Semi- and Self-Supervised Learning

Supervised learning framework requires labelled data, which is a **bottleneck** in many problems.

- Medical imaging, object detection…

How to tap into the **abundant data** (images, documents, video,…) **that is not labelled**?

- Unsupervised learning:
  - We have just unlabelled data. We can learn how data is clustered, density estimation, dimensionality reduction, ….
- Self-supervised learning
  - Where the training signal is derived from the data itself or artificially generated
- Semi-supervised learning
  - Use a small amount of labelled data along with the unlabelled data

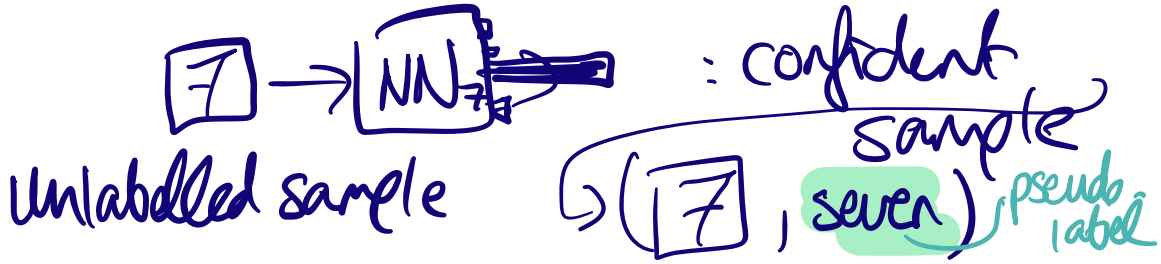
SELF-SUPERVISED LEARNING

Self-Supervised Learning

In self-supervised learning, the supervisory signal can be obtained in two distinct ways:

- **Pretext tasks:**
  - **Labels derived from data:** the system tries to identify a **hidden part of the input** from unhidden parts of the input.
  - E.g. next token prediction
  - **Output**: Representation
  - **Aims to learn good embeddings (representation learning)**

- **Pseudo-labeling:**
  - **Labels derived from model itself or a teacher**
  - E.g. Fixmatch, MeanTeacher, Knowledge Distillation
  - **Output**: Trained model
  - **Typically used in semi-supervised learning with limited data**



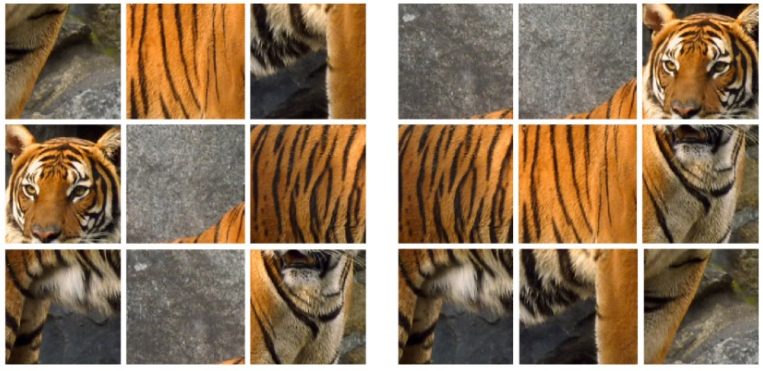
Self-supervised learning with Pretext Tasks

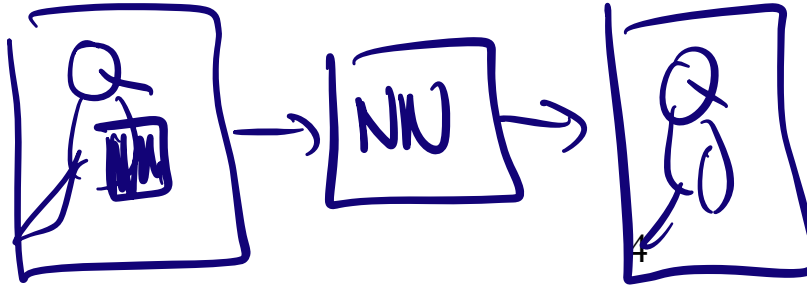
Text

- **Predict the next word given previous words.**
- Predict randomly masked words from sentences.
- Predicts if a given sentence is in continuation to the first sentence it is given.

Image

- **Rearrange the image grids like a jigsaw and train the model to solve it.**
- Rotate the image randomly and train the model to predict the angle.
- Remove small patches from the image and have the network learn to fill it back.
- Decolorize the image and train the model to colorize it.
- Blur the image and train the model to produce a clean image.





In pretext tasks, the network is given a task to learn, but the main goal is to learn useful representations during this process:

- **Rotate the image randomly and train the model to predict the angle.**

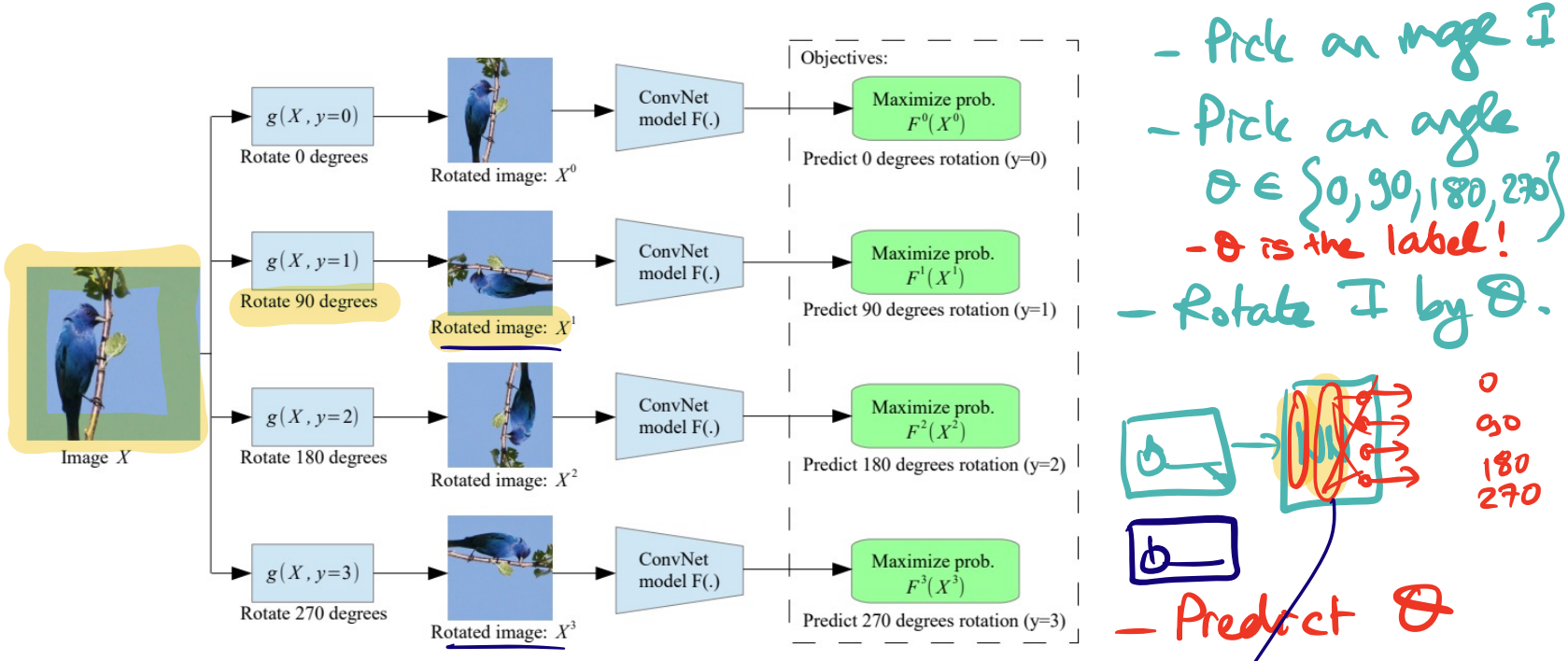


Figure 2: Illustration of the self-supervised task that we propose for semantic feature learning. Given four possible geometric transformations, the 0, 90, 180, and 270 degrees rotations, we train a ConvNet model $F(.)$ to recognize the rotation that is applied to the image that it gets as input. $F^y(X^{y^*})$ is the probability of rotation transformation y predicted by model $F(.)$ when it gets as input an image that has been transformed by the rotation transformation y^*.

CORE PRINCIPLES
FOR EXPLOTING UNLABELLED DATA

Self-Supervised Learning
pseudo-labels

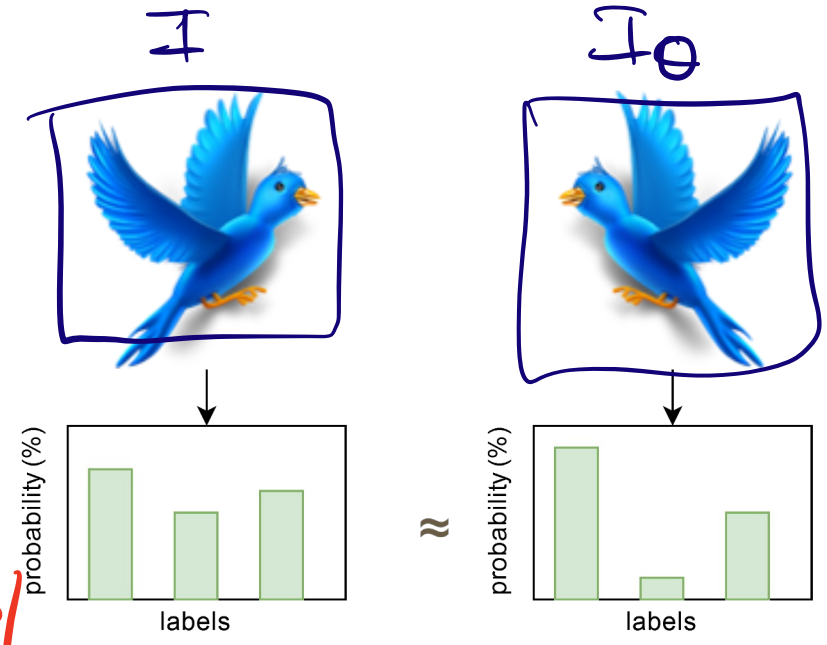
Pseudo-labels: The model or a teacher (a pretrained model) assigns a label (prediction) to an unlabelled data..

Self-Supervised Learning
Consistency regularization

Consistency regularization: **Images that are obtained with small augmentations** should have **similar representations**.

$$\text{loss}\left(\left|p_{\text{model}}(y \mid \text{Augment}(x); \theta) - p_{\text{model}}(y \mid \text{Augment}(x); \theta)\right|\right)$$

x' x''

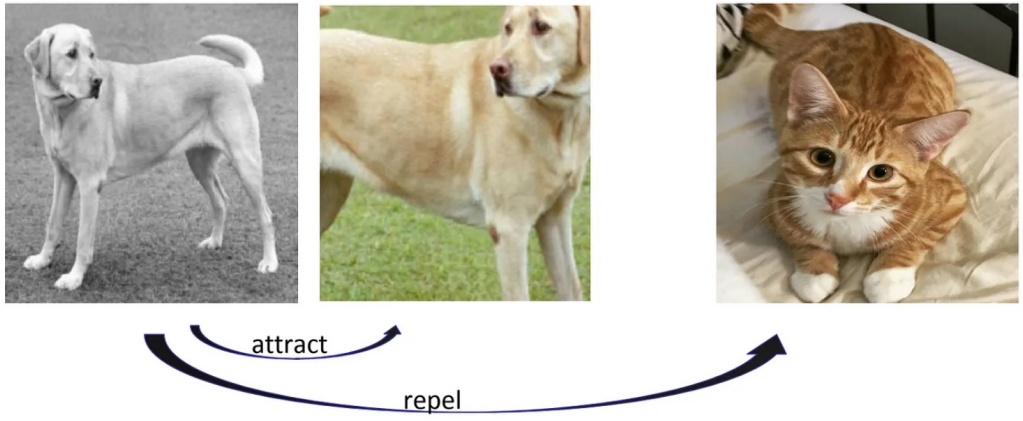


want similar embeddings/ labels.

