Classification

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, Health status, Salary)
(Textual features : Ngrams)
(Perceptual features)
Attributes
(a1, a2,… an)

Classifier

Discrete-valued
Category of document? {Politics, Movies, Biology}
Issue Loan? {Yes, No}
Steer? {Left, Straight, Right}
Class label

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 (2/3rd training, 1/3rd 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

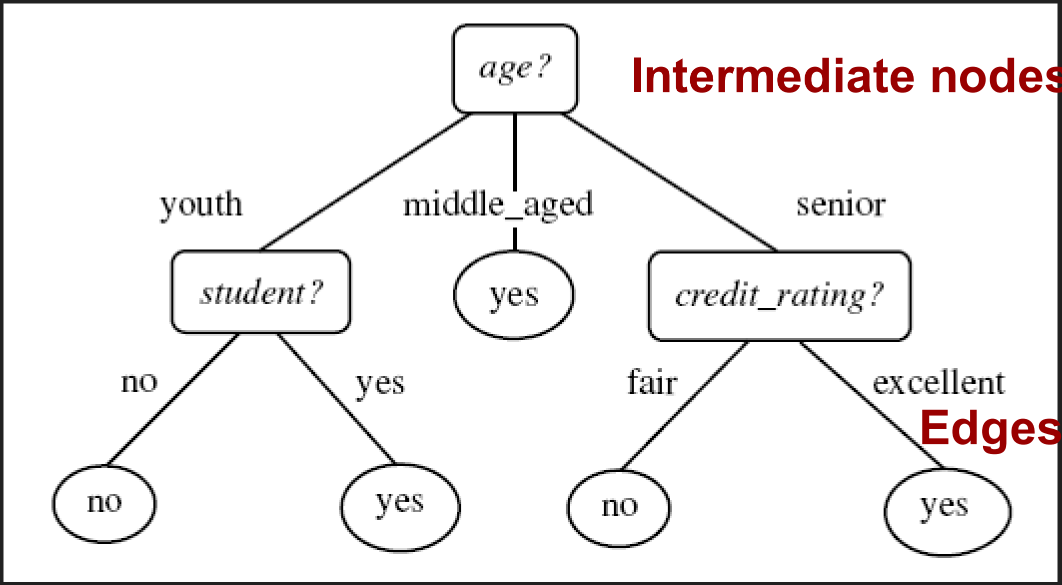
Evaluating classifiers

- 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

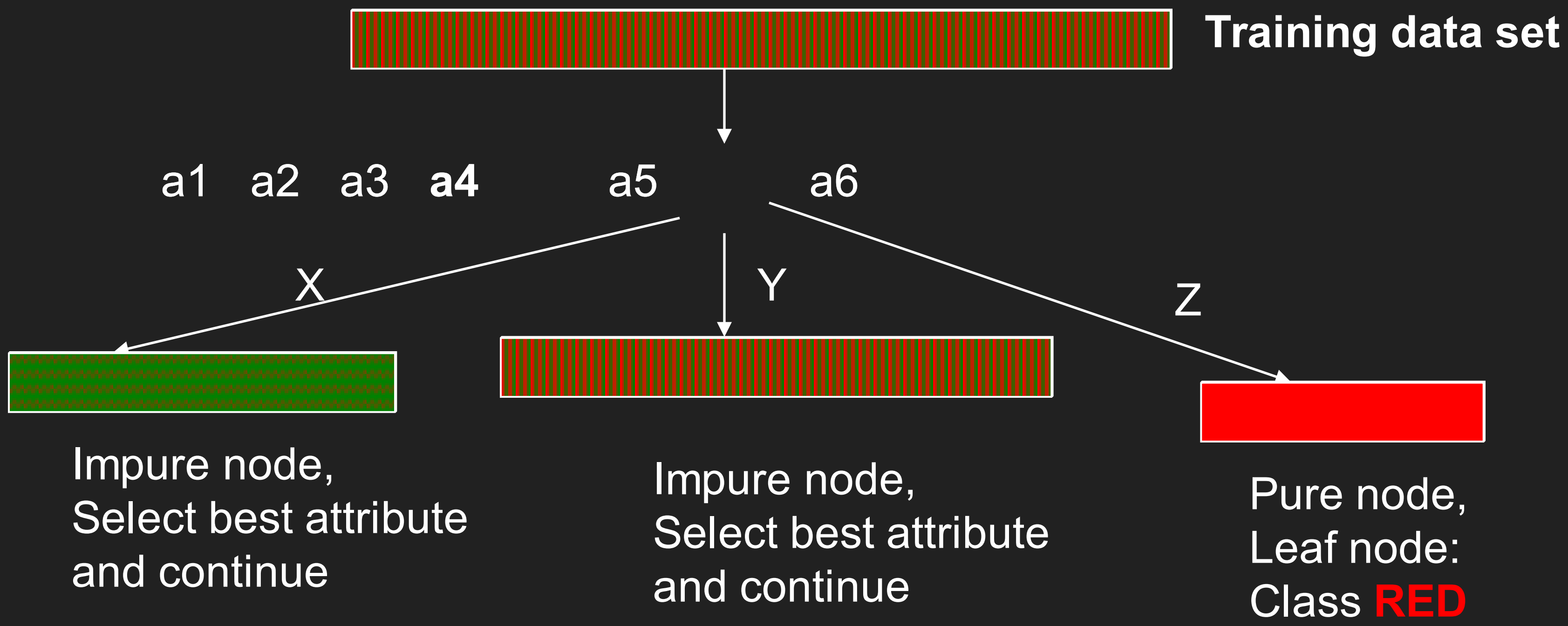
Edges : Attribute value tests

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 neighbor classifier

(K – NN classifier)

"One is known by the company one keeps"

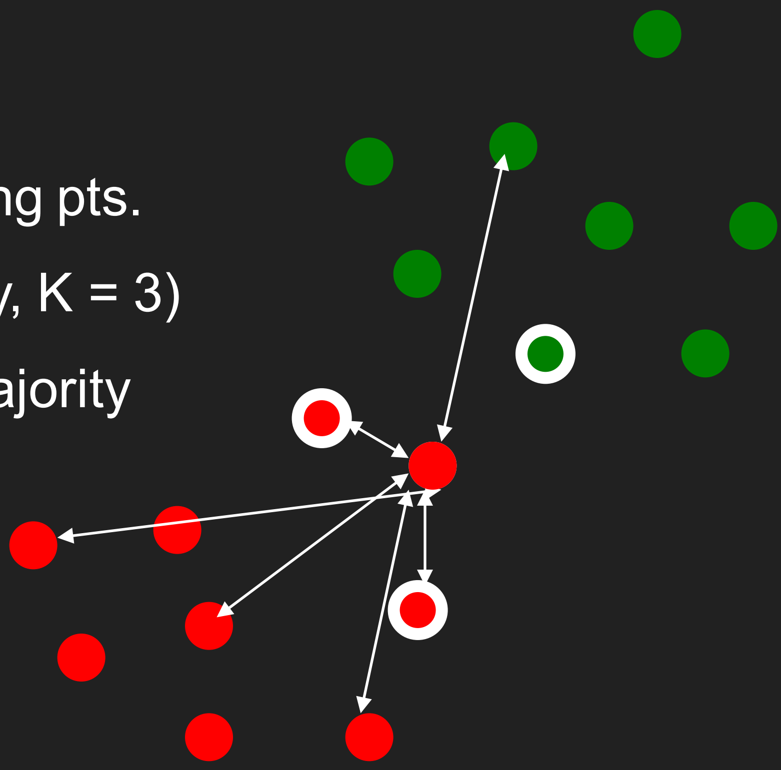
K-NN classifier schematic

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’)

