Ch.5 Learning without supervision

1. Unsupervised learning

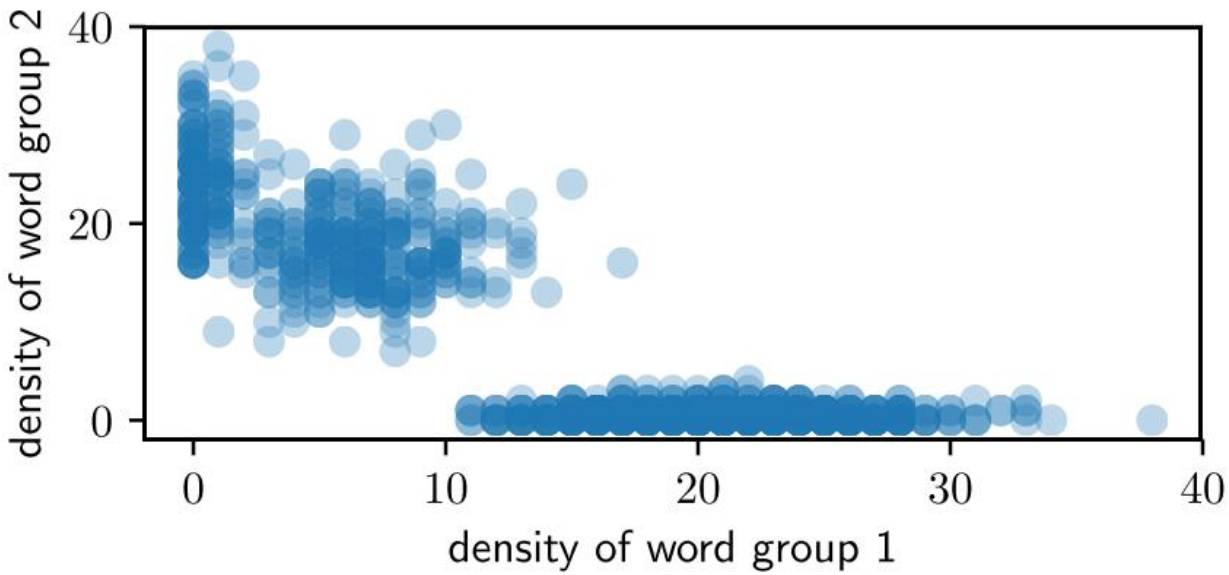
왜 unsupervised learning이 필요한가?

- WSD(Word sense disambiguation)의 사례

- 개별 의미에 대한 충분한 labeled data를 다 모으는 것이 불가능함

- `bank#1`: a financial institution
- `bank#2`: the land bordering a river

어떻게 가능한가?



- WSD는 주로 feature vector를 이용해 수행됨
- 동음이의어를 포함하는 document를 특정 단어 집합의 출현 빈도로 산점도 표현하면 그룹이 만들어짐
- 뚜렷하게 군집이 만들어지는 모습을 통해 비지도 의미 구별의 가능성 확인

1. *financial, deposits, credit, lending, capital, markets, regulated, reserve, liquid, assets*
2. *land, water, geography, stream, river, flow, deposits, discharge, channel, ecology*

대표적인 클러스터링 알고리즘

- **(거리) K-Means Clustering**
- **(거리) Gaussian Mixture Models (GMM)을 사용한 Expectation-Maximization (EM)**
- (밀도) Mean-Shift Clustering
- (밀도) Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
- (거리) Agglomerative Hierarchical Clustering

https://www.nextobe.com/single-post/2018/02/26/데이터-과학자가-알아야-할-5가지-클러스터링-알고리즘

1) K-means

- 가장 구현하기 쉽고 단순한 클러스터링 알고리즘. 최초 중심점은 무작위로 초기화. point가 가장 가까운 centroid에 배정되어 하나의 군집(cluster)을 형성
- 속도가 빠르다는 장점, but 사전에 이용자가 centroid 개수인 k를 결정해야 함

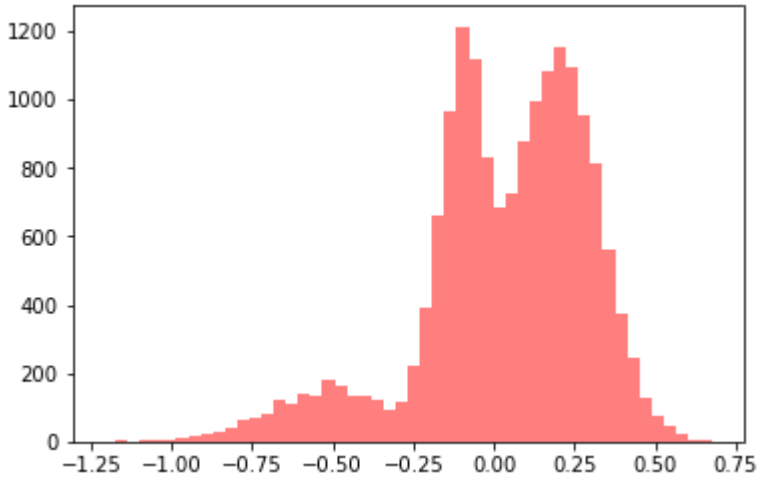
1. each instance is placed in the cluster with the closest center;
2. each center is recomputed as the average over points in the cluster.

