



Introduction to Data Mining

Chapter 4 Classification – Alternative Techniques

by Michael Hahsler

Based in Slides by by Tan,
Steinbach, Karpatne, Kumar



R Code Examples

- Available R Code examples are indicated on slides by the R logo



- The Examples are available at [https://mhahsler.github.io/Introduction to Data Mining R Examples/](https://mhahsler.github.io/Introduction%20to%20Data%20Mining%20R%20Examples/)





Topics

- **Rule-Based Classifier**
- Nearest Neighbor Classifier
- Naive Bayes Classifier
- Artificial Neural Networks
- Support Vector Machines
- Ensemble Methods

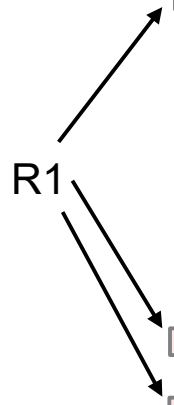
Rule-Based Classifier

- Classify records by using a collection of “if...then...” rules
- Rule: $(Condition) \rightarrow y$
where
 - Condition is a conjunctions of attributes (calles LHS, antecedent or condition)
 - y is the class label (called RHS or consequent)
- Examples of classification rules:
 - $(Blood\ Type = Warm) \wedge (Lay\ Eggs = Yes) \rightarrow Birds$
 - $(Taxable\ Income < 50K) \wedge (Refund = Yes) \rightarrow Evade = No$

Rule-based Classifier (Example)

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

R1



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

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

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

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

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

Application of Rule-Based Classifier

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

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

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

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

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

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

| Name | Blood Type | Give Birth | Can Fly | Live in Water | Class |
|--------------|------------|------------|---------|---------------|-------|
| hawk | warm | no | yes | no | ? |
| grizzly bear | warm | yes | no | no | ? |

The rule R1 covers: *hawk* \rightarrow *Bird*

The rule R3 covers: *grizzly bear* \rightarrow *Mammal*

Ordered Rule Set vs. Voting

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

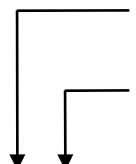
R1: (Give Birth = no) \wedge (Can Fly = yes) \rightarrow Birds

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

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

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

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



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

- Alternative: (weighted) voting by all matching rules.

Rule Coverage and Accuracy

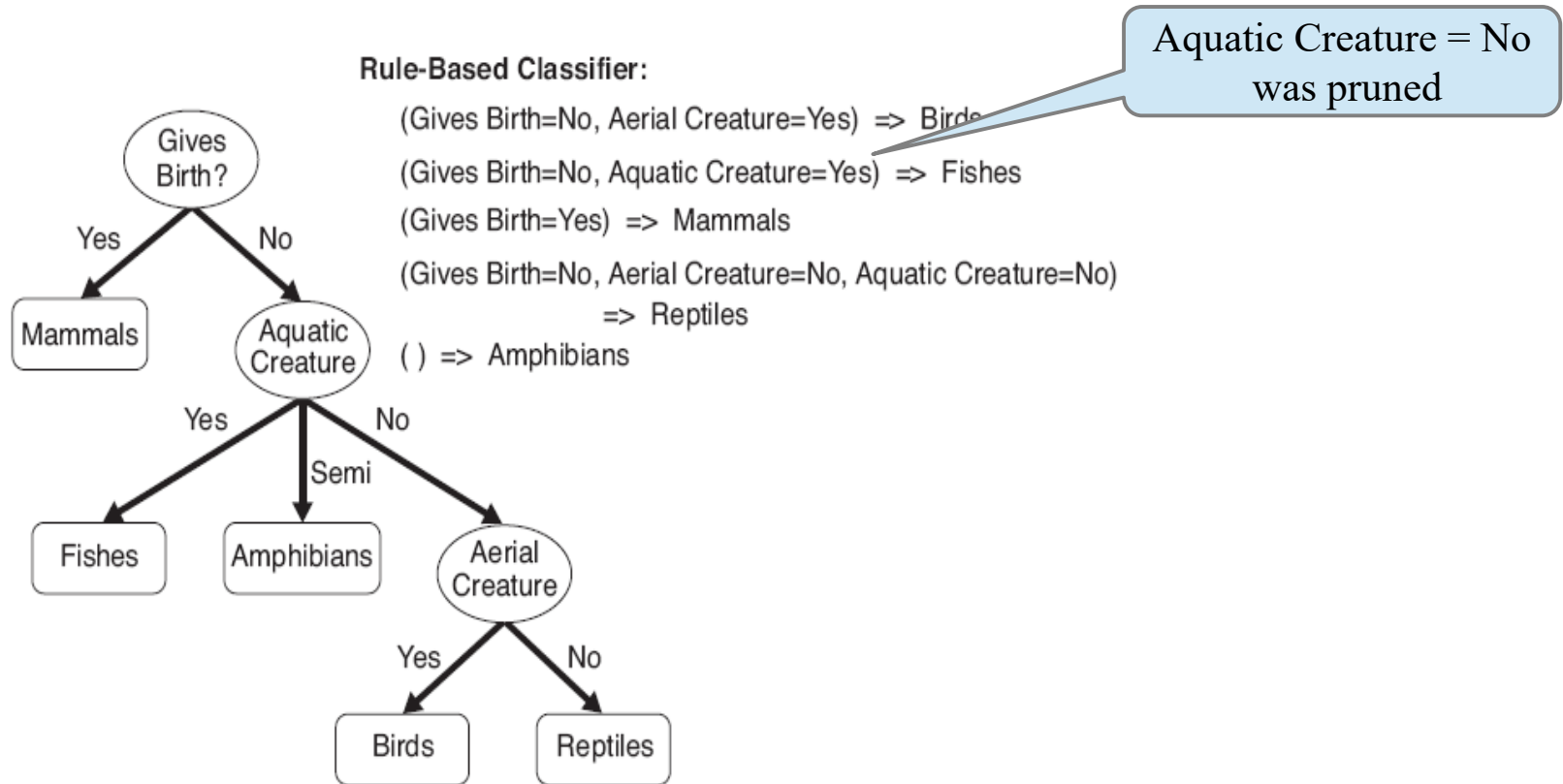
- Coverage of a rule:
 - Fraction of records that satisfy the antecedent of a rule
- Accuracy of a rule:
 - Fraction of records that satisfy both the antecedent and consequent of a rule

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

(Status=Single) → No

Coverage = 40%, Accuracy = 50%

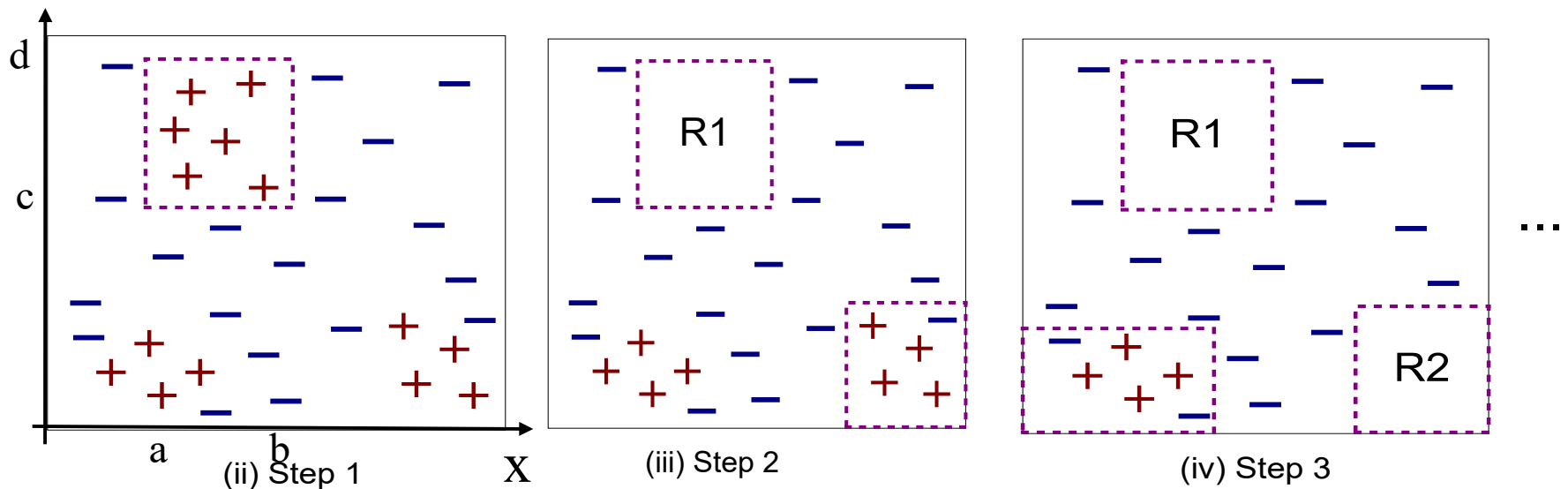
Rules From Decision Trees



- Rules are mutually exclusive and exhaustive (cover all training cases)
- Rule set contains as much information as the tree
- Rules can be simplified (similar to pruning of the tree)
- Example: C4.5rules

Direct Methods of Rule Generation

- Extract rules directly from the data
- Sequential Covering (Example: try to cover class +)



