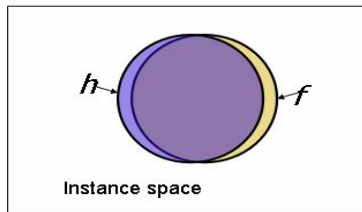




MACHINE LEARNING

Vasant Honavar
Artificial Intelligence Research Laboratory
Department of Computer Science
Bioinformatics and Computational Biology Program
Center for Computational Intelligence, Learning, & Discovery
Iowa State University
honavar@cs.iastate.edu
www.cs.iastate.edu/~honavar/
www.cild.iastate.edu/

Measuring classifier performance



■ $h(x) = f(x)$

To simplify matters, assume that class labels are binary
M-class problem is turned into M 2-class problems

Measuring Classifier Performance

N : Total number of instances in the data set

TP_j : Number of True positives for class j

FP_j : Number of False positives for class j

TN_j : Number of True Negatives for class j

FN_j : Number of False Negatives for class j

$$\begin{aligned} \text{Accuracy}_j &= \frac{TP_j + TN_j}{N} \\ &= P(\text{class} = c_j \wedge \text{label} = c_j) \end{aligned}$$

Perfect classifier \leftrightarrow Accuracy = 1

Popular measure

Biased in favor of the majority class!

Should be used with caution!

Measuring Classifier Performance: Sensitivity

$$\begin{aligned} \text{Sensitivity}_j &= \frac{TP_j}{TP_j + FN_j} \\ &= \frac{\text{Count}(\text{label} = c_j \wedge \text{class} = c_j)}{\text{Count}(\text{class} = c_j)} \\ &= P(\text{label} = c_j \mid \text{class} = c_j) \end{aligned}$$

Perfect classifier \rightarrow Sensitivity = 1

Probability of correctly labeling members of the target class

Also called recall or hit rate

Measuring Classifier Performance: Specificity

$$\begin{aligned}
 \text{Specificity}_j &= \frac{TP_j}{TP_j + FP_j} \\
 &= \frac{\text{Count}(\text{label} = c_j \wedge \text{class} = c_j)}{\text{Count}(\text{label} = c_j)} \\
 &= P(\text{class} = c_j \mid \text{label} = c_j)
 \end{aligned}$$

Perfect classifier \rightarrow Specificity = 1

Also called precision

Probability that a positive prediction is correct

Classifier Learning -- Measuring Performance

Class Label \rightarrow	C_1	$\neg C_1$
C_1 \downarrow	TP= 55	FP=5
$\neg C_1$	FN=10	TN=30

$$N = TP + FN + TN + FP = 100$$

$$\text{sensitivity}_1 = \frac{TP}{TP + FN} = \frac{55}{65}$$

$$\text{specificity}_1 = \frac{TN}{TN + FP} = \frac{30}{35}$$

$$\text{accuracy}_1 = \frac{TP + TN}{N} = \frac{85}{100}$$

Measuring Performance: Precision, Recall, and False Alarm Rate

$$Precision_j = Specificity_j = \frac{TP_j}{TP_j + FP_j} \quad Recall_j = Sensitivity_j = \frac{TP_j}{TP_j + FN_j}$$

Perfect classifier \rightarrow Precision=1 Perfect classifier \rightarrow Recall=1

$$FalseAlarm_j = \frac{FP_j}{TN_j + FP_j}$$

$$= \frac{Count(label = c_j \wedge class = \neg c_j)}{Count(label = \neg c_j)}$$

$$= P(label = c_j | class = \neg c_j)$$

Perfect classifier \rightarrow
False Alarm Rate = 0

Measuring Performance – Correlation Coefficient

$$CC_j = \frac{(TP_j \times TN_j) - (FP_j \times FN_j)}{\sqrt{(TP_j + FN_j)(TP_j + FP_j)(TN_j + FP_j)(TN_j + FN_j)}}$$

$$-1 \leq CC_j \leq 1$$

Perfect classifier \leftrightarrow $CC = 1$, Random guessing \leftrightarrow $CC=0$

