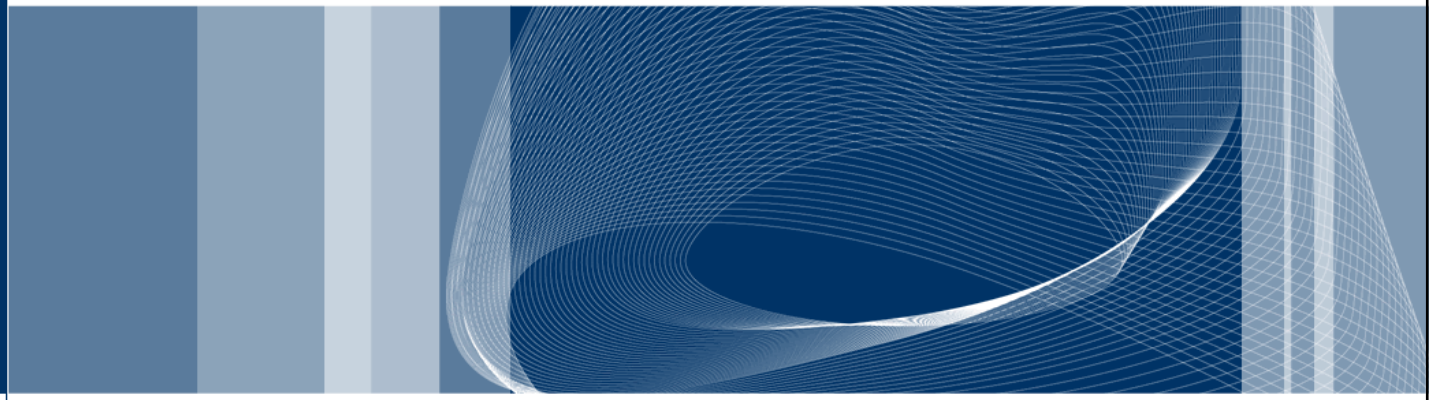
POLITECNICO DI MILANO



Credibility: Robust Evaluation of Models

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

Prof. Pier Luca Lanzi

Evaluation & Credidibility Issues

- What measure should we use?
- How reliable are the predicted results?
- How much should we believe in what was learned?
  - Error on the training data is not a good indicator of performance on future data
  - The classifier was computed from the very same training data, any estimate based on that data will be optimistic.
  - In addition, new data will probably not be exactly the same as the training data!

How to evaluate the performance of a model?

How to obtain reliable estimates?

How to compare the relative performance among competing models?

Given two equally performing models, which one should we prefer?

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the performance of competing models?
- Model Selection
  - Which model should we prefer?

How to evaluate the performance of a model? (Metrics for Performance Evaluation)

Metrics for Evaluation of Numerical Prediction

- Residual Sum of Squares (RSS)

$$RSS(\vec{w}) = \sum_{I=1}^{N} \left(y_i - \sum_{j=0}^{D} w_j h_j(\vec{x_i}) \right)^2$$

- R^2 (coefficient of determination)

$$TSS = \sum_{i=1}^{N} (y_i - \overline{y})^2 \qquad R^2 = 1 - \frac{RSS}{TSS}$$

- R^2 (coefficient of determination – second equation)

$$R_2^2 = \sum_{i=1}^{N} (\hat{y}_i - \overline{y})^2 / \sum_{i=1}^{N} (y_i - \overline{y})^2$$

Metrics for Evaluation of Numerical Prediction

- Mean Square Error

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$

- Root Mean Square Error

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

Classification

Metrics for Performance Evaluation: Confusion Matrix

- Focus on the predictive capability of a model

- Confusion Matrix:

		PREDICTED CLASS	
		Yes	No
TRUE CLASS	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation: Accuracy

		PREDICTED CLASS	
		Yes	No
ACTUAL CLASS	Yes	a (TP)	b (FN)
	No	c (FP)	d (TN)

- Most widely-used metric:

$$Accuracy = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Sometimes Accuracy is not Enough

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %

- Accuracy is misleading because model does not detect any class 1 example

Metrics for Performance Evaluation: Cost

		PREDICTED CLASS	
		Yes	No
ACTUAL CLASS	Yes	C(TP)	C(FN)
	No	C(FP)	C(TN)

C(x): Cost of misclassifying examples of type x

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(.)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

