes are listed in the same order in the rows as in the columns, therefore the correctly classified elements are located on the main diagonal from top left to bottom right and they correspond to the number of times the two raters agree.

		PREDICTED classification				
	Classes	a	b	c	d	Total
ACTUAL classification	a	6	0	1	2	9
	b	3	9	1	1	14
	c	1	0	10	2	13
	d	1	2	1	12	16
	Total	11	11	13	17	**52**

Figure 1: Example of confusion matrix

In the following paragraphs, we review two-class classification concepts, which will come in handy later to understand multi-class concepts.

1.2 Precision & Recall

These metrics will act as building blocks for Balanced Accuracy and F1-Score formulas.
Starting from a two class confusion matrix:

		PREDICTED		
	Classes	Positive (1)	Negative (0)	Total
ACTUAL	Positive (1)	TP = 20	FN = 5	25
	Negative (0)	FP = 10	TN = 15	25
	Total	30	20	**50**

Figure 2: Two-class Confusion Matrix

The Precision is the fraction of True Positive elements divided by the total number of positively predicted units (column sum of the predicted positives). In particular, True Positive are the elements that have been labelled as positive by the model and they are actually positive, while False Positive are the elements that have been labelled as positive by the model, but they are actually negative.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

Precision expresses the proportion of units our model says are Positive and they actually Positive. In other words, Precision tells us how much we can trust the model when it predicts an individual as Positive.

The Recall is the fraction of True Positive elements divided by the total number of positively classified units (row sum of the actual positives). In particular False Negative are the elements that have been labelled as negative by the model, but they are actually positive.

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

The Recall measures the model's predictive accuracy for the positive class: intuitively, it measures the ability of the model to find all the Positive units in the dataset.

Hereafter, we present different metrics for the multi-class setting, outlining pros and cons, with the aim to provide guidance to make the best choice.

2 Accuracy

Accuracy is one of the most popular metrics in multi-class classification and it is directly computed from the confusion matrix.

Referring to Figure 2:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

The formula of the Accuracy considers the sum of True Positive and True Negative elements at the numerator and the sum of all the entries of the confusion matrix at the denominator. True Positives and True Negatives are the elements correctly classified by the model and they are on the main diagonal of the confusion matrix, while the denominator also considers all the elements out of the main diagonal that have been incorrectly classified by the model.
In simple words, consider to choose a random unit and predict its class, Accuracy is the probability that the model prediction is correct.

Referring to Figure 1:

$$Accuracy = \frac{6 + 9 + 10 + 12}{52} \tag{4}$$

The same reasoning is also valid for the multi-class case.

Accuracy returns an overall measure of how much the model is correctly predicting on the entire set of data. 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.

A practical example is represented by 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.

On the other hand, the metric is very intuitive and easy to understand. Both in binary cases and multi-class cases the Accuracy assumes values between 0 and 1, while the quantity missing to reach 1 is called $MisclassificationRate$ [5].

3 Balanced Accuracy

Balanced Accuracy is another well-known metric both in binary and in multi-class classification; it is computed starting from the confusion matrix.

Referring to Figure 2:

$$Balanced\ Accuracy = \frac{\frac{TP}{Total_{row_1}} + \frac{TN}{Total_{row_2}}}{2} \tag{5}$$

Referring to Figure 1:

$$Balanced\ Accuracy = \frac{\frac{6}{9} + \frac{9}{14} + \frac{10}{13} + \frac{12}{16}}{4} \tag{6}$$

The formula of the Balanced Accuracy is essentially an average of recalls. First we evaluate the Recall for each class, then we average the values in order to obtain the Balanced Accuracy score. The value of Recall for each class answers the question "how likely will an individual of that class be classified correctly?". Hence, Balanced Accuracy 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 Accuracy tend to converge to the same value.
In fact, the main difference between Balanced Accuracy and Accuracy emerges when the initial set of data (i.e. the actual classification) shows an unbalanced distribution for the classes.

		PREDICTED classification				
	Classes	a	b	c	d	Total
ACTUAL classification	a	5	23	17	17	62
	b	10	540	21	14	585
	c	166	96	436	110	808
	d	1	2	5	87	95
	Total	182	661	479	228	**1550**

Figure 3: Imbalanced Dataset

Figure 3 shows how the actual classification is unbalanced towards classes "b" and "c". For this setting, Accuracy value is 0.689, whereas Balanced Accuracy is 0.615. The difference is mainly due to the weighting that recall applies on each row/actual class. In this way each class has an equal weight in the final calculation of Balanced Accuracy 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 Accuracy, first we compute a measure of performance (recall) for the algorithm on each class, then we apply the arithmetic mean of these values to find the final Balanced Accuracy score. All in all, Balanced Accuracy 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 Accuracy 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 [2].

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 Accuracy, since they have the same importance as largest classes have in the equation. For example, considering class "a" in the Figure 3, there are 57 misclassified elements and 5 elements which have been rightly predicted, for a total row of 62 elements belonging to the class "a" observing the actual classification. An amount of 57 elements have been assigned to other classes by the model, in fact the recall for this small class is quite low (0.0806).

When the class presents a high number of individuals (i.e. class "c"), its bad performance is caught up also by the Accuracy. Instead, when the class has just few individuals (i.e. class "a"), the model's bad performance on this last class cannot be caught up by Accuracy. If we are interested in achieving good predictions (i.e. class "b" and "d") also for rare classes, the information of Balanced Accuracy guarantees to spot possible predictive problems also for the under-represented classes.