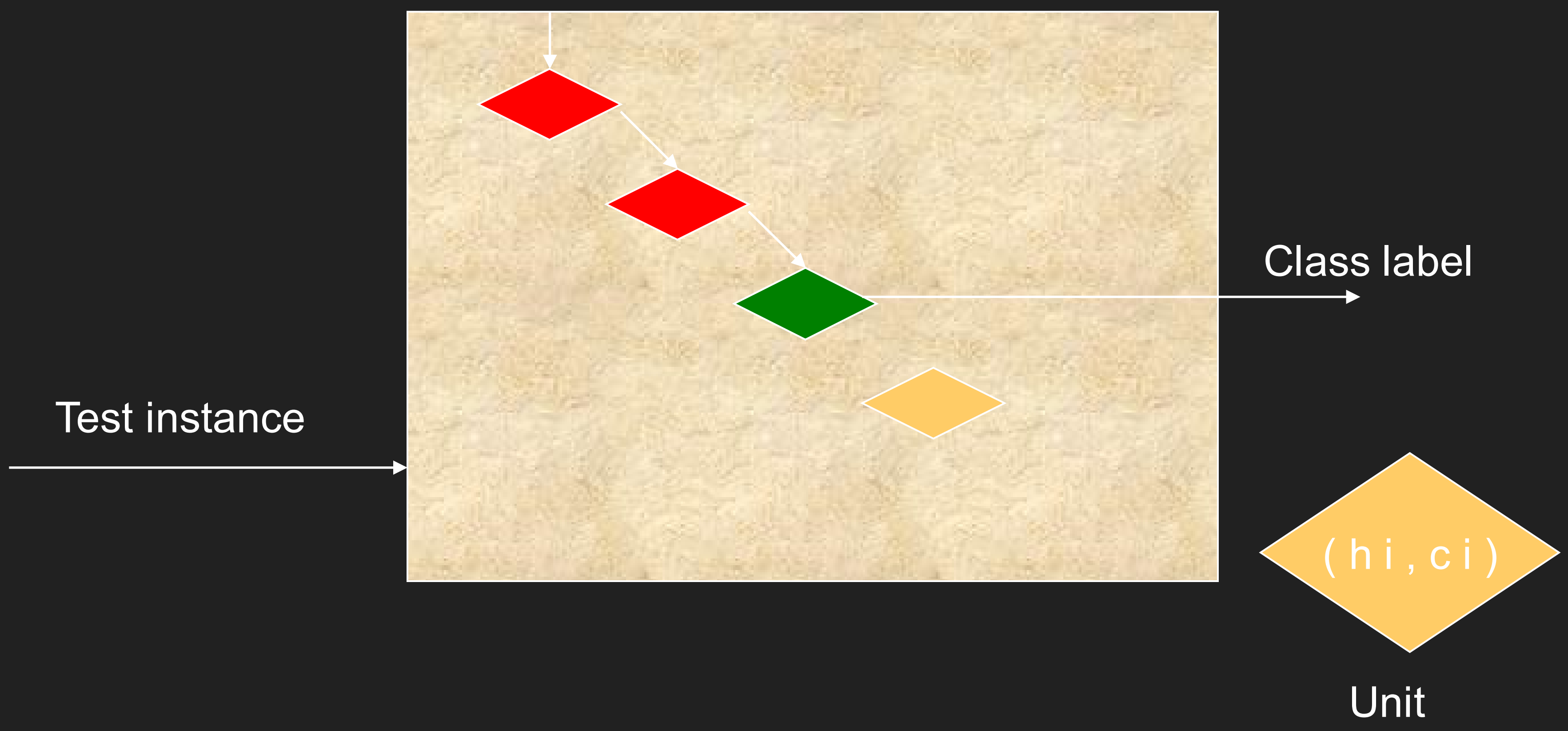
Decision Lists

Decision Lists

- A sequence of boolean functions that lead to a result

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

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

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

- Bayesian belief networks: Attributes ARE dependent
- A directed acyclic graph and conditional probability tables

Diagram from Han-Kamber

BBN learning

(when network structure known)

