

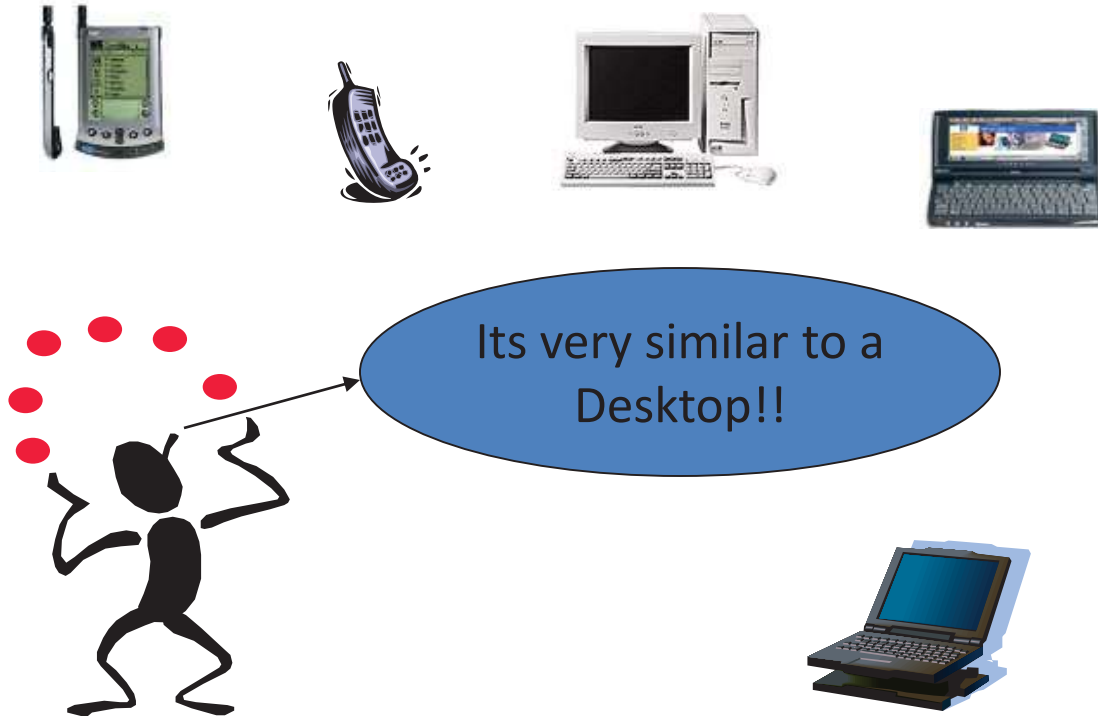
# **Algorithms: K Nearest Neighbors**

# Simple Analogy..

- Tell me about your friends(*who your neighbors are*) and *I will tell you who you are.*



# Instance-based Learning



# KNN – Different names

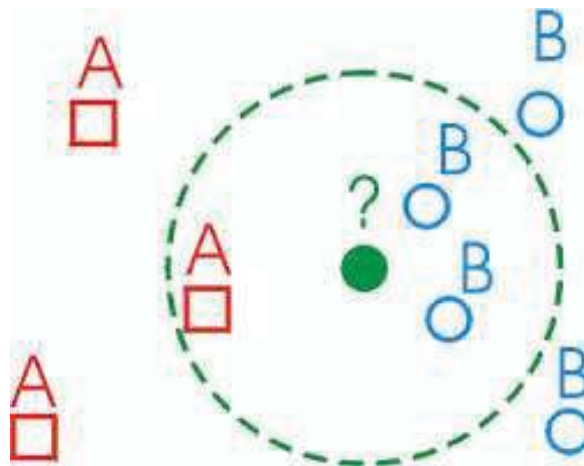
- K-Nearest Neighbors
- Memory-Based Reasoning
- Example-Based Reasoning
- Instance-Based Learning
- Lazy Learning

