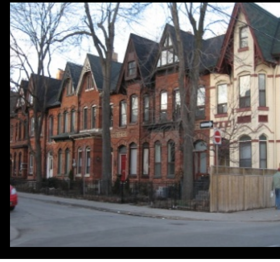



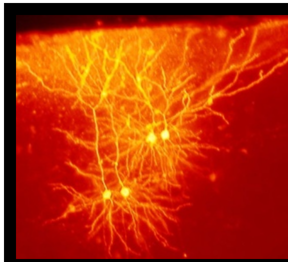
“Classifiers”

Under the guidance of

Prof. Pushpak Bhattacharyya
pushpakbh@gmail.com
IIT Bombay

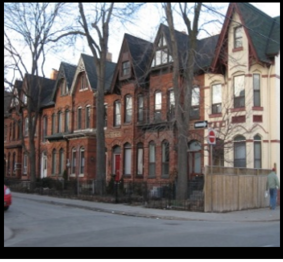
R & D project by

Aditya M Joshi
adityaj@cse.iitb.ac.in
IIT Bombay

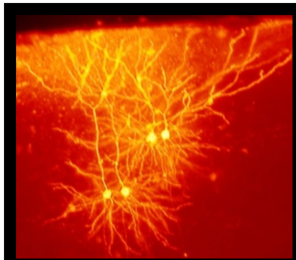




Overview

Introduction to Classification





What is classification?

A machine learning task that deals with identifying the class to which an instance belongs

A classifier performs classification

(Age, Marital status,
Ngrams)

Health status, Salary)

Attributes

(a1, a2,… an)

Classifier

Discrete-valued

Category of document?
{Politics, Movies, Biology}

Class label

Issue Loan? {Yes, No}

Steer? {Left, Straight, Right}

Classification learning

Training phase → **Testing phase**

Learning the classifier from the available data

'Training set'

(Labeled)

Testing how well the classifier performs

'Testing set'

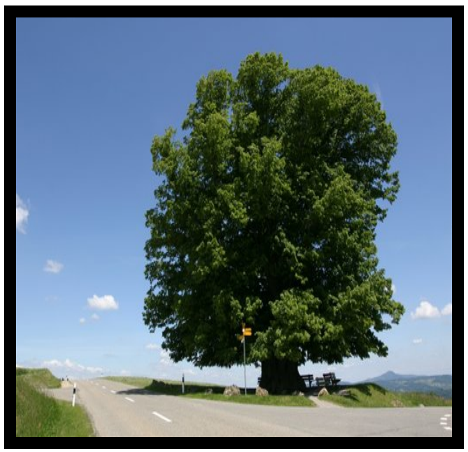
Generating datasets

- Methods:
  - Holdout ($2/3^{rd}$ training, $1/3^{rd}$ testing)
  - Cross validation (n – fold)
    - Divide into n parts
    - Train on (n-1), test on last
    - Repeat for different combinations
  - Bootstrapping
    - Select random samples to form the training set

Evaluating classifiers

- Outcome:
  - Accuracy
  - Confusion matrix
  - If cost-sensitive, the expected cost of classification (attribute test cost + misclassification cost)

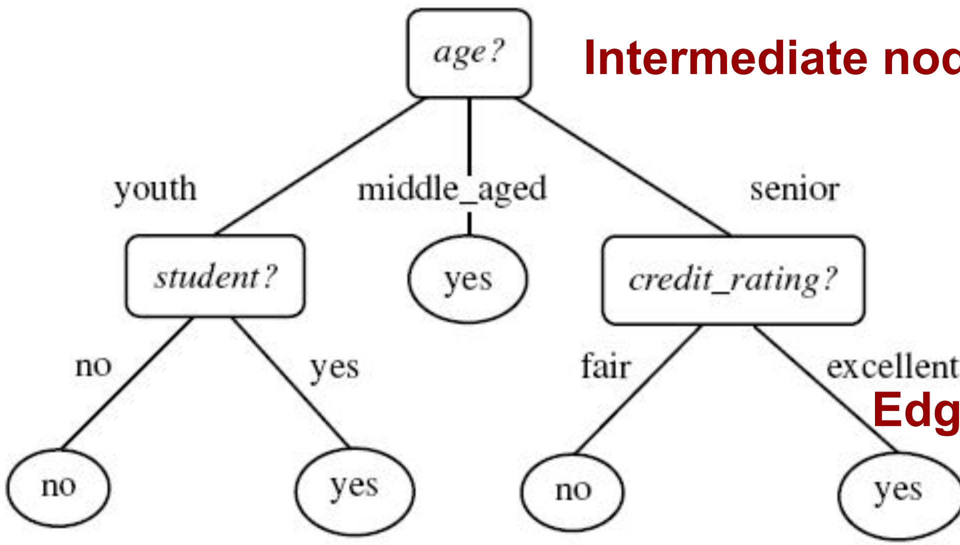
  etc.



Decision Trees

Example tree

Intermediate nodes : Attributes



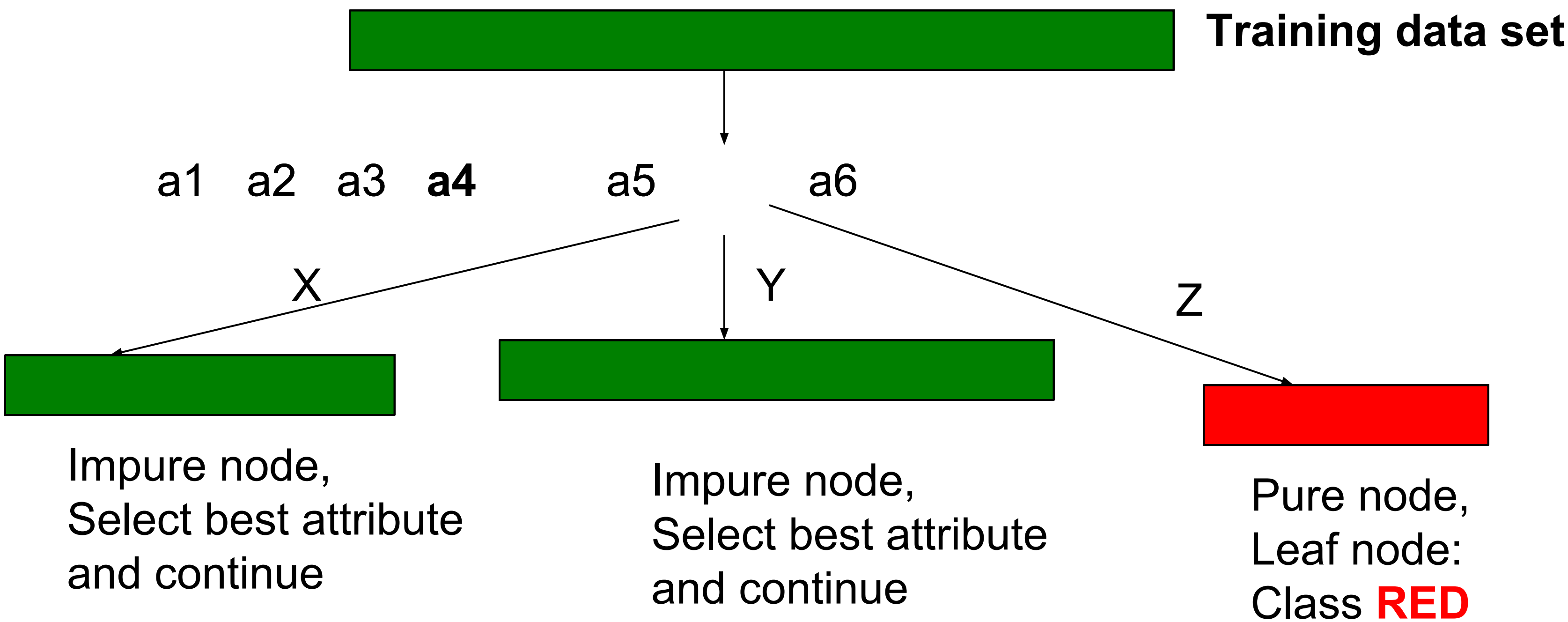
Edges : Attribute value tests

Leaf nodes : Class predictions

Example algorithms: ID3, C4.5, SPRINT, CART

Diagram from Han-Kamber

Decision Tree schematic



Decision Tree Issues

How to determine the attribute for split?