Corresponds to the standard measure of correlation between two random variables *Label* and *Class* estimated from labels **L** and the corresponding class values **C** for the special case of binary (0/1) valued labels and classes

$$CC_j = \sum_{d_i \in D} \frac{(jlabel_i - \overline{jlabel})(jclass_i - \overline{jclass})}{\sigma_{JLABEL} \sigma_{JCLASS}}$$

where $jlabel_i = 1$ iff the classifier assigns d_i to class c_j

$jclass_i = 1$ iff the true class of d_i is class c_j

Beware of terminological confusion in the literature!

- Some bioinformatics authors use “accuracy” incorrectly to refer to recall i.e. sensitivity or precision i.e. specificity
- In medical statistics, specificity sometimes refers to **sensitivity for the negative class** i.e. $\frac{TN_j}{TN_j + FP_j}$
- Some authors use false alarm rate to refer to the probability that a positive prediction is incorrect i.e. $\frac{FP_j}{FP_j + TP_j} = 1 - Precision_j$

When you write

- **provide the formula in terms of TP , TN , FP , FN**

When you read

- **check the formula in terms of TP , TN , FP , FN**

Measuring Classifier Performance

- TP , FP , TN , FN provide the relevant information
- No single measure tells the whole story
- A classifier with 90% accuracy can be useless if 90 percent of the population does not have cancer and the 10% that do are misclassified by the classifier
- **Use of multiple measures recommended**
- **Beware of terminological confusion!**

Measuring Classifier Performance

Micro-averages

Micro averaging gives equal importance to each instance
 → Classes with large number of instances dominate

$$\text{MicroAverage Precision} = \frac{\sum_j TP_j}{\sum_j TP_j + \sum_j FP_j} \quad \text{MicroAverage Recall} = \frac{\sum_j TP_j}{\sum_j TP_j + \sum_j FN_j}$$

$$\text{MicroAverage FalseAlarm} = 1 - \text{MicroAverage Precision}$$

$$\text{MicroAverage Accuracy} = \frac{\sum_j TP_j}{N} \quad \text{Etc.}$$

$$\text{MicroAverage CC} = \frac{\left(\left(\sum_j TP_j \right) \times \left(\sum_j TN_j \right) \right) - \left(\left(\sum_j FP_j \right) \times \left(\sum_j FN_j \right) \right)}{\sqrt{\left(\sum_j TP_j + \sum_j FN_j \right) \left(\sum_j TP_j + \sum_j FP_j \right) \left(\sum_j TN_j + \sum_j FP_j \right) \left(\sum_j TN_j + \sum_j FN_j \right)}}$$

Measuring Classifier Performance

Macro-averaging

Macro averaging gives equal importance to each of the M classes

$$\text{MacroAverage Sensitivity} = \frac{1}{M} \sum_j \text{Sensitivity}_j$$

$$\text{MacroAverageCorrelationCoeff} = \frac{1}{M} \sum_j \text{CorrelationCoeff}_j$$

$$\text{MacroAverage Specificity} = \frac{1}{M} \sum_j \text{Specificity}_j$$

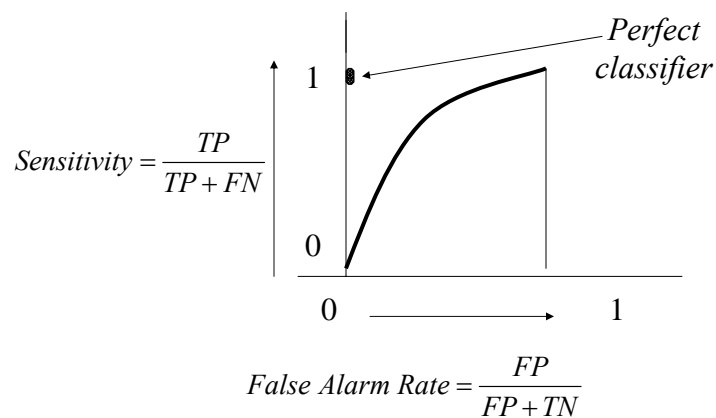
Receiver Operating Characteristic (ROC) Curve

