K Nearest- Neighbor Classifier

Patricia Hoffman, PhD

Most of the slides are from

Tan, Steinbach, and Kumar - Introduction to Data Mining

K Nearest-Neighbor Classifier

- **Slides from Tan, Steinbach, Kumar**
  http://www-users.cs.umn.edu/~kumar/dmbook/dmslides/chap5_alternative_classification.pdf
- **Class Package**
  http://cran.r-project.org/web/packages/class/class.pdf
- **e1071 Package**
  http://cran.r-project.org/web/packages/e1071/e1071.pdf
- **The Elements of Statistical Learning**
  - **Hastie, Tibshirani, Friedman**
  - **Mixture Data Example (Chapter 2)**

  http://www-stat.stanford.edu/~tibs/ElemStatLearn/
- **Project – Canopy Clustering**
  http://www.kamalnigam.com/papers/canopy-kdd00.pdf

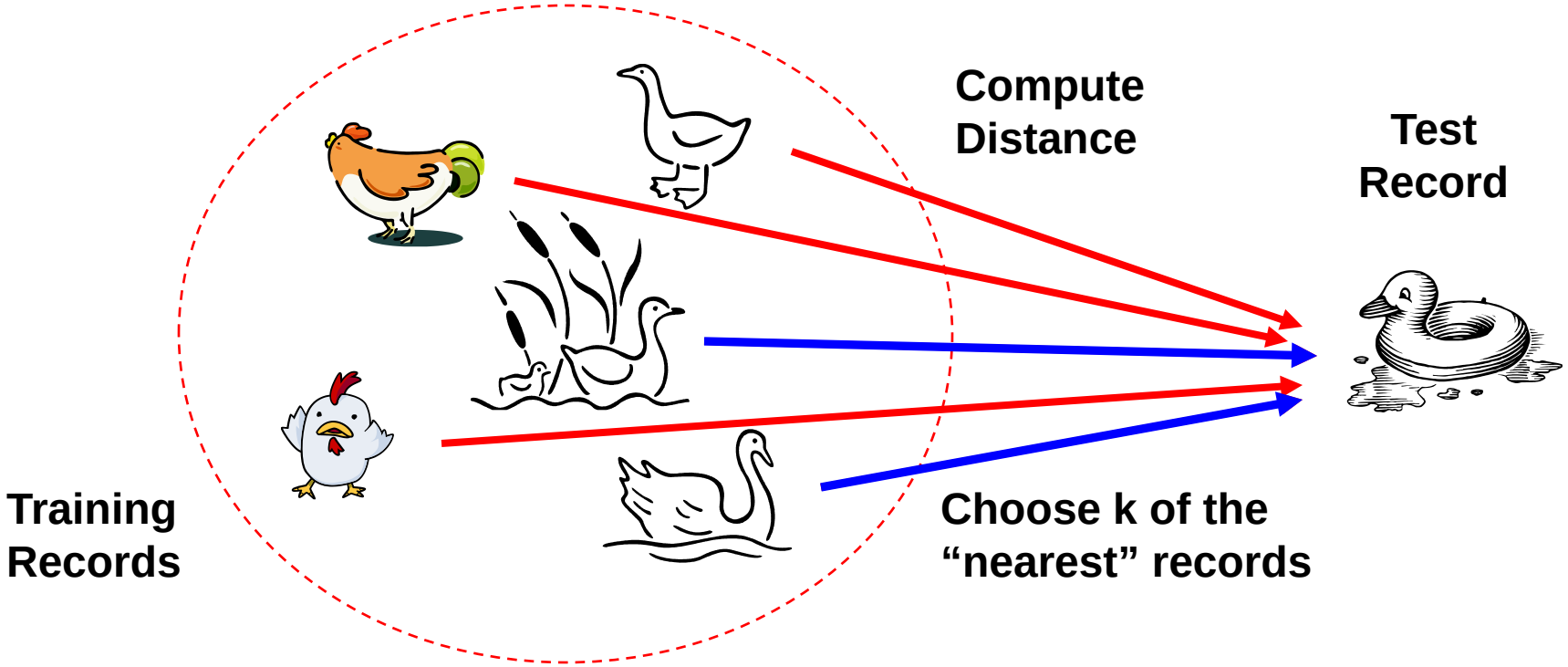
K Nearest-Neighbor Classifier

- **KfirstNearestNeighbor.R**
  -Performs kNN on the sonar data

- **mixSimknn.R**
  -Uses the mixture data from the Elements of Statistical Learning
  -notice what happens for different k values

