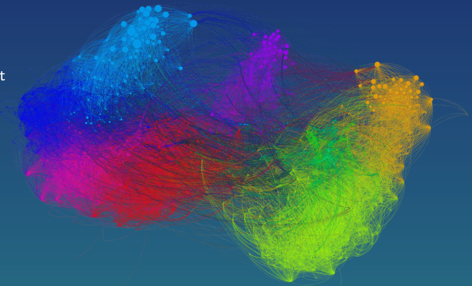
Graphs in Machine Learning

Michal Valko

DeepMind Paris and Inria Lille

TA: Omar Darwiche Domingues with the help of Pierre Perrault

Partially based on material by: Gary Miller,
Mikhail Belkin, Branislav Kveton,
Doyle & Schnell, Daniel Spielman



October 22nd, 2019

MVA 2019/2020

Graph nets lecture

- invited lecture by Marc Lelarge
- including 2019 material
- TD 3 the following week on graph nets
- questions from Marc
  - basic of deep learning?
  - deep learning course at MVA or elsewhere?
  - RNN?
  - VAE?

Previous Lecture

- spectral graph theory
- Laplacians and their properties
  - symmetric and asymmetric normalization
  - random walks
- geometry of the data and the connectivity
- spectral clustering

This Lecture

- manifold learning with Laplacians eigenmaps
- recommendation on a bipartite graph
- resistive networks
  - recommendation score as a resistance?
  - Laplacian and resistive networks
  - resistance distance and random walks
- Gaussian random fields and harmonic solution
- graph-based semi-supervised learning and manifold regularization
- transductive learning
- inductive and transductive semi-supervised learning

$$\mathbb{R}^d \longrightarrow \mathbb{R}^m$$

manifold learning

...discworld

Manifold Learning: Recap

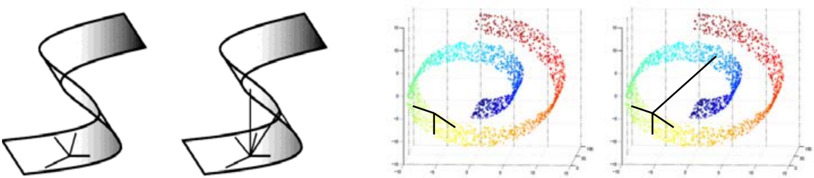
problem: definition reduction/manifold learning

Given $\{\mathbf{x}_i\}_{i=1}^N$ from $\mathbb{R}^d$ find $\{\mathbf{y}_i\}_{i=1}^N$ in $\mathbb{R}^m$, where $m \ll d$.

- What do we know about the **dimensionality reduction**
  - representation/visualization (2D or 3D)
  - an old example: globe to a map
  - often assuming $\mathcal{M} \subset \mathbb{R}^d$
  - feature extraction
  - linear vs. nonlinear dimensionality reduction
- What do we know about linear vs. nonlinear methods?
  - linear: ICA, PCA, SVD, ...
  - nonlinear often preserve only **local** distances

Manifold Learning: Linear vs. Non-linear

Manifold Learning: Preserving (just) local distances



$$d(\mathbf{y}_i, \mathbf{y}_j) = d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{only if} \quad d(\mathbf{x}_i, \mathbf{x}_j) \quad \text{is small}$$

$$\min \sum_{ij} w_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|^2$$

Looks familiar?

Manifold Learning: Laplacian Eigenmaps

Step 1: Solve generalized eigenproblem:

$$\mathbf{Lf} = \lambda \mathbf{Df}$$

Step 2: Assign m new coordinates:

$$\mathbf{x}_i \mapsto \left(f_2\left(i\right), \ldots, f_{m+1}\left(i\right)\right)$$

Note$_1$: we need to get $m+1$ smallest eigenvectors
Note$_2$: $\mathbf{f}_1$ is useless

http://web.cse.ohio-state.edu/~mbelkin/papers/LEM_NC_03.pdf

Manifold Learning: Laplacian Eigenmaps to 1D

Laplacian Eigenmaps 1D objective

$$\min_{\mathbf{f}} \mathbf{f}^{\mathsf{T}}\mathbf{L}\mathbf{f} \quad \text{s.t.} \quad f_i \in \mathbb{R}, \quad \mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{1} = 0, \quad \mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{f} = \mathbf{1}$$

