Evaluating Hypotheses

What to measure?

How to measure it?

Two key difficulties when only limited data is available:

- Bias in the estimate
- Variance in the estimate

Estimating Hypothesis Accuracy

- Given a hypothesis *h* and a data sample containing *n* examples drawn at random from *D*, what is the best estimate of the accuracy of *h* over future instances?
- What is the probable error in this accuracy estimate?

Sample Error and True Error

- Sample error of hypothesis h with respect to target function f and data sample S is

  $error_S(h) = (1/n) \sum_{x \in S} \{\delta_K[f(x), h(x)]\}$

- True error of hypothesis h with respect to target function f and distribution D is

  $error_D(h) = \Pr_{x \in D}[f(x) \neq h(x)]$

How good an estimate of $error_D(h)$ is $error_S(h)$?

Confidence Intervals

For discrete-valued hypotheses, $n \geq 30$ and $error_S(h)=r/n$, statistical theory asserts:

- The <u>most probable value</u> of $error_D(h)$ is $error_S(h)$
- With approximately N% probability, $error_D(h)$ lies in the interval:

$$error_S(h) \pm z_N[error_S(h)(1- error_S(h))/n]^{1/2}$$

Example: $N = 95$ $z_N \cong 1.96$

Valid when (Rule of Thumb): $n\ error_S(h)\ [1-error_S(h)] \geq 5$

In general

We can estimate error measures over finite samples

- With no bias

- With a variance that decreases with sample size

Regression: RMSE

- The Root-Mean Squared Error (RMSE) is a cost function for regression. The formula for the RMSE is:

$$RMSE(f) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(f(\mathbf{x}_i) - y_i)^2}$$

where m is the number of test examples, $f(x_i)$, the predicted output on x_i and y_i the actual values.

RMSE has no scale.

Normalized Root-Mean Squared Error (NRMSE), defined as RMSE over the variance of the data, is a better measure.

Confusion Matrix-Based Performance Measures

True class-> **Hypothesized class**	**Pos**	**Neg**
Yes	TP	FP
No	FN	TN
	P=TP+FN	N=FP+TN

Confusion Matrix

- Multi-Class Focus:
  - **Accuracy** = (TP+TN)/(P+N)
- Single-Class Focus:
  - **Precision** = TP/(TP+FP)
  - **Recall** = TP/P
  - **Fallout** = FP/N
  - **Sensitivity** = TP/(TP+FN)
  - **Specificity** = TN/(FP+TN)

Some issues with performance measures

True class->	Pos	Neg
Yes	**200**	100
No	300	**400**
	P=500	N=500

True class ->	Pos	Neg
Yes	**400**	300
No	100	**200**
	P=500	N=500

- Both classifiers obtain 60% accuracy
- They exhibit very different behaviours:
  - On the left: weak positive recognition rate/strong negative recognition rate
  - On the right: strong positive recognition rate/weak negative recognition rate

Some issues with performance measures (cont'd)

True class →	Pos	Neg
Yes	**500**	5
No	0	**0**
	P=500	N=5

True class →	Pos	Neg
Yes	**450**	1
No	50	**4**
	P=500	N=5

- The classifier on the left obtains 99.01% accuracy while the classifier on the right obtains 89.9%
  - Yet, the classifier on the right is much more sophisticated than the classifier on the left, which just labels everything as positive and misses all the negative examples.

Some issues with performance measures (cont'd)

