Introduction to Data Mining

Chapter 4
Classification – Alternative Techniques

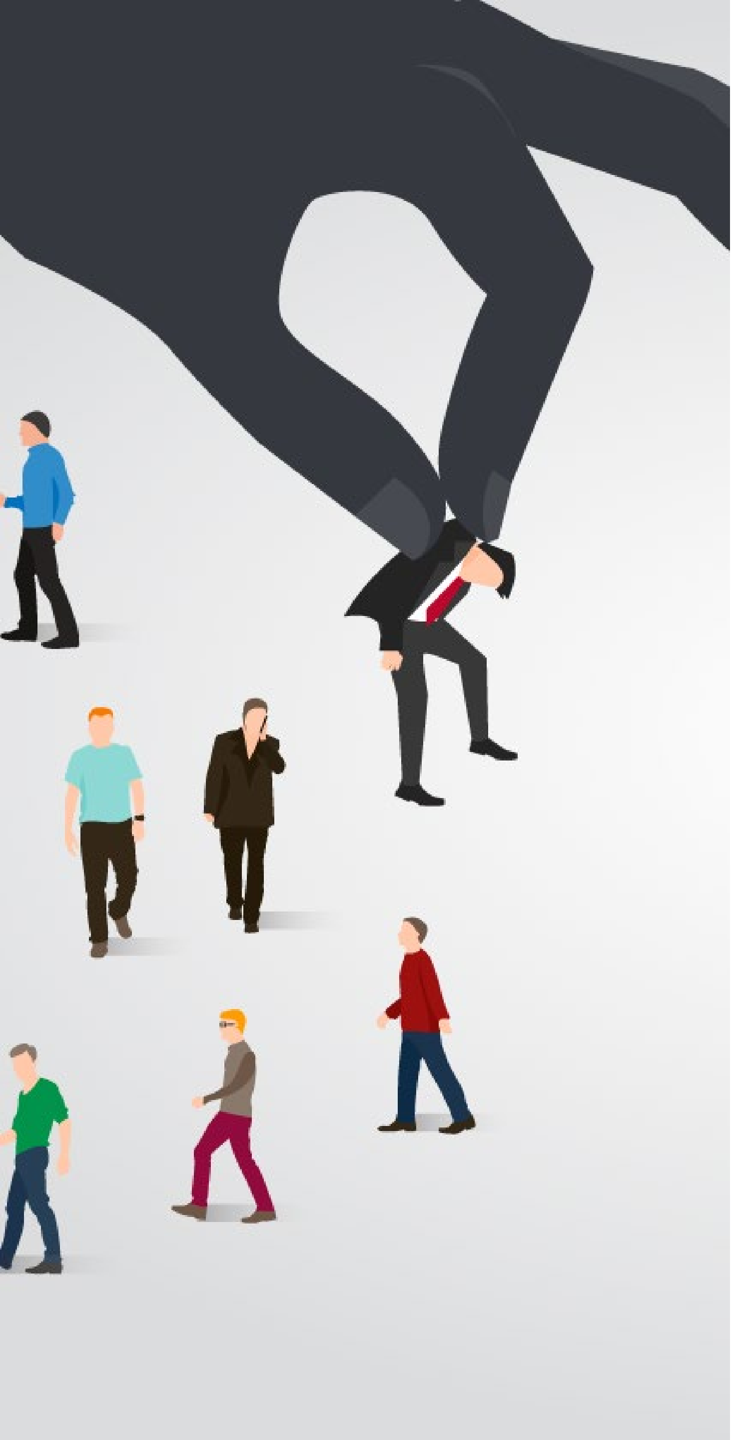
by Michael Hahsler

Based in Slides 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/



Topics

- Other Classification Methods
  - **Rule-Based Classifier**
  - Nearest Neighbor Classifier
  - Naive Bayes Classifier
  - Artificial Neural Networks
  - Support Vector Machines
  - Ensemble Methods
- Class Imbalance Problem

Rule-Based Classifier

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

Using a Rule-Based Classifier

A rule R **covers** an instance x if the attributes of the instance satisfy the condition of the rule. Such a rule can be used for classification.

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

Rule base

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 (R3 is selected below -> Amphibians)
  - 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 = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

R3: (Live in Water = sometimes) $\rightarrow$ Amphibians

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

R5: (Give Birth = no) $\rightarrow$ Amphibians

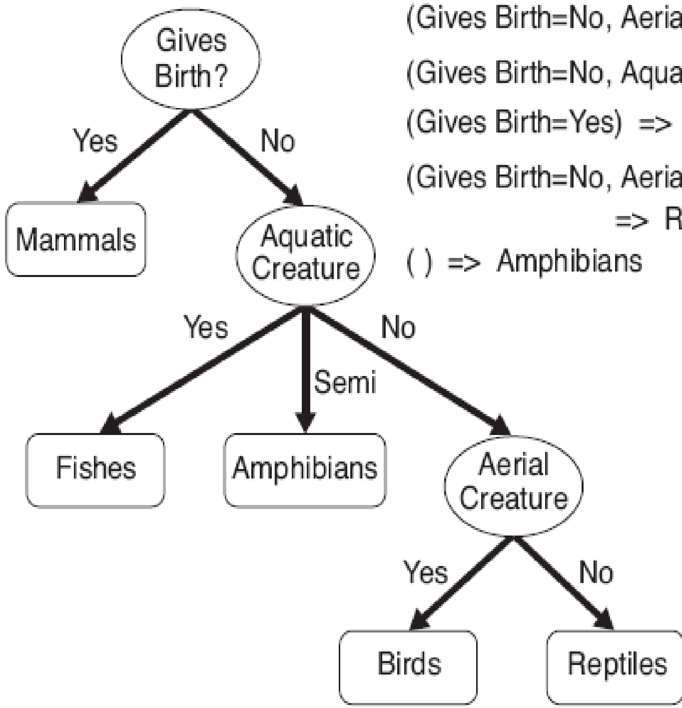
Rule base

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

R3, 4 and 5 cover the observation

- Alternative: (weighted) voting by all matching rules (-> Amphibians)

Rules From Decision Trees



Rule-Based Classifier:

(Gives Birth=No, Aerial Creature=Yes) => Birds

(Gives Birth=No, Aquatic Creature=Yes) => Fishes

(Gives Birth=Yes) => Mammals

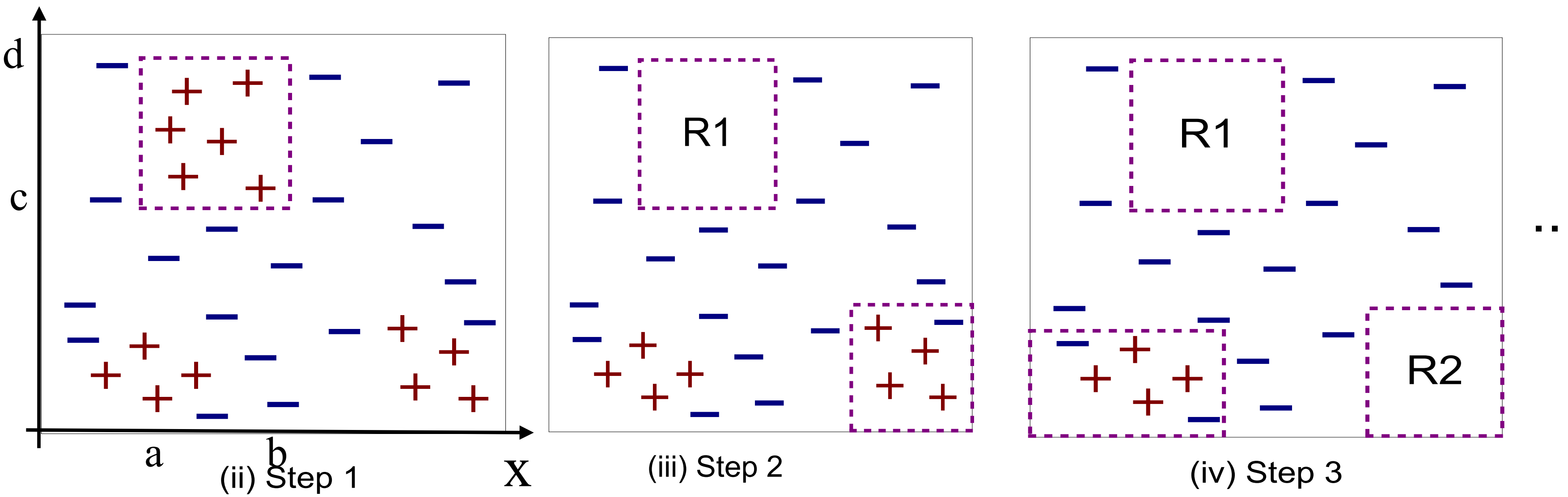
(Gives Birth=No, Aerial Creature=No, Aquatic Creature=No)
=> Reptiles

( ) => Amphibians

- Rules are created by reading the decisions in tree branches from the root to a final node.
- 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

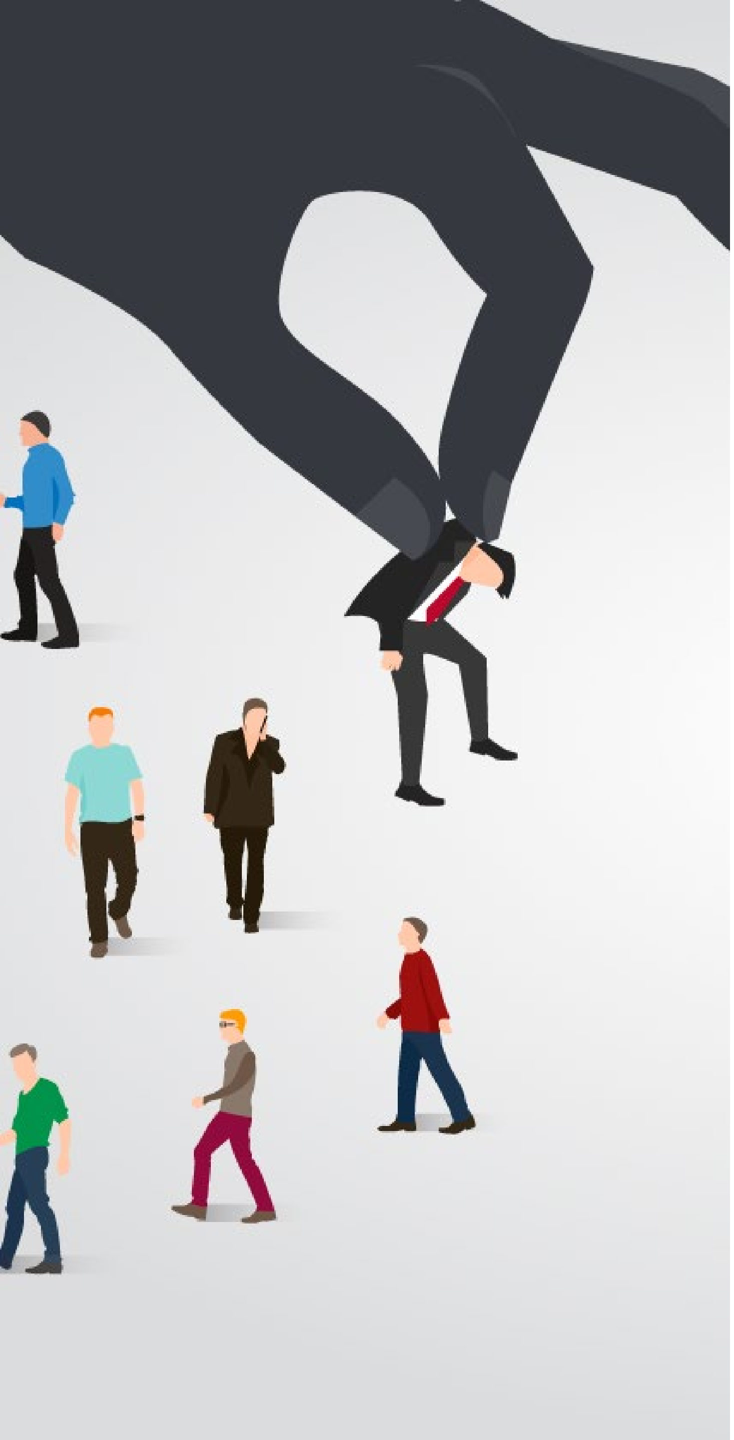
- Extract rules directly from the data
- Sequential Covering (Example: try to cover class +)



