CS522 Advanced Database Systems
Classification

Chengyu Sun
California State University, Los Angeles

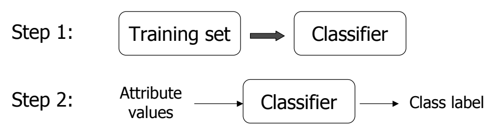
A Classification Problem

- Is a loan to a person who is 45 years old, divorced, renting an apartment, with two kids and annual income of 100K high risk or low risk?

Terminology and Concepts ...

- Record (or tuple)
  - Attributes
    - E.g. age, marital status, # of kids, owns home or not, credit score ...
  - Class label
    - E.g. high risk, low risk ...
- Classification: predict the class label with given attribute values

... Terminology and Concepts



- Classifier (or model)
- Training set: records with known class labels that are used to construct (i.e. *train*) the classifier

Classification vs. Regression

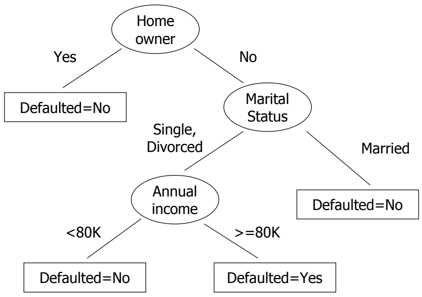
- Classification predicts categorical attribute values
- Regression predicts *continuous* numerical attribute values

SID	HW1	HW2	HW3	Final	Pass/Fail
1	40	60	70	95	Passed
2	10	15	11	65	Failed
3	30	45	40	75	Passed
4	35	50	35	?	?

A Training Set

TID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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

A Decision Tree



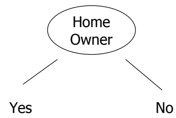
Decision Tree Induction

- Let D be the set of training record associated with current node
  - If all record in D belong to the same class C, current node is a leaf node and is labeled as C.
  - If D contains records that belong to more than one class, select an attribute to split D into subsets, and create a child node for each subset. *Apply the algorithm recursively on each child node.*

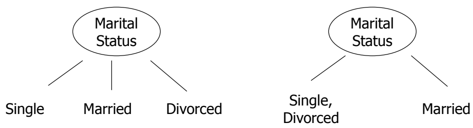
Terminating Conditions

- All records in D belong to the same class
- No more attribute to split
  - *Class label??*
- No records associated with the node
  - *Class label??*

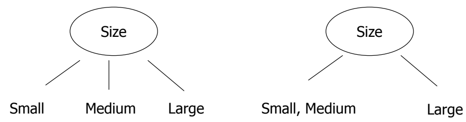
Split on Binary Attributes



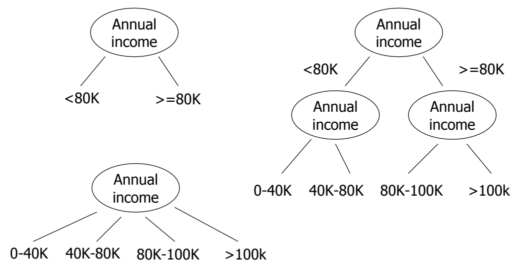
Split on Nominal Attributes



Split on Ordinal Attributes



Split on Continuous Attributes



Discretization of Numerical Data

- Number of partitions is known
  - Equi-width
  - Equi-depth
- Number of partitions is unknown
  - Recursive binary split
    - Determine the best split point
  - χ^2 merge
  - Intuitive partitioning

χ^2 Merge Example ...

	Y	N
70-70	0	1
75-75	0	1

	Y	N
75-75	0	1
85-85	1	0

... χ^2 Merge Example

	Y	N
70-70	0(0.3)	1(0.7)
75-75	0(0.3)	1(0.7)

	Y	N
75-75	0(0.3)	1(0.7)
85-85	1(0.3)	0(0.7)

$\chi_a^2 = (0-0.3)^2/0.3 + (0-0.3)^2/0.3 + (1-0.7)^2/0.7 + (1-0.7)^2/0.7 = 0.857$

$\chi_b^2 = (0-0.3)^2/0.3 + (1-0.3)^2/0.3 + (1-0.7)^2/0.7 + (0-0.7)^2/0.7 = 1.429$

So which pair to merge??

Intuitive Partitioning

- The 3-4-5 Rule
  - 3, 6, 7, or 9 distinct values at the most significant digit ➔ 3 equi-width intervals
  - 2,4,8 ➔ 4 equi-width intervals
  - 1,5,10 ➔ 5 equi-width intervals
- Example: 60 70 75 85 90 95 100 120 125 220
  - *Intervals??*

Splitting Attribute Selection

- After a split, ideally each subset would "pure", i.e. contains only one class of records

Gender	Age	Preferred color
female	20	pink
male	20	black
female	15	pink
male	15	black

Attribute Selection Measures

- Entropy (Information Gain)
- Gini index
- Gain Ratio

Entropy

$$Entropy(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- p_i is the fraction of records in D that belongs to class C_i
- m is the number of classes in D

Entropy Example

- Preferred color
  - 2 black and 2 pink??
  - 3 black and 1 pink??
  - 4 black??

