BITS Pilani

Pilani|Dubai|Goa|Hyderabad

Introduction to Data Science
Classification and Prediction

- *The slides presented here are obtained from the authors of the books and from various other contributors. I hereby acknowledge all the contributors for their material and inputs.*
- *I have added and modified a few slides to suit the requirements of the course.*

Rule-based Classification

Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Neural Networks
  - computational networks that simulate the decision process in neurons (networks of nerve cell)
- Naïve Bayes and Bayesian Belief Networks
  - uses the probability theory to find the most likely of the possible classifications
- Support Vector Machines
  - fits a boundary to a region of points that are all alike; uses the boundary to classify a new point

Rule-Based Classifier

- Classify records by using a collection of “if…then…” rules

- Rule: (Condition) $\rightarrow$ y where
  - Condition is a conjunctions of attributes
  - y is the class label
  - LHS: rule antecedent or condition
  - RHS: rule consequent
  - Examples of classification rules:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) $\rightarrow$ Evade=No

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) $\rightarrow$ 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

Application of Rule-Based Classifier

- A rule ***r*** **covers** an instance **x** if the attributes of the instance satisfy the condition of the rule

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
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

The rule R1 covers a hawk => Bird

The rule R3 covers the grizzly bear => Mammal

Rule Coverage and Accuracy

- Coverage of a rule:
  - Fraction of records that satisfy the antecedent of a rule
- Accuracy of a rule:
  - Fraction of records that satisfy both the antecedent and consequent of a rule

Tid	Refund	Marital Status	Taxable Income	Class
1	Yes	**Single**	125K	**No**
2	No	Married	100K	**No**
3	No	**Single**	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	**Single**	85K	**Yes**
9	No	Married	75K	**No**
10	No	**Single**	90K	**Yes**

(Status=Single) $\rightarrow$ No

Coverage = 40%, Accuracy = 50%

How does Rule-based Classifier Work?

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
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur triggers rule R3, so it is classified as a mammal

A turtle triggers both R4 and R5

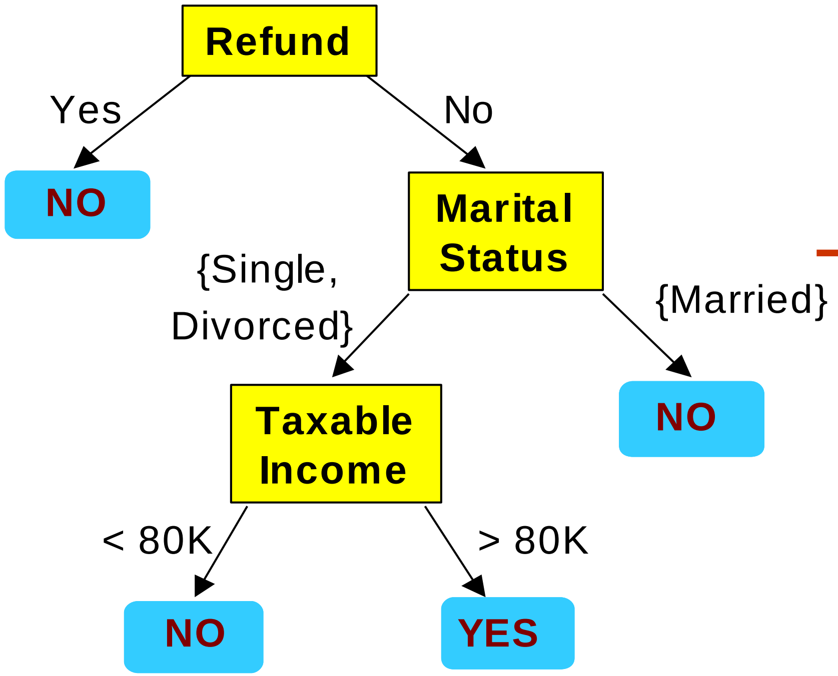
A dogfish shark triggers none of the rules

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

From Decision Trees To Rules



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

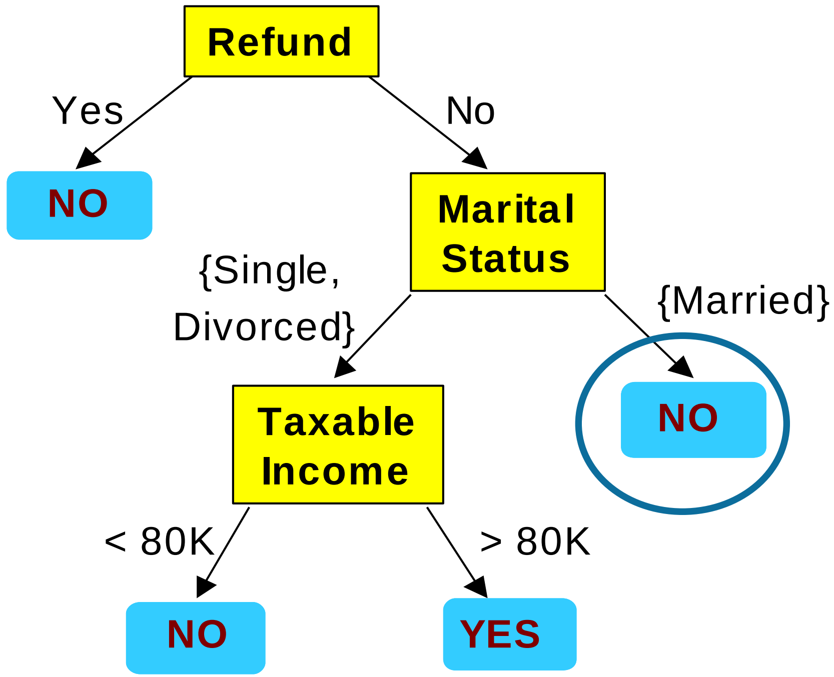
(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree

Rules Can Be Simplified



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Initial Rule: (Refund=No) $\wedge$ (Status=Married) $\rightarrow$ No

Simplified Rule: (Status=Married) $\rightarrow$ No

Further Characterizing Rules

- Rules are not mutually exclusive
  - A record may trigger more than one rule
  - Solution?
    - Ordered rule set
    - Unordered rule set – use voting schemes

- Rules are not exhaustive
  - A record may not trigger any rules
  - Solution?
    - Use a default class

Ordered Rule Set

- Rules are rank ordered according to their priority
  - An ordered rule set is known as a decision list
- When a test record is presented to the classifier
  - It is assigned to the class label of the highest ranked rule it has triggered
  - If none of the rules fired, it is assigned to the default class

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	?

Rule Ordering Schemes

- Rule-based ordering
  - Individual rules are ranked based on their quality
- Class-based ordering
  - Rules that belong to the same class appear together

Rule-based Ordering	**Class-based Ordering**
(Refund=Yes) ==> No	(Refund=Yes) ==> No
(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No	(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No
(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes	(Refund=No, Marital Status={Married}) ==> No
(Refund=No, Marital Status={Married}) ==> No	(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

How to Evaluate Learnt 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
    - favors rules that have high accuracy and cover many positive tuples

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

- Rule pruning based on an independent set of test tuples
    - Pos/neg are # of positive/negative tuples covered by R.
    - If FOIL_Prune is higher for the pruned version of R, prune R

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

How to Evaluate Learnt Rule?

- We can use Likelihood Ratio Statistic, which confirms that effect of rule is not attributed to chance, but represents correlation between attribute value and classes.

$$\text{Likelihood\_Ratio} = 2 * \sum_{j=1}^{m} f_i \log_2\left(\frac{f_i}{e_i}\right)$$

  - m is the number of classes
  - For tuples satisfying the rule, fi is the observed frequency of each class among tuples, ei is the expected frequency if the rule made random predictions
  - Higher the Likelihood Ratio, better the rule is.
  - Used by CN2

Building Classification Rules

- Direct Method:
  - Extract rules directly from data
  - e.g.: RIPPER, CN2, Holte's 1R

- Indirect Method:
  - Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  - e.g: C4.5rules

Direct Method: Sequential Covering

- Sequential covering algorithm: Extracts rules directly from training data
- Typical sequential covering algorithms: FOIL, AQ, CN2, RIPPER
- Rules are learned *sequentially*, each for a given class C_i will cover many tuples of C_i but none (or few) of the tuples of other classes

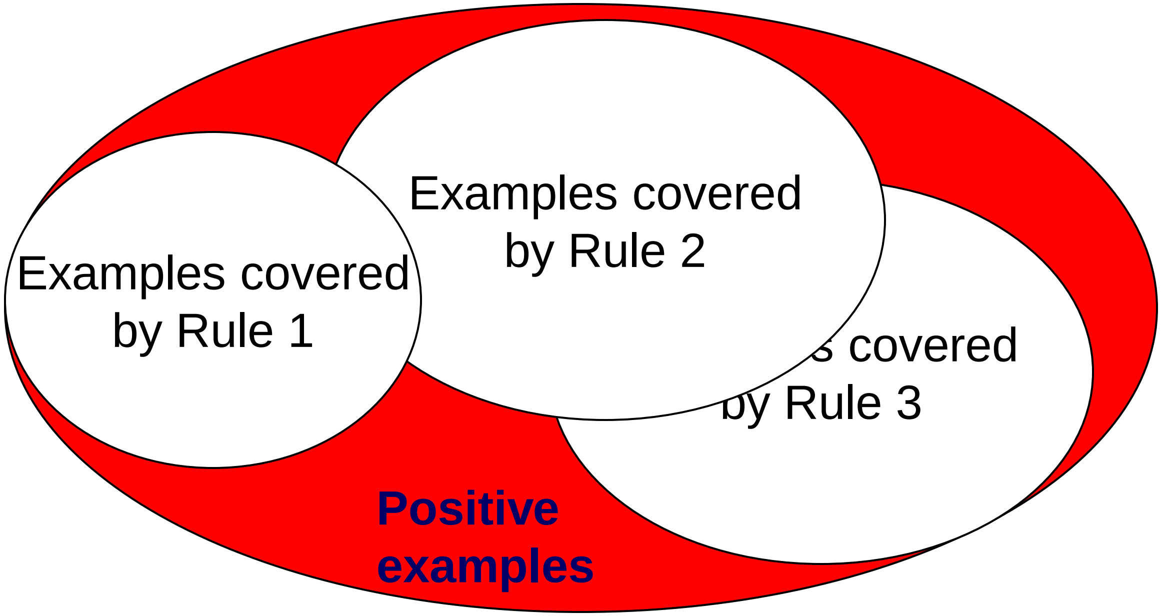
Rule Induction: Sequential Covering Method

- Start with an empty rule set
- Steps:
  - Rules are learned one at a time
  - Each time a rule is learned, the tuples covered by the rules are removed
  - Repeat the process(above steps) on the remaining tuples
    - until *termination condition*, e.g., when no more training examples or when the quality of a rule returned is below a user-specified threshold

- Comparison with decision-tree induction: learning a set of rules *simultaneously*