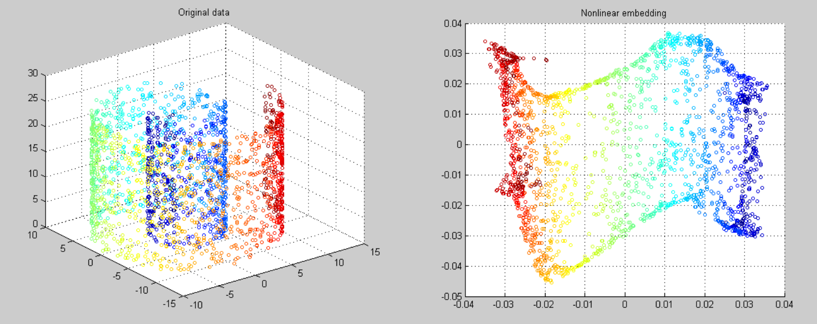
The meaning of the constraints is similar as for spectral clustering:

$\mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{f} = \mathbf{1}$ is for scaling

$\mathbf{f}^{\mathsf{T}}\mathbf{D}\mathbf{1} = 0$ is to not get $\mathbf{v}_1$

What is the solution?

Manifold Learning: Example



`http://www.mathworks.com/matlabcentral/fileexchange/`

`36141-laplacian-eigenmap-~-diffusion-map-~-manifold-learning`

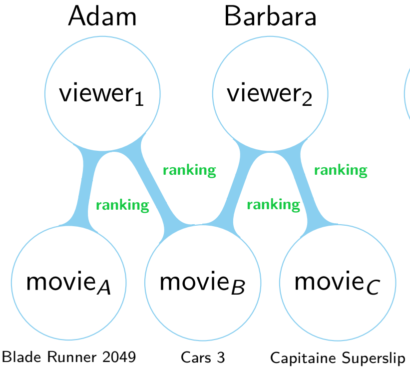
$$\mathrm{score}(v, m)$$

recommendation on a bipartite graph

...with the graph Laplacian!

Use of Laplacians: Movie recommendation

How to do movie recommendation on a bipartite graph?



Question: *Do we recommend Capitaine Superslip to Adam?*

Let's compute some $\text{score}(v, m)$!

Use of Laplacians: Movie recommendation

How to compute the $\mathrm{score}(v, m)$? Using some **graph distance**!

Idea$_1$: maximally weighted path

$\mathrm{score}(v, m) = \max_{vPm} \mathrm{weight}(P) = \max_{vPm} \sum_{e \in P} \mathrm{ranking}(e)$

Idea$_2$: change the path weight

$\mathrm{score}_2(v, m) = \max_{vPm} \mathrm{weight}_2(P) = \max_{vPm} \min_{e \in P} \mathrm{ranking}(e)$

Idea$_3$: consider everything

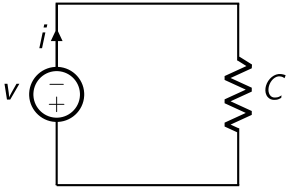
$\mathrm{score}_3(v, m) =$ max flow from m to v

Laplacians and Resistive Networks

How to compute the $\mathrm{score}(v, m)$?

Idea$_4$: view edges as conductors

$\mathrm{score}_4(v, m)$ = effective resistance between m and v



$C \equiv$ conductance

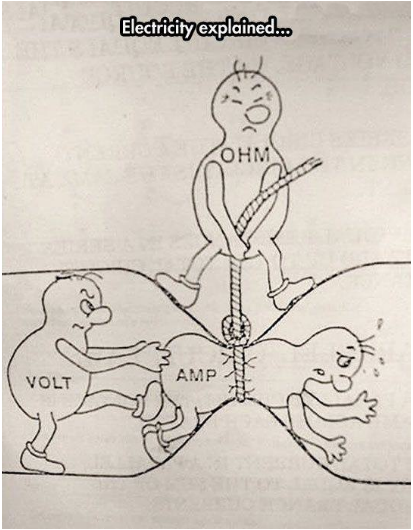
$R \equiv$ resistance

$i \equiv$ current

$V \equiv$ voltage

$$C = \frac{1}{R} \qquad i = CV = \frac{V}{R}$$

Resistive Networks: Some high-school physics



Resistive Networks

resistors in series

$$R = R_1 + \cdots + R_n \qquad C = \frac{1}{\frac{1}{C_1} + \cdots + \frac{1}{C_N}} \qquad i = \frac{V}{R}$$

conductors in parallel

$$C = C_1 + \cdots + C_N \qquad i = VC$$

Effective Resistance on a graph

Take two nodes: $a \neq b$. Let V_{ab} be the voltage between them and i_{ab} the current between them. Define $R_{ab} = \frac{V_{ab}}{i_{ab}}$ and $C_{ab} = \frac{1}{R_{ab}}$.

We treat the entire graph as a resistor!