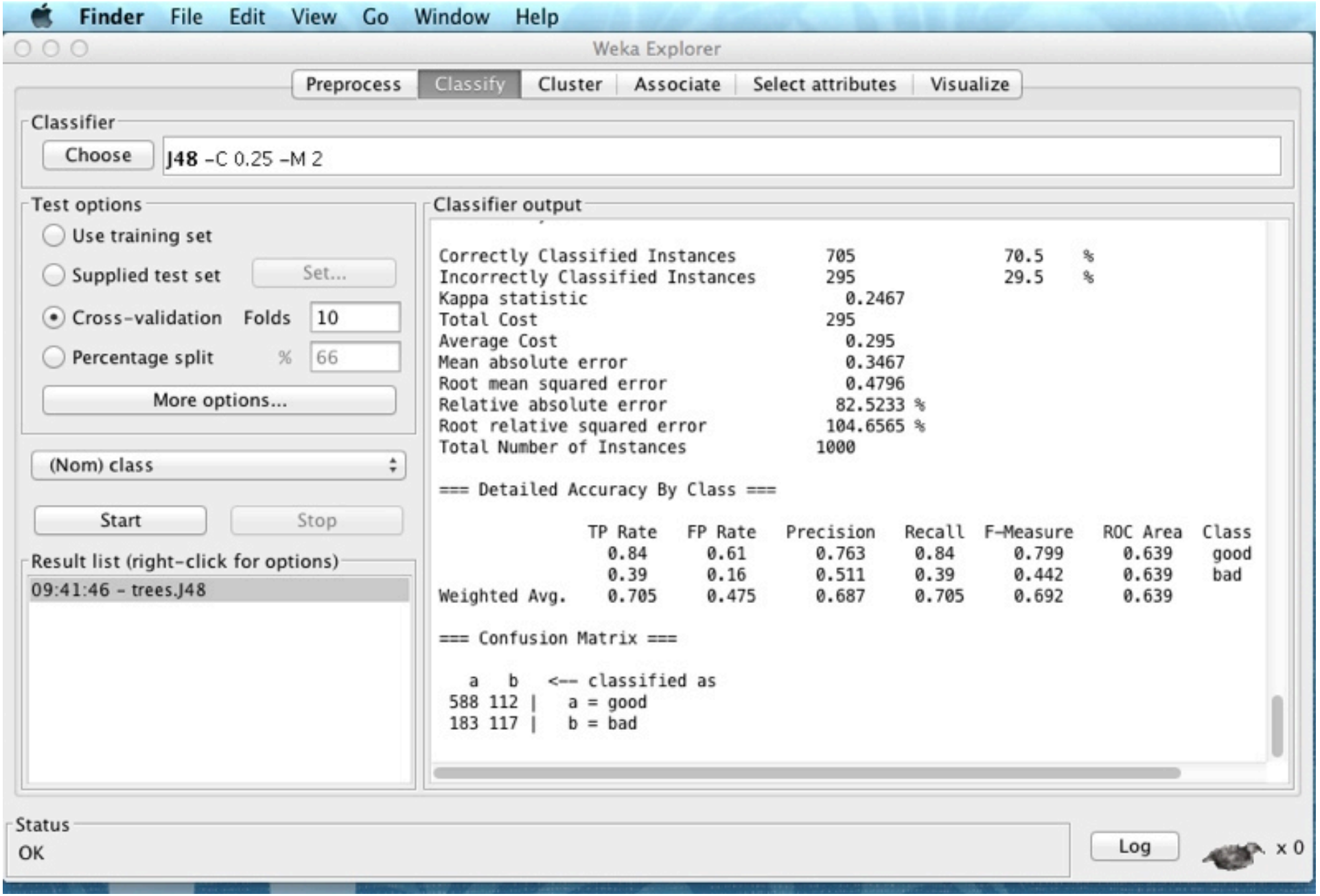
Accuracy = 90%

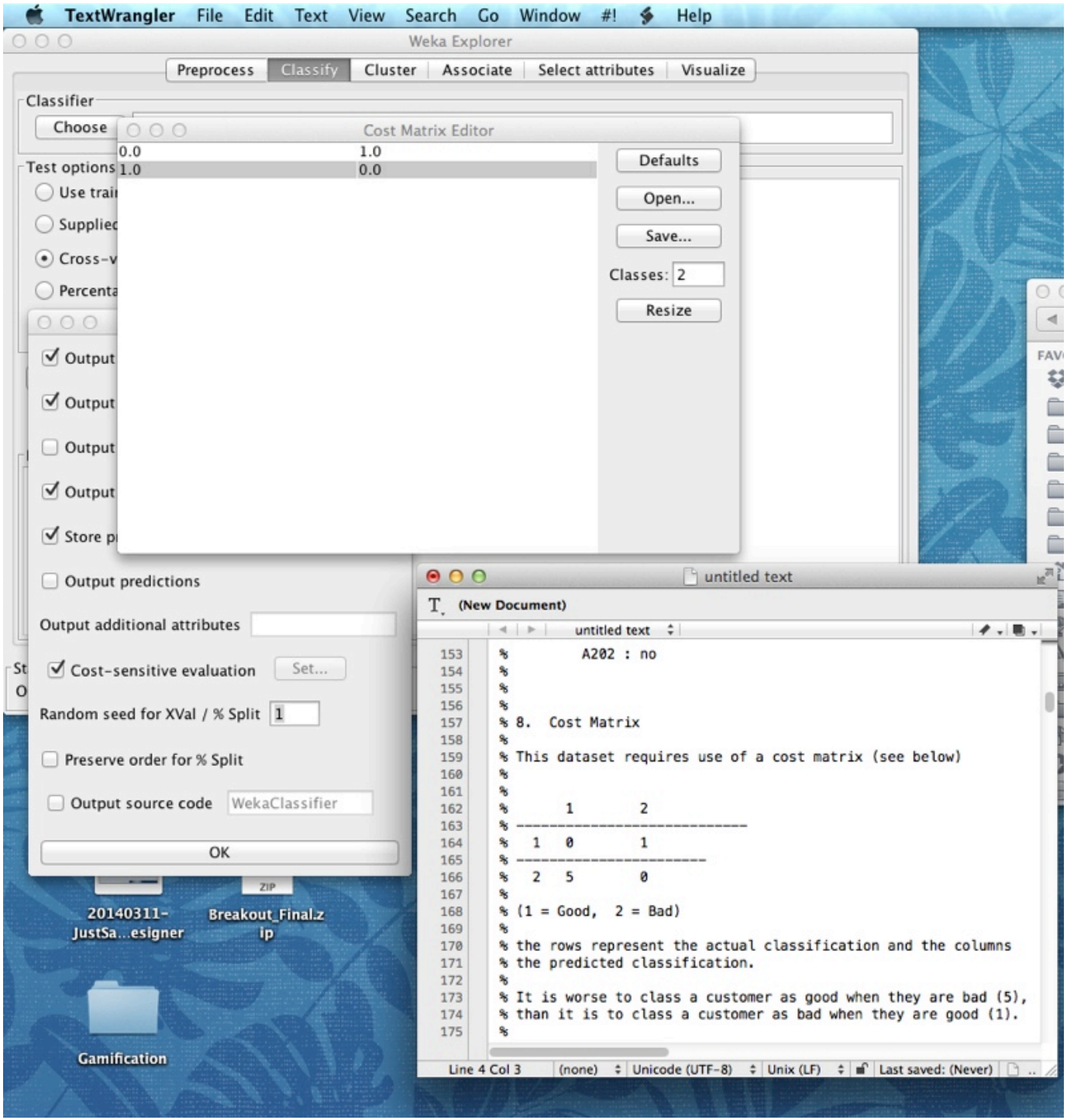
Cost = 4255

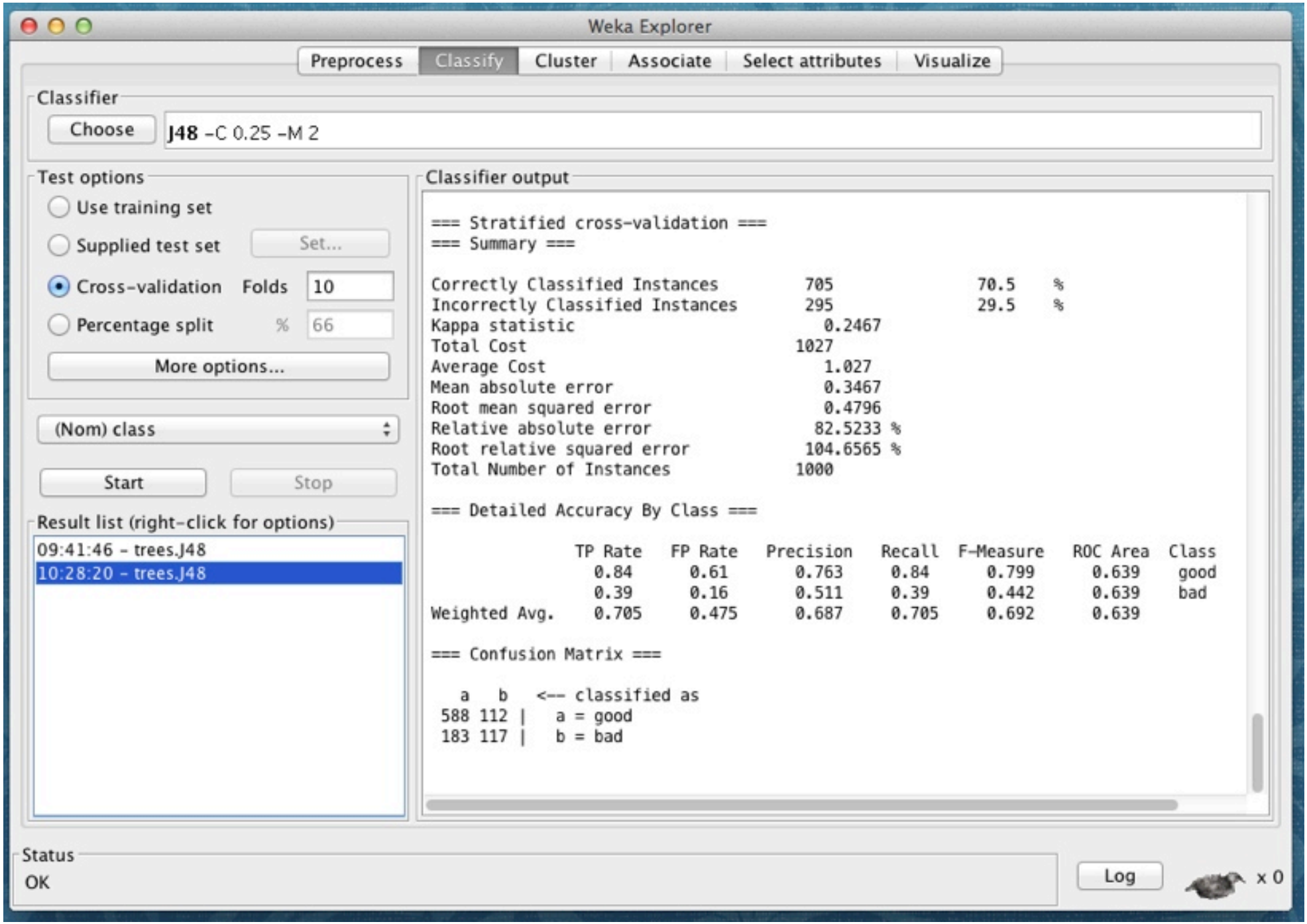
Costs can be used to evaluate
an existing classification models

Or can be used by the algorithm
to guide the search for the model

Some algorithms can use the cost matrix to
build the model (e.g., decision trees)


Finder File Edit View Go Window Help
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation Folds 10
Percentage split % 66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
09:41:46 - trees.J48
Classifier output
Correctly Classified Instances 705 70.5 %
Incorrectly Classified Instances 295 29.5 %
Kappa statistic 0.2467
Total Cost 295
Average Cost 0.295
Mean absolute error 0.3467
Root mean squared error 0.4796
Relative absolute error 82.5233 %
Root relative squared error 104.6565 %
Total Number of Instances 1000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
0.84 0.61 0.763 0.84 0.799 0.639 good
0.39 0.16 0.511 0.39 0.442 0.639 bad
Weighted Avg. 0.705 0.475 0.687 0.705 0.692 0.639
=== Confusion Matrix ===
a b <-- classified as
588 112 | a = good
183 117 | b = bad
Status
OK
Log
x 0


