Lecture 9 - Model Evaluation Techniques

Adnan Ferdous Ashrafi

Stamford University Bangladesh

Table of Contents

Model Evaluation Techniques- Definition

Nestled between the modeling and deployment phases comes the crucial *evaluation phase*, techniques for which are discussed in this Lecture. By the time we arrive at the evaluation phase, the modeling phase has already generated one or more candidate models. It is of critical importance that these models be evaluated for quality and effectiveness before they are deployed for use in the field. Deployment of data mining models usually represents a capital expenditure and investment on the part of the company. If the models in question are invalid, the company's time and money are wasted.

Model Evaluation Techniques- Performance of a classifier

It is often useful to measure the performance of the model on the test set because such a measure provides an unbiased estimate of its generalization error. The accuracy or error rate computed from the test set can also be used to compare the relative performance of different classifiers on the same domain. However, in order to do this, the class labels of the test records must be known. This section reviews some of the methods commonly used to evaluate the performance of a classifier.

Training and Testing

For classification problems, it is natural to measure a classifier's performance in terms of the error rate. The classifier predicts the class of each instance: if it is correct, that is counted as a success; if not, it is an error. The error rate is just the proportion of errors made over a whole set of instances, and it measures the overall performance of the classifier.

Training and Testing- Error Rate on Training

This is a surprising fact, and a very important one.

Error rate on the training set is not likely to be a good indicator of future performance. Why?

Because the classifier has been learned from the very same training data, any estimate of performance based on that data will be optimistic, and may be hopelessly optimistic.

Training and Testing- Testing Set

To predict the performance of a classifier on new data, we need to assess its error rate on a dataset that played no part in the formation of the classifier. This independent dataset is called the test set. We assume that both the training data and the test data are representative samples of the underlying problem.

Training and Testing- Concern

The real problem occurs when there is not a vast supply of data available. In many situations the training data must be classified manually—and so must the test data, of course, to obtain error estimates. This limits the amount of data that can be used for training, validation, and testing, and the problem becomes how to make the most of a limited dataset. From this dataset, a certain amount is held over for testing—this is called the *holdout* procedure—and the remainder is used for training (and, if necessary, part of that is set aside for validation).

There's a dilemma here: to find a good classifier, we want to use as much of the data as possible for training; to obtain a good error estimate, we want to use as much of it as possible for testing.

Predicting Performance- Estimate

Suppose we measure the error of a classifier on a test set and obtain a certain numeric error rate—say 25% so this corresponds to a success rate of 75%. Now, this is only an estimate.

What can you say about the true success rate on the target population? Sure, it's expected to be close to 75%. But how close—within 5%? Within 10%?

It must depend on the size of the test set. Naturally, we would be more confident of the 75% figure if it was based on a test set of 10,000 instances rather than on a test set of 100 instances. But how much more confident would we be?

Predicting Performance- Probability

Imagine that there is a Bernoulli process—a biased coin—whose true (but unknown) success rate is p. Suppose that out of N trials, S are successes: thus the observed success rate is $f = S/N$.

The question is, what does this tell you about the true success rate p?

Predicting Performance- Confidence Intervals

The answer to this question is usually expressed as a confidence interval; that is, p lies within a certain specified interval with a certain specified confidence. For example, if $S = 750$ successes are observed out of $N = 1000$ trials, this indicates that the true success rate must be around 75%. But how close to 75%? It turns out that with 80% confidence, the true success rate p lies between 73.2% and 76.7%. If $S = 75$ successes are observed out of $N = 100$ trials, this also indicates that the true success rate must be around 75%. But the experiment is smaller, and the 80% confidence interval for p is wider, stretching from 69.1% to 80.1%.