Alternatives:

1. Information Gain

Gain (A, S) = Entropy (S) – **Σ** ((Sj/S)*Entropy(Sj))

Other options:

Gain ratio, etc.

Lazy learners

Lazy learners

- ‘**Lazy**’: Do not create a model of the training instances in advance

- When an instance arrives for testing, runs the algorithm to get the class prediction

- Example, K – nearest neighbour classifier

  (K – NN classifier)

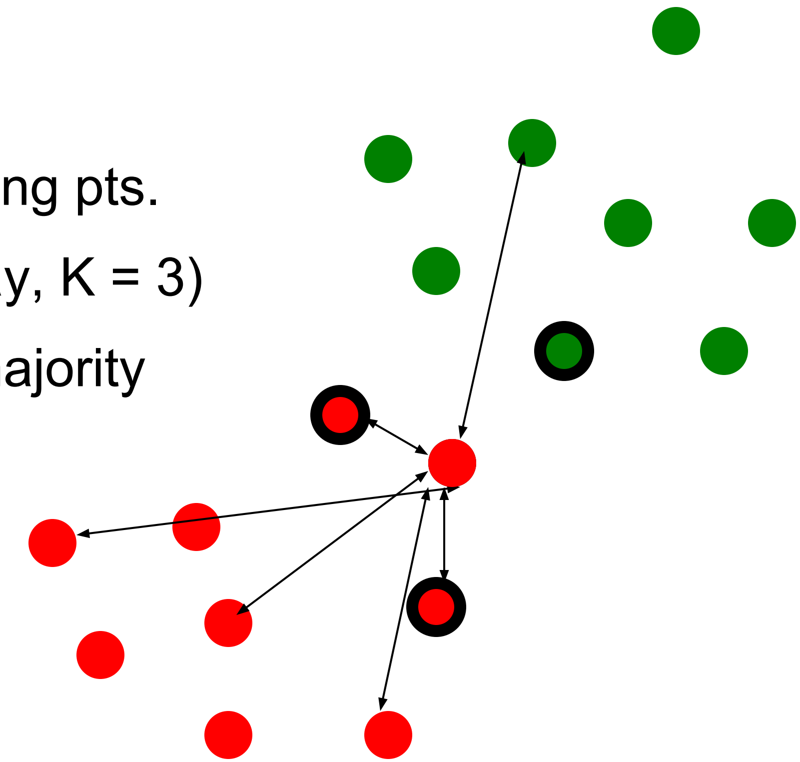
  “One is known by the company

  one keeps”

K-NN classifier schematic

For a test instance,

1) Calculate distances from training pts.
2) Find K-nearest neighbours (say, K = 3)
3) Assign class label based on majority



$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}.$$

$$v' = \frac{v - min_A}{max_A - min_A},$$

K-NN classifier Issues

How to determine distances between values of categorical attributes?

Alternatives:

1. Boolean distance (1 if same, 0 if different)

2. Differential grading (e.g. weather – ‘drizzling’ and ‘rainy’ are closer than ‘rainy’ and ‘sunny’)

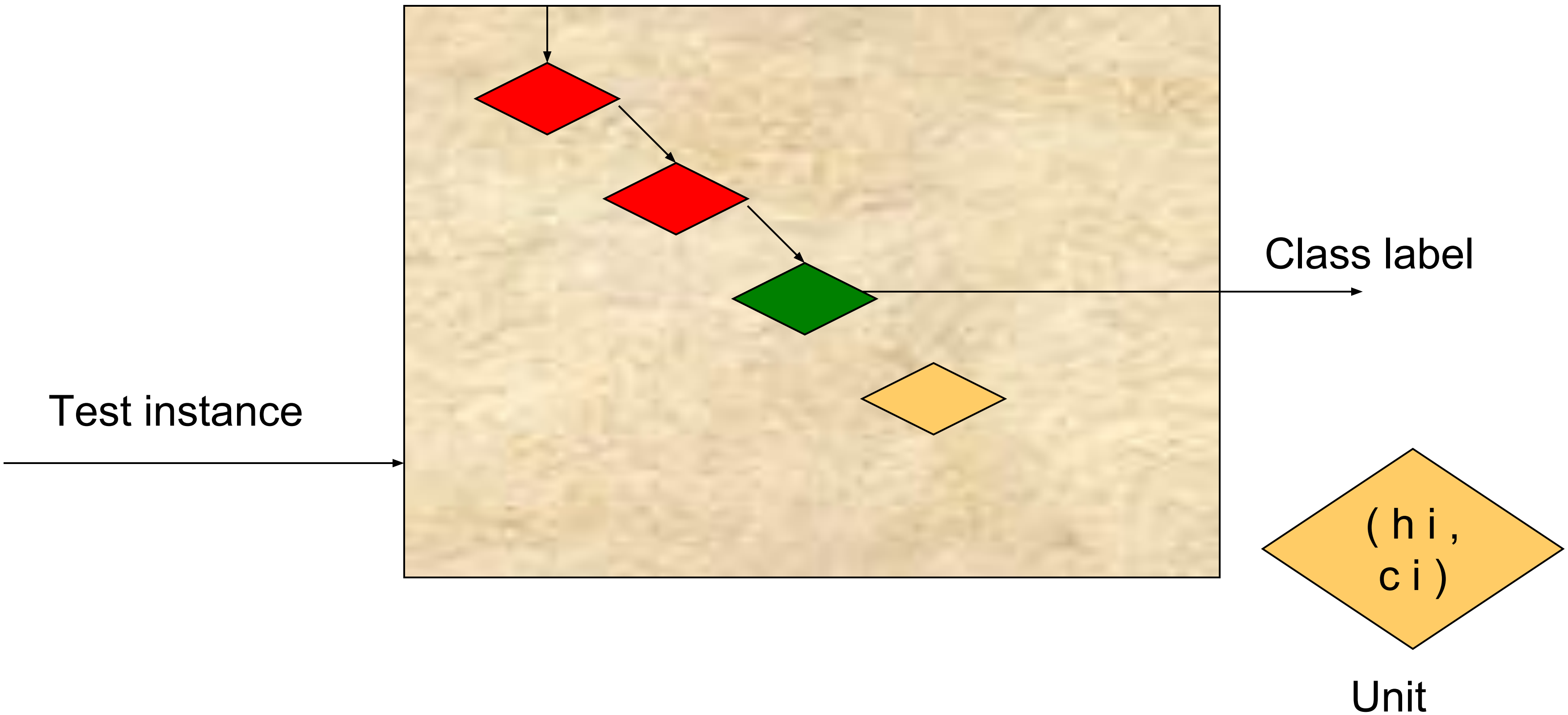
Decision Lists

Decision Lists

- A sequence of boolean functions that lead to a result

$$f(y) = \begin{cases} cj, & \text{if } j = \min \{ i \mid hi(y) = 1 \} \text{ exists} \\ 0 & \text{otherwise} \end{cases}$$

Decision List example



Decision List learning

R $\mathcal{R} = \{(h_i, b_i)\}_{i=1}^{r}$

(h k, 1 / 0)

S’ = S - Qk

training set $S = P \cup N$

$\mathcal{H} = \{h_i(\mathbf{x})\}_{i=1}^{|\mathcal{H}|}$

Set of candidate feature functions

For each hi,

Qi = Pi U Ni

(hi = 1)

Select hk, the feature with highest utility

If
(| Pi| - pn * | Ni |
>
|Ni| - pp *|Pi|)
then 1

else 0

U i = max { | Pi| - pn * | Ni | , |Ni| - pp *|Pi| }

Decision list Issues

What is the terminating condition?