- We can often trade off recall versus precision – e.g., by adjusting classification threshold θ e.g.,

$$\text{label} = c_j \text{ if } \frac{P(c_j | X)}{P(\neg c_j | X)} > \theta$$

- ROC curve is a plot of Sensitivity against False Alarm Rate which characterizes this tradeoff for a given classifier

Receiver Operating Characteristic (ROC) Curve



Measuring Performance of Classifiers – ROC curves

- ROC curves offer a more complete picture of the performance of the classifier as a function of the classification threshold
- A classifier h is better than another classifier g if $\text{ROC}(h)$ **dominates** the $\text{ROC}(g)$
- $\text{ROC}(h)$ **dominates** $\text{ROC}(g) \rightarrow \text{AreaROC}(h) > \text{AreaROC}(g)$

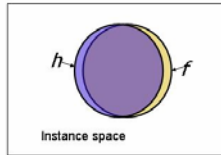
Evaluating a Classifier

- How well can a classifier be expected to perform on *novel* data?
- We can *estimate* the *performance* (e.g., accuracy, sensitivity) of the classifier using a test data set (not used for training)
- How close is the *estimated* performance to the *true* performance?

References:

- Evaluation of discrete valued hypotheses – Chapter 5, Mitchell
- Empirical Methods for Artificial Intelligence, Cohen
Better yet, take Stat 430x

Estimating the performance of a classifier



■ $h(x) = f(x)$ The *true error* of a hypothesis h with respect to a target function f and an instance distribution D is

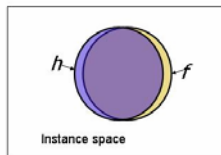
$$Error_D(h) \equiv \Pr_{x \in D}[f(x) \neq h(x)]$$

The *sample error* of a hypothesis h with respect to a target function f and an instance distribution D is

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

$\delta(a, b) = 1$ iff $a \neq b$; $\delta(a, b) = 0$ otherwise

Estimating classifier performance



■ $h(x) = f(x)$

$$Domain(X) = \{a, b, c, d\}$$

$$D(X) = \left\{ \frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{4} \right\}$$

x	a	b	c	d
$h(x)$	0	1	1	0
$f(x)$	1	1	0	0

$$\begin{aligned} error_D(h) &= \Pr_D[h(x) \neq f(x)] \\ &= D(X = a) + D(X = c) \\ &= \frac{1}{8} + \frac{1}{8} = \frac{1}{4} \end{aligned}$$

Evaluating the performance of a classifier

- Sample error estimated from training data is an *optimistic* estimate

$$\text{Bias} = E[\text{Error}_S(h)] - \text{Error}_D(h)$$

- For an *unbiased* estimate, h must be evaluated on an independent sample S (which is not the case if S is the training set!)
- Even when the estimate is unbiased, it can *vary* across samples!

- If h misclassifies 8 out of 100 samples $\text{Error}_S(h) = \frac{8}{100} = 0.08$

How close is the *sample error* to the *true error*?

How close is the *estimated error* to the *true error*?

- Choose a sample S of size n according to distribution D
- Measure $\text{Error}_S(h)$

$\text{Error}_S(h)$ is a random variable (outcome of a random experiment)

Given $\text{Error}_S(h)$, what can we conclude about $\text{Error}_D(h)$?

More generally, given the estimated performance of a hypothesis, what can we say about its actual performance?