Nearest Neighbor Classifiers

- Basic idea:
  - If it walks like a duck, quacks like a duck, then it's probably a duck



Instance Based Classifiers

- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
    - KNN is a method: Does NOT produce a “Model”
  - Nearest neighbor
    - Uses k “closest” points (nearest neighbors) for performing classification

Instance-Based Classifiers

Set of Stored Cases

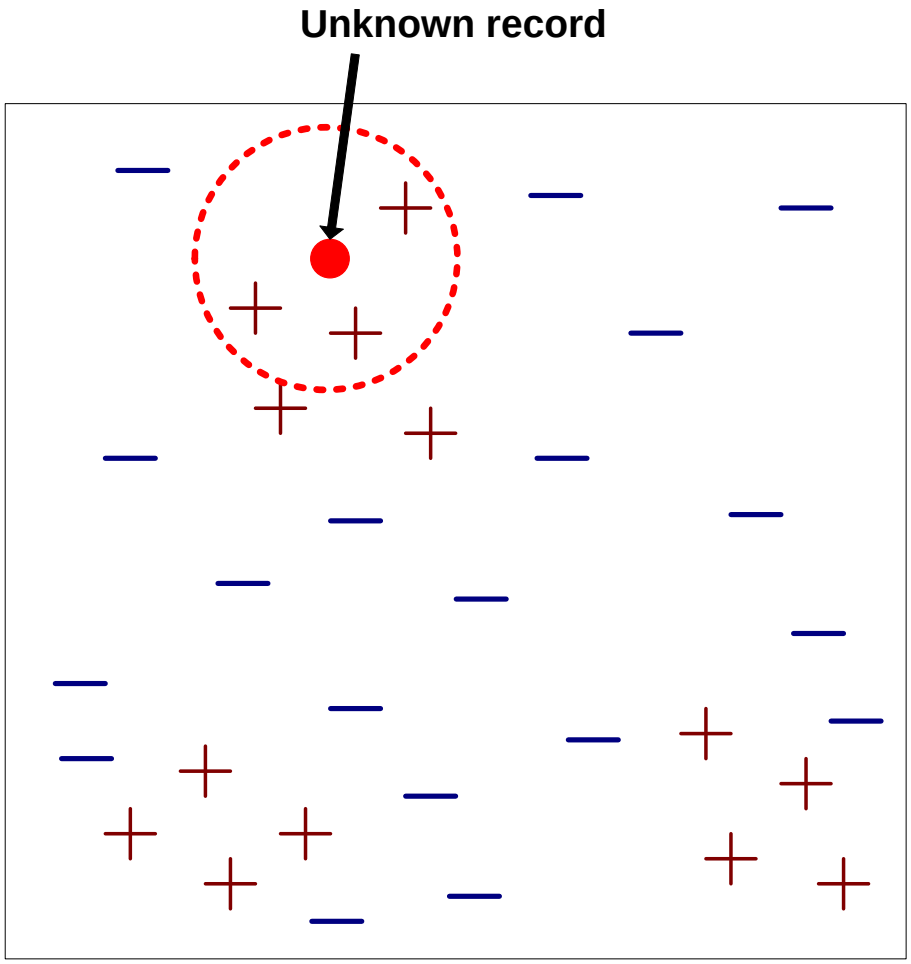
Atr1	……..	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- **Store the training records**
- **Use training records to predict the class label of unseen cases**