1. Size of R (an upper threshold)
2. Q_k = null
3. S' contains examples of same class

Probabilistic classifiers

Probabilistic classifiers : NB

- Based on Bayes rule
- Naïve Bayes : Conditional independence assumption

$$P(C_i \mid X) = \frac{P(X \mid C_i) \cdot P(C_i)}{P(X)}$$

$$P(X \mid C_i) = \prod_{k=1}^{d} P(x_k \mid C_i)$$

Naïve Bayes Issues

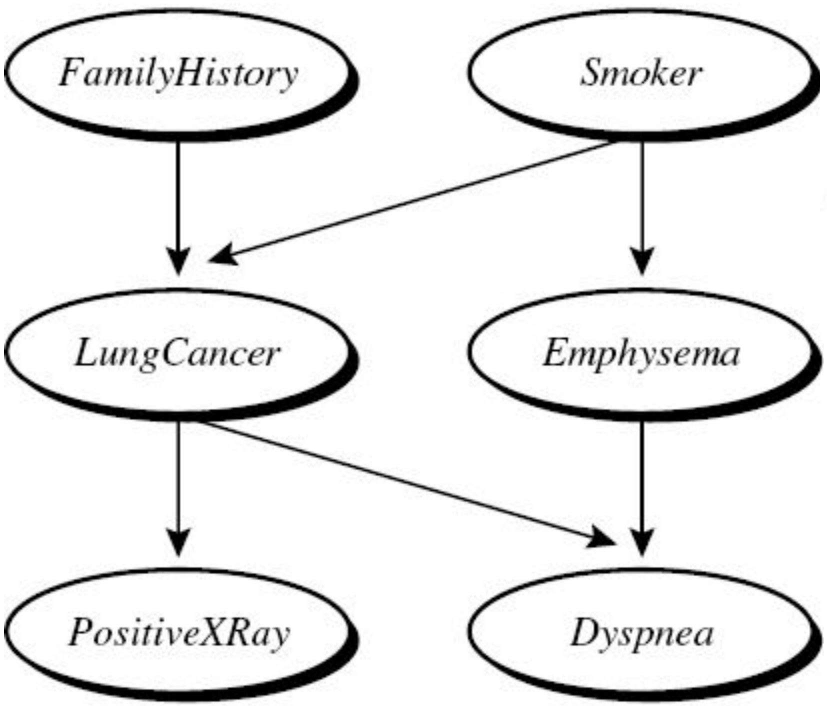
Problems due to sparsity of data?

Problem : Probabilities for some values may be zero

Solution : Laplace smoothing

For each attribute value,
update probability m / n as : (m + 1) / (n + k)
where k = domain of values

Probabilistic classifiers : BBN



	FH, S	FH, ~S	~FH, S	~FH, ~S
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

An added term for conditional probability between attributes:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | Parents(Y_i))$$

Diagram from Han-Kamber

BBN learning

(when network structure known)

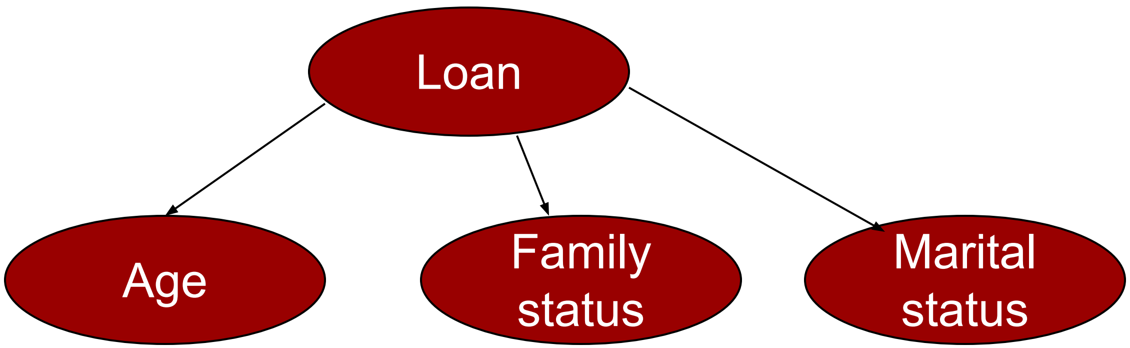
- Input : Network topology of BBN
- Output : Calculate the entries in conditional probability table

(when network structure not known)

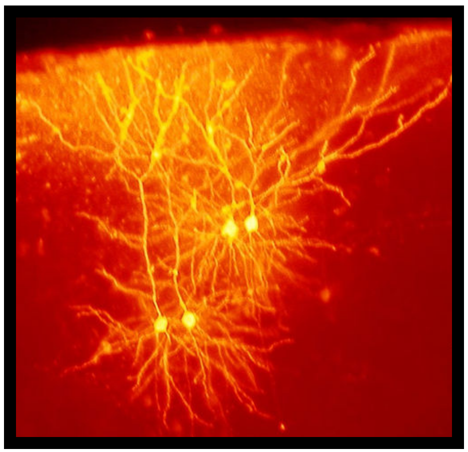
- ???

Learning structure of BBN

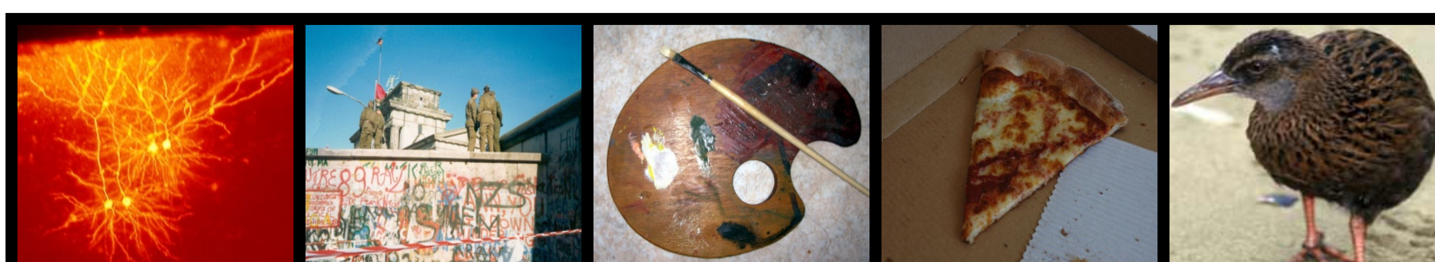
- Use Naïve Bayes as a basis pattern



- Add edges as required
- Examples of algorithms: TAN, K2

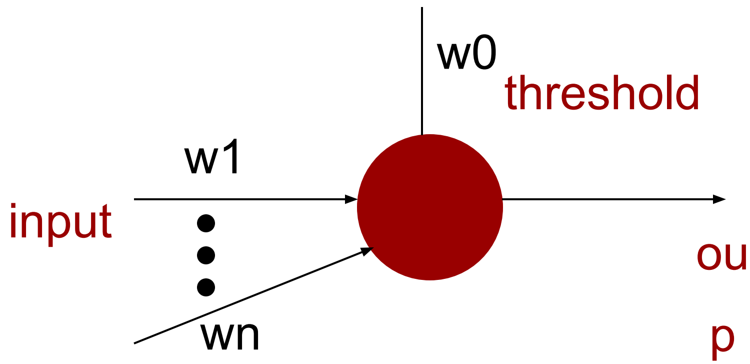


Artificial Neural Networks



Artificial Neural Networks

- Based on biological concept of neurons
- Structure of a fundamental unit of ANN:



output: : activation function

p (v) where

$p(v) = \mathrm{sgn}(w_0 + w_1x_1 + \ldots + w_nx_n)$

Perceptron learning algorithm

- Initialize values of weights
- Apply training instances and get output
- Update weights according to the update rule:

n : learning rate

t : target output

o : observed output

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - o)x_i$$

- Repeat till converges

- Can represent linearly separable functions only

Sigmoid perceptron

- Basis for multilayer feedforward networks

$$\sigma(y) = \frac{1}{1+e^{-y}}$$

$$\frac{d\sigma(y)}{dy} = \sigma(y) \cdot (1 - \sigma(y))$$

Multilayer feedforward networks

- Multilayer? Feedforward?

