Confusion Matrix for Multiclass

Claudio C. Kandza-Tadi

African Master of Machine Intelligence (AMMI), AIMS Rwanda

(March, 2020)

1 INTRODUCTION

A confusion matrix is a table that is often used to summarize the performance of a classification algorithm (or "classifier") on a set of test data whose true values are known. Confusion matrix were first discussed by Townsend [1], and it was in 1998 that the term confusion matrix became popular in Machine Learning thanks to Kohavi and Provost [2]. In supervised machine learning, confusion matrix is a matrix that measures the quality of a classification system. Each row of the matrix corresponds to an actual class and each column corresponds to a predicted class. Classification accuracy alone can be misleading if you have an unequal number of observations in each class or if you have more than two classes in your data set. One of the advantages of the confusion matrix is that it quickly shows whether a classification system manages to classify correctly. The confusion matrix itself is relatively simple to understand, but the associated terminology and the calculation of certain elements of the matrix can be confusing, especially in the case of multiple classes. Manliguez [3] derived the generalized formula to calculate the precision, the recall, the specificity and the global accuracy of the system having multiple classes. His results were used in eight (8) different works ([4] , [5], [6], [7], [8], [9], [10], [11]) to compute the overall accuracy in most cases, but we have realized that it were the only well defined generalized classification metric in Manliguez [3]. The aim of this document is to show the evidence of the mistakes in some generalized classification metrics and also to correct them. To achieve that objective, we have used the analytical approach for binary classification case and the computational approach using python package scikit-learn for the ternary Classification case study.

1.1 Confusion Matrix Form for Multiple Classes

Given that there are n classes, Table 1 shows the format of a confusion matrix for multiple classes.

		Predicted Number			
		Class 1	Class 2	…	Class ***n***
Actual Number	Class 1	x_{11}	x_{12}	…	x_{1n}
	Class 2	x_{21}	x_{22}	…	x_{2n}
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	Class ***n***	x_{n1}	x_{n2}	…	x_{nn}

Table 1. Confusion Matrix for Multiple Classes
Source : Manliguez [3]

2 GENERALIZED CONFUSION MATRIX FOR MULTIPLE CLASSES

2.1 Generalized formula for the number of true/false positive, true/false negative

- The total numbers of false negative (TFN) for each class i is given by

$$TFN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij}$$

- The total numbers of false positive (TFP) for each class i is given by

$$TFP_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ji}$$

- The total numbers of true negative (TTN) for each class i is given by

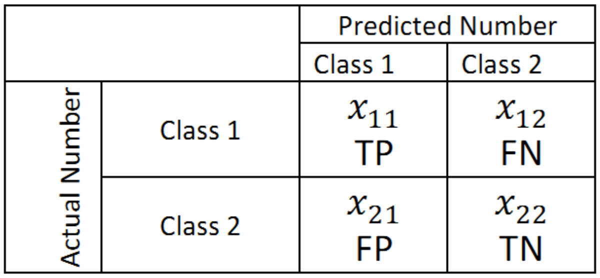
$$TTN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{\substack{k=1 \\ k \neq i}}^{n} x_{jk}$$

- The total numbers of true positive in the system (TTP) is given by

$$TTP_{all} = \sum_{j=1}^{n} x_{jj}.$$

2.2 Generalized Classification Metrics

In this section, we are going to present the results for the analytical and computational aprroaches. For the analytical part we have considered the binary classification case and the table below shows the format of confusion matrix for two (2) classes.



		Predicted Number	
		Class 1	Class 2
Actual Number	Class 1	x_{11} TP	x_{12} FN
	Class 2	x_{21} FP	x_{22} TN

Table 2. Confusion Matrix for 2 Classes

- True positive (TP): correctly classified or detected.
- False positive (FP): incorrectly classified or detected. It represents the type I error.
- False negative (FN): incorrectly rejected. It represents the type II error.
- True negative (TN): correctly rejected.

Precision

- The generalized precision (P) for each class i

  Precision also called sensitivity helps us to measures the proportion of actual positives that are correctly identified. And it has been defined such as

  $$P_i = \frac{TTP_{all}}{TTP_{all} + TFP_i}$$

  This equation does take into account the binary classification. It is even not working for three classes. below we are going to show it analytically for a binary classification problem.

We know that precision for binary classification is given by

$$Precision = \frac{TP}{TP + FP}.$$

Now, considering the generalized formala for precision, we have

$$Precision_1 = \frac{\sum_{j=1}^{2} x_{jj}}{\sum_{j=1}^{2} x_{jj} + \sum_{\substack{j=2 \\ j\neq 1}}^{2} x_{j1}}.$$

And taking into account the elements of the table above, we get

$$\sum_{j=1}^{2} x_{jj} = x_{11} + x_{22} = TP + TN,$$

$$\sum_{\substack{j=2 \\ j\neq 1}}^{2} x_{j1} = x_{21} = FP.$$

Then,

$$Precision_1 = \frac{TP + TN}{TP + TN + FP}.$$

It is clear that,

$$\frac{TP + TN}{TP + TN + FP} \neq \frac{TP}{TP + FP}.$$

Therefore, the generalized precision formala does not work for binary classification.

