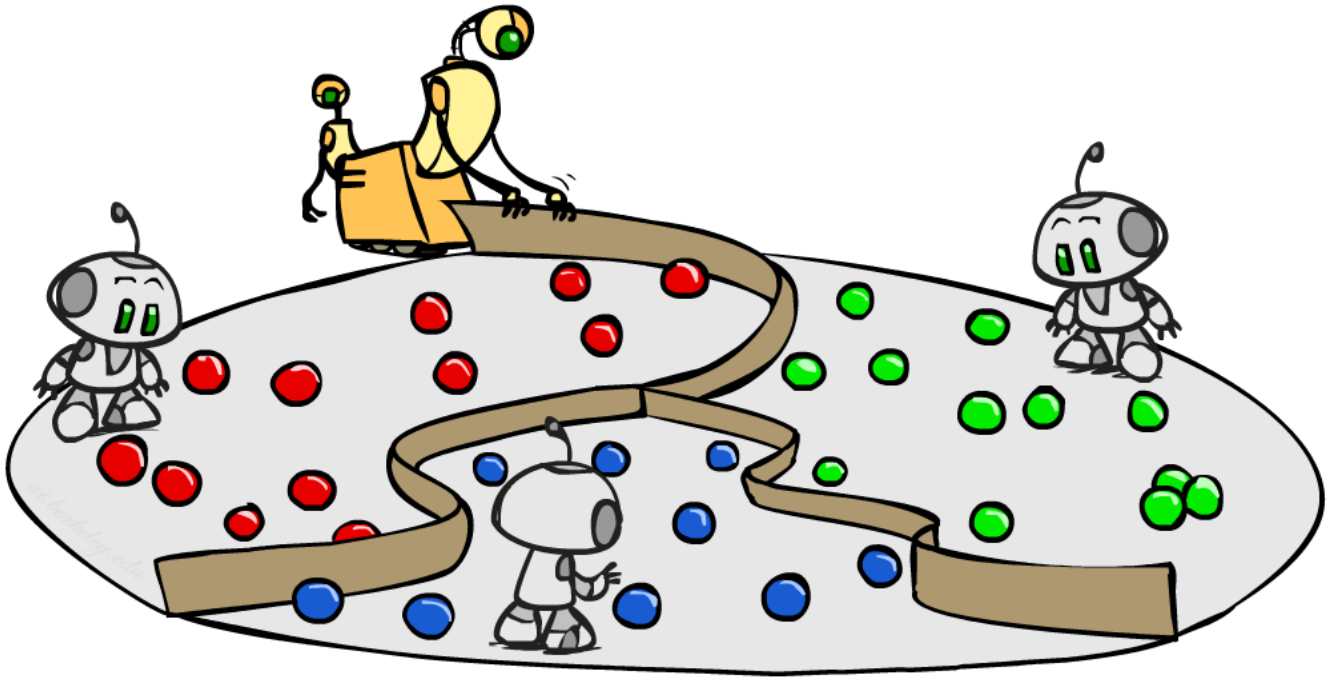
Unsupervised Machine Learning



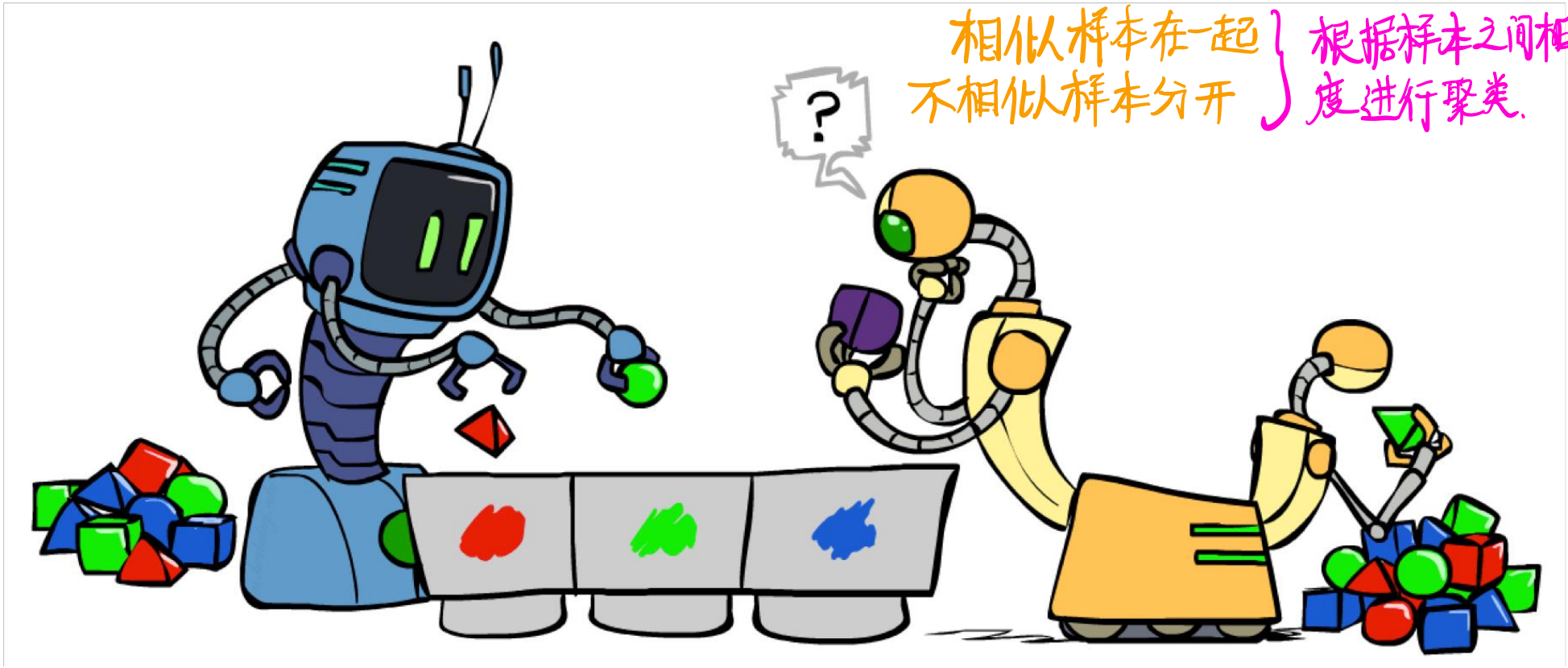
AIMA Chapter 20

[Adapted from slides by Dan Klein and Pieter Abbeel at UC Berkeley and from Daniel Weld, Carlos Guestrin, & Luke Zettlemoyer at U Washington]

Types of Learning

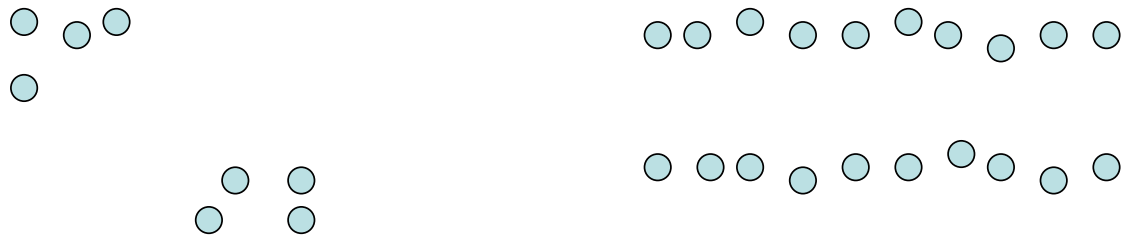
- Supervised learning
  - Training data includes desired outputs
- Unsupervised learning 
  - Training data does not include desired outputs
- Semi-supervised learning
  - Training data includes a few desired outputs
- Reinforcement learning
  - Rewards from sequence of actions