- weak augmentaθ
  - small/simple transformaθ
    - small rotaθs
    - ic crops
    - flipping ---
- strong augmentaθs more drastic change

Self-supervised learning
Contrastive learning

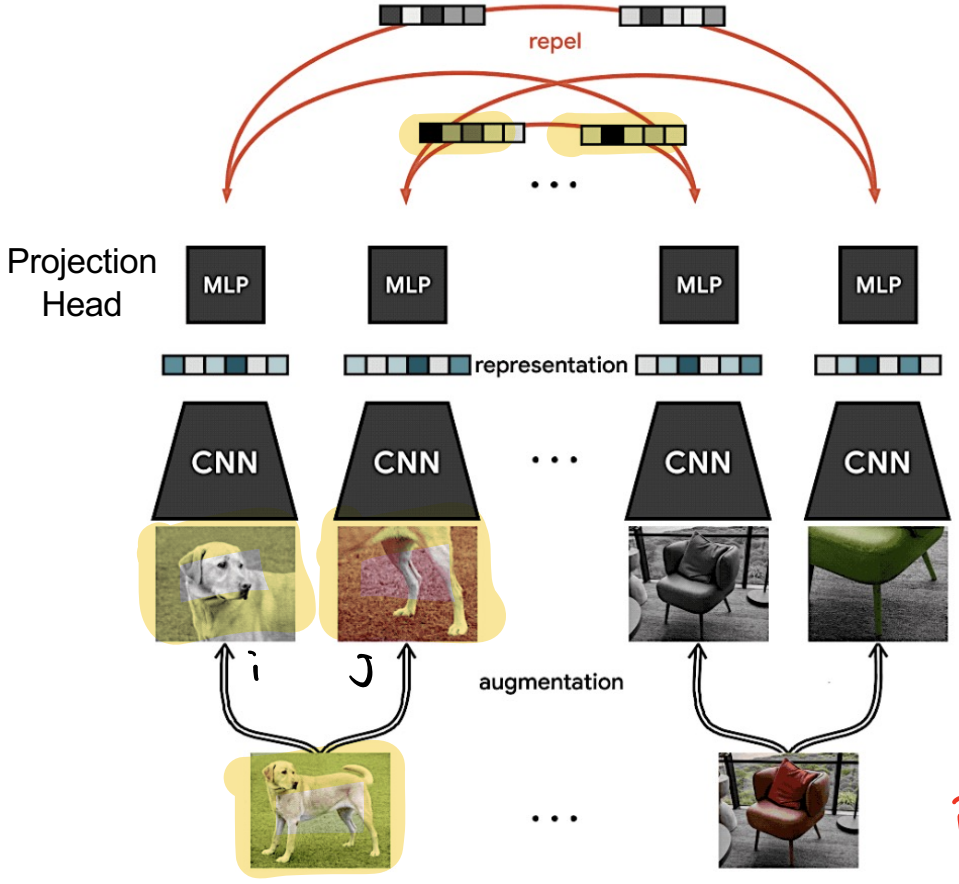
Contrastive Learning: Small transformations of an image should result in embeddings that are more similar to each other than other images.



Instance Discrimination. Image Credits: Deep Unsupervised Learning — P. Abbeel, P. Chen, J. Ho, A. Srinivas, A. Li, W. Yan — L7 Self-Supervised Learning

SimCLR – Chen & Hinton et al, 2020

https://arxiv.org/pdf/2002.05709.pdf



- Data: Unlabeled Dataset
- Maximize the similarity between the augmentations of an image - consistency regularization
- Contrastive Learning

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} {}_{[k \neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)},$$

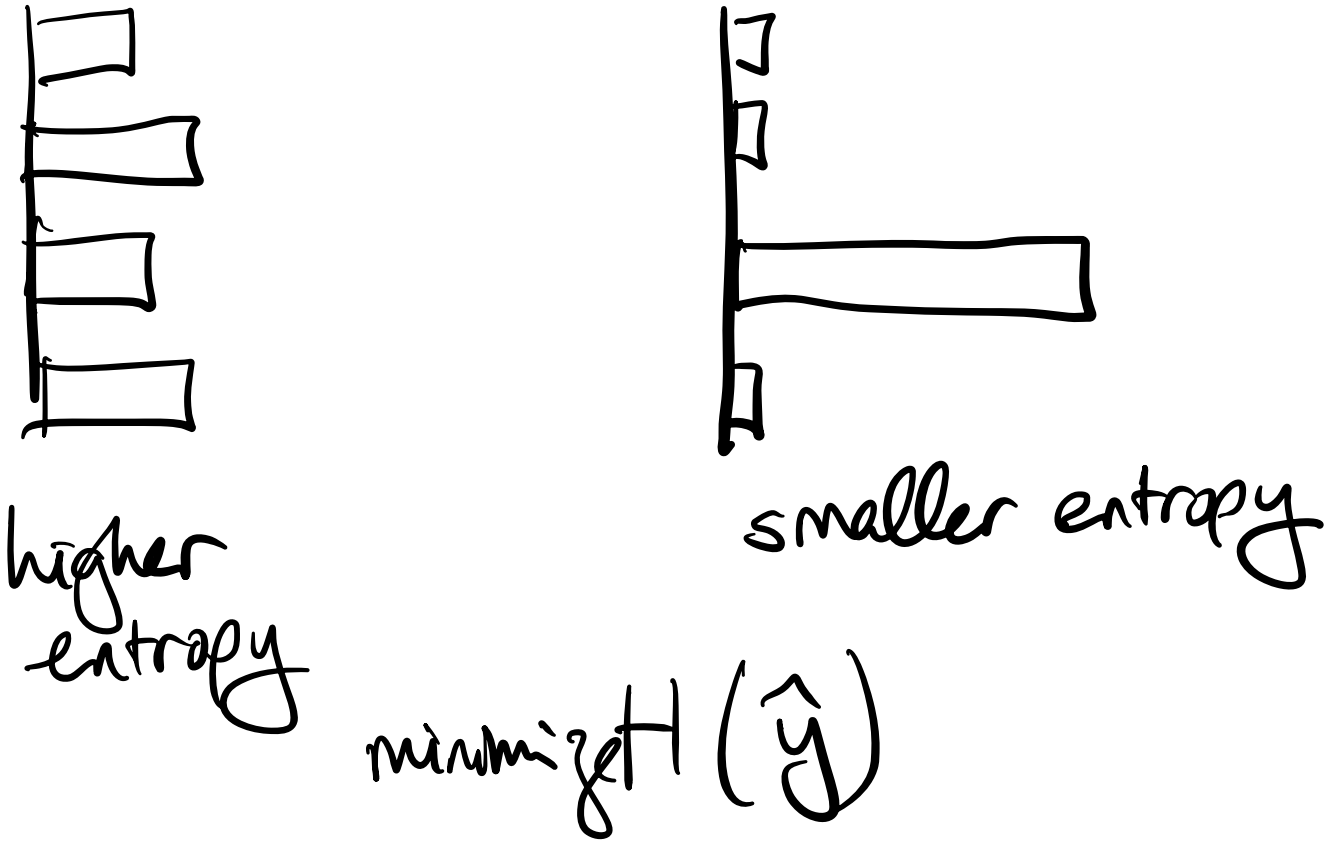
sim. of non-matching pairs

mini-batch of 2N images

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

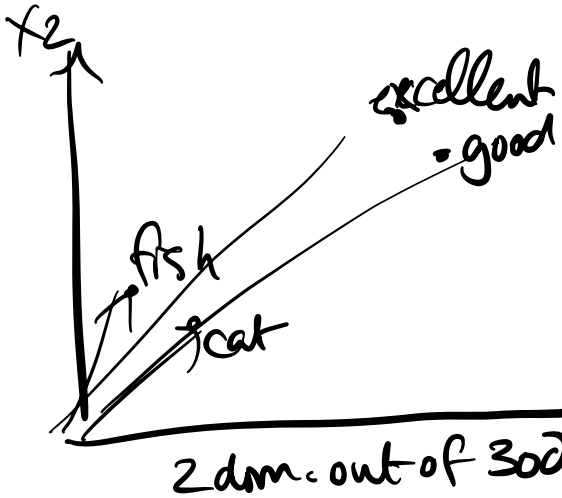
Semi-Supervised Learning

Entropy minimization: Encourage confident predictions on unlabeled data.



