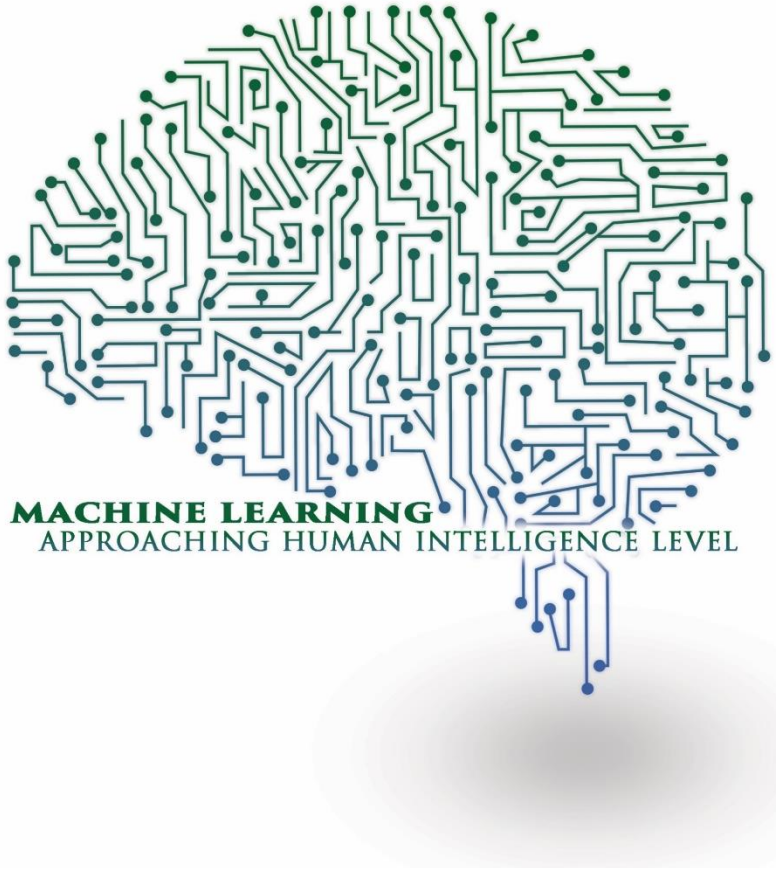
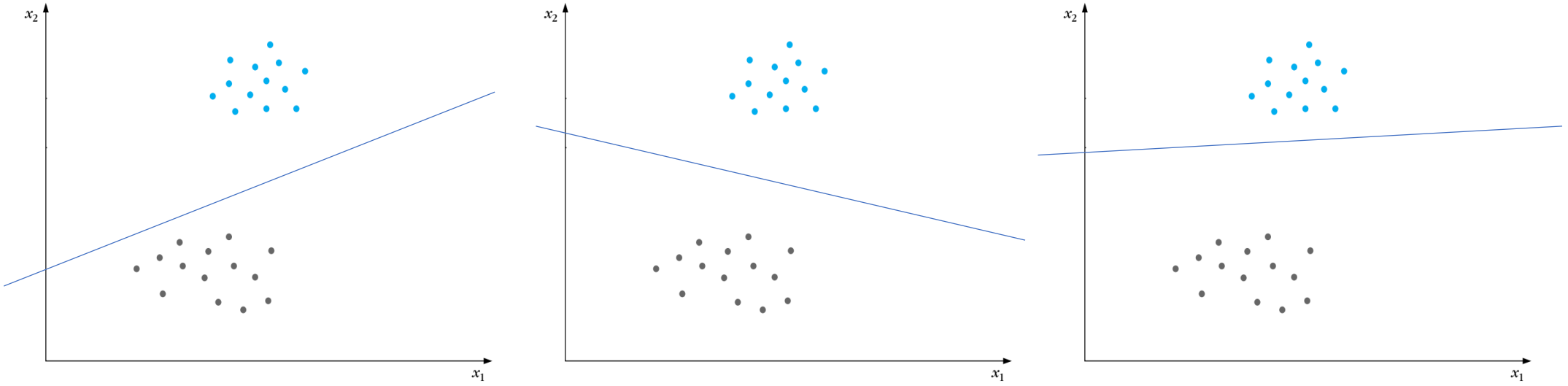