$$R1: a > x > b \wedge c > y > d \rightarrow \text{class } +$$

Advantages of Rule-Based Classifiers

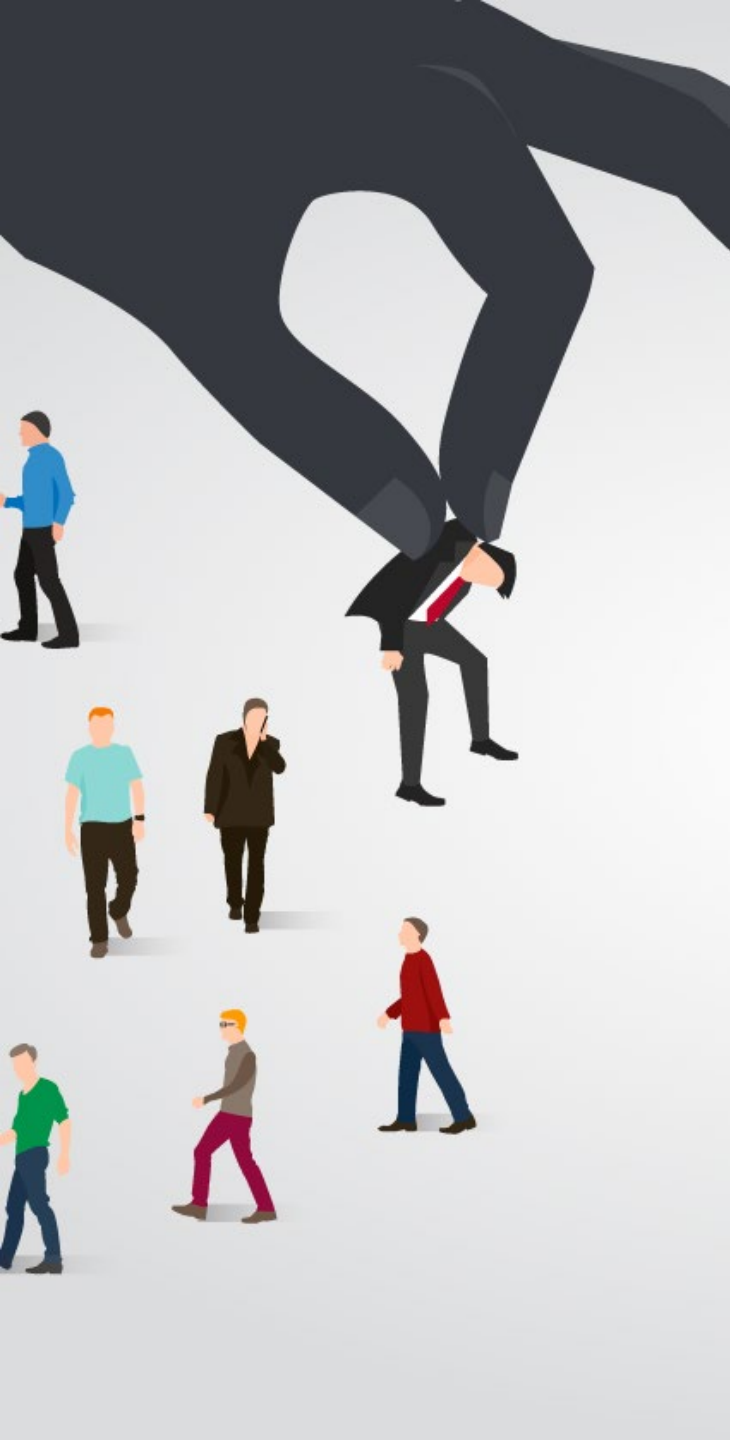
As highly expressive as decision trees

Easy to interpret

Easy to generate

Can classify new instances rapidly

Performance comparable to decision trees



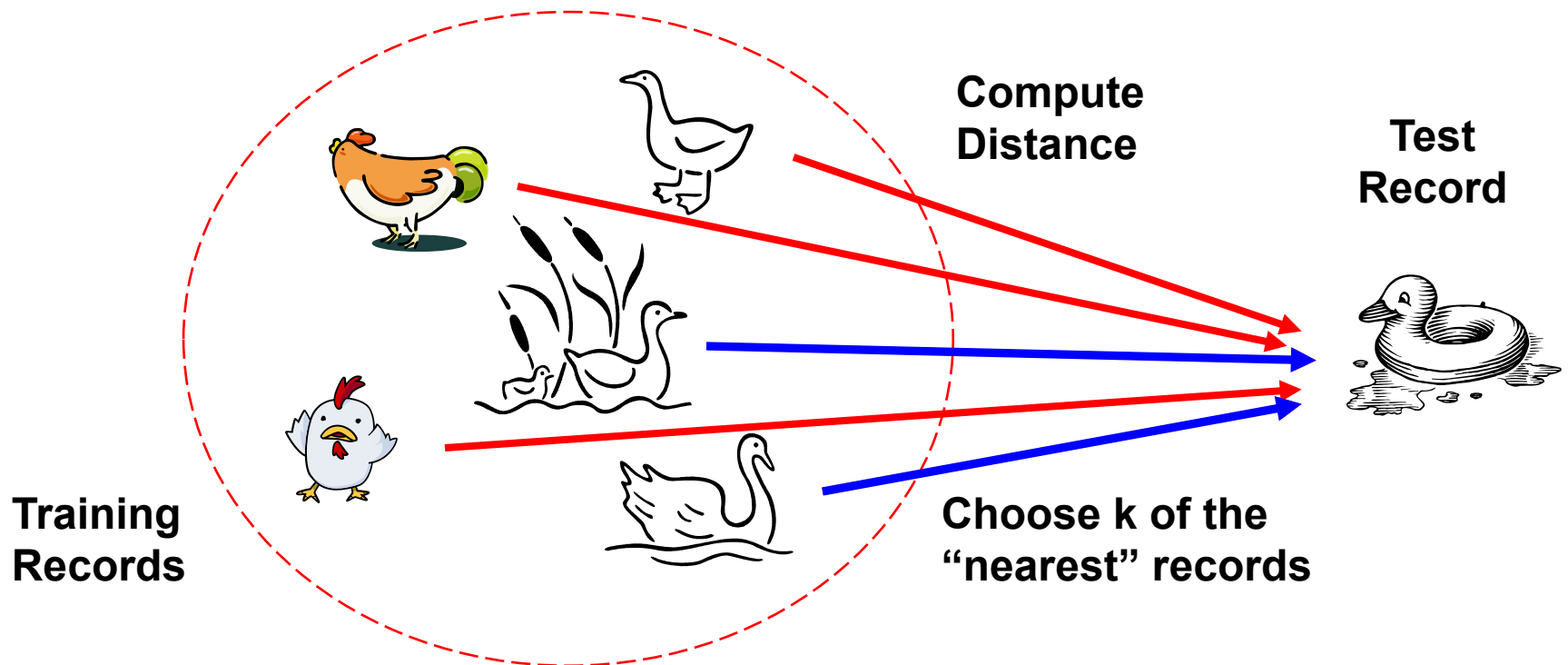
Topics

- Rule-Based Classifier
- **Nearest Neighbor Classifier**
- Naive Bayes Classifier
- Artificial Neural Networks
- Support Vector Machines
- Ensemble Methods

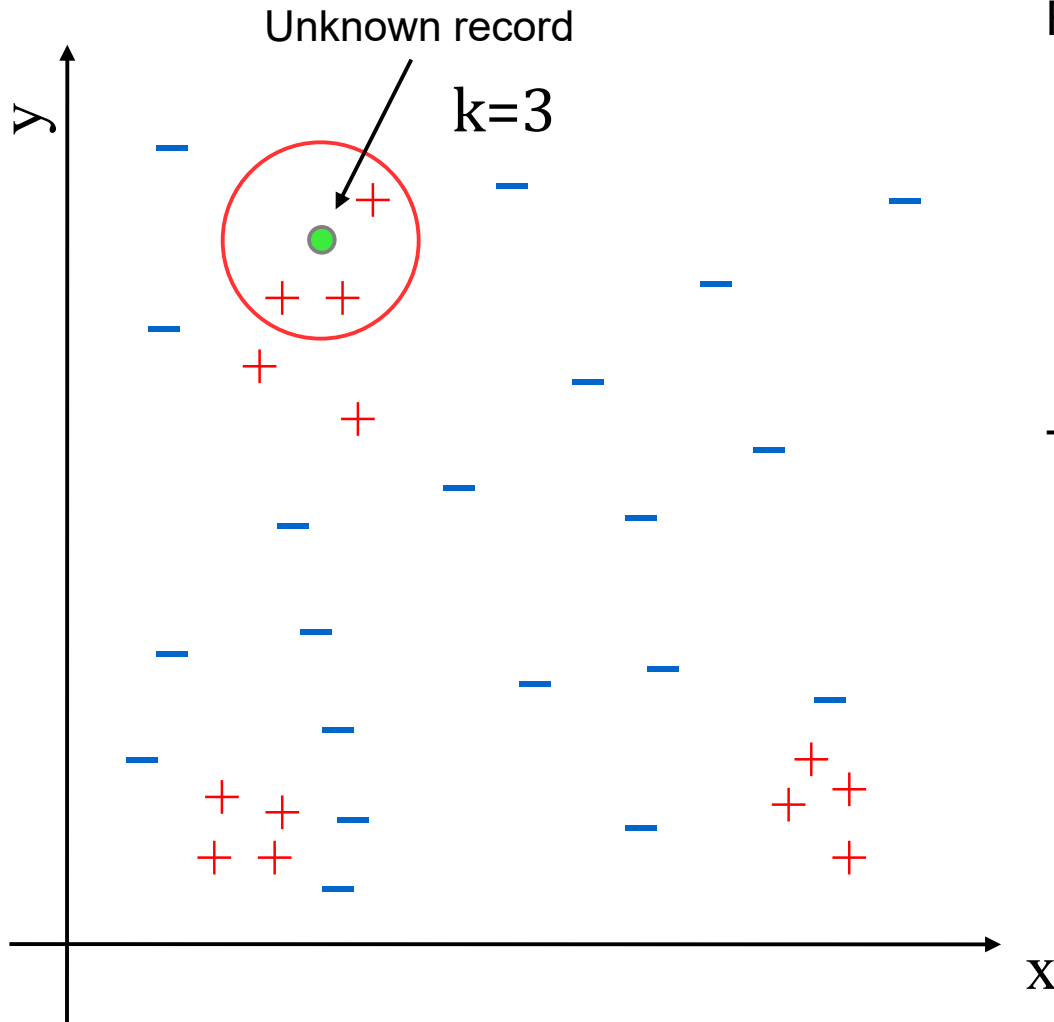
Nearest Neighbor Classifiers

- Basic idea:

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



Nearest-Neighbor Classifiers



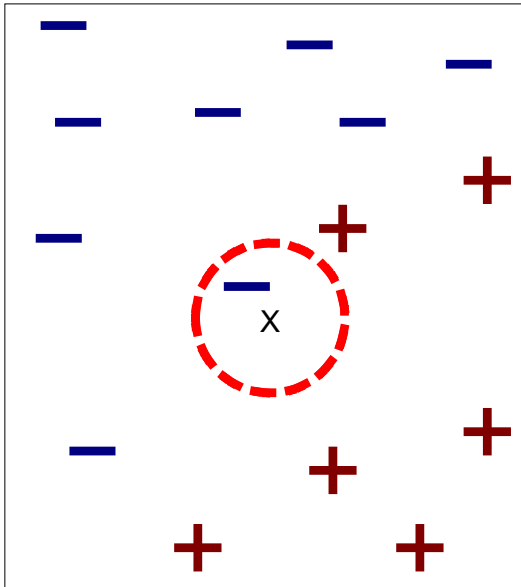
Requires three things

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

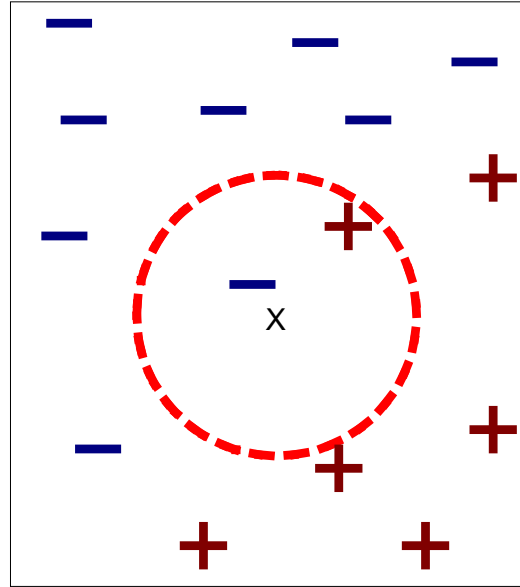
To classify an unknown record:

- 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)

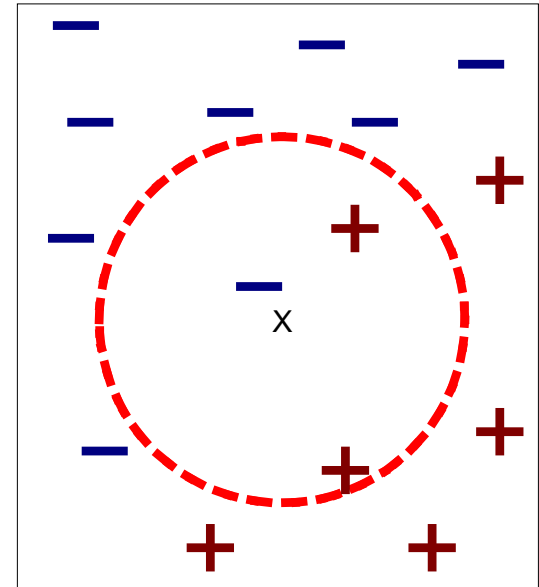
Definition of Nearest Neighbor



(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

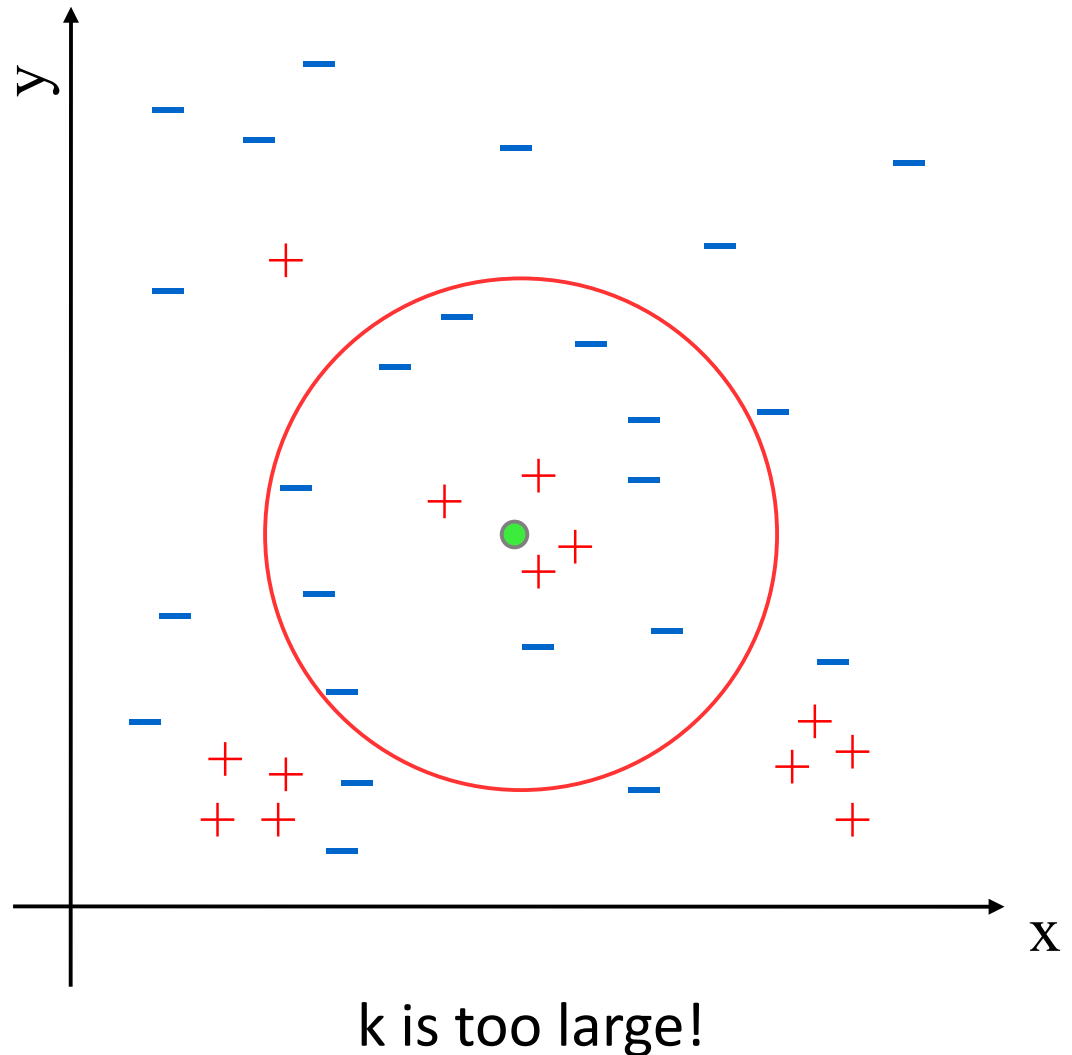
$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_i (p_i - q_i)^2}$$

- 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 (e.g., weight factor $w = 1/d^2$)

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



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
- Income will dominate Euclidean distance!

Solution: scaling/standardization (Z-Score)

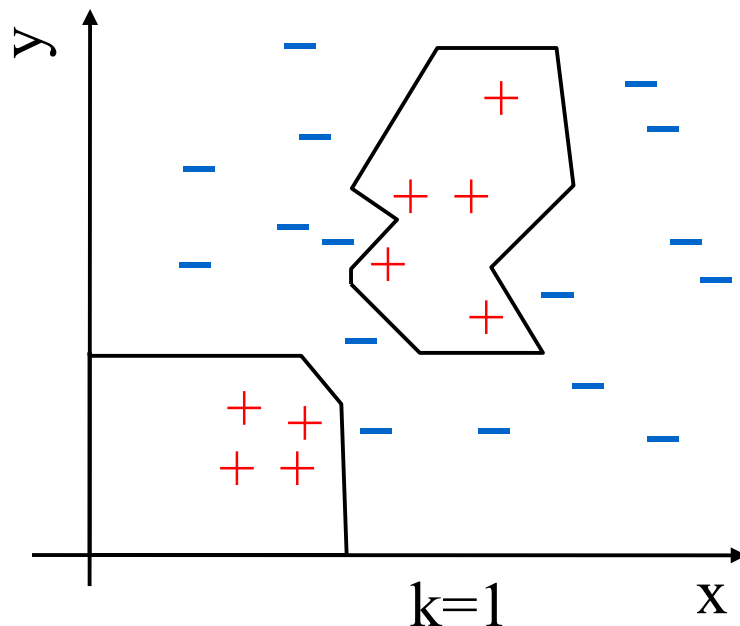
$$z = \frac{x - \bar{x}}{sd(x)}$$

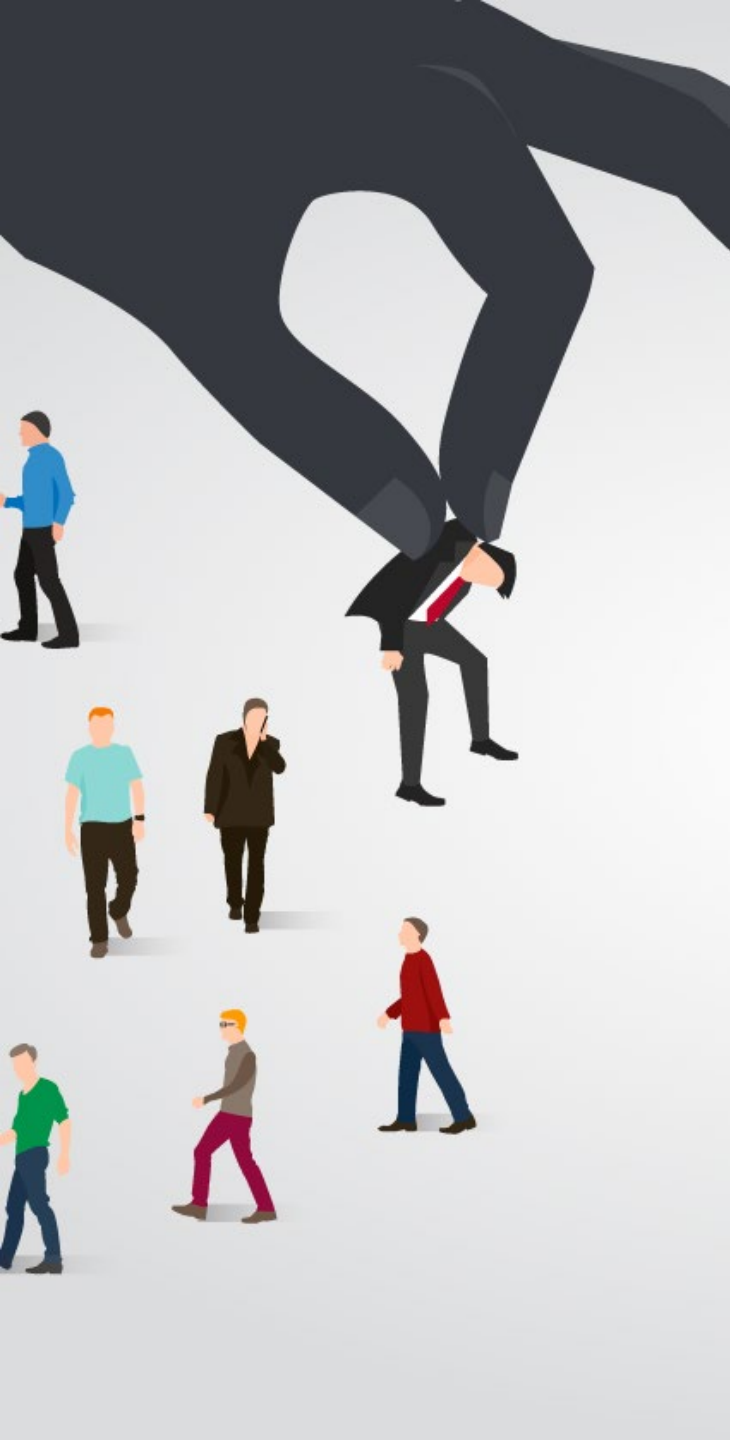
Nearest neighbor Classification...

k-NN classifiers are lazy learners

- It does not build models explicitly (unlike eager learners such as decision trees)
- Needs to store all the training data
- Classifying unknown records are relatively expensive (find the k-nearest neighbors)

Advantage: Can create non-linear decision boundaries





Topics

- Rule-Based Classifier
- Nearest Neighbor Classifier
- **Naive Bayes Classifier**
- Artificial Neural Networks
- Support Vector Machines
- Ensemble Methods

Bayes' Rule

- The product rule gives us two ways to factor a joint distribution:

$$P(A, B) = P(A|B)P(B) = P(B|A)P(A)$$

- Therefore,

Posterior Prob.

