Classification: Basic Concepts, Decision Trees, and Model Evaluation

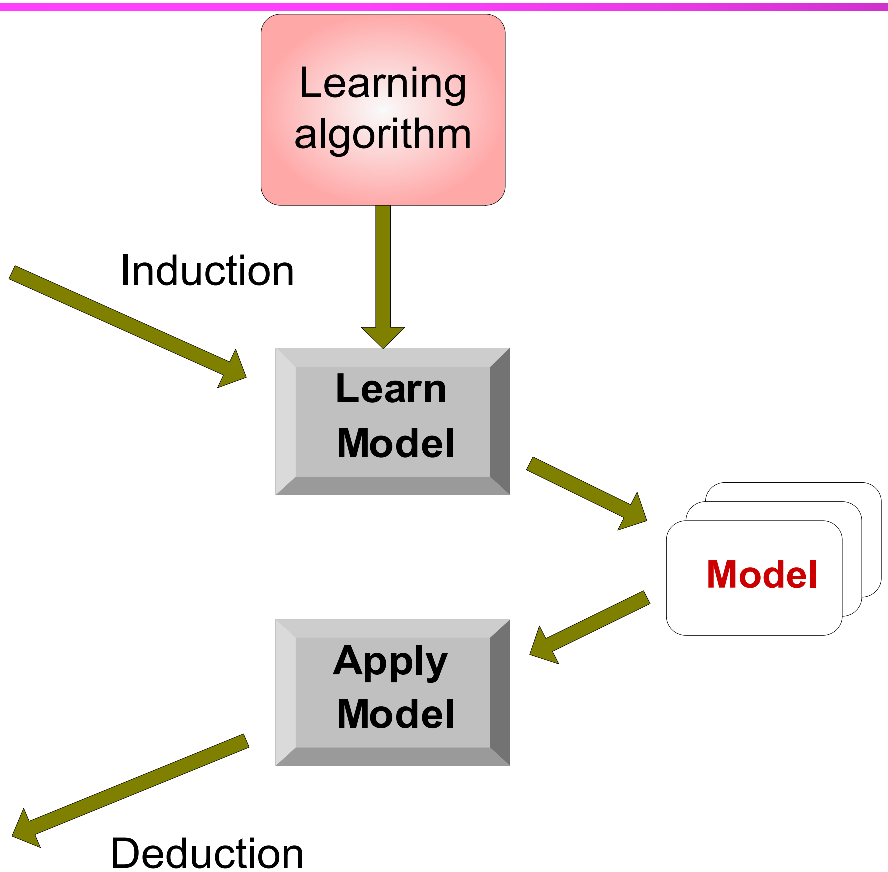
Dr. Meng Qu
Rutgers University

THE STATE UNIVERSITY OF NEW JERSEY
RUTGERS

General Approach for Building Classification Model

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Classification Techniques

- Base Classifiers
  - Decision Tree based Methods
  - Rule-based Methods
  - Nearest-neighbor
  - Neural Networks
  - Naïve Bayes and Bayesian Belief Networks
  - Support Vector Machines
- Ensemble Classifiers
  - Boosting, Bagging, Random Forests

Classification: Alternative Techniques

Bayesian Classifiers

Bayes Classifier

- A probabilistic framework for solving classification problems
- 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 C causes A 50% of the time
  - Prior probability of any patient having C is 1/50,000
  - Prior probability of any patient having A is 1/20
- If a patient has A, what's the probability he/she has C?

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

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

$$\boxed{P(C|A_1A_2\ldots A_n) = \frac{P(A_1A_2\ldots A_n|C)P(C)}{P(A_1A_2\ldots A_n)}}$$

  - Choose value of C that maximizes
    $P(C \mid A_1, A_2, \ldots, A_n)$
  - Equivalent to choosing value of C that maximizes
    $P(A_1, A_2, \ldots, A_n|C)\ P(C)$
- 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:
  - $P(A_1, A_2, \ldots, A_n | C_j) = P(A_1| C_j)\, P(A_2| C_j) \ldots P(A_n| C_j)$
  - Can estimate $P(A_i| C_j)$ for all A_i and C_j.
  - New point is classified to C_j if $P(C_j) \prod P(A_i| C_j)$ is maximal.

Training dataset

Class:
C1:buys_computer= 'yes'
C2:buys_computer= 'no'

Data sample
X =(age<=30,
Income=medium,
Student=yes
Credit_rating=
Fair)

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Naïve Bayesian Classifier: Example

- Compute P(X|Ci) for each class

P(age="<30" | buys_computer="yes") = 2/9=0.222
P(age="<30" | buys_computer="no") = 3/5 =0.6
P(income="medium" | buys_computer="yes")= 4/9 =0.444
P(income="medium" | buys_computer="no") = 2/5 = 0.4
P(student="yes" | buys_computer="yes)= 6/9 =0.667
P(student="yes" | buys_computer="no")= 1/5=0.2
P(credit_rating="fair" | buys_computer="yes")=6/9=0.667
P(credit_rating="fair" | buys_computer="no")=2/5=0.4

X=(age<=30 , income =medium, student=yes, credit_rating=fair)

P(X|Ci) : P(X|buys_computer="yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer="no") = 0.6 x 0.4 x 0.2 x 0.4 =0.019

P(X|Ci)*P(Ci) : P(X|buys_computer="yes") * P(buys_computer="yes") = 0.028
P(X|buys_computer="no") * P(buys_computer="no") = 0.007

X belongs to class "buys_computer=yes"

Naïve Bayes Classifier

- If one of the conditional probability is zero, then the entire expression becomes zero.
- Probability estimation:

$$\text{Original}: P(A_i \mid C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace}: P(A_i \mid C) = \frac{N_{ic} + 1}{N_c + c}$$

$$\text{m - estimate}: P(A_i \mid C) = \frac{N_{ic} + mp}{N_c + m}$$

c: number of classes

p: prior probability

m: parameter (equivalent sample size)

Classification: Model Overfitting and Classifier Evaluation

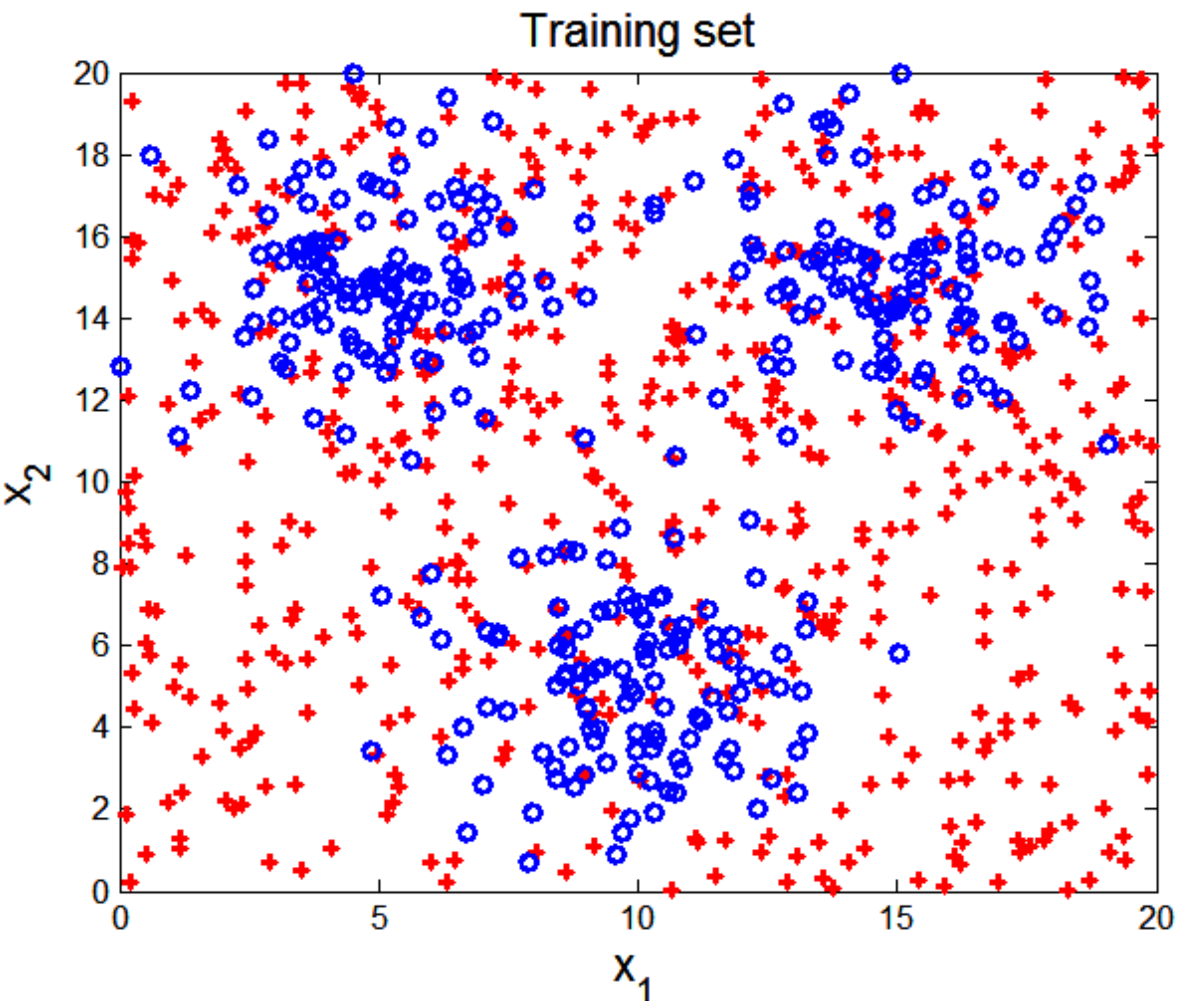
Dr. Meng Qu
Rutgers University

THE STATE UNIVERSITY OF NEW JERSEY
RUTGERS

Classification Errors

- Training errors (apparent errors)
  - Errors committed on the training set
- Test errors
  - Errors committed on the test set
- Generalization errors
  - Expected error of a model over random selection of records from same distribution