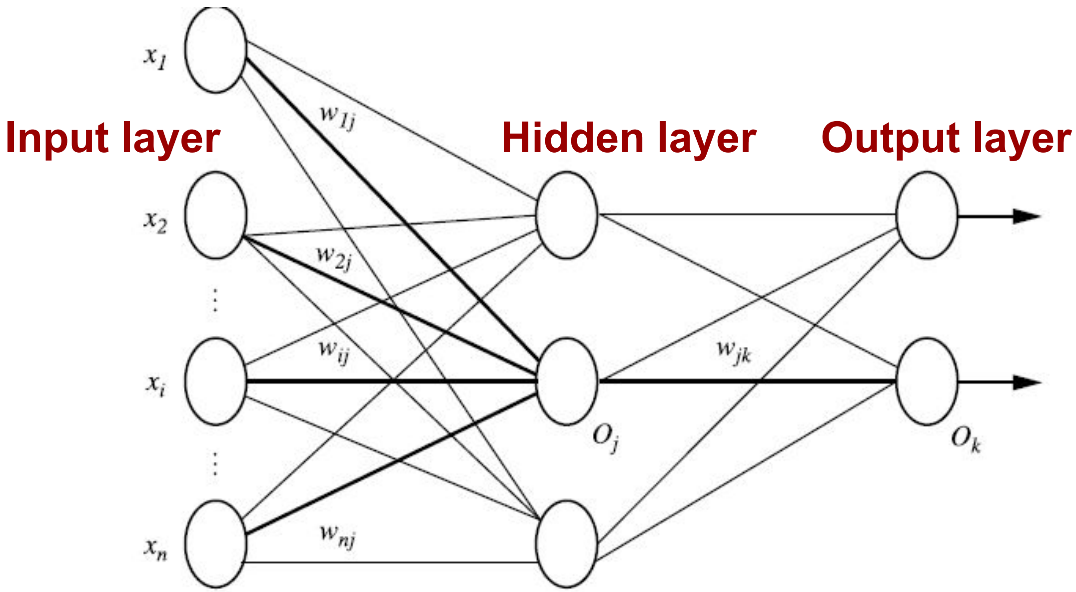
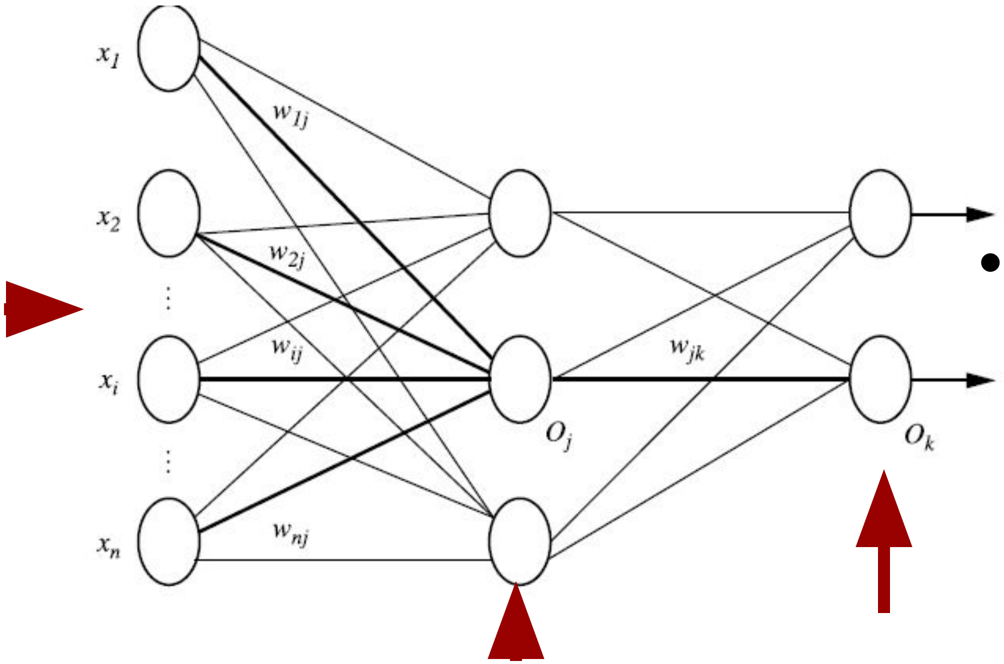


Diagram from Han-Kamber

Backpropagation

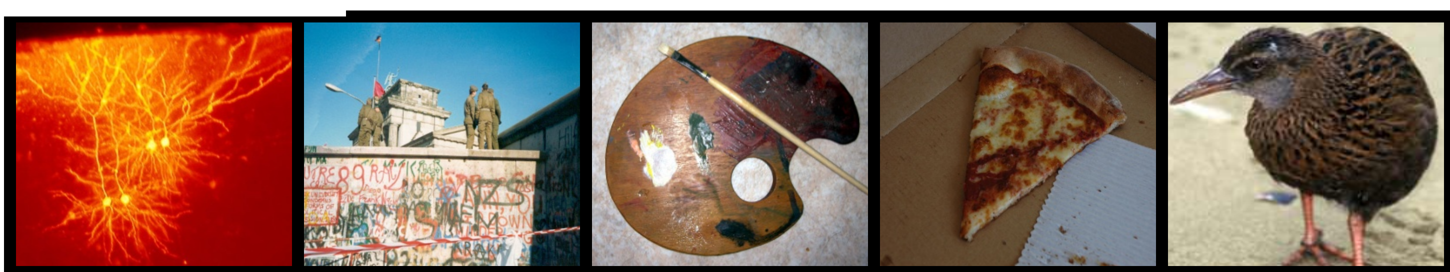


- Apply training instances as input and produce output
- Update weights in the 'reverse' direction as follows:

$$\Delta w_{ji} = \eta \delta j o_i$$

$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} w_{kh}\delta_k \quad {}_j)$$

