

## Introduction

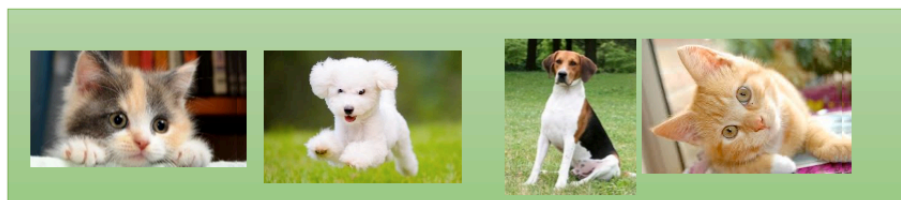
- Supervised learning:  $\{(x^r, \hat{y}^r)\}_{r=1}^R$ 
  - E.g.  $x^r$ : image,  $\hat{y}^r$ : class labels
- Semi-supervised learning:  $\{(x^r, \hat{y}^r)\}_{r=1}^R, \{x^u\}_{u=R}^{R+U}$ 
  - A set of unlabeled data, usually  $U \gg R$
  - Transductive learning: unlabeled data is the testing data
  - Inductive learning: unlabeled data is not the testing data
- Why semi-supervised learning?
  - Collecting data is easy, but collecting “labelled” data is expensive
  - We do semi-supervised learning in our lives

对于猫狗分类问题，如果只有一部分data有label，还有其他很大一部分data是unlabeled，那么我们可以认为unlabeled data对我们网络的训练是无用的吗？

Labelled  
data

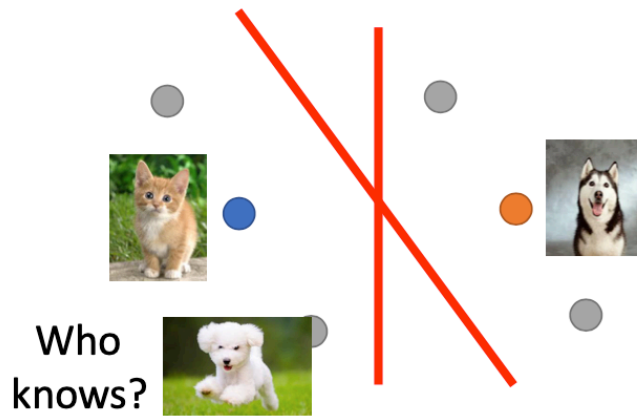


Unlabeled  
data



(Image of cats and dogs without labeling)

Q: Why semi-supervised learning helps ?



The distribution of the unlabeled data tell us ***something***.

Usually with some assumptions

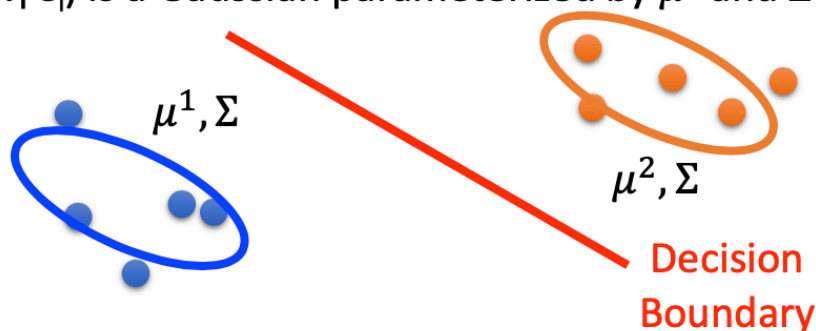
A: 如图所示，图中灰色圆点表示unlabeled data，其他圆点表示labeled data。如果没有unlabeled data，此时可以用一条竖直的线将猫狗进行分类，boundary为竖直的那条线；但unlabeled data的分布也可以告诉我们一些信息，对我们的训练也是有帮助的，有了unlabeled data，此时的boundary为斜直线

