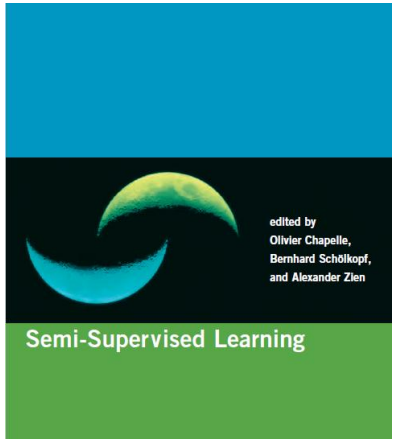
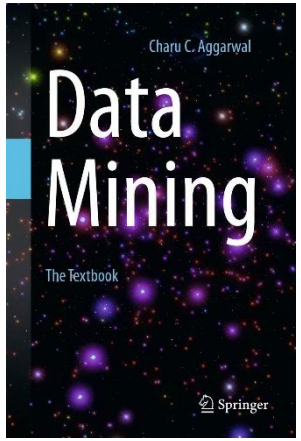
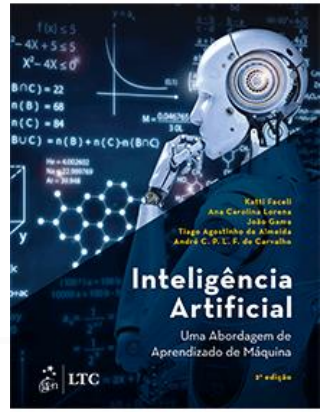




Aprendizado Semissupervisionado

Introdução



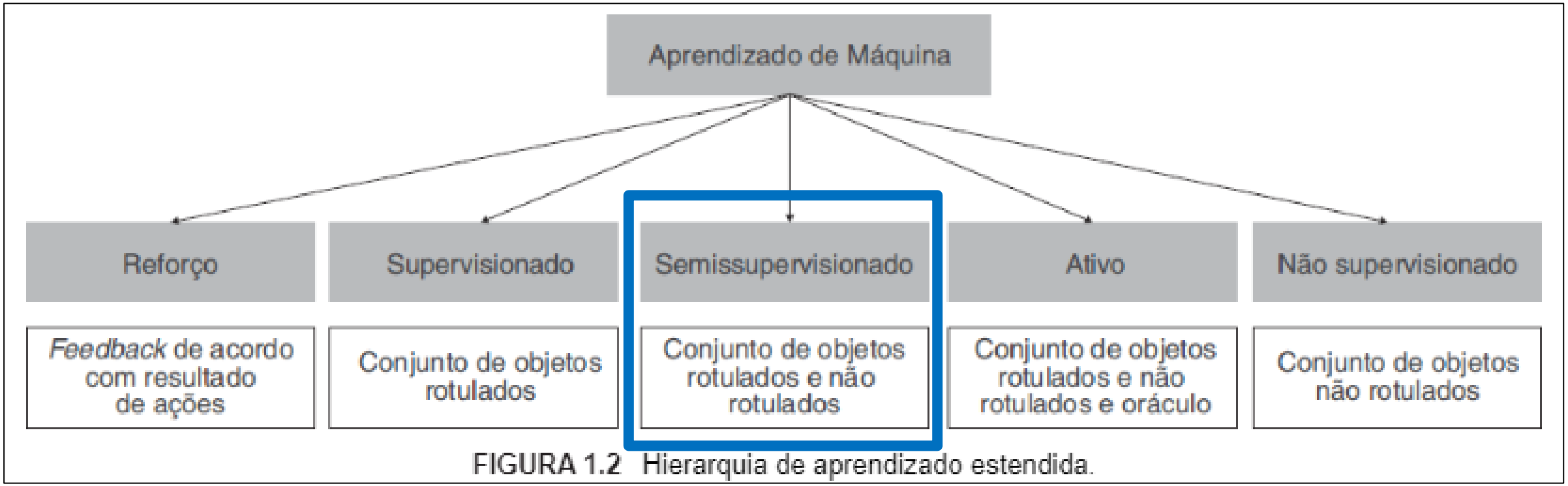
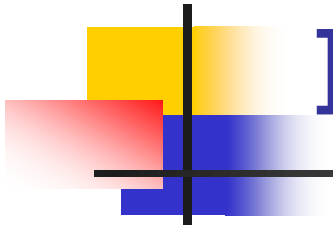


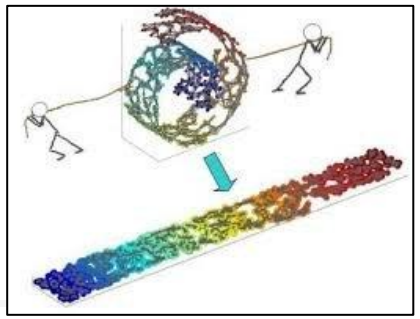
FIGURA 1.2 Hierarquia de aprendizado estendida.