Unseen Case

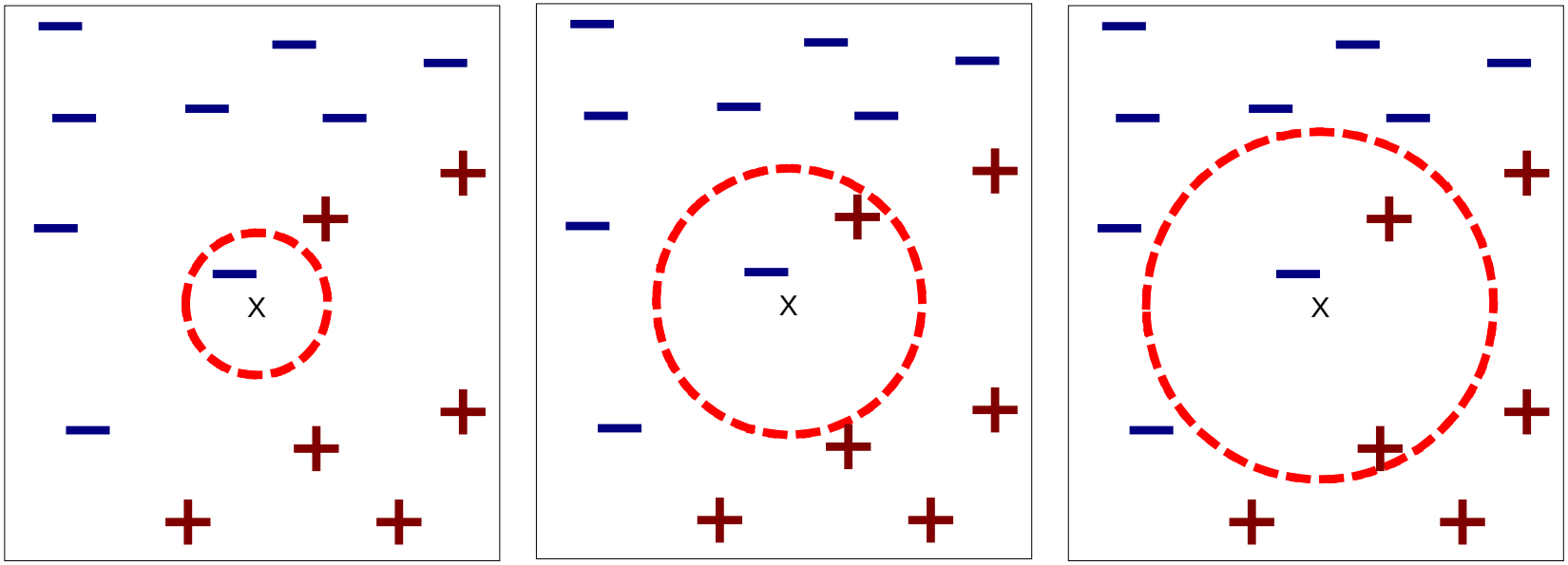
Atr1	……..	AtrN

Nearest-Neighbor Classifiers



- Requires three things
  - The set of stored records
  - Distance Metric to compute distance between records
  - The value of *k*, the number of nearest neighbors to retrieve
- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)