## Semi-supervised Learning for Generative Model

### Intuitive

不考虑unlabeled data，只有labeled data

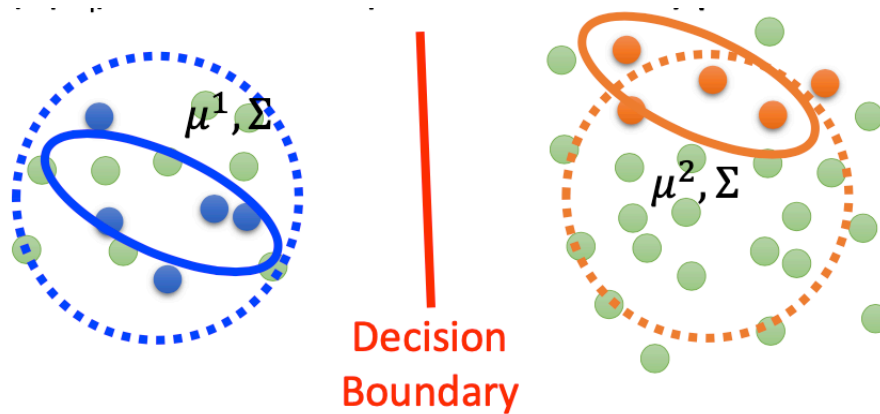
- Given labelled training examples  $x^r \in C_1, C_2$ 
  - looking for most likely prior probability  $P(C_i)$  and class-dependent probability  $P(x|C_i)$
  - $P(x|C_i)$  is a Gaussian parameterized by  $\mu^i$  and  $\Sigma$



With  $P(C_1), P(C_2), \mu^1, \mu^2, \Sigma$

$$P(C_1|x) = \frac{P(x|C_1)P(C_1)}{P(x|C_1)P(C_1) + P(x|C_2)P(C_2)}$$

如果把unlabeled data也考虑进来，此时的boundary 也发生了变化



The unlabeled data  $x^u$  help re-estimate  $P(C_1), P(C_2), \mu^1, \mu^2, \Sigma$

#### Formulation

- Initialization:  $\theta = \{P(C_1), P(C_2), \mu^1, \mu^2, \Sigma\}$

**E**

- Step 1: compute the posterior probability of unlabeled data

$$P_{\theta}(C_1|x^u) \quad \text{Depending on model } \theta$$

**M**

- Step 2: update model

Back to  
step 1

$$P(C_1) = \frac{N_1 + \sum_{x^u} P(C_1|x^u)}{N}$$

$N$ : total number of examples  
 $N_1$ : number of examples  
belonging to  $C_1$

$$\mu^1 = \frac{1}{N_1} \sum_{x^r \in C_1} x^r + \frac{1}{\sum_{x^u} P(C_1|x^u)} \sum_{x^u} P(C_1|x^u) x^u \dots\dots$$

不同的maximum likelihood对比

# Why?

$$\theta = \{P(C_1), P(C_2), \mu^1, \mu^2, \Sigma\}$$

- Maximum likelihood with labelled data **Closed-form solution**

$$\log L(\theta) = \sum_{x^r} \log P_{\theta}(x^r, \hat{y}^r)$$

$$\begin{aligned} P_{\theta}(x^r, \hat{y}^r) \\ = P_{\theta}(x^r | \hat{y}^r) P(\hat{y}^r) \end{aligned}$$

- Maximum likelihood with labelled + unlabeled data

$$\log L(\theta) = \sum_{x^r} \log P_{\theta}(x^r, \hat{y}^r) + \sum_{x^u} \log P_{\theta}(x^u)$$

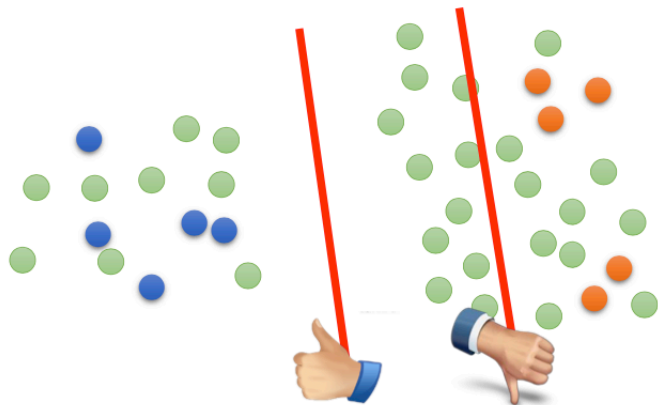
**Solved iteratively**

$$P_{\theta}(x^u) = P_{\theta}(x^u | C_1) P(C_1) + P_{\theta}(x^u | C_2) P(C_2)$$