Introdução

- In many applications, labeled data is expensive and hard to acquire
- On the other hand, unlabeled data is often available
- Semi-supervised learning will be useful whenever there are far more unlabeled data than labeled
  - It turns out that unlabeled data can be used to significantly improve the accuracy of many mining algorithms

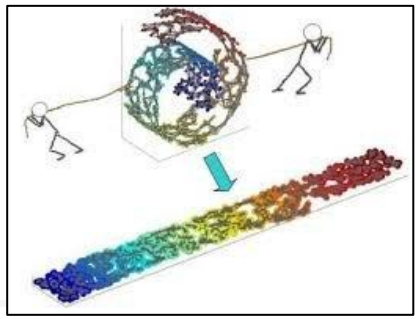
https://www.quora.com/What-is-manifold-learning



Introdução

- Unlabeled data is useful because of the following two reasons:
  - Unlabeled data can be used to estimate the low-dimensional manifold structure of the data
    - Mesmo sem saber as classes, os dados sem rótulo ajudam o modelo a entender o formato, i.e., a estrutura natural do espaço de dados
      - Imagine que cada ponto de dado é uma estrela no céu. Você ainda não sabe quais são "constelações" (rótulos), mas consegue ver como as estrelas se agrupam naturalmente em desenhos. Esses agrupamentos formam a estrutura (ou manifold) onde os dados vivem
        - Com os dados não rotulados, o modelo descobre essa forma — "o mapa" do espaço de dados
        - Com os dados rotulados, ele coloca nomes nos pedaços desse mapa

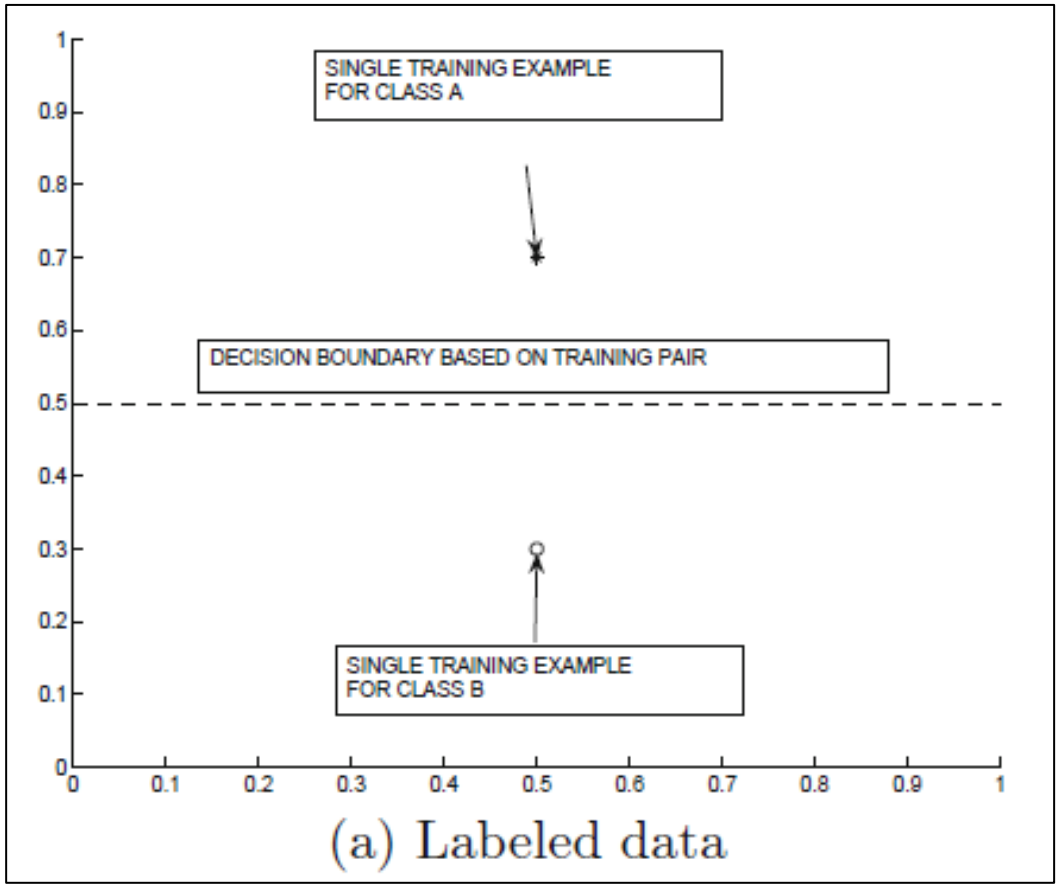
https://www.quora.com/What-is-manifold-learning



Introdução

- Unlabeled data is useful because of the following two reasons (cont.):
  - Unlabeled data can be used to estimate the joint probability distribution of features, useful for indirectly relating feature values to labels
    - Os dados não rotulados ajudam o modelo a entender como as variáveis se relacionam entre si — mesmo sem saber as classes
      - Imagine que você observa centenas de frutas, sem saber o nome delas. Você nota padrões: (i) frutas mais vermelhas costumam ser menores; (ii) frutas mais amarelas são mais compridas; etc.
        - Mesmo sem rótulo (“maçã”, “banana”, etc.), você já entende como as características se combinam. Depois, quando aparece pouca informação rotulada, o modelo já sabe como as características se distribuem, e isso facilita ligar padrões a rótulos

