Data Mining
Classification: Basic Concepts,

Lecture Notes for Chapter 4 - 5

Introduction to Data Mining

by

Tan, Steinbach, Kumar

Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.

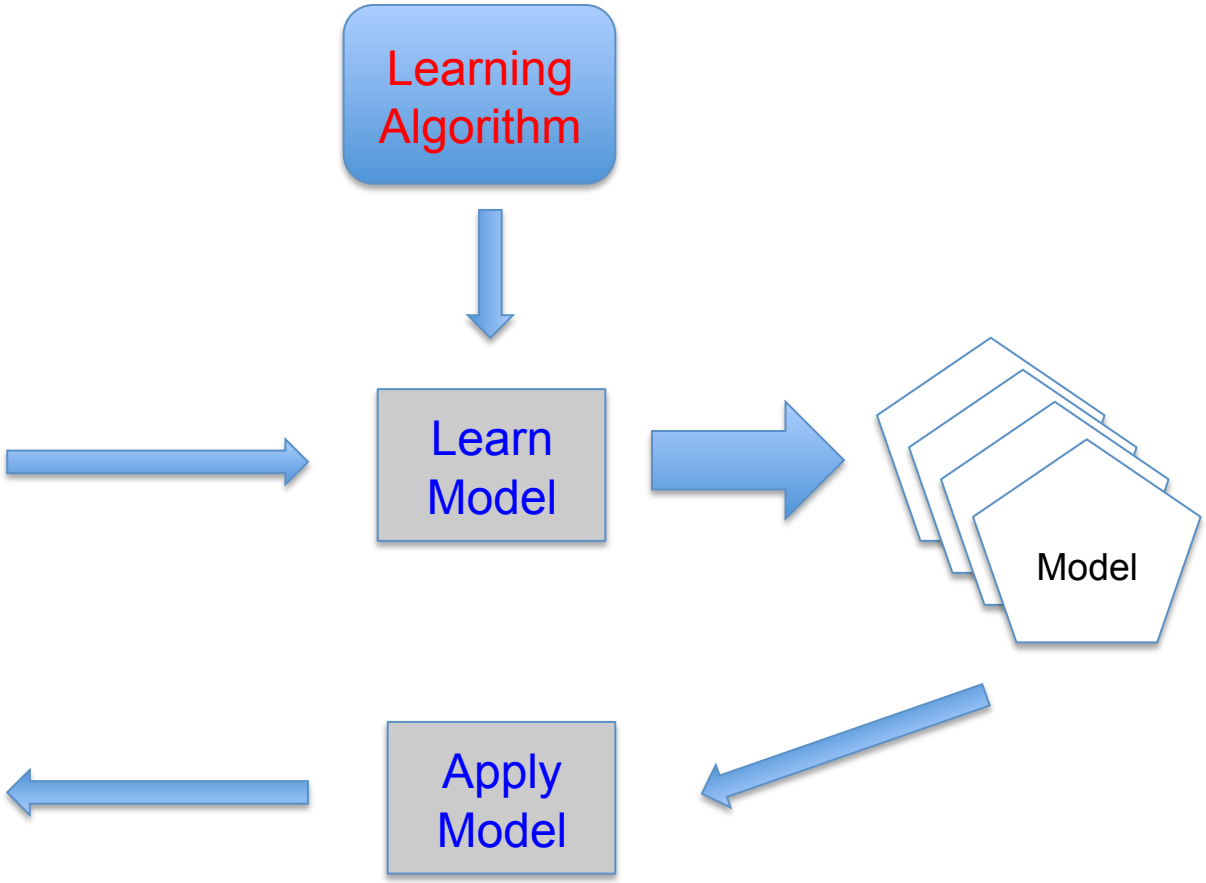
Classification: Definition

- Goal: previously unseen records should be assigned a class as accurately as possible.
  - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

- When the class is numerical, the problem is a regression problem.

Illustrating Classification Task

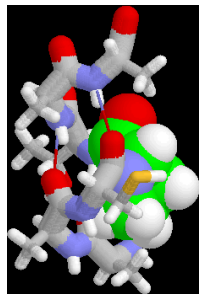
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Tid	Refund	Marital Status	Taxable Income	Cheat
1	No	Married	80K	?
2	No	Single	100K	?
3	Yes	Single	90K	?
4	No	Married	120K	?
5	Yes	Divorced	130K	?

Examples of Classification Task

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc



Classification Techniques

- Decision Tree
- Naïve Bayes
- Instance Based Learning
- Rule-based Methods
- Neural Networks
- Bayesian Belief Networks
- Support Vector Machines

Classification: Measure the quality

Usually the Accuracy measure is used:

$$\text{Accuracy} = \frac{\text{Number of Correctly Classified record}}{\text{Total Number of record in the test set}}$$

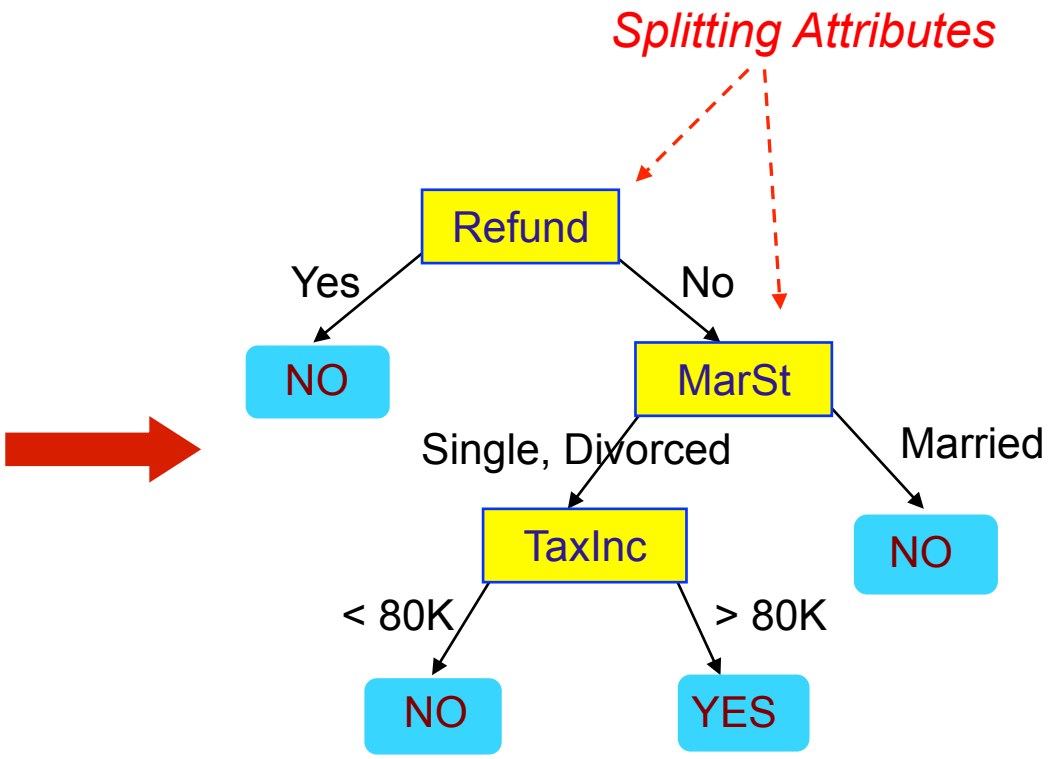
Decision Tree

- Uses a tree structure to model the training set
- Classifies a new record following the path in the tree
- Inner nodes represent attributes and leaves nodes represent the class

Example of a Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
	categorical	categorical	continuous	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

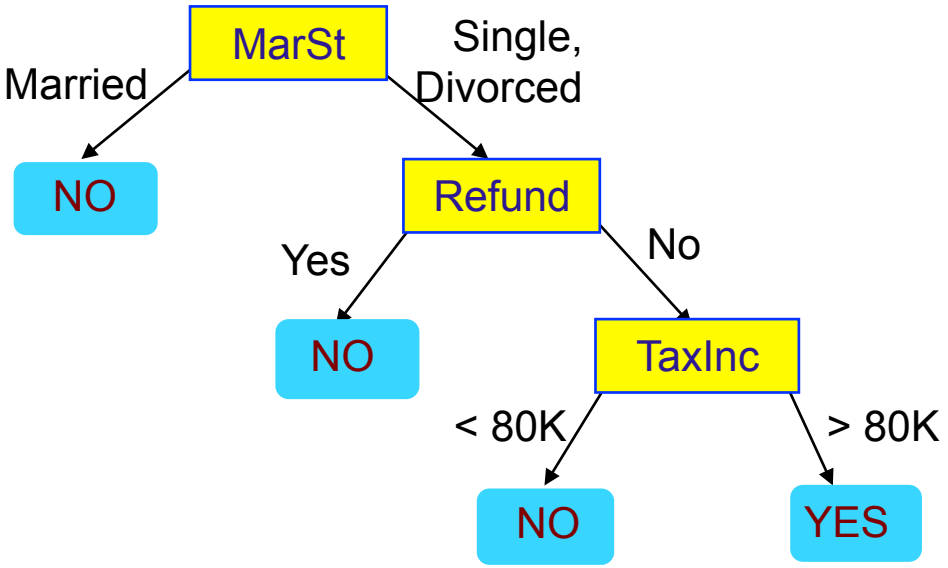
Training Data



Model: Decision Tree

Another Example of Decision Tree

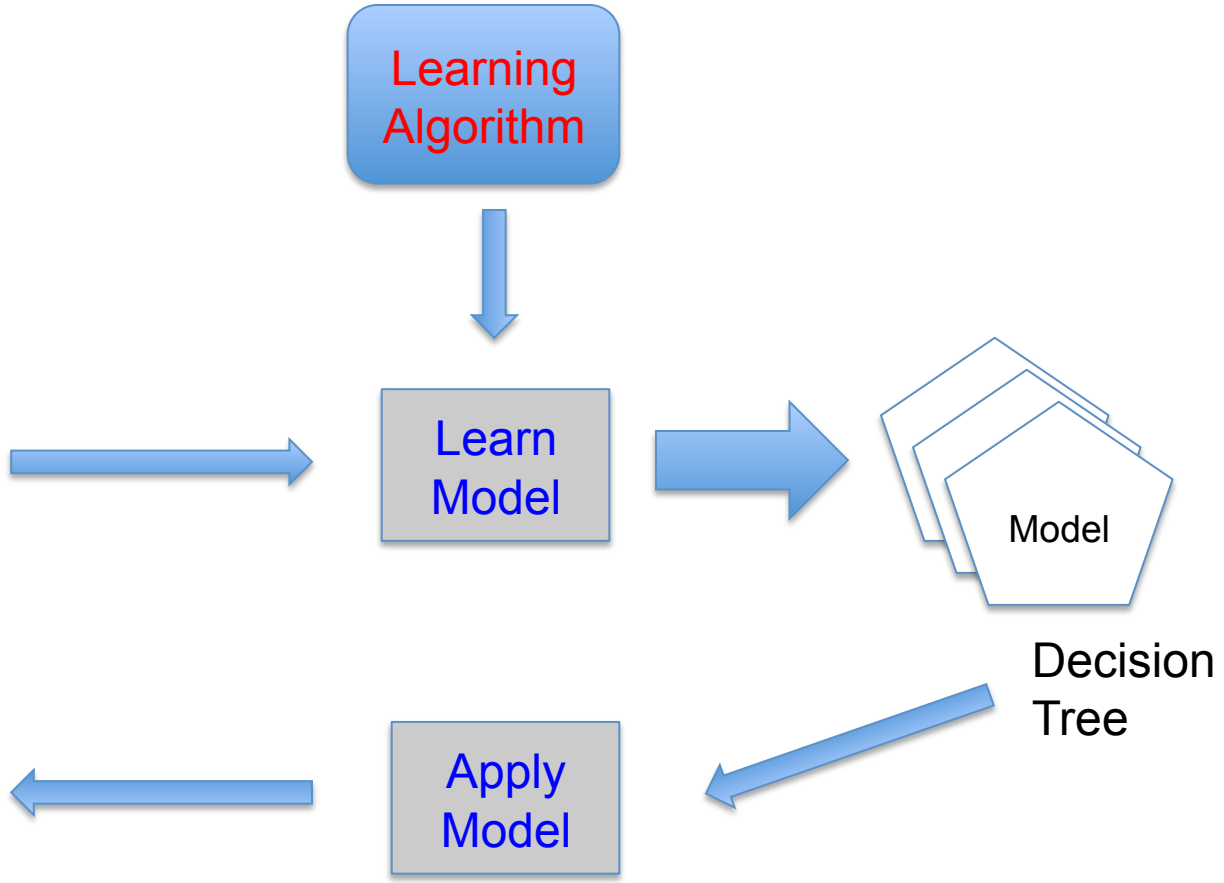
	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**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



There could be more than one tree that fits the same data!