Clustering 聚类，数据组



Clustering

- Basic idea: group together similar instances
- Example: 2D point patterns



- What could "similar" mean?
  - One option: small (squared) Euclidean distance

$$\text{dist}(x, y) = (x - y)^{\top}(x - y) = \sum_i (x_i - y_i)^2$$

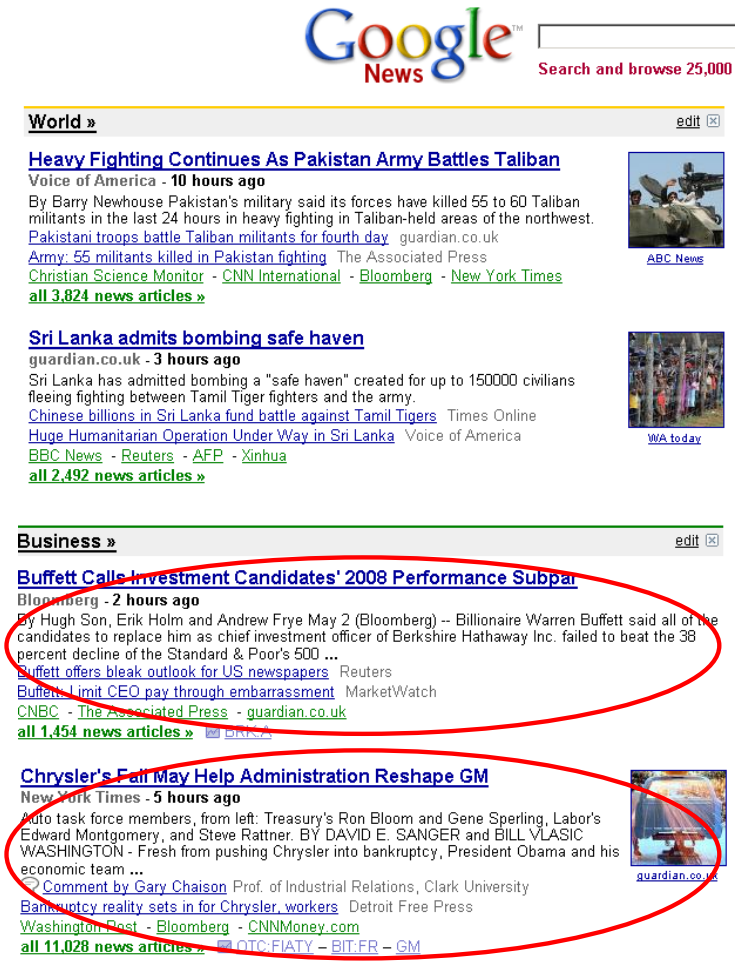
  - Many other options, often domain specific

样本点之间的相似度.