Introdução



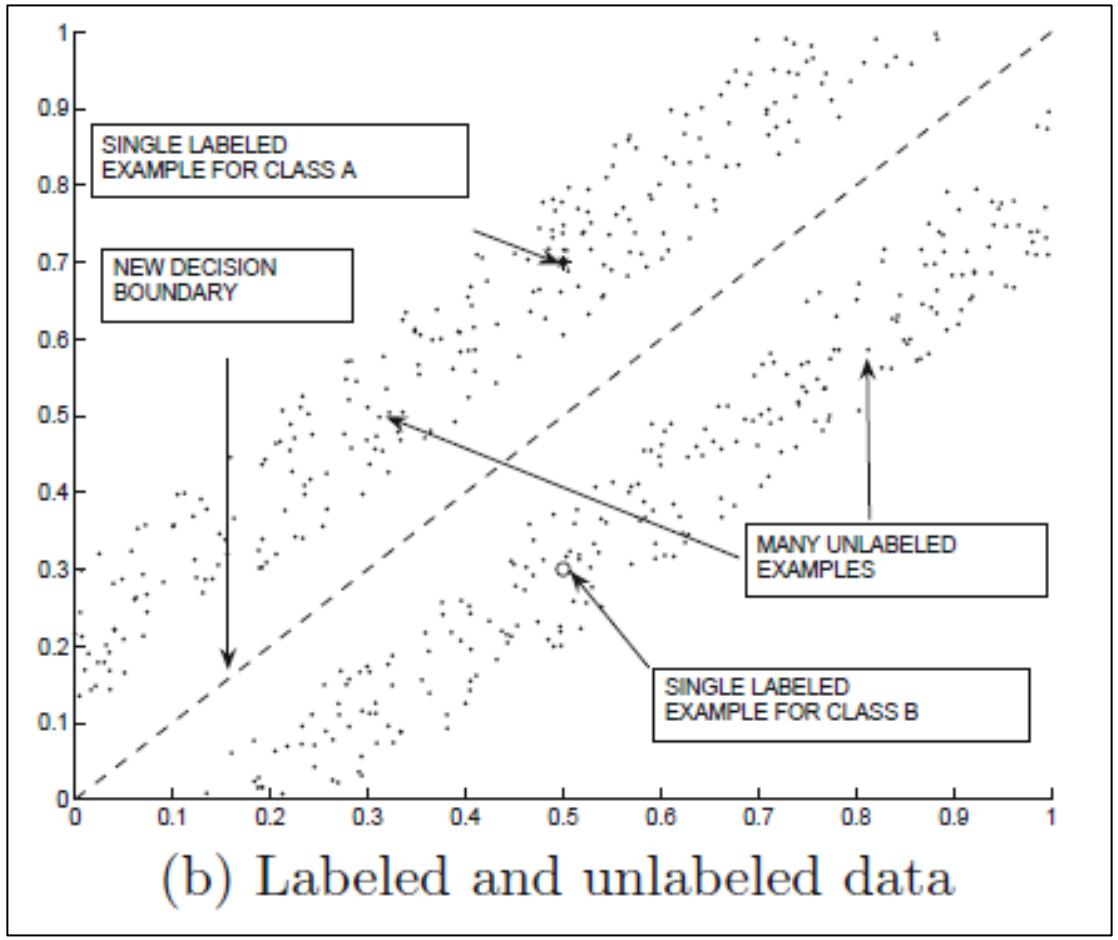
(a) Labeled data

Portions of this decision boundary are in regions of the space where almost no feature values are available

Therefore, the decision boundaries in these regions may not reflect the class behavior of unseen test instances

Introdução

Suppose that a large number of unlabeled examples are added to the training data

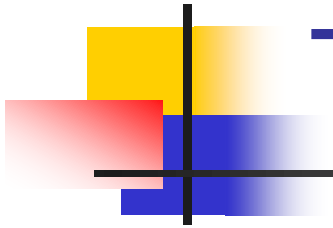


(b) Labeled and unlabeled data

It becomes immediately evident that the data is distributed along two manifolds, each of which contains one of the training examples

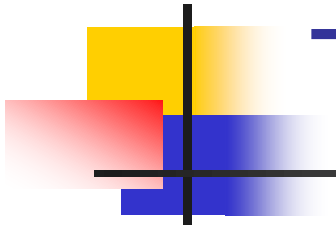
A key assumption here is that the class variables are likely to vary smoothly over dense regions of the space, but it may vary significantly over sparse regions of the space

Leads to a new decision boundary that takes the underlying feature correlations into account in addition to the labeled instances



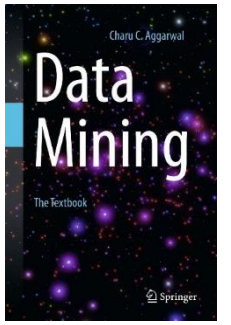
Transduction x Induction

- An inductive method infers a function $f : X \rightarrow Y$ on the entire space X, and afterward returns the evaluations $f(x_i)$ at the test points



Transduction x Induction

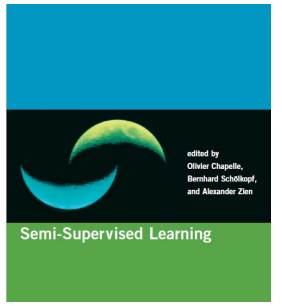
- Transduction consists of directly estimating the finite set of test labels, i.e., a function $f : X_u \rightarrow Y$ only defined on the test set
  - All test instances need to be specified at the time of constructing the training model
    - New out-of-sample instances cannot be classified after the model has been constructed
  - Transduction is not the same as SSL: some semi-supervised algorithms are transductive, but others are inductive



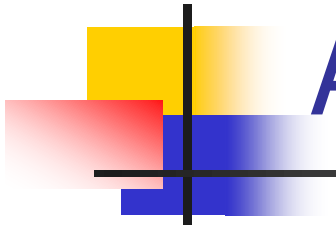
Techniques

- Meta-algorithms
  - Uses any existing classification algorithm as a subroutine, and leverage it to incorporate the impact of unlabeled data
  - Self-training, Co-training
- Variations of Classification Algorithms
  - Modifications are incorporated in specific classifiers to account for the impact of unlabeled data
  - Semisupervised Bayes Classification with EM **[2]**, TSVM **[3]**, Graph-Based **[4]**

Techniques

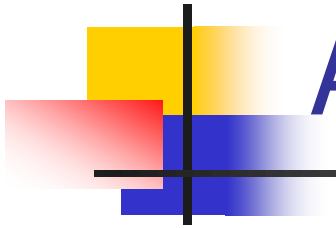


- For semi-supervised learning to work, certain assumptions will have to hold
  - Smoothness assumption **[1]**
    - If two points x_1, x_2 in a high-density region are close, then so should be the corresponding outputs y_1, y_2
  - Cluster assumption **[2]**
  - Low density separation assumption **[3]**
  - Manifold assumption **[4]**