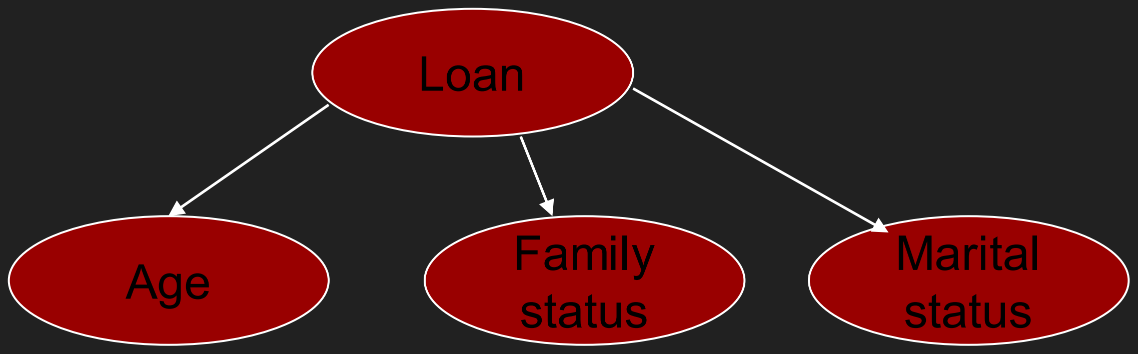
- Input: Network topology of BBN
- Output: Calculate the entries in conditional probability table

(when network structure not known)

- **???**

Learning structure of BBN

- Use Naïve Bayes as a basis pattern



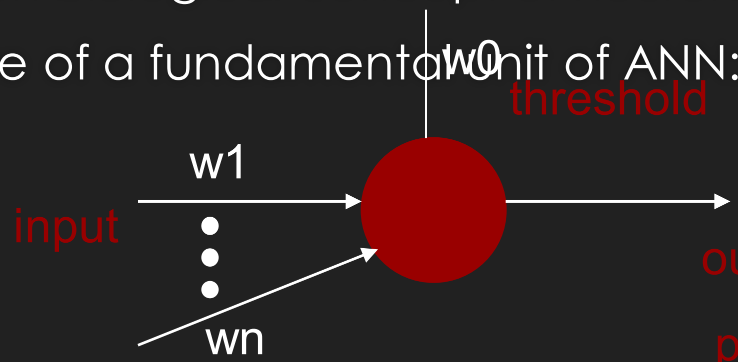
- Add edges as required
- Examples of algorithms: TAN, K2

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

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

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

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

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

- Basis for multilayer feedforward networks

Multilayer feedforward networks

- Multilayer? Feedforward?