TextWrangler File Edit Text View Search Go Window #! Help
Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Classifier
Choose
Test options
Use trai
Supplied
Cross-v
Percenta
Cost Matrix Editor
0.0 1.0
1.0 0.0
Defaults
Open...
Save...
Classes: 2
Resize
Output
Output
Output
Output
Store p
Output predictions
Output additional attributes
Cost-sensitive evaluation Set...
Random seed for XVal / % Split 1
Preserve order for % Split
Output source code WekaClassifier
OK
untitled text
(New Document)
untitled text
153 % A202 : no
154 %
155 %
156 %
157 % 8. Cost Matrix
158 %
159 % This dataset requires use of a cost matrix (see below)
160 %
161 %
162 % 1 2
163 % ------------------------
164 % 1 0 1
165 % ----------------------
166 % 2 5 0
167 %
168 % (1 = Good, 2 = Bad)
169 %
170 % the rows represent the actual classification and the columns
171 % the predicted classification.
172 %
173 % It is worse to class a customer as good when they are bad (5),
174 % than it is to class a customer as bad when they are good (1).
175 %
Line 4 Col 3 (none) Unicode (UTF-8) Unix (LF) Last saved: (Never)
20140311-JustSa...esigner
Breakout_Final.zip
Gamification


Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
J48 -C 0.25 -M 2
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
09:41:46 - trees.J48
10:28:20 - trees.J48
Classifier output
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 705 70.5 %
Incorrectly Classified Instances 295 29.5 %
Kappa statistic 0.2467
Total Cost 1027
Average Cost 1.027
Mean absolute error 0.3467
Root mean squared error 0.4796
Relative absolute error 82.5233 %
Root relative squared error 104.6565 %
Total Number of Instances 1000
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure ROC Area Class
0.84 0.61 0.763 0.84 0.799 0.639 good
0.39 0.16 0.511 0.39 0.442 0.639 bad
Weighted Avg. 0.705 0.475 0.687 0.705 0.692 0.639
=== Confusion Matrix ===
a b <-- classified as
588 112 | a = good
183 117 | b = bad
Status
OK
Log
x 0

Precision & Recall

- Alternatives to accuracy, introduced in the area of information retrieval and search engine
- Precision
  - Focuses on the percentage of examples that have been classified positive examples and are actually positive
  - In the information retrieval context represents the percentage of actually good documents that have been shown as a result.
- Recall
  - Focuses on the percentage of positively classified examples with respect to the number of existing good documents
  - In the information retrieval context, recall represents the percentage of good documents shown with respect to the existing ones.

Cost-Sensitive Measures

$$Precision(p) = \frac{TP}{TP+FP} = \frac{a}{a+c}$$

The higher the precision, the lower the FPs

$$Recall(r) = \frac{TP}{TP+FN} = \frac{a}{a+b}$$

The higher the precision, the lower the FNs

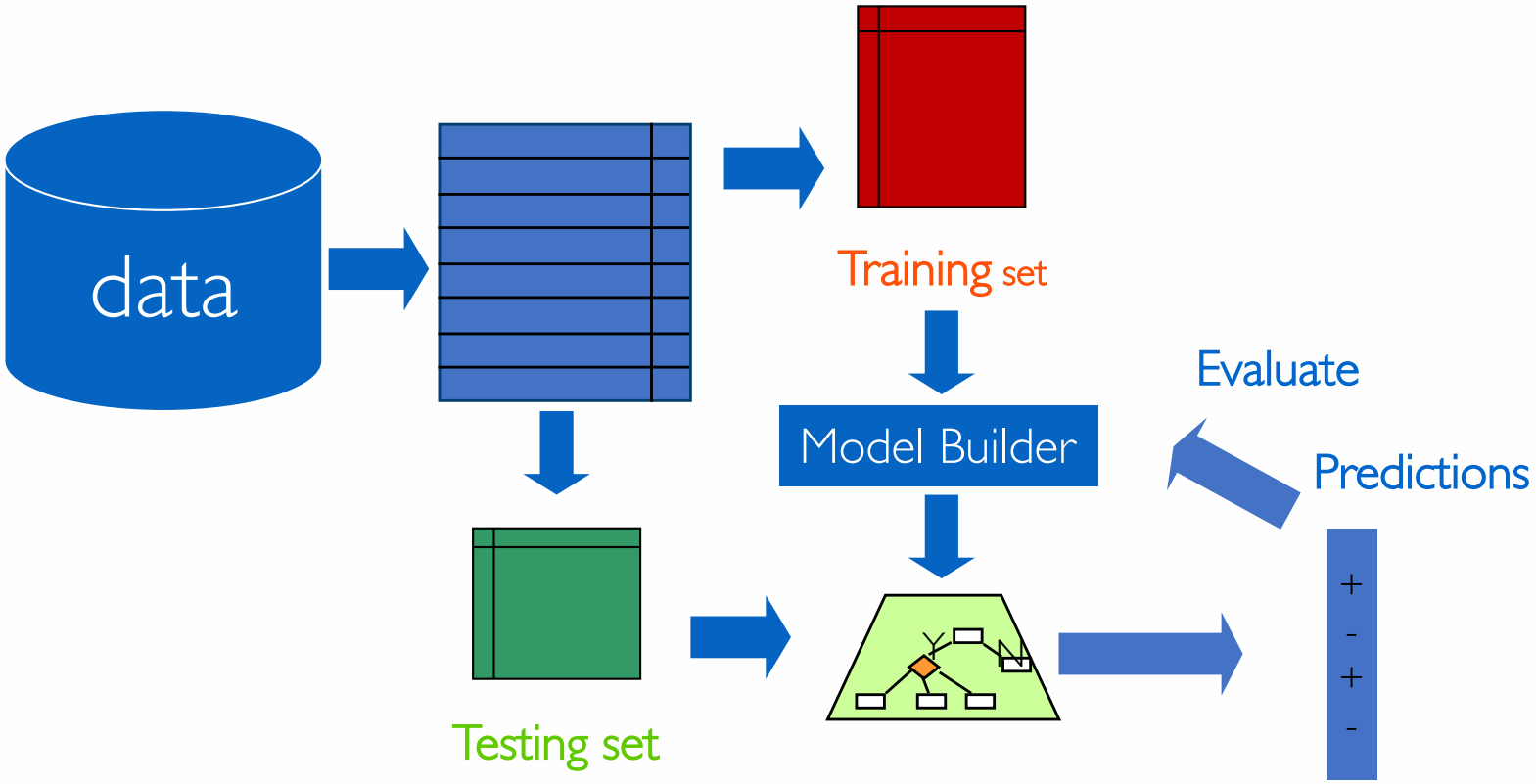
$$F1-measure = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

The higher the F1, the lower the FPs & FNs

- Precision is biased towards TP) & FP
- Recall is biased towards TP & FN
- F1-measure is biased towards all except TN
  it is high when both precision and recall are reasonably high

How to obtain reliable estimates?
(Methods for Performance Evaluation)

