CS570: Introduction to Data Mining

Classification Advanced

Reading: Chapter 8.4 & 8.5 Han, Chapters 4.5 & 4.6 Tan

Anca Doloc-Mihu, Ph.D.

Slides courtesy of Li Xiong, Ph.D.,

Classification and Prediction

- Last lecture
  - Overview
  - Decision tree induction
  - Bayesian classification
- Today
  - Training (learning) Bayesian network
  - kNN classification and collaborative filtering
  - Support Vector Machines (SVM)
  - Neural Networks
  - Regression
  - Model evaluation
  - Rule based methods
- Upcoming lectures
  - Ensemble methods
  - Bagging, Random Forests, AdaBoost

Model Evaluation

- Metrics for Performance Evaluation of a Classifier
- Methods for Model Comparison (selecting the best classifier)
- Methods for Performance Evaluation of a Classifier

Metrics for Performance Evaluation

- Accuracy (recognition rate)
- Error rate (misclassification rate)
- Sensitivity (recall)
- Specificity
- Precision
- F1 score, F-measure or F-score

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
- **Accuracy** of a classifier: percentage of test set tuples that are correctly classified by the model – limitations?
  - Binary classification:

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

  - **Error rate** (misclassification rate) = 1 – accuracy

- Confusion matrix: given *m* classes, *CMi,j*, indicates # of tuples in class *i* that are labeled by the classifier as class *j*
  - Binary classification confusion matrix

	PREDICTED CLASS		
ACTUAL CLASS		positive	negative
	positive	TP	FN
	negative	FP	TN

TP (true positive)

FN (false negative)

FP (false positive)

TN (true negative)

Limitation of Accuracy

- 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

Accuracy is most effective when the class distribution is relatively balanced.

Cost-Sensitive Measures

$$\text{False Positive Rate (FPR)} = \frac{FP}{FP + TN}$$

$$\text{False Negative Rate (FNR)} = \frac{FN}{TP + FN}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{FP + TN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Recall} + \text{Recall}}$$

	PREDICTED CLASS		
		positive	negative
ACTUAL CLASS	positive	TP	FN
	negative	FP	TN

sensitivity/recall/true positive rate

specificity/true negative rate

precision

Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- **Loss function**: measures the error bw. y_i and the predicted value y_i'
  - Absolute error: $| y_i - y_i'|$
  - Squared error: $(y_i - y_i')^2$
- Test error (generalization error): the average loss over the test set
  - Mean absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i'|}{d}$ Mean squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$
  - Relative absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i'|}{\sum_{i=1}^{d} | y_i - \bar{y}|}$ Relative squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$

The mean squared-error exaggerates the presence of outliers

Popularly use (square) root mean-square error, similarly, root relative squared error

Classifier Accuracy Measures

	C_1	C_2
C_1	True positive	False negative
C_2	False positive	True negative

classes	buy_computer = yes	buy_computer = no	total	recognition(%)
buy_computer = yes	6954	46	7000	99.34
buy_computer = no	412	2588	3000	86.27
total	7366	2634	10000	95.52

- Accuracy of a classifier M, acc(M): percentage of test set tuples that are correctly classified by the model M
  - Error rate (misclassification rate) of M = 1 – acc(M)
- Confusion matrix: given *m* classes, $CM_{i,j}$, indicates # of tuples in class *i* that are labeled by the classifier as class *j*
- Alternative accuracy measures (e.g., for cancer diagnosis)

  sensitivity = truePos/pos /* true positive recognition rate */

  specificity = trueNeg/neg /* true negative recognition rate */

  precision = truePos/(truePos + falsePos)