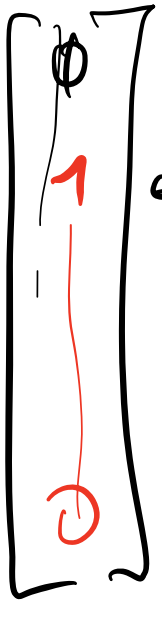
Embeddings

excellent

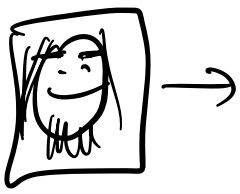


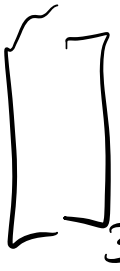
t-SNE

one-hot encoding



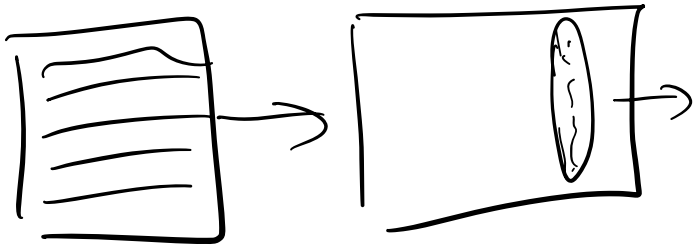
Word Embed.
like Word2Vec

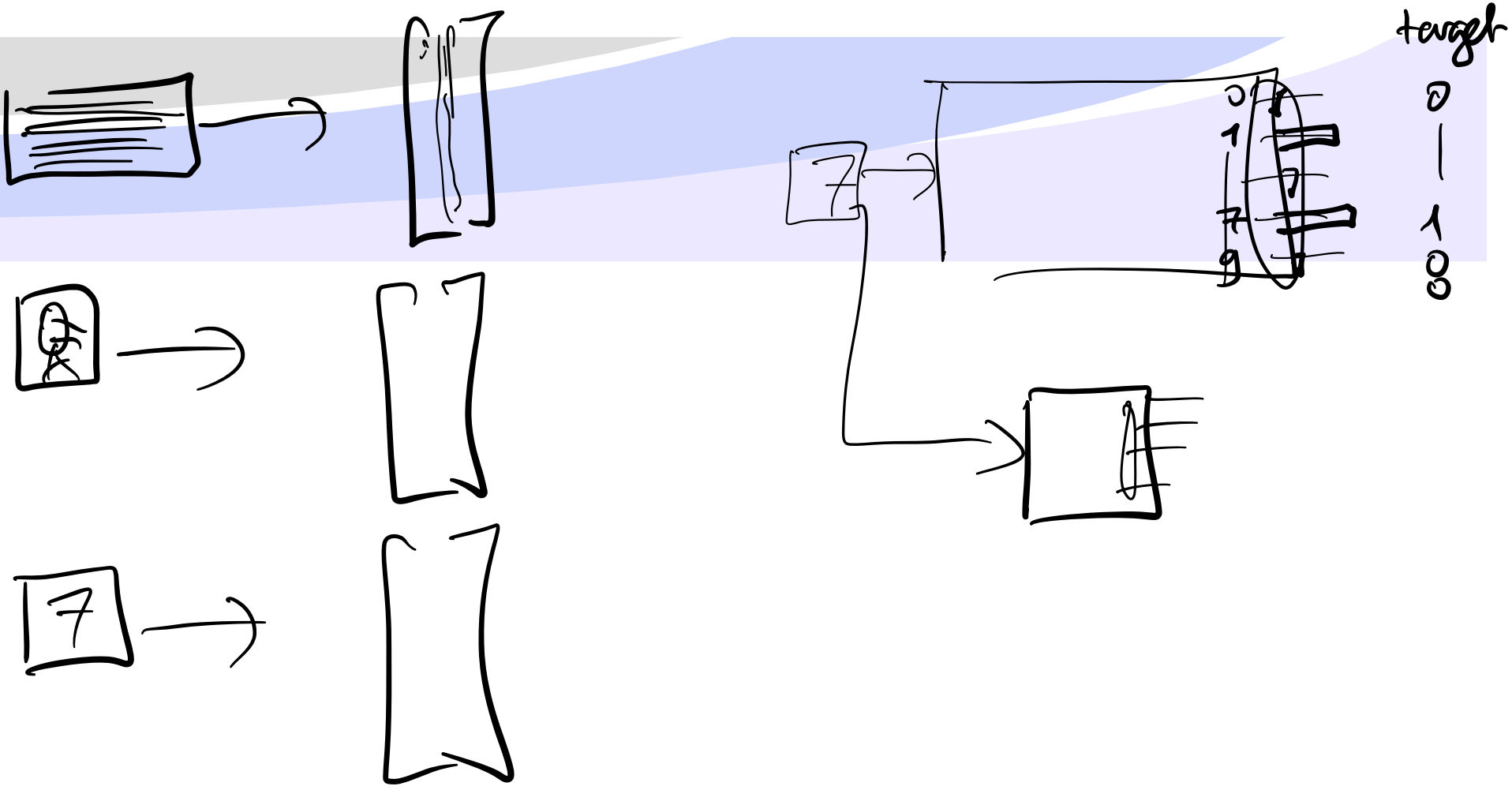




$e_{excellent}$

$\approx$ e_{good}





SEMI-SUPERVISED LEARNING

embedding

Semi-Supervised Learning

In this case, there is **some amount of labelled data.**

The goal is typically to increase performance in the given domain by leveraging the **unlabelled data.**

Early work:

- Self-training / Confident pseudo-labeling
  - Train a classifier on labeled data
  - Predict labels on unlabeled data
  - Accept only high-confidence predictions as true
  - Retrain.
- Expectation-Maximization
- Co-training
- …

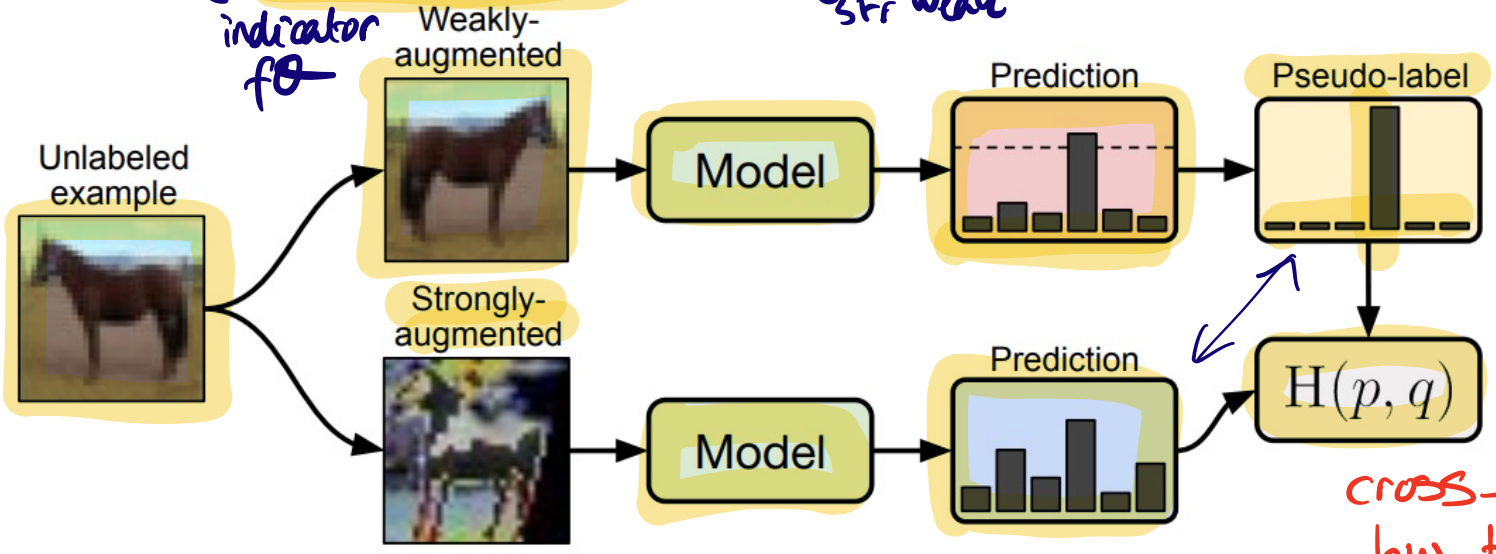
FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence-2020

$$\mathcal{L} = \underbrace{\mathcal{L}_{sup}}_{\text{labeled data}} + \lambda \underbrace{\mathcal{L}_{u}}_{\text{unlabeled data}}$$

- Uses both labelled and unlabelled data
- Pseudo-label from weak augmentation
- Consistency regularization between weak and strong augmentations

$L_u = \sum_u \left(\mathbb{1}(\max(\hat{y}_{weak}) \geq thr) \cdot loss(\hat{y}, \hat{y}_{str\ weak})\right)$

indicator fn



cross-entropy bw two distr.

Figure 1: Diagram of FixMatch. A weakly-augmented image (top) is fed into the model to obtain predictions (red box). When the model assigns a probability to any class which is above a threshold (dotted line), the prediction is converted to a one-hot pseudo-label. Then, we compute the model's prediction for a strong augmentation of the same image (bottom). The model is trained to make its prediction on the strongly-augmented version match the pseudo-label via a cross-entropy loss.

p & q.

Note: Cross-entropy loss $= -\sum y \cdot \log \hat{y}$ $\quad H(p,q) = -\sum_x p(x) \cdot \log_2 q(x)$

Cross-Entropy

(measure of mismatch bw. two distr. p & q)

$$H(p, q) = -\sum_y p(y) \log q(y)$$

- **General concept**: measures how well q approximates p
- Defined for *any* two distributions on the same support

$$H(p, p) = -\sum_y p(y) \log p(y)$$

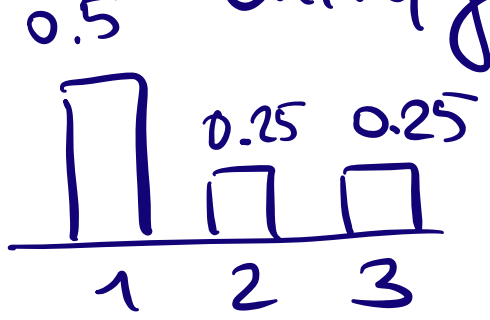
1 for 1 ; 0 1 / 1 0

$H(p,p) = -\Sigma p(y) \cdot \log_2 p(y)$

= entropy.

$$H(p, q) = H(p) + \mathrm{KL}(p \| q)$$

extra bits needed to encode



Entropy of 3-sided biased dice

$= +0.5 \cdot \log_2 \frac{1}{0.25} \cdot 2$

MixMatch: A Holistic Approach to Semi-Supervised Learning - 2019

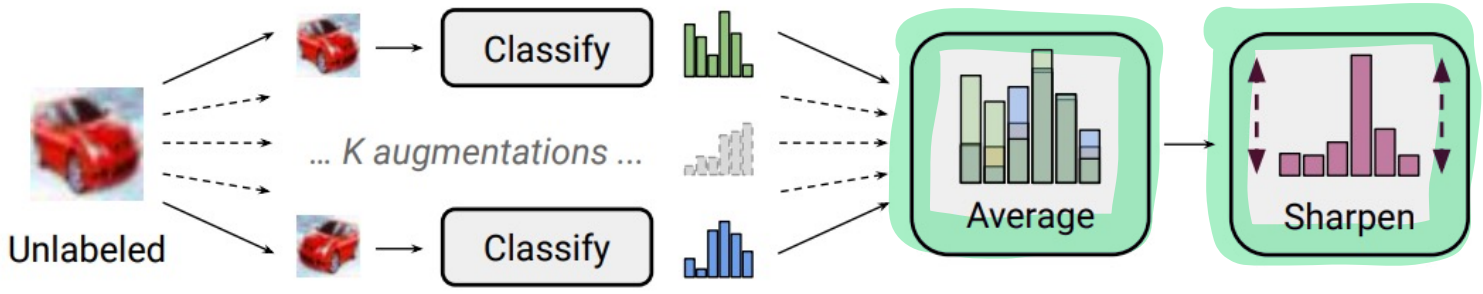


Figure 1: Diagram of the label guessing process used in MixMatch. Stochastic data augmentation is applied to an unlabeled image K times, and each augmented image is fed through the classifier. Then, the average of these K predictions is "sharpened" by adjusting the distribution's temperature.

$$\text{Sharpen}(p, T)_i := p_i^{\frac{1}{T}} \Big/ \sum_{j=1}^{L} p_j^{\frac{1}{T}}$$

As T → 0, the output of Sharpen(p, T) will approach a Dirac ("one-hot") distribution.

