Eva - Automatic Evaluation Metrics Analysis and Explanation for Binary Classification

Emanuel Shakarov, Rom Sharon
Tabular Data Science, Bar Ilan University

March 13, 2023

1 Abstract

Eva is a Python package designed to simplify the analysis and interpretation of evaluation metrics for binary classification tasks. The package automates the process of analyzing commonly used metrics such as accuracy, precision, recall, F1 score, and more, given predicted and actual labels with an optional probabilities vector from the model. Eva's output includes an analysis of the metrics, identifying which metric did not performed well, and providing explanations for potential reasons for such performances. By automating the analysis and interpretation of evaluation metrics, Eva streamlines the evaluation process and reduces the potential for human error and bias, in addition, it can help to get a better decisions about the models and improve overall performance. Eva is a valuable tool for machine learning engineers and data scientists.

2 Problem description

The problem that Eva addresses is the manual and often subjective process of evaluating binary classification models. The element of the data science (DS) pipeline that Eva seeks to improve is the evaluation stage. The evaluation stage is crucial in ensuring that the model is performing optimally and meeting the required performance criteria. However, the evaluation process is often time-consuming, requiring the manual analysis of multiple evaluation metrics. The analysis of these metrics can be subjective and prone to error, leading to inaccurate evaluations and sub-optimal model performance. Furthermore, the evaluation process can be hindered by the lack of clear guidelines on which metrics to prioritize and how to interpret their results. Eva aims to automate the evaluation process, providing an objective and standardized analysis of multiple evaluation metrics and offering suggestions for improving model performance. By doing so, Eva reduces the potential for human error and bias and streamlines the evaluation stage, improving the overall efficiency of the Data science pipeline.

3 Solution overview

The Eva package is a Python package that provides various evaluation metrics for binary classification models. It is easy to use, and the user only needs to pass the true labels, predicted labels, and predicted probabilities (optional) to the Eva class, which automatically computes various evaluation metrics. The package includes several evaluation metrics and each metric has a description, suggestion, and threshold value based on sensitivity level (low, medium, or high). The Eva class initializes all the evaluation metrics with their respective thresholds, descriptions, and suggestions and allows users to evaluate all the metrics using the evaluate() method. If any metric's value falls below its threshold, Eva prints a warning message with a description and a suggestion to improve the model's performance. The Eva package also provides various plot functions to visualize some of the metrics' results.

List of the current calculated evaluation metrics in Eva:

1. **Accuracy**

   Description: Accuracy calculates the proportion of correct predictions out of all the predictions made by the model
   Suggestion: Try to use a more complex model or to add more data to the training set, in addition, you can try to tune your hyperparametrs in order to get a better performance.
   Calculation: $\frac{TP+TN}{TP+FP+TN+FN}$
   Thresholds: low: 0.85, medium: 0.9, high: 0.95

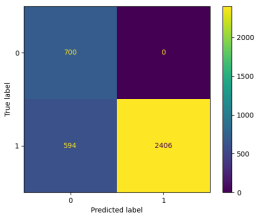

figure 1.1: Accuracy plot example

2. **Precision**

   Description: Precision measures how many observations predicted as positive are in fact positive
   Suggestion: Try to adjust the decision threshold of your model to improve the precision score of your binary classification task. Lowering the threshold can increase the number of true positives but may also increase false positives, while raising the threshold can decrease false positives but may also decrease true positives
   Calculation: $\frac{TP}{TP+FP}$
   Thresholds: low: 0.85, medium: 0.9, high: 0.95

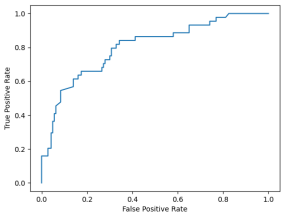

figure 1.2: Precision plot example

3. ## Recall

Description: recall calculates the proportion of true positive predictions out of all the actual positive instances

Suggestion: Try to increase the size of your dataset or apply data augmentation techniques. Increasing the size of your dataset can provide more examples of the positive class, which can improve the model's ability to identify true positives.

Calculation: $\frac{TP}{TP+FN}$

Thresholds: low: 0.85, medium: 0.9, high: 0.95

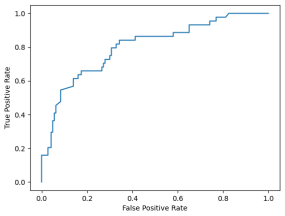

figure 1.3: Recall plot example

4. ## F1 score

Description: F1 score is an harmonic mean of precision and recall. It is commonly used when the dataset is imbalanced.

Suggestion: Try to balance precision and recall to improve F1 score by adjusting the decision threshold of your model or using algorithms that are specifically designed to optimize the F1 score.

