Performance Evaluation for Learning Algorithms:
Techniques, Applications and Issues

Nathalie Japkowicz

University of Ottawa, Canada
Northern Illinois University, USA

nat@site.uottawa.ca

Mohak Shah

GE Global Research, USA

mohak@mohakshah.com

“Evaluation is the key to making real progress in data mining”,

[Witten & Frank, 2005], p. 143

Yet….

- Evaluation gives us a lot of, sometimes contradictory, information. How do we make sense of it all?
- Evaluation standards differ from person to person and discipline to discipline. How do we decide which standards are right for us?
- Evaluation gives us support for a theory over another, but rarely, if ever, certainty. So where does that leave us?

Example I: Which Classifier is better?

There are almost as many answers as there are performance measures! (e.g., UCI Breast Cancer)

Algo	Acc	RMSE	TPR	FPR	Prec	Rec	F	AUC	Info S
NB	71.7	.4534	.44	.16	.53	.44	.48	.7	48.11
C4.5	🟥 75.5	.4324	.27	.04	🟨 .74	🟨 .27	.4	🟥 .59	34.28
3NN	72.4	.5101	.32	.1	.56	.32	.41	.63	43.37
Ripp	71	.4494	.37	.14	.52	.37	.43	.6	22.34
SVM	69.6	🟥 .5515	.33	.15	.48	.33	.39	.59	🟥 54.89
Bagg	67.8	.4518	.17	.1	.4	.17	.23	.63	11.30
Boost	70.3	.4329	🟨 .42	🟨 .18	.5	.42	.46	.7	34.48
RanF	69.23	.47	.33	.15	.48	.33	🟥 .39	🟥 .63	20.78

🟨 : acceptable contradictions

🟥 : questionable contradictions

Example I: Which Classifier is better?
Ranking the results

Algo	Acc	RMSE	TPR	FPR	Prec	Rec	F	AUC	Info S
NB	3	5	🟨 1	🟨 7	3	1	1	1	2
C4.5	🟥 1	1	7	1	🟨 1	🟨 7	5	🟥 7	5
3NN	🟥 2	🟥 7	🟨 6	🟨 2	2	6	4	3	3
Ripp	4	3	3	4	4	3	3	6	6
SVM	6	🟥 8	4	5	5	4	6	7	🟥 1
Bagg	🟥 8	4	8	2	8	8	8	🟥 3	8
Boost	🟥 5	2	🟨 2	🟨 8	7	2	2	🟥 1	4
RanF	7	6	4	5	5	4	🟥 7	🟥 3	7

🟨 : acceptable contradictions

🟥 : questionable contradictions

Example II: What's wrong with our results?

- A new algorithm was designed based on data that had been provided to us by the National Institute of Health (NIH). [Wang and Japkowicz, 2008, 2010]
- According to the standard evaluation practices in machine learning, we found our results to be significantly better than the state-of-the-art.
- The conference version of the paper received a best paper award and the work is receiving a bit of attention in the ML community (35+ citations)
- NIH would not consider our algorithm because it was *probably* not *significantly* better (if at all) than others as they felt we had not conducted our evaluation according to their standards.

Example III: What do our results mean?

- In an experiment comparing the accuracy performance of eight classifiers on ten domains, we concluded that One-Way Repeated-Measure ANOVA rejects the hypothesis that the eight classifiers perform similarly on the ten domains considered at the 95% significance level [Japkowicz & Shah, 2011], p. 273.

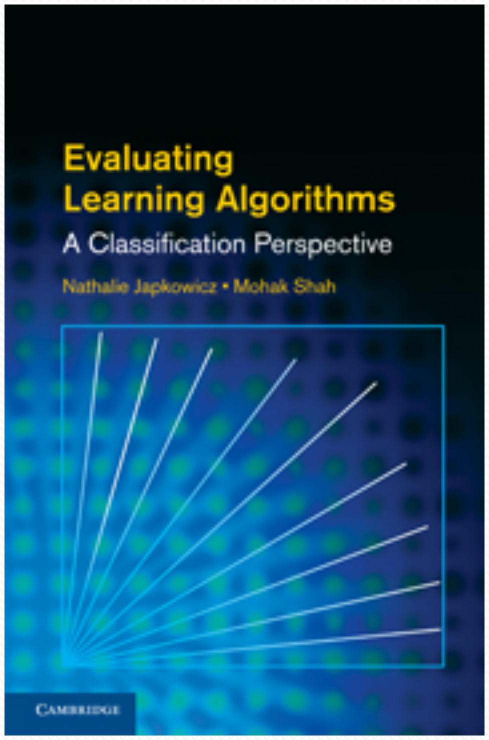
- Can we be sure of our results?
  - What if we are in the 5% not vouched for by the statistical test?
  - Was the ANOVA test truly applicable to our specific data? Is there any way to verify that?
  - Can we have any 100% guarantee on our results?

Purpose of the tutorial

- To give the audience an appreciation for the complexity and uncertainty of evaluation.
- To discuss the different aspects of evaluation and to present an overview of the issues that come up in their context and what the possible remedies may be.
- To present some of the newer lesser known results in classifier evaluation.
- Finally, to direct the audience to some available, easy-to-use, resources that can help improve the robustness of evaluation in the field.

Book on which the tutorial is based

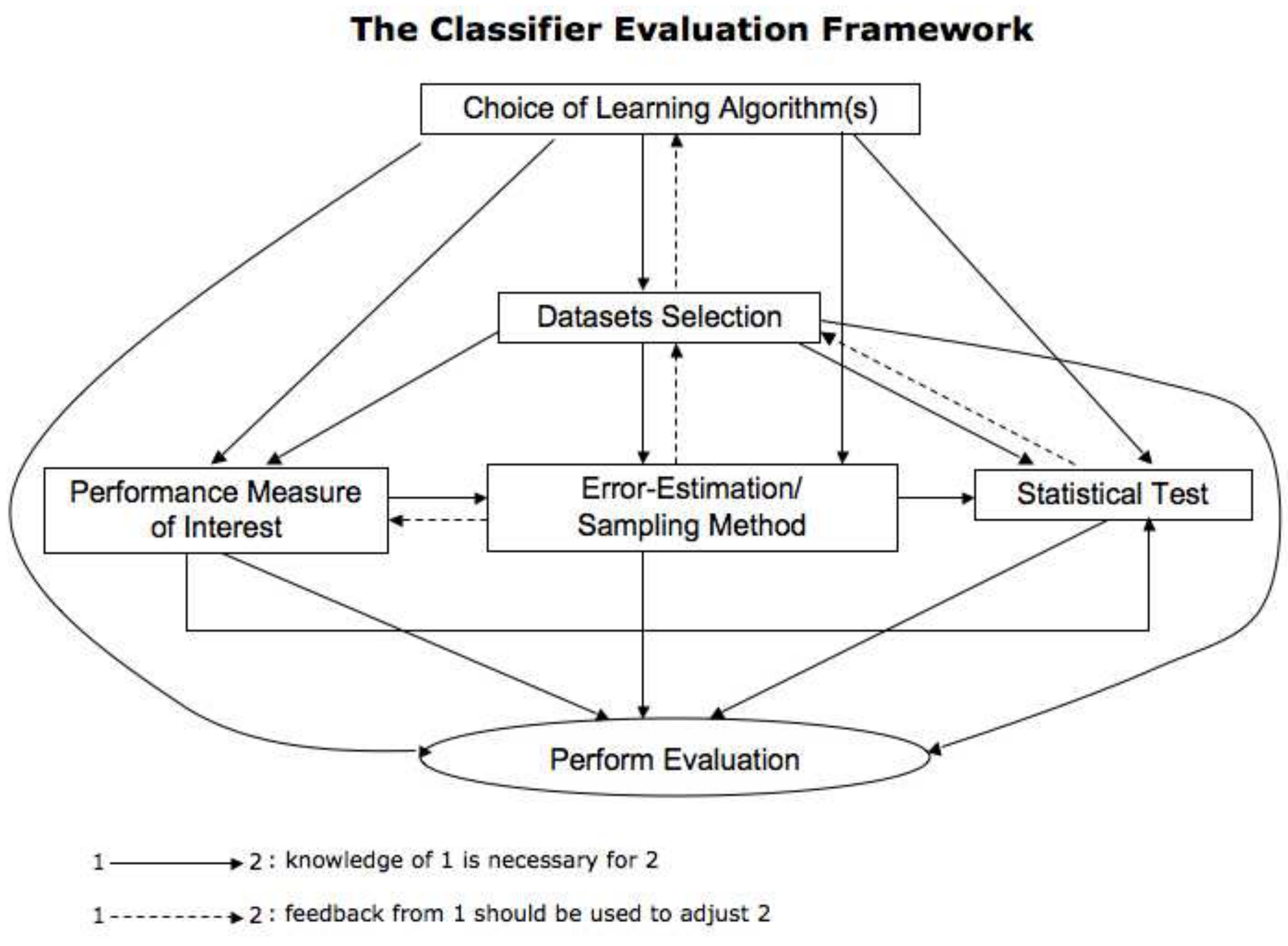
Evaluating Learning Algorithms:
A Classification Perspective
Nathalie Japkowicz & Mohak Shah
Cambridge University Press, 2011



- Review:

  "This treasure-trove of a book covers the important topic of performance evaluation of machine learning algorithms in a very comprehensive and lucid fashion. As Japkowicz and Shah point out, performance evaluation is too often a formulaic affair in machine learning, with scant appreciation of the appropriateness of the evaluation methods used or the interpretation of the results obtained. This book makes significant steps in rectifying this situation by providing a reasoned catalogue of evaluation measures and methods, written specifically for a machine learning audience and accompanied by concrete machine learning examples and implementations in R. This is truly a book to be savoured by machine learning professionals, and required reading for Ph.D students."
  Peter A. Flach, University of Bristol

The main steps of evaluation


The Classifier Evaluation Framework
Choice of Learning Algorithm(s)
Datasets Selection
Performance Measure of Interest
Error-Estimation/ Sampling Method
Statistical Test
Perform Evaluation
1 → 2 : knowledge of 1 is necessary for 2
1 - - - → 2 : feedback from 1 should be used to adjust 2

What these steps depend on

- These steps depend on the purpose of the evaluation:
  - Comparison of a *new algorithm* to other (may be generic or application-specific) classifiers on a *specific domain* (e.g., when proposing a novel learning algorithm)
  - Comparison of a *new generic algorithm* to other generic ones on a set of *benchmark domains* (e.g. to demonstrate general effectiveness of the new approach against other approaches)
  - Characterization of *generic classifiers* on *benchmarks domains* (e.g. to study the algorithms' behavior on general domains for subsequent use)
  - Comparison of *multiple classifiers* on a *specific domain* (e.g. to find the best algorithm for a given application task)

Outline of the tutorial:

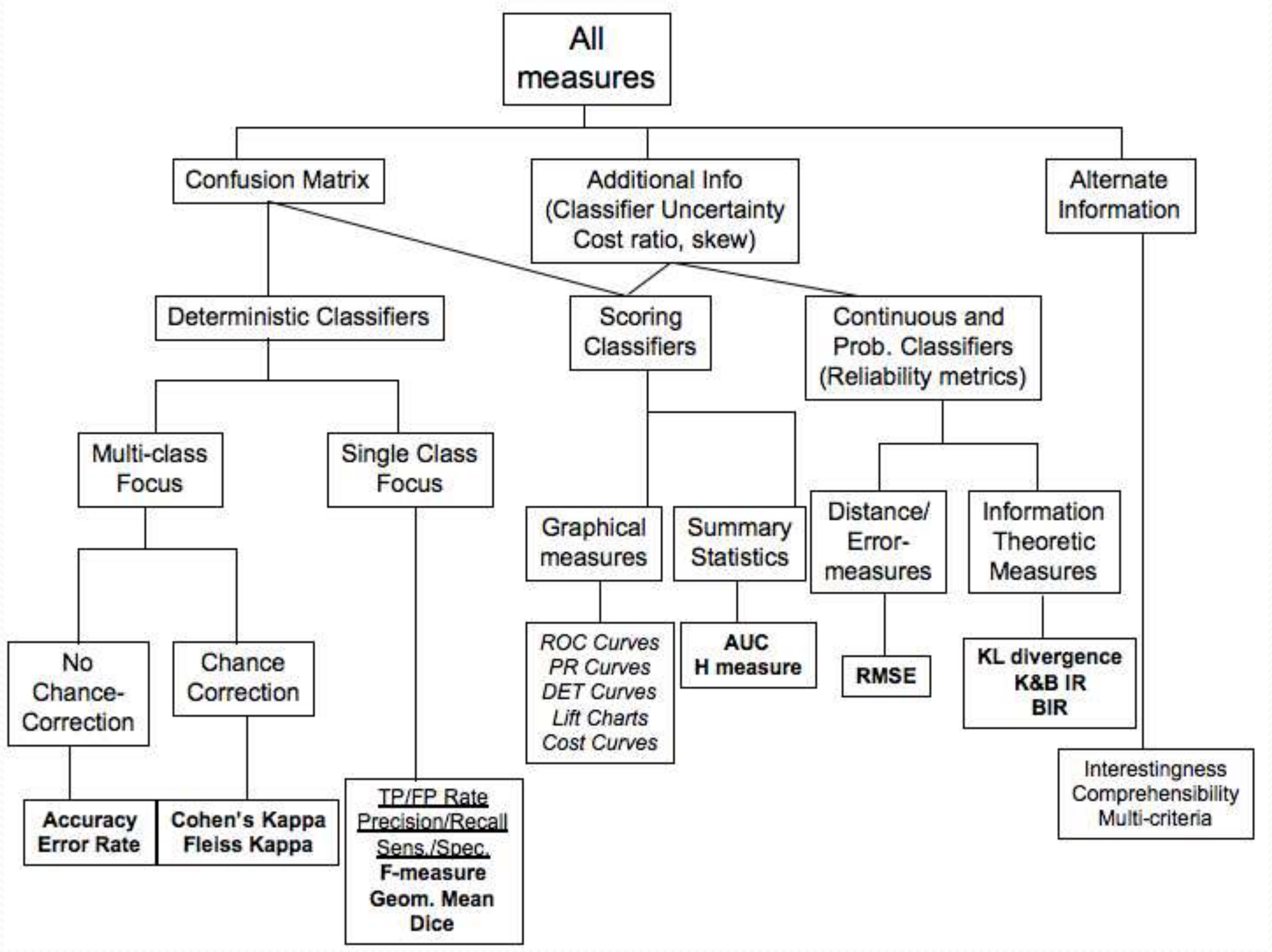
- Performance measures
- Error Estimation/Resampling
- Statistical Significance Testing
- Data Set Selection and Evaluation Benchmark Design
- Available resources

Performance Measures

Performance Measures
Outline

- Ontology
- Confusion Matrix Based Measures
- Graphical Measures:
  - ROC Analysis and AUC, Cost Curves
  - Recent Developments on the ROC Front: Smooth ROC Curves, the H Measure
  - Other Curves: PR Curves, lift, etc..
- Probabilistic Measures
- Others: qualitative measures, combinations frameworks, visualization, agreement statistics

Overview of Performance Measures



Confusion Matrix-Based Performance Measures

True class → Hypothesized | class V	Pos	Neg
Yes	TP	FP
No	FN	TN
	P=TP+FN	N=FP+TN

A 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)

Aliases and other measures

- Accuracy = 1 (or 100%) - Error rate
- Recall = TPR = Hit rate = Sensitivity
- Fallout = FPR = False Alarm rate
- Precision = Positive Predictive Value (PPV)

- Negative Predictive Value (NPV) = TN/(TN+FN)
- Likelihood Ratios:
  - LR+ = Sensitivity/(1-Specificity)
  - LR- = (1-Sensitivity)/Specificity

Pairs of Measures and Compounded Measures

- Precision / Recall
- Sensitivity / Specificity
- Likelihood Ratios (LR+ and LR-)
- Positive / Negative Predictive Values

- F-Measure:
  - $F\alpha = [(1+ \alpha)\ (\text{Precision} \times \text{Recall})] / [(\alpha \times \text{Precision}) + \text{Recall}]$ $\alpha = 1, 2, 0.5$
- G-Mean: 2-class version single-class version
  - G-Mean = Sqrt(TPR x TNR) or Sqrt(TPR x Precision)

Skew and Cost considerations

- Skew-Sensitive Assessments: e.g., Class Ratios
  - Ratio+ = (TP + FN)/(FP + TN)
  - Ratio- = (FP + TN)/(TP + FN)
  - TPR can be weighted by Ratio+ and TNR can be weighted by Ratio-
- Asymmetric Misclassification Costs:
  - If the costs are known,
    - Weight each non-diagonal entries of the confusion matrix by the appropriate misclassification costs
    - Compute the weighted version of any previously discussed evaluation measure.