Sequential Covering Algorithm

while (enough target tuples left)
    generate a rule
    remove positive target tuples satisfying this rule



Rule Generation

- To generate a rule

  while(true)

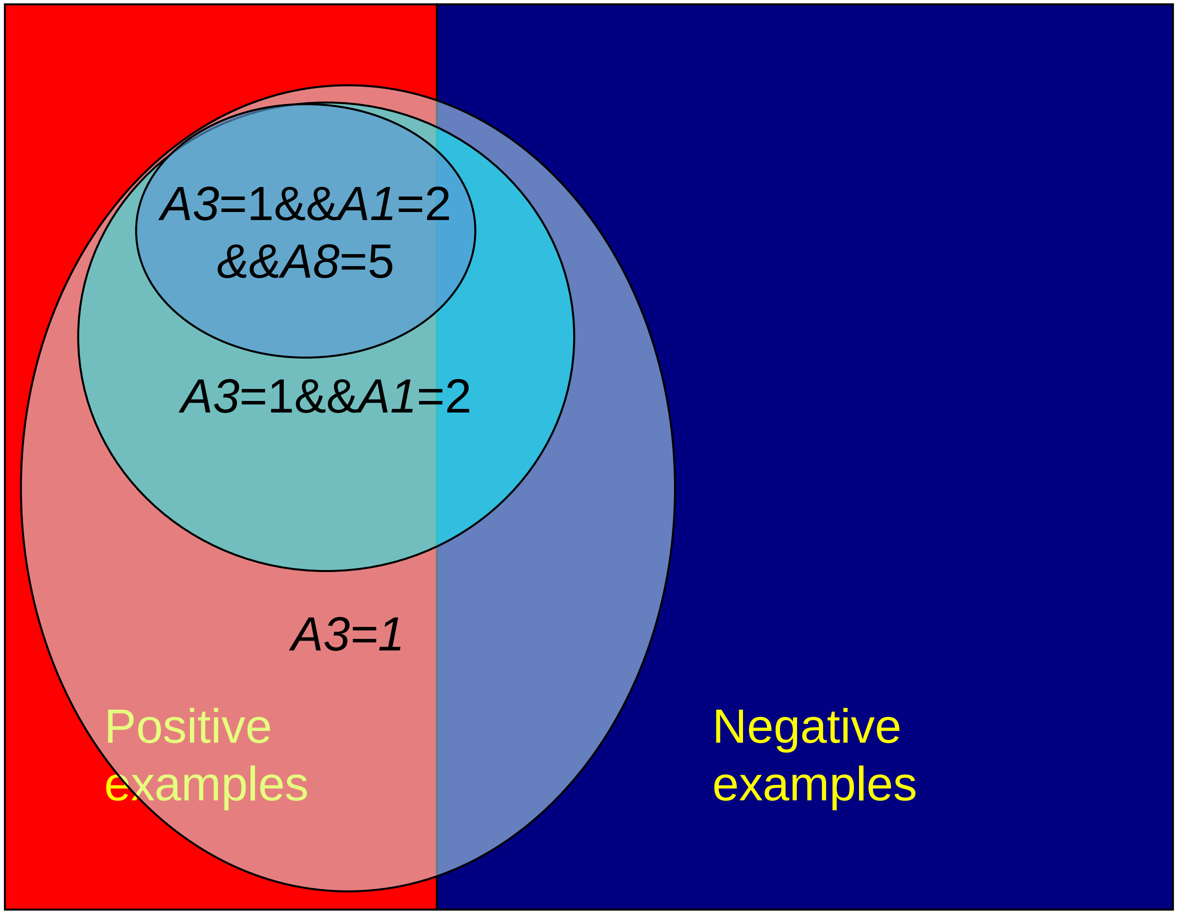
  find the best predicate *p*

  if foil-gain(*p*) > threshold **then** add *p* to current rule

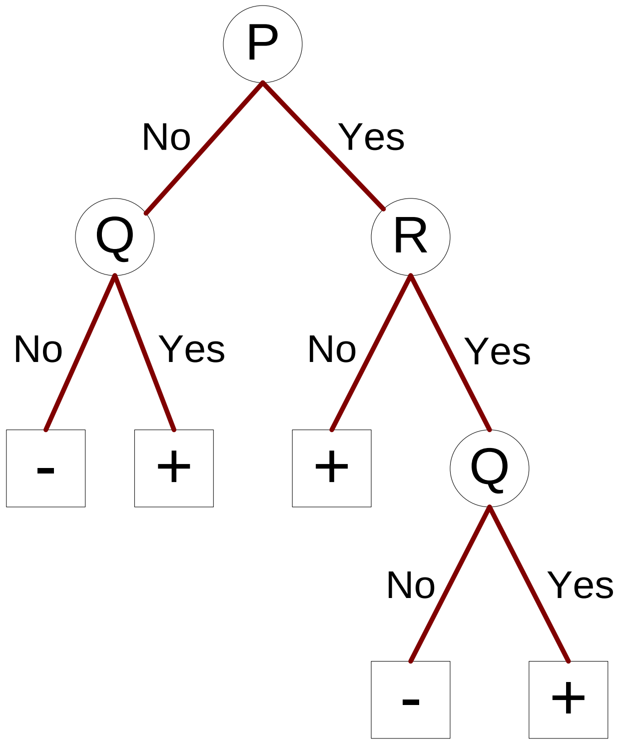
  else break

Predicates considered may be independent of each other (as in previous slide) or progressively restrictive (as in the next slide)