True class →	**Pos**	**Neg**
Yes	200	100
No	300	**400**
	P=500	N=500

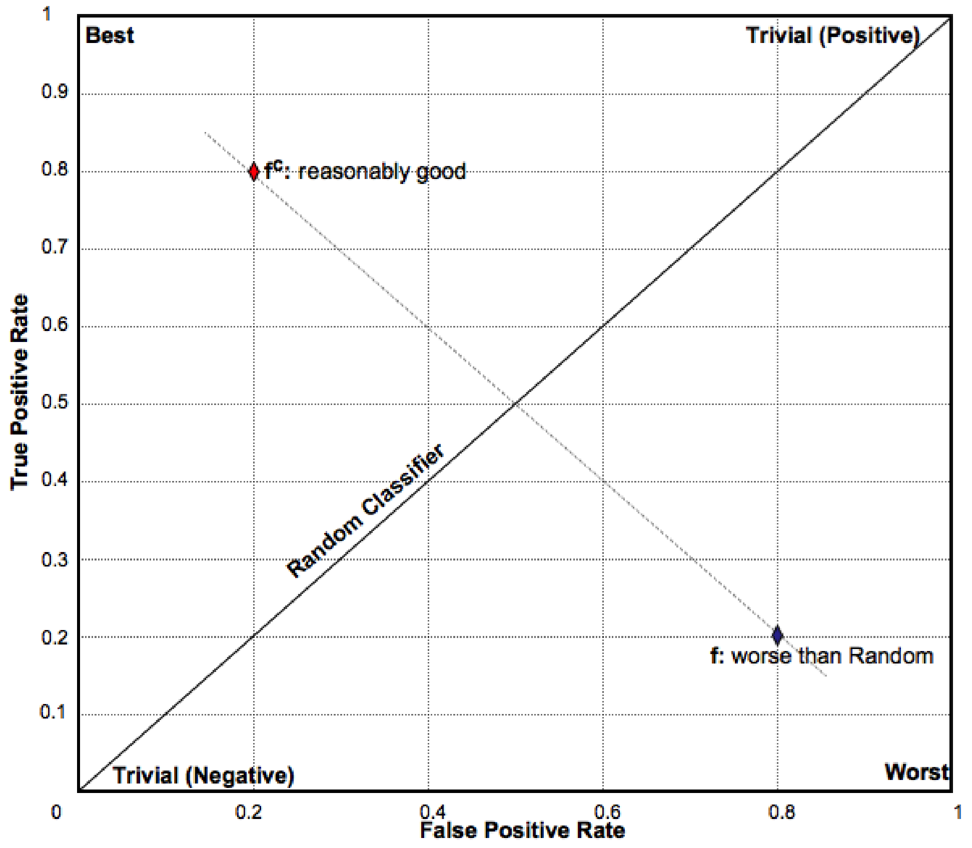
True class →	**Pos**	**Neg**
Yes	200	100
No	300	**0**
	P=500	N=100

- Both classifiers obtain the same precision and recall values of 66.7% and 40% (Note: the data sets are different)
- They exhibit very different behaviours:
  - Same positive recognition rate
  - Extremely different negative recognition rate: strong on the left / nil on the right
- Note: Accuracy has no problem catching this!