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!

Some issues with performance measures (cont’d)

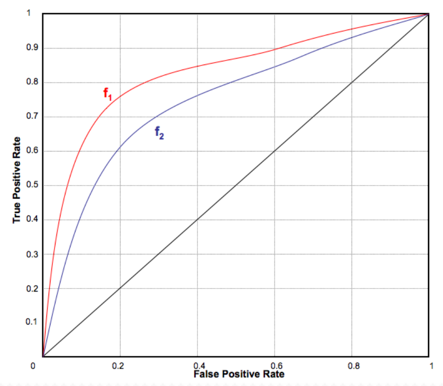
- This and similar analyses reveal that each performance measure will convey some information and hide other.
- Therefore, there is an information trade-off carried through the different metrics.
- A practitioner has to choose, quite carefully, the quantity s/he is interested in monitoring, while keeping in mind that other values matter as well.
- Classifiers may rank differently depending on the metrics along which they are being compared.

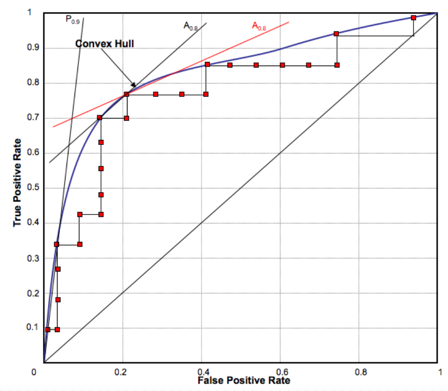
Graphical Measures
ROC Analysis, AUC, Cost Curves

- Many researchers have now adopted the AUC (the area under the ROC Curve) to evaluate their results.
- The principal advantage of the AUC is that it is more robust than Accuracy in class imbalanced situations where the default classifier that always outputs the most probable label yields exceedingly good estimates of performance.
- The AUC takes the class distribution into consideration and, therefore, gives more weight to correct classification of the minority class, thus outputting fairer results.

ROC Analysis

- ROC Analysis is applicable to scoring rather than merely deterministic classifiers
- ROC graphs are insensitive to class imbalances (or skew) since they consider the TPR and FPR independently and do not take into account the class distribution. They, therefore, give very nice overall comparisons of two systems.
- However, practically speaking, ROC graphs ignore the skew which the performance measures of interest (*pmi*) usually takes into consideration. Therefore, at model selection time, it is wise to consider isometrics for *pmi* which are lines in the ROC space along which the same performance value is obtained for that *pmi*. Different skew ratios are represented by different isolines, making the selection of the optimal operating point quite easy.



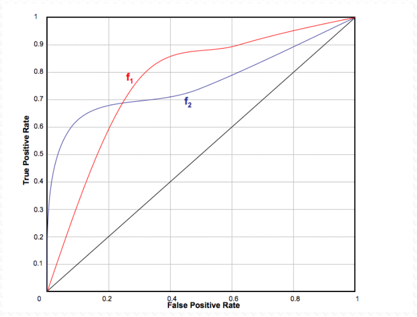


AUC

- 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.
- The AUC can also be interpreted as the probability of the classifier assigning a higher rank to a randomly chosen positive example than to a randomly chosen negative example
- Using the second interpretation, the AUC can be estimated as follows:

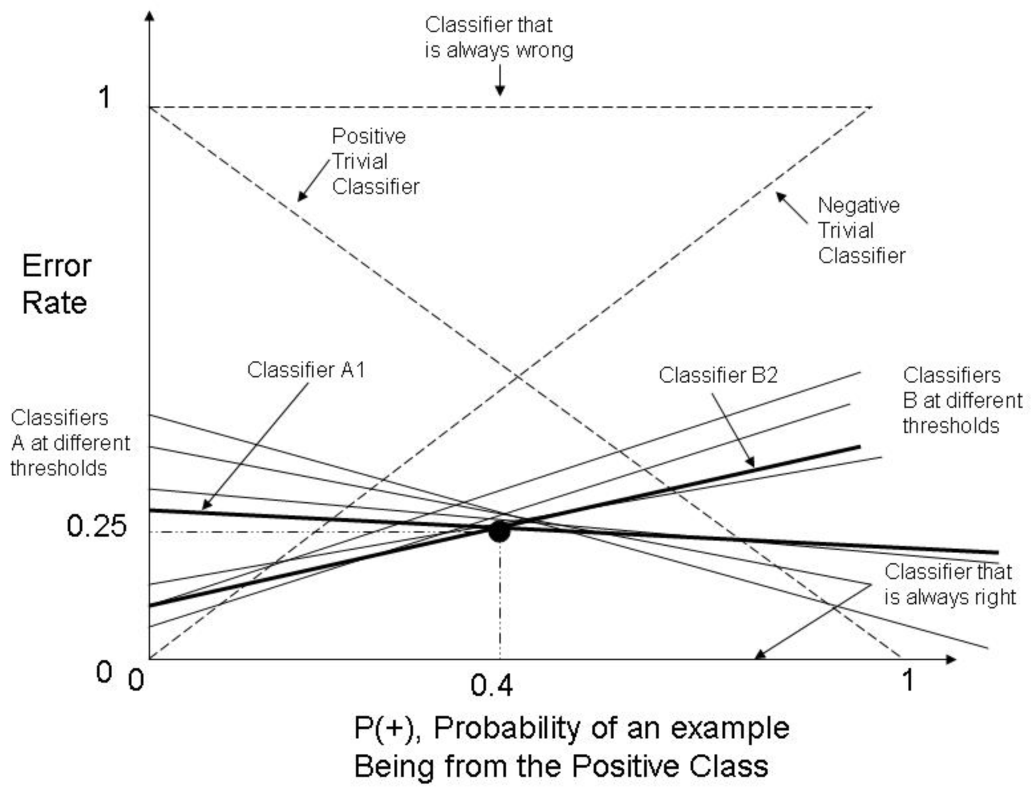
$$\boxed{\text{AUC} = \frac{\sum_{i=1}^{|Tp|}(Ri - i)}{|Tp||Tn|}}$$

Cost Curves



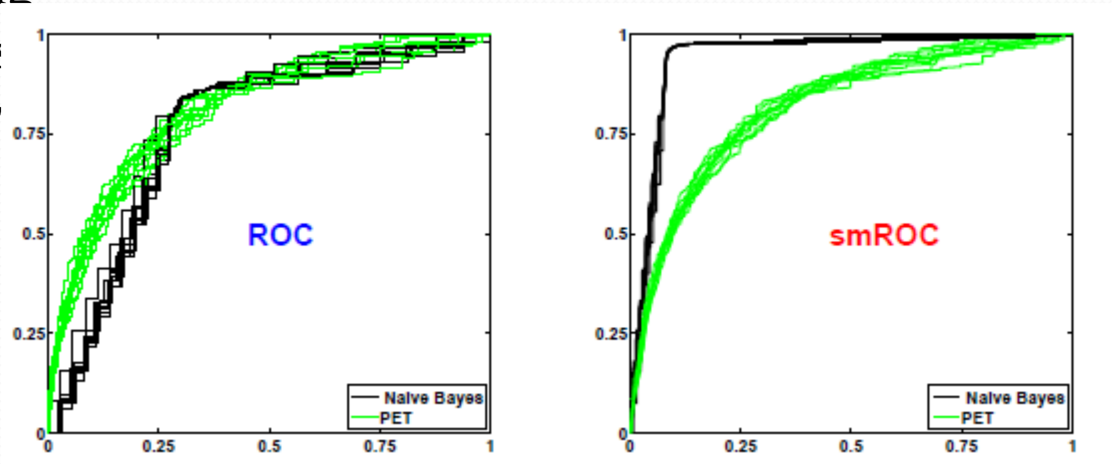
Cost-curves are more practical than ROC curves because they tell us *for what class probabilities* one classifier is preferable over the other.

ROC Curves only tell us that *sometimes* one classifier is preferable over the other



Recent Developments I: Smooth ROC Curves (smROC) [Klement, Flach, Japkowicz & Matwin, ECML'2011]

- ROC Analysis visualizes the ranking performance of classifiers but ignores any kind of scoring information.
- Scores are important as they tell us whether the difference between two rankings is large or small, but they are difficult to integrate because unlike probabilities, each scoring system is different from one clas to the next and their integr is not obvious.
- smROC extends the ROC curve to include normalized scores as smoothing weights added to its segments.
- smROC was shown to capture the similarity between similar classification models better than ROC and that it can capture difference between classification models that ROC is barely sensitive to, if at all.



Recent Developments II: The H Measure I

- A criticism of the AUC was given by [Hand, 2009]. The argument goes along the following lines:

  The misclassification cost distributions (and hence the skew-ratio distributions) used by the AUC are different for different classifiers. Therefore, we may be comparing apples and oranges as the AUC may give more weight to misclassifying a point by classifier A than it does by classifier B

- To address this problem, [Hand, 2009] proposed the H-Measure.
- In essence, The H-measure allows the user to select a cost-weight function that is equal for all the classifiers under comparison and thus allows for fairer comparisons.

Recent Developments II: The H Measure II

- [Flach, Hernandez-Orallo and Ferri, ICML'2011] argued that
  - [Hand'2009]'s criticism holds when the AUC is interpreted as a classification measure but not when it is interpreted as a pure ranking measure.
  - When [Hand'2009]'s criticism holds, the problem occurs only if the threshold used to construct the ROC curve are assumed to be optimal.
  - When constructing the ROC curve using all the points in the data set as thresholds rather than only optimal ones, then the AUC measure can be shown to be coherent and the H-Measure unnecessary, at least from a theoretical point of view.
  - The H-Measure is a linear transformation of the value that the area under the Cost-Curve would produce.

Other Curves

- Other research communities have produced and used graphical evaluation techniques similar to ROC Curves.
  - Lift charts plot the number of true positives versus the overall number of examples in the data sets that was considered for the specific true positive number on the vertical axis.
  - PR curves plot the precision as a function of its recall.
  - DET curves are like ROC curves, but plot the FNR rather than the TPR on the vertical axis. They are also typically log-scaled.
  - Relative Superiority Graphs are more akin to Cost-Curves as they consider costs. They map the ratios of costs into the [0,1] interval.

Probabilistic Measures I: RMSE

- The Root-Mean Squared Error (RMSE) is usually used for regression, but can also be used with probabilistic classifiers. The formula for the RMSE is:

$$RMSE(f) = sqrt(1/m \Sigma_{i=1}^{m}(f(x_i) - y_i)^2))$$

where m is the number of test examples, $f(x_i)$, the classifier's probabilistic output on x_i and y_i the actual label.

ID	$f(x_i)$	y_i	$(f(x_i) - y_i)^2$
1	.95	1	.0025
2	.6	0	.36
3	.8	1	.04
4	.75	0	.5625
5	.9	1	.01

RMSE(f) = sqrt(1/5 * (.0025+.36+.04+.5625+.01))
= sqrt(0.975/5) = 0.4416

Probabilistic Measures II: Information Score

- Kononenko and Bratko's Information Score assumes a prior P(y) on the labels. This could be estimated by the class distribution of the training data. The output (posterior probability) of a probabilistic classifier is P(y|f), where f is the classifier. I(a) is the indicator function.
- $IS(x) = I(P(y|f) \geq P(y)) * (-\log(P(y))+\log(P(y|f)) +$
  $+ I(P(y|f) < P(y)) * (-\log(1-P(y))+\log(1-P(y|f)))$
- $IS_{avg} = 1/m \sum_{i}^{m}(IS(x_i))$

x	$P(y_i\|f)$	y_i	IS(x)
1	.95	1	0.66
2	.6	0	0
3	.8	1	.42
4	.75	0	.32
5	.9	1	.59

$P(y=1) = 3/5 = 0.6$
$P(y=0) = 2/5 = 0.4$
$IS(x_1) = 1 * (-\log(.6) + \log(.95)) +$
$\quad 0 * (-\log(.4) + \log).05))$
$\quad = 0.66$

$IS_{avg} = 1/5\ (0.66+0+.42+.32+.59)$
$\quad = 0.40$

Other Measures I:
A Multi-Criteria Measure – The Efficiency Method

- The efficiency method is a framework for combining various measures including qualitative metrics, such as interestingness. It considers the positive metrics for which higher values are desirable (e.g, accuracy) and the negative metrics for which lower values are desirable (e.g., computational time).

$$\varepsilon_S(f) = \Sigma_i w_i pm_i^+(f) \; / \; \Sigma_j w_j nm_j^-(f)$$

pm_i^+ are the positive metrics and nm_j^- are the negative metrics. The w_i's and w_j's have to be determined and a solution is proposed that uses linear programming.

Other Measures II:

Visualization-Based Measure *[Japkowicz, Sanghi, Tischer, ECML'08, ISAIM'08]*

- Classifier evaluation can be viewed as a problem of analyzing high-dimensional data.
- The performance measures currently used are but one class of projections that can be applied to these data.
- Why not apply other (standard or not) projections to the data with various kinds (standard or not) distance measures?

