2. Classification Evaluation

References
T. Mitchell. Machine Learning. Chapter 5

2.1 Statistical Evaluation

Performance evaluation in classification based on accuracy or error rate.

Consider a typical classification problem:

f: X→Y

X instance space

D: prob distribution over X

S: dataset: a sample from X

Consider a hypothesis h, **solution of a learning algorithm** obtained from S and estimate the accuracy:

Errors

True error: is the prob that h will misclassify an instance drawn at random according to D. (true error cannot be computed)

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[f(x) \neq h(x)]$$

Sample error: is the proportion of examples h misclassifies (computed only on a small data sample)

$$error_S(h) \equiv \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

where delta = 1 if f(x)≠h(x), 0 otherwise. **Accuracy(h)** = 1 - error(h)

the goal of a learning system is to be accurate in h(x), for every x not in S
If accuracyS (h) is very high, but accuracyD(h) is poor, then our system would not be very useful.

Probabilities

bias = E[errorS(h)]-errorD(h)

1. If S is the training set used to compute h, errorS (h) is optimistically biased
2. For unbiased estimate, h and S must be chosen independently
   E[errorS (h)] = errorD(h)
3. Even with unbiased S, errorS (h) may still vary from errorD(h).
   The smaller the set S, the greater the expected variance.

Confidence intervals

With approximately N% probability, $error_{\mathcal{D}}(h)$ lies in interval

$$error_S(h) \pm z_N \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

where

$N\%$:	50%	68%	80%	90%	95%	98%	99%
z_N:	0.67	1.00	1.28	1.64	1.96	2.33	2.58

Estimators

How to compute errorS (h)

1. Partition the data set D (D = T Union S, T Inters. S = Vuoto, |T| = 2/3|D|)
2. Compute a hypothesis h using training set T
3. Evaluate errorS (h)

errorS (h) is an **unbiased estimator** for errorD(h); since true error is not computable, but we need it to evaluate h, we must use an estimate

Trade off between training and testing

- More training and less testing improve performance: but Es does not approximate well Ed
- More training and less testing reduces variance

USE: 2/3 TRAINING, 1/3 TESTING

Comparisons

True comparison: d = Ed(h1) - Ed(h2)

Its estimator: d^= Es1(h1) - Es2(h2)

d^ is unbiased iff all parameters are independent: E[d^] = d

Overfitting: h in H **overfits** training data if there is h' in H:

Es(h) < Es(h') Ed(h) > Ed(h')

Rem.: h is solution of learning algorithm L when using training set T → h = L(T)

K-fold Cross Validation

1. Partition data set D into k disjoint sets S1, S2, . . . , Sk (|Si | > 30)
2. For i = 1, . . . , k do

   use Si as test set, and the remaining data as training set Ti
   Ti = {D − Si}
   hi = L(Ti)

   deltai = Esi(hi)

3. Return

   error(L,D) = 1/k Sum deltai

Comparing Algorithm

E(S in D)[errorD(LA(S)) − errorD(LB(S))]

where L(S) is the hypothesis output by learner L using training set S
i.e., the expected difference in true error between hypotheses output by learners LA and LB; can be approximated by a K-Fold Cross Validation.

Using A and B →

$$\bar{\delta} \equiv \frac{1}{k} \sum_{i=1}^{k} \delta_i$$

Note: if $\bar{\delta} < 0$ we can estimate that L_A is better than L_B.

2.2 Performance metrics

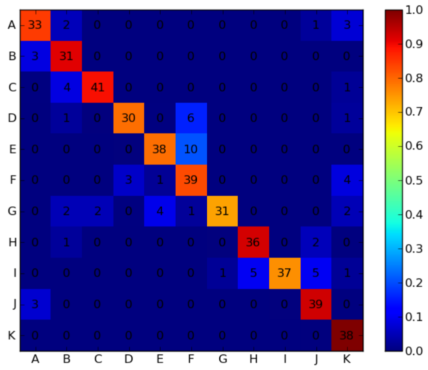
Accuracy is not always a good performance metric, especially with unbalanced data.

	Predicted class	
True Class	Yes	No
Yes	TP: True Positive	FN: False Negative
No	FP: False Positive	TN: True Negative

Error rate = | errors | / | instances | = (FN + FP) / (TP + TN + FP +FN)
Accuracy = 1 - Error rate = (TP + TN) / (TP + TN + FP + FN)
Problems when datasets are unbalanced.

Confusion Matrix

In a classification problem with many classes, we can compute how many times an instance of class Ci is classified in class Cj.



Main diagonal contains accuracy; outside the diagonal the errors.