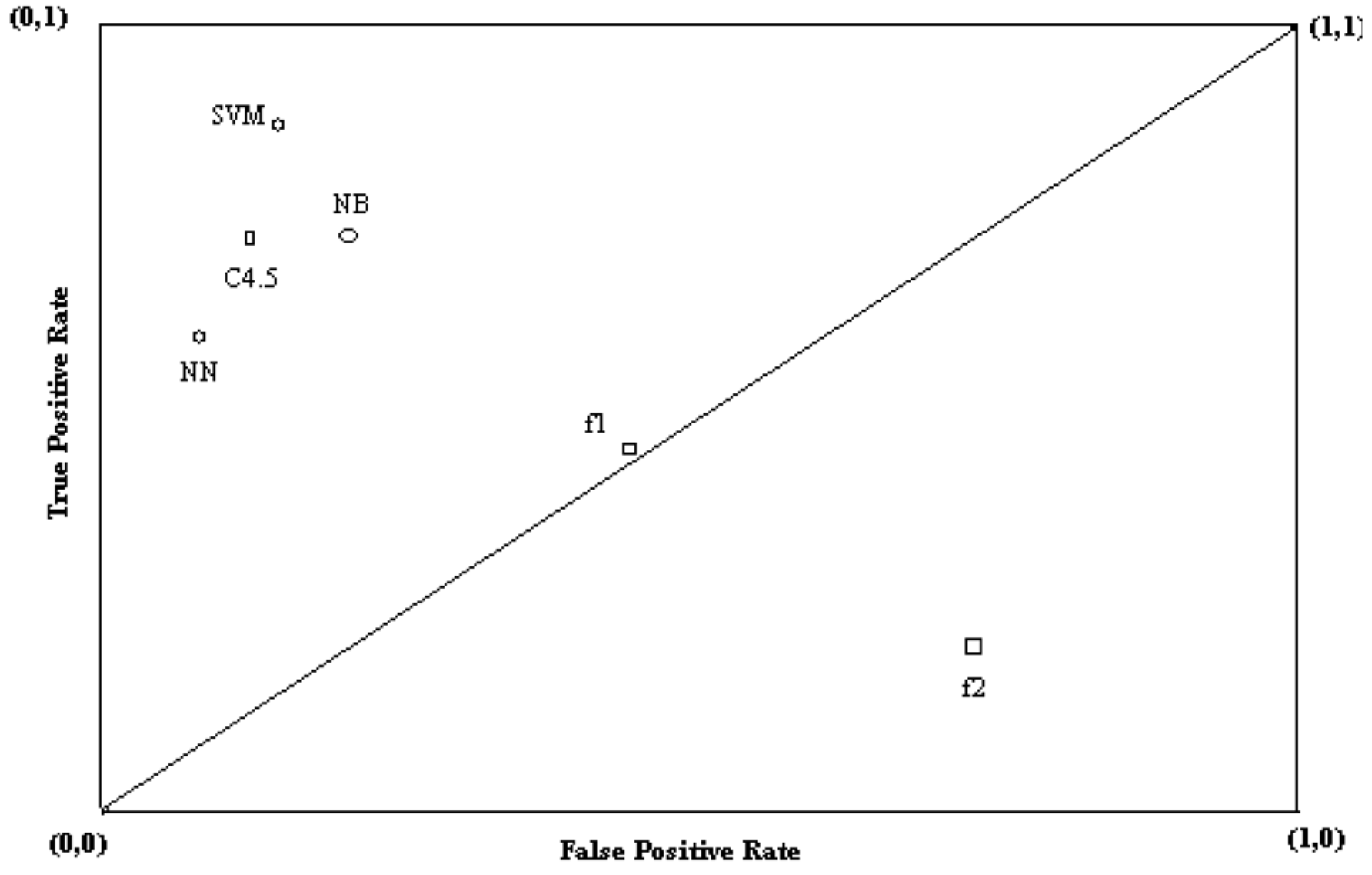
ROC Curves

- Performance measures for scoring classifiers
  - most classifiers are in fact scoring classifiers
  - Scores needn't be in pre-defined intervals, or even likelihood or probabilities
- ROC analysis has origins in Signal detection theory to set an operating point for desired signal detection rate
  - Signal assumed to be corrupted by noise (Normally distributed)
- ROC maps FPR on horizontal axis and TPR on the vertical axis; Recall that
  - FPR = FP/(FP+TN) = 1 – Specificity
  - TPR = TP/(TP+FN) = Sensitivity