Evaluation of a classifier with limited data

- There is extensive literature on how to estimate classifier performance from samples and how to assign confidence to estimates (See Mitchell, Chapter 5)
- Holdout method – use part of the data for training, and the rest for testing
- We may be unlucky – training data or test data may not be *representative*
- Solution – Run multiple experiments with disjoint training and test data sets in which each class is represented in roughly the same proportion as in the entire data set

Estimating the performance of the learned classifier

K-fold cross-validation

Partition the data (multi) set S into K equal parts $S_1 \dots S_K$ with roughly the same class distribution as S .

$Errorc = 0$

For $i=1$ to K do

$$\left\{ \begin{array}{l} S_{Test} \leftarrow S_i \quad S_{Train} \leftarrow S - S_i; \\ \alpha \leftarrow Learn(S_{Train}) \\ Errorc \leftarrow Errorc + Error(\alpha, S_{Test}) \end{array} \right\}$$

$$Error \leftarrow \left(\frac{Errorc}{K} \right); \quad Output(Error)$$

Leave-one-out cross-validation

- K -fold cross validation with $K = n$ where n is the total number of samples available
- n experiments – using $n-1$ samples for training and the remaining sample for testing
- Leave-one-out cross-validation does not guarantee the same class distribution in training and test data!
 - Extreme case: 50% class 1, 50% class 2
 - Predict majority class label in the training data
 - True error – 50%;
 - Leave-one-out error estimate – 100%!!!!

Estimating classifier performance

Recommended procedure

- Use K -fold cross-validation ($K=5$ or 10) for estimating performance estimates (accuracy, precision, recall, points on ROC curve, etc.) and 95% confidence intervals around the mean
- Compute mean values of performance estimates and standard deviations of performance estimates
- Report mean values of performance estimates and their standard deviations or 95% confidence intervals around the mean
- Be skeptical – repeat experiments several times with different random splits of data into K folds!

Evaluating a hypothesis or a learning algorithm

How well can the decision tree be expected to perform on *novel* data?

We can *estimate* the *performance* (e.g., accuracy) of the decision tree using a test data set (not used for training)

How close is the *estimated* performance to the *true* performance?

Reference: Evaluation of discrete valued hypotheses – Chapter 5, Mitchell

Evaluating performance when we can afford to test on a large independent test set

The *true* error of a hypothesis h with respect to a target function f and an instance distribution D is

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

The sample error of a hypothesis h with respect to a target function f and an instance distribution D is

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

$$\delta(a, b) = 1 \text{ iff } a \neq b; \delta(a, b) = 0 \text{ otherwise}$$

Evaluating Classifier performance

$$\text{Bias} = E[\text{Error}_S(h)] - \text{Error}_D(h)$$

Sample error estimated from training data is an *optimistic* estimate

For an *unbiased* estimate, h must be evaluated on an independent sample S (which is not the case if S is the training set!)

Even when the estimate is unbiased, it can *vary* across samples!

If h misclassifies 8 out of 100 samples $\text{Error}_S(h) = \frac{8}{100} = 0.08$

How close is the *sample error* to the *true error*?

How close is estimated error to its true value?

Choose a sample S of size n according to distribution D

Measure $\text{Error}_S(h)$

$\text{Error}_S(h)$ is a random variable (outcome of a random experiment)

Given $\text{Error}_S(h)$, what can we conclude about $\text{Error}_D(h)$?

More generally, given the estimated performance of a hypothesis, what can we say about its actual performance?

How close is estimated accuracy to its true value?

Question: How close is p (the true probability) to \hat{p} ?

This problem is an instance of a well-studied problem in statistics – the problem of estimating the proportion of a population that exhibits some property, given the observed proportion over a random sample of the population. In our case, the property of interest is that h correctly (or incorrectly) classifies a sample.

Testing h on a single random sample x drawn according to D amounts to performing a random experiment which succeeds if h correctly classifies x and fails otherwise.

How close is estimated accuracy to its true value?