Information Gain

- Suppose D is split into v subsets on attribute A

$$Gain(A) = Entropy(D) - \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Entropy(D_j)$$

Information Gain Example

- Preferred color
  - Gain(Gender)??
  - Gain(Age)??

Split Information

- Information gain favors attributes with lots of distinct values
- *Split information* can be used to "normalized" information gain

$$SplitInfo(A) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

Gain Ratio

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

Gini Index

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

- Used in the CART algorithm for *binary split*

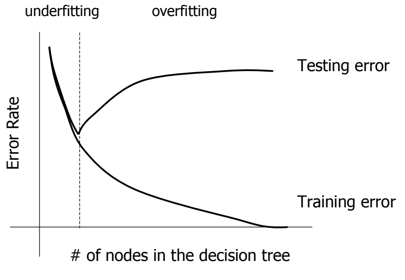
Gini Index Example

- Preferred color
  - 2 black and 2 pink??
  - 3 black and 1 pink??
  - 4 black??
  - Split on gender??
  - Split on age??

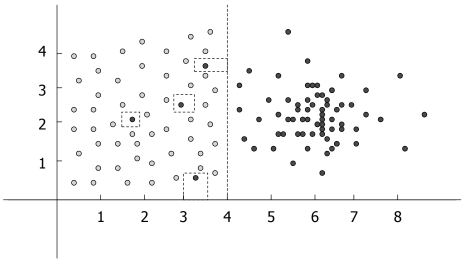
Training Error and Testing Error

- Training error
  - Misclassification of training records
- Testing (Generalization) error
  - Misclassification of testing records

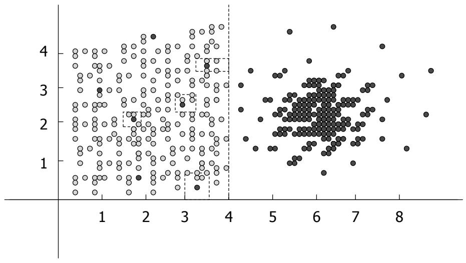
Model Overfitting and Underfitting



Overfitting Due to Outliers/Noise ...



... Overfitting Due to Outliers/Noise

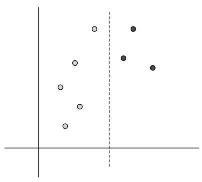


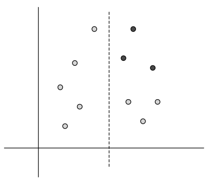
Occam's Razor

- A.K.A. Principle of Parsimony
- Given two models with the same generalization errors, the simpler model is preferred over the more complex model

Tree Pruning – Prepruning

- Prune during decision tree construction
  - Number of records < threshold
  - "Purity gain" < threshold





Tree Pruning – Postpruning

- Buttom-up pruning of a fully constructed tree
  - Replace a subtree with a leaf node if it reduces testing error
    - *How do we know whether it reduces testing error or not??*
  - Pruning based on Minimum Description Length (MDL)

Estimate Testing Errors ...

- Use a *pruning/validation set* in addition to the training set
  - Usually 1/3 of the original training set
  - Good for algorithms that can be parameterized to obtains models with different levels of complexity

... Estimate Testing Errors

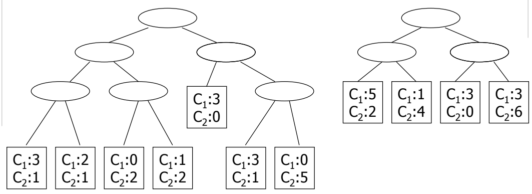
- Optimistic error estimation
  - The training set is a good representation of the overall data (optimistic!), so the training error is the testing error
- Pessimistic error estimation
  - Training error + penalty term for model complexity

Pessimistic Error Estimation

$$e_g(T) = \frac{\sum_i [e(t_i) + \Omega(t_i)]}{\sum_i n(t_i)} = \frac{e(T) + \Omega(T)}{N}$$

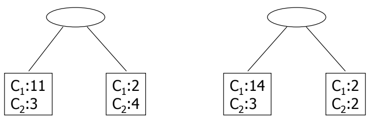
- T – A decision tree
- $n(t_i)$ – # of training records at leaf node t_i
- $e(t_i)$ – # of misclassified training records at t_i
- $\Omega(t_i)$ – Penalty term associated with t_i

Example of Pessimistic Error Estimation



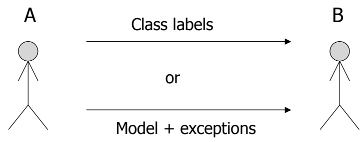
- $e_g(T)$ with $\Omega(t_i)$=0.5?? $\Omega(t_i)$=1??
- Range of Ω for a binary decision tree??