( $x^u$  can come from either  $C_1$  and  $C_2$ )

## Low-density Separation Assumption

非黑即白  
“Black-or-white”



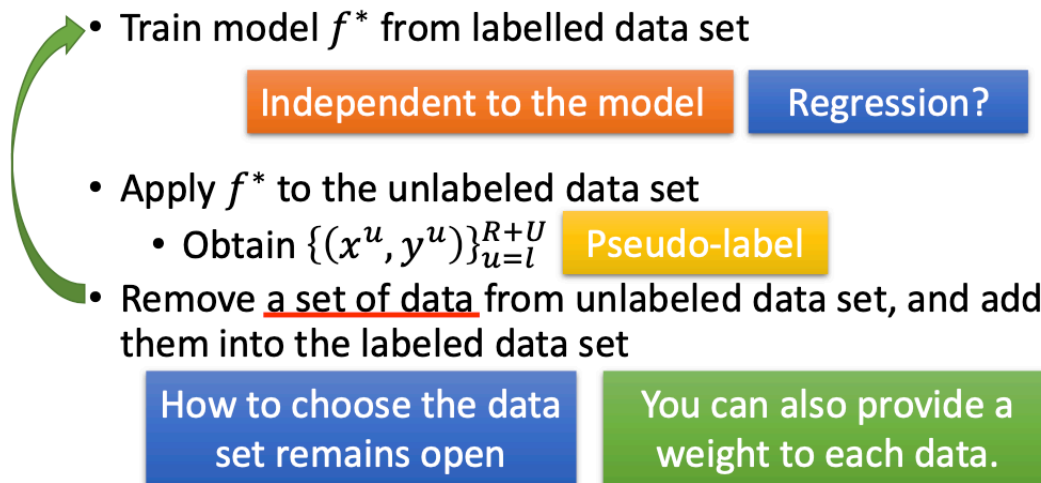
## Self-training

有labeled data和unlabeled data，重复以下过程：

- 从labeled data中训练了模型  $f^*$ ；
- 将  $f^*$  应用到unlabeled data，得到带label的数据，称为Pseudo-label
- 从unlabeled data中移出这部分data，并加入labeled data；要移除哪部分data，要根据具体的限制条件而定
- 有了更多的label data，就可以继续训练我们的模型，返回第一步

- Given: labelled data set  $= \{(x^r, \hat{y}^r)\}_{r=1}^R$ , unlabeled data set  $= \{x^u\}_{u=l}^{R+U}$

• Repeat:



Q: 这种训练方式对regression 有用吗？

W: 不能, regression输出的是一个真实的值

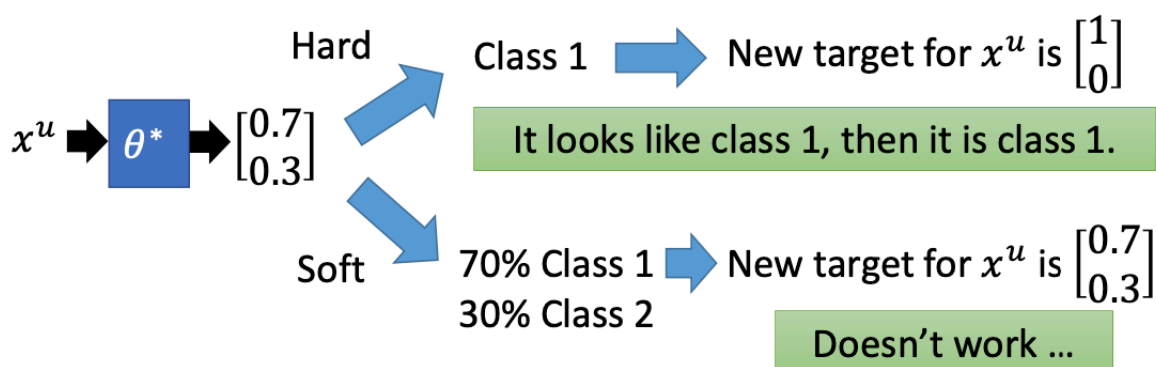
### hard label vs soft label

self-training用的是hard label; generative model用的是soft label

- Similar to semi-supervised learning for generative model
- Hard label v.s. Soft label