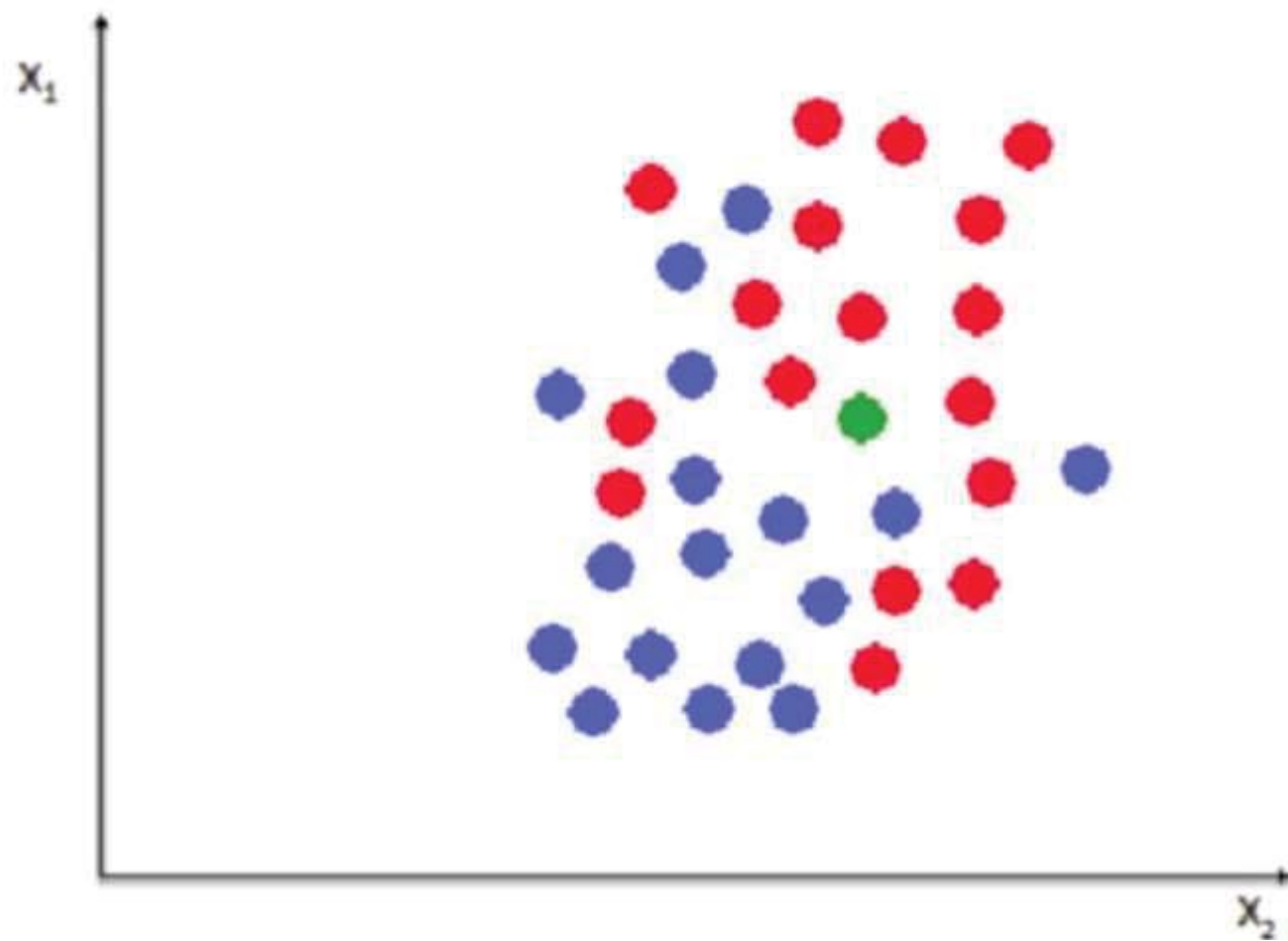
# What is KNN?

- A powerful classification algorithm used in pattern recognition.
- K nearest neighbors stores all available cases and classifies new cases based on a *similarity measure* (e.g. **distance function**)
- One of the **top data mining algorithms** used today.
- A **non-parametric** lazy learning algorithm (An Instance-based Learning method).