Prior Prob.

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

- Why is this useful?
 - Can get diagnostic probability $P(\text{cavity} \mid \text{toothache})$ from causal probability $P(\text{toothache} \mid \text{cavity})$
 - We can update our beliefs based on evidence.
 - Important tool for probabilistic inference .

Example of Bayes Theorem

- A doctor knows that meningitis causes stiff neck 50% of the time → $P(S|M)=.5$
- Prior probability of any patient having meningitis is $P(M) = 1/50,000=0.00002$
- Prior probability of any patient having stiff neck is $P(S) = 1/20=0.05$
- If a patient has stiff neck, what's the probability he/she has meningitis?

$$P(M | S) = \frac{P(S | M) P(M)}{P(S)} = \frac{.5 \times 0.00002}{0.05} = 0.0002$$

Increases the probability by x10!

Bayesian Classifiers

- Consider each attribute and class label as random variables
- Given a record with attributes (A_1, A_2, \dots, A_n)
 - Goal is to predict class C
 - Specifically, we want to find the value of C that maximizes

$$P(C | A_1, A_2, \dots, A_n)$$

Bayesian Classifiers

- compute the posterior probability $P(C | A_1, A_2, \dots, A_n)$ for all values of C using the Bayes theorem

$$P(C | A_1, A_2, \dots, A_n) = \frac{P(A_1, A_2, \dots, A_n | C) P(C)}{P(A_1, A_2, \dots, A_n)}$$

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

this is a constant!

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

- How to estimate $P(A_1, A_2, \dots, A_n | C)$?

Naïve Bayes Classifier

Assume independence among attributes A when class is given:

$$P(A_1, A_2, \dots, A_n | C) = P(A_1 | C) P(A_2 | C) \dots P(A_n | C) = \prod_i P(A_i | C)$$

We can estimate $P(A_i | C_j)$ for all A_i and C_j .

New point is classified to C_j such that:

$$\max_j (P(C_j) \prod_i P(A_i | C_j))$$

How to Estimate Probabilities from Data?

- Class: $P(C_j) = N_{C_j} / N$

e.g., $P(C=\text{No}) = 7/10$,
 $P(C=\text{Yes}) = 3/10$

- For discrete attributes:

$$P(A_i | C_j) = \frac{|A_{ij}|}{N_{C_j}}$$

where $|A_{ij}|$ is number of instances having attribute A_i and belongs to class C_j

e.g.

$P(\text{Status}=\text{Married} \mid C=\text{No}) = 4/7$

$P(\text{Refund}=\text{Yes} \mid C=\text{Yes})=0$

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

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 $P(A_i | C_j)$

Example of Naïve Bayes Classifier

Given a Test Record what is the most likely class?

$$X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120K)$$

naive Bayes Classifier:

$$P(\text{Refund}=\text{Yes}|\text{No}) = 3/7$$

$$P(\text{Refund}=\text{No}|\text{No}) = 4/7$$

$$P(\text{Refund}=\text{Yes}|\text{Yes}) = 0$$

$$P(\text{Refund}=\text{No}|\text{Yes}) = 1$$

$$P(\text{Marital Status}=\text{Single}|\text{No}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{No}) = 1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{No}) = 4/7$$

$$P(\text{Marital Status}=\text{Single}|\text{Yes}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{Yes}) = 1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{Yes}) = 0$$

For taxable income:

If class=No: sample mean=110

sample variance=2975

If class=Yes: sample mean=90

sample variance=25

$$\begin{aligned} P(X|\text{Class}=\text{No}) &= P(\text{Refund}=\text{No}|\text{Class}=\text{No}) \\ &\quad * P(\text{Married}|\text{Class}=\text{No}) \\ &\quad * P(\text{Income}=120K|\text{Class}=\text{No}) \\ &= 4/7 * 4/7 * 0.0072 = 0.0024 \end{aligned}$$

$$\begin{aligned} P(X|\text{Class}=\text{Yes}) &= P(\text{Refund}=\text{No}|\text{Class}=\text{Yes}) \\ &\quad * P(\text{Married}|\text{Class}=\text{Yes}) \\ &\quad * P(\text{Income}=120K|\text{Class}=\text{Yes}) \\ &= 1 * 0 * 1.2 * 10^{-9} = 0 \end{aligned}$$

Since $P(X|\text{No})P(\text{No}) > P(X|\text{Yes})P(\text{Yes})$

Therefore $P(\text{No}|X) > P(\text{Yes}|X)$

=> Class = No

Naïve Bayes Classifier

Probability estimation:

Original: $P(A_i | C_j) = \frac{N_{ij}}{N_j}$

Issue: If one of the conditional probabilities is zero, then the entire expression becomes zero.

Laplace: $P(A_i | C_j) = \frac{N_{ij}+1}{N_j+c}$

c: number of classes

p: prior probability

m-estimate: $P(A_i | C_j) = \frac{N_{ij}+mp}{N_j+m}$

m: parameter

Naïve Bayes (Summary)

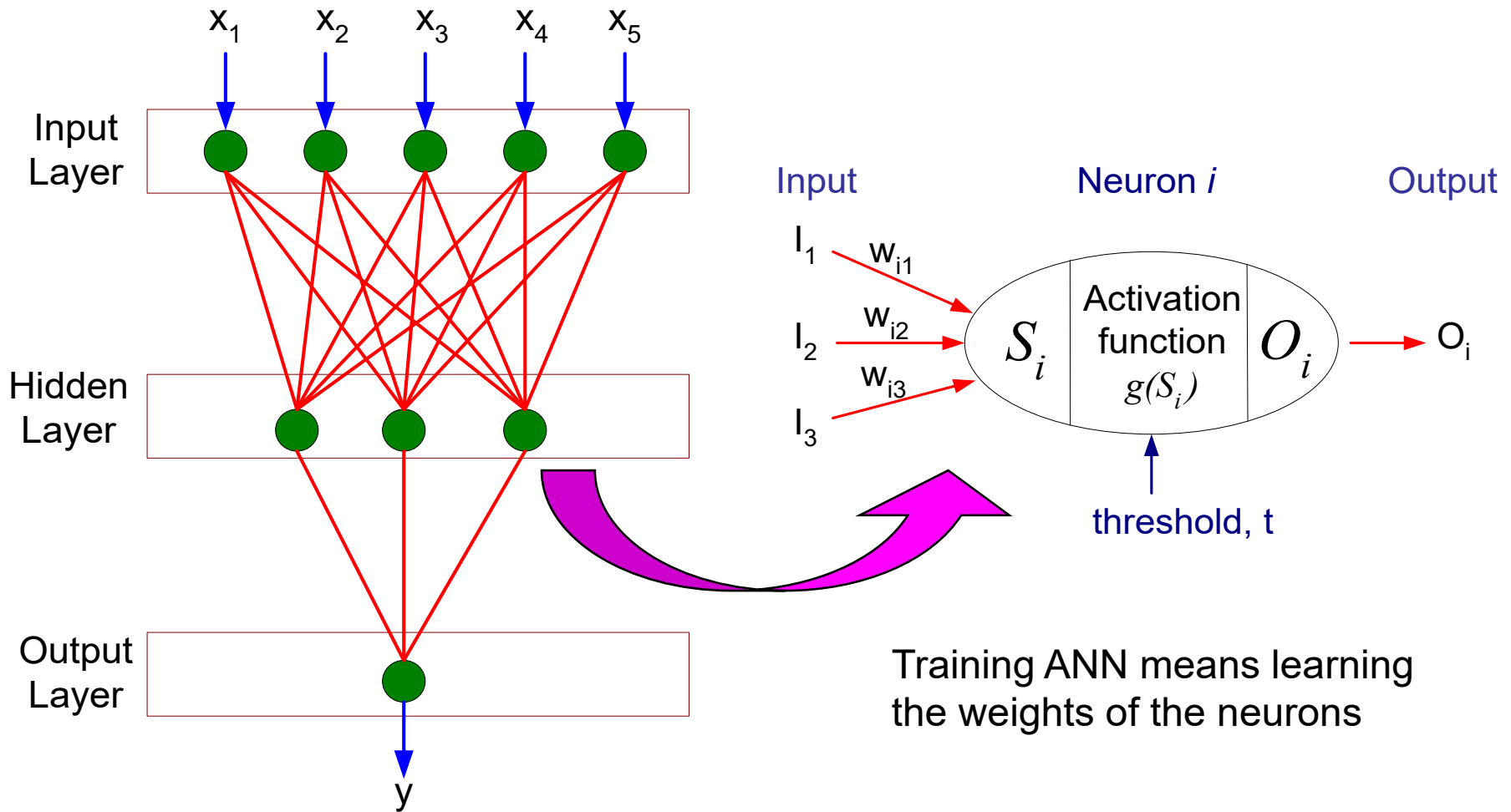
- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
 - Use other techniques such as Bayesian Belief Networks (BBN)



Topics

- Rule-Based Classifier
- Nearest Neighbor Classifier
- Naive Bayes Classifier
- **Artificial Neural Networks**
- Support Vector Machines
- Ensemble Methods

General Structure of ANN



Algorithm for learning ANN

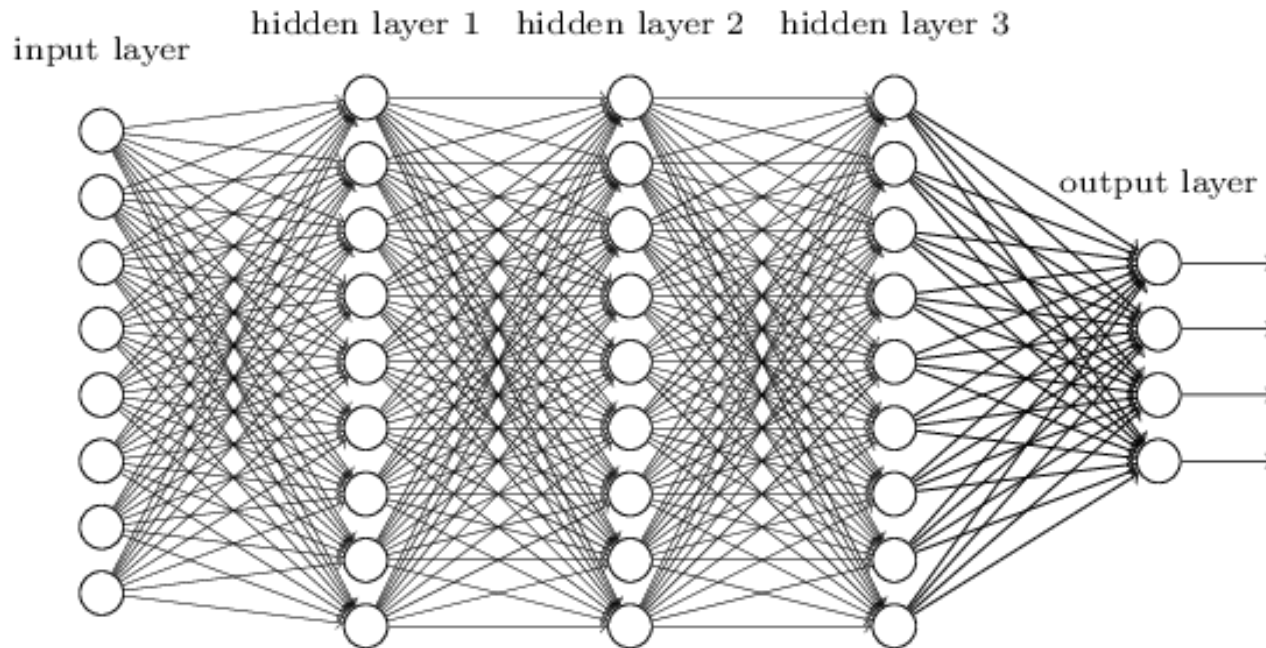
- Initialize the weights (w_0, w_1, \dots, w_k)
- Adjust the weights in such a way that the output of ANN is consistent with class labels of training examples