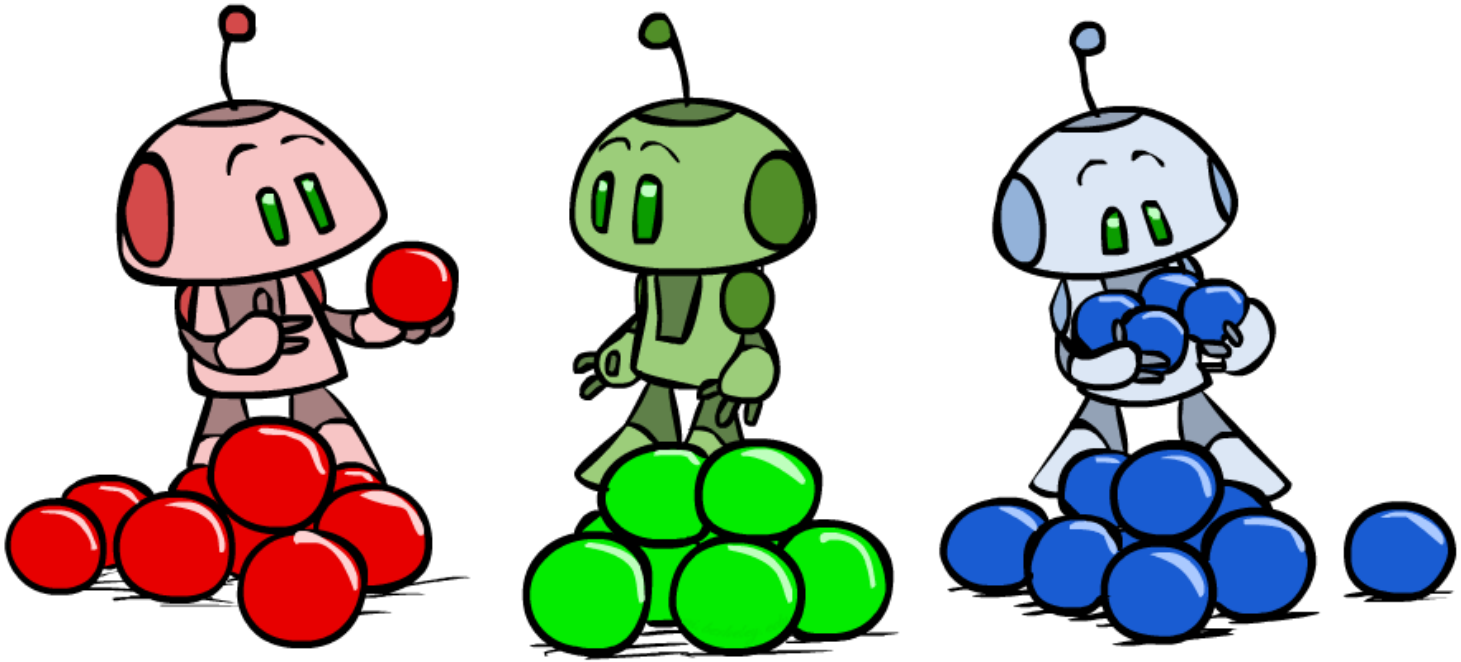
Clustering

- Applications
  - Group emails
  - Group search results
  - Find categories of customers
  - Detect anomalous program executions