Example Data Set



Two class problem:

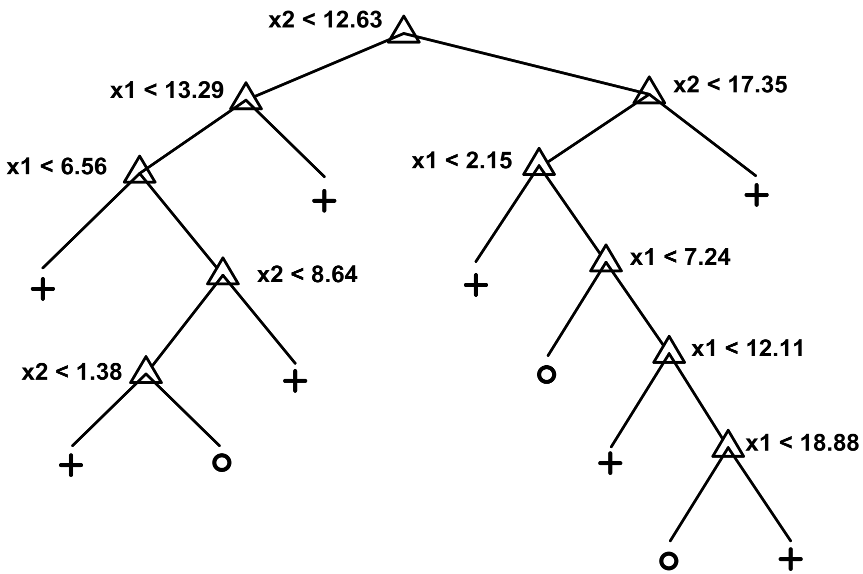
+, o

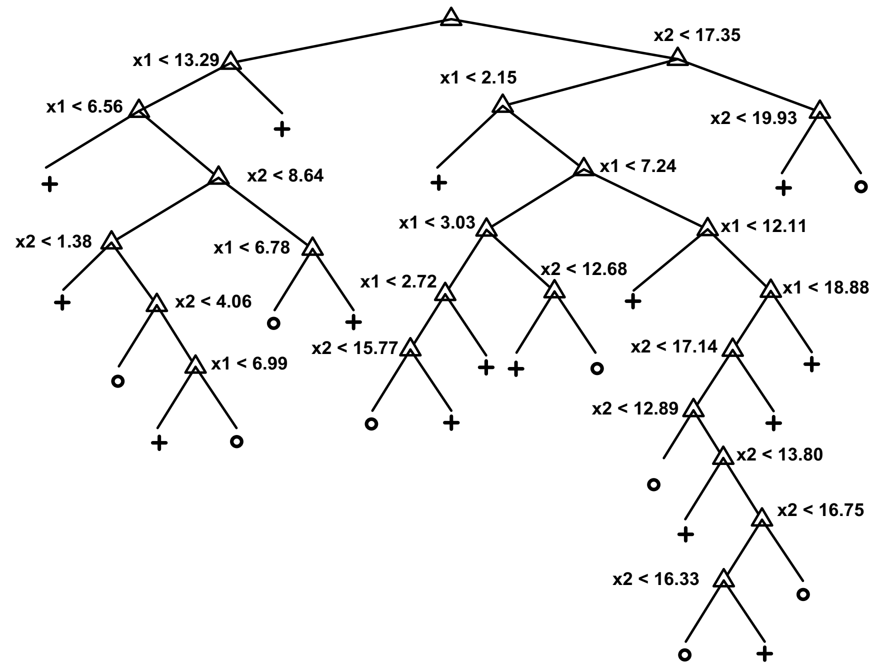
3000 data points (30% for training, 70% for testing)

Data set for + class is generated from a uniform distribution

Data set for o class is generated from a mixture of 3 gaussian distributions, centered at (5,15), (10,5), and (15,15)

Decision Trees



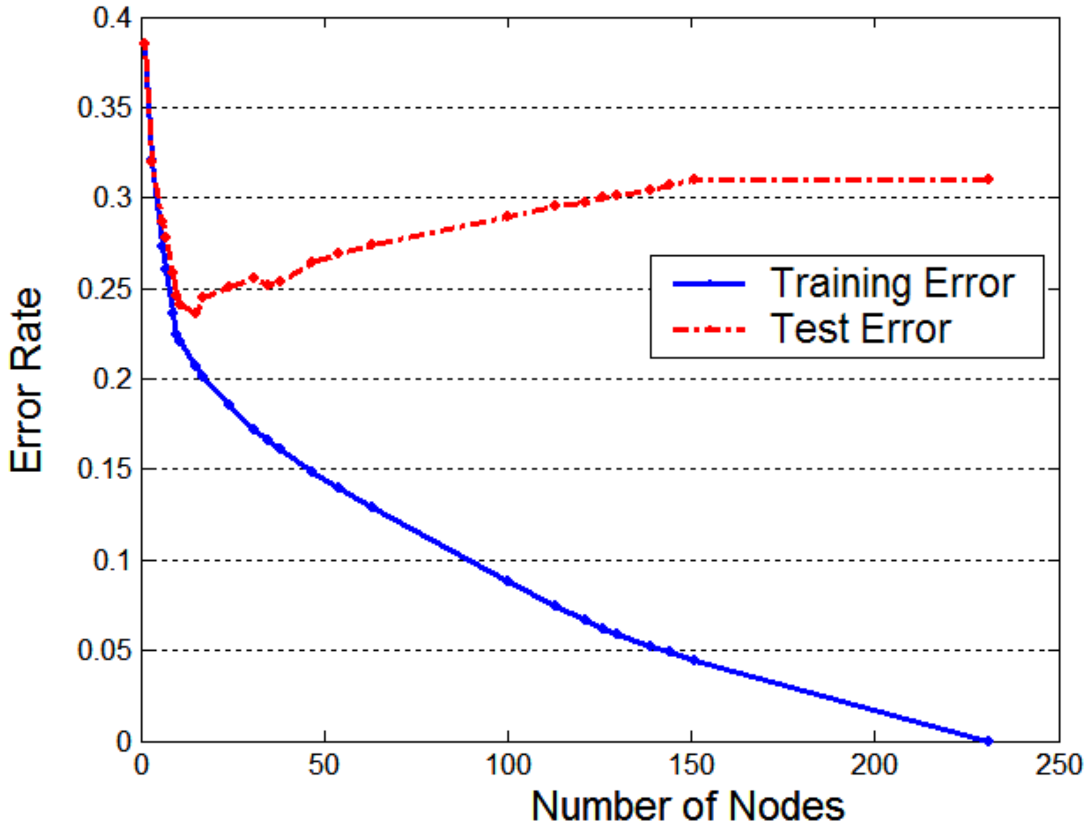


Decision Tree with 11 leaf nodes

Decision Tree with 24 leaf nodes

Which tree is better?

