Machine Learning for Language Technology 2015

http://stp.lingfil.uu.se/~santinim/ml/2015/ml4lt_2015.htm

Machine Learning in Practice (1)

Marina Santini
santinim@stp.lingfil.uu.se

Department of Linguistics and Philology
Uppsala University, Uppsala, Sweden

Autumn 2015

Acknowledgements

- Weka's slides
- Witten et al. (2011): Ch 5 (156-180)
- Daume' III (2015): ch 4 pp. 65-67.

Outline

- Comparing schemes: the t-test
- Predicting probabilities
- Cost-sensitive measures
- Occam's razor

Comparing data mining schemes

- Frequent question: which of two learning schemes performs better?
- Note: this is domain dependent!
- Obvious way: compare 10-fold CV estimates
- Generally sufficient in applications (we don't loose if the chosen method is not truly better)
- However, what about machine learning research?
  - Need to show convincingly that a particular method works better

Comparing schemes II

- Want to show that scheme A is better than scheme B in a particular domain
  - For a given amount of training data
  - On average, across all possible training sets
- Let's assume we have an infinite amount of data from the domain:
  - Sample infinitely many dataset of specified size
  - Obtain cross-validation estimate on each dataset for each scheme
  - Check if mean accuracy for scheme A is better than mean accuracy for scheme B

Paired t-test

- In practice we have limited data and a limited number of estimates for computing the mean
- *Student's t-test* tells whether the means of two samples are significantly different
- In our case the samples are cross-validation estimates for different datasets from the domain
- Use a *paired* t-test because the individual samples are paired
  - The same CV is applied twice

William Gosset

Born: 1876 in Canterbury; Died: 1937 in Beaconsfield, England

Obtained a post as a chemist in the Guinness brewery in Dublin in 1899. Invented the t-test to handle small samples for quality control in brewing. Wrote under the name "Student".

Distribution of the means

- $x_1\ x_2\ \ldots\ x_k$
- $y_1\ y_2\ \ldots\ y_k$

- m_x and m_y are the means

- With enough samples, the mean of a set of independent samples is normally distributed

- Estimated variances of the means are σ_x^2/k and σ_y^2/k

- If μ_x and μ_y are the true means then → → → $\frac{m_x-\mu_x}{\sqrt{\sigma_x^2/k}}\ \frac{m_y-\mu_y}{\sqrt{\sigma_y^2/k}}$ are *approximately* normally distributed with mean 0, variance 1

Student's distribution

- With small samples ($k < 100$) the mean follows *Student's distribution with k–1 degrees of freedom*
- Confidence limits:

Assuming we have 10 estimates

9 degrees of freedom

Pr[$X \geq z$]	z
0.1%	4.30
0.5%	3.25
1%	2.82
5%	1.83
10%	1.38
20%	0.88

normal distribution

Pr[$X \geq z$]	z
0.1%	3.09
0.5%	2.58
1%	2.33
5%	1.65
10%	1.28
20%	0.84

Distribution of the differences

- Let $m_d = m_x - m_y$
- The difference of the means (m_d) also has a Student's distribution with k–1 degrees of freedom
- The standardized version of m_d is called the t-statistic: $$t = \frac{m_d}{\sqrt{\sigma_d^2 / k}}$$
- We use t to perform the t-test
- σ_d^2 = the variance of the difference samples

Performing the test

- Fix a significance level
  - If a difference is significant at the α% level, there is a (100-α)% chance that the true means differ
- Divide the significance level by two because the test is two-tailed
  - i.e. the true difference can be +ve or – ve
- Look up the value for z that corresponds to $\alpha/2$
- If $t \leq -z$ or $t \geq z$ then the difference is significant
  - I.e. the *null hypothesis* (that the difference is zero) can be rejected

Unpaired observations

- If the CV estimates are from different datasets, they are no longer paired (or maybe we have *k* estimates for one scheme, and *j* estimates for the other one)
- Then we have to use an *un* paired t-test with min(*k* , *j*) – 1 degrees of freedom
- The estimate of the variance of the difference of the means becomes….:

$$\frac{\sigma_x^2}{k} + \frac{\sigma_y^2}{j}$$

Predicting probabilities

- Performance measure so far: success rate
- Also called *0-1 loss function*:

$$\sum_i \begin{cases} 0 \text{ if prediction is correct} \\ 1 \text{ if prediction is incorrect} \end{cases}$$