Story groupings:
unsupervised clustering



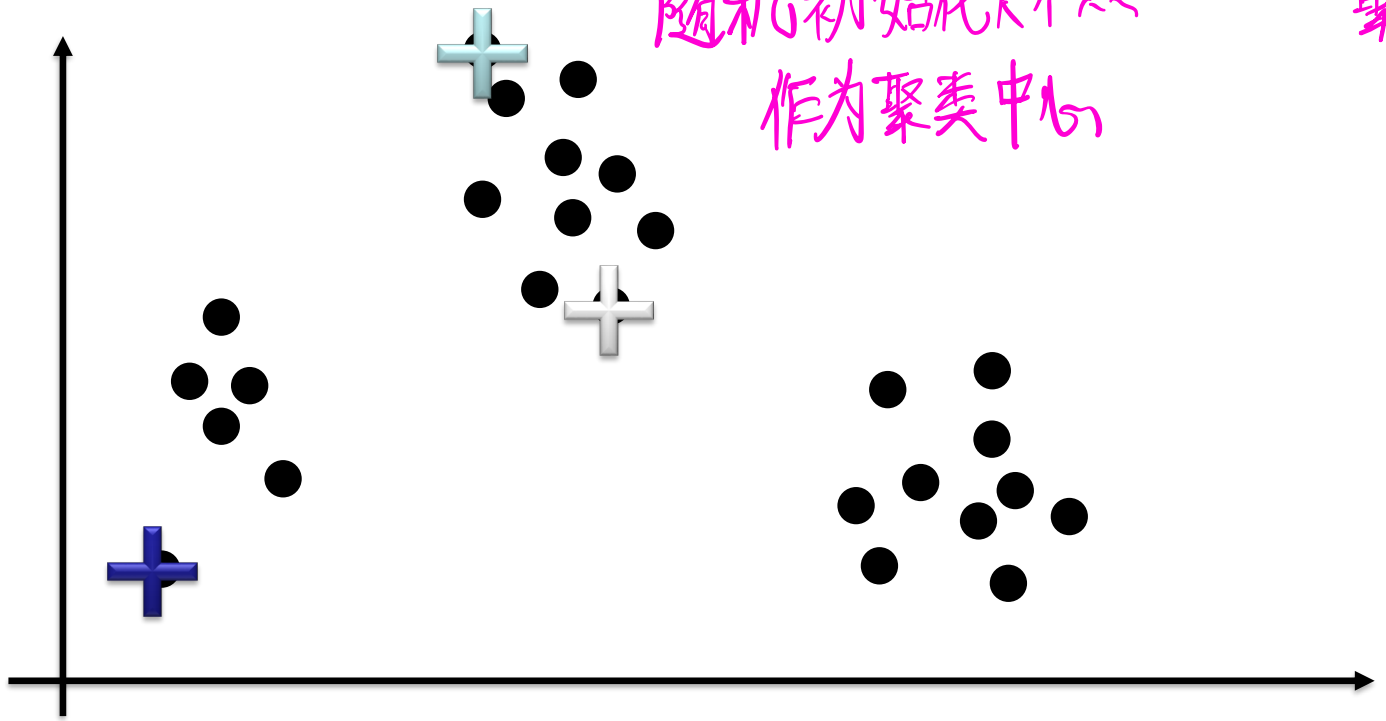
K-Means



K-Means Clustering: *Intuition*

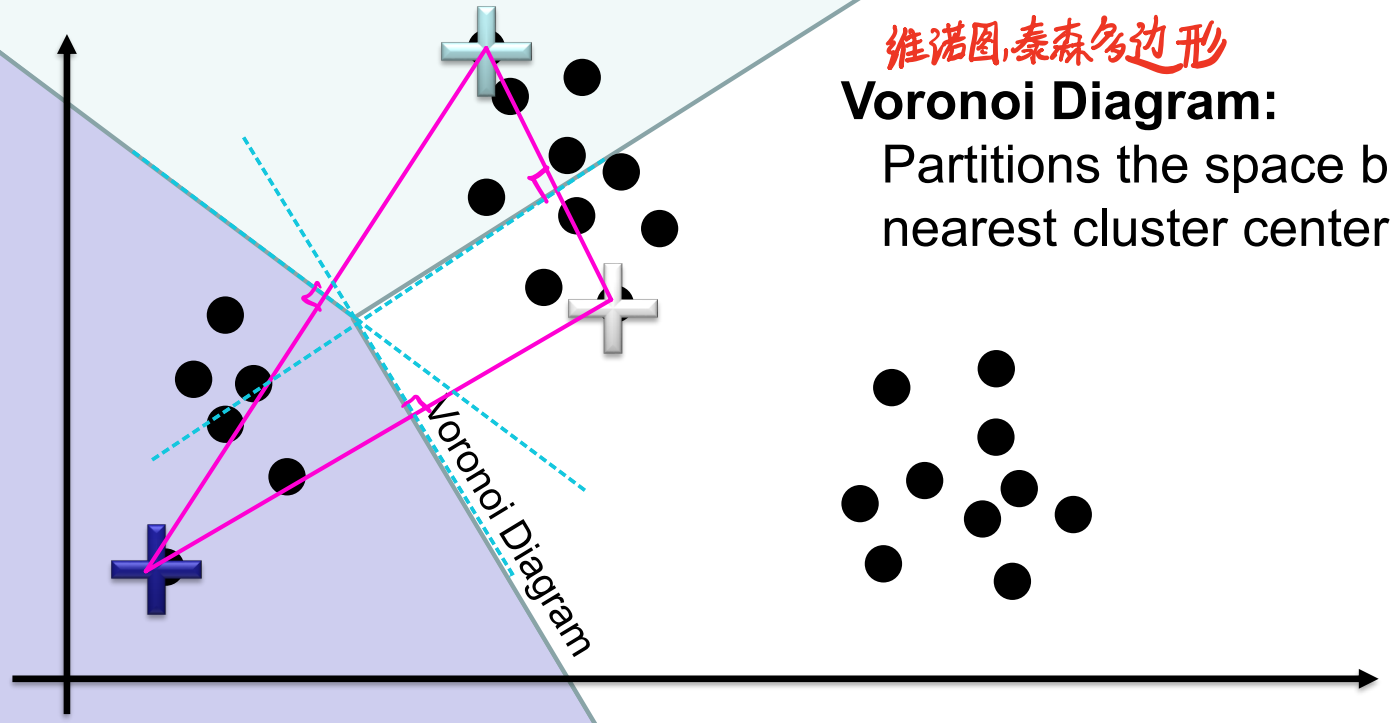
- Input K: The number of clusters to find (人为定义: K: 共of clusters)
- Pick an initial set of points as cluster centers 随机初始化K个点作为聚类中心

