classifier II

Arend Hintze

how good is my classifier?

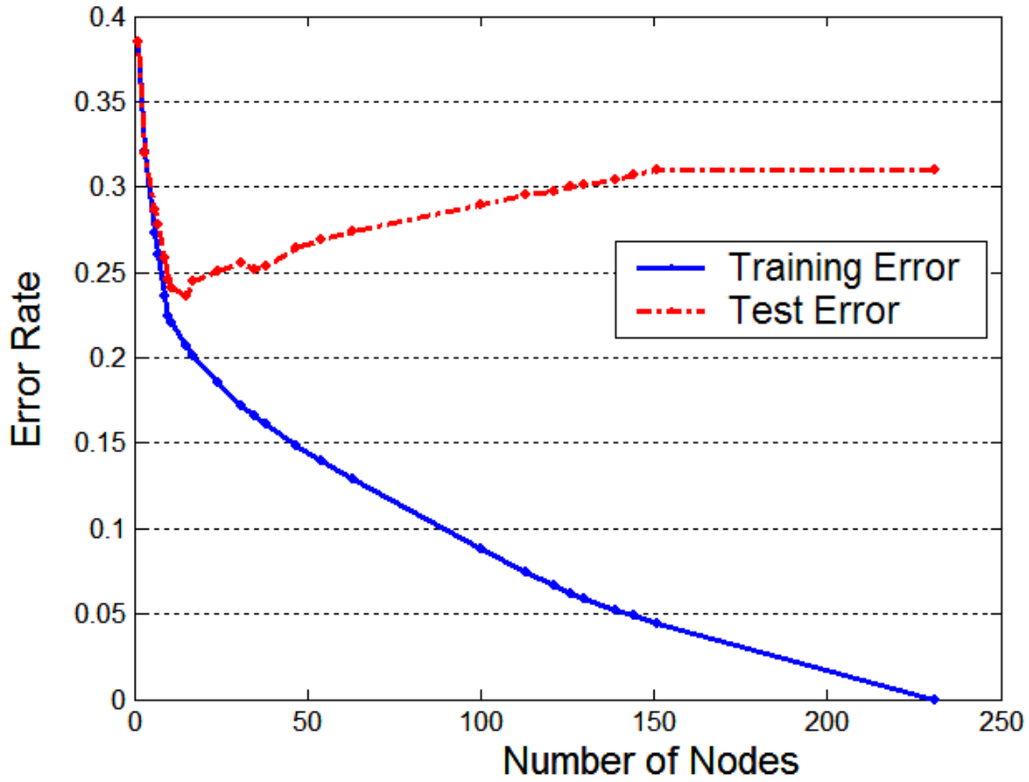
- confusion matrix:

		PREDICTED CLASS	
		Class = +	Class = -
ACTUAL CLASS	Class = +	a	b
	Class = -	c	d

$$\text{Accuracy} = \frac{\#\text{Correct Predictions}}{\#\text{Instances Predicted}} = \frac{\boldsymbol{a+d}}{\boldsymbol{a+b+c+d}}$$

$$\text{Error rate} = 1 - \text{Accuracy} = \frac{\boldsymbol{b+c}}{\boldsymbol{a+b+c+d}}$$

over and under fitting



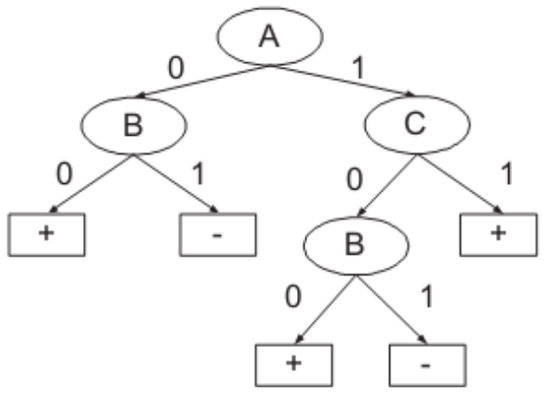
Underfitting: when model is too simple, both training and test errors are large

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

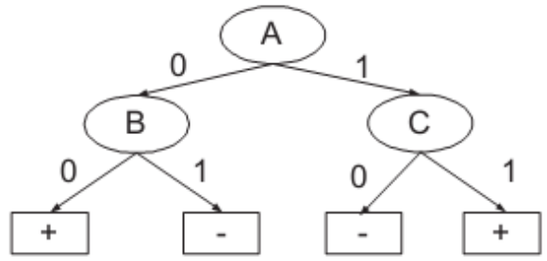
what if I only have trainings data?