- b와 w 중 하나를 고정시킨 상태에서 번갈아 update를 진행(jointly optimization의 차선책) → local optima에 빠지는 문제 발생

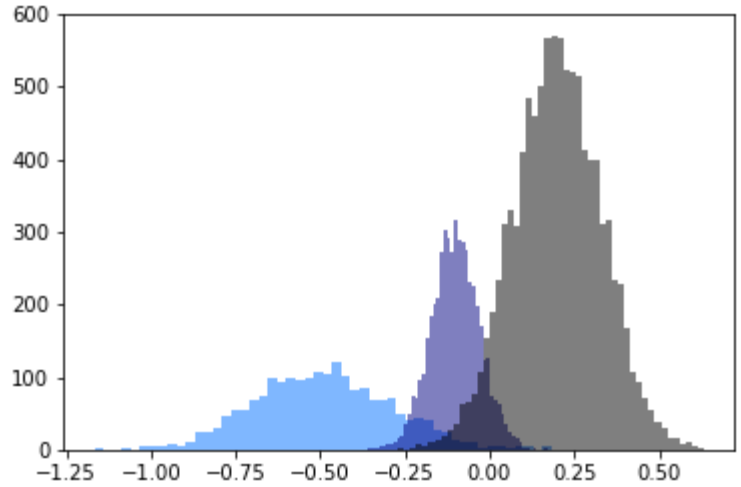
$$\min_{b,w} \sum_{i}^{n} \sum_{j}^{k} w_{ij} \|x_i - b_j\|_2^2 \text{ s.t. } \sum_{j} w_{ij} = 1, \forall j$$

http://sanghyukchun.github.io/69/

2-1) Gaussian Mixture Model (GMM)



실제 보는 데이터



우리가 상상할 수 있는 하위 분포들

- Mixture Model: 전체 분포에서 하위 분포가 존재한다고 보는 모델
- GMM: 데이터가 K개의 random한 하위 정규분포들의 혼합으로 이루어졌다고 가정 (K는 사람이 정해야 함)
- 두 종류의 모수: **1) 정규분포 중 확률적으로 어디에서 속해있는가를 나타내는 Weight 값(잠재변수)**, 2) 각각의 정규분포의 모수(평균, 분산)

2-2) Expectation-Maximization (EM)

- 원래는 연립방정식의 해를 구하는 방법으로 responsibility를 포함한 모수 추정
- 그러나 식의 형태가 responsibility를 알고 있다면 모수를 추정하는 것이 간단하도록 만들어져 있기 때문에 iterative한 방식인 EM(Expectation-Maximization)을 사용하면 더 쉽게 모수 추정 가능
- 모수와 responsibility를 번갈아가며 업데이트하는 방법

  - E step 에서는 우리가 현재까지 알고 있는 모수가 정확하다고 가정하고 이를 사용하여 각 데이터가 어느 카테고리에 속하는지 즉, responsiblity를 추정한다.

$$(\theta_k, \mu_k, \Sigma_k) \quad \Longrightarrow \quad \gamma_{ik}$$

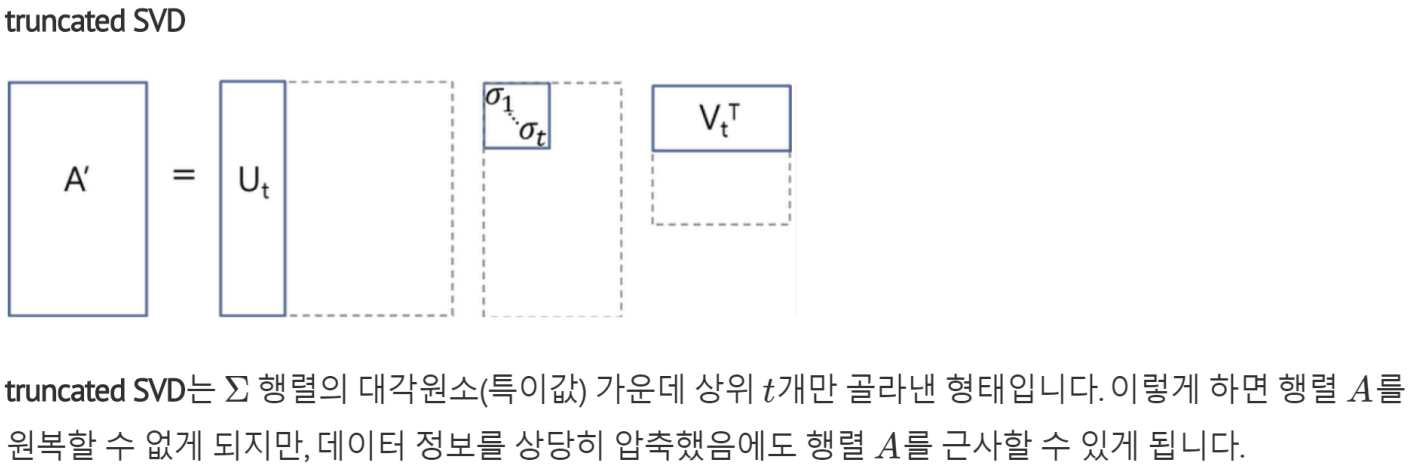
  - M step 에서는 우리가 현재까지 알고 있는 responsibility가 정확하다고 가정하고 이를 사용하여 모수값을 추정한다.

