Genetic Algorithms & Supervised Learning

- ❑ **Supervised genetic learning** iteratively modifies the elements of the population
- ❑ Step 1. initialize a population P of elements
- ❑Step 2. apply a fitness function to evaluate each element currently in the population. The fitness function uses a set of training data to help with the evaluation. At each iteration, elements not satisfying the fitness criteria are eliminated from P.
- ❑Step 3. add new elements to P, to replace any elements eliminated in Step 2. The new elements are formed from previously deleted elements by applying crossover and mutation.
(selection may or may not be used)

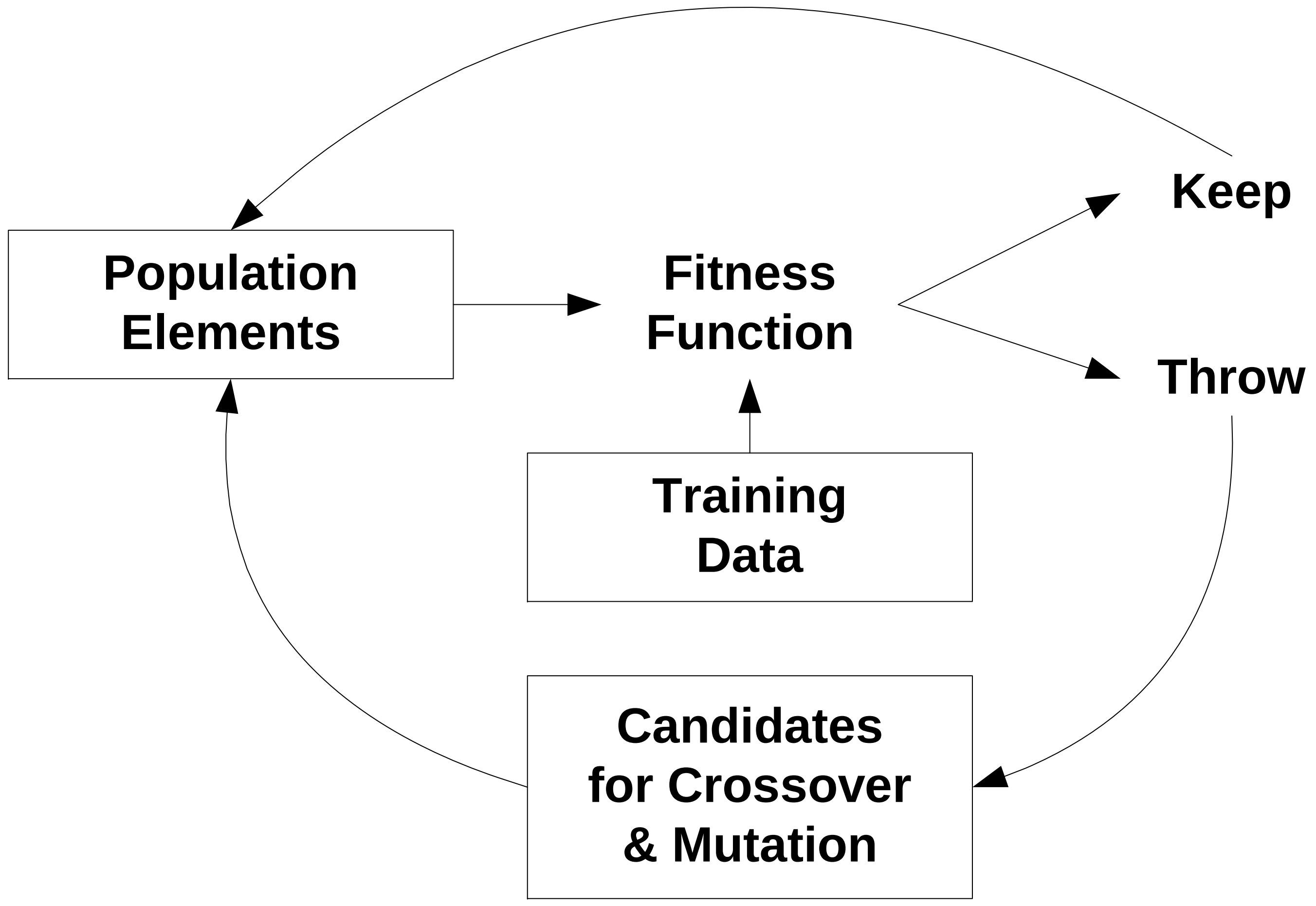


Figure 3.8 Supervised genetic learning

Example: credit card promotion database.
The aim is to create a model to be able to differentiate individuals who have accepted the life insurance promotion from those who have not.

Suppose that the initial population is represented by the 4 elements below.