Model Overfitting



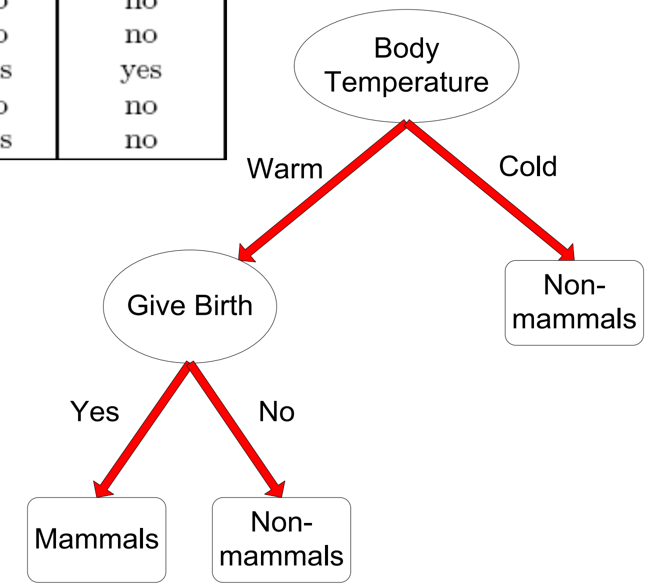
Underfitting: when model is too simple, both training and test errors are large

Overfitting: when model is too complex, training error is small but test error is large

Mammal Classification Problem

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Mammal
human	warm-blooded	hair	yes	no	no	yes	no	yes
python	cold-blooded	scales	no	no	no	no	yes	no
salmon	cold-blooded	scales	no	yes	no	no	no	no
whale	warm-blooded	hair	yes	yes	no	no	no	yes
frog	cold-blooded	none	no	semi	no	yes	yes	no
komodo dragon	cold-blooded	scales	no	no	no	yes	no	no
bat	warm-blooded	hair	yes	no	yes	yes	yes	yes
pigeon	warm-blooded	feathers	no	no	yes	yes	no	no
cat	warm-blooded	fur	yes	no	no	yes	no	yes
leopard shark	cold-blooded	scales	yes	yes	no	no	no	no
turtle	cold-blooded	scales	no	semi	no	yes	no	no
penguin	warm-blooded	feathers	no	semi	no	yes	no	no
porcupine	warm-blooded	quills	yes	no	no	yes	yes	yes
eel	cold-blooded	scales	no	yes	no	no	no	no
salamander	cold-blooded	none	no	semi	no	yes	yes	no

Training Set



Decision Tree Model

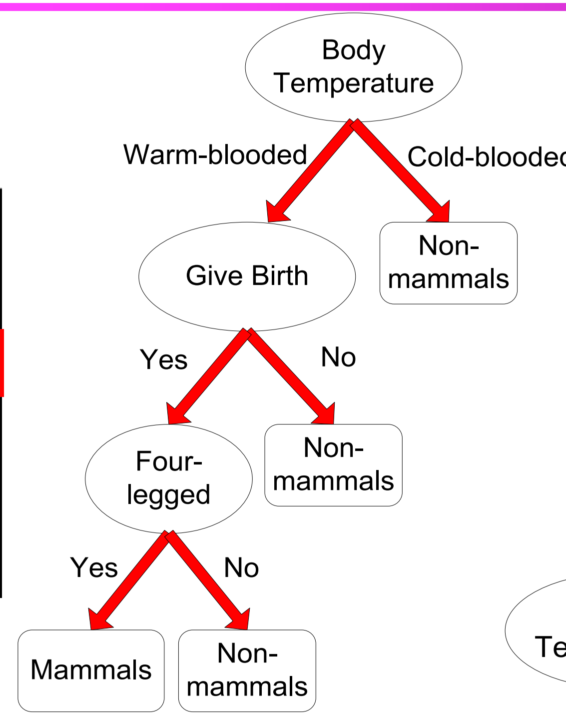
training error = 0%

Effect of Noise

Example: Mammal Classification problem

Training Set:

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
porcupine	warm-blooded	yes	yes	yes	yes
cat	warm-blooded	yes	yes	no	yes
bat	warm-blooded	yes	no	yes	no*
whale	warm-blooded	yes	no	no	no*
salamander	cold-blooded	no	yes	yes	no
komodo dragon	cold-blooded	no	yes	no	no
python	cold-blooded	no	no	yes	no
salmon	cold-blooded	no	no	no	no
eagle	warm-blooded	no	no	no	no
guppy	cold-blooded	yes	no	no	no



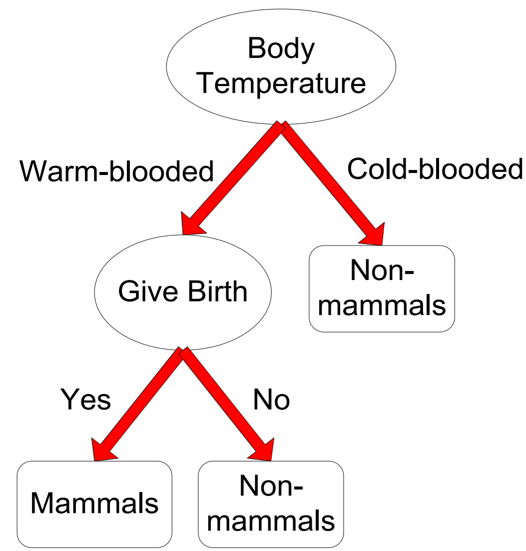
Model M1:

train err = 0%,

test err = 30%

Test Set:

Name	Body Temperature	Gives Birth	Four-legged	Hibernates	Class Label
human	warm-blooded	yes	no	no	yes
pigeon	warm-blooded	no	no	no	no
elephant	warm-blooded	yes	yes	no	yes
leopard shark	cold-blooded	yes	no	no	no
turtle	cold-blooded	no	yes	no	no
penguin	cold-blooded	no	no	no	no
eel	cold-blooded	no	no	no	no
dolphin	warm-blooded	yes	no	no	yes
spiny anteater	warm-blooded	no	yes	yes	yes
gila monster	cold-blooded	no	yes	yes	no



Model M2:

train err = 20%,

test err = 10%

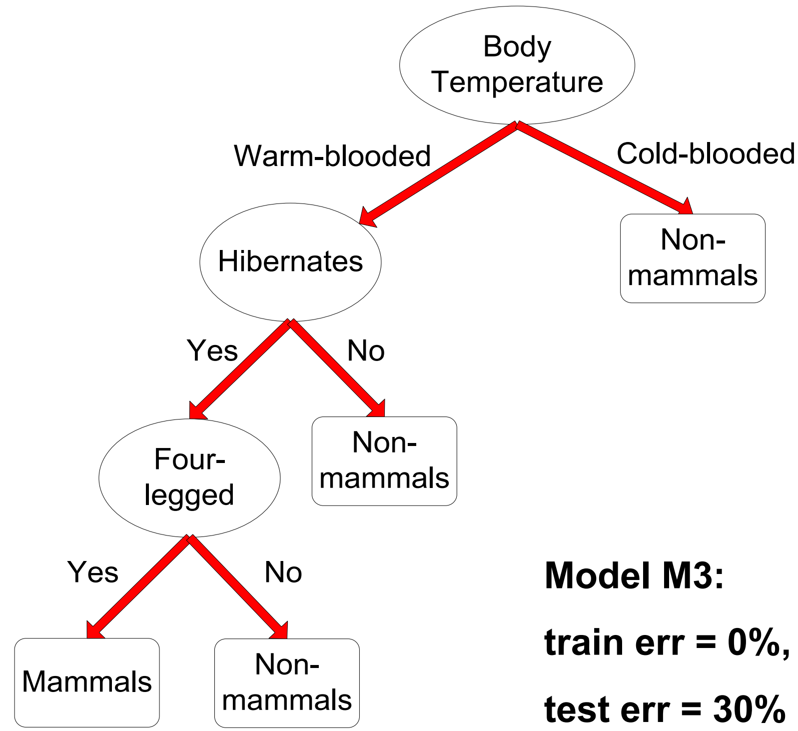
Lack of Representative Samples

Training Set:

Name	Body Temperature	Four-legged	Hibernates	Class Label
salamander	cold-blooded	yes	yes	no
guppy	cold-blooded	no	no	no
eagle	warm-blooded	no	no	no
poorwill	warm-blooded	no	yes	no
platypus	warm-blooded	yes	yes	yes

Test Set:

Name	Body Temperature	Four-legged	Hibernates	Class Label
human	warm-blooded	no	no	yes
pigeon	warm-blooded	no	no	no
elephant	warm-blooded	yes	no	yes
leopard shark	cold-blooded	no	no	no
turtle	cold-blooded	yes	no	no
penguin	cold-blooded	no	no	no
eel	cold-blooded	no	no	no
dolphin	warm-blooded	no	no	yes
spiny anteater	warm-blooded	yes	yes	yes
gila monster	cold-blooded	yes	yes	no



Model M3:

train err = 0%,

test err = 30%

Lack of training records at the leaf nodes for making reliable classification

Effect of Multiple Comparison Procedure

- Consider the task of predicting whether stock market will rise/fall in the next 10 trading days

- Random guessing:

  $P(correct) = 0.5$

- Make 10 random guesses in a row:

$$P(\# correct \geq 8) = \frac{\binom{10}{8} + \binom{10}{9} + \binom{10}{10}}{2^{10}} = 0.0547$$

Day 1	Up
Day 2	Down
Day 3	Down
Day 4	Up
Day 5	Down
Day 6	Down
Day 7	Up
Day 8	Up
Day 9	Up
Day 10	Down

Effect of Multiple Comparison Procedure

- Approach:
  - Get 50 analysts
  - Each analyst makes 10 random guesses
  - Choose the analyst that makes the most number of correct predictions