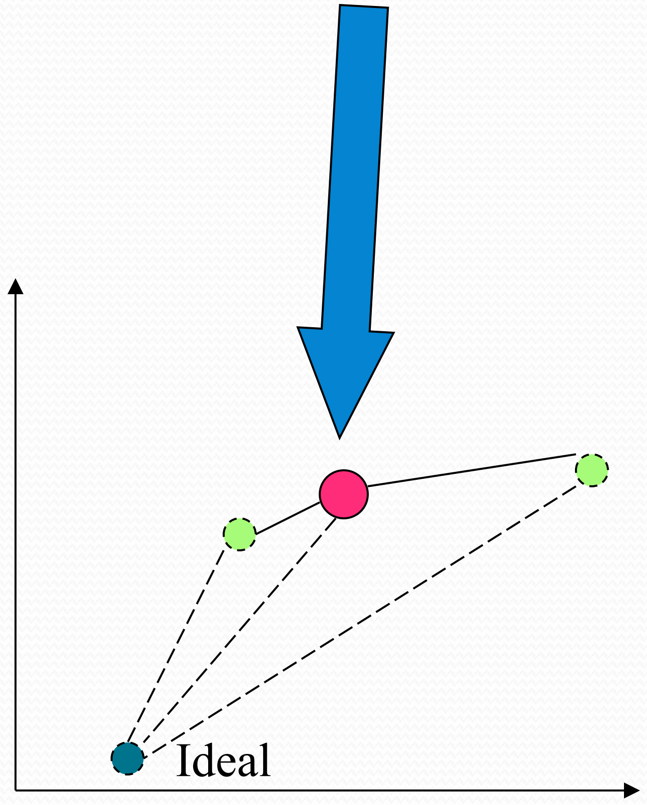
Illustration of the Framework

True class →	Pos	Neg
Yes	82	17
No	12	114

True class →	Pos	Neg
Yes	15	5
No	25	231

True class →	Pos	Neg
Yes	99	6
No	1	94

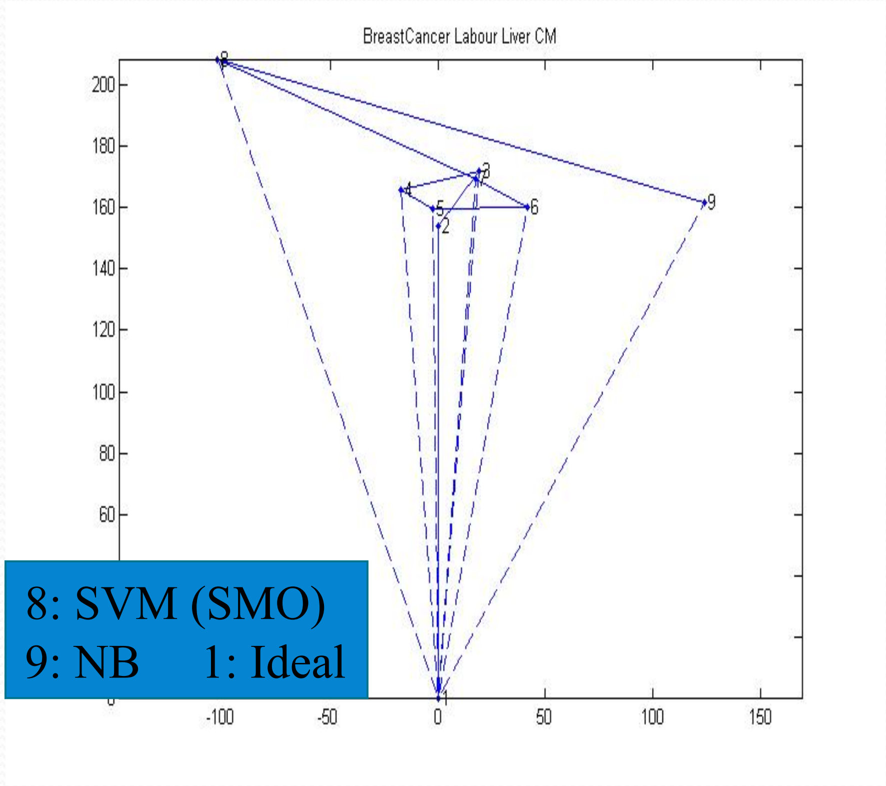
82	17	12	114	15	5	25	231	99	6	1	94



Confusion matrices for a single classifier on three domains

Illustration on Multiple domains:

Breast Cancer, Labor and Liver

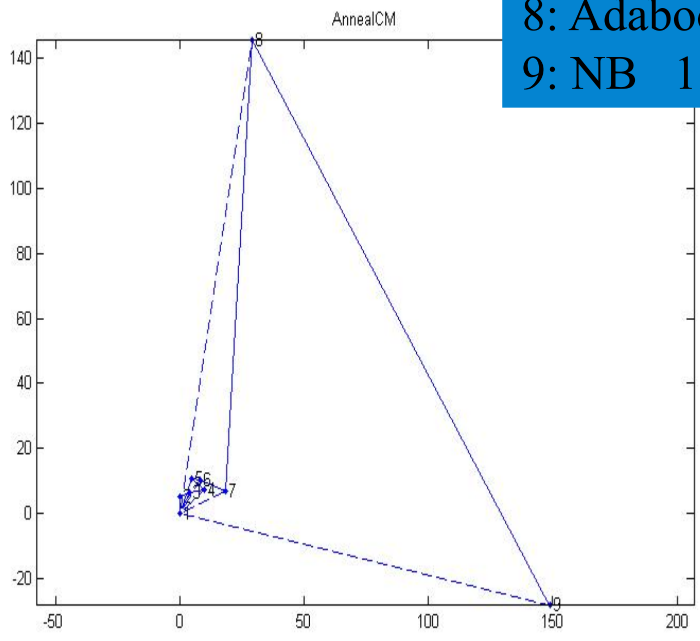


	Acc.	F-Meas.	AUC
NB BC:	71.7	.48	.7
La:	89.5	.92	.97
Li:	55.4	.6	.64
Avg:	72.2	.67	.77
SMO BC:	69.6	.39	.59
La:	89.5	.92	.87
Li:	58.3	.014	.5
Avg:	72.46	.44	.65
Boost. BC:	70.3	.46	.7
La:	87.7	.91	.87
Li:	66.1	.534	.68
Avg:	74.7	.64	.75

Abnormality detection with our new approach is a lot easier and accurate than it is, when relying on Accuracy, F-Measure, or AUC listings on each domain or their average on all domains.

Also, our new approach Allows us to mix binary and multi-class domains. Averaging does not!

Illustration on a Multiclass domain: Anneal



8: Adaboost
9: NB 1: Ideal

Adaboost:

a	b	c	d	e	f	← classified as
0	0	8	0	0	0	| a
0	0	99	0	0	0	| b
0	0	684	0	0	0	| c
0	0	0	0	0	0	| d
0	0	0	0	67	0	| e
0	0	40	0	0	0	| f

NB:

a	b	c	d	e	f	← classified as
7	0	1	0	0	0	| a
0	99	0	0	0	0	| b
3	38	564	0	0	79	| c
0	0	0	0	0	0	| d
0	0	0	0	67	0	| e
0	0	2	0	0	38	| f

NB	Other Classifiers	Ada boost
86.3	97.4 to 99.3	83.6

Accuracy

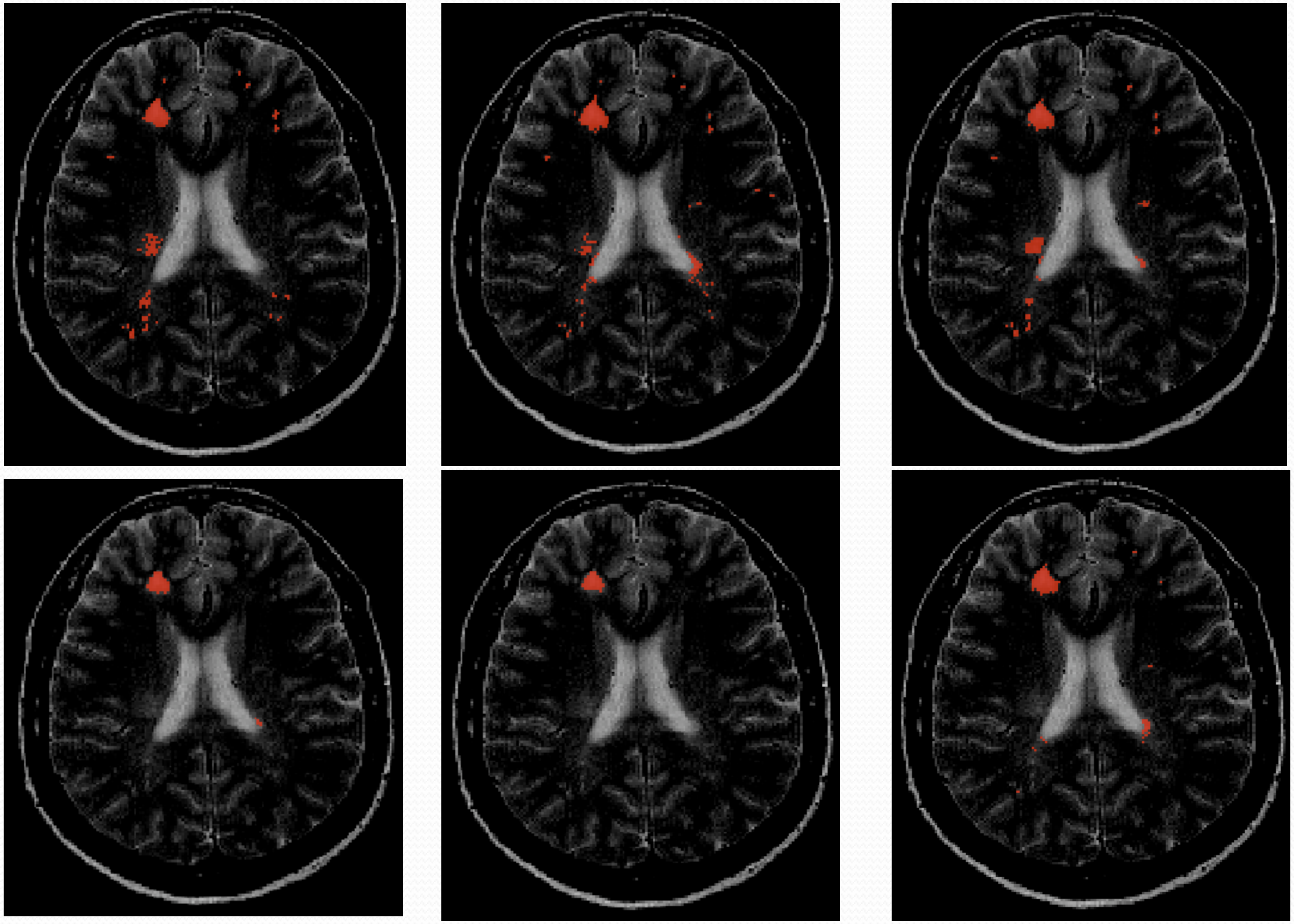
Accuracy does not tell us whether NB and Adaboost make the same kind of errors!

Other Measures III:

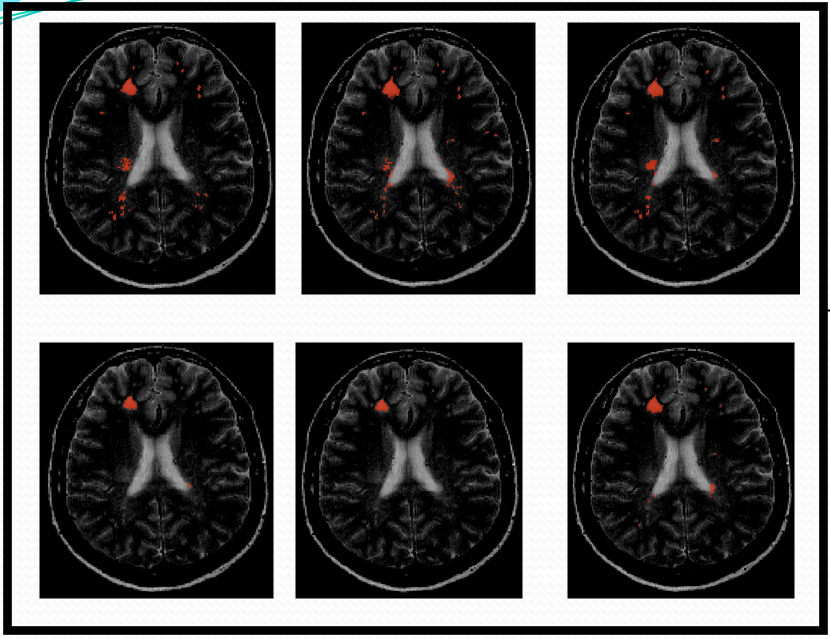
Accounting for chance and evaluating under imprecise labeling

- Measures considered so far assume existence of ground truth labels (against which, for instance, accuracy is measured)
- In the absence of ground truth, typically experts label the instances

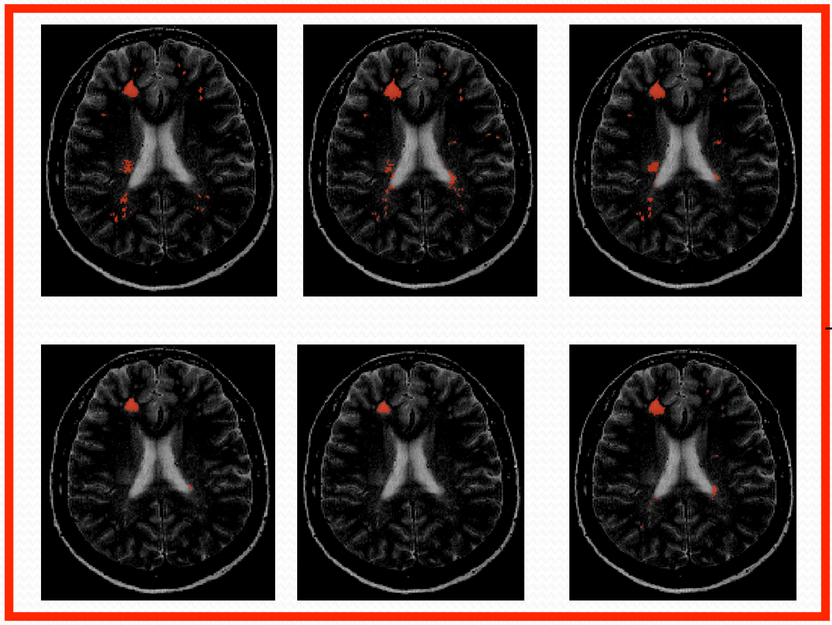
Multiple Annotations

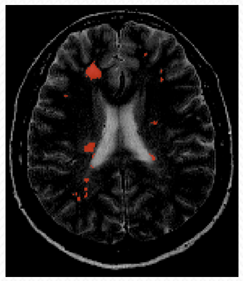


2 Questions of interest



Inter-expert agreement:
Overall Agreement
of the group





Classifier Agreement
Against the group

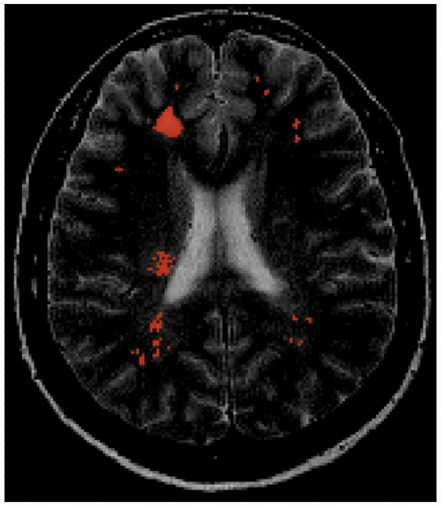
Such measurements are also desired when

- Assessing a classifier in a skewed class setting
- Assessing a classifier against expert labeling
- There are reasons to believe that bias has been introduced in the labeling process (say, due to class imbalance, asymmetric representativeness of classes in the data,…)
- …

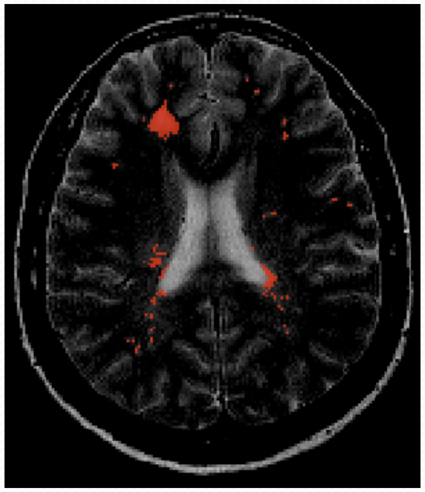
Agreement statistic aim to address this by accounting for chance agreements (coincidental concordances)