3.1 Balanced Accuracy Weighted

The Balanced Accuracy Weighted takes advantage of the Balanced Accuracy formula multiplying each recall by the weight of its class w_k, namely the frequency of the class on the entire dataset. We add also the sum of the weights W at the denominator, with respect to the Balanced Accuracy.

$$Balanced\ Accuracy\ Weighted = \frac{\sum_{k=1}^{K} \frac{TP_k}{Total_{row_k} \cdot w_k}}{K \cdot W} \tag{7}$$

Referring to Figure 1:

$$Balanced\ Accuracy\ Weighted = \frac{\frac{6}{9} \cdot w_a + \frac{9}{14} \cdot w_b + \frac{10}{13} \cdot w_c + \frac{12}{16} \cdot w_d}{4 \cdot W} \tag{8}$$

Once recalls have been weighted by the frequency of each class (w_k), the average of recall is no longer dirtied by low frequency classes: large classes will have a proportional weight to their size, and small ones will have a resized effect, compared with the Balanced Accuracy formula.

Since every recall is weighted by the class frequency of the initial dataset, Balanced Accuracy 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. At the same time, it keeps track of the importance of each class thanks to the frequency. 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.

4 F1-Score

Also F1-Score assesses classification model's performance starting from the confusion matrix, it aggregates Precision and Recall measures under the concept of harmonic mean.

$$\text{F1-Score} = \left(\frac{2}{precision^{-1} + recall^{-1}} \right) = 2 \cdot \left(\frac{precision \cdot recall}{precision + recall} \right) \tag{9}$$

The formula of F1-score can be interpreted as a weighted average between Precision and Recall, where F1-score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall are equal onto the F1-score and the harmonic mean is useful to find the best trade-off between the two quantities [11].

The addends "Precision" and "Recall" could refer both to binary classification and to multi-class classification, as shown in Chapter 1.2: in the binary case we only consider the Positive class (therefore the True Negative elements have no

importance), while in the multi-class case we consider all the classes one by one and, as a consequence, all the entries of the confusion matrix.

To give some intuition about the F1-Score behaviour, we review the effect of the harmonic mean on the final score. It has been observed from previous studies that it gives large weight to smaller classes and it mostly rewards models that have similar Precision and Recall values. As an example, we consider Model A with Precision equal to Recall (80%), and Model B whose precision is 60% and recall is 100%. Arithmetically, the mean of the precision and recall is the same for both models, but using the harmonic mean, i.e. computing the F1-Score, Model A obtains a score of 80%, while Model B has only a score 75% [12].

Moreover, Precision and Recall take values in the range [0;1] and when one of them assumes values close to 0, the final F1-Score suffers a huge drop. In fact the harmonic mean tends to give more weight to lower values.

4.1 F1-Score Binary case

Referring to confusion matrix in Figure 2, since Precision and Recall do not consider the True Negative elements, we calculate the binary F1-Score as follows:

$$Precision = \frac{20}{30} = 0.66 \qquad Recall = \frac{20}{25} = 0.80 \tag{10}$$

$$\text{F1-Score} = 2 \cdot \left(\frac{0.66 \cdot 0.80}{0.66 + 0.80} \right) = 0.72 \tag{11}$$

The F1-Score for the binary case takes into account both Precision and Recall. Thanks to these metrics, we can be quite confident that F1-Score will spot weak points of the prediction algorithm, if any of those points exists.
In our example a score of 0.72 demonstrates a quite good model ability in predicting the correct class.

4.2 F1-Score Multi-class case

When it comes to multi-class cases, F1-Score should involve all the classes. To do so, we require a multi-class measure of Precision and Recall to be inserted into the harmonic mean. Such metrics may have two different specifications, giving rise to two different metrics: Micro F1-Score and Macro F1-Score [9].

4.2.1 Macro F1-Score

In order to obtain Macro F1-Score, we need to compute Macro-Precision and Macro-Recall before. They are respectively calculated by taking the average precision for each predicted class and the average recall for each actual class. Hence, the Macro approach considers all the classes as basic elements of the calculation: each class has the same weight in the average, so that there is no distinction between highly and poorly populated classes.

ACTUAL classification	PREDICTED classification: Classes	a	b	c	d
	a	TN	FP	TN	TN
	b	FN	TP	FN	FN
	c	TN	FP	TN	TN
	d	TN	FP	TN	TN

ACTUAL classification	PREDICTED classification: Classes	a	b	c	d	Total
	a	50	37	24	39	150
	b	10	480	5	3	498
	c	14	10	765	1	790
	d	0	2	9	101	112
	Total	74	529	803	144	**1550**

Figure 4: Macro-Precision and Macro-Recall
Class b is the reference and we show how to compute its Precision and Recall

For the required computations, we will use the Confusion Matrix focusing on one class at a time and labelling the tiles accordingly. In particular, we consider True Positive (TP) as the only correctly classified units for our class, whereas

False Positive (FP) and False Negative (FN) are the wrongly classified elements on the column and the row of the class respectively. True Negative (TN) are all the other tiles, as shown in Figure 4 where we are considering the class "b" as reference focus.
When we switch from one class to another one, we compute the quantities again and the labels for the Confusion Matrix tiles are changed accordingly.