Input layer **Hidden layer** **Output layer**

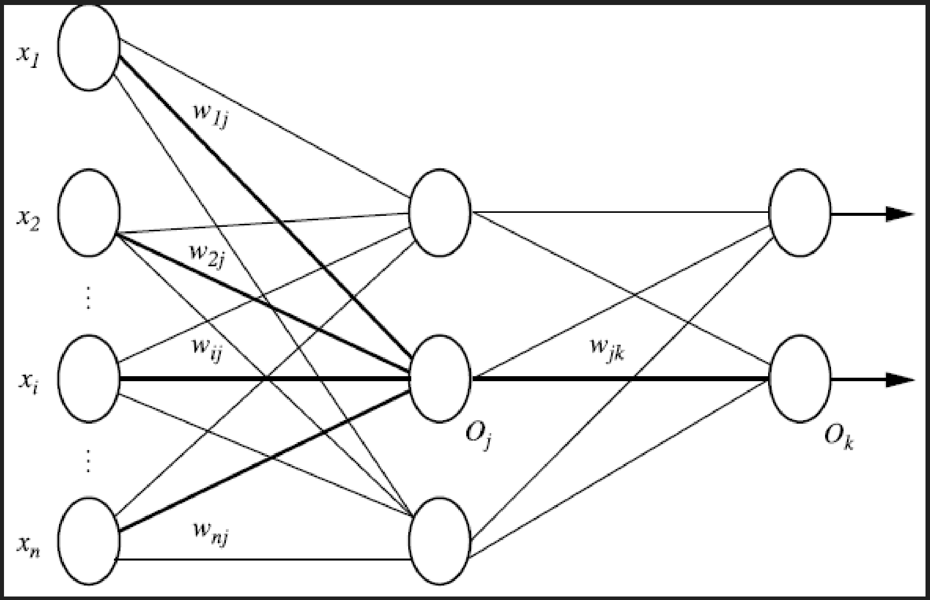
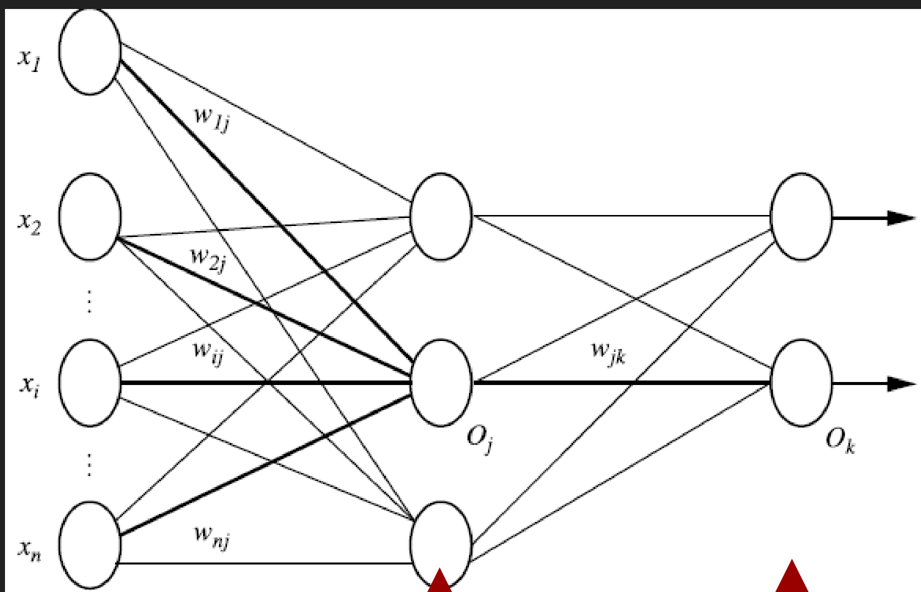