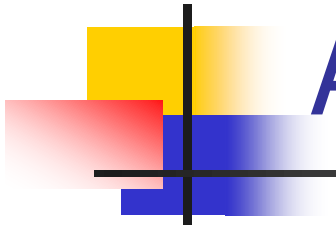
Assumptions

- **Cluster assumption [2]**
  - If points are in the same cluster, they are likely to be of the same class
  - Algoritm: Generative Models
    - Semisupervised Bayes Classification with EM



Assumptions

- **Low density separation assumption [3]**
  - The decision boundary should lie in a low-density region
    - Although equivalent with assumption [2], they can inspire different algorithms
  - Algorithm: TSVM (Transductive SVM)

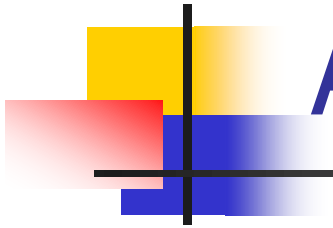


Assumptions

- **Manifold assumption [4]**
  - The (high-dimensional) data lie (roughly) on a low-dimensional manifold
    - Curse of dimensionality = pairwise distances tend to become more similar, and thus less expressive
    - Ensures more appropriate distances and, thus, "if two points x_1, x_2 are close, then so should be the corresponding outputs y_1, y_2" (smoothness assumption of supervised learning)

https://en.wikipedia.org/wiki/Distance_(graph_theory)

The **distance** between two vertices in a graph is the number of edges in a shortest path (also called a **graph geodesic**) connecting them



Assumptions

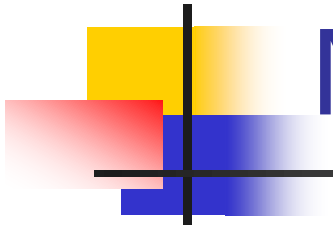
- **Manifold assumption [4]**
  - The most active area of research in semi-supervised learning has been in graph-based methods
  - Graph methods can be argued to build on the manifold assumption
    - If the distance of two points is computed by minimizing the path distance over all paths connecting the two points, this can be seen as an approximation of the geodesic distance of the two points with respect to the manifold of data points

Self-Training: A Survey [Paper]
Enhancing Self-Training Methods [Paper]

https://scikit-learn.org/stable/modules/semi_supervised.html

Meta-algorithms: Self-training

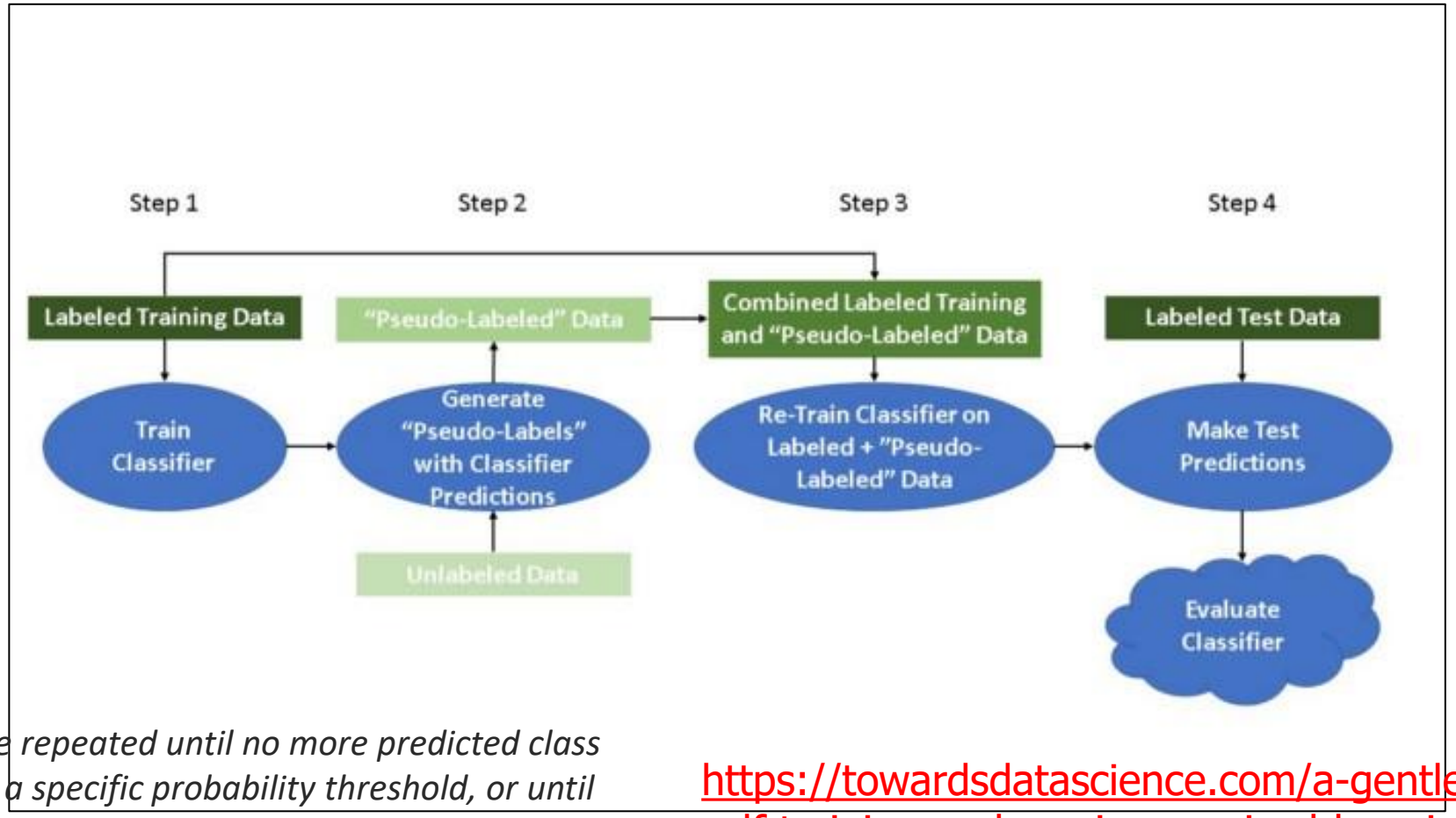
- The self-training procedure can use any existing classification algorithm A as input
- The classifier A is used to incrementally assign labels to unlabeled examples for which it has the most confident prediction
- As input, the self-training procedure uses the initial labeled set L, the unlabeled set U, and a user-defined parameter k that may sometimes be set to 1