—Objective function:

$$E = \sum_i [Y_i - f(w_i, X_i)]^2$$

- Find the weights w_i 's that minimize the above objective function. Methods: backpropagation algorithm, gradient descend

Deep Learning / Deep Neural Networks



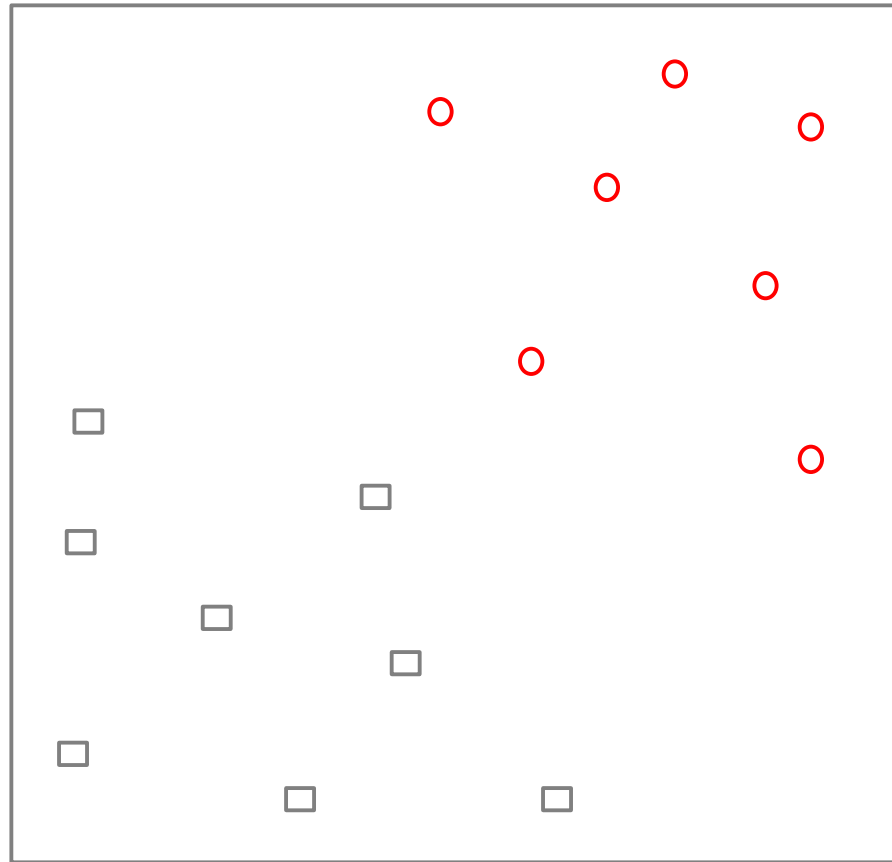
- Needs lots of data + computation (GPU)
- Applications: computer vision, speech recognition, natural language processing, audio recognition, machine translation, bioinformatics, ...
- Tools: Keras, Tensorflow and many others.
- Related: Deep belief networks, recurrent neural networks (RNN), convolutional neural network (CNN)



Topics

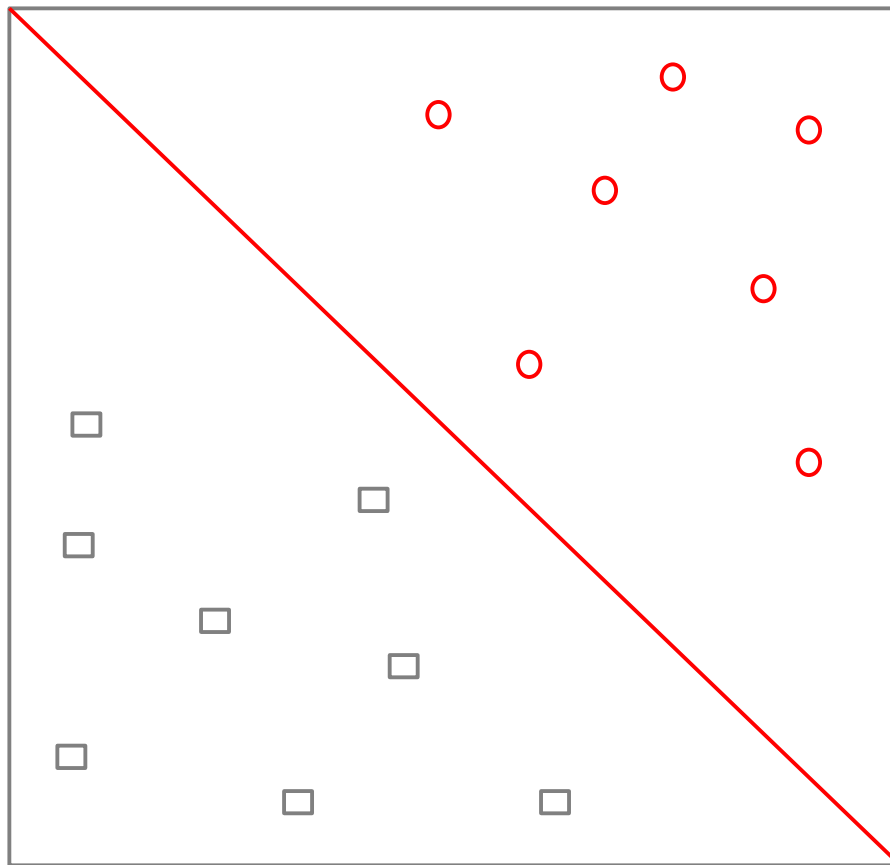
- Rule-Based Classifier
- Nearest Neighbor Classifier
- Naive Bayes Classifier
- Artificial Neural Networks
- **Support Vector Machines**
- Ensemble Methods

Support Vector Machines



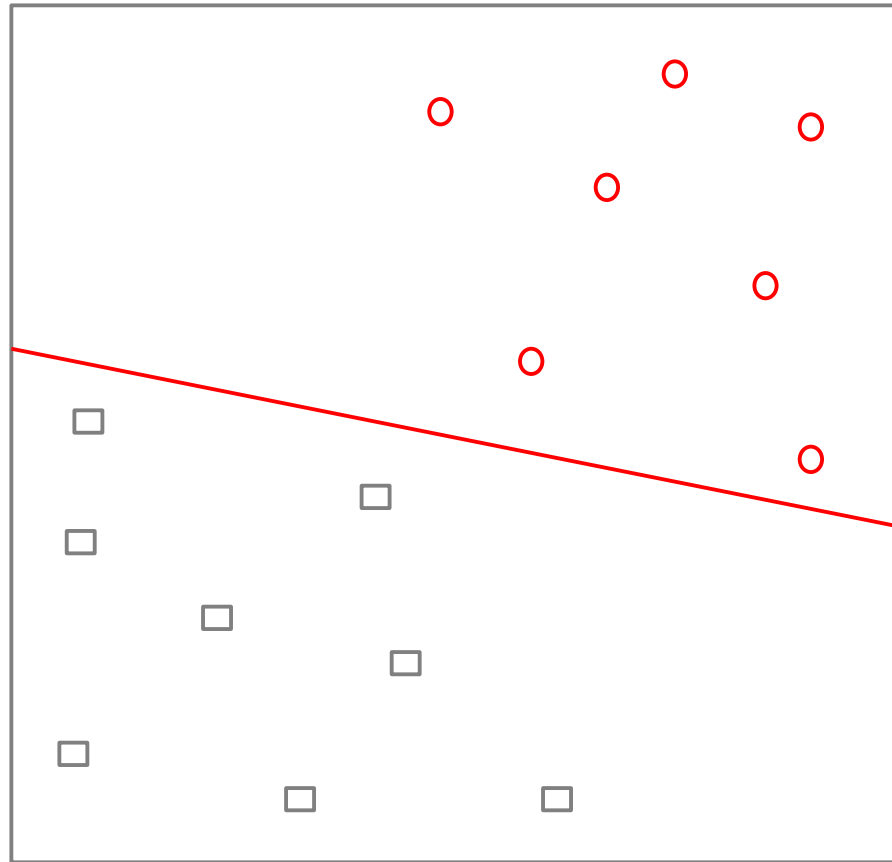
Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