$$\gamma_{ik} \quad \Longrightarrow \quad (\theta_k, \mu_k, \Sigma_k)$$

Application: Word Sense Induction (WSI)

- 단어 의미 추론: 특정 단어가 같은 의미로 사용된 문맥들끼리 묶어서 다른 의미로 사용된 문맥을 구별해내는 작업 (사전적 의미 라벨링까지는 하지 않음)
- 각각의 인스턴스는 모호한 단어들을 지칭. x(i)는 해당 단어 주변에 나타난 단어들의 빈도로 이루어진 벡터 (Schutze (1998)는 50 윈도우 사용)
- Sparse한 많은 매트릭스이므로 특이값 분해(singular value decomposition)를 사용해 압축하면 더 좋은 성능 얻을 수 있음 (truncated SVD)

https://ratsgo.github.io/from%20frequency%20to%20semantics/2017/04/06/pcasvdlsa/



truncated SVD는 Σ 행렬의 대각원소(특이값) 가운데 상위 t개만 골라낸 형태입니다. 이렇게 하면 행렬 A를 원복할 수 없게 되지만, 데이터 정보를 상당히 압축했음에도 행렬 A를 근사할 수 있게 됩니다.

2. semi-supervised learning

Semi-supervised learning 기법들

1. Self-training
2. **Generative models**
3. S3VMs(=TSVMs)
4. **Graph-based algorithms**
5. **Multiview algorithms**

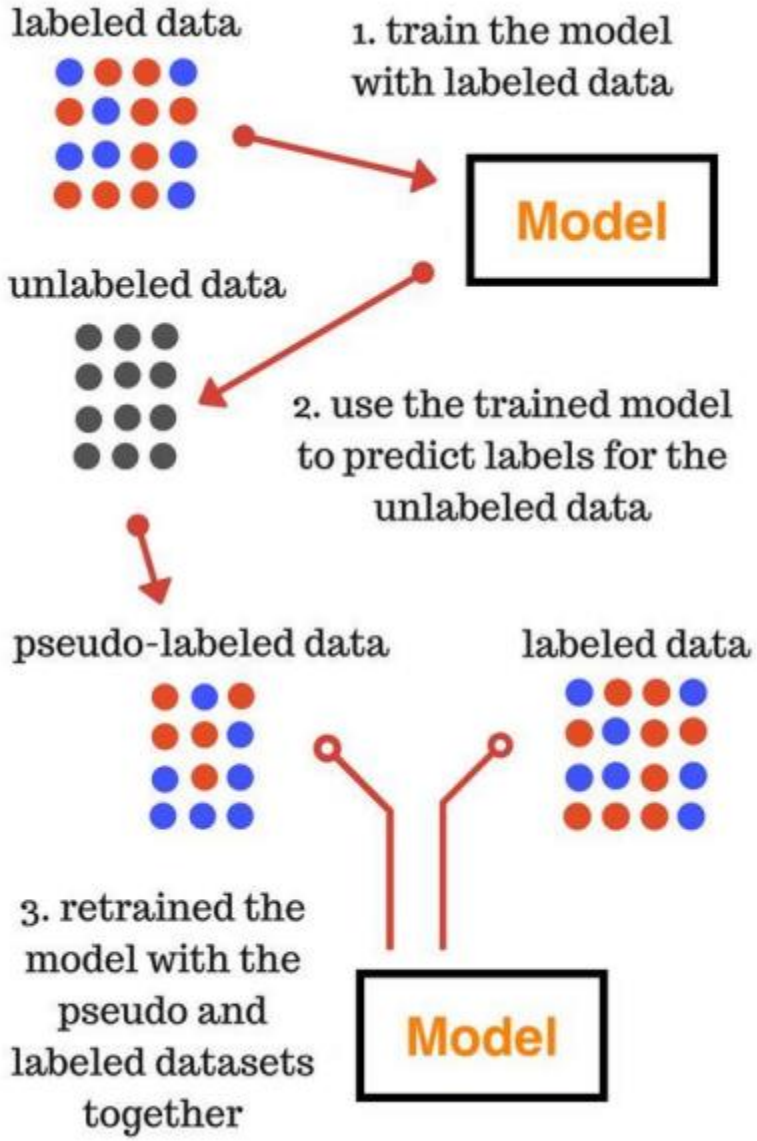
http://pages.cs.wisc.edu/~jerryzhu/pub/sslicml07.pdf