Predicting Performance- Confidence Intervals

These figures are easy to relate to qualitatively, but how are they derived quantitatively? We reason as follows: the mean and variance of a single Bernoulli trial with success rate p are p and $p(1-p)$, respectively. If N trials are taken from a Bernoulli process, the expected success rate $f = S/N$ is a random variable with the same mean p; the variance is reduced by a factor of N to $p(1-p)/N$.

For large N, the distribution of this random variable approaches the normal distribution. These are all facts of statistics: we will not go into how they are derived.

Predicting Performance- Confidence Intervals

The probability that a random variable X, with zero mean, lies within a certain confidence range of width $2z$ is

$$P_r[-z \leq X \leq z] = c$$

For a normal distribution, values of c and corresponding values of z are given in tables printed at the back of most statistical texts. However, the tabulations conventionally take a slightly different form: they give the confidence that X will lie outside the range, and they give it for the upper part of the range only:

$$P_r[X \geq z]$$

This is called a one-tailed probability because it refers only to the upper "tail" of the distribution. Normal distributions are symmetric, so the probabilities for the lower tail

$$P_r[X \leq -z]$$

are just the same.

Methods for evaluating the performance of a Classifier- Holdout Method

In the holdout method, the original data with labeled examples is partitioned into two disjoint sets, called the training and the test sets, respectively. A classification model is then induced from the training set and its performance is evaluated on the test set. The proportion of data reserved for training and for testing is typically at the discretion of the analysts (e.g., 50-50 or two- thirds for training and one-third for testing). The accuracy of the classifier can be estimated based on the accuracy of the induced model on the test set.

Holdout Method- Limitations

The holdout method has several well-known limitations. First, fewer labeled examples are available for training because some of the records are withheld for testing. As a result, the induced model may not be as good as when all the labeled examples are used for training. Second, the model may be highly dependent on the composition of the training and test sets. The smaller the training set size, the larger the variance of the model. On the other hand, if the training set is too large, then the estimated accuracy computed from the smaller test set is less reliable. Such an estimate is said to have a wide confidence interval. Finally, the training and test sets are no longer independent of each other. Because the training and test sets are subsets of the original data, a class that is over-represented in one subset will be under-represented in the other, and vice versa.

Random Subsampling

The holdout method can be repeated several times to improve the estimation of a classifier's performance. This approach is known as random subsampling. Let acc_i be the model accuracy during the i^{th} iteration. The overall accuracy is given by $acc_{\text{sub}} = \sum_{i=1}^{k} acc_i/k$.

Random Subsampling- Limitations

Random subsampling still encounters some of the problems associated with the holdout method because it does not utilize as much data as possible for training. It also has no control over the number of times each record is used for testing and training. Consequently, some records might be used for training more often than others.

Cross-Validation

An alternative to random subsampling is cross-validation. In this approach, each record is used the same number of times for training and exactly once for testing. To illustrate this method, suppose we partition the data into two equal-sized subsets. First, we choose one of the subsets for training and the other for testing. We then swap the roles of the subsets so that the previous training set becomes the test set and vice versa. This approach is called a two- fold cross-validation. The total error is obtained by summing up the errors for both runs. In this example, each record is used exactly once for training and once for testing.

Cross-Validation- k-fold cross-validation

The k-fold cross-validation method generalizes this approach by segmenting the data into k equal-sized partitions. During each run, one of the partitions is chosen for testing, while the rest of them are used for training. This procedure is repeated k times so that each partition is used for testing exactly once. Again, the total error is found by summing up the errors for all k runs.

Cross-Validation- Leave one out - k-fold cross-validation

A special case of the k-fold cross-validation method sets k = N , the size of the data set. In this so-called leave-one-out approach, each test set contains only one record. This approach has the advantage of utilizing as much data as possible for training. In addition, the test sets are mutually exclusive and they effectively cover the entire data set.

Cross-Validation- Limitations

The drawback of this approach is that it is computationally expensive to repeat the procedure N times. Furthermore, since each test set contains only one record, the variance of the estimated performance metric tends to be high.