Recall

- The generalized recall (R) for each class i

  Recall helps us to answer the following question: what proportion of actual Positives is correctly classified? And it have been defined such as

  $$R_i = \frac{TTP_{all}}{TTP_{all} + TFN_i}$$

In the case of binary classification problem, the generalized recall is given by

$$R_1 = \frac{\sum_{j=1}^{2} x_{jj}}{\sum_{j=1}^{2} x_{jj} + \sum\limits_{\substack{j=2 \\ j \neq i}}^{2} x_{1j}} = \frac{x_{11} + x_{22}}{x_{11} + x_{22} + x_{12}}$$

Then,

$$R_1 = \frac{TP + TN}{TP + TN + FN} \neq \frac{TP}{TP + FN}$$

with $\frac{TP}{TP+FN}$ the known formula for recall in the case of two classes.
We can see, the generalized recall is not working for binary classification problem.

Specificity

- The generalized specificity(S) for each class i have been defined as

  Specificity, also called the true negative rate helps us to measures the proportion of actual negatives that are correctly identified.

$$S_i = \frac{TTN_{all}}{TTN_{all} + TFP_i}.$$

Unfortunately, the document did not provide the generalized formular for Total number of True Negative TTN_{all}.

Overall Accuracy

- The overall accuracy of the classifier

  Accuracy helps us to measure the proportion of correctly predicted observation to the total observations. And the overall accuracy has been defined as

$$\text{Overall Accuracy} = \frac{TTP_{all}}{\text{Total Number of Testing Entries}}.$$

$$\text{Overall Accuracy} = \frac{\sum_{j=1}^{n} x_{jj}}{\sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}}.$$

For the above binary classification problem we have,

$$\text{Overall Accuracy} = \frac{\sum_{j=1}^{2} x_{jj}}{\sum_{i=1}^{2} \sum_{j=1}^{2} x_{ij}} = \frac{x_{11} + x_{22}}{x_{11} + x_{12} + x_{21} + x_{22}}.$$

It follows that,

$$\text{Overall Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}.$$

Clearly, the generalized overall accuracy have been well defined by the author.

2.3 Computational Approach

Python function for the generalized total number of true/false positive, true/false negative

```
        import numpy as np
        import pandas as pd

        #The total numbers of false negative (TFN) for each class i
def TFN(A, i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number for each class
        rows : represent the actual number for each class"""
        n = A.shape[0]
        sum_ = 0
        for j in range(n):
                if i != j:
                        sum_ += A[i,j]
        return sum_


        #The total numbers of false positive (TFP) for each class i
def TFP(A, i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        n = A.shape[0]
        sum_ = 0
        for j in range(n):
                if i != j:
                        sum_ += A[j,i]
        return sum_


        #The total numbers of true negative (TTN) for each class i
def TTN(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        n = A.shape[0]
        sum_ = 0
        for j in range(n):
                for k in range(n):
                        if (j != i) and (k != i):
                                sum_ += A[j,k]
        return sum_
        #The total numbers of true positive in the system (TTP)
def TTP_all(A):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return np.trace(A)
```

Python function for generalized classification metrics for multi classes

```
        #The generalized precision (P)for each class i
def Precision_G(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return TTP_all(A)/(TTP_all(A)+TFP(A,i))

        #The generalized recall (R)for each class i
def Recall_G(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return TTP_all(A)/(TTP_all(A)+TFN(A,i))

        #The generalized specificity(S) for each class i
def Specificity_G(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return TTN_all(A)/(TTN_all(A)+TFP(A,i))

        #The Overall accuracy
def OverallAccuracy(A):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        n = A.shape[0]
        sum_ = 0
        for j in range(n):
                for k in range(n):
                        sum_ += A[j,k]
        return TTP_all(A)/sum_

        #Display the results
def display_G(A):
```

```
"""A : confusion matrix (square matrix)
columns: represent the predicted number
rows : represent the actual number"""
dic = {'Precision':[Precision_G(A,i) for i in range(A.shape[0])],
        'Recall':[Recall_G(A,i) for i in range(A.shape[0])]}
result = pd.DataFrame(dic, index=['Cat','Fish','Hen'])
return result.head()
```

Comparison between Scikit-learn and Generalized confusion matrix for multiple classes