- Probability that at least one analyst makes at least 8 correct predictions

$$P(\#\, correct \geq 8) = 1 - (1 - 0.0547)^{50} = 0.9399$$

Effect of Multiple Comparison Procedure

- Many algorithms employ the following greedy strategy:
  - Initial model: M
  - Alternative model: $M' = M \cup \gamma$,
    where γ is a component to be added to the model (e.g., a test condition of a decision tree)
  - Keep M' if improvement, $\Delta(M,M') > \alpha$

- Often times, γ is chosen from a set of alternative components, $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_k\}$

- If many alternatives are available, one may inadvertently add irrelevant components to the model, resulting in model overfitting

Notes on Overfitting

- Overfitting results in decision trees that are <u>more complex</u> than necessary

- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records

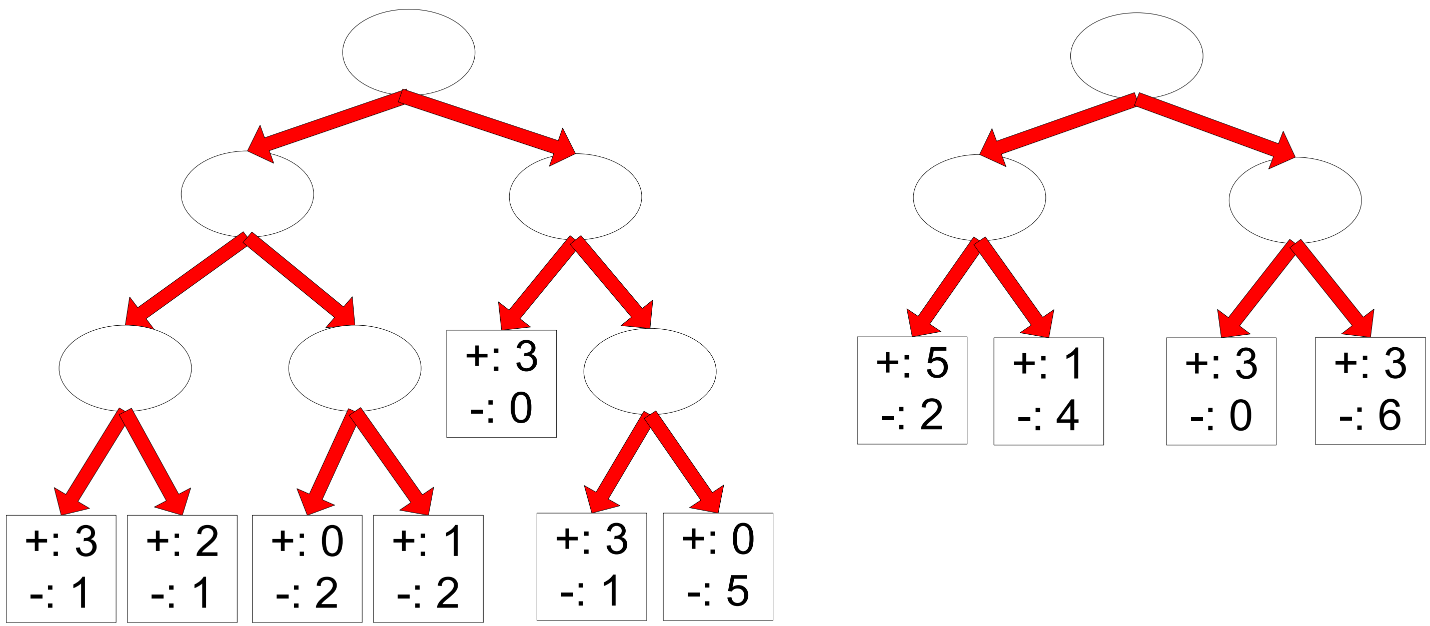
- Need new ways for estimating generalization errors

Estimating Generalization Errors

- Resubstitution Estimate
- Incorporating Model Complexity
- Estimating Statistical Bounds
- Use Validation Set

Resubstitution Estimate

- Using training error as an optimistic estimate of generalization error



$e(T_L) = 4/24$

$e(T_R) = 6/24$

Decision Tree, T_L

Decision Tree, T_R

Incorporating Model Complexity

- Rationale: Occam's Razor
  - Given two models of similar generalization errors, one should prefer the simpler model over the more complex model
  - A complex model has a greater chance of being fitted accidentally by errors in data
  - Therefore, one should include model complexity when evaluating a model

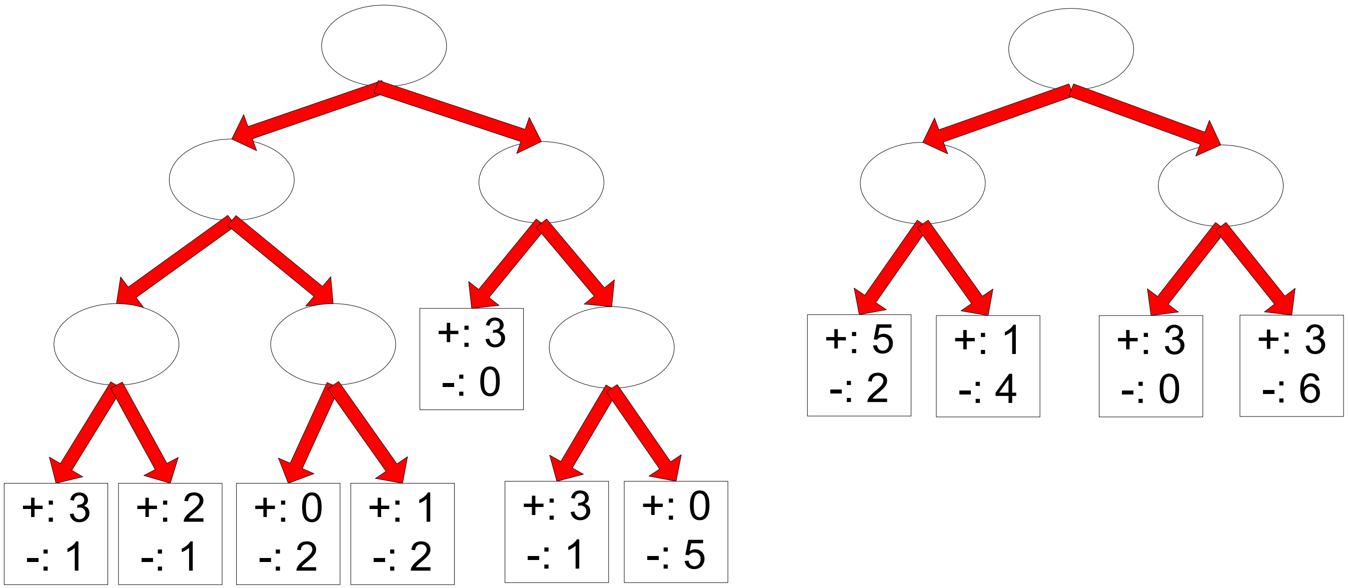
Pessimistic Estimate

- Given a decision tree node t
  - $n(t)$: number of training records classified by t
  - $e(t)$: misclassification error of node t
  - Training error of tree T:

$$e'(T) = \frac{\sum_i \left[e(t_i) + \Omega(t_i) \right]}{\sum_i n(t_i)} = \frac{e(T) + \Omega(T)}{N}$$

    - Ω: is the cost of adding a node
    - N: total number of training records

Pessimistic Estimate



Decision Tree, T_L

Decision Tree, T_R

$e(T_L) = 4/24$

$e(T_R) = 6/24$

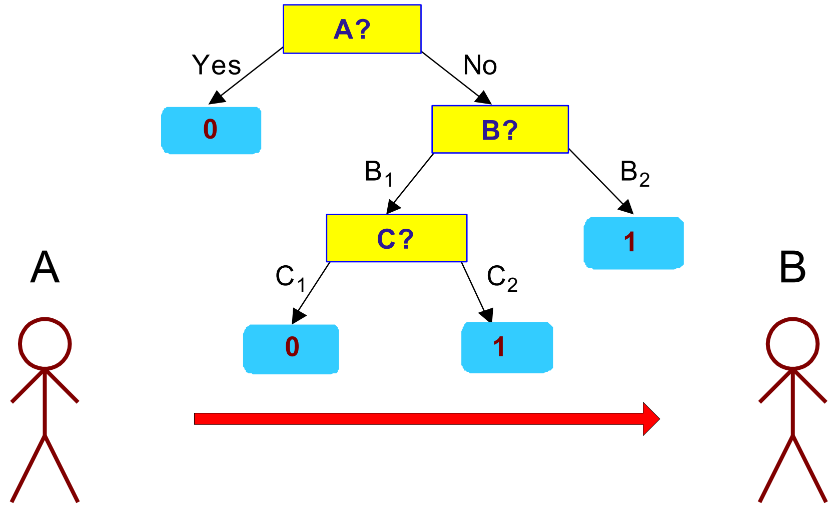
$\Omega = 1$

$e'(T_L) = (4 + 7 \times 1)/24 = 0.458$

$e'(T_R) = (6 + 4 \times 1)/24 = 0.417$

Minimum Description Length (MDL)

X	y
X_1	1
X_2	0
X_3	0
X_4	1
…	…
X_n	1



X	y
X_1	?
X_2	?
X_3	?
X_4	?
…	…
X_n	?