Precision and Recall for each class are computed using the same formulas of the binary setting and the labelling, as described above. The Formulas 12 and 13 represent the two quantities for a generic class k.

$$Precision_k = \frac{TP_k}{TP_k + FP_k} \tag{12}$$

$$Recall_k = \frac{TP_k}{TP_k + FN_k} \tag{13}$$

Macro Average Precision and Recall are simply computed as the arithmetic mean of the metrics for single classes.

$$MacroAveragePrecision = \frac{\sum_{k=1}^{K} Precision_k}{K} \tag{14}$$

$$MacroAverageRecall = \frac{\sum_{k=1}^{K} Recall_k}{K} \tag{15}$$

Eventually, Macro F1-Score is the harmonic mean of Macro-Precision and Macro-Recall:

$$Macro \text{ F1-Score} = 2 * (\frac{MacroAveragePrecision * MacroAverageRecall}{MacroAveragePrecision^{-1} + MacroAverageRecall^{-1}}) \tag{16}$$

It is possible to derive some intuitions from the equation.
Macro-Average methods tend to calculate an overall mean of different measures, because the numerators of Macro Average Precision and Macro Average Recall are composed by values in the range $[0, 1]$. There is no link to the class size, because classes with different size are equally weighted at the numerator. This implies that the effect of the biggest classes have the same importance as small ones have. The obtained metric evaluates the algorithm from a class standpoint: high Macro-F1 values indicate that the algorithm has good performance on all the classes, whereas low Macro-F1 values refers to poorly predicted classes.

4.3 Micro F1-Score

In order to obtain Micro F1-Score, we need to compute Micro-Precision and Micro-Recall before.
The idea of Micro-averaging is to consider all the units together, without taking into consideration possible differences between classes. Therefore, the Micro-Average Precision is computed as follows:

$$Micro\ Average\ Precision = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{K} Total\ Column_k} = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \tag{17}$$

What about the Micro-Average Recall? When we try to evaluate it, we observe the measure is exactly equal to the Micro-Average Precision, in fact summing the two measures rely on the sum of the True Positives, whereas the difference should be in the denominator: we consider the Column Total for the Precision calculation and the Row Total for the Recall calculation. Although, using the units all together ends up in having the Grand Total in both the Formulas.

$$Micro\ Average\ Recall = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{K} Total\ Row_k} = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \tag{18}$$

Long story short, we may see that Micro-Average Precision and Recall are just the same values, therefore the Micro-Average F1-Score is just the same as well (the harmonic mean of two equal values is just the value).

$$Micro\ Average\ F1 = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \tag{19}$$

Taking a look to the formula, we may see that Micro-Average F1-Score is just equal to Accuracy. Hence, pros and cons are shared between the two measures. Both of them give more importance to big classes, because they just consider all the units together. In fact a poor performance on small classes is not so important, since the number of units belonging to those classes is small compared to the dataset size.

All in all, we may regard the Macro F1-Score as an average measure of the average precision and average recall of the classes. This measure is calculated at class level, so that each class has the same weight. Small classes are equivalent to big ones and the algorithm performance on them is equally important, regardless of the class size.
On the contrary, trying to reverse the concept and build the Micro F1-score, just give us the Accuracy Formula. So that we have a new interpretation of the Accuracy as the average of Precision and Recall above the entire dataset. The Accuracy, as stated above, is calculated at a dataset level and each unit has the same importance. Therefore, the Accuracy gives different importance to different classes, based on their frequency in the dataset.

4.4 Cross Entropy

From a theoretical point of view, Cross-Entropy is used to evaluate the similarity between two distribution functions. Consider two generic distributions $p(x)$ and $q(x)$, the Cross-Entropy is given by the formula 20-21, to respectively suit continuous or discrete X variables.

$$H(p, q) = -\int_{D_x} p(x) \log q(x) \tag{20}$$

$$H(p, q) = -\sum_{D_x} p(x) \log q(x) \tag{21}$$

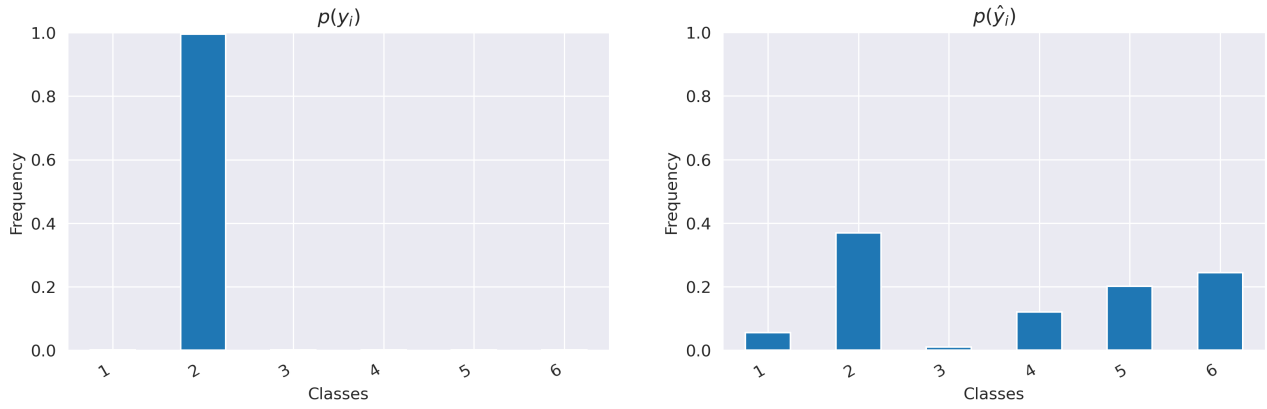
The metric compares the two distributions over the entire domain D_X of the X variable and it only assumes positive values. In particular, small values of the Cross-Entropy function denote very similar distributions.