- Divide training data into two parts:
  - Training
  - Test / Validation set
- Drawback: Less data available for training

example:



(a) Decision Tree A



(b) Decision Tree B

#	A	B	C	Class
1	0	0	0	+
2	0	0	0	+
3	0	1	0	-
4	0	1	1	-
5	1	0	0	+
6	1	0	1	+
7	1	0	1	-
8	1	1	0	-
9	1	1	0	-
10	1	1	1	+

(c) Training data

#	A	B	C	Class
11	0	0	0	+
12	0	0	1	-
13	0	1	0	-
14	0	1	1	-
15	0	1	1	+
16	1	0	0	-
17	1	0	0	-
18	1	0	1	-
19	1	1	0	-
20	1	1	1	+

(c) Validation data

Training errors:

Tree A: 10%

Tree B: 20%

Validation errors:

Tree A: 50%

Tree B: 30%

So, tree B is preferred over A

Rule-Based Classifier

- use a set if “if … then …” rules (called the rule set)
- Data tuple (X, Y) where X = attribute Y = class

example:

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hiber-nates	Class Label
human	warm-blooded	hair	yes	no	no	yes	no	Mammals
python	cold-blooded	scales	no	no	no	no	yes	Reptiles
salmon	cold-blooded	scales	no	yes	no	no	no	Fishes
whale	warm-blooded	hair	yes	yes	no	no	no	Mammals
frog	cold-blooded	none	no	semi	no	yes	yes	Amphibians
komodo dragon	cold-blooded	scales	no	no	no	yes	no	Reptiles
bat	warm-blooded	hair	yes	no	yes	yes	yes	Mammals
pigeon	warm-blooded	feathers	no	no	yes	yes	no	Birds
cat	warm-blooded	fur	yes	no	no	yes	no	Mammals
guppy	cold-blooded	scales	yes	yes	no	no	no	Fishes
alligator	cold-blooded	scales	no	semi	no	yes	no	Reptiles
penguin	warm-blooded	feathers	no	semi	no	yes	no	Birds
porcupine	warm-blooded	quills	yes	no	no	yes	yes	Mammals
eel	cold-blooded	scales	no	yes	no	no	no	Fishes
salamander	cold-blooded	none	no	semi	no	yes	yes	Amphibians

Rule set:

r_1: (Gives Birth = no) $\wedge$ (Aerial Creature = yes) $\longrightarrow$ Birds
r_2: (Gives Birth = no) $\wedge$ (Aquatic Creature = yes) $\longrightarrow$ Fishes
r_3: (Gives Birth = yes) $\wedge$ (Body Temperature = warm-blooded) $\longrightarrow$ Mammals
r_4: (Gives Birth = no) $\wedge$ (Aerial Creature = no) $\longrightarrow$ Reptiles
r_5: (Aquatic Creature = semi) $\longrightarrow$ Amphibians

application:

- a test instance is predicted based on the rule it triggers:

r_1: (Gives Birth = no) $\wedge$ (Aerial Creature = yes) $\longrightarrow$ Birds
r_2: (Gives Birth = no) $\wedge$ (Aquatic Creature = yes) $\longrightarrow$ Fishes
r_3: (Gives Birth = yes) $\wedge$ (Body Temperature = warm-blooded) $\longrightarrow$ Mammals
r_4: (Gives Birth = no) $\wedge$ (Aerial Creature = no) $\longrightarrow$ Reptiles
r_5: (Aquatic Creature = semi) $\longrightarrow$ Amphibians

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates
hawk	warm-blooded	feather	no	no	yes	yes	no
grizzly bear	warm-blooded	fur	yes	no	no	yes	yes

Hawk: triggers rule r1 -> Bird
Grizzly: triggers rule r3 -> mammal

and now?

r_1: (Gives Birth = no) $\wedge$ (Aerial Creature = yes) $\longrightarrow$ Birds
r_2: (Gives Birth = no) $\wedge$ (Aquatic Creature = yes) $\longrightarrow$ Fishes
r_3: (Gives Birth = yes) $\wedge$ (Body Temperature = warm-blooded) $\longrightarrow$ Mammals
r_4: (Gives Birth = no) $\wedge$ (Aerial Creature = no) $\longrightarrow$ Reptiles
r_5: (Aquatic Creature = semi) $\longrightarrow$ Amphibians

Name	Body Temperature	Skin Cover	Gives Birth	Aquatic Creature	Aerial Creature	Has Legs	Hibernates
lemur	warm-blooded	fur	yes	no	no	yes	yes
turtle	cold-blooded	scales	no	semi	no	yes	no
dogfish shark	cold-blooded	scales	yes	yes	no	no	no

Lemur: triggers rule r3 -> mammal
Turtle: triggers rule r3 and r4 ???
Dogfish: none ???

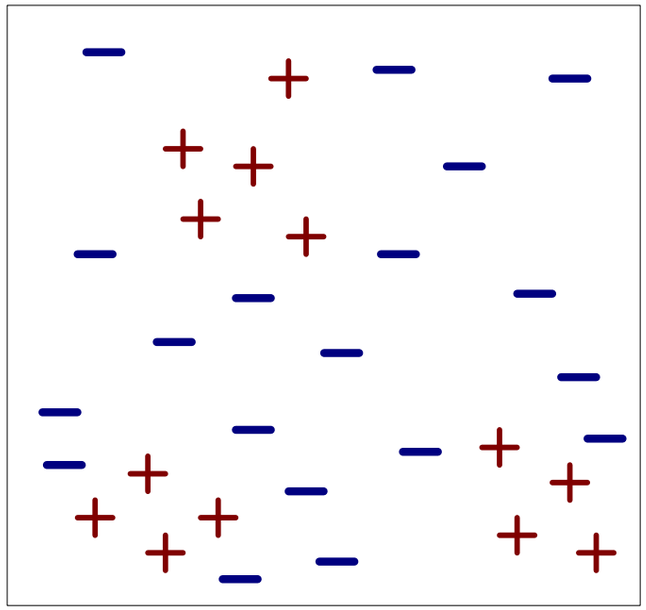
Ordered rule set and default!

