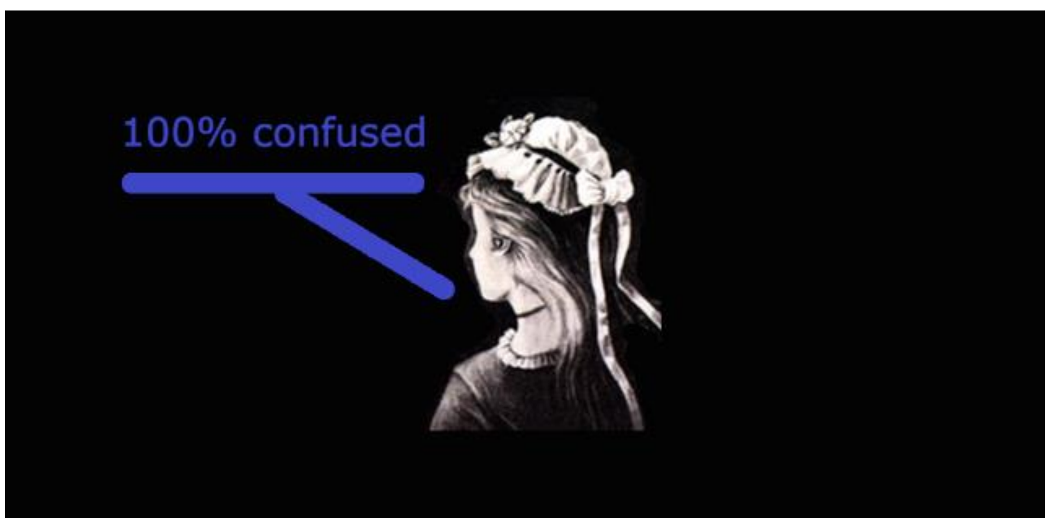
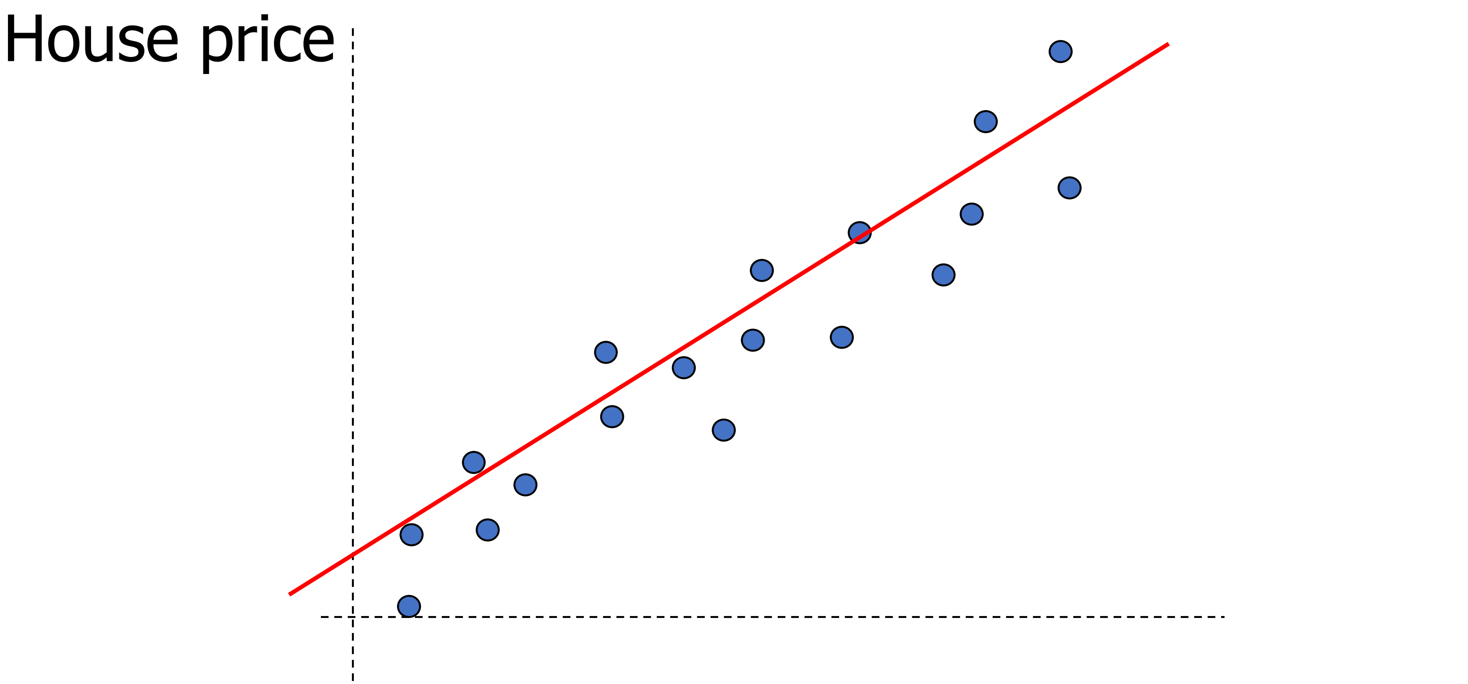
CISC 372
Advanced Data Analytics
L2- Review

https://l1nna.com/course/cisc372/



Linear Regression

- House Price Prediction (a prediction problem)