Diagram from Han-Kamber



ANN Issues

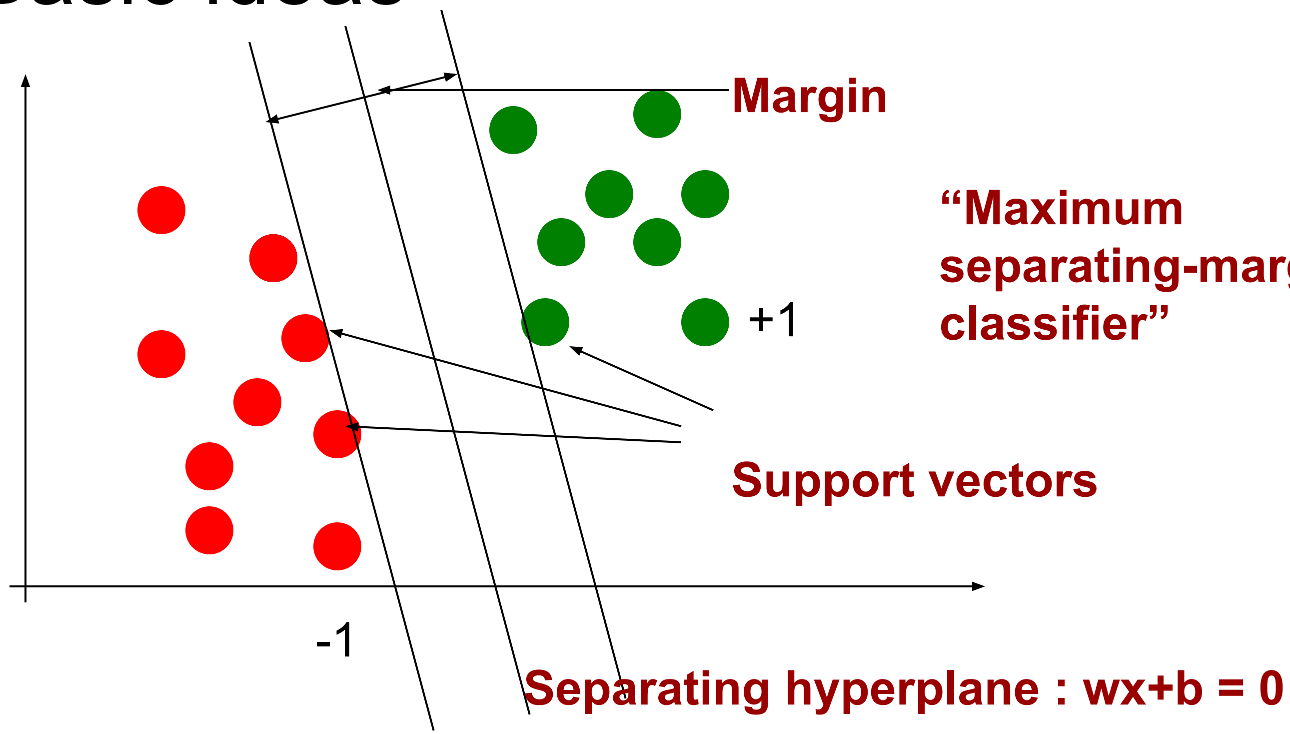
Learning the structure of the network

1. Construct a complete network
2. Prune using heuristics:
    - Remove edges with weights nearly zero
    - Remove edges if the removal does not affect accuracy

Support vector machines

Support vector machines

- Basic ideas



"Maximum separating-margin classifier"

SVM training

Maximize $\sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l Q_{kl}$ where $Q_{kl} = y_k y_l (\mathbf{x}_k . \mathbf{x}_l)$

Subject to these constraints: $0 \leq \alpha_k \leq C \quad \forall k \qquad \sum_{k=1}^{R} \alpha_k y_k = 0$

Then define:

$$\mathbf{w} = \sum_{k=1}^{R} \alpha_k y_k \mathbf{x}_k$$

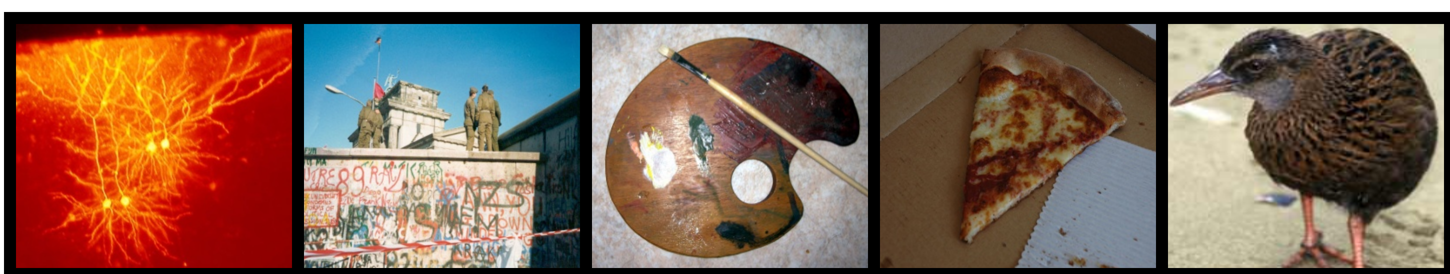
$$b = y_K (1 - \varepsilon_K) - \mathbf{x}_K . \mathbf{w}_K$$

$$\text{where} \quad K = \arg\max_k \alpha_k$$

Lagrangian multipliers are

her

Dot product of xk and xl



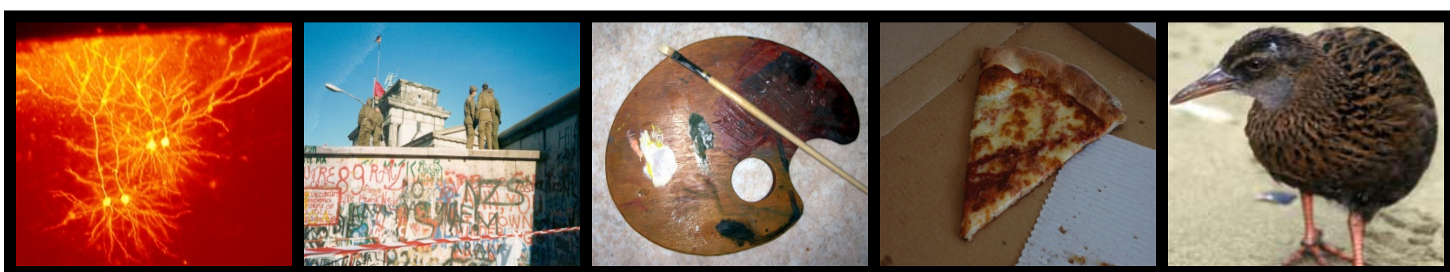
Focussing on dot product

$$\text{Maximize } \sum_{k=1}^{R}\alpha_k - \frac{1}{2}\sum_{k=1}^{R}\sum_{l=1}^{R}\alpha_k\alpha_l Q_{kl} \text{ where } Q_{kl} = y_k y_l(\Phi(\mathbf{x}_k).\Phi(\mathbf{x}_l))$$

- For non-linear separable points,

we plan to map them to a higher dimensional (and linearly separable) space

- The product $\Phi(\mathbf{x}_k).\Phi(\mathbf{x}_l)$ n be time-consuming. Therefore, we use kernel functions



Kernel functions

$$\text{Maximize} \sum_{k=1}^{R} \alpha_k - \frac{1}{2} \sum_{k=1}^{R} \sum_{l=1}^{R} \alpha_k \alpha_l Q_{kl} \text{ where } \quad Q_{kl} = y_k y_l \, k(\mathbf{x}, \mathbf{x}')$$

- Without having to know the non-linear mapping, apply kernel function, say,

$$k(\mathbf{x}, \mathbf{x}') = \left(\text{scale} \cdot \langle \mathbf{x}, \mathbf{x}' \rangle + \text{offset}\right)^{\text{degree}}$$

- Reduces the number of computations required to generate Q kl values.

