Implementation of Model Evaluation using Confusion Matrix in Python

Ahmad Farhan AlShammari
Department of Computer and Information Systems
College of Business Studies, PAAET
Kuwait

ABSTRACT
The goal of this research is to develop a model evaluation program using confusion matrix in Python. Model evaluation is used to measure the performance of the applied model by comparing the predicted data with the actual data. Confusion matrix is used to summarize the predictions of the applied model and compute the evaluation metrics.

The basic steps of model evaluation using confusion matrix are explained: preparing data (actual and predicted), computing confusion matrix, computing totals (sum of items, diagonal, rows, and columns), computing evaluation metrics (accuracy, precision, recall, and F1-score), printing evaluation metrics, and plotting confusion matrix.

The developed program was tested on an experimental dataset. The program successfully performed the basic steps of model evaluation using confusion matrix and provided the required results.

Keywords
Artificial Intelligence, Machine Learning, Model Evaluation, Confusion Matrix, Evaluation Metrics, Accuracy, Precision, Recall, F1-Score, Python, Programming.

1. INTRODUCTION
In recent years, machine learning has played a major role in the development of computer systems. Machine learning (ML) is a branch of Artificial Intelligence (AI) which is focused on the study of computer algorithms to improve the performance and efficiency of computer programs [1-15].

Model evaluation is an important area in the field of machine learning. It is sharing the knowledge of many related fields: machine learning, programming, data science, mathematics, statistics, and numerical methods [16-21].

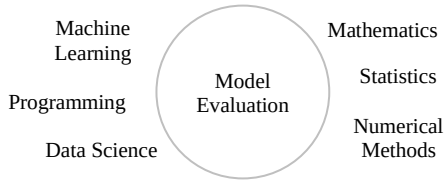

Fig 1: Area of Model Evaluation

Model evaluation is used to measure the performance of the applied model using confusion matrix which is used to summarize the predictions of the applied model and compute the evaluation metrics.

Model evaluation using confusion matrix is applied in the applications of machine learning, for example: classification and regression.

2. LITERATURE REVIEW
The study of literature reviewed the basic concepts of model evaluation using confusion matrix [22-28].

Model evaluation is very essential in the applications of machine learning. It is used to measure the performance of the applied model. Model evaluation is performed using confusion matrix which is used to summarize the predictions of the applied model and compute the evaluation metrics.

The fundamental concepts of model evaluation using confusion matrix are explained in the following section.

Model Evaluation:
Model evaluation is the process of measuring the performance of the applied model. The predicted data is compared with the actual data to count the number of correct (true) and incorrect (false) predictions. The resulting outcomes are summarized in the confusion matrix to further compute the evaluation metrics.

The concept of model evaluation using confusion matrix is illustrated in the following diagram:

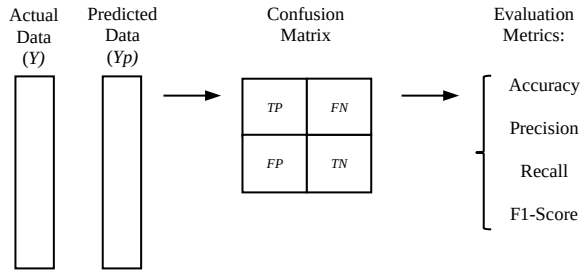

Fig 2: Explanation of Model Evaluation

Confusion Matrix:
Confusion matrix is a statistical table used to summarize the predictions of the applied model. The basic confusion matrix is the "binary" form (n=2) where it has two classes (Yes/No, Positive/Negative, Spam/Not Spam, etc.). It is represented as a matrix of size (2×2) as shown in the following diagram:

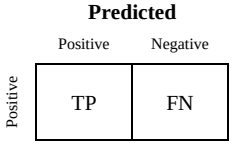

Fig 3: Explanation of Confusion Matrix

where: TP: is the number of true positive predictions,
TN: is the number of true negative predictions,
FP: is the number of false positive predictions, and
FN: is the number of false negative predictions.