R1: $a>x>b \wedge c>y>d \longrightarrow class +$

Advantages of Rule-Based Classifiers

- As expressive as decision trees
- Easy to interpret
- Easy to generate
- Can classify new instances rapidly
- Performance comparable to decision trees

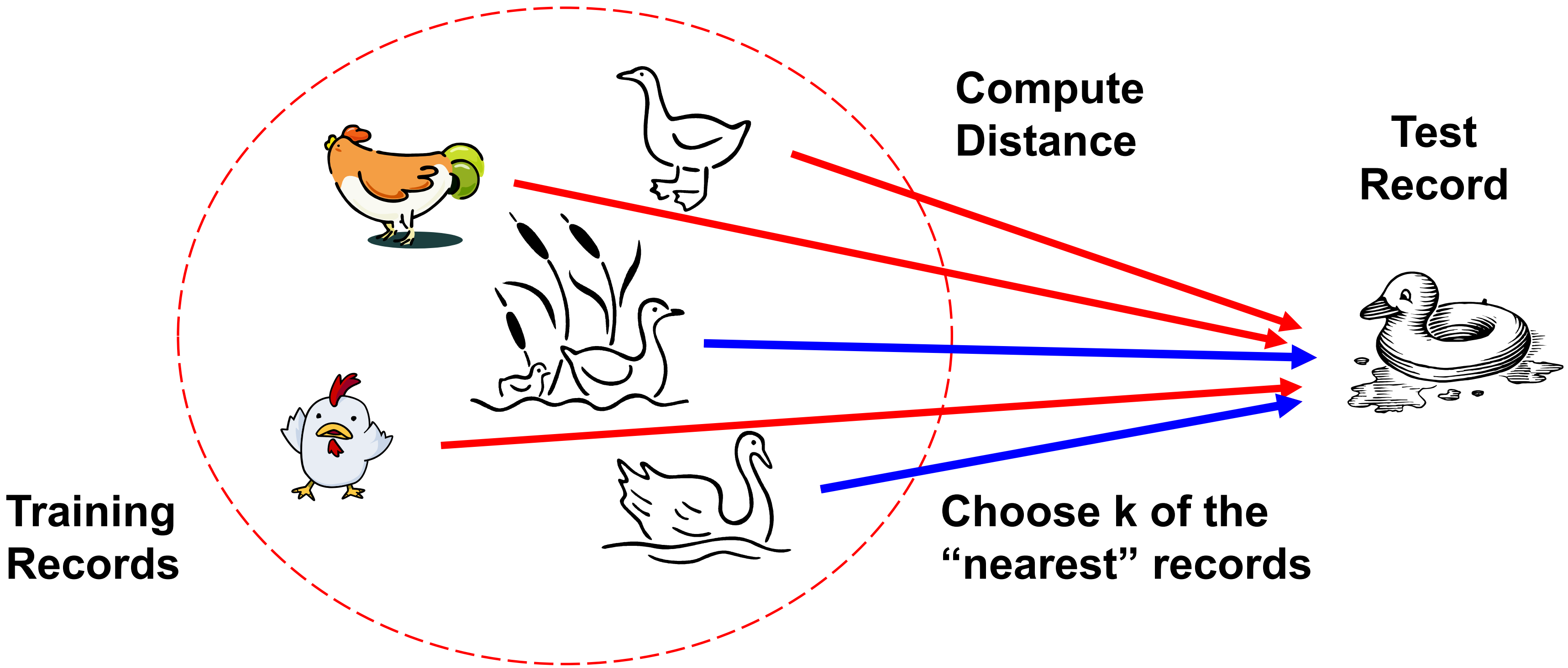


Topics

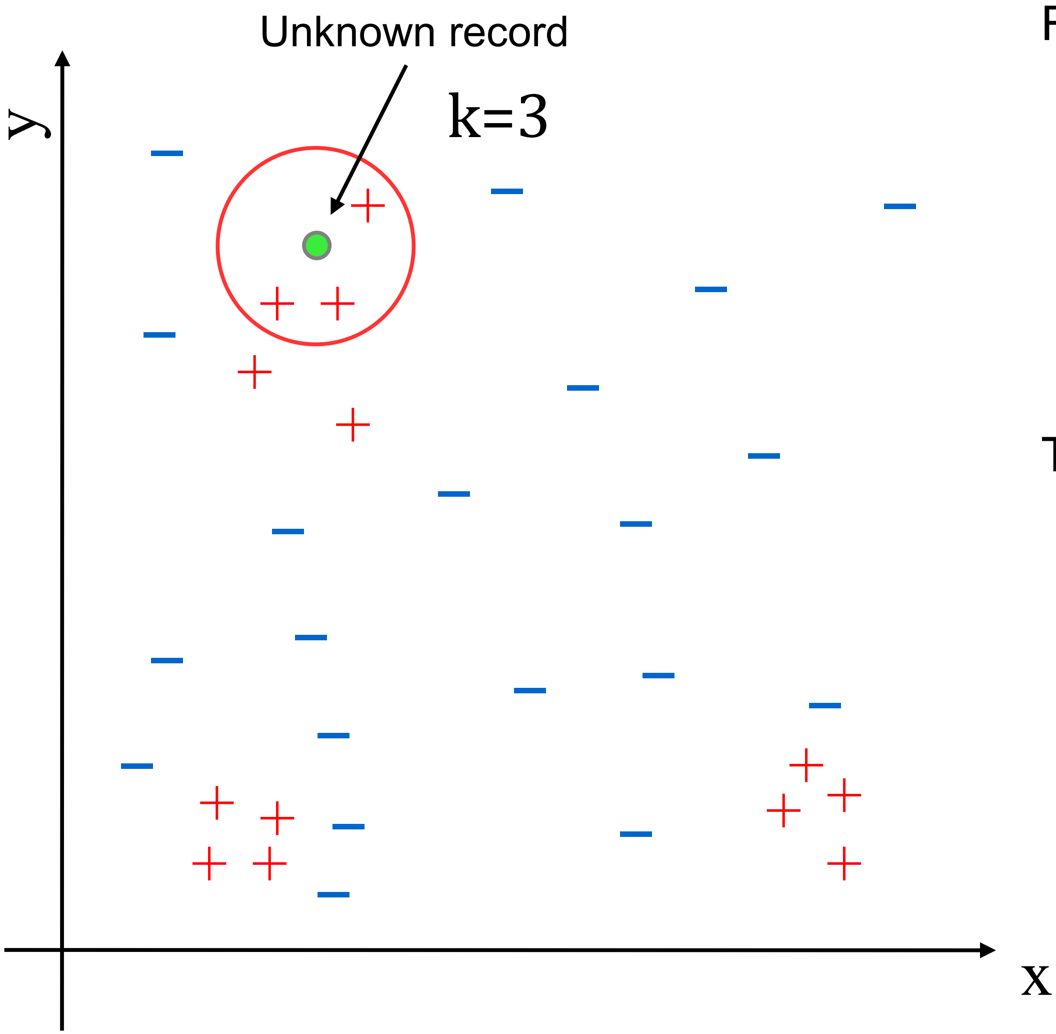
- Rule-Based Classifier
- **Nearest Neighbor Classifier**
- Naive Bayes Classifier
- Artificial Neural Networks
- Support Vector Machines
- Ensemble Methods

Nearest Neighbor Classifiers

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



Nearest-Neighbor Classifiers



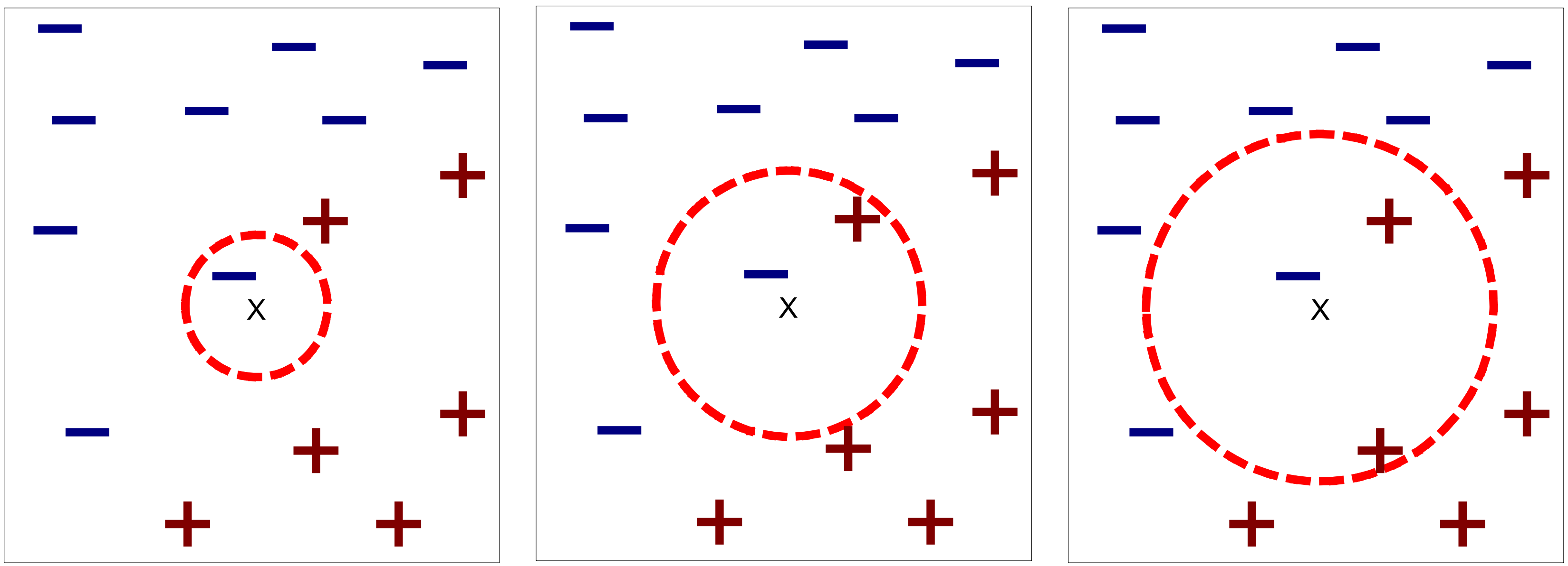
Requires three things

- 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

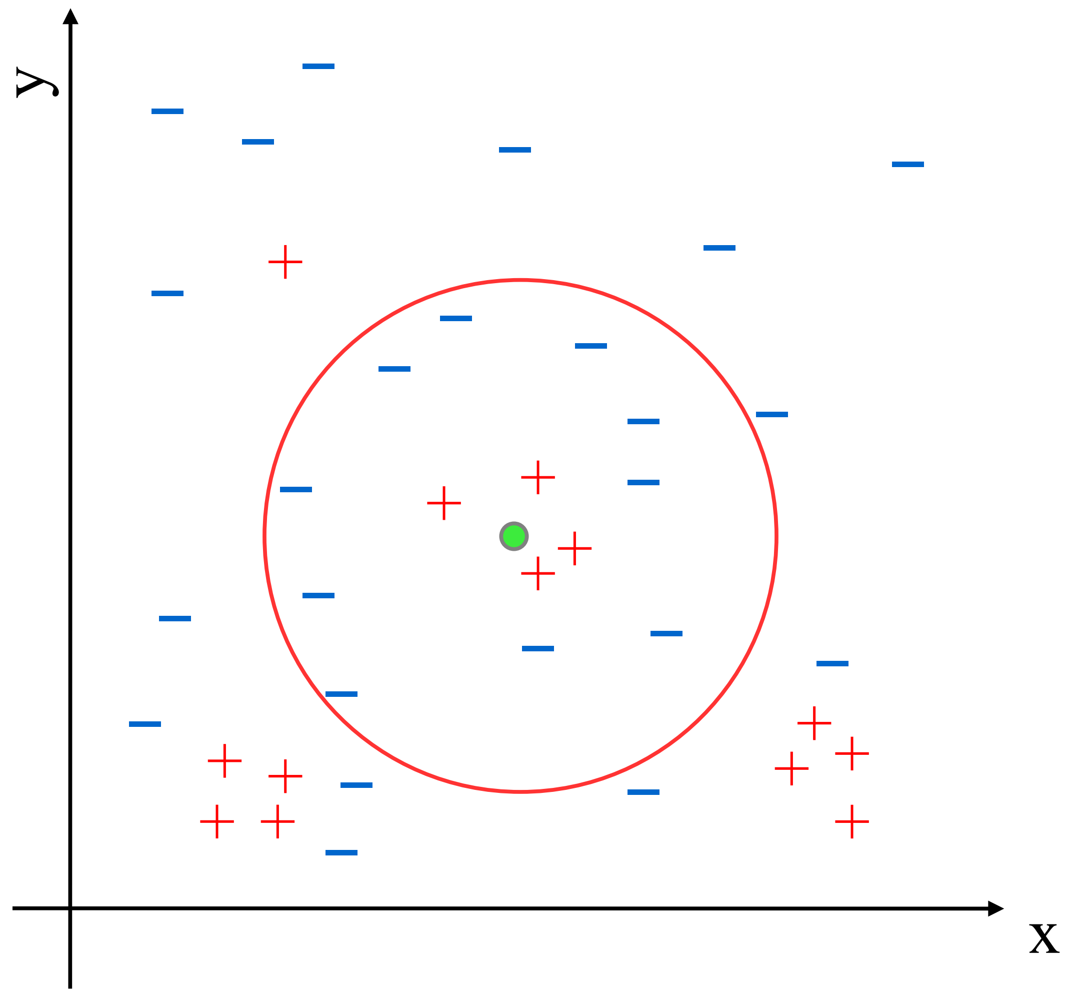
$$d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{\sum_i (p_i - q_i)^2}$$