- 학습목적에 따라 Transductive Learning과 Inductive Learning으로 구분
- Transductive Learning: 학습 데이터 중 레이블이 없는 데이터들에 대해 최대한 정확한 레이블이 부여되도록 학습을 수행하는 것이 목적
- Inductive Learning: 주어진 데이터를 최대한 잘 활용하여 레이블들을 분류하는 분류경계선이나 예측모델을 구축하는 학습을 수행

1) self-training

1. 레이블이 달린 데이터로 모델을 학습
2. 이 모델을 이용해 레이블이 달리지 않은 데이터를 예측
3. 이 중에서 가장 확률값이 높은 데이터들만 레이블 데이터로 다시 가져감
4. 위 과정을 계속 반복 (반복할 수록 모델이 정확해짐)

- 장점: 가장 단순함. 어떤 알고리즘에도 적용 가능
- 단점: 잘못된 아웃라이어(노이즈)가 한 개만 포함되어 있어도 잘못된 결과가 나올 수 있음. 완전히 수렴한다고 말할 수 없음



2) generative models

- 우도(likelihood)나 사후 확률(posterior probability)를 사용하여 분류 경계선 (decision boundary)을 생성

- 장점: 제대로 학습된다면 굉장히 효과적, 오랜 역사를 가진 확률방법론을 사용함

- 단점: 모델이 얼마나 잘 만들어졌는지 확인 어려움, 좋지 않은 local optima가 산출될 수 있음

1. Start from MLE $\theta = \{w, \mu, \Sigma\}_{1:2}$ on (X_l, Y_l), repeat:
2. The E-step: compute the expected label $p(y|x,\theta) = \frac{p(x,y|\theta)}{\sum_{y'} p(x,y'|\theta)}$ for all $x \in X_u$
   - label $p(y = 1|x, \theta)$-fraction of x with class 1
   - label $p(y = 2|x, \theta)$-fraction of x with class 2
3. The M-step: update MLE θ with (now labeled) X_u
   - w_c=proportion of class c
   - μ_c=sample mean of class c
   - Σ_c=sample cov of class c

3) Transductive support vector machine(TSVM)

- **전제: 각기 다른 클래스에 속하는 unlabeled 데이터는 큰 경계(margin)으로 구분되어 있을 것**
- SVM의 경계최대화(margin maximization) 방법을 이용. 밀도가 희박한 지역(sparse region)으로 분류선이 지나도록 설계. 클래스 없는 데이터를 이용해 결정 경계의 마진이 커지도록 학습

    1. 먼저 클래스 있는 데이터만을 이용해 SVM을 학습하고 클래스 없는 데이터를 레이블링
    2. 레이블링 데이터 중 클래스를 바꾸어 슬랙 값이 작아질 경우, 두 데이터의 클래스를 교환

    3. 레이블 전파와 달리 레이블링을 수행하면서 학습하는 과정을 반복

- 단점: 계산 시간이 오래 걸림, local optima에 빠질 수 있음

3) Transductive support vector machine(TSVM)

- **전제: 각기 다른 클래스에 속하는 unlabeled 데이터는 큰 경계(margin)으로 구분되어 있을 것**
- SVM의 경계최대화(margin maximization) 방법을 이용. 밀도가 희박한 지역(sparse region)으로 분류선이 지나도록 설계. 클래스 없는 데이터를 이용해 결정 경계의 마진이 커지도록 학습

