Evaluation

Charles Sutton
Data Mining and Exploration
Spring 2012

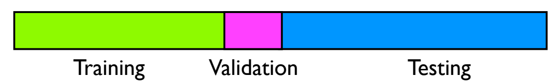
Evaluate what?

- Do you want to evaluate a **classifier** or a **learning algorithm**?
- Do you want to **predict accuracy** or **predict which one is better**?
- Do you have a **lot of data** or **not much**?
- Are you interested in **one domain** or in understanding accuracy **across domains**?

For really large amounts of data....

- You could use training error to estimate your test error
  - But this is stupid, so don't do it
- Instead split the instances randomly into a training set and test set
- But then suppose you need to:
  - Compare 5 different algorithms
  - Compare 5 different feature sets
  - Each of them have different knobs in the training algorithm (e.g., size of neural network, gradient descent step size, k in k-nearest neighbour, etc., etc.)

A: Use a validation set.



- When you first get the data, put the test set away and don't look at it.
- The validation set lets you compare the "tweaking" parameters of different algorithms.

This is a fine way to work, **if** you have lots of data.

1. Hypothesis Testing

Variability

Classifier A: 81% accuracy

Classifier B: 84% accuracy

Which classifier do you think is best?

Variability

Classifier A: 81% accuracy

Classifier B: 84% accuracy

But then suppose I tell you
- Only 100 examples in the test set
- After 400 more test examples, I get

	0-100	101-200	201-300	301-400	401-500
A:	0.81	0.77	0.78	0.81	0.78
B:	0.84	0.75	0.75	0.76	0.78

Sources of Variability

- Choice of training set
- Choice of test set
- Inherent randomness in learning algorithm
- Errors in data labeling

	0-100	101-200	201-300	301-400	401-500
A:	0.81	0.77	0.78	0.81	0.78
B:	0.84	0.75	0.75	0.76	0.78

Key point:

Your measured testing error is a random variable (you sampled the testing data)

Want to infer the "true test error" based on this sample

This is another learning problem!

Next slide: Make this more formal...

Test error is a random variable

Call your test data $x_1, x_2, \ldots, x_N$

where $x_i \sim \mathcal{D}$ independently

h the classifier

f the true function

$\mathbf{1}_{\text{foo}}$ delta function

True error

$$e = \Pr_{x \sim \mathcal{D}}[f(x) \neq h(x)]$$

Test error

$$\hat{e} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{[f(x_i) \neq h(x_i)]}$$

Theorem: As $N \to \infty$ then $\hat{e} \to e$ [Why?]

Test error is a random variable

Call your test data $x_1, x_2, \ldots, x_N$

where $x_i \sim \mathcal{D}$ independently

h the classifier

f the true function

$\mathbf{1}_{\text{foo}}$ delta function

True error

$$e = \Pr_{x \sim \mathcal{D}}[f(x) \neq h(x)]$$

Test error

$$\hat{e} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{[f(x_i) \neq h(x_i)]}$$

Theorem: $\hat{e} \sim \text{Binomial}(N, e)$

Main question

Suppose

Classifier A: 81% accuracy

Classifier B: 84% accuracy

Is that difference real?

Rough-and-ready variability

Classifier A: 81% accuracy

Classifier B: 84% accuracy

Is that difference real?

Answer 1:

If doing c-v, report both mean and standard deviation of error across folds.

If doing c-v, report both mean and standard deviation of error across folds.

	Learning	**Evaluation**
World	Original problem (e.g., Difference between spam and normal emails)	True error
Sample	Inboxes for multiple users	Classifier performance on each example
Estimation	Classifier	Avg error on test set

Hypothesis testing

Want to know whether $\hat{e}_A$ and $\hat{e}_B$ are significantly different.

