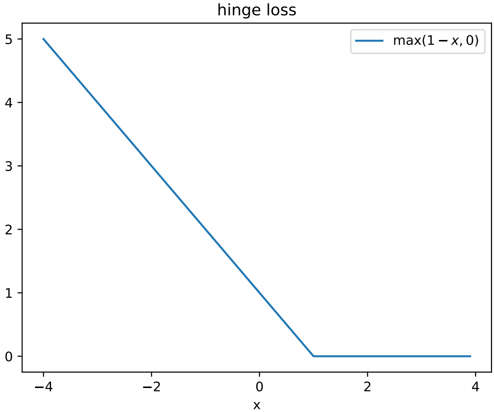
Machine learning II, unsupervised learning and agents: metrics



Metrics

Let $D = \{x_1, \dots, x_n\} \subset \mathcal{X}$ be a dataset of n samples, with labels $\{y_1, \dots, y_n\} \subset \mathcal{Y}$.
There is a metric in the input space $\mathcal{X}$ and in the output space $\mathcal{Y}$.

- The **metric** in $\mathcal{X}$ determines to what extent two samples x_i and x_j should be considered similar or dissimilar.
- The **metric** in $\mathcal{Y}$ determines to what extent two labels y_i and y_j should be considered similar or dissimilar.

In all machine learning, the choice of the metric is very important !

Metrics in output space

A **loss function** l is a map that measures the discrepancy between to elements of a set (for instance of a linear space).

$$l : \begin{cases} \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+ \\ (y, z) \mapsto l(y, z) \end{cases}$$

Typically, z can represent our prediction for a given input x, $z = \tilde{f}(x)$, and y the correct label.

Most common supervised learning losses

"0-1" loss for binary classification.
$\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{-1, 1\}$.

$$l(y, z) = 1_{y \neq z} \tag{1}$$

Squared loss for regression
$\mathcal{Y} = \mathbb{R}$.

$$l(y, z) = (y - z)^2 \tag{2}$$

absolute loss for regression.
$\mathcal{Y} = \mathbb{R}$.

$$l(y, z) = |y - z| \tag{3}$$

In **unsupervised learning**, there is notion of **output space** !

Geometric distances

Often, $\mathcal{X} = \mathbb{R}^p$ (input space). In this case, **geometric** metrics are used. $x = (x_1, ..., x_p)$ and $y = (y_1, ..., y_p)$ are p-dimensional **vectors**.

- L2 : $||x - y||_2 = \sqrt{\sum_{k=1}^{p}(x_k - y_k)^2}$ (Euclidian distance, 2-norm distance)
- L1 : $||x - y||_1 = \sum_{k=1}^{p}|x_k - y_k|$ (Manhattan distance, 1-norm distance)
- weighted L_1 : $\sum_{k=1}^{p} w_k|x_k - y_k|$
- $||x - y||_\infty$: $\max(|x_i - y_i|, i \in [1, n])$ (infinity norm distance, Chebyshev distance)

Choice of the metric

In some contexts, some usual metrics such as $L2$ might not be meaningful !

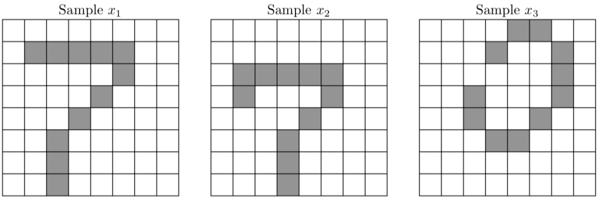


Figure – In $\mathbb{R}^{64}$, those three points form an equilateral triangle, [Fix et al., ,]

Non-geometric data

Not all data are geometric !

Hamming distance

- $\#\{x_i \neq y_i\}$ (Hamming distance)
- Levenshtein distance for strings (allows deletions and additions)

General definition of a distance

A **distance** on a set E is an application $d : E \times E \rightarrow \mathbb{R}_+$ that must :

- be **symetric** : $\forall x, y, d(x, y) = d(y, x)$
- **separate the values** : $\forall x, y, d(x, y) = 0 \Leftrightarrow x = y$
- respect the **triangular inequality**
  $\forall x, y, z, d(x, y) \leq d(x, z) + d(y, z)$

General definition of a distance

We could verify that :

- L2 is a distance
- Hamming is a distance

Similarities

Sometimes, it is not possible to define a proper **distance** in the input space $\mathcal{X}$! This may happen for instance is $\mathcal{X}$ is a dataset of texts.

- When distances are unavailable, we can use **Similarities** or **Dissimilarity** to compare points.
- Dissimilarites are more general and don't always abide by the distance axioms.
- Other examples : Adjacency in an oriented graph, Custom agregated score to compare data.

Example : cosine similarity

The **cosine similarity** may be used to compare texts.
If u and v are vectors,

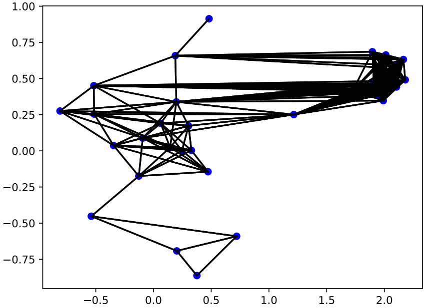
$$S_C(u, v) = \frac{(u|v)}{||u||||v||} \tag{4}$$

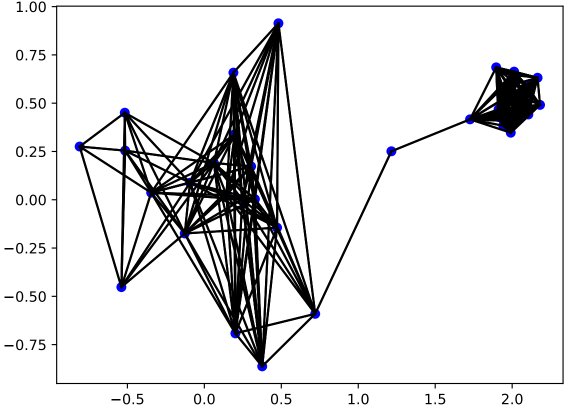
- the **bag of words representation** allows us to build a vector from a text (one hot encoding).
- **cosine_similarity/scraper.py**
- **cosine_similarity/similarity.py**

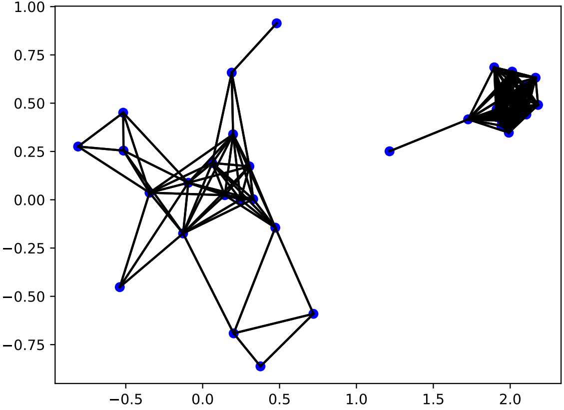
Hybrid data

Sometimes each sample contains both numerical data and non-numerical data (text, categorical data.)
See **hybrid_data/**
This is often the case in machine learning applications! (database of customers, cars, countries, etc.)

Exercice 1 : Using **metrics/geometric_data/build_graph_2.py**, choose the metric and the threshold so that this graph (and the ones on the next slides) are built.




1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.5
0.0
0.5
1.0
1.5
2.0


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−0.5
0.0
0.5
1.0
1.5
2.0

References I

Fix, J., Frezza-Buet, H., Geist, M., and Pennerath, F.
Machine Learning.pdf.