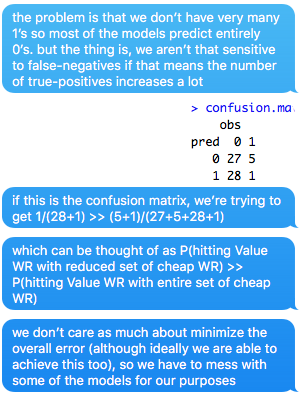
**classifyValueWR.R**



Basically, we want to remove the players that have no chance of hitting.

**1. http://www.win-vector.com/blog/2015/02/does-balancing-classes-improve-classifier-performance/**

Accuracy is simply the fraction of datums classified correctly.

Precision is the fraction of datums classified as positive that really were; equivalently, it’s an estimate of the conditional probability of a datum being in the positive class, given that it was classified as positive.

Recall (also called sensitivity or the true positive rate) is the fraction of positive datums in the population that were correctly identified.

Specificity is the true negative rate, or one minus the false positive rate: the number of negative datums correctly identified as such.

**2. http://stats.stackexchange.com/questions/38412/the-general-approaches-for-improving-a-svm-based-classifier-which-is-low-precisi**

For example, if I'm classifying 100 data points, 95 of which belong to class A, and 5 of which belong to class B, many machine learning algorithms (SVM included) will just classify everything/most things as class A, **yielding great recall but awful precision.**

Used this paper to solve: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1839342/

"...Since we will be dealing with very unbalanced numbers of positive and negative examples, we introduce cost factors C\_+ and C\_- to be able to adjust the cost of false positives vs. false negatives...". (paper: http://www.cs.cornell.edu/People/tj/publications/morik\_etal\_99a.pdf). **When C\_+ > C\_-, precision is increased.**

**Now, if you want just an out-of-the-box solution, the ratio of C\_+ to C\_- can be passed as the -j parameter to SVMLight: http://svmlight.joachims.org/**

Consider sparse matrix transformations too.