Genetic Algorithms for Classification and Feature Extraction

Min Pei, Erik D. Goodman, William F. Punch III and Ying Ding, (1995), "Genetic Algorithms For Classification and Feature Extraction", Michigan State University, Genetic Algorithms Research and Applications Group (GARAGe),CSNA-95.

Group 4

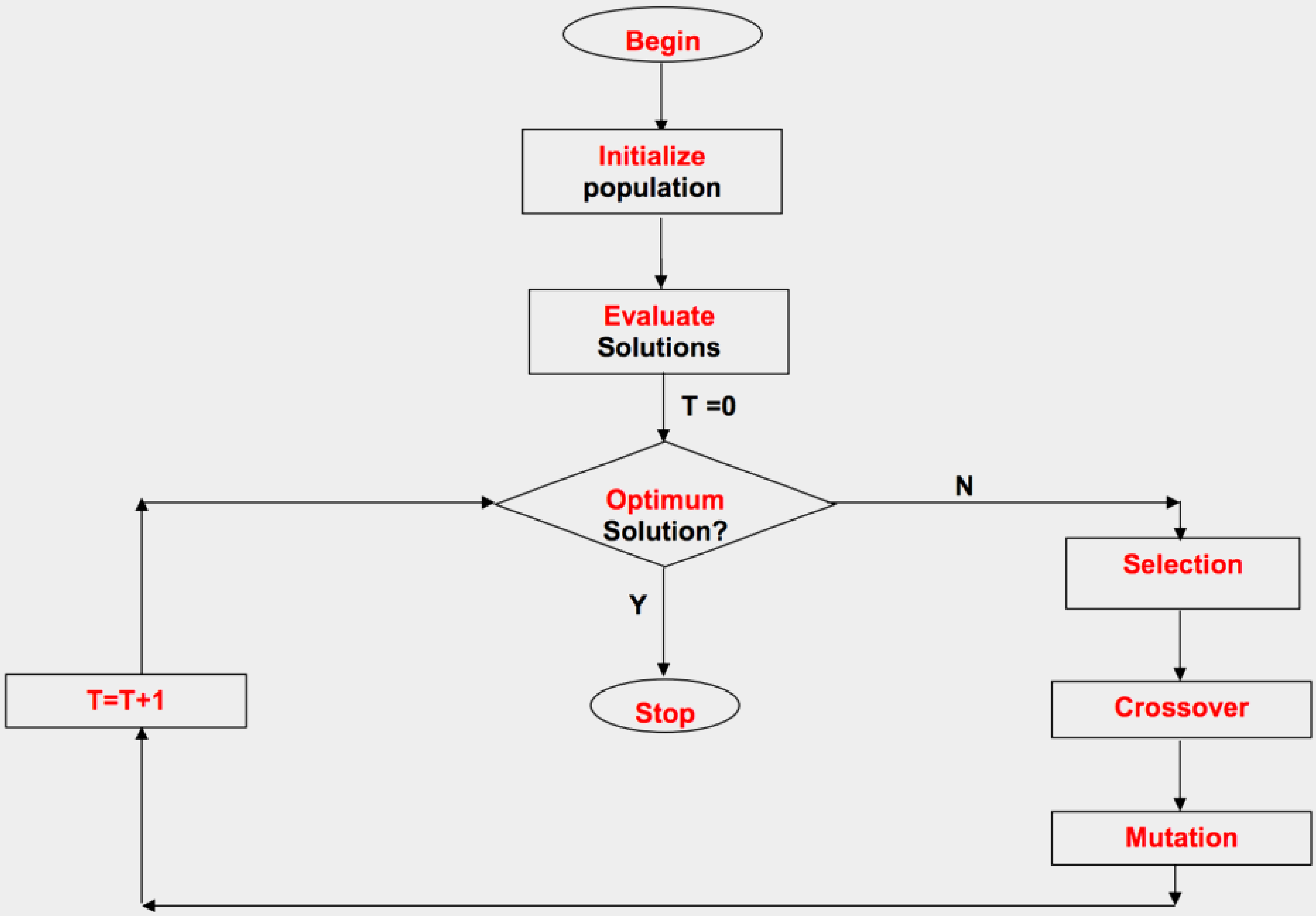
Hari Prasath Raman - 110283168

Venkatakrishnan Rajagopalan - 110455765

What is GA?

"Select The Best, Discard The Rest"

- Randomized heuristic search/optimization strategy
- Simulate natural selection
    - Initial population is composed of candidate solutions.
    - Evolve population from which strong and diverse candidates can emerge via mutation and crossover (mating).


Begin
Initialize population
Evaluate Solutions
T =0
Optimum Solution?
N
Y
Selection
Crossover
Mutation
Stop
T=T+1

```
Simple_Genetic_Algorithm()
    {
    Initialize the Population;
    Calculate Fitness Function;

    While(Fitness Value != Optimal Value)
        {
        Selection;//Natural Selection, Survival Of
   Fittest
        Crossover;//Reproduction, Propagate favorable
   characteristics

        Mutation;//Mutation
        Calculate Fitness Function;
        }
    }
```

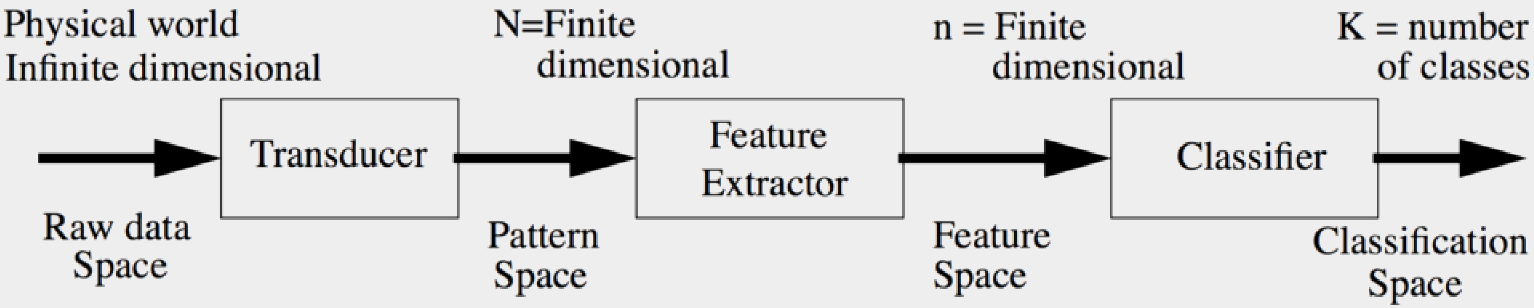
Basic Components

- Encoding
  - representing the solution in the form of a string
- Successor function(s)
  - Mutation, crossover
- Fitness function
- Some parameters
  - Population size
  - Generation limit

A different approach...