Considering using neural network

$\theta^*$  (network parameter) from labelled data



### Entropy-based Regularization

如果输出的每个类别的概率是相近的, 那么这个模型就比较bad; 输出的类别差距很大, 比如某个类别的概率为1, 其他都是0; 我们可以用  $E(y^u)$  来衡量

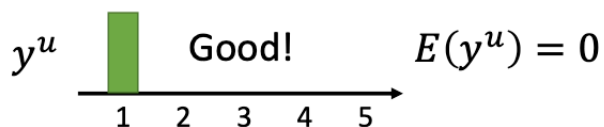
$$E(y^u) = - \sum_{m=1}^5 y_m^u \ln(y_m^u)$$

对于第一个和第二个distribution, 那么 $E(y^u) = 0$ ;

对于第三个distribution, 那么 $E(y^u) = -\ln(\frac{1}{5}) = \ln 5$

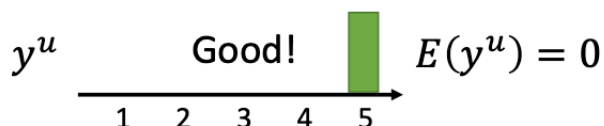


Entropy of  $y^u$  :  
Evaluate how concentrate  
the distribution  $y^u$  is



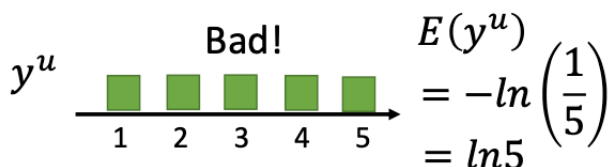
$$E(y^u) = - \sum_{m=1}^5 y_m^u \ln(y_m^u)$$