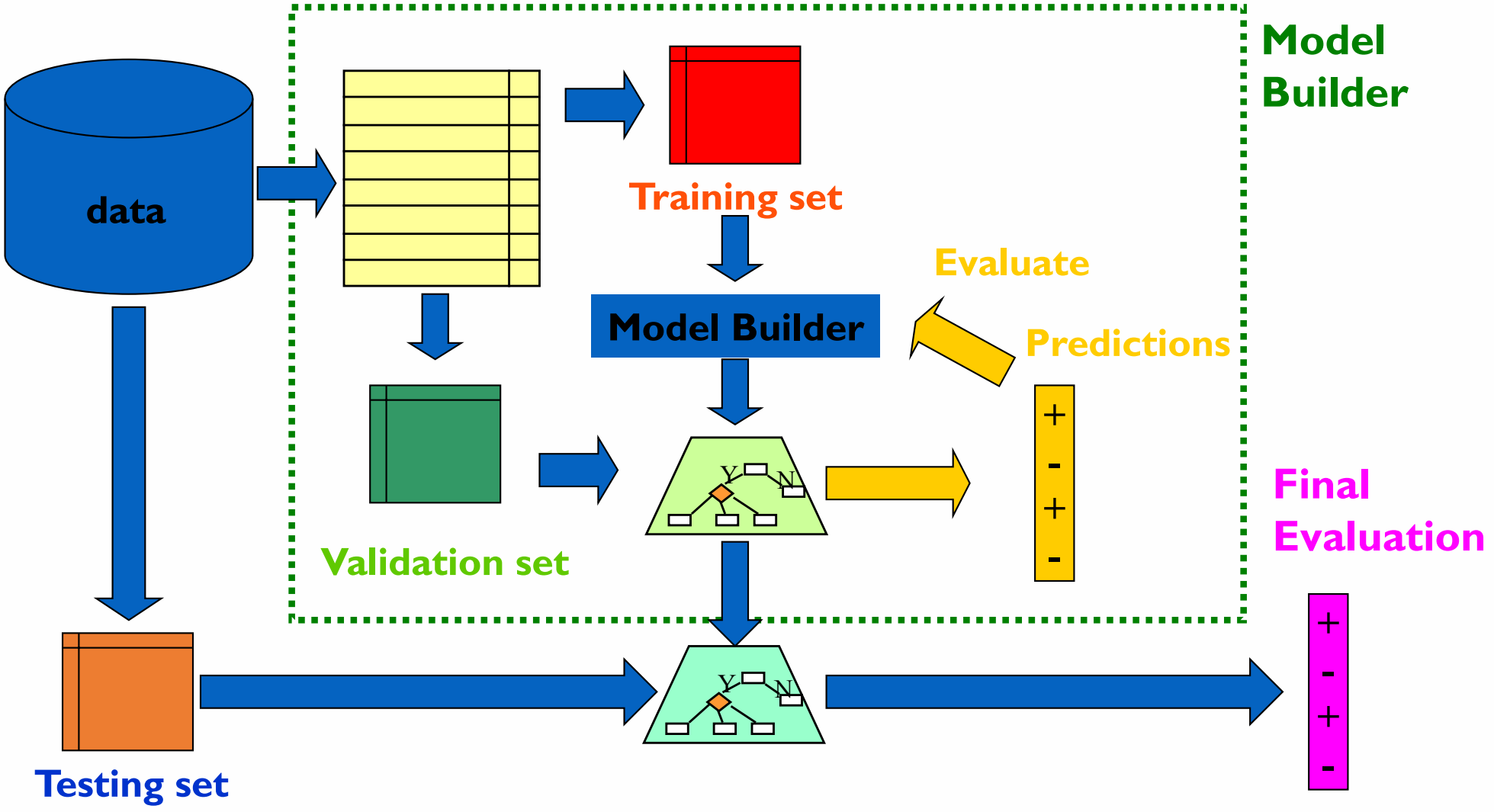
Performance Evaluation: Train & Test



What if We Need to Tune Parameters?

- Test data must not used in any way to create the classifier
- But what happens if we need to tune some parameters like for instance the regularization term in logistic regression?
- What if we need to choose between two classifiers?
- These schemes operate in two stages:
  - Stage 1: builds the basic structure
  - Stage 2: optimizes parameter settings or choose among different classification models
- Test data can't be used for parameter tuning!
- Proper procedure uses three sets: training data, validation data, and test data. Validation data is used to optimize parameters

The Full Process: Training, Validation, and Testing



Methods for Performance Evaluation

- How to obtain a reliable estimate of performance?

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

Making the most of the data

- Once evaluation is complete, all the data can be used to build the final classifier
- Generally, the larger the training data the better the classifier (but returns diminish)
- The larger the test data the more accurate the error estimate

Methods of Estimation

- Holdout
  - Reserve ½ for training and ½ for testing
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
- Bootstrap
  - Sampling with replacement

Holdout Evaluation & "Small" Datasets

- Reserves a certain amount for testing and uses the remainder for training, typically,
  - Reserve ½ for training and ½ for testing
  - Reserve 2/3 for training and 1/3 for testing
- For small or "unbalanced" datasets, samples might not be representative
- For instance, it might generate training or testing datasets with few or none instances of some classes
- Stratified sampling
  - Makes sure that each class is represented with approximately equal proportions in both subsets

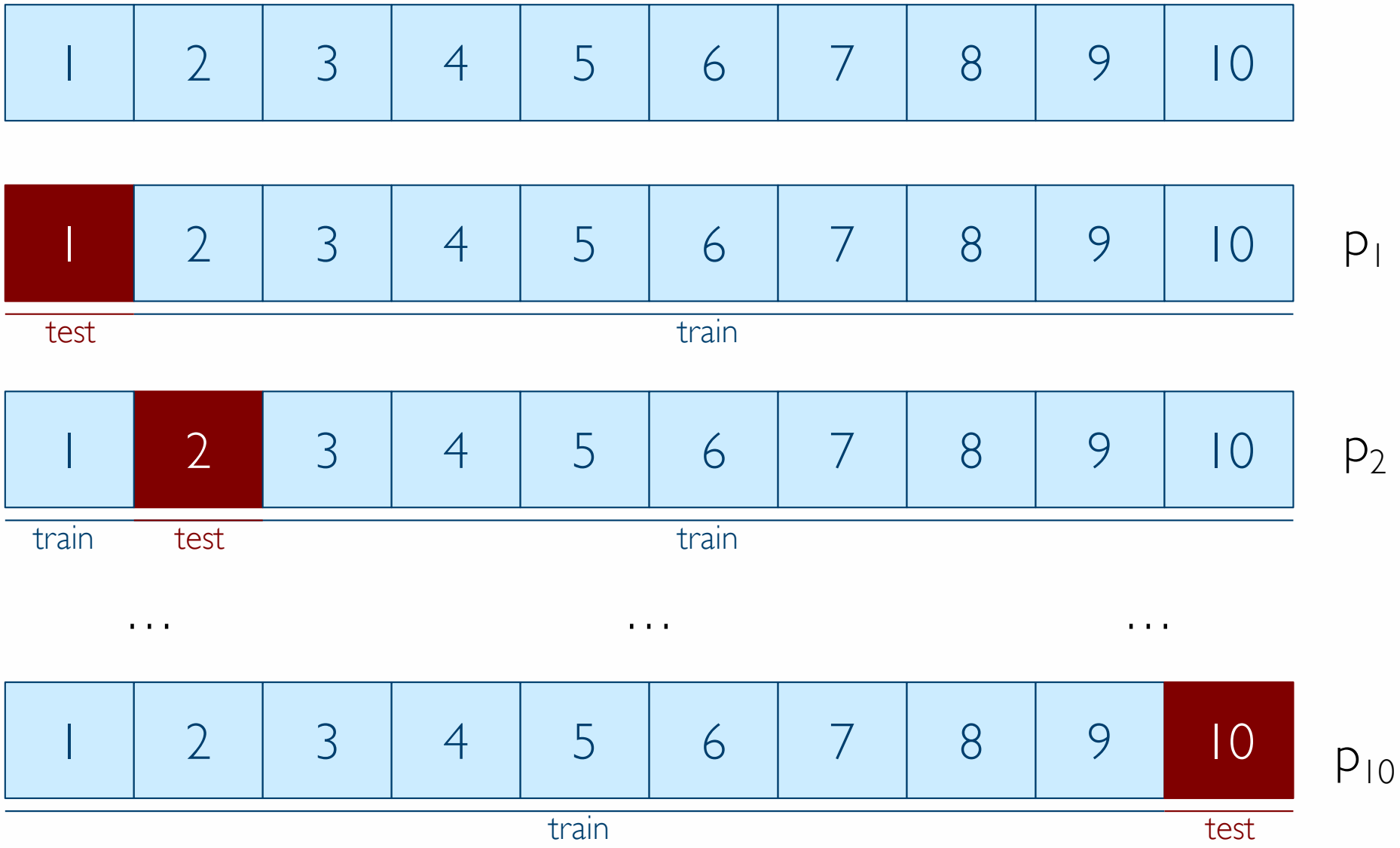
Repeated Holdout

- Holdout estimate can be made more reliable by repeating the process with different subsamples
- In each iteration, a certain proportion is randomly selected for training (possibly with stratification)
- The error rates on the different iterations are averaged to yield an overall error rate
- Still not optimum since the different test sets overlap

Cross-Validation

- First step
  - Data is split into k subsets of equal size
- Second step
  - Each subset in turn is used for testing and the remainder for training
- This is called k-fold cross-validation and avoids overlapping test sets
- Often the subsets are stratified before cross-validation is performed
- The error estimates are averaged to yield an overall error estimate

Ten-fold Crossvalidation



The final performance is computed as the average p_i

Cross-Validation

- Standard method for evaluation stratified ten-fold cross-validation
- Why ten? Extensive experiments have shown that this is the best choice to get an accurate estimate
- Stratification reduces the estimate's variance
- Even better: repeated stratified cross-validation
- E.g. ten-fold cross-validation is repeated ten times and results are averaged (reduces the variance)
- Other approaches appear to be robust, e.g., 5x2 crossvalidation

Leave-One-Out Cross-Validation

- It is a particular form of cross-validation
  - Set number of folds to number of training instances
  - I.e., for n training instances, build classifier n times
- Makes best use of the data
- Involves no random subsampling
- Computationally expensive

Leave-One-Out and Stratification

- Disadvantage of Leave-One-Out is that: stratification is not possible
- It guarantees a non-stratified sample because there is only one instance in the test set!
- Extreme example: random dataset split equally into two classes
  - Best inducer predicts majority class
  - 50% accuracy on fresh data
  - Leave-One-Out-CV estimate is 100% error!