MixMatch

- **Inputs:**
  - Labeled dataset $\mathcal{X}$
  - (same size) Unlabeled dataset $\mathcal{U}$
- **Outputs:**
  - Augmented labeled dataset $\mathcal{X}'$
  - Augmented and "guessed" unlabeled dataset $\mathcal{U}'$

One can then use these datasets to train a model using a special loss function:

$$\mathcal{L}_{\mathcal{X}} = \frac{1}{|\mathcal{X}'|} \sum_{x,p \in \mathcal{X}'} \mathrm{H}(p, \mathrm{p}_{\mathrm{model}}(y \mid x; \theta))$$

Cross-entropy loss

$$\mathcal{L}_{\mathcal{U}} = \frac{1}{L|\mathcal{U}'|} \sum_{u,q \in \mathcal{U}'} \|q - \mathrm{p}_{\mathrm{model}}(y \mid u; \theta)\|_2^2$$

Entropy minimization

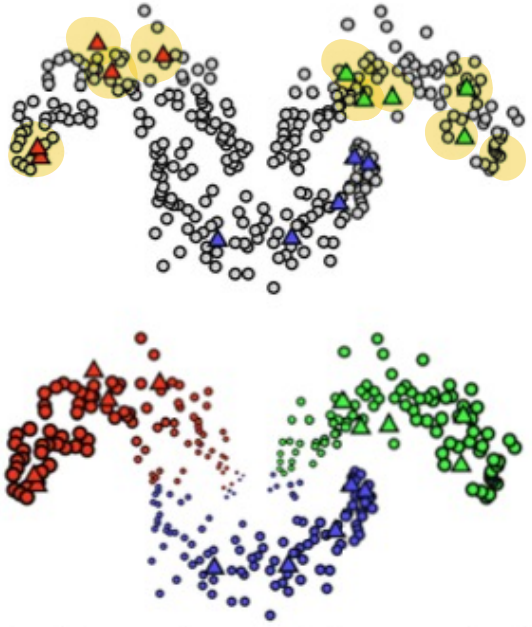
$$\mathcal{L} = \mathcal{L}_{\mathcal{X}} + \lambda_{\mathcal{U}} \mathcal{L}_{\mathcal{U}}$$

Semi-Supervised Learning

3) Graph-based methods (classical SSL)

Build a similarity graph; propagate labels by minimizing a graph Laplacian energy.

Examples: Label Propagation, Label Spreading.



Unsupervised Learning

Some of the slides are thanks to Öznur Taştan, Emily Fox and Carlos Guestrin

Unsupervised Learning

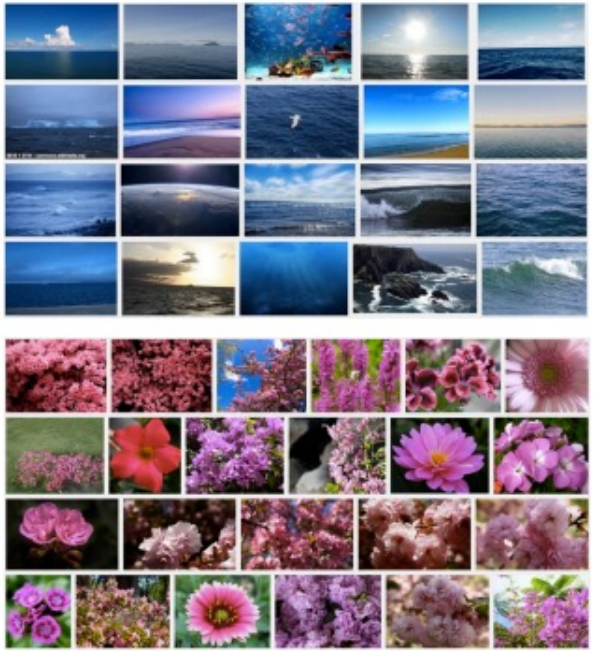
In unsupervised clustering we do not have labelled data

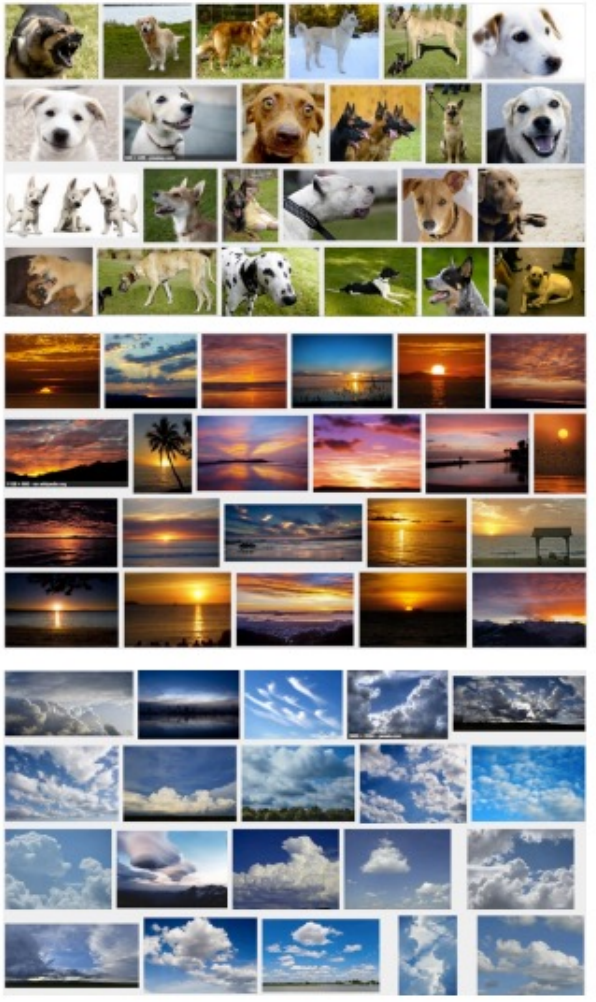
$\{(x^{(i)},y^{(i)})\}$ versus $\{(x^{(i)})\}$

- Clustering:
  - Identify unknown structure in data
- Dimensionality reduction:
  - Reduce the feature dimensionality using the structural characteristics of the data (reduce computation, visualize in 2/3D,…)
- Semi-supervised learning
- Self-supervised learning
- …

Clustering images

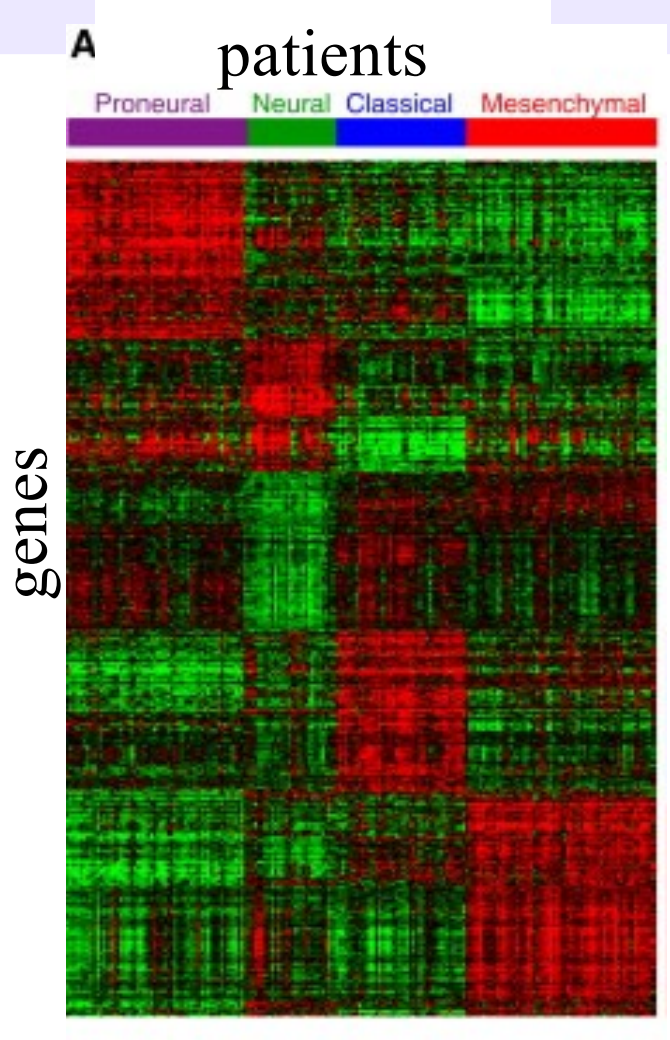
- For search, group as:
  - Ocean
  - Pink flower
  - Dog
  - Sunset
  - Clouds
  - ...





Clustering patients

Find subgroups of patients that have very highly similar molecular profiles



Verhaak et al., Cancer Cell, 2010.

Clustering pixels - Image segmentation

Goal: Break up the image into meaningful or perceptually similar regions

[Slide from James Hayes]

Clustering Applications

Anomaly detection

- Detect samples that are far from learned clusters to detect anomaly
  - Credit card fraud
  - Event detection

Customer Segmentation

- See what kinds of customers there are and possibly target them separately etc.
- Even if we do not have labels for customer groups, we can group similar customers in a group

Clustering

No labels provided

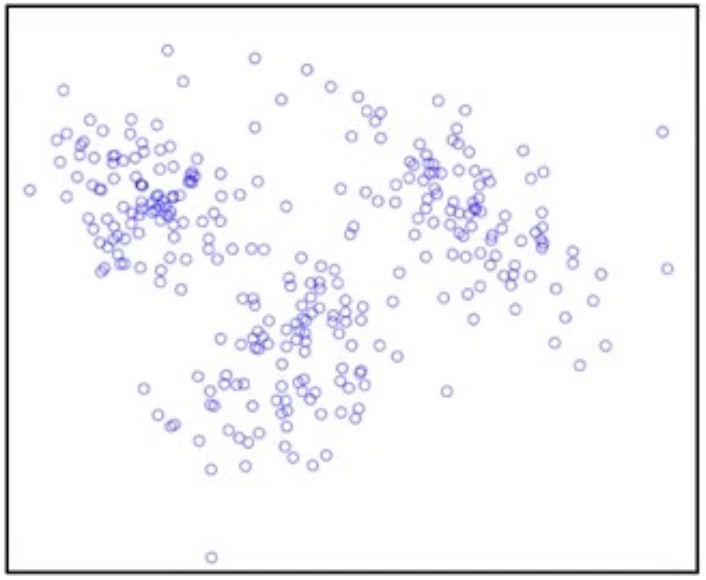
...uncover cluster structure from input alone

Input: docs as vectors $\mathbf{x}_i$

Output: cluster labels z_i

An **unsupervised learning** task

Unsupervised clustering of documents



Slide from Emily Fox & Carlos Guestrin.