1. Suppose not. ["null hypothesis"]
2. Define a test statistic, in this case $T = |e_A - e_B|$
3. Measure a value of the statistic $\hat{T} = |\hat{e}_A - \hat{e}_B|$
4. Derive the distribution of $\hat{T}$ assuming #1.
5. If $p = \Pr[T > \hat{T}]$ is really low, e.g., < 0.05, "reject the null hypothesis"

   p is your p-value

If you reject, then the difference is "statistically significant"

Example

Classifier A: 81% accuracy

Classifier B: 84% accuracy

$$\hat{T} = |\hat{e}_A - \hat{e}_B| = 0.03$$

4. Derive the distribution of $\hat{T}$ assuming the null.

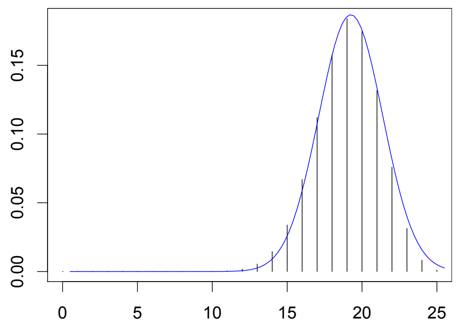
What we know:

$$\hat{e}_A \sim \mathrm{Binomial}(N, e_A)$$

$$\hat{e}_B \sim \mathrm{Binomial}(N, e_B)$$

$$e_A = e_B$$

Approximation to the rescue



Approximate binomial by normal

$$\hat{e}_A \sim \mathcal{N}(Ne_A, Ne_A(1-e_A))$$

Distribution under the null

4. Derive the distribution of $\hat{T}$ assuming the null.

What we know:

$$\hat{e}_A \sim \mathcal{N}(Ne_A, s_A^2)$$

$$\hat{e}_B \sim \mathcal{N}(Ne_B, s_B^2)$$

$$e_A = e_B$$

where

$$s_A^2 = Ne_A(1-e_A)$$

Distribution under the null

4. Derive the distribution of $\hat{T}$ assuming the null.

What we know:

$$\hat{e}_A \sim \mathcal{N}(Ne_A, s_A^2)$$

$$\hat{e}_B \sim \mathcal{N}(Ne_B, s_B^2)$$

$$e_A = e_B$$

where

$$s_A^2 = Ne_A(1-e_A)$$

$$s_{AB}^2 = \frac{2e_{AB}(1-e_{AB})}{N}$$

$$e_{AB} = \frac{1}{2}(e_A + e_B)$$

But this means

$$\hat{e}_A - \hat{e}_B \sim \mathcal{N}(0, s_{AB}^2)$$

(assuming the two are independent...)

Computing the p-value

5. If $p = \Pr[T > \hat{T}]$ is really low, e.g., < 0.05, "reject the null hypothesis"

In our example

$$\hat{e}_A - \hat{e}_B \sim \mathcal{N}(0, s_{AB}^2)$$

$$s_{AB}^2 \approx 0.0029$$

So one line of R (or MATLAB):

```
> pnorm(-0.03, mean=0, sd=sqrt(0.0029))
[1] 0.2887343
```

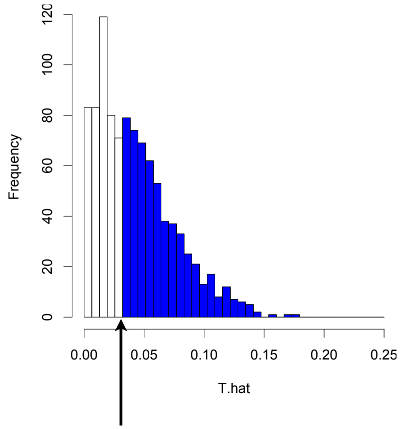
Frequentist Statistics

What does

$$p = \Pr[T > \hat{T}]$$

really mean?

Generated 1000 test sets for classifiers A and B, computed error under the null:



Our example: $\hat{T} = 0.03$

p-value is shaded area

Frequentist Statistics

What does

$$p = \Pr[T > \hat{T}]$$

really mean?

Refers to the "frequency" behaviour if the test is applied over and over for different data sets.

Fundamentally different (and more orthodox) than Bayesian statistics.

Errors in the hypothesis test

Type I error: False rejects

Type II error: False non-reject

Logic is to fix the Type I error $\alpha = 0.05$

Design the test to minimise Type II error

Summary

- Call this test "difference in proportions test"
- An instance of a "z-test"
- This is OK, but there are tests that work better in practice...

