INFORMATICS SEMINAR
SEPT. 27 & OCT. 4, 2010

Introduction to Semi-Supervised Learning Review

L.D. Miller

Overview—Citation

- X. Zhu and A.B. Goldberg, *Introduction to Semi-Supervised Learning*, Morgan & Claypool Publishers, 2009
- Why use this book?
  - Provides an excellent introduction to semi-supervised learning
    - Easy to understand examples
    - Numerous references
  - Recently published & free to download!

Overview—Schedule

- Machine Learning (ML) Introduction (Sept. 27)
- Semi-Supervised Learning (SSL) (Sept. 27)
  - Self-Training
- Mixture Models (Sept. 27)
  - Cluster-then-Label
- Co-Training (Sept. 27)
- Graph-Based SSL (Oct. 4)
- Semi-Supervised Support Vector Machines (Oct. 4)
- Software Implementations (Oct. 4)

ML Introduction

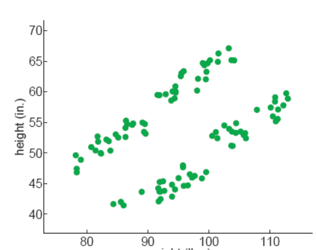
- Dataset Definition
- Unsupervised Learning
- Supervised Learning

ML Intro—Dataset

- Dataset consists of set of instances
- An instance (i.e., data point) consists of *D*-dimensional feature vector (x)
- Features (i.e., attributes) can be numeric or discrete values
- An instance may have a desired prediction or label (y)
- **<u>Assumption</u>**: instances in training sample are sampled independently from underlying distribution

ML Intro—Dataset

- Example Dataset "Little Green Men"



ML Intro—Unsupervised Learning

7

- Uses training sample of instances **<u>without</u>** labels
- Common Tasks:
  - Novelty Detection
  - Dimensionality reduction
  - Clustering (book focus)
    - Partitions data points into clusters where instances in the same cluster are more “similar” than instances in different clusters
    - Number of clusters either pre-specified or inferred from data

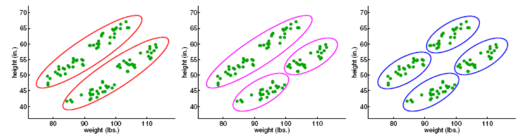
ML Intro—Unsupervised Learning

8

- Hierarchical Agglomerative Clustering

Input: a training sample $\{\mathbf{x}_i\}_{i=1}^{n}$; a distance function $d()$.
1. Initially, place each instance in its own cluster (called a singleton cluster).
2. **while** (number of clusters > 1) **do:**
3. Find the closest cluster pair A, B, i.e., they minimize $d(A, B)$.
4. Merge A, B to form a new cluster.

Output: a binary tree showing how clusters are gradually merged from singletons to a root cluster, which contains the whole training sample.



ML Intro—Supervised Learning

9

- Uses training sample of instances **<u>with</u>** labels
- Common Tasks:
  - Regression
  - Classification (book focus)
    - Train a function (i.e., classifier) to predict the correct label for unknown data points from the same joint probability distribution as the training sample
    - Function divides feature space into decision regions where instances share the same label

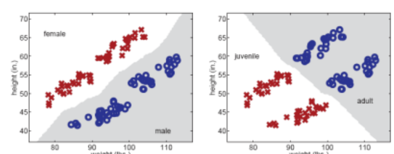
ML Intro—Supervised Learning

10

- K-Nearest-Neighbor Classifier

Input: Training data $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$; distance function $d()$; number of neighbors k; test instance $\mathbf{x}^*$

1. Find the k training instances $\mathbf{x}_{i_1}, \ldots, \mathbf{x}_{i_k}$ closest to $\mathbf{x}^*$ under distance $d()$.
2. Output y^* as the majority class of $y_{i_1}, \ldots, y_{i_k}$. Break ties randomly.



(a) classification by gender (b) classification by age

SSL

11

- Introduction
- Inductive vs. Transductive
- Self-Training

SSL—Intro

12

- Uses training sample of instances **<u>with</u>** and **<u>without</u>** labels
- Common Tasks:
  - Constrained Clustering
    - Improve clustering using label information
    - Example: use must-link and cannot-link constraints
  - Semi-Supervised Classification (book focus)
    - Improve classification using unlabeled instances
    - Example: self-training discussed later “bootstraps” the training sample by labeling the unlabeled instances