- Data mining techniques we learnt till now are based on modern database based languages
- Due to high dimensionality of the data, a numerical approach is needed
- The approach used here is 'statistical'
- Data Mining applies to many kinds of data and the type of data determines the language to be used
- Data used here is always a set of SPECIAL tuples
  - a data vector is similar to record in this unified table
- It does no apply to any set of data (collection of different tuples)
- We represent attributes as features (just a different term)

Pattern Recognition and Classification System



Curse of Dimensionality

- Extra information often leads to "non-optimal" use of data
- Removal of redundant and irrelevant features, increases classifier reliability
- But use of too few features or of non representative features can make the classification difficult

Feature(Attribute) Selection

- Task of finding the "best" subset of size *d* of features from the initial N features in the data pattern space

- Criterion used to define the best subset is usually the probability of misclassification

Feature (Attribute) Extraction

- Transformation from pattern space to feature space such that the new feature set gives better separation of pattern classes and reduces dimensionalty

- Feature Extraction is superset of feature selection (identity transformation of feature extraction)

Formal Definition of Feature Selection

- Define the set of N features representing the pattern in the pattern space as a vector x

$$\mathbf{x = [x_1, x_2, \cdots, x_N]}$$

- Goal is to find the best subset of features y that satisfies some optimal performance assigned by criterion function J(.), ideally the probability of misclassification

$$\mathbf{y = [y_1, y_2, \cdots, y_n]}$$

$$\mathbf{\min J(y) = \min \{(\forall y)\ J(y)\}}$$

Formal Definition of Feature Extraction

- Goal is to y
  ield a new feature vector of lower dimension by defining a mapping function M( ) such that y = M(x) which satisfies some optimal performance assigned by criterion function J( )
- The result of applying M is to create y such that |y| ≤ |x|

y = M(x)

min J{M(x)} = min {(∀M(x)) J(M(x))}

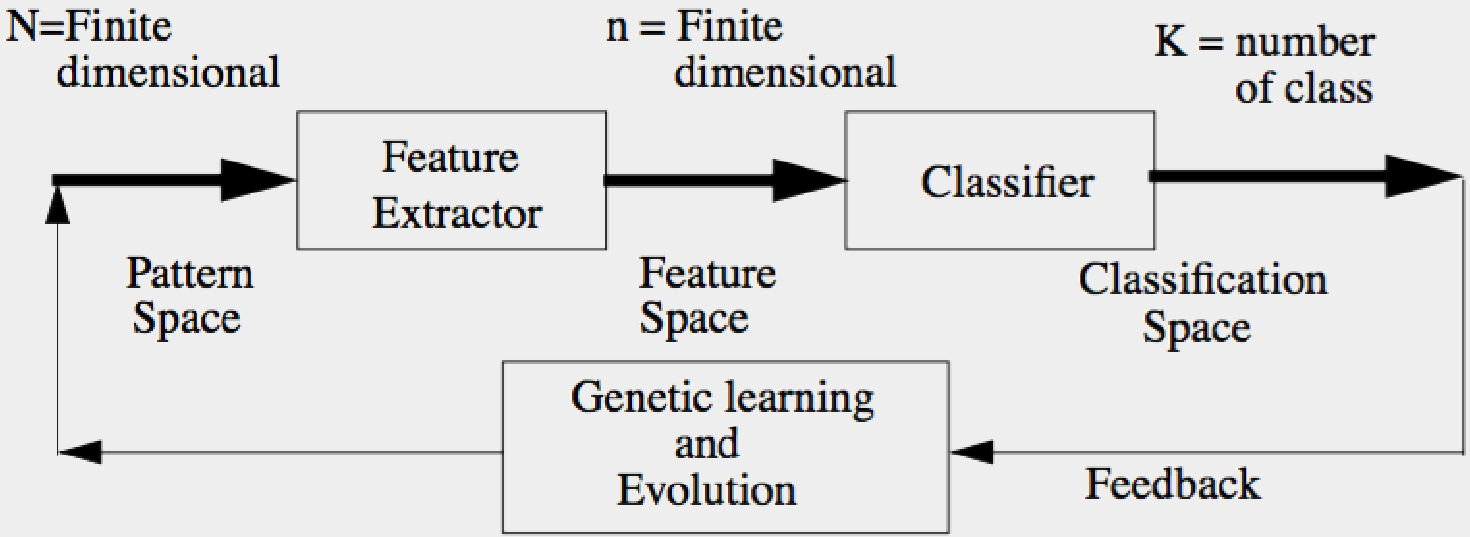
- In general, M(x) can be linear or non linear but majority of existing methods restrict to linear mapping

y = W x, where W is a (NxN) matrix of coefficients

Classification & Feature Evaluation

- Feature selection and extraction are crucial in optimizing performance and strongly affect classifier design.
- Inherent relation: Feedback linkage between feature evaluation and classification
- Feature extraction and classifier design carried out simultaneously through "Genetic learning and evolution"