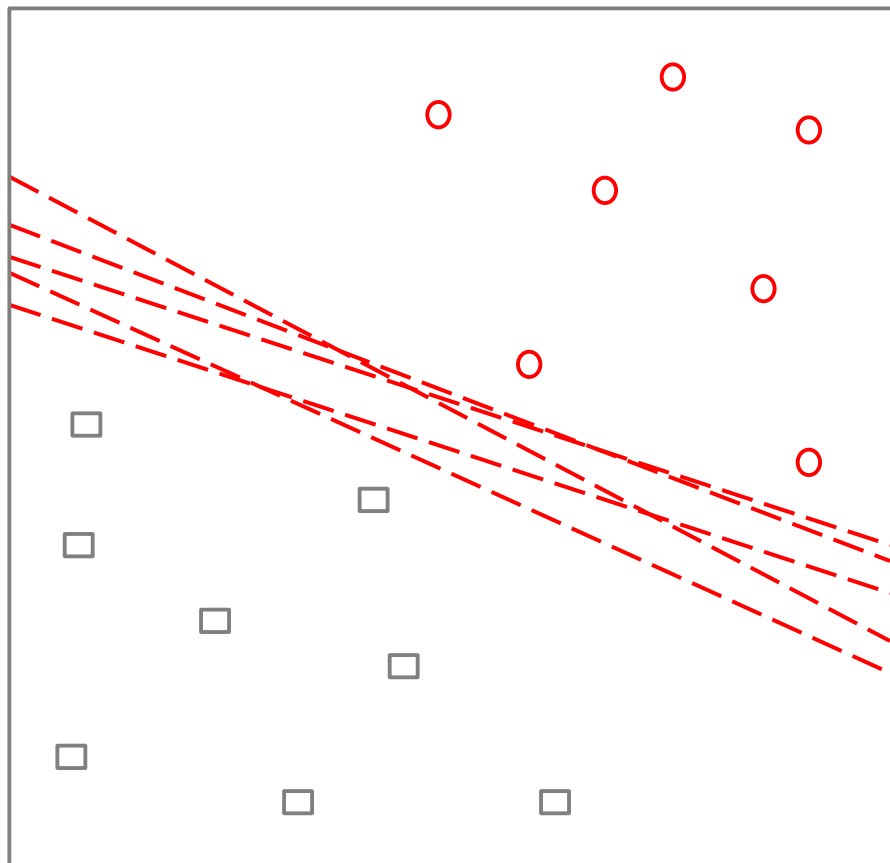
One Possible Solution

Support Vector Machines



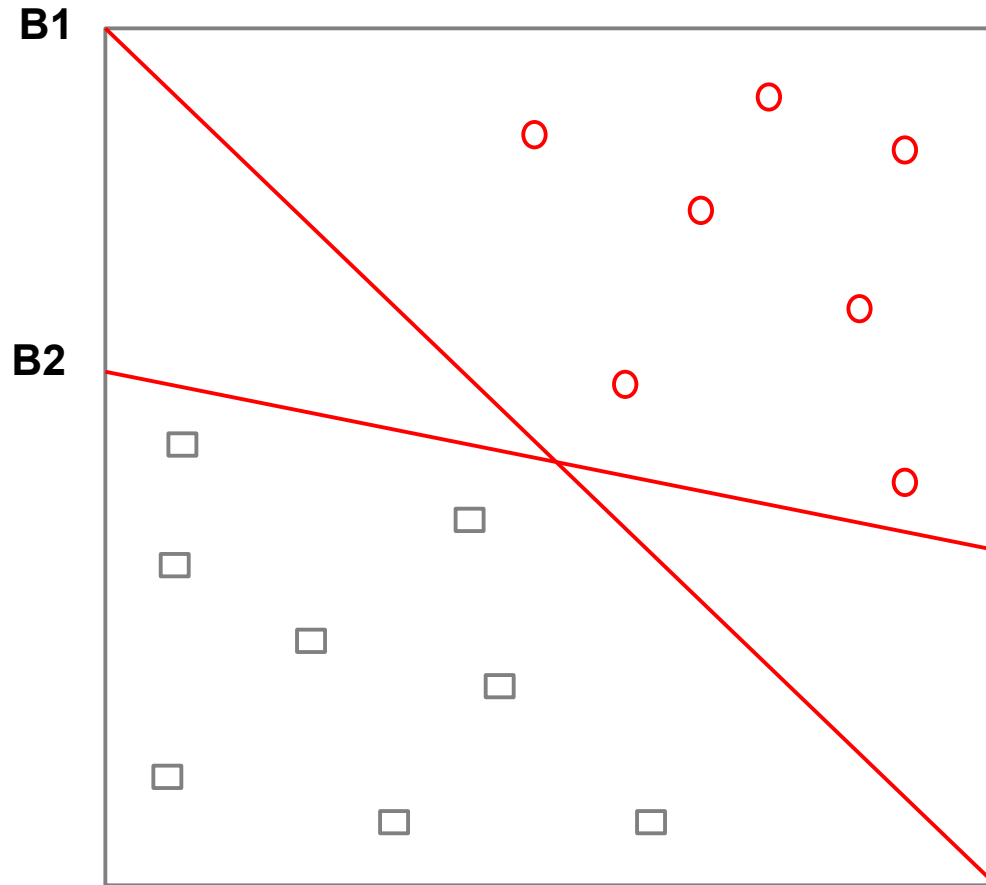
Another possible solution

Support Vector Machines



Other possible solutions

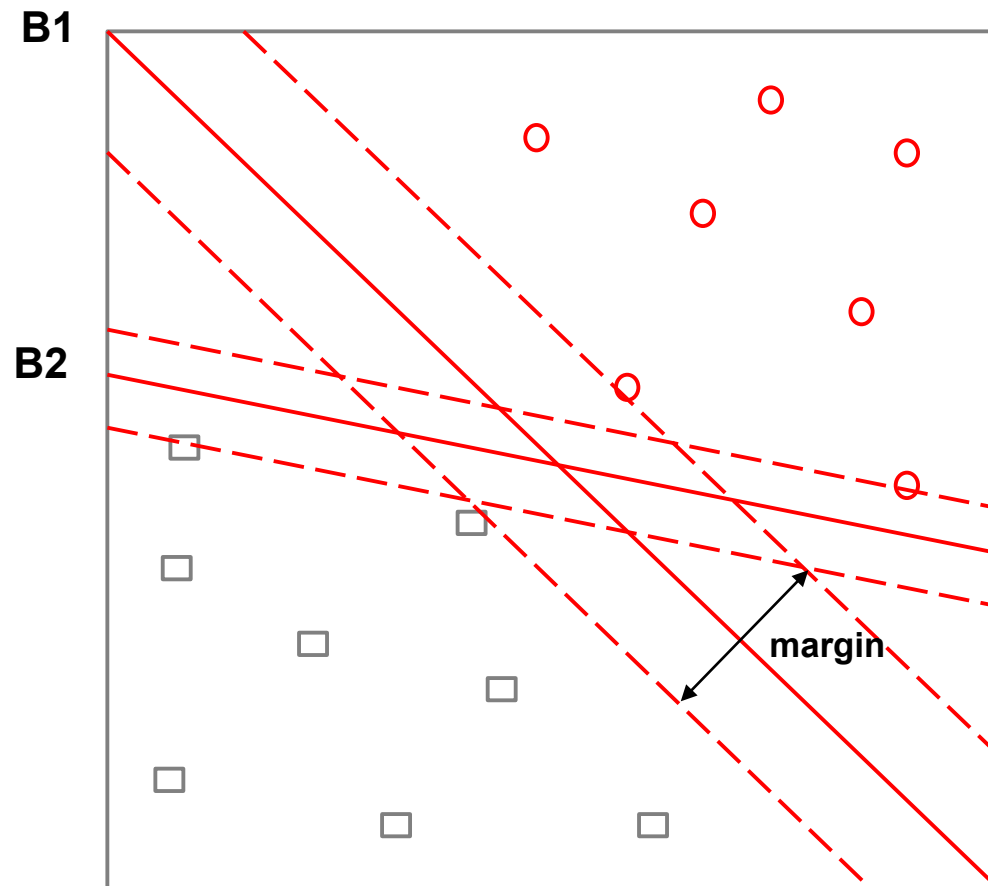
Support Vector Machines



Which one is better? B1 or B2?

How do you define better?

Support Vector Machines

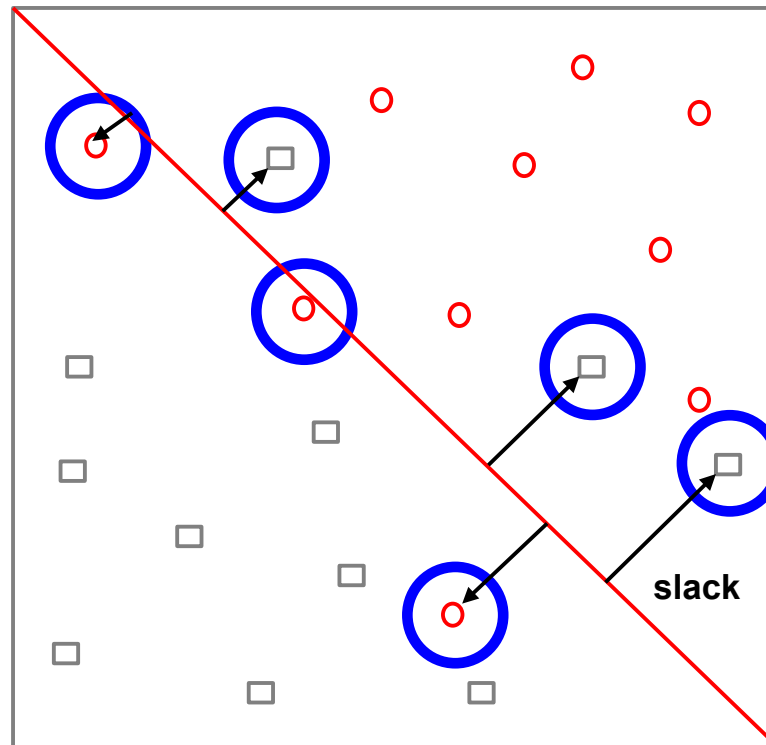


Find hyperplane **maximizes** the margin => B1 is better than B2

Larger margin = more robust = less expected generalization error

Support Vector Machines

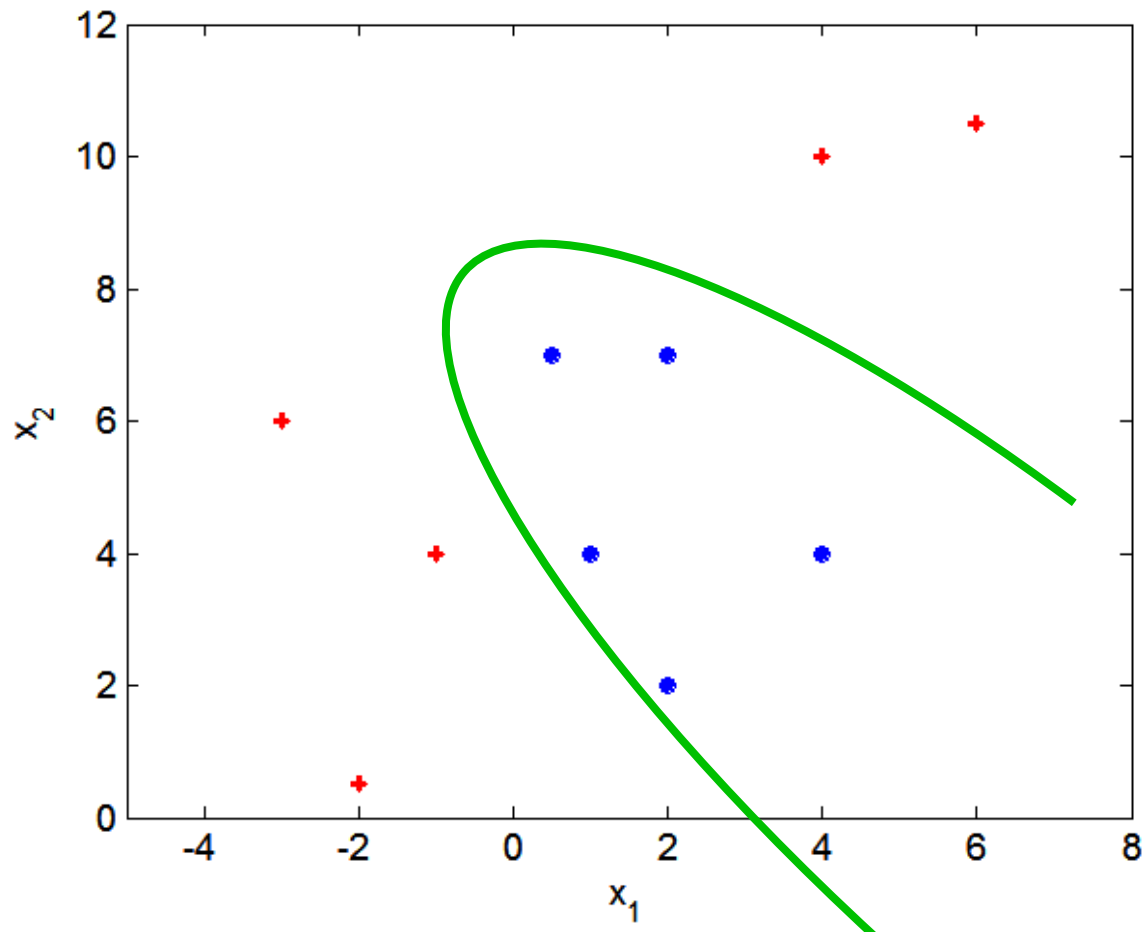
What if the problem is not linearly separable?