Clustering

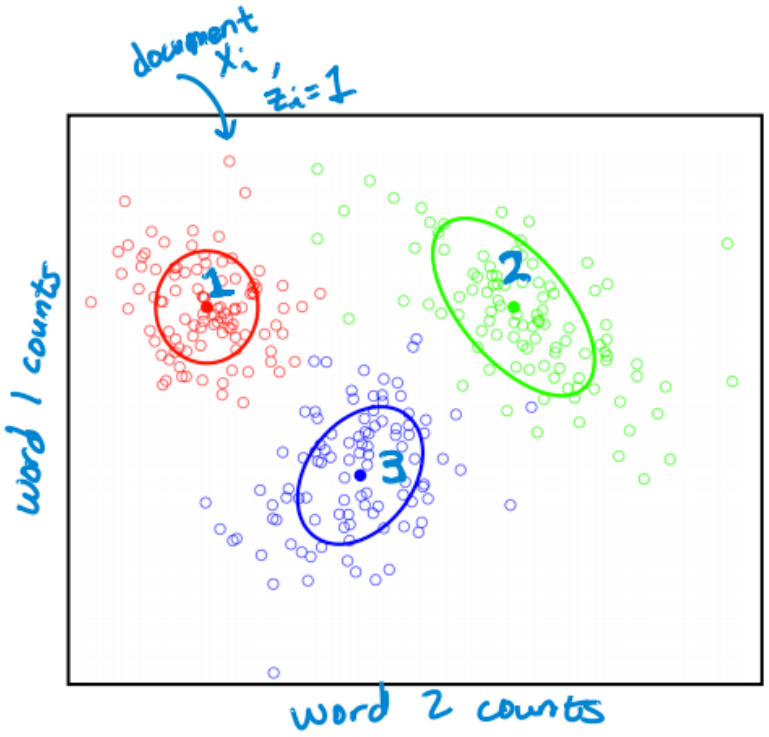
No labels provided

...uncover cluster structure from input alone

Input: docs as vectors $\mathbf{x}_i$

Output: cluster labels z_i

An **unsupervised learning** task



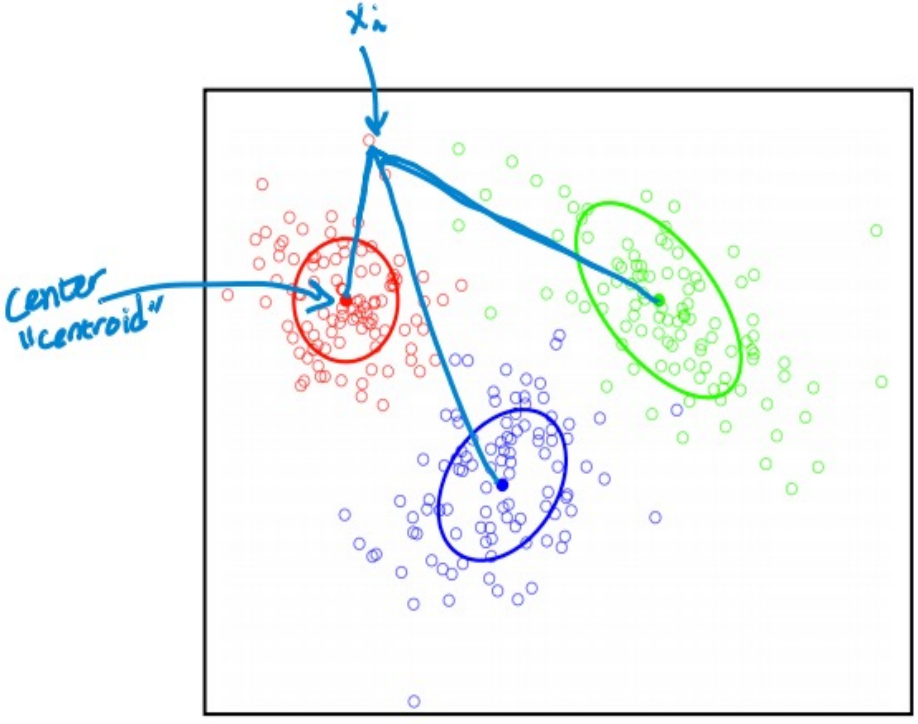
Slide from Emily Fox & Carlos Guestrin.

Clustering

Cluster defined by
center & shape/spread

Assign observation $\mathbf{x}_i$ (doc) to cluster k (topic label) if

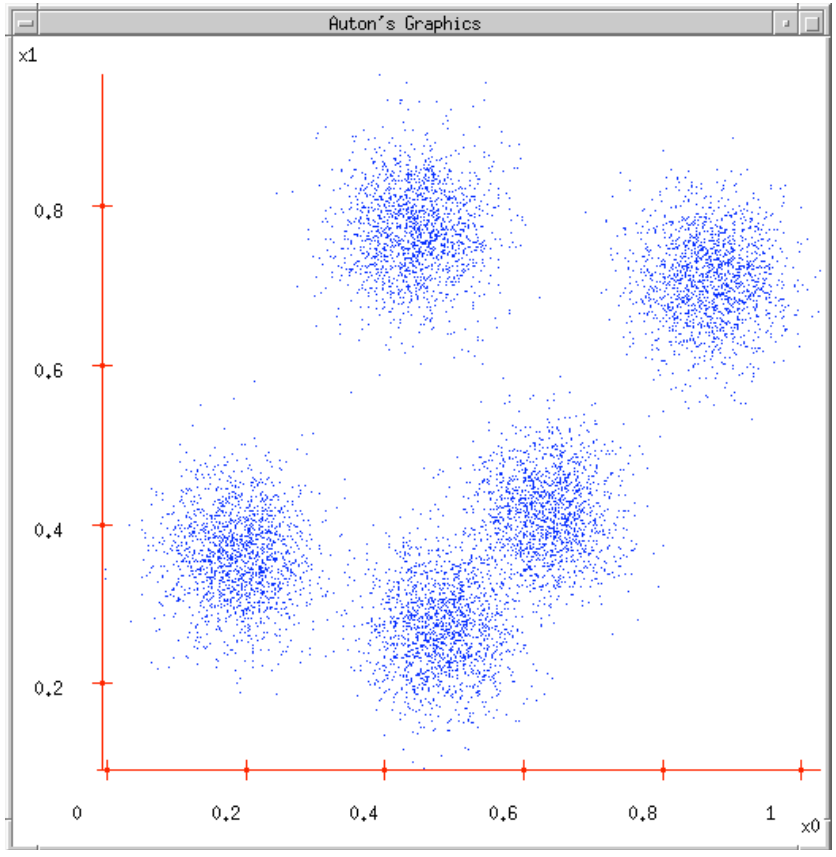
- Score under cluster k is higher than under others
- For simplicity, often define score as distance to cluster center (ignoring shape)



Slide from Emily Fox & Carlos Guestrin.

K-means clustering algorithm

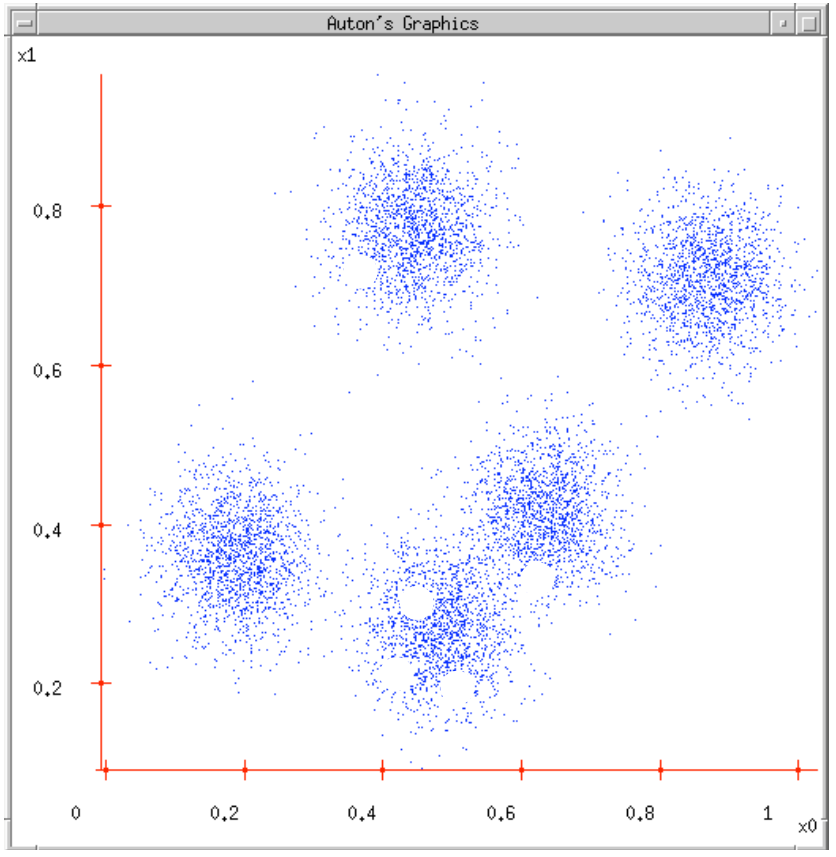
The most commonly used clustering algorithm.



K-Means Clustering

K-Means (k , data)

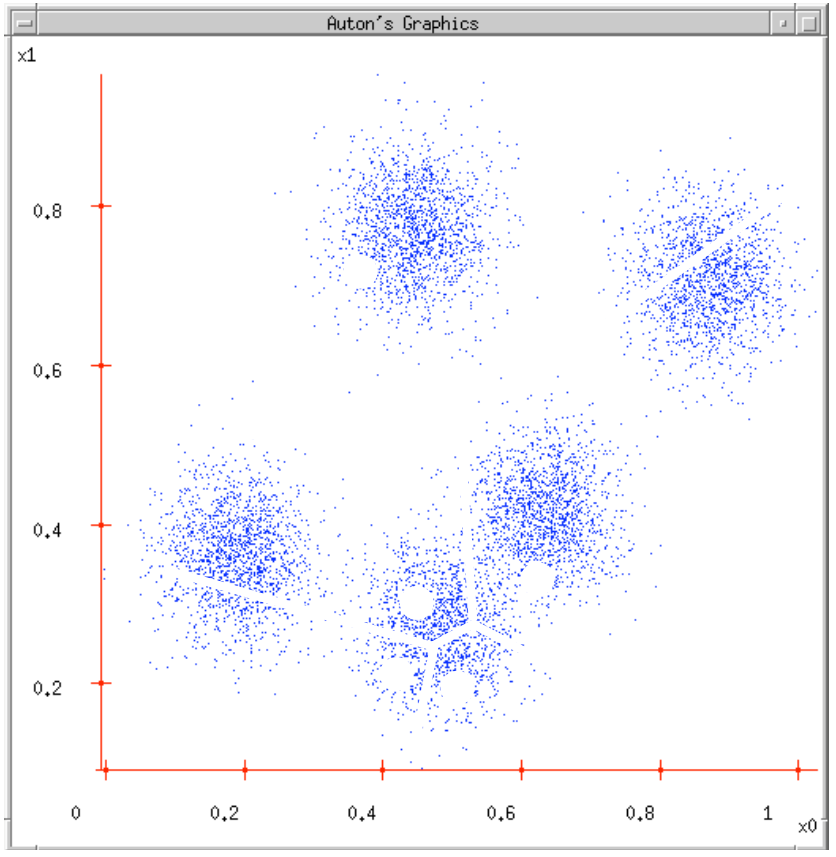
- Randomly choose k cluster center locations (centroids).
  - Can be better initialized
- Loop until convergence
  - Assign each point to the cluster of the closest centroid.
  - Reestimate the cluster centroids based on the data assigned to each.