1. 먼저
2. 레이
3. 레이

- 단점: 계산

Try to keep labeled points outside the margin, while maximizing the margin:

$$\min_{h,b,\xi} \sum_{i=1}^{l} \xi_i + \lambda \|h\|^2_{\mathcal{H}_K}$$

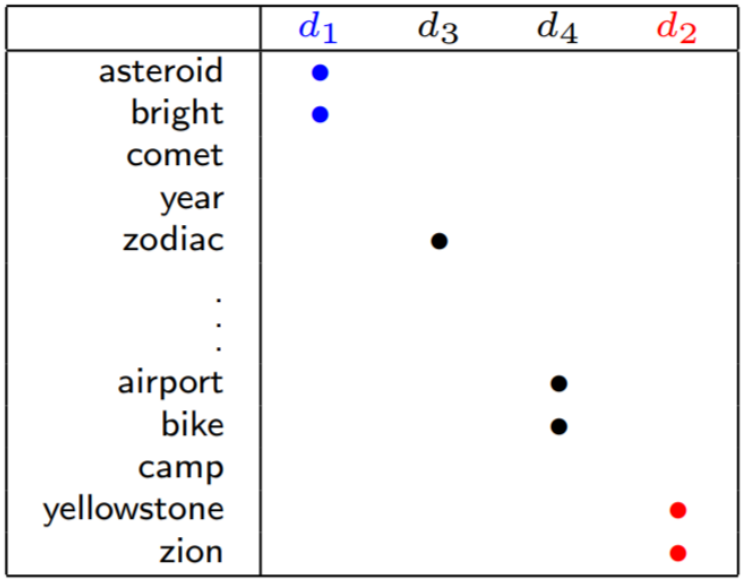
$$\text{subject to } \; y_i(h(x_i) + b) \geq 1 - \xi_i \quad , \forall i = 1 \ldots l$$

$$\xi_i \geq 0$$

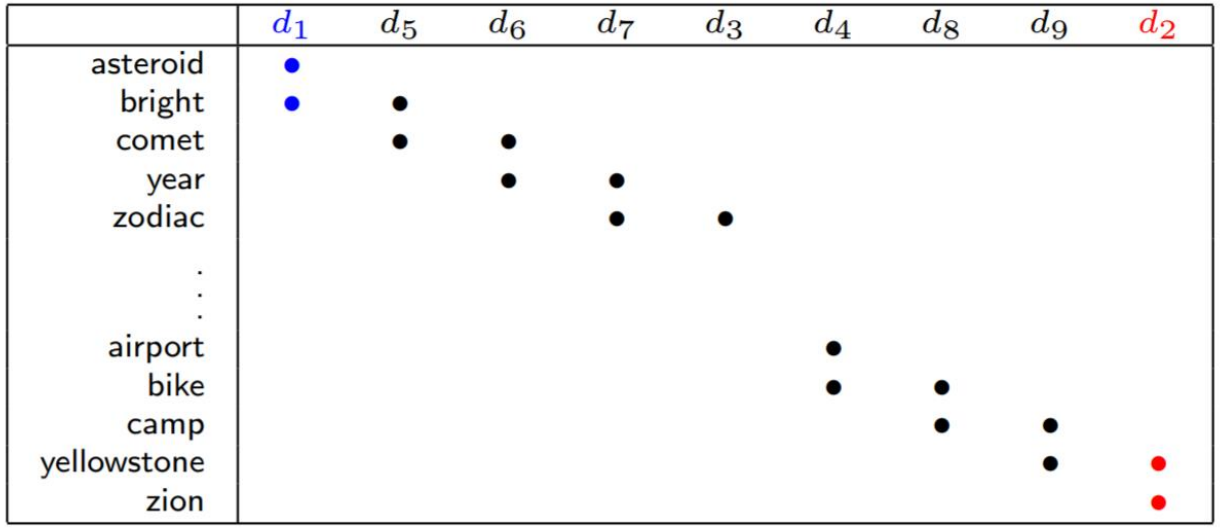
The ξ's are slack variables.

4) Graph-based algorithm

- When labeled data alone fails (겹치는 단어들이 없어 유사도를 구할 수 없음)

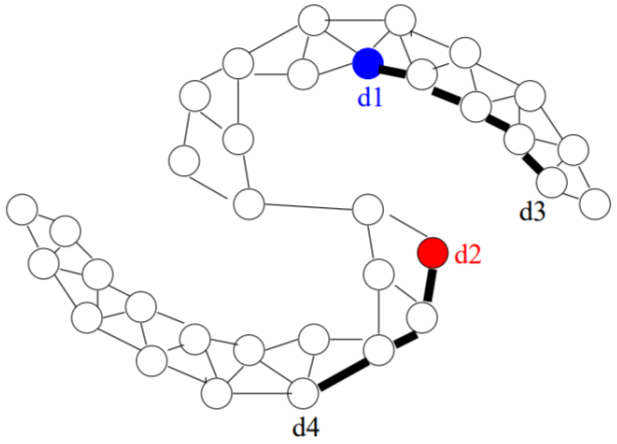


- Labels “propagate” via similar unlabeled articles.