Feature Extractor and Classifier with Feedback Learning System



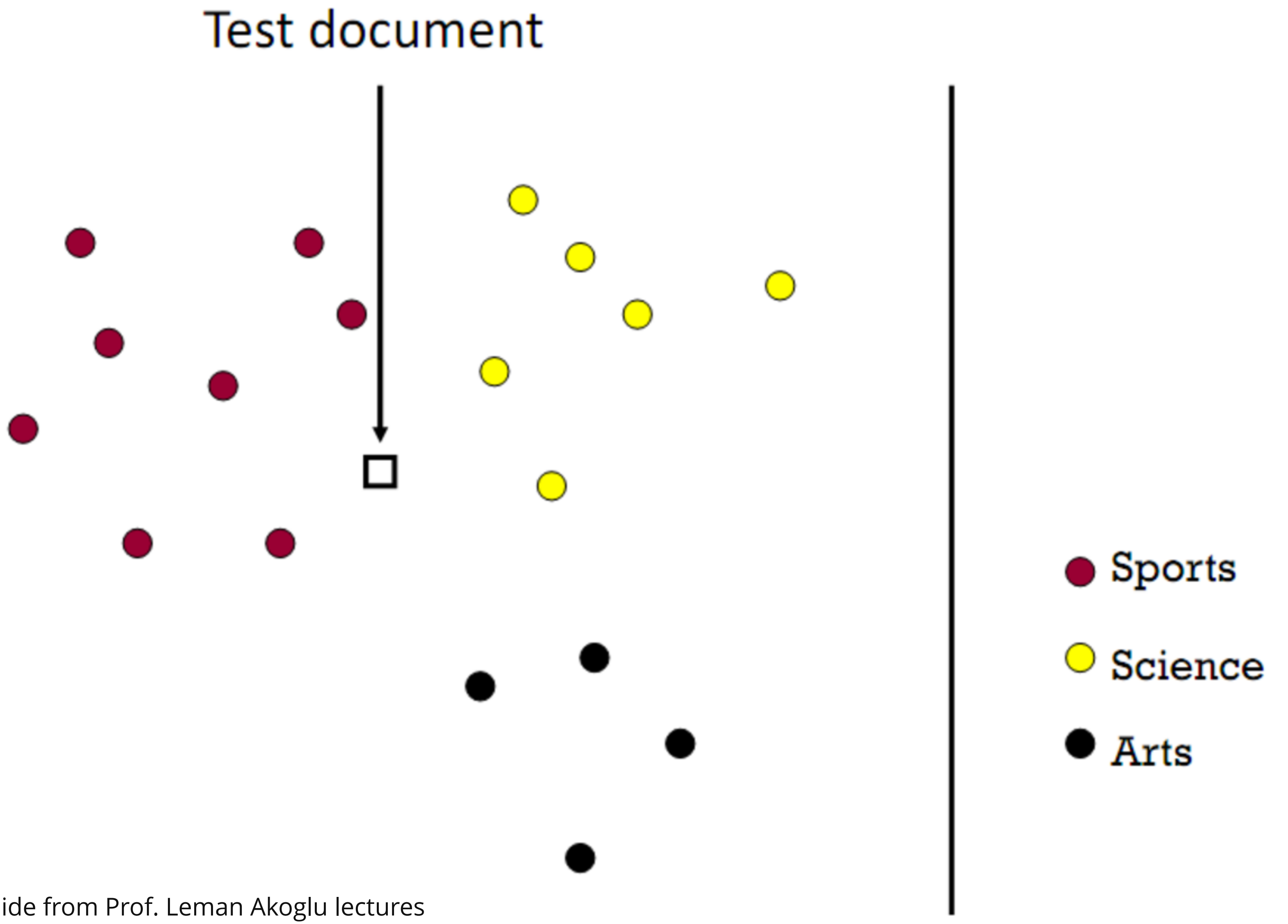
Benefits of GA

- GAs search from population not a single point.
- Discover new solutions by speculating on many combination of best possible solutions from within current pop.
- Useful in multi class high dimensionality which guarantees performance.
- A global optimum search method.

Approach

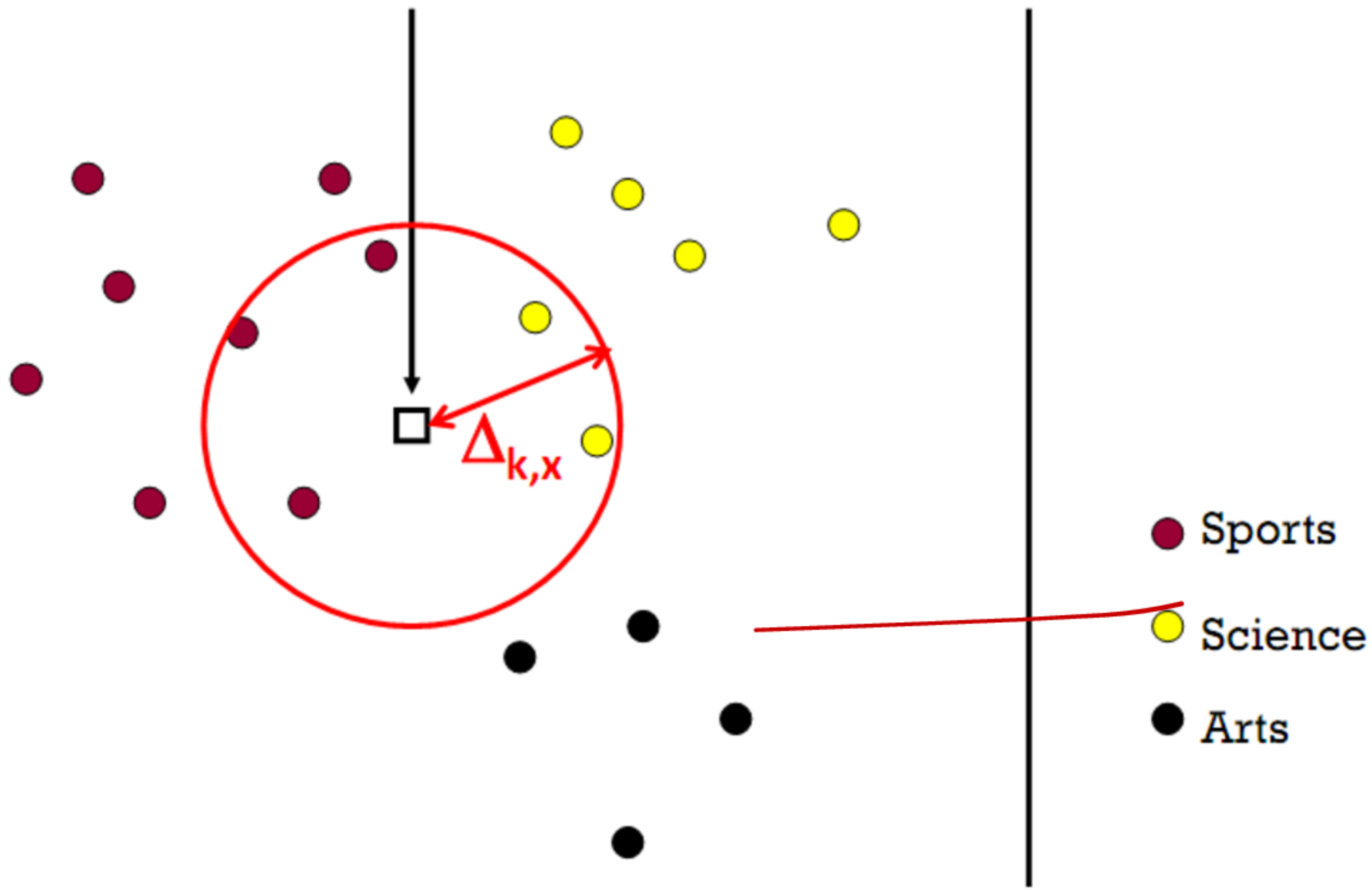
- GA/KNN hybrid approach
- GA/RULE approach

k-NN classifier



*Slide from Prof. Leman Akoglu lectures

k-NN classifier (k=5)