Evaluation Metrics:

The evaluation metrics are statistical measures used to describe the performance of the applied model, for example: accuracy, precision, recall, and F-1 score. They are explained in the following section.

Accuracy:

The accuracy is defined as the number of true predictions divided by the number of predictions. It is computed by the following formula:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \quad (1)$$

Accuracy is simply illustrated in the following diagram:

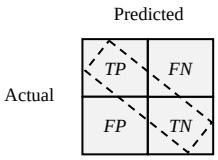

Fig 4: Explanation of Accuracy

Precision:

The precision is defined as the number of true positive predictions divided by the number of positive predictions. It is computed by the following formula:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2)$$

Precision is simply illustrated in the following diagram:

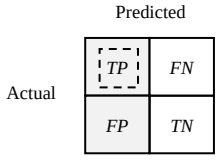

Fig 5: Explanation of Precision

Recall:

The recall is defined as the number of true positive predictions divided by the number of actual positive instances. It is computed by the following formula:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (3)$$

Recall is simply illustrated in the following diagram:

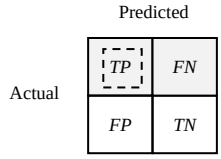

Fig 6: Explanation of Recall

F1-Score:

The F1-score is the harmonic mean of precision and recall. It is computed by the following formula:

$$\text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (4)$$

In general, for multiple classes (n>2), the confusion matrix is expanded to a matrix of size ($n \times n$) as shown in the following form:

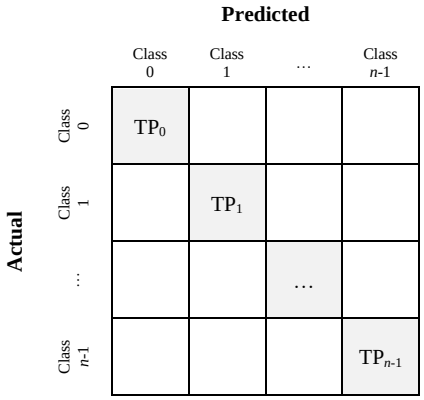

Fig 7: Confusion Matrix of Multiple Classes

where (TP_i) is the number of true positive predictions for class (i).

The totals (sum of items, diagonal, rows, and columns) are calculated as shown in the following diagram:

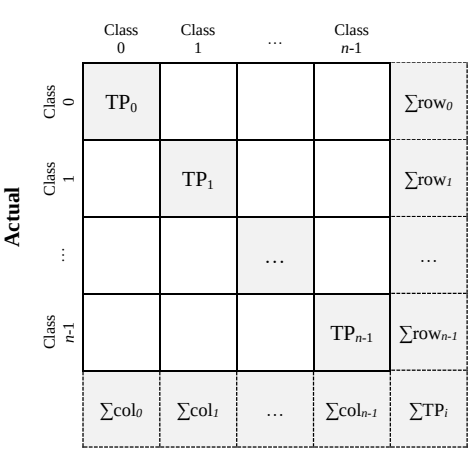

Fig 8: Computing Totals (Sum of Items, Diagonal, Rows, and Columns)

where: ($\sum TP_i$) is the sum of items in diagonal, ($\sum row_i$) is the sum of items in row (i), and ($\sum col_j$) is the sum of items in column (j).

Therefore, the evaluation metrics are computed by the following formulas:

$$\text{Accuracy} = \frac{\sum TP_i}{\text{Total}} \quad (5)$$

$$\text{Precision}_i = \frac{TP_i}{\sum col_i} \quad (6)$$

$$\text{Recall}_i = \frac{TP_i}{\sum row_i} \quad (7)$$

$$\text{F1-Score}_i = \frac{2 \times \text{Precision}_i \times \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} \quad (8)$$

Model Evaluation System:

The model evaluation system is explained in the following outline:

Input: Actual data (Y) and predicted data (Yp).
Output: Evaluation Metrics.
Processing: First, the predicted data is compared with the actual data to compute the confusion matrix. Then, the totals (sum of items, diagonal, rows, and columns) are computed. After that, the evaluation metrics (accuracy, precision, recall, and F1-score) are computed. Finally, the evaluation metrics are printed and the confusion matrix is plotted.