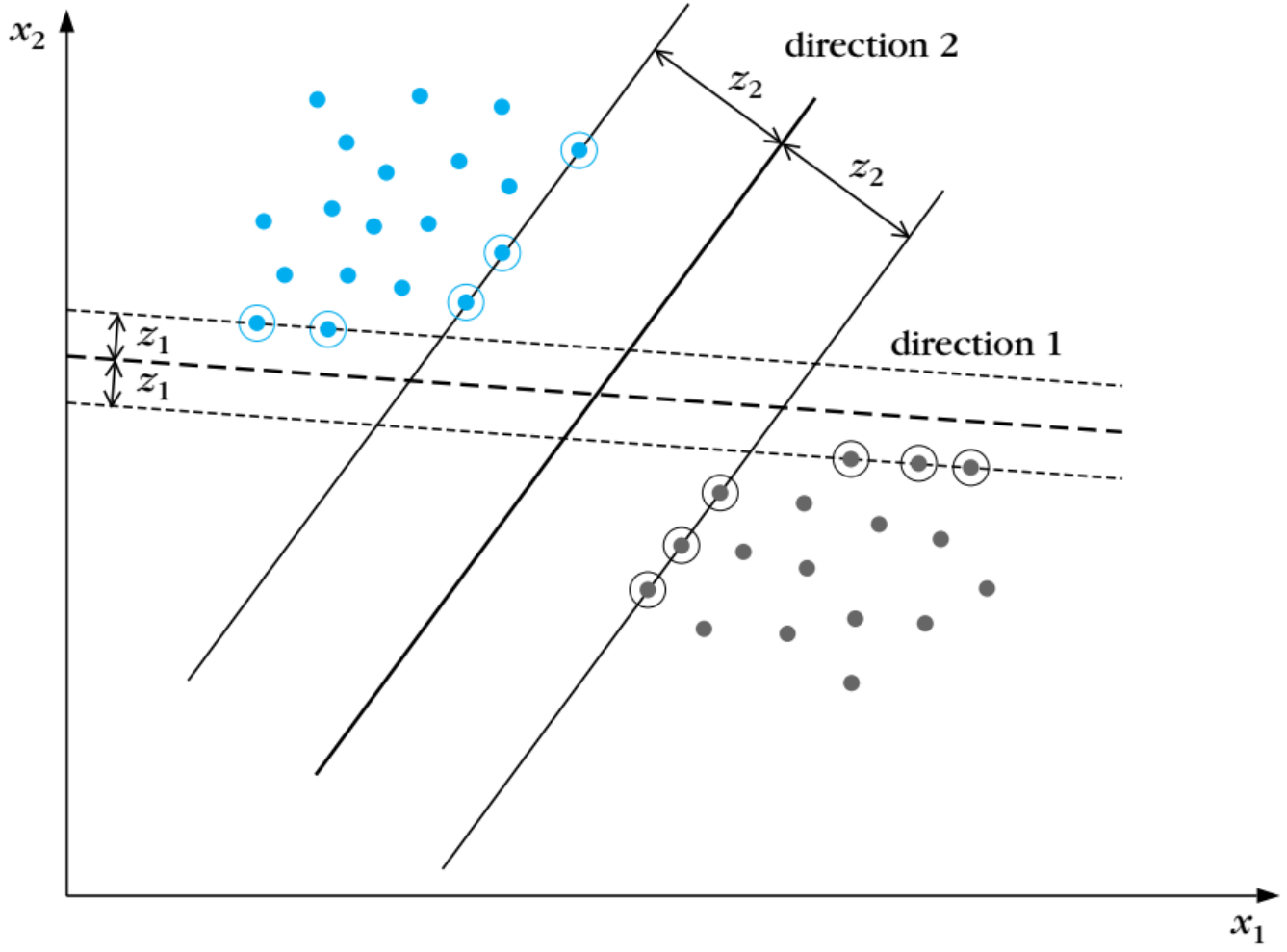
Support Vector Machine

Which classifier would you choose ?