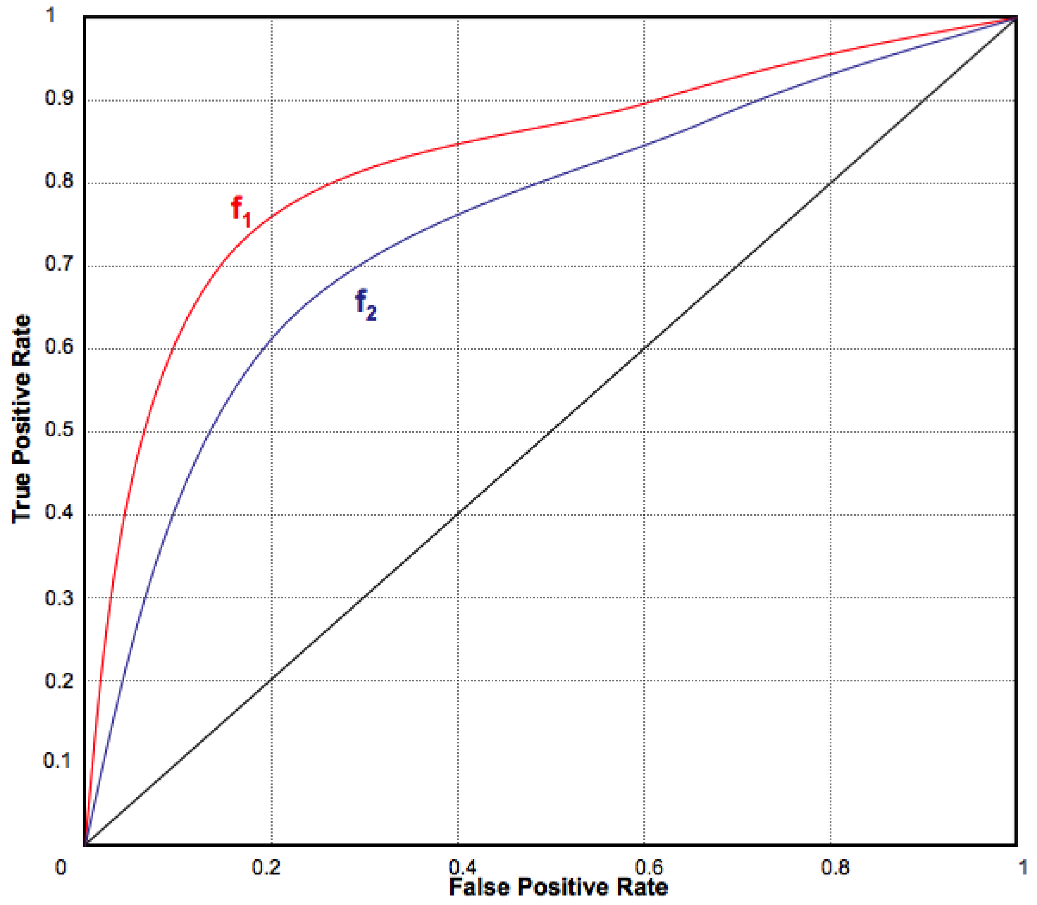
ROC Space


Best
Trivial (Positive)
f^c: reasonably good
Random Classifier
f: worse than Random
Trivial (Negative)
Worst
True Positive Rate
False Positive Rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

ROC Plot for discrete classifiers



ROC plot for two hypothetical scoring classifiers



AUC

- ROC Analysis allows a user to visualize the performance of classifiers over their operating ranges.
- However, it does not allow the quantification of this analysis, which would make comparisons between classifiers easier.
- The Area Under the ROC Curve (AUC) allows such a quantification: it represents the performance of the classifier averaged over all the possible cost ratios.

Se we decided on a performance measure.

- How do we estimate it in an unbiased manner?

- What if we used all the data?
  - Re-substitution: Overly optimistic (best performance achieved with complete over-fit)

Hold-out approach

- Set aside a separate test set T. Evaluate your measure on this set

- Pros
  - independence from training set
  - Generalization behavior can be characterized
  - Estimates can be obtained for *any* classifier

- Cons
  - We loose data for learning

The need for re-sampling

- Too few training examples -> learning a poor classifier
    - Having too few examples in the training set affects the bias of algorithm by making its average prediction unreliable