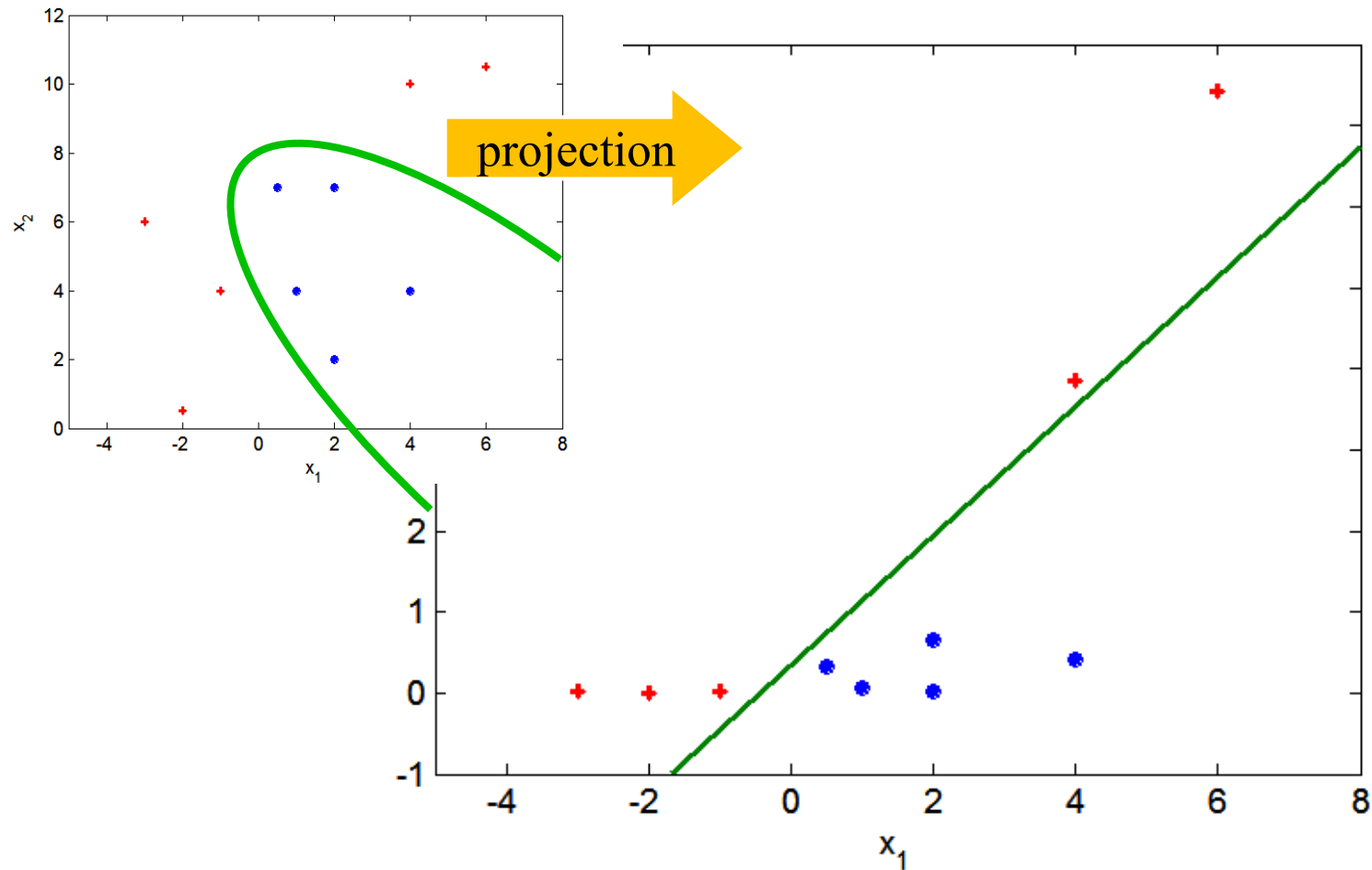
- Use slack variables to account for violations
- Use hyperplane that minimizes slack

Nonlinear Support Vector Machines

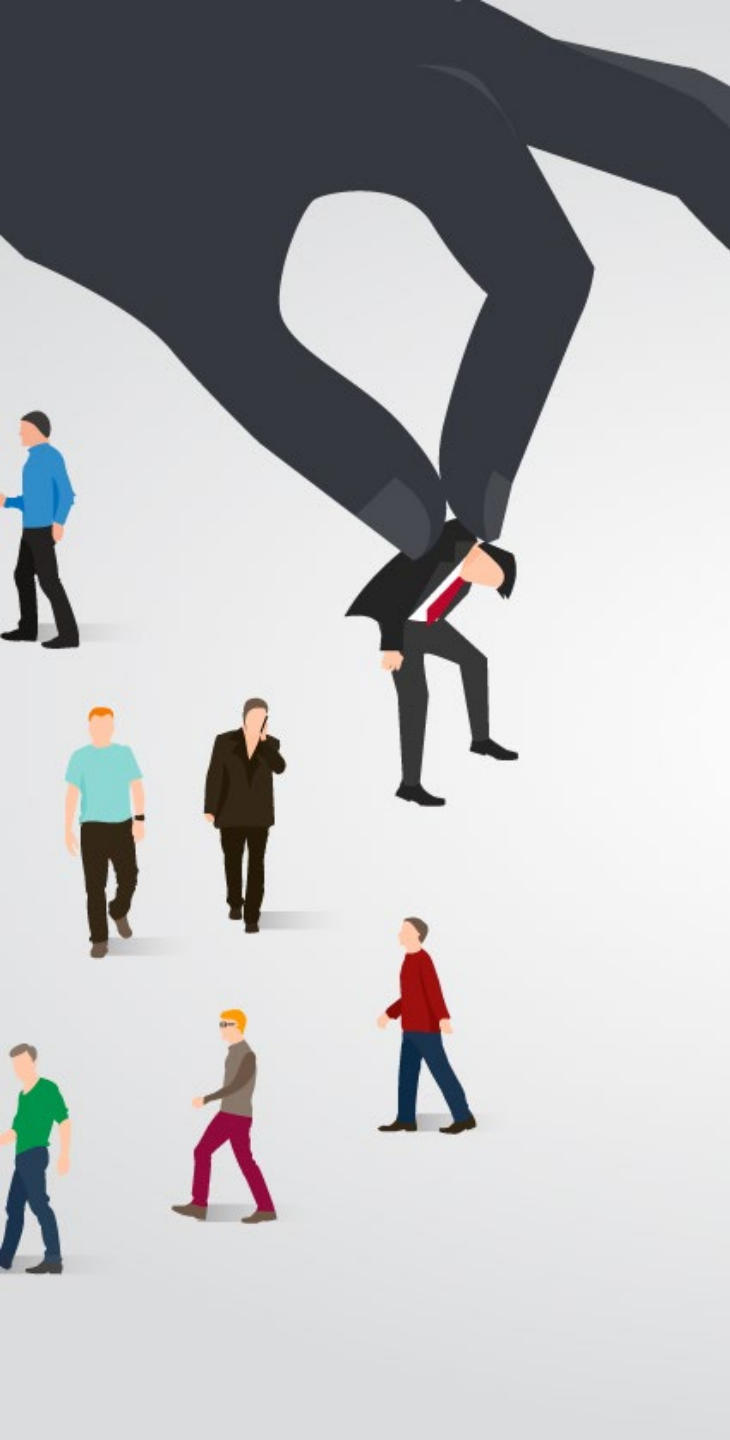
- What if decision boundary is not linear?



Nonlinear Support Vector Machines



- Project data into higher dimensional space
- Using the Kernel trick!



Topics

- Rule-Based Classifier
- Nearest Neighbor Classifier
- Naive Bayes Classifier
- Artificial Neural Networks
- Support Vector Machines
- **Ensemble Methods**