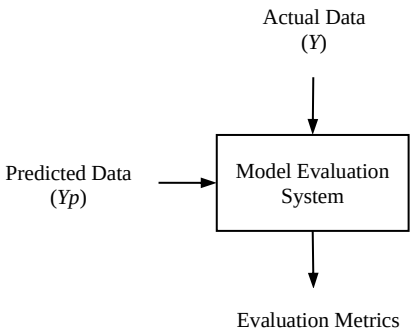

Fig 9: Diagram of Model Evaluation System

Python:

Python [29] is a general high-level programming language. It is very simple to code, easy to learn, and powerful. It is the most popular programming language especially for the development of machine learning applications.

Python provides additional libraries for different purposes for example: Numpy [30], Pandas [31], Matplotlib [32], NLTK [33], SciPy [34], and SK Learn [35].

Note: In this research, the standard functions of Python are used without any additional library.

3. RESEARCH METHODOLOGY

The basic steps of model evaluation using confusion matrix are: (1) preparing data (actual and predicted), (2) computing confusion matrix, (3) computing totals (sum of items, diagonal, rows, and columns), (4) computing evaluation metrics (accuracy, precision, recall, and F1-score), (5) printing evaluation metrics, and (6) plotting confusion matrix.

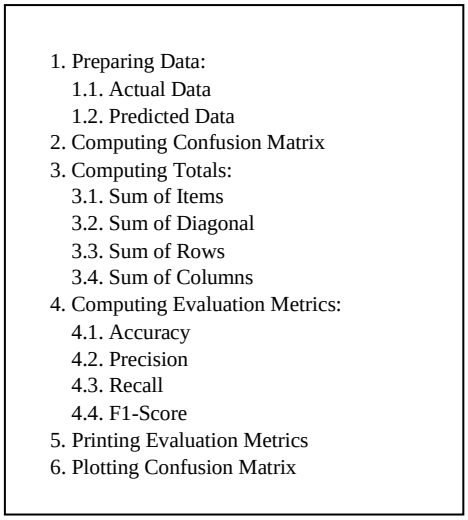

Fig 10: Steps of Model Evaluation

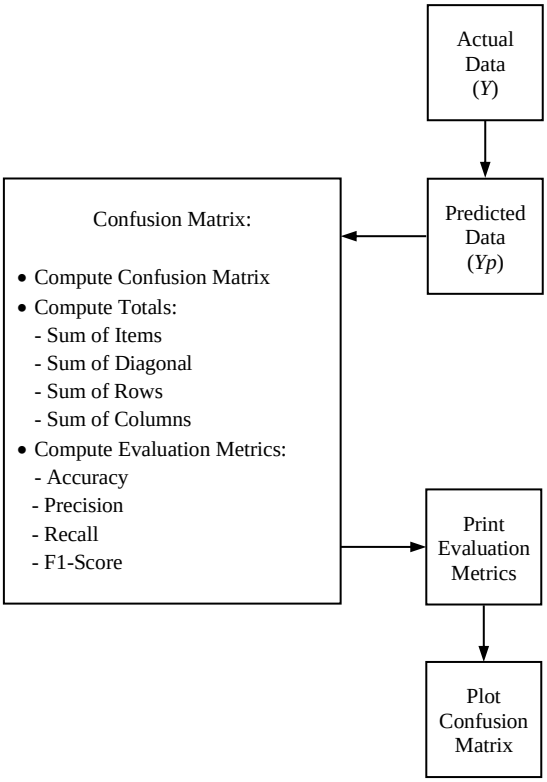

Fig 11: Flowchart of Model Evaluation

The basic steps of model evaluation using confusion matrix are explained in the following section.