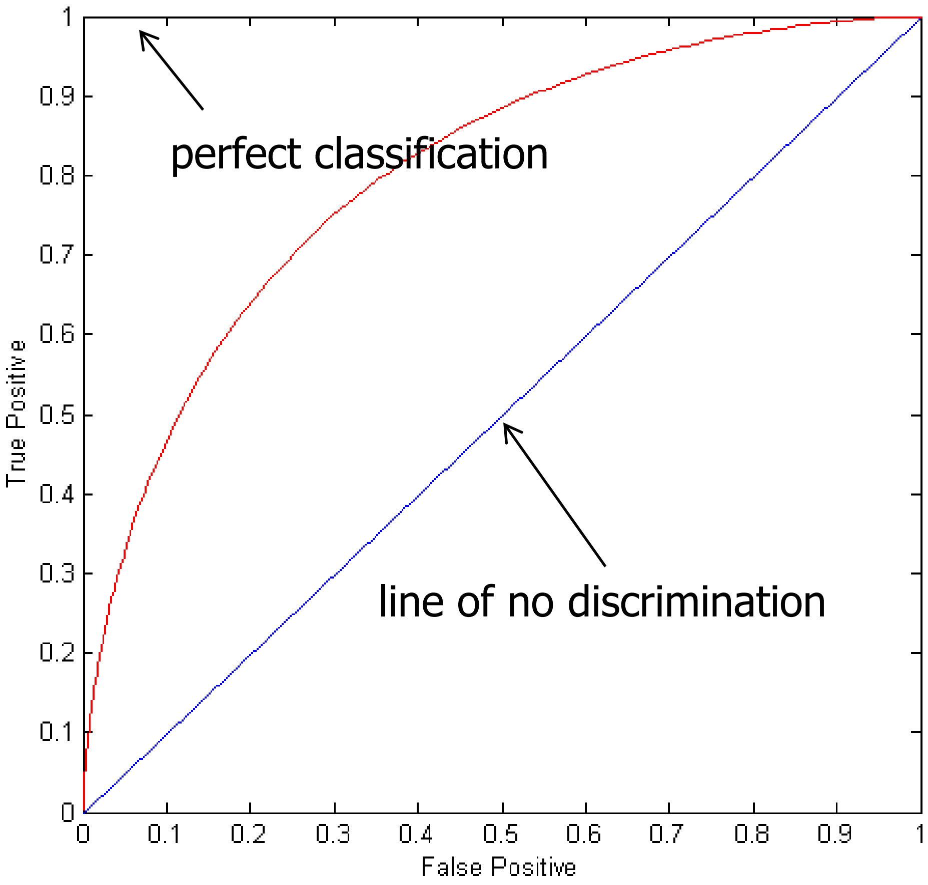
  accuracy = sensitivity * pos/(pos + neg) + specificity * neg/(pos + neg)

Model Evaluation

- Metrics for Performance Evaluation
- Methods for Model Comparison
- Methods for Performance Evaluation

Model Comparison: ROC (Receiver Operating Characteristic)

- From signal detection theory
- True positive rate vs. false positive rate
- Sensitivity vs (1 - specificity)
- Each prediction result represents one point (varying threshold, sample distribution,
- etc)