As small as possible



$$L = \sum_{x^r} C(y^r, \hat{y}^r)$$

labelled data



$$+ \lambda \sum_{x^u} E(y^u)$$

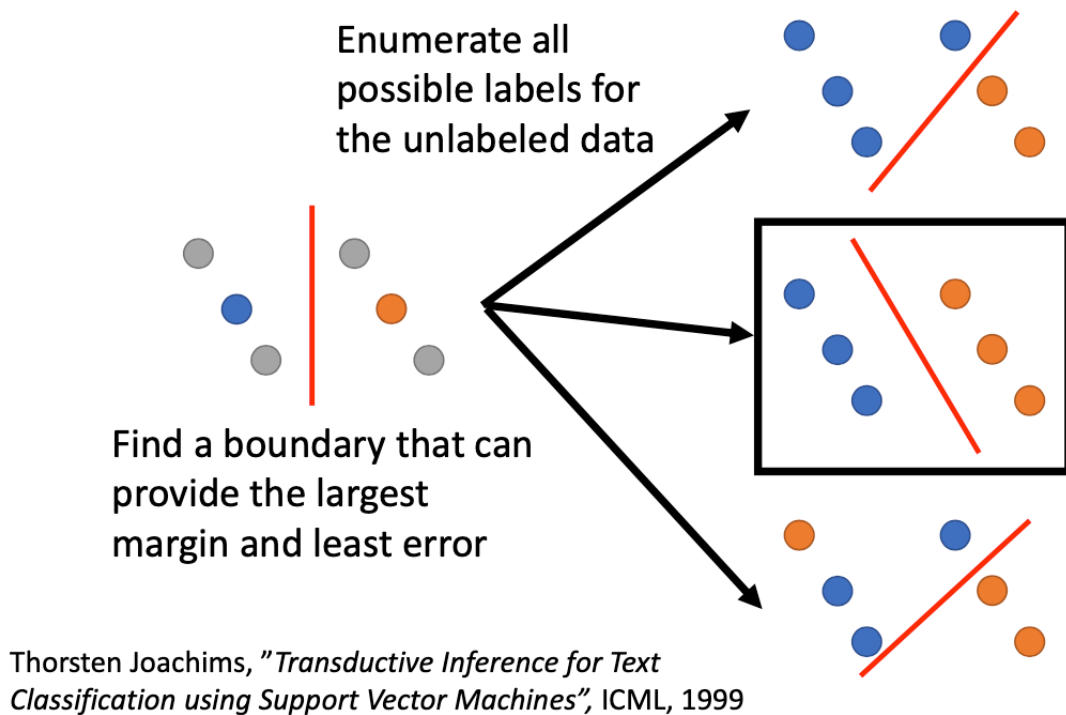
unlabeled data

那么我们现在就可以重新设计loss function, 用cross entropy来估计 $y^r, \hat{y}^r$ 之间的差距, 即 $C(y^r, \hat{y}^r)$ , 使用labeled data, 还加上了一个regularization term

$$L = \sum_{x^r} C(y^r, \hat{y}^r) + \lambda \sum_{x^u} E(y^u)$$

### Outlook: Semi-supervised SVM

对于unlabeled data, 如果是SVM 二分类问题, 可以把所有的unlabeled data都穷举为Class1或Class2, 列举出所有可能的方案, 再找出对应的boundary, 计算loss, 可以发现下图中黑色方框图具有最小的loss



## Smoothness Assumption

### Introduction

近朱者赤，近墨者黑

"You are known by the company you keep"

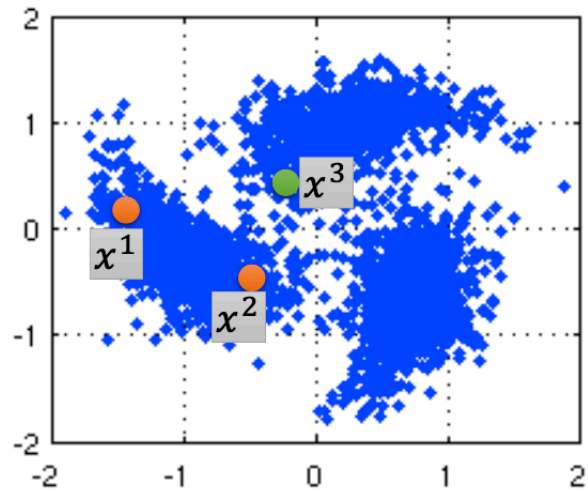
假设：如果 $x$ 是similar的，那么他们的 $y$ 也是一样的

这样的假设是非常不精确的，下面我们做出一个更加精确的假设：

- $x$ 是分布不均匀的，有的地方很密集，有的地方很稀疏
- $x^1, x^2$ 中间有个high density region, 那么label  $y^1, y^2$ 就很可能是一样的；但 $x^2, x^3$ 中间没有high density region, 其label相同的概率就非常小

- Assumption: “similar”  $x$  has the same  $\hat{y}$
- More precisely:
  - $x$  is not uniform.
  - If  $x^1$  and  $x^2$  are close in a high density region,  $\hat{y}^1$  and  $\hat{y}^2$  are the same.