- Rules are ordered by priority
- A default rule has to be set

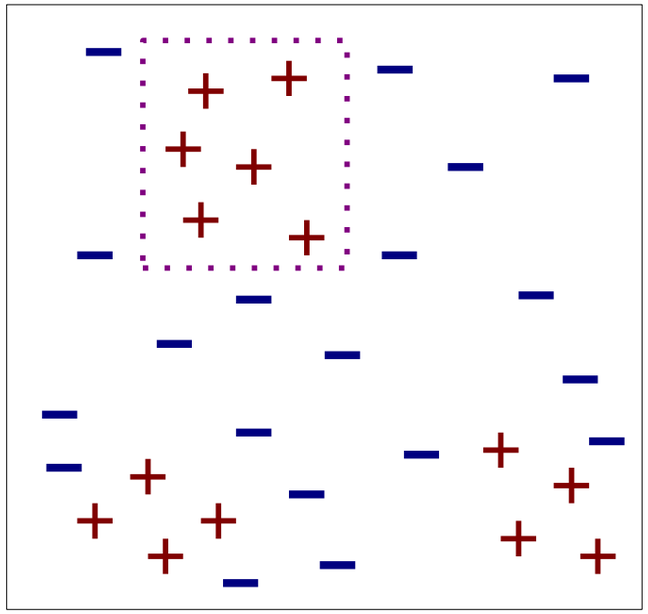
Sequential Covering Algorithm

- Many rule-based algorithms use it
- Algorithm:
  - initialize rule set R to be empty
  - repeat until no rule can be added:
    - grow a single rule r
    - eliminate all training instances covered by rule r
    - add rule r to the current rule set R