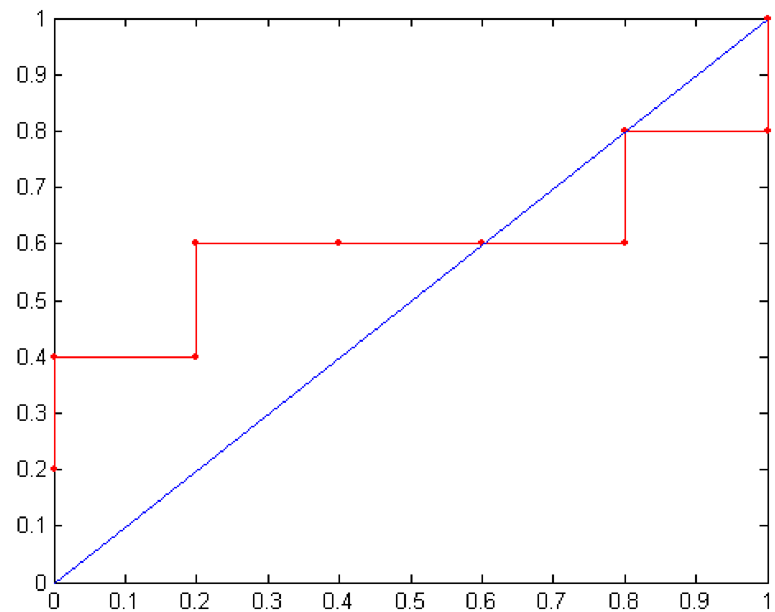
How to Construct an ROC curve

Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

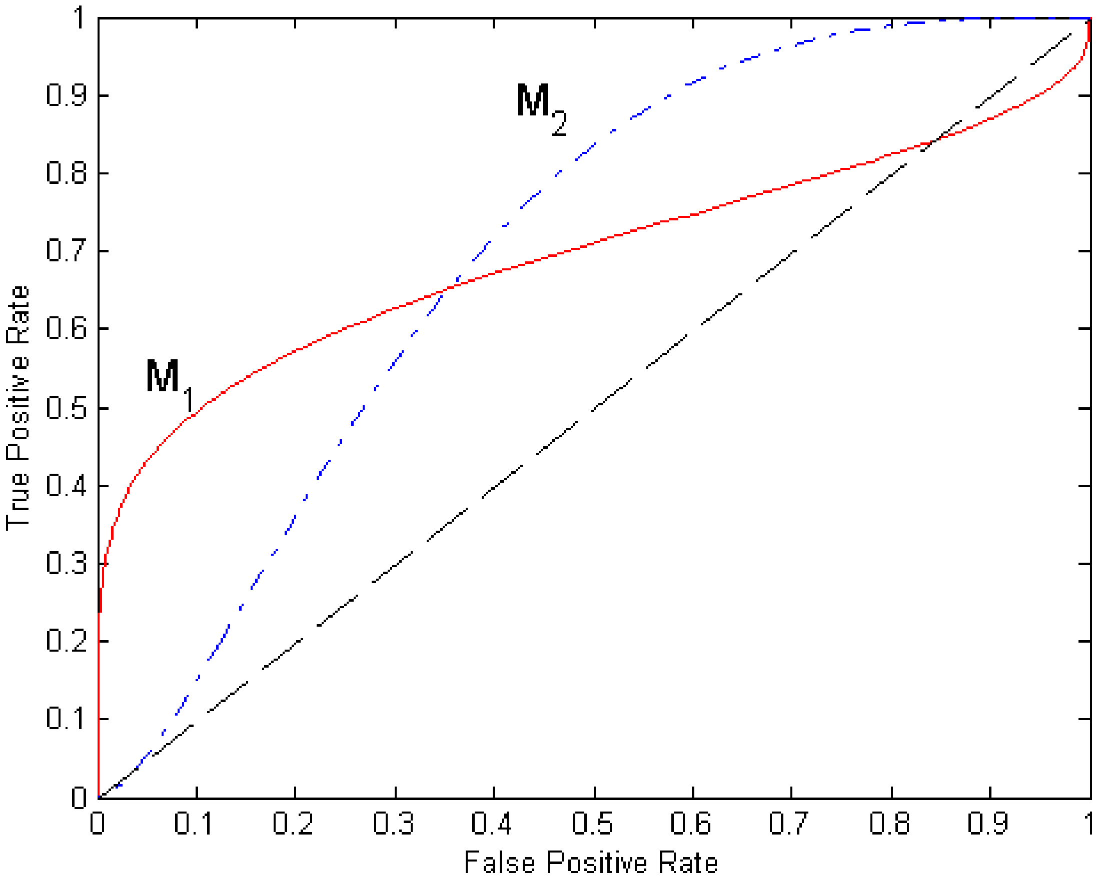
- Sort instances according to posterior probability P(+|A) in decreasing order
- Apply threshold at each unique value of P(+|A)
- Compute and plot TPR and FPR

How to construct an ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0



Using ROC for Model Comparison



- Area Under the ROC curve
  - Ideal: Area = 1
  - Diagonal: Area = 0.5
- M1 vs. M2?

Test of Significance

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances

- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Confidence Interval for Accuracy

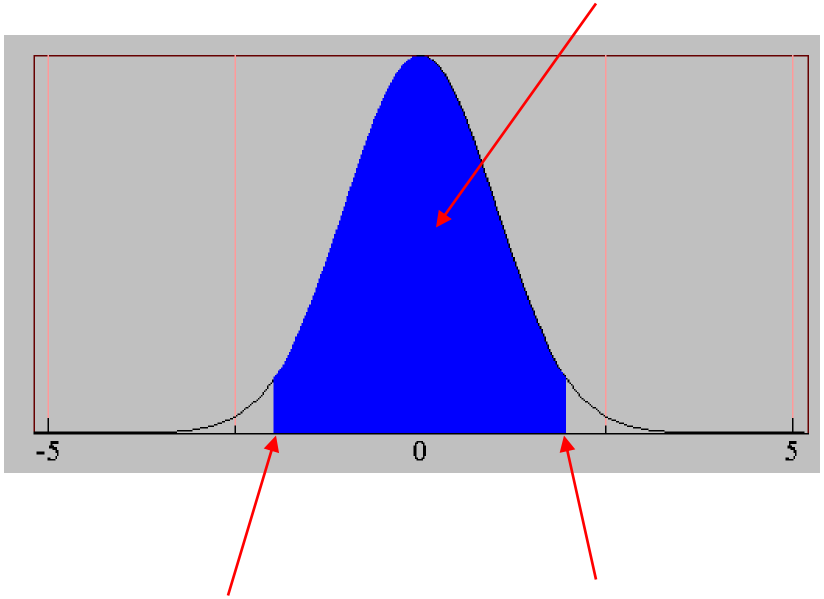
- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
  - Possible outcomes for prediction: correct or wrong
  - Collection of Bernoulli trials has a Binomial distribution