Resistive Networks: Optional Homework (ungraded)

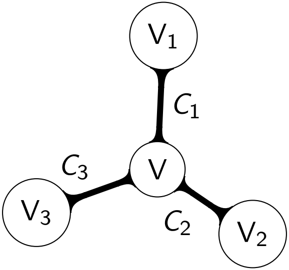
Show that R_{ab} is a metric space.

1. $R_{ab} \geq 0$
2. $R_{ab} = 0$ iff $a = b$
3. $R_{ab} = R_{ba}$
4. $R_{ac} \leq R_{ab} + R_{bc}$

The effective resistance is a distance!

How to compute effective resistance?

Kirchhoff's Law $\equiv$ flow in $=$ flow out



$V = \frac{C_1}{C}V_1 + \frac{C_2}{C}V_2 + \frac{C_3}{C}V_3$ (convex combination)

residual current $= CV - C_1V_1 - C_2V_2 - C_3V_3$

Kirchhoff says: This is zero! **There is no residual current!**

Resistors: Where is the link with the Laplacian?

General case of the previous! $d_i = \sum_j c_{ij} =$ sum of conductances

$$\mathbf{L}_{ij} = \begin{cases} d_i & \text{if } i = j, \\ -c_{ij} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

$\mathbf{v}$ = **voltage setting** of the nodes on graph.

$(\mathbf{Lv})_i$ = residual current at $\mathbf{v}_i$ — as we derived

Use: setting voltages and getting the current

Inverting $\equiv$ injecting current and getting the voltages

The net injected has to be zero $\equiv$ Kirchhoff's Law.

Resistors and the Laplacian: Finding R_{ab}

Let's calculate R_{1N} to get the **movie recommendation score**!

$$\mathbf{L}\begin{pmatrix} 0 \\ v_2 \\ \vdots \\ v_{n-1} \\ 1 \end{pmatrix} = \begin{pmatrix} i \\ 0 \\ \vdots \\ 0 \\ -i \end{pmatrix}$$

$$i = \frac{V}{R} \qquad V = 1 \qquad R = \frac{1}{i}$$

Return $R_{1N} = \dfrac{1}{i}$

Doyle and Snell: Random Walks and Electric Networks

`https://math.dartmouth.edu/~doyle/docs/walks/walks.pdf`

Resistors and the Laplacian: Finding R_{1N}

$\mathbf{Lv} = (i, 0, \ldots, -i)^\mathsf{T} \equiv$ **boundary valued problem**

For R_{1N}

V_1 and V_N are the **boundary**

$(v_1, v_2, \ldots, v_N)$ is **harmonic:**

$V_i \in$ **interior** (not boundary)

V_i is a **convex combination of its neighbors**

Resistors and the Laplacian: Finding R_{1n}

From the properties of electric networks (cf. Doyle and Snell) we inherit the useful properties of the Laplacians!

Example: Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions (later in the course)

Maximum Principle

If $\mathbf{f} = \mathbf{v}$ is harmonic then min and max are on the boundary.

Uniqueness Principle

If $\mathbf{f}$ and $\mathbf{g}$ are harmonic with the same boundary then $\mathbf{f} = \mathbf{g}$

Resistors and the Laplacian: Finding R_{1N}

Alternative method to calculate R_{1N}:

$$\mathbf{L}\mathbf{v} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ -1 \end{pmatrix} \stackrel{\text{def}}{=} \mathbf{i}_{\text{ext}} \quad \text{Return} \quad R_{1N} = v_1 - v_N$$

Why?

Question: Does $\mathbf{v}$ exist? $\mathbf{L}$ does not have an inverse :(.
Not unique: $\mathbf{1}$ in the nullspace of $\mathbf{L}$: $\mathbf{L}(\mathbf{v} + c\mathbf{1}) = \mathbf{L}\mathbf{v} + c\mathbf{L}\mathbf{1} = \mathbf{L}\mathbf{v}$
Moore-Penrose pseudo-inverse solves LS
Solution: Instead of $\mathbf{v} = \mathbf{L}^{-1}\mathbf{i}_{\text{ext}}$ we take $\mathbf{v} = \mathbf{L}^{+}\mathbf{i}_{\text{ext}}$
We get: $R_{1N} = v_1 - v_N = \mathbf{i}_{\text{ext}}^{\mathsf{T}}\mathbf{v} = \mathbf{i}_{\text{ext}}^{\mathsf{T}}\mathbf{L}^{+}\mathbf{i}_{\text{ext}}$.
Notice: We can reuse $\mathbf{L}^{+}$ to get resistances for any pair of nodes!

What? A pseudo-inverse?