connected by a  
high density path

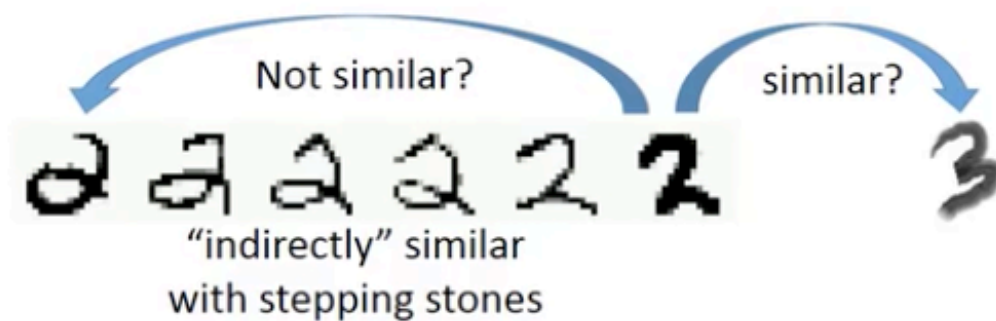


Source of image:  
<http://hips.seas.harvard.edu/files/pinwheel.png>

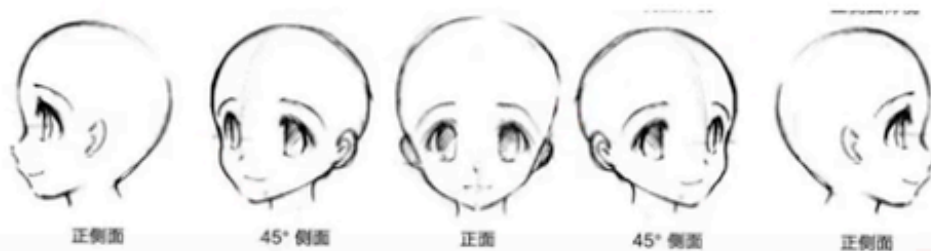
$x^1$  and  $x^2$  have the same label

$x^2$  and  $x^3$  have different labels

对于下图中的数字，2之间是有过渡形态的，所以这两个图片是similar的；而2与3之间没有过渡形态，因此是不similar的



(The example is from the tutorial slides of Xiaojin Zhu.)

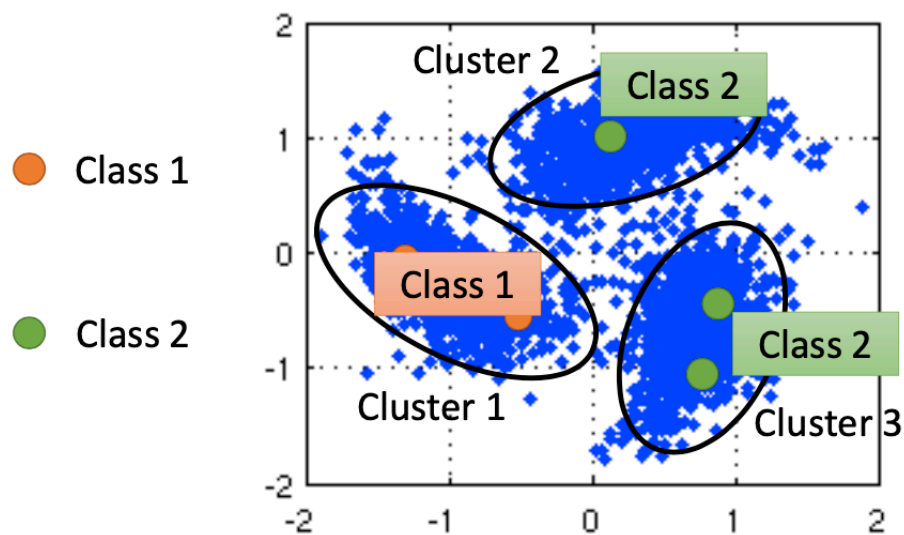


Source of image: <http://www.moehui.com/5833.html/5/>

Created with EverCam

比较直观的做法是先进cluster，再进行label





Using all the data to learn a classifier as usual

### Graph-based Approach

那么我们到底要怎么才能知道 $x^1, x^2$ 到底在high density region是不是close呢？

我们可以把data point用图来表示，图的表示有时是比较nature，有时需要我们自己找出来point之间的联系

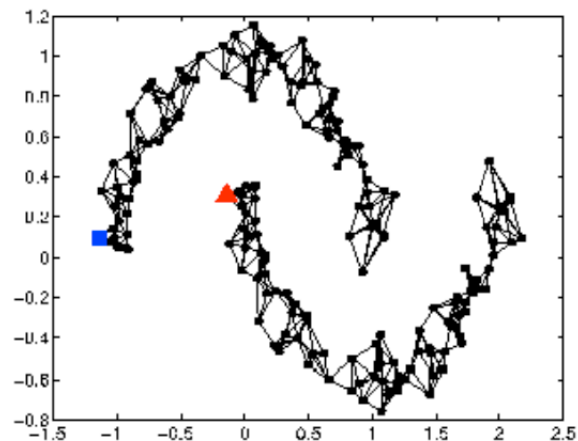
- How to know  $x^1$  and  $x^2$  are close in a high density region (connected by a high density path)

Represented the data points as a **graph**

Graph representation is nature sometimes.