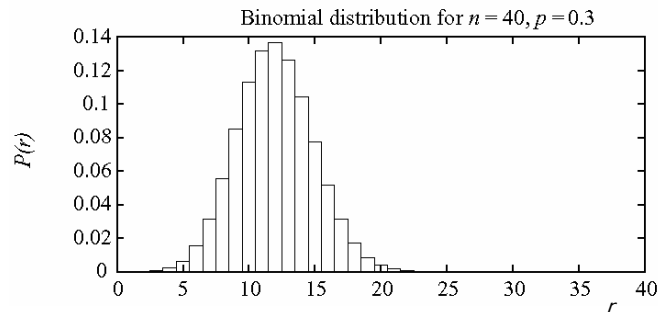
The output of a hypothesis whose true error is p as a binary *random variable* which corresponds to the outcome of a Bernoulli trial with a *success rate* p (the probability of correct prediction)

The *number of successes* r observed in N trials is a random variable Y which follows the Binomial distribution

$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

$Error_S(h)$ is a Random Variable

Probability of observing r misclassified examples in a sample of size n :



$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

Recall basic statistics

Consider a random experiment with discrete valued outcomes

$$y_1, y_2, \dots, y_M$$

The expected value of the corresponding random variable Y is

$$E(Y) \equiv \sum_{i=1}^M y_i \Pr(Y = y_i)$$

The variance of Y is

$$\text{Var}(Y) \equiv E[(Y - E[Y])^2]$$

The standard deviation of Y is

$$\sigma_Y \equiv \sqrt{\text{Var}(Y)}$$

How close is estimated accuracy to its true value?

The *mean* of a Bernoulli trial with success rate $p = p$

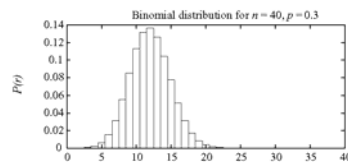
Variance = $p(1-p)$

If N trials are taken from the same Bernoulli process, the observed success rate \hat{p} has the same mean p

and variance $\frac{p(1-p)}{N}$

For large N , the distribution of \hat{p} follows a Gaussian distribution

Binomial Probability Distribution



$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

Probability $P(r)$ of r heads in n coin flips, if $p = Pr(\text{heads})$

- Expected, or mean value of X , $E[X]$, is

$$E[X] \equiv \sum_{i=0}^N iP(i) = np$$

- Variance of X is

$$\text{Var}(X) \equiv E[(X - E[X])^2] = np(1-p)$$

- Standard deviation of X , σ_X , is

$$\sigma_X \equiv \sqrt{E[(X - E[X])^2]} = \sqrt{np(1-p)}$$

Estimators, Bias, Variance, Confidence Interval

$$\sigma_{Error_S(h)} = \sqrt{\frac{p(1-p)}{n}}$$

$$Error_S(h) = \frac{r}{n}$$

$$Error_D(h) = p$$

$$\sigma_{Error_S(h)} = \sqrt{\frac{Error_D(h)(1-Error_D(h))}{n}}$$

$$\sigma_{Error_S(h)} \approx \sqrt{\frac{Error_S(h)(1-Error_S(h))}{n}}$$

An $N\%$ confidence interval for some parameter p that is the interval which is expected with probability $N\%$ to contain p

Normal distribution approximates binomial

$Error_S(h)$ follows a **Binomial** distribution, with

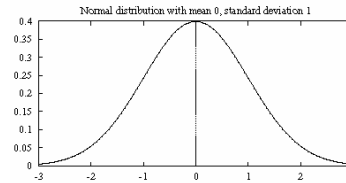
- mean $\mu_{Error_S(h)} = Error_D(h)$
- standard deviation

$$\sigma_{Error_S(h)} = \sqrt{\frac{Error_D(h)(1-Error_D(h))}{n}}$$

We can approximate this by a **Normal** distribution with the same mean and variance when $np(1-p) \geq 5$

Normal distribution

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



The probability that X will fall in the interval (a, b) is given by $\int_a^b p(x)dx$

Expected, or mean value of X is given by $E[X] = \mu$

Variance of X is given by $\text{Var}(X) = \sigma^2$

Standard deviation of X is given by $\sigma_X = \sigma$

How close is the estimated accuracy to its true value?