Bootstraping

- Cross-validation uses sampling without replacement
  - The same instance, once selected, can not be selected again for a particular training/test set
- Bootstrap uses sampling with replacement
  - Sample a dataset of n instances n times with replacement to form a new dataset of n instances
  - Use this data as the training set
  - Use the instances from the original dataset that don't occur in the new training set for testing

The 0.632 Bootstrap

- An instance has a probability of 1–1/n of not being picked
- Thus its probability of ending up in the test data is:

$$\left(1-\frac{1}{N}\right)^{N} \approx e^{-1} = 0.368$$

- This means the training data will contain approximately 63.2% of the instances

Estimating Error using Bootstrap

- The error estimate on the test data will be very pessimistic, since training was on just around 63% of the instances
- Therefore, we compute the overall error ε by combining error on the train and test as,

$$\varepsilon = 0.632 \times \varepsilon_{test} + 0.368 \times \varepsilon_{train}$$

- The training error gets less weight than the error on the test data
- Repeat process several times with different replacement samples; average the results

How to compare the relative performance among competing models?

How to Compare the Performance of Two Models?

- Suppose we have two models
  - Model M_A with an accuracy = 82% computed using 10-fold crossvalidation
  - Model M_B with an accuracy = 80% computed using 10-fold crossvalidation
- How much confidence can we place on accuracy of M_A and M_B?
- Can we say M_A is better than M_B?
- Can the difference in performance measure be explained as a result of random fluctuations in the test set?

How do we know that the difference in performance is not just due to chance?

We computes the odds of it!

Apply the t-test and compute the p-value

The p-value represents the probability that the reported difference is due to chance

Paired t-Test Evaluation using k-fold Crossvalidation

- Generate the k folds and for each configuration compute the performance of model A and B,

$$\theta_1^A \ldots \theta_k^A \qquad \theta_1^B \ldots \theta_k^B$$

- We define,

$$\delta_i = \theta_i^A - \theta_i^A \quad \mu_\delta = \frac{1}{k}\sum_i \delta_i \quad \sigma_\delta = \frac{1}{k}(\delta_i - \mu_\delta)^2$$

- And two hypotheses,

$$H_0 : \mu_\delta = 0 \qquad H_a : \mu_\delta \neq 0$$

- We apply the t-test to check whether we can reject the null hypothesis H_0 with a target confidence

ALGORITHM 22.4. Paired *t*-Test via Cross-Validation

PAIRED *t*-TEST(α, K, $\mathbf{D}$):

1 $\mathbf{D} \leftarrow$ randomly shuffle $\mathbf{D}$
2 $\{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_K\} \leftarrow$ partition $\mathbf{D}$ in K equal parts
3 **foreach** $i \in [1, K]$ **do**
4 $\quad M_i^A, M_i^B \leftarrow$ train the two different classifiers on $\mathbf{D} \setminus \mathbf{D}_i$
5 $\quad \theta_i^A, \theta_i^B \leftarrow$ assess M_i^A and M_i^B on $\mathbf{D}_i$
6 $\quad \delta_i = \theta_i^A - \theta_i^B$
7 $\hat{\mu}_\delta = \frac{1}{K} \sum_{i=1}^{K} \delta_i$
8 $\hat{\sigma}_\delta^2 = \frac{1}{K} \sum_{i=1}^{K} (\delta_i - \hat{\mu}_\delta)^2$
9 $Z_\delta^* = \frac{\sqrt{K}\hat{\mu}_\delta}{\hat{\sigma}_\delta}$
10 **if** $Z_\delta^* \in \left(-t_{\alpha/2,K-1}, t_{\alpha/2,K-1}\right)$ **then**
11 $\quad$ Accept H_0; both classifiers have similar performance
12 **else**
13 $\quad$ Reject H_0; classifiers have significantly different performance

Comparing the Performance of Two Models Using k-fold Crossvalidation

- Apply k-fold crossvalidation to each model obtaining k evaluations for each algorithm over the same folder configuration
- Fix a confidence level c (e.g., 0.95 corresponding to 95%)
- Apply Student's t-test and compute the p-value to test whether the reported difference is statistically significant
  - If the p-value is larger than 1-c (0.05 in the example), then the difference is not significant
  - If the p-value is smaller than 1-c (smaller than 0.05 in the example), then the difference is significant.
- Note that the t-test can be paired or unpaired

"paired" when the estimates
are from the same datasets

"unpaired" when the estimates
are from different datasets

run the python notebook dedicated to classifier comparison

Multiple Testing

- Say that you perform a statistical test with a 0.05 threshold, but you repeat the test on twenty different observations.
- For example, you want to compare the performance of several classification algorithms
- Assume that all of the observations are explainable by the null hypothesis
- What is the chance that at least one of the observations will receive a p-value less than 0.05?

Multiple Testing - Example

- Say that you perform a statistical test with a 0.05 threshold (95% confidence level), but you repeat the test on 20 different observations. What is the chance that at least one of the observations will receive a p-value less than 0.05?

- P(making a mistake) = 0.05
- P(not making a mistake) = 0.95
- P(not making any mistake) = 0.95^{20} = 0.358
- P(making at least one mistake) = 1 - 0.358 = 0.642
- There is a 64.2% chance of making at least one mistake.

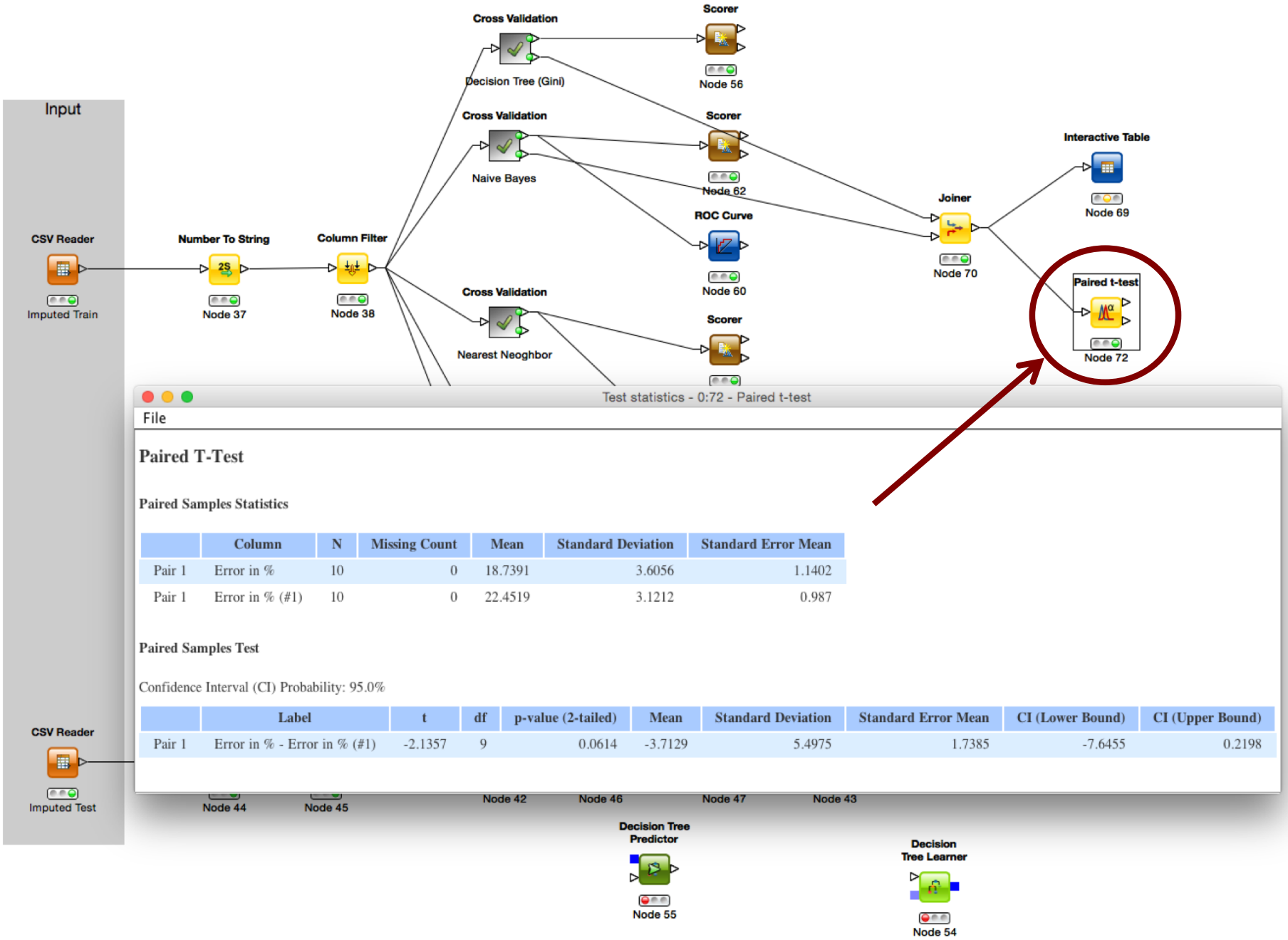
Bonferroni Correction

- Assume that individual tests are independent.
- Divide the desired p-value threshold by the number of tests performed.
- Example
  - We now have, the threshold set to 0.05/20 = 0.0025.
  - P(making a mistake) = 0.0025
  - P(not making a mistake) = 0.9975
  - P(not making any mistake) = 0.9975^{20} = 0.9512
  - P(making at least one mistake) = 1 - 0.9512 = 0.0488