Meta-algorithms: Self-training

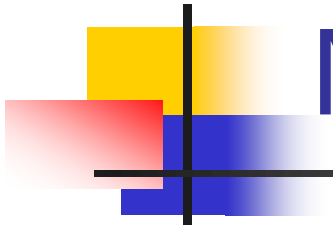
- The self-training procedure iteratively uses the following steps:
  - (a) use algorithm A on the current labeled set L to identify the k instances in the unlabeled data U for which the classifier A is the most confident
  - (b) assign labels to the k most confidently predicted instances and add them to L. Remove these instances from U

Meta-algorithms: Self-training



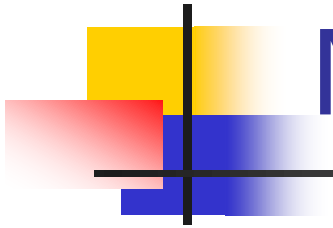
Steps 1 through 4 can be repeated until no more predicted class labels from Step 2 meet a specific probability threshold, or until no more unlabeled data remains

https://towardsdatascience.com/a-gentle-introduction-to-self-training-and-semi-supervised-learning-ceee73178b38



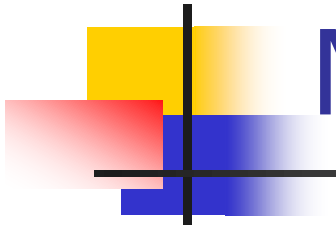
Meta-algorithms: Self-training

- The major drawback of self-training is that the addition of predicted labels to the training data can lead to propagation of errors in the presence of noise



Meta-algorithms: Co-training

- In Co-training, it is assumed that the feature set can be partitioned into two disjoint groups F_1 and F_2, such that each of them is sufficient to learn the target classification function
- It is important to select the two feature subsets so that they are as independent from one another as possible



Meta-algorithms: Co-training

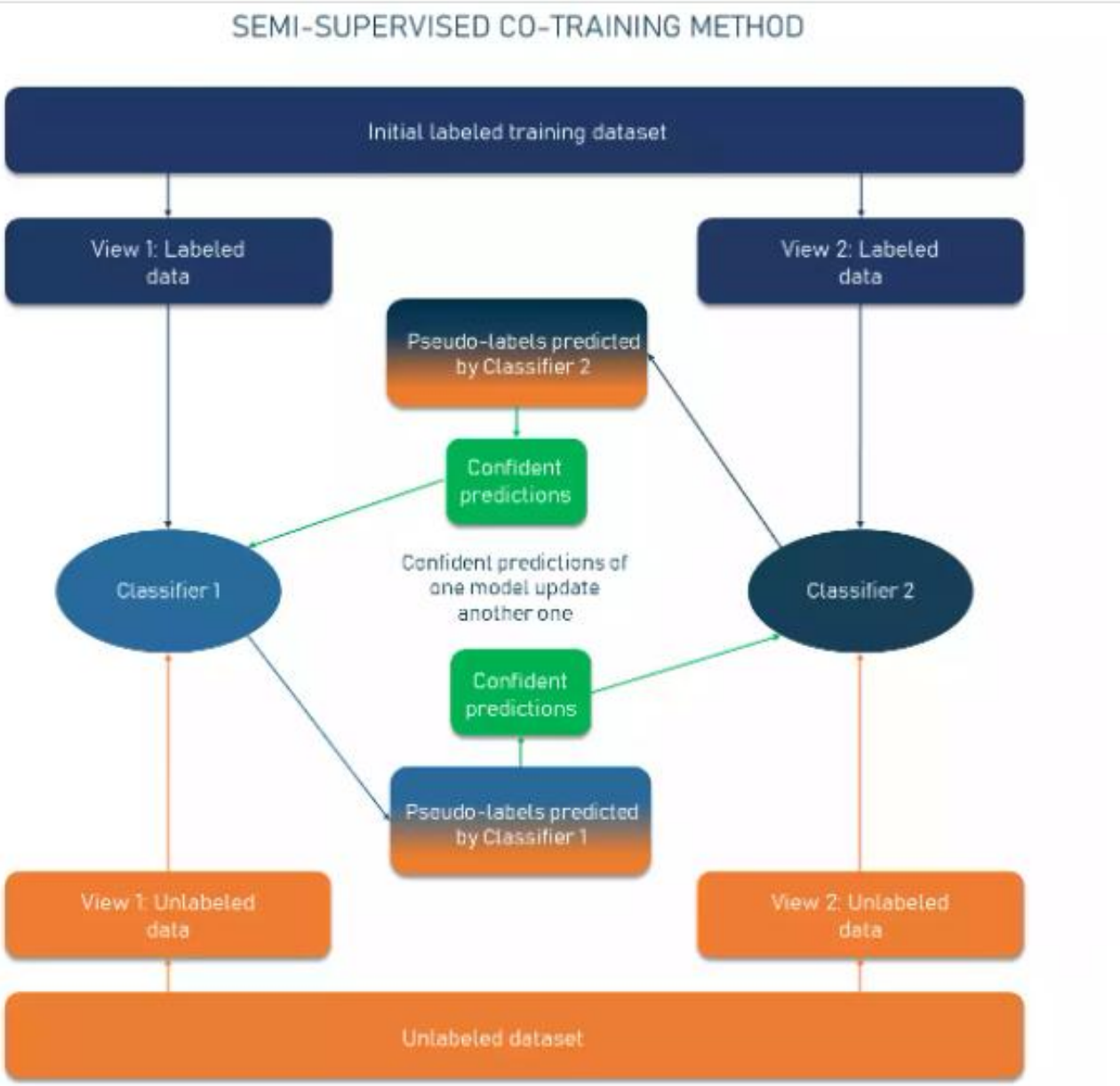
- Two classifiers are constructed, such that one classifier is constructed on each of these groups
- These classifiers are not allowed to interact with one another directly for prediction of unlabeled examples though they are used to build up training sets for each other
- This is the reason that the approach is referred to as Co-training