Bootstrap

The methods presented so far assume that the training records are sampled without replacement. As a result, there are no duplicate records in the training and test sets.

In the bootstrap approach, the training records are sampled with replacement; i.e., a record already chosen for training is put back into the original pool of records so that it is equally likely to be redrawn. If the original data has N records, it can be shown that, on average, a bootstrap sample of size N contains about 63.2% of the records in the original data. This approximation follows from the fact that the probability a record is chosen by a bootstrap sample is $1-(1-1/N)^N$.

Bootstrap

When N is sufficiently large, the probability asymptotically approaches $1 - e^{-1} = 0.632$. Records that are not included in the bootstrap sample become part of the test set. The model induced from the training set is then applied to the test set to obtain an estimate of the accuracy of the bootstrap sample, i . The sampling procedure is then repeated b times to generate b bootstrap samples.

Bootstrap- Variations

There are several variations to the bootstrap sampling approach in terms of how the overall accuracy of the classifier is computed. One of the more widely used approaches is the **0.632 bootstrap**, which computes the overall accuracy by combining the accuracies of each bootstrap sample (i) with the accuracy computed from a training set that contains all the labeled examples in the original data (acc_s):

$$\text{Accuracy}, acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632\times \in_i + 0.368 \times acc_s)$$

Comparing data mining methods- Definition

We often need to compare two different learning methods on the same problem to see which is the better one to use. It seems simple: estimate the error using cross-validation (or any other suitable estimation procedure), perhaps repeated several times, and choose the scheme whose estimate is smaller. This is quite sufficient in many practical applications: if one method has a lower estimated error than another on a particular dataset, the best we can do is to use the former method's model.

Comparing data mining methods- Definition

However, it may be that the difference is simply caused by estimation error, and in some circumstances it is important to determine whether one scheme is really better than another on a particular problem. This is a standard challenge for machine learning researchers. If a new learning algorithm is proposed, its proponents must show that it improves on the state of the art for the problem at hand and demonstrate that the observed improvement is not just a chance effect in the estimation process.

Cost of making wrong decisions- Definition

The evaluations that have been discussed so far do not take into account the cost of making wrong decisions, wrong classifications. Optimizing classification rate without considering the cost of the errors often leads to strange results.

Cost of making wrong decisions- Examples

Examples in which errors cost different amounts include loan deci- sions: the cost of lending to a defaulter is far greater than the lost-business cost of refusing a loan to a nondefaulter.

And oil-slick detection: the cost of failing to detect an environment-threatening real slick is far greater than the cost of a false alarm. And load forecasting: the cost of gearing up electricity generators for a storm that doesn't hit is far less than the cost of being caught completely unprepared.

And diagnosis: the cost of misidentifying problems with a machine that turns out to be free of faults is less than the cost of overlooking problems with one that is about to fail.

Cost of making wrong decisions- Examples

And promotional mailing: the cost of sending junk mail to a household that doesn't respond is far less than the lost-business cost of not sending it to a household that would have responded.

In truth, it would be hard pressed to find an application in which the costs of different kinds of error were the same.

Predictive Outcomes- Two-class Predictions

In the two-class case with classes *yes* and *no*, lend or not lend, mark a suspicious patch as an oil slick or not, and so on, a single prediction has the four different possible outcomes shown in Figure 1.

The **true positives (TP)** and **true negatives (TN)** are correct classifications.

A **false positive (FP)** occurs when the outcome is incorrectly predicted as yes (or positive) when it is actually no (negative).

A **false negative (FN)** occurs when the outcome is incorrectly predicted as negative when it is actually positive.

Predictive Outcomes- Two-class Predictions

		Predicted class: yes	Predicted class: no
Actual class	yes	true positive	false negative
	no	false positive	true negative

Figure 1: Different outcomes of a two-class prediction

Predictive Outcomes- Two-class Predictions

The **true positive rate** is TP divided by the total number of positives, which is TP + FN;

the **false positive rate** is FP divided by the total number of negatives, FP + TN.