Non Parametric Tests

- They do not make any assumption about the distribution of the variable in the population
- Mann-Whitney U Test
  - Nonparametric equivalent of the independent t-test
- Wilcoxon matched-pairs signed rank test
  - Used to compare two related groups

	Nonparametric tests		Parametric tests
	Nominal data	Ordinal data	Ordinal, interval, ratio data
One group	Chi square goodness of fit	Wilcoxon signed rank test	One group t-test
Two unrelated groups	Chi square	Wilcoxon rank sum test, Mann-Whitney test	Student's t-test
Two related groups	McNemar's test	Wilcoxon signed rank test	Paired Student's t-test
K-unrelated groups	Chi square test	Kruskal -Wallis one-way analysis of variance	ANOVA
K-related groups		Friedman matched samples	ANOVA with repeated measurements


Input
CSV Reader
Imputed Train
Number To String
Node 37
Column Filter
Node 38
Cross Validation
Decision Tree (Gini)
Scorer
Node 56
Cross Validation
Naive Bayes
Scorer
Node 62
ROC Curve
Node 60
Cross Validation
Nearest Neoghbor
Scorer
Joiner
Node 70
Interactive Table
Node 69
Paired t-test
Node 72
Test statistics - 0:72 - Paired t-test
File
Paired T-Test
Paired Samples Statistics
Column | N | Missing Count | Mean | Standard Deviation | Standard Error Mean
Pair 1 | Error in % | 10 | 0 | 18.7391 | 3.6056 | 1.1402
Pair 1 | Error in % (#1) | 10 | 0 | 22.4519 | 3.1212 | 0.987
Paired Samples Test
Confidence Interval (CI) Probability: 95.0%
Label | t | df | p-value (2-tailed) | Mean | Standard Deviation | Standard Error Mean | CI (Lower Bound) | CI (Upper Bound)
Pair 1 | Error in % - Error in % (#1) | -2.1357 | 9 | 0.0614 | -3.7129 | 5.4975 | 1.7385 | -7.6455 | 0.2198
CSV Reader
Imputed Test
Node 44
Node 45
Node 42
Node 46
Node 47
Node 43
Decision Tree Predictor
Node 55
Decision Tree Learner
Node 54

Probabilistic Classifiers

Probabilistic Classifiers & Classification Thresholds

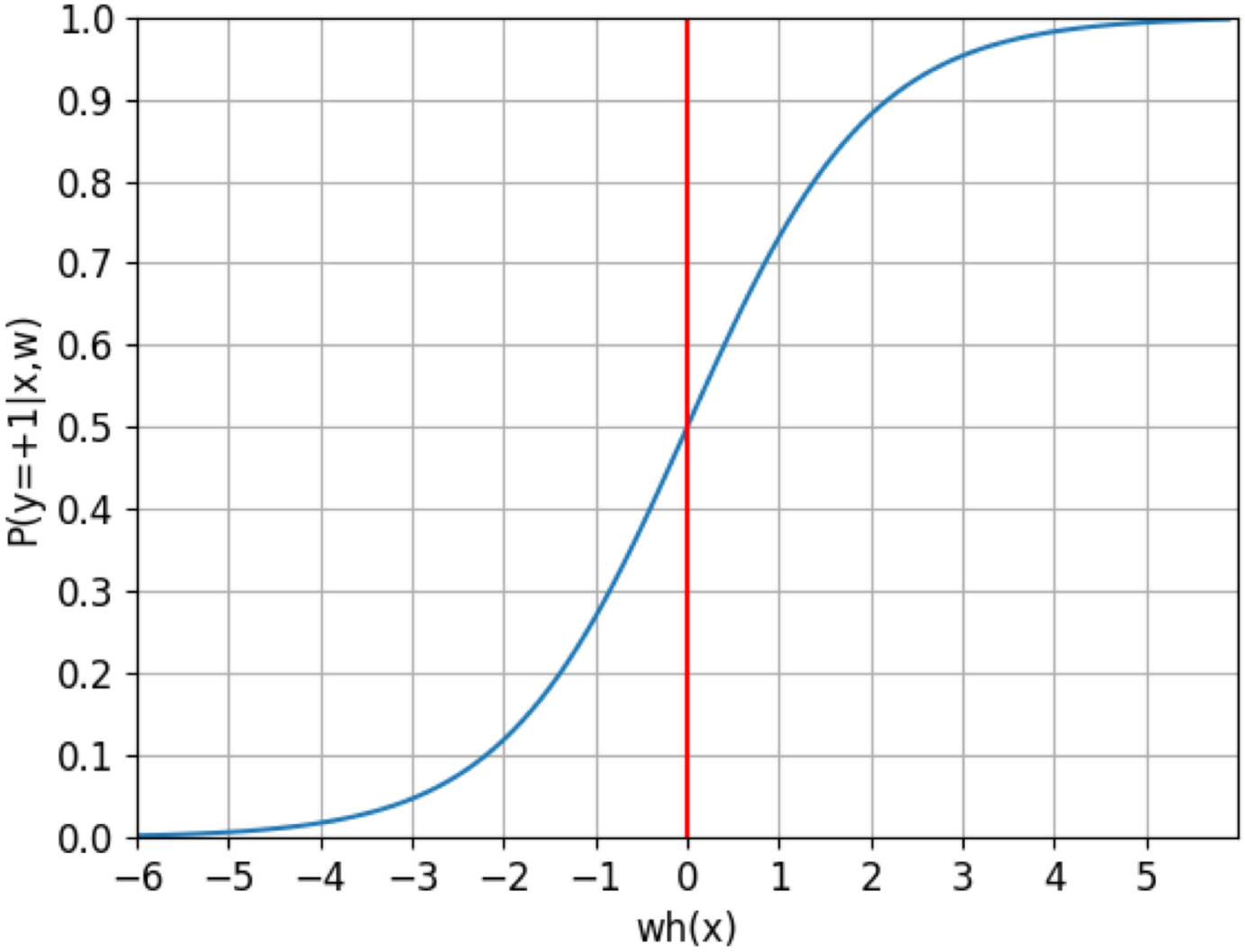
- Up to now we used logistic regression to predict classifier labels, however, logistic regression returns a probability

$$P(y_i|\vec{x_i})$$

- Given an example x_i its predicted class is the label with the largest probability so it is equivalent to using a threshold of 0.5 to decide which class to assign to an example

- However, we can use a different threshold and for instance label as positive only examples we return a 1 only when $P(+1|x)>0.75$

- This would label as positive only cases for which we are more confident that should be labeled as positive.


1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
P(y=+1|x,w)
−6
−5
−4
−3
−2
−1
0
1
2
3
4
5
wh(x)

How the Classification Threshold Influence Precision and Recall?

- Suppose we use a near one threshold to classify positive examples
- Then, we will classify as positives only examples for which we are very confident (this is a pessimistic classifier)
- Precision will be high
  - In fact, we are not likely produce false positives
- Recall will be low
  - In fact, we are likely to produce more false negatives

How the Classification Threshold Influence Precision and Recall?

