Evaluation of classifier performance

As different algorithms have varying strengths and weaknesses, it is necessary to use tests that reveal the distinctions among the learners when measuring how a learner will perform on future data.

To measure classification performance we used a measure of accuracy that divided the proportion of correct predictions by the total number of predictions. This number indicates the percentage of cases, where the learning was done correctly or otherwise. For example, suppose a classifier correctly identified, whether or not 99990 out of 100,000 newborn babies are carriers of a treatable genetic defect. This would mean an accuracy of 99.99% and an error rate of only 0.01 percent.

This appears to indicate an extremely accurate classifier. What if the genetic defect is found in only 10 out of every 100,000 babies? A test that predicts 'no defect' regardless of the circumstances will still be correct for 99.99 percent of all cases. In this case, even though the predictions are correct for the large majority of data, the classifier is not very useful for it's intended purpose, which is to identify children with birth defects. This is one consequence of the 'class imbalance problem' which refers to the issues associated with data having a large majority of records belonging to a single class.

The best measure of classifier performance is whether the classifier is successful at it's intended purpose. For this reason, it is crucial to have measures of model performance that measure utility rather than only accuracy.

A confusion matrix is a table that categorizes predictions according to whether they match the actual value in the data. One of the table's dimensions indicates the possible categories of predicted values while the other dimension indicates the same for actual values. A confusion matrix can be created for a model predicting any number of classes.

When the predicted value is the same as the actual value, this is a correct classification. Correct predictions fall on the diagonal in the confusion matrix. The off-diagonal matrix cells indicate cases where the predicted value differs from the actual value. These are incorrect predictions. Performance measures for classification models are based on the counts of predictions falling on and off the diagonal in these tables.

The most common performance measures consider the model's ability to discern one class versus all others. The class of interest is known as the positive class, while all others are known as negative. Note that the use of the terminology positive and negative is not intended to imply any value judgement (i.e. good versus bad), nor does it necessarily suggest that the outcome is present or absent (i.e. birth defect versus none). The choice of the positive outcome can even be arbitrary, as in cases where a model is predicting categories such as sunny versus rainy, or dog versus cat etc.

The relationship between positive class and negative class predictions can be shown as a 2×2 confusion matrix that tabulates whether predictions fall into categories -

True Positive (TP): Correctly classified as class of interest
True Negative (TN): Correctly classified as not class of interest
False Positive (FP): Incorrectly classified as class of interest
False Negative (FN): Incorrectly classified as not class of interest

For the birth defect classifier mentioned previously, the confusion matrix would tabulate whether the model's predicted birth defect status matches the patient's actual birth defect status, as shown -

		Predicted Birth Defect	
		no	yes
Actual Birth Defect	no	TN	FP
	yes	FN	TP

Using confusion matrices to measure performance -

With the 2x2 confusion matrix, the definition of accuracy, called success rate is given as

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

In this formula, the terms refer to the number of times the model's predictions fell into each of these categories.

The error rate, or the proportion of incorrectly classified examples is given as

$$\text{Error rate} = 1 - \text{accuracy} = \frac{FP + FN}{TP + TN + FP + FN}$$

Consider a dataset that includes the text of SMS messages along with a label indicating whether the message is unwanted. Junk messages are labeled spam, while legitimate messages are labeled ham. (Ref. - J.M. Gomez Hidalgo, T.A. Almeida and A. Yamakami, "On the validity of a New SMS Spam Collection", Proceedings of the 11th IEEE International conference on Machine Learning and Applications, 2012.)

The confusion matrix values are as shown below

$$\begin{array}{ccc} & \text{ham} & \text{spam} \\ \text{ham} & 1202 & 5 \\ \text{spam} & 29 & 154 \end{array}$$

$$\text{Here, accuracy} = \frac{154+1202}{154+1202+5+29} = 0.9755$$

$$\text{error rate} = \frac{5+29}{154+1202+5+29} = 0.0245$$

Beyond accuracy, countless measures have been developed and used for specific purposes in disciplines as diverse as medicine, information retrieval, marketing, signal detection theory.