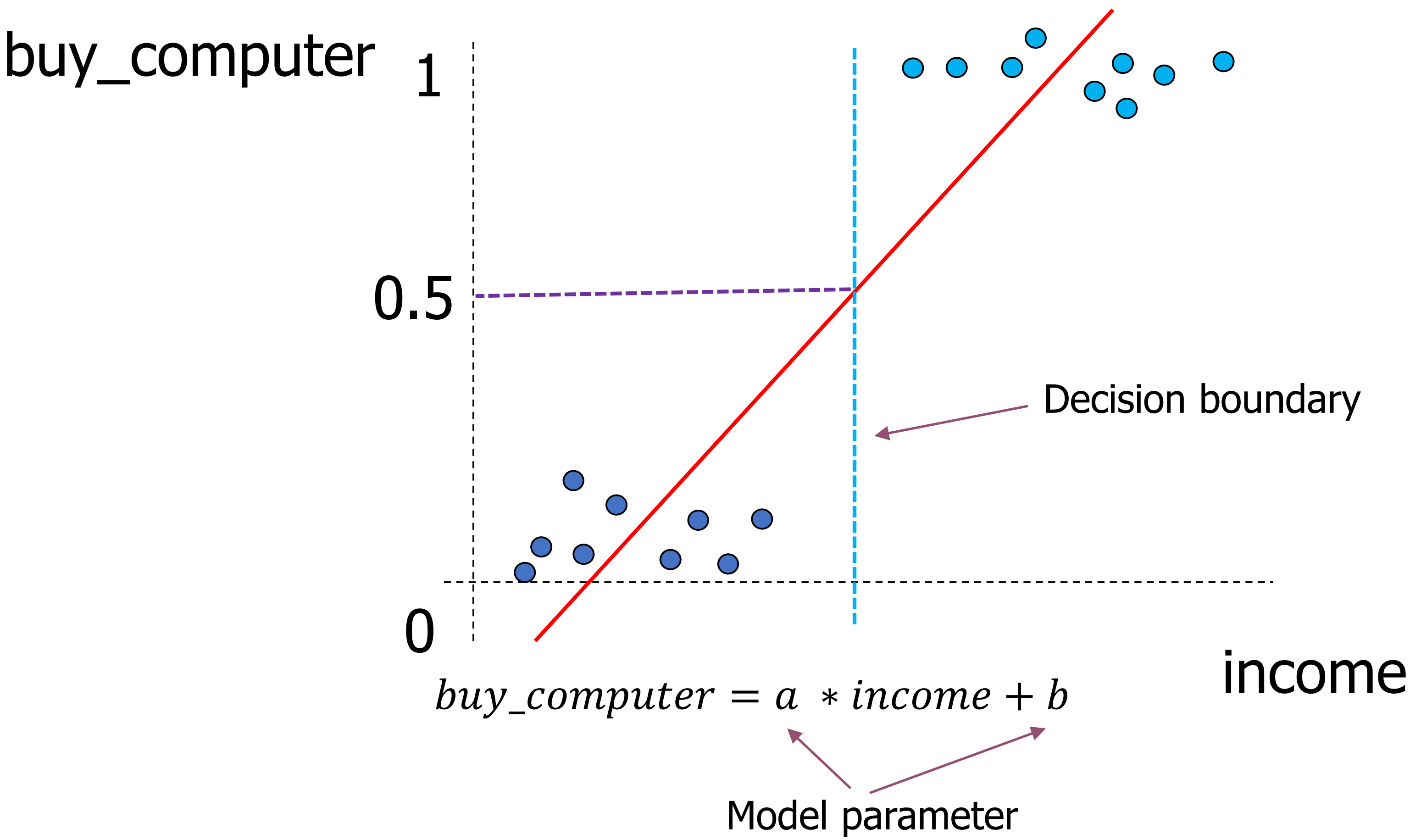


$$price = a * size + b$$

Model parameter

Linear Regression

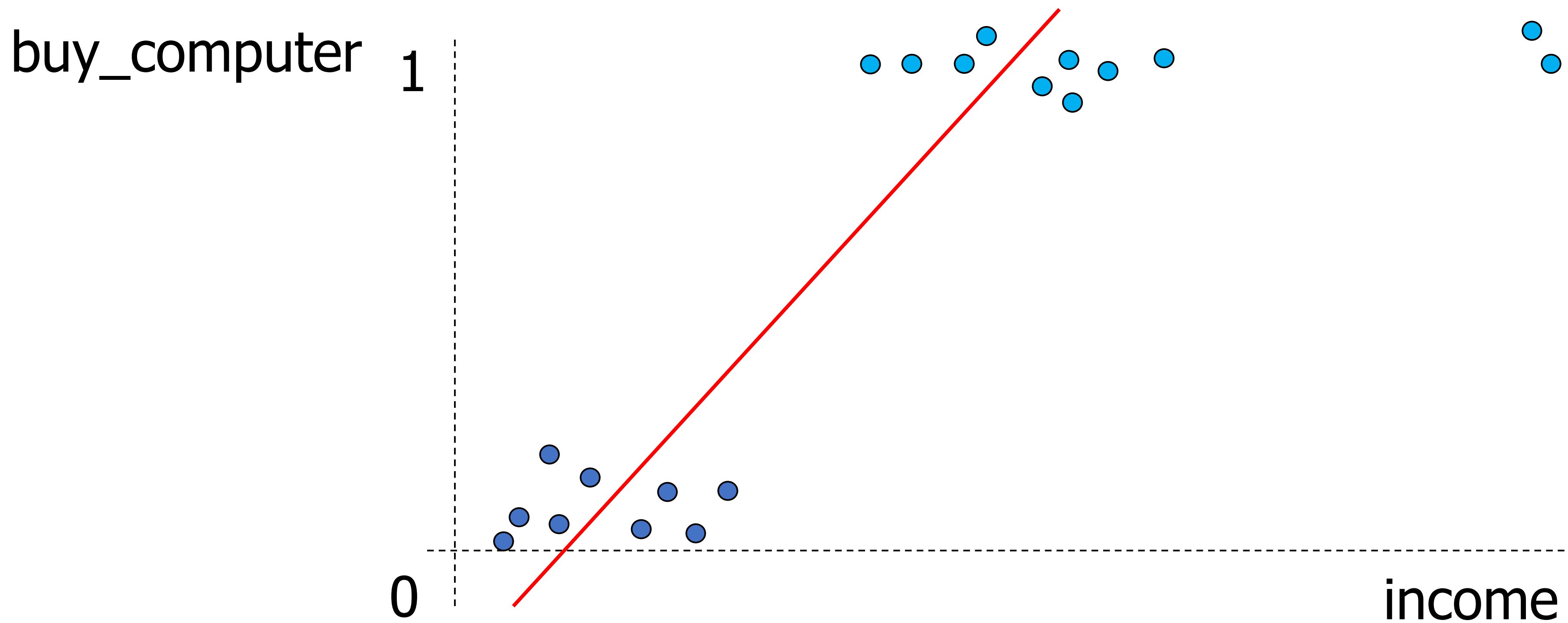
- Linear Regression for Classification



$$buy\_computer = a * income + b$$

Linear Regression

- Linear Regression for Classification

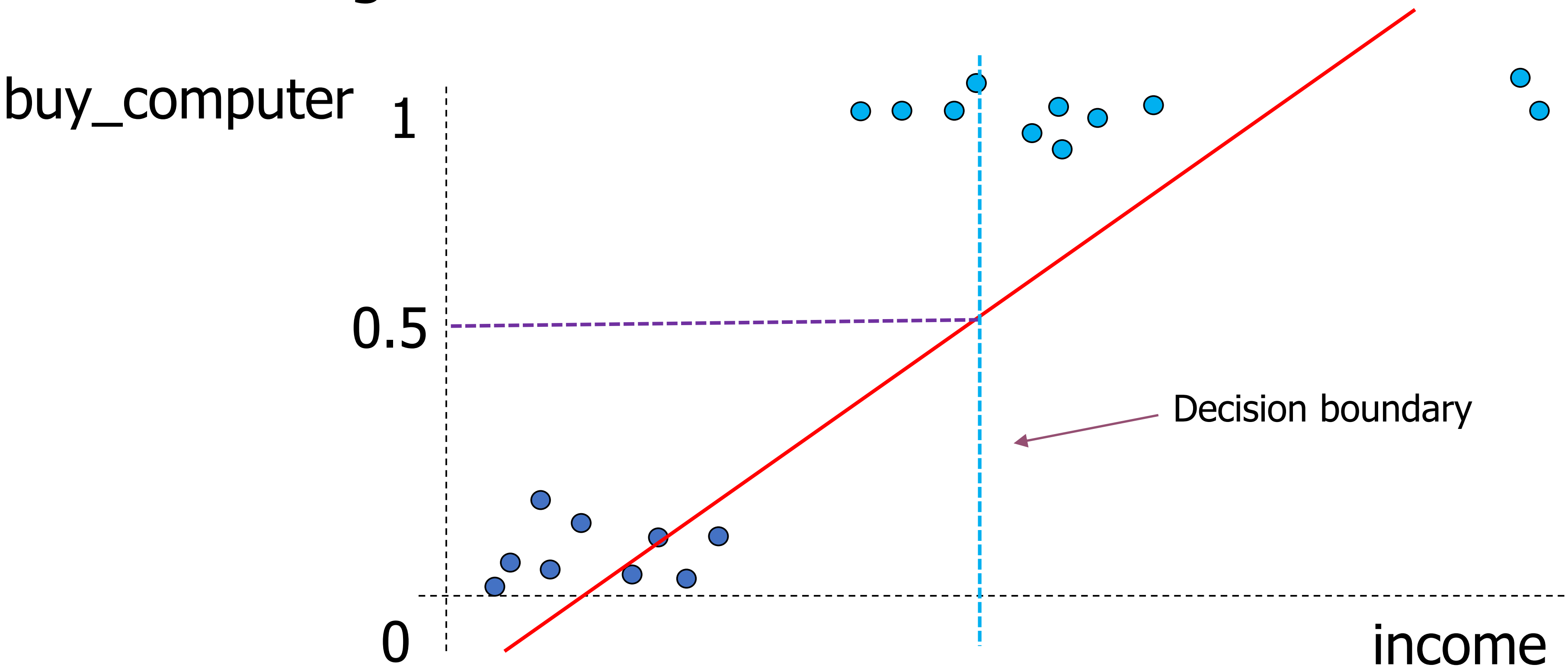


$$buy\_computer = a * income + b$$

Model parameter

Linear Regression

- Linear Regression for Classification

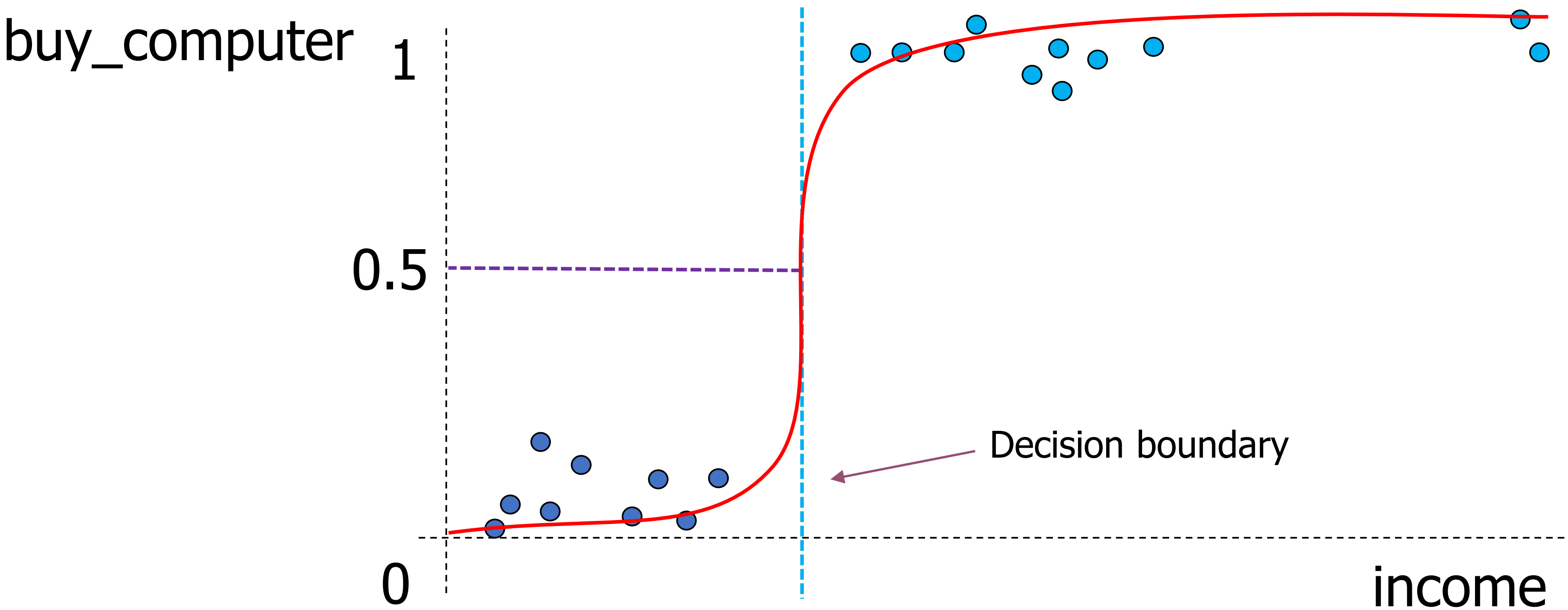


