THE UNIVERSITY OF TEXAS AT AUSTIN
Department of Statistics and Data Sciences
College of Natural Sciences

SDS 385: Stat Models for Big Data

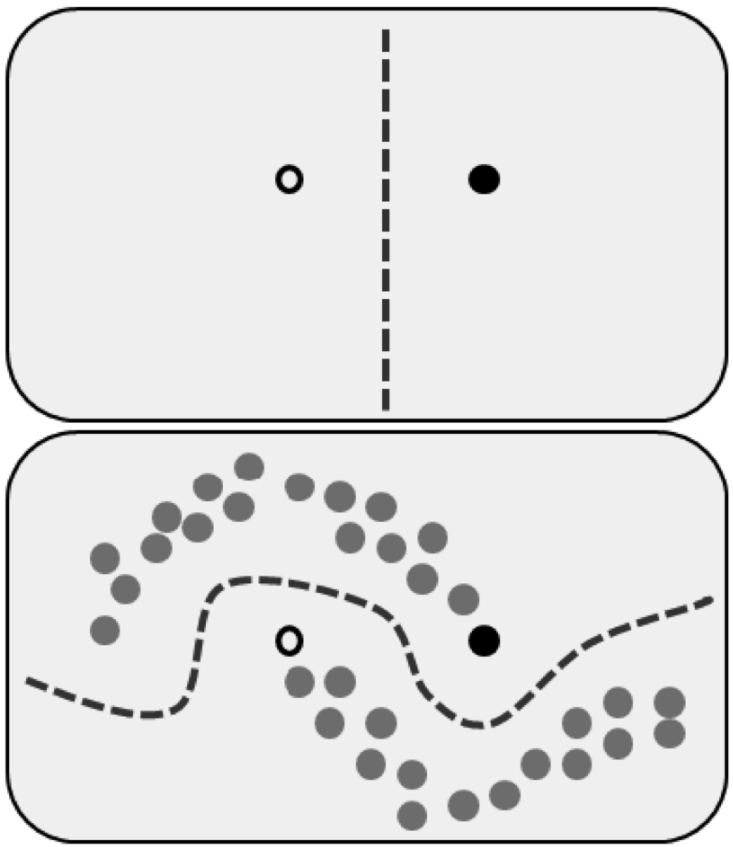
Lecture 10a: Semisupervised learning

Purnamrita Sarkar
Department of Statistics and Data Science
The University of Texas at Austin

https://psarkar.github.io/teaching

Semi-supervised learning

- You are given a lot of unlabeled data.
- Only a few points are labeled.
- Is this useful?



Semi-supervised learning

- Two broad ways
  - Label propagation:
    - Graph Based algorithm
    - Does not generalize to unseen data, i.e. Transductive
  - Manifold regularization
    - Graph Based regularization
    - Does generalize to unseen data, i.e. Inductive

Semi-supervised learning

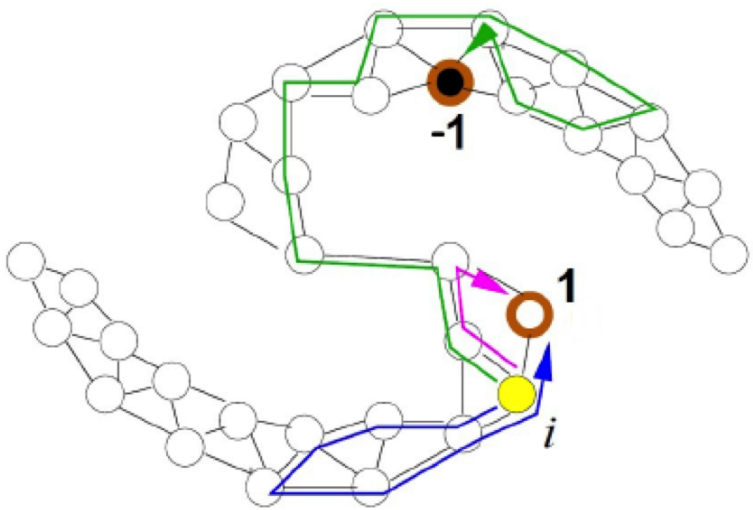
- Input n data points $x_1, \ldots, x_n$
- Define similarity matrix $S \in \mathbb{R}^{n \times n}$

$$S_{ij} = \exp(-\|x_i - x_j\|^2 / 2\sigma^2)$$

- Since S is dense, often k nearest neighbor graphs are also used. (Your homework!)