In Multi-class classification, we may regard the response variable Y and the prediction $\hat{Y}$ as two discrete random variables: they assume values in $\{1, \cdots, K\}$ and each number represents a different class.
Considering the generic i-th unit of the dataset: it has specific values $(x_1^{(i)}, \cdots, x_m^{(i)})$ of the $\mathbf{X}$ variables and the number $y^{(i)}$ represents the class the unit belongs to. Since we only observe the true class, we consider the unit to have probability equal to 1 for this class and probability equal to 0 for the remaining classes. Doing so, $y^{(i)}$ may be rewritten as a vector of probabilities, as shown in Panel 5a. On the other hand, the algorithm prediction itself generates a numeric vector $\hat{y}^{(i)}$, with the probability for the i-th unit to belong to each class.

$y^{(i)}$ and $\hat{y}^{(i)}$ are generated respectively from the conditioned random variables $Y|\mathbf{X}$ and $\hat{Y}|\mathbf{X}$. The conditioning reflects the fact that we are considering a specific unit, with specific characteristics, namely the unit's values for the $\mathbf{X}$ variables.

In the following figures we will regard respectively $p(y_i)$ and $p(\hat{y}_i)$ as the probability distributions of the conditioned variables above. In Figure 5, a representation of the two distributions, for a fictitious unit.

The Cross-Entropy for the i-th unit (Formula 22) is calculated between the true distribution $p(y_i)$ and the prediction $p(\hat{y}_i)$.



(a) Distribution of $Y_i|\mathbf{X}$

(b) Distribution of $\hat{Y}_i|\mathbf{X}$

Figure 5

$$H(p(y_i), p(\hat{y}_i)) = -\sum_{k=1}^{K} p(Y_i = k|\mathbf{X_i}) \log p(\hat{Y}_i = k|\mathbf{X_i}) \tag{22}$$

Each class is considered in the formula above, however the quantity $p(Y_i = k|\mathbf{X_i})$ is 0 for all the classes except the true one, making all the terms but one disappear. Cross-Entropy exploits only the value of $p(\hat{Y}_i = k|\mathbf{X_i})$ for the k value representing the true class.

Eventually we consider the average of the Cross-Entropy values for the single units, to obtain a measure of agreement on the entire dataset (Formula 23).

$$H(p(y), p(\hat{y})) = -\sum_{i}^{N} \sum_{k=1}^{K} p(Y_i = k|\mathbf{X_i}) \log p(\hat{Y}_i = k|\mathbf{X_i}) \tag{23}$$

It is worth noting that the technique does not rely on the Confusion Matrix, instead it employs directly the variables Y and $\hat{Y}$. Therefore, Cross-Entropy does not evaluate the goodness of the classification rule (the rule which translates the probabilities into the predicted class).

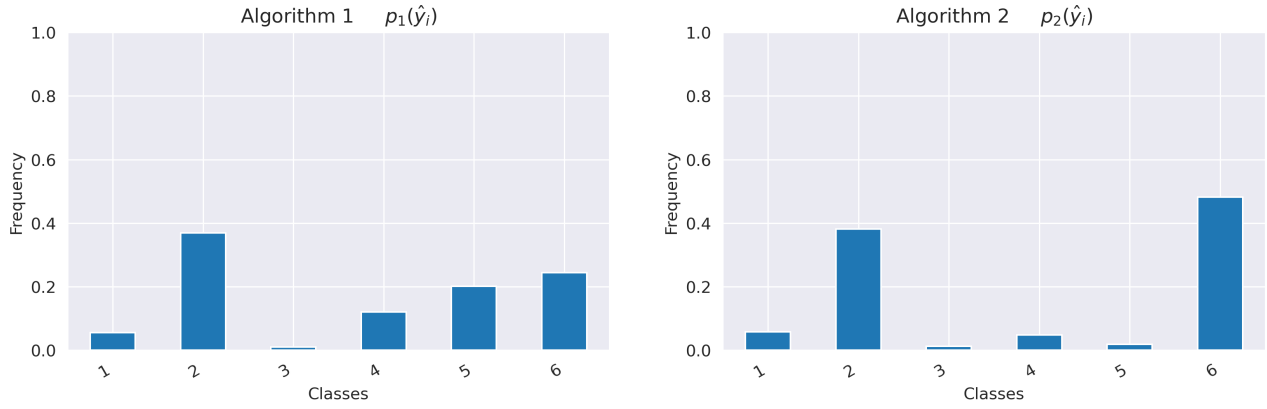
From a practical perspective, Cross-Entropy is widely employed thanks to its fast calculation. Although, it takes into account only the true class probability $p(\hat{y}_i = k)$ without caring about the probability mass distribution among the remaining classes.
This may have some drawbacks, as shown in Figure 6: the i-th unit gets predicted by two different algorithms, obtaining two distinct distributions. The true label is $y_i = 2$, referring to the same unit of Figure 5.