Sensitivity and Specificity - Classification often involves a balance between being overly conservative and overly aggressive in decision making. For example, an e-mail filter could guarantee to eliminate every spam message by aggressively eliminating nearly every ham message at the same time. On the other hand, a guarantee that no ham messages will be inadvertently filtered might allow an unacceptable amount of spam to pass through the filter. This tradeoff is captured by sensitivity and specificity.

The sensitivity of a model, also called the true positive rate, measures the proportion of positive examples that were correctly classified. It is given as

$$\text{sensitivity} = \frac{TP}{TP + FN}$$

The specificity of a model, also called the true negative rate, measures the proportion of negative examples that were correctly classified. It is given as

$$\text{specificity} = \frac{TN}{TN + FP}$$

Based on the confusion matrix for the SMS classifier data, we get

$$\text{sensitivity} = \frac{154}{154 + 29} = 0{\cdot}8415$$

$$\text{specificity} = \frac{1202}{1202 + 5} = 0{\cdot}9958$$

Sensitivity and specificity range from 0 to 1, with values close to 1 being more desirable. However, it is important to find an appropriate balance between the two, which is often a context-specific task.

For example, sensitivity of 0·8415 implies that 84% of spam messages were correctly classified. Similarly the specificity of 0·996 implies that 99·6% of ham messages were correctly classified i.e. 0·4% of valid messages were rejected as spam. It is based on the intended purpose, if rejecting 0·4% of valid SMS messages is acceptable or otherwise.

Precision and recall - Closely related to sensitivity and specificity are other two performance measures, related to the compromises made in classification - precision and recall. These statistics are primarily used in the context of information retrieval, where in they provide an indication of how interesting and relevant a model's results are, or whether the predictions are diluted by meaningless noise.

The precision, also known as the positive predictive value, is defined as the proportion of positive examples that are truly positive. In other words, when a model predicts the positive class, how often is it correct ? A precise model will only predict the positive class in cases, very likely to be positive. It will be very trustworthy. Consider, what would happen if the model was very imprecise. Overtime, the results would be less likely to be trusted. In the context of information retrieval, this would be similar to a search engine returning unrelated results. Eventually, the users would switch to some other search engine. In the case of SMS spam filter, high precision means that the model is carefully able to target only the spam, while ignoring the ham.

$$\text{precision} = \frac{TP}{TP + FP}$$

Recall is a measure of how complete the results are.

$$\text{recall} = \frac{TP}{TP + FN}$$

This expression is the same as sensitivity, only the interpretation differs.

A model with high recall captures a large portion of the positive examples, meaning that it has wide breadth. For example, a search engine with high recall returns a large number of documents pertinent to the search query. Similarly, the SMS spam filter has high recall if the majority of spam messages are correctly identified.

For the given confusion matrix, assuming that spam is the positive class,

$$\text{precision} = \frac{154}{154+5} = 0.9685$$

$$\text{recall} = \frac{154}{154+29} = 0.8415$$

Similar to the inharent tradeoff between sensitivity & specificity, for most real world problems, it is difficult to build a model with both high precision and high recall. It is easy to be precise if you target only the low hanging fruit' - the easy to classify examples. Similarly, it is easy for a model to have high recall by casting a very wide net, meaning that the model is overly aggressive at predicting the positive cases. In contrast, having both high precision and recall at the same time is very challenging. It is therefore important to test a variety of models in order to find the combination of precision and recall to meet the needs of the project.

F-measure - A measure of model performance that combines precision and recall into a single number is known as the F-measure or F-score.