Meta-algorithms: Co-training

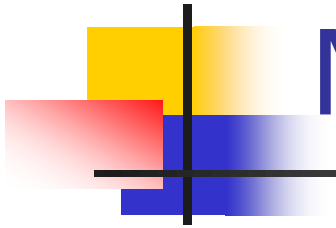
- The Co-training procedure iteratively uses the following steps:
  - (a) train classifier A_1 using labeled set L_1, and add k most confidently predicted instances from unlabeled set $U - L_2$ to training data set L_2 for classifier A_2
  - (b) train classifier A_2 using labeled set L_2, and add k most confidently predicted instances from unlabeled set $U - L_1$ to training data set L_1 for classifier A_1

The sets L_1 and L_2 are initialized to the available labeled data L. As different examples from the initially unlabeled set U are added to L1 and L2, respectively, the training instances in L_1 and L_2 may vary from one another

Meta-algorithms: Co-training

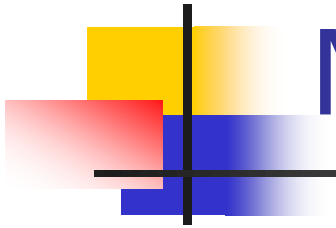


https://www.altexsoft.com/blog/semi-supervised-learning/ altexsoft



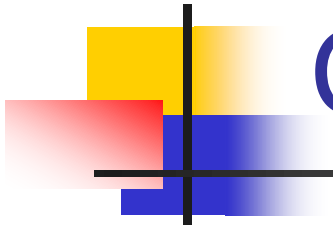
Meta-algorithms: Co-training

- At the end, the two classifiers are then retrained with the expanded training data sets and, therefore, two classifiers are returned
  - Can be used to label not only the unlabeled data set U, but also unseen test instances
- For an unseen test instance, each classifier may be used to determine the class label scores
- The score for a test instance is determined by combining the scores of the two classifiers
  - For example, if the Bayes method is used as the base classifier, then the product of the posterior probabilities returned by the two classifiers may be used



Meta-algorithms: Co-training

- The Co-training approach is more robust to noise because of the disjoint feature sets used by the two algorithms
  - An important assumption is that of conditional independence of the features in the two sets with respect to a particular class
  - The intuition is that instances generated by one classifier appear to be randomly distributed to the other, and vice-versa



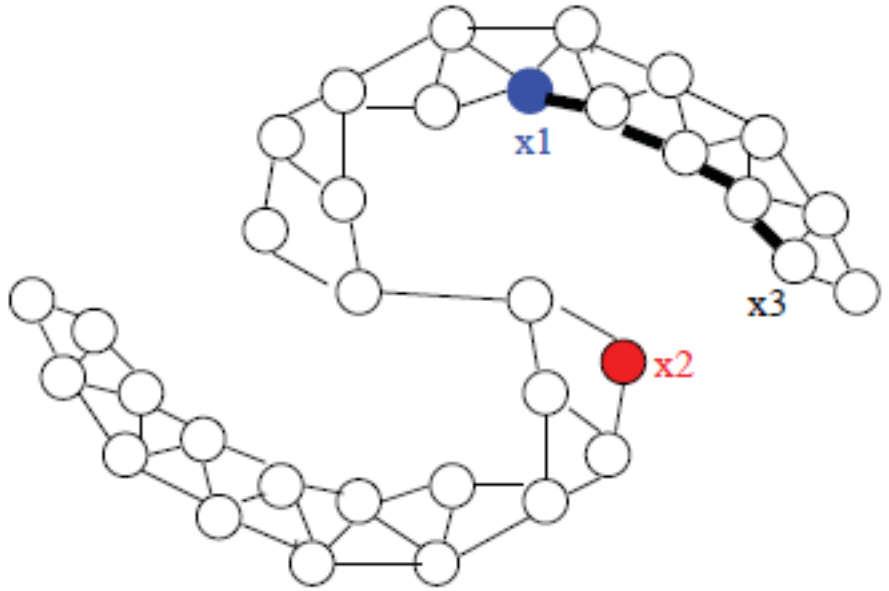
Graph-Based

- One advantage of this approach is that it can be used for semisupervised classification of arbitrary data types, as long as a distance function is available for quantifying proximity between data objects

https://scikit-learn.org/stable/modules/semi_supervised.html

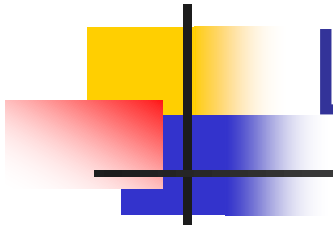
Graph-Based

- Label Propagation
  - Vértices similares devem possuir a mesma classe
  - Dois passos:
    - Construção do grafo
    - Propagação dos rótulos



Introduction to Semi-Supervised Learning [ref.]