Criterial of the optimal classifier ?

Margins and support vectors



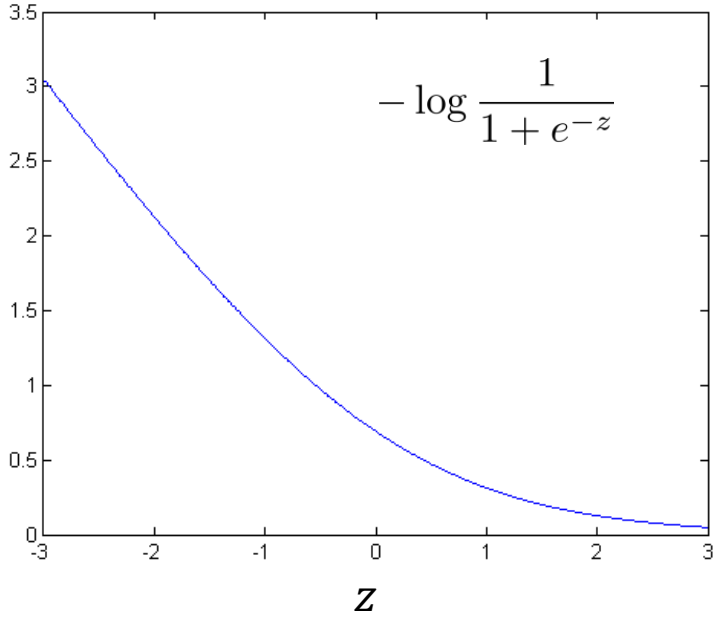
Recall logistic regression cost function

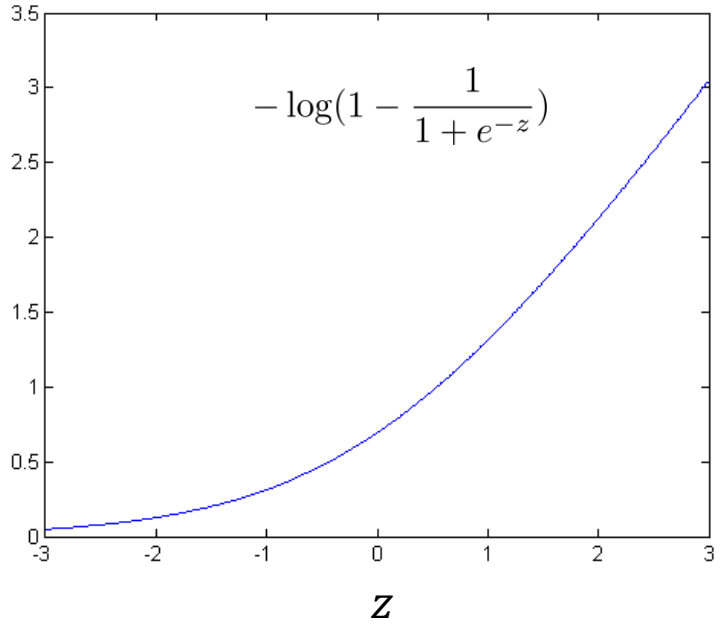
Cost of example: $-(y \log h_\theta(x) + (1-y) \log(1 - h_\theta(x)))$

$$= -y \log \frac{1}{1+e^{-\theta^T x}} - (1-y) \log(1 - \frac{1}{1+e^{-\theta^T x}})$$