Definition of Nearest Neighbor

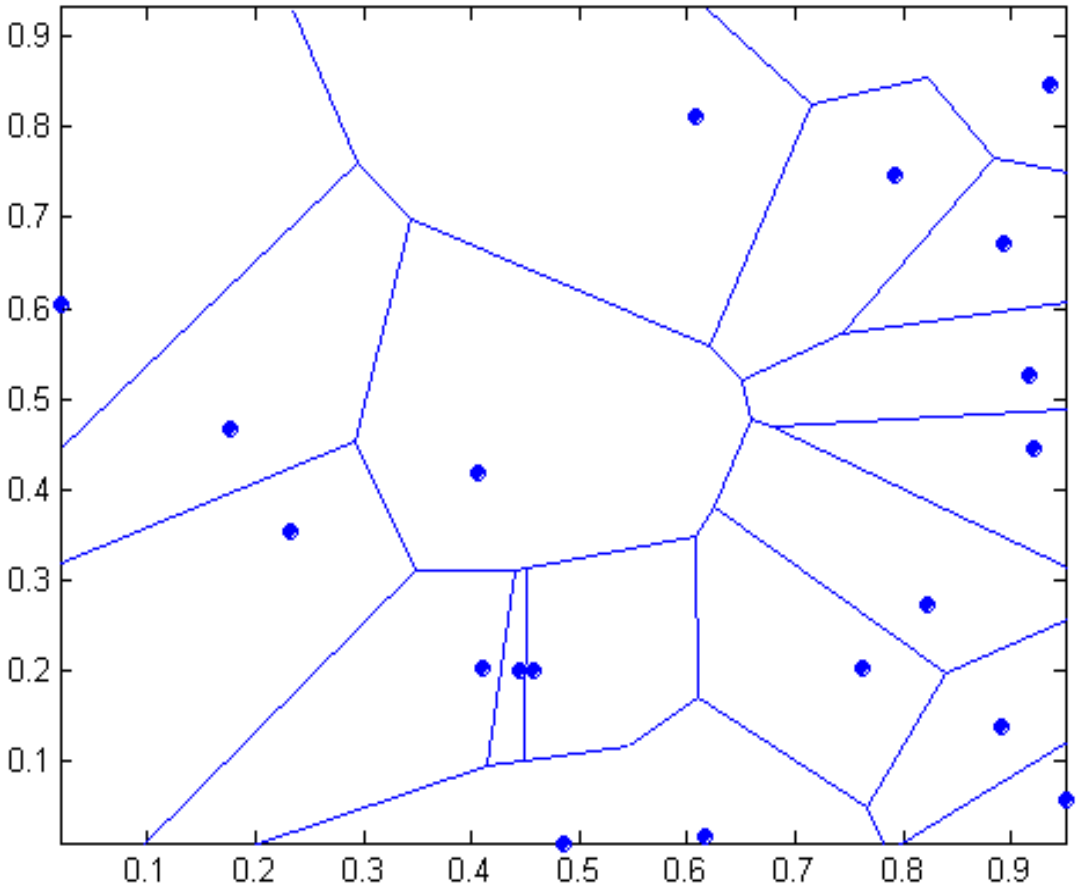


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-nearest neighbors of a record x are data points that have the k smallest distance to x

1 nearest-neighbor

Voronoi Diagram



Nearest Neighbor Classification

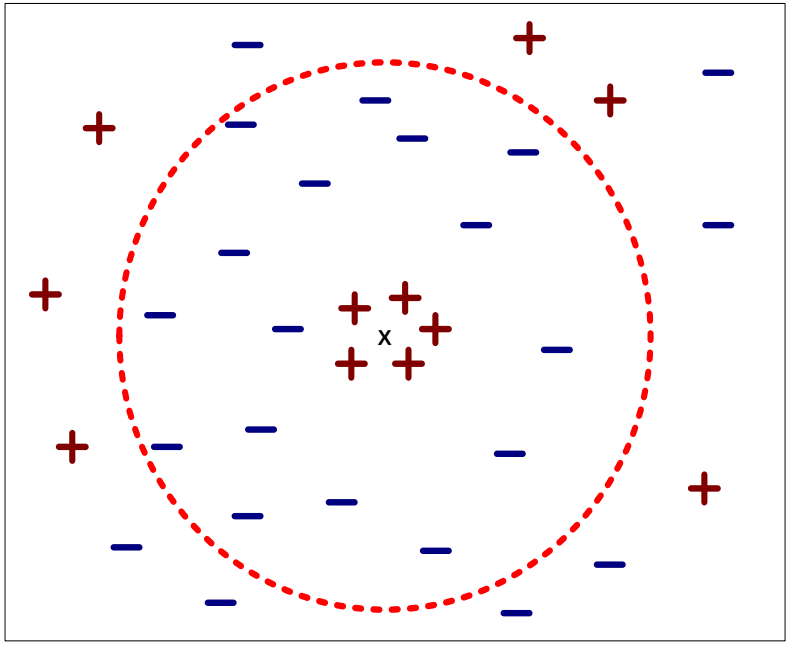
- Compute distance between two points:
  - Euclidean distance

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine the class from nearest neighbor list
  - take the majority vote of class labels among the k-nearest neighbors
  - Weigh the vote according to distance
    - weight factor, $w = 1/d^2$

Nearest Neighbor Classification…

- Choosing the value of k:
  - If k is too small, sensitive to noise points
  - If k is too large, neighborhood may include points from other classes



Nearest Neighbor Classification…

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M

Nearest Neighbor Classification

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results
    - Solution Normalize the vectors to unit length

1 1 1 1 1 1 1 1 1 1 1 0

1 0 0 0 0 0 0 0 0 0 0 0

vs

0 1 1 1 1 1 1 1 1 1 1 1

0 0 0 0 0 0 0 0 0 0 0 1

d = 1.4142

d = 1.4142

Nearest Neighbor Classification

- k-NN classifiers are lazy learners
  - It does not build models explicitly
  - Unlike eager learners such as decision tree
  - Classifying unknown records are relatively expensive

Modified Value Difference Metric

- Works with both continuous and nominal features

  -For nominal features, distance between two nominal values is computed using modified value difference metric (MVDM)

  - For example, if an attribute *color* has three values *red, green* and *blue*, and the application is to identify whether or not an object is an apple, *red* and *green* would be considered closer than *red* and *blue* because the former two both have similar correlations with the output class *apple*.