- Most classifiers produces class probabilities
- Depending on the application, we might want to check the accuracy of the probability estimates
- 0-1 loss is not the right thing to use in those cases

Quadratic loss function

- $p_1 \ldots p_k$ are probability estimates for an instance
- c is the index of the instance's actual class
- $a_1 \ldots a_k = 0$, except for a_c which is 1

Quadratic loss is: $\Sigma_j (p_j - a_j)^2 = \Sigma_{j!=c} p_j^2 + (a - p_c)^2$

Want to minimize $E[\Sigma_j (p_j - a_j)^2]$

Informational loss function

- The informational loss function is $-\log(p_c)$, where c is the index of the instance's actual class
- Let p_1^* … p_k^* be the true class probabilities
- Then the expected value for the loss function is:

$$-p_1^* \log_2 p_1 - \ldots - p_k^* \log_2 p_k$$

Discussion

- Which loss function to choose?
  - Quadratic loss function takes into account all class probability estimates for an instance
  - Informational loss focuses only on the probability estimate for the actual class

The kappa statistic

- Two confusion matrices for a 3-class problem: actual predictions (left) vs. random predictions (right)

		Predicted class			
		a	*b*	*c*	*total*
Actual class	*a*	88	10	2	*100*
	b	14	40	6	*60*
	c	18	10	12	*40*
	total	*120*	*60*	*20*	

		Predicted class			
		a	*b*	*c*	*total*
Actual class	*a*	60	30	10	*100*
	b	36	18	6	*60*
	c	24	12	4	*40*
	total	*120*	*60*	*20*	

- Number of successes: sum of entries in diagonal (*D*)
- *Kappa* statistic: $$\frac{D_{observed} - D_{random}}{D_{perfect} - D_{random}}$$

  measures relative improvement over random predictions

K statistic: Calculations

- Proportions of the class ”a” = 0.5 (ie 100 instances out of 200 → 50% → 50/100 → 0.5)
- Proportions of the class ”b” = 0.3 (ie 60 instances out of 200 → 30% → 30/100 → 0.3)
- Proportions of the class ”c” = 0.2 (ie 40 instances out of 200 → 20% → 20/100 → 0.2)

Both classifiers (see below) returns 120 a’s, 60 b’s and 20 c’s, but one classifier is random. How much the actual classifier improves on the random classifier?

A classifier **randomly guessing** would return the predictions in the table on the RHS:

0.5*120=60; 0.3*60=18; 0.2*20=4 → 60+18+4 = 82

The actual classifier returns the predictions in the table on the LHS, 140 correct predictions (see diagonal), ie 70% success rate. However: **k statistic = 140-82/200-82 = 58/118=0.49=49%**

- **So the actual success rate of 70% repesents an improvement of 49% on random guessing!**

actual predictions (left) vs. random predictions (right)

		Predicted class			
		a	*b*	*c*	*total*
Actual class	*a*	88	10	2	*100*
	b	14	40	6	*60*
	c	18	10	12	*40*
	total	*120*	*60*	*20*	

		Predicted class			
		a	*b*	*c*	*total*
Actual class	*a*	60	30	10	*100*
	b	36	18	6	*60*
	c	24	12	4	*40*
	total	*120*	*60*	*20*	

$$\frac{D_{observed} - D_{random}}{D_{perfect} - D_{random}}$$

In summary

- A k statistic of 100% (or 1) implies a perfect classifier.
- A k statistic of 0 implies that the classifier provides no information and behaves as if it were guessing randomly.

- The Kappa statistic is used to measure the agreement between predicted and observed categorizations of a dataset, and corrects the agreement that occurs by chance.
- Weka provides the k statistic value to assess the success rate beyond the chance.

Quiz 1: k statistic

		Predicted		
		Red	Green	Blue
Actual	Red	37	1	2
	Green	3	16	11
	Blue	1	12	17
	Total	41	29	30

Our classifier predicts Red 41 times, Green 29 times and Blue 30 times. The **actual numbers** for the sample are: 40 Red, 30 Green and 30 Blue.