- Given x (# of correct predictions) or equivalently, acc=x/N, and N (# of test instances),

  Can we predict p (true accuracy of model)?

Confidence Interval for Accuracy

- For large test sets (N large),
  - acc has a normal distribution with mean p and variance p(1-p)/N



$$P\left(Z_{\alpha/2} < \frac{acc - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2}\right)$$
$$= 1 - \alpha$$

- Confidence Interval for p:

$$p = \frac{2 \times N \times acc + Z_{\alpha/2}^2 \pm \sqrt{Z_{\alpha/2}^2 + 4 \times N \times acc - 4 \times N \times acc^2}}{2(N + Z_{\alpha/2}^2)}$$

Confidence Interval for Accuracy

- Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:
  - N=100, acc = 0.8
  - Let $1-\alpha = 0.95$ (95% confidence)
  - From probability table, $Z_{\alpha/2}=1.96$

$1-\alpha$	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65

N	50	100	500	1000	5000
p(lower)	0.670	0.711	0.763	0.774	0.789
p(upper)	0.888	0.866	0.833	0.824	0.811

Comparing Performance of 2 Models

- Given two models, say M1 and M2, which is better?
  - M1 is tested on D1 (size=n1), found error rate = e_1
  - M2 is tested on D2 (size=n2), found error rate = e_2
  - *Assume D1 and D2 are independent, is the observed difference bw e_1 and e_2 statistically significant?*
  - If n1 and n2 are sufficiently large, then we can approximate

$$e_1 \sim N(\mu_1, \sigma_1)$$
$$e_2 \sim N(\mu_2, \sigma_2)$$

$$\hat{\sigma}_i = \frac{e_i(1-e_i)}{n_i}$$

Comparing Performance of 2 Models

- To test if performance difference is statistically significant: d = e1 – e2
  - d ~ $N(d_t,\sigma_t)$ where d_t is the true difference
  - Since D1 and D2 are independent, their variance adds up:

$$\sigma_t^2 = \sigma_1^2 + \sigma_2^2 \cong \hat{\sigma}_1^2 + \hat{\sigma}_2^2$$

$$= \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}$$

  - At $(1-\alpha)$ confidence level, $$d_t = d \pm Z_{\alpha/2}\hat{\sigma}_t$$

An Illustrative Example

- Given: M1: $n1 = 30$, $e1 = 0.15$
  M2: $n2 = 5000$, $e2 = 0.25$
- $d = |e2 - e1| = 0.1$ (2-sided test)

$$\hat{\sigma}_d = \frac{0.15(1-0.15)}{30} + \frac{0.25(1-0.25)}{5000} = 0.0043$$

- At 95% confidence level, $Z_{\alpha/2}=1.96$

$$d_t = 0.1 \pm 1.96 \times \sqrt{0.0043} = 0.1 \pm 0.128$$

=> Interval contains 0 => difference may not be statistically significant

Comparing Performance of 2 Algorithms

- Each learning algorithm may produce k models:
  - L1 may produce M11 , M12, …, M1k
  - L2 may produce M21 , M22, …, M2k
- If models are generated on the same test sets D1,D2, …, Dk (e.g., via cross-validation)
  - For each set: compute $d_j = e_{1j} - e_{2j}$
  - d_j has mean d_t and variance σ_t
  - Estimate:

$$\hat{\sigma}_t^2 = \frac{\sum_{j=1}^{k}(d_j - \overline{d})^2}{k(k-1)}$$

$$d_t = d \pm t_{1-\alpha, k-1}\hat{\sigma}_t$$

Model Evaluation

- Metrics for Performance Evaluation
- Methods for Model Comparison
- Methods for Performance Evaluation