SSL—Classification

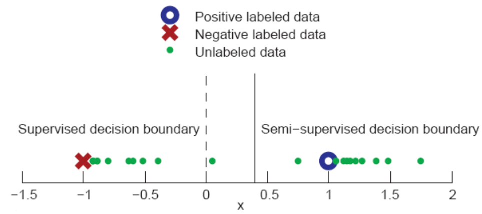
13

- Motivation:
  - Understand learning in humans/machines
  - Build better ML algorithms (book focus)
    - Supervised learning requires labeled instances
    - Labels are difficult to obtain because they require human annotators, special devices, expensive experiments, etc.
    - Unlabeled instances are available in large quantity and easy to collect
    - Leverage unlabeled instances to improve the performance for supervised learning
- **Assumption**: Instances with the same label "form coherent groups" (i.e., smoothness)

SSL—Classification

14

- SSL Example:



SSL—Inductive vs. Transductive

15

- Two different SSL settings:
  - Inductive
    - Learns a function to predict labels for unknown instances using labeled/unlabeled training sample
    - Similar to supervised learning
  - Transductive
    - Learns a function to predict the labels for the unlabeled instances in the training sample

Algorithm*	I	T
Self-training		✓
Mixture Models	✓	✓
Co-training		✓
Graph Based		✓
S3VM	✓	✓

*Emphasized in this book

SSL—Self-Training

16

- A self-training algorithm uses its own predictions "to teach itself"
  - Step 1: train a function using only the labeled instances.
  - Step 2: use the function to label some of the unlabeled instances
  - Step 3: retrain the function on the expanded, labeled instances
- **Assumption**: Own predictions tend to be correct

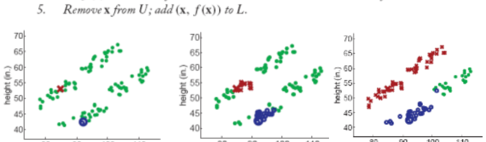
SSL—Self-Training

17

- Propagating 1-Nearest Neighbor

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, *unlabeled data* $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$, *distance function* $d()$.

1. *Initially, let* $L = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ *and* $U = \{\mathbf{x}_j\}_{j=l+1}^{l+u}$.
2. *Repeat until* U *is empty:*
3. *Select* $\mathbf{x} = \mathrm{argmin}_{\mathbf{x}\in U} \min_{\mathbf{x}'\in L} d(\mathbf{x}, \mathbf{x}')$.
4. *Set* $f(\mathbf{x})$ *to the label of* $\mathbf{x}$*'s nearest instance in* L. *Break ties randomly.*
5. *Remove* $\mathbf{x}$ *from* U; *add* $(\mathbf{x}, f(\mathbf{x}))$ *to* L.



(a) Iteration 1 (b) Iteration 25 (c) Iteration 74

Mixture Models

18

- Gaussian Mixture Models (GMM)
- Cluster-then-Label

Mixture Models—GMM

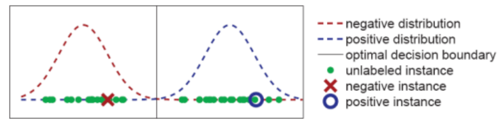
19

- Motivation:
  - Unlabeled data points contain a mixed distribution from <u>all</u> the labels
  - If we could decompose this mixed distribution into separate distributions <u>for each</u> label then we could predict labels for unlabeled data points using these distributions
    - Similar to unsupervised clustering!
- **<u>Assumption:</u>** Data comes from a mixture model with Gaussian distributions for the labels

Mixture Models—GMM

20

- Separate Distributions for the Labels



Mixture Models—GMM

21

- One commonly used criterion for solving mixture models is maximum likelihood estimate (MLE).

$$\log p(\mathcal{D}|\theta) = \log \prod_{i=1}^{l} p(\mathbf{x}_i, y_i|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta) p(\mathbf{x}_i|y_i, \theta),$$

- MLE gives the estimated set of parameters for each distribution (mean and covariance matrix)
- Does not use unlabeled training data
- For SSL use MLE with marginal probability for generating the unlabeled instances

$$\log p(\mathcal{D}|\theta) = \sum_{i=1}^{l} \log p(y_i|\theta) p(\mathbf{x}_i|y_i, \theta) + \sum_{i=l+1}^{l+u} \log p(\mathbf{x}_i|\theta).$$

Mixture Models—GMM

22

- Cannot solve new MLE analytically because labels are unknown so use Expectation Maximization (EM) to find parameters that (locally) maximize the probability distributions
  - In E we assign soft labels to unlabeled data using current parameters
  - In M we compute new parameters using MLE on labeled data and soft assignments