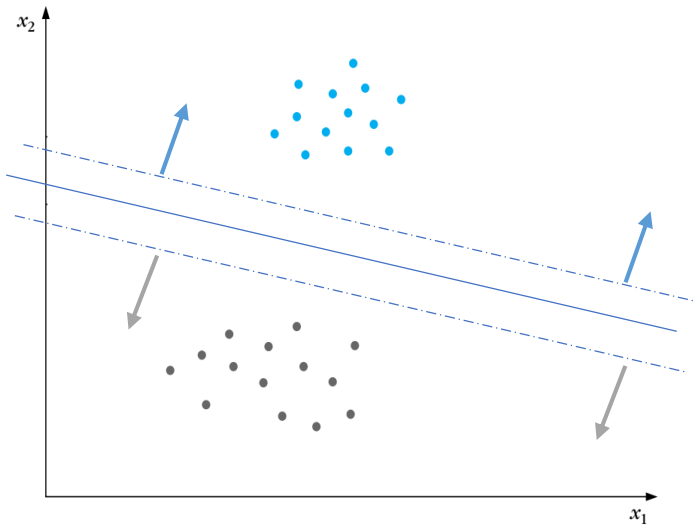
If $y = 1$ (want $\theta^T x \gg 0$): $\theta^T x \geq 1$

If $y = 0$ (want $\theta^T x \ll 0$): $\theta^T x \leq -1$





The mathematics behind large margin classification



SVM Decision Boundary

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

$$\text{s.t.} \quad \theta^T x^{(i)} \geq 1 \qquad \text{if } y^{(i)} = 1$$

$$\theta^T x^{(i)} \leq -1 \qquad \text{if } y^{(i)} = 0$$

Let t_i *be an indicator variable, meaning* :

$$t_i = \begin{cases} 1 & \textit{if } x_i \textit{ in the first class} \\ -1 & \textit{if } x_i \textit{ in the second calass} \end{cases}$$

$$\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2$$

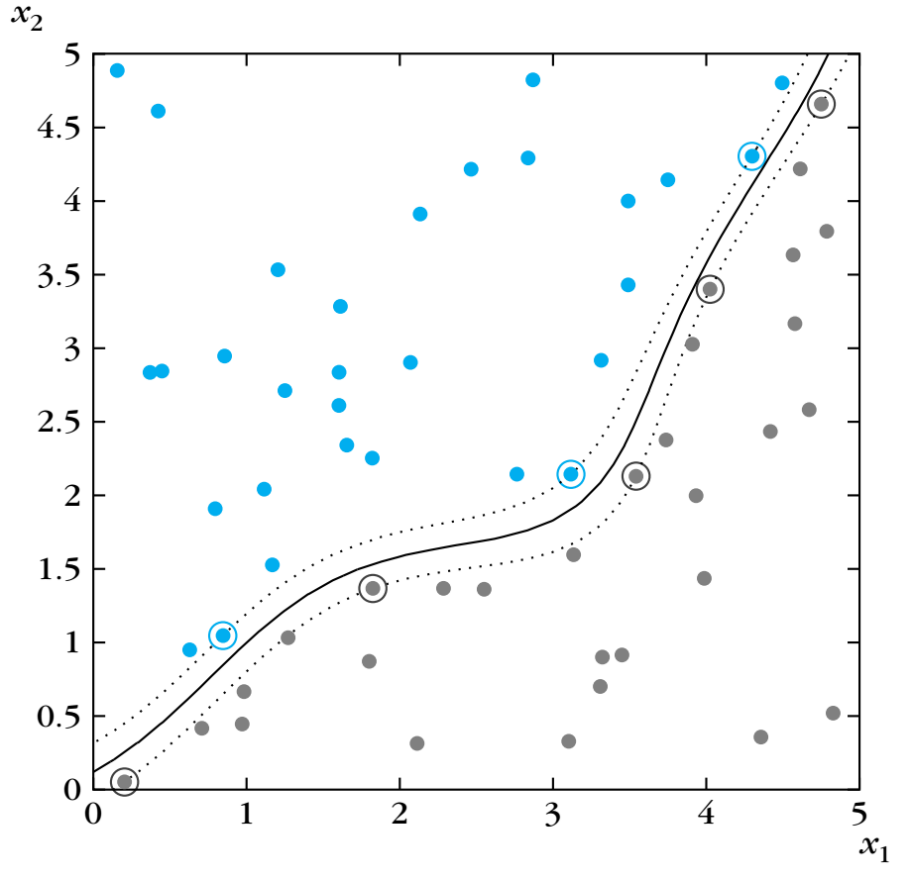
$$\text{s.t.} \quad t_i \left(\theta^T x^{(i)}\right) \geq 1$$