Decision Tree Classification Task

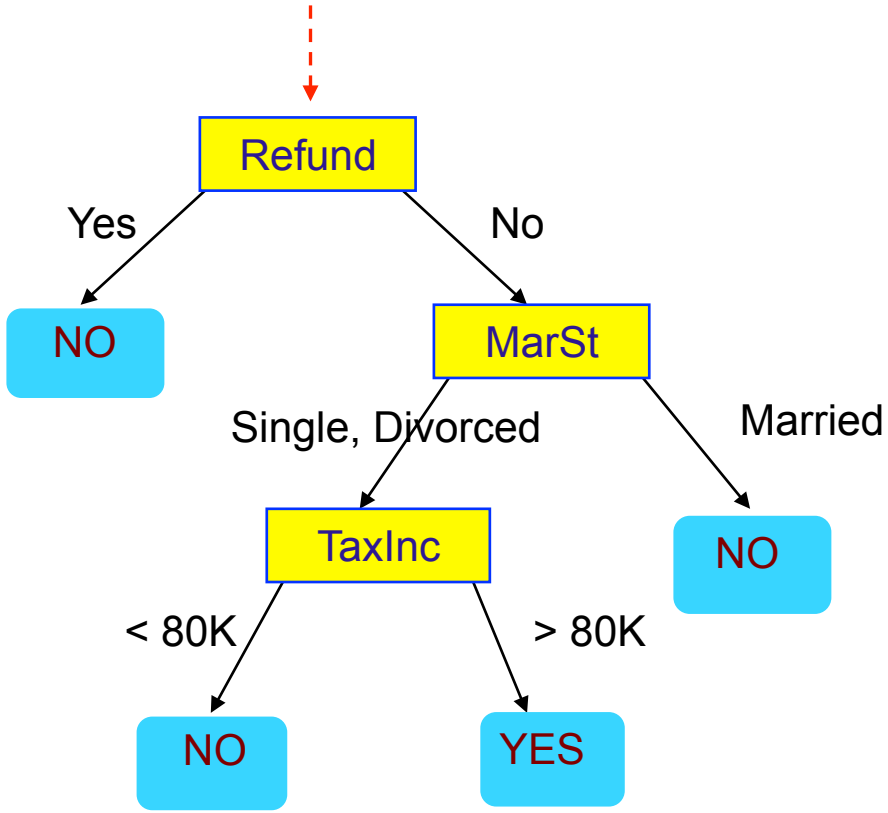
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Tid	Refund	Marital Status	Taxable Income	Cheat
1	No	Married	80K	?
2	No	Single	100K	?
3	Yes	Single	90K	?
4	No	Married	120K	?
5	Yes	Divorced	130K	?

Apply Model to Test Data

Start from the root of tree.



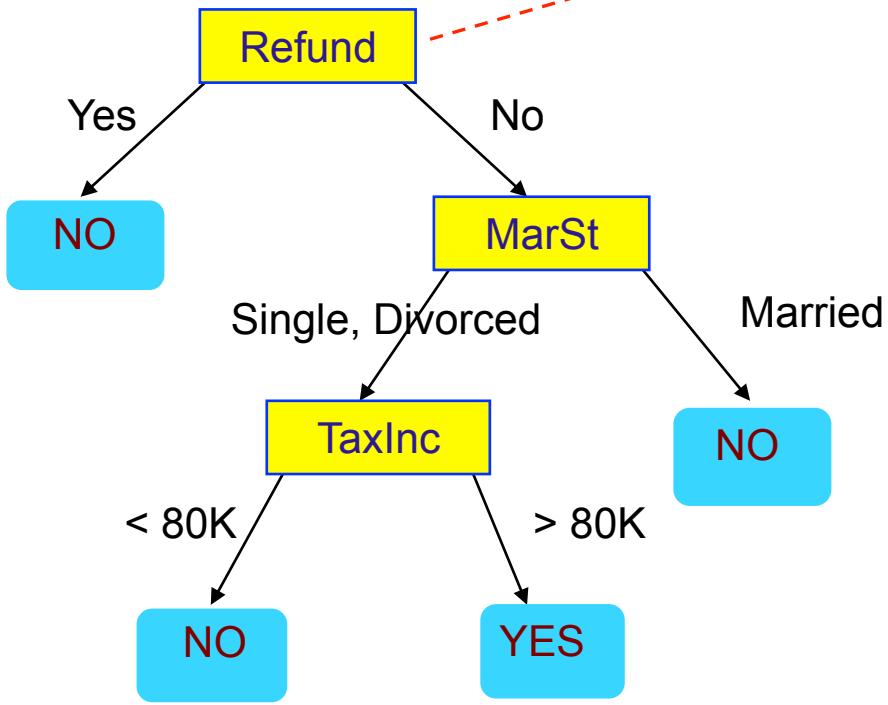
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

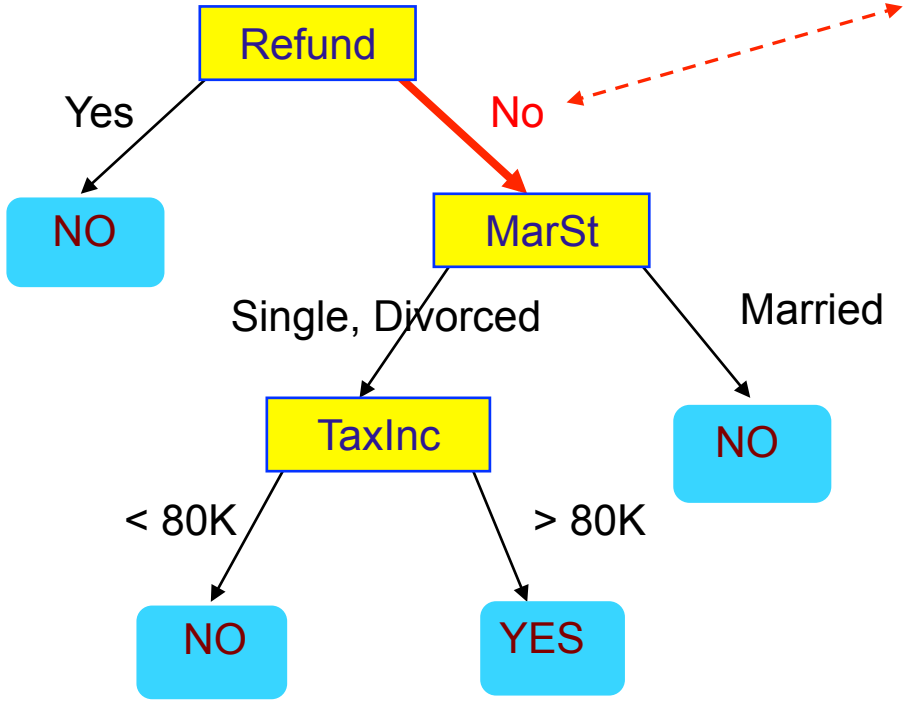
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

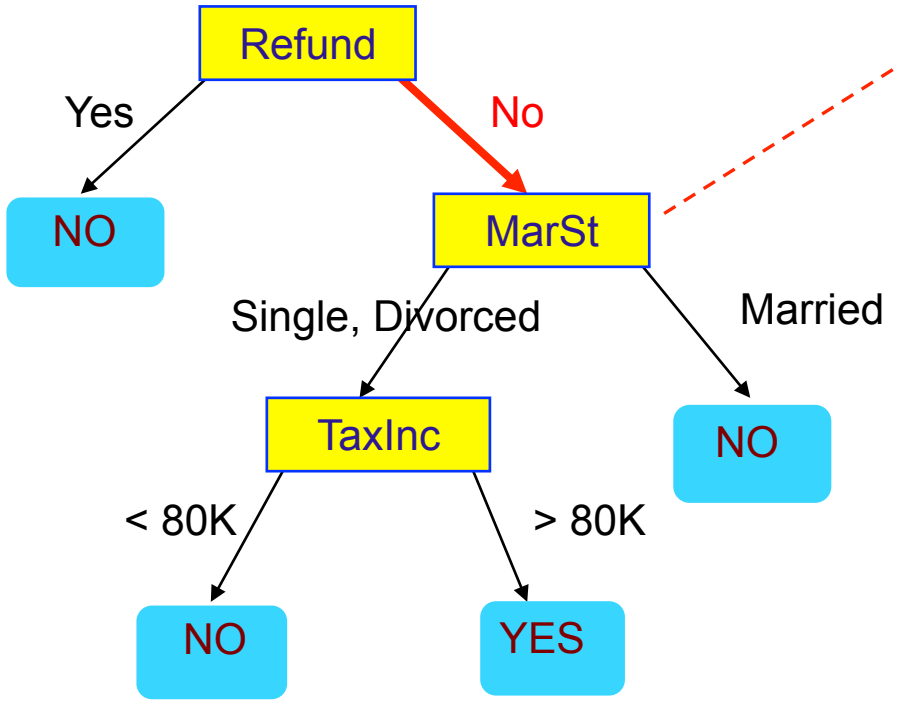
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

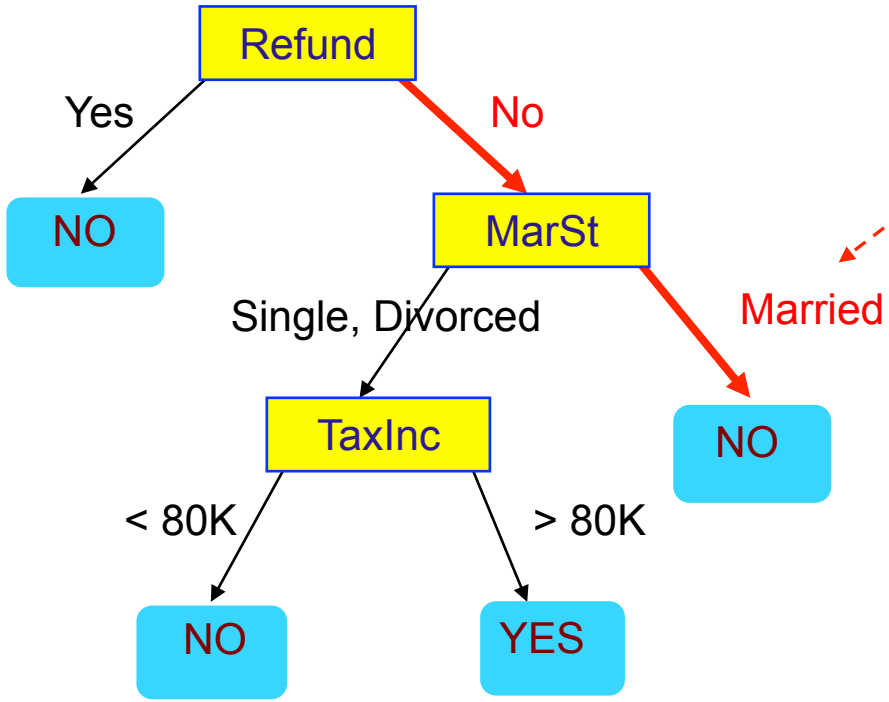
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

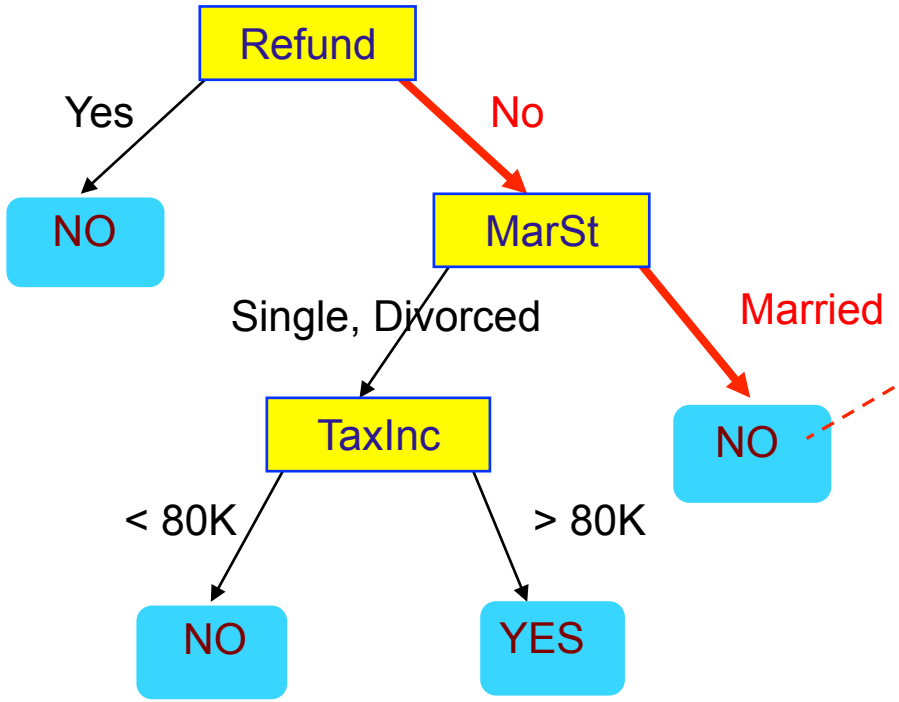
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