Eigendecomposition of the Laplacian:

$$\mathbf{L} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^\mathsf{T} = \sum_{i=1}^{N} \lambda_i \mathbf{q}_i \mathbf{q}_i^\mathsf{T} = \sum_{i=2}^{N} \lambda_i \mathbf{q}_i \mathbf{q}_i^\mathsf{T}$$

Pseudo-inverse of the Laplacian:

$$\mathbf{L}^+ = \mathbf{Q}\boldsymbol{\Lambda}^+\mathbf{Q}^\mathsf{T} = \sum_{i=2}^{N} \frac{1}{\lambda_i} \mathbf{q}_i \mathbf{q}_i^\mathsf{T}$$

Moore-Penrose pseudo-inverse solves a least squares problem:

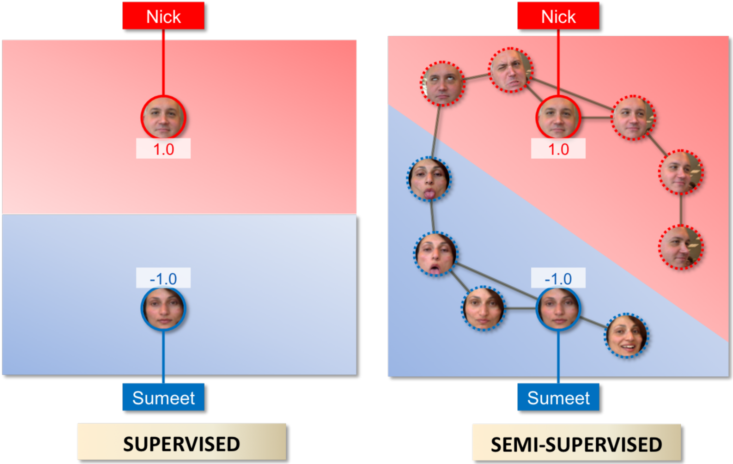
$$\mathbf{v} = \arg\min_{\mathbf{x}} \|\mathbf{L}\mathbf{x} - \mathbf{i}_{\mathrm{ext}}\|_2 = \mathbf{L}^+\mathbf{i}_{\mathrm{ext}}$$

SSL

semi-supervised learning

...our running example for learning with graphs

Semi-supervised learning: How is it possible?



This is how children learn! hypothesis

Semi-supervised learning (SSL)

SSL problem: definition

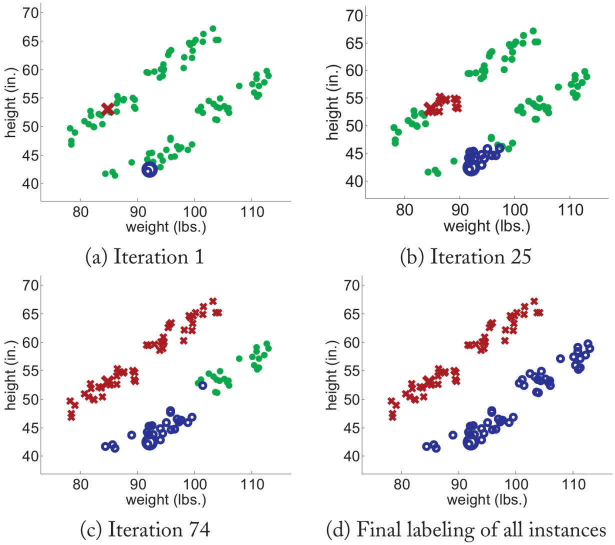
Given $\{\mathbf{x}_i\}_{i=1}^{N}$ from $\mathbb{R}^d$ and $\{y_i\}_{i=1}^{n_l}$, with $n_l \ll N$, find $\{y_i\}_{i=n_l+1}^{n}$ (**transductive**) or find f predicting y well beyond that (**inductive**).

Some facts about SSL

- assumes that the unlabeled data is useful
- works with data geometry assumptions
  - cluster assumption — low-density separation
  - manifold assumption
  - smoothness assumptions, generative models, ...
- now it helps now, now it does not (sic)
  - provable cases when it helps
- inductive or transductive/out-of-sample extension

http://olivier.chapelle.cc/ssl-book/discussion.pdf

SSL: Self-Training



(a) Iteration 1 (b) Iteration 25

(c) Iteration 74 (d) Final labeling of all instances

SSL: Overview: Self-Training

SSL: Self-Training

Input: $\mathcal{L} = \{\mathbf{x}_i, y_i\}_{i=1}^{n_l}$ and $\mathcal{U} = \{\mathbf{x}_i\}_{i=n_l+1}^{N}$