Coincidental concordances (Natural Propensities) need to be accounted for

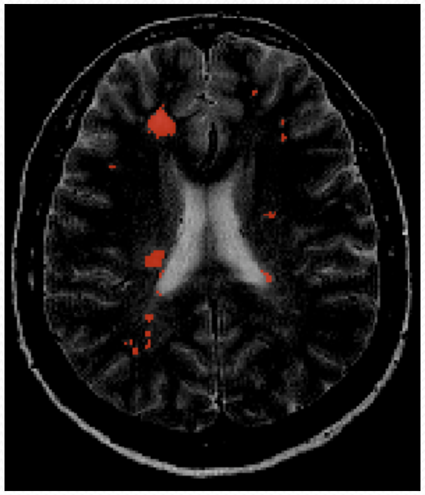
R1: 1526

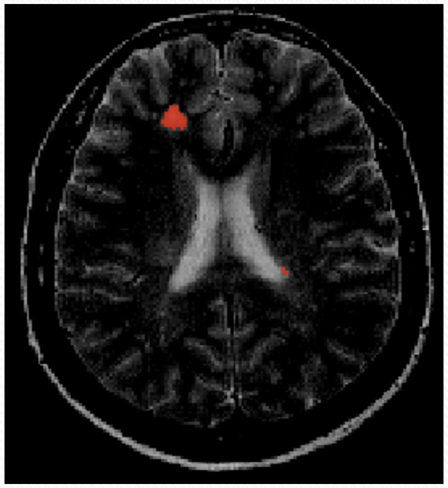


R2: 2484

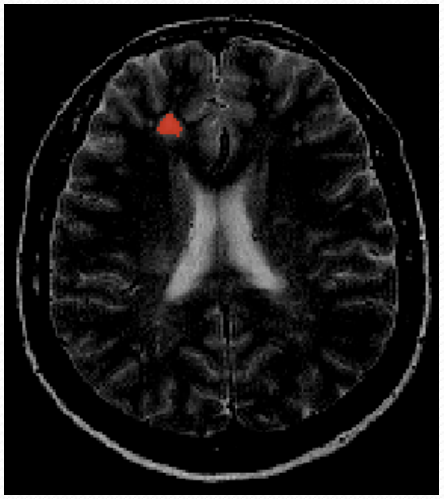


R3: 1671

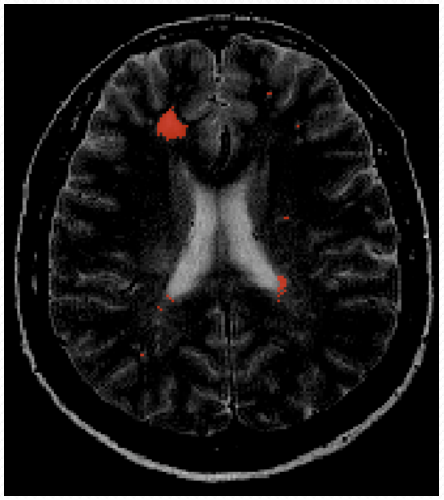




R4: 816



R5: 754



R6:1891

General Agreement Statistic

$$\kappa = \frac{\mathbf{E}_S(\mathbb{A}) - \mathbf{E}(\mathbb{A})}{\max_S(\mathbb{A}) - \mathbf{E}(\mathbb{A})}$$

General Agreement Statistic

Agreement measure

$$\kappa = \frac{\mathbf{E}_S(\mathbb{A}) - \mathbf{E}(\mathbb{A})}{\boxed{\max_S(\mathbb{A})} - \mathbf{E}(\mathbb{A})}$$

Maximum Achievable Agreement

Chance Agreement

Examples: Cohen's kappa, Fleiss Kappa, Scott's pi, ICC…

Multiple raters over multiple classes

- Various generalizations to estimate inter-rater agreement in general multi-class multi-rater scenario
  - E.g., Fleiss kappa (Fleiss 1971); ICC, Scott's pi; (Vanbelle and Albert, 2009);
  - Typically relying on marginalization argument (fixed number of raters assumed to be sampled from a pool of raters to label each instance)
  - Recently generalization for the fixed rater scenarios have been proposed (κ_S measure (Shah, 2011)) to address this

- For assessing agreement against a group of raters
  - Typically consensus argument (deterministic label set formed via majority vote for evaluation and the classifier is evaluated against this label set) or marginalization based measures
  - Issues with these argument were highlighted for fixed rater setting (Shah, 2011)
    - A novel measure, the *S* measure, has been proposed to address this

Error Estimation/ Resampling

Se we decided on a performance measure.

- How do we estimate it in an unbiased manner?

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

- Which factors affect such estimation choices?
  - Sample Size/Domains
  - Random variations: in training set, testing set, learning algorithms, class-noise
  - …

Hold-out approach

- Set aside a separate test set T. Estimate the empirical risk:

$$R_T(f) = \int L(y, f_S(\mathbf{x}))dD(\mathbf{x}, y)$$

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

Hold-out approach

- Confidence intervals can be found too (based on binomial approximation):

$$\left|R_T(f) - R(f)\right| \leq t_{1-\delta} = \epsilon = \sqrt{\frac{1}{2m'} \ln\left(\frac{2}{\delta}\right)}$$

A Tighter bound

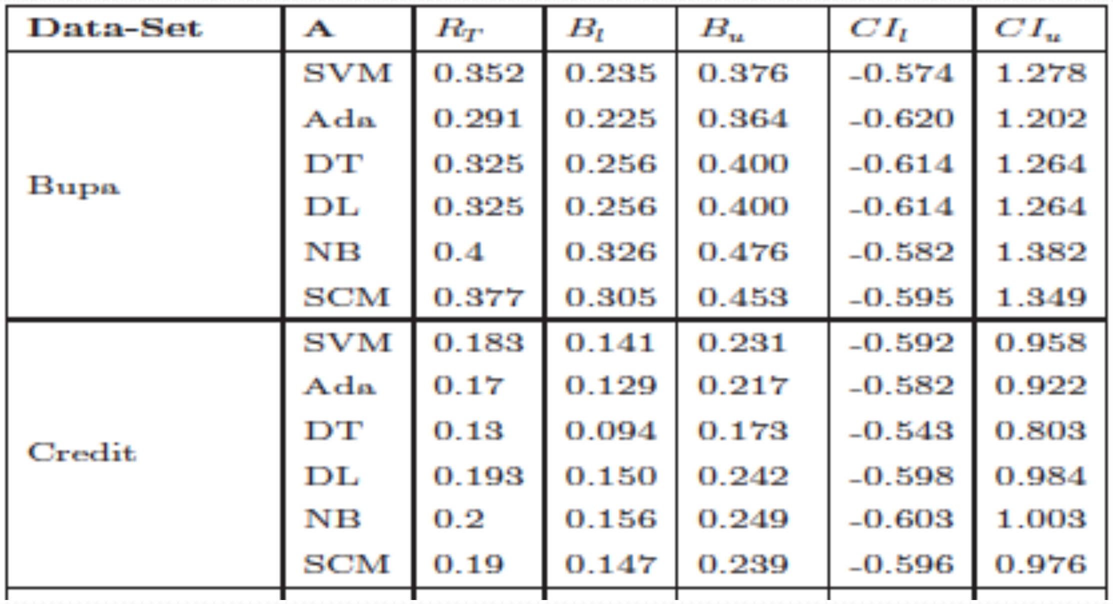
- Based on binomial inversion:

$$\Pr_{T\sim\mathcal{D}^m}(R(f) \leq \overline{\mathrm{Bin}}(m, \lambda, \delta)) \geq 1 - \delta$$

- More realistic than asymptotic Gaussian assumption ($\mu \pm 2\sigma$)

Binomial vs. Gaussian assumptions

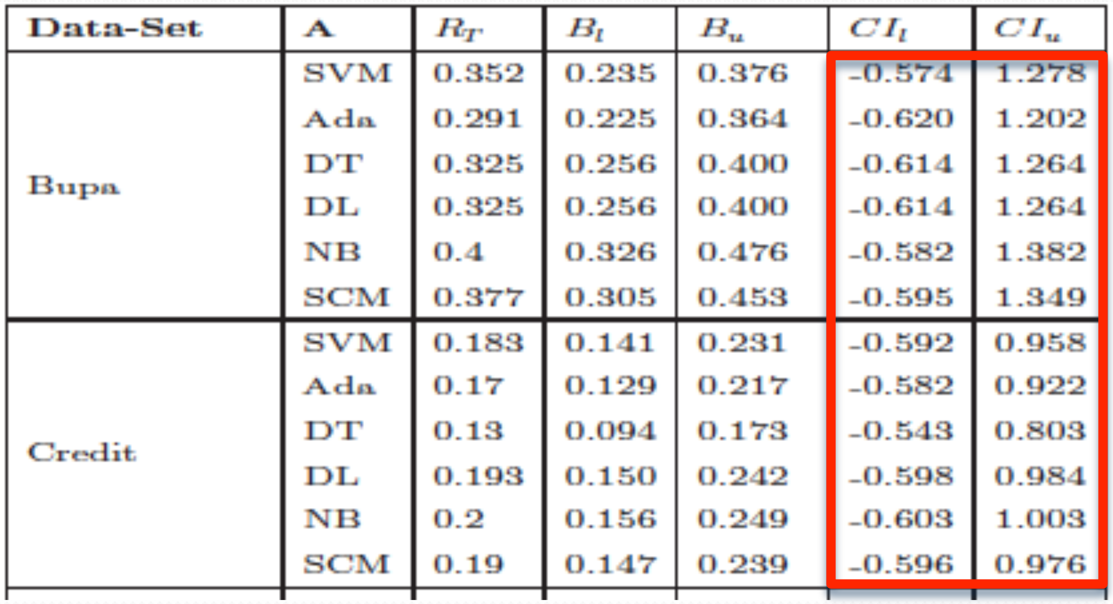
- B_l and B_u shows binomial lower and upper bounds
- CI_l and CI_u shod lower and upper confidence intervals based on Gaussian assumption



Data-Set	A	R_T	B_l	B_u	CI_l	CI_u
Bupa	SVM	0.352	0.235	0.376	-0.574	1.278
	Ada	0.291	0.225	0.364	-0.620	1.202
	DT	0.325	0.256	0.400	-0.614	1.264
	DL	0.325	0.256	0.400	-0.614	1.264
	NB	0.4	0.326	0.476	-0.582	1.382
	SCM	0.377	0.305	0.453	-0.595	1.349
Credit	SVM	0.183	0.141	0.231	-0.592	0.958
	Ada	0.17	0.129	0.217	-0.582	0.922
	DT	0.13	0.094	0.173	-0.543	0.803
	DL	0.193	0.150	0.242	-0.598	0.984
	NB	0.2	0.156	0.249	-0.603	1.003
	SCM	0.19	0.147	0.239	-0.596	0.976

Binomial vs. Gaussian assumptions

- B_l and B_u shows binomial lower and upper bounds
- CI_l and CI_u shod lower and upper confidence intervals based on Gaussian assumption



Data-Set	A	R_T	B_l	B_u	CI_l	CI_u
Bupa	SVM	0.352	0.235	0.376	-0.574	1.278
	Ada	0.291	0.225	0.364	-0.620	1.202
	DT	0.325	0.256	0.400	-0.614	1.264
	DL	0.325	0.256	0.400	-0.614	1.264
	NB	0.4	0.326	0.476	-0.582	1.382
	SCM	0.377	0.305	0.453	-0.595	1.349
Credit	SVM	0.183	0.141	0.231	-0.592	0.958
	Ada	0.17	0.129	0.217	-0.582	0.922
	DT	0.13	0.094	0.173	-0.543	0.803
	DL	0.193	0.150	0.242	-0.598	0.984
	NB	0.2	0.156	0.249	-0.603	1.003
	SCM	0.19	0.147	0.239	-0.596	0.976

Hold-out sample size requirements

- The bound:

$$|R_T(f) - R(f)| \leq t_{1-\delta} = \epsilon = \sqrt{\frac{1}{2m'} \ln\left(\frac{2}{\delta}\right)}$$

gives

$$m' \geq \frac{1}{2\epsilon^2} \ln\left(\frac{2}{\delta}\right)$$

Hold-out sample size bound

- The bound:

$$|R_T(f) - R(f)| \leq t_{1-\delta} = \epsilon = \sqrt{\frac{1}{2m'} \ln\left(\frac{2}{\delta}\right)}$$

gives

$$\boxed{m' \geq \frac{1}{2\epsilon^2} \ln\left(\frac{2}{\delta}\right)}$$

This sample size bound grows very quickly with ε and δ

The need for re-sampling

- Too few training examples -> learning a poor classifier
- Too few test examples -> erroneous error estimates

- Hence: Resampling
  - Allows for accurate performance estimates while allowing the algorithm to train on most data examples
  - Allows for closer duplication conditions

What implicitly guides re-sampling: a quick look into relationship with Bias-variance behavior

- Bias-Variance analysis helps understand the behavior of learning algorithms

- In the classification case:
  - Bias: difference between the most frequent prediction of the algorithm and the optimal label of the examples
  - Variance: Degree of algorithm's divergence from its most probable (average) prediction in response to variations in training set

- Hence, bias is independent of the training set while variance isn't

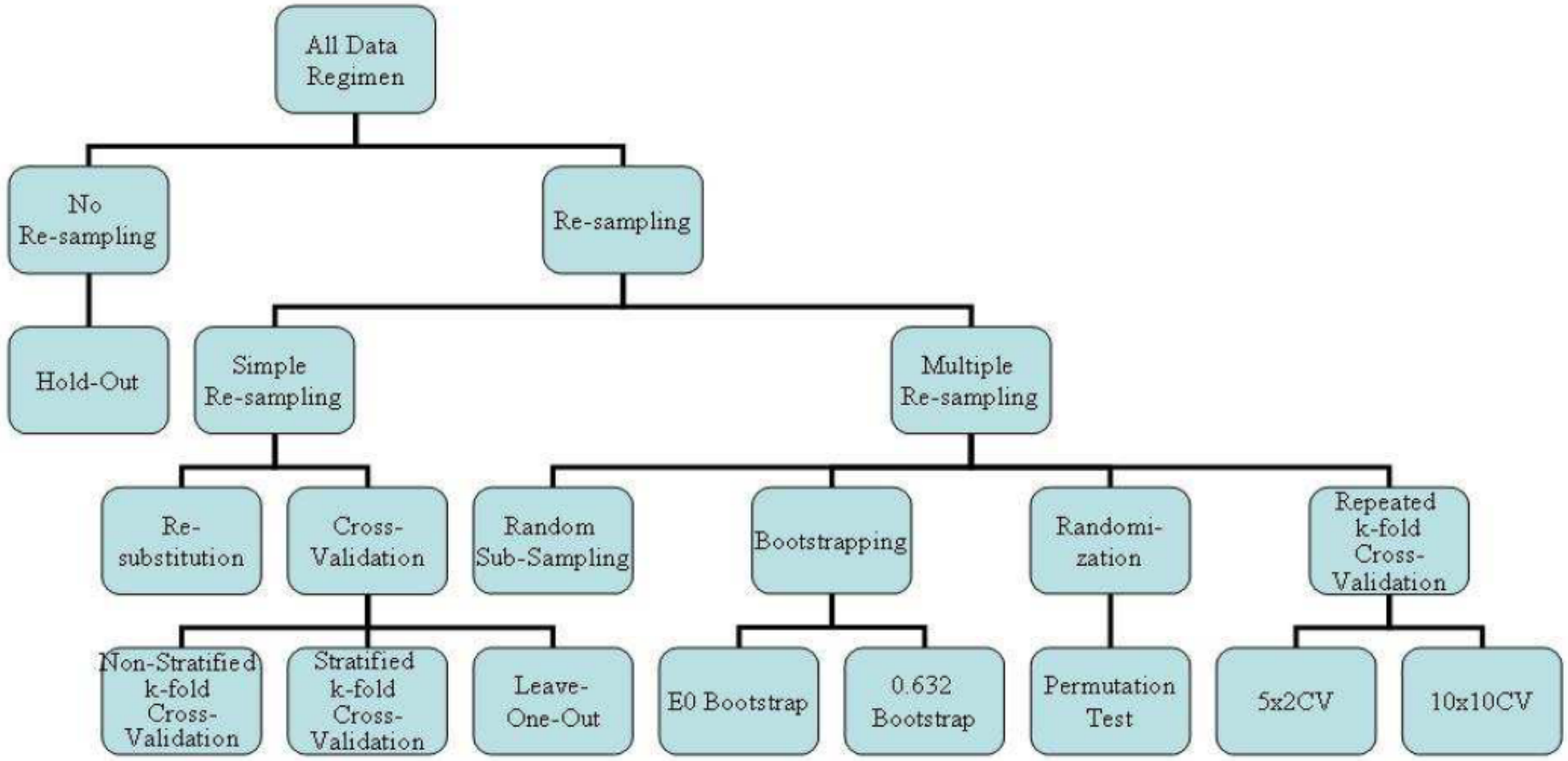
What implicitly guides re-sampling: a quick look into relationship with Bias-variance behavior