- Suppose we use a near zero threshold to classify positive examples
- Then, we will classify everything as positives
  (this is an optimistic classifier)
- Precision will be low as we are going to generate the maximum number of false positives (everything is positive!)
- Recall will be high since by classifying everything as positive we are going to generate the maximum number of false negatives

We can use the threshold to optimize our precision and recall

a higher the threshold, increases precision and lower recall

a lower threshold, decreases precision and increase recall

Precision-Recall Tradeoff Example

- In the notebook, a simple logistic regression model applied to the loans data returns the confusion matrix below, corresponding to a precision of 0.82 and recall of 0.99

	Classified -1	Classified +1
Labeled -1	1212	21910
Labeled +1	1057	98283

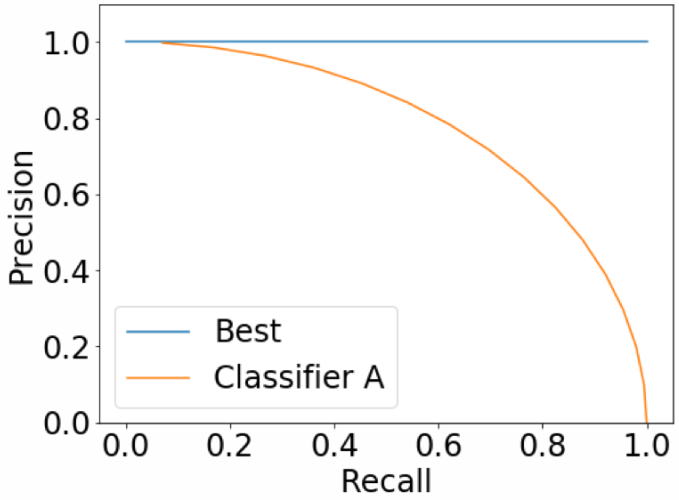
- By increasing the classification threshold for positive (+1) examples to 0.75 we obtained a new confusion matrix with precision of 0.86 and a recall of 0.99

	Classified -1	Classified +1
Labeled -1	10632	12490
Labeled +1	20106	79234

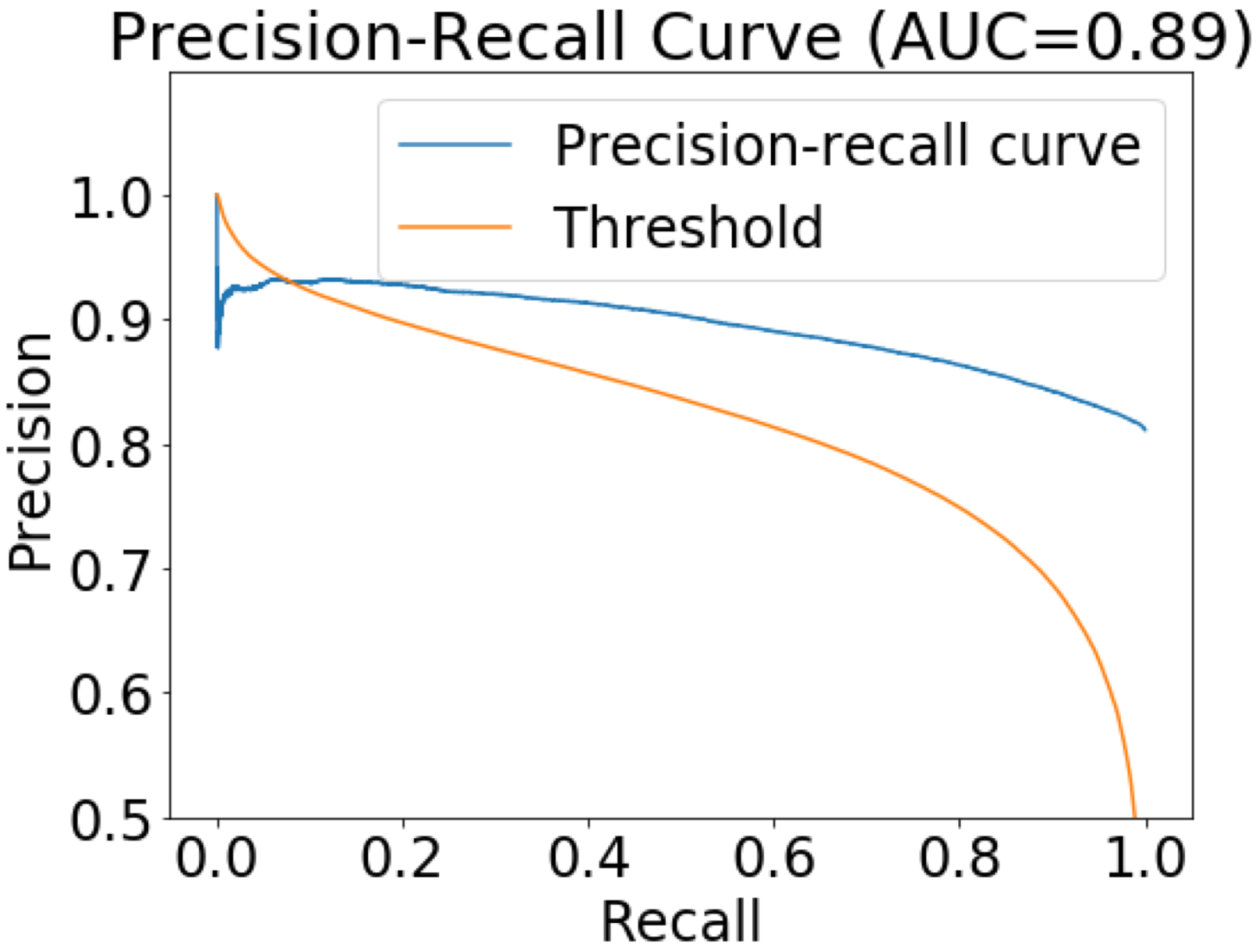
- Overall, we reduced the number of false positives (we did not accept risky loans and we were better at identifying risky loans)

Precision-Recall Curves

- Plot precision as a function of recall for varying threshold values



- The best classifier would be the one that has always a precision equal to one (but never happens)
- More in general classifiers will show of different shapes
- How to decide among more classifiers?
  - Use the area under the curve (the nearer to one, the better)
  - Use F1 measure



Precision-recall curve for the loan dataset (run the python notebook for details)

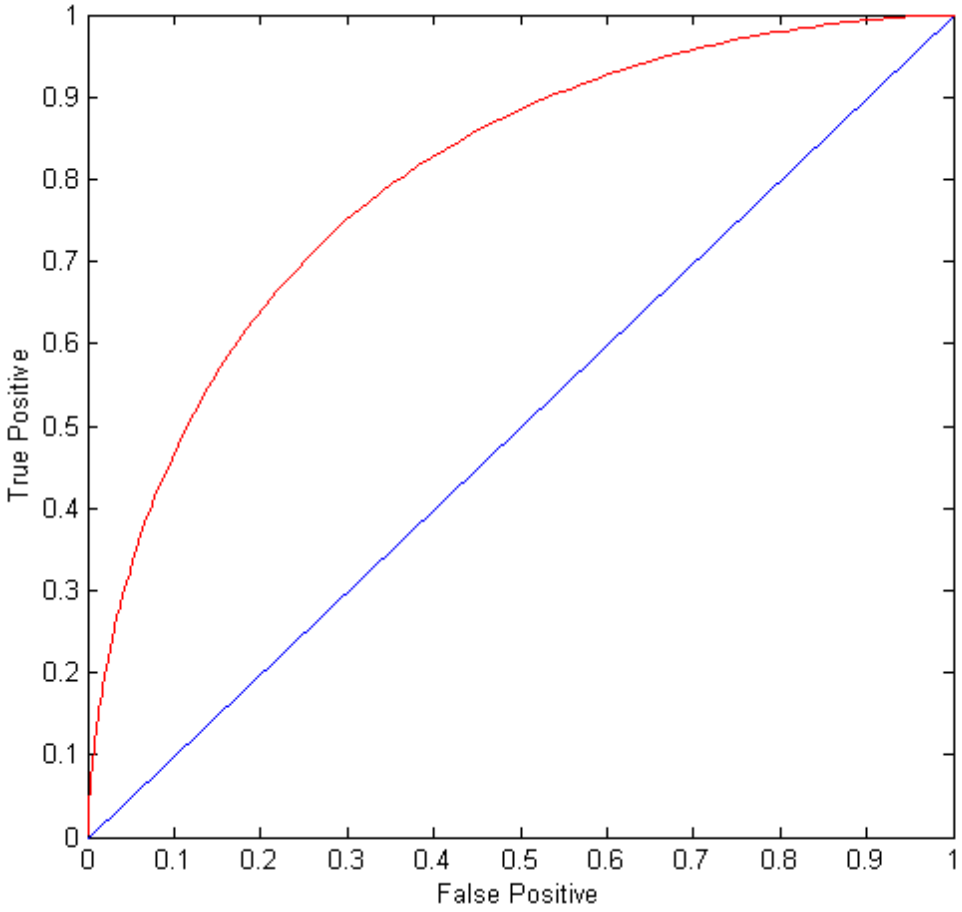
Receiver Operating Characteristic (ROC) Curves

ROC Curve (Receiver Operating Characteristic)