K-Means Clustering

K-Means (k , data)

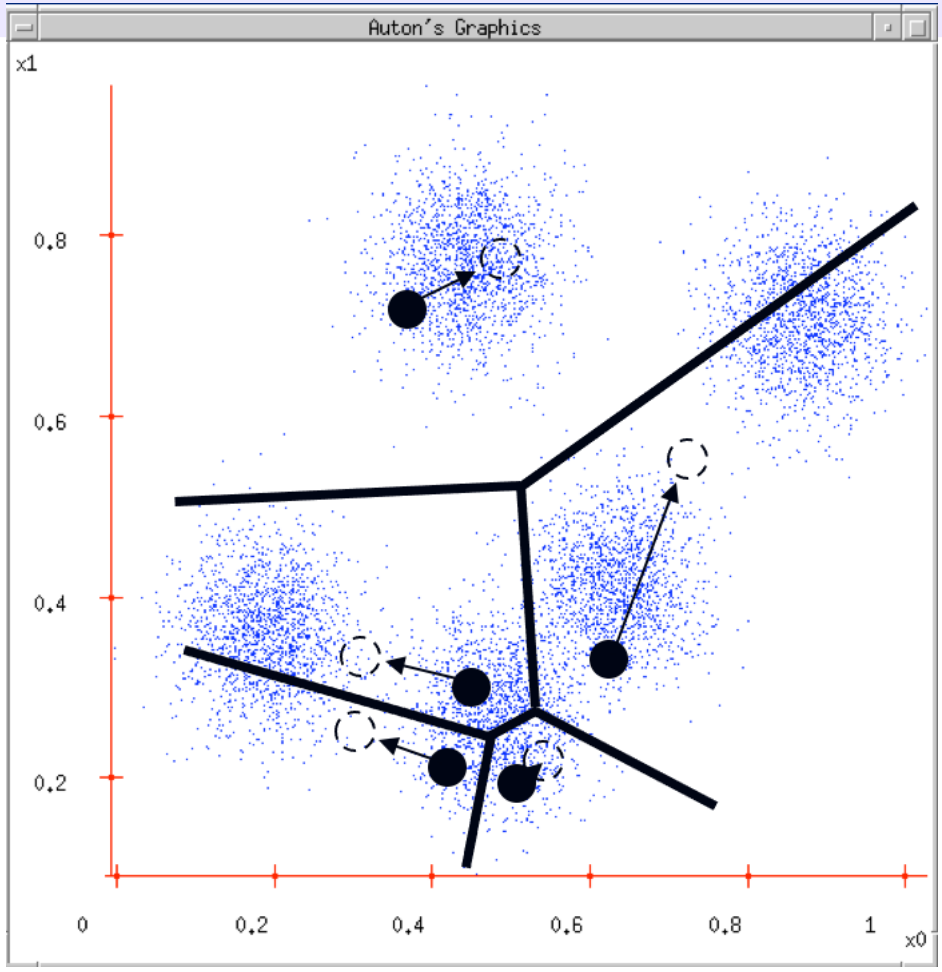
- Randomly choose k cluster center locations (centroids).
  - Loop until convergence
    - Assign each point to the cluster of the closest centroid.
    - Reestimate the cluster centroids based on the data assigned to each.



K-Means Clustering

K-Means (k , data)

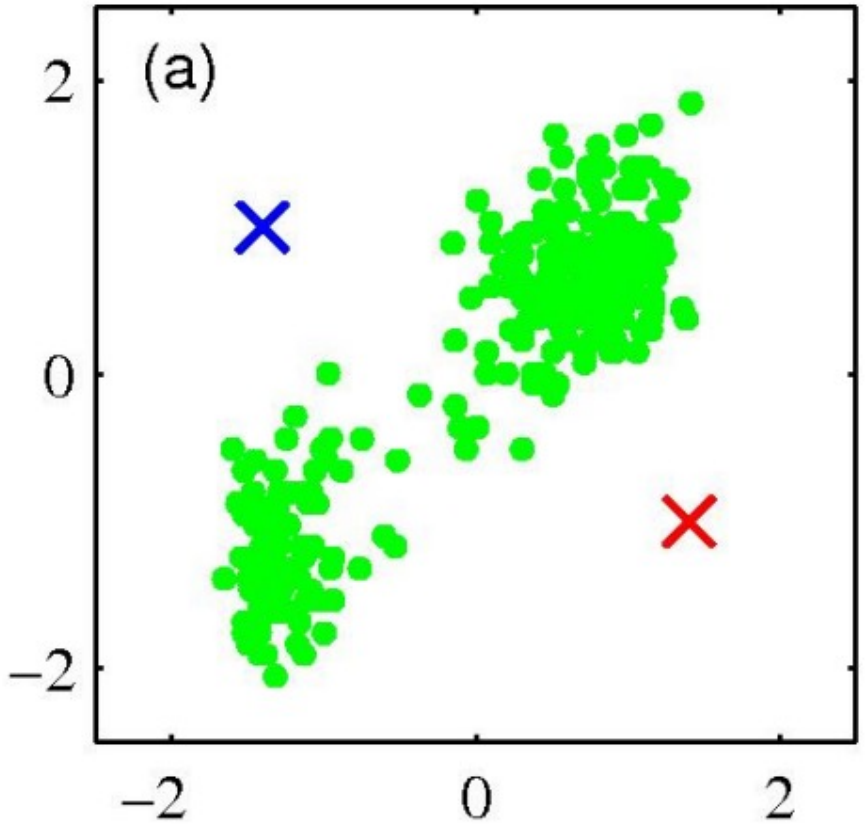
- Randomly choose k cluster center locations (centroids).
- Loop until convergence
  - Assign each point to the cluster of the closest centroid.
  - Reestimate the cluster centroids based on the data assigned to each.



K-means clustering algorithm

0. Initialize cluster centers

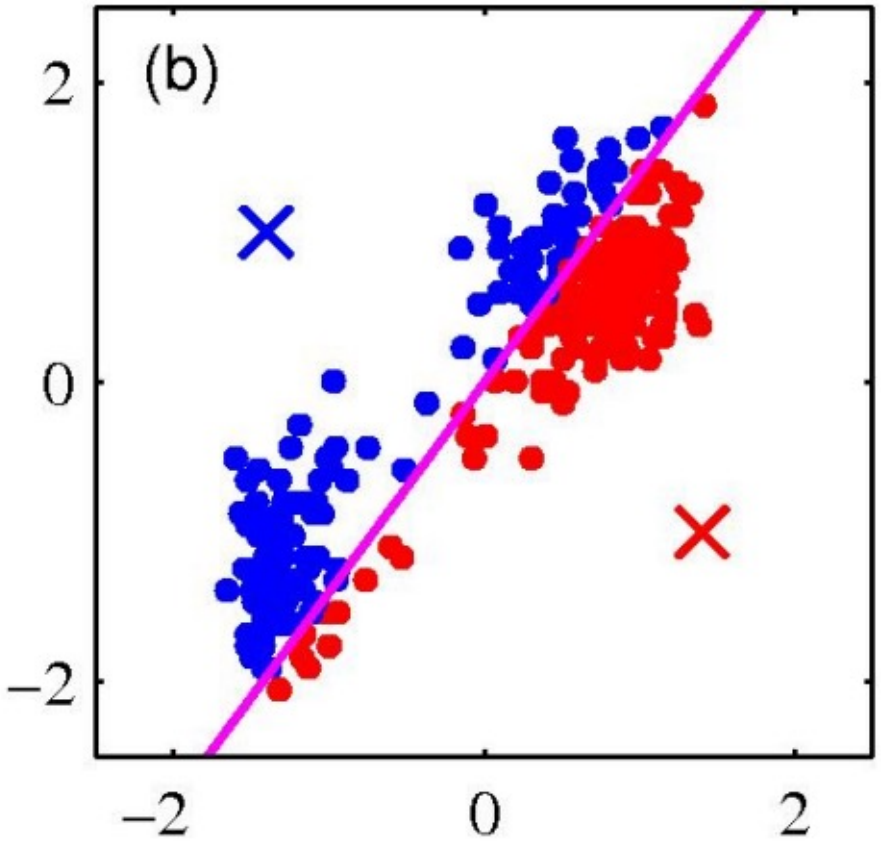
$$\mu_1, \mu_2, \ldots, \mu_k$$



K-means clustering algorithm

0. Initialize cluster centers
1. Assign observations to closest cluster center

$$z_i \leftarrow \arg\min_j ||\mu_j - \mathbf{x}_i||_2^2$$

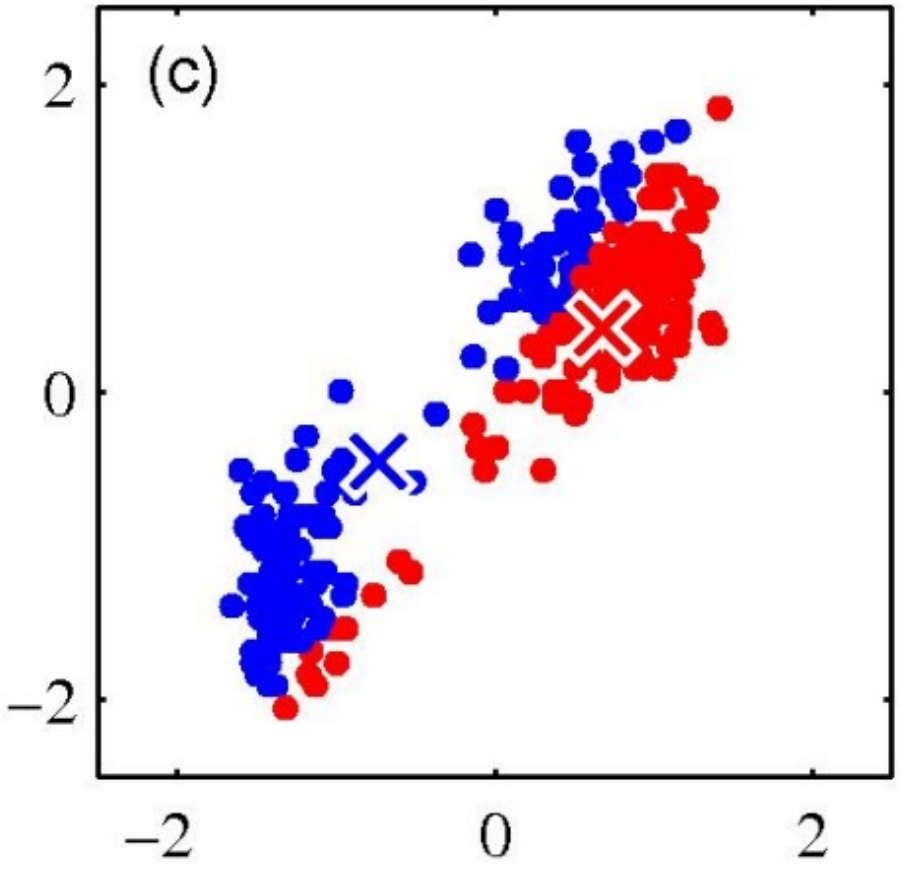


Slide adapted from Emily Fox & Carlos Guestrin.

