K-Nearest Neighbours

Classification algorithm

Procheta Sen

UNIVERSITY OF LIVERPOOL

World's simplest classifier!

- Given a training dataset D_{train} of N instances $(\overline{X}, y)$, simply "remember" the entire N instance in the memory (or hard-disk): **memory-based learning**

- When we want to classify a test (previously unseen, not in D_{train}) instance $\overline{X'}$, we would simply check whether $\overline{X'}$ is in D_{train}
    - if $\overline{X'} \in D_{train}$, then return the label of $\overline{X'}$
    - otherwise make a random guess

The Nearest Neighbour classifier

- **Training**: store entire training set D_{train}

- **Classification**: for an input object $\overline{X}'$, find in the training set the “**closest**” **object** $\overline{X}$ to $\overline{X}'$ and classify $\overline{X}'$ same as $\overline{X}$.

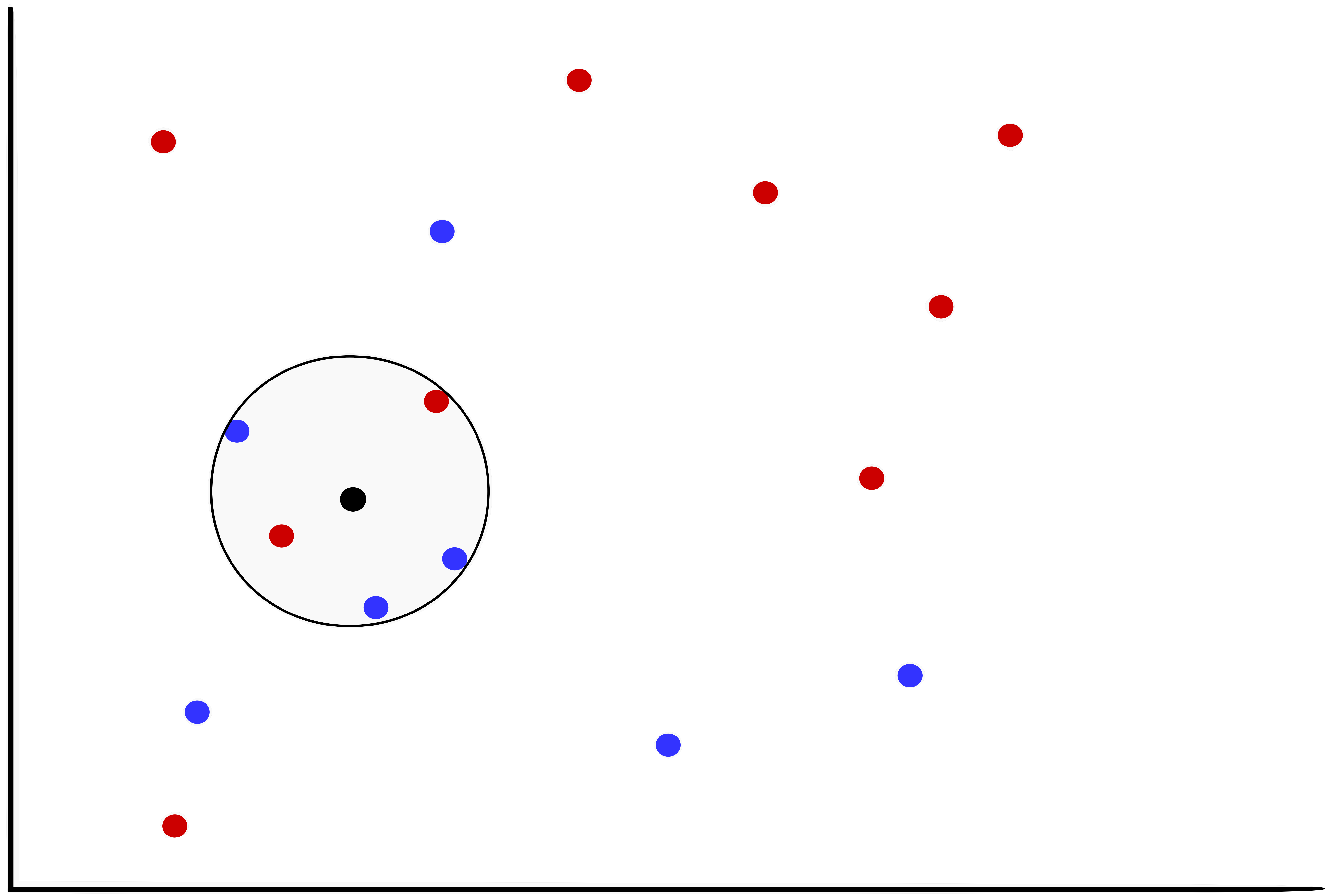
The **k**-Nearest Neighbour classifier

- **Training**: store entire training set

- **Classification**: for an input object $\overline{X'}$,
  - find in the training set the **k closest (nearest) objects** to $\overline{X'}$
  - find the majority label among those nearest neighbours
  - the majority label is predicted to the test instances $\overline{X'}$