Let the probability that a Gaussian random variable X , with zero mean, takes a value between $-z$ and z ,

$$\Pr[-z \leq X \leq z] = c$$

$\Pr[X \geq z] = 5\%$ implies there is a 5% chance that X lies more than 1.65 standard deviations from the mean, or

$$\Pr[-1.65 \leq X \leq 1.65] = 90\%$$

$\Pr[X \geq z]$	z
0.001	3.09
0.005	2.58
0.01	2.33
0.05	1.65
0.10	1.28

How close is the estimated accuracy to its true value?

But \hat{p} does not have zero mean and unit variance so we normalize to get

$$\Pr \left[-z < \frac{\hat{p} - p}{\sqrt{\frac{p(1-p)}{n}}} < z \right] = c$$

How close is the estimated accuracy to its true value?

To find confidence limits:

Given a particular confidence figure c , use the table to find the z corresponding to the probability $\frac{1}{2}(1-c)$.

Use linear interpolation for values not in the table

$$p = \frac{\left[\hat{p} + \frac{z^2}{2n} \pm z \sqrt{\frac{\hat{p}}{n} - \frac{\hat{p}^2}{n} + \frac{z^2}{4n^2}} \right]}{\left[1 + \frac{z^2}{n} \right]}$$

How close is the estimated accuracy to its true value?

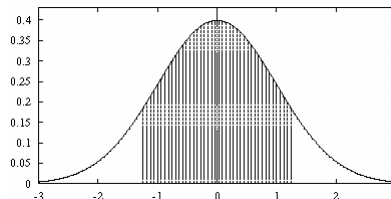
Example

$$\hat{p} = 0.75; \quad n = 1000; \quad c = 0.80; \quad z = 1.28$$

Then with 80% confidence, we can say that the value of p lies in the interval $[0.733, 0.768]$

Note: the normal distribution assumption is valid only for large n (i.e. $np(1-p) \geq 5$ or $n > 30$) so estimates based on smaller values of n should be taken with a generous dose of salt

Estimating confidence intervals



80% of area (probability) lies in $\mu \pm 1.28\sigma$

N% of area (probability) lies in $\mu \pm z_N\sigma$

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

Confidence intervals

If

S contains n examples, drawn independently of h and each other
and $n \geq 30$ or $np(1-p) \geq 5$,

Then

With approximately $N\%$ probability, $Error_S(h)$ lies in interval

$$Error_D(h) \pm Z_N \sqrt{\frac{Error_D(h)(1-Error_D(h))}{n}}$$

equivalently, $Error_D(h)$ lies in interval

$$Error_S(h) \pm Z_N \sqrt{\frac{Error_D(h)(1-Error_D(h))}{n}}$$

which is approximately

$$Error_S(h) \pm Z_N \sqrt{\frac{Error_S(h)(1-Error_S(h))}{n}}$$

One sided confidence intervals

What is the probability that $Error_D(h)$ is at most U ?

Symmetry of Gaussian distribution implies that
confidence interval with $100(1-\alpha)\%$ confidence
with lower bound L and upper bound U corresponds
to a confidence interval with confidence $100(1-\frac{\alpha}{2})\%$
and with upper bound U but no lower bound (or vice
versa)

General approach to deriving confidence intervals

1. Identify the population parameter p to be estimated
e.g., $Error_D(h)$
2. Define a suitable estimator W – preferably unbiased,
minimum variance
3. Determine the distribution D_W obeyed by W , and the
mean and variance of W
4. Determine the confidence interval by finding the
thresholds L and U such that $N\%$ of the mass of the
probability distribution D_Y falls within the interval $[L,U]$.