We require that after each iteration, exactly two elements from each class, (life-insurance-promotion Y/N) remain in the population.

The income range ? value, means that this attribute is not important in the learning process.

Table 3.8 • **An Initial Population for Supervised Genetic Learning**

Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
1	20–30K	No	Yes	Male	30–39
2	30–40K	Yes	No	Female	50–59
3	?	No	No	Male	40–49
4	30–40K	Yes	Yes	Male	40–49

Genetic training involves repeated modification of the population by applying a fitness function (FF) to each element.

FF implementation: compare each element from P with the 6 elements of the training data below.

The fitness value (score) of an element E of P is defined as the **ratio** of the number of matches of the attribute values of E to its own class, to the number of matches to all training data from the competing class. (we add 1 to the denominator, to avoid div. by 0)

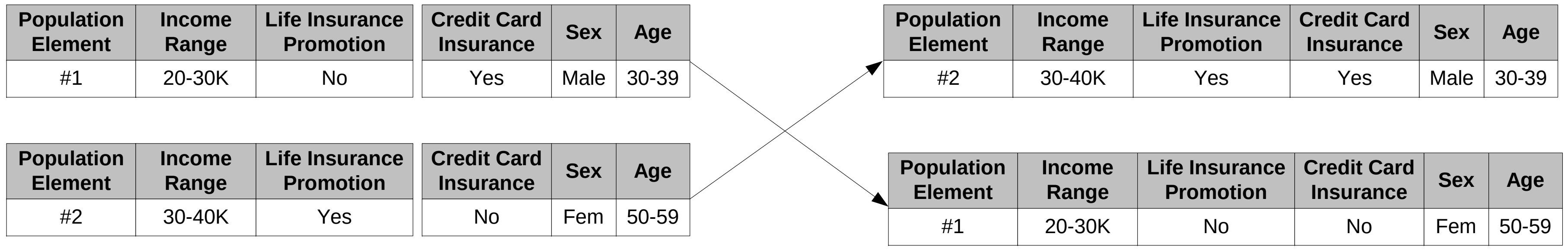
If the fitness score of an element E is not ≥ predefined value, then E is **eliminated** from P and becomes a candidate for crossover/mutation

Table 3.9 • **Training Data for Genetic Learning**

Training Instance	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
1	30–40K	Yes	Yes	Male	30–39
2	30–40K	Yes	No	Female	40–49
3	50–60K	Yes	No	Female	30–39
4	20–30K	No	No	Female	50–59
5	20–30K	No	No	Male	20–29
6	30–40K	No	No	Male	40–49

- Computation of the fitness score of element 1 (noted by E1).
- E1 is a member of the class (life-insurance-promotion N)
- Compute N = # of matches of E1 with the 3 members of the class (life-insurance-promotion N) in the training data.   N = 4
- 4 = (2 matches for income range=20-30) + (2 matches for sex=male)
- Compute M = # of matches of E1 with the 3 members of the class (life-insurance-promotion Y) in the training data.   M = 4 +1
- 4 = (1 match for credit card insurance=Y) + (1 match for sex=male) + (2 matches for Age=30-39)
- The fitness score of E1 is equal to N/M = 4/5 = 0.80
- The fitness scores of E2, E3, E4 are: 6/7, 6/5, 5/5 respectively.
- We eliminate E1, E2, as they have the lowest scores, for their respective classes.

Now we use the crossover operator, among E1 and E2.
We will use crossover after the Life Insurance Promotion attribute.



Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
#1	20-30K	No	Yes	Male	30-39

Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
#2	30-40K	Yes	No	Fem	50-59

Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
#2	30-40K	Yes	Yes	Male	30-39

Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
#1	20-30K	No	No	Fem	50-59

The resulting 2nd generation population is shown below.

The fitness scores of the two new elements are 7/5 and 6/4

Table 3.10 • **A Second-Generation Population**

Population Element	Income Range	Life Insurance Promotion	Credit Card Insurance	Sex	Age
1	20–30K	No	No	Female	50–59
2	30–40K	Yes	Yes	Male	30–39
3	?	No	No	Male	40–49
4	30–40K	Yes	Yes	Male	40–49

Figure 3.9 A crossover operation

- The algorithm stops when a termination condition is satisfied (e.g. evolve 500 generations)
- Once the algorithm terminates, we have a model for classifying new instances or predicting future outcomes.
- To use the model, we compare a new unknown instance with the elements of the final population P. Two options:

(1) give the unknown instance the same classification as the element of P to which it is most similar.