$$buy\_computer = a \ * income + b$$

Model parameter

Logistic Regression

- Logistic Regression for Classification



$$buy\_computer = logistic(a * income + b)$$

Model parameter

Logistic Regression

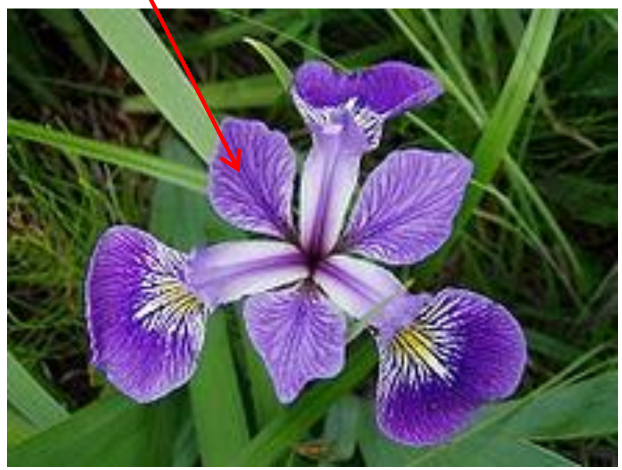
- Logistic Regression for Classification
  - The logistic function rescales its input to between 0 and 1.
  - The logistic function transforms a straight curve into a ‘S’-shape curve.
  - By default, it only handles binary classification task.