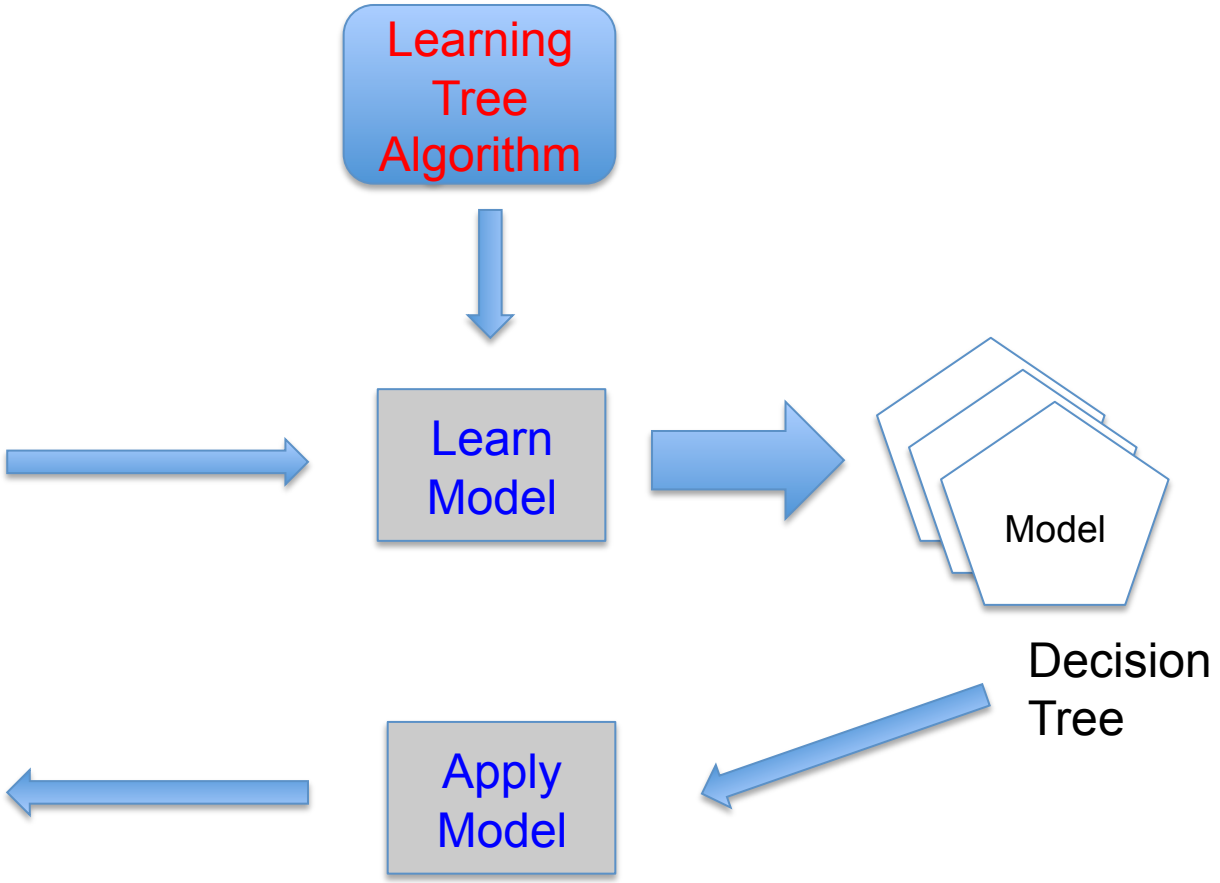
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Assign Cheat to “No”

Decision Tree Classification Task

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



Tid	Refund	Marital Status	Taxable Income	Cheat
1	No	Married	80K	?
2	No	Single	100K	?
3	Yes	Single	90K	?
4	No	Married	120K	?
5	Yes	Divorced	130K	?

Decision Tree Induction

- Many Algorithms:
  - Hunt's Algorithm (one of the earliest)
  - CART
  - ID3, C4.5 (J48 on WEKA)
  - SLIQ,SPRINT

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

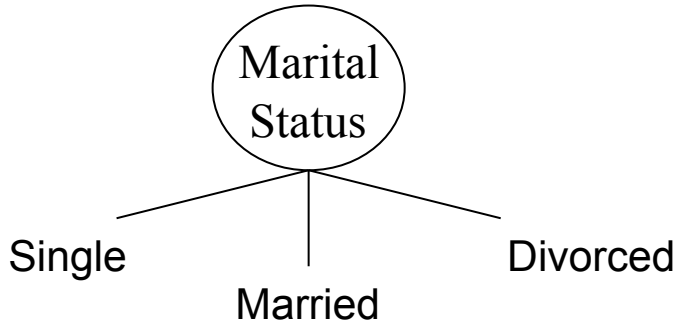
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

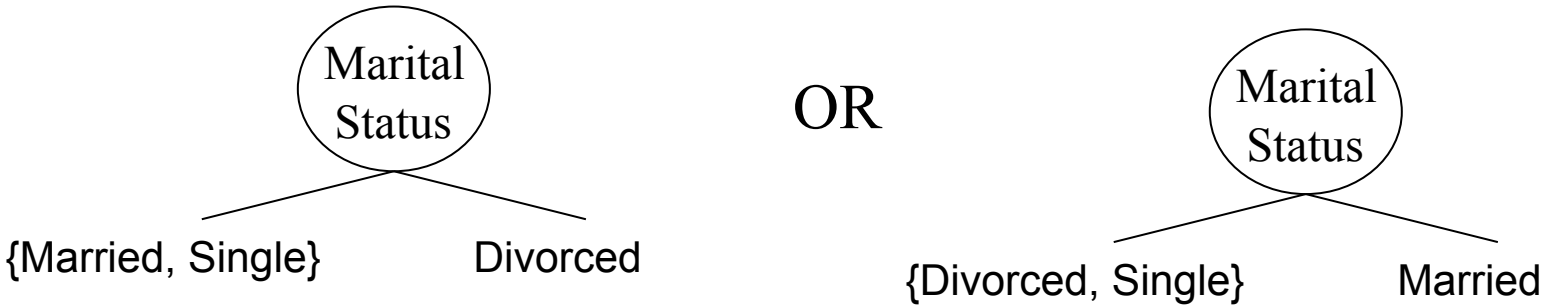
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

- Multi-way split: Use as many partitions as distinct values.



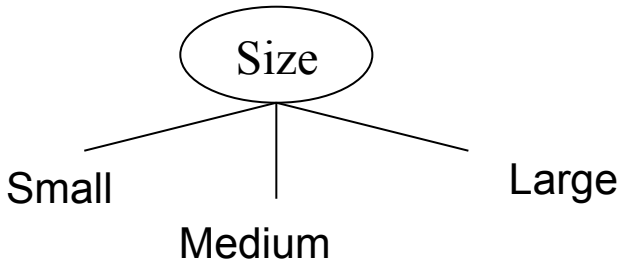
- Binary split: Divides values into two subsets. Need to find optimal partitioning.



Splitting Based on Ordinal Attributes

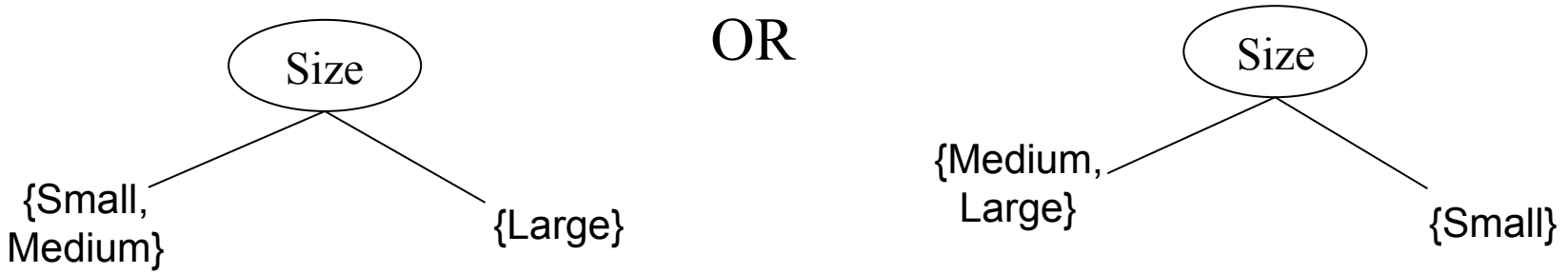
We can imagine an attribute SIZE defined over the ordered set {Small, Medium, Large}

- Multi-way split: Use as many partitions as distinct values.

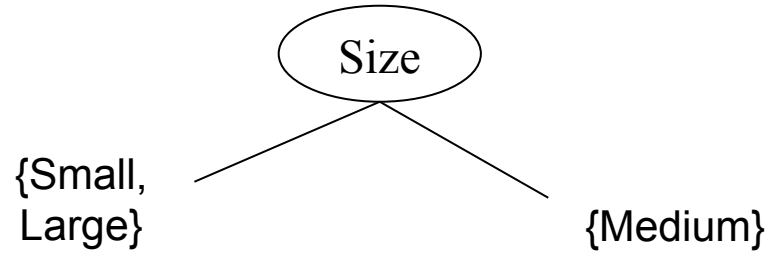


Splitting Based on Ordinal Attributes

Binary split: Divides values into two subsets. Need to find optimal partitioning.



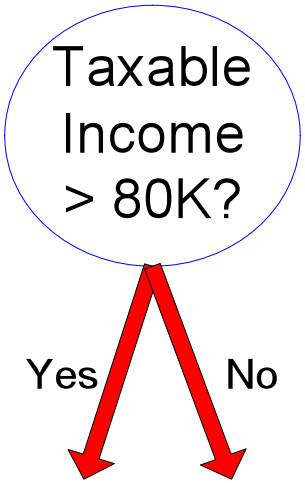
What about this split?



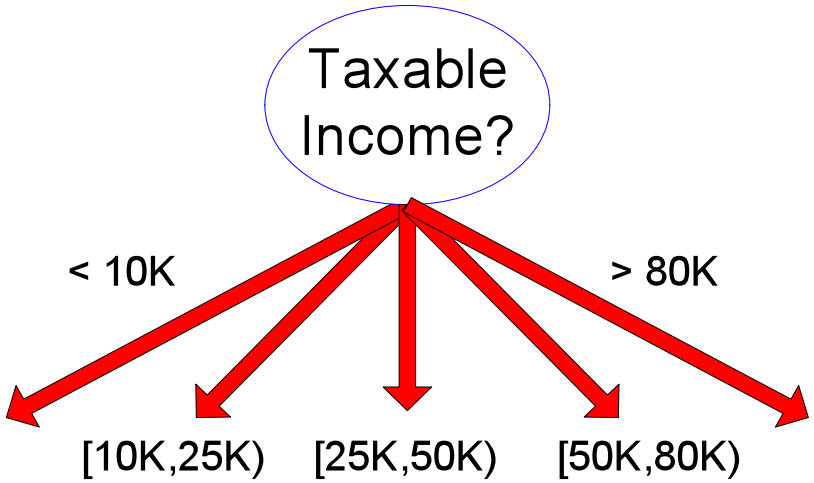
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes



(i) Binary split



(ii) Multi-way split

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

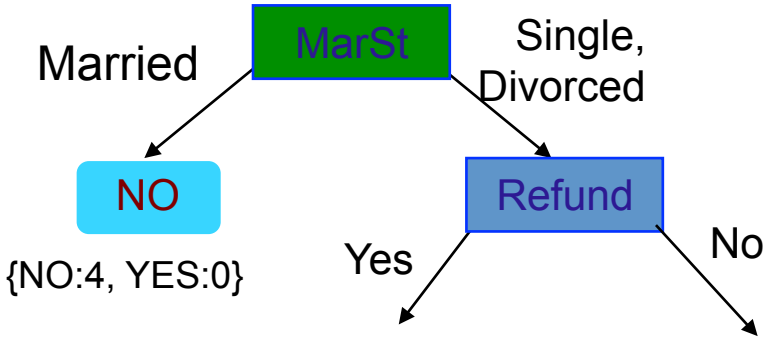
Example

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

	Yes	No
Married	0	4
Single, Divorced	3	3

Example

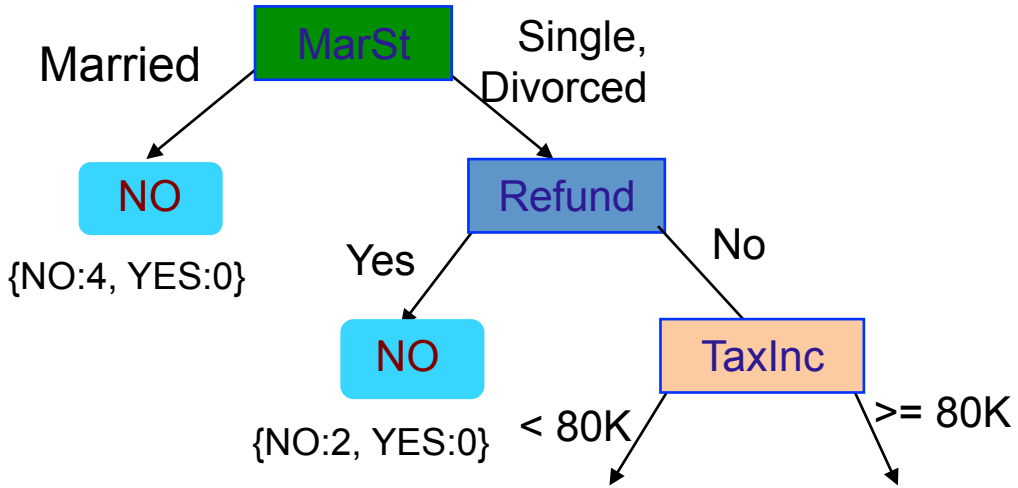
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



		YES	NO
Single, Divorced	**Refund = NO**	3	1
Single, Divorced	**Refund = Yes**	0	2

Example

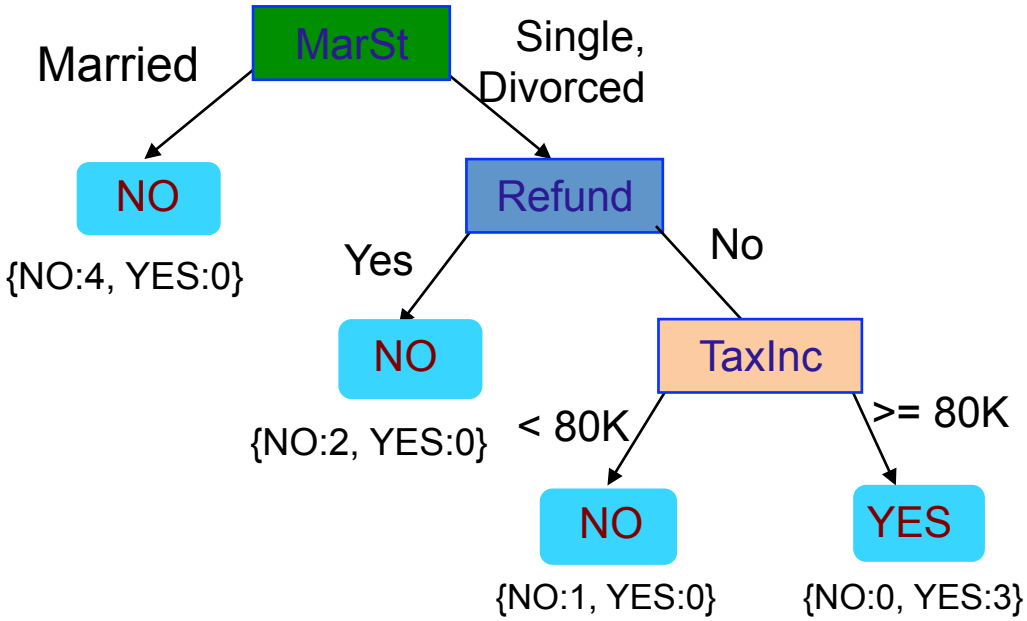
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



			YES	NO
Single, Divorced	**Refund = NO**	**TaxInc = < 80k**	0	1
Single, Divorced	**Refund = NO**	**TaxInc = >= 80k**	3	0

Example

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



How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, High degree of impurity	Homogeneous, Low degree of impurity

Measures of Node Impurity

Given a node t

- Gini Index $$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$
- Entropy $$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$
- Misclassification error $$Error(t) = 1 - \max_j P(j \mid t)$$

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class

- Stop expanding a node when all the records have similar attribute values

Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Accuracy is comparable to other classification techniques for many simple data sets

Naive Bayes

- Uses probability theory to model the training set
- Assumes independence between attributes
- Produces a model for each class

Bayes Theorem

Conditional Probability:

$$P(C \mid A) = \frac{P(A,C)}{P(A)}$$

$$P(A \mid C) = \frac{P(A,C)}{P(C)}$$

Bayes theorem:

$$P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)}$$

Example of Bayes Theorem

- Given:
  - A doctor knows that meningitis causes stiff neck 50% of the time
  - Prior probability of any patient having meningitis is 1/50,000
  - Prior probability of any patient having stiff neck is 1/20

- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M \mid S) = \frac{P(S \mid M)P(M)}{P(S)} = \frac{0.5 \times 1/50000}{1/20} = 0.0002$$

Bayesian Classifiers

- Consider each attribute and class label as random variables

- Given a record with attributes (A_1, A_2,…,A_n)
  - Goal is to predict class C
  - Specifically, we want to find the value of C that maximizes P(C| A_1, A_2,…,A_n)

- Can we estimate P(C| A_1, A_2,…,A_n) directly from data?

Bayesian Classifiers

- Approach:
  - compute the posterior probability $P(C \mid A_1, A_2, \ldots, A_n)$ for all values of C using the Bayes theorem

$$P(C \mid A_1 A_2 \mathbf{K}\ A_n) = \frac{P(A_1 A_2 \mathbf{K}\ A_n \mid C) P(C)}{P(A_1 A_2 \mathbf{K}\ A_n)}$$

  - Choose value of C that maximizes
    $P(C \mid A_1, A_2, \ldots, A_n)$

  - Equivalent to choosing value of C that maximizes
    $P(A_1, A_2, \ldots, A_n \mid C)\ P(C)$

- How to estimate $P(A_1, A_2, \ldots, A_n \mid C)$?

Naïve Bayes Classifier

- Assume independence among attributes A_i when class is given:
  - $P(A_1, A_2, \ldots, A_n \mid C) = P(A_1 \mid C_j)\, P(A_2 \mid C_j) \ldots P(A_n \mid C_j)$
  - Can estimate $P(A_i \mid C_j)$ for all A_i and C_j.
  - New point is classified to C_j if $P(C_j) \prod P(A_i \mid C_j)$ is maximal.

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

- Class: $P(C) = N_c/N$
  - e.g., P(No) = 7/10,
    P(Yes) = 3/10
- For discrete attributes:

  $P(A_i \mid C_k) = |A_{ik}|/ N_c$

  - where $|A_{ik}|$ is number of instances having attribute A_i and belongs to class C_k
  - Examples:

    P(Status=Married|No) = 4/7
    P(Refund=Yes|Yes)=0

How to Estimate Probabilities from Data?

- For continuous attributes:
  - Discretize the range into bins
    - one ordinal attribute per bin
    - violates independence assumption
  - Two-way split: $(A < v)$ or $(A > v)$
    - choose only one of the two splits as new attribute
  - Probability density estimation:
    - Assume attribute follows a normal distribution
    - Use data to estimate parameters of distribution (e.g., mean and standard deviation)
    - Once probability distribution is known, can use it to estimate the conditional probability $P(A_i|c)$

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Compute:

P(Status=Married|Yes) = ?

P(Refund=Yes|No) = ?

P(Status=Divorced|Yes) = ?

P(TaxableInc > 80K|Yes) = ?

P(TaxableInc > 80K|NO) = ?

How to Estimate Probabilities from Data?

Tid	Refund	Marital Status	Taxable Income	Evade
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

Compute:

P(Status=Married|Yes) = 0/3

P(Refund=Yes|No) = 3/7

P(Status=Divorced|Yes) = 1/3

P(TaxableInc > 80K|Yes) = 3/3

P(TaxableInc > 80K|NO) = 4/7

Example of Naïve Bayes Classifier

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} >= 80\text{K})$

REFUND	P(Refund=Yes|No) = 3/7 P(Refund=No|No) = 4/7 P(Refund=Yes|Yes) = 0 P(Refund=No|Yes) = 1
MARITAL STATUS	P(Marital Status=Single|No) = 2/7 P(Marital Status=Divorced|No) = 1/7 P(Marital Status=Married|No) = 4/7 P(Marital Status=Single|Yes) = 2/7 P(Marital Status=Divorced|Yes) = 1/7 P(Marital Status=Married|Yes) = 0
TAXABLE INCOMING	P(TaxableInc >= 80K|Yes) = 3/3 P(TaxableInc >= 80K|NO) = 4/7 P(TaxableInc < 80K|Yes) = 0/3 P(TaxableInc < 80K|NO) = 3/7

Class=No	7/10
Class=Yes	3/10

$$P(C_j|A_1, \ldots, A_n) = P(A_1, \ldots, A_n|C_j)\, P(C_j) = P(A_1|C_j) \ldots P(A_n|C_j)\, P(C_j)$$

Example of Naïve Bayes Classifier

- **P(X|Class=No)** = P(Refund=No|Class=No)
  × P(Married| Class=No)
  × P(Income>=80K| Class=No)
  = 4/7 × 4/7 × 4/7 = **0.1865**

P(X|No)P(No) =
0.1865 * 0.7 = 0.1306

- **P(X|Class=Yes)** = P(Refund=No| Class=Yes)
  × P(Married| Class=Yes)
  × P(Income>=80K| Class=Yes)
  = 1 × 0 × 1 = **0**

P(X|Yes)P(Yes) =
0 * 0.3 = 0

Since P(X|No)P(No) > P(X|Yes)P(Yes)

Therefore P(No|X) > P(Yes|X)
=> **Class = No**

Example of Naïve Bayes Classifier(2)

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Single}, \text{Income} >= 80\text{K})$

REFUND	P(Refund=Yes|No) = 3/7 P(Refund=No|No) = 4/7 P(Refund=Yes|Yes) = 0 P(Refund=No|Yes) = 3/3
MARITAL STATUS	P(Marital Status=Single|No) = 2/7 P(Marital Status=Divorced|No) = 1/7 P(Marital Status=Married|No) = 4/7 P(Marital Status=Single|Yes) = 2/7 P(Marital Status=Divorced|Yes) = 1/7 P(Marital Status=Married|Yes) = 0
TAXABLE INCOMING	P(TaxableInc >= 80K|Yes) = 3/3 P(TaxableInc >= 80K|NO) = 4/7 P(TaxableInc < 80K|Yes) = 0/3 P(TaxableInc < 80K|NO) = 3/7

Class=No	7/10
Class=Yes	3/10

$$P(A_1, A_2, \ldots, A_n \mid C)\, P(C) = P(A_1 \mid C_j)\, P(A_2 \mid C_j) \ldots P(A_n \mid C_j)\, P(C)$$

Example of Naïve Bayes Classifier(2)

- **P(X|Class=No)** = P(Refund=No|Class=No)
  × P(Single| Class=No)
  × P(Income>=80K| Class=No)
  = 4/7 × 2/7 × 4/7 = 0.0933

P(X|No)P(No) =
0.0933 * 0.7 = 0.06531

- **P(X|Class=Yes)** = P(Refund=No| Class=Yes)
  × P(Single| Class=Yes)
  × P(Income>=80K|Class=Yes)
  = 1 × 2/3 × 1 =0.666

P(X|Yes)P(Yes) =
0.666* 0.3 = 0.08571

Since P(X|No)P(No) < P(X|Yes)P(Yes)

Therefore P(No|X) < P(Yes|X)
=> Class = Yes

Naïve Bayes (Summary)