K-means clustering algorithm

0. Initialize cluster centers
1. Assign observations to closest cluster center
2. Revise cluster centers as mean of assigned observations

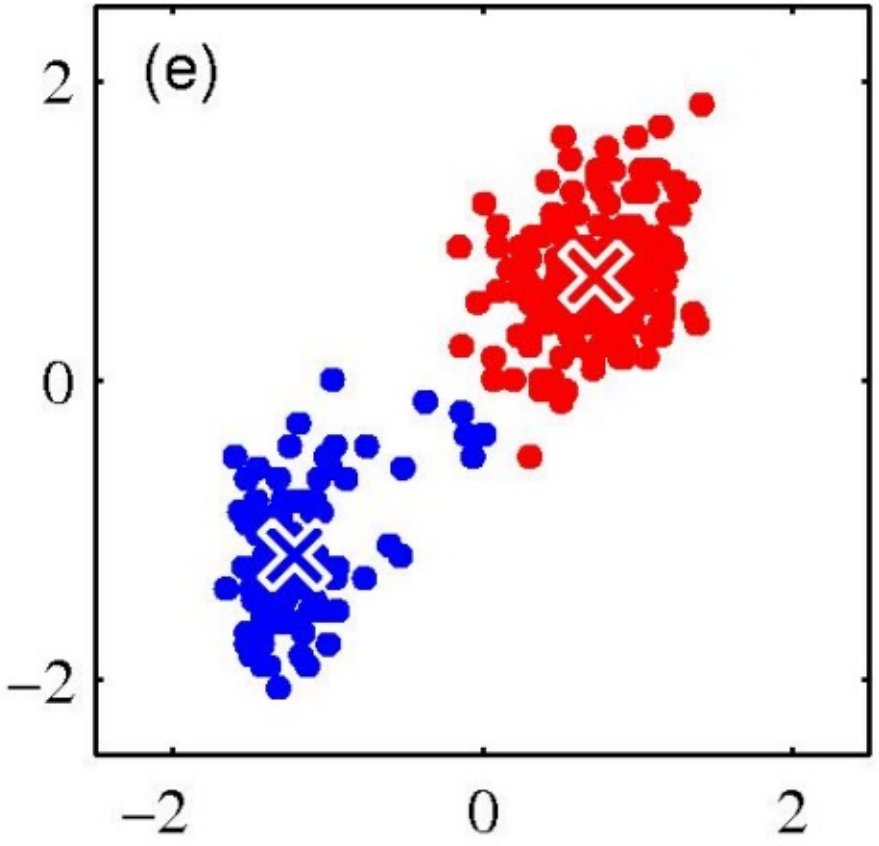
$$\mu_j = \frac{1}{n_j} \sum_{i:z_i=j} \mathbf{x}_i$$



Slide adapted from Emily Fox & Carlos Guestrin.

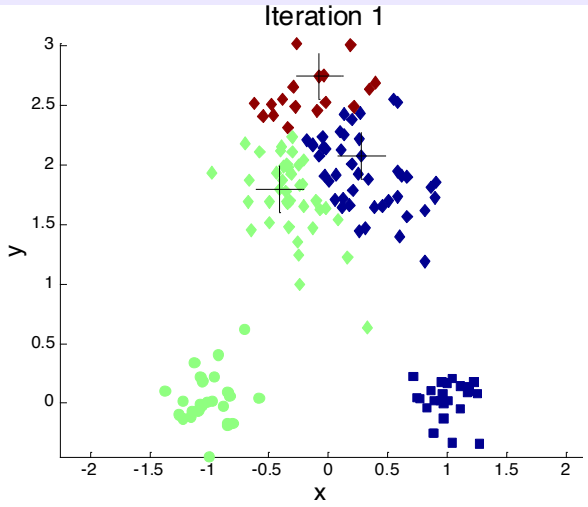
K-means clustering algorithm

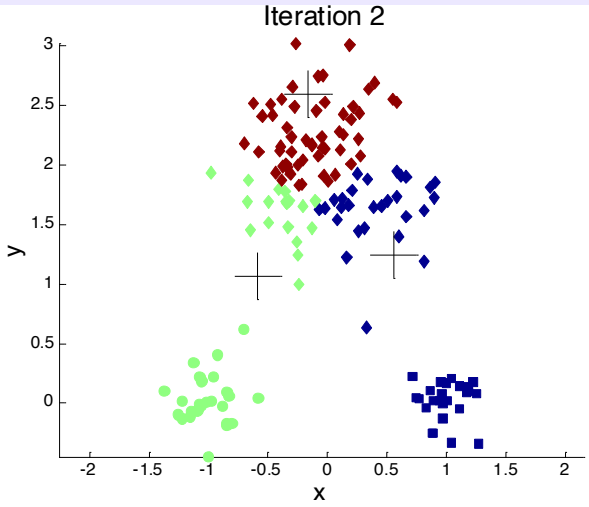
0. Initialize cluster centers
1. Assign observations to closest cluster center
2. Revise cluster centers as mean of assigned observations
3. Repeat 1.+2. until convergence

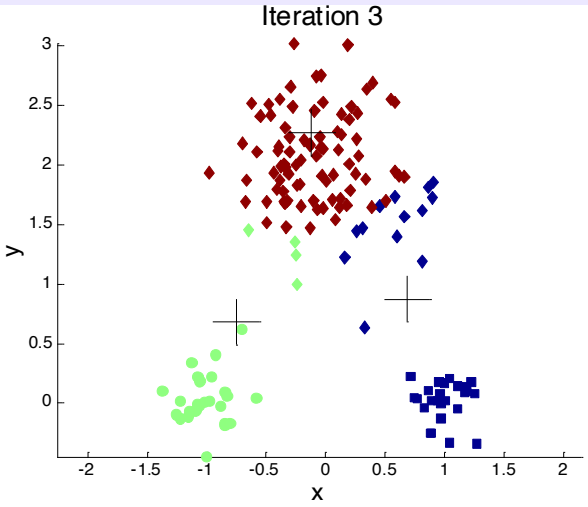


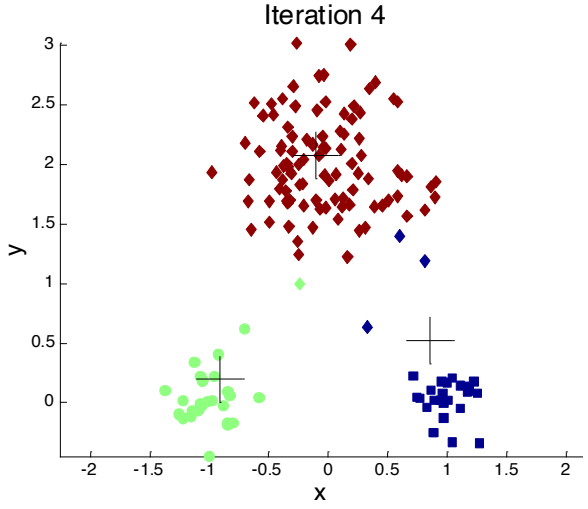
Slide adapted from Emily Fox & Carlos Guestrin.