  - Note: This means that the data needs to be scaled!

- 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



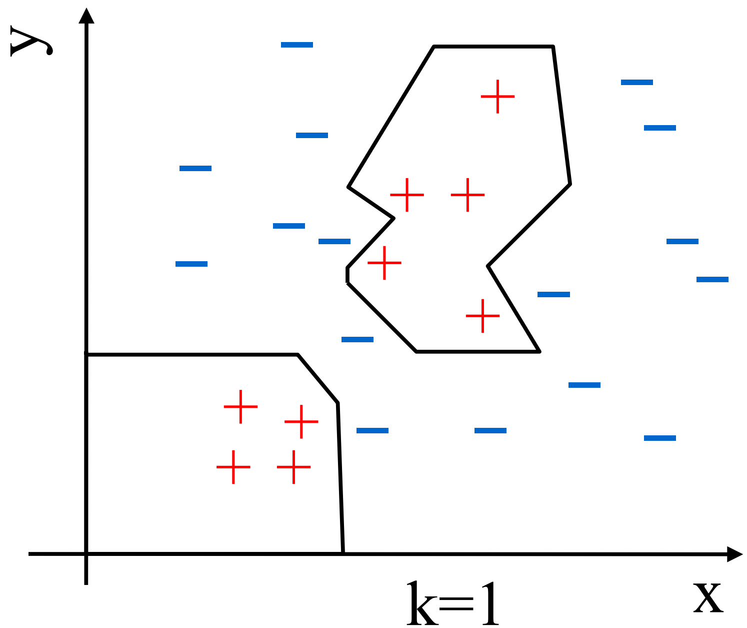
k is too large!

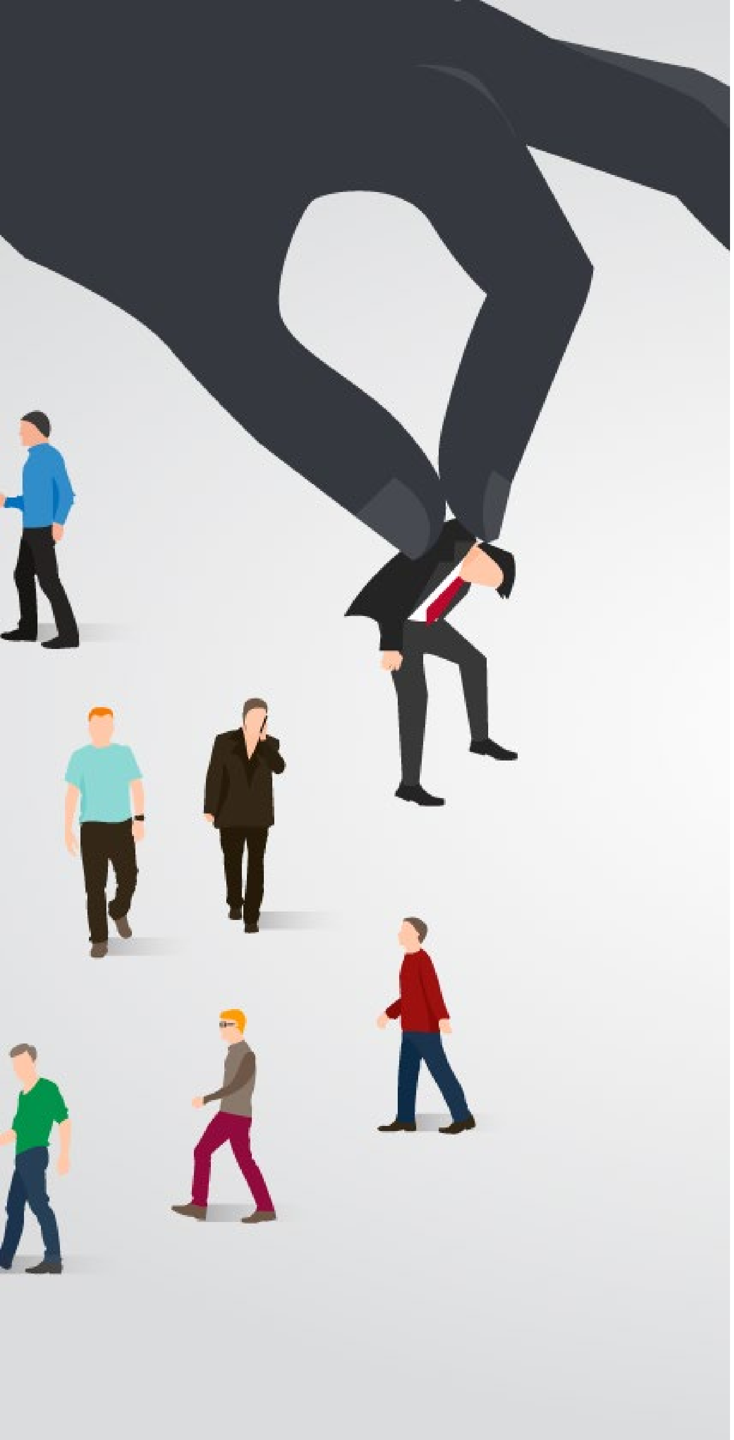
Nearest neighbor Classification…

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 arbitrary non-linear decision boundaries.

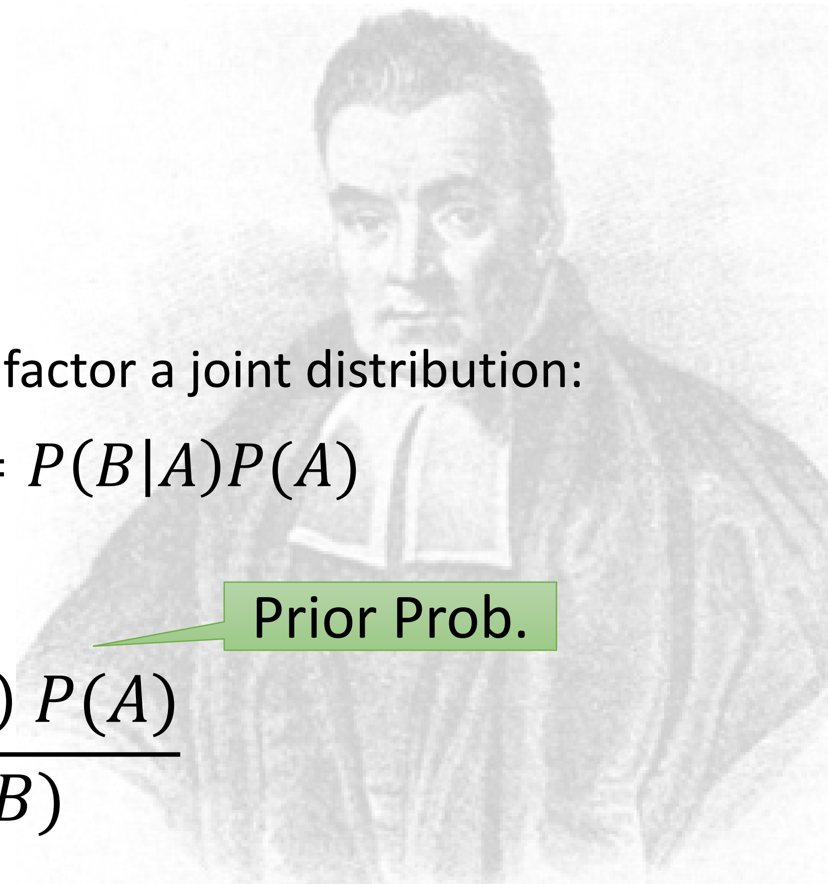




Topics

- 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(cavity | toothache) from causal probability P(toothache | 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?

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

Increases the probability by x10!

Bayesian Classifiers

- Consider each attribute and class label as a random variable.
- Classification problem: Given a record with attributes $(A_1, A_2, \ldots, A_n)$ predict class C.
- This can be done by finding the most likely class that has the largest

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

Bayesian Classifiers

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

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

- 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

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

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

Naïve Bayes Classifier

Assume independence among attributes A when class is given:

$$P(A_1, A_2, \ldots, A_n \,|C) \;=\; P(A_1|\, C)\, P(A_2|\, C) \ldots\, P(A_n|\, C) = \prod_i P(A_i \mid C)$$

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

A new observation is classified to C_j such that:

$$\max_j \; P\left(C_j\right) \prod_i P(A_i \,|C_j)$$

How to Estimate Probabilities from Data?

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

  e.g., P(C=No) = 7/10,
  P(C=Yes) = 3/10

- For discrete attributes:

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

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

e.g.
P(Status=Married | C=No) = 4/7
P(Refund=Yes | C=Yes)=0

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

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 $P(A_i \mid C_j)$

Example of Naïve Bayes Classifier

Given a Test Record what is the most likely class?

$$X = (Refund = No, Married, Income = 120K)$$

naive Bayes Classifier:

P(Refund=Yes|No) = 3/7
P(Refund=No|No) = 4/7
P(Refund=Yes|Yes) = 0
P(Refund=No|Yes) = 1
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

For taxable income:
If class=No: sample mean=110
sample variance=2975
If class=Yes: sample mean=90
sample variance=25

P(X|Class=No) = P(Refund=No|Class=No)
** P(Married| Class=No)*
** P(Income=120K| Class=No)*
*= 4/7 * 4/7 * 0.0072 = 0.0024*

P(X|Class=Yes) = P(Refund=No| Class=Yes)
** P(Married| Class=Yes)*
** P(Income=120K| Class=Yes)*
*= 1 * 0 * 1.2 * 10^{-9} = 0*

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

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

=> Class = No

Naïve Bayes Classifier

Probability estimation:

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

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

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

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

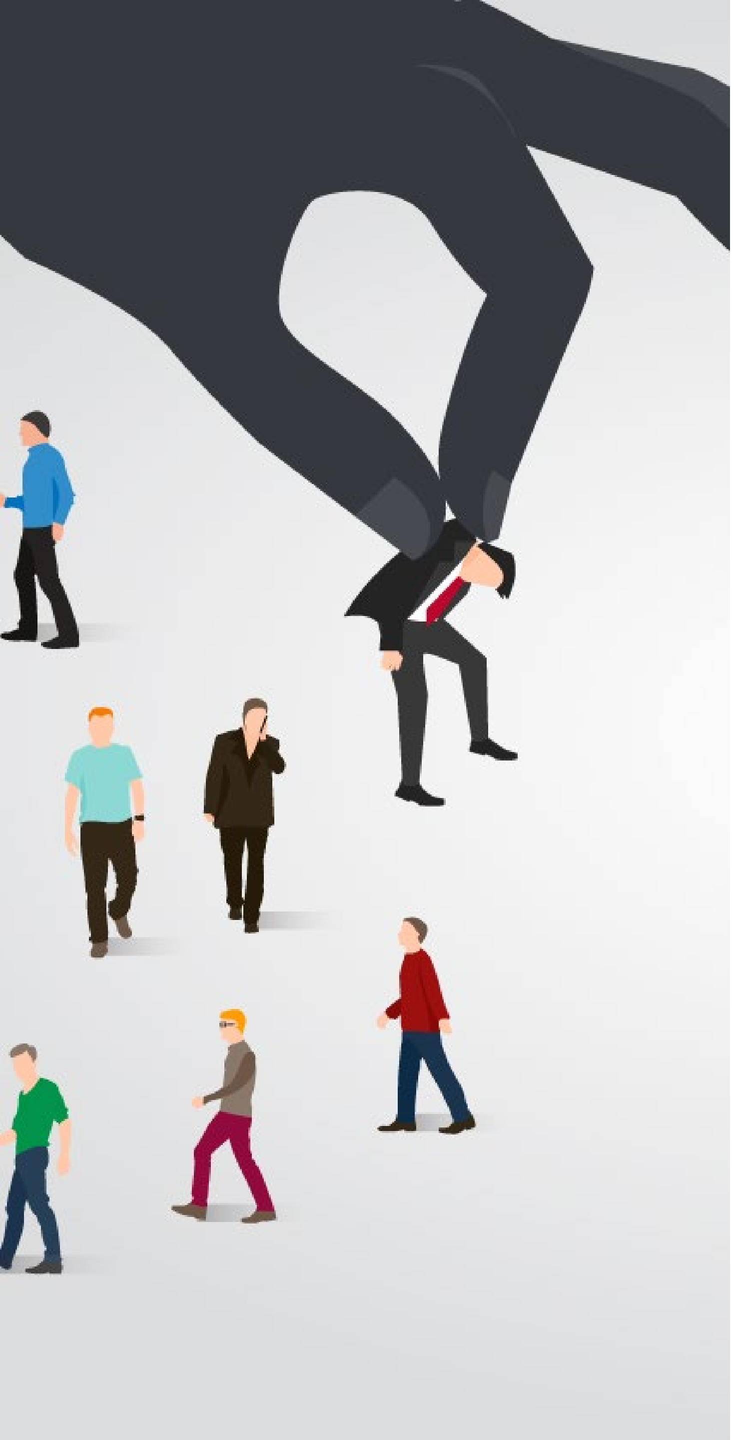
c: number of classes

p: prior probability

m: parameter

Naïve Bayes (Summary)

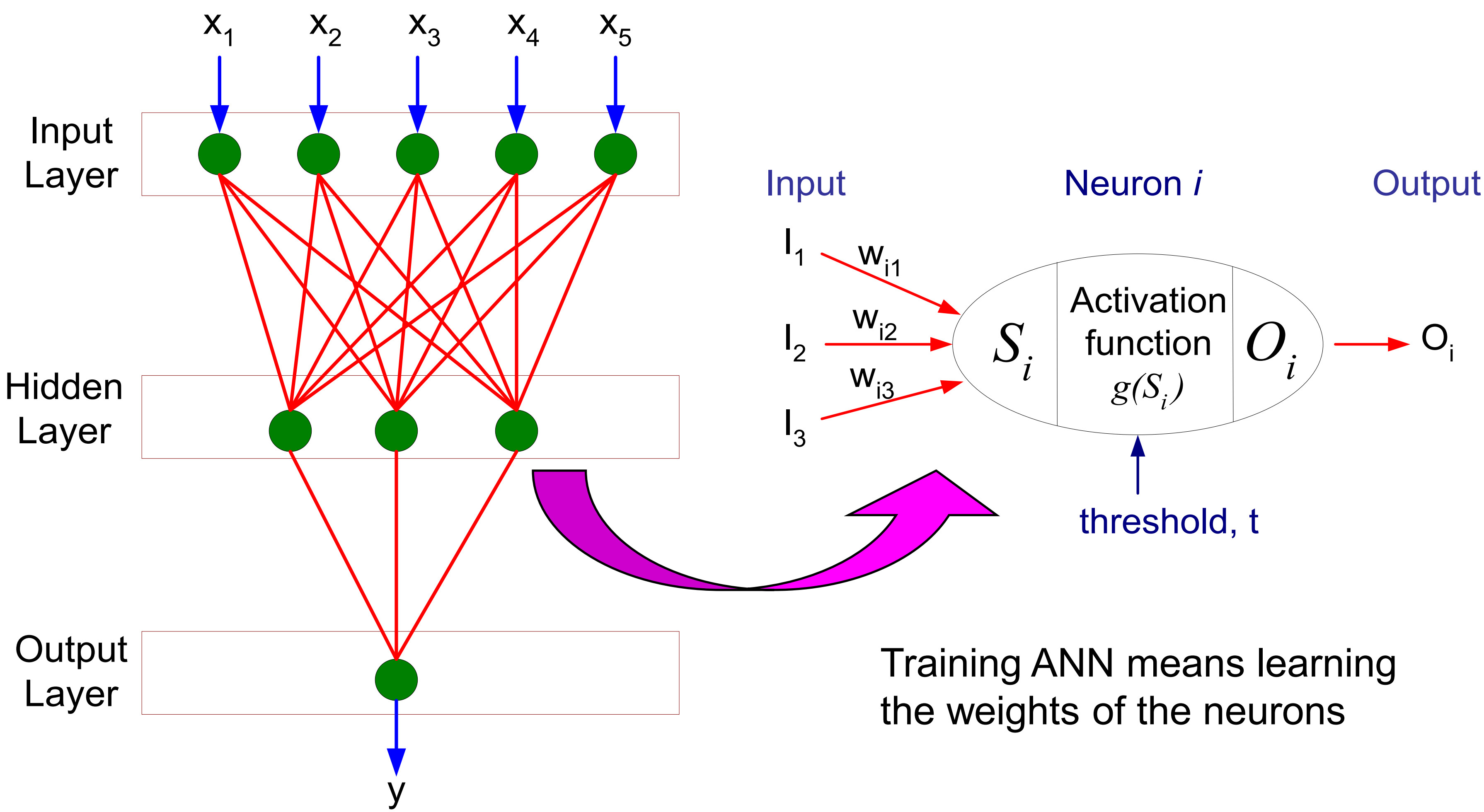
- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Independence assumption may not hold for some attributes
  - Use other techniques such as Bayesian Belief Networks (BBN)



Topics

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

General Structure of ANN



Training ANN means learning the weights of the neurons

Algorithm for learning ANN

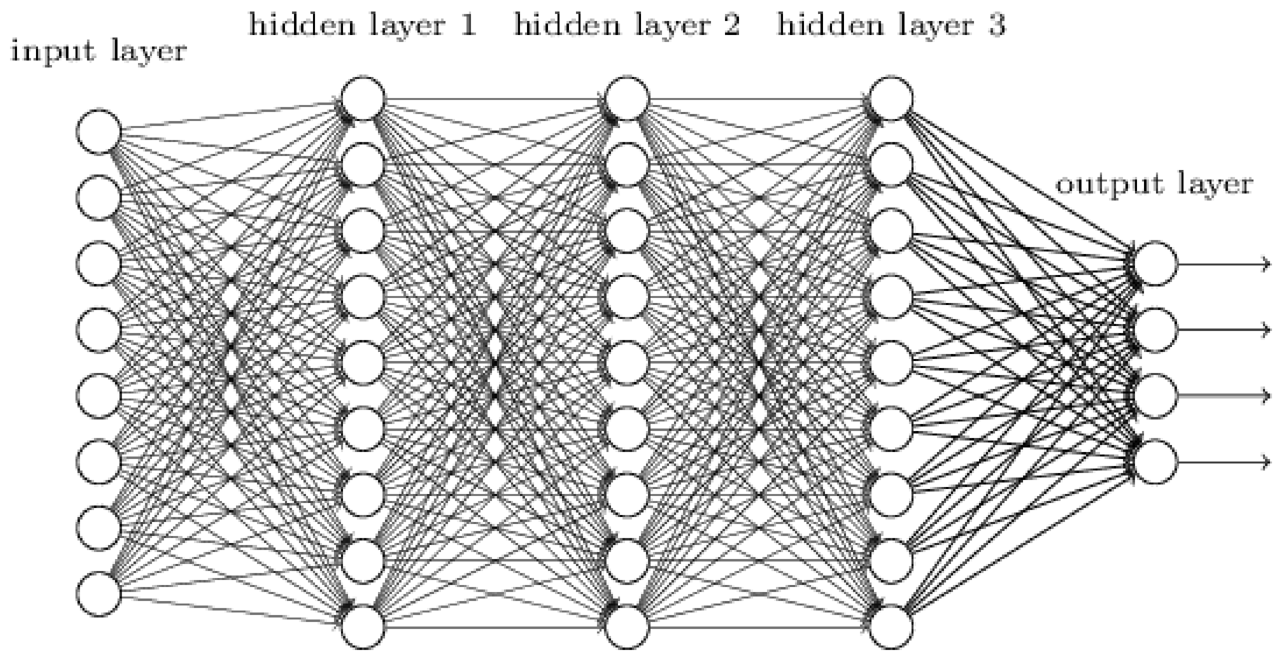
- Initialize the weights $(w_0, w_1, \ldots, w_k)$

- Adjust the weights in such a way that the output of ANN is consistent with class labels of training examples
  - Objective function:

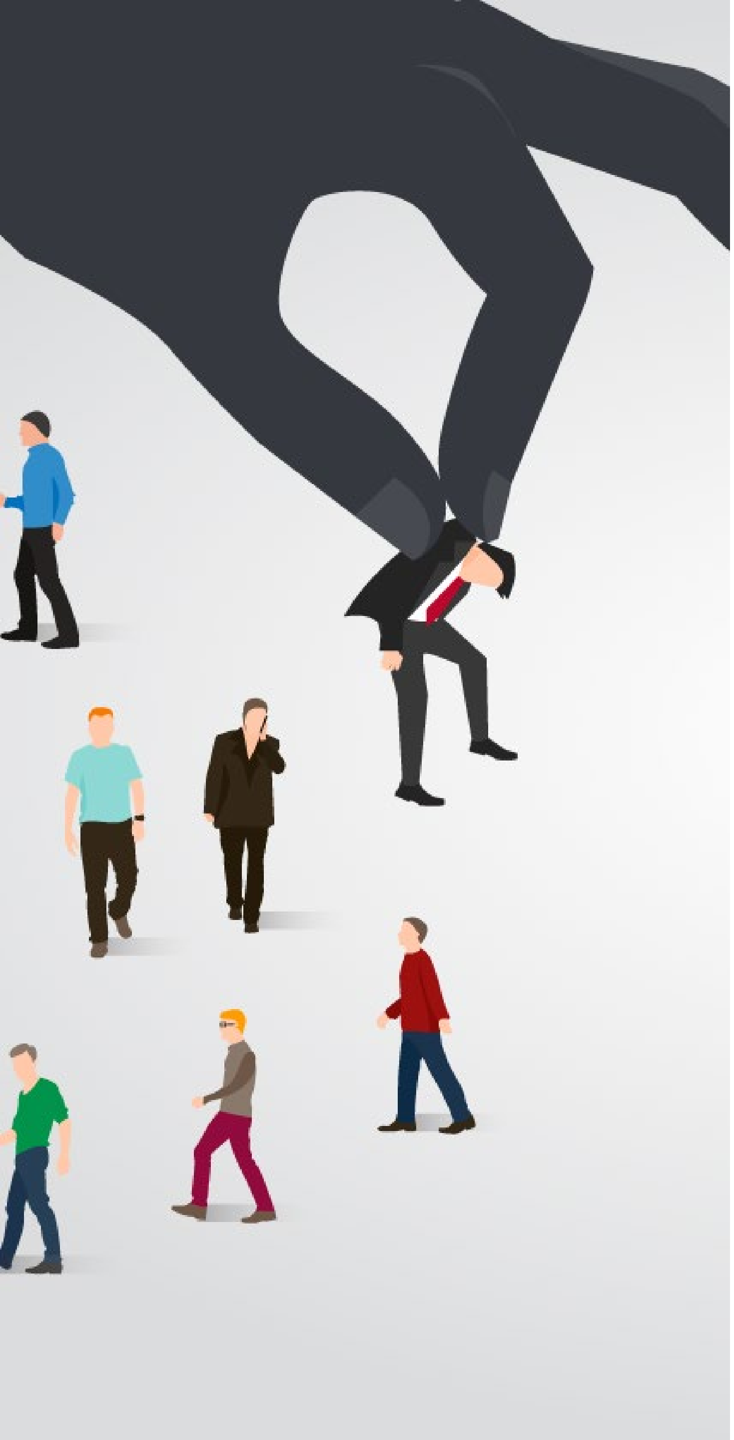
$$E = \sum_i [Y_i - f(w_i, X_i)]^2$$

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

Deep Learning / Deep Neural Networks



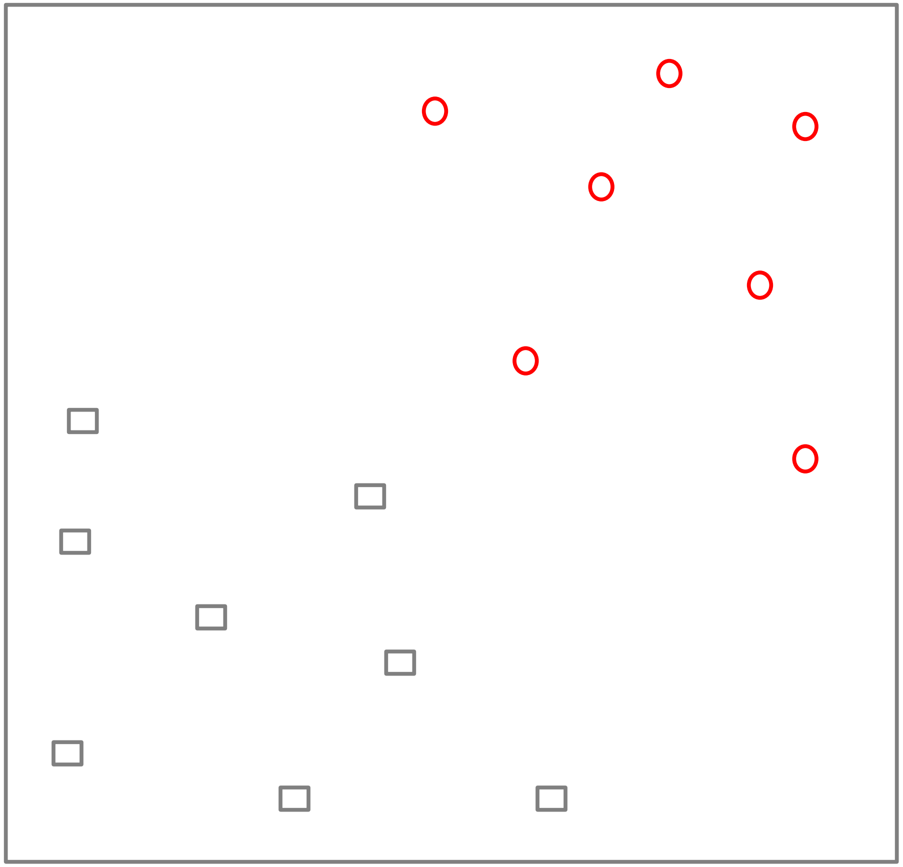
- Needs lots of data + computation (GPU)
- Applications: computer vision, speech recognition, natural language processing, audio recognition, machine translation, bioinformatics, …
- Tools: Keras, Tensorflow and many others.
- Related: Deep belief networks, recurrent neural networks (RNN), convolutional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



Topics

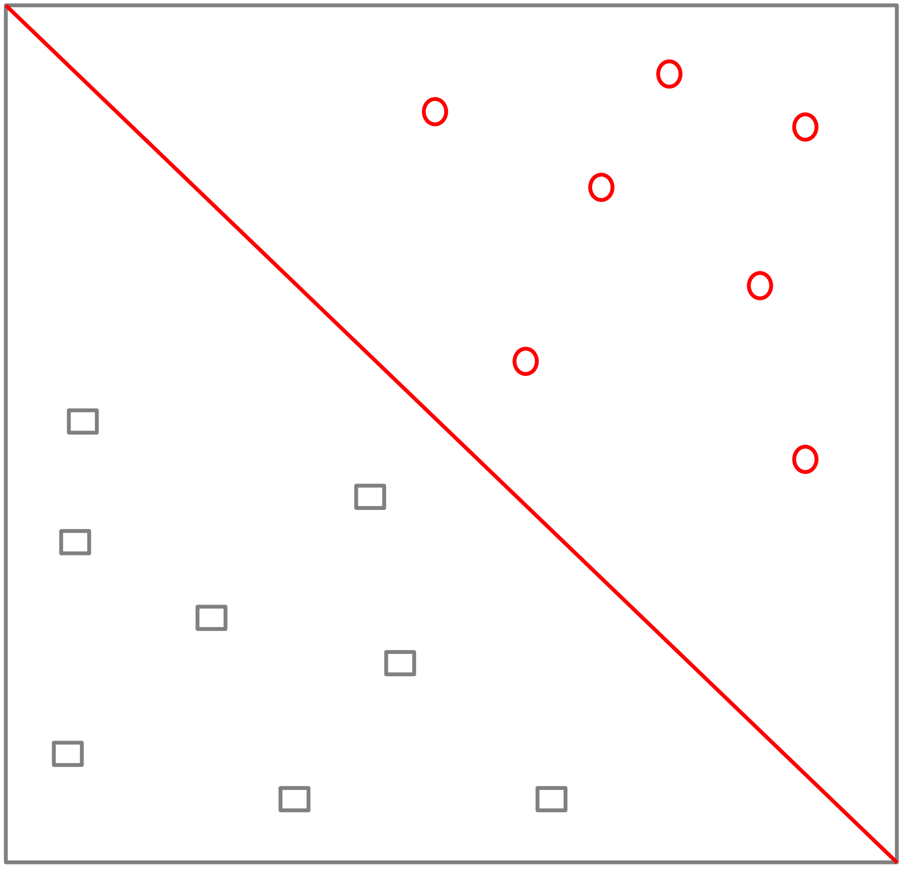
- Rule-Based Classifier
- Nearest Neighbor Classifier
- Naive Bayes Classifier
- Artificial Neural Networks
- **Support Vector Machines**
- Ensemble Methods

Support Vector Machines



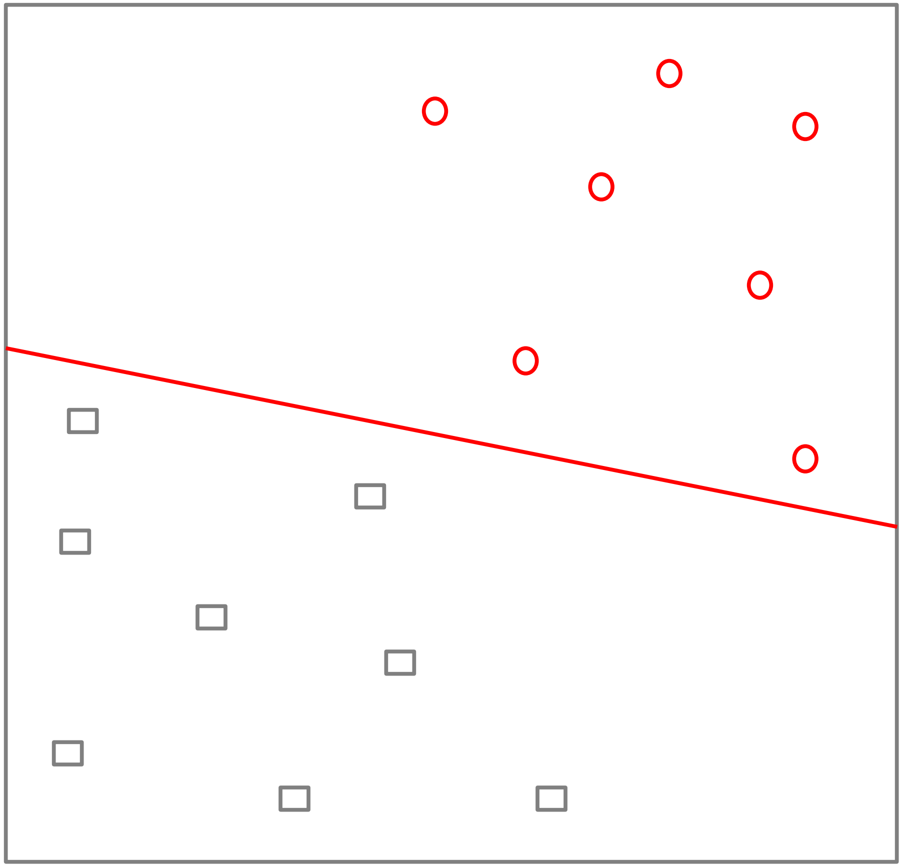
Find a linear hyperplane (decision boundary) that will separate the data