Label propagation [Zhu et al, 2003]

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$



Label propagation algorithm

- Compute $P \in [0,1]^{(\ell+u)\times(\ell+u)}$
- $P_{ij} = \dfrac{S_{ij}}{\sum_j S_{ij}}$
- Harmonic function:
  - Function value at an unlabeled node is an average of function values at its neighbors
  - For $j = \ell + 1 : \ell + u$,

$$f(j) = \frac{\sum_i S_{ji} f(i)}{\sum_i S_{ji}} = \underbrace{\sum_i P_{ji} f(i)}_{\text{convex combination of values of neighbors}}$$

  - In other words, for the unlabeled nodes, this fixed point equation is satisfied

$$f[U] = Pf[U] \qquad f[L] = y[L]$$

  - For a vector v and set S, we denote by $v[S]$ the subset of values in S
  - U and L denote the set of unlabeled and labeled points respectively.

Closed form

- Assume there are just two classes. Set $y[L] \in \{0,1\}^{\ell}$ accordingly.
- We have:

$$\begin{bmatrix} P_{LL} & P_{LU} \\ P_{UL} & P_{UU} \end{bmatrix} \begin{bmatrix} Y_L \\ Y_U \end{bmatrix} = \begin{bmatrix} Y_L \\ Y_U \end{bmatrix}$$

- Expanding, we get:

$$P_{UL} Y_L + P_{UU} Y_U = Y_U$$

- Moving things around:

$$Y_U = (I - P_{UU})^{-1} P_{UL} Y_L$$

- Can use a linear system solver

Label propagation - Random walk interpretation

- Think of the labeled nodes as absorbing states
- Use

$$Y_U = \sum_{t=0}^{\infty} P_{UU}^t P_{UL} Y_L$$

$$= \underbrace{P_{UL} Y_L}_{\text{Probability of reaching "1"s in one step}} + \underbrace{P_{UU} P_{UL} Y_L}_{\text{Probability of reaching in two steps}} +$$

$$= \text{Probability of reaching a label "1" in a long random walk}$$

- Why is this useful?
  - If the labels are all reachable, a long walk must hit a "0" or a "1"
  - So if $Y_i > 1/2$, that means from i, its more likely to reach a "1" than a "0"

Graph Laplacian interpretation

- Graph Laplacian: $L = D - S$, where $D_{ii} = \sum_j S_{ij}$ is a diagonal matrix
- L is positive semi-definite (we are assuming $S_{ij} > 0$)
- Why?
  - For any vector v,
$$v^T L v = \sum_{ij} S_{ij}(v_i - v_j)^2$$
  - So this measures how unsmooth v is w.r.t S
  - L is also singular, why?

Label propagation algorithm

- We can also frame label propagation as

$$\arg\min_{v} v^T L v \qquad s.t. v[L] = y_L$$

- Why?

Label propagation algorithm

- We can also frame label propagation as

$$\arg\min_{v} v^T L v \qquad s.t. v[L] = y_L$$

- Why?
- Write $v = \begin{bmatrix} y_L & v_U \end{bmatrix}$

Label propagation algorithm

- We can also frame label propagation as

$$\arg\min_{v} v^T L v \qquad s.t. v[L] = y_L$$

- Why?
- Write $v = \begin{bmatrix} y_L & v_U \end{bmatrix}$
- Now set the derivative to zero.
  - $Lv = 0$ such that $v[L] = y_L$
  -

$$\begin{bmatrix} D_{LL} - S_{LL} & -S_{LU} \\ -S_{UL} & D_{UU} - S_{UU} \end{bmatrix} \begin{bmatrix} y_L \\ v_U \end{bmatrix} = 0$$