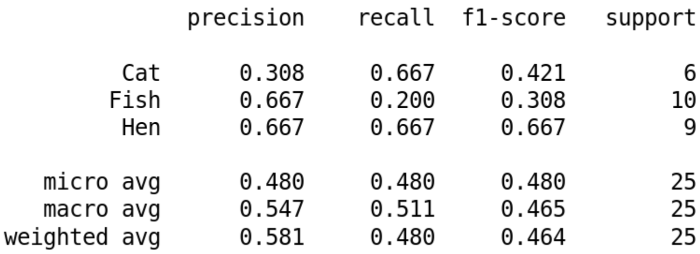
```
from sklearn import metrics

# Constants
C="Cat"
F="Fish"
H="Hen"

# True values
y_true = [C,C,C,C,C,C, F,F,F,F,F,F,F,F,F,F, H,H,H,H,H,H,H,H,H]
# Predicted values
y_pred = [C,C,C,C,H,F, C,C,C,C,C,C,H,H,F,F, C,C,C,H,H,H,H,H,H]

# Print the confusion matrix
print(metrics.confusion_matrix(y_true, y_pred))
```

```
# Print the precision and recall, among other metrics
#using Scikit-learn
print(metrics.classification_report(y_true, y_pred, digits=3))
```



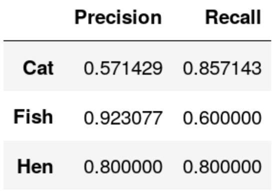
```
              precision    recall  f1-score   support

         Cat      0.308     0.667     0.421         6
        Fish      0.667     0.200     0.308        10
         Hen      0.667     0.667     0.667         9

   micro avg      0.480     0.480     0.480        25
   macro avg      0.547     0.511     0.465        25
weighted avg      0.581     0.480     0.464        25
```

Table 3. Using Scikit-learn function

```
# The confusion matrix for Cat , Fish and Hen
#ternary classification
Confusion_matrix = metrics.confusion_matrix(y_true , y_pred)
display_G(Confusion_matrix)
```



	Precision	Recall
Cat	0.571429	0.857143
Fish	0.923077	0.600000
Hen	0.800000	0.800000

Table 4. Generalized Confusion Matrix for Multiple Classes

It has been observed that even for ternary classification problem, the generalized classification metrics are not well defined. In the following line, we are going to correct that.

3 CORRECTED GENERALIZED CONFUSION MATRIX FOR MULTI CLASS CLASSIFICATION

3.1 Corrected Classification Metrics for Mutiple Classes

- The corrected generalized precision (P) for each class i is defined as

$$P_i = \frac{TP_i}{TP_i + FP_i}$$

where $TP_i = x_{ii}$ and $FP_i = \sum_{\substack{j=1\\ j\neq i}}^{n} x_{ji}$

- The corrected generalized recall (R) for each class i is defined as

$$R_i = \frac{TP_i}{TP_i + FN_i}$$

where $TP_i = x_{ii}$ and $FN_i = \sum_{\substack{j=1\\ j\neq i}}^{n} x_{ij}$

- The corrected generalized Specificity (Sp) for each class i is defined as

$$Sp_i = \frac{TN_i}{TN_i + FP_i}$$

where $TN_i = \sum_{\substack{j=1\\ j\neq i}}^{n} \sum_{\substack{k=1\\ k\neq i}}^{n} x_{jk}$

- The generalized Accuracy (Accur) for each class i is defined as

$$Accur_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}$$

where $TP_i = x_{ii}$

- The Overall Accuracy of the system is given by

$$Overall\ Accuracy = \frac{TTP_{all}}{TTE}$$

where $TTE = \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}$ (total numbers of testing entries in the system) and $TTP_{all} = \sum_{j=1}^{n} x_{jj}$.
NB: TTN = TN, TFP = FP and TFN = FN, they are well defined.

3.2 Python function for the corrected generalized classification metrics

```
        #The corrected generalized precision (P) function
        #for each class i
def Precision_CG(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number
        """
        return A[i,i]/(A[i,i]+TFP(A,i))

        #The corrected generalized recall (R) function
        #for each class i
def Recall_CG(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return A[i,i]/(A[i,i]+TFN(A,i))

        #The corrected generalized Specificity (Sp) function
        #for each class i
def Specificity_CG(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return TTN(A,i)/(TTN(A,i)+TFP(A,i))

        #The generalized Accuracy (Accur)for each class i
def Accuracy_CG(A,i):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return (A[i,i] + TTN(A,i))/(A[i,i] + TTN(A,i)+TFP(A,i)+TFN(A,i))

        # Accuracy of the all system (Overall Accuracy)
```