Modified Value Difference Metric

d(Single, Married) = {|P(y|s) – P(y|m)|} + {|P(n|s) – P(n|m)|}

The distance between Single and Married depends on probabilities that Single and Married affect the outcome

d(Single, Married) =
+ |{ (2 yes and single)/4 total single} - {(0 yes and married)/ 4 total married}|
+ |{ (2 no and single)/4 total single} - {(4 no and married)/ 4 total married}|

Note:

The distance between Divorced and Single is zero as they effect the outcome in the same way. Neither being divorced or single is providing any extra information about the target class.

The distance between Single and Married is the same as the distance between Divorced and Married. Being Married does provide information about the class. If fact, if you are married then you are NOT a tax cheat from the training data that has been provided.

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

Example: Modified Value Distance Metric

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Distance between nominal attribute values:

d(Single,Married)

= | 2/4 – 0/4 | + | 2/4 – 4/4 | = 1

d(Single,Divorced)

= | 2/4 – 1/2 | + | 2/4 – 1/2 | = 0

d(Married,Divorced)

= | 0/4 – 1/2 | + | 4/4 – 1/2 | = 1

d(Refund=Yes,Refund=No)

= | 0/3 – 3/7 | + | 3/3 – 4/7 | = 6/7

Class	Marital Status		
	Single	Married	Divorced
Yes	2	0	1
No	2	4	1

Class	Refund	
	Yes	No
Yes	0	3
No	3	4

$$d(V_1, V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

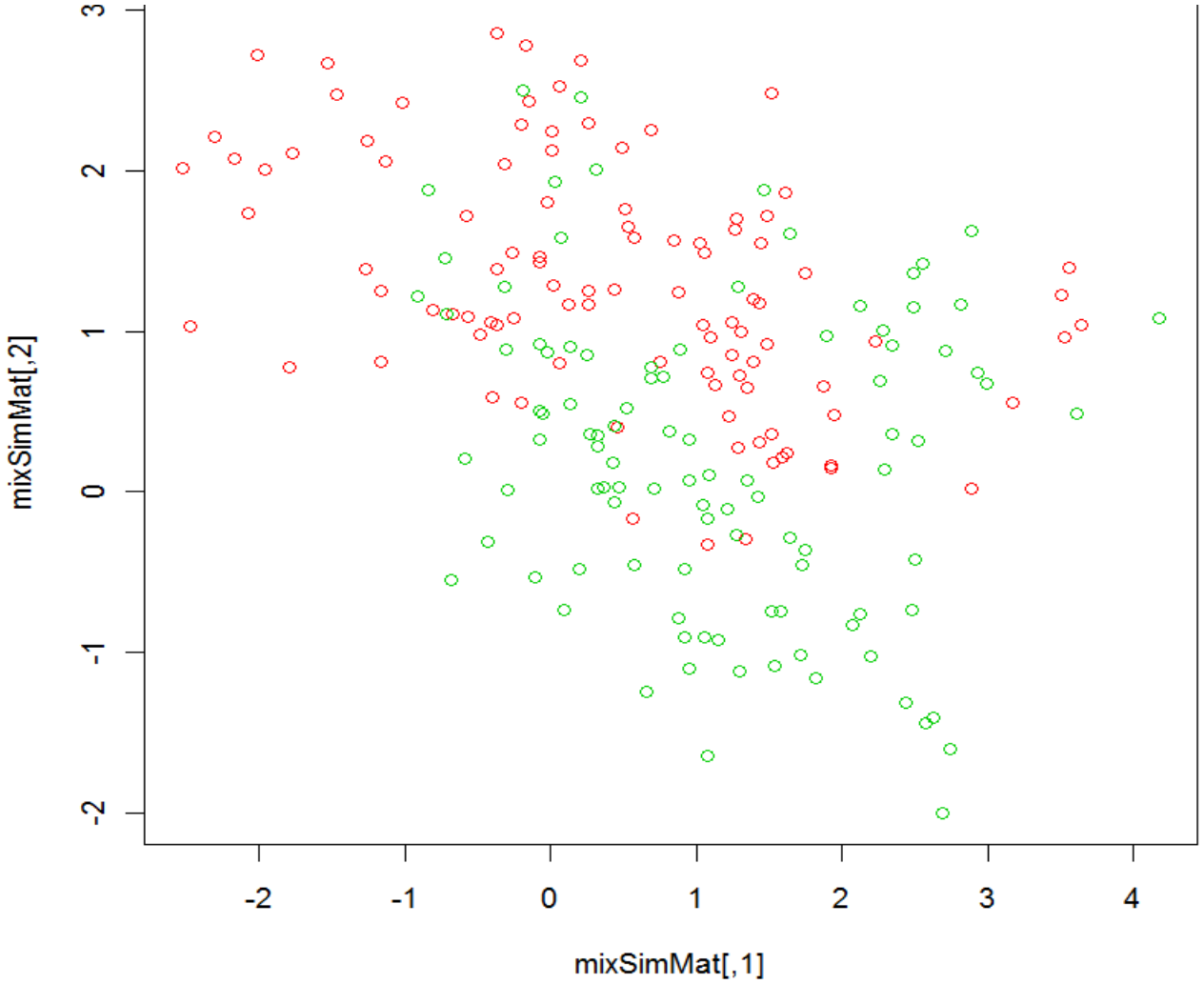
Nearest Neighbor

Nearest neighbor methods work very poorly when the dimensionality is large (meaning there are a large number of attributes)

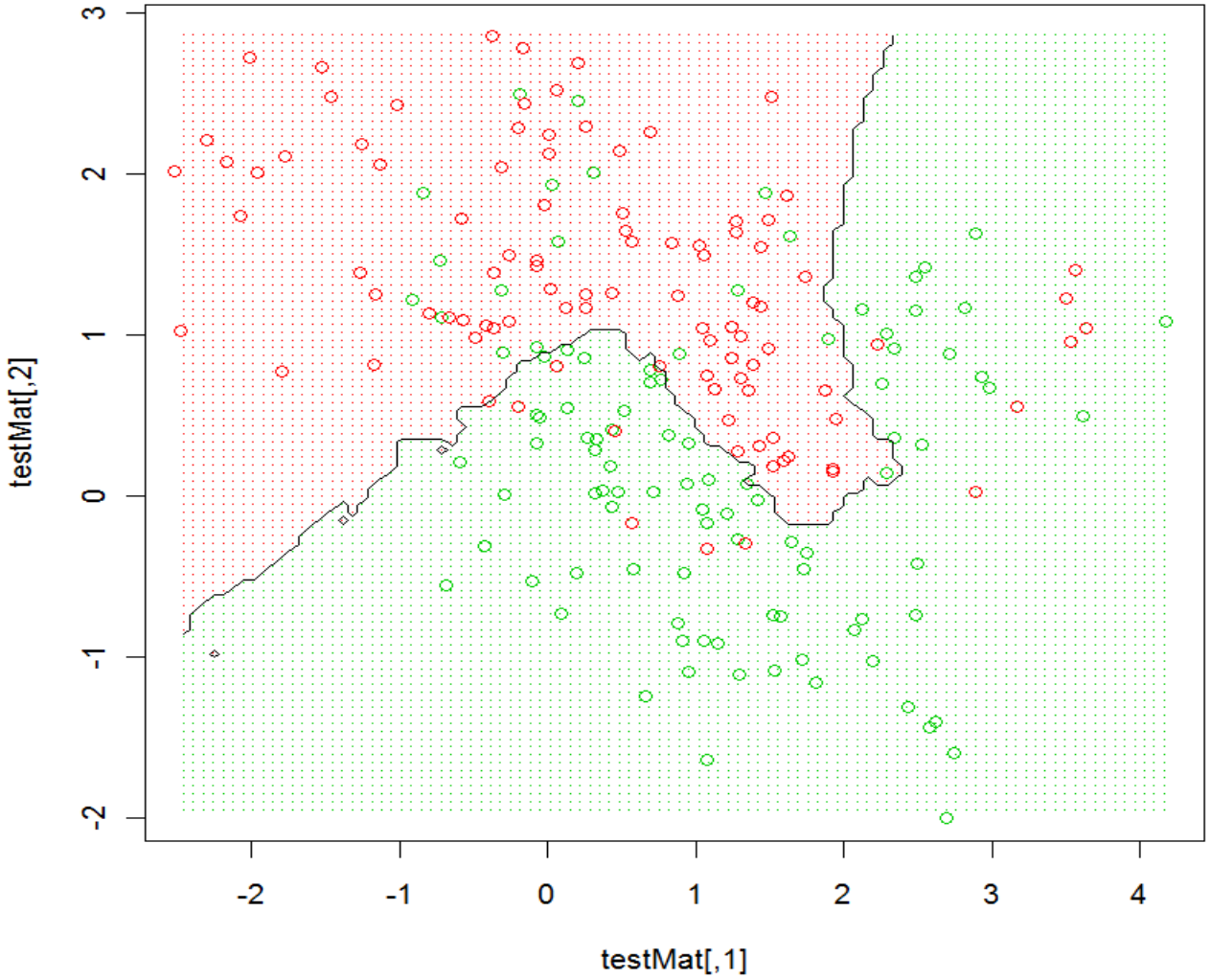
The scales of the different attributes are important. If a single numeric attribute has a large spread, it can dominate the distance metric. A common practice is to scale all numeric attributes to have equal variance.