Testing SVM

$$\sum_{i=1}^{N_S} \alpha_i y_i K(\mathbf{s}_i, \mathbf{x}) + b$$

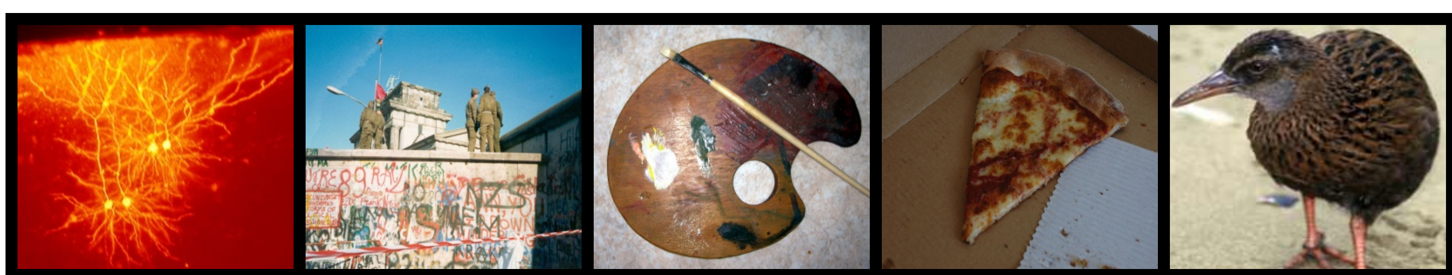
SVM Issues

What if n-classes are to be predicted?

Problem : SVMs deal with two-class classification

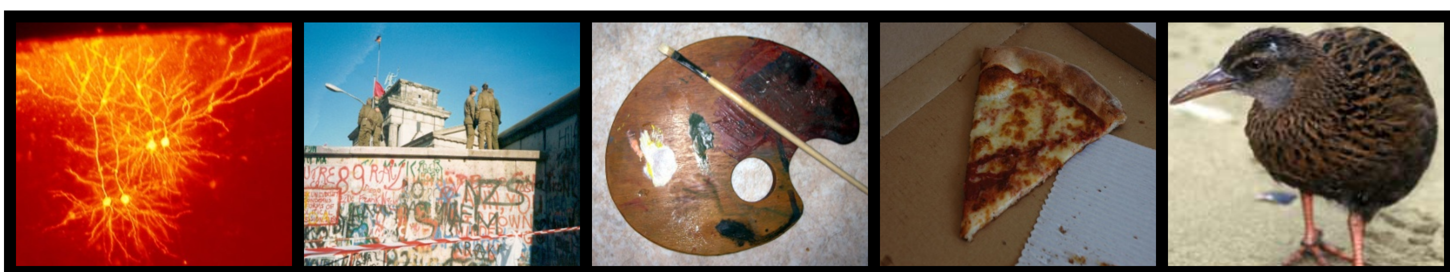
Solution : Have multiple SVMs each for one class

Combining classifiers

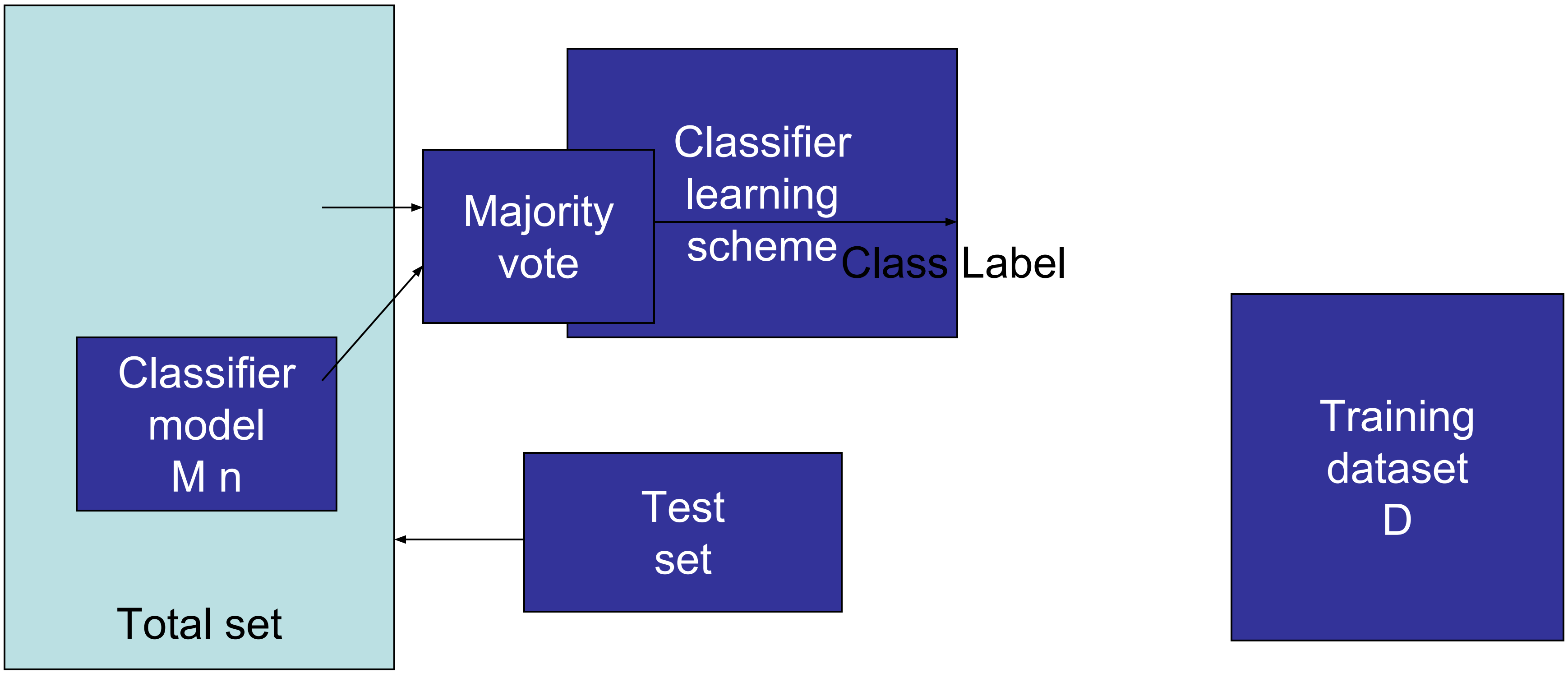


Combining Classifiers

- ‘Ensemble’ learning
- Use a combination of models for prediction
  - Bagging : Majority votes
  - Boosting : Attention to the ‘weak’ instances
- Goal : An improved combined model