- Cost(Model,Data) = Cost(Data|Model) + Cost(Model)
  - Cost is the number of bits needed for encoding.
  - Search for the least costly model.
- Cost(Data|Model) encodes the misclassification errors.
- Cost(Model) uses node encoding (number of children) plus splitting condition encoding.

Using Validation Set

- Divide training data into two parts:
  - Training set:
    - use for model building
  - Validation set:
    - use for estimating generalization error
    - Note: validation set is not the same as test set

- Drawback:
  - Less data available for training

Handling Overfitting in Decision Tree

- Pre-Pruning (Early Stopping Rule)
  - Stop the algorithm before it becomes a fully-grown tree
  - Typical stopping conditions for a node:
    - Stop if all instances belong to the same class
    - Stop if all the attribute values are the same
  - More restrictive conditions:
    - Stop if number of instances is less than some user-specified threshold
    - Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
    - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain).
    - Stop if estimated generalization error falls below certain threshold

Handling Overfitting in Decision Tree

- Post-pruning
  - Grow decision tree to its entirety
  - Subtree replacement
    - Trim the nodes of the decision tree in a bottom-up fashion
    - If generalization error improves after trimming, replace sub-tree by a leaf node
    - Class label of leaf node is determined from majority class of instances in the sub-tree
  - Subtree raising
    - Replace subtree with most frequently used branch

Example of Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

Training Error (Before splitting) = 10/30

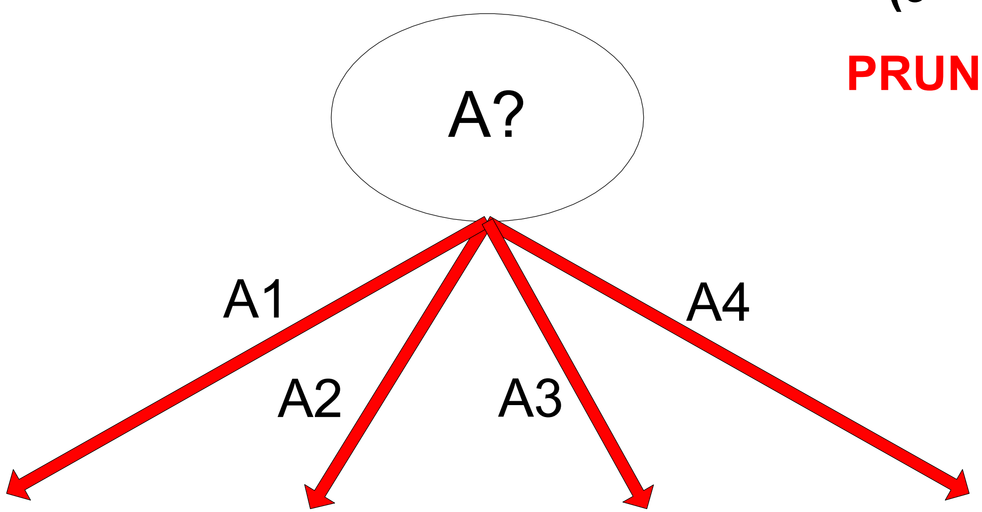
Pessimistic error = (10 + 0.5)/30 = 10.5/30

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

= (9 + 4 $\times$ 0.5)/30 = 11/30

PRUNE!



Class = Yes	8
Class = No	4

Class = Yes	3
Class = No	4

Class = Yes	4
Class = No	1

Class = Yes	5
Class = No	1

Examples of Post-pruning

Decision Tree:

```
depth = 1 :
|   breadth > 7 : class 1
|   breadth <= 7 :
|   |   breadth <= 3 :
|   |   |   ImagePages > 0.375 : class 0
|   |   |   ImagePages <= 0.375 :
|   |   |   |   totalPages <= 6 : class 1
|   |   |   |   totalPages > 6 :
|   |   |   |   |   breadth <= 1 : class 1
|   |   |   |   |   breadth > 1 : class 0
|   |   width > 3 :
|   |   |   MultiIP = 0:
|   |   |   |   ImagePages <= 0.1333 : class 1
|   |   |   |   ImagePages > 0.1333 :
|   |   |   |   |   breadth <= 6 : class 0
|   |   |   |   |   breadth > 6 : class 1
|   |   |   MultiIP = 1:
|   |   |   |   TotalTime <= 361 : class 0
|   |   |   |   TotalTime > 361 : class 1
depth > 1 :
|   MultiAgent = 0:
|   |   depth > 2 : class 0
|   |   depth <= 2 :
|   |   |   MultiIP = 1: class 0
|   |   |   MultiIP = 0:
|   |   |   |   breadth <= 6 : class 0
|   |   |   |   breadth > 6 :
|   |   |   |   |   RepeatedAccess <= 0.0322 : class 0
|   |   |   |   |   RepeatedAccess > 0.0322 : class 1
|   MultiAgent = 1:
|   |   totalPages <= 81 : class 0
|   |   totalPages > 81 : class 1
```

Subtree Raising

Subtree Replacement

Simplified Decision Tree:

```
depth = 1 :
|   ImagePages <= 0.1333 : class 1
|   ImagePages > 0.1333 :
|   |   breadth <= 6 : class 0
|   |   breadth > 6 : class 1
depth > 1 :
|   MultiAgent = 0: class 0
|   MultiAgent = 1:
|   |   totalPages <= 81 : class 0
|   |   totalPages > 81 : class 1
```

Evaluating Performance of Classifier

- Model Selection
  - Performed during model building
  - Purpose is to ensure that model is not overly complex (to avoid overfitting)
  - Need to estimate generalization error

- Model Evaluation
  - Performed after model has been constructed
  - Purpose is to estimate performance of classifier on previously unseen data (e.g., test set)

Methods for Classifier Evaluation

- Holdout
  - Reserve k% for training and (100-k)% 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
- Bootstrap
  - Sampling with replacement
  - .632 bootstrap:

$$acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}\left(0.632 \times acc_i + 0.368 \times acc_s\right)$$

Methods for Comparing Classifiers

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances
- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?

Confidence Interval for Accuracy

- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
    - Coin toss – head/tail
    - Prediction – correct/wrong
  - Collection of Bernoulli trials has a Binomial distribution:
    - $x \sim \text{Bin}(N, p)$ x: number of correct predictions

- Estimate number of events
  - Given N and p, find $P(x=k)$ or $E(x)$
  - Example: Toss a fair coin 50 times, how many heads would turn up?

    Expected number of heads $= N \times p = 50 \times 0.5 = 25$

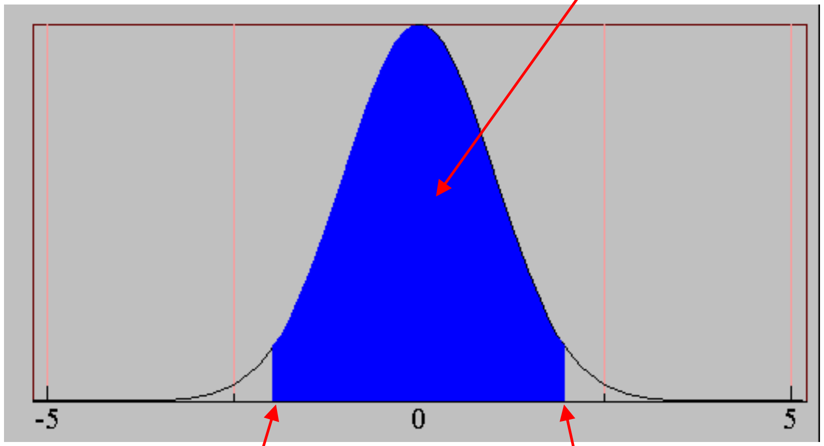
Confidence Interval for Accuracy

- Estimate parameter of distribution
  - Given x (# of correct predictions)
    or equivalently, acc=x/N, and
    N (# of test instances),
  - Find upper and lower bounds of p (true accuracy of model)

Confidence Interval for Accuracy

- For large test sets (N > 30),
  - acc has a normal distribution with mean p and variance p(1-p)/N

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



- Confidence Interval for p:

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

prove

Confidence Interval for Accuracy

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

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

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

Comparing Performance of 2 Models

- Given two models, say M1 and M2, which is better?
  - M1 is tested on D1 (size=n1), found error rate = e_1
  - M2 is tested on D2 (size=n2), found error rate = e_2
  - Assume D1 and D2 are independent
  - If n1 and n2 are sufficiently large, then