McNemar's Test

	Classifier B correct	Classifier B wrong
Classifier A correct	n_{11}	n_{01}
Classifier A wrong	n_{10}	n_{00}

McNemar's Test

p_A probability A is correct GIVEN A and B disagree

Null hypothesis: $p_A = 0.5$

Test statistic:

$$\frac{(|n_{10} - n_{01}| - 1)^2}{n_{01} + n_{10}}$$

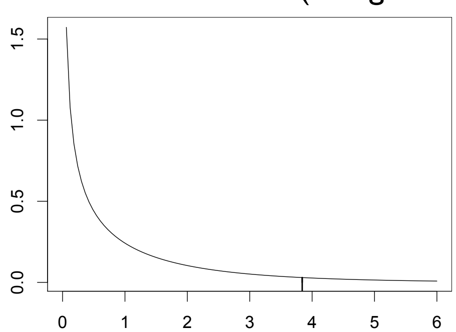
Distribution under null?

McNemar's Test

Test statistic:

$$\frac{(|n_{10} - n_{01}| - 1)^2}{n_{01} + n_{10}}$$

Distribution under null? χ^2 (1 degree of freedom)



Pros/Cons McNemar's test

Pros

- Doesn't require the independence assumptions of the difference-of-proportions test
- Works well in practice [Dietterich, 1997]

Cons

- Does not assess training set variability

Accuracy is not the only measure

Accuracy is great, but not always helpful
e.g., Two class problem. 98% instances negative

Alternative: for every class C, define

Precision $$P = \frac{\#\text{ instances of } C \text{ that classifier got right}}{\#\text{ instances that classifier predicted } C}$$

Recall $$R = \frac{\#\text{ instances of } C \text{ that classifier got right}}{\#\text{ true instances of } C}$$

F-measure $$F_1 = \frac{2}{\frac{1}{P} + \frac{1}{R}} = \frac{2PR}{P + R}$$

Calibration

Sometimes we care about the confidence of a classification.

If the classifier outputs probabilities, can use cross-entropy:

$$H(p) = \frac{1}{N} \sum_{i=1}^{N} \log p(y_i|x_i)$$

where

(x_i, y_i) feature vector, true label for each instance i

$p(y_i|x_i)$ probabilities output by the classifier

An aside

- We've talked a lot about overfitting.
- What does this mean for well-known contest data sets? (Like the ones in your mini-project.)
- Think about the paper publishing process. I have an idea, implement it, try it on a standard train/test set, publish a paper if it works.
- Is there a problem with this?

2. ROC curves

(Receiver Operating Characteristic)

Problems in what we've done so far

- Skewed class distributions
- Differing costs

Classifiers as rankers

- Most classifiers output a real-valued score as well as a prediction
  - e.g., decision trees: proportion of classes at leaf
  - e.g., logistic regression: P(class | x)
- Instead of evaluating accuracy at a single threshold, evaluate how good the score is at ranking

More evaluation measures

Assume two classes. (Hard to do ROC with more.)

		True +	True -
Predicted	+	TP	FP
	-	FN	TN

TP: True positives
FP: False positives
TN: True negatives
FN: False negatives

More evaluation measures

		True +	True -
Predicted	+	TP	FP
	-	FN	TN

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

More evaluation measures

		True +	True -
Predicted	+	TP	FP
	-	FN	TN

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$P = \frac{TP}{TP + FP}$$

More evaluation measures

		True +	True -
Predicted	+	TP	FP
	-	FN	TN

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$TPR = \frac{TP}{TP + FN}$$ a.k.a., recall