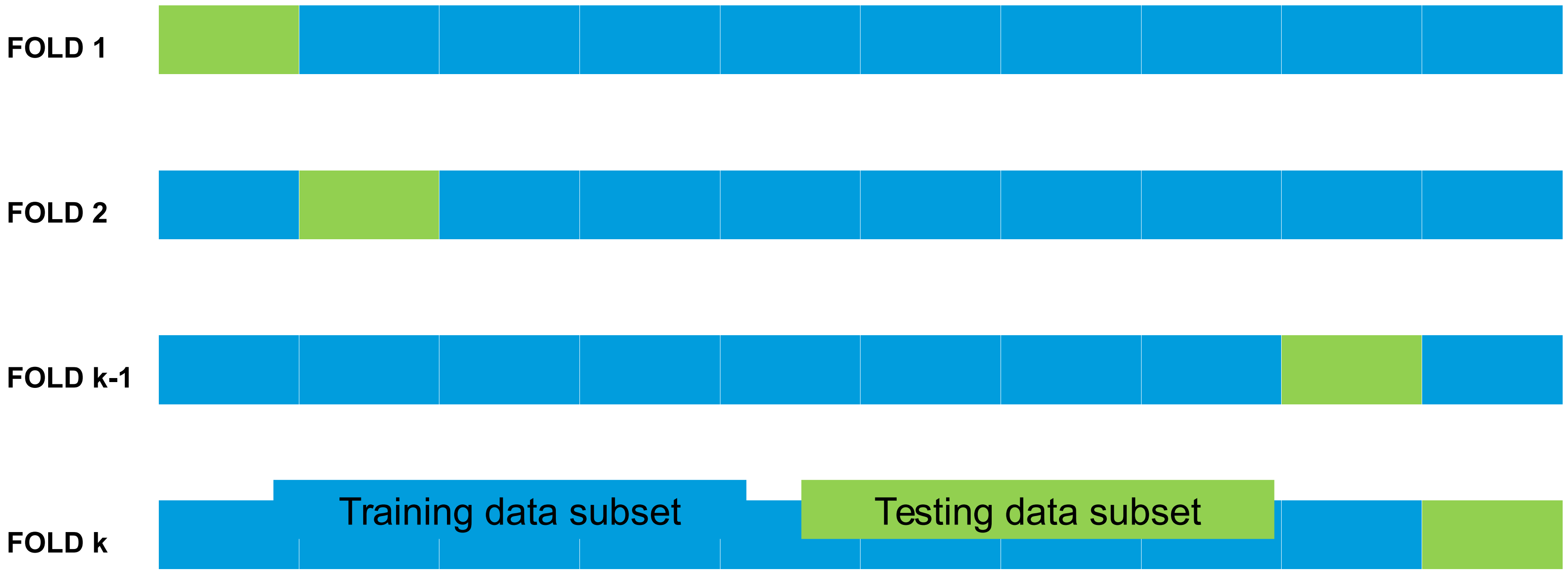
- Too few test examples -> bogus error estimates
    - Having too few examples in test set results in high variance in the estimation

- Hence: **Resampling**
    - Delivers accurate performance estimates while allowing the algorithm to train on most data examples

Simple Resampling: Repeated hold-out

- Set aside a separate random test set T. Evaluate your measure on this set
- Repeat n times
- Average your estimations
- Test and training sets have partial overlap.
  No independence.

Simple Resampling: K-fold Cross Validation



In Cross-Validation, the data set is divided into k folds and at each iteration, a different fold is reserved for testing while all the others are used for training the classifiers.

Simple Resampling: Some variations of Cross-Validation

- Stratified k-fold Cross-Validation:
  - Maintain the class distribution from the original dataset when forming the k-fold CV subsets
  - useful when the class-distribution of the data is imbalanced/skewed.

- Leave-One-Out
  - This is the extreme case of k-fold CV where each subset is of size 1.
  - Also known a Jackknife estimate
  - Useful for very small datasets

Observations

- k-fold CV is arguable the best known and most commonly used resampling technique
  - With k of reasonable size, less computer intensive than Leave-One-Out
  - Easy to apply
- In all the variants, the testing sets are independent of one another, as required, by many statistical testing methods
  - but the training sets are highly overlapping. This can affect the bias of the error estimates (generally mitigated for large dataset sizes).
- Results in an averaged estimate over k different classifiers

Multiple Resampling

- Multiple trials of simple resampling
  - Potentially Higher replicability and more stable estimates
  - Multiple runs (how many?): 5x2 cv, 10x10 cv
  - No more independence

Statistical Significance Testing

- Statistical Significance Testing can enable ascertaining whether the observed results are statistically significant (within certain constraints)
- We primarily focus on Null Hypothesis Significance Testing (NHST)

Statistical Significance Testing

- Why?

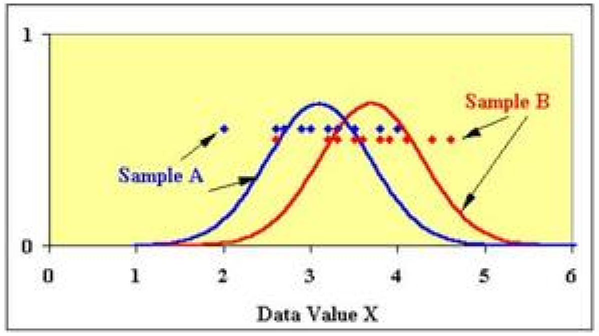
Escenario: some data, we train two classifiers A and B. A is better than B in a holdout set.
Q: Is A really better than B?

Statistical Significance Testing

- Why?

Escenario: some data, we train two classifiers A and B. A is better than B in a holdout set.
Q: Is A really better than B?