The two algorithms have the same prediction for class 2, i.e. 0.4, but, substantially, they have different performance on the aggregate perspective: in Panel 6a the highest probability class is 2, for 6b it is 6. Depending on the classification rule, the two algorithms are likely to assign different predicted classes to the unit.

Regardless of such differences, Cross-Entropy assigns the same value to each algorithm, since $p(\hat{y}_i = 2)$ is the same. In our example, Panel 6a achieves right predictions and 6b a wrong one, without being reported by the Cross-Entropy.

5 Independence between two Random Discrete Variables

The last two metrics in this paper were built starting from the confusion matrix and relying on two different statistical concepts. Matthews Correlation Coefficient takes advantage of the Phi-Coefficient [7], while Cohen's Kappa Score



(a) Distribution of $\hat{Y}_i^{(1}|\mathbf{X}$

(b) Distribution of $\hat{Y}_i^{(2}|\mathbf{X}$

Figure 6

relates to the probabilistic concept of dependence between two random variables. It will be demonstrated that both the scores come up with a measure of how much the model's predictions are dependent on the ground truth classification of a given dataset. Moreover, both the metrics take into account the True Negative (TN) values in the binary case, so they may be preferable to F1-Score when the aim is to assessing the performance of a binary classifier.

5.1 Mattheus Correlation Coefficient

Brian W. Mattheus developed the Mattheus Correlation Coefficient (MCC) in 1975, exploiting Karl Pearson's Phi-Coefficient in order to compare different chemical structures. Only in the 2000s MCC became a widely employed metric to test the performance of Machine Learning techniques with some extensions to the multi-class case [4].

MCC has a range of $[-1, 1]$. Values close to 1 indicate very good prediction, in fact there is a strong positive correlation between the Prediction and the True Labels. Strong correlation implies that the two variables strongly agree, therefore the predicted values will be very similar to the Actual Classification. On the contrary, when MCC is equal to 0, there is no correlation between our variables: the classifier is randomly assigning the units to the classes without any link to their true class value. [1].
MCC may also be negative, in this case the relation between true and predicted classes is of an inverse type. Even if this is an highly undesirable situation, this often happens because of setting errors in the modelling: strong inverse correlation means that the model learnt how to classify the data but it systematically switches all the labels. But it is also possible to solve the problem by fixing the implementation errors.

5.1.1 Mattheus Correlation Coefficient for binary classification

MCC could be seen as the Phi-Coefficient applied to binary classification problems: as described above, we consider the "Predicted" classification and "Actual" classification as two discrete random variables and we evaluate their association.

MCC = 0.408	PREDICTED		
	Classes	Positive (1)	Negative (0)	Total
ACTUAL	Positive (1)	TP = 20	FN = 5	25
	Negative (0)	FP = 10	TN = 15	25
	Total	30	20	**50**

Figure 7: Predicted and Actual Random Variables

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN)(TP + FP)(TN + FN)(TN + FP)}} \tag{24}$$

In Formula 24, we notice that MCC takes into account all the confusion matrix cells. In particular, the numerator consists of two products including all the four inner cells of the confusion matrix in Figure 7, while the denominator consists of the four outer cells (row totals and column totals).

A practical demonstration of the concept that there is an effective support regarding the equivalence of MCC and Phi-coefficient in the binary case is given by [8].

5.1.2 Mattheus Correlation Coefficient for Multi-class Classification

Some changes happen when it comes to multi-class classification: the True and the Predicted class distributions are no longer binary and a higher number of classes has been taken into account. In this case numerator and denominator take a different shape compared to the binary case and this can partially help to find more stable results inside the range [-1; +1] of MCC.

In the multi-class case, the Matthews correlation coefficient can be defined in terms of a confusion matrix C for K classes.

Matrix C		PREDICTED classification				
	K Classes	k=1	k=2	k=3	k=4	Total
ACTUAL classification	k=1	C_{11}=50	37	24	39	150
	k=2	10	C_{22}=480	5	3	$t_{k=2}$ = 498
	k=3	14	10	C_{33}=765	1	790
	k=4	0	2	9	C_{44}=101	112
	Total	74	$p_{k=2}$ = 529	803	144	s =**1550**

Figure 8: Multi-class Confusion Matrix C

$$MCC = \frac{c \times s - \sum_k^K p_k \times t_k}{\sqrt{(s^2 - \sum_k^K p_k^2)(s^2 - \sum_k^K t_k^2)}} \tag{25}$$

To simplify the definition, it is necessary to consider the following intermediate variables [6]:

- $c = \sum_k^K C_{kk}$ the total number of elements correctly predicted
- $s = \sum_i^K \sum_j^K C_{ij}$ the total number of elements
- $p_k = \sum_i^K C_{ki}$ the number of times that class k was predicted (column total)
- $t_k = \sum_i^K C_{ik}$ the number of times that class k truly occurred (row total)

The setting of this formula has raised some intuitions.
In the multi-class case MCC seems to depend on correctly classified elements, because the total number of elements correctly predicted are multiplied by the total number of elements at the numerator and the weight of this product is more powerful than the sum $\sum_k^K p_k \times t_k$. This sum includes also the elements wrongly classified by the model and covers multiplicative entities that are weaker than the product $c \times s$.

Regarding the denominator, it is employed to rescale the fraction in the interval $[-1, +1]$, in fact it corresponds to the maximum absolute value the numerator may assume. The first term in the denominator entirely depends on the

Predicted classification, whereas the second one depends on the True classes, which can be considered as property of the dataset since they do not change when we apply different models on the same dataset.