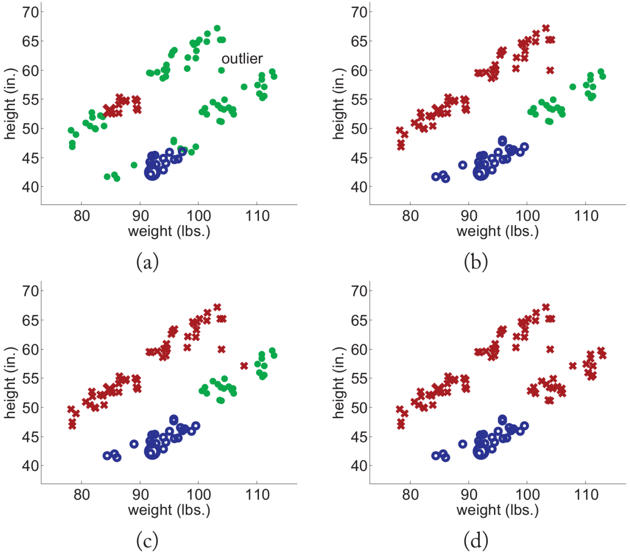
Repeat:

- train f using $\mathcal{L}$
- apply f to (some) $\mathcal{U}$ and add them to $\mathcal{L}$

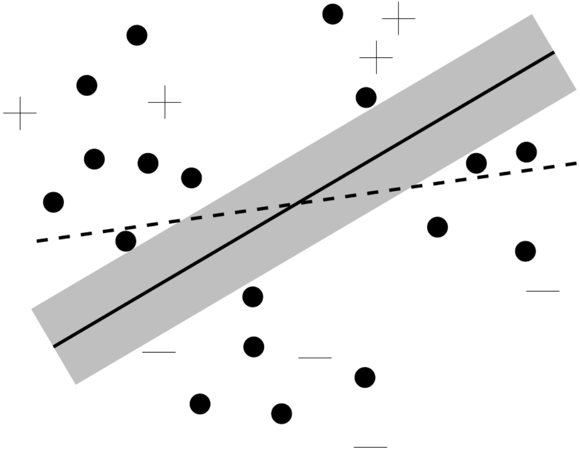
What are the properties of self-training?

- its a wrapper method
- heavily depends on the the internal classifier
- some theory exist for specific classifiers
- nobody uses it anymore
- errors propagate (unless the clusters are well separated)

SSL: Self-Training: Bad Case



SSL: Transductive SVM: S3VM



SSL: Transductive SVM: Classical SVM

Linear case: $f = \mathbf{w}^\mathsf{T}\mathbf{x} + b \quad \rightarrow \quad$ we look for $(\mathbf{w}, b)$

max-margin classification

$$\max_{\mathbf{w},b} \quad \frac{1}{\|\mathbf{w}\|}$$
$$s.t. \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 \quad \forall i = 1, \ldots, n_l$$

note the difference between functional and geometric margin

max-margin classification

$$\min_{\mathbf{w},b} \quad \|\mathbf{w}\|^2$$
$$s.t. \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 \quad \forall i = 1, \ldots, n_l$$

SSL: Transductive SVM: Classical SVM

max-margin classification: separable case

$$\min_{\mathbf{w},b} \quad \|\mathbf{w}\|^2$$
$$s.t. \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 \quad \forall i = 1, \dots, n_l$$

max-margin classification: non-separable case

$$\min_{\mathbf{w},b} \quad \lambda\|\mathbf{w}\|^2 + \sum_i \xi_i$$
$$s.t. \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1, \dots, n_l$$
$$\xi_i \geq 0 \quad \forall i = 1, \dots, n_l$$

SSL: Transductive SVM: Classical SVM

max-margin classification: non-separable case

$$\min_{\mathbf{w},b} \quad \lambda\|\mathbf{w}\|^2 + \sum_i \xi_i$$

$$s.t. \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1, \ldots, n_l$$

$$\xi_i \geq 0 \quad \forall i = 1, \ldots, n_l$$

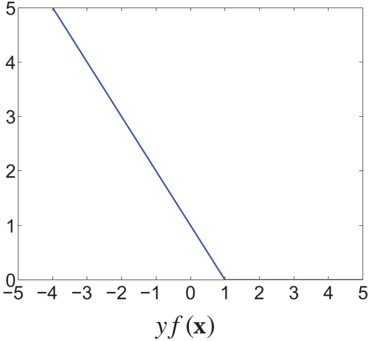
Unconstrained formulation using **hinge loss**:

$$\min_{\mathbf{w},b} \sum_i^{n_l} \max\left(1 - y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right), 0\right) + \lambda\|\mathbf{w}\|^2$$

In general?

$$\min_{\mathbf{w},b} \sum_i^{n_l} V(\mathbf{x}_i, y_i, f\left(\mathbf{x}_i\right)) + \lambda\Omega(f)$$

SSL: Transductive SVM: Classical SVM: Hinge loss



(a) the hinge loss

$$V(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) = \max\left(1 - y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right), 0\right)$$

SSL: Transductive SVM: Unlabeled Examples

$$\min_{\mathbf{w},b} \sum_{i}^{n_l} \max\left(1 - y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right), 0\right) + \lambda\|\mathbf{w}\|^2$$

How to incorporate unlabeled examples?