Calculation: $\frac{2\times PERCISION\times RECALL}{PRECISION+RECALL}$

Thresholds: low: 0.85, medium: 0.9, high: 0.95

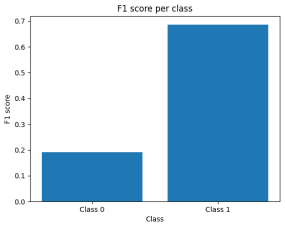

figure 1.3: Recall plot example

5. ## AUC (Area Under the ROC Curve)

Description: AUC (Area Under the ROC Curve) calculates the area under the ROC curve, which plots the true positive rate against the false positive rate, the closer to 1, the better. A score of 0.5 is equivalent to random guessing.

Suggestion: Try to use algorithms that directly optimize AUC to improve the AUC metric of your binary classification task. The ROC-AUC maximization algorithm is one such approach that can directly optimize AUC metric score.

Calculation: $\frac{\sum_{i=1}^{m-1}(FPR_{i+1}-FPR_i)\cdot(TPR_i+TPR_{i+1})}{2}$

Thresholds: Youden's J statistic used to emulate the AUC metric by calculating the area under a curve generated by plotting the Youden's J score against different threshold values. However, if the Youden's J score is less than 0.5(medium sensitivity), the AUC metric may perform poorly, indicating worse than random classifier performance.

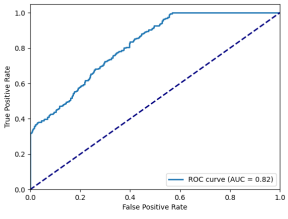

figure 1.3: Recall plot example

6. ## Matthew's Correlation Coefficient (MCC)

Description: Matthew's Correlation Coefficient (MCC) measures the quality of a binary classification by taking into account true positives, true

negatives, false positives, and false negatives and it is commonly used when the dataset is imbalanced. A high value for MCC (close to 1) indicates good performance, while a low value (close to 0 or below) indicates poor performance.

Suggestion: Try to adjust the decision threshold of the model to balance precision and recall. Feature selection or regularization techniques can also reduce the impact of irrelevant or noisy features on MCC.

Calculation: $\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

Thresholds: low: 0.3, medium: 0.35, high: 0.4

7. Mean Squared Error (MSE)

Description: Mean Squared Error (MSE) is a popular regression metric which measures the average squared difference between the true and predicted values.

Suggestion: Try to use more complex models like neural networks or gradient boosting to improve the Mean Squared Error (MSE) metric. These models can capture complex nonlinear relationships in the data and improve predictive accuracy.

Calculation: $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

Thresholds: If the mean squared error (MSE) score of a regression model is less than or equal to the variance of the true values y_{true}, then the model's performance can be considered poor. This is because the MSE measures the average squared difference between the predicted values and the true values, and if the MSE is close to or less than the variance of y_{true}, it indicates that the model is not accurately capturing the variability in the true values and is likely underfitting the data. In other words, the model is not complex enough to capture the underlying patterns in the data and is not able to make accurate predictions.

8. Brier Score

Description: Brier score is used to check the goodness of a predicted probability score. This is very similar to the mean squared error, but only applied for prediction probability scores, whose values range between 0 and 1.

Suggestion: Try to adjust the model's hyperparameters, such as the regularization strength, or to use a different algorithm that is better suited to the data, in addtion, try to increase the amount of data or improve the quality of the input features may also help to improve the model's performance.

Calculation: $\frac{1}{N}\sum_{i=1}^{N}(f_i - o_i)^2$

Thresholds: If the Brier score falls below the defined threshold for each sensitivity level, such as low: 0.18, medium: 0.15, high: 0.1, it suggests that the model's performance is poor.

4 Experimental evaluation

To evaluate the performance of Eva, we conducted four tests for each of four datasets, resulting in a total of 12 tests. Each test involved training a binary classification model using a different dataset and model configuration. The datasets are: **1.** Titanic dataset [4] **2.** Go To College dataset [5] **3.** Dataset Surgical binary classification [6] **4.** Breast Cancer Wisconsin (Diagnostic) Data Set [7]

1. **Test 1: Logistic Regression (80%/20%)**
   This is the basic test where the Logistic Regression model is trained on 80% of the dataset and tested on the remaining 20%. This test serves as a baseline to compare the performance of the other tests.

2. **Test 2: Logistic Regression (10%/90%)**
   In this test, the Logistic Regression model is trained on only 10% of the dataset and tested on the remaining 90%. The purpose of this test is to see how Eva performed when the model results are poorly.