*Slide from Prof. Leman Akoglu lectures

What should we predict? … Average? Majority? Why?

k-NN classifier

- Optimal Classifier: $$f^*(x) = \arg\max_y P(y|x) = \arg\max_y p(x|y)P(y)$$

- k-NN Classifier: $$\widehat{f}_{kNN}(x) = \arg\max_y \widehat{p}_{kNN}(x|y)\widehat{P}(y) = \arg\max_y k_y$$ (Majority vote)

$$\widehat{p}_{kNN}(x|y) = \frac{k_y}{n_y \Delta_{k,x}}$$

k_y → # training pts of class y that lie within Δ_k ball

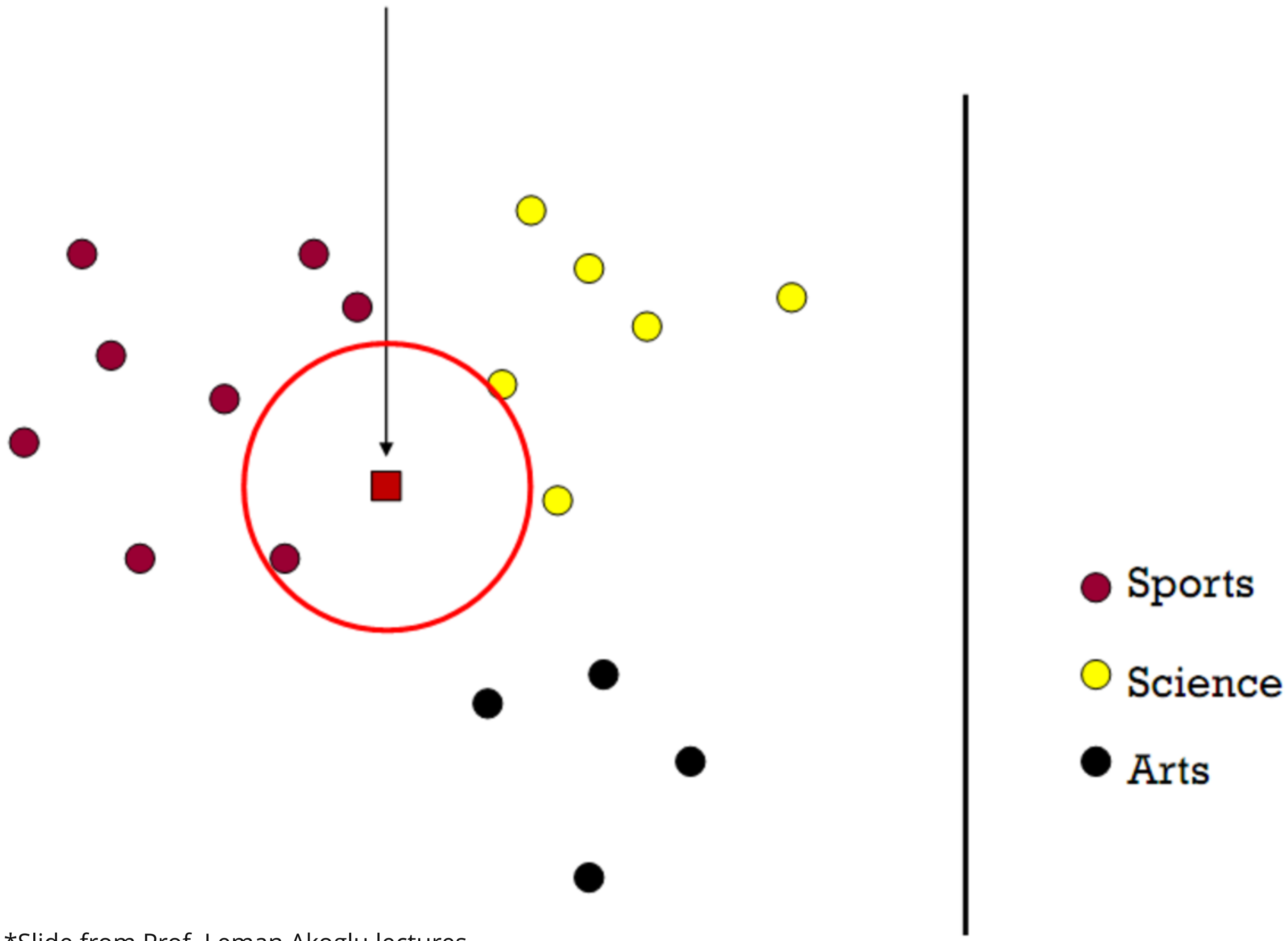
n_y → # total training pts of class y

$$\sum_y k_y = k$$

$$\widehat{P}(y) = \frac{n_y}{n}$$

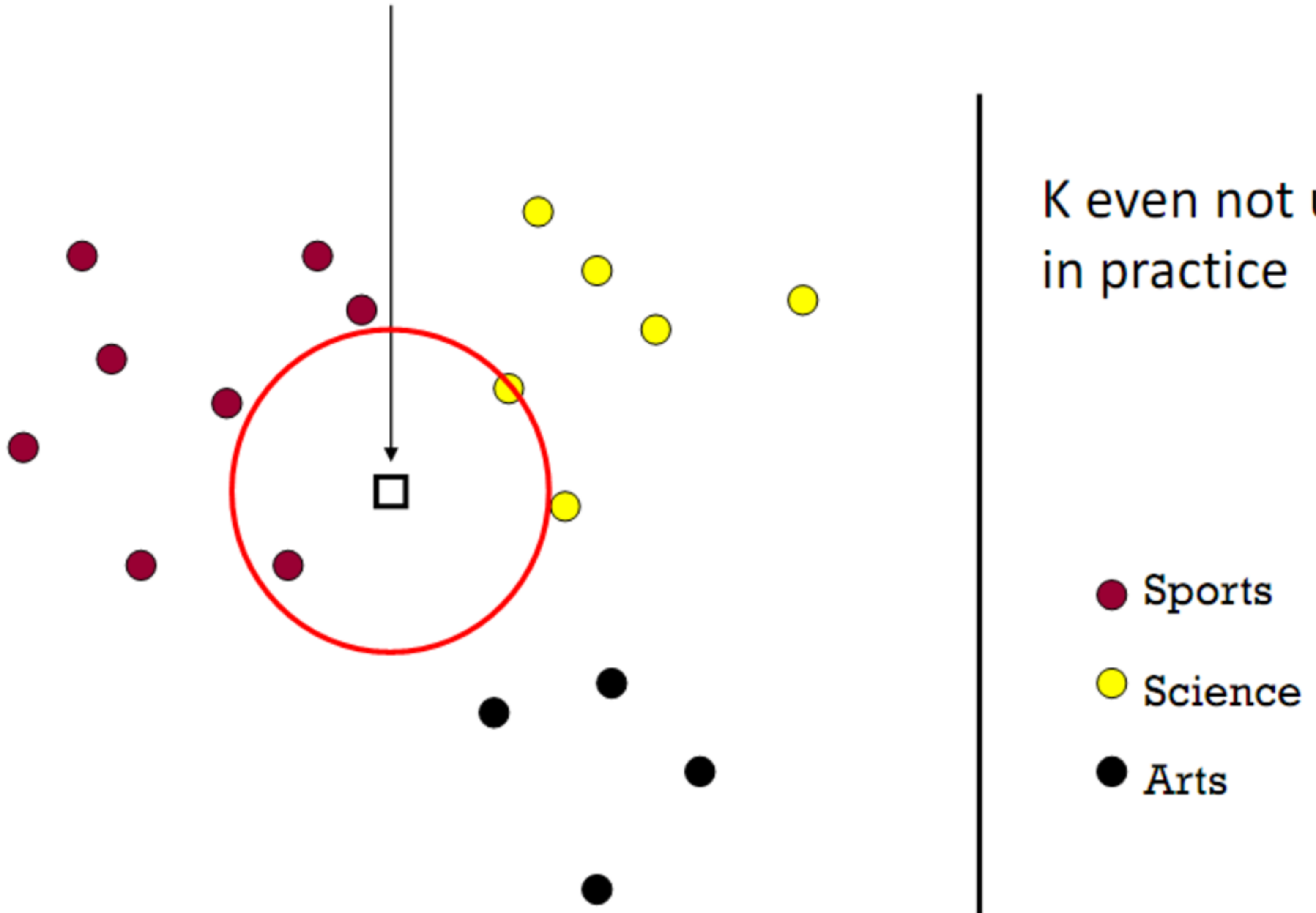
*Slide from Prof. Leman Akoglu lectures

1-Nearest Neighbor (kNN) classifier



*Slide from Prof. Leman Akoglu lectures

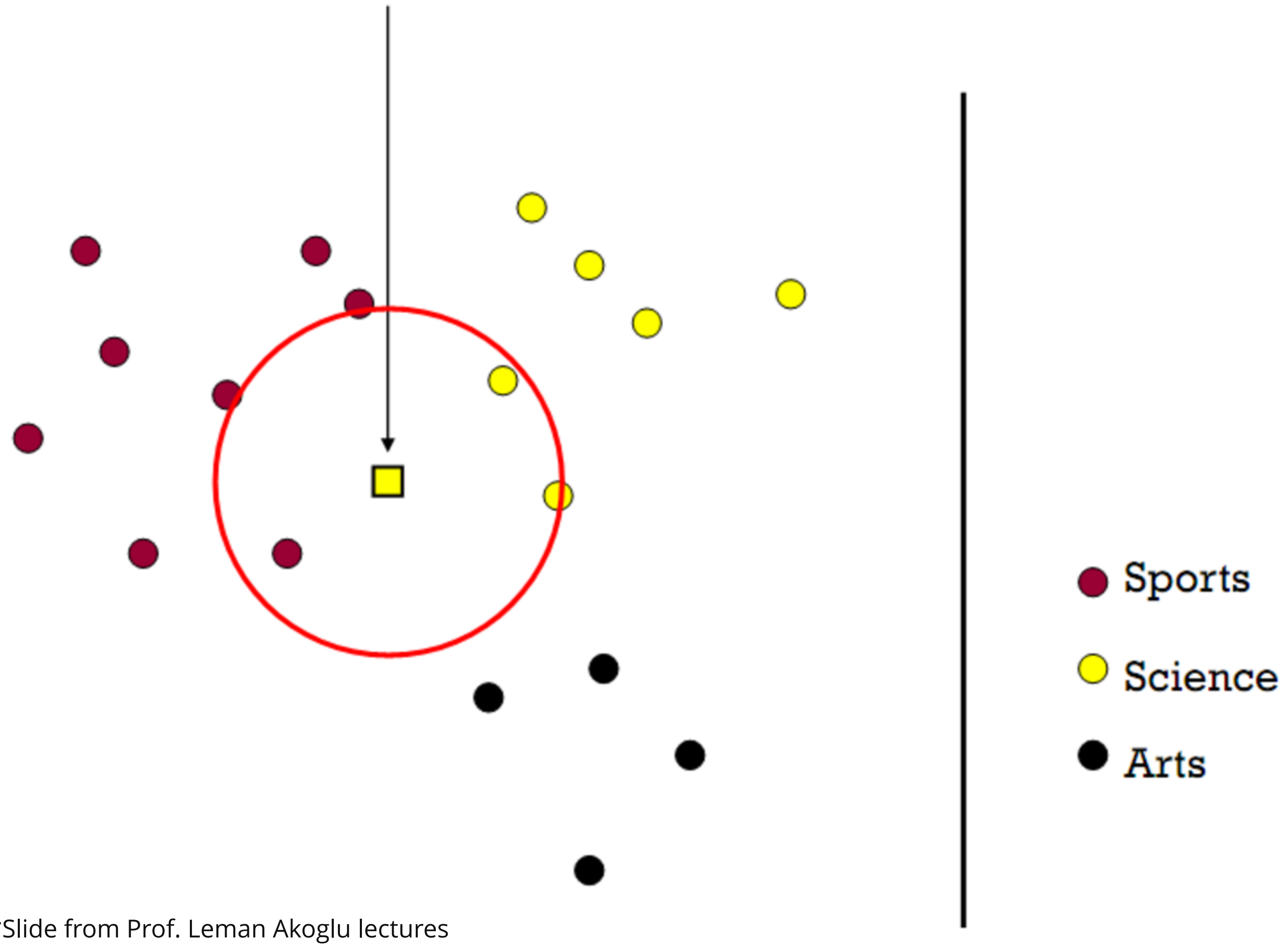
2-Nearest Neighbor (kNN) classifier



K even not used in practice

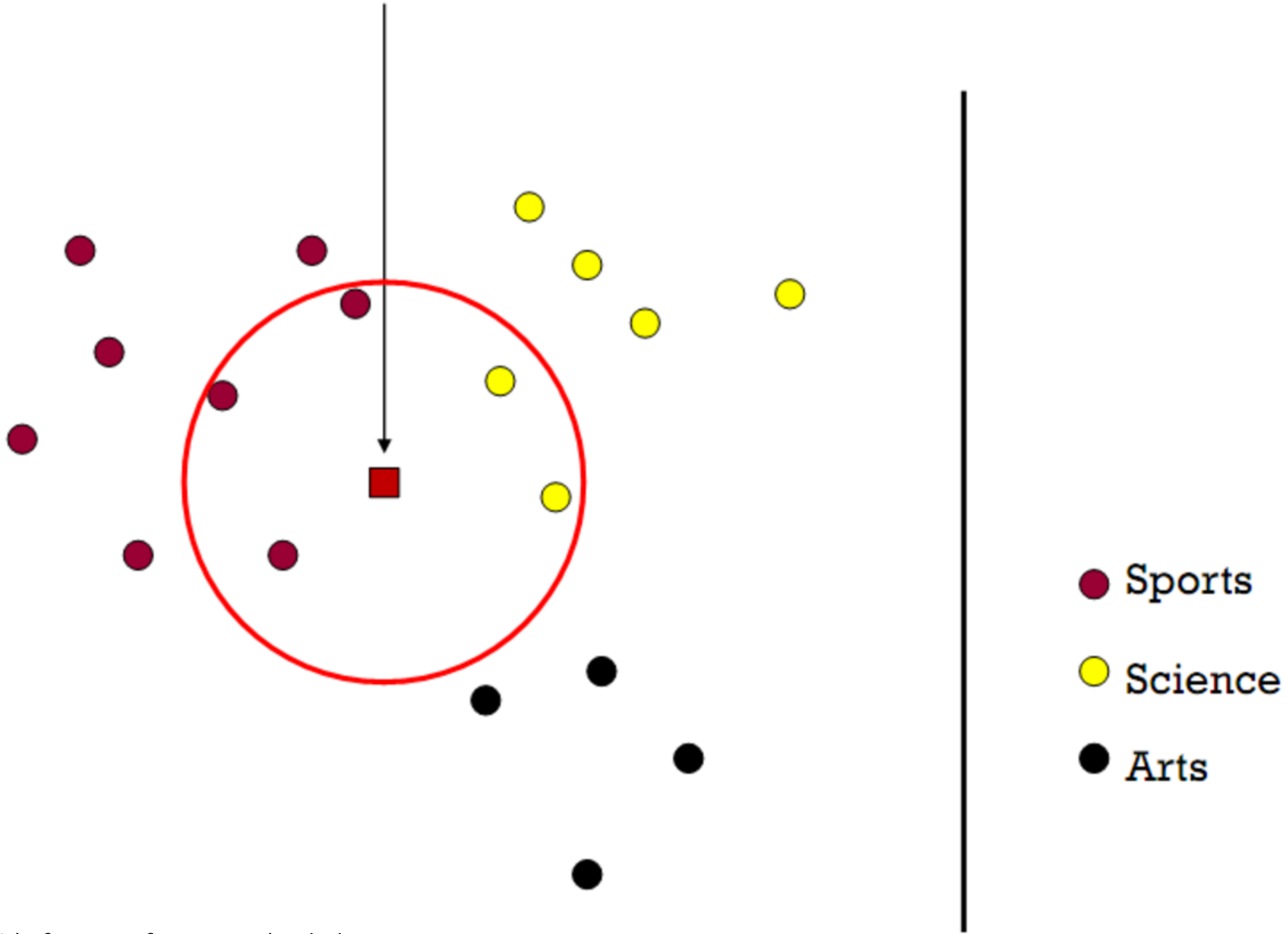