Central Limit Theorem Simplifies Confidence Interval Calculations

Consider a set of independent, identically distributed random variables Y_1, \dots, Y_n , all governed by an arbitrary probability distribution with mean μ and finite variance σ^2 . Define the sample mean,

$$\bar{Y} \equiv \frac{1}{n} \sum_{i=1}^n Y_i$$

Central Limit Theorem As $n \rightarrow \infty$, the distribution governing \bar{Y} approaches a Normal distribution, with mean μ and variance σ^2/n

Evaluation of a classifier with limited data

Holdout method – use part of the data for training, and the rest for testing

We may be unlucky – training data or test data may not be *representative*

Solution – Run multiple experiments with disjoint training and test data sets in which each class is represented in roughly the same proportion as in the entire data set

Estimating the performance of the learned classifier

K-fold cross-validation

Partition the data (multi) set S into K equal parts $S_1 \dots S_K$ where each part has roughly the same class distribution as S .

$A = 0$

For $i=1$ to K do

$$\left. \begin{array}{l} S_{Train} \leftarrow S - S_i; \quad S_{Test} \leftarrow S_i \\ \alpha \leftarrow Learn(S_{Train}) \\ A \leftarrow A + Accuracy(\alpha, S_{Test}) \end{array} \right\}$$

$Accuracy \leftarrow A/K$; Output ($Accuracy$)

K-fold cross-validation

Recommended procedure for evaluating classifiers when data are limited

Use K -fold cross-validation ($K=5$ or 10)

Better still, repeat K -fold cross-validation R times and average the results

Difference in error between two hypotheses

We wish to estimate $d \equiv Error_D(h_1) - Error_D(h_2)$

Suppose h_1 has been tested on a sample S_1 of size n_1 drawn according to D and h_2 has been tested on a sample S_2 of size n_2 drawn according to D

An unbiased estimator $\hat{d} \equiv Error_{S_1}(h_1) - Error_{S_2}(h_2)$

For large n_1 and large n_2 the corresponding error estimates follow Normal distribution

Difference of two Normal distributions yields a normal distribution with variance equal to the sum of the variances of the individual distributions

Difference between errors of two hypotheses

$$d \equiv \text{Error}_D(h_1) - \text{Error}_D(h_2)$$

$$\hat{d} \equiv \text{Errors}_{S_1}(h_1) - \text{Errors}_{S_2}(h_2)$$

$$\sigma_{\hat{d}} \approx \sqrt{\frac{\text{Errors}_{S_1}(h_1)(1 - \text{Errors}_{S_1}(h_1))}{n_1} + \frac{\text{Errors}_{S_2}(h_2)(1 - \text{Errors}_{S_2}(h_2))}{n_2}}$$

$$\hat{d} \pm z_N \sqrt{\frac{\text{Errors}_{S_1}(h_1)(1 - \text{Errors}_{S_1}(h_1))}{n_1} + \frac{\text{Errors}_{S_2}(h_2)(1 - \text{Errors}_{S_2}(h_2))}{n_2}}$$

When $S_1=S_2$, the variance of \hat{d} is smaller and the confidence interval correct but overly conservative

Hypothesis testing

Is one hypothesis likely to be better than another?

What is the probability that $\text{Error}_D(h_1) > \text{Error}_D(h_2)$?

Suppose $\text{Errors}_{S_1}(h_1) = 0.30$; $\text{Errors}_{S_2}(h_2) = 0.20$; $\hat{d} = 0.10$

What is the probability that $d > 0$ given that $\hat{d} = 0.10$?

$$\Pr(d > 0 \mid \hat{d} = 0.10) = \Pr(\hat{d} < \mu_{\hat{d}} + 0.10)$$

Hypothesis testing

If $n_1 = n_2 = 100, \sigma_{\hat{d}} \approx 0.061$

$$\Pr(d > 0 \mid \hat{d} = 0.10) \approx \Pr(\hat{d} < \mu_{\hat{d}} + 1.64\sigma_{\hat{d}}) = 0.95$$

We accept the hypothesis that

$$Error_D(h_1) > Error_D(h_2)$$

with 95% confidence

Equivalently, we reject the opposite hypothesis –
the null hypothesis at a $(1-0.95) = 0.05$ level of significance

Comparing learning algorithms L_A and L_B

Which learning algorithm is better at learning f ?