3. **Test 3: Balanced Logistic Regression**
   In this test, the target variable is balanced with equal representation of both classes. This test helps to evaluate the performance of the model on a balanced dataset and to assess whether the model is biased towards the majority class in the original imbalanced dataset.

4. **Test 4: XGBoost (80%/20%)**
   This test uses the XGBoost algorithm to train the model on 80% of the dataset and test on the remaining 20%. The purpose of this test is to compare the performance of the Logistic Regression model with a more complex algorithm and to evaluate whether the additional complexity improves the model's performance.

For each test, we evaluated the performance of the model using Eva.evalute function. We created a bar chart for each metric with all 12 tests and plotted the threshold line to visually assess the performance of the models.

Overall, the evaluation of Eva demonstrated its ability to simplify the analysis and interpretation of evaluation metrics for binary classification tasks.

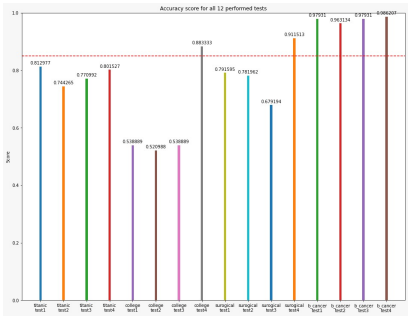

Figure 1: Accuracy metric results and the Eva's medium threshold in red.

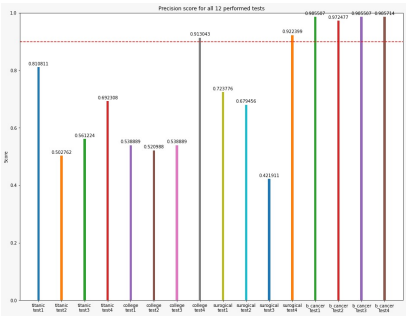

Figure 2: Precision metric results and the Eva's medium threshold in red.

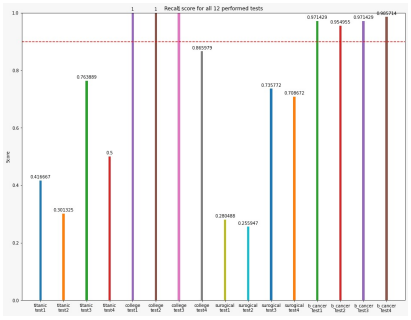

Figure 3: Recall metric results and the Eva's medium threshold in red.

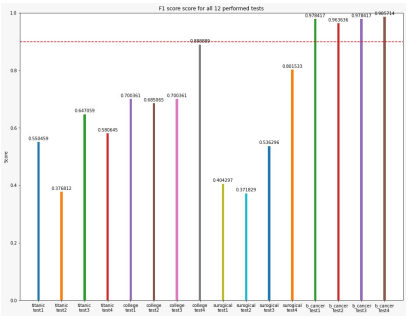

Figure 4: F1 metric results and the Eva's medium threshold in red.

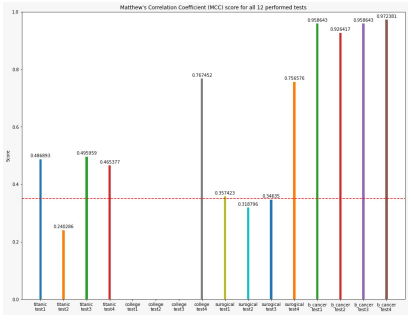

Figure 5: Matthew's Correlation Coefficient metric results and the Eva's medium threshold in red.

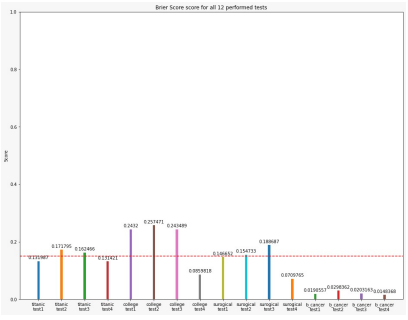

Figure 6: Brier score metric results and the Eva's medium threshold in red.

brier

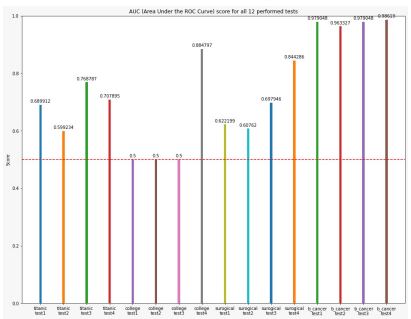

Figure 7: Area Under the ROC(AUC) Curve metric results and the Eva's medium threshold in red.