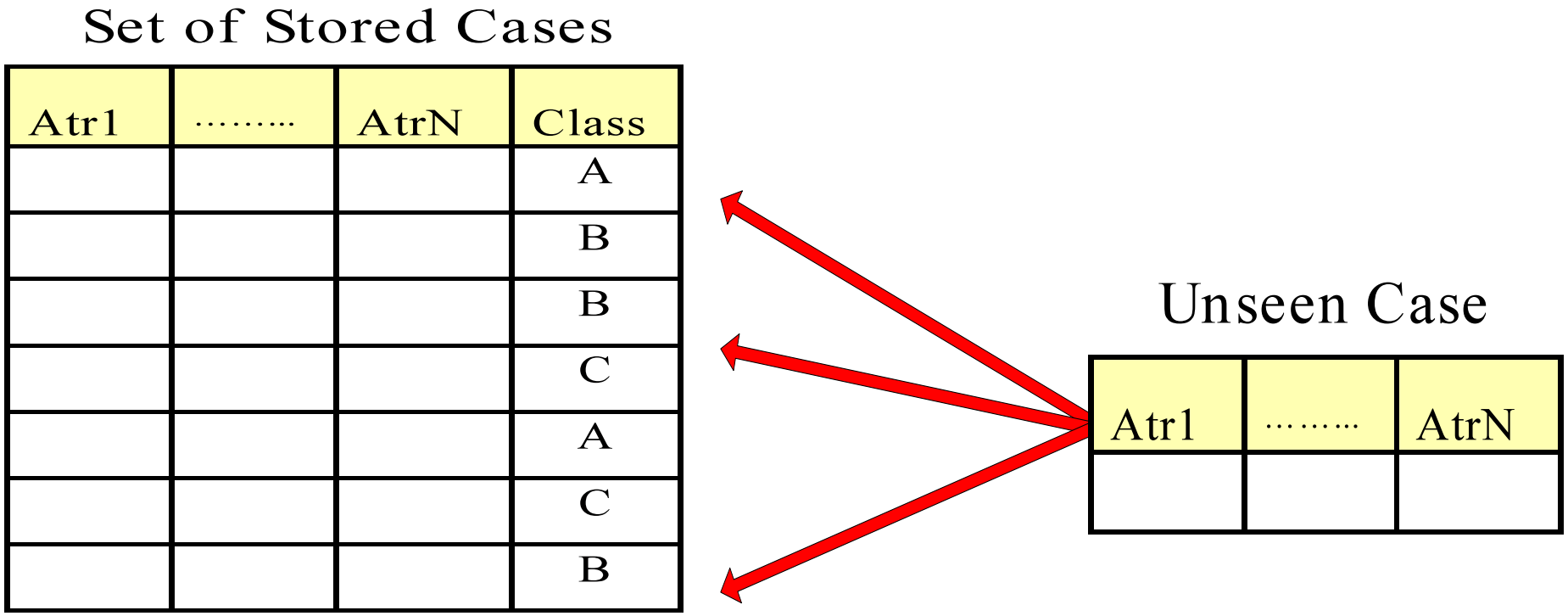
- Robust to isolated noise points
- Model each class separately
- Robust to irrelevant attributes
- Use the whole set of attribute to perform classification
- Independence assumption may not hold for some attributes

Instance-Based Classifier

- Lazy approach to classification
- Uses all the training set to peform classificaiton
- Uses distances between training and test records

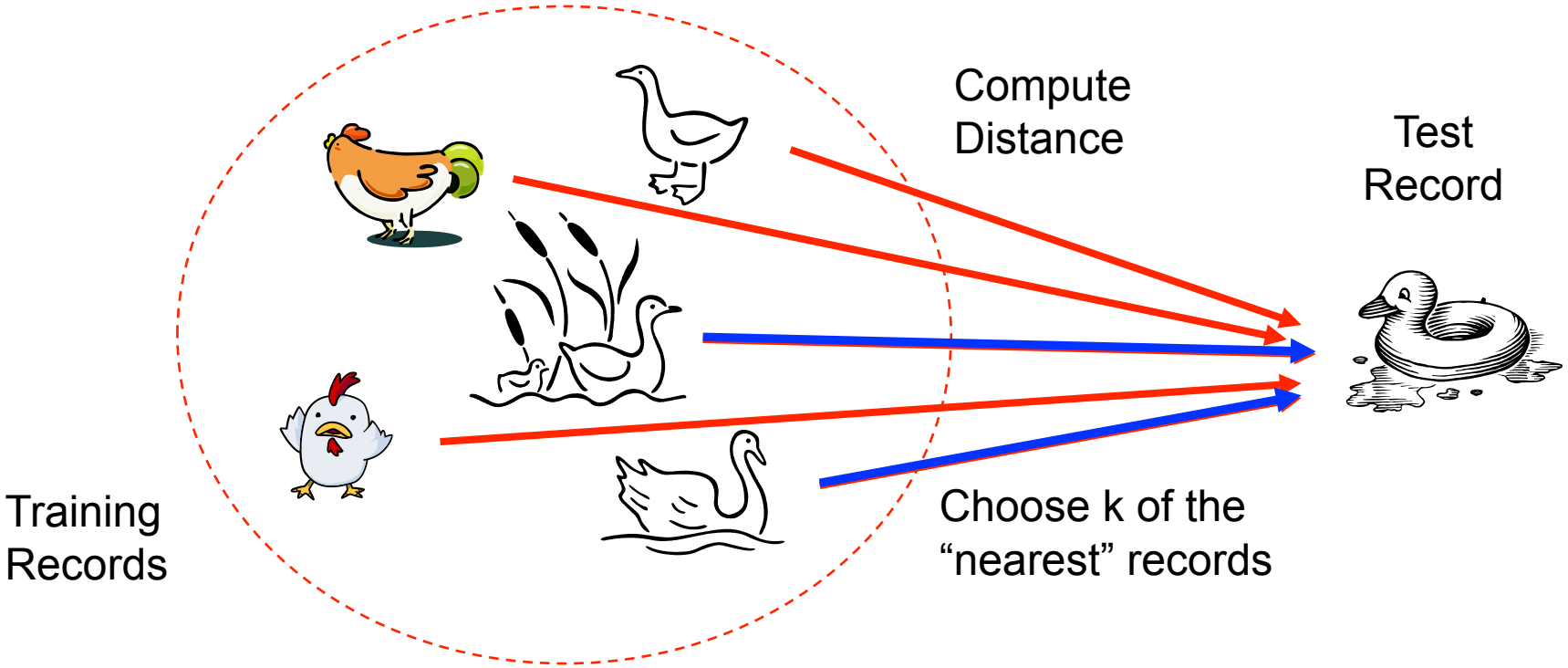
K-nearest neighbors

Uses k “closest” points (nearest neighbors) for performing classification

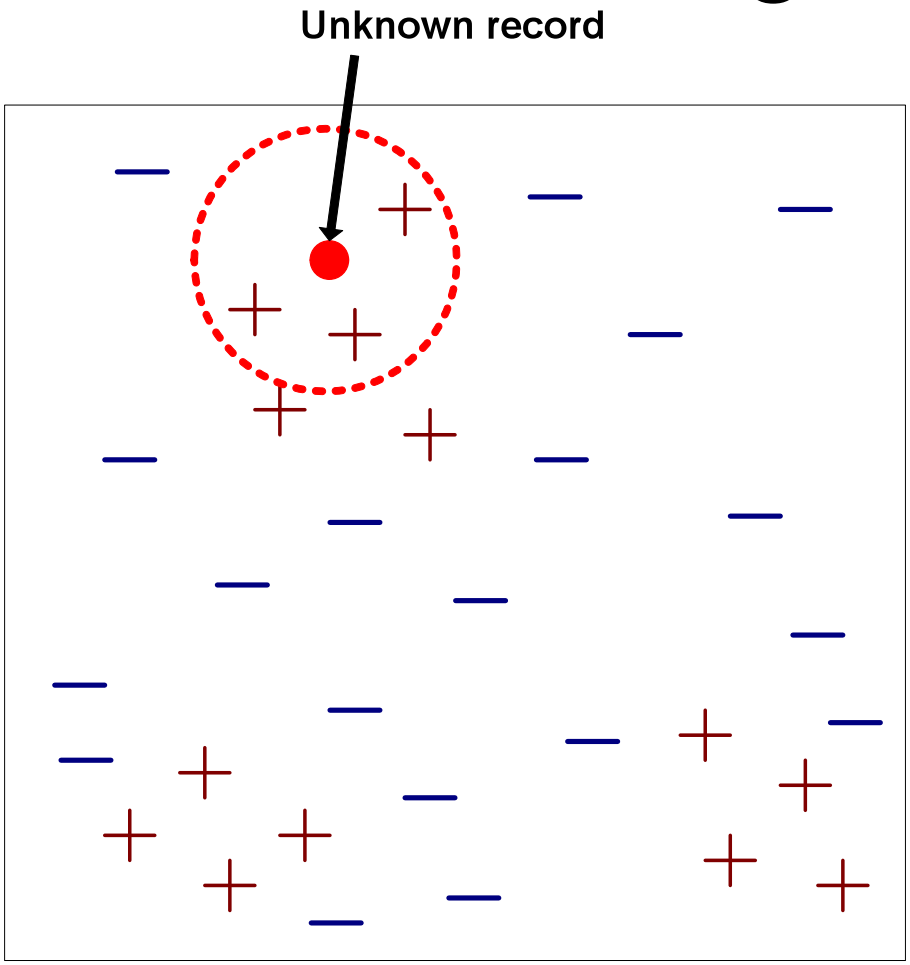


Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



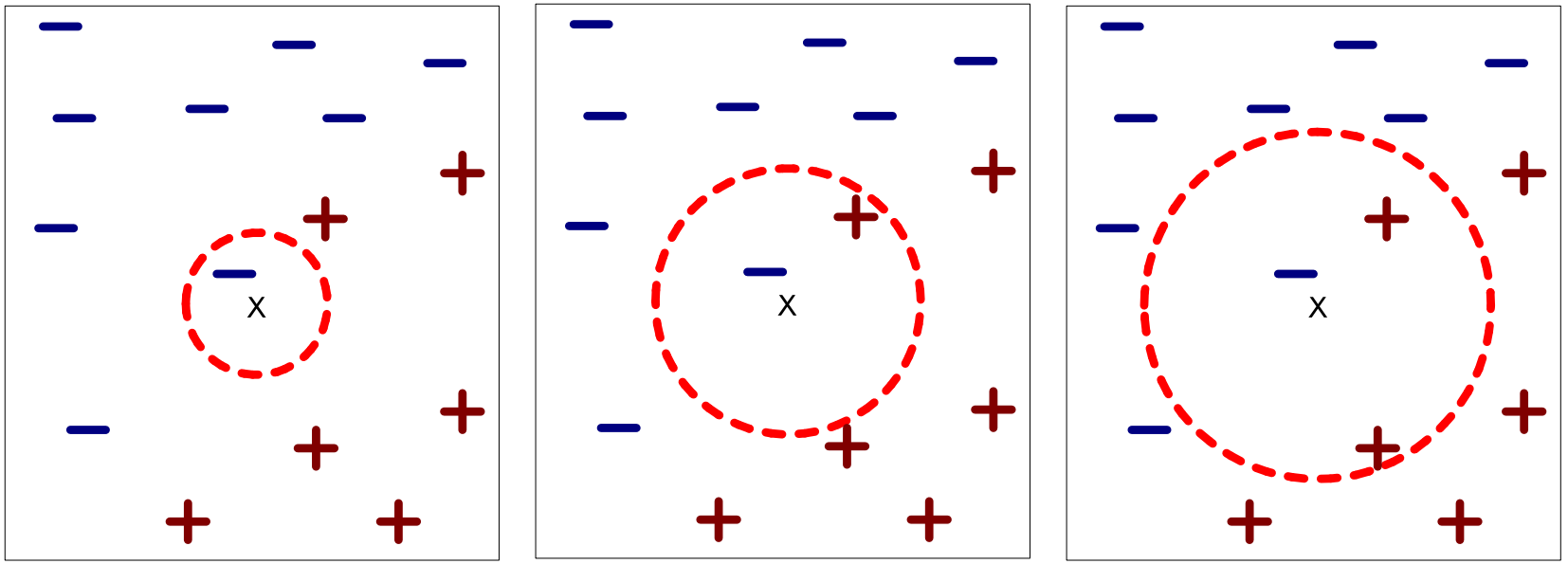
Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve

- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor



(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

Nearest Neighbor Classification

- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Example of K-NN Classification

Training set

Id record	Attr1	Attr2	Class
1	10	2	Yes
2	4	4	No
3	1	9	Yes
4	3	10	Yes
5	4	6	No
6	8	8	No
7	1	8	Yes

Test set

Id Record	Attr1	Attr2	Class
8	2	7	?
9	7	7	?
10	1	11	?

Example of K-NN Classification

Use K =3

Euclidean Dist	8	9	10
1	9.43	5.83	12.72
2	3.61	4.24	7.62
3	2.2	6.32	2.0
4	3.16	5.0	2.24
5	2.23	3.16	5.83
6	6.08	1.41	7.62
7	1.41	6.08	3.0

Id record	Class
1	Yes
2	No
3	Yes
4	Yes
5	No
6	No
7	Yes

3-NN(8) = {7,3,5} = {Yes, Yes, No} = Yes

3-NN(9) = ?