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

Methods of Evaluation

- Holdout method
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

- Cross-validation (*k*-fold, where k = 10 is most popular)
  - Randomly partition the data into *k mutually exclusive* subsets, each approximately equal size
  - At *i*-th iteration, use k-1 sets as training set and remaining one as test set
  - Leave-one-out: k folds where k = # of tuples, for small sized data
  - Stratified cross-validation: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data

Evaluating the Accuracy of a Classifier or Predictor (II)

- Bootstrap - Sampling with replacement
  - Works well with small data sets
  - Samples the given training tuples uniformly *with replacement*
    - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
- Several boostrap methods, and a common one is **.632 boostrap**
  - Suppose we are given a data set of d tuples. The data set is sampled d times, with replacement, resulting in a training set of d samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data will end up in the bootstrap, and the remaining 36.8% will form the test set (since $(1 - 1/d)^d \approx e^{-1} = 0.368$)
  - Repeat the sampling procedue k times, overall accuracy of the model:

$$acc(M) = \sum_{i=1}^{k} (0.632 \times acc(M_i)_{test\_set} + 0.368 \times acc(M_i)_{train\_set})$$

Classification and Prediction

- Last lecture
  - Overview
  - Decision tree induction
  - Bayesian classification
- Today
  - Training (learning) Bayesian network
  - kNN classification and collaborative filtering
  - Support Vector Machines (SVM)
  - Neural Networks
  - Regression
  - Model evaluation
  - Rule based methods
- Upcoming lectures
  - Ensemble methods
  - Bagging, Random Forests, AdaBoost

Rule-Based Classifier

- Classify records by a collection of IF-THEN rules
- Basic concepts
  - IF (*Condition*) THEN y
  - (*Condition*) $\rightarrow$ y
    - LHS: rule antecedent or condition
    - RHS: rule consequent
  - E.g. IF *age* = youth AND *student* = yes THEN *buys_computer* = yes
- Using the rules
- Learning the rules

Rule-based Classifier: Example

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$?

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$?

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$?

Assessment of a Rule

- **Coverage** of a rule:
  - Fraction of records that satisfy the antecedent of a rule
  - coverage(R) = n_{covers} /|D| where n_{covers} = # of tuples covered by R and D is the training data set

- **Accuracy** of a rule:
  - Fraction of records that satisfy both the antecedent and consequent of a rule
  - accuracy(R) = $n_{correct}$ / n_{covers} where $n_{correct}$ = # of tuples correctly classified by R

Characteristics of Rule-Based Classifier

- Mutually exclusive rules
  - Classifier contains mutually exclusive rules if the rules are independent of each other
  - Every record is covered by at most one rule

- Exhaustive rules
  - Classifier has exhaustive coverage if it accounts for every possible combination of attribute values
  - Each record is covered by at least one rule

Using the Rules

- Rules that are mutually exclusive and exhaustive
- Rules that are not mutually exclusive
  - A record may trigger more than one rule
  - Solution? – Conflict resolution
    - Ordered rules (decision list) - in decreasing order of their priority
    - Unordered rule set – use voting schemes
- Rules that are not exhaustive
  - A record may not trigger any rules
  - Solution? - Use a default class (rule)

Rule-Based Ordering

- Rule-based ordering
  - Individual rules are ranked based on their quality
  - Rule set is known as a decision list
- Class-based ordering
  - Classes are sorted in order of decreasing importance
  - Rules are sorted by the classes

R1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

R2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

R3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

R5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?

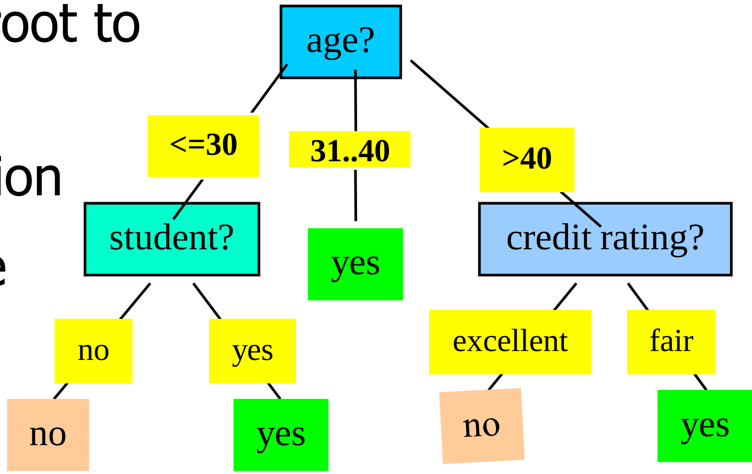
Building Classification Rules

- Indirect Method: Extract rules from other classification models
  - Decision trees. E.g. C4.5 Rules