Rule Generation



Indirect Methods



Rule Set

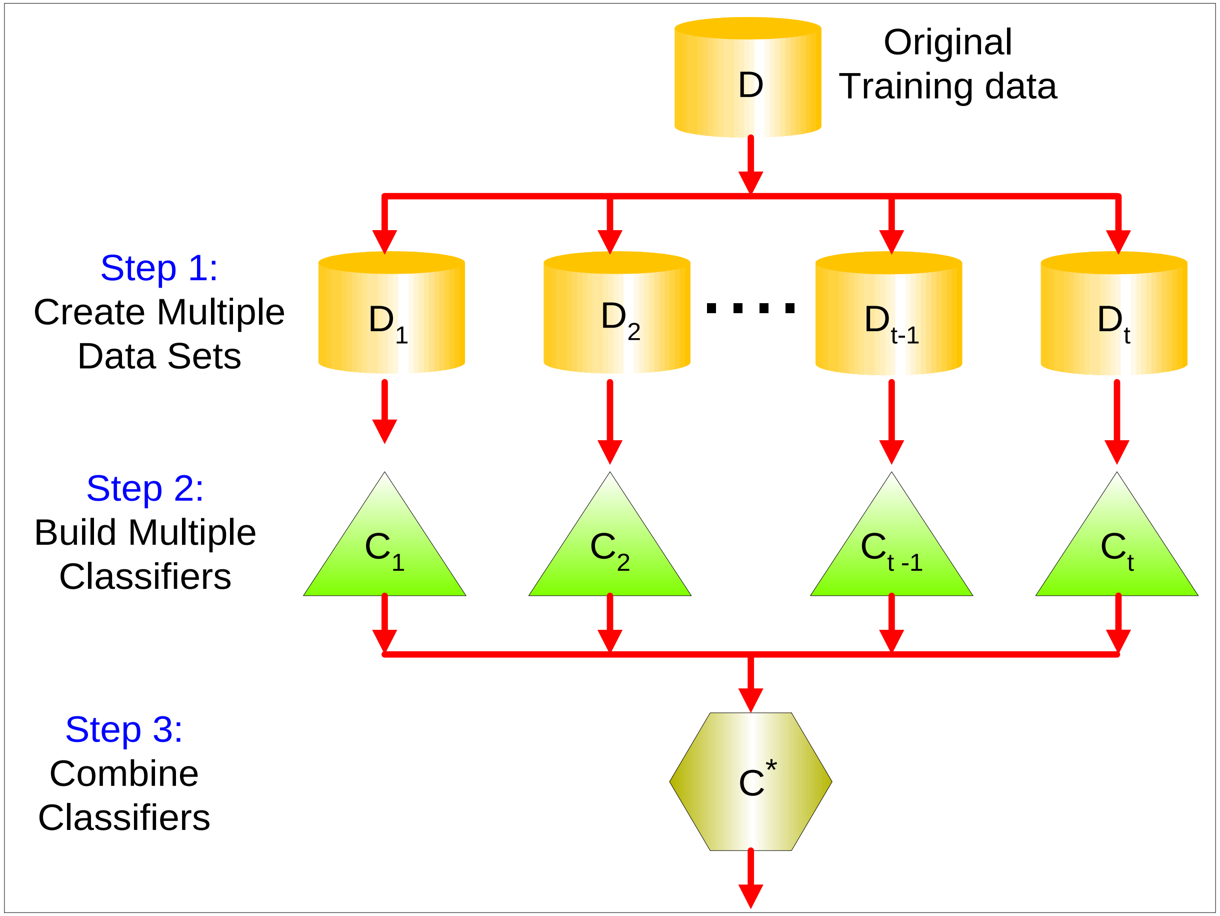
r1: (P=No,Q=No) ==> -
r2: (P=No,Q=Yes) ==> +
r3: (P=Yes,R=No) ==> +
r4: (P=Yes,R=Yes,Q=No) ==> -
r5: (P=Yes,R=Yes,Q=Yes) ==> +

Ensemble Methods

Ensemble Methods

- Construct a set of classifiers from the training data
- Predict class label of previously unseen records by aggregating predictions made by multiple classifiers

General Idea



Why does it work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, ε = 0.35
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^{i} (1-\varepsilon)^{25-i} = 0.06$$

When error rate differs…

- Suppose there are k base classifiers
  - Each classifier has different error rate, ε_i
  - Again, assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:
    - Majority of classifiers have to make wrong prediction
    - Compute the probability for each combination that can make wrong prediction (brute force method)
    - Sum up for all possible combinations

Model Evaluation and Selection

Model Evaluation and Selection

Evaluation metrics: How can we measure accuracy? Other metrics to consider?

Use **validation test set** of class-labeled tuples instead of training set when assessing accuracy

Methods for estimating a classifier's accuracy:

- Holdout method, random subsampling
- Cross-validation
- Bootstrap

Comparing classifiers:

- Cost-benefit analysis and ROC Curves

Classifier Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Given *m* classes, an entry, $CM_{i,j}$ in a **confusion matrix** indicates # of tuples in class *i* that were labeled by the classifier as class *j*

May have extra rows/columns to provide totals

Predicted class ->	C_1	¬ C_1
Actual class⇓		
C_1	**True Positives (TP)**	**False Negatives (FN)**
¬ C_1	**False Positives (FP)**	**True Negatives (TN)**

Classifier Evaluation Metrics: Confusion Matrix

Example of Confusion Matrix:

Predicted class ->	buy_computer = yes	buy_computer = no	Total
Actual class ⇓			
buy_computer = yes	**6954**	**46**	7000
buy_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