Overall, our classifier is right 70% of the time.

Suppose these predictions had been random guesses. Our classifier have been randomly right: 0.4 x 41 + 0.3 x 29 + 0.3 x 30 = 34.1 (random guess)

So the actual success rate of 70% represents an improvement of 35.9% on random guessing.

What is the k statistic for our classifier?

1. 0.54
2. 0.60
3. 0.70

Counting the cost

- In practice, different types of classification errors often incur different costs
- Examples:
  - Promotional mailing
  - Terrorist profiling
    - "Not a terrorist" correct 99.99% of the time, but if you miss 0.01% the cost will be very high
  - Loan decisions
  - etc.
- There are many other types of cost!
  - E.g.: cost of collecting training data

Counting the cost

- The *confusion matrix*:

		Predicted class	
		Yes	No
Actual class	Yes	True positive	False negative
	No	False positive	True negative

Classification with costs

- Two cost matrices:

		Predicted class	
		yes	*no*
Actual class	*yes*	0	1
	no	1	0

		Predicted class		
		a	*b*	*c*
Actual class	*a*	0	1	1
	b	1	0	1
	c	1	1	0

- Success rate is replaced by average cost per prediction
  - Cost is given by appropriate entry in the cost matrix

Cost-sensitive classification

- Can take costs into account when making predictions
  - Basic idea: only predict high-cost class when very confident about prediction
- Given: predicted class probabilities
  - Normally we just predict the most likely class
  - Here, we should make the prediction that minimizes the expected cost
    - Expected cost: dot product of vector of class probabilities and appropriate column in cost matrix
    - Choose column (class) that minimizes expected cost

Cost-sensitive learning

- So far we haven't taken costs into account at training time
- Most learning schemes do not perform cost-sensitive learning
  - They generate the same classifier no matter what costs are assigned to the different classes
  - Example: standard decision tree learner
- Simple methods for cost-sensitive learning:
  - Resampling of instances according to costs
  - Weighting of instances according to costs
- Some schemes can take costs into account by varying a parameter, e.g. naïve Bayes

Lift charts

- In practice, costs are rarely known
- Decisions are usually made by comparing possible scenarios
- Example: promotional mailout to 1,000,000 households
  - Mail to all; 0.1% respond (1000)
  - Data mining tool identifies subset of 100,000 most promising, 0.4% of these respond (400)
    40% of responses for 10% of cost may pay off
  - Identify subset of 400,000 most promising, 0.2% respond (800)
- A *lift chart* allows a visual comparison

Data for a lift chart

Rank	Predicted probability	Actual class	Rank	Predicted probability	Actual class
1	0.95	*yes*	11	0.77	*no*
2	0.93	*yes*	12	0.76	*yes*
3	0.93	*no*	13	0.73	*yes*
4	0.88	*yes*	14	0.65	*no*
5	0.86	*yes*	15	0.63	*yes*
6	0.85	*yes*	16	0.58	*no*
7	0.82	*yes*	17	0.56	*yes*
8	0.80	*yes*	18	0.49	*no*
9	0.80	*no*	19	0.48	*yes*
10	0.79	*yes*	. . .	. . .	. . .