- Direct Method: Extract rules directly from data
  - Sequential Covering. E.g.: CN2, RIPPER
  - Associative Classification

Rule Extraction from a Decision Tree

- One rule is created for each path from the root to a leaf - each attribute-value pair forms a conjunction, the leaf holds the class prediction
- Rules are mutually exclusive and exhaustive
- Pruning (C4.5): class-based ordering



- Example: Rule extraction from our *buys_computer* decision-tree

IF *age* = young AND *student* = *no* THEN *buys_computer* = *no*

IF *age* = young AND *student* = *yes* THEN *buys_computer* = *yes*

IF *age* = mid-age THEN *buys_computer* = *yes*

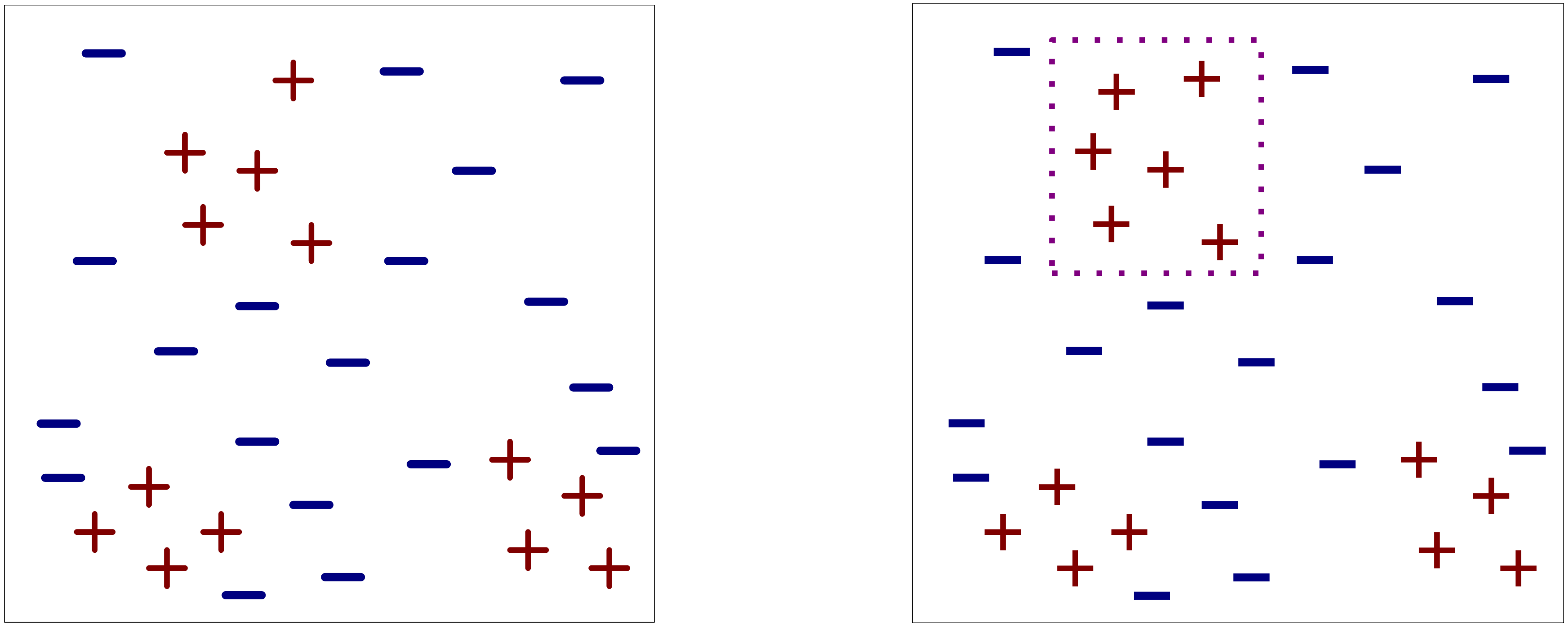
IF *age* = old AND *credit_rating* = *excellent* THEN *buys_computer* = *no*