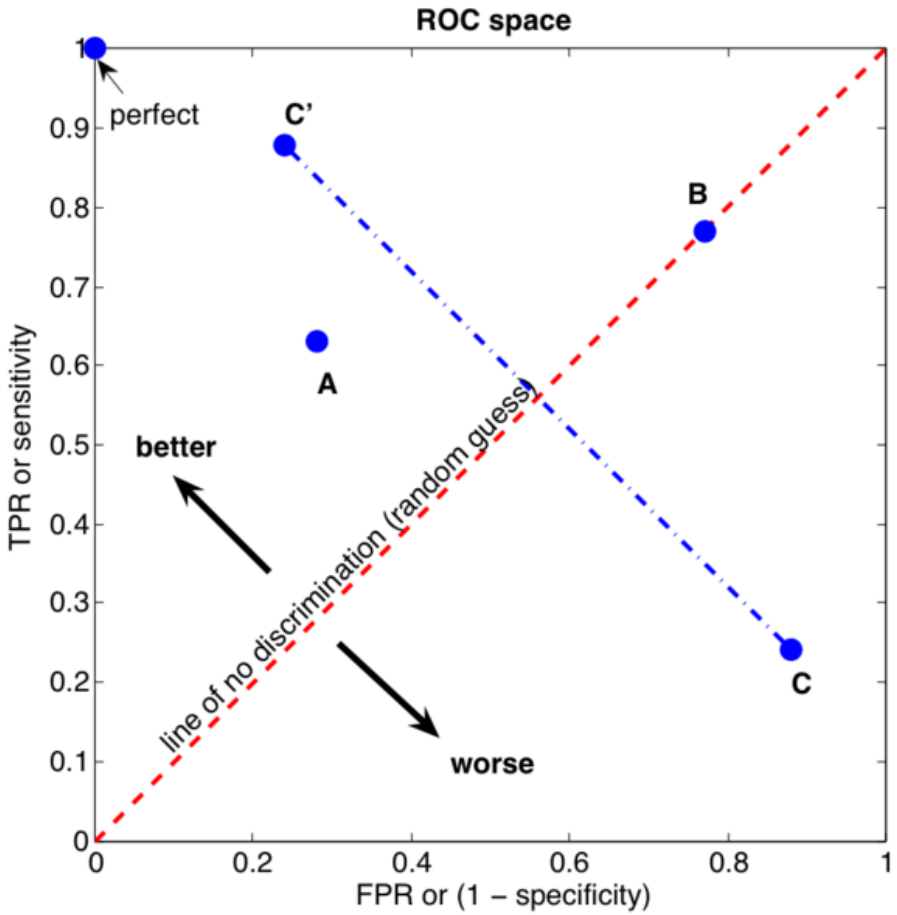
- Developed in 1950s for signal detection theory to analyze signals
- Plot the True Positive Rate (TPR=TP/(TP+FN)) against the False Positive Rate (FPR=FP/(TN+FP))
- Performance of each classifier represented as a point on the ROC curve
- Changing the classification threshold, sample distribution or cost matrix changes the location of the point

ROC Curve

- (FPR, TPR)
  - (0,0): declare everything to be negative class
  - (1,1): declare everything to be positive class
  - (0,1): ideal
- Diagonal line:
  - Random guessing
  - Below diagonal line, prediction is opposite of the true class



ROC Curve



A

TP=63	FP=28	91
FN=37	TN=72	109
100	100	200

TPR = 0.63

FPR = 0.28

ACC = 0.68

B

TP=77	FP=77	154
FN=23	TN=23	46
100	100	200

TPR = 0.77

FPR = 0.77

ACC = 0.50

C

TP=24	FP=88	112
FN=76	TN=12	88
100	100	200

TPR = 0.24

FPR = 0.88

ACC = 0.18

C'

TP=88	FP=24	112
FN=12	TN=76	88
100	100	200

TPR = 0.88

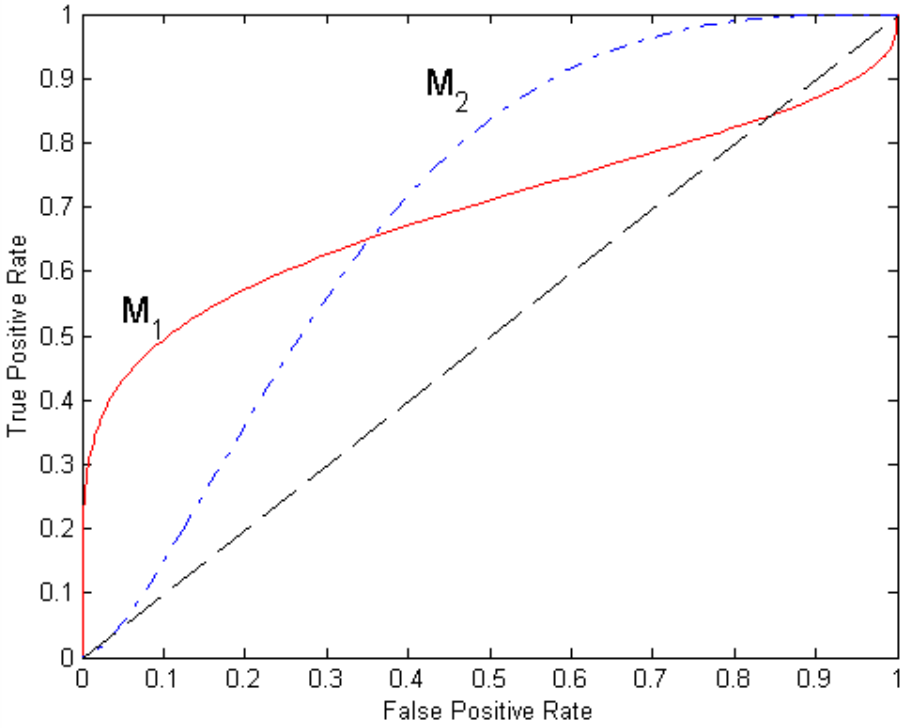
FPR = 0.24

ACC = 0.82

Using ROC for Model Comparison

- No model consistently outperform the other
- M_1 is better for small FPR
- M_2 is better for large FPR

- Area Under the ROC curve
- Ideal, area = 1
- Random guess, area = 0.5



There are techniques similar to ROC in other areas

Lift charts are an example

Run the notebook dedicated
to the precision-recall tradeoff

Which model should we prefer?
(Model selection)

Model Selection Criteria

- Model selection criteria attempt to find a good compromise between:
  - The complexity of a model
  - Its prediction accuracy on the training data
- Reasoning: a good model is a simple model that achieves high accuracy on the given data
- Also known as Occam's Razor : the best theory is the smallest one that describes all the facts

William of Ockham, born in the village of Ockham in Surrey (England) about 1285, was the most influential philosopher of the 14th century and a controversial theologian.

Which Classification Algorithm?

- Among the several algorithms, which one is the "best"?
  - Some algorithms have a lower computational complexity
  - Different algorithms provide different representations
  - Some algorithms allow the specification of prior knowledge
- If we are interested in the generalization performance, are there any reasons to prefer one classifier over another?
- Can we expect any classification method to be superior or inferior overall?
- According to the No Free Lunch Theorem, the answer to all these questions is known

No Free Lunch Theorem

- If the goal is to obtain good generalization performance, there are no context-independent or usage-independent reasons to favor one classification method over another
- If one algorithm seems to outperform another in a certain situation, it is a consequence of its fit to the particular problem, not the general superiority of the algorithm
- When confronting a new problem, this theorem suggests that we should focus on the aspects that matter most
  - Prior information
  - Data distribution
  - Amount of training data
  - Cost or reward
- The theorem also justifies skepticism regarding studies that "demonstrate" the overall superiority of a certain algorithm

No Free Lunch Theorem

- "[A]ll algorithms that search for an extremum of a cost [objective] function perform exactly the same, when averaged over all possible cost functions." [1]
- "[T]he average performance of any pair of algorithms across all possible problems is identical." [2]
- Wolpert, D.H., Macready, W.G. (1995), No Free Lunch Theorems for Search, Technical Report SFI-TR-95-02-010 (Santa Fe Institute).
- Wolpert, D.H., Macready, W.G. (1997), No Free Lunch Theorems for Optimization, IEEE Transactions on Evolutionary Computation 1, 67.

Optional Material

Lift Charts

- Lift is a measure of the effectiveness of a predictive model calculated as the ratio between the results obtained with and without the predictive model
- Cumulative gains and lift charts are visual aids for measuring model performance
- Both charts consist of a lift curve and a baseline
- The greater the area between the lift curve and the baseline, the better the model

Example: Direct Marketing

- Mass mailout of promotional offers (1000000)
- The proportion who normally respond is 0.1%, that is 1000 responses
- A data mining tool can identify a subset of a 100000 for which the response rate is 0.4%, that is 400 responses
- In marketing terminology, the increase of response rate is known as the lift factor yielded by the model.
- The same data mining tool, may be able to identify 400000 households for which the response rate is 0.2%, that is 800 respondents corresponding to a lift factor of 2.
- The overall goal is to find subsets of test instances that have a high proportion of true positive

An Example from Direct Marketing