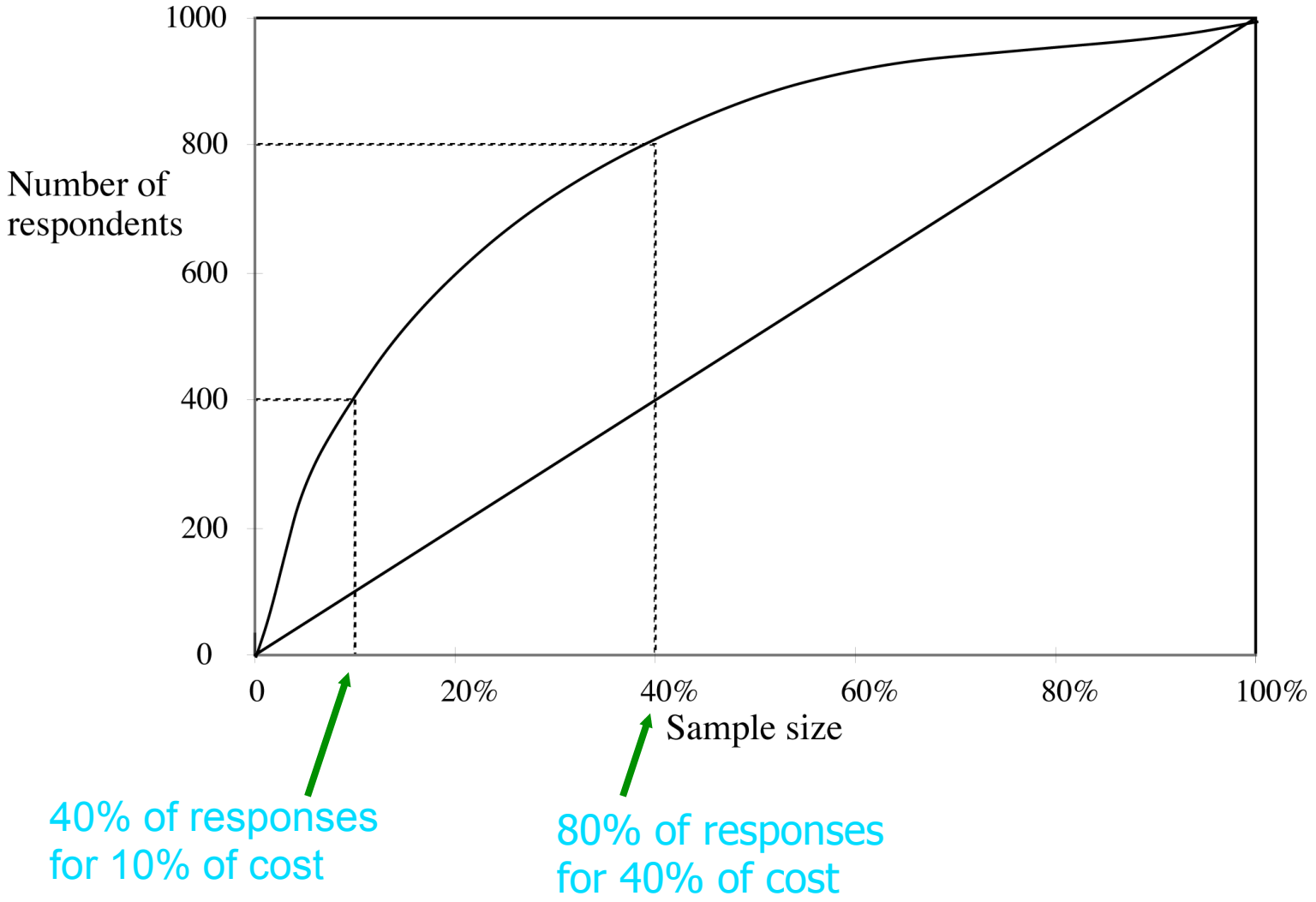
Generating a lift chart

- Sort instances according to predicted probability of being positive:

	Predicted probability	Actual class
1	0.95	Yes
2	0.93	Yes
3	0.93	No
4	0.88	Yes
…	…	…