The weakness of MCC involves its lower limits. There is no fixed minimum value and it changes every time between -1 and 0 depending on the number and on the Actual distribution of the classes in the initial dataset [6].

5.1.3 Pros and Cons of MCC

Among the Advantages of this technique, we can see that MCC includes all the entries of the confusion matrix both at the numerator and the denominator. This means that MCC is generally regarded as a balanced measure which can be used in binary classification even if the classes are very different in size [4].

Moreover, MCC is a good indicator of total unbalanced prediction models. We have shown this topic in Figure 9, where the model assigns all the elements to only one class and the value of MCC falls to 0, even if the Accuracy achieves a great value (0.80) and the Recall for the first class assumes the highest value (1).

MCC = 0	PREDICTED		
Classes	Positive (1)	Negative (0)	Total
ACTUAL Positive (1)	TP = 40	FN = 0	40
Negative (0)	FP = 10	TN = 0	10
Total	50	0	**50**

Figure 9: Total Unbalanced Prediction

However, the weaknesses of MCC could be found in some extreme cases and they are mainly related to its construction. If there are unbalanced results in the model's prediction, the final value of MCC shows very wide fluctuations inside its range of [-1; +1] during the training period of the model [3].

5.2 Cohen's Kappa

Cohen's Kappa builds on the idea of measuring the concordance between the Predicted and the True Labels, which are regarded as two random categorical variables [10]. It is possible to compare two categorical variables building the confusion matrix and calculating the marginal rows and the marginal columns distributions.

		Model's PREDICTION					
	K Modes	p=1	...	p = h	...	p=K	Total
ACTUAL classification	a=1	n_{11}	...	n_{1h}	...	n_{1K}	n_{1T}
	...	...	...	...	...	...	...
	a = v	n_{v1}	...	n_{vh}	...	n_{vK}	n_{vT}
	...	...	...	...	...	...	...
	a=K	n_{K1}	...	n_{Kh}	...	n_{KK}	n_{KT}
	Total	n_{T1}	...	n_{Th}	...	n_{TK}	***N***

Figure 10: Confusion matrix for two General Categorical Distributions

In particular two distributions of the same character are independent if they assume the same relative frequencies at the same character model.

$$\frac{n_{vh}}{n_{Th}} = \frac{n_{vT}}{N} \tag{26}$$

Also, two characters (i.e. model's Prediction & Actual classification) are independent variables in distribution if this relationship is true:

$$n_{vh}{}^{*} = \frac{n_{Th} \times n_{vT}}{N} \tag{27}$$

And $n_{vh}*$ stands for a relative frequency that we expect to find if two categorical distributions are independent.

Given this definition of independence between categorical variables, we can start dealing with Cohen's Kappa indicators as rating values of the dependence (or independence) between the model's Prediction and the Actual classification.

The marginal columns distribution can be regarded as the distribution of the Predicted values (how many elements are predicted in each possible class), while the Marginal rows represent the distribution of the True classes.

Moreover, we will see in this chapter why Cohen's Kappa could be also useful in evaluating the performance of two different models when they are applied on two different databases and it allows to make a comparison between them.

5.2.1 Cohen's Kappa for binary classification

Starting from a simple confusion matrix:

K=0.13	PREDICTION		
Classes	Positive (1)	Negative (0)	Total
ACTUAL Positive (1)	TP = 45	FN = 15	60
ACTUAL Negative (0)	FP = 25	TN = 15	40
Total	70	30	**N = 100**

Figure 11: Confusion Matrix for binary Cohen's Kappa

Cohen (1960) evaluated the classification of two raters (i.e. model's Prediction & Actual distribution) in order to find a measure of agreement between them, or rather, he looked for a statistic giving the degree of concordance between two (or more) sets of measurements. He calculated the inter-observer agreement taking into account the expected agreement by chance as follows [10]:

$$K = \frac{P_o - P_e}{1 - P_e} \tag{28}$$

Where

- P_o is the proportion of observed agreement, in other words it is the Accuracy achieved by the model
- P_e is the Expected Accuracy, i.e. the level of Accuracy we expect to obtain by chance. If we have a model that classifies the units in random classes, preserving just the distribution of the predicted classes, its Accuracy should be close to P_e
- $1 - P_e$ is essentially the difference between the maximum value and the minimum value of the numerator, in this way we are re-scaling the final index between -1 and +1.

$$P_e = P_{Positive} + P_{Negative} \tag{29}$$

$$P_{Positive} = \frac{TP + FN}{N} \times \frac{TP + FP}{N} \tag{30}$$

$$P_{Negative} = \frac{TN + FP}{N} \times \frac{TN + FN}{N} \tag{31}$$

The K statistic can take values from -1 to $+1$ and is interpreted somewhat arbitrarily as follows: 0 is the agreement equivalent to chance, from 0.10 to 0.20 is a slight agreement, from 0.21 to 0.40 is a fair agreement, from 0.41 to 0.60 is a moderate agreement, from 0.61 to 0.80 is a substantial agreement, from 0.81 to 0.99 is a near perfect agreement and 1.00 is a perfect agreement. Negative values indicate that the observed agreement is worse than what would be expected by chance. An alternative interpretation is offered by [10] saying that kappa values below 0.60 indicate a significant level of disagreement.