Diagram from Han-Kamber

Backpropagation



$$\delta_j = (t_j - o_j)o_j(1 - o_j)$$

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

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

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

Diagram from Han-Kamber

ANN Issues

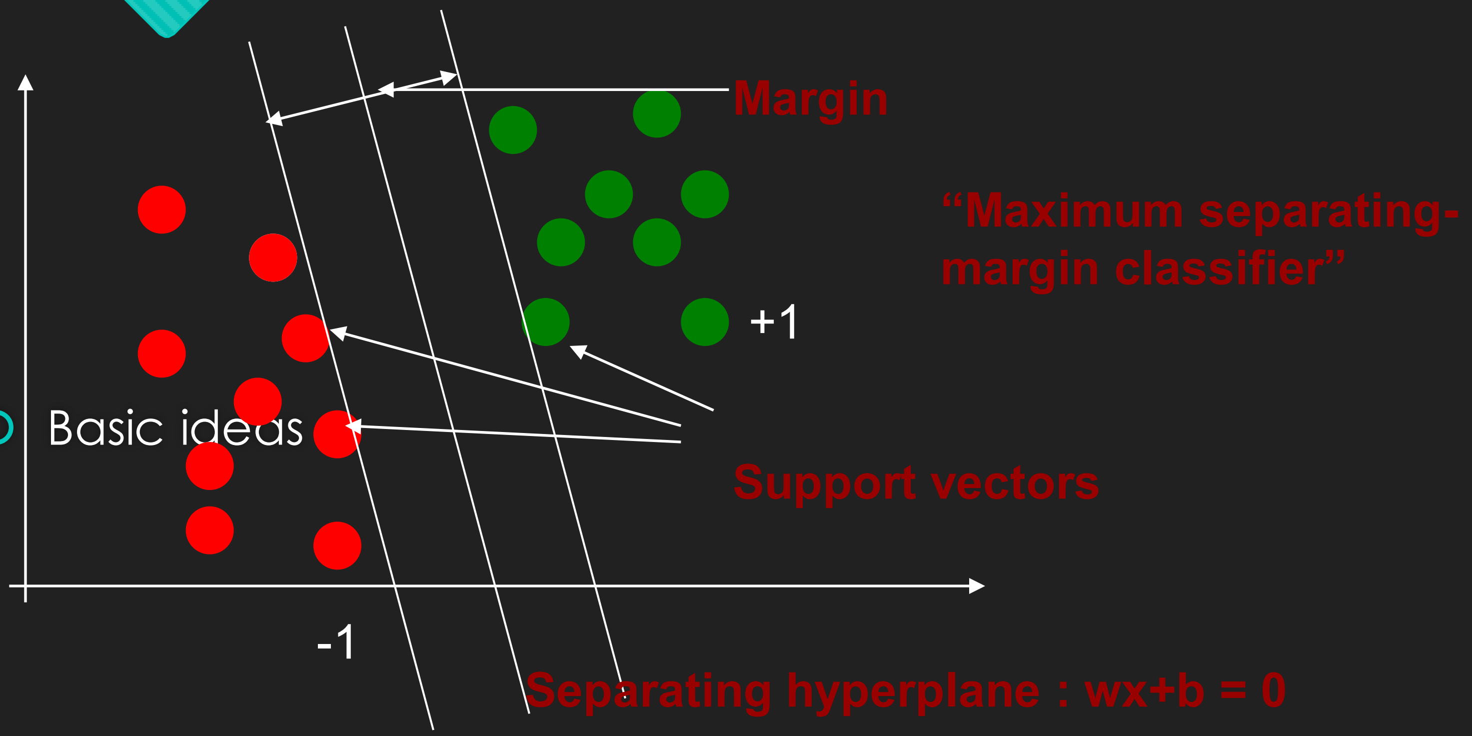
Learning the structure of the network

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



SVM Issues

What if n-classes are to be predicted?

Problem : SVMs deal with two-class classification

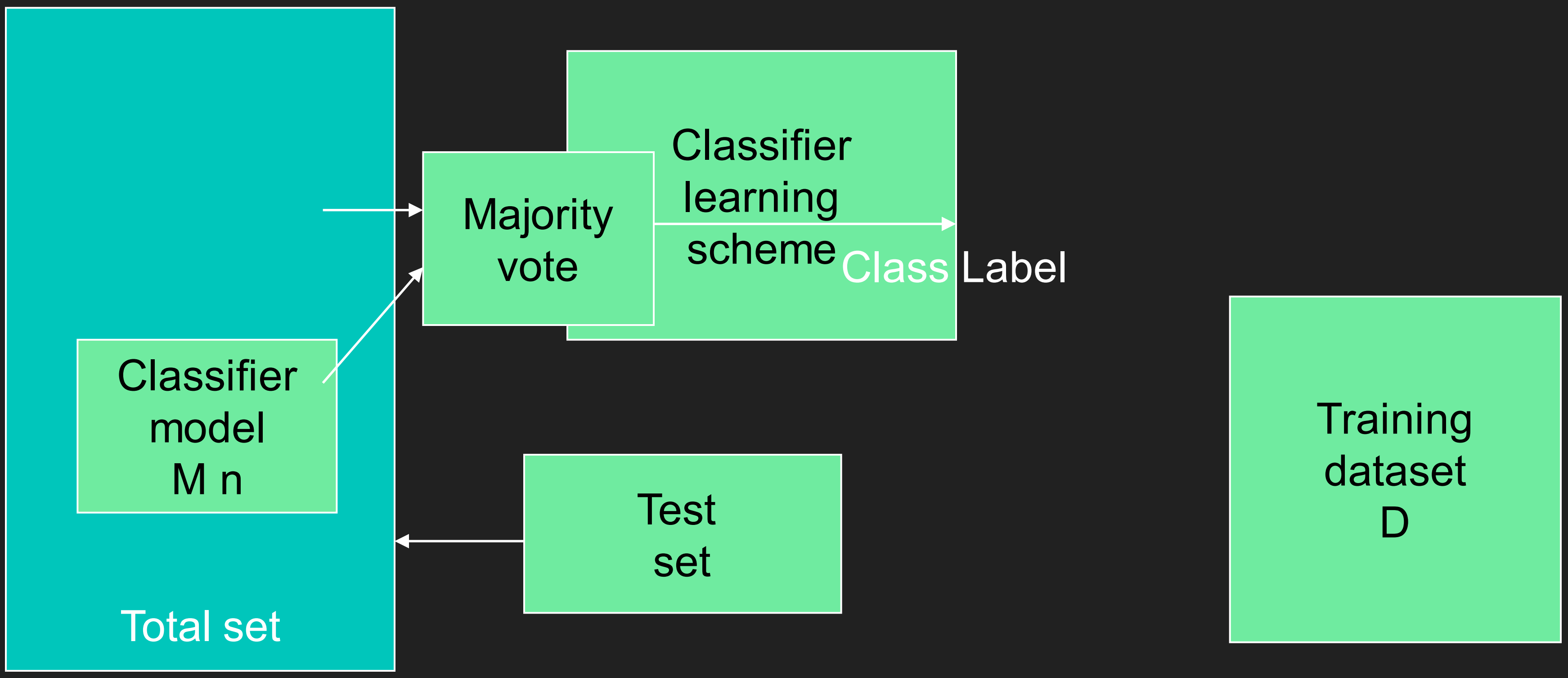
Solution : Have multiple SVMs each for one class