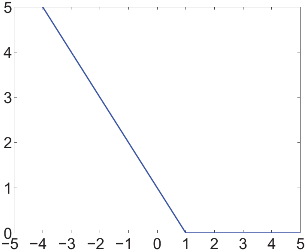
No y's for unlabeled $\mathbf{x}$.

Prediction of f for (any) $\mathbf{x}$? $\widehat{y} = \operatorname{sgn}\left(f\left(\mathbf{x}\right)\right) = \operatorname{sgn}\left(\mathbf{w}^\mathsf{T}\mathbf{x} + b\right)$

Pretending that $\operatorname{sgn}\left(f\left(\mathbf{x}\right)\right)$ is the true label ...

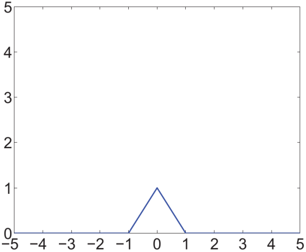
$$\begin{aligned} V(\mathbf{x}, \widehat{y}, f\left(\mathbf{x}\right)) &= \max\left(1 - \widehat{y}\left(\mathbf{w}^\mathsf{T}\mathbf{x} + b\right), 0\right) \\ &= \max\left(1 - \operatorname{sgn}\left(\mathbf{w}^\mathsf{T}\mathbf{x} + b\right)\left(\mathbf{w}^\mathsf{T}\mathbf{x} + b\right), 0\right) \\ &= \max\left(1 - \left|\mathbf{w}^\mathsf{T}\mathbf{x} + b\right|, 0\right) \end{aligned}$$

SSL: Transductive SVM: Hinge and Hat Loss



$yf(\mathbf{x})$

(a) the hinge loss



$f(\mathbf{x})$

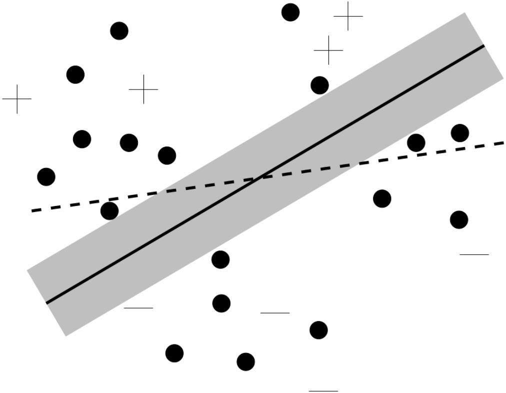
(b) the hat loss

What is the difference in the objectives?

Hinge loss penalizes?

Hat loss penalizes?

SSL: Transductive SVM: S3VM



This is what we wanted!

SSL: Transductive SVM: Formulation

Main SVM idea stays the same: penalize the margin

$$\min_{\mathbf{w},b} \sum_{i=1}^{n_l} \max\left(1 - y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right), 0\right) + \lambda_1 \|\mathbf{w}\|^2 + \lambda_2 \sum_{i=n_l+1}^{n_l+n_u} \max\left(1 - \left|\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right|, 0\right)$$

What is the loss and what is the regularizer?

$$\min_{\mathbf{w},b} \sum_{i=1}^{n_l} \max\left(1 - y_i\left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right), 0\right) + \lambda_1 \|\mathbf{w}\|^2 + \lambda_2 \sum_{i=n_l+1}^{n_l+n_u} \max\left(1 - \left|\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right|, 0\right)$$

Think of **unlabeled data** as the **regularizers** for your classifiers!

Practical hint: Additionally enforce the class balance.

What it the main issue of TSVM?

recent advancements: `http://jmlr.org/proceedings/papers/v48/hazanb16.pdf`

SSL($\mathcal{G}$)

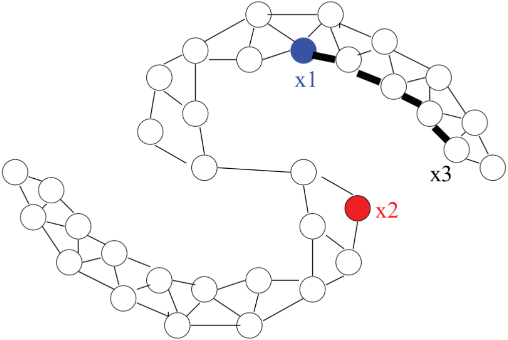
semi-supervised learning with graphs and harmonic functions

...our running example for learning with graphs

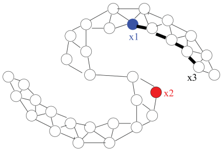
SSL with Graphs: Prehistory

Blum/Chawla: Learning from Labeled and Unlabeled Data using Graph Mincuts
`http://www.aladdin.cs.cmu.edu/papers/pdfs/y2001/mincut.pdf`

*following some insights from vision research in 1980s



SSL with Graphs: MinCut



MinCut SSL: an idea similar to MinCut clustering

Where is the link?