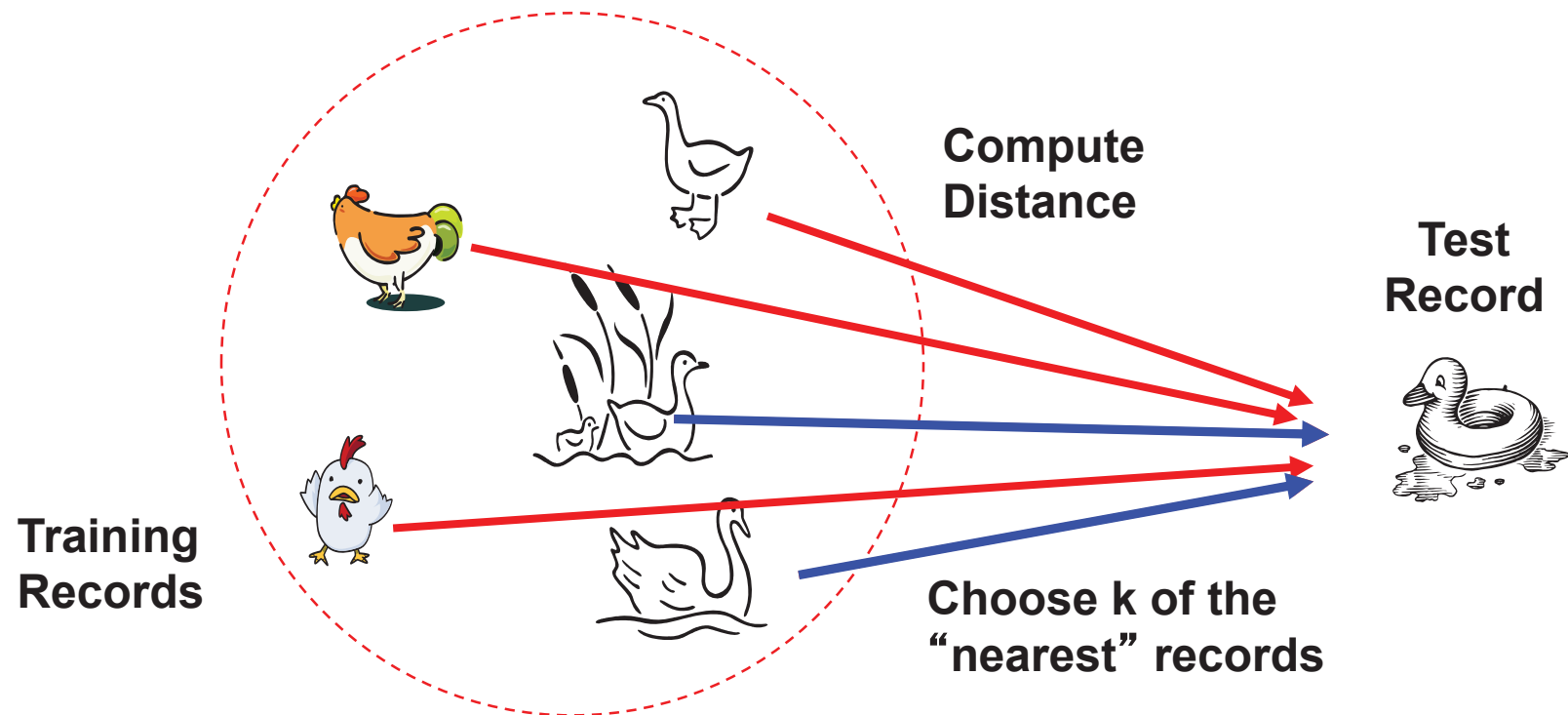
# KNN: Classification Approach

- An object (a new instance) is classified by a majority votes for its neighbor classes.
- The object is assigned to the most common class amongst its K nearest neighbors.(*measured by a distant function* )





# Distance Measure





# Distance measure for Continuous Variables

## Distance functions

Euclidean

$$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$$

Manhattan

$$\sum_{i=1}^k |x_i - y_i|$$

Minkowski

$$\left( \sum_{i=1}^k (|x_i - y_i|)^q \right)^{1/q}$$

# Distance Between Neighbors

- Calculate the distance between new example (E) and all examples in the training set.
- *Euclidean* distance between two examples.
  - $X = [x_1, x_2, x_3, \dots, x_n]$
  - $Y = [y_1, y_2, y_3, \dots, y_n]$
  - The Euclidean distance between  $X$  and  $Y$  is defined

as:

$$D(X, Y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

# K-Nearest Neighbor Algorithm

- All the instances correspond to points in an  $n$ -dimensional feature space.
- Each instance is represented with a set of numerical attributes.
- Each of the training data consists of a set of vectors and a class label associated with each vector.
- Classification is done by comparing feature vectors of different  $K$  nearest points.
- Select the  $K$ -nearest examples to  $E$  in the training set.
- Assign  $E$  to the most common class among its  $K$ -nearest neighbors.