Graph-Based Label Propagation



- Construção do Grafo
  - ε-vizinhança
  - κ-vizinhos mais próximos (κNN)

Graph-Based Label Propagation

- Construção do Grafo
  - **ε-vizinhança**
    - Dois vértices x_i e x_j são conectados se a distância entre os mesmos é menor do que ε (threshold)
    - Em geral, não é muito utilizada, pois na maioria das vezes gera-se uma rede desconexa
      - Alguns vértices não possuirão ligação com vértices rotulados

Graph-Based Label Propagation

- Construção do Grafo
  - **κ-vizinhos mais próximos (κNN)**
    - Dois vértices x_i e x_j são conectados se x_j está entre os κ vizinhos mais próximos de x_i
      - A matriz de adjacência pode não ser simétrica, pois x_j ser vizinho de x_i não implica em x_i ser vizinho de x_j
        - κNN mútuo
        - κNN simétrico

Graph-Based Label Propagation

- Construção do Grafo
  - **κ-vizinhos mais próximos (κNN)**
    - κNN mútuo
      - Verificam-se os casos em que existe uma conexão entre x_i e x_j e não existe uma conexão entre x_j e x_i e adiciona-se uma conexão entre x_j e x_i
    - κNN simétrico
      - Verificam-se os casos em que existe uma conexão entre x_i e x_j e não existe uma conexão entre x_j e x_i e remove-se a conexão entre x_j e x_i

Graph-Based Label Propagation

https://en.wikipedia.org/wiki/Radial_basis_function_kernel

https://scikit-learn.org/stable/modules/semi_supervised.html

- Construção do Grafo
  - Existem métodos que alteram o peso das conexões
    - Visam ponderar os valores das conexões entre dois vértices
      - Kernel RBF (medida de similaridade [0, 1])
        - Aplica uma função Gaussiana no peso, ponderando seu valor de acordo com a abertura σ da Gaussiana

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right)$$

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$$

$$\gamma = \frac{1}{2\sigma^2}$$

Graph-Based Label Propagation

Learning from labeled and unlabeled data with label propagation
https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

- Setup
  - $(x_1,\ldots,y_1) \ldots (x_l,\ldots,y_l)$ = labeled data, $Y_L = \{y_1,\ldots,y_l\}$ = class labels
    - Assume that the number of classes C is known and that all classes are present in the labeled data
  - $(x_{l+1},\ldots,y_{l+1}) \ldots (x_{l+u},\ldots,y_{l+u})$ = unlabeled data, $Y_U = \{y_{l+1},\ldots,y_{l+u}\}$ are unknown, $u \gg l$
  - $X = \{x_1,\ldots,x_{l+u}\}$
- Problem
  - Estimate Y_U from X and Y_L

Graph-Based Label Propagation

Learning from labeled and unlabeled data with label propagation
https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

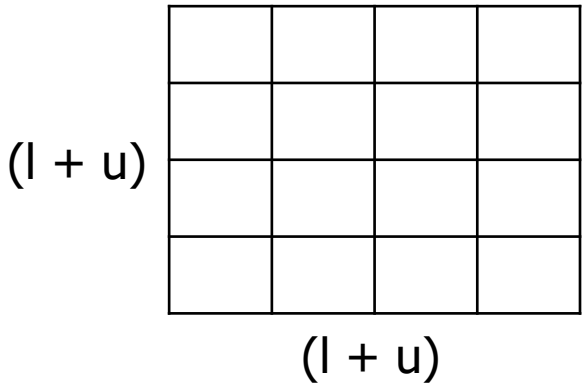
- Construção do Grafo
  - Create a fully connected graph where the nodes are all data points – labeled+unlabeled
  - W_{ij} é computada via distância euclidiana
  - W_{ij} é normalizada via Kernel RBF

Graph-Based Label Propagation

Learning from labeled and unlabeled data with label propagation
https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

- Propagação
  - All nodes have soft labels = a distribution over labels
    - Thus, labels of a node are propagate to all nodes through the edges

Matriz T (n X n)



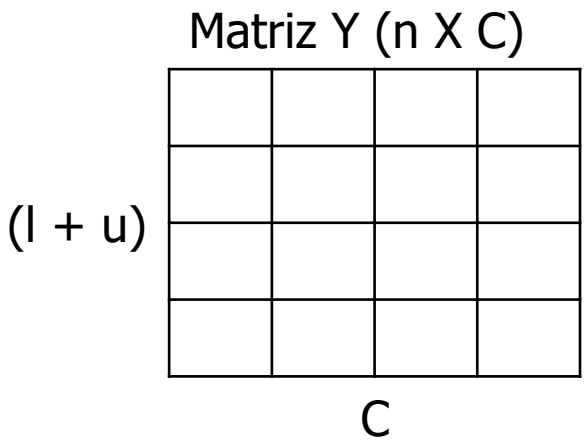
$$T_{ij} = P(j \rightarrow i) = \frac{w_{ij}}{\sum_{k=1}^{l+u} w_{kj}}$$