Combining classifiers

Combining Classifiers

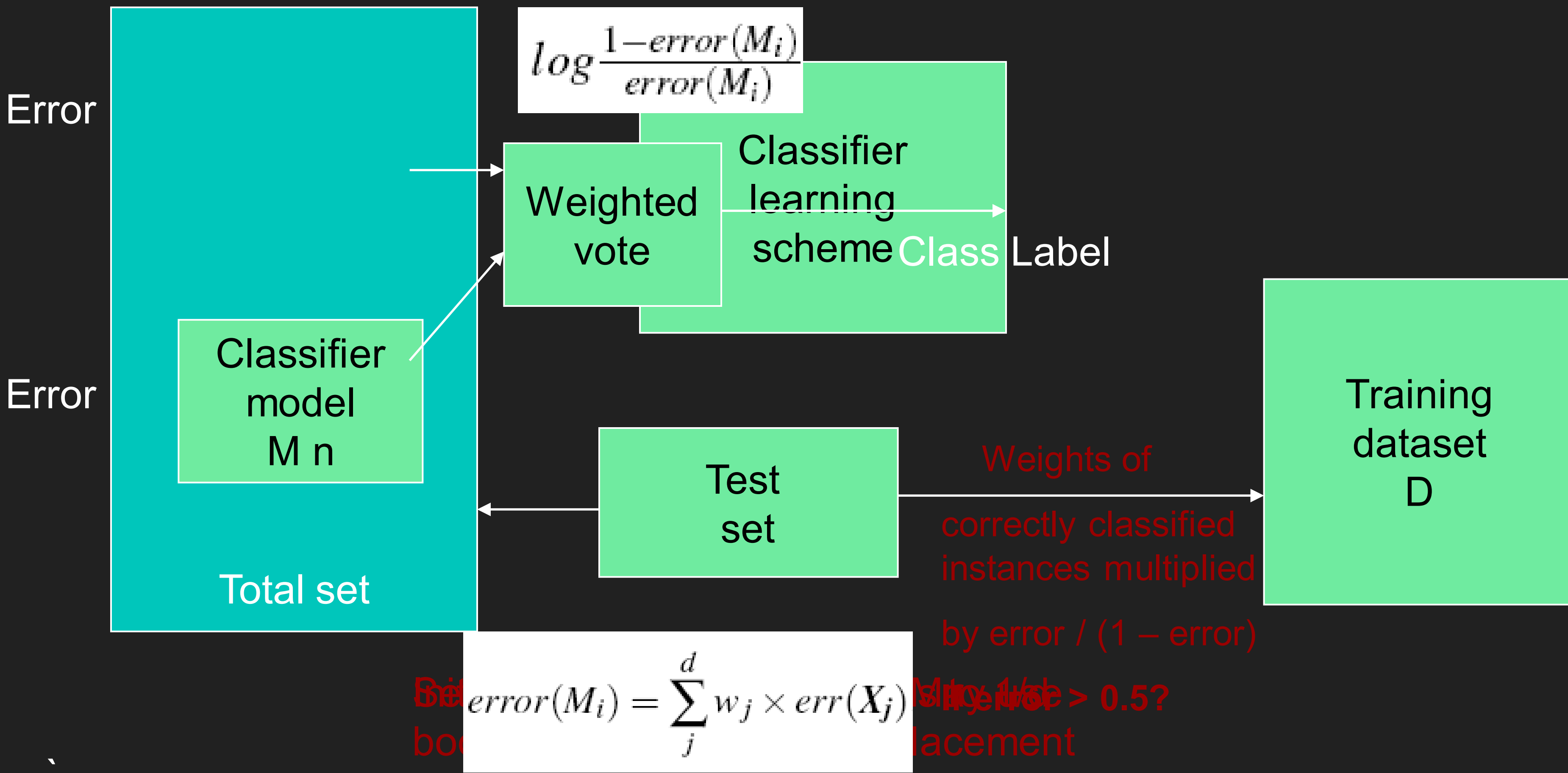
- ‘Ensemble’ learning
- Use a combination of models for prediction
  - Bagging : Majority votes
  - Boosting : Attention to the ‘weak’ instances
- Goal : An improved combined model