Pruning with Estimated Testing Error



- With optimistic error estimation??
- With pessimistic error estimation??
- With a pruning set??

Minimum Description Length (MDL)

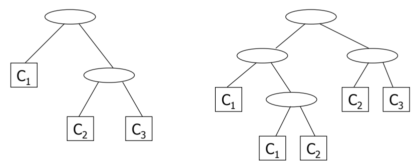


- The best model is the one that minimizes the number of bits to encode both the model and the exceptions to the model

MDL Example ...

- n records
- m binary attributes
- k classes
- Cost(Internal Node) = $\log_2 m$
- Cost(Leaf node) = $\log_2 k$
- Cost(Error) = $\log_2 n$
- Cost = Cost(All Nodes)+Cost(All Errors)

... MDL Example



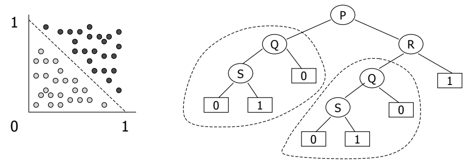
- 16 binary attributes and 3 classes
- Left tree has 7 errors and right tree has 4 errors

About Decision Tree Classification ...

- 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

... About Decision Tree Classification

- Data fragmentation
- Finding an optimal decision tree is NP-hard
- Limitation on expressiveness
- Tree replication



Rule-based Classification Example ...

- The Vertebrate dataset

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

... Rule-based Classification Example

- The Rules

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

©Tan, Steinbach, Kumar Introduction to Data Mining 2004

Terminology

Rule set: $R = (r_1 \vee r_2 \vee ... \vee r_k)$

Rule: r_i: (Condition$_i$) $\rightarrow c_i$

- Condition$_i$=(A_1 op v_1) $\wedge$ (A_2 op v_2) $\wedge$ ... $\wedge$ (A_k op v_k)
  - Rule antecedent, precondition
  - **Conjunct**: (A_i op v_i), op$\in\{=,\neq,<,>,\leq,\geq\}$
- c_i
  - Class label
  - Rule consequent

Coverage and Accuracy

- A rule r *covers* a record x if the precondition of r matches the attributes of x
  - A.K.A. r is *fired/triggered* by x
- Coverage(r) = |A| / |D|
  - |A|: # of records covered by r
- Accuracy(r) = |A$\cap$y| / |A|
  - |A$\cap$y|: # of records that satisfy both the antecedent and consequent of r
- Example
  - *coverage and accuracy of r3??*

How a Rule-based Classifier Works

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	?

- Lemur: ??
- Turtle: ??
- Dogfish shark: ??

Two Properties of a Rule-based Classifier

- Exhaustive Rules
  - Every combination of the attribute values is covered by at least one rule
- Mutually Exclusive Rules
  - No two rules are triggered by the same record

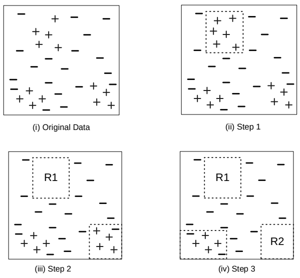
Make a Rule Set Exhaustive/Mutually Exclusive

- Default rule: $() \rightarrow c_d$
- Ordered rules
  - Quality-based ordering
  - Class-based ordering
- Unordered rules
  - Majority votes
    - Weighted by the rule's accuracy

The Sequential Covering Algorithm

- Order the classes $\{c_1, c_2, ..., c_k\}$
- For each class c_i, $i<k$
  - Find the best rule r for c_i
  - Remove the records covered by r
  - Add r to the rule list
  - Repeat until the stop condition is met
- Add a default rule $() \rightarrow c_k$

Sequential Covering Example



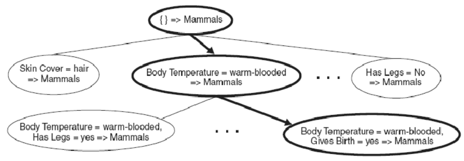
Ordering Classes and Rules

- Class ordering
  - Based on frequency
- Rule ordering
  - Based on classes
  - Based on quality of the rules

Rule Growing

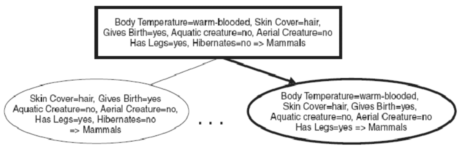
- From general to specific
  - Start with $() \rightarrow c_i$
  - *Greedily* add one conjunct at a time
- From specific to general
  - Start with any positive record
  - *Greedily* remove one conjunct at a time
- Augmented by *beam search* with k best candidates

Rule Growing Example (a)



(a) General-to-specific

Rule Growing Example (b)