Example

5-NN: find 5 closest to the black point. 3 blue and 2 red, so predict blue



Measures of similarity/distance

Measures of similarity/distance

General problem

Given two objects $\overline{X}$ and $\overline{Y}$ determine the value of the **similarity** or **distance** between the two objects.

Similarity functions: larger values imply greater similarity

Distance functions: smaller values imply greater similarity

In some domains, such as spatial data, it is more natural to talk about distance functions, whereas in other domains, such as text, it is more natural to talk about similarity functions.

Similarity and distance functions are often expressed in closed form (easy to compute formula), but in some domains, such as time-series data, they are defined algorithmically and cannot be expressed in closed form (may be computationally expensive).

Measures of similarity/distance for numerical data

Euclidean distance between the vectors $\overline{X}$ and $\overline{Y}$.

$$EucDist(\overline{X}, \overline{Y}) = \|\overline{X} - \overline{Y}\|_2 = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} = \sqrt{(\overline{X} - \overline{Y})^T(\overline{X} - \overline{Y})}$$

where $\|\overline{T}\|_2 = \sqrt{\sum_{i=1}^{d} t_i^2}$ denotes the L^2-norm of the vector $\overline{T} = (t_1, t_2, \ldots, t_d)^T$.

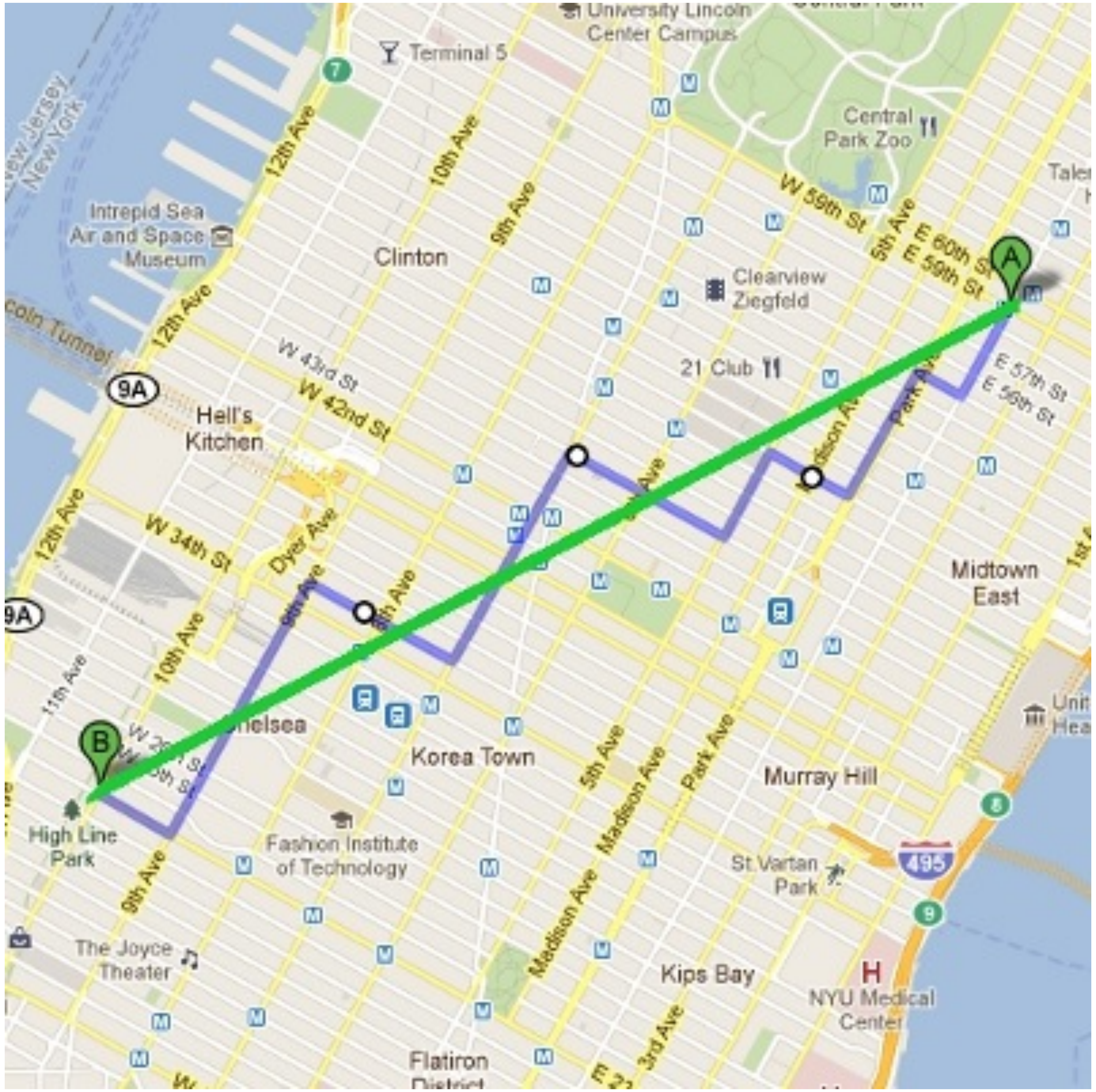
Measures of similarity/distance for numerical data