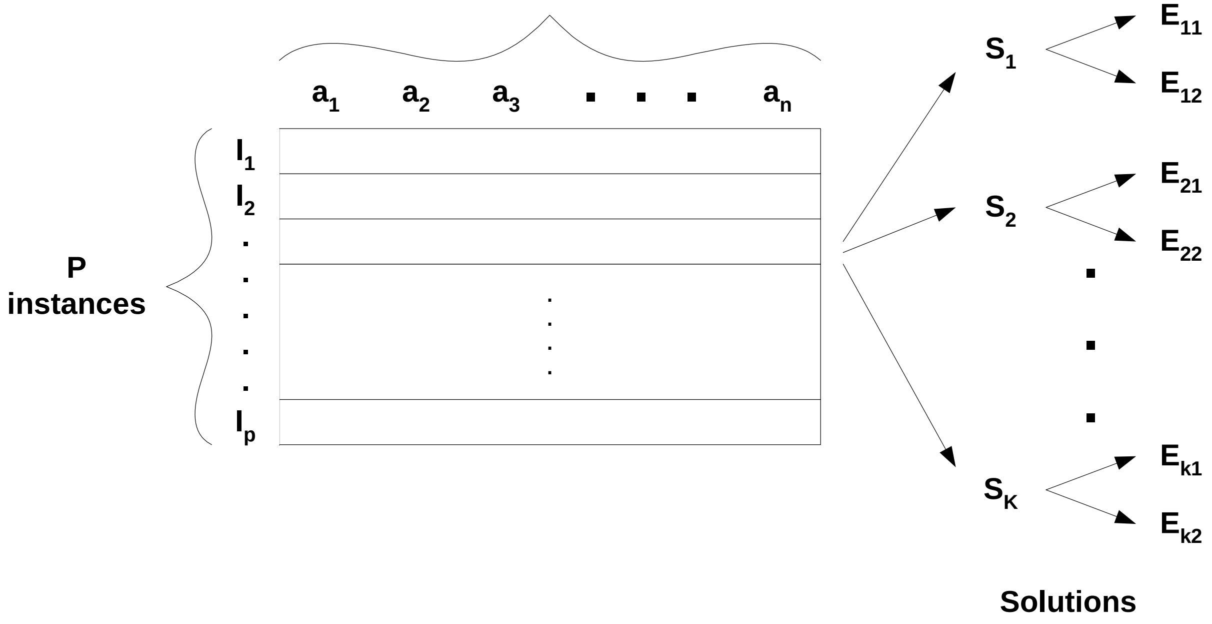
(2) choose randomly one of the m elements of P which are most similar to the new instance, and give its classification to the unknown instance

Genetic Algorithms & Unsupervised Clustering

- Use unsupervised genetic learning to develop a set of clusters for numerical data in n-dimensional space.

- Suppose there are P data instances, each consisting of n attribute values. Suppose we want to partition the P data in m clusters.

- The algorithm generates k possible solutions. Each solution contains m (n-dim.) data points, where each point is a best current representative for one of the m clusters. Illustrate with m = 2.



- **Crossover** operation: move data points from solution S_i to solution S_j.
- **Fitness function** for solution S_j: average Euclidean distance of the P data points from their closest element within S_j.
- FF computation for solution S_j: for each of the m elements in S_j compute the Euclidean distance from a data element and save the lowest such distance. Repeat for all P elements. The average of these P distances is the FF value
- Lower fitness function values, represent better fitness scores.
- At the end of genetic learning, the best of the k possible solutions is selected as the final solution.
- Each of the P data points is then assigned to the cluster associated with its closest element in the final solution.

Apply this algorithm to the 6 data points: assume m=2 clusters and k=3 solutions

11.31 = average of 6 smallest distances

2nd gener. crossover between S1 and S3

3rd gener. mutate S1 (exchange x,y coord)

Table 3.6 • **K-Means Input Values**

Instance	X	Y
1	1.0	1.5
2	1.0	4.5
3	2.0	1.5
4	2.0	3.5
5	3.0	2.5
6	5.0	6.0

Table 3.11 • **A First-Generation Population for Unsupervised Clustering**

	S_1	S_2	S_3
Solution elements (initial population)	(1.0,1.0) (5.0,5.0)	(3.0,2.0) (3.0,5.0)	(4.0,3.0) (5.0,1.0)
Fitness score	11.31	9.78	15.55
Solution elements (second generation)	(5.0,1.0) (5.0,5.0)	(3.0,2.0) (3.0,5.0)	(4.0,3.0) (1.0,1.0)
Fitness score	17.96	9.78	11.34
Solution elements (third generation)	(5.0,5.0) (1.0,5.0)	(3.0,2.0) (3.0,5.0)	(4.0,3.0) (1.0,1.0)
Fitness score	13.64	9.78	11.34