Classifier Evaluation Metrics: Accuracy, Error Rate, Sensitivity and Specificity

Classifier Accuracy, or recognition rate: percentage of test set tuples that are correctly classified

Accuracy = (TP + TN)/All

Error rate: *1 – accuracy*, or

Error rate = (FP + FN)/All

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P’	**N’**	**All**

- **Class Imbalance Problem**:
  - One class may be *rare*, e.g. fraud, or HIV-positive
  - Significant *majority of the negative class* and minority of the positive class
  - **Sensitivity**: True Positive recognition rate
    - **Sensitivity = TP/P**
  - **Specificity**: True Negative recognition rate
    - **Specificity = TN/N**

Classifier Evaluation Metrics: Precision and Recall, and F-measures

Precision: exactness – what % of tuples that the classifier labeled as positive are actually positive

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

Recall: completeness – what % of positive tuples did the classifier label as positive?

$$recall = \frac{TP}{TP + FN} = \frac{TP}{P}$$

Perfect score is 1.0

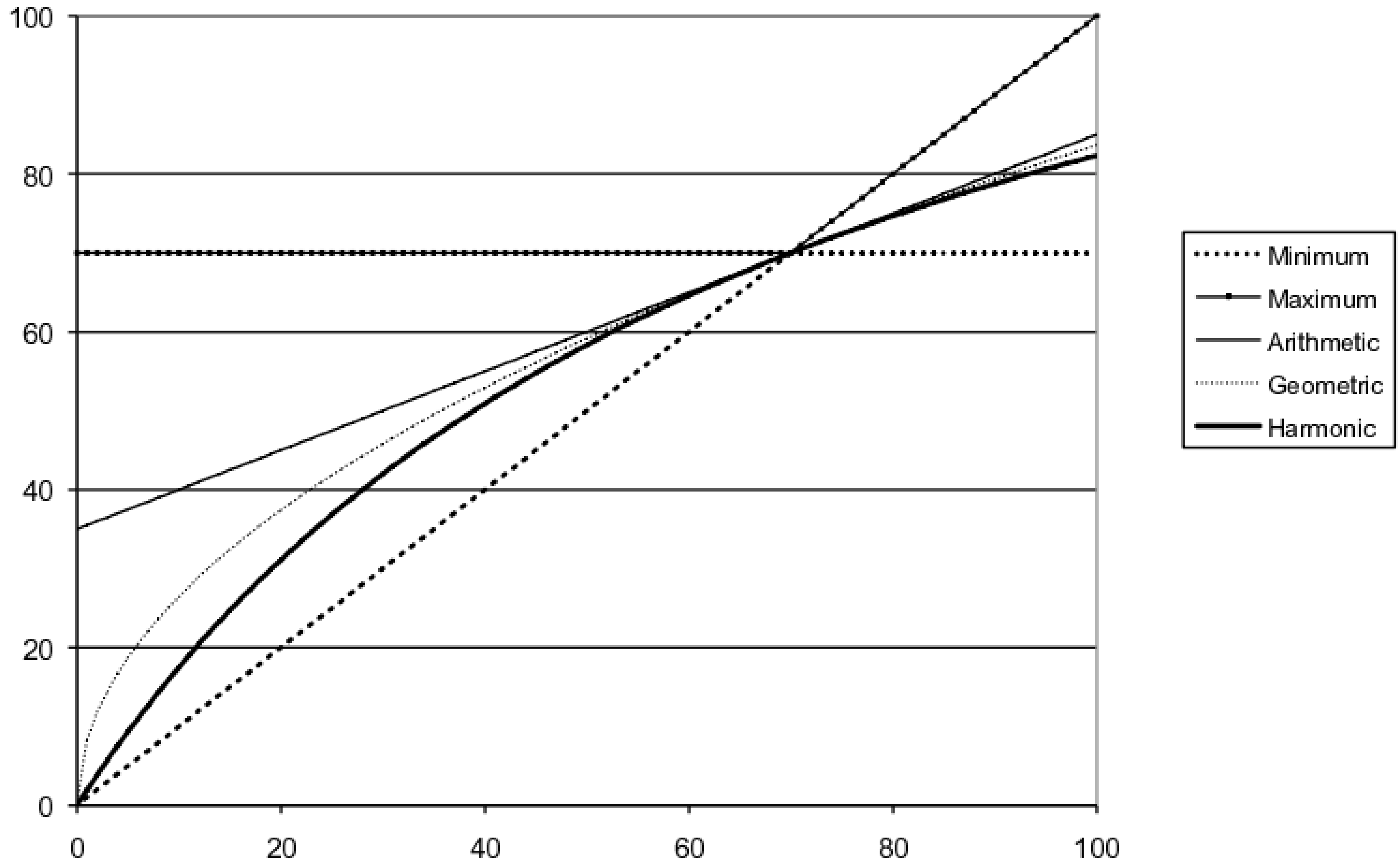
Inverse relationship between precision & recall

***F* measure (*F_1* or *F*-score)**: harmonic mean of precision and recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

Why a harmonic mean, but not arithmetic or geometric mean?