What if the data is not linearly separable ?

We can make a non linear transformation for the data to other space where it is *linearly separable.*

$$\boldsymbol{x} \in \mathcal{R}^{l} \longrightarrow \boldsymbol{y} \in \mathcal{R}^{k}$$

Kernels



Linear kernel (no kernel): $\theta^T x^{(i)}$

Gaussian kernel (rbf): $\exp(\frac{\|x - x^{(i)}\|}{2\sigma^2})$

need to choose σ

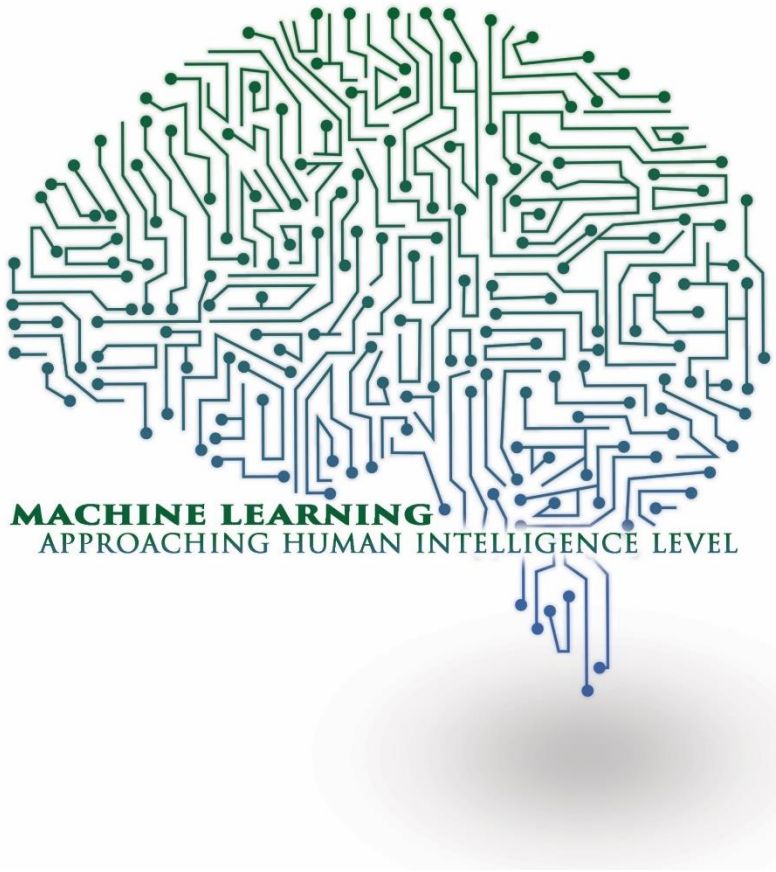
Polynomial kernel: $(\theta^T x^{(i)} + const)^d$

d is the degree

String kernel, chi-square kernel, histogram intersection kernel, …

Not all similarity functions make valid kernels.
(Need to satisfy technical condition called “Mercer’s Theorem” to make sure SVM packages’ optimizations run correctly, and do not diverge).