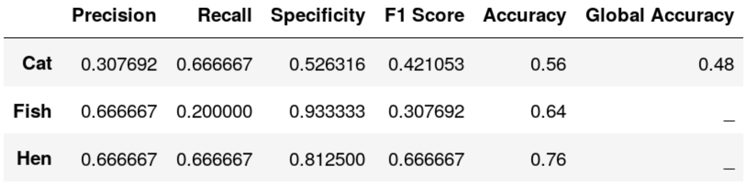
```
def Accuracy_Sys(A):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        return np.trace(A)/np.sum(A)

        # F1 Score(Harmonic mean of precision and recall)
def F1_Score(A,i,b):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number
        b : commonly 0.5, 1, 2"""
        N_ = (1+b)*Precision_CG(A,i)*Recall_CG(A,i)
        D_ = (b**2)*Precision_CG(A,i)+Recall_CG(A,i)
        return N_ /D_

        # Display the results for the corrected generalized
        # confusion matrix
def display_CG(A):
        """A : confusion matrix (square matrix)
        columns: represent the predicted number
        rows : represent the actual number"""
        P = [Precision_CG(A,i) for i in range(A.shape[0])]
        R = [Recall_CG(A,i) for i in range(A.shape[0])]
        Sp = [Specificity_CG(A,i) for i in range(A.shape[0])]
        F1 = [F1_Score(A,i,1) for i in range(A.shape[0])]
        Accur = [Accuracy_CG(A,i) for i in range(A.shape[0])]
        dic = {'Precision':P,'Recall':R, 'Specificity':Sp,
        'F1 Score':F1,'Accuracy':Accur,
        'Global Accuracy':[Accuracy_Sys(A),'_' , '_']}
        result = pd.DataFrame(dic, index=['Cat','Fish','Hen'])
        return result.head()
```

3.3 Comparison between Scikit-learn and Corrected generalized classification metrics for multiple classes

```
        # Print the precision and recall, among other metrics
        # using corrected generalized confusion matrix
        display_CG(Confusion_matrix)
```



	Precision	Recall	Specificity	F1 Score	Accuracy	Global Accuracy
Cat	0.307692	0.666667	0.526316	0.421053	0.56	0.48
Fish	0.666667	0.200000	0.933333	0.307692	0.64	_
Hen	0.666667	0.666667	0.812500	0.666667	0.76	_

Table 5. Corrected generalized confusion matrix for multiple classes

It has been observed that the corrected generalized classification metrics for multi classes is given the same results by using Scikit-learn function (see Table 3 and Table 5).

4 CONCLUSION

Considering the importance of the confusion matrix in the fields of machine learning, we felt that it was useful point out the mistakes on the classification metrics and then correct them.

References

1. Townsend, J. T. (1971). Theoretical analysis of an alphabetic confusion matrix. Perception & Psychophysics, 9(1), 40-50. https://doi.org/10.3758/BF03213026
2. Ron, K., & Foster, P. (1998). Special issue on applications of machine learning and the knowledge discovery process. Journal of Machine Learning, 30, 271-274.
3. Manliguez, C. (2016). Generalized confusion matrix for multiple classes.
4. Abiyoga, A. W., & Iswari, N. M. S. (2019). Decision Support System for Choosing an Elective Course Using Naive Bayes Classifier. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 850, 97.
5. Pratama, B. T., Utami, E., & Sunyoto, A. (2019). The Impact of Using Domain Specific Features on Lexicon Based Sentiment Analysis on Indonesian App Review. In 2019 International Conference on Information and Communications Technology (ICOIACT) (pp. 474-479). IEEE.
6. Pratama, B. T., Utami, E., & Sunyoto, A. (2019). A Comparison of the Use of Several Different Resources on Lexicon Based Indonesian Sentiment Analysis on App Review Dataset. In 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT) (pp. 282-287). IEEE.
7. Sardar, V. M., & Shirbahadurkar, S. D. (2018). Speaker identification of whispering speech: an investigation on selected timbrel features and KNN distance measures. International Journal of Speech Technology, 21(3), 545-553. https://doi.org/10.1007/s10772-018-9527-4
8. Elalfi, A.E., Elalmi, M.E., & Zahran, F.A. (2019). A Novel Intelligent System for Diagnosing some of Humans' Respiratory System Diseases.International Journal of Computer Applications 181(41):19-29.
9. Kharis, S. A. A., Hadi, I., & Hasanah, K. A. (2019). Multiclass Classification of Brain Cancer with Multiple Multiclass Artificial Bee Colony Feature Selection and Support Vector Machine. In Journal of Physics: Conference Series (Vol. 1417, No. 1, p. 012015). IOP Publishing.
10. Dalarmelina, N. D. V., Teixeira, M. A., & Meneguette, R. I. (2020). A Real-Time Automatic Plate Recognition System Based on Optical Character Recognition and Wireless Sensor Networks for ITS. Sensors, 20(1), 55.
11. Grifoni, P., Caschera, M. C., & Ferri, F. (2020). DAMA: A Dynamic Classification of Multimodal Ambiguities. International Journal of Computational Intelligence Systems.