*Slide from Prof. Leman Akoglu lectures

3-Nearest Neighbor (kNN) classifier



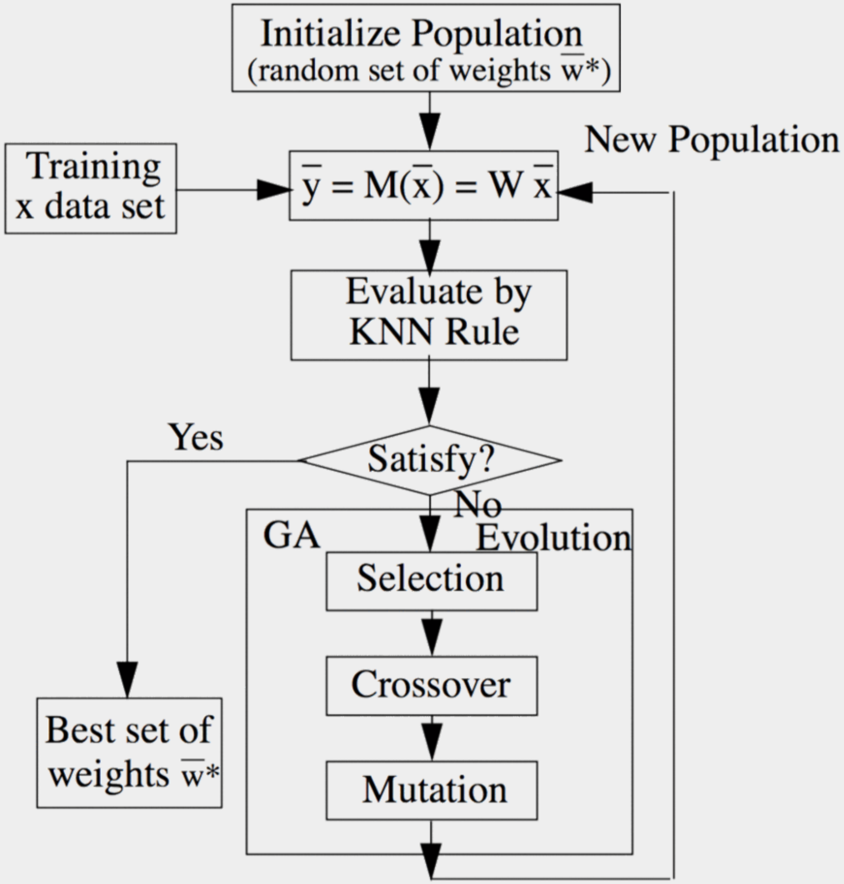
*Slide from Prof. Leman Akoglu lectures

5-Nearest Neighbor (kNN) classifier



*Slide from Prof. Leman Akoglu lectures

GA/KNN Approach



Non Linear Transformation

$$\bar{y} = M(\bar{x}) = W\bar{x}^*$$

where $\bar{x}^* = [x_1, x_2, \cdots, x_N, (x_i \bullet x_j)_1, (x_n \bullet x_m)_2, \cdots, (x_r \bullet x_s)_k]$