What is the formal statement? We look for $f(\mathbf{x}) \in \{\pm 1\}$

$$\mathrm{cut} = \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2 = \Omega(f)$$

Why $\left(f\left(\mathbf{x}_i\right) - f\left(\mathbf{x}_j\right)\right)^2$ and not $|f(\mathbf{x}_i) - f(\mathbf{x}_j)|$?

SSL with Graphs: MinCut

We look for $f(\mathbf{x}) \in \{\pm 1\}$ to minimize the cut $\Omega(\mathbf{f})$

$$\Omega(\mathbf{f}) = \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2$$

Clustering was unsupervised, here we have supervised data.

Recall the general objective-function framework:

$$\min_{\mathbf{w},b} \sum_{i}^{n_l} V(\mathbf{x}_i, y_i, f(\mathbf{x}_i)) + \lambda \Omega(\mathbf{f})$$

It would be nice if we match the prediction on labeled data:

$$V(\mathbf{x}, y, f(\mathbf{x})) = \infty \sum_{i=1}^{n_l} \left(f(\mathbf{x}_i) - y_i\right)^2$$

SSL with Graphs: MinCut

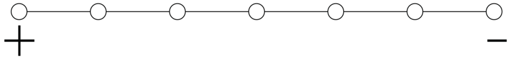
Final objective function:

$$\min_{\mathbf{f}\in\{\pm 1\}^{n_l+n_u}} \infty \sum_{i=1}^{n_l} (f(\mathbf{x}_i) - y_i)^2 + \lambda \sum_{i,j=1}^{n_l+n_u} w_{ij} (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2$$

This is an integer program :(

Can we solve it?

Are we happy?



We need a better way to reflect the confidence.

SSL with Graphs: Harmonic Functions

Zhu/Ghahramani/Lafferty: Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions (ICML 2013)

http://mlg.eng.cam.ac.uk/zoubin/papers/zgl.pdf

*a seminal paper that convinced people to use graphs for SSL

Idea 1: Look for a **unique** solution.

Idea 2: Find a smooth one. (**harmonic** solution)

Harmonic SSL

1): As before, we constrain f to match the supervised data:

$$f(\mathbf{x}_i) = y_i \qquad \forall i \in \{1, \dots, n_l\}$$

2): We enforce the solution f to be harmonic:

$$f(\mathbf{x}_i) = \frac{\sum_{i\sim j} f(\mathbf{x}_j) w_{ij}}{\sum_{i\sim j} w_{ij}} \qquad \forall i \in \{n_l + 1, \dots, n_u + n_l\}$$

SSL with Graphs: Harmonic Functions

The harmonic solution is obtained from the mincut one ...

$$\min_{\mathbf{f}\in\{\pm 1\}^{n_l+n_u}} \infty \sum_{i=1}^{n_l} \left(f(\mathbf{x}_i) - y_i\right)^2 + \lambda \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2$$

...if we just relax the integer constraints to be real ...

$$\min_{\mathbf{f}\in\mathbb{R}^{n_l+n_u}} \infty \sum_{i=1}^{n_l} \left(f(\mathbf{x}_i) - y_i\right)^2 + \lambda \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2$$

...or equivalently (note that $f(\mathbf{x}_i) = f_i$) ...

$$\min_{\mathbf{f}\in\mathbb{R}^{n_l+n_u}} \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2$$

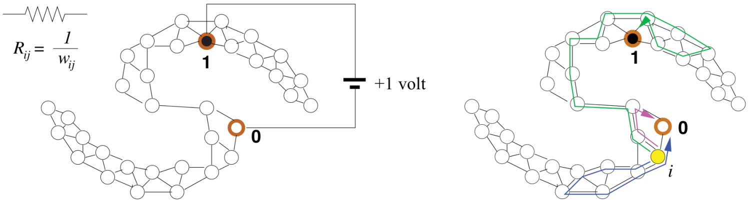
$$s.t. \quad y_i = f(\mathbf{x}_i) \quad \forall i = 1, \dots, n_l$$

