A List of [Binary] Classification Models' Evaluation Metrics Calculated by WEKA

67 metrics

By: Noureddin Sadawi

31 Jul 2014

This is a list of metrics that WEKA generates for evaluating [binary] classifiers. Notice that a simple definition has been provided for each of them (definitions were extracted from WEKA's source files).

I have studied WEKA's source code and have worked out which metrics can be considered primary (i.e. calculated from the data) and which metrics are secondary (i.e. calculated from primary metrics). The results of this study are reported in the file: weka-params.pdf

1. **Basic performance stats - right vs wrong**
   (a) **Correct:** Gets the number of instances correctly classified (that is, for which a correct prediction was made).
   (b) **pct Correct:** Gets the percentage of instances correctly classified (that is, for which a correct prediction was made).
   (c) **pct Incorrect:** Gets the percentage of instances incorrectly classified (that is, for which an incorrect prediction was made).
   (d) **pct Unclassified:** Gets the percentage of instances not classified (that is, for which no prediction was made by the classifier).
   (e) **Incorrect:** Gets the number of instances incorrectly classified (that is, for which an incorrect prediction was made).
   (f) **Kappa:** Returns value of kappa statistic if class is nominal.
   (g) **Unclassified:** Gets the number of instances not classified (that is, for which no prediction was made by the classifier).
2. **IR stats**

(a) **Area Under ROC [requires class index]:** Returns the area under ROC for those predictions that have been collected in the evaluateClassifier(Classifier, Instances) method.

(b) **false Negative Rate [requires class index]:** Calculate the false negative rate with respect to a particular class.

(c) **false Positive Rate [requires class index]:** Calculate the false positive rate with respect to a particular class.

(d) **fMeasure [requires class index]:** Calculate the F-Measure with respect to a particular class.

(e) **num True Negatives [requires class index]:** Calculate the number of true negatives with respect to a particular class.

(f) **num True Positives [requires class index]:** Calculate the number of true positives with respect to a particular class.

(g) **numFalseNegatives [requires class index]:** Calculate number of false negatives with respect to a particular class.

(h) **num False Positives [requires class index]:** Calculate number of false positives with respect to a particular class.

(i) **precision [requires class index]:** Calculate the precision with respect to a particular class.

(j) **recall [requires class index]:** Calculate the recall with respect to a particular class.

(k) **true Negative Rate [requires class index]:** Calculate the true negative rate with respect to a particular class.

(l) **true Positive Rate [requires class index]:** Calculate the true positive rate with respect to a particular class.

3. **Weighted IR stats**

(a) **weighted Area Under ROC:** Calculates the weighted (by class size) AUC.

(b) **weighted False Negative Rate:** Calculates the weighted (by class size) false negative rate.

(c) **weighted False Positive Rate:** Calculates the weighted (by class size) false positive rate.

(d) **weighted FMeasure:** Calculates the macro weighted (by class size) average F-Measure.

(e) **weighted Precision:** Calculates the weighted (by class size) precision.

(f) **weighted Recall:** Calculates the weighted (by class size) recall.

(g) **weighted True Negative Rate:** Calculates the weighted (by class size) true negative rate.

(h) **weighted True Positive Rate:** Calculates the weighted (by class size) true positive rate.

4. **SF stats**

   (a) **SF Entropy Gain:** Returns the total SF, which is the null model entropy minus the scheme entropy.

   (b) **SF Mean Entropy Gain:** Returns the SF per instance, which is the null model entropy minus the scheme entropy, per instance.

   (c) **SF Mean Prior Entropy:** Returns the entropy per instance for the null model.

   (d) **SF Mean Scheme Entropy:** Returns the entropy per instance for the scheme

   (e) **SF Prior Entropy:** Returns the total entropy for the null model.

   (f) **SF Scheme Entropy:** Returns the total entropy for the scheme.

5. **Sensitive stats - certainty of predictions**

   (a) **relative Absolute Error:** Returns the relative absolute error.

   (b) **root Mean Squared Error:** Returns the root mean squared error.

   (c) **root Relative Squared Error:** Returns the root relative squared error if the class is numeric.

   (d) **mean Absolute Error:** Returns the mean absolute error.

6. **K&B stats**

   (a) **KB Information:** Return the total Kononenko & Bratko Information score in bits.

   (b) **KB Mean Information:** Return the Kononenko & Bratko Information score in bits per instance.

   (c) **KB Relative Information:** Return the Kononenko & Bratko Relative Information score.

7. **area Under PRC [requires class index]:** Returns the area under precision-recall curve (AUPRC) for those predictions that have been collected in the evaluateClassifier(Classifier, Instances) method.

8. **avg Cost:** Gets the average cost, that is, total cost of misclassifications (incorrect plus unclassified) over the total number of instances.

9. **confusion Matrix:** Returns a copy of the confusion matrix.
10. **correlation Coefficient:** Returns the correlation coefficient if the class is numeric.
11. **coverage Of Test Cases By Predicted Regions:** Gets the coverage of the test cases by the predicted regions at the confidence level specified when evaluation was performed.
12. **error Rate:** Returns the estimated error rate or the root mean squared error (if the class is numeric).
13. **getClassPriors:** Get the current weighted class counts.
14. **matthews Correlation Coefficient [requires class index]:** Calculates the matthews correlation coefficient (sometimes called phi coefficient) for the supplied class
15. **mean Prior Absolute Error:** Returns the mean absolute error of the prior.
16. **num Instances:** Gets the number of test instances that had a known class value (actually the sum of the weights of test instances with known class values
17. **prior Entropy:** Calculate the entropy of the prior distribution.
18. **root Mean Prior Squared Error:** Returns the root mean prior squared error.
19. **size Of Predicted Regions:** Gets the average size of the predicted regions, relative to the range of the target in the training data, at the confidence level specified when evaluation was performed
20. **total Cost:** Gets the total cost, that is, the cost of each prediction times the weight of the instance, summed over all instances.
21. **unweighted Macro Fmeasure:** Unweighted macro-averaged F-measure.
22. **unweighted Micro Fmeasure:** Unweighted micro-averaged F-measure.
23. **weighted Area Under PRC:** Calculates the weighted (by class size) AUPRC.
24. **weighted Matthews Correlation:** Calculates the weighted (by class size) matthews correlation coefficient.
25. Number_of_training_instances

26. Number_of_testing_instances
27. Elapsed_Time_training
28. Elapsed_Time_testing
29. UserCPU_Time_training
30. UserCPU_Time_testing
31. Serialized_Model_Size
32. Serialized_Train_Set_Size
33. Serialized_Test_Set_Size