4) Graph-based algorithm

- Nodes: $X_l \cup X_u$
- Edges: similarity weights computed from features, e.g.,
  - k-nearest-neighbor graph, unweighted (0, 1 weights)
  - fully connected graph, weight decays with distance $w = \exp\left(-\|x_i - x_j\|^2/\sigma^2\right)$
- Want: implied similarity via all paths

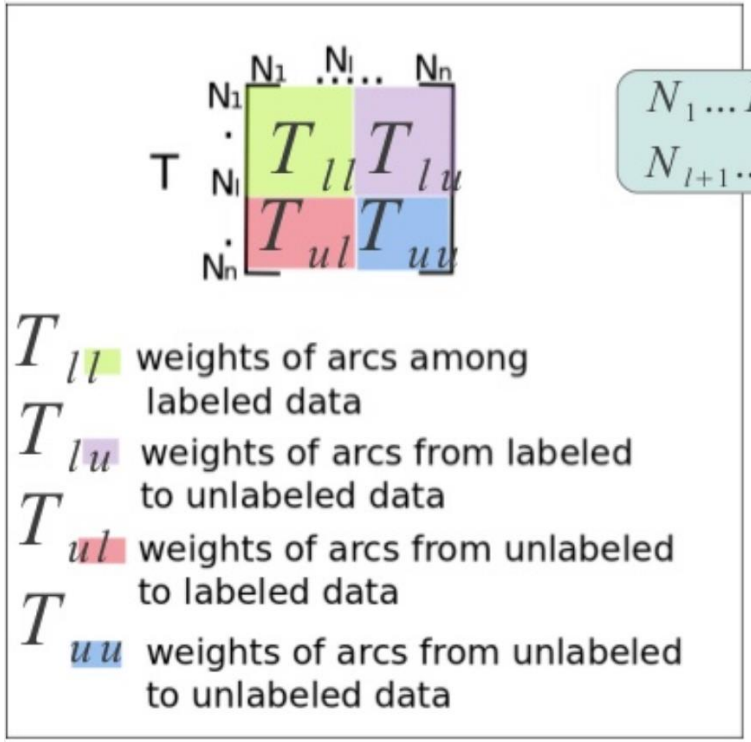


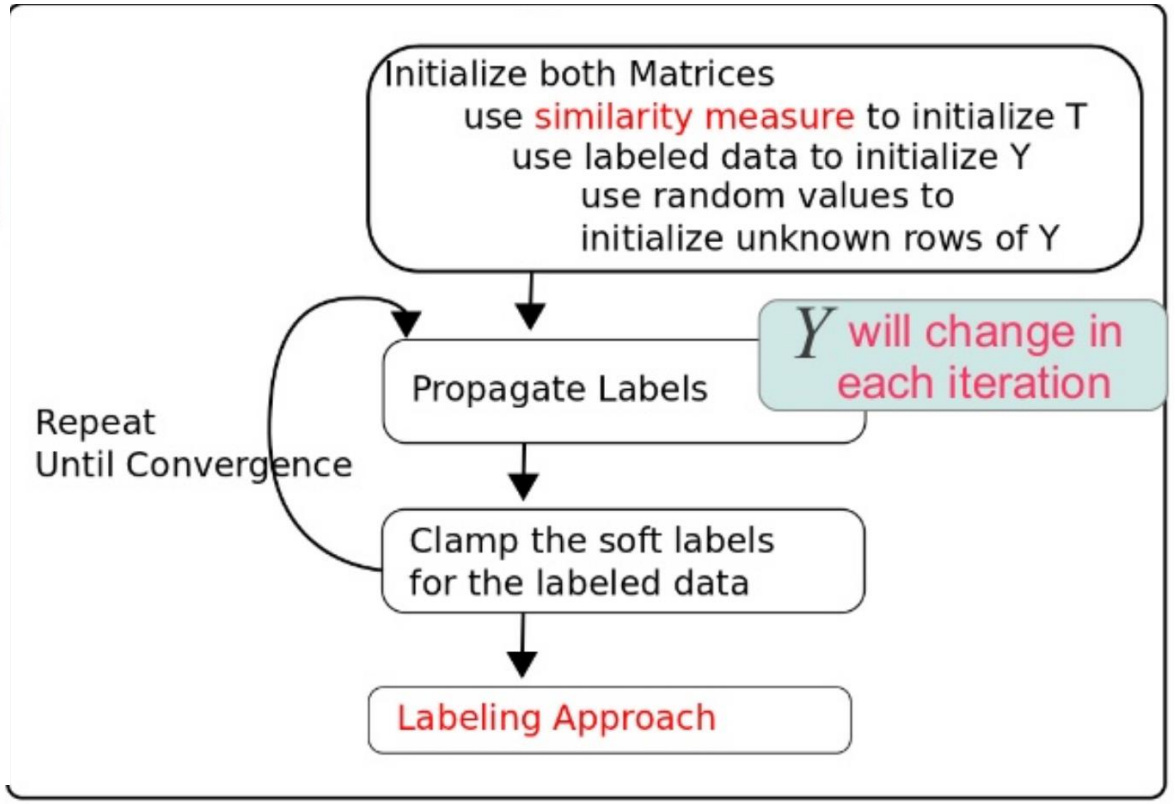
4) Graph-based algorithm

- ‘Smoothness Assumption’: 인접한 이웃일 수록 동일한 레이블을 가질 확률이 높다는 가정(knn의 기본 아이디어와 동일)
- 초기 방식
  - Label Propagation(라벨 전파)
    - 레이블이 부여된 데이터들의 레이블을 이 데이터들과 인접한 레이블이 부여되지 않은 데이터들에 대해서 전파
    - Transition matirx를 활용하여 레이블을 전파, 이는 Markov Chain이 작동하는 방식과 유사
  - Gaussian Field Harmonic Function (GFHF)
    - Weight Matrix로부터 Laplacian Matrix를 도출하고 이를 바탕으로 학습을 수행
  - 한계
    - 레이블이 부여된 데이터의 레이블을 고정하여 학습을 수행하는데, 이는 레이블을 지나치게 신뢰
    - 실제 문제에서는 엔지니어나 설비의 오류로 인해서 레이블이 잘못 부여된 경우가 존재할 수 있음