- Having too few examples in the training set affects the bias of algorithm by making its average prediction unreliable

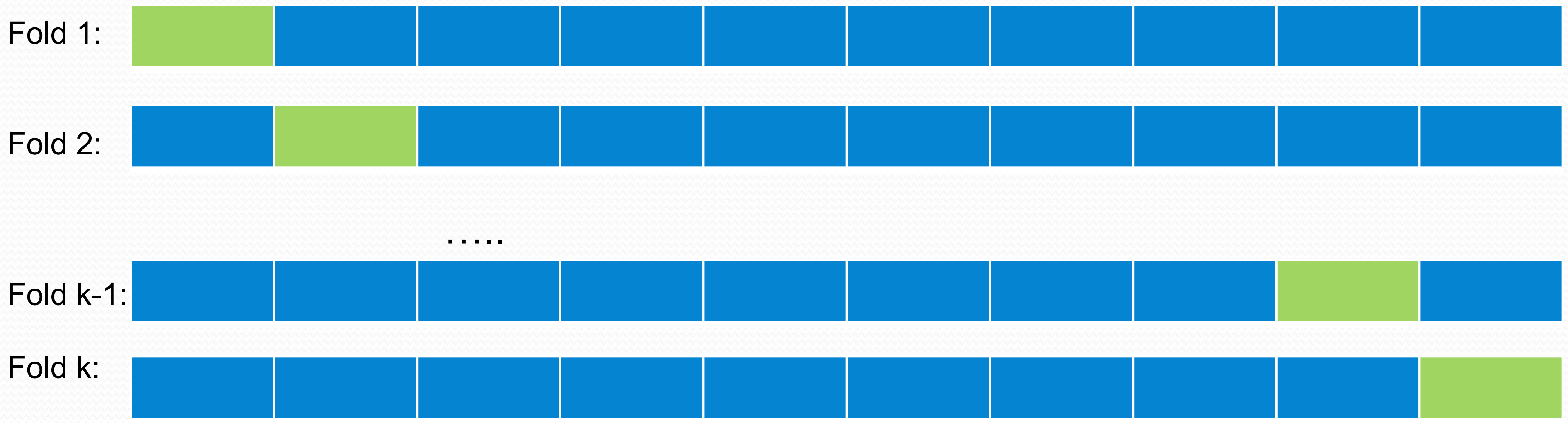
- Having too few examples in test set results in high variance

- Hence, the choice of resampling can introduce different limitations on the bias and variance estimates of an algorithm and hence on the reliability of error estimates

An ontology of error estimation techniques



Simple Resampling: k-fold Cross-Validation



🟦 : training data subset

🟩 : testing data subset

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 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

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 averaged estimate over k different classifiers

Observations

- Leave-One-Out can result in estimates with high variance given that the testing folds contains only one example
    - On the other hand, the classifier is practically unbiased since each training fold contains almost all the data
- Leave-One-Out can be quite effective for moderate dataset sizes
    - For small datasets, the estimates have high variance while for large datasets application becomes too computer-intensive
- Leave-One-Out can also be beneficial, compared to k-fold CV, when
    - There is wide dispersion of data distribution
    - Data set contains extreme values

Limitations of Simple Resampling

- Do not estimate the risk of a classifier but rather expected risk of algorithm over the size of partitions
  - While stability of estimates can give insight into robustness of algorithm, classifier comparisons in fact compare the averaged estimates (and not individual classifiers)

- Providing rigorous guarantees is quite problematic
  - No confidence intervals can be established (unlike, say, hold-out method)
  - Even under normality assumption, the standard deviation at best conveys the uncertainty in estimates

Multiple Resampling: Bootstrapping

- Attempts to answer: What can be done when data is too small for application of k-fold CV or Leave-One-Out?

- Assumes that the available sample is representative

- Creates a large number of new sample by drawing with replacement

Multiple Resampling: ε0 Bootstrapping

- Given a data D of size *m*, create *k* bootstrap samples B_i, each of of size *m* by sampling with replacement

- At each iteration i:
  - B_i represents the training set while the test set contains single copy of examples in D that are not present in B_i
  - Train a classifier on Bi and test on the test set to obtain error estimate $εo_i$

- Average over $εo_i$'s to obtain εo estimate

Multiple Resampling: ε0 Bootstrapping

- In every iteration the probability of an example not being chosen after *m* samples is:

$$(1 - \frac{1}{m})^m \approx \frac{1}{e} \approx 0.368$$

- Hence, average distinct examples in the training set is m(1-0.368)m = 0.632m

- The ε0 measure can hence be pessimistic since in each run the classifier is typically trained on about 63.2% of data

Multiple Resampling: e632 Bootstrapping

- e632 estimate corrects for this pessimistic bias by taking into account the optimistic bias of resubstitution error over the remaining 0.368 fraction:

$$e632 = 0.632 \cdot \epsilon 0 + 0.368 \cdot err(f)$$

Discussion

- Bootstrap can be useful when datasets are too small
  - In these cases, the estimates can also have low variance owing to (artificially) increased data size
- ε0: effective in cases of very high true error rate; has low variance than k-fold CV but more biased
- e632 (owing to correction it makes): effective over small true error rates
- An interesting observation: these error estimates can be algorithm-dependent
  - e.g., Bootstrap can be a poor estimator for algorithms such as nearest neighbor or FOIL that do not benefit from duplicate instances

Multiple Resampling: other approaches

- Randomization
  - Over labels: Assess the dependence of algorithm on actual label assignment (as opposed to obtaining similar classifier on chance assignment)
  - Over Samples (Permutation): Assess the stability of estimates over different re-orderings of the data
- Multiple trials of simple resampling
  - Potentially Higher replicability and more stable estimates
  - Multiple runs (how many?): 5x2 cv, 10x10 cv (main motivation is comparison of algorithms)

What to watch out when selecting error estimation method

- Domain (e.g., for data distribution, dispersion)
- Dataset Size
- Bias-Variance dependence
- (Expected) true error rate
- Type of learning algorithms to be evaluated (e.g. can be robust to, say, permutation test; or do not benefit over bootstrapping)
- Computational Complexity

The relations between the evaluation methods, whether statistically significantly different or not, varies with data quality. Therefore, one cannot replace one test with another, and only evaluation methods appropriate for the context may be used.

Reich and Barai (1999, Pg 11)

Statistical Significance Testing

Statistical Significance Testing

- Error estimation techniques allows us to obtain an estimate of the desired performance metric(s) on different classifiers
- A difference is seen between the metric values over classifiers
- Questions
  - Can the difference be attributed to real characteristics of the algorithms?
  - Can the observed difference(s) be merely coincidental concordances?

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) typically w.r.t.:
  - Evaluate algorithm A of interest against other algorithms on a specific problem
  - Evaluate generic capabilities of A against other algorithms on benchmark datasets
  - Evaluating performance of multiple classifiers on benchmark datasets or a specific problem of interest

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 that 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.

Choosing a Statistical Test

- There are several aspects to consider when choosing a statistical test.
  - What kind of problem is being handled?
  - Whether we have enough information about the underlying distributions of the classifiers' results to apply a parametric test?
  - …

- Statistical tests considered can be categorized by the task they address, i.e., comparing
  - 2 algorithms on a single domain
  - 2 algorithms on several domains
  - multiple algorithms on multiple domains

Issues with hypothesis testing

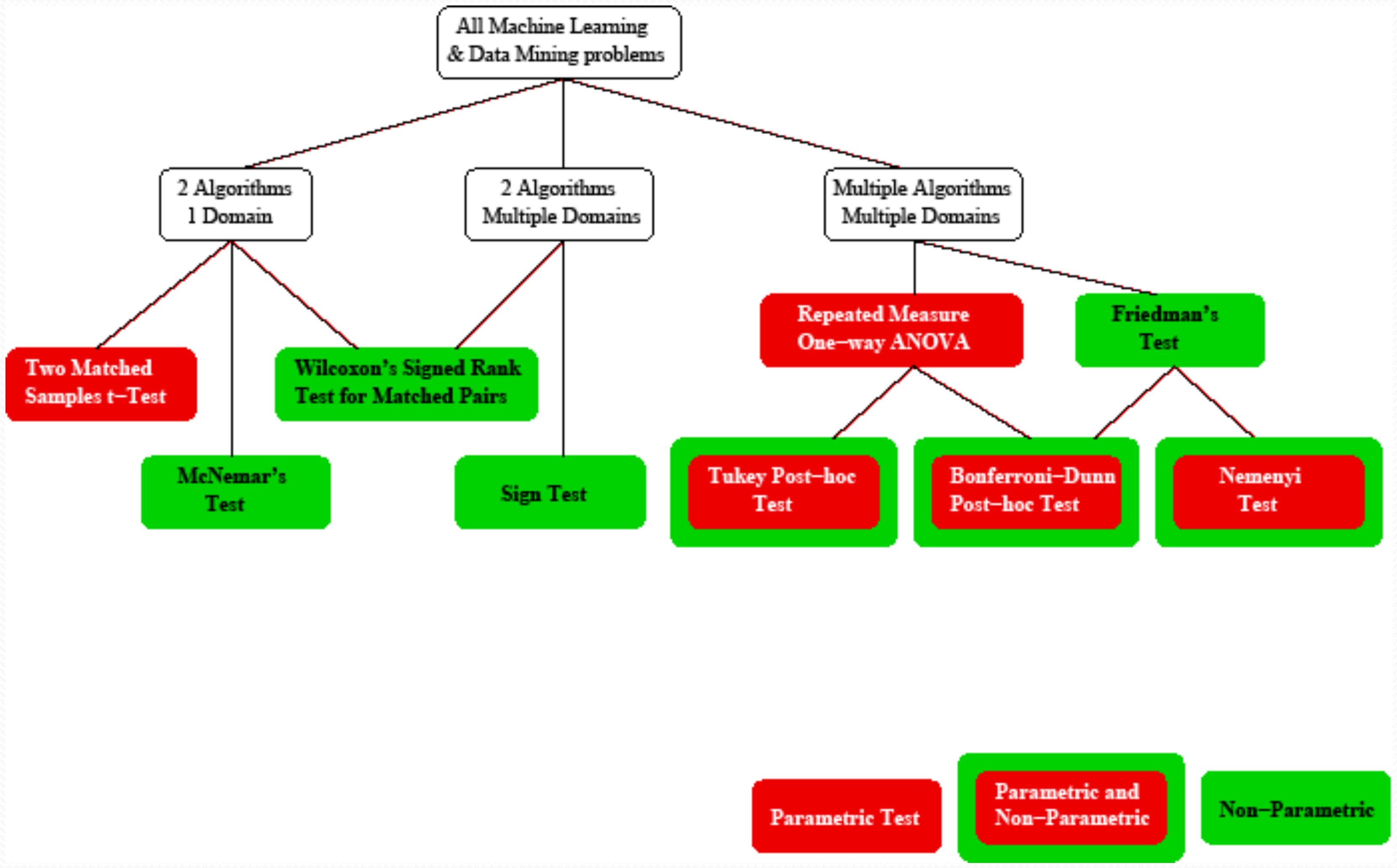
- NHST never constitutes a proof that our observation is valid
- Some researchers (Drummond, 2006; Demsar, 2008) have pointed out that NHST:
  - overvalues the results; and
  - limits the search for novel ideas (owing to excessive, often unwarranted, confidence in the results)
- NHST outcomes are prone to misinterpretation
  - Rejecting null hypothesis with confidence p does not signify that the performance-difference holds with probability 1-p
  - 1-p in fact denotes the likelihood of evidence from the experiment being correct if the difference indeed exists
- Given enough data, a statistically significant difference can always be shown

Issues with hypothesis testing: But NHST can still be helpful

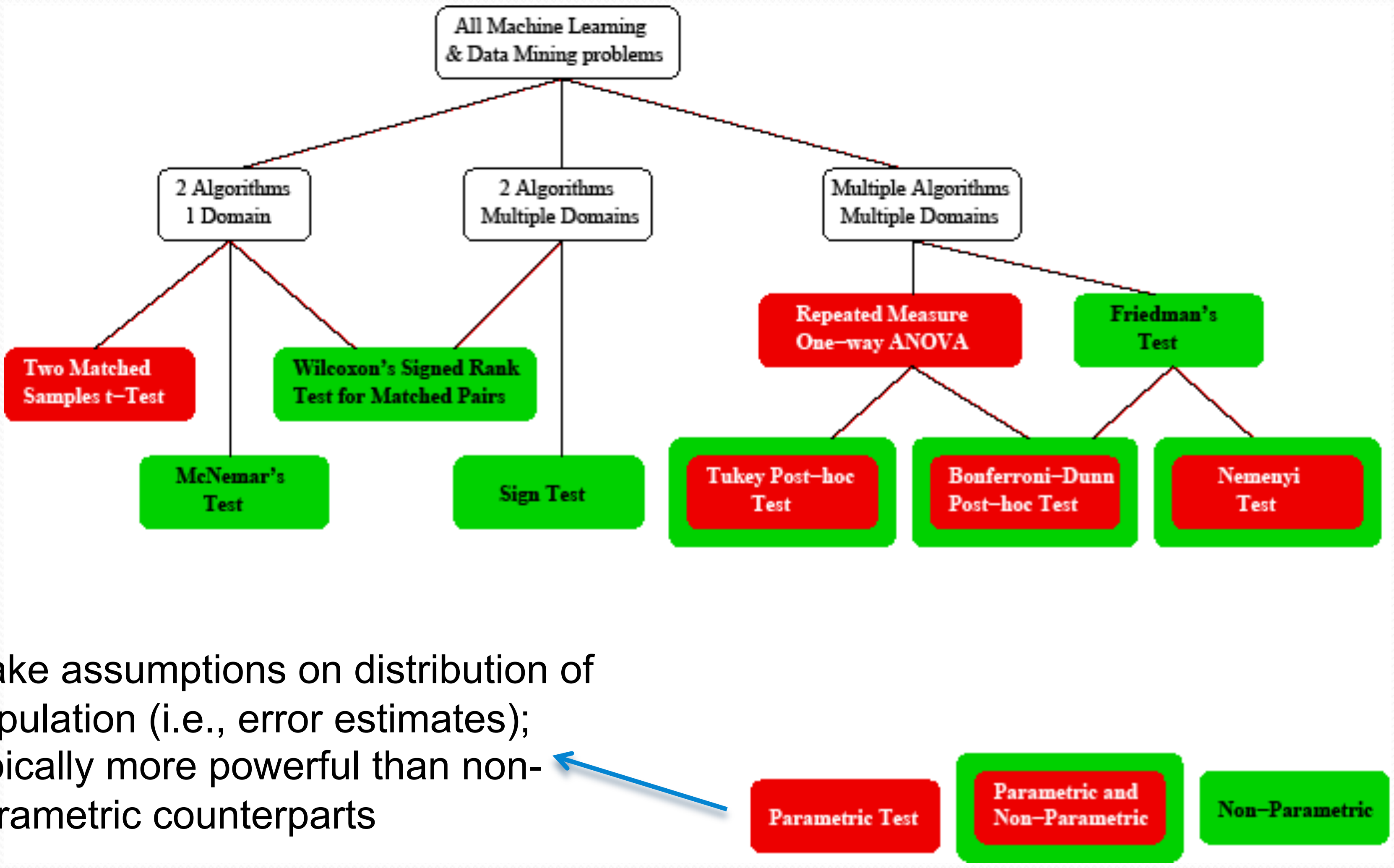
- NHST never constitutes a proof that our observation is valid (but provides added concrete support to observations)
- Some researchers (Drummond, 2006; Demsar, 2008) have pointed out that NHST:
  - overvalues the results; and
  - limits the search for novel ideas (owing to excessive, often unwarranted, confidence in the results)
- NHST outcomes are prone to misinterpretation (to which a better sensitization is necessary)
- Given enough data, a statistically significant difference can always be shown