  - Solving:

$$-S_{UL} y_L + (D_{UU} - S_{UU}) v_U = 0$$

Label propagation algorithm

- We can also frame label propagation as

$$\arg\min_{v} v^T L v \qquad s.t. v[L] = y_L$$

- Why?
- Write $v = \begin{bmatrix} y_L & v_U \end{bmatrix}$
- Now set the derivative to zero.
  - $Lv = 0$ such that $v[L] = y_L$
  -

$$\begin{bmatrix} D_{LL} - S_{LL} & -S_{LU} \\ -S_{UL} & D_{UU} - S_{UU} \end{bmatrix} \begin{bmatrix} y_L \\ v_U \end{bmatrix} = 0$$

  - Solving:

$$-S_{UL} y_L + (D_{UU} - S_{UU}) v_U = 0$$

  - rearranging: $v_U = (D_{UU} - S_{UU})^{-1} S_{UL} y_L = (I - P_{UU})^{-1} P_{UL} y_L$

Experimentally?

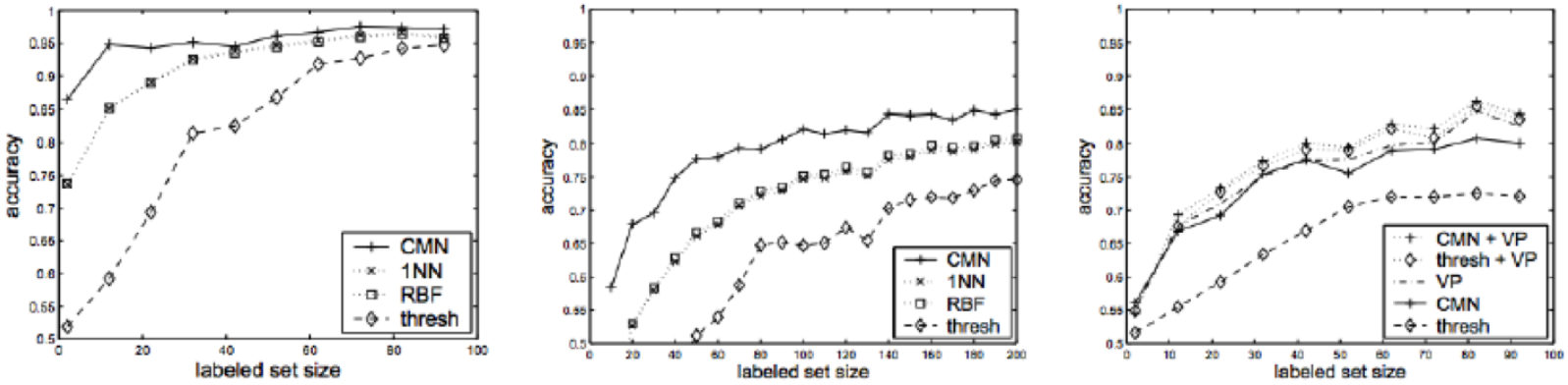


Figure 3. Harmonic energy minimization on digits "1" vs. "2" (left) and on all 10 digits (middle) and combining voted-perceptron with harmonic energy minimization on odd vs. even digits (right)