SVM



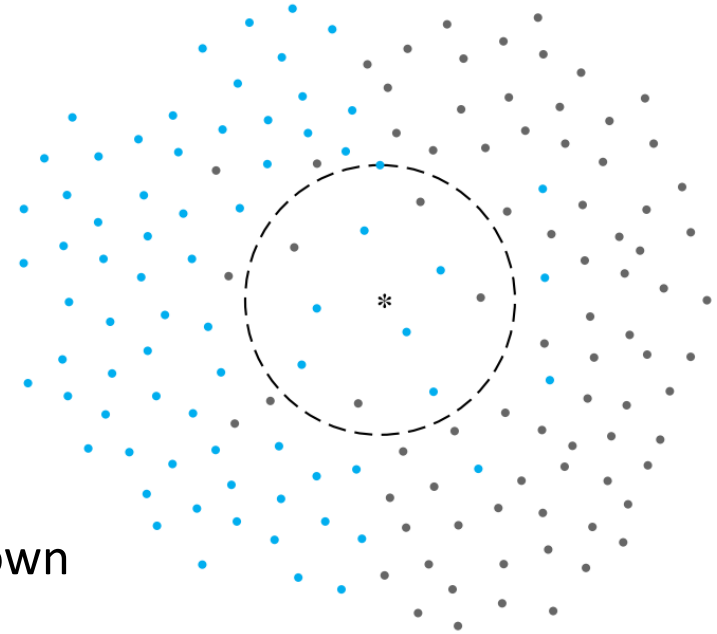
K Nearest Neighbour

KNN

The algorithm for the so-called *nearest neighbour rule* is summarized as follows.

Given an unknown feature vector ***x*** and a ***distance measure***, then:

- Out of the N training vectors, identify the k nearest neighbours, regardless of class label.
- Out of these *k* samples, identify the number of vectors, k_i that belong to class ω_i , $i = 1, 2, \ldots, M$.

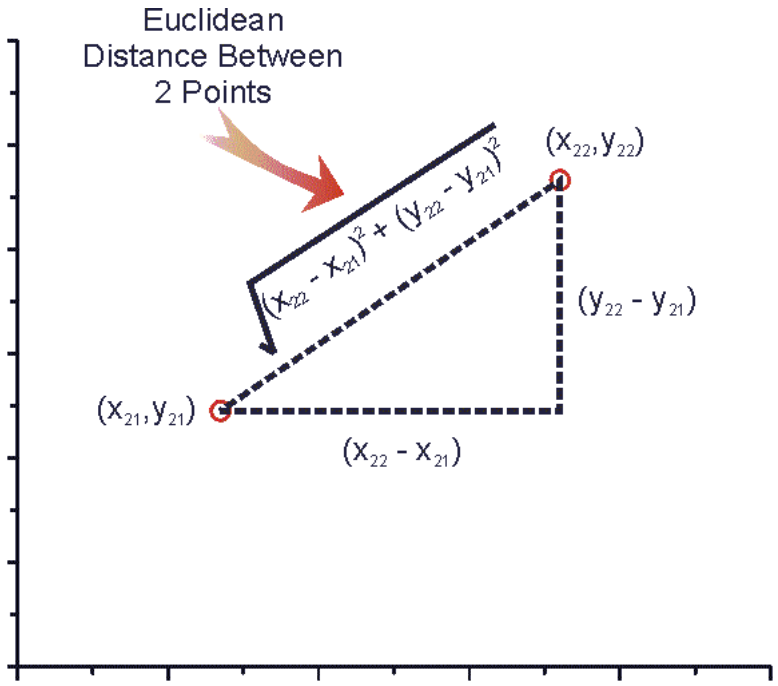


The simplest version of the algorithm is for $\mathrm{k} = 1$,known as the nearest neighbour (NN) rule

