Metrics To Evaluate

There are standard metrics that are widely used for evaluating classification predictive models, such as classification accuracy or classification error.

Standard metrics work well on most problems, which is why they are widely adopted. But all metrics make assumptions about the problem or about what is important in the problem. Therefore an evaluation metric must be chosen that best captures what you or your project stakeholders believe is important about the model or predictions, which makes choosing model evaluation metrics challenging.

This challenge is made even more difficult when there is a skew in the class distribution. The reason for this is that many of the standard metrics become unreliable or even misleading when classes are imbalanced, or severely imbalanced, such as 1:100 or 1:1000 ratio between a minority and majority class.

Unlike standard evaluation metrics that treat all classes as equally important, imbalanced classification problems typically rate classification errors with the minority class as more important than those with the majority class. As such performance metrics may be needed that focus on the minority class, which is made challenging because it is the minority class where we lack observations required to train an effective model.

Classification Metrics

1.Classification Accuracy.
2.Log Loss.
3.Area Under ROC Curve.
4.Confusion Matrix.

1. Classification Accuracy

Classification accuracy is the number of correct predictions made as a ratio of all predictions made.
This is the most common evaluation metric for classification problems, it is also the most misused. It is really only suitable when there are an equal number of observations in each class (which is rarely the case) and that all predictions and prediction errors are equally important, which is often not the case.

2. Log Loss

Logistic loss (or log loss) is a performance metric for evaluating the predictions of probabilities of membership to a given class.
The scalar probability between 0 and 1 can be seen as a measure of confidence for a prediction by an algorithm. Predictions that are correct or incorrect are rewarded or punished proportionally to the confidence of the prediction.
(It is the negative log-likelihood of a logistic model that returns y_pred probabilities for its training data y_true. We multiply this by negative 1 to maintain a common convention that lower loss scores are better)

3. Area Under ROC Curve

Area Under ROC Curve (or ROC AUC for short) is a performance metric for binary classification problems.
The AUC represents a model's ability to discriminate between positive and negative classes. An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model as good as random.
A ROC Curve is a plot of the true positive rate and the false positive rate for a given set of probability predictions at different thresholds used to map the probabilities to class labels. The area under the curve is then the approximate integral under the ROC Curve.

4. Confusion Matrix

The confusion matrix is a handy presentation of the accuracy of a model with two or more classes.
The table presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.
For example, a machine learning algorithm can predict 0 or 1 and each prediction may actually have been a 0 or 1. Predictions for 0 that were actually 0 appear in the cell for prediction=0 and actual=0, whereas predictions for 0 that were actually 1 appear in the cell for prediction = 0 and actual=1. And so on.

Sensitivity-Specificity Metrics

Sensitivity refers to the true positive rate and summarizes how well the positive class was predicted.
Sensitivity = TruePositive / (TruePositive + FalseNegative)
Specificity is the complement to sensitivity, or the true negative rate, and summarises how well the negative class was predicted.
Specificity = TrueNegative / (FalsePositive + TrueNegative)

For imbalanced classification, the sensitivity might be more interesting than the specificity.
Sensitivity and Specificity can be combined into a single score that balances both concerns, called the geometric meanor G-Mean.
G-Mean = sqrt(Sensitivity * Specificity)

Precision

Precision is a metric that quantifies the number of correct positive predictions made.
It is calculated as the ratio of correctly predicted positive examples divided by the total number of positive examples that were predicted.
Precision = TruePositives / (TruePositives + FalsePositives)

Recall

Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made.
It is calculated as the ratio of correctly predicted positive examples divided by the total number of positive examples that could be predicted.
Recall = TruePositives / (TruePositives + FalseNegatives)

F-Measure

Precision and recall measure the two types of errors that could be made for the positive class.
Maximizing precision minimizes false positives and maximizing recall minimizes false negatives.
F-Measure or F-Score provides a way to combine both precision and recall into a single measure that captures both properties.
F-Measure = (2 * Precision * Recall) / (Precision + Recall)
This is the harmonic mean of the two fractions.

The intuition for F-measure is that both measures are balanced in importance and that only a good precision and good recall together result in a good F-measure.

Fbeta-Measure

The F-measure balances the precision and recall.