- **Insights**

Given the similarity of the last operations to the concept of independence between two events,

$$P_{Positive} = P(Prediction_1 \cap Actual_1) \tag{32}$$

$$P_{Positive} = P(Prediction_1) \times P(Actual_1) \tag{33}$$

$$P_{Positive} = \frac{45 + 15}{100} \times \frac{45 + 25}{100} = 0.42 \tag{34}$$

we have noticed that the Expected Accuracy P_e plays the main role in the Cohen's Kappa Score because it brings with it two components of independence ($P_{Positive}$ and $P_{Negatives}$) which are subtracted from the observed agreement P_o.

It is important to remove the Expected Accuracy (the random agreement component for Cohen and the two independent components for us) from the Accuracy for two reasons: the Expected Accuracy is related to a classifier that assigns units to classes completely at random, it is important to find a model's Prediction that is as dependent as possible to the Actual distribution. So Cohen's Kappa results to be a measure of how much the model's prediction is dependent on the Actual distribution, with the aim to identify the best learning algorithm of classification.

These are the basic intuitions on Cohen's Kappa score and they have to be supported by demonstrations:

$$P_e = P_{Positive} + P_{Negative} \tag{35}$$

$$P_e = \frac{TP + FN}{N} \times \frac{TP + FP}{N} + \frac{TN + FP}{N} \times \frac{TN + FN}{N} \tag{36}$$

$$if\ {n_{vh}}^* = \frac{n_{Th} \times n_{vT}}{N}\ then\ TP^* = \frac{(45 + 15) \times (45 + 25)}{100} \tag{37}$$

$$and\ P_{Positive} = TP^* \times \frac{1}{100}\ and\ P_{Negative} = TN^* \times \frac{1}{100} \tag{38}$$

Coming back to the formula 35:

$$P_e = P_{Positive} + P_{Negative} = \frac{TP^*}{100} + \frac{TN^*}{100} = \frac{TP^* + TN^*}{100} \tag{39}$$

If two random and categorical variables are independent they should have this Accuracy $\frac{TP^*+TN^*}{100}$. But, since we want that the Predicted and Actual distribution to be as dependent as possible, Cohen's Kappa score directly subtracts this previous Accuracy from the observed agreement at the numerator of the formula. In this way, we have obtained an Accuracy value related only to the goodness of the model and we have already deleted the part ascribed to chance (the Expected Accuracy).

Just as a reminder, two dependent variables are also correlated and identified by reciprocal agreement. In our case a high correlation is observed when the model's Prediction assigns a unit to one class, and the same unit has been also assigned to the same class by the Actual classification.

5.2.2 Cohen's Kappa for multi-class cases

In the multi-class case, the calculation of Cohen's Kappa Score changes its structure and it becomes more similar to Mattheus Correlation Coefficient [13].

Referring to Multi-class Confusion Matrix C in Figure 8:

$$K = \frac{c \times s - \sum_k^K p_k \times t_k}{s^2 - \sum_k^K p_k \times t_k} \tag{40}$$

Where:

- $c = \sum_k^K C_{kk}$ the total number of elements correctly predicted
- $s = \sum_i^K \sum_j^K C_{ij}$ the total number of elements
- $p_k = \sum_i^K C_{ki}$ the number of times that class k was predicted (column total)
- $t_k = \sum_i^K C_{ik}$ the number of times that class k truly occurs (row total)

MCC and Cohen's Kappa coincides in the multi-class cases apart from the denominator that is slightly lower in Cohen's Kappa score justifying slightly higher final scores. However some evidences of the binary case still holds: when K is equal to 0 the model's Prediction is totally independent from the Actual classification and if K is equal to 1 the model's Prediction is totally dependent from the Actual classification. Instead K is negative when the agreement between the algorithm and the true labels distribution is worse than the random agreement, so that there is no accordance between the model's Prediction and the Actual classification.

As before, the advantage of Cohen's Kappa score must be sought through the measure of Expected Accuracy as an intrinsic characteristic of a given dataset. In the multi-class case the Expected Accuracy assumes the shape of the sum applied on the row and column totals multiplication for each class k (40).

5.2.3 Useful Applications

Cohen's Kappa finds useful applications in many classification problems.
Firstly it allows the joint comparison of two models for which it has registered the same accuracy, but different values of Cohen's Kappa. Figure 12 is a simplified binary example, where K increases more the errors are unbalanced towards one class. This is true also for multi-class settings.

K = 0.13		PREDICTION	
	N = 100	Positive (1)	Negative (0)
GROUND TRUTH	Positive (1)	TP = 45	FN = 15
	Negative (1)	FP = 25	TN = 15

K = 0.259		PREDICTION	
	N = 100	Positive (1)	Negative (0)
GROUND TRUTH	Positive (1)	TP = 25	FN = 35
	Negative (1)	FP = 5	TN = 35

Figure 12: Cohen's Kappa Matrix for comparison

Secondly, the Expected Accuracy re-scales the score and represents the intrinsic characteristics of a given dataset.
We consider both the number of classes and the fact to be balanced or unbalanced towards a group of classes as the two

main representative characteristics of a dataset. Subtracting the Expected Accuracy we are also removing the intrinsic dissimilarities of different datasets and we are making two different classification problems comparable. As a result, K can compare the performances of two different model on two different cases.

6 Conclusions

The aim of this essay is an in-depth analysis of different choices to evaluate the performance of different classification algorithms on multi-class datasets.