4) Graph-based algorithm

- T is a matrix, holding all the weights of the graph



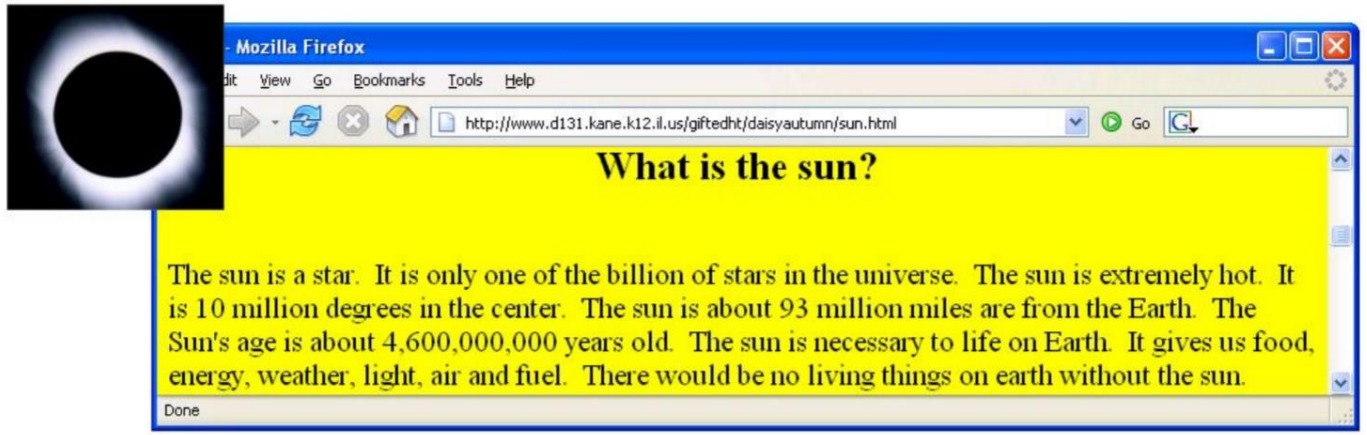


4) Graph-based algorithm

- 한계 극복을 위해 Local and Global Consistency (LGC) 고안
  - Smoothness Function과 레이블이 잘못 부여된 경우를 교정할 수 있도록 하기 위한 Regularization Term을 모두 활용하여 학습을 수행
  - Cluster Assumption을 도입하고, 이 Cluster Assumption에 위배되는 레이블들을 레이블이 잘못 부여된 경우라고 가정하여 이를 교정할 수 있도록 Regularization Term을 도입
    - Cluster assumption: Cluster나 Submanifold와 같이 동일한 구조에 포함된 데이터는 동일한 레이블을 가질 확률이 높다는 가정
  - 한계: 노이즈에 취약

5) Multi-view learning (co-training)

Two views of an item: image and HTML text



5) Multi-view learning (co-training)

- Feature split

Each instance is represented by two sets of features $x = [x^{(1)}; x^{(2)}]$

- $x^{(1)} =$ image features
- $x^{(2)} =$ web page text
- This is a natural feature split (or multiple views)