Input: observed data $\mathcal{D}$, hidden data $\mathcal{H}$, initial parameter $\theta^{(0)}$
1. Initialize $t = 0$.
2. Repeat the following steps until $p(\mathcal{D}|\theta^{(t)})$ converges:
3. E-step: compute $q^{(t)}(\mathcal{H}) \equiv p(\mathcal{H}|\mathcal{D}, \theta^{(t)})$
4. M-step: find $\theta^{(t+1)}$ that maximizes $\sum_{\mathcal{H}} q^{(t)}(\mathcal{H}) \log p(\mathcal{D}, \mathcal{H}|\theta^{(t+1)})$
5. $t = t + 1$
Output: $\theta^{(t)}$

Mixture Models—CTL

23

- Clusters found by unsupervised clustering are similar to the distributions found by GMM
- The Cluster-then-Label algorithm uses such clusters for semi-supervised classification
- The addition of EM style approach to CTL (GACS) compensates for sensitivity in the clustering algorithms

Co-Training

24

- Motivation:
  - An instance can contain two distinct feature sets or "views"
    - Name and context (from named entity classification)
    - Words in webpage and links to webpage
    - Etc.
  - If we train a separate classifier on each view they could teach each other!

instance	$\mathbf{x}^{(1)}$	$\mathbf{x}^{(2)}$	y
1.	Washington State	headquartered in	Location
2.	Mr. Washington	vice president	Person
3.	Kazakhstan	headquartered in	?
4.	Kazakhstan	flew to	?
5.	Mr. Smith	partner at	?

Co-Training

25

- Co-Training Algorithm

Input: labeled data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$, a learning speed k.
Each instance has two views $\mathbf{x}_i = [\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}]$.
1. Initially let the training sample be $L_1 = L_2 = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_l, y_l)\}$.
2. Repeat until unlabeled data is used up:
3. Train a view-1 classifier $f^{(1)}$ from L_1, and a view-2 classifier $f^{(2)}$ from L_2.
4. Classify the remaining unlabeled data with $f^{(1)}$ and $f^{(2)}$ separately.
5. Add $f^{(1)}$'s top k most-confident predictions $(\mathbf{x}, f^{(1)}(\mathbf{x}))$ to L_2.
Add $f^{(2)}$'s top k most-confident predictions $(\mathbf{x}, f^{(2)}(\mathbf{x}))$ to L_1.
Remove these from the unlabeled data.

- **Assumption:** Views are conditionally independent given the class label
- Assumption often violated but results are generally good even with feature splits on single "view" dataset (Ling et al., 2009)

Graph-Based SSL

26

- Introduction
- Edge Weight Heuristics (EWH)
- SSL Algorithms
- Weakness

Graph-Based SSL—Intro

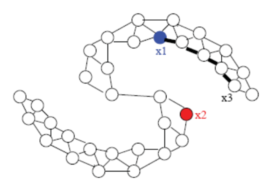
27

- Motivation:
  - Model the relationship between instances by constructing a graph from all the training data
    - Vertices are instances
    - Edges are similarity between instances
  - Propagate labels from the labeled vertices through the edges to nearby unlabeled vertices
- **Assumption:** Labels are "smooth" with respect to graph such that two instances connected by the strong edge should have same label

Graph-Based SSL—Intro

28

- Label Propagation Example



Graph-Based SSL—EWH

29

- Fully connected
  - For each x_i, x_j, create edge with weight that decreases as Euclidean distance increases
  - One popular variant is Radial Basis Function because weight is normalized between 0 and 1
    - Bandwidth (σ) controls how quickly weight decreases

$$w_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

k Nearest Neighbor (kNN)
  - For each x_i, find k most similar instances using Euclidean distance
  - Create edge for x_i, x_j iff x_j is in kNN (not symmetric!)
  - Automatically adapts to density of feature space

Graph-Based SSL—EWH

30

- εNN
  - For each xi, xj, create edge iff distance $\|\mathbf{x}_i - \mathbf{x}_j\| \leq \epsilon$
  - Easier to construct than kNN graphs
- Which should I use?
  - No definitive answer
  - "Best" graph requires knowledge of the problem domain
  - RBF and kNN seem the most popular

Graph-Based SSL—Algorithms

31

- Mincut
  - Treat positive labeled instances (i.e., vertices) as fluid "source" and negative as "sink"
  - Find minimum set of edges (i.e., cut) whose removal blocks flow from sources to sink
  - Solve integer programming problem or use Edmond-Karp