  - The prediction can be interpreted as probability
  - For multinominal class attribute, we use generalized linear regression with multinominal family (skip, but available in RapidMiner).

Iris Dataset – Binary Classification

Feature:
- Petal.Length
- Petal.With

Species:
Versicolor : 1
Virginica : 0

Petal



Iris versicolor

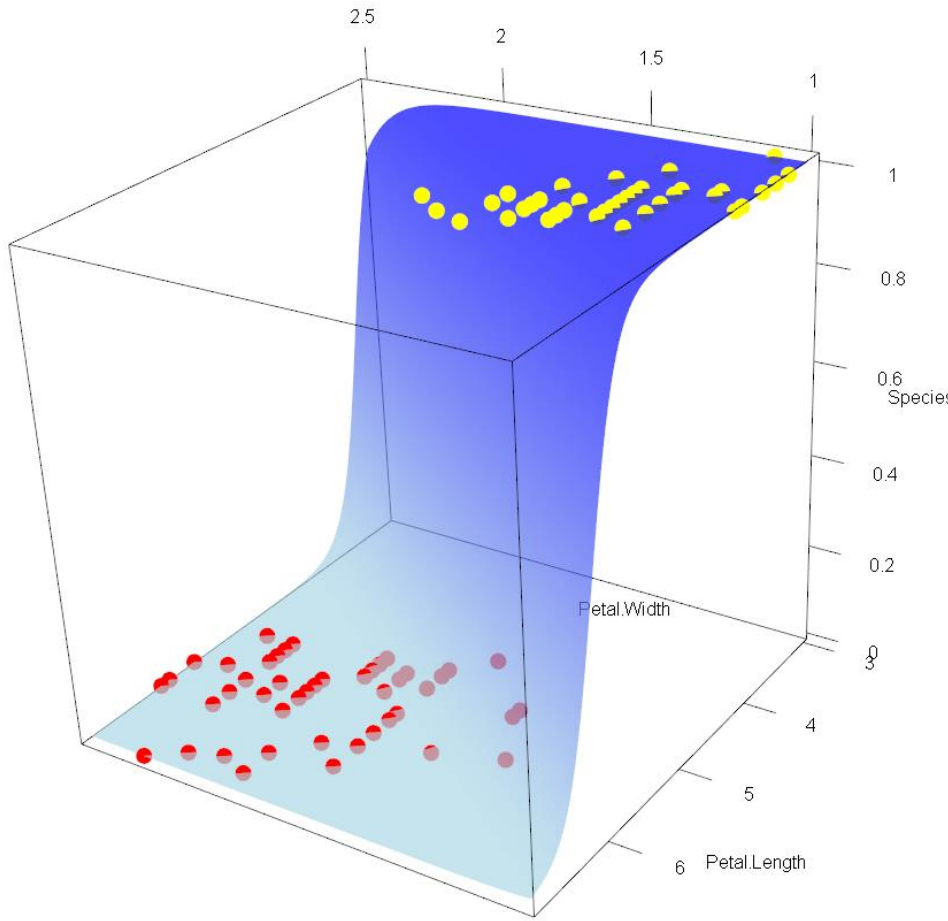
Iris virginica

	Petal.Width	Petal.Length	Species
99	1.1	3.0	1
100	1.3	4.1	1
101	2.5	6.0	0
102	1.9	5.1	0