The **overall success rate** is the number of correct classifications divided by the total number of classifications:

$$\frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}$$

Finally, the error rate is one minus this.

Predictive Outcomes- Multiclass Predictions

In a multiclass prediction, the result on a test set is often displayed as a two- dimensional **confusion matrix** with a row and column for each class. Each matrix element shows the number of test examples for which the actual class is the row and the predicted class is the column.

Good results correspond to large numbers down the main diagonal and small, ideally zero, off-diagonal elements.

Figure 2 shows a numeric example with three classes. In this case the test set has 200 instances (the sum of the nine numbers in the matrix), and 88 + 40 + 12 = 140 of them are predicted correctly, so the success rate is 70%.

Predictive Outcomes- Multiclass Predictions

		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	

(a)

		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	

(b)

Figure 2: Different outcomes of a three-class prediction: (a) actual and (b) expected.

Cost of making wrong decisions- Cost-sensitive classification

If the costs are known, they can be incorporated into a financial analysis of the decision-making process. In the two-class case, in which the confusion matrix is like that of Figure 2, the two kinds of error—false positives and false negatives—will have different costs; likewise, the two types of correct classification may have different benefits. In the two-class case, costs can be summarized in the form of a 2×2 matrix in which the diagonal elements represent the two types of correct classification and the off-diagonal elements represent the two types of error. In the multiclass case this generalizes to a square matrix whose size is the number of classes, and again the diagonal elements represent the cost of correct classification. Figure 3 (a) and (b) shows default cost matrices for the two- and three-class cases whose values simply give the number of errors: misclassification costs are all 1.

Cost of making wrong decisions- Cost-sensitive classification

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

(a)

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

(b)

Figure 3: Default cost matrixes: (a) a two-class case and (b) a three-class case.

Cost of making wrong decisions- Cost-sensitive classification

Given a cost matrix, you can calculate the cost of a particular learned model on a given test set just by summing the relevant elements of the cost matrix for the model's prediction for each test instance. Here, the costs are ignored when making predictions, but taken into account when evaluating them.

Cost of making wrong decisions- Cost-sensitive learning

We have seen how a classifier, built without taking costs into consideration, can be used to make predictions that are sensitive to the cost matrix. In this case, costs are ignored at training time but used at prediction time. An alternative is to do just the opposite: take the cost matrix into account during the training process and ignore costs at prediction time. In principle, better performance might be obtained if the classifier were tailored by the learning algorithm to the cost matrix.

Cost of making wrong decisions- Cost-sensitive learning

In the two-class situation, there is a simple and general way to make any learning method cost sensitive. The idea is to generate training data with a different proportion of yes and no instances.

Cost of making wrong decisions- Cost-sensitive learning

Suppose that you artificially increase the number of no instances by a factor of 10 and use the resulting dataset for training. If the learning scheme is striving to minimize the number of errors, it will come up with a decision structure that is biased toward avoiding errors on the no instances, because such errors are effectively penalized 10-fold. If data with the original proportion of no instances is used for testing, fewer errors will be made on these than on yes instances—that is, there will be fewer false positives than false negatives—because false positives have been weighted 10 times more heavily than false negatives.

Varying the proportion of instances in the training set is a general technique for building cost-sensitive classifiers.

Lift Charts- Definition

Lift charts and gains charts are graphical evaluative methods for assessing and com- paring the usefulness of classification models. Lift is a concept, originally from the marketing field, which seeks to compare the response rates with and without using the classification model.