IF *age* = young AND *credit_rating* = *fair* THEN *buys_computer* = *yes*

Direct Method: Sequential Covering

1. Start from an empty rule
2. Grow a rule using the Learn-One-Rule function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) until stopping criterion is met

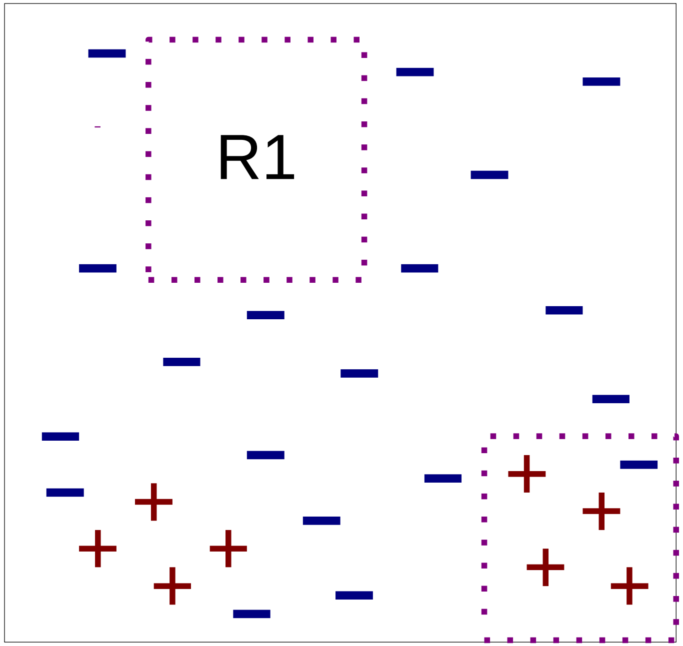
Example of Sequential Covering



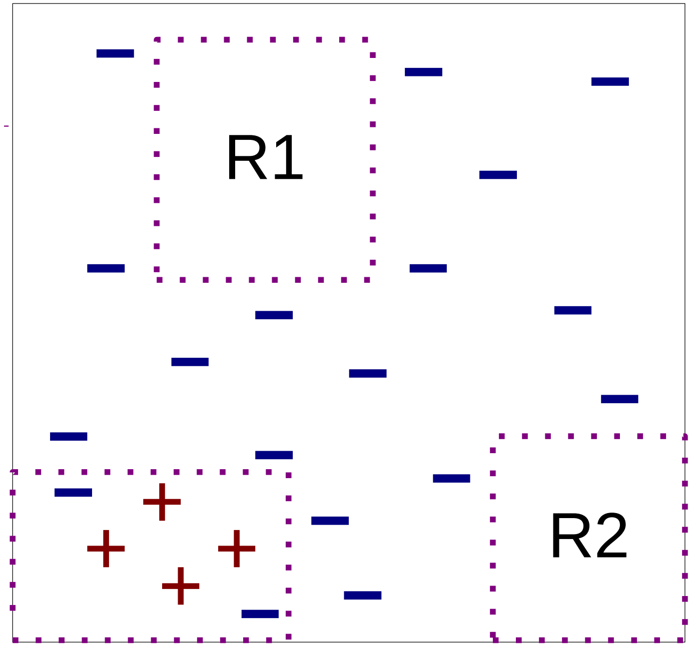
(i) Original Data

(ii) Step 1

Example of Sequential Covering



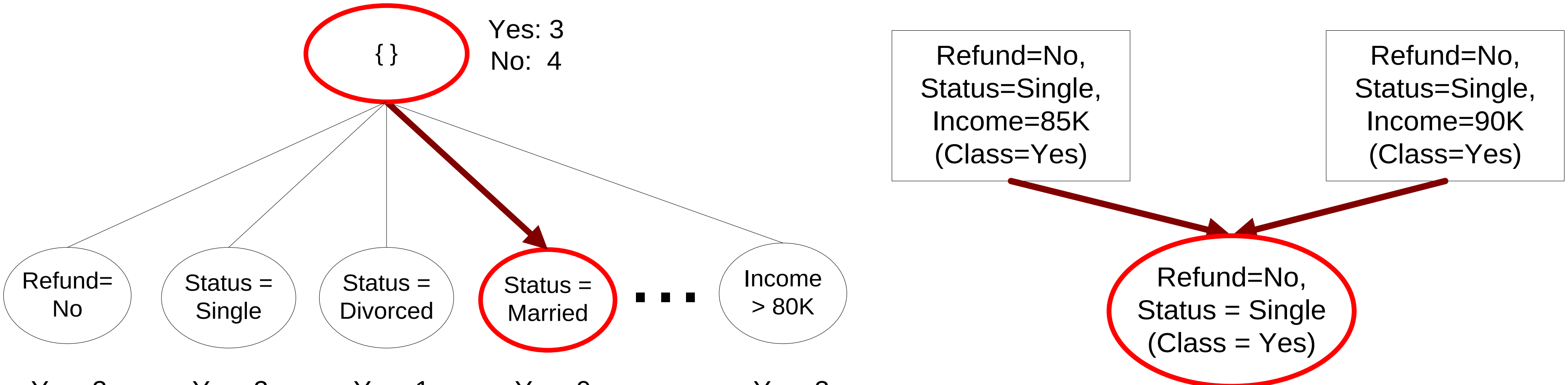
(iii) Step 2



(iv) Step 3

Rule Growing

- Two common strategies



(a) General-to-specific

(b) Specific-to-general

Learn-One-Rule

- Start with the most general rule possible: condition = empty
- Adding new attributes by adopting a greedy depth-first strategy
  - Picks the one that most improves the rule quality
- Rule-Quality measures: consider both coverage and accuracy
  - Foil-gain (in FOIL & RIPPER): assesses info_gain by extending condition

$$FOIL\_Gain = pos' \times (\log_2 \frac{pos'}{pos'+neg'} - \log_2 \frac{pos}{pos+neg})$$

    It favors rules that have high accuracy and cover many positive tuples
- Rule pruning based on an independent set of test tuples

$$FOIL\_Prune(R) = \frac{pos - neg}{pos + neg}$$

    Pos/neg are # of positive/negative tuples covered by R.

    If *FOIL_Prune* is higher for the pruned version of R, then prune R.

Direct Method: Multi-Class

- For 2-class problem, choose one of the classes as positive class, and the other as negative class
  - Learn rules for positive class
  - Negative class will be default class

- For multi-class problem
  - Order the classes according to increasing class prevalence (fraction of instances that belong to a particular class)
  - Learn the rule set for smallest class first, treat the rest as negative class
  - Repeat with next smallest class as positive class

Associative Classification

- Associative classification
  - Search for strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels
  - Classification: Based on evaluating a set of rules in the form of

$$P_1 \wedge p_2 \ldots \wedge p_l \rightarrow \text{“}A_{class} = C\text{”} \text{ (conf, sup)}$$

- Why effective?
  - It explores highly confident associations among multiple attributes and may overcome some constraints introduced by decision-tree induction, which considers only one attribute at a time
  - In many studies, associative classification has been found to be more accurate than some traditional classification methods, such as C4.5

Typical Associative Classification Methods

- CBA (Classification By Association: Liu, Hsu & Ma, KDD'98)
  - Mine association possible rules in the form of
    - Cond-set (a set of attribute-value pairs) → class label
  - Build classifier: Organize rules according to decreasing precedence based on confidence and then support
- CMAR (Classification based on Multiple Association Rules: Li, Han, Pei, ICDM'01)
  - Classification: Statistical analysis on multiple rules
- CPAR (Classification based on Predictive Association Rules: Yin & Han, SDM'03)
  - Generation of predictive rules (FOIL-like analysis)
  - High efficiency, accuracy similar to CMAR
- RCBT (Mining top-*k* covering rule groups for gene expression data, Cong et al. SIGMOD'05)
  - Explore high-dimensional classification, using top-k rule groups
  - Achieve high classification accuracy and high run-time efficiency

Rule-Based Classifiers: Comments

- As highly expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

Classification and Prediction

- Last lecture
  - Overview
  - Decision tree induction
  - Bayesian classification
- Today
  - Training (learning) Bayesian network
  - kNN classification and collaborative filtering
  - Support Vector Machines (SVM)
  - Neural Networks
  - Regression
  - Model evaluation
  - Rule based methods
- Upcoming lectures
  - Ensemble methods
  - Bagging, Random Forests, AdaBoost