http://blog.revolutionanalytics.com/2016/03/com\_class\_eval\_metrics\_r.html#macro

1. logLoss,
2. AUC,
3. Accuracy,
   1. It is defined as the fraction of instances that are correctly classified.
   2. accuracy = sum(diag) / n
4. Kappa,
   1. is a measure of agreement between the predictions and the actual labels. It can also be interpreted as a comparison of the overall acurracy to the expected random chance accuracy. The higher the Kappa metric is, the better your classifier is compared to a random chance classifier.
5. F1,
6. Sensitivity  (also known as recall),
7. Specificity,
8. Pos\_Pred-Value (also known as precision),
9. Neg\_Pred-Value,
10. Detection\_Rate,
11. Balanced\_Accuracy
12. Per-class Precision, Recall(sensitivity), and F-1

In order to assess the performance with respect to every class in the dataset, we will compute common per-class metrics such as precision, recall, and the F-1 score. These metrics are particularly useful when the class labels are not uniformly distributed (most instances belong to one class, for example). In such cases, accuracy could be misleading as one could predict the dominant class most of the time and still achieve a relatively high overall accuracy but very low precision or recall for other classes.

* 1. **Precision** is defined as the fraction of correct predictions for a certain class, whereas
  2. **Recall(Sensitivity)** is the fraction of instances of a class that were correctly predicted.

Notice that there is an obvious trade off between these 2 metrics. When a classifier attempts to predict one class, say class a, most of the time, it will achieve a high recall for a (most of the instances of that class will be identified). However, instances of other classes will most likely be incorrectly predicted as a in that process, resulting in a lower precision for a.

* 1. **F-1** score is also commonly reported. It is defined as the harmonic mean (or a weighted average) of precision and recall.
  2. precision = diag / colsums

recall = diag / rowsums

f1 = 2 \* precision \* recall / (precision + recall)

* 1. These metrics are computed for each class e.g

## precision recall f1

## a 0.8888889 0.8888889 0.8888889

## b 0.8108108 0.8108108 0.8108108

## c 0.8611111 0.8611111 0.8611111

Note that this is an example of multi-class classification evaluation and that some of the variables we compute are vectors that contain multiple values representing each class. For example, precision contains 3 values corresponding to the classes a, b, and c. The code can generalize to any number of classes.

However, in binary classification tasks, one would look at the values of the positive class when reporting such metrics. In that case, the overall precision, recall and F-1, are those of the positive class.

* 1. Macro-averaged Metrics

The per-class metrics can be averaged over all the classes resulting in macro-averaged precision, recall and F-1.

macroPrecision = mean(precision)

macroRecall = mean(recall)

macroF1 = mean(f1)

data.frame(macroPrecision, macroRecall, macroF1)

## macroPrecision macroRecall macroF1

## 1 0.8536036 0.8536036 0.8536036

* 1. Yyy
  2. huu

Then F1 can be easily computed, as stated above, as: F1 <- (2 \* precision \* recall) / (precision + recall)

https://towardsdatascience.com/what-metrics-should-we-use-on-imbalanced-data-set-precision-recall-roc-e2e79252aeba

Precision is more focused in the positive class than in the negative class, it actually measures **the probability of correct detection of positive values**, while FPR and TPR (ROC metrics) measure **the ability to distinguish between the classes**.

**Final intuition to metric selection**

1. **Use precision and recall to focus on small positive class —**When the positive class is smaller and the ability to detect correctly positive samples is our main focus (correct detection of negatives examples is less important to the problem) we should use precision and recall.
2. **Use ROC when both classes detection is equally important —** When we want to give equal weight to both classes prediction ability we should look at the ROC curve.
3. **Use ROC when the positives are the majority or switch the labels and use precision and recall —** When the positive class is larger we should probably use the ROC metrics because the precision and recall would reflect mostly the ability of prediction of the positive class and not the negative class which will naturally be harder to detect due to the smaller number of samples. If the negative class (the minority in this case) is more important, we can switch the labels and use precision and recall (As we saw in the examples above — switching the labels can change everything).

<https://www.ritchieng.com/machine-learning-evaluate-classification-model/>

**Conclusion:**

* Confusion matrix gives you a **more complete picture** of how your classifier is performing
* Also allows you to compute various **classification metrics**, and these metrics can guide your model selection

**Which metrics should you focus on?**

* Choice of metric depends on your **business objective**
  + Identify if FP or FN is more important to reduce
  + Choose metric with relevant variable (FP or FN in the equation)
* **Spam filter** (positive class is "spam"):
  + Optimize for **precision or specificity**
    - precision
      * false positive as variable
    - specificity
      * false positive as variable
  + Because false negatives (spam goes to the inbox) are more acceptable than false positives (non-spam is caught by the spam filter)
* **Fraudulent transaction detector** (positive class is "fraud"):
  + Optimize for **sensitivity**
    - FN as a variable
  + Because false positives (normal transactions that are flagged as possible fraud) are more acceptable than false negatives (fraudulent transactions that are not detected)

**Adjusting the classification threshold**

**Conclusion:**

* **Threshold of 0.5** is used by default (for binary problems) to convert predicted probabilities into class predictions
* Threshold can be **adjusted** to increase sensitivity or specificity
* Sensitivity and specificity have an **inverse relationship**
  + Increasing one would always decrease the other
* Adjusting the threshold should be one of the last step you do in the model-building process
  + The most important steps are
    - Building the models
    - Selecting the best model

### Receiver Operating Characteristic (ROC) Curves[¶](https://www.ritchieng.com/machine-learning-evaluate-classification-model/#8.-Receiver-Operating-Characteristic-(ROC)-Curves)

**Question:** Wouldn't it be nice if we could see how sensitivity and specificity are affected by various thresholds, without actually changing the threshold?

**Answer:** Plot the ROC curve.

* Receiver Operating Characteristic (ROC)
  + ROC curve can help you to **choose a threshold** that balances sensitivity and specificity in a way that makes sense for your particular context
  + You can't actually **see the thresholds** used to generate the curve on the ROC curve itself