1. Preparing Data:

The actual and predicted data are prepared by the following steps:

1.1. Actual Data:

The actual data (*Y*) is obtained from the original source and converted into list in the following form:

$Y = [y_0, y_1, y_2, \ldots, y_{n-1}]$

1.2. Predicted Data:

The predicted data (*Yp*) is obtained from the applied model and converted into list in the following form:

$Yp = [yp_0, yp_1, yp_2, \ldots, yp_{n-1}]$

2. Computing Confusion Matrix:

The confusion matrix is computed by the following code:

```
CM = zeros(n, n)
for i in range(len(Y)):
    CM[Y[i]][Yp[i]] += 1
```

3. Computing Totals:

The totals (sum of items, diagonal, rows, and columns) are computed by the following steps:

3.1. Sum of Items:

The sum of items (total) is computed by the following code:

```
def compute_total(CM):
    sum = 0
    for i in range(n):
        for j in range(n):
            sum += CM[i][j]
    return sum
```

3.2. Sum of Diagonal:

The sum of diagonal (diagonal) is computed by the following code:

```
def compute_diagonal(CM):
    sum = 0
    for i in range(n):
        sum += CM[i][i]
    return sum
```

3.3. Sum of Rows:

The sum of rows (sumrows) is computed by the following code:

```
def compute_sumrows(CM):
    sumrows = []
    for i in range(n):
        sum = 0
        for j in range(n):
            sum += CM[i][j]
        sumrows.append(sum)
    return sumrows
```

3.4. Sum of Columns:

The sum of columns (sumcols) is computed by the following code:

```
def compute_sumcols(CM):
    sumcols = []
    for j in range(n):
        sum = 0
        for i in range(n):
            sum += CM[i][j]
        sumcols.append(sum)
    return sumcols
```

4. Computing Evaluation Metrics:

The evaluation metrics (accuracy, precision, recall, and F1-score) are computed by the following steps:

4.1. Accuracy:

The accuracy is computed by the following code:

```
total = compute_total(CM)
diagonal = compute_diagonal(CM)
accuracy = diagonal/total
```

4.2. Precision:

The precision is computed by the following code:

```
def compute_precision(CM, sumcols):
    precision = []
    for i in range(n):
        precision.append(CM[i][i]/sumcols[i])
    return precision
```

4.3. Recall:

The recall is computed by the following code:

```
def compute_recall(CM, sumrows):
    recall = []
    for i in range(n):
        recall.append(CM[i][i]/sumrows[i])
    return recall
```

4.4. F1-Score:

The F1-score is computed by the following code:

```
def compute_f1score(precision, recall):
    f1score = []
    for i in range(n):
        f1score.append(2*precision[i]*recall[i]
                    /(precision[i] + recall[i]))
    return f1score
```

5. Printing Evaluation Metrics:

The evaluation metrics (accuracy, precision, recall, and F1-score) are printed by the following code:

```
print("Accuracy = ", accuracy)
print()
print("Class\tPrecision\tRecall\tF1-Score")
print("-"*32)
for i in range(n):
    print(f"{i}:
            \t{precision[i]:.2f}
            \t{recall[i]:.2f}
            \t{f1score[i]:.2f}")
print()
print("Average:")
print("-"*32)
print(f"\t{sum(precision)/n:.2f}
        \t{sum(recall)/n:.2f}
        \t{sum(f1score)/n:.2f}")
```

6. Plotting Confusion Matrix:

The confusion matrix is plotted (using the matplotlib library) by the following code:

```
import matplotlib.pyplot as plt

threshold = get_max(CM)/2
for i in range(n):
    for j in range(n):
```

```
        plt.text(j,i, CM[i][j],
        ha="center",
        va="center",
        color="white" if (CM[i][j] > threshold)
        else "black")

plt.imshow(CM, cmap="Blues")
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.xticks(range(n))
plt.yticks(range(n))
plt.colorbar()
plt.show()
```

4. RESULTS AND DISCUSSION

The developed program was tested on an experimental dataset from Kaggle [36]. The program performed the basic steps of model evaluation using confusion matrix and provided the required results. The resulting output is explained in the following section.