Manhattan Distance between the vectors $\overline{X}$ and $\overline{Y}$.

$$ManDist(\overline{X}, \overline{Y}) = \|\overline{X} - \overline{Y}\|_1 = \sum_{i=1}^{d} |x_i - y_i|,$$

where $\|\overline{T}\|_1 = \sum_{i=1}^{d} |t_i|$ denotes the L^1-norm of the vector $\overline{T} = (t_1, t_2, \ldots, t_d)^T$.

Manhattan



$EucDist(A, B)$

$ManDist(A, B)$

Vector norms

- "norm" is a mathematical concept that expresses the "size/length" of a vector
- Popular vector norms in data mining are as follows

$$L^1\text{-norm } \|\overline{X}\|_1 = \sum_{i=1}^{d} |x_i|$$

$$L^2\text{-norm } \|\overline{X}\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$$

$$L^0\text{-norm } \|\overline{X}\|_0 = \text{ no. of non-zero elements in } \overline{X}$$

$$L^\infty\text{-norm } \|\overline{X}\|_\infty = \max\{\,|x_1|, |x_2|, \ldots, |x_d|\,\}$$

From vector norms to distances

From a vector norm, one can construct a distance function between pairs of vectors (say, $\overline{X}$ and $\overline{Y}$) as the norm of their difference

L^1-norm $\|\overline{X}\|_1 = \sum_{i=1}^{d} |x_i|$ L^1-distance $\|\overline{X} - \overline{Y}\|_1 = \sum_{i=1}^{d} |x_i - y_i|$

L^2-norm $\|\overline{X}\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$ L^2-distance $\|\overline{X} - \overline{Y}\|_2 = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}$

L^0-norm $\|\overline{X}\|_0 =$ no. of non-zero elements in $\overline{X}$

L^∞-norm $\|\overline{X}\|_\infty = \max\{ |x_1|, |x_2|, \ldots, |x_d| \}$

Measures of similarity/distance for numerical data

Cosine similarity

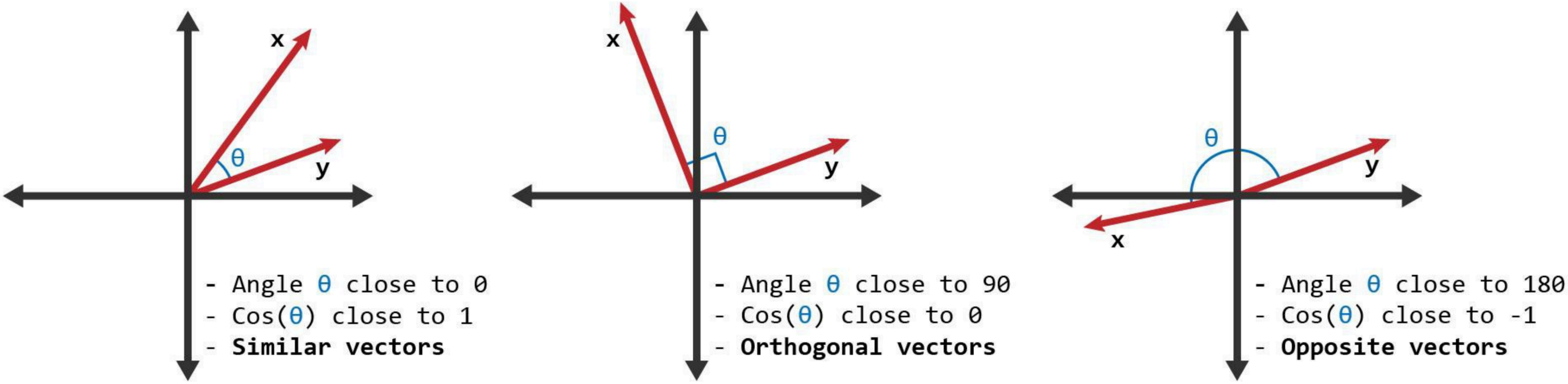
Let $\overline{X} = (x_1, x_2, \ldots, x_d)$ and $\overline{Y} = (y_1, y_2, \ldots, y_d)$ be vectors in $\mathbb{R}^n$