example



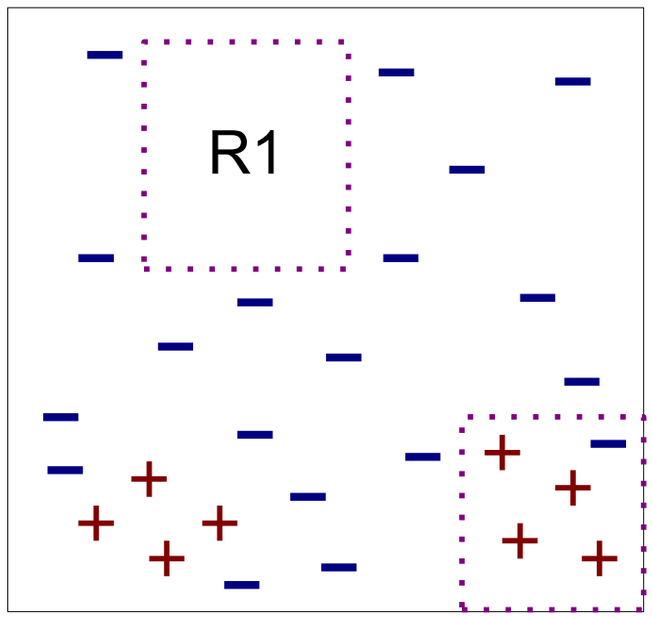
(i) Original Data



(ii) Step 1

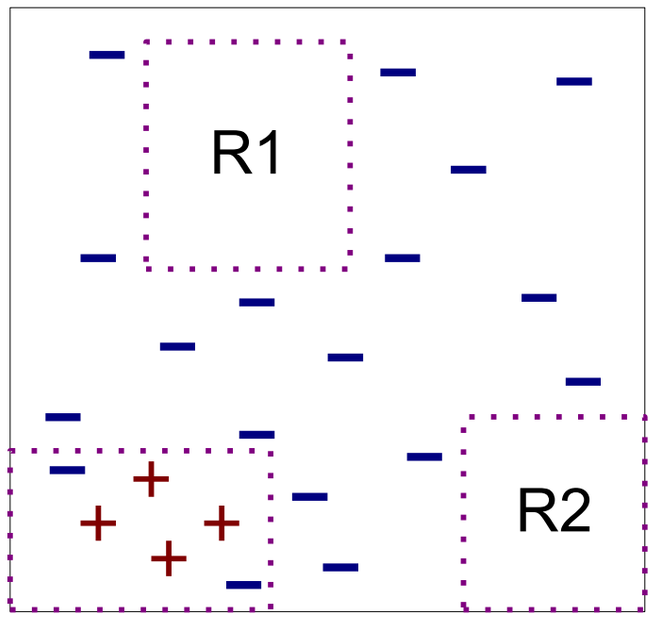
Extract a rule with high accuracy

example



(iii) Step 2

Eliminate instances covered by the rule



(iv) Step 3

Extract the next high accuracy rule

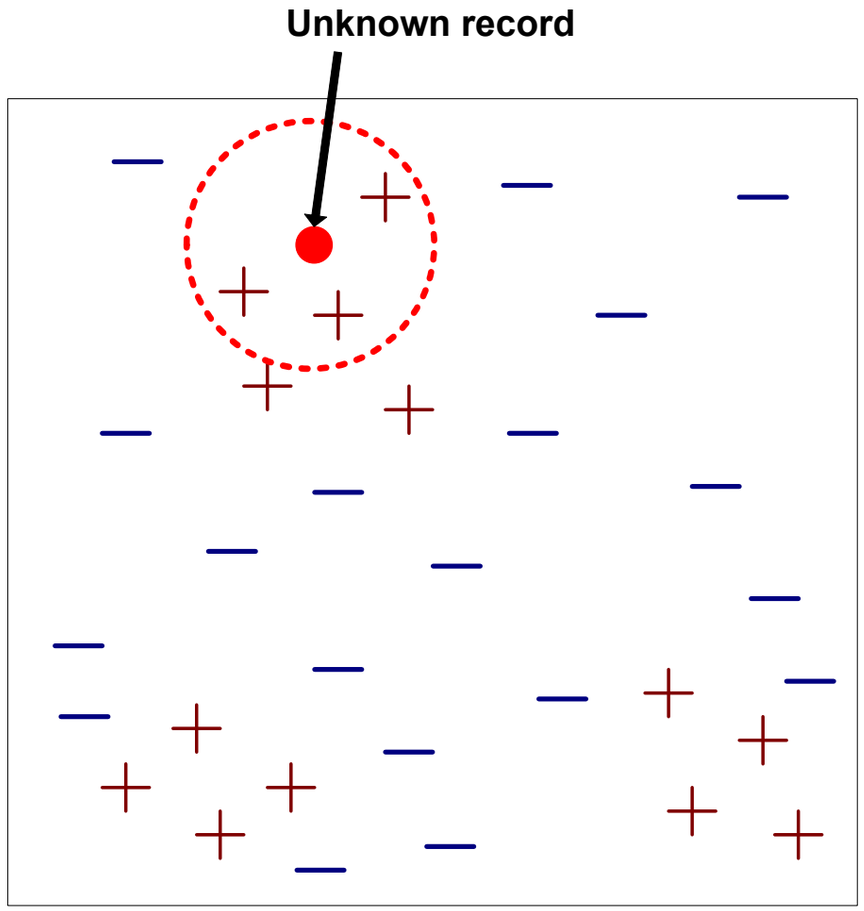
Question: Which class (rule) should we extract first?

Class-based ordering

- learn the rules for the smallest class first, followed by the next highest, and so forth

- default is the largest class remaining

Nearest-Neighbor Classifier



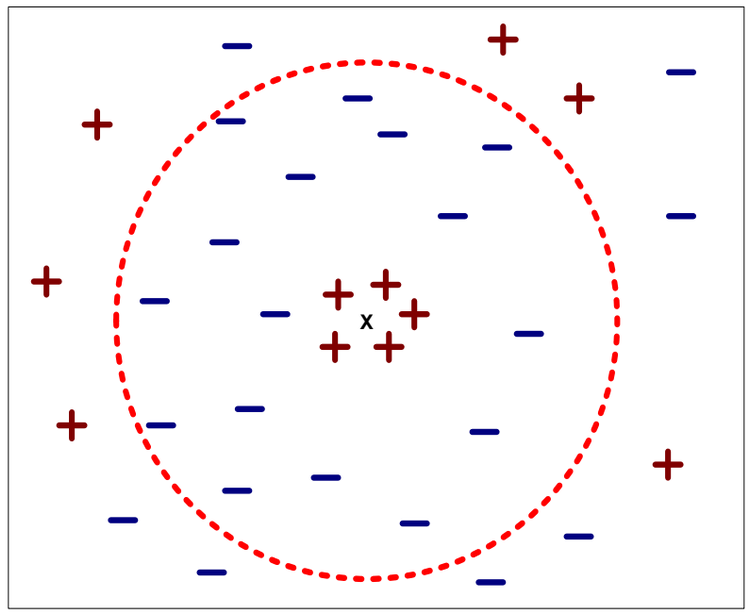
- Given a test instance:
  - Compute its distance to all the training instances
  - Identify its *k* nearest neighbors
  - Use class labels of k nearest neighbors to predict the class label of test instance (e.g., by taking majority vote)

Nearest-Neighbor Classifier

- Requires a distance/similarity measure
    - euclidian, Mahalanobis, cosine similarity …
- specify parameter k (number of nearest neighbors to consider)

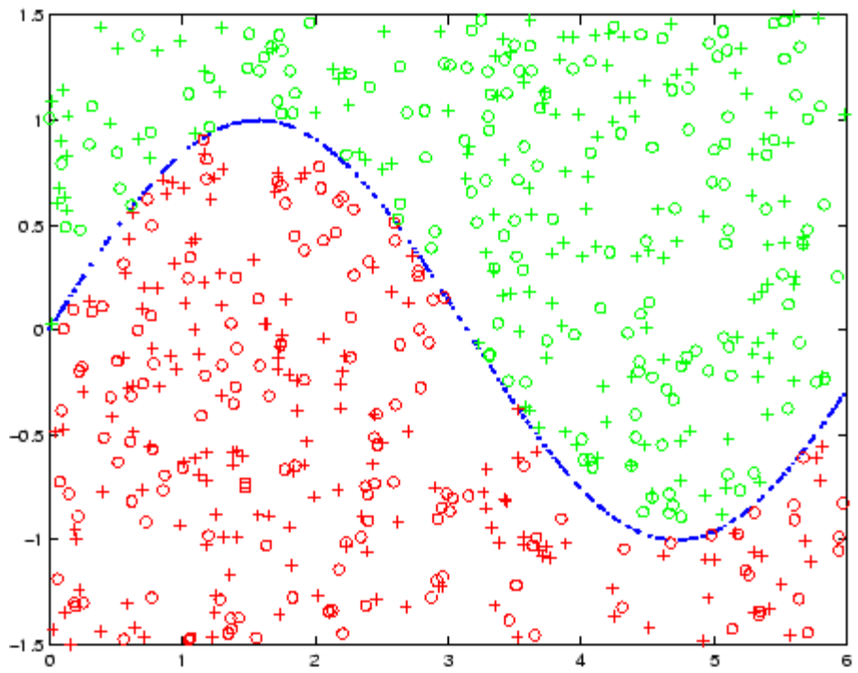
Nearest-Neighbor Classifier choosing value for k