Support Vector Machines



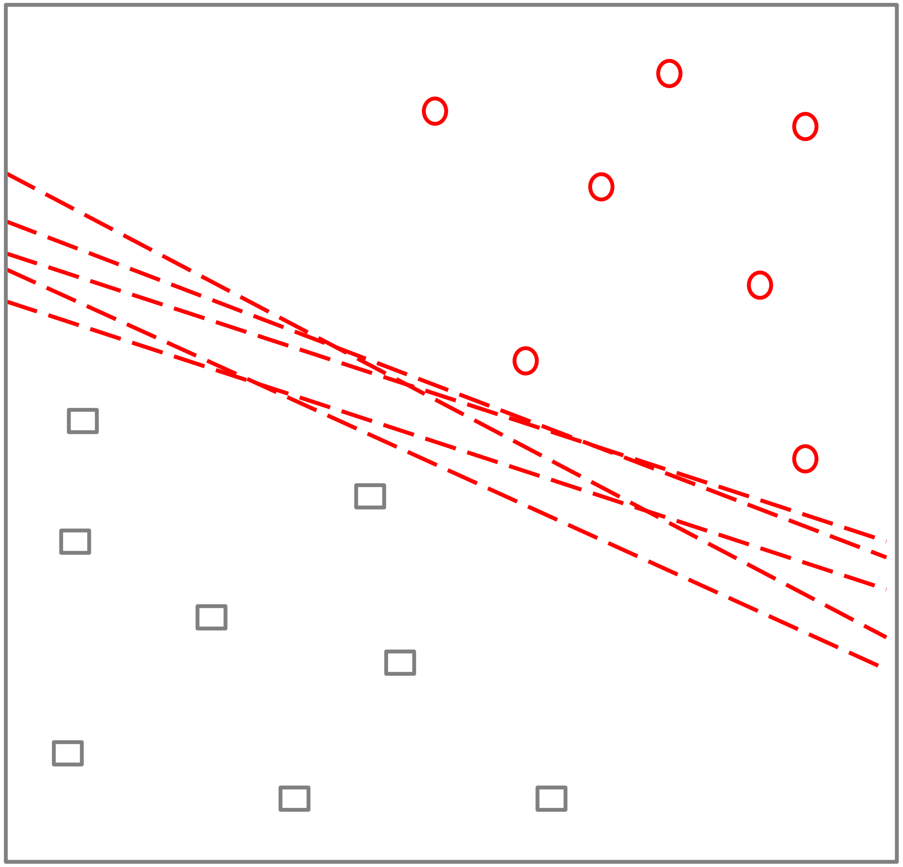
One Possible Solution

Support Vector Machines



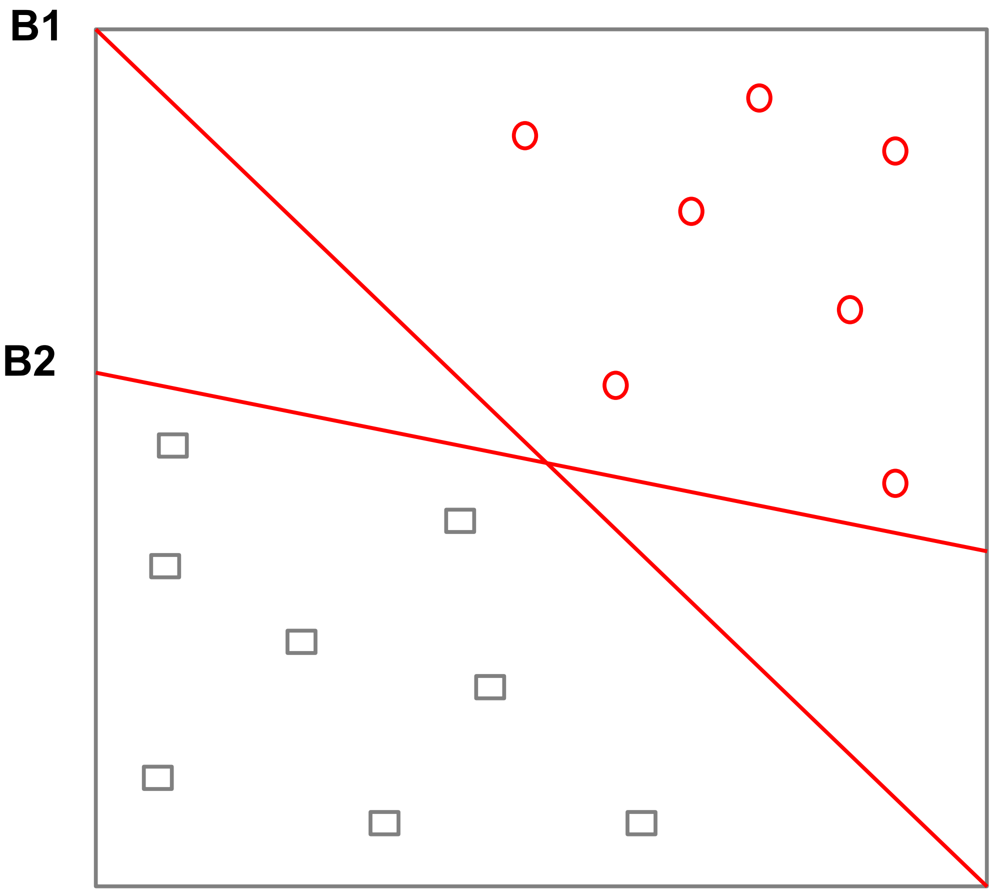
Another possible solution

Support Vector Machines



Other possible solutions

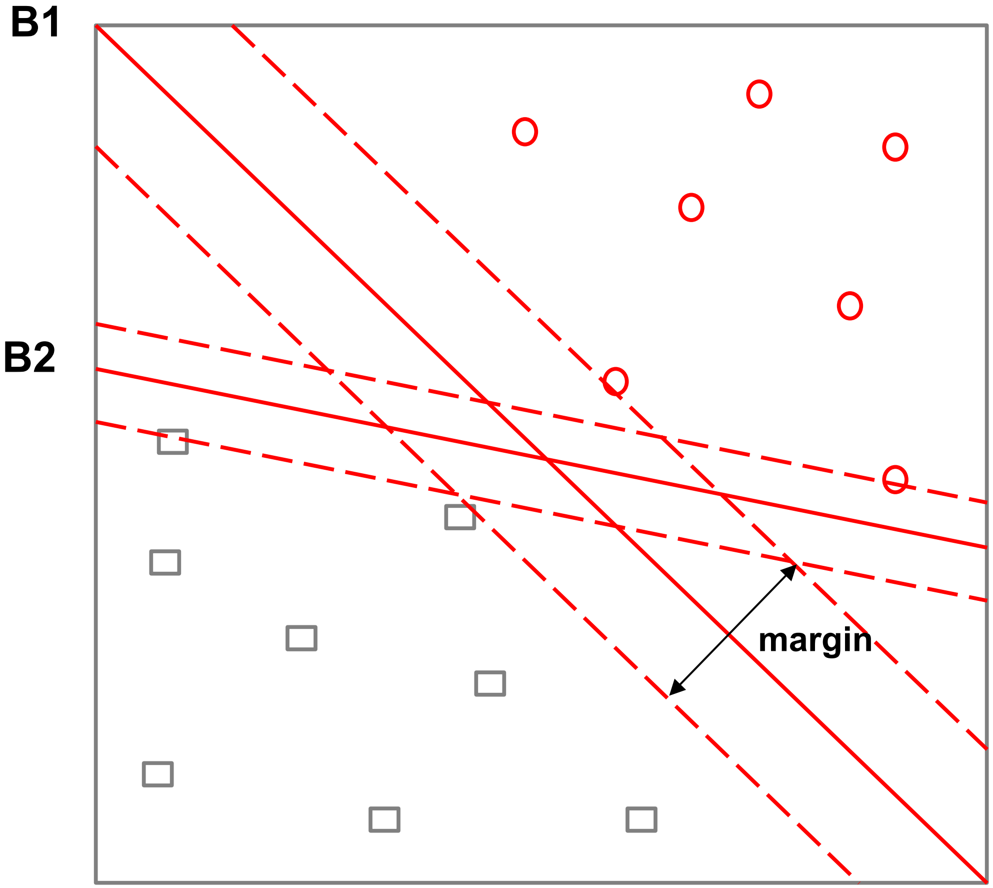
Support Vector Machines



Which one is better? B1 or B2?

How do you define better?

Support Vector Machines

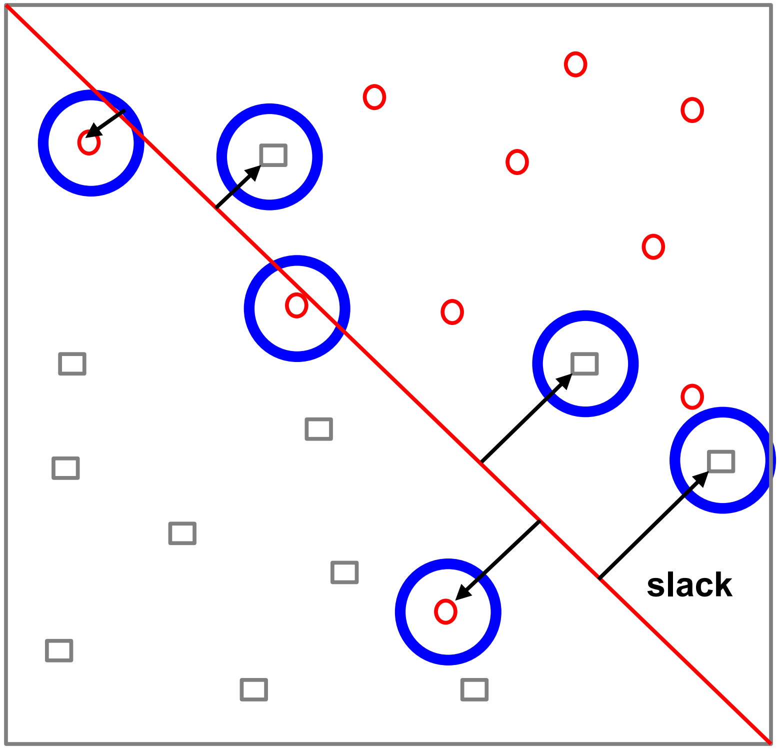


Find hyperplane maximizes the margin => B1 is better than B2

Larger margin = more robust = less expected generalization error

Support Vector Machines

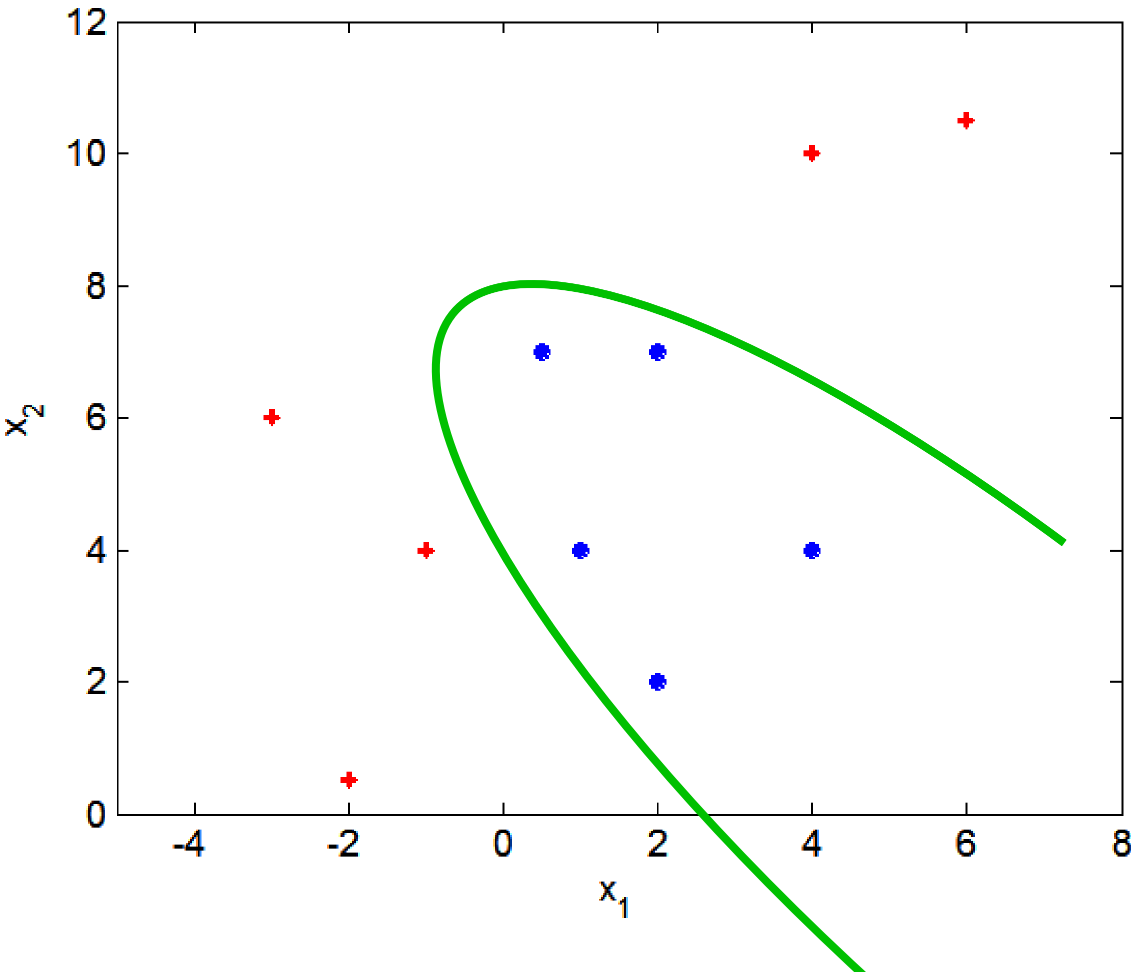
What if the problem is not linearly separable?