Ensemble Methods

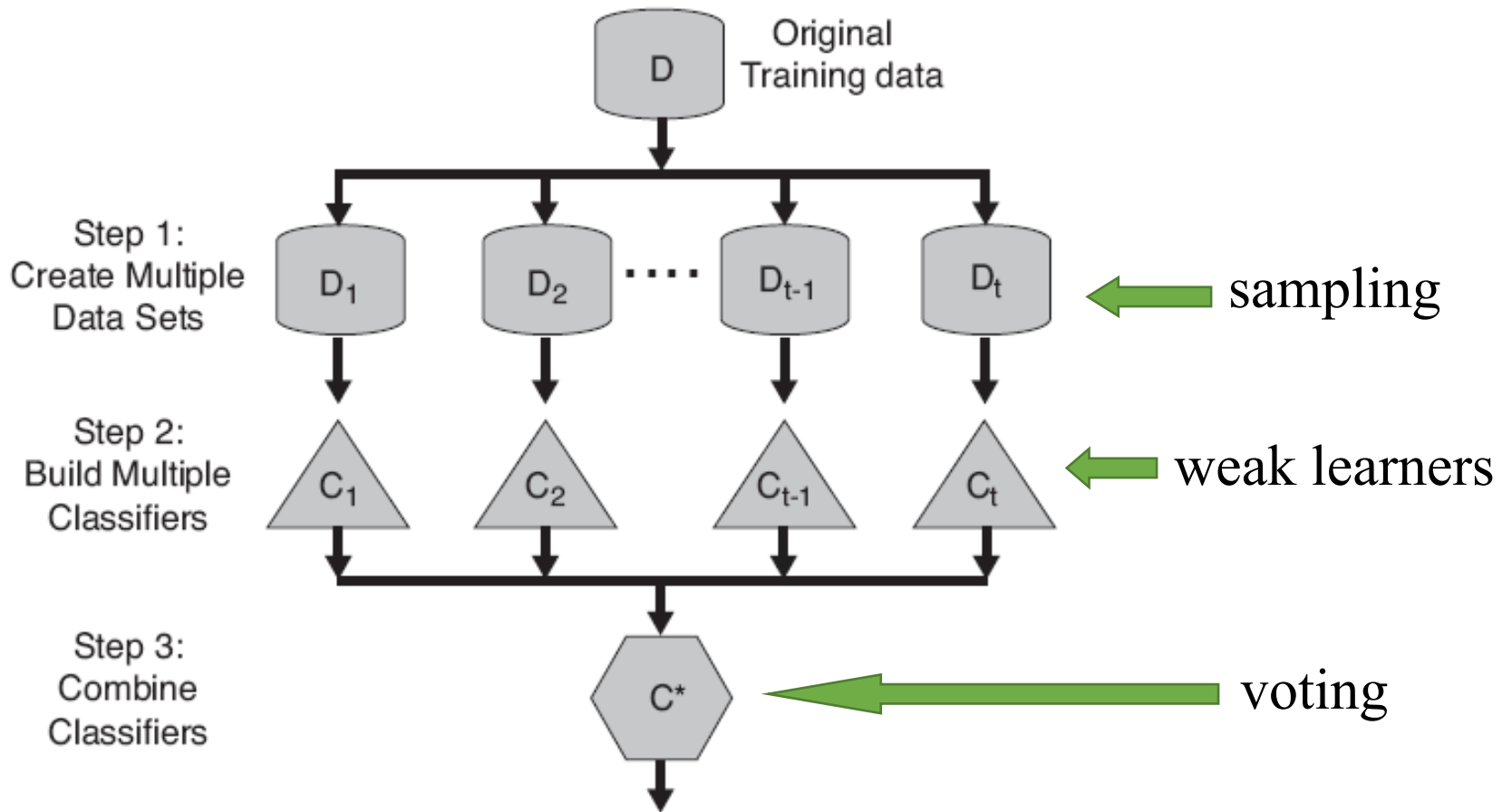
Method

1. Construct a set of (possibly weak) classifiers from the training data
2. Predict class label of previously unseen records by aggregating predictions made by multiple classifiers

Advantages

- Improve the stability and often also the accuracy of classifiers.
- Reduces variance in the prediction
- Reduces overfitting

General Idea



Why does it work?

- Suppose there are 25 base classifiers
 - Each classifier has error rate, $\epsilon = 0.35$
 - Assume classifiers are independent (different features and/or training data)
 - Probability that the ensemble classifier makes a wrong prediction:

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

= Probability that 13 or more classifier make the wrong decision

Notes

- 13 is the majority vote
- The binomial coefficient gives the number of ways you can choose i out of 25

Examples of Ensemble Methods

- How to generate an ensemble of classifiers?
 - Bagging
 - Boosting
 - Random Forests

Bagging (Bootstrap Aggregation)

1. **Sampling** with replacement (bootstrap sampling)

| | | | | | | | | | | |
|--------------------------|---|---|----|----|---|---|----|----|---|----|
| Original Data | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Bagging (Round 1) | 7 | 8 | 10 | 8 | 2 | 5 | 10 | 10 | 5 | 9 |
| Bagging (Round 2) | 1 | 4 | 9 | 1 | 2 | 3 | 2 | 7 | 3 | 2 |
| Bagging (Round 3) | 1 | 8 | 5 | 10 | 5 | 5 | 9 | 6 | 3 | 7 |

Note: some objects are chosen multiple times in a bootstrap sample while others are not chosen! A typical bootstrap sample contains about 63% of the objects in the original data.

2. **Build classifiers**, one for each bootstrap sample (classifiers are hopefully independent since they are learned from different subsets of the data)

3. **Aggregate** the classifiers' results by averaging or voting

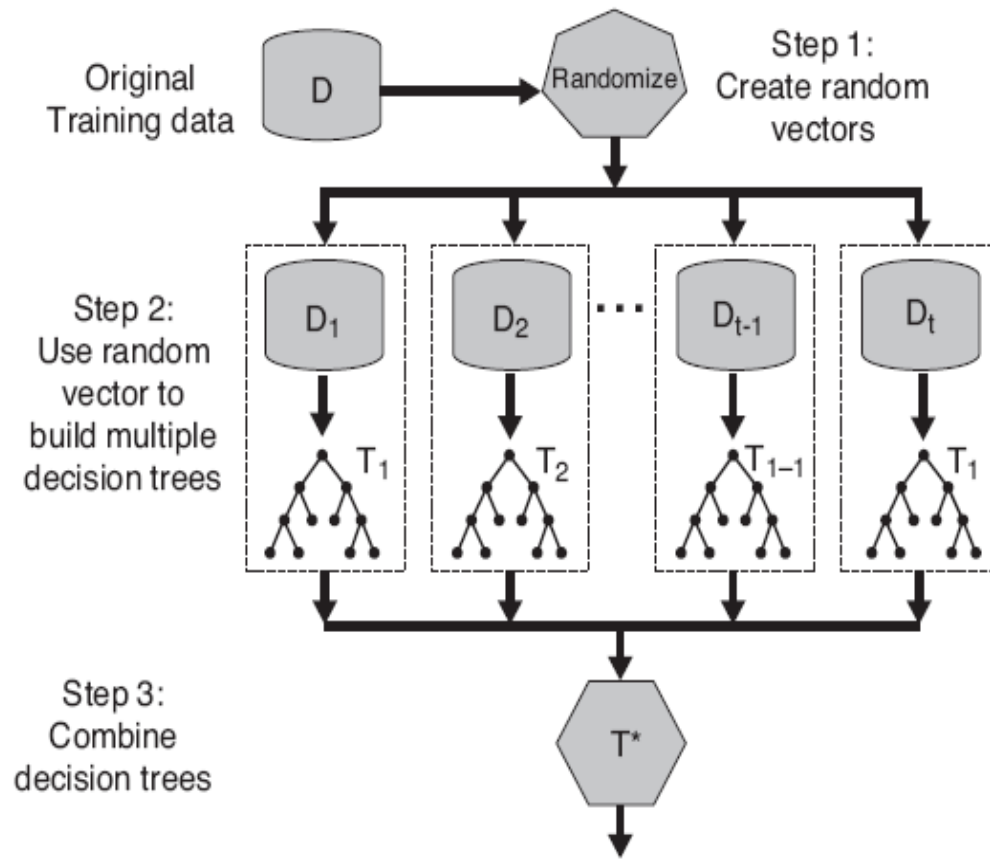
Boosting

- Records that are incorrectly classified in one round will have their weights increased in the next

| | | | | | | | | | | |
|--------------------|---|---|---|----|---|---|---|----|---|----|
| Original Data | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Boosting (Round 1) | 7 | 3 | 2 | 8 | 7 | 9 | 4 | 10 | 6 | 3 |
| Boosting (Round 2) | 5 | 4 | 9 | 4 | 2 | 5 | 1 | 7 | 4 | 2 |
| Boosting (Round 3) | 4 | 4 | 8 | 10 | 4 | 5 | 4 | 6 | 3 | 4 |

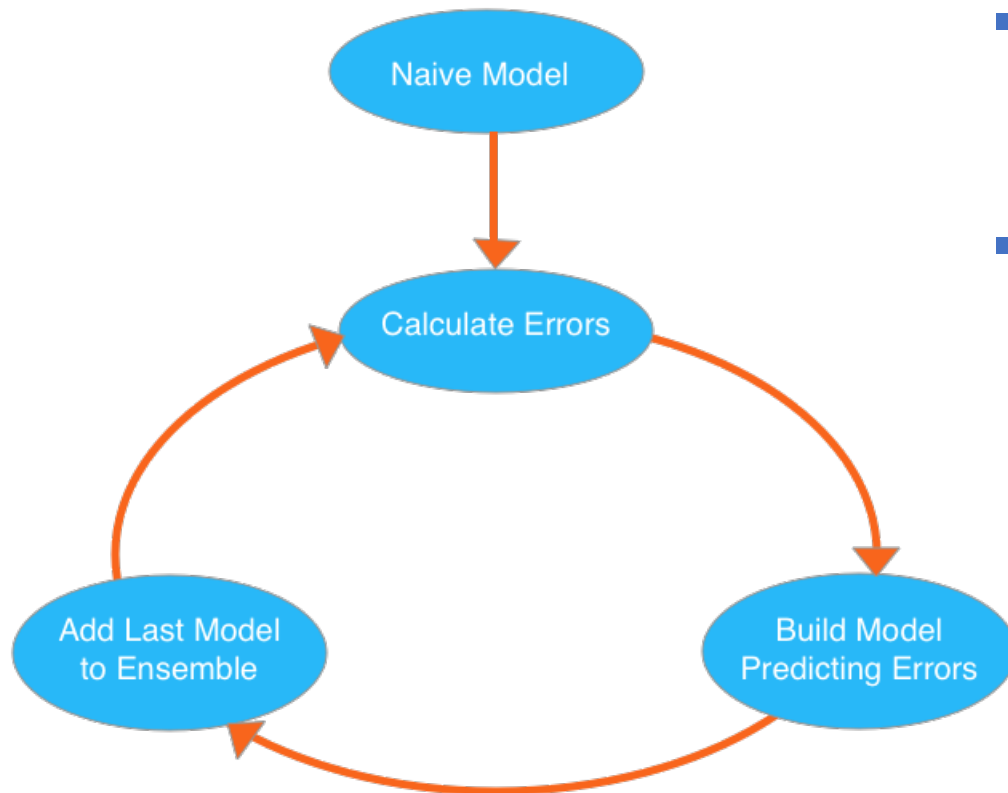
- Example 4 is hard to classify. Its weight is increased; therefore it is more likely to be chosen again in subsequent rounds
- Popular algorithm:** AdaBoost (Adaptive Boosting) typically uses decision trees as the weak learner.

Random Forests



- Introduce two sources of randomness: “Bagging” and “Random input vectors”
- **Bagging method:** each tree is grown using a bootstrap sample of training data
- **Random vector method:** At each node, best split is chosen only from a random sample of the m possible attributes.

Gradient Boosted Decision Trees (XGBoost)



- **Idea:** build models to predict (correct) errors (= boosting).

- **Approach:**

1. Start with a naive (weak) model
2. Calculate errors for each observation in the dataset.
3. Build a new model to predict these errors and add to the ensemble.
4. Go to 2.

Other Popular Approaches

- Logistic Regression
- Linear Discriminant Analysis
- Regularized Models (Shrinkage)
- Stacking