- k too small, sensitive to noise points
- k too large, neighborhood includes points from other classes

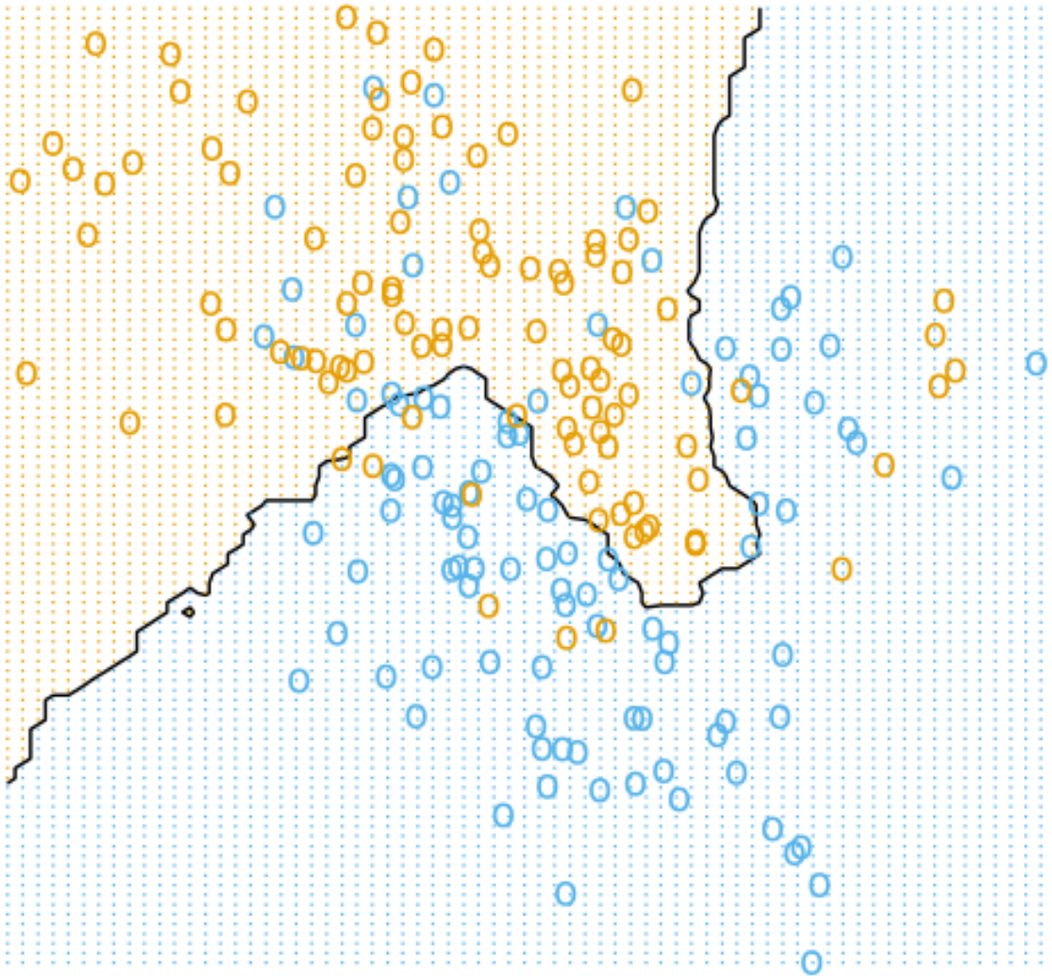


Decision Boundary Classifier

- Define a boundary
- boundary divides space into classes



NN meets DB

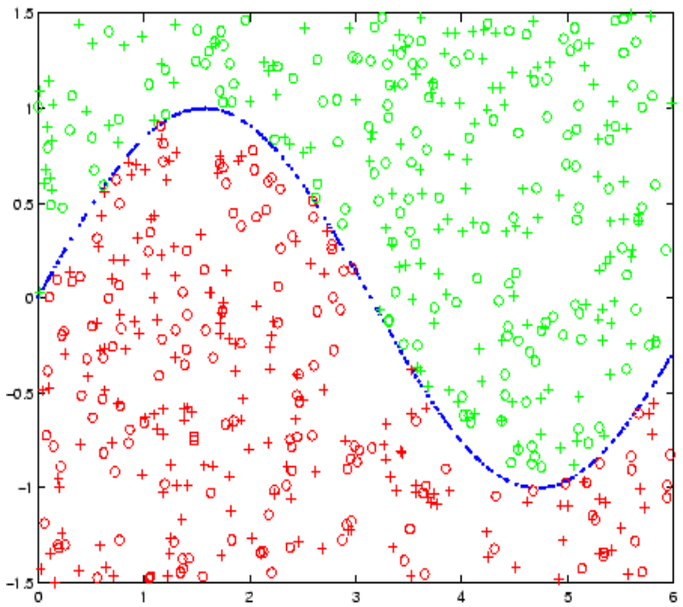


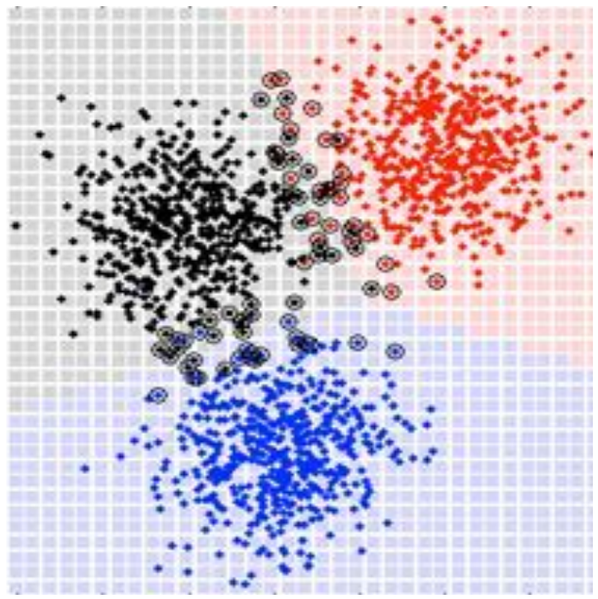
Decision boundary using k=15

classification as a learning problem

- Classification can be viewed as the problem of learning f(x) that defines the decision boundary

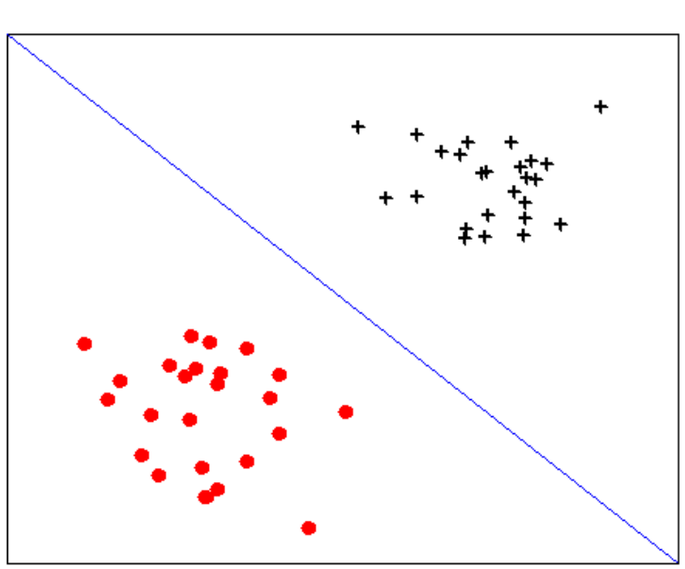