We define lift as the proportion of positive hits in the set of the model's positive classifications, divided by the proportion of positive hits in the data set overall:

$$\text{lift} = \frac{\text{proportion of positive hits in set of positive classifications}}{\text{proportion of positive hits in data set as a whole}}$$

Lift Charts- Example

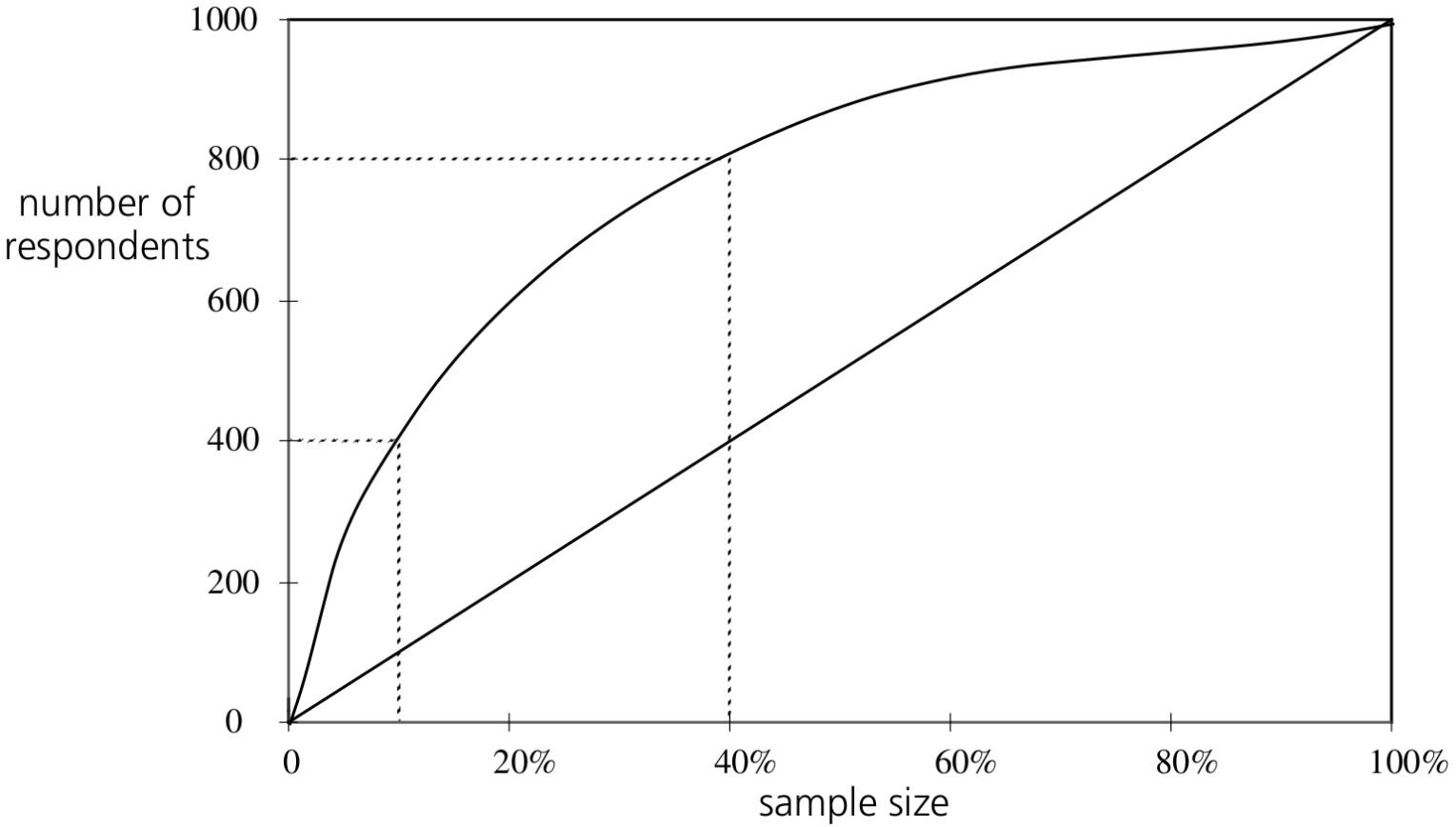

Figure 4: A hypothetical lift chart.