E.g. Hyperlink of webpages, citation of papers

Sometimes you have to construct the graph yourself.

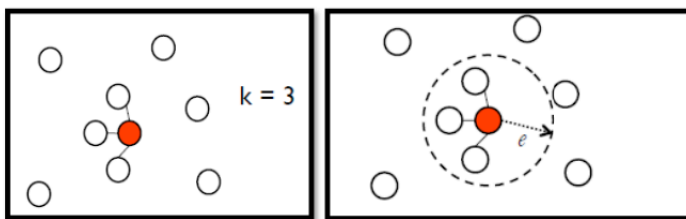


### Graph Construction

首先定义不同point之间的相似度 $s(x^i, x^j)$ ，可以通过以下两个算法来添加edge：

- KNN，对于图中红色的圆点，与其最相近的三个（ $k=3$ ）neighbor相连接
- e-Neighborhood，对于周围的neighbor，只有和他相似度大于1的才会被连接起来

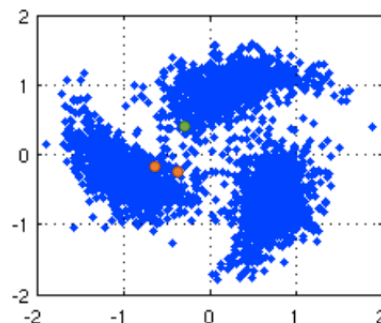
- Define the similarity  $s(x^i, x^j)$  between  $x^i$  and  $x^j$
- Add edge:
  - K Nearest Neighbor
  - $\epsilon$ -Neighborhood



- Edge weight is proportional to  $s(x^i, x^j)$

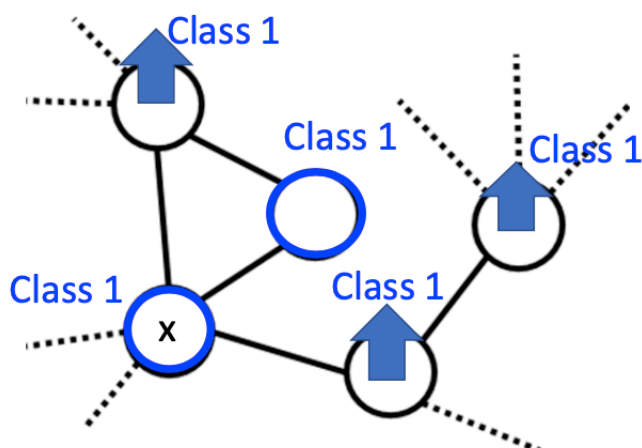
Gaussian Radial Basis Function:

$$s(x^i, x^j) = \exp(-\gamma \|x^i - x^j\|^2)$$



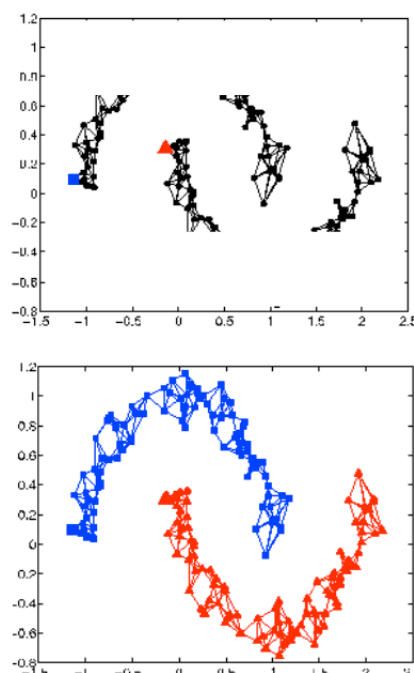
edge并不是只有相连和不相连两种选择而已，也可以给edge一些weight，让这个weight和这两个point之间的相似度成正比

labeled data会影响他的邻居，如果这个point是class1，那么他周围的某些point也可能是class1



The labelled data influence their neighbors.