f(x)





Linear Classifier

- Construct a linear decision boundary to separate instances from different classes



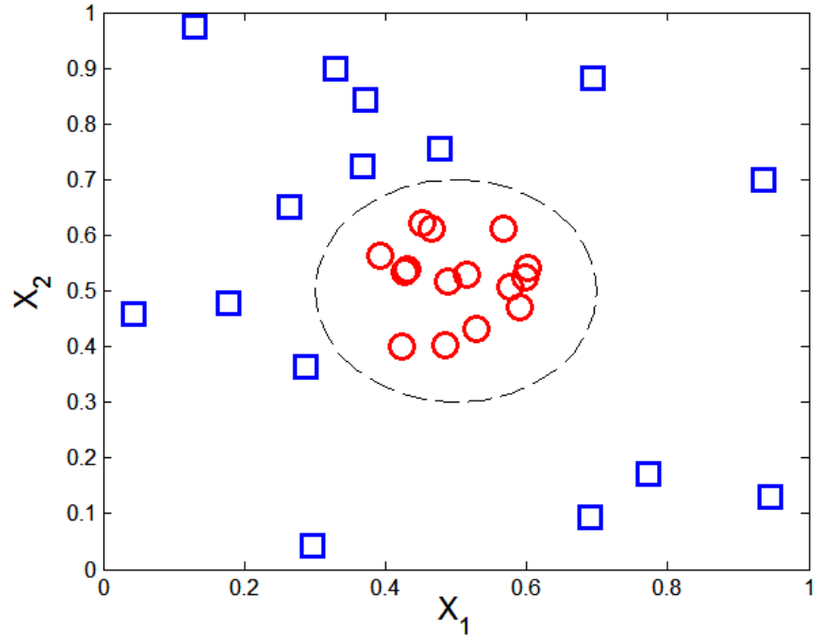
$$\text{Linear Model}: \quad f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_o$$

$$\text{Predicted class}: \quad \hat{y} = \begin{cases} +1 & \text{if } f(\boldsymbol{x}) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

- perceptron, linear SVM, logistic regression, …

Nonlinear Classifier

- same idea as linear, but uses a nonlinear model



Nonlinear Model (example):