- *x* axis is sample size
  y axis is number of true positives

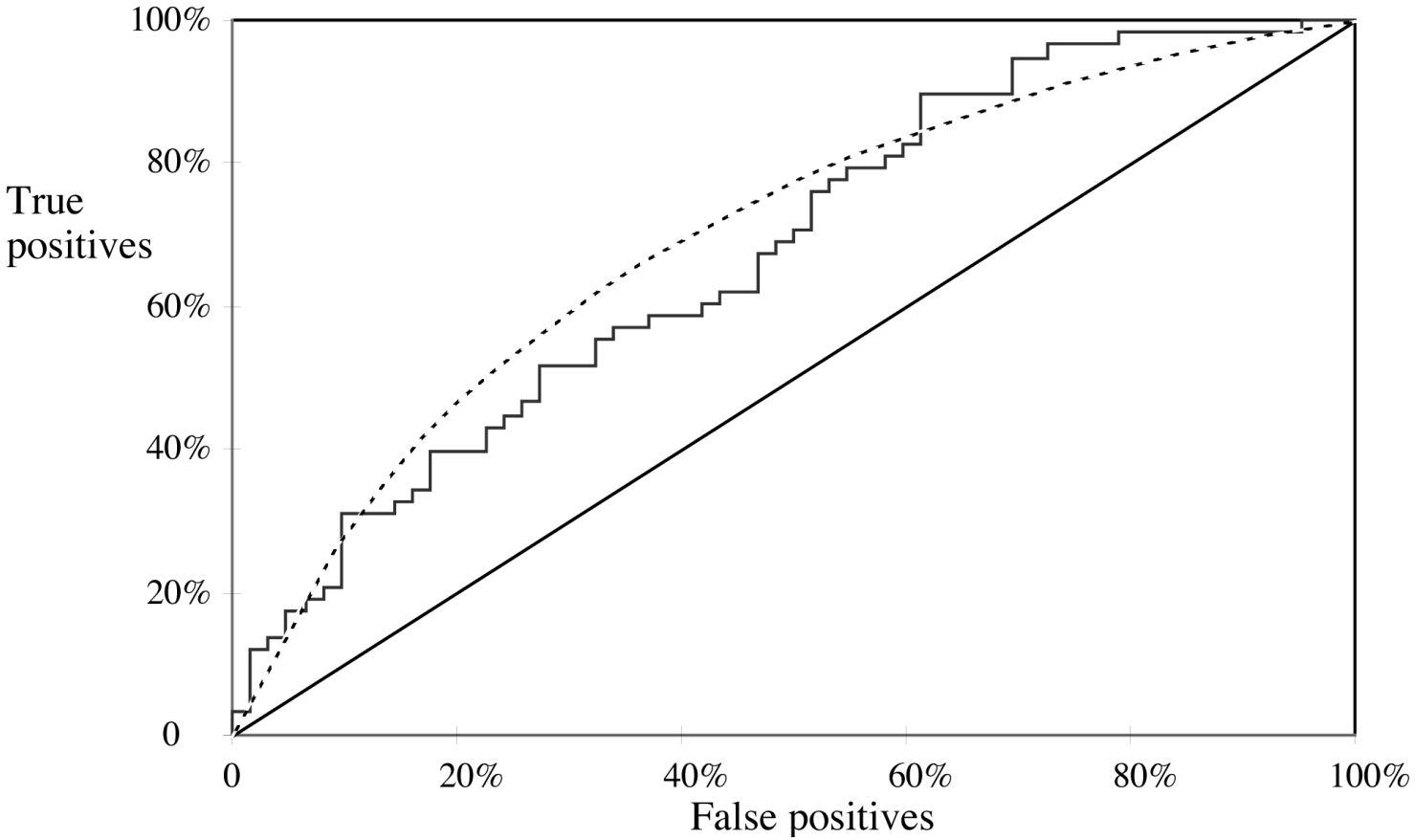
A hypothetical lift chart



ROC curves

- *ROC curves* are similar to lift charts
  - Stands for “receiver operating characteristic”
  - Used in signal detection to show tradeoff between hit rate and false alarm rate over noisy channel
- Differences to lift chart:
  - *y* axis shows percentage of true positives in sample *rather than absolute number*
  - *x* axis shows percentage of false positives in sample *rather than sample size*