Actual and Predicted Data:

The actual data (*Y*) and the predicted data (*Yp*) are prepared and printed as shown in the following view:

```
        Y       Yp
------------------
0:      0       0
1:      0       0
2:      0       2
3:      1       1
4:      2       0
5:      2       2
6:      2       3
7:      3       3
8:      3       3
9:      3       3
...
```

Confusion Matrix:

The confusion matrix is computed and printed as shown in the following view:

```
Confusion Matrix:
0:      52      2       5       1
1:      3       28      2       1
2:      7       2       25      9
3:      2       0       12      40
```

Totals:

The totals (sum of items, diagonal, rows, and columns) are computed as shown in the following diagram:

Predicted

Actual	Class 0	Class 1	Class 2	Class 3	
Class 0	52	2	5	1	60
Class 1	3	28	2	1	34
Class 2	7	2	25	9	43
Class 3	2	0	12	40	54
	64	32	44	51	145

Fig 12: Totals (Sum of Items, Diagonal, Rows, and Columns)

Evaluation Metrics:

The evaluation metrics (accuracy, precision, recall, and F1-score) are computed and printed as shown in the following view:

```
Accuracy = 0.76

Class   Precision Recall    F1-Score
-----------------------------------
0:      0.81      0.87      0.84
1:      0.88      0.82      0.85
2:      0.57      0.58      0.57
3:      0.78      0.74      0.76

Average:
-----------------------------------
        0.76      0.75      0.76
```

The accuracy is (76%) which indicates that the model is highly correct in most of the predictions.

The precision, recall, and F1-score are (76%, 75%, and 76%) respectively. They indicate that the model is highly correct in most of the predictions.

The confusion matrix is plotted and displayed as shown in the following chart:

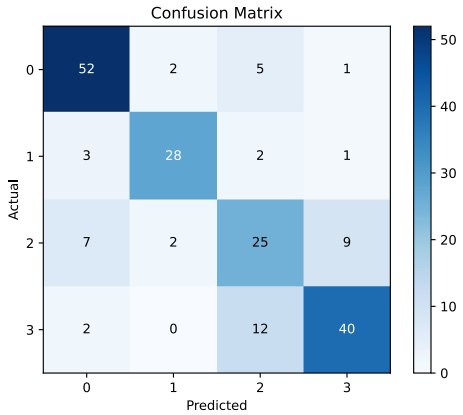

Fig 13: Confusion Matrix Plot

In summary, the program output shows that the program has successfully performed the basic steps of model evaluation using confusion matrix and provided the required results.

5. CONCLUSION

Machine learning is playing a major role in the development of computer systems. It helps to improve the performance and efficiency of computer programs.

Model evaluation is very important in the applications of machine learning. It is used to measure the performance of the applied model by comparing the predicted data with the actual data. Confusion matrix is used to summarize the predictions of the applied model and compute the evaluation metrics.

In this research, the author developed a program to perform model evaluation using confusion matrix in Python. The developed program performed the basic steps of model evaluation using confusion matrix: preparing data (actual and predicted), computing confusion matrix, computing totals (sum of items, diagonal, rows, and columns), computing evaluation metrics (accuracy, precision, recall, and F1-score), printing evaluation metrics, and plotting confusion matrix.

The program was tested on an experimental dataset and provided the required results: confusion matrix, totals (sum of items, diagonal, rows, and columns), and evaluation metrics (accuracy, precision, recall, and F1-score).

In future work, more research is really needed to improve the current methods of model evaluation using confusion matrix. In addition, they should be more investigated on different fields, domains, and datasets.

6. REFERENCES

[1] Sammut, C., & Webb, G. I. (2011). "Encyclopedia of Machine Learning". Springer Science & Business Media.

[2] Jung, A. (2022). "Machine Learning: The Basics". Singapore: Springer.