The formula for F-measure is

$$F\text{-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{recall} + \text{precision}}$$

$$= \frac{2 \times TP}{(2 \times TP) + FP + FN}$$

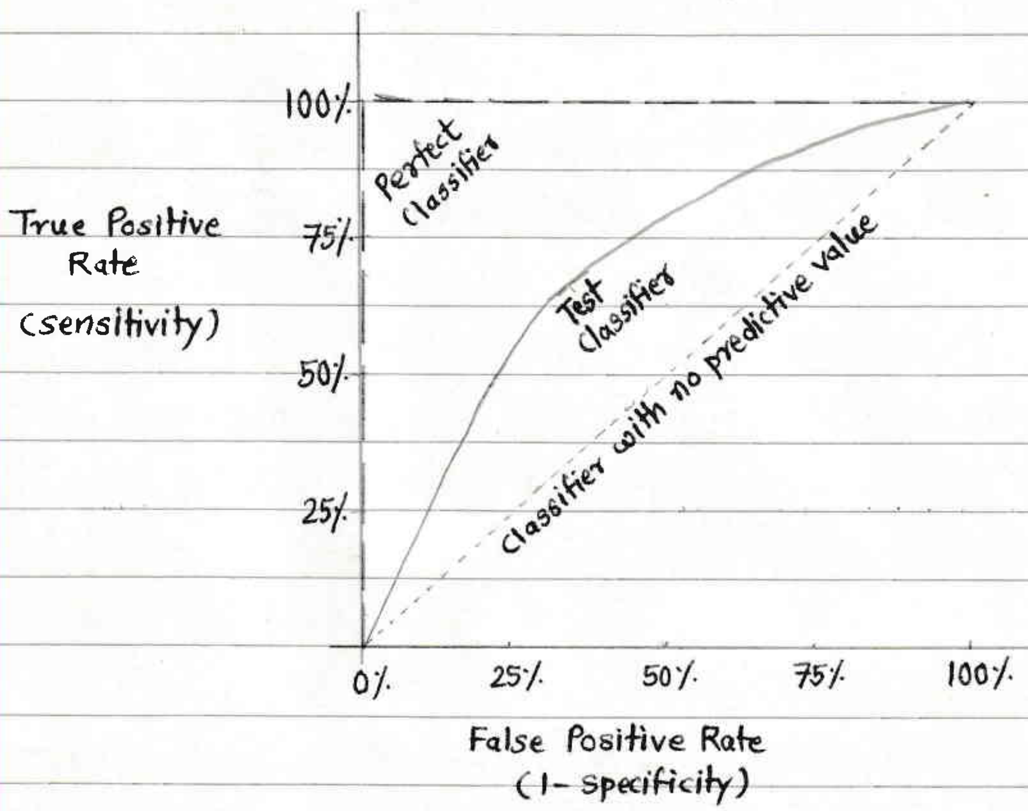
Using the data from the confusion matrix,

$$F\text{-measure} = \frac{2 \times 154}{(2 \times 154) + 5 + 29} = 0{\cdot}9005$$

Since the F-measure reduces model performance to a single number, it provides a convenient way to compare several models side-by-side. However, this assumes that equal weight should be assigned to precision and recall, which is not a valid assumption everytime. It is possible to calculate F-scores using different weights for precision and recall, but it becomes an arbitrary activity. A better practice is to use such measures in combination with methods that consider a model's strength and weaknesses more globally.

ROC curves – The ROC curve (Receiver Operating Characteristic) is commonly used to examine the tradeoff between the detection of true positives, while avoiding false positives. The ROC curves were developed by engineers in the field of communications around the time of world war II. Receivers of radar and radio signals needed a method to discriminate between true signals and false alarms. The same technique is useful today for visualizing the efficiency of machine learning models.

The characteristics of a typical ROC diagram are as shown. Curves are defined on a plot with the proportion of true positives on the vertical axis, and the proportion of false positives on the horizontal axis. Because these values are equivalent to 'sensitivity' and '1-specificity', the diagram is also known as a sensitivity/specificity plot.