A sample ROC curve

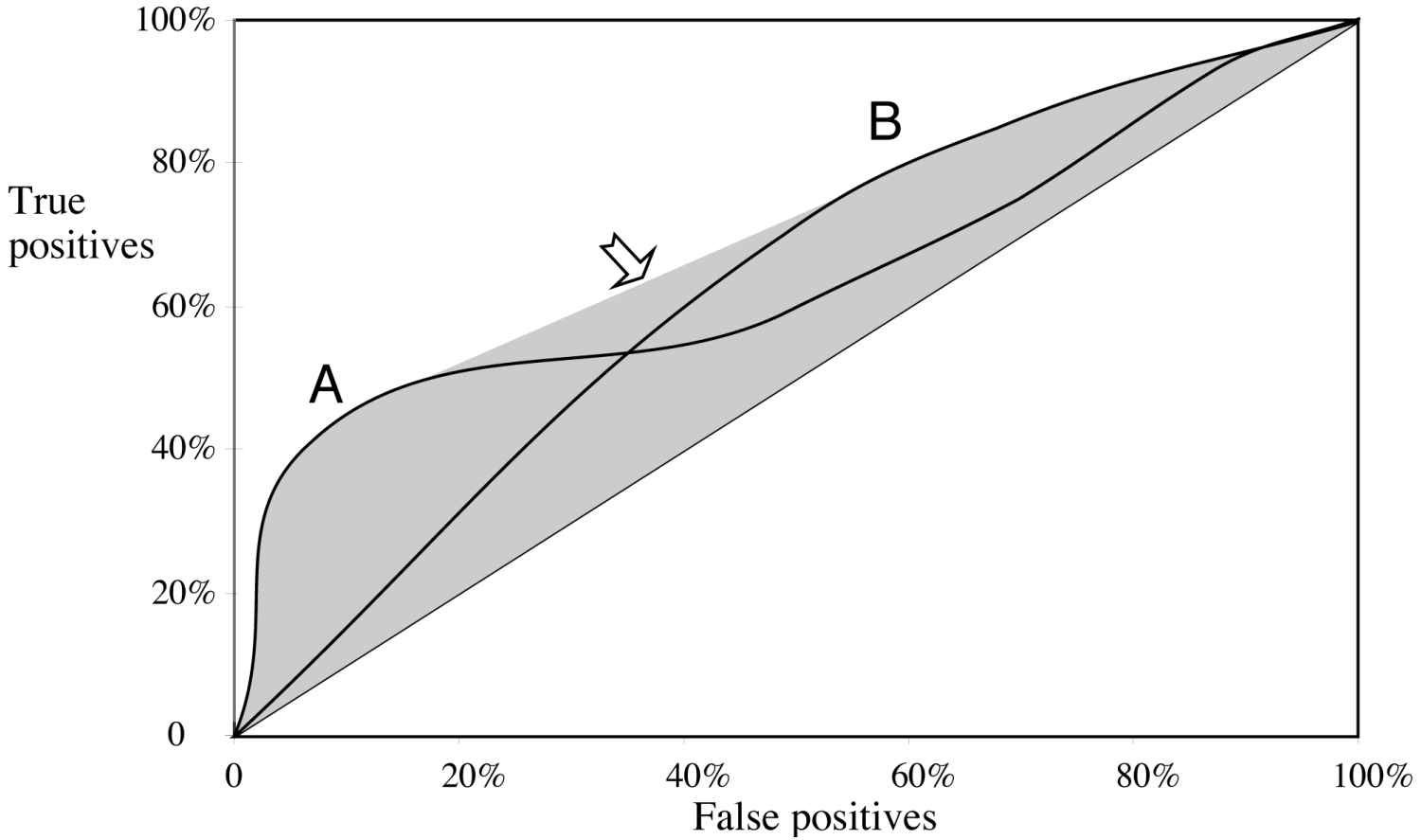


- Jagged curve—one set of test data
- Smooth curve—use cross-validation

Cross-validation and ROC curves

- Simple method of getting a ROC curve using cross-validation:
  - Collect probabilities for instances in test folds
  - Sort instances according to probabilities
- This method is implemented in WEKA
- However, this is just one possibility
  - Another possibility is to generate an ROC curve for each fold and average them

ROC curves for two schemes



- For a small, focused sample, use method A
- For a larger one, use method B
- In between, choose between A and B with appropriate probabilities

Recall-Precision Curves

- Percentage of retrieved documents that are relevant: *precision*=TP/(TP+FP)
- Percentage of relevant documents that are returned: *recall* =TP/(TP+FN)
- Precision/recall curves have hyperbolic shape
- Summary measures: average precision at 20%, 50% and 80% recall (*three-point average recall*)
- *F-measure*=(2 × recall × precision)/(recall+precision)
- *sensitivity* × *specificity* = (TP / (TP + FN)) × (TN / (FP + TN))
- Area under the ROC curve (*AUC*):
  probability that randomly chosen positive instance is ranked above randomly chosen negative one

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.

Elegance vs. errors

- Model 1: very simple, elegant model that accounts for the data almost perfectly
- Model 2: significantly more complex model that reproduces the data without mistakes

- Model 1 is probably preferable.

The End