(b) Specific-to-general

Rule Evaluation

- Decide which conjunct should be added (or removed)

Rule Evaluation Example

- A training set contains 60 records in class c_1 and 100 records in class c_2
- Compare two rules
  - r_1: covers 50 c_1 and 5 c_2
  - r_2: covers 2 c_1 and 0 c_2

Rule Evaluation Measure (a)

- Observed frequency vs. expected frequency

$$R(r) = 2\sum_{i=1}^{k} f_i \log(f_i / e_i)$$

For r1:

$f_1 = 50$, $f_2 = 5$

$e_1 = 55 \times 60/160$, $e_2 = 55 \times 100/160$

Rule Evaluation Measure (b)

- Accuracy and coverage

$$m-estimate(r)=\frac{n_c+kp_c}{n+k} \quad Laplace(r)=\frac{n_c+1}{n+k}$$

n: number of records covered by r
n_c: number of positive records covered by r
k: number of classes
p_c: prior probability of class c

Rule Evaluation Measure (c)

- Accuracy improvement and coverage

FOIL's information gain:

$$FGain(r)=n_c'\times(\log_2\frac{n_c'}{n'}-\log_2\frac{n_c}{n})$$

Stop Conditions

- Stop growing a rule
- Stop adding a rule for class c_i
  - Minimum Description Length (MDL)

Rule Pruning

- Similar to post-pruning of decision trees
- Remove a conjunct if the accuracy rate improves based on a validation set

Indirect Rule Extraction

- Using decision tree
  - Rule generation
  - *Exhaustive?? Mutually Exclusive??*
- Using association rule mining
  - Find association rules in the form of $\mathbf{A}\rightarrow c_i$
  - Select a subset of the rules to form a classifier
    - Sort the rules based on confidence, support, and length
    - Add to a rule list one at a time
    - Add a default rule

Probabilistic Relationship between Attributes and Class

- Ten middle-aged, divorced, male borrowers have defaulted on their loans, but would the 11th one default as well?

Bayes' Theorem

$$P(A|B)=\frac{P(B|A)P(A)}{P(B)}$$

- Prior and posterior probabilities
  - P(A) and P(A|B)
  - P(B) and P(B|A)

Bayesian Classification

$$P(C_i|\mathbf{X})=\frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- **X** is a given record with attribute values $(x_1, x_2, ..., x_n)$, and C_i is a class
- $P(C_i|\mathbf{X})$ is the probability of **X** belonging to class C_i given **X**'s attribute values
- We predict that **X** belong to C_i if $P(C_i|X)>P(C_j|X)$ for $j\neq i$

Calculate P(C_i|**X**)

- P(X) does not need to be calculated
  - *Why??*
- P(C_i)??
- P(**X**|C_i)??

Naive Bayesian Classification

- **X**=(x_1,x_2,...,x_n)
- Assume the attribute values are conditionally independent of one another (the *naive* assumption)

$$P(\mathbf{X}|C_i)=\prod_{i=1}^{n}P(x_i|C_i)$$
$$=P(x_1|C_i)\times P(x_2|C_i)\times\cdots\times P(x_n|C_i)$$

Attribute A_k is Categorical

- $P(x_k|C_i)$ is the fraction of number of records in C_i with value x_k for attribute A_k

Attribute A_k is Continuous-valued ...

- Assume A_k follows a Gaussian distribution with a mean μ and standard deviation σ

$$\mu=\frac{1}{N}\sum_{i=1}^{N}x_i \qquad s=\sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i-\mu)^2}$$

s: sample standard deviation

... Attribute A_k is Continuous-valued

$$g(x,\mu,\sigma)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(x_k \mid C_i) \rightarrow g(x_k,\mu_{c_i},\sigma_{c_i})$$

Sample Data

TID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
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
11	No	Married	120K	??

Naive Bayesian Classification Example ...

- P(Default=N|HO=N,MS=M,AI=120K)
- P(Default=Y|HO=N,MS=M,AI=120K)

... Naive Bayesian Classification Example

- Annual Income, Default=No
  - μ=110, σ=54.54
  - P(AI=120K|No)=0.0072
- Annual Income, Default=Yes
  - μ=90, σ=5
  - P(AI=120K|Yes)=1.2×10^{-9}

Avoid Zero $P(x_k|C_i)$

- A zero $P(x_k|C_i)$ would make the whole $P(\mathbf{X}|C_i)$ zero
- To avoid this problem, add 1 to to each count, assuming the training set is sufficiently large that the effect of adding one is negligible
- Example
  - Low income:0
  - Medium income: 990
  - High income: 10

About Naive Bayesian Classification

- The most accurate classification *if the conditional independence assumption holds*
- In practice, some attributes may be correlated
  - E.g. education level and annual income