Iris Dataset – Logistic Regression

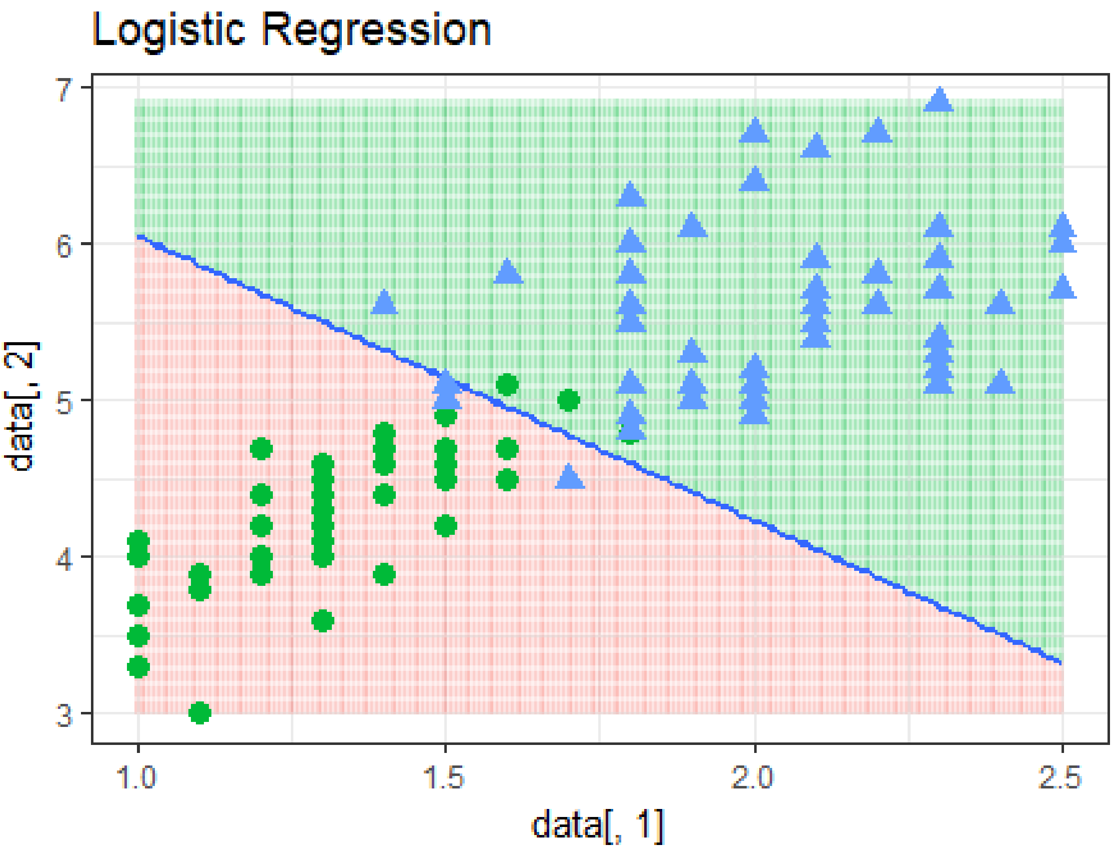
Feature:
- Petal.Length
- Petal.With

Species:
Versicolor : 1
Virginica : 0



$$prob(species) = logistic(a_0 * Petal.Width + a_1 * Petal.Length + b)$$

Iris Dataset – Logistic Regression



Feature:
- Petal.Length
- Petal.With

Species:
Versicolor : 1
Virginica : 0

Iris Dataset – Logistic Regression

$$prob(species) = logistic(a_0 * Petal.Width + a_1 * Petal.Length + b)$$

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -45.272     13.610  -3.327 0.000879 ***
Petal.Width    10.447      3.755   2.782 0.005405 **
Petal.Length    5.755      2.306   2.496 0.012565 *
```

- Logistic Regression for Classification
  - Coefficient indicates how much the probability will increase/decrease if you increase/decrease the corresponding feature.
  - P-value (Pr) indicates if the correlation is significant between a feature and the class attribute.

Iris Dataset – Logistic Regression

- We can add more feature combinations by introducing polynomial terms.

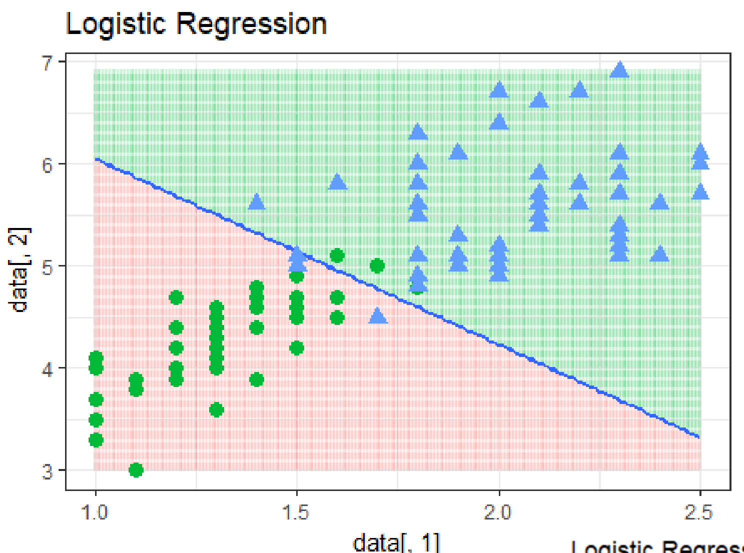
M1: $$prob(species) = logistic(a_0 * Petal.Width + a_1 * Petal.Length + b)$$

M2: $$prob(species) = logistic(a_0 * Petal.Width + a_1 * Petal.Length + \\ a_2 * (Petal.Width)^2 + a_3 * (Petal.Length)^2 + b)$$

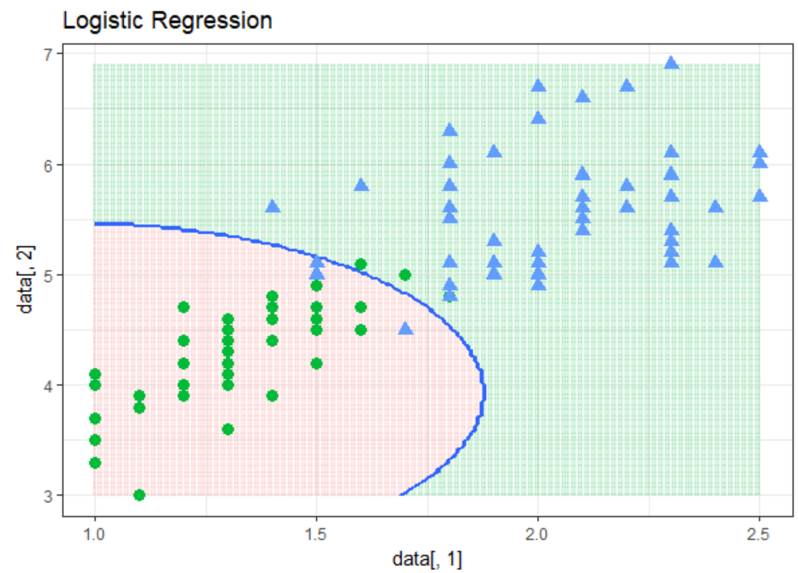