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

  - Approximate: $\hat{\sigma}_i^{\,2} = \frac{e_i(1-e_i)}{n_i}$

Comparing Performance of 2 Models

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

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

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

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

An Illustrative Example

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

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

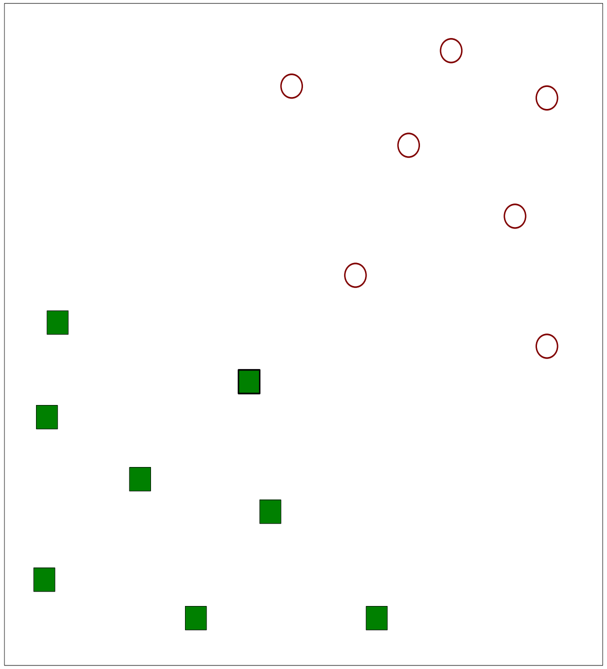
- At 95% confidence level, $Z_{\alpha/2}$=1.96

$$d_t = 0.100 \pm 1.96 \times \sqrt{0.0043} = 0.100 \pm 0.128$$

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

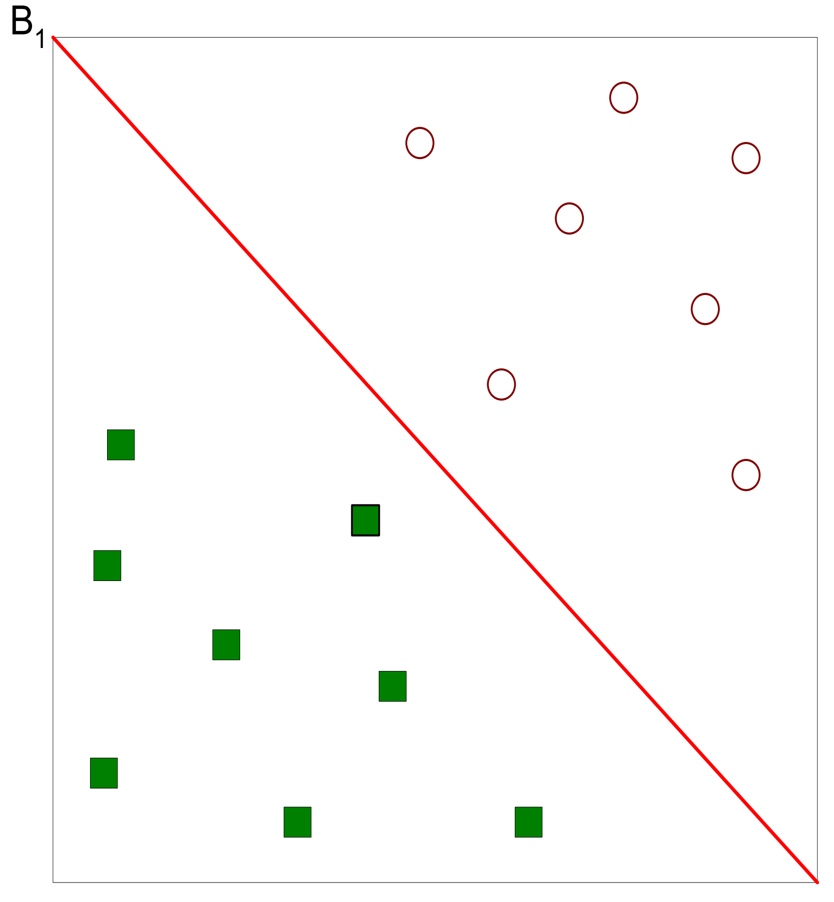
Support Vector Machines (SVMs)

SVMs are a rare example of a methodology where geometric intuition, elegant mathematics, theoretical guarantees, and practical use meet.



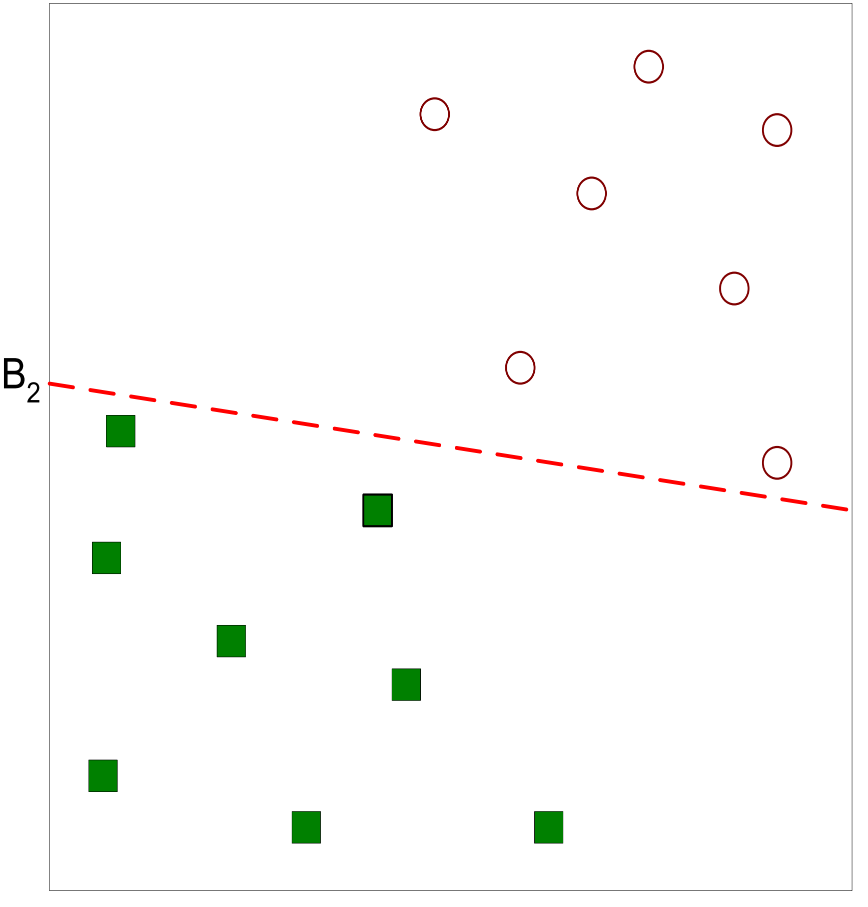
- Find a linear hyperplane (decision boundary) that separates the data