The knn() function in R in the library "class" does a *k*-nearest neighbor classification using Euclidean distance.

K Nearest- Neighbor Classifier Example: mixSimknn.R

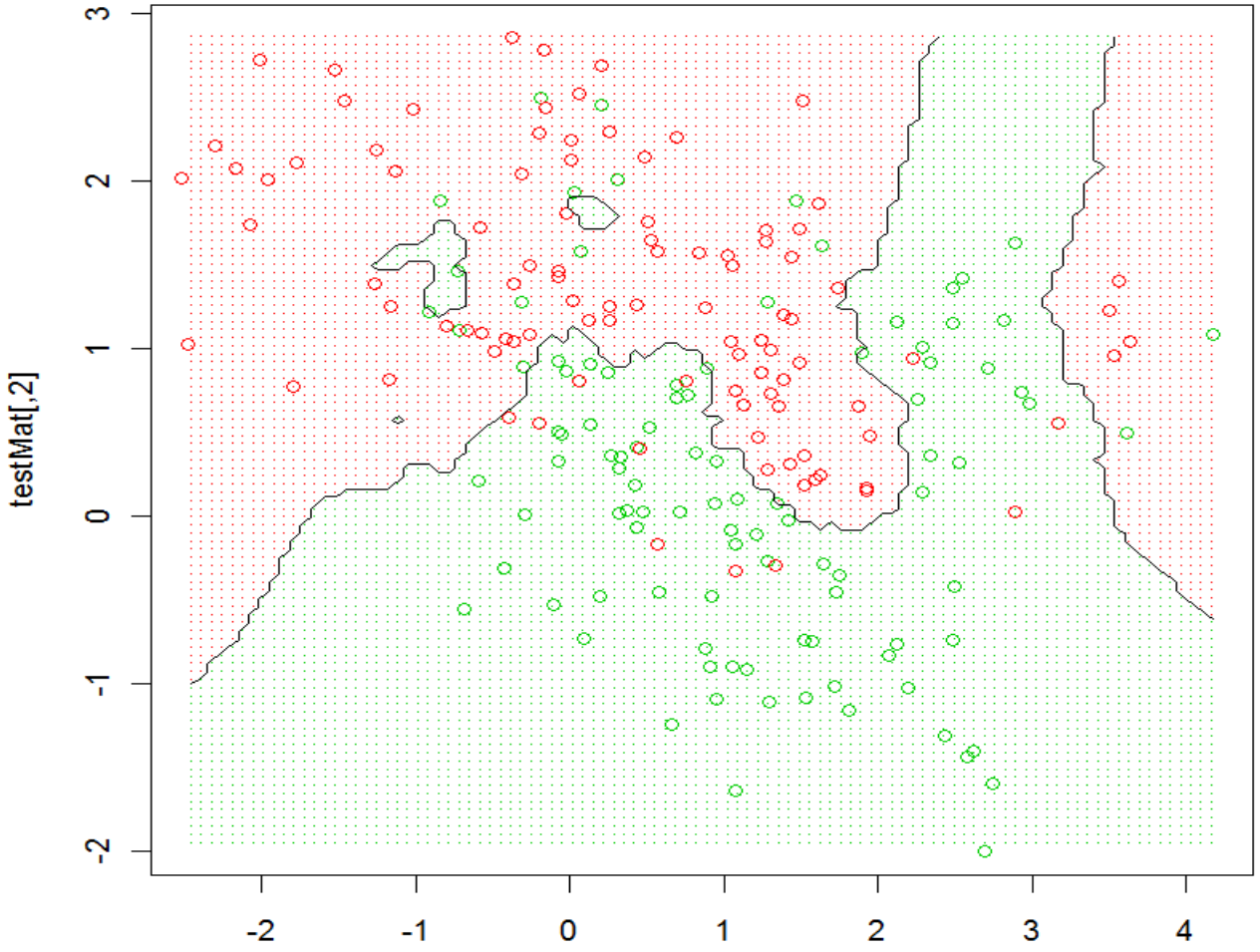


K Nearest- Neighbor Classifier