M3: $$\begin{gathered} prob(species) = logistic(a_0 * Petal.Width + a_1 * Petal.Length + \\ a_2 * (Petal.Width)^2 + a_3 * (Petal.Length)^2 \\ a_4 * (Petal.Width)^3 + a_5 * (Petal.Length)^3 \\ a_6 * (Petal.Width)^4 + a_7 * (Petal.Length)^4 \\ a_8 * (Petal.Width)^5 + a_9 * (Petal.Length)^5 \\ a_{10} * (Petal.Width)^6 + a_{11} * (Petal.Length)^6 \\ \dots + b \,) \end{gathered}$$

Iris Dataset – Logistic Regression

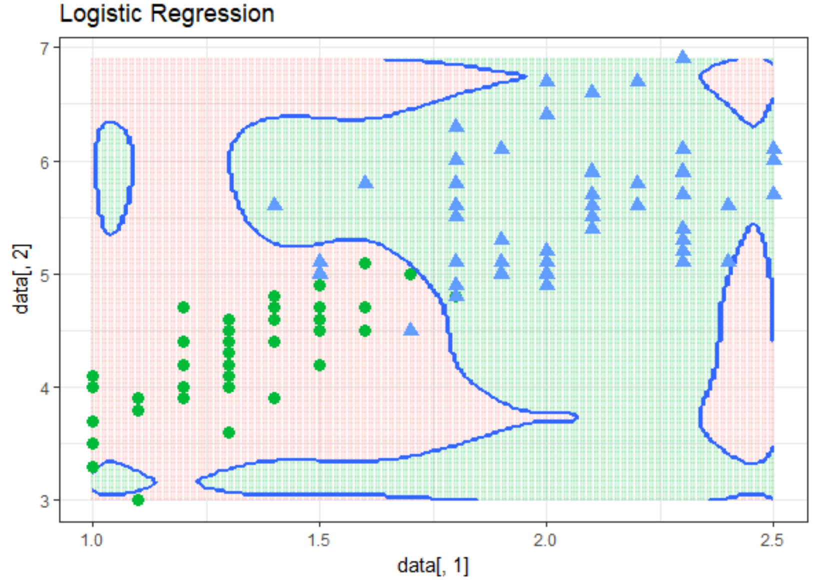
M1:



M2:



M3:



Logistic Regression

- Logistic Regression for Classification
  - As the complexity of the model increases, the chance of over-fitting the training data increases.
- Solution Regularization:
  - Penalize the norm of the parameters. (i.e. scale)

$$
\begin{aligned}
price = logistic(&a_0 * Petal.Width + a_1 * Petal.Length + \\
&a_2 * (Petal.Width)^2 + a_3 * (Petal.Length)^2 \\
&a_4 * (Petal.Width)^3 + a_5 * (Petal.Length)^3 \\
&a_6 * (Petal.Width)^4 + a_7 * (Petal.Length)^4 \\
&a_8 * (Petal.Width)^5 + a_9 * (Petal.Length)^5 \\
&a_{10} * (Petal.Width)^6 + a_{11} * (Petal.Length)^6 \\
&\ldots + b\ )
\end{aligned}
$$

Regularization

- L-1 Regularization (Lasso)
  - Panelize the absolute norm of parameters.
  - $\sum_{i=1}^{k} |a_k|$
  - Encourage model sparsity (turn on/off some features)
- L-2 regularization (Ridge)
  - Panelize the squares of parameters.
  - $\sum_{i=1}^{k} (a_k)^2$
  - Make the parameters small in scale.
  - Make the decision boundary less curved.

- Notebook Intro

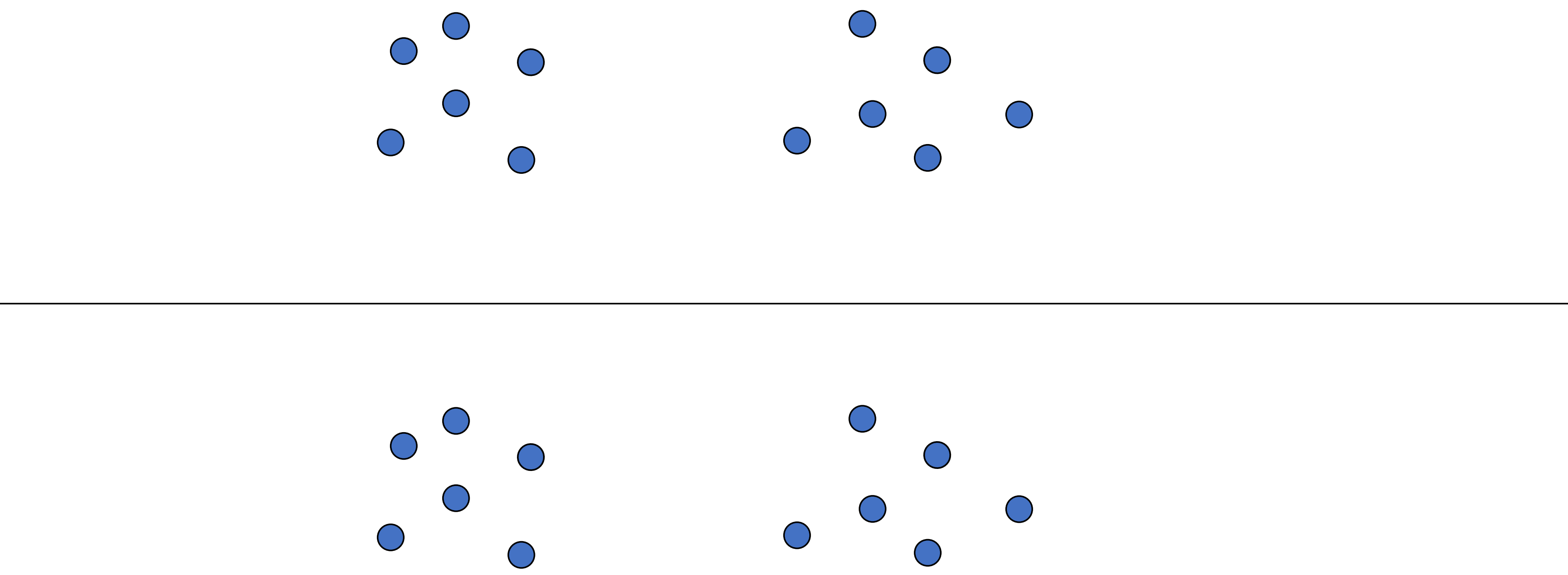
Partitioning Algorithms: Basic Concept