Propagate through the graph



### Definition

对于下图中的两幅图，如果从直观上看，我们可以认为左边的图更smooth

现在我们用数字来定量描述，S的定义如下

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2$$

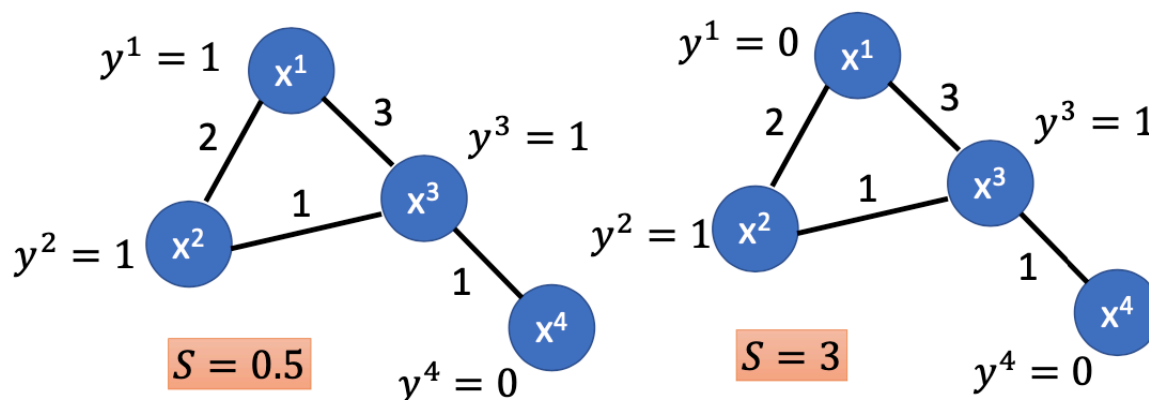
根据公式我们可以算出左图的S=0.5，右图的S=3，值越小越smooth，越小越好

- Define the smoothness of the labels on the graph

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2$$

Smaller means smoother

For all data (no matter labelled or not)



对原来的S进行改造一下,  $S = y^T L y$

其中  $L = D - W$ , W为权重矩阵, D表示将weight每行的和放到对角线的位置

- Define the smoothness of the labels on the graph

$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2 = y^T L y$$

**y:** (R+U)-dim vector

$$\mathbf{y} = [\dots y^i \dots y^j \dots]^T$$

**L:** (R+U) x (R+U) matrix

Graph Laplacian

$$L = \underline{D} - \underline{W}$$

$$W = \begin{bmatrix} 0 & 2 & 3 & 0 \\ 2 & 0 & 1 & 0 \\ 3 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 5 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

loss function其中一项就包括cross entropy计算的loss; smoothness的量S, 前面再乘上一个可以调整的参数 $\lambda$ ,  $\lambda S$ 就表示一个regularization term

网络的整体目标是使loss function 取得最小值, 即cross entropy项和smoothness都必须要达到最小值, 和其他的网络一样, 计算相应的gradient, 做gradient descent即可

如果要计算smoothness不一定非要在output的地方, 也可以是其他位置, 比如hidden layer拿出来进行一些transform, 或者直接拿hidden layer, 都可以计算smoothness

- Define the smoothness of the labels on the graph

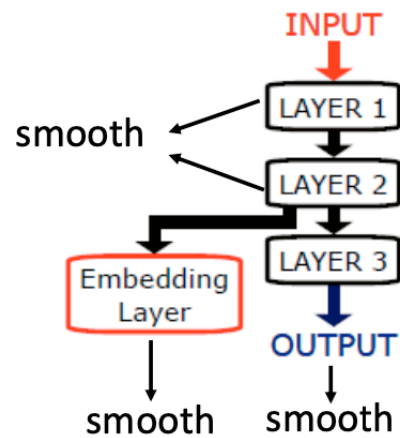
$$S = \frac{1}{2} \sum_{i,j} w_{i,j} (y^i - y^j)^2 = \mathbf{y}^T L \mathbf{y}$$

Depending on network parameters

$$L = \sum_{x^r} C(y^r, \hat{y}^r) + \lambda S$$

As a regularization term

J. Weston, F. Ratle, and R. Collobert, "Deep learning via semi-supervised embedding," ICML, 2008



## Better Representation

- Find the latent factors behind the observation
- The latent factors (usually simpler) are better representations

observation

Better representation  
(Latent factor)