$$W = \begin{bmatrix} w_1 & 0 & \dots & 0 \\ 0 & w_2 & \dots & 0 \\ 0 & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & 0 \\ 0 & 0 & 0\ 0 & w_{N+k} \end{bmatrix}, \quad \text{or} \quad \bar{w}^* = [w_1, w_2, \dots, w_N, w_{N+1}, \dots, w_{N+k}],$$

- weight components that move towards 0 indicate that their corresponding features are not important for the discrimination task and are dropped in feature extraction
- resulting weights indicate the usefulness of a particular feature, its discriminatory power

Problems in Running GA

- Chromosome Encoding
  - For feature selection, $w_i = \{0,1\}$ $1 <= i <= N$, the chromosome encoding requires a single bit for each w_i, a component of weight vector w*
  - For feature extraction $w_i = [0,10]$ $1 <= i <= N$, weight's resolution is determined by number of bits used
- Normalization of Training Datasets
  - KNN evaluation is affected by scaling so we need to pre-normalize the training data to some range such as [0,1]
- Evaluation Criteria -- Fitness functions

Fitness Function

- Classifier is defined as a function from pattern space to feature space then to classification space

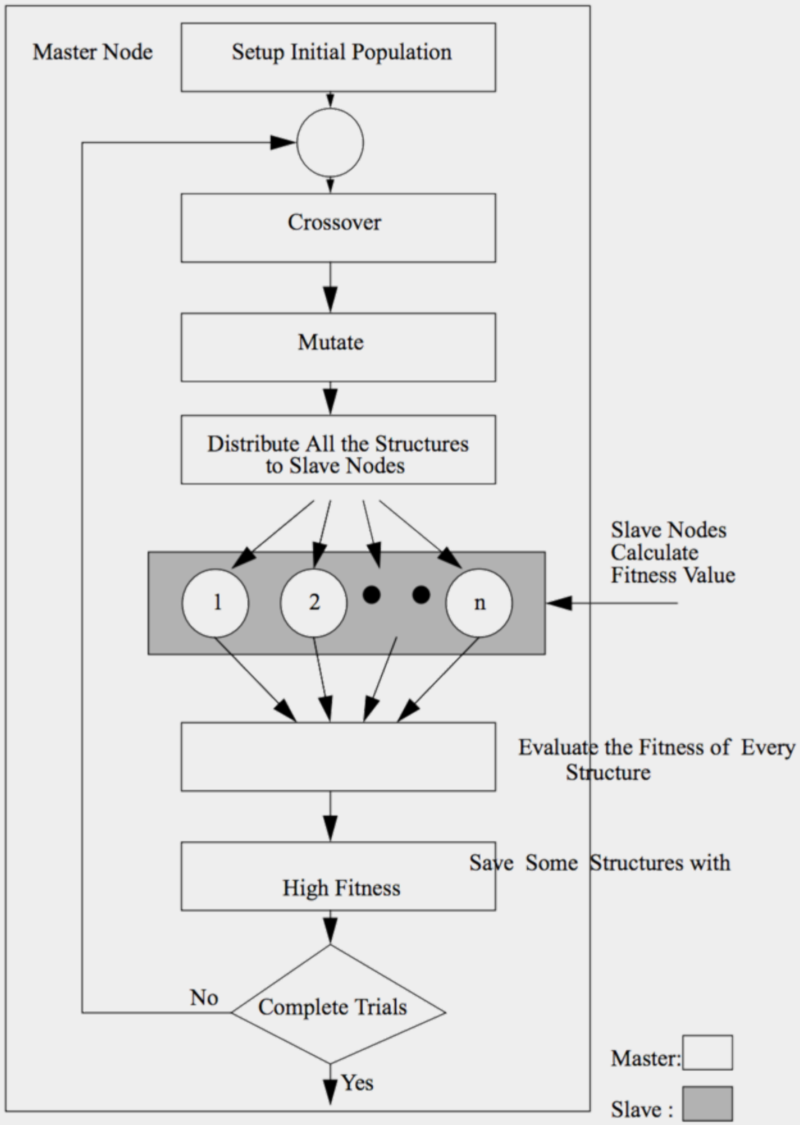
$$Fitness = J(w^*) = (TotPats - CorrectPats) / Totpats$$

$$Fitness = J(w^*) + T = \gamma \frac{(TotPats - CorrectPats)}{Totpats} + \delta \frac{nmin/K}{Totpats}$$

nmin is the cardinality of the near-neighbor minority set and *K* is the number of nearest neighbors. Constants *gamma* and *delta* are used for tuning the algorithm

Need for Speed

- GA can be implemented a parallel algorithm easily
- Most of the computational time is spent in evaluating the chromosome
- Idea is to distribute the evaluation of individuals in the population to several nodes (processors)



GA/KNN is not good enough

- Computational cost of the GA/KNN method is very high and requires parallel or distributed processing
- The very high computational cost comes from the problem of feature selection and extraction of high dimensionality data patterns
- Decrease computational cost without sacrificing performance by directly generating rules i.e, using a GA combined with a production (rule) system

GA/RULE Approach

- GA combined with a production rule system
- Focuses on the classification of binary feature patterns
- A single, integrated rule format
  - uses a known “training” sample set
  - result is a small set of “best” classification rules
- Directly manipulates a rule representation used for classification rather than transformation on KNN rule

Advantages

- Simpler to implement
- Requires substantially fewer computation cycles to achieve answers of similar quality.
- Accuracy of this method is significantly better than the classical KNN method
- “good” rules created for classifying “unknown” data
- Reveal those features which are important for the classification, based on the features used by the rules.