T = probabilistic transition matrix
T_{ij} = probability to jump from node j to i

Graph-Based Label Propagation

Learning from labeled and unlabeled data with label propagation
https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

- Propagação
  - All nodes have soft labels = a distribution over labels
    - Thus, labels of a node are propagate to all nodes through the edges



Y = label matrix
Each row representes the label probability distribution of node x_i
Initialization of rows corresponding to unlabeled data points is random

Graph-Based Label Propagation

Learning from labeled and unlabeled data with label propagation
https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

- Algoritmo

1. Propagate $Y \leftarrow TY$
2. Row-normalize Y.
3. Clamp the labeled data. Repeat from step 1 until Y converges.

It is importante to maintain labels sources from labeled data

https://machinelearningmastery.com/semi-supervised-learning-with-label-propagation/

Graph-Based Label Propagation

Introduction to Semi-Supervised Learning [ref.]

Separar os dados em dois conjuntos de rótulo, de modo a cortar arestas de menor peso total. Isso significa que ele procura evitar separar pontos muito similares, respeitando a ideia de suavidade local (smoothness).

- Existem vários algoritmos

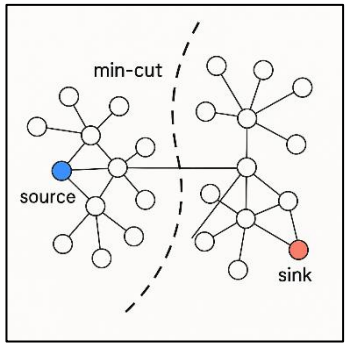
$$\sum_{i,j:f(\mathbf{x}_i)\neq f(\mathbf{x}_j)} w_{ij}.$$

The "cut size" is measured by the sum of the weights on the edges defining the cut

$$\min_{f:f(\mathbf{x})\in\{-1,1\}} \infty \sum_{i=1}^{l}(y_i - f(\mathbf{x}_i))^2 + \sum_{i,j=1}^{l+u} w_{ij}(f(\mathbf{x}_i) - f(\mathbf{x}_j))^2,$$

MinCut

-> duas classes ({+1, -1})
-> objetivo é encontrar o menor conjunto de arestas cuja remoção isole totalmente as duas classes



If x_i and x_j are not connected, then $w_{ij} = 0$
If the edge exists and is not cut, then $f(x_i) - f(x_j) = 0$

Graph-Based Label Propagation

Introduction to Semi-Supervised Learning [ref.]

$$\min_{f:f(\mathbf{x})\in\mathbb{R}} \sum_{i,j=1}^{l+u} w_{ij}(f(\mathbf{x}_i)-f(\mathbf{x}_j))^2 \quad \text{subject to} \quad f(\mathbf{x}_i)=y_i \ \text{ for } i=1\ldots l.$$

- Existem vários algoritmos

$$\min_{f:f(\mathbf{x})\in\mathbb{R}} \infty \sum_{i=1}^{l}(y_i-f(\mathbf{x}_i))^2 + \sum_{i,j=1}^{l+u} w_{ij}(f(\mathbf{x}_i)-f(\mathbf{x}_j))^2.$$

Now, f produces real values

GFHF (Gaussian Fields and Harmonic Functions)

-> The value assigned to each unlabeled vertex is the weighted average of its neighbors' values

$$f(\mathbf{x}_i) = y_i, \quad i=1\ldots l$$
$$f(\mathbf{x}_j) = \frac{\sum_{k=1}^{l+u} w_{jk} f(\mathbf{x}_k)}{\sum_{k=1}^{l+u} w_{jk}}, \quad j=l+1\ldots l+u.$$

Graph-Based Label Propagation

Learning with local and global consistency
https://proceedings.neurips.cc/paper_files/paper/2003/file/87682805257e619d49b8e0dfdc14affa-Paper.pdf

https://scikit-learn.org/stable/modules/semi_supervised.html

- Existem vários algoritmos

$$F = \mu \sum_{x_i \in L} (f_i - y_i)^2 + \frac{1}{2} \sum_{x_i; x_j \in D} w_{ij} \left\| \frac{f_i}{\sqrt{d_{ii}}} - \frac{f_j}{\sqrt{d_{jj}}} \right\|^2$$

LLGC (Learning with Local and Global Consistency)
-> d_{ii} = grau do vértice i

Permite que os rótulos originais se alterem

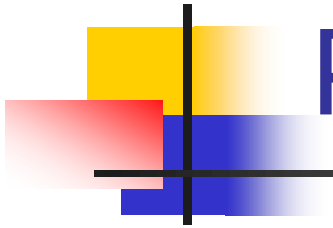
Evita que vértices com um alto grau não dominem a rede, i.e., diminuem sua influência na propagação das classes

Demais Referências

- A survey on semi-supervised learning
  - https://link.springer.com/article/10.1007/s10994-019-05855-6
- Introduction to Semi-Supervised Learning
  - https://link.springer.com/book/10.1007/978-3-031-01548-9
- Pós-processamento de regras de associação via redes e propagação de rótulos
  - https://teses.usp.br/teses/disponiveis/55/55134/tde-14102016-165710/pt-br.php

Implementações

- scikit-learn
  - https://scikit-learn.org/stable/modules/semi_supervised.html
- NetworkX
  - https://networkx.org/
  - https://networkx.org/documentation/stable/reference/algorithms/node_classification.html



Papers

- Graph-based Semi-supervised Learning: A Comprehensive Review
  - https://arxiv.org/abs/2102.13303
- A Survey on Deep Semi-supervised Learning
  - https://arxiv.org/abs/2103.00550
- A Survey of Data-Efficient Graph Learning
  - https://arxiv.org/abs/2402.00447