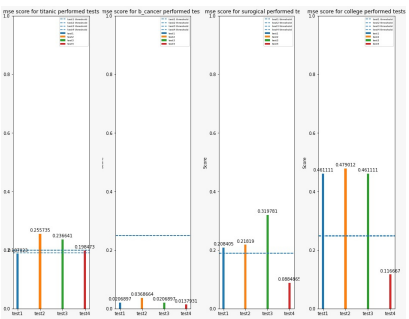

Figure 8: Mean Squared Error (MSE) metric results and Eva's medium threshold in red.

Based on the tests conducted, the results indicate that the breast cancer dataset performed well in almost all metrics (6/8), suggesting that it is a good dataset for binary classification tasks. Additionally, the results show that when XGBoost was used, it generally performed well in the majority of tests (25/32), indicating that it may be a good choice for binary classification tasks. Also, In test 3, where the data was balanced, the F1 score was higher than in test 1 with the basic model. This indicates that the model in test 3 performed better at finding true positives (i.e., correctly identifying positive cases) while also correctly identifying a higher proportion of all actual positives. Balancing the data likely helped the model better capture the patterns and relationships in the minority class, leading to improved performance.

5 Related Work

5.1 Evidently

Evidently is an open-source Python library for data scientists and ML engineers. It helps evaluate, test, and monitor the performance of ML models from validation to production. It works with tabular and text data. [2]. While Evidently and our proposed solution share the goal of monitoring machine learning performance, they differ in their specific approaches. Our solution, Eva, is designed to simplify the analysis and interpretation of evaluation metrics for binary classification tasks. Additionally, Eva offers automated analysis of commonly used metrics and identifies potential areas of improvement. It is possible that both libraries could be used together to provide a comprehensive solution for monitoring machine learning performance.

5.2 Evaluating machine learning models and their diagnostic value

The article "Evaluating machine learning models and their diagnostic value" [3] provides a comprehensive overview of model validation in machine learning. The authors emphasize the importance of model validation in selecting the best model and assessing the performance of a given model. The article describes various performance metrics for different machine learning tasks, including classification and regression, and explains how to interpret these metrics, taking into account class imbalance, varying prevalence, and asymmetric cost-benefit trade-offs.

Additionally, the authors discuss how to obtain confidence intervals of performance metrics, distinguishing between internal validation or evaluation of learning algorithms and external validation or evaluation of resulting prediction models. The article provides valuable insights into the process of model validation and can serve as a useful resource for data scientists and machine learning engineers.

Although Eva and this article both focus on model validation and performance evaluation, Eva is a Python package designed specifically to simplify the analysis and interpretation of evaluation metrics for binary classification tasks. In contrast, this article provides a broader overview of model validation and covers a wider range of machine learning tasks. While Eva can automate the analysis of commonly used metrics, this article provides a more in-depth explanation of the underlying principles and techniques for model validation. As such, both Eva and this article can complement each other in providing a comprehensive understanding of model validation and performance evaluation in machine learning.

6 Conclusion

In conclusion, Eva is a Python package that simplifies the analysis and interpretation of evaluation metrics for binary classification tasks. The package automates the process of analyzing commonly used metrics and provides an analysis of the metrics, identifying which metric did not perform well and gives a suggestion of how to improve it. Eva streamlines the evaluation process and reduces the potential for human error and bias. As a valuable tool for machine learning engineers and data scientists, Eva can help to make better decisions about the models and improve overall performance. During the project, we learned how to use Eva and how it can be useful in evaluating and interpreting model performance. We also gained insights into the importance of evaluation metrics and the potential impact on decision-making in machine learning projects. Eva can be a valuable addition to the tools available to data scientists and machine learning engineers for evaluating and improving model performance.

References

[1] Jakub Czakon. *Evaluating machine learning models and their diagnostic value.* Neptune.AI. https://neptune.ai/blog/evaluation-metrics-binary-classification. Submitted on January 31, 2023.

[2] EvidentlyAI, Evidently. (2023). *GitHub Repository.* https://github.com/evidentlyai/evidently.

[3] Varoquaux, G., Colliot, O. (2023). *Evaluating machine learning models and their diagnostic value.* HAL archives ouvertes. https://hal.archives-ouvertes.fr/hal-03682454v4. Submitted on January 21, 2023.

[4] Kaggle *Titanic dataset* . https://www.kaggle.com/datasets/heptapod/titanic.

[5] Kaggle *Go To College Dataset.* https://www.kaggle.com/datasets/saddamazyazy/go-to-college-dataset.

[6] Kaggle *Dataset Surgical binary classification.* https://www.kaggle.com/datasets/omnamahshivai/surgical-dataset-binary-classification.

[7] Kaggle *Breast Cancer Wisconsin (Diagnostic) Data Set.* https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data.