As the most famous classification performance indicator, the **Accuracy** returns an overall measure of how much the model is correctly predicting the classification of a single individual above the entire set of data. It is an average measure which is suitable for balanced datasets because it does not consider the class distribution.

As a simple arithmetic mean of Recalls, the **Balanced Accuracy** gives the same weight to each class and its insensibility to class distribution helps to spot possible predictive problems also for rare and under-represented classes.

As weighted average of Recall, the **Balanced Accuracy Weighted** keeps track of the importance of each class thanks to the frequency. In this case, large and small classes have a proportional effect on the result in relation to their size and the metric can be applied during the training phase of the algorithm on a wide number of classes.

As harmonic mean of Macro Precision and Macro Recall, **Macro-Average** methods tend to calculate an overall mean of different measures without taking into account the class size. The effect of the biggest classes is shifted by the smallest ones which have the same weight.

Regarding **Micro F1-Score**, it is possible to show that the harmonic mean of Micro Precision and Micro Recall just boils to the Accuracy formula, giving a new interpretation of it.

As average of **Cross Entropy** for each unit in a dataset, it is a measure of agreement between two probability distributions (predicted and true classification). Cross Entropy is detached from the confusion matrix and it is widely employed thanks to his fast calculation. Although it just takes into account the prediction probability of the right class, without considering how the probability distribution behaves on the other classes, this may cause issues especially when a unit is misclassified.

It may be considered as the successor of Karl Pearson's Phi-Coefficient, the **Mattheus Correlation Coefficient** expresses the degree of correlation between two categorical random variables (predicted and true classification). Its result covers the range [-1; +1] pointing out different model behaviors during the training phase of the algorithm.

Its value represents the dependence between the predicted and the true classification, **Cohen's Kappa** exploits the Expected Accuracy, namely a measure representing the dependence obtained by chance between the predicted and the true classification measure, to delete any intrinsic characteristic of the dataset. This allows for the comparison between different models applied on different samples of data.

Funding

We acknowledge financial support by CRIF S.p.A. and Università degli Studi di Bologna.

References

[1] Sabri Boughorbel, Fethi Jarray, and Mohammed El-Anbari. "Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric". In: *PLOS ONE* 12.6 (June 2017), pp. 1–17. DOI: 10.1371/journal.pone.0177678. URL: https://doi.org/10.1371/journal.pone.0177678.

[2] K. H. Brodersen et al. "The Balanced Accuracy and Its Posterior Distribution". In: *2010 20th International Conference on Pattern Recognition*. 2010, pp. 3121–3124.

[3] J. B. Brown. "Classifiers and their Metrics Quantified". In: *Molecular Informatics* 37.1-2 (2018), p. 1700127. DOI: 10.1002/minf.201700127. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/minf.201700127. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/minf.201700127.

[4] Davide Chicco and Giuseppe Jurman. "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation". eng. In: *BMC genomics* 21.1 (Jan. 2020), pp. 6–6. ISSN: 1471-2164. DOI: 10.1186/s12864-019-6413-7. URL: https://doi.org/10.1186/s12864-019-6413-7.

[5] S.C. CHOI. "DISCRIMINATION AND CLASSIFICATION: OVERVIEW". In: *Statistical Methods of Discrimination and Classification*. Ed. by SUNG C. CHOI. Pergamon, 1986, pp. 173–177. ISBN: 978-0-08-034000-5. DOI: https://doi.org/10.1016/B978-0-08-034000-5.50005-8. URL: http://www.sciencedirect.com/science/article/pii/B9780080340005500058.

[6] J. Gorodkin. "Comparing two K-category assignments by a K-category correlation coefficient". In: *Computational Biology and Chemistry* 28.5 (2004), pp. 367–374. ISSN: 1476-9271. DOI: https://doi.org/10.1016/j.compbiolchem.2004.09.006. URL: http://www.sciencedirect.com/science/article/pii/S1476927104000799.

[7] B.W. Matthews. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". In: *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2 (1975), pp. 442–451. ISSN: 0005-2795. DOI: https://doi.org/10.1016/0005-2795(75)90109-9. URL: http://www.sciencedirect.com/science/article/pii/0005279575901099.

[8] Nechushtan Moran. *Ok I got it...* Mar. 2020. URL: https://medium.com/@morannechushtan/ok-i-got-it-9445e36d6c95.

[9] Juri Opitz and Sebastian Burst. *Macro F1 and Macro F1*. 2019. arXiv: 1911.03347 [cs.LG].

[10] Priya Ranganathan, C. S. Pramesh, and Rakesh Aggarwal. "Common pitfalls in statistical analysis: Measures of agreement". eng. In: *Perspectives in clinical research* 8.4 (2017). PCR-8-187[PII], pp. 187–191. ISSN: 2229-3485. DOI: 10.4103/picr.PICR_123_17. URL: https://doi.org/10.4103/picr.PICR_123_17.

[11] Yutaka Sasaki et al. *The truth of the f-measure. 2007*. 2007.

[12] Boaz Shmueli. *Multi-Class Metrics Made Simple, Part II: the F1-score*. July 2019. URL: https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1.

[13] Antonio J. Tallón-Ballesteros and José Riquelme. "Data Mining Methods Applied to a Digital Forensics Task for Supervised Machine Learning". In: *Studies in Computational Intelligence* 555 (Jan. 2014), pp. 413–428. DOI: 10.1007/978-3-319-05885-6-17.