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

Graph-Based SSL—Algorithms

32

- Harmonic Function
  - Similar to Mincut except *f* can produce real values
  - Interesting Interpretations:
    - Electrical network where edges are resistors
    - Random walk on a graph
  - Iterative procedure to solve where we update unlabeled vertices with weight average of neighbors (see book for proof of convergence)

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

Graph-Based SSL—Algorithms

33

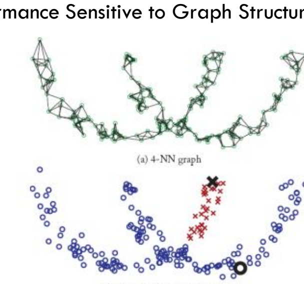
- Performance Sensitive
  - Treat positive labeled instances (i.e., vertices) as fluid "source" and negative as "sink"
  - Find minimum set of edges (i.e., cut) whose removal blocks flow from sources to sink
  - Solve integer programming problem or use Edmond-Karp

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

Graph-Based SSL—Weakness

34

- Performance Sensitive to Graph Structure!



(a) 4-NN graph

(b) Harmonic function predictions

S3VM

35

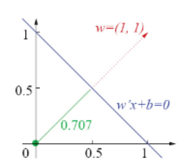
- Support Vector Machines (SVM)
- Semi-Supervised Support Vector Machines (S3VM)

Source: http://www.payroll-bureau-int.com/

S3VM—SVM

36

- Linear **decision boundary** in 2-space



- Decision boundary cuts feature space into two halves
  - Labels depend on which side instance is on
  - Measure distance between instance and boundary to find the **margin**

S3VM—SVM

37

- Training sample is **linearly separable** when decision boundary separates instances with different labels
  - Solve using quadratic programming
- What happens when training sample is not linearly separable?
  - Relax constraints with slack variables (this book) and solve using **hinge loss**
  - Remap into higher dimensional space using kernel trick (Cristianini & Shawe-Taylor, 2000)
- Motivation:
  - Find decision boundary that maximizes the margin for labeled training sample

S3VM—S3VM

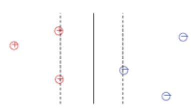
38

- Also called Transductive Support Vector Machines
- Uses a **hat loss** function to tentatively label the unlabeled instances
  - Does not require real label
  - Similar to unsupervised clustering
- Motivation:
  - Find decision boundary that maximizes the margin for entire training sample

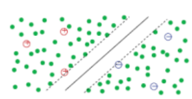
S3VM—S3VM

39

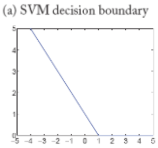
- Difference between SVM and S3VM



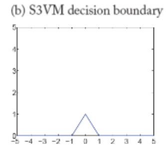
(a) SVM decision boundary



(b) S3VM decision boundary



(a) the hinge loss

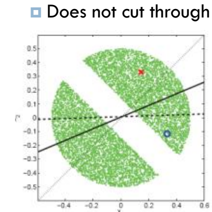


(b) the hat loss

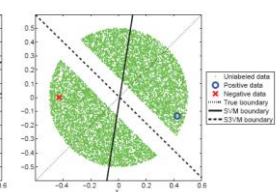
S3VM—S3VM

40

- Assumption: Decision boundary falls in a low density region of the feature space
  - Does not cut through dense labeled data



(a) S3VM in local minimum



(b) S3VM in “wrong” low density region

Software Implementations

41

Conclusions

42

- SSL algorithms discussed use instances **with** and **without** labels to train classifier
- All five categories rely on strong assumptions
  - **Self-Training:** Own predictions tend to be correct
  - **Gaussian Mixture Models:** Data comes from a mixture model with Gaussian distributions for the labels
  - **Co-Training:** Views are conditionally independent given the class label
  - **Graph-Based:** Labels are “smooth” with respect to graph
  - **S3VM:** Decision boundary falls in a low density region of the feature space
- When assumptions are violated accuracy is reduced!

For More Information…

43

- Machine Learning Textbook
  - T.M. Mitchell, *Machine Learning*, McGraw-Hill Science/Engineering/Math, 1997
- Department Faculty



Questions?

References

- N. Cristianini and J. Shaw-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods,* Cambridge University Press, 2000
- C. Ling, J. Du, and Z. Zhou, "When does Co-training Work in Real Data?," *Advances in Knowledge Discovery and Data Mining*, 2009, pp. 596-603
- T.M. Mitchell, Machine Learning, McGraw-Hill Science/Engineering/Math, 1997
- X. Zhu and A.B. Goldberg, *Introduction to Semi-Supervised Learning*, Morgan & Claypool Publishers, 2009