# 3-KNN: Example(1)

Customer	Age	Income	No. credit cards	Class
George	35	35K	3	No
Rachel	22	50K	2	Yes
Steve	63	200K	1	No
Tom	59	170K	1	No
Anne	25	40K	4	Yes
John	37	50K	2	YES

## Distance from John

$$\text{sqrt} [(35-37)^2 + (35-50)^2 + (3-2)^2] = 15.16$$

$$\text{sqrt} [(22-37)^2 + (50-50)^2 + (2-2)^2] = 15$$

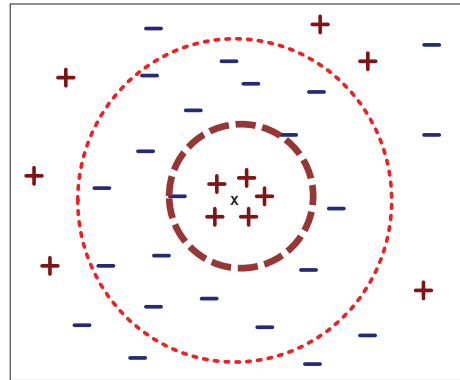
$$\text{sqrt} [(63-37)^2 + (200-50)^2 + (1-2)^2] = 152.23$$

$$\text{sqrt} [(59-37)^2 + (170-50)^2 + (1-2)^2] = 122$$

$$\text{sqrt} [(25-37)^2 + (40-50)^2 + (4-2)^2] = 15.74$$

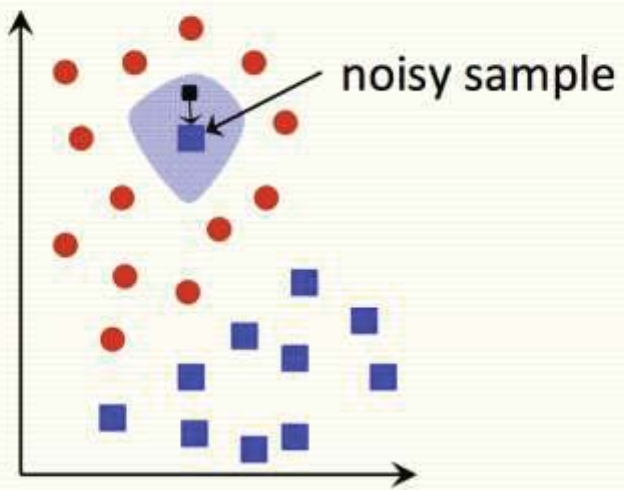
# How to choose K?

- If K is too small it is sensitive to noise points.
- Larger K works well. But too large K may include majority points from other classes.



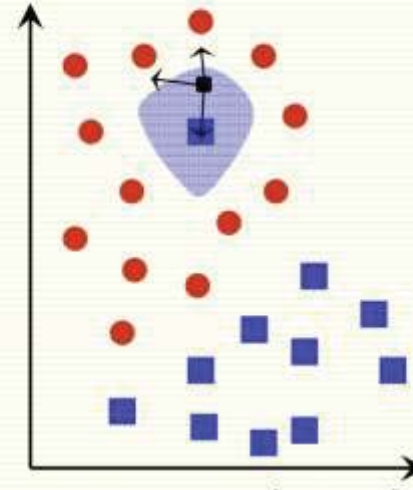
- Rule of thumb is  $K < \sqrt{n}$ ,  $n$  is number of examples.

**1 NN**

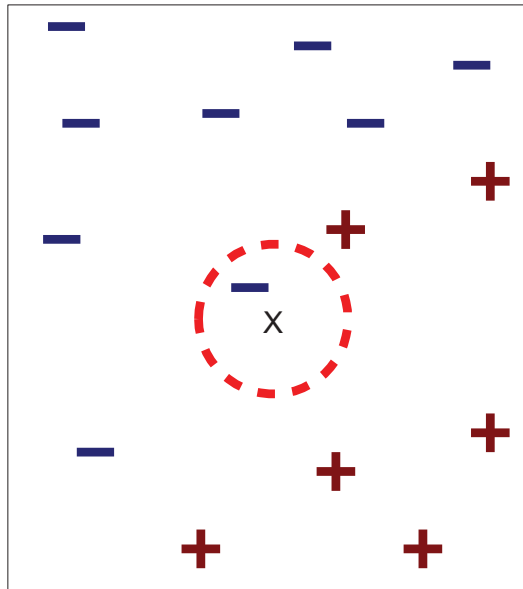


every example in the blue shaded area will be misclassified as the **blue** class