The points comprising ROC curves indicate the true positive rate at varying false positive thresholds. To create the curves, a classifier's predictions are sorted by the model's estimated probability of the positive class, with the largest values first. Beginning at the origin, each prediction's impact on the true positive rate and false positive rate will result in a curve tracing vertically (for a correct prediction) or else horizontally (for an incorrect prediction).

For illustrating this concept, three hypothetical classifiers are compared in the ROC plot. First, the diagonal line from the bottom-left to the top-right corner of the diagram represents a classifier with no predictive value. This type of classifier detects true positives and false positives at exactly the same rate, implying that the classifier cannot discriminate between the two. This is the baseline by which other classifiers may be judged. ROC curves falling close to this line indicates models that are not very useful. Similarly the perfect classifier has a curve that passes through the point at 100% true positive rate and 0% false positive rate. It is able to correctly identify all of the true positives before it incorrectly classifies any negative result. Most real-world classifiers are similar to the test classifier, and they fall somewhere in the zone in between perfect and useless.

The closer the curve is to the perfect classifier, the better it is at identifying positive values. This can be measured using a statistic known as the 'area under the ROC curve' (AUC). It ranges from 0·5 for a classifier with no predictive value to a 1·0 for a perfect classifier.

A convention for interpreting AUC scores uses a system similar to academic letter grades.

0·9 – 1·0 = A (Outstanding)
0·8 – 0·9 = B (Excellent/good)
0·7 – 0·8 = C (Acceptable/fair)
0·6 – 0·7 = D (Poor)
0·5 – 0·6 = F (No discrimination)

Classifier performance -

It is important to know how well our classifier performs. The performance of a classifier is a compound characteristic, whose most important component is the classification accuracy. If we were able to try the classifier on all possible input objects, we would know exactly how accurate it is. Unfortunately, this is hardly a possible scenario, so an estimate of the accuracy is used instead.

Assume that a labeled data set Z of size $N \times n$ is available for testing the accuracy of our classifier D. The most natural way to calculate an estimate of the error, is to run D on all the objects in Z and find the proportion of misclassified objects. To look at the error from a probabilistic point of view, we can adopt the following model. The classifier commits an error with probability P_D on any object $x \in \mathbb{R}^n$, which is a useful assumption. Then the estimate of P_D is

$$\hat{P}_D = \frac{N_{error}}{N}$$

For the given data set Z of size $N \times n$, containing n-dimensional feature vectors describing N objects. We would like to use as much as possible of the data to build the classifier (training), and also as much as possible unseen data to test it's performance more thoroughly (testing). However, if we use all data for training and the same data for testing, we might overtrain the classifier so that it perfectly learns the available data and fails on unseen data. That is why it is important to have a separate data set on which to examine the final product. The main alternatives for making the best use of Z can be summarized as follows-

1> Resubstitution (R- method) - Design classifier D on Z and test it on Z. $\hat{P}_D$ is optimistically biased.

2> Holdout (H- method) - Traditionally split Z into halves, use one half for training, and the other half for calculating $\hat{P}_D$. Splits in other proportions are also used in practice. We can swap the two subsets, get another estimate $\hat{P}_D$ and average the two. A version of this method is the data shuffle, where we do L random splits of Z into training and testing parts and average all L estimates of P_D, as calculated on the respective testing parts.

3> K-Fold Cross-validation (K Fold CV) (Rotation method or π-method) - We choose an integer K (most preferably a factor of N) and randomly divide Z into K subsets of size N/K. Then we use one subset to test the performance of D trained on the union of the remaining K-1 subsets. This procedure is repeated K times, choosing a different part for testing each time. To get the final value of $\hat{P}_D$, we average the K estimates. When K=N, the method is called the 'Leave-one-out' method.

4> Bootstrap (B- method) - This is done by randomly generating L sets of cardinality N from the original set with replacement. Then we assess and average the error rate of the classifiers built on these sets.