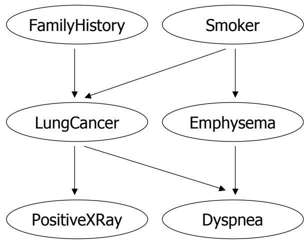
Bayesian Belief Network (BBN)

- A *directed acyclic graph* (dag) encoding the dependencies among a set of variables
- A *conditional probability table* (CPT) for each node given its immediate parent nodes

A BBN Example

CPT for LungCancer

	Yes	No
FH,S	0.8	0.2
FH,!S	0.5	0.5
!FH,S	0.7	0.3
!FH,!S	0.1	0.9



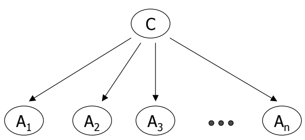
BBN Terminology

- If there is a directed *arc* from X to Y
  - X is a parent of Y
  - Y is a child of X
- If there is a directed *path* from X to Y
  - X is an ancestor of Y
  - Y is a descendent of X

Conditional Independence in BBN

- A node in a Bayesian network is conditionally independent of its non-descendants if its parents are known

Naive Bayesian is a Special Case of BBN



Construct a BBN

- Create the structure of the network
  - From domain knowledge
  - From training data
- Calculate the CPT for each node X
  - P(X) if X does not have any parent
  - P(X|Y) if X has one parent Y
  - $P(X|Y_1,Y_2,\ldots,Y_k)$ if X has multiple parents $\{Y_1,Y_2,\ldots,Y_k\}$

Another BBN Example

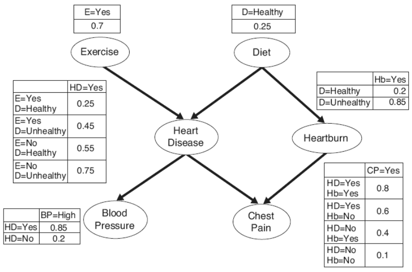


Figure 5.13. A Bayesian belief network for detecting heart disease and heartburn in patients.

Bayesian Classification Examples

- ◈Output node – Heart Disease
- ◈Testing data
  - ()
  - (BP=high)
  - (BP=high,D=Healthy,E=Yes)

Bayesian Classification Examples – 1

$$P(HD=Yes)=\sum_{i=1}^{n}\sum_{j=1}^{m}P(HD=Yes\mid E=a_i,D=b_j)P(E=a_i,D=b_j)$$
$$=\sum_{i=1}^{n}\sum_{j=1}^{m}P(HD=Yes\mid E=a_i,D=b_j)P(E=a_i)P(D=b_j)$$
$$=0.49$$

Bayesian Classification Examples – 2

$$P(HD=Yes\mid BP=High)$$
$$=\frac{P(BP=High\mid HD=Yes)P(HD=Yes)}{P(BP=High)}$$
$$=\frac{P(BP=High\mid HD=Yes)P(HD=Yes)}{\sum_{i=1}^{n}P(BP=High\mid HD=a_i)P(HD=a_i)}$$
$$=0.80$$

Bayesian Classification Examples – 3

$$P(HD=Yes\mid BP=High,D=Healthy,E=Yes)$$
$$=\frac{P(BP=High\mid HD=Yes,D=Healthy,E=Yes)P(HD=Yes\mid D=Healthy,E=Yes)}{P(BP=High\mid D=Healthy,E=Yes)}$$
$$=\frac{P(BP=High\mid HD=Yes)P(HD=Yes\mid D=Healthy,E=Yes)}{\sum_{i=1}^{n}P(BP=High\mid HD=a_i)P(HD=a_i\mid D=Healthy,E=Yes)}$$
$$=0.59$$

About BBN

- ◈Does not assume attribute independence
- ◈Provides a way to encode domain knowledge
  - Robust to model overfitting
- ◈Any node can be used an output node

Bayes Error Rate

- If the relationship between attributes and class is probabilistic, it is impossible to be 100% correct.
- Bayes Error Rate – minimum achievable error rate for a given classification problem

Bayes Error Rate Example ...

- Identify alligators and crocodiles based on their lengths X
- P(X|Crocodile) is Gaussian with μ=15 and σ=2
- P(X|Alligator) is Gaussian with μ=12 and σ=2

...Bayes Error Rate Example...

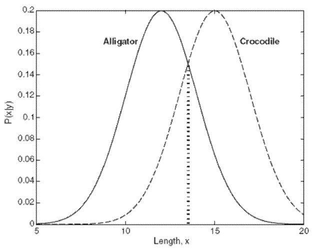


Figure 5.11. Comparing the likelihood functions of a crocodile and an alligator.