[3] Kubat, M. (2021). "An Introduction to Machine Learning". Cham, Switzerland: Springer.

[4] Li, H. (2023). "Machine Learning Methods". Springer Nature.

[5] Dey, A. (2016). "Machine Learning Algorithms: A Review". International Journal of Computer Science and Information Technologies, 7 (3), 1174-1179.

[6] Bonaccorso, G. (2018). "Machine Learning Algorithms: Popular Algorithms for Data Science and Machine Learning". Packt Publishing.

[7] Jo, T. (2021). "Machine Learning Foundations: Supervised, Unsupervised, and Advanced Learning". Springer.

[8] Jordan, M. I., & Mitchell, T. M. (2015). "Machine Learning: Trends, Perspectives, and Prospects". Science, 349(6245), 255-260.

[9] Forsyth, D. (2019). "Applied Machine Learning". Cham, Switzerland: Springer.

[10] Chopra, D., & Khurana, R. (2023). "Introduction to Machine Learning with Python". Bentham Science Publishers.

[11] Müller, A. C., & Guido, S. (2016). "Introduction to Machine Learning with Python: A Guide for Data Scientists". O'Reilly Media.

[12] Zollanvari, A. (2023). "Machine Learning with Python: Theory and Implementation". Springer Nature.

[13] Raschka, S. (2015). "Python Machine Learning". Packt Publishing.

[14] Sarkar, D., Bali, R., & Sharma, T. (2018). "Practical Machine Learning with Python". Apress.

[15] Swamynathan, M. (2019). "Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics using Python". Apress.

[16] Kong, Q., Siauw, T., & Bayen, A. (2020). "Python Programming and Numerical Methods: A Guide for Engineers and Scientists". Academic Press.

[17] Yale, K., Nisbet, R., & Miner, G. D. (2017). "Handbook of Statistical Analysis and Data Mining Applications". Elsevier.

[18] Unpingco, J. (2022). "Python for Probability, Statistics, and Machine Learning". Cham, Switzerland: Springer.

[19] Brandt, S. (2014). "Data Analysis: Statistical and Computational Methods for Scientists and Engineers". Springer.

[20] VanderPlas, J. (2017). "Python Data Science Handbook: Essential Tools for Working with Data". O'Reilly Media.

[21] James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). "An Introduction to Statistical Learning: With Applications in Python". Springer Nature.

[22] Aggarwal, C. C. (2020). "Data Classification: Algorithms and Applications". CRC Press

[23] Novaković, J. D., Veljović, A., Ilić, S. S., Papić, Ž., & Tomović, M. (2017). "Evaluation of Classification Models in Machine Learning". Theory and Applications of Mathematics & Computer Science, 7(1), 39.

[24] Zheng, A. (2015). "Evaluating Machine Learning Models: A Beginner's Guide to Key Concepts and Pitfalls". O'Reilly Media.

[25] Raschka, S. (2018). "Model Evaluation, Model Selection, and Algorithm Selection in Machine Larning". arXiv preprint arXiv:1811.12808.

[26] Rainio, O., Teuho, J. & Klén, R. (2024). "Evaluation Metrics and Statistical Tests for Machine Learning". Scientific Reports, 14 (1), 6086.

[27] Naidu, G., Zuva, T., & Sibanda, E. M. (2023). "A Review of Evaluation Metrics in Machine Learning Algorithms". In Computer Science On-line Conference (pp. 15-25). Cham: Springer International Publishing.

[28] Hossin, M., & Sulaiman, M.N, (2015). "A Review on Evaluation Metrics for Data Classification Evaluations". International Journal of Data Mining & Knowledge Management Process, 5, 01-11.

[29] Python: https://www.python.org

[30] Numpy: https://www.numpy.org

[31] Pandas: https:// pandas.pydata.org

[32] Matplotlib: https://www. matplotlib.org

[33] NLTK: https://www.nltk.org

[34] SciPy: https://scipy.org

[35] SK Learn: https://scikit-learn.org

[36] Kaggle: https://www.kaggle.com