- Partitioning method: Partitioning a database ***D*** of ***n*** objects into a set of ***k*** clusters, such that the sum of squared distances is minimized (where c_i is the centroid or medoid of cluster C_i)

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in C_i} (dist(p, c_i))^2$$

- Given *k*, find a partition of *k clusters* that optimizes the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - Heuristic methods: *k-means* and *k-medoids* algorithms
  - *k-means* (MacQueen'67, Lloyd'57/'82): Each cluster is represented by the center of the cluster
  - *k-medoids* or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects in the cluster

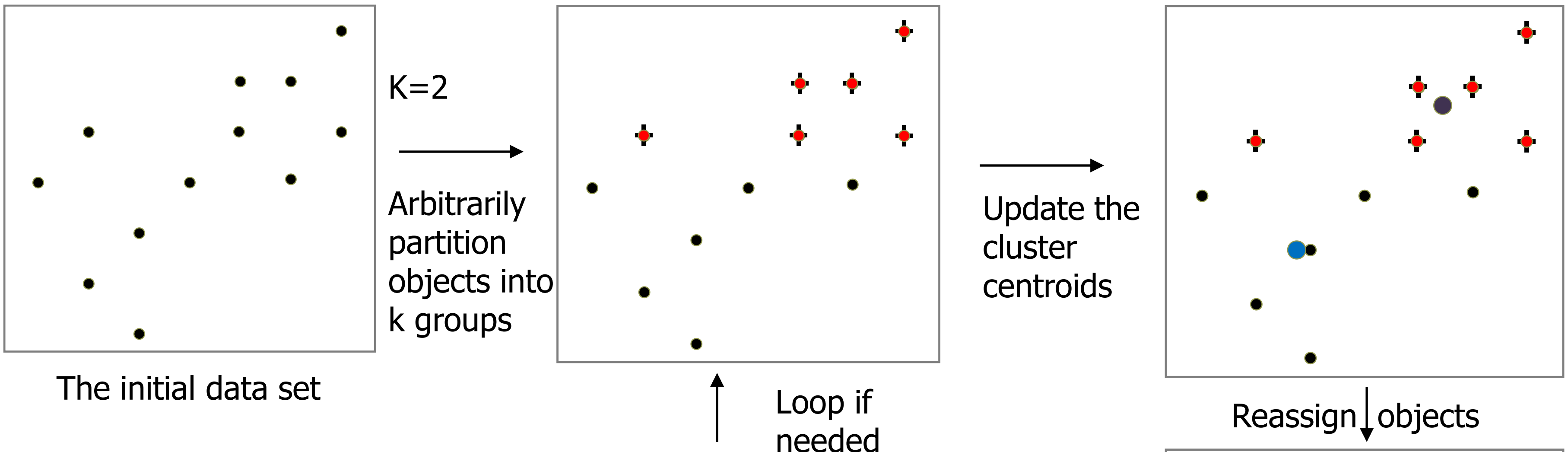
Partitioning Algorithms: Basic Concept

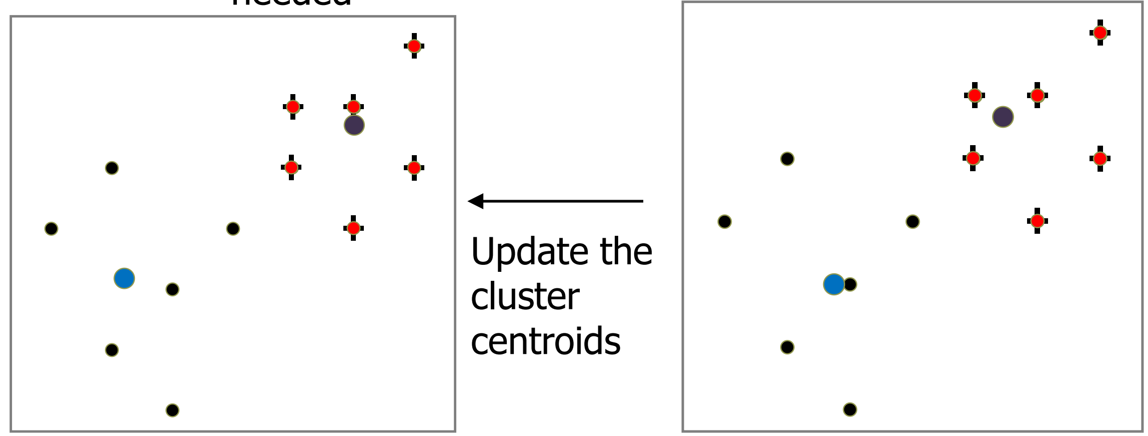


The *K-Means* Clustering Method

- Given *k*, the *k-means* algorithm is implemented in four steps:
  1. Partition objects into *k* nonempty subsets
  2. Compute seed points as the centroids of the clusters of the current partitioning (the centroid is the center, i.e., *mean point*, of the cluster)
  3. Assign each object to the cluster with the nearest seed point
  4. Go back to Step 2, stop when the assignment does not change

An Example of *K-Means* Clustering

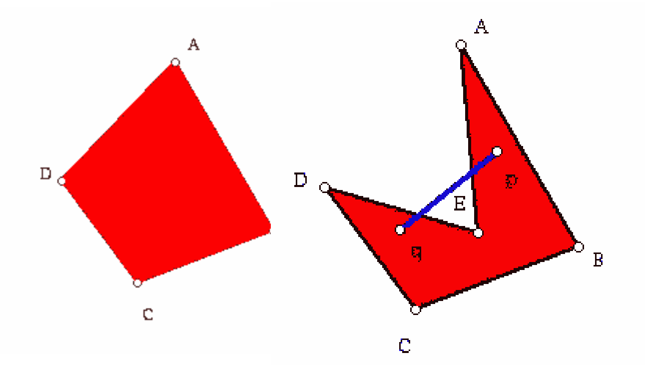




- Partition objects into *k* nonempty subsets
- Repeat
  - Compute centroid (i.e., mean point) for each partition
  - Assign each object to the cluster of its nearest centroid
- Until no change