... Bayes Error Rate Example

$$Error = \int_0^{\hat{x}} P(X \mid Crocodile)dX + \int_{\hat{x}}^{\infty} P(X \mid Alligator)dX$$

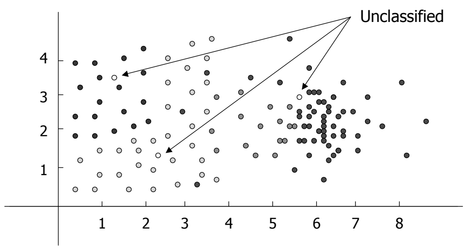
⇩

$$P(X = \hat{x} \mid Crocodile) = P(X = \hat{x} \mid Alligator)$$

⇩

$$\hat{x} = 13.5$$

k Nearest Neighbor (kNN) Classification Example



- *What is the class of each unclassified sample??*

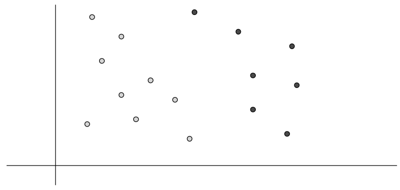
kNN Classification

- Find the k nearest neighbors of the test sample
- Classify the test sample with the majority class of its k nearest neighbors

About kNN

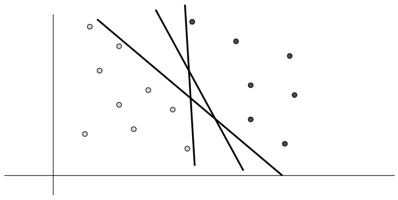
- Similarity/distance measures
  - More on this when we talk about clustering
- Index structures
- Local decision – susceptible to noise
- Error rate <= (2 * Bayes Error Rate) if k=1 and $n \rightarrow \infty$

Support Vector Machine (SVM)



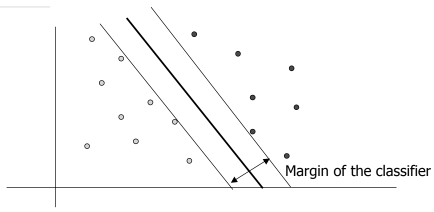
- Find a hyperplane (decision boundary) that will separate the data.

Which Hyperplane to Choose?



- There are an infinite number of hyperplanes with zero training error.

Maximum Margin Hyperplanes

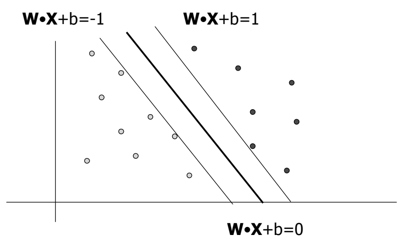


- Maximum margin hyperplane minimizes the worst-case generalization error.

Linear SVM ...

- Binary classification
- Record: $\{x_1, x_2, \ldots, x_n, y\}$
  - Attribute values: $\mathbf{X}=(x_1, x_2, \ldots, x_n)$
  - Class label: $y \in \{1, -1\}$
- Decision boundary: $\mathbf{W} \bullet \mathbf{X}+b=0$
- Classification
  - $y=1$ if $\mathbf{W} \bullet \mathbf{X}+b > 0$
  - $y=-1$ if $\mathbf{W} \bullet \mathbf{X}+b < 0$

... Linear SVM



$$\text{Margin} = \frac{2}{\|W\|} = \frac{2}{\sqrt{w_1^2 + w_2^2 + \cdots + w_n^2}}$$

Training Linear SVM

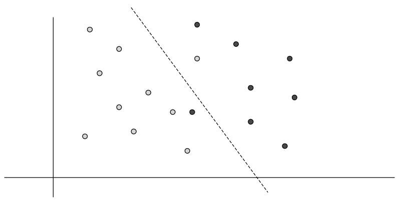
- Maximize margin given the conditions $y_i(\mathbf{W} \bullet \mathbf{X_i} + b) \geq 1$ for all training records
  - Constrained (convex) quadratic optimization problem
  - Solvable by numerical methods such as quadratic programming

Decision Boundary of Linear SVM

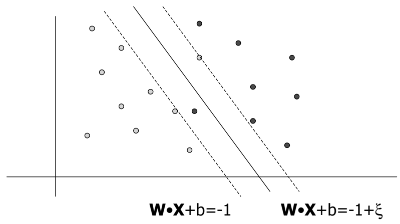
$$(\sum_{i=1}^{N} \lambda_i y_i \mathbf{X_i} \bullet \mathbf{X}) + b = 0$$

- $(\mathbf{X_i}, y_i)$ are training records that satisfy $y_i(\mathbf{W} \bullet \mathbf{X_i} + b) = 1$ ➔ Support Vectors

Linear SVM – Non-separable Case



Introducing a Slack Variable ξ



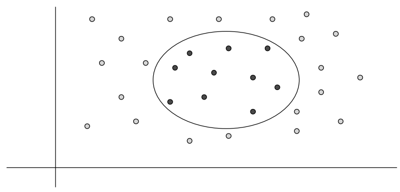
Training Non-separable Linear SVM

- Minimize the following objective function under the constraint $y_i(\mathbf{W} \bullet \mathbf{X_i} + b) \geq (1 - \xi)$

$$f(\mathbf{W}) = \frac{\|\mathbf{W}\|^2}{2} + C(\sum_{i=1}^{N} \xi_i)^k$$

C and k are user-specified parameters representing the penalty of misclassifying the training records.