Support Vector Machines



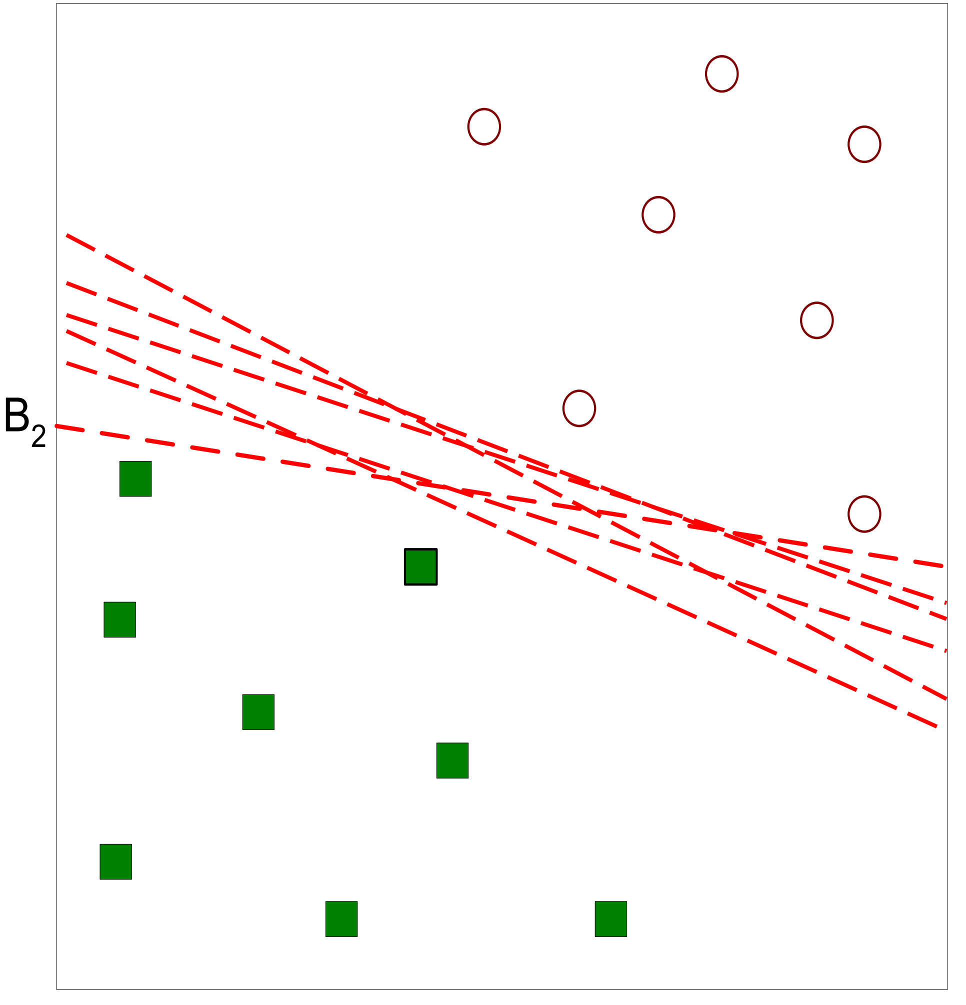
- One Possible Solution

Support Vector Machines



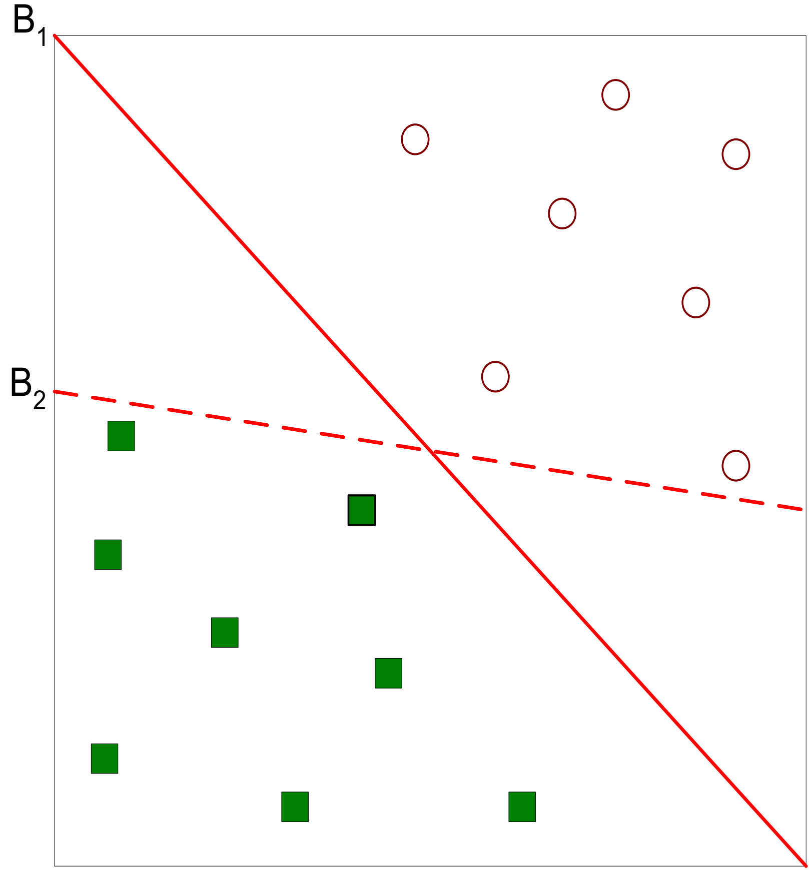
- Another possible solution

Support Vector Machines



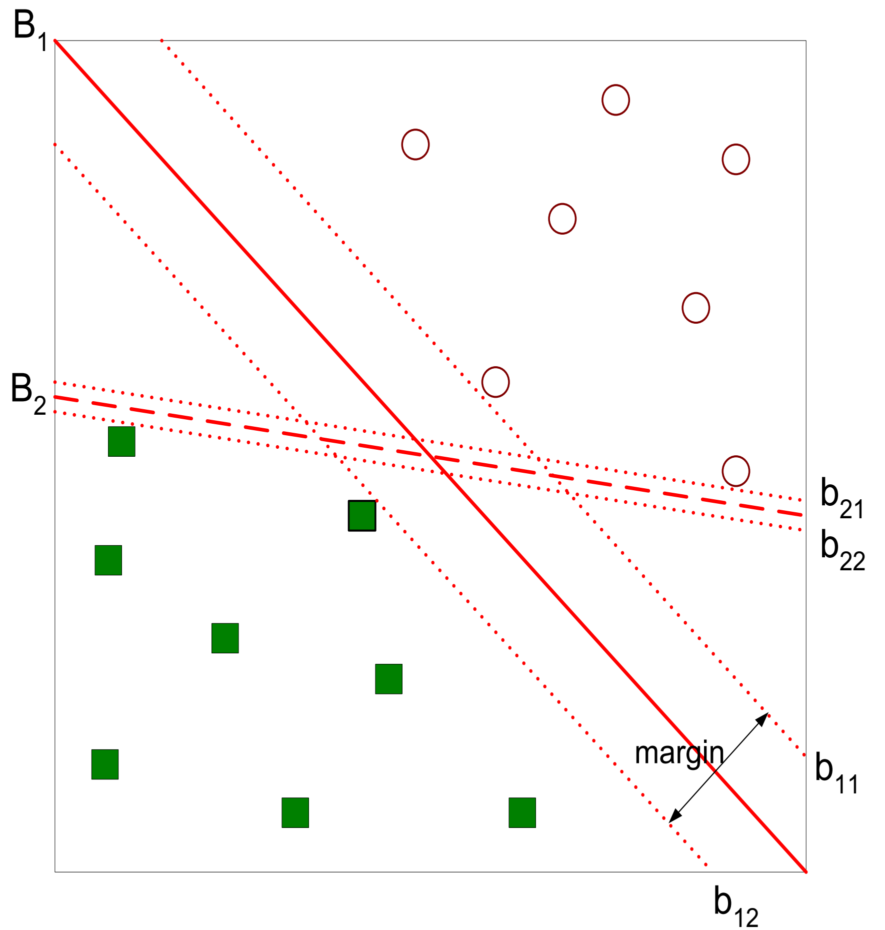
- Other possible solutions

Support Vector Machines



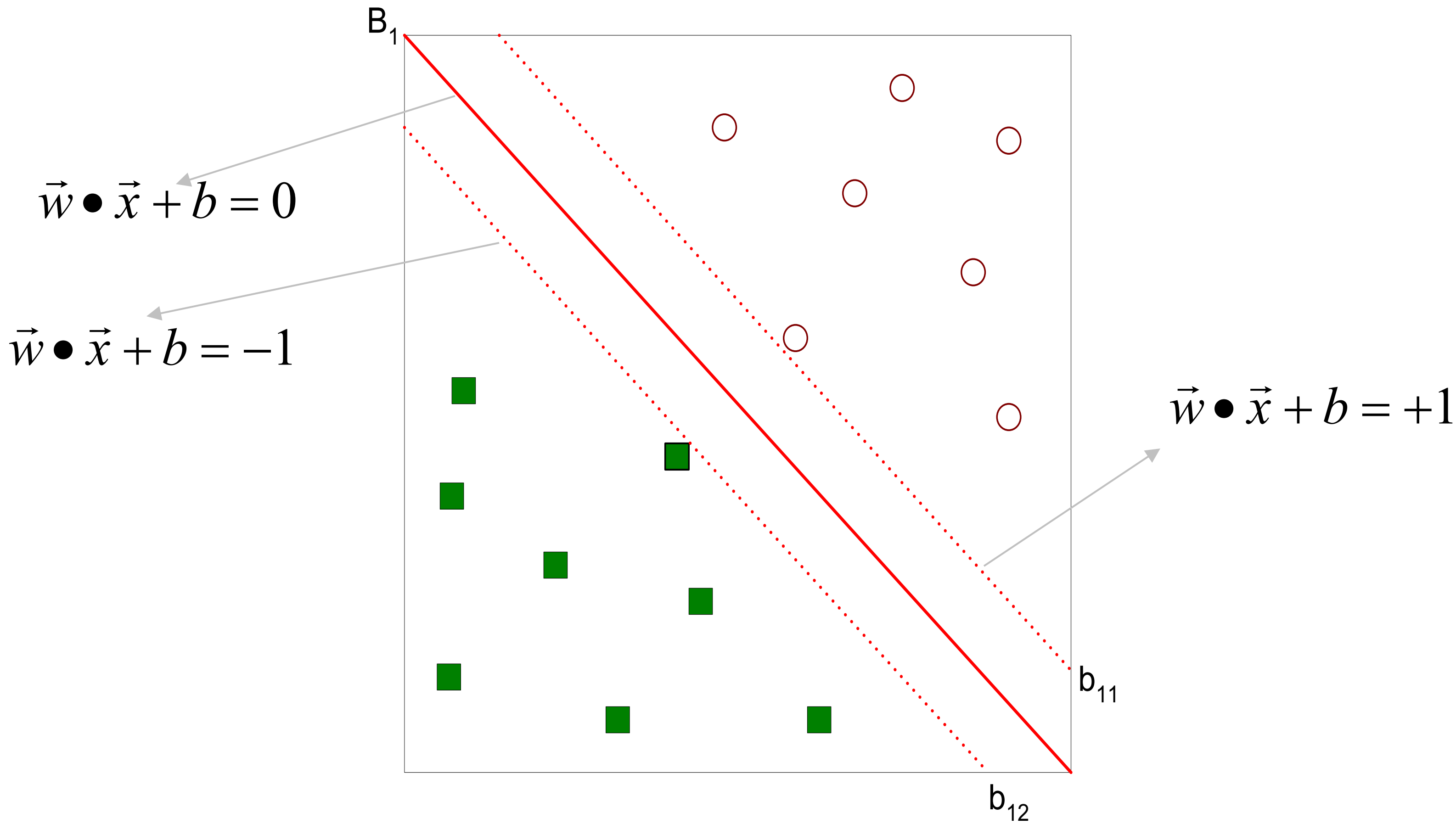
- Which one is better? B1 or B2?
- How do you define better?