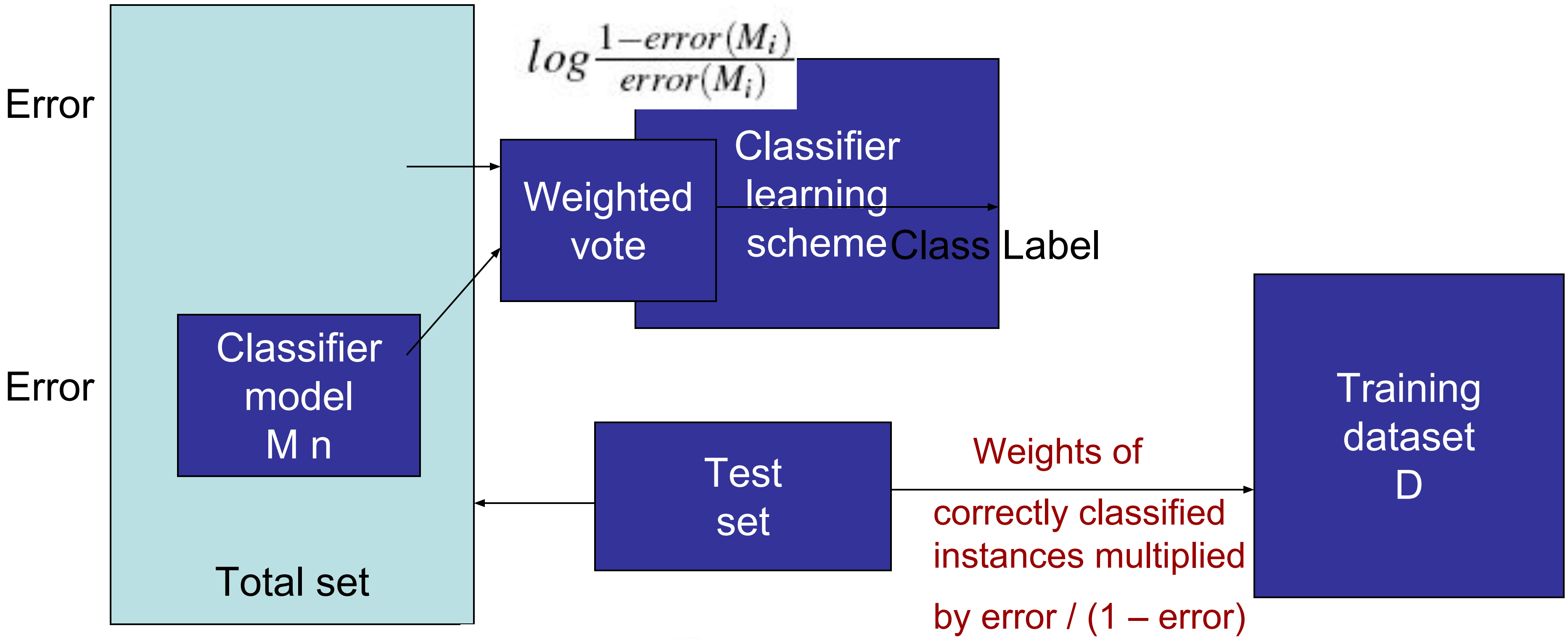


Bagging



At random. May use bootstrap sampling with replacement

Boosting (AdaBoost)

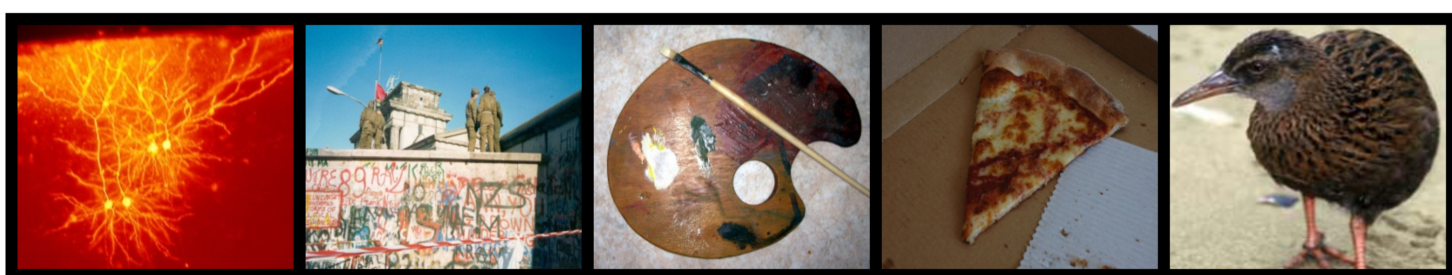


$$error(M_i) = \sum_{j}^{d} w_j \times err(X_j)$$

If error > 0.5?

lacement

The last slice



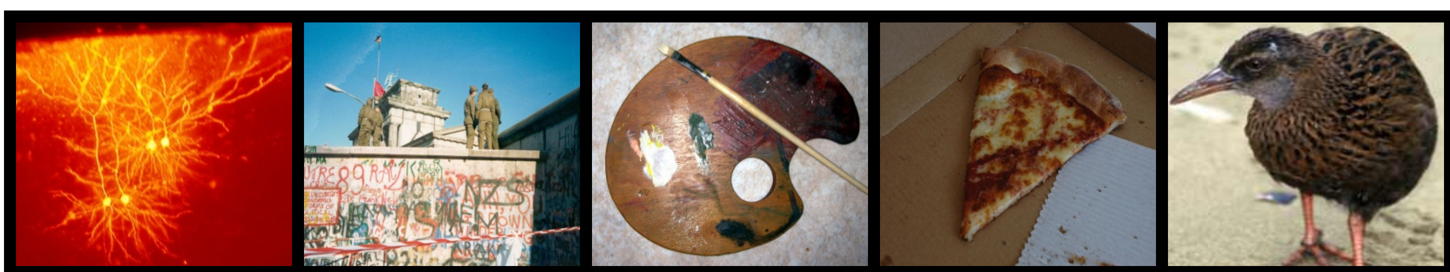
Data preprocessing

- Attribute subset selection
  - Select a subset of total attributes to reduce complexity
- Dimensionality reduction
  - Transform instances into ‘smaller’ instances

Attribute subset selection

- Information gain measure for attribute selection in decision trees

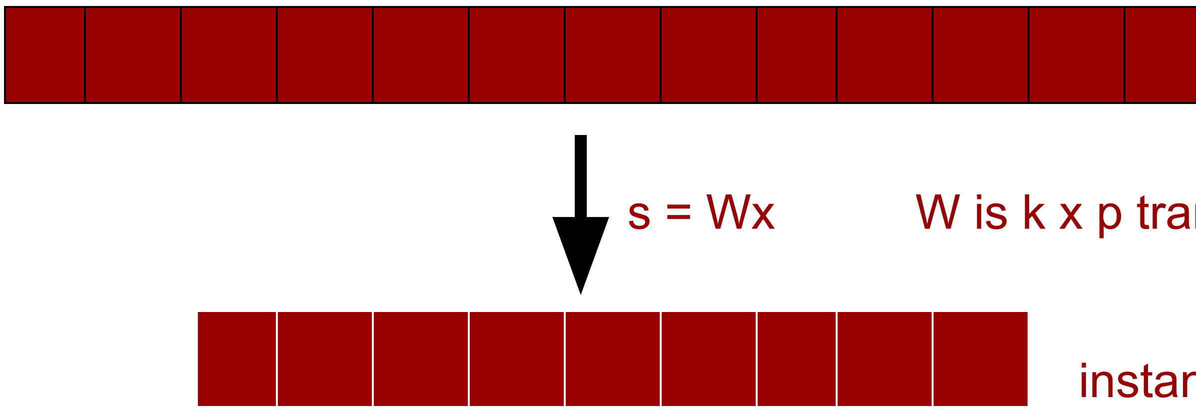
- Stepwise forward / backward elimination of attributes



Dimensionality reduction

Number of attributes of a data instance

- High dimensions : Con complexity



instance x in p-dimensions

$s = Wx$

W is k x p transformation mtrx.

instance x in k-dimensions

$k < p$

Principal Component Analysis

- Computes k orthonormal vectors : Principal components
- Essentially provide a new set of axes – in decreasing order of variance

$$S = U^T X.$$

$$w_1 = \arg\max_{||w=1||} \mathrm{Var}\{x^T w\}$$

$$s = Wx$$

(k X n) (p X n) (k X p)

Eig

F

(p X n)

(p X n)