The impossibility to reject the null hypothesis while using a reasonable effect size is telling

An overview of representative tests

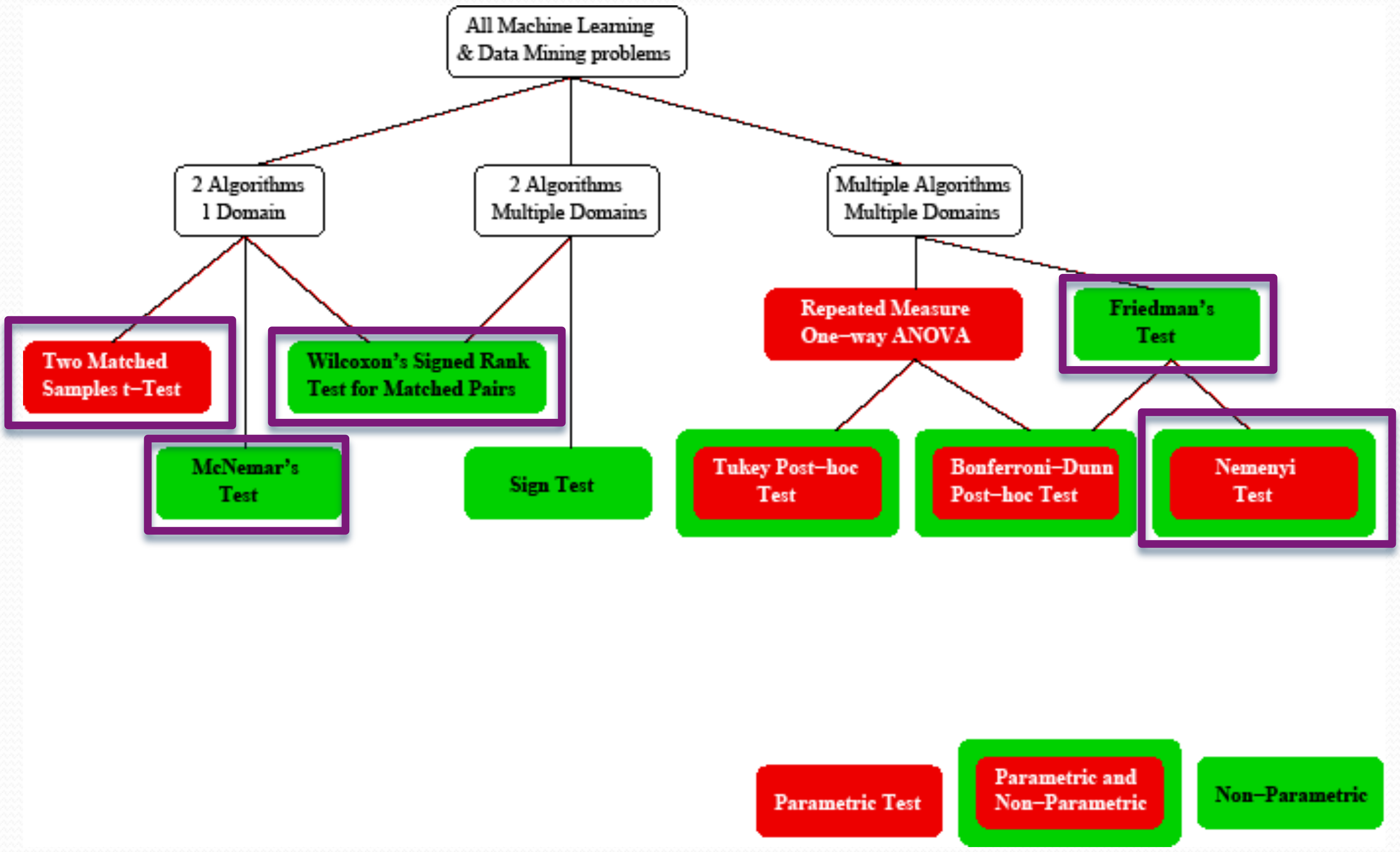

All Machine Learning & Data Mining problems
2 Algorithms 1 Domain
2 Algorithms Multiple Domains
Multiple Algorithms Multiple Domains
Two Matched Samples t–Test
McNemar's Test
Wilcoxon's Signed Rank Test for Matched Pairs
Sign Test
Repeated Measure One–way ANOVA
Friedman's Test
Tukey Post–hoc Test
Bonferroni–Dunn Post–hoc Test
Nemenyi Test
Parametric Test
Parametric and Non–Parametric
Non–Parametric

Parametric vs. Non-parametric tests


All Machine Learning & Data Mining problems
2 Algorithms 1 Domain
2 Algorithms Multiple Domains
Multiple Algorithms Multiple Domains
Two Matched Samples t–Test
McNemar's Test
Wilcoxon's Signed Rank Test for Matched Pairs
Sign Test
Repeated Measure One–way ANOVA
Friedman's Test
Tukey Post–hoc Test
Bonferroni–Dunn Post–hoc Test
Nemenyi Test
Parametric Test
Parametric and Non–Parametric
Non–Parametric

Make assumptions on distribution of population (i.e., error estimates); typically more powerful than non-parametric counterparts

Tests covered in this tutorial


All Machine Learning & Data Mining problems
2 Algorithms 1 Domain
2 Algorithms Multiple Domains
Multiple Algorithms Multiple Domains
Two Matched Samples t–Test
McNemar's Test
Wilcoxon's Signed Rank Test for Matched Pairs
Sign Test
Repeated Measure One–way ANOVA
Friedman's Test
Tukey Post–hoc Test
Bonferroni–Dunn Post–hoc Test
Nemenyi Test
Parametric Test
Parametric and Non–Parametric
Non–Parametric

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:

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

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: Illustration

- We apply C4.5 and Naïve Bayes (NB) on the Labor dataset (from UCI repository)
  - 10 runs of 10-fold CV (maintaining same training and testing folds across algorithms).
  - The result of each 10-fold CV is considered a single result. Hence, each run of 10-fold CV is a different trial. Consequently n=10
- We have:

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

- 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)

Effect size

- *t*-test measured the *effect* (i.e., the different in the sample means is indeed statistically significant) but not the *size* of this effect (i.e., the extent to which this difference is practically important)
- Statistics offers many methods to determine this effect size including Pearson's correlation coefficient, Hedges' G, etc.
- In the case of *t*-test, we use: Cohen's *d* statistic

$$d_{cohen} = \frac{\overline{pm}(f_1) - \overline{pm}(f_2)}{\sigma_p}$$

where

$$\sigma_p = \sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}}$$

Pooled standard deviation estimate

Effect size

- A typical interpretation proposed by Cohen for effect size was the following discrete scale:
  - A value of about 0.2 or 0.3 : small effect, but probably meaningful
  - ≈0.5: medium effect that is noticeable
  - ≥ 0.8: large effect

- In the case of Labor dataset example, we have:

$$d_{cohen} = \frac{0.2175 - 0.0649}{\sqrt{\frac{0.00261+0.00133}{2}}} = 3.4381$$

Comparing 2 algorithms on a single domain: *t*-test: so far so good; but what about Assumptions made by *t*-test?

Assumptions

- The Normality or pseudo-normality of populations
- The randomness of samples (representativeness)
- Equal Variances of the population

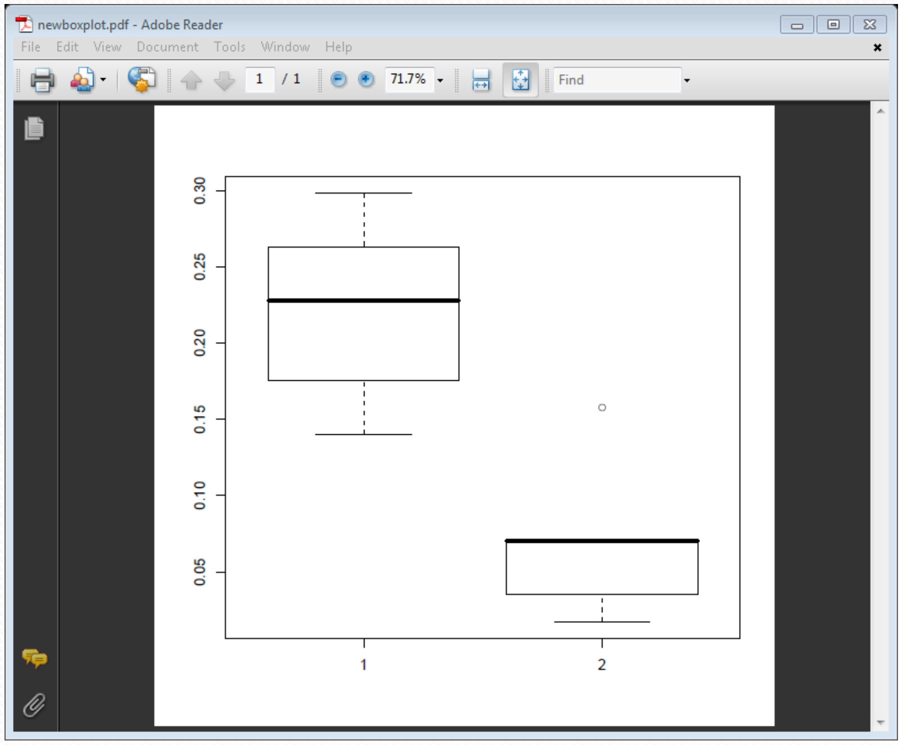
Let's see if these assumptions hold in our case of Labor dataset:

- Normality: we may assume pseudo-normality since each trial eventually tests the algorithms on each of 57 instances (note however, we compare averaged estimates)
- Randomness: Data was collected in 1987-1988. All the collective agreements were collected. No reason to assume this to be non-representative

Comparing 2 algorithms on a single domain: *t*-test: so far so good; but what about Assumptions made by *t*-test?

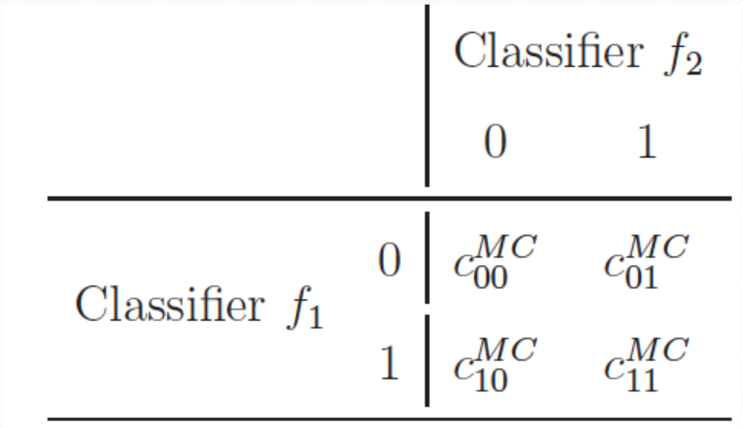
However,

- The sample variances cannot be considered equal
- We also ignored: correlated measurement effect (due to overlapping training sets) resulting in higher Type I error.
- Were we warranted to use *t*-test? Probably Not (even though t-test is quite robust over some assumptions).
  - Other variations of t-test (e.g., Welch's *t*-test)
  - **McNemar's test (non-parametric)**



Comparing 2 algorithms on a single domain: McNemar’s test

- Non-parametric counterpart to *t*-test
- Computers the following contingency matrix



		Classifier f_2 0	Classifier f_2 1
Classifier f_1	0	c_{00}^{MC}	c_{01}^{MC}
	1	c_{10}^{MC}	c_{11}^{MC}

- 0 here denotes misclassification by concerned classifier: i.e.,
  - c_{00} denotes the number of instances misclassified by both f_1 and f_2
  - C_{10} denotes the number of instances correctly classified by f_1 but not by f_2
  - …

Comparing 2 algorithms on a single domain: McNemar's test

- Null hypothesis

$$c_{01}^{Mc} = c_{10}^{Mc} = c_{null}^{Mc}$$

- McNemar's statistic

$$\chi^2_{McNemar} = \frac{(|c_{01}^{Mc} - c_{10}^{Mc}| - 1)^2}{c_{01}^{Mc} + c_{10}^{Mc}}$$

- distributed approximately as χ^2 if $c_{01}^{Mc} + c_{10}^{Mc}$ is greater than 20
- To reject the null hypothesis: $\chi^2_{McNemar}$ should be greater than $\chi^2_{1,1-\alpha}$ for desired α (typically 0.05)

Comparing 2 algorithms on a single domain: McNemar's test

- Back to the Labor dataset: comparing C4.5 and NB, we obtain

$$\chi^2_{McNemar} = \frac{(|11-2|-1)^2}{11+2} = 64/13 = 4.92$$

- Which is greater than the desired value of 3.841 allowing us to reject the null hypothesis at 95% confidence

Comparing 2 algorithms on a single domain: McNemar's test

To keep in mind:

- Test aimed for matched pairs (maintain same training and test folds across learning algorithms)
- Using resampling instead of independent test sets (over multiple runs) can introduce some bias
- A constraint: $c_{01}^{Mc} + c_{10}^{Mc}$ (the diagonal) should be at least 20 which was not true in this case.
    - In such a case, **sign test** (also applicable for comparing 2 algorithms on multiple domains) should be used instead

Comparing 2 algorithms on a multiple domains

- No clear parametric way to compare two classifiers on multiple domains:
  - *t*-test is not a very good alternative because it is not clear that we have commensurability of the performance measures across domains
  - The normality assumption is difficult to establish
  - The t-test is susceptible to outliers, which is more likely when many different domains are considered.
- Therefore we will describe two non-parametric alternatives
  - ➔ The Sign Test
  - ➔ Wilcoxon's signed-Rank Test

Comparing 2 algorithms on a multiple domains: Sign test

- Can be used to compare two classifiers on a single domain (using the results at each fold as a trial)
- More commonly used to compare two classifiers on multiple domains
- Calculate:
  - n_{f1}: the number of times that f1 outperforms f2
  - n_{f2}: the number of times that f2 outperforms f1

- Null hypothesis holds if the number of wins follow a binomial distribution.
- Practically speaking, a classifier should perform better on at least w_α (the critical value) datasets to reject null hypothesis at α significance level

Comparing 2 algorithms on a multiple domains: Sign test: Illustration

Dataset	NB	SVM	Adaboost	Rand Forest
Anneal	96.43	99.44	83.63	99.55
Audiology	73.42	81.34	46.46	79.15
Balance Scale	72.30	91.51	72.31	80.97
Breast Cancer	71.70	66.16	70.28	69.99
Contact Lenses	71.67	71.67	71.67	71.67
Pima Diabetes	74.36	77.08	74.35	74.88
Glass	70.63	62.21	44.91	79.87
Hepatitis	83.21	80.63	82.54	84.58
Hypothyroid	98.22	93.58	93.21	99.39
Tic-Tac-Toe	69.62	99.90	72.54	93.94

- NB vs SVM: n_{f1}= 4.5, n_{f2}= 5.5 and $w_{0.05}$ = 8 ➔ we cannot reject the null hypothesis (1-tailed)
- Ada vs RF: n_{f1}=1 and n_{f2}=8.5 ➔ we can reject the null hypothesis at level α=0.05 (1-tailed) and conclude that RF is significantly better than Ada on these data sets at that significance level.