Support Vector Machines



- Find hyperplane maximizes the margin => B1 is better than B2

Support Vector Machines



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}} + \mathrm{b} \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Support Vector Machines

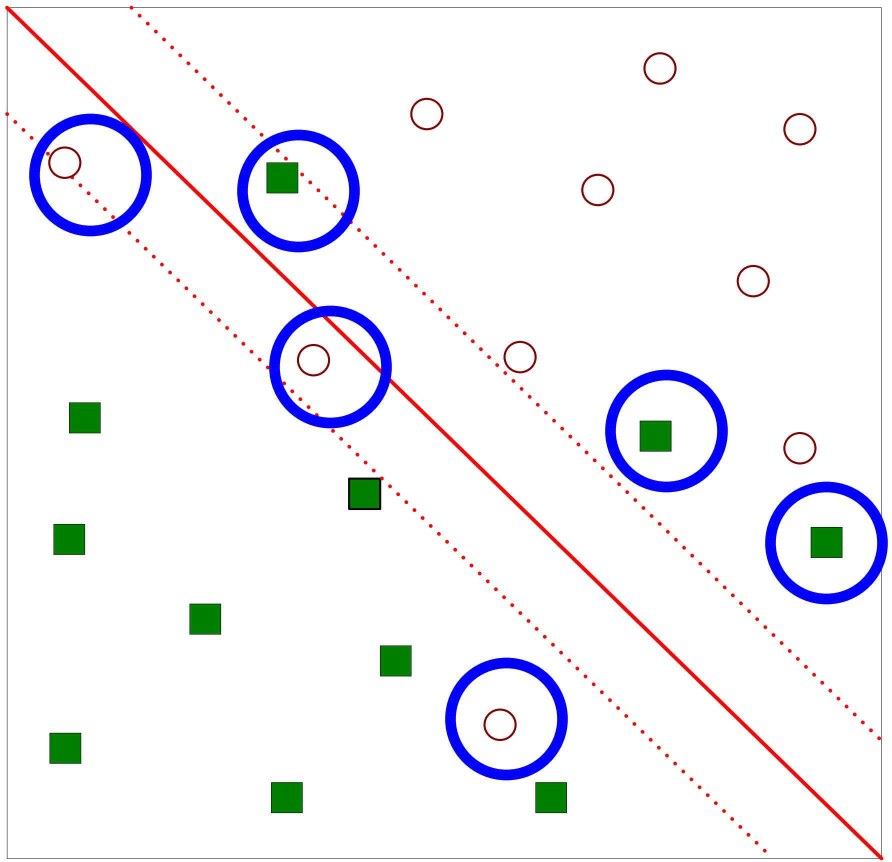
- We want to maximize: $\text{Margin} = \dfrac{2}{\|\vec{w}\|^2}$
  - Which is equivalent to minimizing: $L(w) = \dfrac{\|\vec{w}\|^2}{2}$
  - But subjected to the following constraints:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 \end{cases}$$

    - This is a constrained optimization problem
      - Numerical approaches to solve it

Support Vector Machines

- What if the problem is not linearly separable?



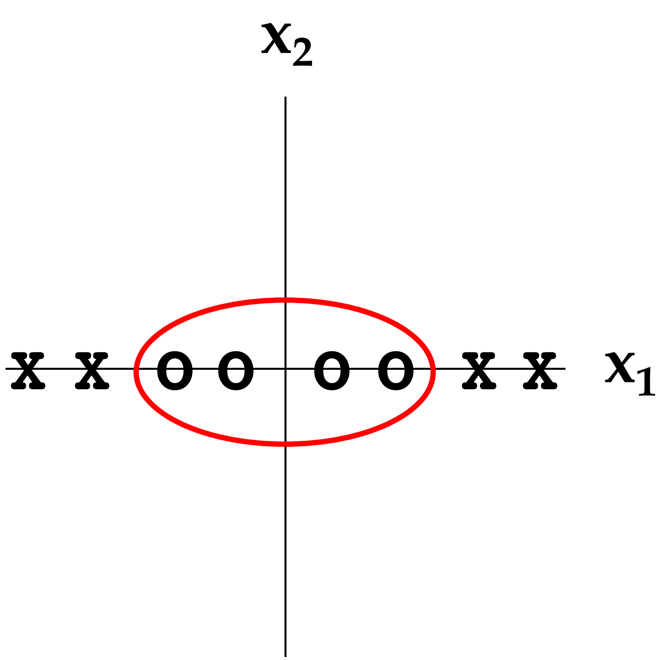
Support Vector Machines

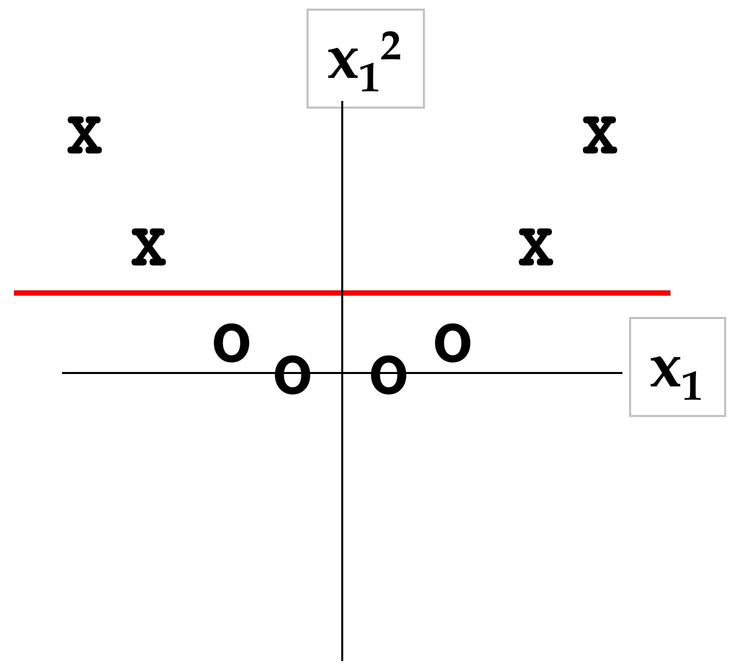
- What if the problem is not linearly separable?
  - Introduce slack variables
    - Need to minimize: $L(w) = \frac{\|\vec{w}\|^2}{2} + C\left(\sum_{i=1}^{N} \xi_i^k\right)$
    - Subject to:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \geq 1 - \xi_{\mathrm{i}} \\ -1 & \text{if } \vec{\mathrm{w}} \bullet \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} \leq -1 + \xi_{\mathrm{i}} \end{cases}$$

Nonlinear Support Vector Machines

- What if decision boundary is not linear?





Mapping into a New Feature Space

$$\Phi : x \rightarrow X = \Phi(x)$$

$$\Phi(x_1,x_2) = (x_1,x_2,x_1{}^2,x_2{}^2,x_1x_2)$$

- Rather than run SVM on x_i, run it on $\Phi(x_i)$
- Find non-linear separator in input space
- What if $\Phi(x_i)$ is really big?
- Use kernels!

Examples of Kernel Functions

- Polynomial kernel with degree *d*

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^d$$

- Radial basis function kernel with width σ

$$K(\mathbf{x}, \mathbf{y}) = \exp(-||\mathbf{x} - \mathbf{y}||^2/(2\sigma^2))$$

- Sigmoid $K(\mathbf{x}, \mathbf{y}) = \tanh(\kappa\mathbf{x}^T\mathbf{y} + \theta)$
  - It does not satisfy the Mercer condition on all κ and θ

Characteristics of SVMs

- Perform best on the average or outperform other techniques across many important applications
- The results are stable, reproducible, and largely independent of the specific optimization algorithm
- A convex optimization problem
  - Lead to the global optimum
- The parameter selection problem
  - The type of kernels (including its parameters)
  - The attributes to be included.
- The results are hard to interpret
- Computational challenge
  - Typically quadratic and multi-scan of the data