A: We only have one measure from a population.



We need several measurement to evaluate the difference between the distribution of observed errors.

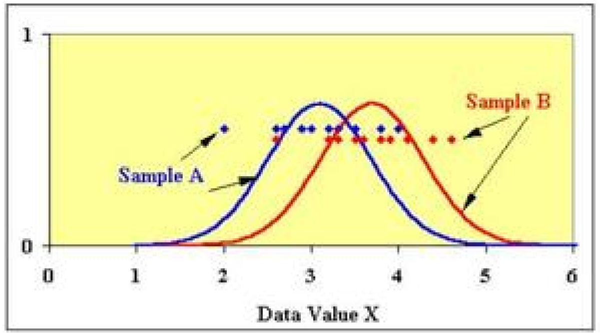
Statistical Significance Testing

- How?

-We want to probe that both samples of mean errors come from different distributions.

-We analyze the difference between measured error (and its variance).

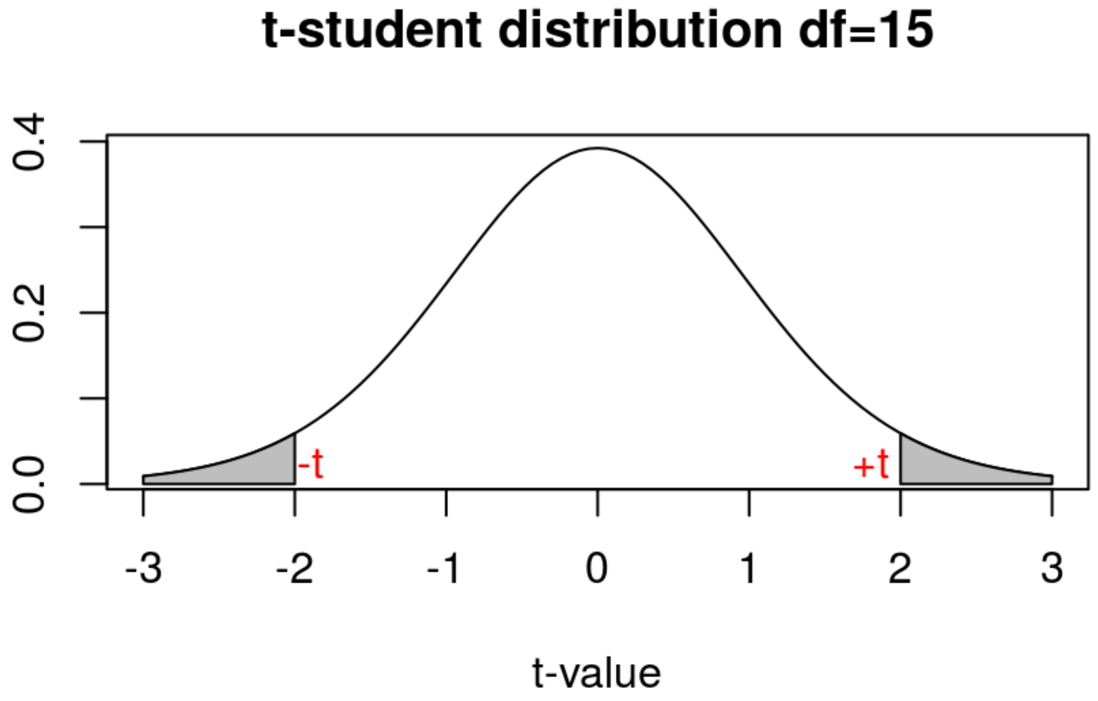
The difference follows a Student's t distribution



Statistical Significance Testing

- How?

- If methods are equal, the probability of having a big mean difference, relative to its variance, is really small.



NHST

- State a null hypothesis
  - Usually the opposite of what we wish to test (for example, classifiers A and B perform equivalently)
- Choose a suitable statistical test and statistic that will be used to (possibly) reject the null hypothesis
- Choose a critical region for the statistic to lie in, which is extreme enough for the null hypothesis to be rejected.
- Calculate the test statistic from the data
- If the test statistic lies in the critical region: reject the null hypothesis.
  - If not, we **fail to reject** the null hypothesis, but do not accept it either.

