## Cost-sensitive example weighting

* Method: altering the original example distribution by multiplying it by a factor proportional to the relative cost of each example
* Transparent Box: supply the weights (costs) directly to the classification algorithm – can’t be applied to all classifiers, gives good results
* Black Box: resample according to the weights – can be applied to all classifiers, sometimes leads to overfitting due to repeated examples
* Cost-proportional rejection sampling: uses black box, achieves at least as good cost minimization as base classifier applied on whole sample, runtime savings allow to use costing
* Costing – running the classifier on multiple subsamples and averaging the results; allows us to use any cost-insensitive classifier in order to accomplish cost-sensitive learning; achieves very good results; computational time savings
* We formulate the cost-sensitive learning in terms of one number per example instead of cost matrix (more general approach)  
  A text on a white background

  Description automatically generated
* Instead of TP, FP, TN and FN we have only two entries: (FP, TN) or (FN, TP), which can be further reduced to (FP-TN) or (FN-TP) – those differences will be our importance*c* of correct classification  
    
  A math problem with a number of lines and symbols

  Description automatically generated with medium confidence  
  A black text on a white background

  Description automatically generated  
  A math equations and formulas

  Description automatically generated
* Right side of translation theorem – the expectation we want to control through the choice of h (classifier); left side – probability that h errs under another distribution
* Choosing h to minimize the rate of errors under the resampled distribution = choosing h to minimize the expected cost under the original distribution
* Transparent box: easy to apply for classifiers that follow the statistical query model, for example neural nets, decision trees, naïve bayes classifier; it can be difficult to apply for classifiers which are dependent on individual examples rather than on statistics derived from entire sample (SVM)  
  A diagram of a cloud with arrows

  Description automatically generated
* Black box: used when we don’t have transparent box access to classification model; classical sampling with replacement leads to overfitting while sampling without replacement leads to obtaining exactly same distribution or smaller sample;
* COST-PROPORTIONATE REJECTION SAMPLING:
  + D – original distribution, D^ - resampled distribution
  + We sample from D and we keep the sample with probability c/Z, where Z is a constant chosen to satisfy

  
S – training set

* + We obrain set S’ as a result  
    A white background with black text

    Description automatically generated  
    A math equations on a white background

    Description automatically generated
  + Guarantees obtaining an approximately cost-minimizing classifier
* COST-PROPORTIONATE REJECTION SAMPLING WITH AGGREGATION (COSTING) – cprs produces different training set each time, and each time it is very small; to take advantage of that, we can create an ensemble learning algorithm in order to improve performance  
  A white background with black text

  Description automatically generated

## Instance weighting – cost-sensitive trees

* N – number of instances in the training set, Ni – number of instances of class i, Nj – number of instances of class j
* We want to assign an initial weight of an instance proportional to cost of missclassifying its class; sum of all weights = N  
  A math equations on a white background

  Description automatically generated
* We use the standard tree procedure, but instead of number of instances Nj(t) at given node, we use Wj(t) when computing the test selection criterion (calculating entropy) and estimating the error in the pruning process
* Instead of minimizing the number of errors, we minimize the number of errors with high cost – as a result, usually the number of low cost errors is increased
* C4.5 already uses Wj(t) instead of Nj(t), the only difference for C4.5CS is the initialization of weights
* Cost matrix needs to be converted to cost vector in order to use (2)
* High-cost errors – missclassification cost > 1
* According to experiments conducted in [Ting 1998], the cost-sensitive C4.5 performes better than C4.5 for binary classification, but comparably in case of multiple classes (due to cost matix -> cost vector conversion)
* This weighthing method can be regarded as sampling because the instances with weights > 1 can be viewed as instance duplication