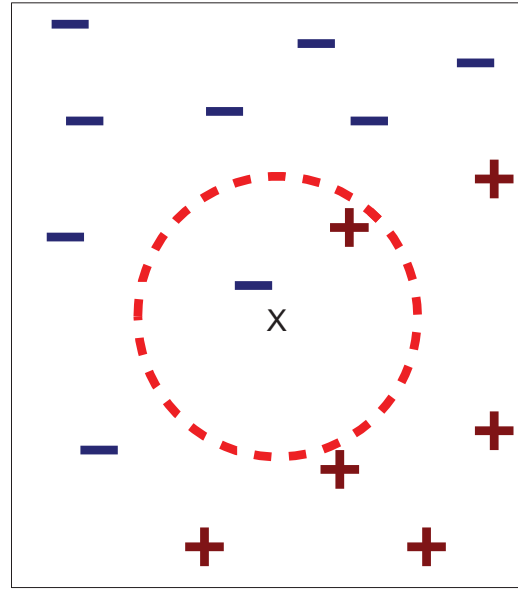
**3 NN**



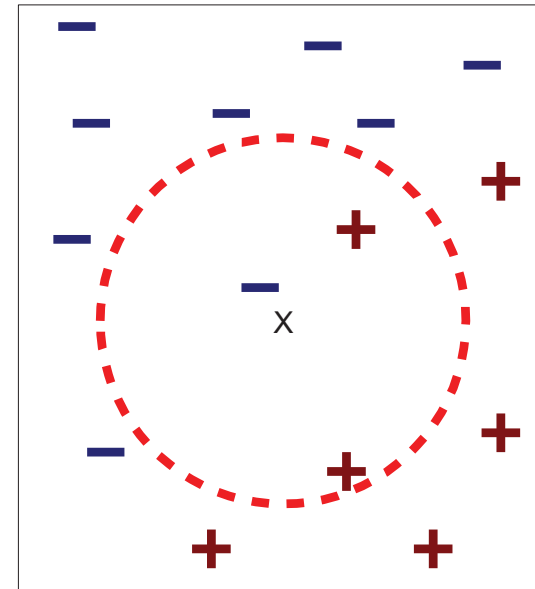
every example in the blue shaded area will be classified correctly as the **red** class



(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  $x$

# KNN Feature Weighting

- Scale each feature by its importance for classification

$$D(a, b) = \sqrt{\sum_k w_k (a_k - b_k)^2}$$

- Can use our prior knowledge about which features are more important
- Can learn the weights  $w_k$  using **cross-validation** (to be covered later)



# Feature Normalization

- Distance between neighbors could be dominated by some attributes with relatively large numbers.
  - e.g., income of customers in our previous example.

$$a_i = \frac{v_i - \min v_i}{\max v_i - \min v_i}$$

- Arises when two features are in different scales.
- Important to normalize those features.
  - Mapping values to numbers between 0 – 1.