$$P = \frac{TP}{TP + FP}$$

$$FPR = \frac{FP}{FP + TN}$$

TPR: True positive rate, FPR: False positive rate

More evaluation measures

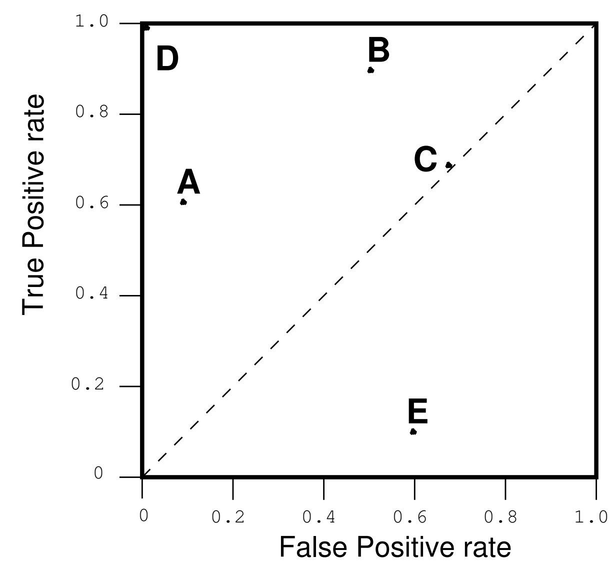
		True +	True -
Predicted	+	TP	FP
	-	FN	TN

$$TPR = \frac{TP}{TP + FN}$$ ← total + instances

$$FPR = \frac{FP}{FP + TN}$$ ← total - instances

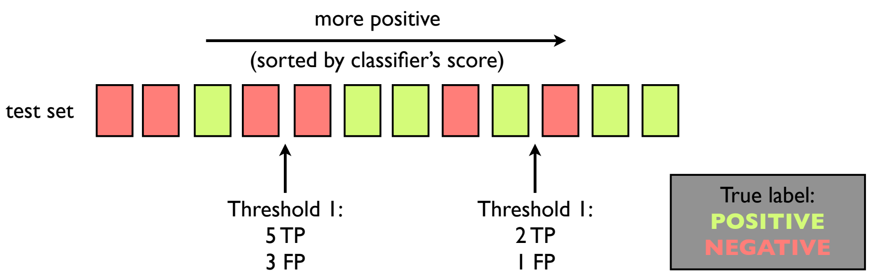
TPR: True positive rate, FPR: False positive rate

"ROC space"



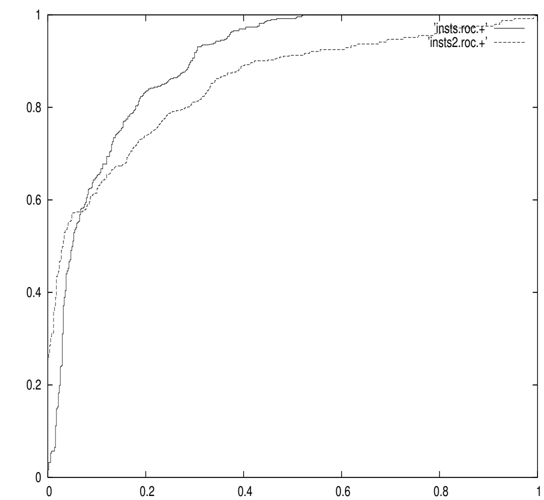
[Fawcett, 2003]

ROC curves



Add a point for every possible threshold, and....

ROC curves



[Fawcett, 2003]

Class skew

ROC curves insensitive to class skew.

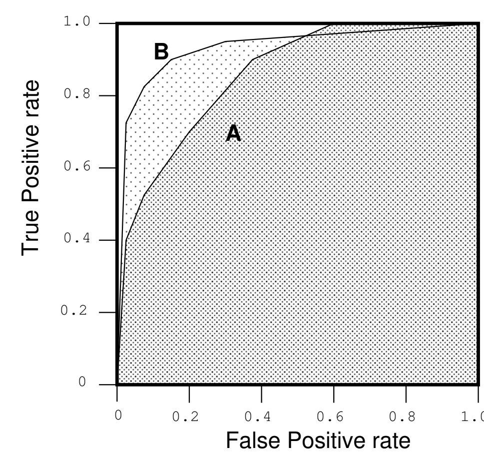
		True +	True -
Predicted	+	TP	FP
	-	FN	TN
		=P	=N

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

Area under curve (AUC)

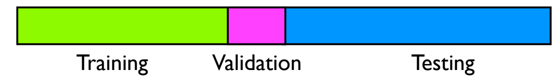
Sometimes you want a single number



[Fawcett, 2003]

3. Cross Validation

A: Use a validation set.



- When you first get the data, put the test set away and don't look at it.
- The validation set lets you compare the "tweaking" parameters of different algorithms.

This is a fine way to work, **if** you have lots of data.

Problem: You don't have lots of data

This causes two problems:

- You don't want to set aside a test set (waste of perfectly good data).
- There's lots of variability in your estimate of the error.