Classifier Evaluation Metrics: Precision and Recall, and F-measures



Precision (Recall fixed at 70 %)

Classifier Evaluation Metrics: Precision and Recall, and F-measures

Harmonic mean can be recomputed by applying weights to precision and recall

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}}$$

By substituting $\beta^2 = (1 - \alpha)/\alpha$,

We get

***F_β*:** weighted measure of precision and recall

– assigns β^2 times as much weight to recall as to precision

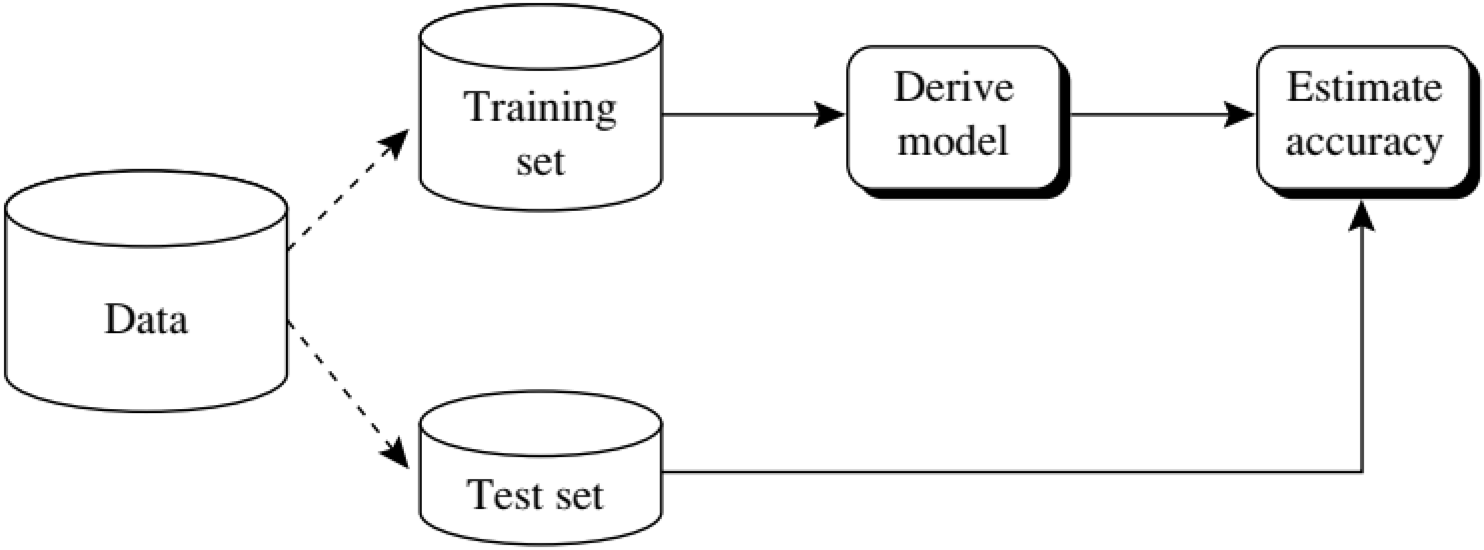
$$F_\beta = \frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

Classifier Evaluation Metrics: Example

Precision = 90/230 = 39.13% *Recall* = 90/300 = 30.00%

Actual Class\Predicted class	cancer = yes	cancer = no	Total	Recognition(%)
cancer = yes	**90**	**210**	300	30.00 (*sensitivity*
cancer = no	**140**	**9560**	9700	98.56 (*specificity)*
Total	230	9770	10000	96.40 (*accuracy*)

Evaluating Classifier Accuracy: Holdout Method



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

Evaluating Classifier Accuracy: Cross-Validation Methods

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 D_i as test set and others as training 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 Classifier Accuracy: Bootstrap

Bootstrap

- 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 bootstrap methods, and a common one is **.632 boostrap**

- A data set with *d* tuples 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 end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 – 1/d)^d \approx e^{-1} = 0.368$)
- Repeat the sampling procedure *k* times, overall accuracy of the model:

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

ROC (Receiver Operating Characteristic)

ROC Curve

(TPR,FPR):

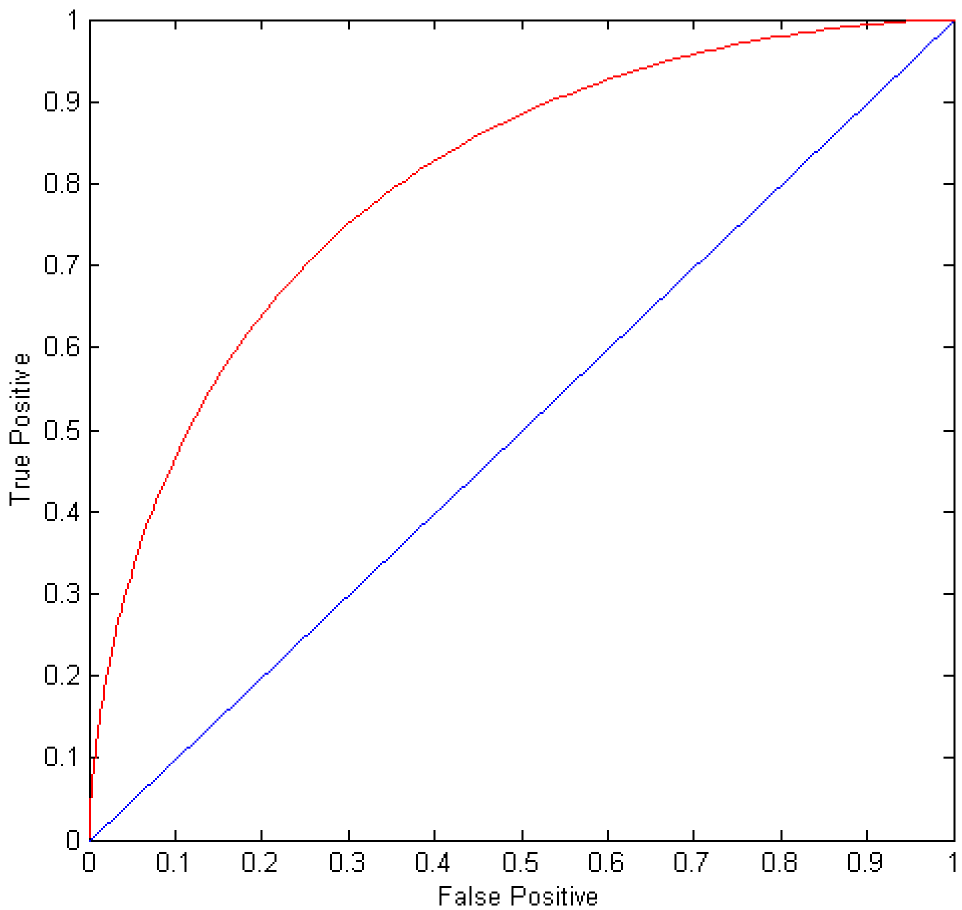
(0,0): declare everything to be negative class