Lift Charts- Ideal Scenario

Where you'd like to be in a lift chart is near the upper left-hand corner: at the very best, 1000 responses from a mailout of just 1000, where you send only to those households that will respond and are rewarded with a 100% success rate. Any selection procedure worthy of the name will keep you above the diagonal—otherwise, you'd be seeing a response that was worse than for random sampling. So the operating part of the diagram is the upper triangle, and the farther to the northwest the better.

ROC curves- Definition

Lift charts are a valuable tool, widely used in marketing. They are closely related to a graphical technique for evaluating data mining schemes known as ROC curves, which are used in just the same situation as the preceding one, in which the learner is trying to select samples of test instances that have a high proportion of positives. The acronym stands for **receiver operating characteristic**, a term used in signal detection to characterize the trade-off between hit rate and false alarm rate over a noisy channel.

ROC curves depict the performance of a classifier without regard to class distribution or error costs.

ROC curves- Example

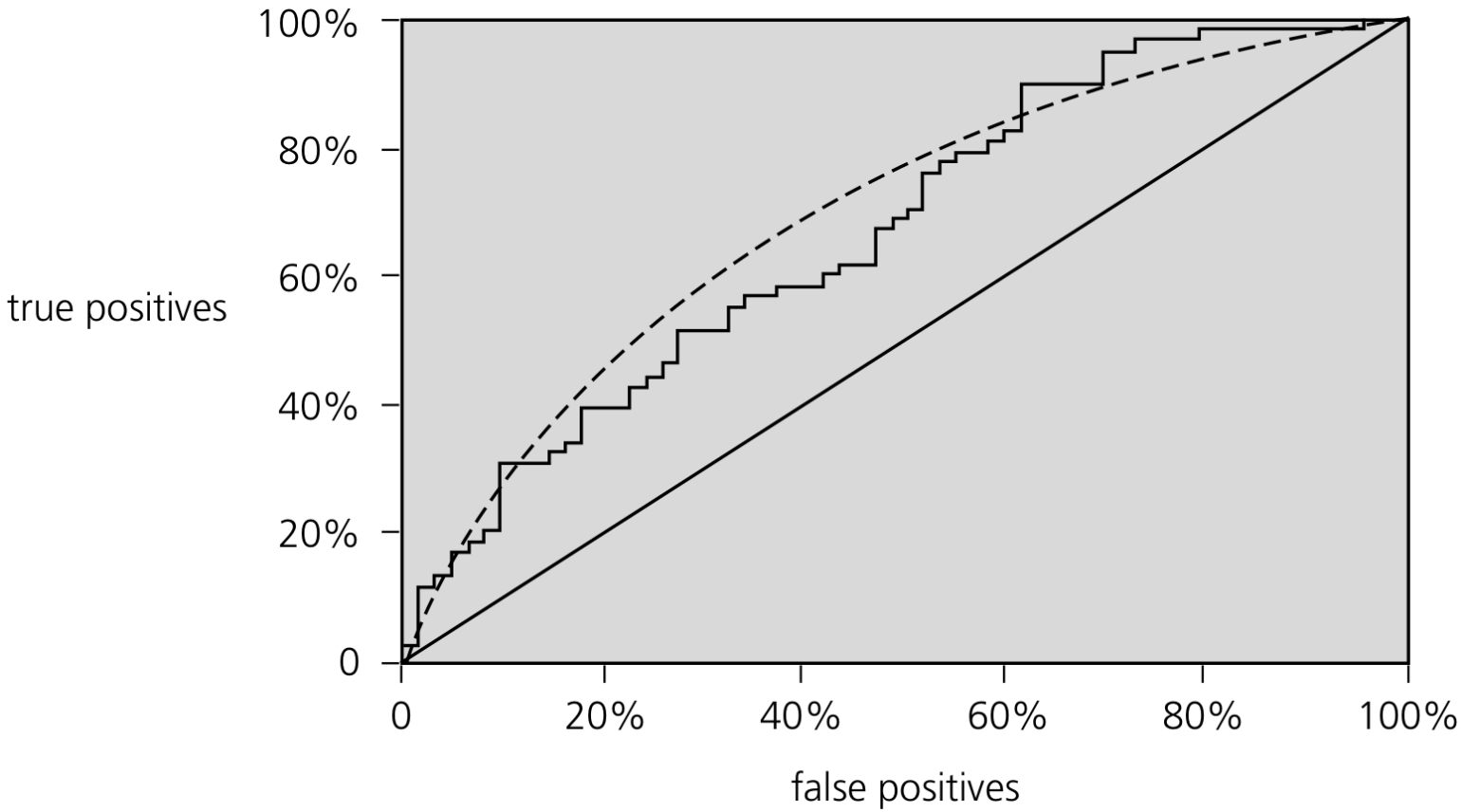

Figure 5: A sample ROC curve.

ROC curves- Comparing Models

It is instructive to look at cross-validated ROC curves obtained using different learning methods. For example, in Figure 6, method A excels if a small, focused sample is sought; that is, if you are working toward the left-hand side of the graph. Clearly, if you aim to cover just 40% of the true positives you should choose method A, which gives a false positive rate of around 5%, rather than method B, which gives more than 20% false positives. But method B excels if you are planning a large sample: if you are covering 80% of the true positives, method B will give a false positive rate of 60% as compared with method A's 80%.

The shaded area is called the **convex hull** of the two curves, and you should always operate at a point that lies on the upper boundary of the convex hull.