Non-linear Decision Boundary



- Transform the data to another coordinate space so a linear boundary can be found

Transformation Example

Non-linear Decision Boundary in 2D space:

$$(x_1 - 1)^2 + (x_2 - 1)^2 - 1 = 0$$

$x'_1 = x_1$
$x'_2 = x_2$
$x'_3 = x_1^2$
$x'_4 = x_2^2$

Linear Decision Boundary in 4D space:

$$x'_3 - 2x'_1 + x'_4 + 2x'_2 + 1 = 0$$

Problems of Transformation

- We don't know the non-linear decision boundary (so we don't know how to do the transformation)
- Computation becomes more costly with more dimensions

Kernel Function to the Rescue

- Training records only appear in the optimization process in the form of dot product $\phi(\mathbf{X_i}) \bullet \phi(\mathbf{X_j})$
  - ϕ is the transformation function
- Kernel function $K(\mathbf{X_i}, \mathbf{X_j}) = \phi(\mathbf{X_i}) \bullet \phi(\mathbf{X_j})$
- So we can do the computation in the original space *without even knowing what the transformation function is*

Kernel Functions

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

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

Sigmoid kernel: $K(\mathbf{X_i}, \mathbf{X_j}) = \tanh(\kappa \mathbf{X_i} \bullet \mathbf{X_j} - \delta)$

Kernel Functions and SVM Classifiers

- Use of different kernel functions result in different classifiers
- There's no golden rule to determine which kernel function is better
- The accuracy difference by using different kernel functions is usually not significant in practice

Multiclass Classification with a Binary Classifier ...

- For k classes $\{c_1, c_2, \ldots, c_k\}$, train k binary classifiers, each classifies $\{c_i, \text{not-}c_i\}$
- *So how does the classification work??*

... Multiclass Classification with a Binary Classifier

- Positive classification by classifier $\{c_i, \text{not-}c_i\}$ ➔ one vote for c_i
- Negative classification by classifier $\{c_i, \text{not-}c_i\}$ ➔ one vote for each c_j where $j \neq i$
- Example:

c_1	c_2	c_3	c_4
+	-	-	-

Error-Correcting Output Coding (ECOC) Example

Class	Codeword
c_1	1 1 1 1 1 1 1
c_2	0 0 0 0 1 1 1
c_3	0 0 1 1 0 0 1
c_4	0 1 0 1 0 1 0

- Classifiers' output: 0 1 1 1 1 1 1

Error-Correcting Output Coding (ECOC)

- Encode each class label with a n-bit code word
- Train n binary classifiers, one for each bit
- The predicted class is the one whose codeword is the closest in Hamming distance to the classifiers' output

About ECOC

- If d is the minimum distance between any pair of code words, ECOC can correct up to $\lfloor (d-1)/2 \rfloor$ errors
- There are many algorithms in coding theory to generate n-bit code words with given Hamming distance
- For multiclass classification, column-wise separation is also important

Other Classification Methods

- Rule-based
- Artificial Neural Network (ANN)
- Association rule analysis
- Genetic algorithms
- Rough Set and Fuzzy Set theory
- ...

Ensemble Methods

- Use a number of *base* classifiers, and make a predication by combining the predications of all the classifiers
- Example
  - Binary classification
  - 3 classifiers, each with error rate 30%
  - Predict by majority vote
  - *Error rate of the ensemble classifier??*

Construct an Ensemble Classifier ...

- By manipulating the training set
  - Use a different subset of the training set to train each classifier
  - E.g. Bagging and Boosting
- By manipulating the input features
  - Use a different subset of the attributes to train each classifier

... Construct an Ensemble Classifier

- By manipulating the class labels
  - E.g. ECOC.
- By manipulating the learning algorithm
  - E.g. use of different kernel functions, introducing randomness in attribute selection in decision tree induction

Why Bagging/Boosting?

- How can we use one training set to train k classifiers?
  - Use the same training set for each classifier??
  - Evenly divide the training set into k subsets??

Bootstrap Sampling

- Uniformly samples the training set D *with replacement*
  - After a record is selected, it is added back to the training set ("replacement")
  - A record may be selected multiple times
- A bootstrap sample D_i
  - $|D_i|=|D|$
  - Contains roughly 63.2% of the original records
    - $1-(1-1/N)^N \rightarrow 1-1/e=0.632$

Bagging

- Use a bootstrap sample for each classifier

Bagging Example

- Record (x,y)
  - x: attribute
  - y: class label
- Base classifier: decision tree with one level x≤k
- Ensemble classifier: 10 classifiers, majority vote

Bagging Example – Dataset

X	Y
0.1	1
0.2	1
0.3	1
0.4	-1
0.5	-1
0.6	-1
0.7	-1
0.8	1
0.9	1
1.0	1

Bagging Example – Bagging

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.8	0.9	1	1	1
y	1	1	1	-1	-1	1	1	1	1	1

x <= 0.65 ==> y = 1
x > 0.65 ==> y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ==> y = 1
x > 0.3 ==> y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ==> y = 1
x > 0.35 ==> y = -1

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ==> y = -1
x > 0.75 ==> y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 ==> y = -1
x > 0.05 ==> y = 1

Figure 5.35. Example of bagging.