Example: K = 15 mixSimknn.R



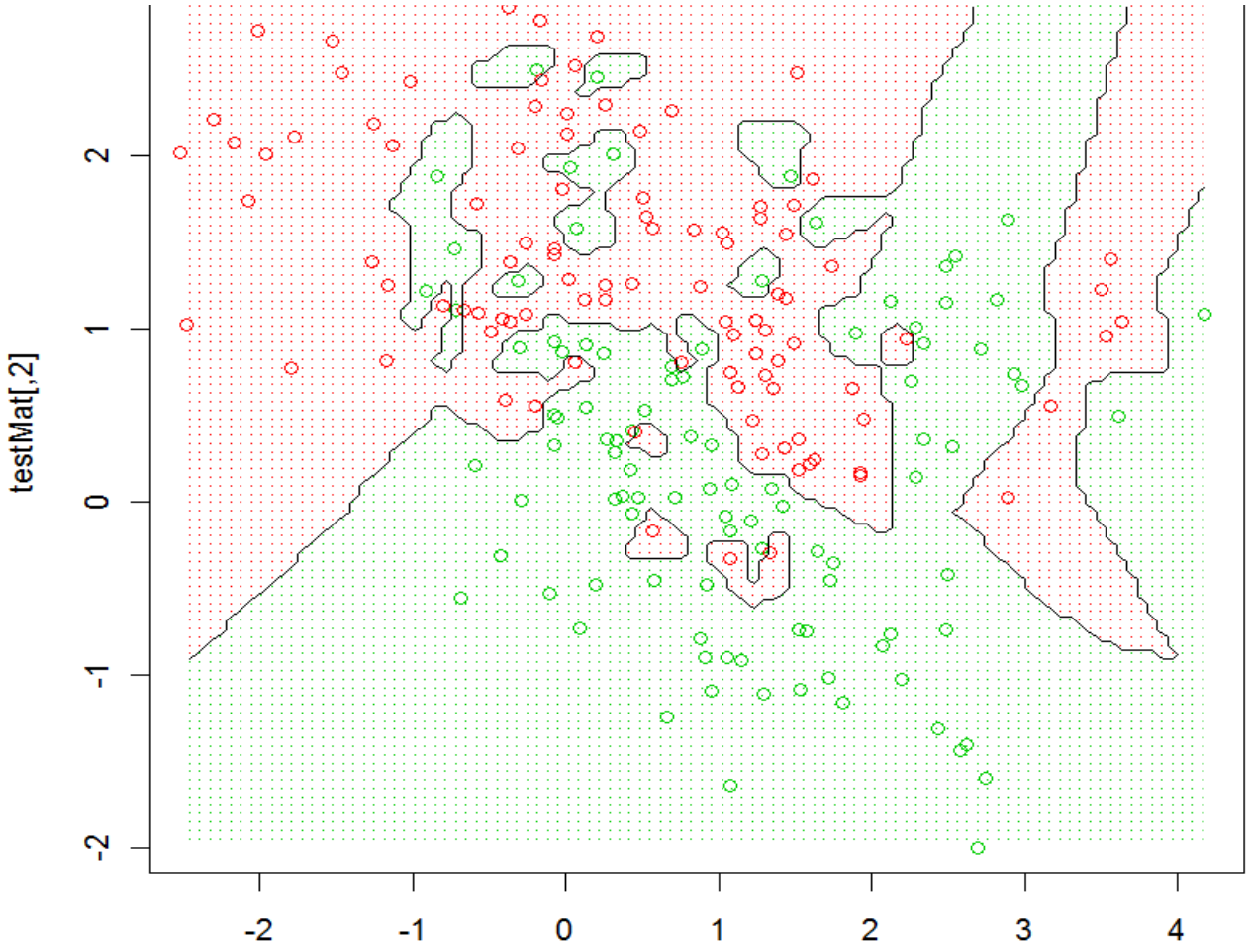
K = 5

K Nearest- Neighbor Classifier
Example: K = 5 mixSimknn.R



K = 15

K Nearest- Neighbor Classifier
Example: K = 1 mixSimknn.R



K = 1