$$CosSim(\overline{X}, \overline{Y}) = \frac{\overline{X}^T\overline{Y}}{||\overline{X}||_2||\overline{Y}||_2} = \cos(\theta),$$

Where $||\overline{X}||_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$ and θ is the the angle between the vectors $\overline{X}$ and $\overline{Y}$.

Measuring similarity/distance for numerical data

Cosine similarity $CosSim(\overline{X}, \overline{Y}) = \dfrac{\overline{X}^T\overline{Y}}{||\overline{X}||\,||\overline{Y}||} = \cos(\theta)$,



- Angle θ close to 0
- Cos(θ) close to 1
- **Similar vectors**

- Angle θ close to 90
- Cos(θ) close to 0
- **Orthogonal vectors**

- Angle θ close to 180
- Cos(θ) close to -1
- **Opposite vectors**

Source: https://www.learndatasci.com/glossary/cosine-similarity/

Measuring similarity/distance for numerical data

Cosine similarity example (text mining domain):

1. We have a fixed set of **terms (keywords)** of interest
2. Each **term** is assigned a different **dimension**
3. A **document is characterised by a vector** where the value in each dimension corresponds to the frequency of the term appearance in the document
4. **Cosine similarity** then gives a useful measure of how similar two documents are likely to be with respect to their subject matter

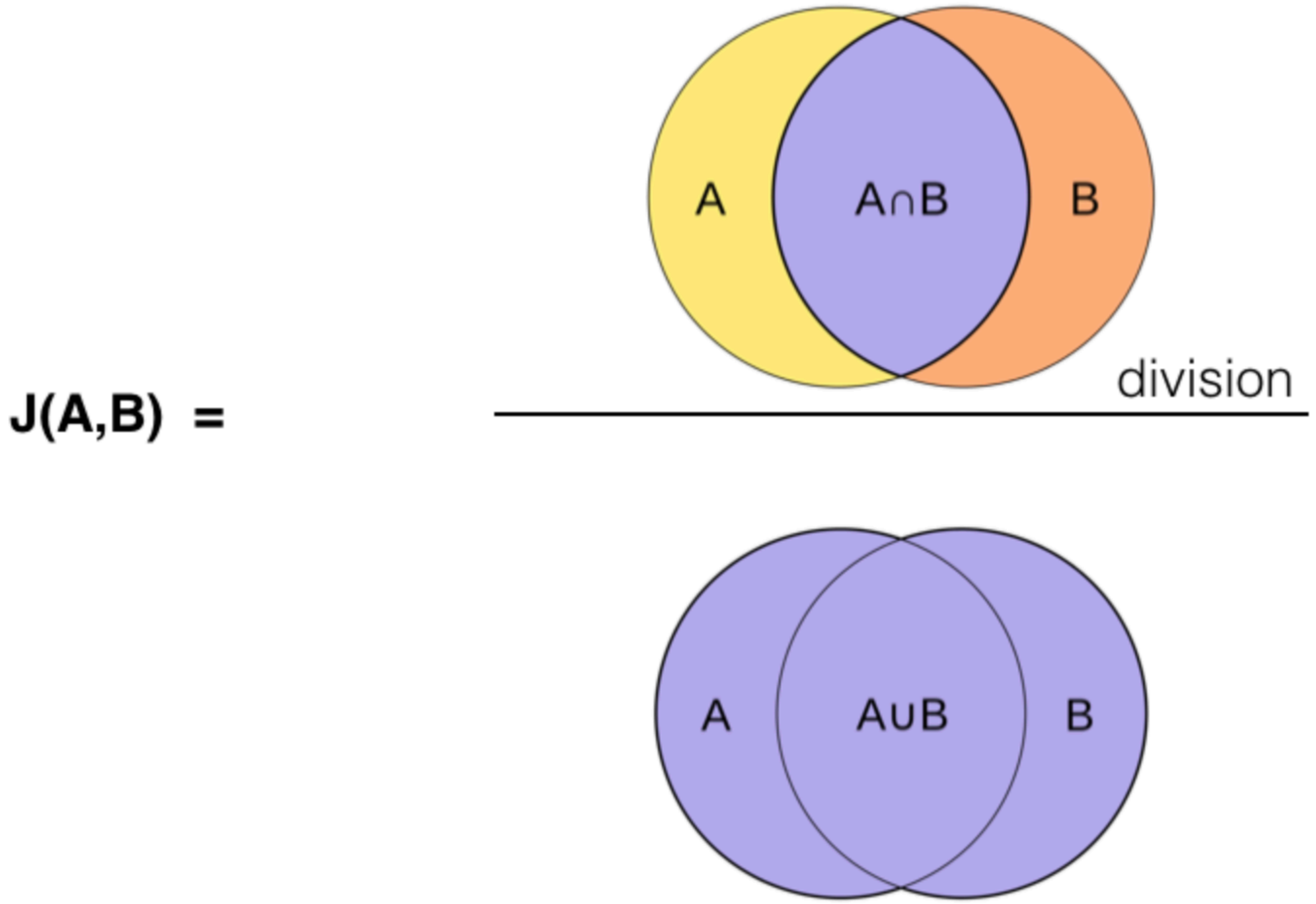
Measures of similarity/distance for numerical data