Bagging Example – Classification

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1
True Class	1	1	1	-1	-1	-1	-1	1	1	1

Figure 5.36. Example of combining classifiers constructed using the bagging approach.

About Bagging

- Reduces the errors associated with random fluctuations in the training data for *unstable classifiers*, e.g. decision trees, rule-based classifiers, ANN
- May degrade the performance of *stable classifiers*, e.g. Bayesian network, SVM, k-NN

Intuition for Boosting

- Sample with weights
  - hard-to-classify records should be chosen more often
- Combine the prediction of the base classifiers with weights
  - Classifiers with lower error rates get more voting power

Boosting – Training

- For `k` classifiers, do `k` rounds of
  - Assign a weight to each record
  - Sample with replacement according to the weights
  - Train a classifier M_i
  - Calculate `error(`M_i`)`
  - Update the weights of the records
    - Increase the weights of the misclassified records
    - Decrease the weights of the correctly classified records

Boosting – Classification

- For each class, sum up the weights of the classifiers that vote for that class
- The class that gets the highest sum is the predicted class

Boosting Implementation

- How the record weights are updated
- How the classifier weights are calculated

Adaboost

Error rate of classifier M_i: $error(M_i) = \sum w_j \times err(\mathbf{X_j})$

Update the weights of the correctly classified records: $w_j \times \frac{error(M_i)}{1 - error(M_i)}$

Weight of classifier M_i: $\log \frac{1 - error(M_i)}{error(M_i)}$

- Initial w_j=1/|D|
- Classifiers with `error(Mi)>0.5` are dropped
- Normalize the weights of all records after updating the weights of the correctly classified records

Evaluate the Accuracy of a Classifier

- Accuracy measures
  - Accuracy rate and error rate
  - Confusion matrix
  - Precision and Recall (for binary classification)

Example of Accuracy Measures

- Example
  - Two classes C_1 and C_2
  - 100 testing records with 50 C_1 records and 50 C_2 records
  - 20 C_1 records misclassified as C_2, and 10 C_2 records misclassified as C_1
- Accuracy measures
  - *Accuracy and error rates??*
  - *Confusion matrix??*
  - *Precision and Recall??*

Evaluate the Accuracy of a Classifier

- The Holdout Method
  - Given a set of records with known class labels, use half of them for training and the other half for testing (or 2/3 for training and 1/3 for testing)

Problems of the Holdout Method

- More records for training means less for testing, and vice versa
- Distribution of the data in the training/testing set may be different from the original dataset
- Some classifiers are sensitive to random fluctuations in the training data

Random Subsampling

- Repeat the holdout method k times
- Take the average accuracy over the k iterations
- Random subsampling methods
  - Cross-validation
  - Bootstrap

K-fold Cross-validation

- Divide the original dataset into k non-overlapping subsets
- Each iteration uses (k-1) subsets for training, and the remaining subset for testing
- Total errors are the sum of the errors in each iteration

Bootstrap

- Each iteration uses a bootstrap sample to train the classifier, and the remaining records for testing
- Calculate the overall accuracy:

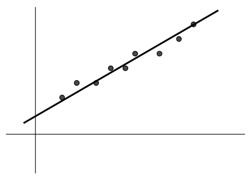
$$\frac{1}{k}\sum_{i=1}^{k}(0.632\times Acc(M_i)_{test\_set} + 0.368\times Acc(M_i)_{all\_records})$$

Predicating Continuous Values

- Regression methods
  - Linear regression
  - Non-linear regression
- Other methods
  - Some classification methods can be adapted to predict continuous values

Linear Regression

- Record (x,y)
  - x: predictor variable
  - y: response variable
- Model
  - $y = w_0 + w_1 x$



Linear Regression Using Least-Squares Method

$$w_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2}$$

$$w_0 = \bar{y} - w_1\bar{x}$$

Multiple Linear Regression

- Record $(x_1, \ldots, x_n, y)$
- Model:

$$y = w_0 + \sum_{i=1}^{n} w_i x_i$$

Summary

- Classification
  - Problem definition and terminology
  - Decision tree and rule-base classification
  - Naive Bayesian classification and BNN
  - kNN
  - SVM and multiclass classification with binary classifier
  - Ensemble methods
  - Evaluation of classification accuracy
- Linear regression