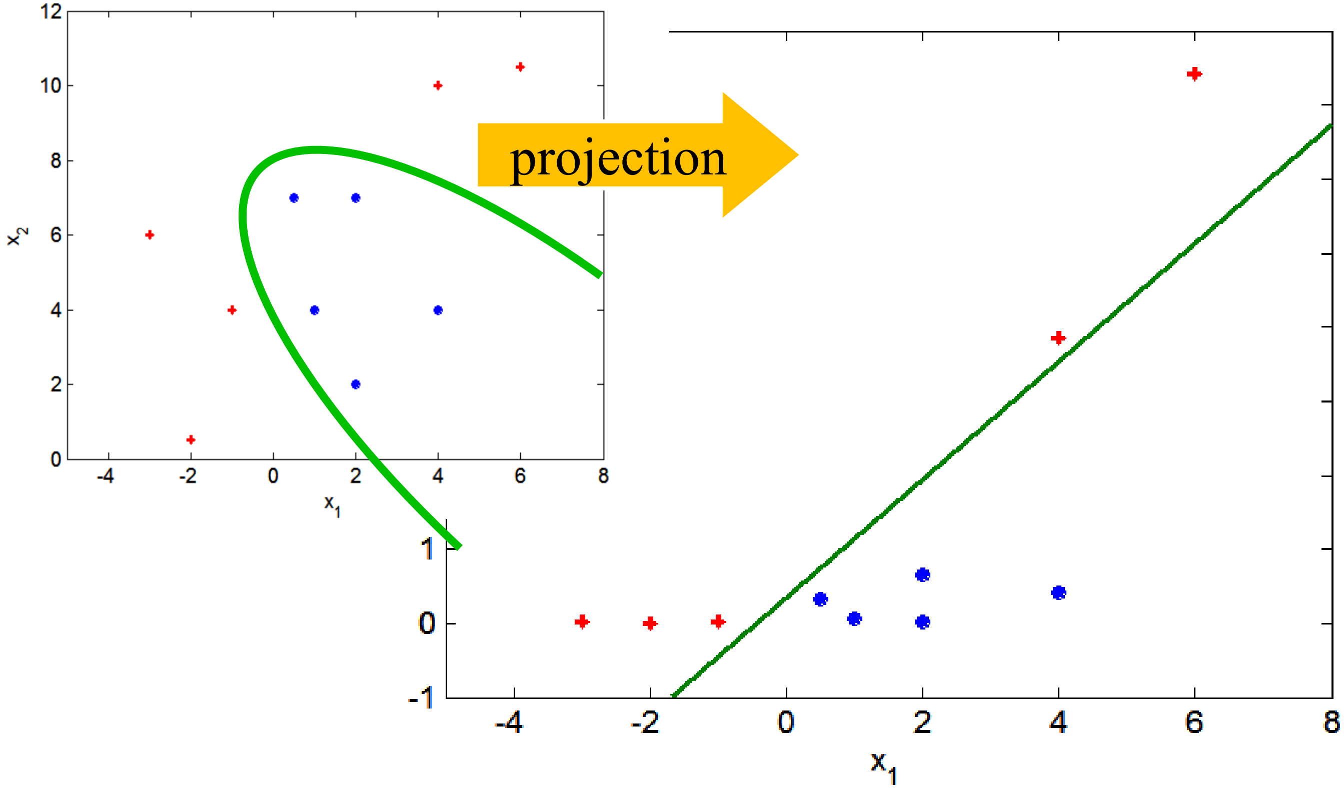
- Use slack variables to account for violations
- Use hyperplane that minimizes slack

Nonlinear Support Vector Machines

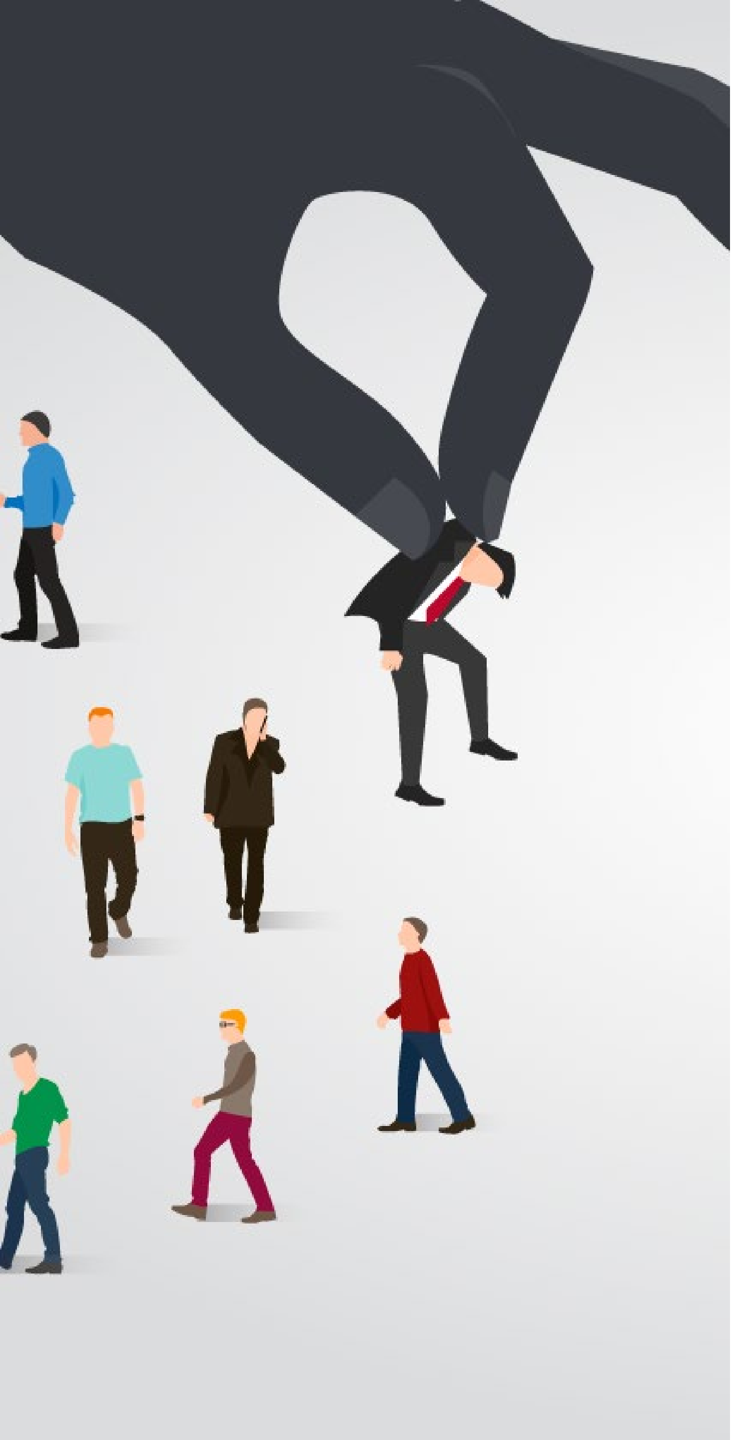
- What if decision boundary is not linear?



Nonlinear Support Vector Machines



- Project data into a higher dimensional space where the classes are linearly separable.
- Using the Kernel trick!



Topics

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

Ensemble Methods

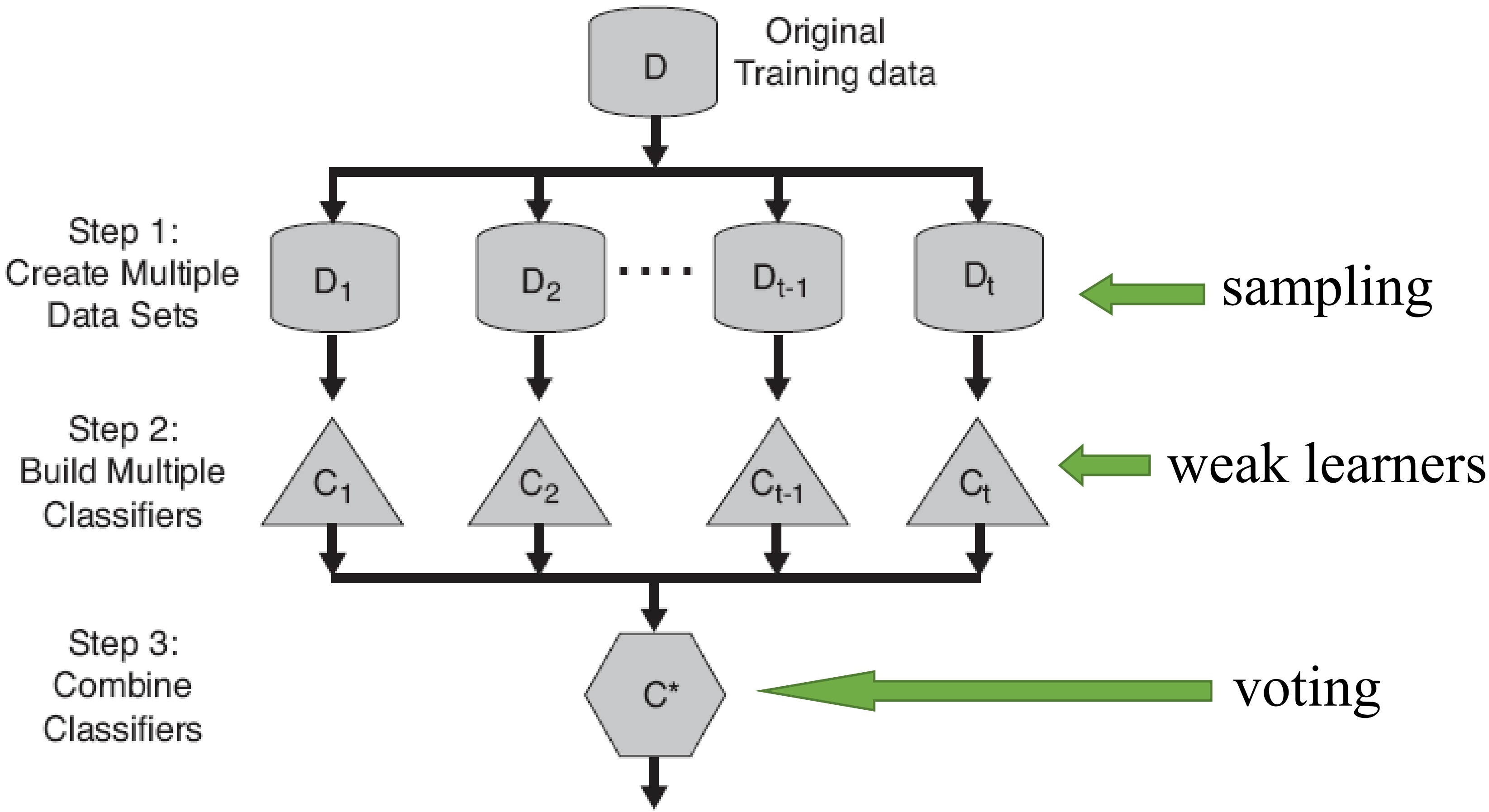
Method

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 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

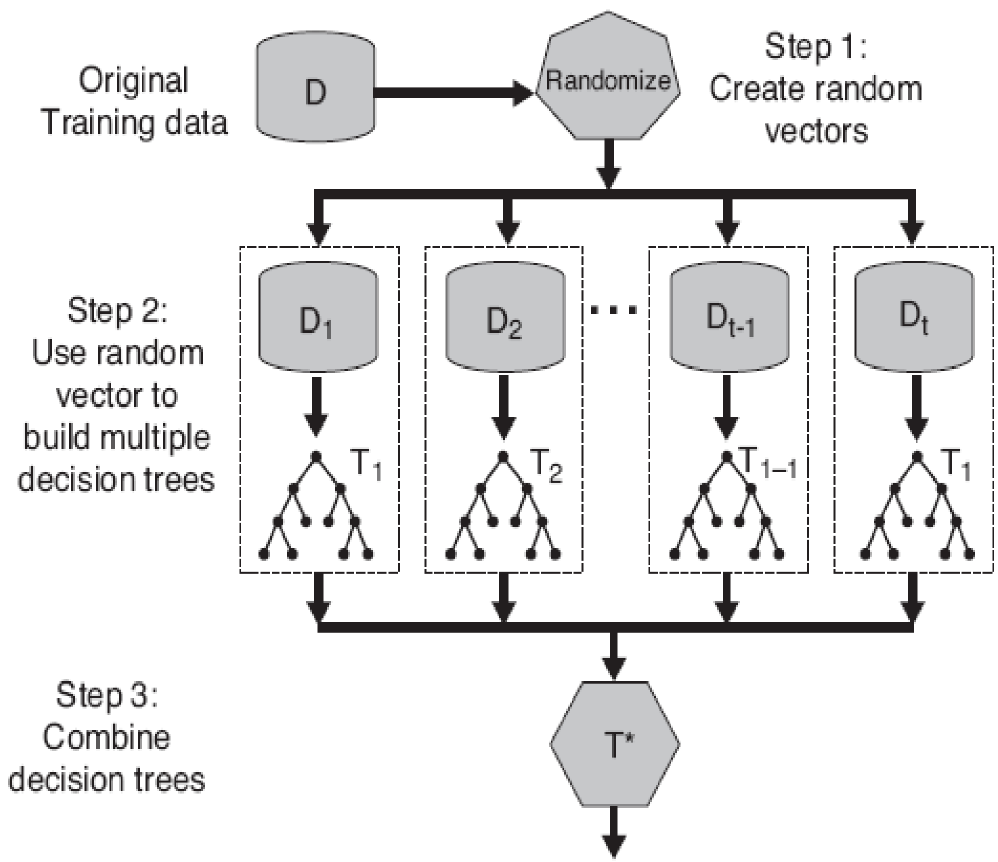
- Records that are incorrectly classified in one round will have their weights increased in the next