Comparing 2 algorithms on a multiple domains: Wilcoxon's Signed-Rank test

- Non-parametric, and typically more powerful than sign test (since, unlike sign test, it quantifies the differences in performances)
- Commonly used to compare two classifiers on multiple domains

Method

- For each domain, calculate the difference in performance of the two classifiers
- Rank the absolute differences and graft the signs in front of these ranks
- Calculate the sum of positive ranks W_{s1} and sum of negative ranks W_{s2}
- Calculate $T_{wilcox} = \min(W_{s1}, W_{s2})$
- Reject null hypothesis if $T_{wilcox} \leq V_{\alpha}$ (critical value)

Comparing 2 algorithms on a multiple domains: Wilcoxon's Signed-Rank test: Illustration

Data	NB	SVM	NB-SVM	|NB-SVM|	Ranks	± Ranks
1	.9643	.9944	-0.0301	0.0301	3	-3
2	.7342	.8134	-0.0792	0.0792	6	-6
3	.7230	.9151	-0.1921	0.1921	8	-8
4	.7170	.6616	+0.0554	0.0554	5	+5
5	.7167	.7167	0	0	Remove	Remove
6	.7436	.7708	-0.0272	0.0272	2	-2
7	.7063	.6221	+0.0842	0.0842	7	+7
8	.8321	.8063	+0.0258	0.0258	1	+1
9	.9822	.9358	+0.0464	0.0464	4	+4
10	.6962	.9990	-0.3028	0.3028	9	-9

$W_{S1} = 17$ and $W_{S2} = 28$ ➔ $T_{\text{Wilcox}} = \min(17, 28) = 17$

For n= 10-1 degrees of freedom and α = 0.05, V = 8 for the 1-sided test.
Since 17 > 8. Hence, we cannot reject the null hypothesis

Comparing multiple algorithms on a multiple domains

- Two alternatives
  - Parametric: (one-way repeated measure) ANOVA ;
  - Non-parametric: Friedman's Test.
- Such multiple-hypothesis tests are called Omnibus tests.
  - Their null hypotheses: is that all the classifiers perform similarly, and its rejection signifies that: there exists at least one pair of classifiers with significantly different performances.
- In case of rejection of this null hypothesis, the omnibus test is followed by a Post-hoc test to identify the significantly different pairs of classifiers.
- In this tutorial, we will discuss Friedman's Test (omnibus) and the Nemenyi test (post-hoc test).

Friedman's Test

- Algorithms are ranked on each domain according to their performances (best performance = lowest rank)
  - In case of a d-way tie at rank r: assign ((r+1)+(r+2)+…+(r+d))/d to each classifier
- For each classifier j, compute $R_{.j}$: the sum of its ranks on all domains
- Calculate Friedman's statistic:

$$\chi_F^2 = \left[\frac{12}{n \times k \times (k+1)} \times \sum_{j=1}^{k} (R_{.j})^2\right] - 3 \times n \times (k+1)$$

over n domains and k classifiers

Illustration of the Friedman test

Domain	Classifier fA	Classifier fB	Classifier fC
1	85.83	75.86	84.19
2	85.91	73.18	85.90
3	86.12	69.08	83.83
4	85.82	74.05	85.11
5	86.28	74.71	86.38
6	86.42	65.90	81.20
7	85.91	76.25	86.38
8	86.10	75.10	86.75
9	85.95	70.50	88.03
19	86.12	73.95	87.18

Domain	Classifier fA	Classifier fB	Classifier fC
1	1	3	2
2	1.5	3	1.5
3	1	3	2
4	1	3	2
5	2	3	1
6	1	3	2
7	2	3	1
8	2	3	1
9	2	3	1
10	2	3	1
$R_{.j}$	15.5	30	14.5

$\chi_F^2 = [\frac{12}{10 \times 3 \times (3+1)} \times \sum_{j=1}^{3} (R_{.j})^2] - 3 \times 10 \times (3+1) = 15.05$ For a 2-tailed test at the 0.05 level of significance, the critical value is 7.8. $\chi_F^2 > 7.8$, i.e., Rejection of the NH.

Nemenyi Test

- The omnibus test rejected the null hypothesis
- Follow up by post-hoc test: Nemenyi test to identify classifier-pairs with significant performance difference
- Let R_{ij} : rank of classifier f_j on data S_i; Compute mean rank of f_j on all datasets

$$\overline{R}_{.j} = \frac{1}{n}\sum_{i=1}^{n} R_{ij}$$

- Between classifier f_{j1} and f_{j2}, calculate

$$q = \frac{\overline{R}_{.j_1} - \overline{R}_{.j_2}}{\sqrt{\frac{k(k+1)}{6n}}}$$

Nemenyi Test: Illustration

- Computing the q statistic of Nemenyi test for different classifier pairs, we obtain:
  - $q_{AB} = -32.22$
  - $q_{BC} = 34.44$
  - $q_{AC} = 2.22$
- At $\alpha = 0.05$, we have $q_\alpha = 2.55$.
- Hence, we can show a statistically significantly different performance between classifiers A and B, and between classifiers B and C, but not between A and C

Data Set Selection and Evaluation Benchmark Design

Considerations to keep in mind while choosing an appropriate test bed

- Wolpert's "No Free Lunch" Theorems: if one algorithm tends to perform better than another on a given class of problems, then the reverse will be true on a different class of problems.

- LaLoudouana and Tarate (2003) showed that even mediocre learning approaches can be shown to be competitive by selecting the test domains carefully.

➔The purpose of data set selection should not be to demonstrate an algorithm's superiority to another in all cases, but rather to identify the areas of strengths of various algorithms with respect to domain characteristics or on specific domains of interest.

Where can we get our data from?

- Repository Data: Data Repositories such as the UCI repository and the UCI KDD Archive have been extremely popular in Machine Learning Research.
- Artificial Data: The creation of artificial data sets representing the particular characteristic an algorithm was designed to deal with is also a common practice in Machine Learning
- Web-Based Exchanges: Could we imagine a multi-disciplinary effort conducted on the Web where researchers in need of data analysis would "lend" their data to machine learning researchers in exchange for an analysis?

Pros and Cons of Repository Data

- Pros:
  - Very easy to use: the data is available, already processed and the user does not need any knowledge of the underlying field.
  - The data is not artificial since it was provided by labs and so on. So in some sense, the research is conducted in real-world setting (albeit a limited one)
  - Replication and comparisons are facilitated, since many researchers use the same data set to validate their ideas.
- Cons:
  - The use of data repositories does not guarantee that the results will generalize to other domains.
  - The data sets in the repository are not representative of the data mining process which involves many steps other than classification.
  - Community experiment/Multiplicity effect: since so many experiments are run on the same data set, by chance, some will yield interesting (though meaningless) results

Pros and Cons of Artificial Data

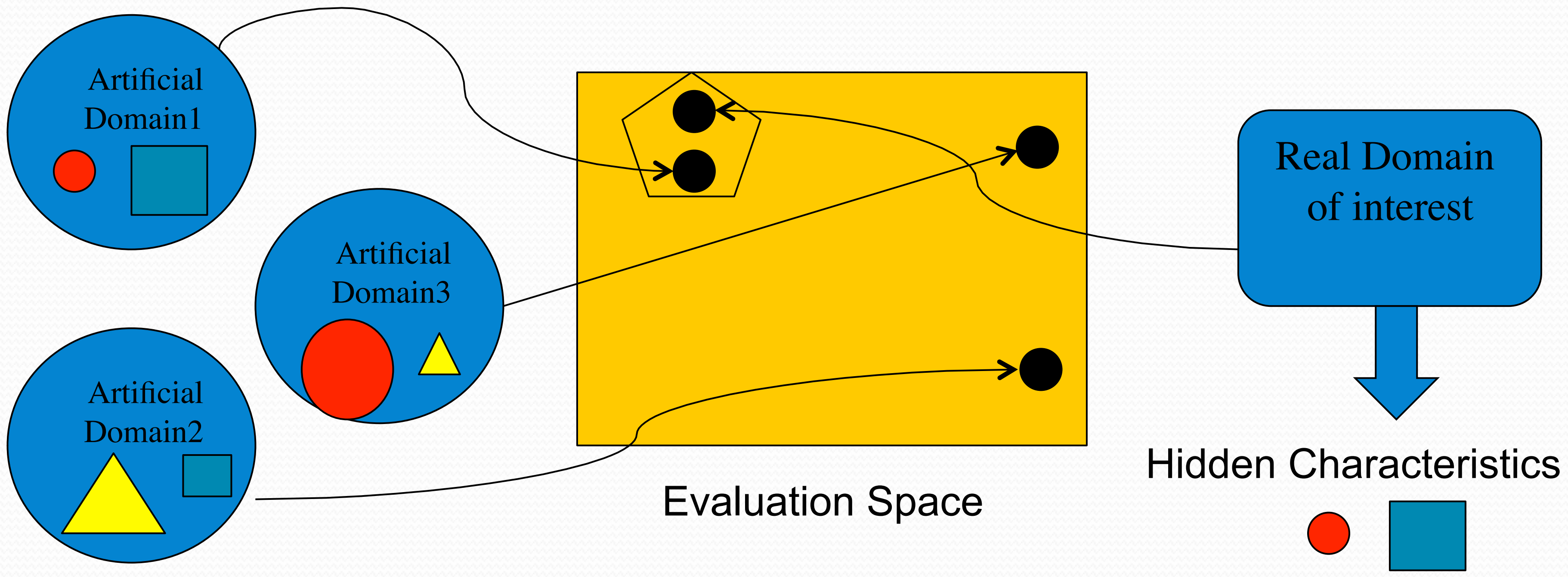
- Pros:
  - Data sets can be created to mimic the traits that are expected to be present in real data (which are unfortunately unavailable)
  - The researcher has the freedom to explore various related situations that may not have been readily testable from accessible data.
  - Since new data can be generated at will, the multiplicity effect will not be a problem in this setting.
- Cons:
  - Real data is unpredictable and does not systematically behave according to a well defined generation model.
  - Classifiers that happen to model the same distribution as the one used in the data generation process have an unfair advantage.

Pros and Cons of Web-Based Exchanges

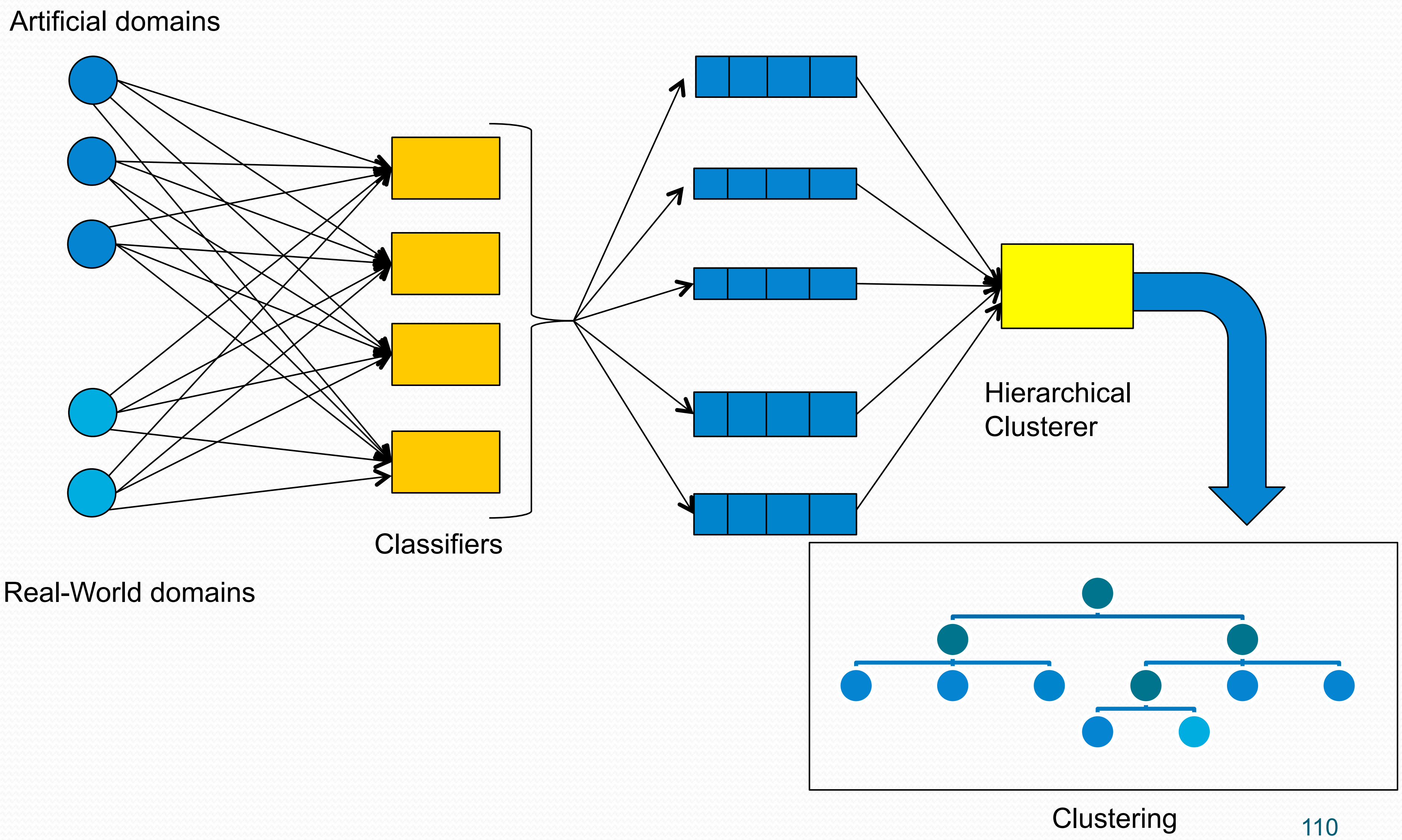
- Pros:
  - Would be useful for three communities:
    - Domain experts, who would get interesting analyses of their problem
    - Machine Learning researchers who would be getting their hand on interesting data, and thus encounter new problems
    - Statisticians, who could be studying their techniques in an applied context.
- Cons:
  - It has not been organized. Is it truly feasible?
  - How would the quality of the studies be controlled?