3-NN(10) = ?

Example of K-NN Classification

Use K =3

Euclidean Dist	8	9	10
1	9.43	5.83	12.72
2	3.61	4.24	7.62
3	2.2	6.32	2.0
4	3.16	5.0	2.24
5	2.23	3.16	5.83
6	6.08	1.41	7.62
7	1.41	6.08	3.0

Id record	Class
1	Yes
2	No
3	Yes
4	Yes
5	No
6	No
7	Yes

3-NN(8) = {7,3,5} = {Yes, Yes, No} = Yes

3-NN(9) = {6,5,2} = {No, No, No} = No

3-NN(10) = {3,4,7} = {Yes,Yes,Yes} = Yes

Example of K-NN Classification

Use K =5

Euclidean Dist	8	9	10
1	9.43	5.83	12.72
2	3.6	4.24	7.62
3	2.2	6.32	2.0
4	3.16	5.0	2.24
5	2.23	3.16	5.83
6	6.08	1.41	7.62
7	1.41	6.08	3.0

Id record	Class
1	Yes
2	No
3	Yes
4	Yes
5	No
6	No
7	Yes

3-NN(8) = {7,3,5,4,2} = {Yes, Yes, No, Yes, No} = Yes

3-NN(9) = ?

3-NN(10) = ?

Example of K-NN Classification

Use K =5

Euclidean Dist	8	9	10
1	9.43	5.83	12.72
2	3.61	4.24	7.62
3	2.2	6.32	2.0
4	3.16	5.0	2.24
5	2.23	3.16	5.83
6	6.08	1.41	7.62
7	1.41	6.08	3.0

Id record	Class
1	Yes
2	No
3	Yes
4	Yes
5	No
6	No
7	Yes

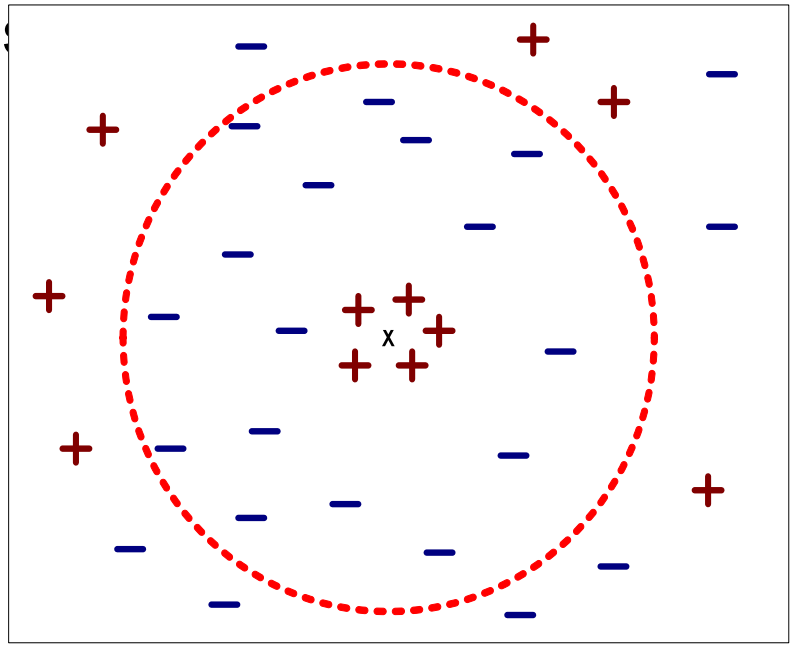
3-NN(8) = {7,3,5,4,2} = {Yes, Yes, No, Yes, No} = Yes

3-NN(9) = {6,5,2,4,1} = {No, No, No, Yes, Yes} = No

3-NN(10) = {3,4,7,5,2} = {Yes,Yes,Yes,No,No} = Yes

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Nearest Neighbor Classification…

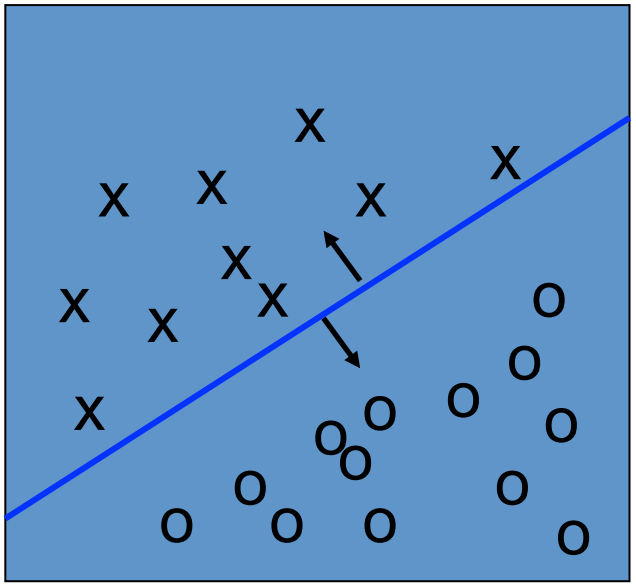
- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest neighbor Classification…

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive

Classification: A Mathematical Mapping

- Classification: predicts categorical class labels
  - E.g., Personal homepage classification
    - $x_i = (x_1, x_2, x_3, \ldots)$, $y_i = +1$ or -1
    - x_1 : # of word "homepage"
    - x_2 : # of word "welcome"
- Mathematically, $x \in X = \Re^n$, $y \in Y = \{+1, -1\}$,
  - We want to derive a function f: X → Y
- Linear Classification
  - Binary Classification problem
  - Data above the blue line belongs to class 'x'
  - Data below blue line belongs to class 'o'
  - Examples: SVM, Perceptron, Probabilistic Classifiers



SVM—Support Vector Machines

- A relatively new classification method for both linear and nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating **hyperplane** (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using **support vectors** (“essential” training tuples) and **margins** (defined by the support vectors)

SVM—History and Applications

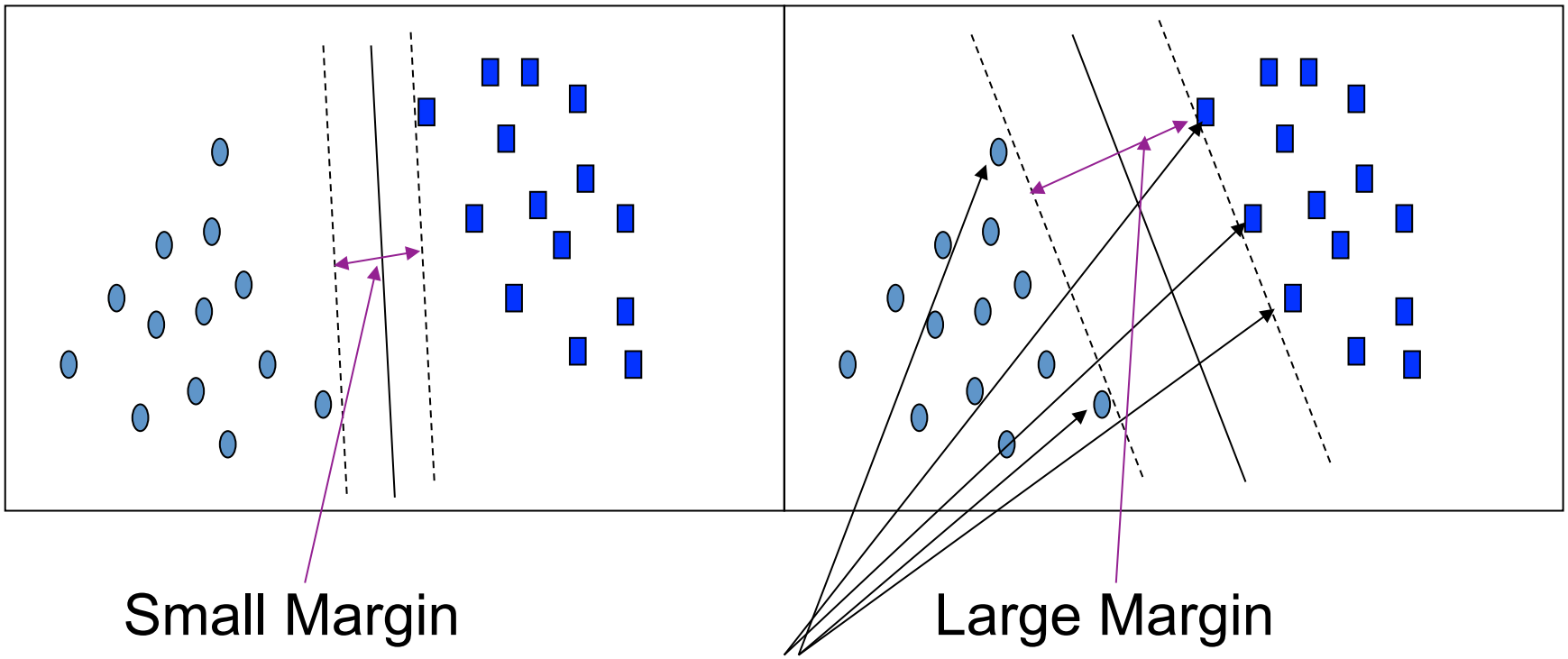
- Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis' statistical learning theory in 1960s

- Features: training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)

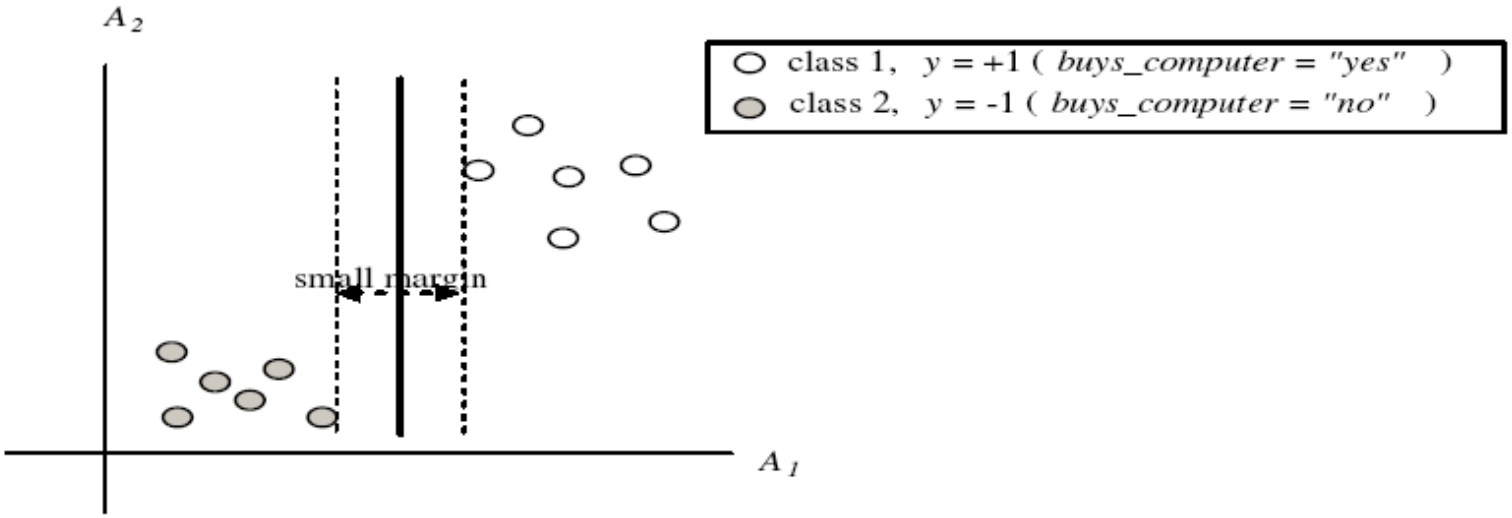
- Used for: classification and numeric prediction