On some problems, we might be interested in an F-measure with more attention put on precision, such as when false positives are more important to minimize, but false negatives are still important.
On other problems, we might be interested in an F-measure with more attention put on recall, such as when false negatives are more important to minimize, but false positives are still important.

The solution is the Fbeta-measure.
The Fbeta-measure measure is an abstraction of the F-measure where the balance of precision and recall in the calculation of the harmonic mean is controlled by a coefficient called beta.

Fbeta = ((1 + beta^2) * Precision * Recall) / (beta^2 * Precision + Recall)
The choice of the beta parameter will be used in the name of the Fbeta-measure.
For example, a beta value of 2 is referred to as F2-measure or F2-score. A beta value of 1 is referred to as the F1-measure or the F1-score.

Three common values for the beta parameter are as follows:
F0.5-Measure(beta=0.5): More weight on precision, less weight on recall.
F1-Measure(beta=1.0): Balance the weight on precision and recall.
F2-Measure(beta=2.0): Less weight on precision, more weight on recall

BrierScore
The benefit of the Brier score is that it is focused on the positive class, which for imbalanced classification is the minority class. This makes it more preferable than log loss, which is focused on the entire probability distribution.

The Brier score is calculated as the mean squared error between the expected probabilities for the positive class (e.g. 1.0) and the predicted probabilities. Recall that the mean squared error is the average of the squared differences between the values.
BrierScore = 1/N * Sum i to N (yhat_i – y_i)^2

Are you predicting probabilities?

- **Do you need class labels?**
- **Is the positive class more important?**
  - Use Precision-Recall AUC
- **Are both classes important?**
  - Use ROC AUC
- **Do you need probabilities?**
  - Use Brier Score and Brier Skill Score
- **Are you predicting class labels?**
- **Is the positive class more important?**
- **Are False Negatives and False Positives Equally Important?**
  - Use F1-Measure
- **Are False Negatives More Important?**
  - Use F2-Measure

- **Are False Positives More Important?**
- Use F0.5-Measure
- **Are both classes important?**
- **Do you have < 80%-90% Examples for the Majority Class?**
- Use Accuracy
- **Do you have > 80%-90% Examples for the Majority Class?**
- Use G-Mean

Threshold

On a binary classification problem with class labels 0 and 1, normalized predicted probabilities and a threshold of 0.5, then values less than the threshold of 0.5 are assigned to class 0 and values greater than or equal to 0.5 are assigned to class 1.

- Prediction < 0.5 = Class 0
- Prediction >= 0.5 = Class 1

The problem is that the default threshold may not represent an optimal interpretation of the predicted probabilities.

This might be the case for a number of reasons, such as:

- The predicted probabilities are not calibrated, e.g. those predicted by an SVM or decision tree.
- The metric used to train the model is different from the metric used to evaluate a final model.
- The class distribution is severely skewed.
- The cost of one type of misclassification is more important than another type of misclassification

The ROC Curve is a useful diagnostic tool for understanding the trade-off for different thresholds and the ROC AUC provides a useful number for comparing models based on their general capabilities.

If crisp class labels are required from a model under such an analysis, then an optimal threshold is required. This would be a threshold on the curve that is closest to the top-left of the plot.

There are many ways we could locate the threshold with the optimal balance between false positive and true positive rates.

One approach would be to test the model with each threshold returned from the call roc_auc_score() and select the threshold with the largest G-Mean value.

It turns out there is a much faster way to get the same result, called the Youden's J statistic.
The statistic is calculated as:

- J = Sensitivity + Specificity – 1

Given that we have Sensitivity (TPR) and the complement of the specificity (FPR), we can calculate it

as:

- J = Sensitivity + (1 – FalsePositiveRate) – 1

Which we can restate as:

- J = TruePositiveRate – FalsePositiveRate

We can then choose the threshold with the largest J statistic value

If we are interested in a threshold that results in the best balance of precision and recall, then this is the same as optimizing the F-measure that summarizes the harmonic mean of both measures.

- F-Measure = (2 * Precision * Recall) / (Precision + Recall)

As in the previous section, the naive approach to finding the optimal threshold would be to calculate the F-measure for each threshold. We can achieve the same effect by converting the precision and recall measures to F-measure directly