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

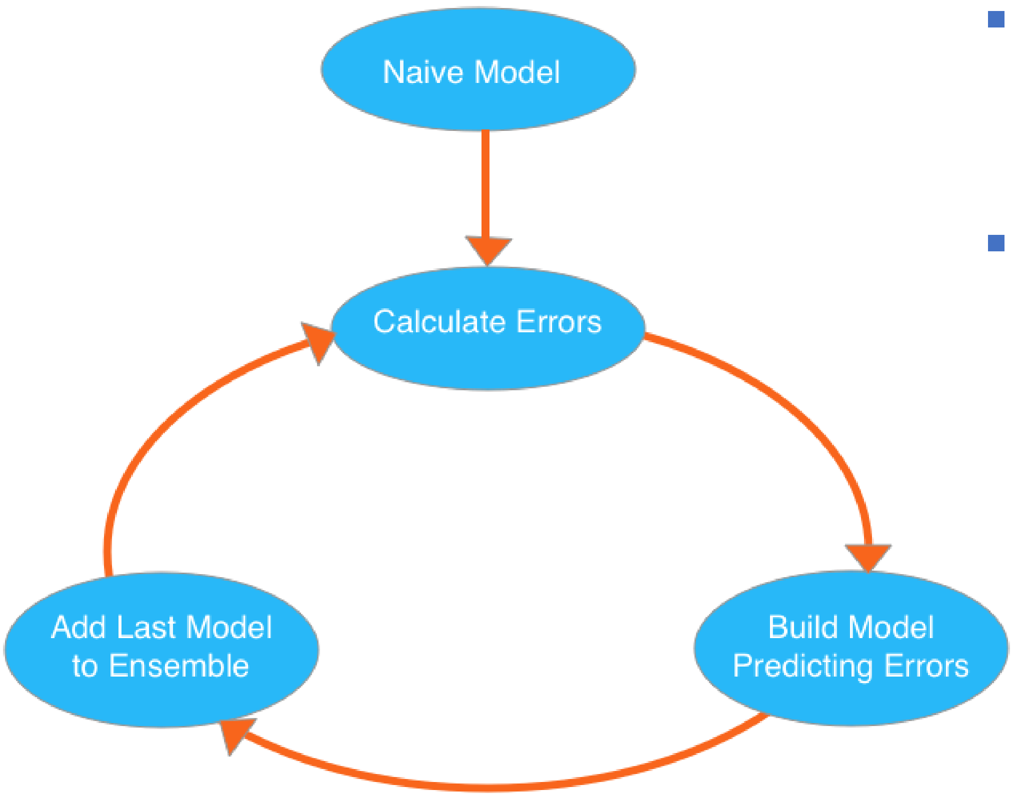
- **Popular algorithm**: AdaBoost (Adaptive Boosting) typically uses decision trees as the weak learner.

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.

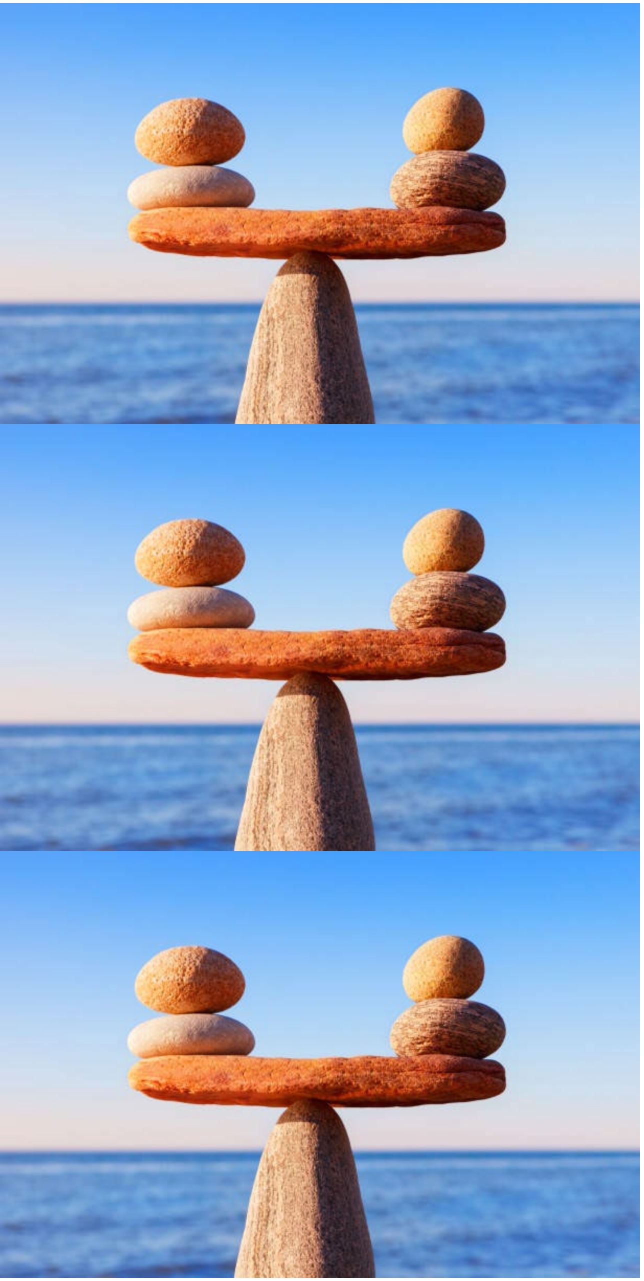
Gradient Boosted Decision Trees (XGBoost)



- **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



Topics

- Other Classification Methods
  - Rule-Based Classifier
  - Nearest Neighbor Classifier
  - Naive Bayes Classifier
  - Artificial Neural Networks
  - Support Vector Machines
  - Ensemble Methods
- **Class Imbalance Problem**

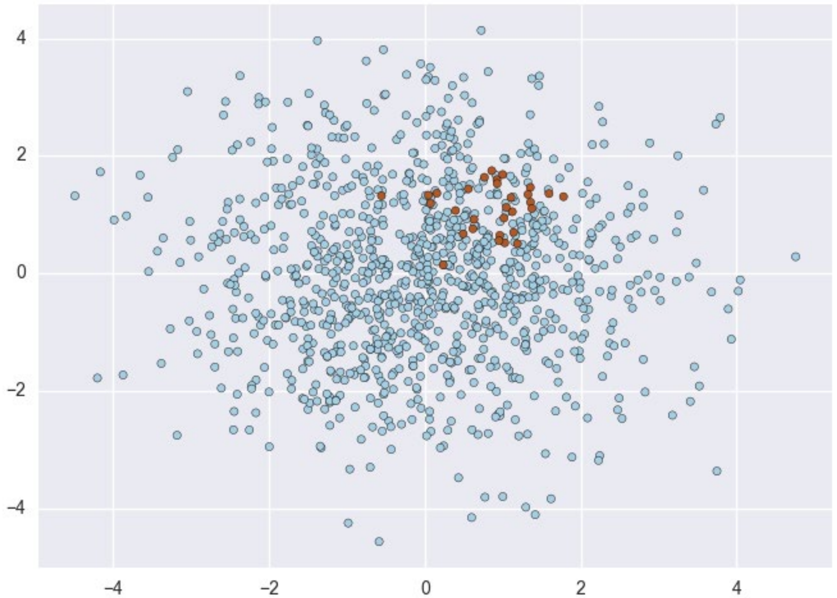
Class Imbalance Problem

Consider a 2-class problem

- Number of Class 0 examples = 9990
- Number of Class 1 examples = 10

A simple model:

- Always predict Class 0
- accuracy = 9990/10000 = 99.9 %
- error = 0.1%



Issues:

1. Evaluation: accuracy is misleading.
2. Learning: Most classifiers try to optimize accuracy/error. **These classifiers will not learn how to find examples of Class 1!**

Class Imbalance Problem: Evaluation

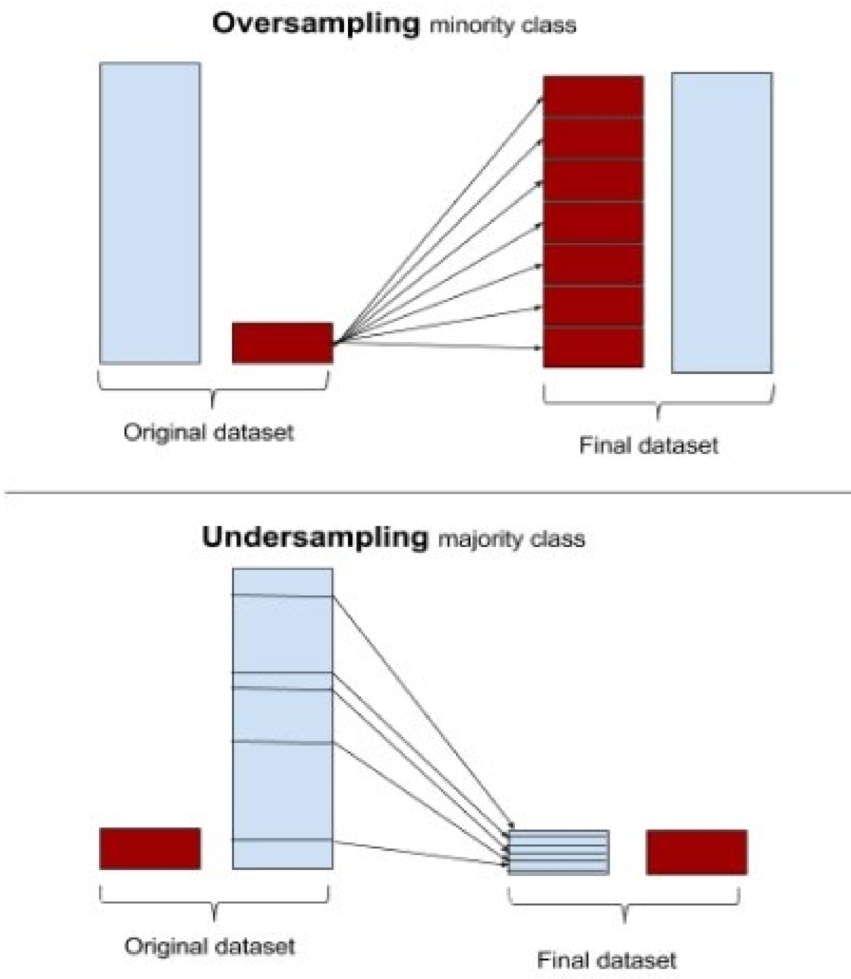
Do not use accuracy to evaluate for problems with strong class imbalance!

Use instead:

- ROC curves and AUC (area under the curve)
- Precision/Recall plots or the F1 Score
- Cohen's Kappa
- Misclassification cost

Class Imbalance Problem: Learning

- **Do nothing**. Sometimes you get lucky!
- **Balance the data set**: Down-sample the majority class and/or up-sample the minority class (use sampling with replacement). Synthesize new examples with SMOTE.
  This will artificially increase the error for a mistake in the minority class.
- Use **algorithms** that can deal with class imbalance (see next slide).
- Throw away minority examples and switch to an **anomaly detection** framework.



Class Imbalance Problem: Learning

Algorithms that can deal with class imbalance:

- Use a classifier that **predict a probability** and lower the decision threshold (from the default of .5). We can estimate probabilities for decision trees using the positive and negative training examples in each leaf node.
- Use a **cost-sensitive classifier** that considers a cost matrix (not too many are available).
- Use boosting techniques like **AdaBoost**.

Conclusion

- There are many ways to implement the classification function.
- Each of them has a different inductive bias and often benefits from specifically created feature (e.g., interaction effects inn linear models).
- Accuracy is a big problematic for **imbalanced data sets**. Rebalancing the data may be necessity.