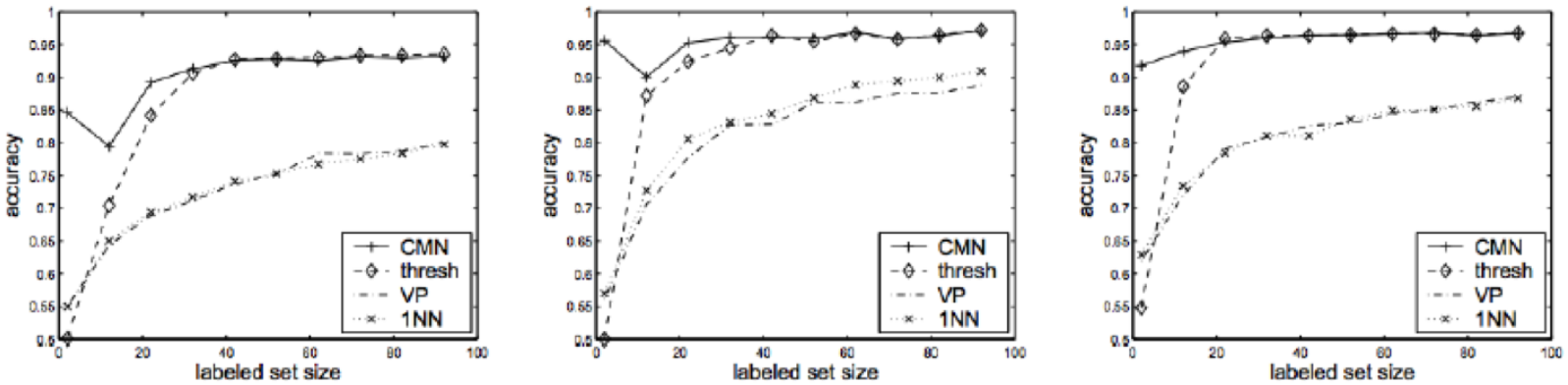


Figure 4. Harmonic energy minimization on PC vs. MAC (left), baseball vs. hockey (middle), and MS-Windows vs. MAC (right)

Manifold regularization

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$

Manifold regularization

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$

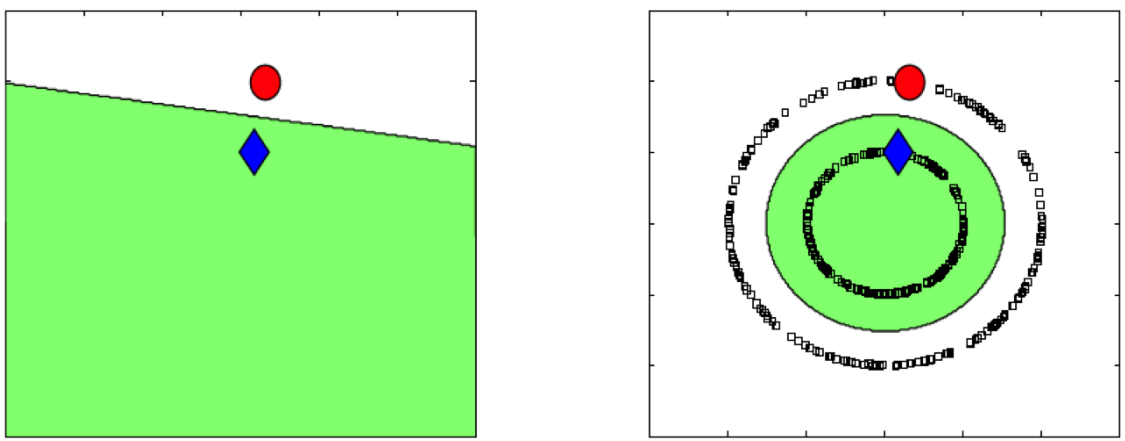


Figure 1: Unlabeled data and prior beliefs

Manifold regularization

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$

Manifold regularization

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$
- Before the constraint was $v[L] = y_L$, now instead we will use a loss function and learn a classifier on the labeled data

$$\min_w \underbrace{\sum_{i=1}^{\ell} \text{loss}(y_i, w^T x_i)}_{\textit{loss}} + \lambda \underbrace{R(w)}_{\textit{regularization}}$$

Manifold regularization

- Input:
  - ℓ labeled datapoints $(x_1, y_1), \ldots, (x_\ell, y_\ell)$
  - u unlabeled datapoints $x_{\ell+1}, \ldots, x_{\ell+u}$
- Predict: labels $y_{\ell+1}, \ldots, y_{\ell+u}$
- Before the constraint was $v[L] = y_L$, now instead we will use a loss function and learn a classifier on the labeled data

$$\min_w \underbrace{\sum_{i=1}^{\ell} \text{loss}(y_i, w^T x_i)}_{\textit{loss}} + \lambda \underbrace{R(w)}_{\textit{regularization}}$$

- How about the unlabeled data?