A Technique for Characterizing our Data Sets
[Japkowicz, Canadian AI'2012]

- The approach can be characterized as a meta-level transductive process driven by classifier evaluation:

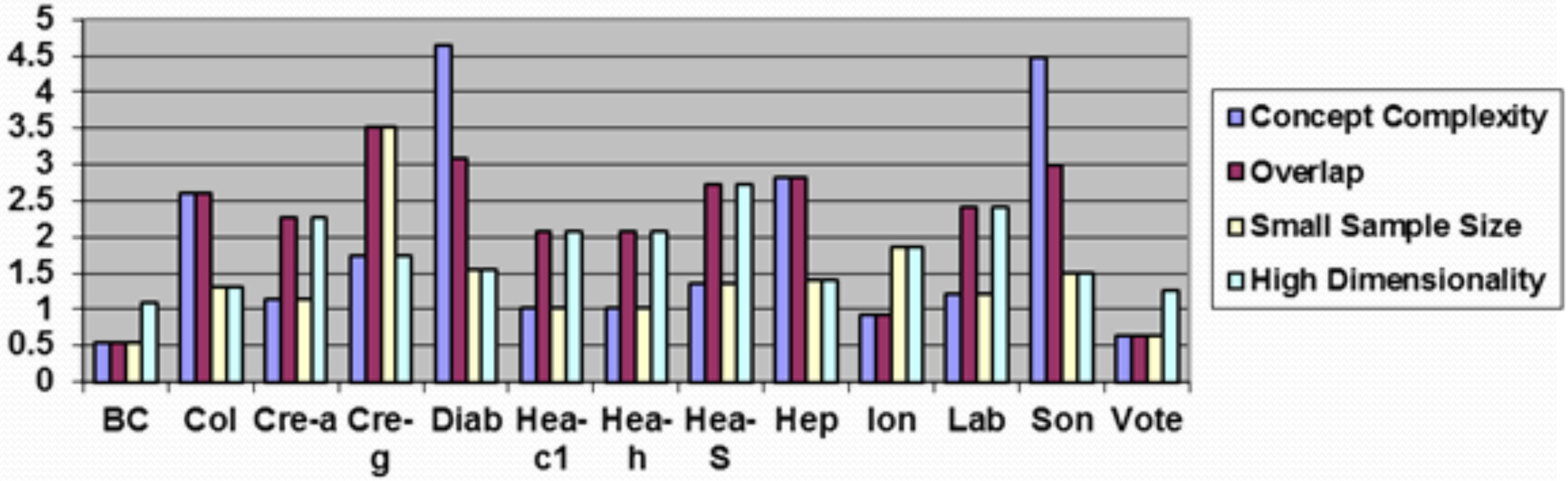


Evaluation Space Mapping



Results

The graph below shows the hidden characteristics that we have extracted from each of the UCI domains we considered, along with the relative strength of these characteristics (which were derived from the distance of each domain mapping to the ideal domain in the evaluation space by way of the hierarchical clustering obtained for each domain)



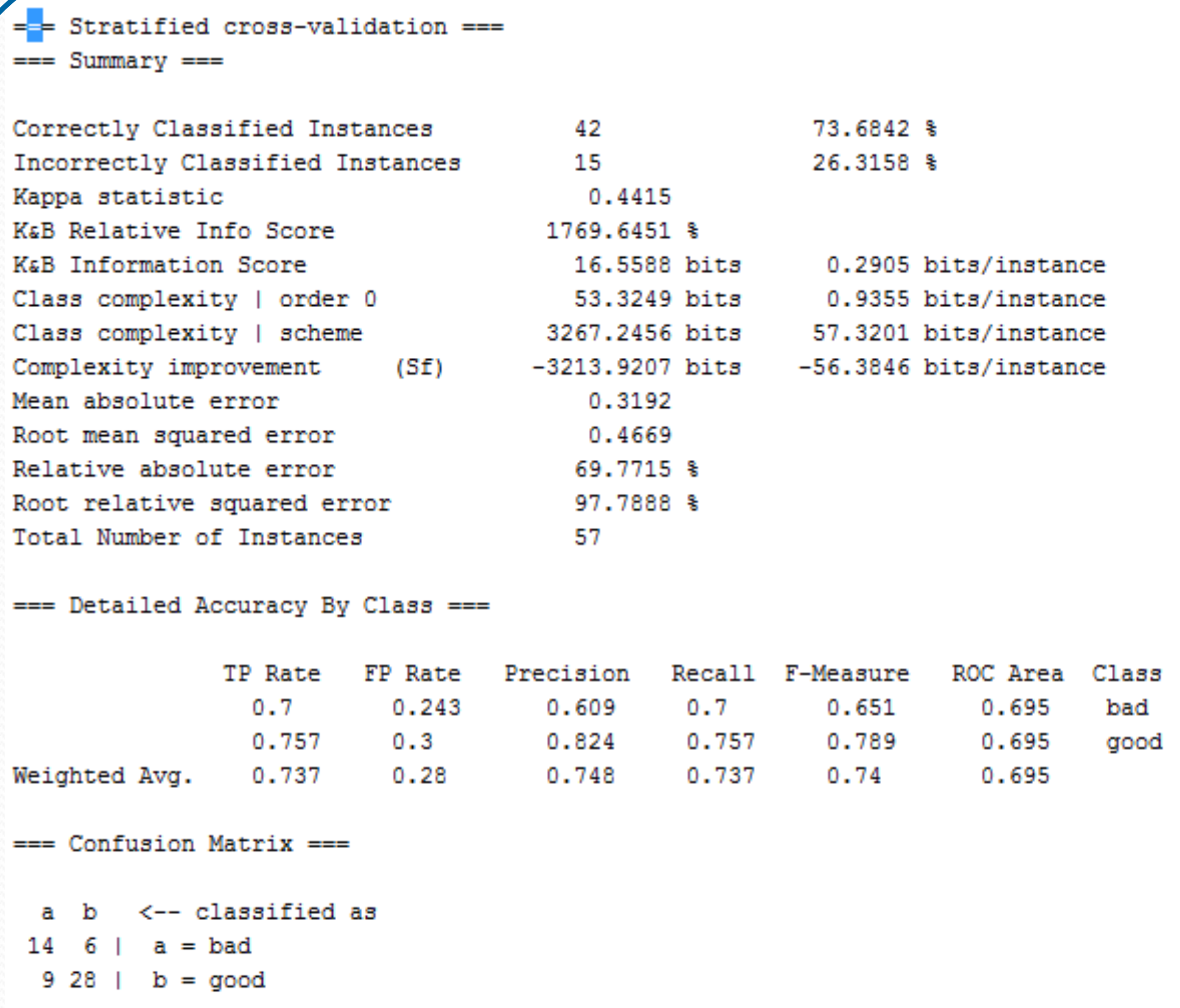
Available Resources

What help is available for conducting proper evaluation?

- There is no need for researchers to program all the code necessary to conduct proper evaluation nor to do the computations by hand.
- Resources are readily available for every steps of the process, including:
  - Evaluation metrics
  - Statistical tests
  - Re-sampling
  - And of course, Data repositories
- Pointers to these resources and explanations on how to use them are discussed in our book: <*"Evaluating Learning Algorithms: A Classification Perspective"* by Japkowicz and Shah, Cambridge University Press, 2011>.

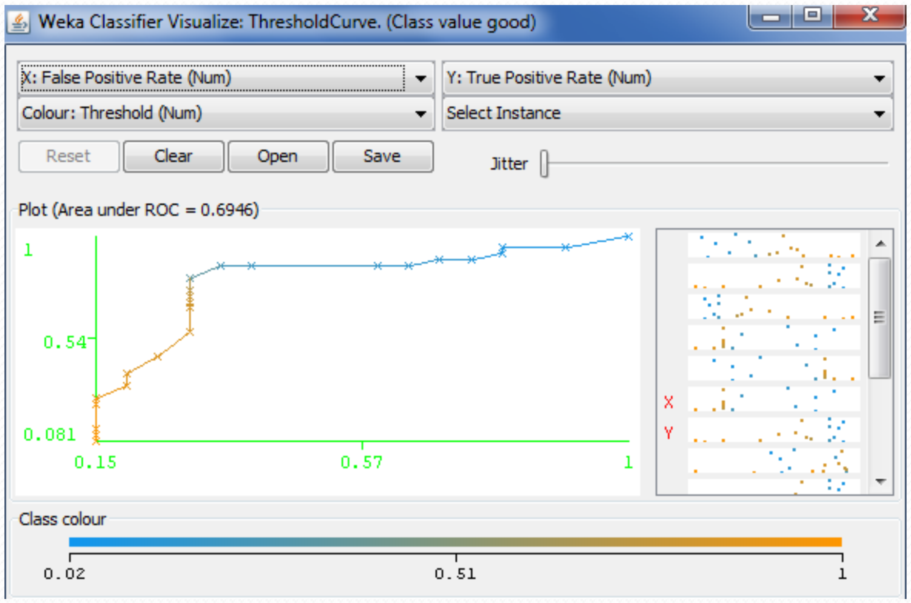
Where to look for evaluation metrics?

Actually, as most people know, Weka is a great source for, not only, classifiers, but also computation of the results according to a variety of evaluation metrics

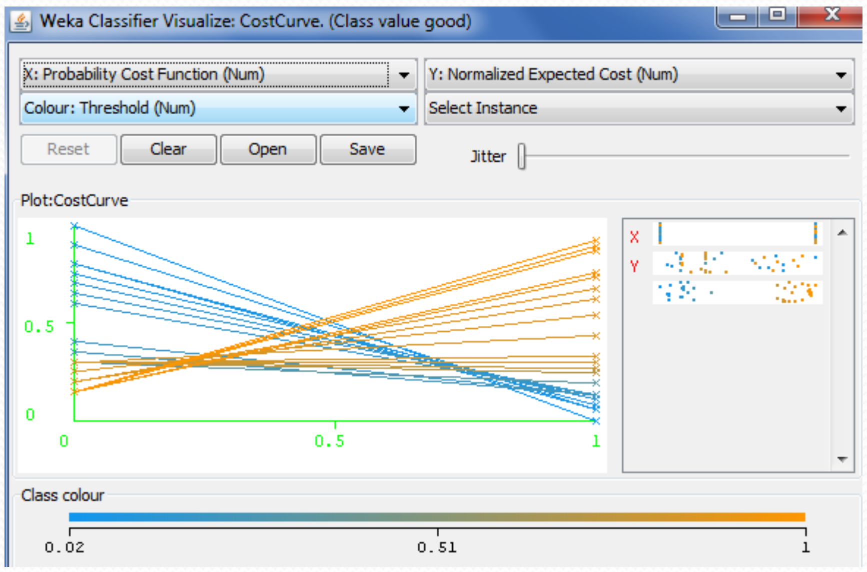


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          42               73.6842 %
Incorrectly Classified Instances        15               26.3158 %
Kappa statistic                          0.4415
K&B Relative Info Score               1769.6451 %
K&B Information Score                   16.5588 bits      0.2905 bits/instance
Class complexity | order 0              53.3249 bits      0.9355 bits/instance
Class complexity | scheme             3267.2456 bits     57.3201 bits/instance
Complexity improvement     (Sf)      -3213.9207 bits    -56.3846 bits/instance
Mean absolute error                      0.3192
Root mean squared error                  0.4669
Relative absolute error                 69.7715 %
Root relative squared error             97.7888 %
Total Number of Instances               57

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.7       0.243      0.609     0.7       0.651      0.695    bad
                 0.757     0.3        0.824     0.757     0.789      0.695    good
Weighted Avg.    0.737     0.28       0.748     0.737     0.74       0.695

=== Confusion Matrix ===

  a  b   <-- classified as
 14  6 |  a = bad
  9 28 |  b = good
```

WEKA even performs ROC Analysis and draws Cost-Curves


Weka Classifier Visualize: ThresholdCurve. (Class value good)
Reset
Clear
Open
Save
Jitter
Class colour
0.02
0.51
1

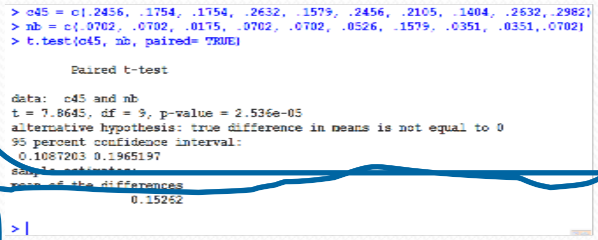
X: False Positive Rate (Num)
Y: True Positive Rate (Num)
Colour: Threshold (Num)
Select Instance
Plot (Area under ROC = 0.6946)



X: Probability Cost Function (Num)
Y: Normalized Expected Cost (Num)
Colour: Threshold (Num)
Select Instance
Plot:CostCurve

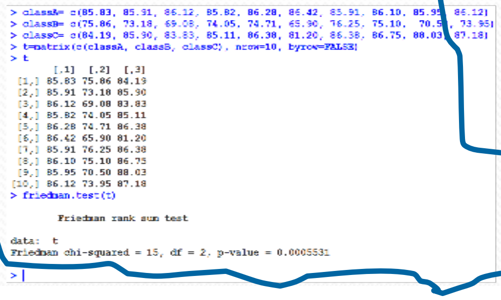
Although better graphical analysis packages are available in R, namely the ROCR package, which permits the visualization of many versions of ROC and Cost Curves, and also Lift curves, P-R Curves, and so on

Where to look for Statistical Tests?

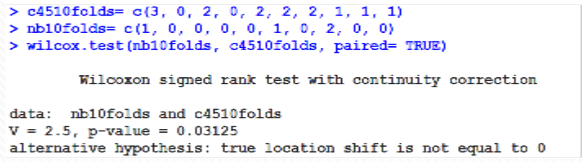


```
> c45 = c(.2456, .1754, .1754, .2632, .1579, .2456, .2105, .1404, .2632,.2982)
> nb = c(.0702, .0702, .0175, .0702, .0702, .0526, .1579, .0351, .0351,.0702)
> t.test(c45, nb, paired= TRUE)

        Paired t-test

data:  c45 and nb
t = 7.8645, df = 9, p-value = 2.536e-05
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.1087203 0.1965197
sample estimates:
mean of the differences
              0.15262

> |
```

The R Software Package contains implementations of all the statistical tests discussed here and many more. They are very simple to run.



```
> classA= c(85.83, 85.91, 86.12, 85.82, 86.28, 86.42, 85.91, 86.10, 85.95, 86.12)
> classB= c(75.86, 73.18, 69.08, 74.05, 74.71, 65.90, 76.25, 75.10,  70.50, 73.95)
> classC= c(84.19, 85.90, 83.83, 85.11, 86.38, 81.20, 86.38, 86.75, 88.03, 87.18)
> t=matrix(c(classA, classB, classC), nrow=10, byrow=FALSE)
> t
      [,1]  [,2]  [,3]
 [1,] 85.83 75.86 84.19
 [2,] 85.91 73.18 85.90
 [3,] 86.12 69.08 83.83
 [4,] 85.82 74.05 85.11
 [5,] 86.28 74.71 86.38
 [6,] 86.42 65.90 81.20
 [7,] 85.91 76.25 86.38
 [8,] 86.10 75.10 86.75
 [9,] 85.95 70.50 88.03
[10,] 86.12 73.95 87.18
> friedman.test(t)

        Friedman rank sum test

data:  t
Friedman chi-squared = 15, df = 2, p-value = 0.0005531

> |
```



```
> c4510folds= c(3, 0, 2, 0, 2, 2, 2, 1, 1, 1)
> nb10folds= c(1, 0, 0, 0, 0, 1, 0, 2, 0, 0)
> wilcox.test(nb10folds, c4510folds, paired= TRUE)

        Wilcoxon signed rank test with continuity correction

data:  nb10folds and c4510folds
V = 2.5, p-value = 0.03125
alternative hypothesis: true location shift is not equal to 0
```

Where to look for Re-sampling methods?

- In our book, we have implemented all the re-sampling schemes described in this tutorial. The actual code is also available upon request. (e-mail us!)

```
eoBoot = function(iter, dataSet, setSize, dimension, classifier1, classifier2){
 classifier1eoBoot <- numeric(iter)
classifier2eoBoot <- numeric(iter)
 for(i in 1:iter) {
 Subsamp <-  sample(setSize, setSize, replace=TRUE)
 Basesamp <- 1:setSize
  oneTrain <- dataSet[Subsamp ,1:dimension ]
 oneTest <- dataSet[setdiff(Basesamp,Subsamp), 1:dimension]
 classifier1model <- classifier1(class~., data=oneTrain)
 classifier2model <- classifier2(class~., data=oneTrain)
    classifier1eval <- evaluate_Weka_classifier(classifier1model, newdata=oneTest)
  classifier1acc <- as.numeric(substr(classifier1eval$string, 70,80))
  classifier2eval <- evaluate_Weka_classifier(classifier2model,
                   newdata=oneTest)
    classifier2acc <- as.numeric(substr(classifier2eval$string, 70,80))
 classifier1eoBoot[i]= classifier1acc
 classifier2eoBoot[i]= classifier2acc
 }
return(rbind(classifier1eoBoot, classifier2eoBoot))}
```

The End

Some Concluding Remarks

If you need help, advice, etc…

- Please, contact us at:

  nat@site.uottawa.ca or

  mohak@mohakshah.com

- Thank you for attending the tutorial!!!

References

- Too many to put down here, but pp. 393-402 of the book.
- If you want specific ones, come and see us at the end of the tutorial or send us an e-mail.