(1,1): declare everything to be positive class

(1,0): ideal

Diagonal line:

- Random guessing
- Below diagonal line:
  - prediction is opposite of the true class

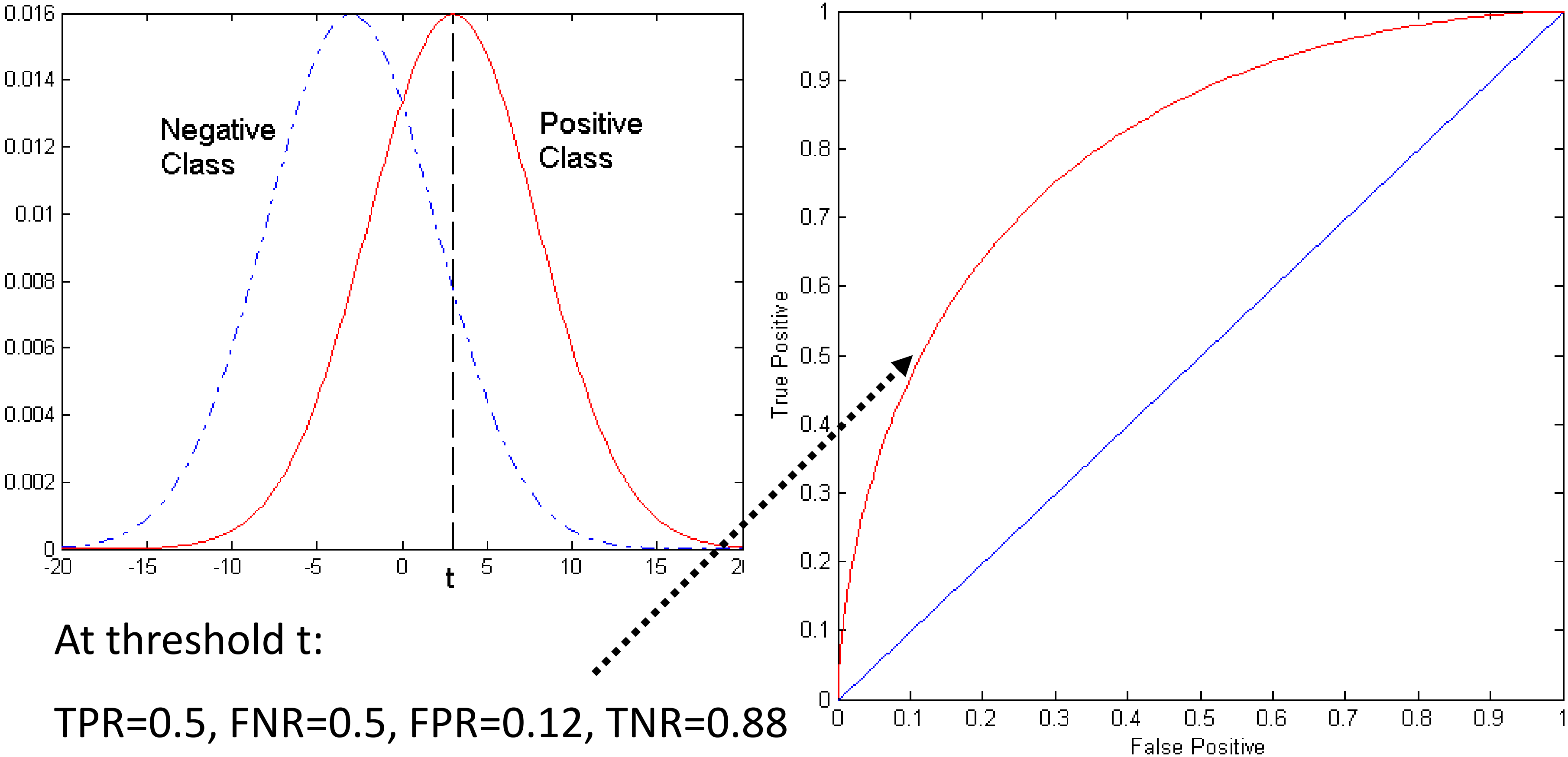


ROC (Receiver Operating Characteristic)

- To draw ROC curve, classifier must produce continuous-valued output
  - Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record
  - By using different thresholds on this value, we can create different variations of the classifier with TPR/FPR tradeoffs
- Many classifiers produce only discrete outputs (i.e., predicted class)

ROC Curve Example

- 1-dimensional data set containing 2 classes (positive and negative)
- Any points located at x > t is classified as positive



At threshold t:

TPR=0.5, FNR=0.5, FPR=0.12, TNR=0.88

How to Construct an ROC curve

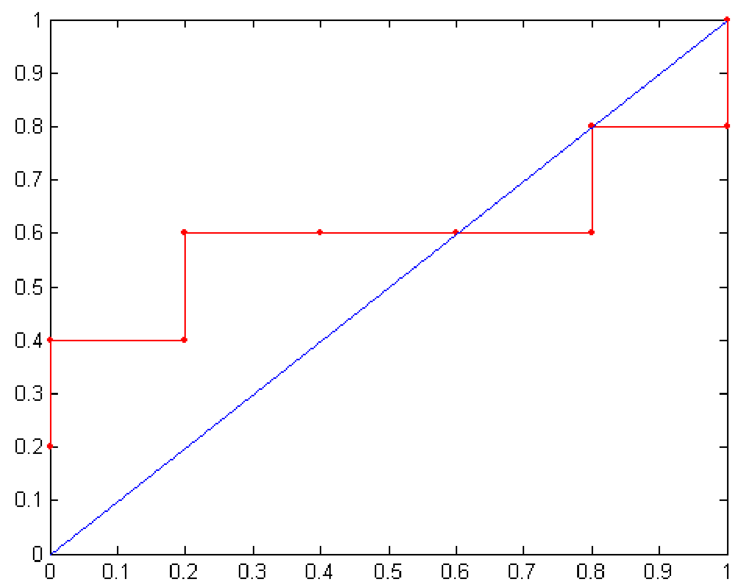
Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use a classifier that produces a continuous-valued score for each instance
  - The more likely it is for the instance to be in the + class, the higher the score
- Sort the instances in decreasing order according to the score
- Apply a threshold at each unique value of the score
- Count the number of TP, FP, TN, FN at each threshold
  - TPR = TP/(TP+FN)
  - FPR = FP/(FP + TN)

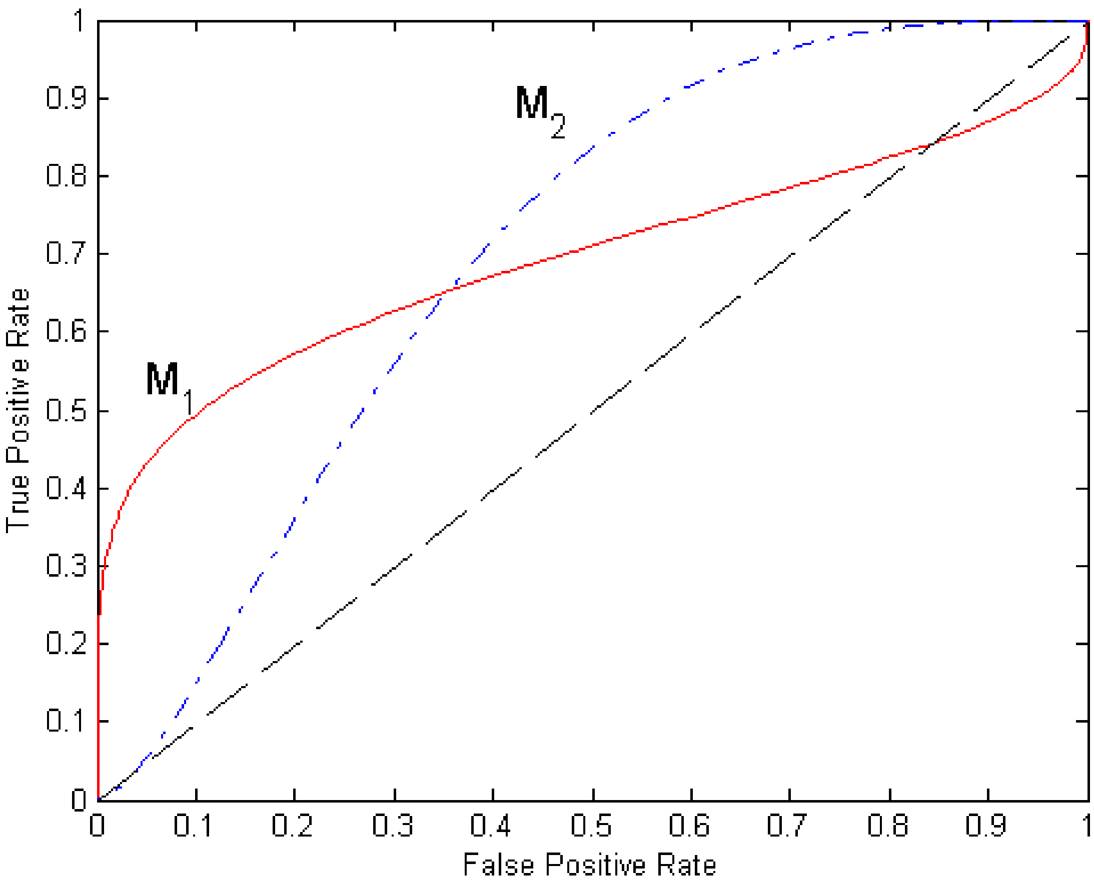
How to construct an ROC curve

Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	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

ROC Curve:



Using ROC for Model Comparison



- No model consistently outperforms the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve (AUC)
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

Prescribed Text Books

	Author(s), Title, Edition, Publishing House
T1	Tan P. N., Steinbach M & Kumar V. “Introduction to Data Mining” Pearson Education
T4	Data Mining: Concepts and Techniques, Third Edition by Jiawei Han, Micheline Kamber and Jian Pei Morgan Kaufmann Publishers
	Principles of Data Mining, Second Edition by Max Bramer Springer © 2013

Thank You