Manifold regularization: Belkin et al 2006

$$\min_w \underbrace{\sum_{i=1}^{\ell} \text{loss}(y_i, w^T x_i)}_{loss} + \lambda \underbrace{R(w)}_{regularization} + \beta (Xw)^T L(Xw)$$

- Assume a linear predictor $w^T x$
- Idea: close/similar points have similar predicted labels.
- LapSVM:

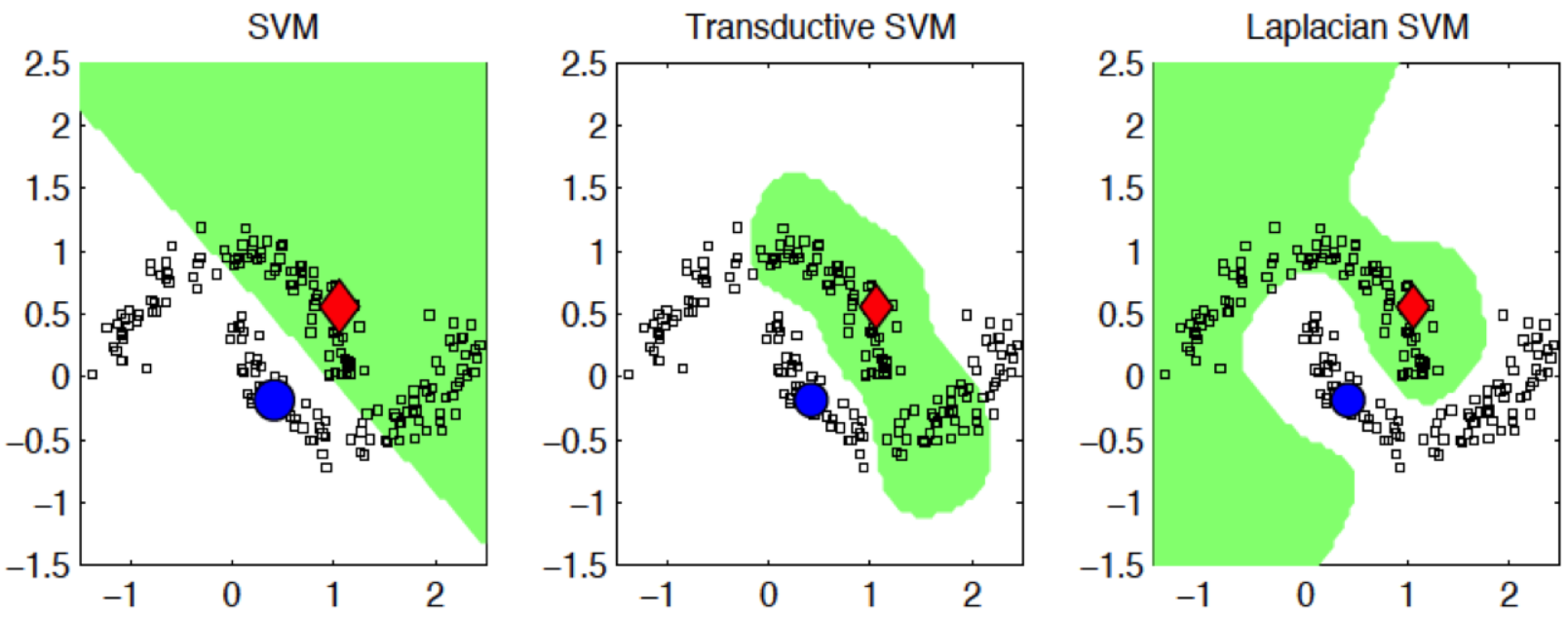
$$\min_w \sum_{i=1}^{\ell} (1 - y_i f(x_i))_+ + \lambda \underbrace{\|f\|_K^2}_{regularization} + \beta f^T L f$$

Transductive SVM: Joachims et al 1999

$$\min_{w, y_{\ell+1}, \ldots, y_{\ell+u}} \sum_{i=1}^{\ell} (1 - y_i f(x_i))_+ + C' \sum_{i=\ell+1}^{n} (1 - y_i f(x_i))_+ + \lambda \underbrace{\|f\|_K^2}_{\textit{regularization}}$$

- Iteratively solves SVM quadratic programs
- Switches labels to improve objective function
- Suffers from local optima, inherently combinatorial problem

Transductive SVM VS LapSVM



Acknowledgments

- Cho-Jui Hsieh's lecture notes from UC Davis
- Zhu et al's paper in ICML "Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions"
- Ng et al's paper on ranking Stability "Link Analysis, Eigenvectors and Stability", IJCAI 2001
- Belkin, Niyogi and Sindhwani's paper "Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples" in JMLR 2006
- My old talk on Random walks.