Bagging



At random. May use bootstrap sampling with replacement

Boosting (AdaBoost)



The last slice

Data preprocessing

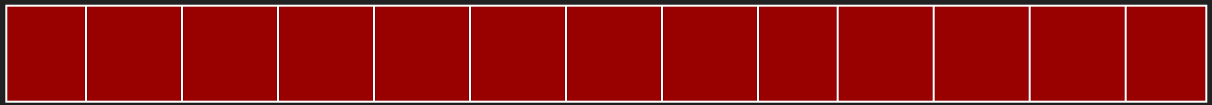
- Attribute subset selection
  - Select a subset of total attributes to reduce complexity
- Dimensionality reduction
  - Transform instances into ‘smaller’ instances

Attribute subset selection

- Information gain measure for attribute selection in decision trees
- Stepwise forward / backward elimination of attributes

Dimensionality reduction

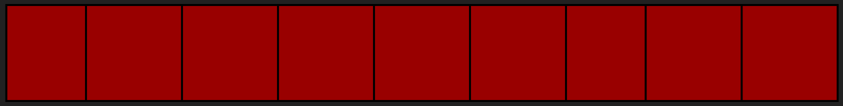
Number of attributes of
a data instance



instance x in
p-dimensions

s = Wx W is k x p transformation mtrx.

- High dimensions : Computational complexity



instance x in
k-dimensions

k < p

Principal Component Analysis

- Computes k orthonormal vectors : Principal c
- Ess $\mathbf{S} = \mathbf{U}^T\mathbf{X}.$ ovid $\mathbf{w}_1 = \arg\max_{||\mathbf{w}=1||} \mathrm{Var}\{\mathbf{x}^T\mathbf{w}\}$ dec variance

$$\mathbf{s} = \mathbf{W}\mathbf{x}$$

(k X n) (k X p) Eig F (p X n) (p X n)

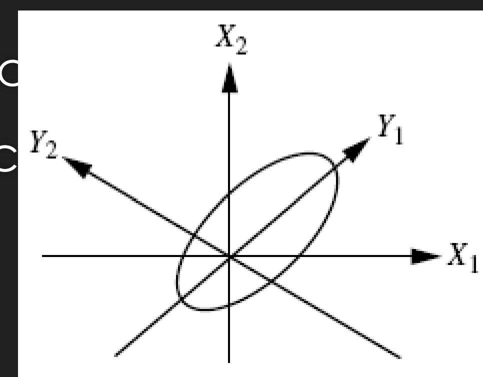


Diagram from Han-Kamber

end of slideshow