ROC curves- Comparing Models- Example

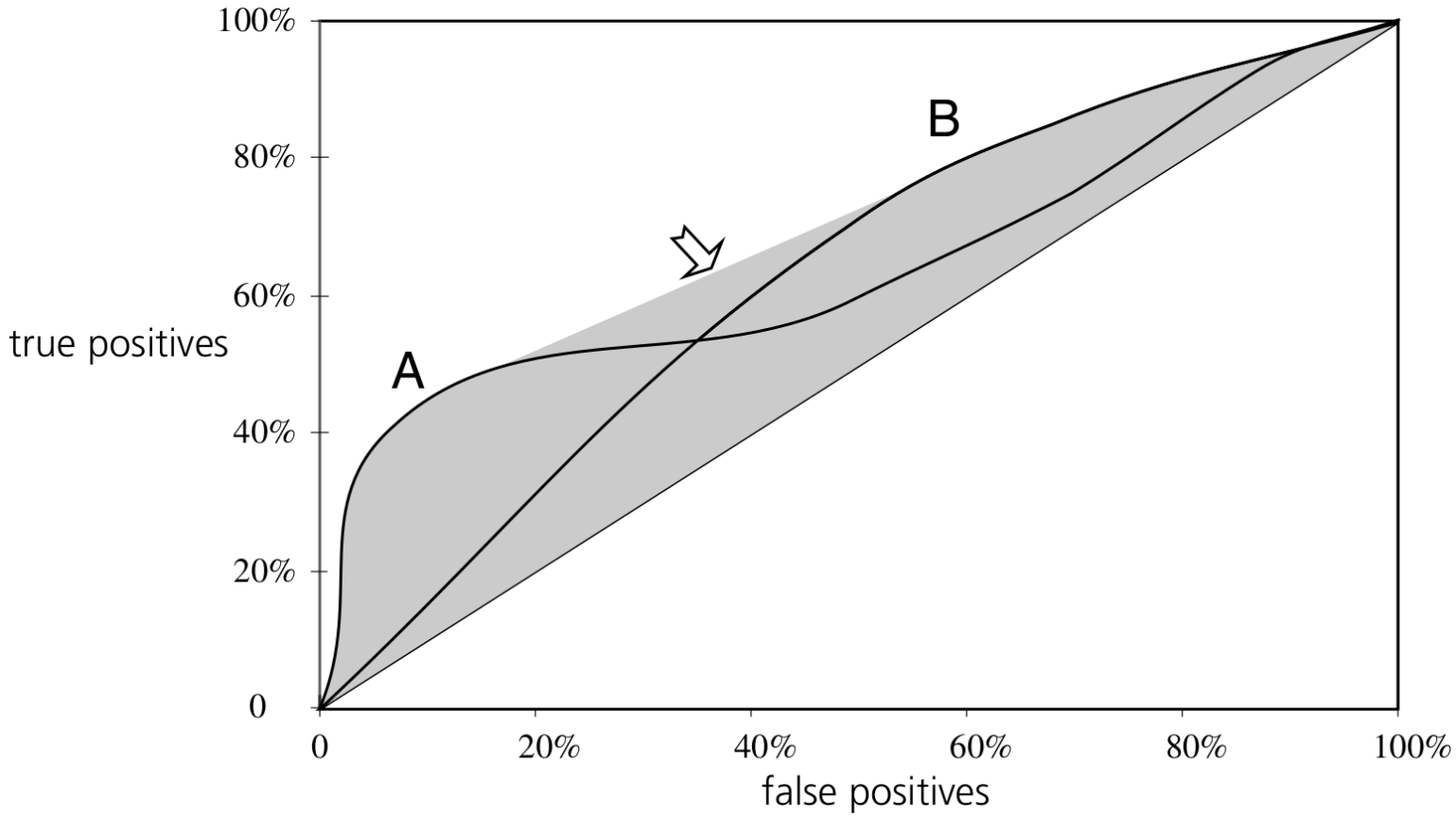

Figure 6: ROC curves for two learning methods.

Recall–precision curves- Definition

Information retrieval researchers define parameters called *recall* and *precision*:

$$\text{recall} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are relevant}}$$

$$\text{precision} = \frac{\text{number of documents retrieved that are relevant}}{\text{total number of documents that are retrieved}}$$

For example, if the list of yes's and no's in Figure 7 represented a ranked list of retrieved documents and whether they were relevant or not, and the entire collection contained a total of 40 relevant documents, then "recall at 10" would refer to recall for the top ten documents, that is, 8/40 = 5%; while "precision at 10" would be 8/10 = 80%. Information retrieval experts use recall–precision curves that plot one against the other, for different numbers of retrieved documents, in just the same way as ROC curves and lift charts—except that because the axes are different, the curves are hyperbolic in shape and the desired operating point is toward the upper right.

Recall–precision curves- Definition

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*	. . .	. . .	. . .

Figure 7: Data for a lift chart.

Recall–precision curves- Example

	Domain	Plot	Axes	Explanation of axes
lift chart	marketing	TP vs. subset size	TP	number of true positives
			subset size	$\frac{TP+FP}{TP+FP+TN+FN} \times 100\%$
ROC curve	communications	TP rate vs. FP rate	TP rate	$tp = \frac{TP}{TP+FN} \times 100\%$
			FP rate	$fp = \frac{FP}{FP+TN} \times 100\%$
recall–precision curve	information retrieval	recall vs. precision	recall	same as TP rate *tp*
			precision	$\frac{TP}{TP+FP} \times 100\%$

Figure 8: Different measures used to evaluate the false positive versus the false negative trade-off.

Further Reading

- Chapter 4.8 of Data Mining - Practical Machine Learning Tools and Techniques, Second Edition - Ian H. Witten, Eibe Frank
- Chapter 8 of DISCOVERING KNOWLEDGE IN DATA - An Introduction to Data Mining - DANIEL T. LAROSE
- Chapter 8 of Introduction to Data Mining (Second Edition) - Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar

Thank you.
Any Questions?