SSL with Graphs: Harmonic Functions

Properties of the relaxation from ± 1 to $\mathbb{R}$

- there is a closed form solution for $\mathbf{f}$
- this solution is unique
- globally optimal
- it is either constant or has a maximum/minimum on a boundary
- $f(\mathbf{x}_i)$ may not be discrete
  - but we can threshold it
- electric-network interpretation
- random-walk interpretation

SSL with Graphs: Harmonic Functions



(a) The electric network interpretation (b) The random walk interpretation

Random walk interpretation:

1) start from the vertex you want to label and randomly walk

2) $P(j|i) = \frac{w_{ij}}{\sum_k w_{ik}} \qquad \equiv \qquad \mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$

3) finish when **a** labeled vertex is hit

absorbing random walk

f_i = probability of reaching a positive labeled vertex

SSL with Graphs: Harmonic Functions

How to compute HS? **Option A:** iteration/propagation

Step 1: Set $f(\mathbf{x}_i) = y_i$ for $i = 1, \ldots, n_l$
Step 2: Propagate iteratively (only for unlabeled)

$$f(\mathbf{x}_i) \leftarrow \frac{\sum_{i\sim j} f(\mathbf{x}_j) w_{ij}}{\sum_{i\sim j} w_{ij}} \qquad \forall i \in \{n_l + 1, \ldots, n_u + n_l\}$$

Properties:

- this will converge to the harmonic solution
- we can set the initial values for unlabeled nodes arbitrarily
- an interesting option for large-scale data

SSL with Graphs: Harmonic Functions

How to compute HS? **Option B:** Closed form solution

Define $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_{n_l+n_u})) = (f_1, \ldots, f_{n_l+n_u})$

$$\Omega(\mathbf{f}) = \sum_{i,j=1}^{n_l+n_u} w_{ij} \left(f(\mathbf{x}_i) - f(\mathbf{x}_j)\right)^2 = \mathbf{f}^\mathsf{T}\mathbf{L}\mathbf{f}$$

$\mathbf{L}$ is a $(n_l + n_u) \times (n_l + n_u)$ matrix:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_{ll} & \mathbf{L}_{lu} \\ \mathbf{L}_{u1} & \mathbf{L}_{uu} \end{bmatrix}$$

How to compute this **constrained** minimization problem?

SSL with Graphs: Harmonic Functions

Let us compute **harmonic** solution using **harmonic** property!

How did we formalize the harmonic property of a circuit?

$$(\mathbf{L}\mathbf{f})_u = \mathbf{0}_u$$

In matrix notation

$$\begin{bmatrix} \mathbf{L}_{ll} & \mathbf{L}_{lu} \\ \mathbf{L}_{ul} & \mathbf{L}_{uu} \end{bmatrix} \begin{bmatrix} \mathbf{f}_l \\ \mathbf{f}_u \end{bmatrix} = \begin{bmatrix} \dots \\ \mathbf{0}_u \end{bmatrix}$$

$\mathbf{f}_l$ is constrained to be $\mathbf{y}_l$ and for $\mathbf{f}_u$ ……

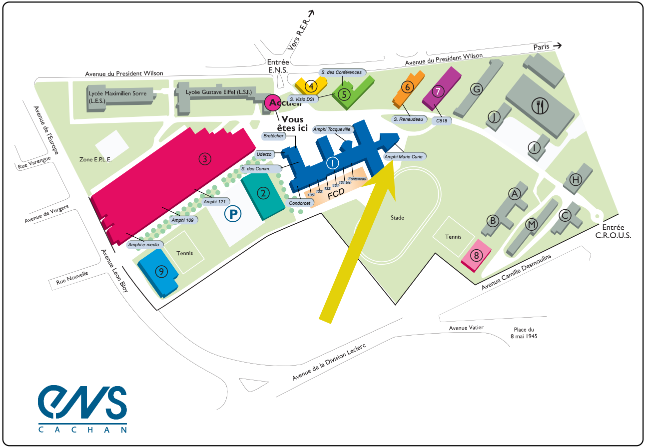
$$\mathbf{L}_{ul}\mathbf{f}_l + \mathbf{L}_{uu}\mathbf{f}_u = \mathbf{0}_u$$

…from which we get

$$\mathbf{f}_u = \mathbf{L}_{uu}^{-1}(-\mathbf{L}_{ul}\mathbf{f}_l) = \mathbf{L}_{uu}^{-1}(\mathbf{W}_{ul}\mathbf{f}_l).$$

Note that this does not depend on $\mathbf{L}_{ll}$.

Next class: Tuesday, October 29th at 13:30!



Michal Valko
contact via Piazza