Comments on the *K-Means* Method

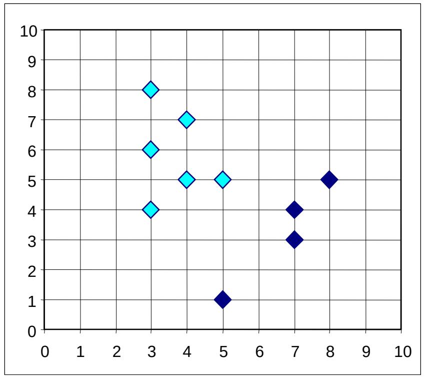
- Strength: *Efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
- Comment: Often terminates at a *local optimal*

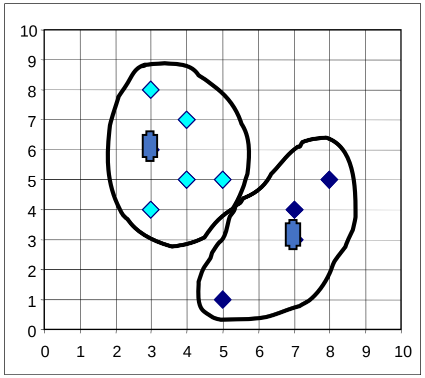


- Weakness
  - Applicable only to objects in a continuous n-dimensional space
    - Using the k-modes method for categorical data
    - In comparison, k-medoids can be applied to a wide range of data
  - Need to specify *k,* the *number* of clusters, in advance (there are ways to automatically determine the best k (see Hastie et al., 2009)
  - Sensitive to *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

What Is the Problem of the K-Means Method?

- The k-means algorithm is sensitive to outliers !
  - Since an object with an extremely large value may substantially distort the distribution of the data
- K-Medoids: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster





PAM (Partitioning Around Medoids) (1987)

- Find *representative* objects, called <u>medoids</u>, in clusters
- Use real object to represent the cluster
  - arbitrarily select ***k*** representative objects
  - repeat
    - assign each remaining object to nearest representative object o_j
    - randomly select a non-representative object o_{random}
    - compute the total cost, *TC*, of swapping o_j with o_{random}
    - if $TC < 0$, ***i*** is replaced o_j by o_{random}
  - until there is no change

PAM: A Typical K-Medoids Algorithm