Rejecting the null hypothesis gives us some confidence in the belief that our observations did not occur merely by chance.

Comparing 2 algorithms on a single domain: *t*-test

- Arguably, one of the most widely used statistical tests
- Comes in various flavors (addressing different experimental settings)
- In our case: **two matched samples *t*-test**
  - to compare performances of two classifiers applied to the same dataset with matching randomizations and partitions
- Measures if the difference between two means (mean value of performance measures, e.g., error rate) is meaningful
- Null hypothesis: the two samples (performance measures of two classifiers over the dataset) come from the same population

Comparing 2 algorithms on a single domain: *t*-test

- The *t*-statistic:

$$t = \frac{\bar{d} - 0}{\frac{\bar{s_d}}{\sqrt{n}}}$$

number of trials

- where

$$\bar{d} = \overline{pm}(f_1) - \overline{pm}(f_2)$$

Average Performance measure, e.g., error rate of classifier f_1

Comparing 2 algorithms on a single domain: *t*-test

- The *t*-statistic:

$$t = \frac{\bar{d} - 0}{\frac{\bar{s}_d}{\sqrt{n}}}$$

- where

$$\bar{s}_d = \sqrt{\frac{\sum_{i=1}^{n} (d_i - \bar{d})^2}{n-1}}$$

Standard Deviation over the difference in performance measures

Difference in Performance measures at trial i

Comparing 2 algorithms on a single domain: *t*-test: Illustration

- C4.5 and Naïve Bayes (NB) algorithms on the Labor dataset
- 10-fold CV (maintaining same training and testing folds across algorithms).
- The result of each fold is a single result. Consequently n=10

- We have:

- and hence:

$$\bar{d} = 0.2175 - 0.0649 = 0.1526$$

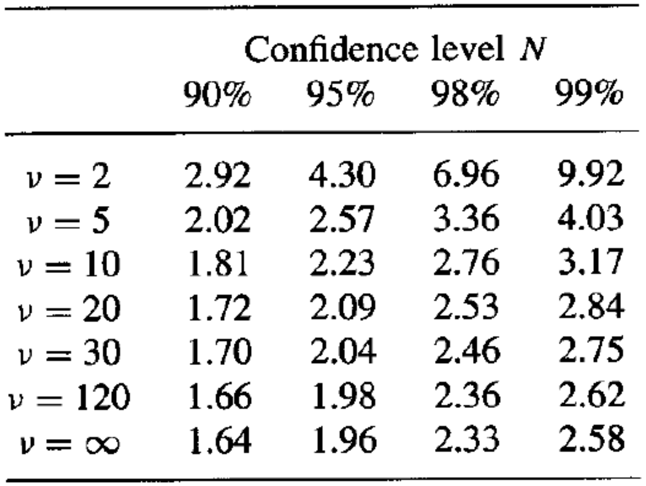
$$\bar{s}_d = \sqrt{\frac{\sum_{i=1}^{n}(d_i - \bar{d})^2}{n-1}} = \sqrt{\frac{0.0321}{10-1}} = 0.05969$$

$$t = \frac{\bar{d} - 0}{\frac{\bar{s}_d}{\sqrt{n}}} = \frac{0.1526 - 0}{\frac{0.05969}{\sqrt{10}}} = 8.0845$$

Comparing 2 algorithms on a single domain: *t*-test: Illustration

$$t = \frac{\bar{d} - 0}{\frac{\bar{s_d}}{\sqrt{n}}} = \frac{0.1526 - 0}{\frac{0.05969}{\sqrt{10}}} = 8.0845$$

- Referring to the t-test table, for n-1=9 degrees of freedom, we can reject the null hypothesis at 0.001 significance level (for which the observed t-value should be greater than 4.781)



	Confidence level N			
	90%	95%	98%	99%
$\nu = 2$	2.92	4.30	6.96	9.92
$\nu = 5$	2.02	2.57	3.36	4.03
$\nu = 10$	1.81	2.23	2.76	3.17
$\nu = 20$	1.72	2.09	2.53	2.84
$\nu = 30$	1.70	2.04	2.46	2.75
$\nu = 120$	1.66	1.98	2.36	2.62
$\nu = \infty$	1.64	1.96	2.33	2.58