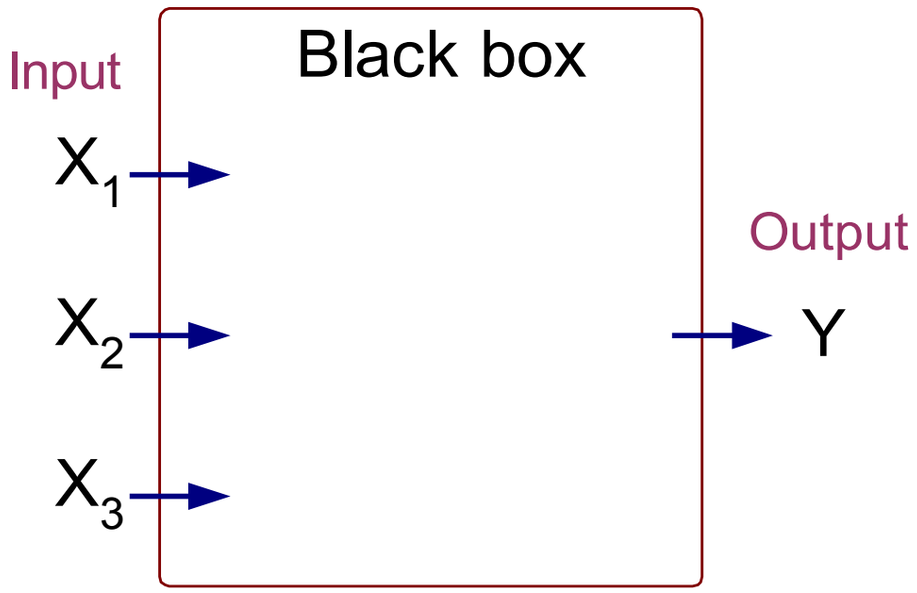
$$f(x) = (w_0 + w_1 x_1 + w_2 x_2)^2 - w_3 x_1 x_2 - w_4 (x_1 + x_2) - w_5$$

$$\text{Predicted class: } \hat{y} = \begin{cases} +1 & \text{if } f(x) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

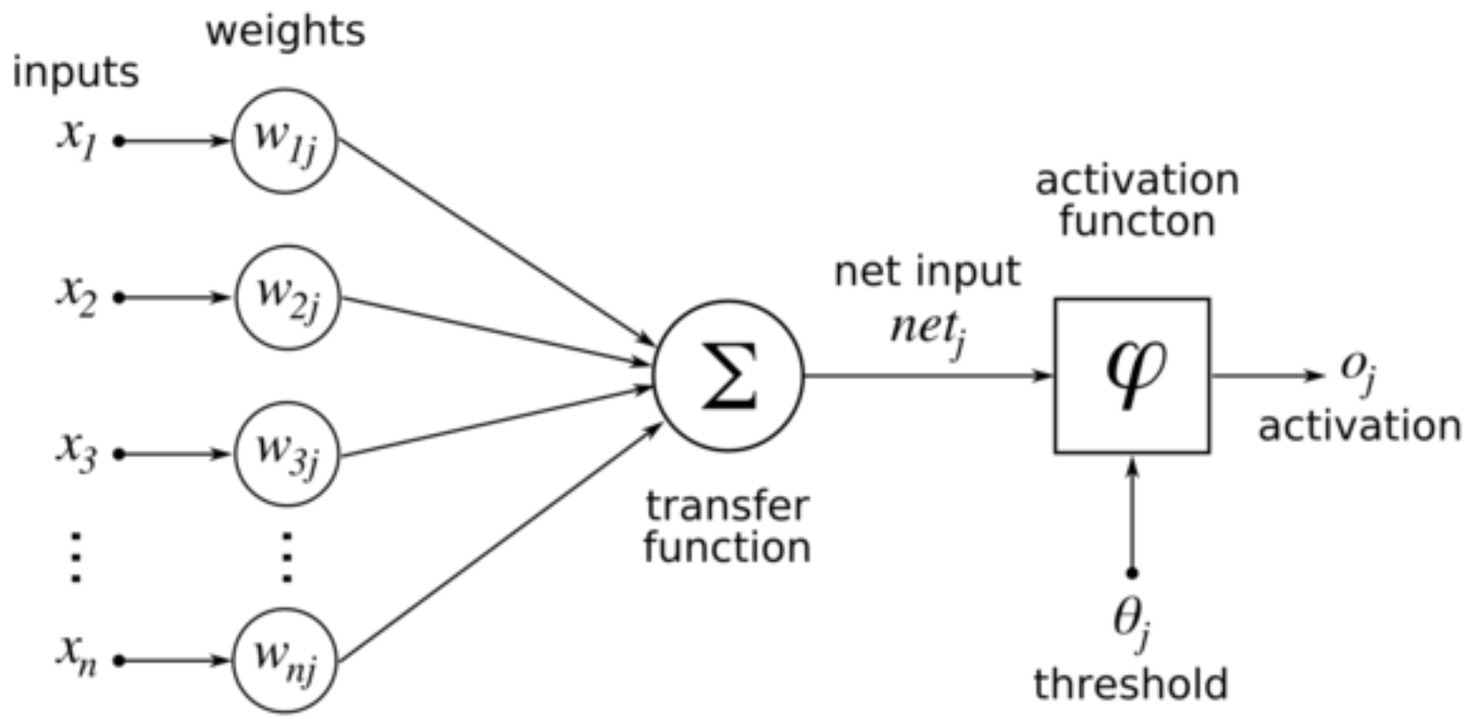
- nonlinear SVM, artificial neural network, …