Cosine distance

$$CosDist(\overline{X}, \overline{Y}) = 1 - CosSim(\overline{X}, \overline{Y})$$

Measures of similarity/distance for set data

Jaccard similarity coefficient: $J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$



Jaccard distance: $d_J(A, B) = 1 - J(A, B)$

https://www.kaggle.com/pashupatigupta/evaluation-metric-implementation-and-demo

Measures of similarity/distance for set data

Overlap coefficient: $\text{overlap}(A, B) = \dfrac{|A \cap B|}{\min(|A|, |B|)}$

Measures of similarity/distance for binary data

Hamming distance is defined for binary vectors of the same length— it is equal to the number of coordinates where the two vectors differ

$$\text{Hamming}(\overline{X}, \overline{Y}) = |\{i : x_i \neq y_i\}|$$

Example:

$$\overline{X} = (1,0,1,0,1,1)^T$$

$$\overline{Y} = (0,0,1,1,1,1)^T$$

$$\text{Hamming}(\overline{X}, \overline{Y}) = 2$$

Measures of similarity/distance for categorical data

Key differences with respect to numerical data:

- no ordering on the values;
- no natural ‘difference’ operation.

One approach to compute similarity between two objects $\overline{X} = (x_1, \ldots, x_d)^T$ and $\overline{Y} = (y_1, \ldots, y_d)$ with categorical features is to define the similarity function as

$$Sim(\overline{X}, \overline{Y}) = \sum_{i=1}^{d} S(x_i, y_i),$$

where $S(x_i, y_i)$ is the similarity between the feature values x_i and y_i

Measures of similarity/distance for categorical data

$$Sim(\overline{X}, \overline{Y}) = \sum_{i=1}^{d} S(x_i, y_i),$$

Specific $S(x_i, y_i)$ (example 1):

$$S(x_i, y_i) = \begin{cases} 1, \text{ if } x_i = y_i \\ 0, \text{ otherwise} \end{cases}$$

Major drawback:

- does not account for the relative frequencies among different feature values;
- E.g. if the value ‘Normal’ of a fixed feature appears in 99% of the objects and the rest have values ‘Cancer’ or ‘Diabetes’, the fact that two objects have this feature equal to ‘Normal’ tells us less than if they both would have the feature equal to ‘Cancer’

Measures of similarity/distance for categorical data

$$Sim(\overline{X}, \overline{Y}) = \sum_{i=1}^{d} S(x_i, y_i),$$

Specific $S(x_i, y_i)$ (example 2):

Let $p_k(x)$ be the fraction of objects, in which the k-th feature takes on the values of x in the data set

$$S(x_i, y_i) = \begin{cases} 1/p_i(x_i)^2, \text{ if } x_i = y_i \\ 0, \text{ otherwise} \end{cases}$$