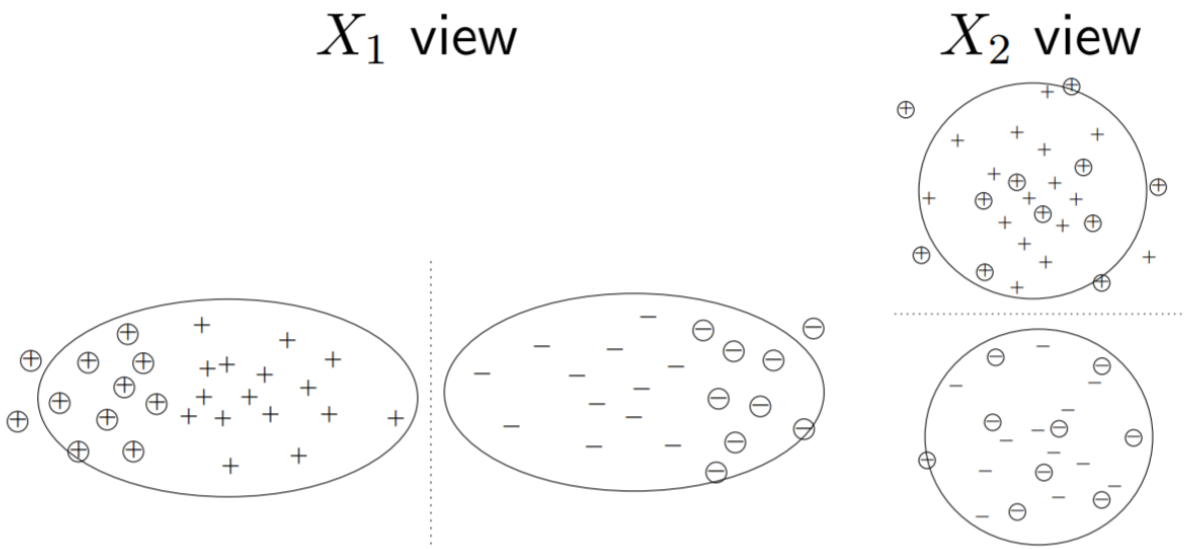
Co-training idea:

- Train an image classifier and a text classifier
- The two classifiers teach each other

5) Multi-view learning (co-training)

Assumptions

- feature split $x = [x^{(1)}; x^{(2)}]$ exists
- $x^{(1)}$ or $x^{(2)}$ alone is sufficient to train a good classifier
- $x^{(1)}$ and $x^{(2)}$ are conditionally independent given the class



5) Multi-view learning (co-training)

Co-training algorithm

1. Train two classifiers: $f^{(1)}$ from $(X_l^{(1)}, Y_l)$, $f^{(2)}$ from $(X_l^{(2)}, Y_l)$.
2. Classify X_u with $f^{(1)}$ and $f^{(2)}$ separately.
3. Add $f^{(1)}$'s k-most-confident $(x, f^{(1)}(x))$ to $f^{(2)}$'s labeled data.
4. Add $f^{(2)}$'s k-most-confident $(x, f^{(2)}(x))$ to $f^{(1)}$'s labeled data.
5. Repeat.

5) Multi-view learning (co-training)

- 장점
  - 간단하고 거의 모든 분류기에 적용이 가능
  - Self-training 방식보다 실수에 덜 민감함
- 단점
  - 자연적인 feature의 분할(split)은 거의 존재하지 않음
  - 두가지 feature 모두 사용하는 모델의 더 성능이 좋아야 함! (실제로는?)

Domain adaptation (DA)

- 많은 실제 시나리오에서 레이블이 지정된 데이터는 훈련된 모델을 적용할 데이터와 몇 가지 주요 측면에서 다름
- 예시
  - 영화에 대한 리뷰 (source domain)를 이용해 가전 제품 (target domain)의 리뷰를 예측하고 싶은 경우
  - 도메인 분류가 된 뉴스 데이터를 이용해 소셜미디어 데이터를 분류하고 싶은 경우 등
- Direct transfer
  - 가장 단순한 방법. source domain을 학습시키고 그 모델을 직접 이용해 target domain에 적용
  - Source와 target 간에 feature를 공유하는 정도에 따라 성능이 좌우됨
- Domain adaptation algorithm
  - 단순한 direct transfer보다 더 잘 전이시키기 위한 방법들
  - 대상 도메인에서 레이블이 지정된 데이터를 사용할 수 있는지 여부에 따라 두 가지 주요 군으로 나누어짐

1) Supervised domain adaptation

- Target 도메인에 소량의 레이블이 지정된 데이터가 있고 source 도메인에는 많은 양의 데이터가 있는 경우. 가장 간단한 접근법은 도메인 차이를 무시하고 소스 및 대상 도메인의 교육 데이터를 병합

2) Unsupervised domain adaptation

- Target 도메인에 레이블이 지정된 데이터가 없는 경우
- 소스 및 대상 도메인의 데이터를 가능한한 유사하게 만들려고 시도.
  일반적으로 소스와 대상 데이터를 공유 공간에 배치하는 투영 함수를 학습

https://medium.com/@sharaf/a-paper-a-day-6-unsupervised-domain-adaptation-by-backpropagation-c004f0d9d9f3
http://theonly1.tistory.com/301