Distance (similarity) measure

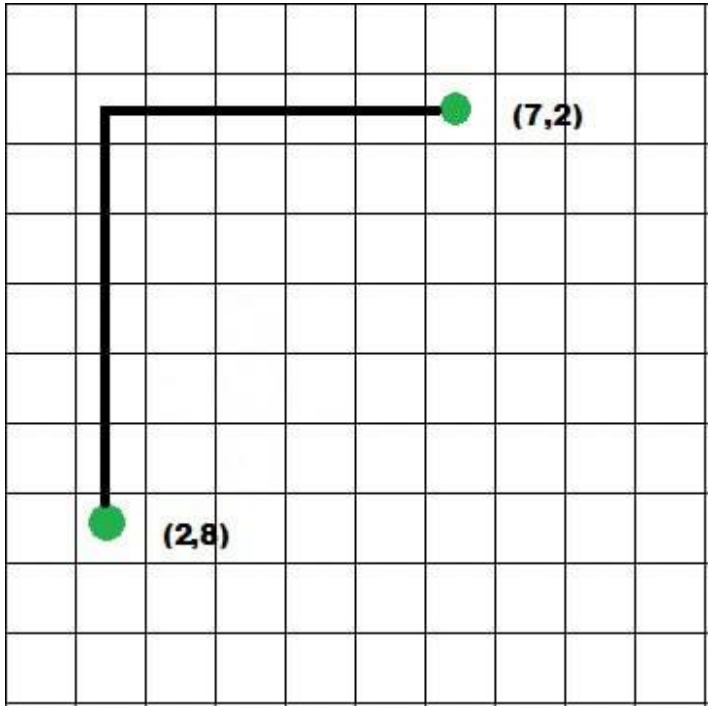
Euclidean distance:

$$d = \sqrt{\sum_{i=1}^{m}(x_i^1 - x_i^2)^2}$$



Manhattan distance:

$$d = \sum_{i=1}^{m}\left|x_i^{(1)} - x_i^{(2)}\right|$$

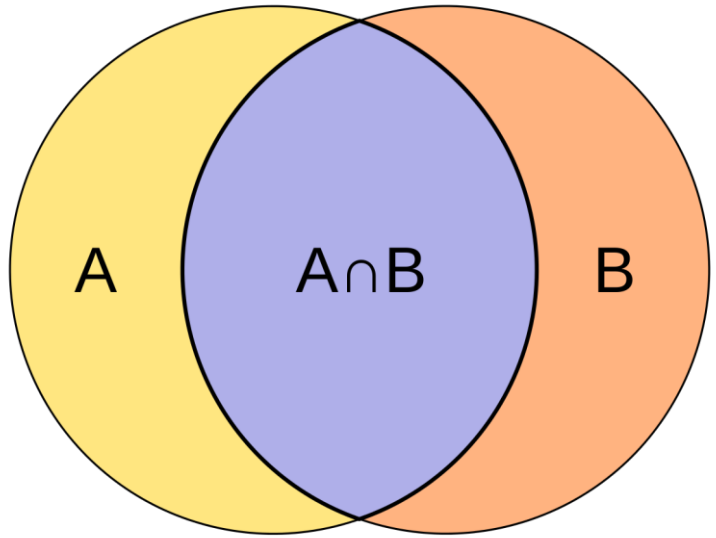


Distance (similarity) measure

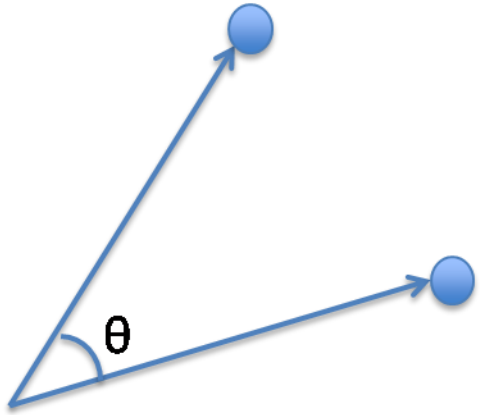
Minkowski distance:

$$d = \left(\sum_{i=1}^{m} (x_i^1 - x_i^2)^p \right)^{1/p}$$

Jaccard similarity



Cosine similarity:



$$sim(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$

Implementation