Unlimited data –

Run L_A and L_B on *large* training set S_{train} drawn according to D

Test the resulting hypotheses on a *large independent* test set
 S_{Test} drawn according to D

Estimate $\Pr[Error_D(L_A(S_{Train})) > Error_D(L_B(S_{Train}))]$ Using

$$Error_{S_{Test}}(L_A(S_{Train})) \text{ and } Error_{S_{Test}}(L_B(S_{Train}))$$

Comparing learning algorithms L_A and L_B

Estimate the expected value of the difference in errors of L_A and L_B where expectation is taken over training sets S_{Train} drawn according to D

$$E_{S_{Train} \subset D} [Error_D(L_A(S_{Train})) - Error_D(L_B(S_{Train}))]$$

We have a limited data set S drawn from an unknown D !!

Comparing learning algorithms L_A and L_B

Limited data – Paired t-test

Run L_A and L_B on *large* training set S_{Train} drawn according to D

Test the resulting hypotheses on a *large independent* test set S_{Test} drawn according to D

Estimate

$$\Pr[Error_D(L_A(S_{Train})) > Error_D(L_B(S_{Train}))] \quad \text{using}$$

$$Error_{S_{Test}}(L_A(S_{Train})) \text{ and } Error_{S_{Test}}(L_B(S_{Train}))$$

Comparing learning algorithms L_A and L_B

Paired t-test

Partition S into k disjoint test sets T_1, T_2, \dots, T_k of equal size

For i from 1 to k do {

$$S_{Test} \leftarrow T_i ; S_{Train} \leftarrow S - T_i$$

$$\delta_i \leftarrow Error_{S_{Test}}(L_A(S_{Train})) - Error_{S_{Test}}(L_B(S_{Train}))$$

}

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

Comparing learning algorithms L_A and L_B

For large test sets, each δ_i has Normal distribution

$\bar{\delta}$ has Normal distribution if δ_i are independent

Can we estimate confidence interval for $\bar{\delta}$ as before?

δ_i are not exactly independent because of sampling from S as opposed to the distribution D (but we will pretend that they are)

We don't know the standard deviation of this distribution.

So we estimate it from sample ..But when the estimated variance is used, the distribution is no longer Normal unless K is large (which typically it is not)

Comparing learning algorithms L_A and L_B

Approximate $N\%$ confidence interval for

$$E_{S_{Train} \subset S} [Error_D(L_A(S_{Train})) - Error_D(L_B(S_{Train}))]$$

is given by $\bar{\delta} \pm t_{N, k-1} \psi_{\bar{\delta}}$

$$\text{where } \psi_{\bar{\delta}} \equiv \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^k (\delta_i - \bar{\delta})^2}$$

is the estimate of standard deviation of the t distribution governing Z_N and $t_{N, K-1}$ plays a role analogous to that of

$$\bar{\delta} \quad \text{As } K \rightarrow \infty, t_{N, K-1} \rightarrow Z_N \text{ and } \psi_{\bar{\delta}} \rightarrow \sigma_{\bar{\delta}}$$

Performance evaluation summary

Rigorous statistical evaluation is extremely important in experimental computer science in general and machine learning in particular

How good is a learned hypothesis?

Is one hypothesis better than another?

Is one learning algorithm better than another on a particular learning task? (No learning algorithm outperforms all others on all tasks – No free lunch theorem)

Different procedures for evaluation are appropriate under different conditions (large versus limited versus small sample) – Important to know when to use which evaluation method and be aware of pathological behavior (tendency to grossly overestimate or underestimate the target value under specific conditions)