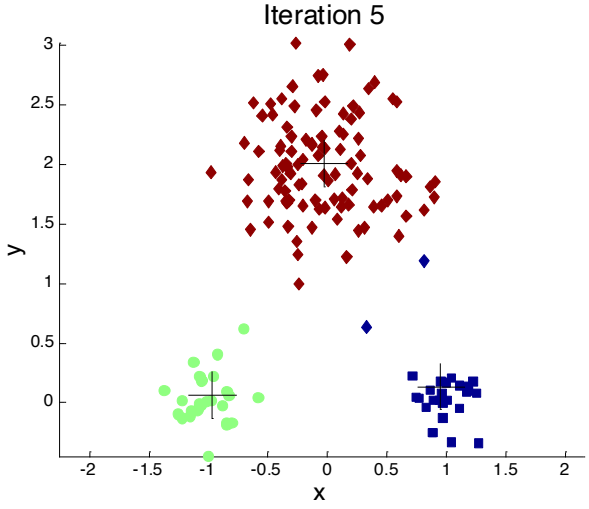
Impact of initial choice of centroids

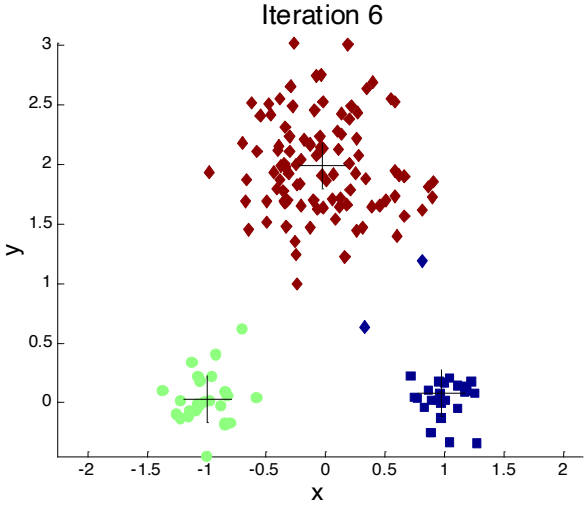






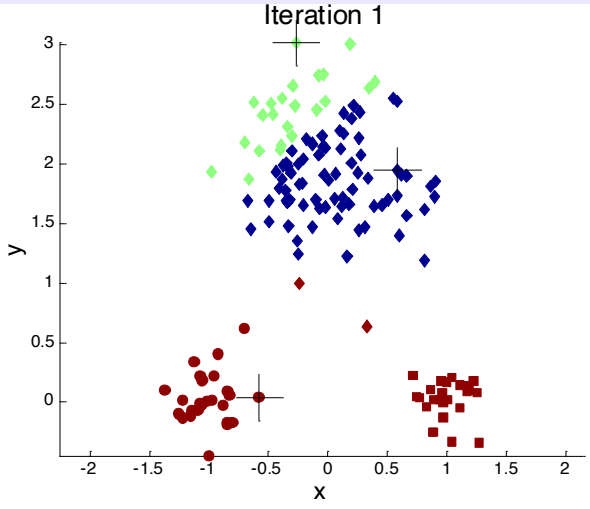


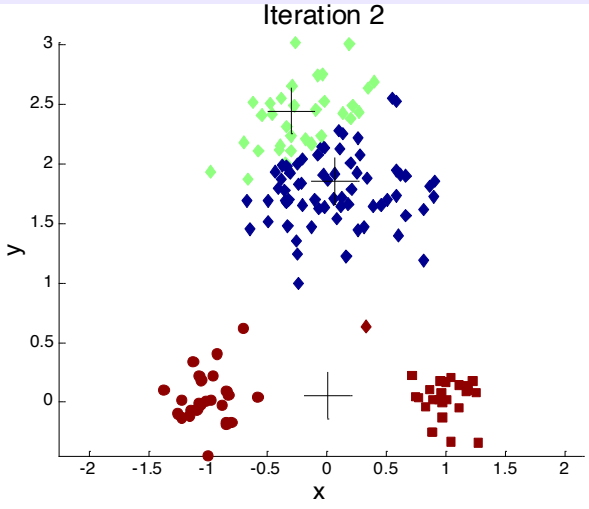


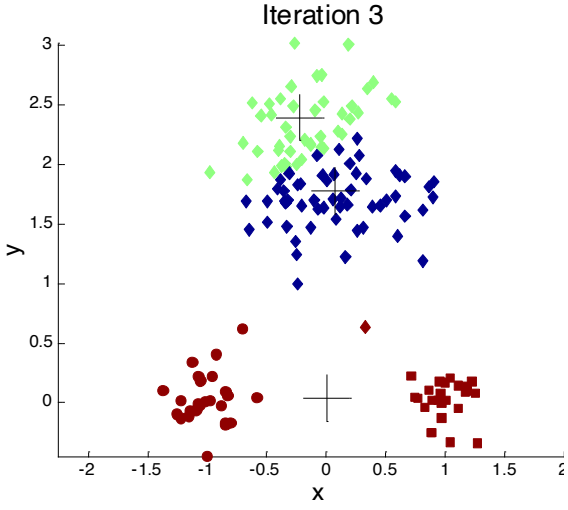


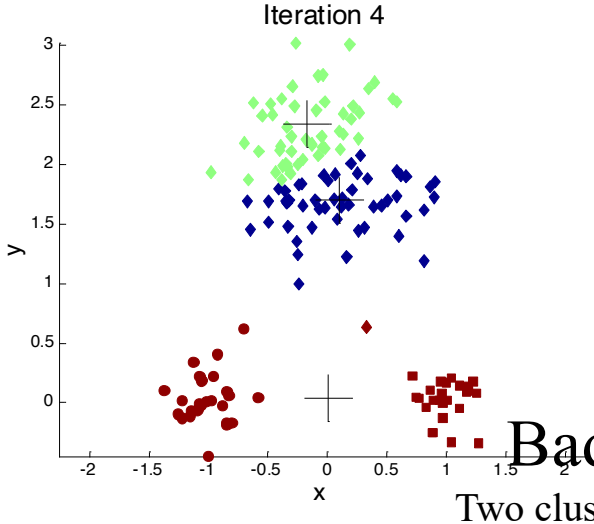
Good result

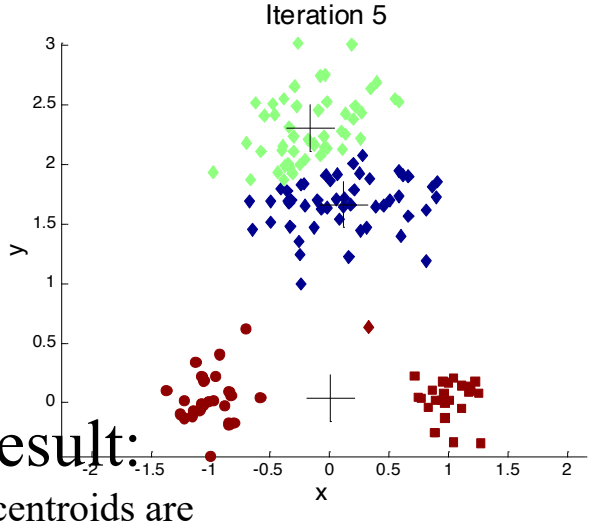
Impact of initial choice of centroids











Bad result:

Two cluster centroids are assigned to the same cluster and one cluster centroid represents two clusters

Problems with K-Means

Sensitive to the initial points

- Do many runs of k-Means, each with different initial centroids.
- Seed the centroids using a better method than random. (select farther centroids to begin with : K-Means++)

Must manually choose k

- Sometimes known (say assigning to known groups/categories…)
- Think of over-clustering
- Use distortion, or inertia measures

Hard assignments

- Use Gaussian Mixture Model with soft assignments

K-means has problems when clusters are of differing…

- Sizes
- Densities
- Non-spherical shapes (it just looks at the centroid and not the shape of the distribution)

Gaussian Mixture Models – Summary

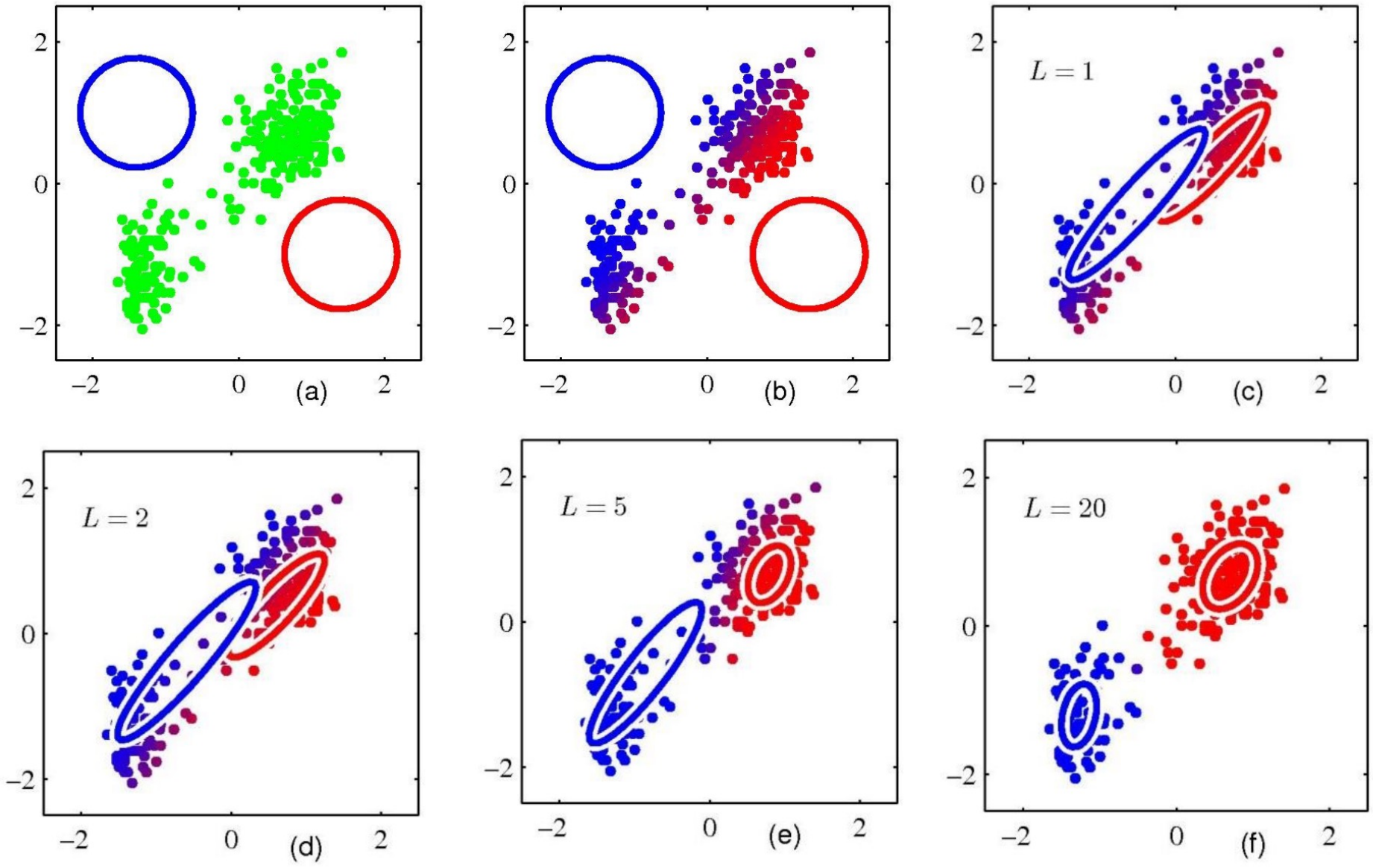
Initialization: We not only decide on the K cluster centroids, but also on the shape of the distributions

- assuming Normal distribution, we use mean and covariance matrices

Soft assignment: We don't just assign a data point (observation) to one cluster with a hard assignment, but assign them with soft assignments that indicate the posterior probability of the cluster given the observation

Update: We update both the center and shape of the distribution knowing the observations assigned to that centroid

Gaussian Mixture Models – Visual Expl.

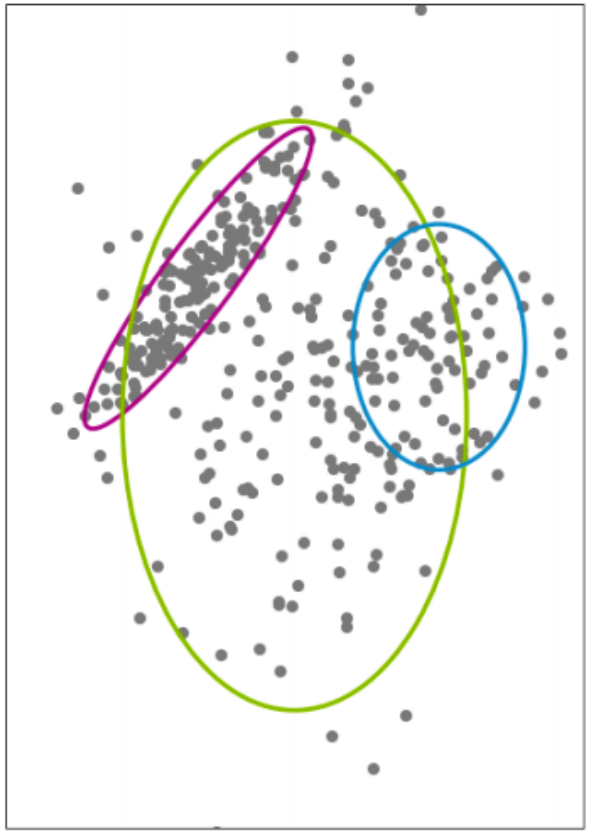


[Bishop]

Cluster responsibilities are easy to compute if parameters are known (π_i, μ_i, Σ_i)

$$p(x) = \sum_{i=0}^{K} \pi_i N(x \mid \mu_i, \Sigma_i)$$

$$0 \le \pi_i \le 1 \,, \quad \sum_{i=1}^{k} \pi_i = 1$$



r_ik = P(Component k) x p(x | Component k) / p(x)

Responsibility cluster k takes for observation i

$$r_{ik} = \frac{\pi_k \; N(x_i \mid \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_i \mid \mu_j, \Sigma_j)}$$

Normalized over all possible cluster assignments

Expectation Maximization

Motivates an iterative algorithm:

1. **E-step:** estimate cluster responsibilities given current parameter estimates

$$\hat{r}_{ik} = \frac{\hat{\pi}_k N(x_i \mid \hat{\mu}_k, \hat{\Sigma}_k)}{\sum_{j=1}^{K} \hat{\pi}_j N(x_i \mid \hat{\mu}_j, \hat{\Sigma}_j)}$$

2. **M-step:** maximize likelihood over parameters given current responsibilities

$$\hat{\pi}_k, \hat{\mu}_k, \hat{\Sigma}_k \mid \{\hat{r}_{ik}, x_i\}$$

R	G	B	r_{i1}	r_{i2}	r_{i3}
$x_1[1]$	$x_1[2]$	$x_1[3]$	0.30	0.18	0.52
$x_2[1]$	$x_2[2]$	$x_2[3]$	0.01	0.26	0.73
$x_3[1]$	$x_3[2]$	$x_3[3]$	0.002	0.008	0.99
$x_4[1]$	$x_4[2]$	$x_4[3]$	0.75	0.10	0.15
$x_5[1]$	$x_5[2]$	$x_5[3]$	0.05	0.93	0.02
$x_6[1]$	$x_6[2]$	$x_6[3]$	0.13	0.86	0.01
Total weight in cluster: (effective # of obs)			**1.242**	**2.8**	**2.42**

52% chance this obs is in cluster 3

Maximum Likelihood Estimates of the Unknown Parameters with soft assignments

$$\hat{\pi}_k = \frac{N_k^{\text{soft}}}{N}$$

$$N_k^{\text{soft}} = \sum_{i=1}^{N} r_{ik}$$

if $\{0,1\}$ just count obs i in cluster k if $r_{ik}=1$ $= N_k$ ✓

$$\hat{\mu}_k = \frac{1}{N_k^{\text{soft}}} \sum_{i=1}^{N} r_{ik} x_i$$

only add x_i if i in k $(r_{ik}=1)$

$\rightarrow \frac{1}{N_k} \sum_{i \text{ in } k} x_i$ ✓

$$\hat{\Sigma}_k = \frac{1}{N_k^{\text{soft}}} \sum_{i=1}^{N} r_{ik} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$$

✓

same as above

$= \frac{1}{N_k} \sum_{i \text{ in } k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$