Cross-validation

Has several goals:

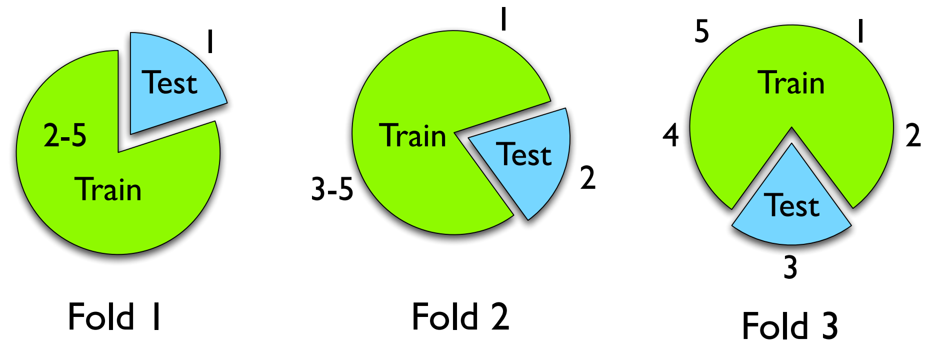
- Don't "waste" examples by never using them for training
- Get some idea of variation due to training sets
- Allow tweaking classifier parameters without use of a separate validation set

Cross-validation

- Randomly split data into K equal-sized subsets (called “folds”). Call these $D_1, D_2, \ldots D_k$
- For i = 1 to K
  - Train on all $D_1, D_2, \ldots D_k$ except D_i
  - Test on D_i. Let e_i be testing error
- Final estimate of test error:

$$\hat{e} = \frac{1}{K}\sum_{i=1}^{K} e_i$$

Cross-validation (prettily)



Final error estimate: Mean of test error in each fold

How to pick k?

- Bigger K (e.g., K=N called leave-one-out)
  - Bigger training sets (good if training data is small)
- Smaller K means
  - Bigger test sets (good)
  - Less computationally expensive
  - Less overlap in training sets
- I typically use 5 or 10
- N.B. Can use more than one fold for testing

Comments about C-V

- Tune parameters of your learning algorithm via cross-validation error
- Note that the different training sets are (highly) dependent
- Sometimes need to be careful about exactly which data goes into training-test splits (e.g., fMRI data, University HTML pages)

Some question about C-V

Say I'm doing 5-fold cv.

- I get 5 classifiers out. Which one is my "final" one for my problem?
- Let's say I want to choose the pruning parameter for my decision tree. I use c-v. How do I then estimate the error of my final classifier?

4. Evaluating clustering

How to evaluate clustering?

- If you really knew what you wanted, you'd be doing classification instead of clustering.
- Option 1: Measure how well the clusters do for some other task (e.g., as features for a classifier, or for ranking documents in IR)
  - Not always what you want to do.
- Option 2: Measure "goodness of fit"
- Option 3: Compare to an external set of labels

Evaluation for Clustering

Suppose that we do have labeled data for evaluation, called "ground truth", that we don't use in the clustering algorithm.

(More for evaluating an algorithm than a clustering.)

Each example has features X, cluster label C, and "ground truth label Y

$$C \in \{1, 2, \dots K\}$$

$$Y \in \{1, 2, \dots J\}$$

C_k set of examples in cluster k

Y_j set of examples with true label j

Purity

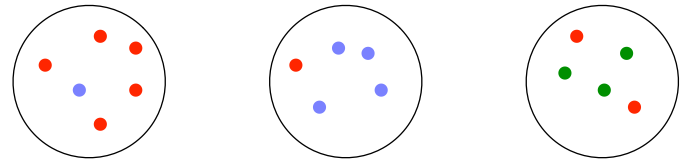
Essentially, your best possible accuracy if clusters are mapped to ground truth labels.

$$\text{Purity} = \frac{1}{N}\sum_{k=1}^{K} \max_{j\in[1,J]} |C_k \cap Y_j|$$

Reminder:

- C_k set of examples in cluster k
- Y_j set of examples with true label j
- N number of data points

