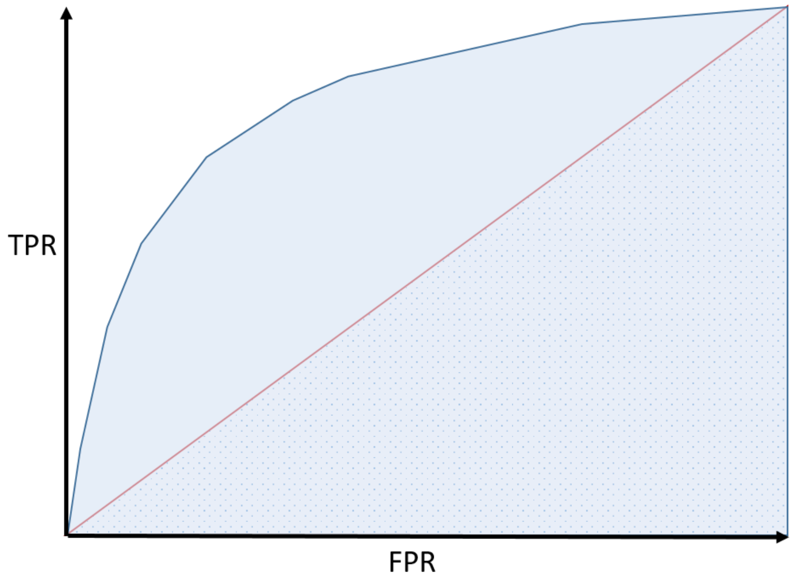
Microsoft: DAT203x Data Science and Machine Learning Essentials

KEY POINTS

- For a classification function to work accurately, when operating on the data in the training and test datasets, the number of times that the sign of ***f(x)*** does not equal ***y*** must be minimized. In other words, for a single entity, if ***y*** is positive, ***f(x)*** should be positive; and if ***y*** is negative, ***f(x)*** should be negative. Formally, we need to minimize cases where $y \neq sign(f(x))$.
- Because same-signed numbers when multiplied together always produce a positive, and numbers of different signs multiplied together always produce a negative, we can simplify our goal to minimize cases where $yf(x) < 0$; or for the whole data set $\Sigma_i y_i f(x_i) < 0$. This general approach is known as a loss function.
- As with regression algorithms, some classification algorithms add a regularization term to avoid over-fitting so that the function achieves a balance of accuracy and simplicity.
- Each classification algorithm (for example AdaBoost, Support Vector Machines, and Logistic Regression) uses a specific loss function implementation, and it's this that distinguishes classification algorithms from one another.
- Decision Trees are classification algorithms that define a sequence of branches. At each branch intersection, the feature value (***x***) is compared to a specific function, and the result determines which branch the algorithm follows. All branches eventually lead to a predicted value (-1 or +1). Most decision trees algorithms have been around for a while, and many produce low accuracy. However, boosted decision trees (AdaBoost applied to a decision tree) can be very effective.
- You can use a "one vs. all" technique to extend binary classification (which predicts a Boolean value) so that it can be used in multi-class classification. This approach involves applying multiple binary classifications (for example, "is this a chair?", "is this a bird?", and so on) and reviewing the results produced by ***f(x)*** for each test. Since ***f(x)*** produces a numeric result, the predicted value is a measure of confidence in the prediction (so for example, a high positive result for "is this a chair?" combined with a low positive result for "is this a bird?" and a high negative result for "is this an elephant?" indicates a high degree of confidence that the object is more likely to be a chair than a bird, and very unlikely to be an elephant.)
- When the training data is imbalanced (so a high proportion of the data has the same True/False value for ***y***) , the accuracy of the classification algorithm can be compromised. To overcome this problem, you can "over-sample" or "weight" data with the minority ***y*** value to balance the algorithm.
- The quality of a classification model can be assessed by plotting the *True Positive Rate* (the number of positives that were classified by the ML algorithm as positives divided by total number of positives) against the *False Positive Rate* (the number of negatives that were classified by the ML algorithm as positives divided by the total number of negatives) for various parameter values on a chart to create a receiver operator characteristic (ROC) curve. The quality of the model is reflected in the area under the curve. The larger this area is than the area under a straight diagonal line (representing a 50% accuracy rate that can be achieved purely by guessing), the better the model; as shown below:


TPR
FPR