Classification rule format

<classification_rule> : : = <condition> : <message>

- A classification rule is a production rule which is used to make a decision assigning a pattern x to one of many classes
- The <condition> part of the rule is a string which consists of k class-attribute vectors
- Each vector consists of "n" elements, where "n" is the number of attributes (features) being used for the classification of that class
- Class-attribute vectors determine features to be used in this rule’s decision for classification i.e act as classification predicate.
- The <message> indicates the class into which the rule classifier places an input pattern matching against the feature vector

Classification Rule Format

<classification_rule> : : = <condition> : <message>

$$(Y_{11}, \ldots, Y_{1i}, \ldots, Y_{1n}), \ldots ,(Y_{j1}, \ldots, Y_{ji}, \ldots, Y_{jn}), \ldots ,(Y_{k1}, \ldots, Y_{ki}, \ldots Y_{kn}) : \omega$$

where $i = 1, 2, \cdots, n$; $j = 1, 2, \cdots, k$

ω is a variable whose value can be one of the 'k' classes.

The alphabet of the class-feature vector consists of 0, 1 and a "don’t care" character, i.e., $Y_{ji} \in \{0, 1, \#\}$

Training and Evaluation using GA

The training data consists of a training vector(record) X with a known classification label.

$\mathbf{X = [X_1 , X_2 , \cdot \cdot \cdot , X_i , \cdot \cdot \cdot , X_n]}$ where $i = 1, 2, \cdot \cdot \cdot , n$ and $X_i \in \{0,1\}$

Each rule is evaluated by matching it against the training data set.

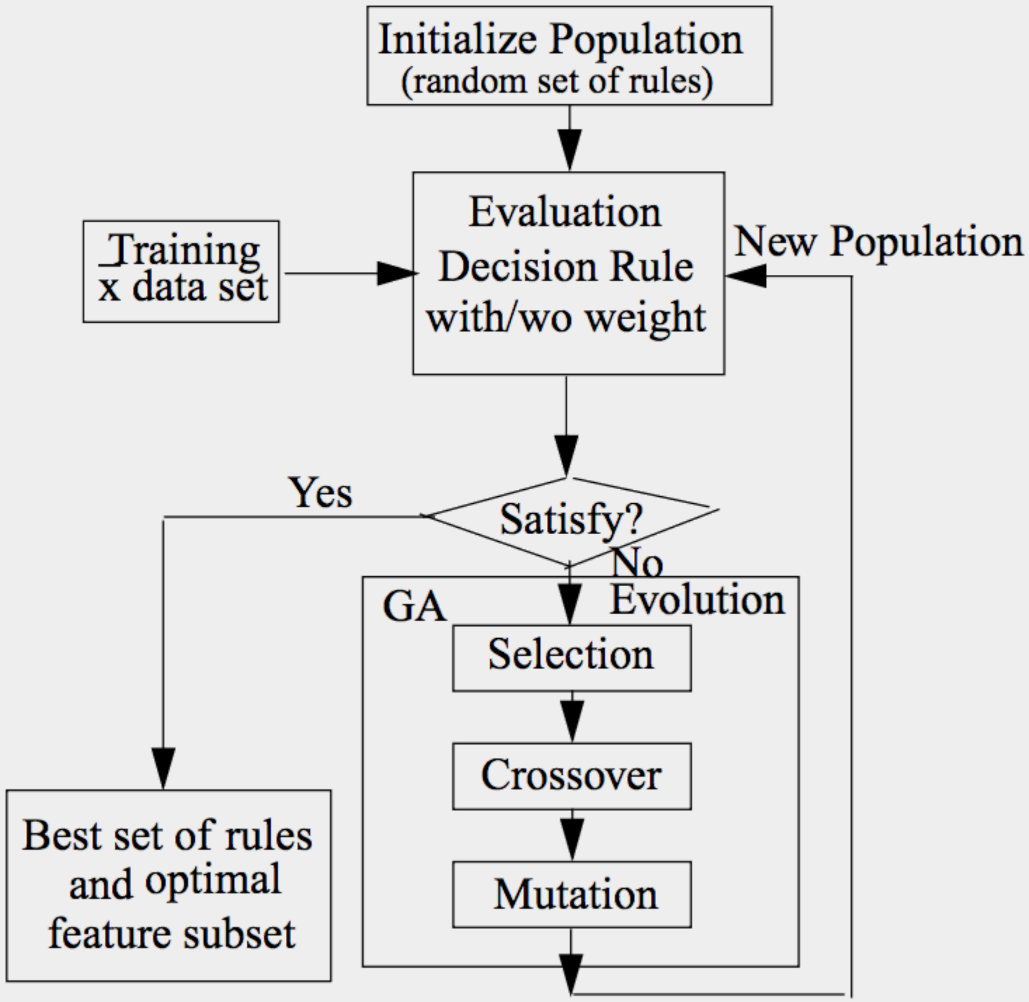
Every class-feature vector of the rule’s condition is compared with the training vector at every position

0 matches a 0, a 1 matches 1 and # don’t-care matches either 0 or 1

Training and Evaluation using GA

- For a training set with three classes, the training vector would be compared with the three vectors in each rule.
- The number of matching features in each vector is counted and the vector with the highest number of matches determines the class of the sample.
- Since the class of each training sample is already known, this classification can then be judged correct or not.
- Based on the accuracy of classification, the decision rule can be directly rewarded or punished.
- Based on this rule “strength”, the GA can evolve new rules.

GA/RULE Approach



Genetic Operators

- **Crossover**
  - Standard one-point crossover
- **Mutation**
  - Standard bit-modification
- The entire population (except for the best solution) is replaced each generation
- **Fitness Function**
  - Fitness = CorrectPats/TotPats + $\alpha * $ n_don’tcare/TotPats
  - This fitness function guides the GA to search out best rule sets as a multiobjective optimization problem.

Determining Invalid Features

For each rule, we arrange the class vectors as a matrix, then determine whether every class vector has the same value (1 or 0) or a don’tcare (#) at any position (column).

If they do, the n_don’tcare variable is incremented, as this feature is useless for classification

Summary

- GA plays an important role in classification and feature extraction for high dimensionality and multi-class data patterns
- An automatic pattern recognition method utilizing feedback information from classifier being used to change the decision space
- GA/KNN works by transforming the decision space and reducing its dimensionality
- GA/RULE works by modifying the decision rule using inductive learning and evolution
- GA can be used with other classifier to solve other complex pattern recognition problems