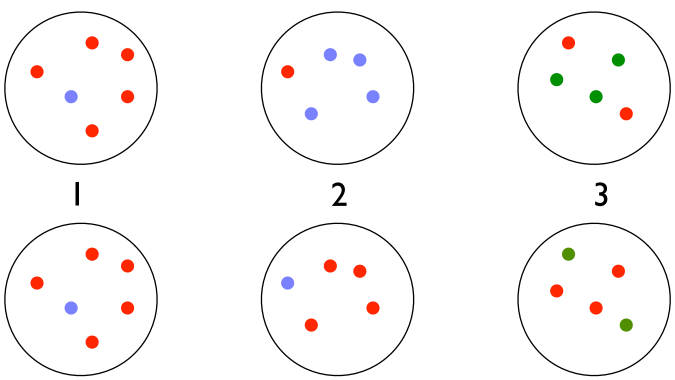
Purity example



Cluster 1 2 3

Purity is: $\frac{5+4+3}{6+5+5} = \frac{12}{16} = 0.75$

Problem with Purity



Alg A

Cluster 1 2 3

Alg B

Both of these have the same purity, but Alg B is doing no better than predicting majority class.

Lessons: a) Try simple baselines
b) Look at multiple evaluation metrics

Rand Index

Consider pairwise decisions

		Ground truth: same	Ground truth: different
Clustering	same	TP	FP
Clustering	different	FN	TN

Now can compute P, R, F_1

Accuracy in this table called: *Rand Index*

5. Other issues in evaluation

Ceiling effects

Decision tree	97%
AdaBoost	98%
Mystery algorithm	96%

Ceiling effects

Decision tree	97%
AdaBoost	98%
Mystery algorithm	96%

Moral: If your test set is too easy, it won't tell you anything about the algorithms.
Always compare to simpler baselines to evaluate how easy (or hard) the testing problem is.

Ceiling effects

Decision tree	97%
AdaBoost	98%
Mystery algorithm	96%

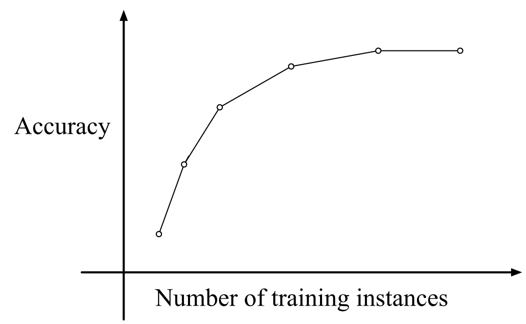
Moral: If your test set is too easy, it won't tell you anything about the algorithms.
Always compare to simpler baselines to evaluate how easy (or hard) the testing problem is.
Always ask yourself what chance performance would be.

Floor effects

- Similarly, your problem could be so hard that no algorithm does well.
- Example: Stock picking. Here there are no experts.
- One way to get at this is inter-annotator agreement.

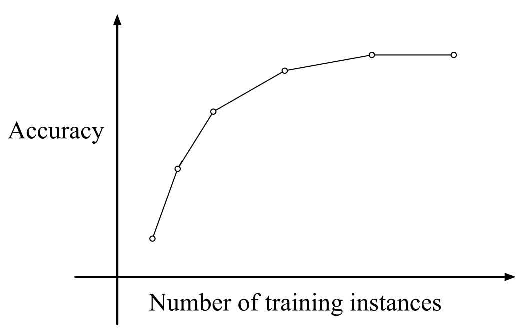
Learning Curves

- It can be interesting to look at how learning performance differs as you get more data.
- This can tell you whether it's worth spending money to gather more data.
- Some algorithms are better with small training sets, but worse with large ones.



Learning curves usually have this shape. Why?

Learning Curves



Learning curves usually have this shape. Why?

You learn "easy" information from the first few examples

(e.g., word "Viagra" usually means the email is spam)

References

- Dietterich, T. G., (1998). Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. Neural Computation, 10 (7) 1895-1924
- Fawcett, T. (2003). ROC Graphs: Notes and Practical Considerations for Researchers Tom Fawcett. HP Labs Tech Report HPL-2003-4. http://home.comcast.net/~tom.fawcett/public_html/papers/ROC101.pdf
- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, (2008). Introduction to Information Retrieval, Cambridge University Press. http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html