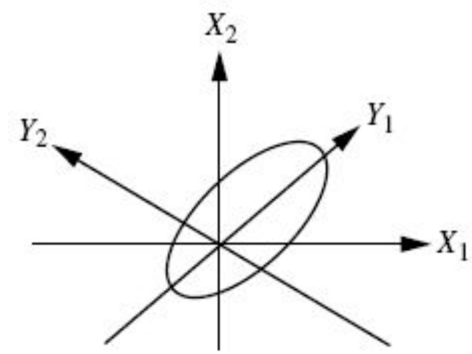
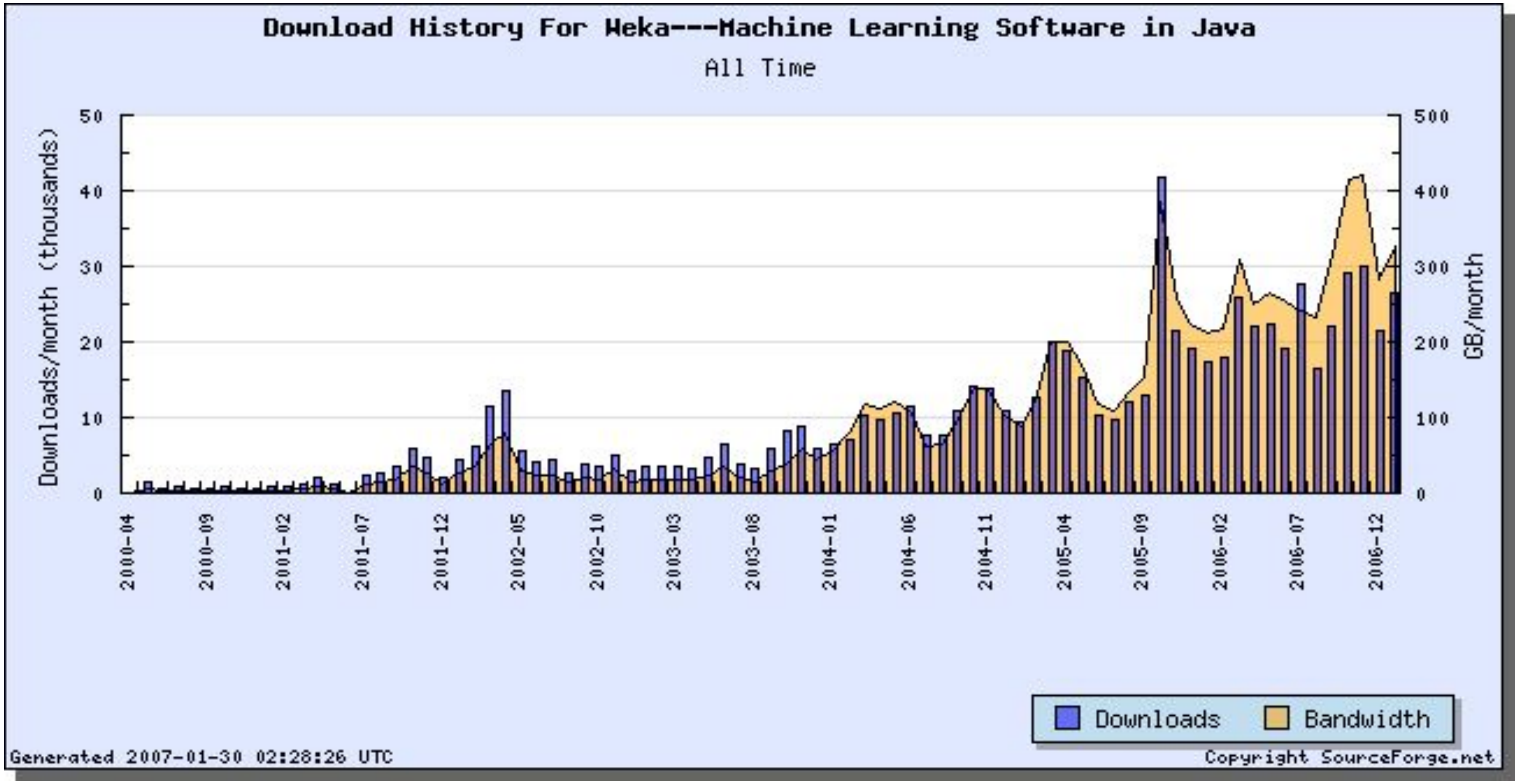


Diagram from Han-Kamber

Weka &

Weka Demo

Weka & Weka Demo



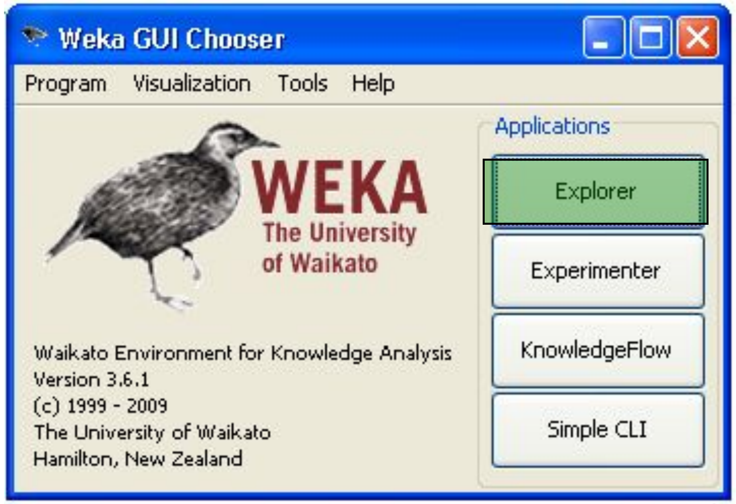
ARFF file format

@RELATION nursery Name of the relation

@ATTRIBUTE children numeric Attribute definition
@ATTRIBUTE housing {convenient, less_conv, critical}
@ATTRIBUTE finance {convenient, inconv}
@ATTRIBUTE social {nonprob, slightly_prob, problematic}
@ATTRIBUTE health {recommended, priority, not_recom}
@ATTRIBUTE pr_val {recommend,priority,not_recom,very_recom,spec_prior}

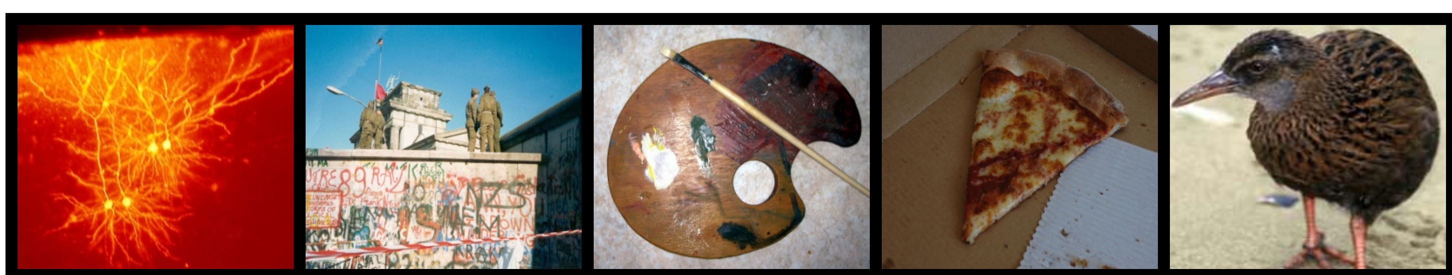
@DATA Data instances : Comma separated, each on a new line
3,less_conv,convenient,slightly_prob,recommended,spec_prior

Parts of weka



Knowledge Flow

Similar to Work Flow
'Customized' to one's needs



Weka demo

Key References

- Data Mining – Concepts and techniques; Han and Kamber, Morgan Kaufmann publishers, 2006.

- Machine Learning; Tom Mitchell, McGraw Hill publications.

- Data Mining – Practical machine learning tools and techniques; Witten and Frank, Morgan Kaufmann publishers, 2005.

end of slideshow

Extra slides 1

Difference between decision lists and decision trees:

1. Lists are functions tested sequentially (More than one attributes at a time)

   Trees are attributes tested sequentially

2. Lists may not require a ‘complete’ coverage for values of an attribute.

   All values of an attribute correspond to atleast one branch of the attribute split.

Learning structure of BBN

- K2 Algorithm :
  - Consider nodes in an order
  - For each node, calculate utility to add an edge from previous nodes to this one
- TAN :
  - Use Naïve Bayes as the baseline network
  - Add different edges to the network based on utility
- Examples of algorithms: TAN, K2

Delta rule

- Delta rule enables to converge to a best fit if points are not linearly separable
- Uses gradient descent to choose the hypothesis space

$$\Delta w_i = \eta \sum_{d \in D} (t_d - o_d)\, x_{id}$$