Artificial Neural Network

X_1	X_2	X_3	Y
1	0	0	-1
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	-1
0	1	0	-1
0	1	1	1
0	0	0	-1

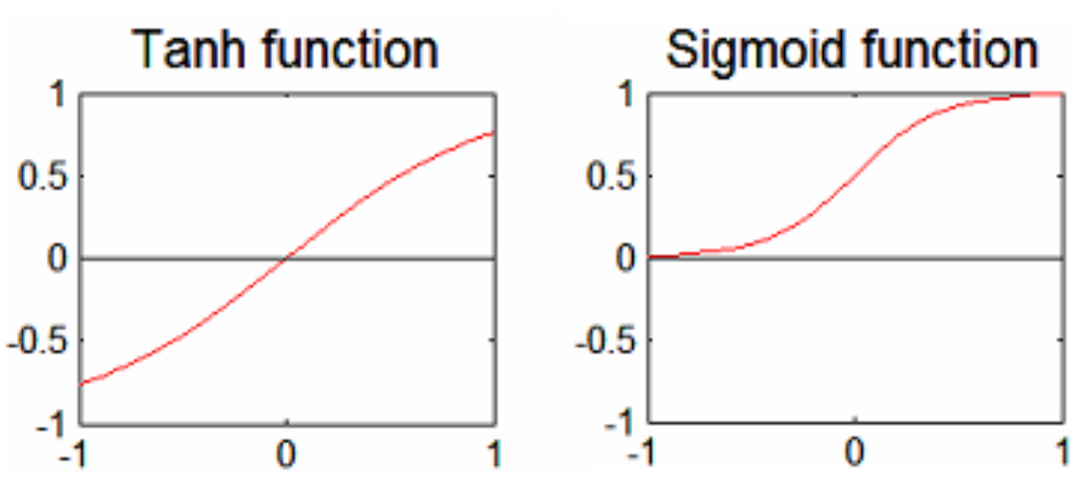


inside the black box

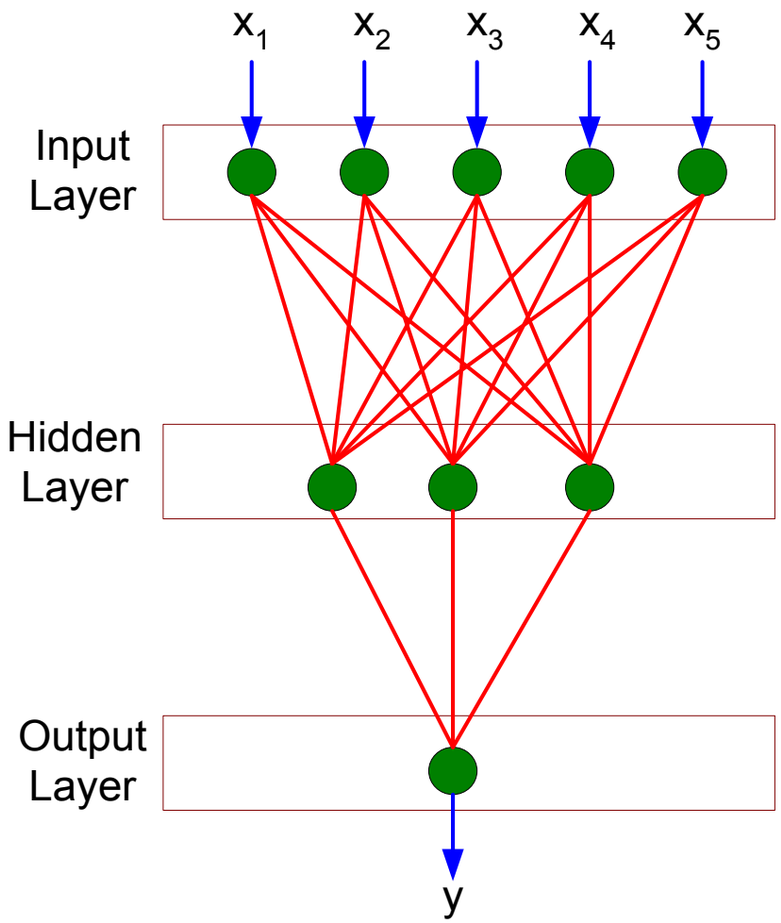


typical function

- transfer:
    - sum, product
- activation:
    - sigmoid, tanh, sin, cos, threshold, …



layers in ANN



where do the weights come from?

- hill climbing, simulated annealing, Manhattan, evolution

- Baum-Welch, Back propagation, forward - backward algorithm

Choosing the Right Classifier

- Sophisticated classifiers like multilayer ANN or nonlinear SVM: great accuracy but non descriptive (black box)

- Simple ones like Decision Tree or Rule based: mediocre accuracy but descriptive model (unless overfit…)

Choosing the Right Classifier

- Sophisticated classifiers like multilayer ANN or nonlinear SVM: assume continuous and numerical data -> transform categories into 0.0 1.0 or -1.0 1.0

- Simple ones like Decision Tree or Rule based: can take everything, finds bins/thresholds automatically (depends on algorithm)

limits and contraints

- imbalanced data (99% in one class 1% in the other)
- noise, uncertainty, discrepancy between learning and testing data
- high dimensionality
- time and storage complexity
- parameter optimization, local maxima, complexity
- sometimes they find simple heuristics (are they true?)