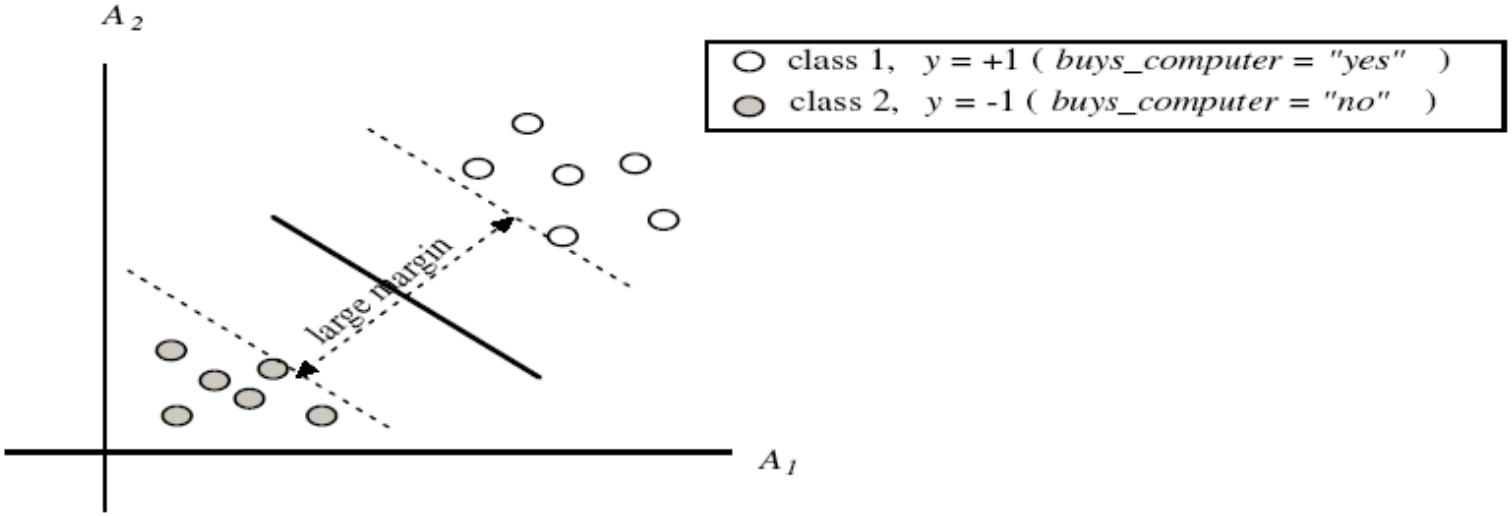
- Applications:
  - handwritten digit recognition, object recognition, speaker identification, benchmarking time-series prediction tests

SVM—General Philosophy

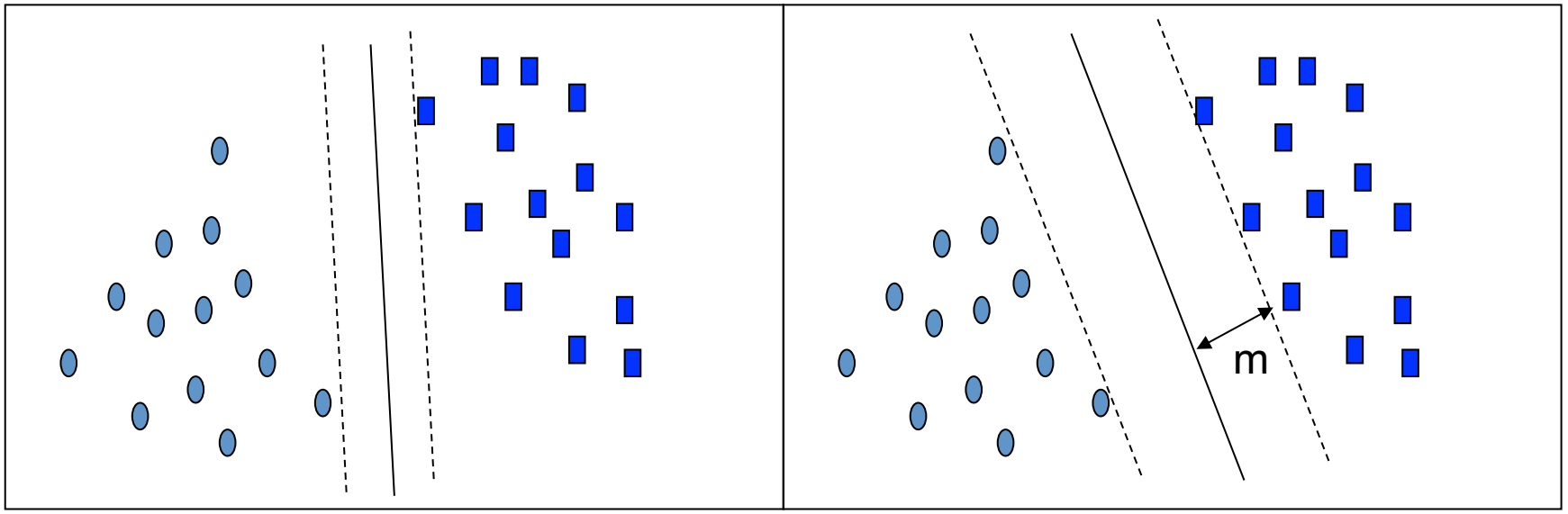


SVM—Margins and Support Vectors





SVM—When Data Is Linearly Separable



Let data D be ($\mathbf{X}_1$, y_1), …, ($\mathbf{X}_{|D|}$, $y_{|D|}$), where $\mathbf{X}_i$ is the set of training tuples associated with the class labels y_i

There are infinite lines (hyperplanes) separating the two classes but we want to find the best one (the one that minimizes classification error on unseen data)

SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane** (MMH)

SVM—Linearly Separable

- A separating hyperplane can be written as

  $\mathbf{W} \bullet \mathbf{X} + b = 0$

  where $\mathbf{W}=\{w_1, w_2, \ldots, w_n\}$ is a weight vector and b a scalar (bias)
- For 2-D it can be written as

  $w_0 + w_1 x_1 + w_2 x_2 = 0$
- The hyperplane defining the sides of the margin:

  $H_1: w_0 + w_1 x_1 + w_2 x_2 \geq 1$ for $y_i = +1$, and

  $H_2: w_0 + w_1 x_1 + w_2 x_2 \leq -1$ for $y_i = -1$
- Any training tuples that fall on hyperplanes H_1 or H_2 (i.e., the sides defining the margin) are **support vectors**
- This becomes a **constrained (convex) quadratic optimization** problem: Quadratic objective function and linear constraints → *Quadratic Programming (QP)* → Lagrangian multipliers

Why Is SVM Effective on High Dimensional Data? (I)

- The **complexity** of trained classifier is characterized by the # of support vectors rather than the dimensionality of the data

- The **support vectors** are the essential or critical training examples —they lie closest to the decision boundary (MMH)

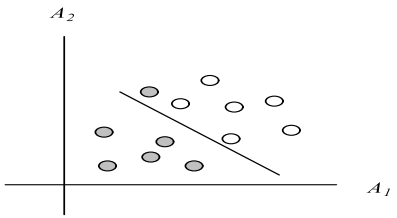
- If all other training examples are removed and the training is repeated, the same separating hyperplane would be found

Why Is SVM Effective on High Dimensional Data? (II)

- The number of support vectors found can be used to compute an (upper) bound on the expected error rate of the SVM classifier, which is independent of the data dimensionality

- Thus, an SVM with a small number of support vectors can have good generalization, even when the dimensionality of the data is high

SVM—Linearly Inseparable



- Transform the original input data into a higher dimensional space

Example 6.8 Nonlinear transformation of original input data into a higher dimensional space. Consider the following example. A 3D input vector $\mathbf{X} = (x_1, x_2, x_3)$ is mapped into a 6D space Z using the mappings $\phi_1(X) = x_1, \phi_2(X) = x_2, \phi_3(X) = x_3, \phi_4(X) = (x_1)^2, \phi_5(X) = x_1x_2$, and $\phi_6(X) = x_1x_3$. A decision hyperplane in the new space is $d(\mathbf{Z}) = \mathbf{WZ} + b$, where $\mathbf{W}$ and $\mathbf{Z}$ are vectors. This is linear. We solve for $\mathbf{W}$ and b and then substitute back so that we see that the linear decision hyperplane in the new $(\mathbf{Z})$ space corresponds to a nonlinear second order polynomial in the original 3-D input space,

$$\begin{aligned} d(Z) &= w_1x_1 + w_2x_2 + w_3x_3 + w_4(x_1)^2 + w_5x_1x_2 + w_6x_1x_3 + b \\ &= w_1z_1 + w_2z_2 + w_3z_3 + w_4z_4 + w_5z_5 + w_6z_6 + b \end{aligned}$$

■

- Search for a linear separating hyperplane in the new space

SVM: Different Kernel functions

- Instead of computing the dot product on the transformed data, it is math. equivalent to applying a kernel function K ($\mathbf{X}_i$, $\mathbf{X}_j$) to the original data, i.e., K($\mathbf{X}_i$, $\mathbf{X}_j$) = Φ($\mathbf{X}_i$) Φ($\mathbf{X}_j$)
- Typical Kernel Functions

$$\textbf{Polynomial kernel of degree } h: \quad K(X_i, X_j) = (X_i \cdot X_j + 1)^h$$

$$\textbf{Gaussian radial basis function kernel}: \quad K(X_i, X_j) = e^{-\|X_i - X_j\|^2/2\sigma^2}$$

$$\textbf{Sigmoid kernel}: \quad K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$$

- SVM can also be used for classifying multiple (> 2) classes and for regression analysis (with additional parameters)

SVM Classification…

- Hard to learn – learned in batch mode using quadratic programming techniques
- Using kernels can learn very complex functions
- Deterministic algorithm
- Nice generalization properties

Discriminative Classifiers

- Advantages
  - Prediction accuracy is generally high
    - As compared to Bayesian methods – in general
  - Robust, works when training examples contain errors
  - Fast evaluation of the learned target function

- Criticism
  - Long training time
  - Difficult to understand the learned function (weights)
  - Not easy to incorporate domain knowledge
    - Easy in the form of priors on the data or distributions

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.

- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation…

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

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

Cost Matrix

	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	**Class=Yes**	**Class=No**
	Class=Yes	C(Yes|Yes)	C(No|Yes)
	Class=No	C(Yes|No)	C(No|No)

C(i|j): Cost of misclassifying class j example as class i

Computing Cost of Classification

Cost Matrix	PREDICTED CLASS		
	C(i|j)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Cost vs Accuracy

Count	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

Cost	PREDICTED CLASS		
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	p	q
	Class=No	q	p

Accuracy is proportional to cost if
1. C(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

$N = a + b + c + d$

$Accuracy = (a + d)/N$

$$\begin{aligned} Cost &= p\,(a + d) + q\,(b + c) \\ &= p\,(a + d) + q\,(N - a - d) \\ &= q\,N - (q - p)(a + d) \\ &= N\,[q - (q-p) \times Accuracy] \end{aligned}$$

Cost-Sensitive Measures

Count	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F - measure (F)} = \frac{2a}{2a+b+c}$$

Model Evaluation

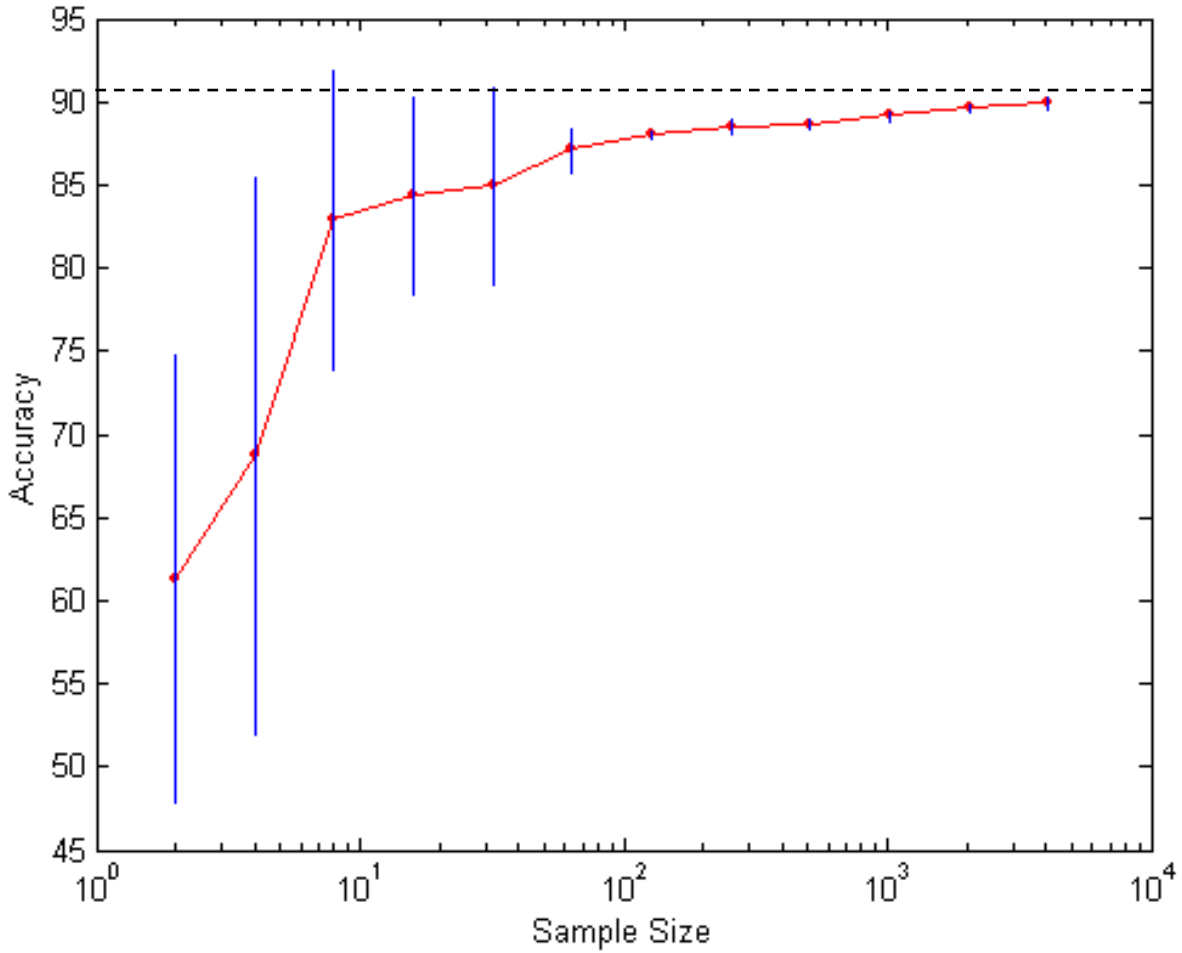
- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

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

Learning Curve



- Learning curve shows how accuracy changes with varying sample size
- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling (Langley, et al)
  - Geometric sampling (Provost et al)

Effect of small sample size:

- Bias in the estimate
- Variance of estimate

Methods of Estimation

- Holdout
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?
- Methods for Model Comparison
  - How to compare the relative performance among competing models?

ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms

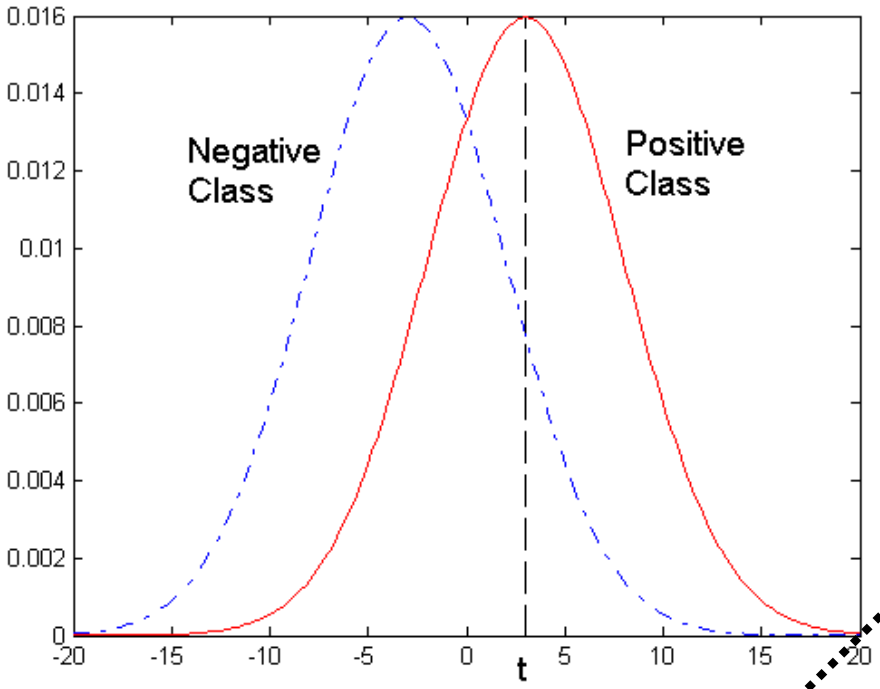
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)

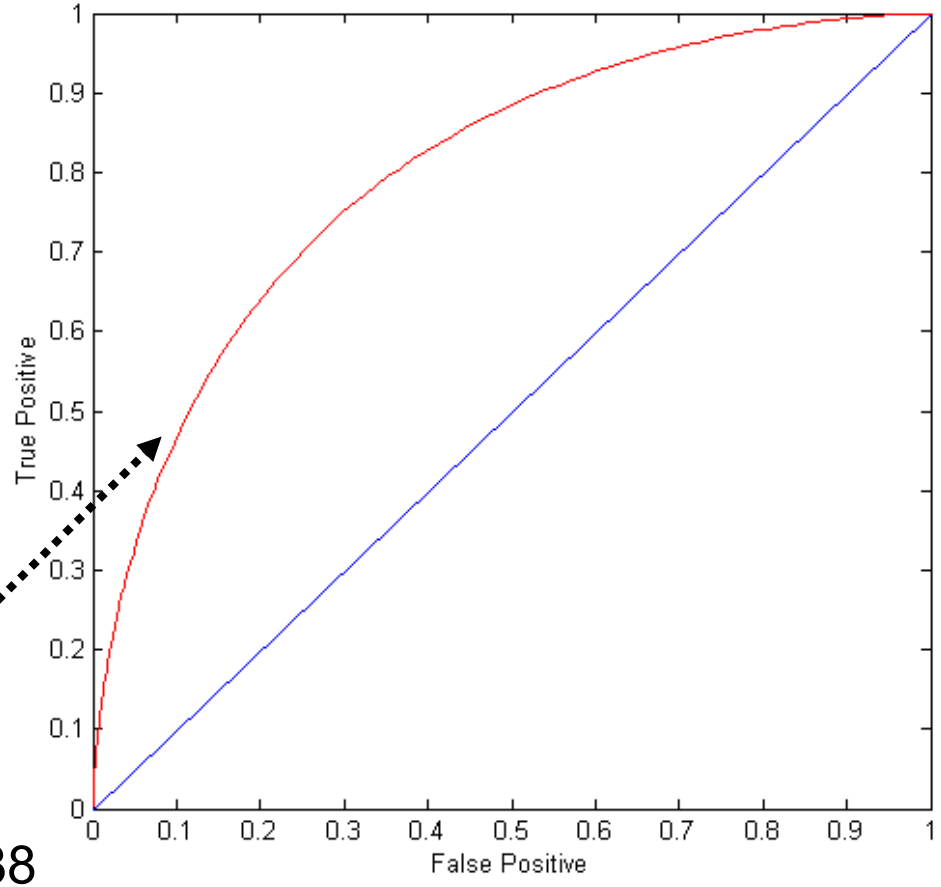
- Performance of each classifier represented as a point on the ROC curve
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)

- any points located at x > t is classified as positive





At threshold t:

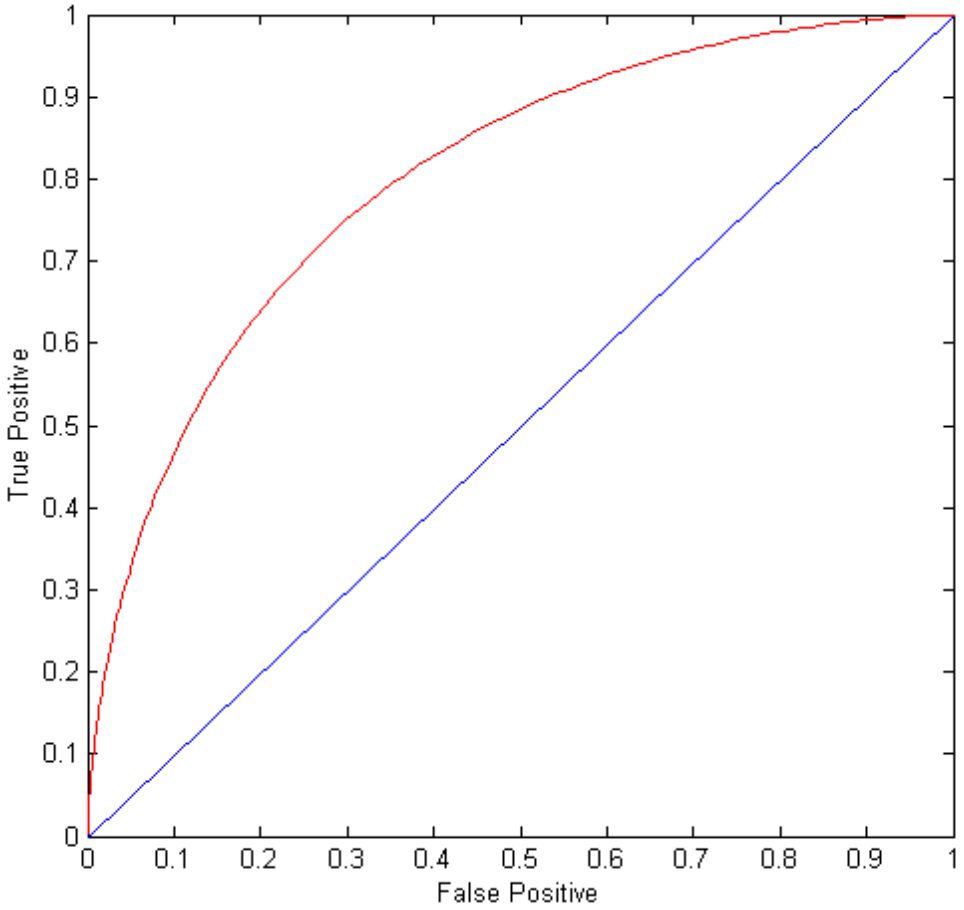
TP=0.5, FN=0.5, FP=0.12, FN=0.88

ROC Curve

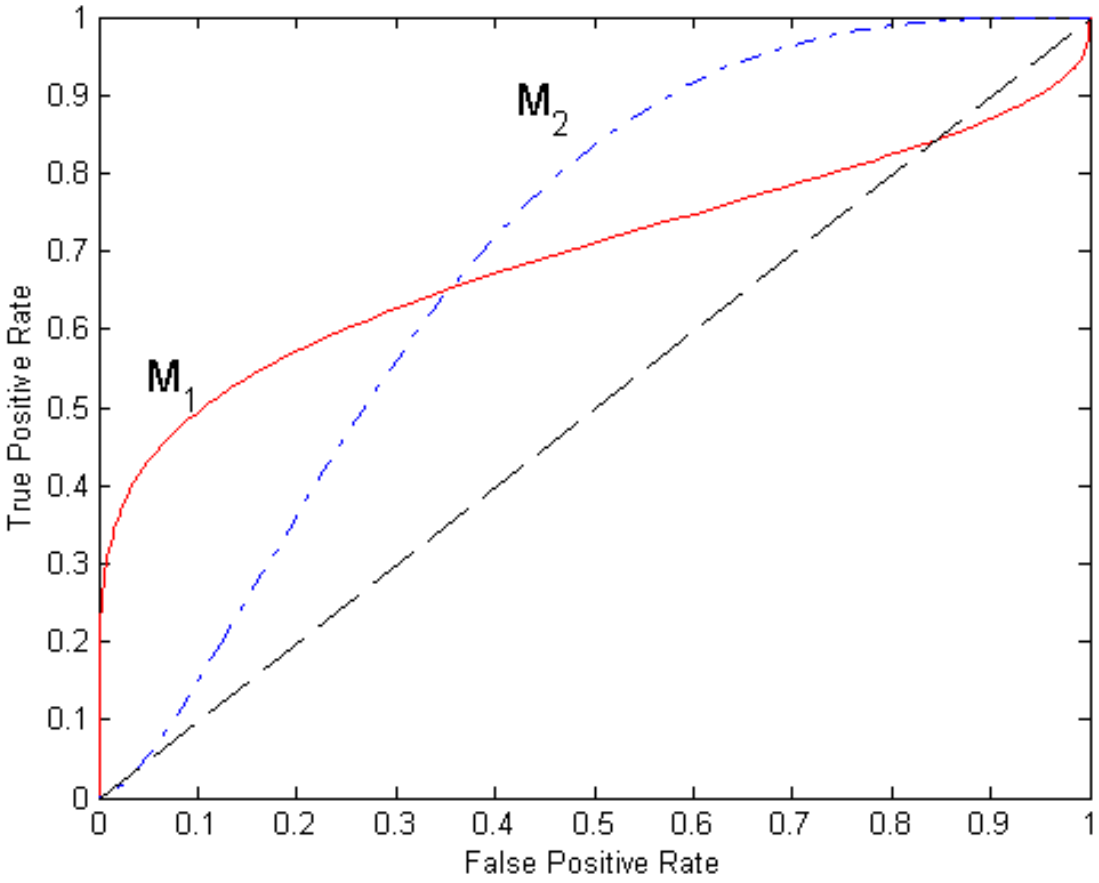
(TP,FP):

- (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



Using ROC for Model Comparison



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