# Nominal/Categorical Data

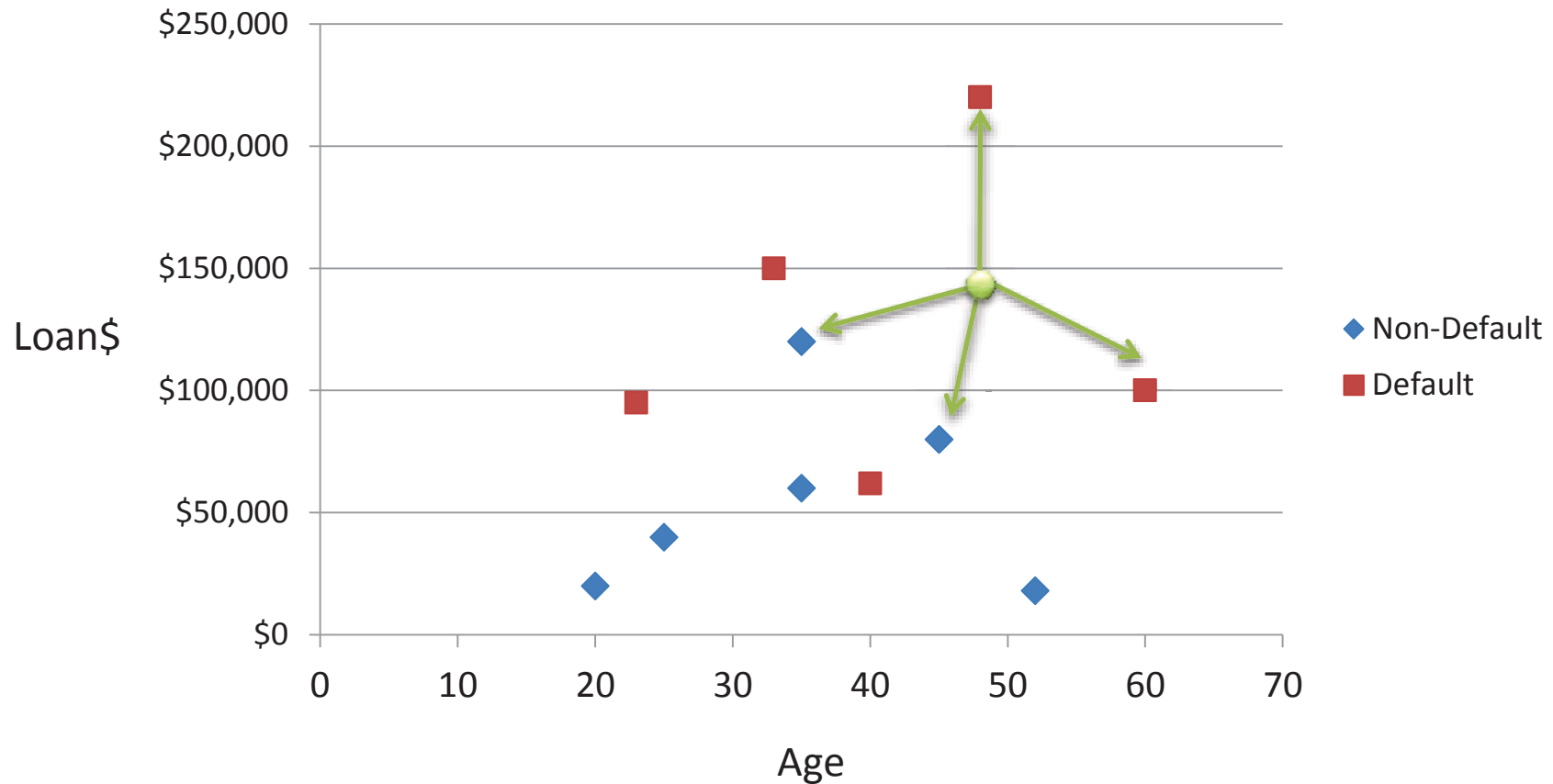
- Distance works naturally with numerical attributes.
- Binary value categorical data attributes can be regarded as 1 or 0.

**Hamming Distance**

$$D_H = \sum_{i=1}^k |x_i - y_i|$$
$$x = y \Rightarrow D = 0$$
$$x \neq y \Rightarrow D = 1$$

X	Y	Distance
Male	Male	0
Male	Female	1

# KNN Classification



# KNN Classification – Distance

Age	Loan	Default	Distance
25	\$40,000	N	102000
35	\$60,000	N	82000
45	\$80,000	N	62000
20	\$20,000	N	122000
35	\$120,000	N	22000
52	\$18,000	N	124000
23	\$95,000	Y	47000
40	\$62,000	Y	80000
60	\$100,000	Y	42000
48	\$220,000	Y	78000
33	\$150,000	Y	8000
<b>48</b>	<b>\$142,000</b>	<b>?</b>	

Euclidean Distance

$$D = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

# KNN Classification – Standardized Distance

Age	Loan	Default	Distance
0.125	0.11	N	0.7652
0.375	0.21	N	0.5200
0.625	0.31	N	0.3160
0	0.01	N	0.9245
0.375	0.50	N	0.3428
0.8	0.00	N	0.6220
0.075	0.38	Y	0.6669
0.5	0.22	Y	0.4437
1	0.41	Y	0.3650
0.7	1.00	Y	0.3861
0.325	0.65	Y	0.3771
<b>0.7</b>	<b>0.61</b>	<b>?</b>	

Standardized Variable

$$X_s = \frac{X - Min}{Max - Min}$$

## Strengths of KNN

- Very simple and intuitive.
- Can be applied to the data from any distribution.
- Good classification if the number of samples is large enough.

## Weaknesses of KNN

- Takes more time to classify a new example.
  - need to calculate and compare distance from new example to all other examples.
- Choosing  $k$  may be tricky.
- Need large number of samples for accuracy.