聚成K类



K-Means Clustering: *Intuition*

- For each data point find the cluster nearest center

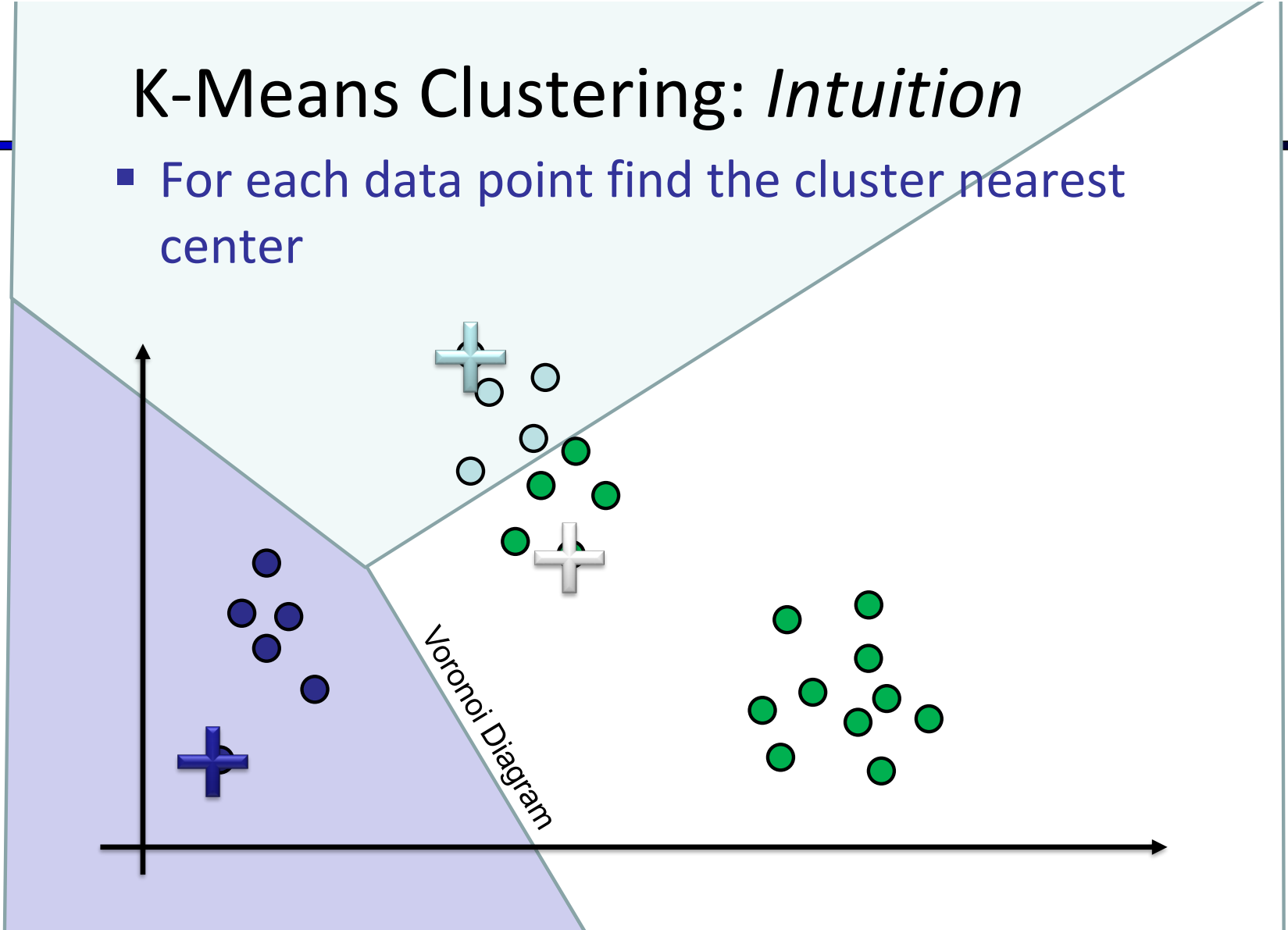


维诺图,泰森多边形

Voronoi Diagram:
Partitions the space by nearest cluster center

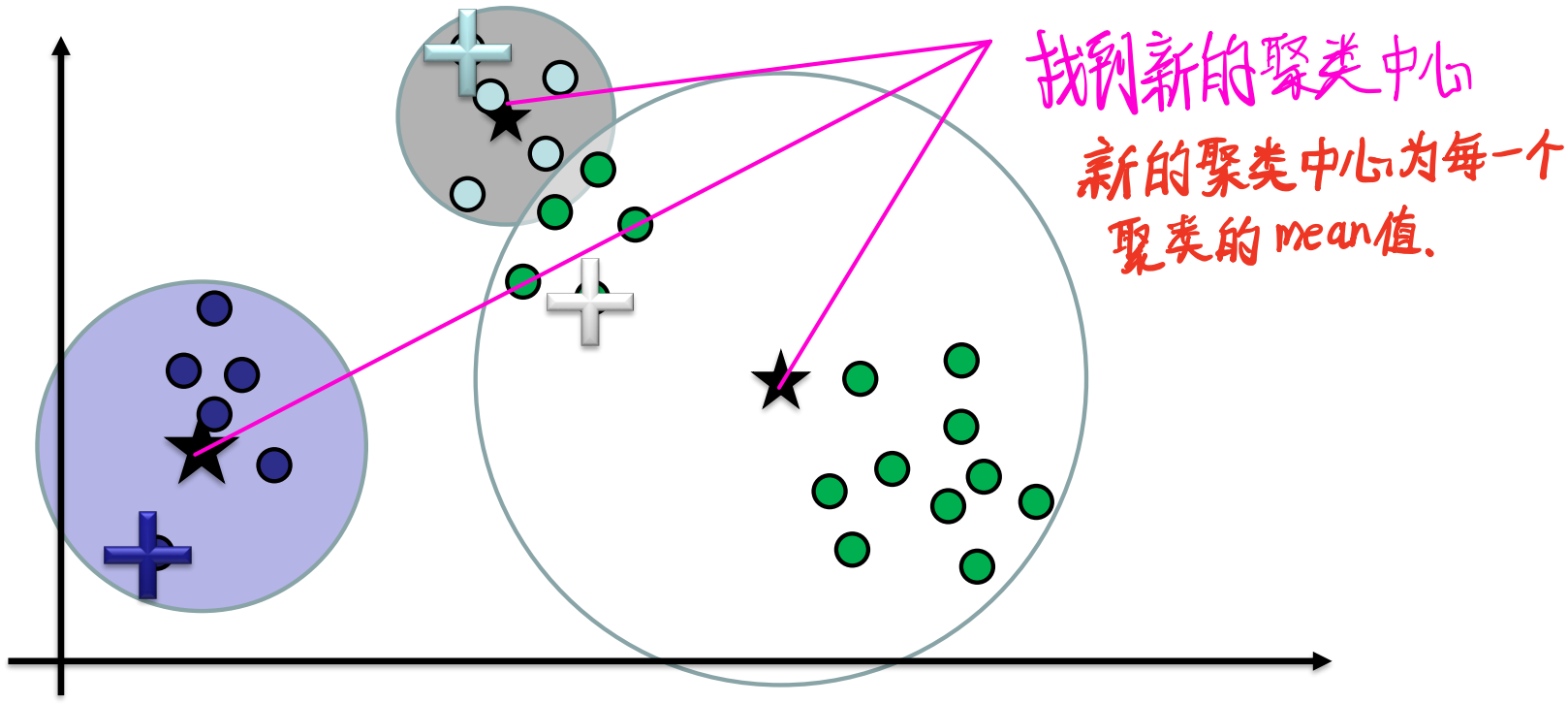
K-Means Clustering: *Intuition*

- For each data point find the cluster nearest center



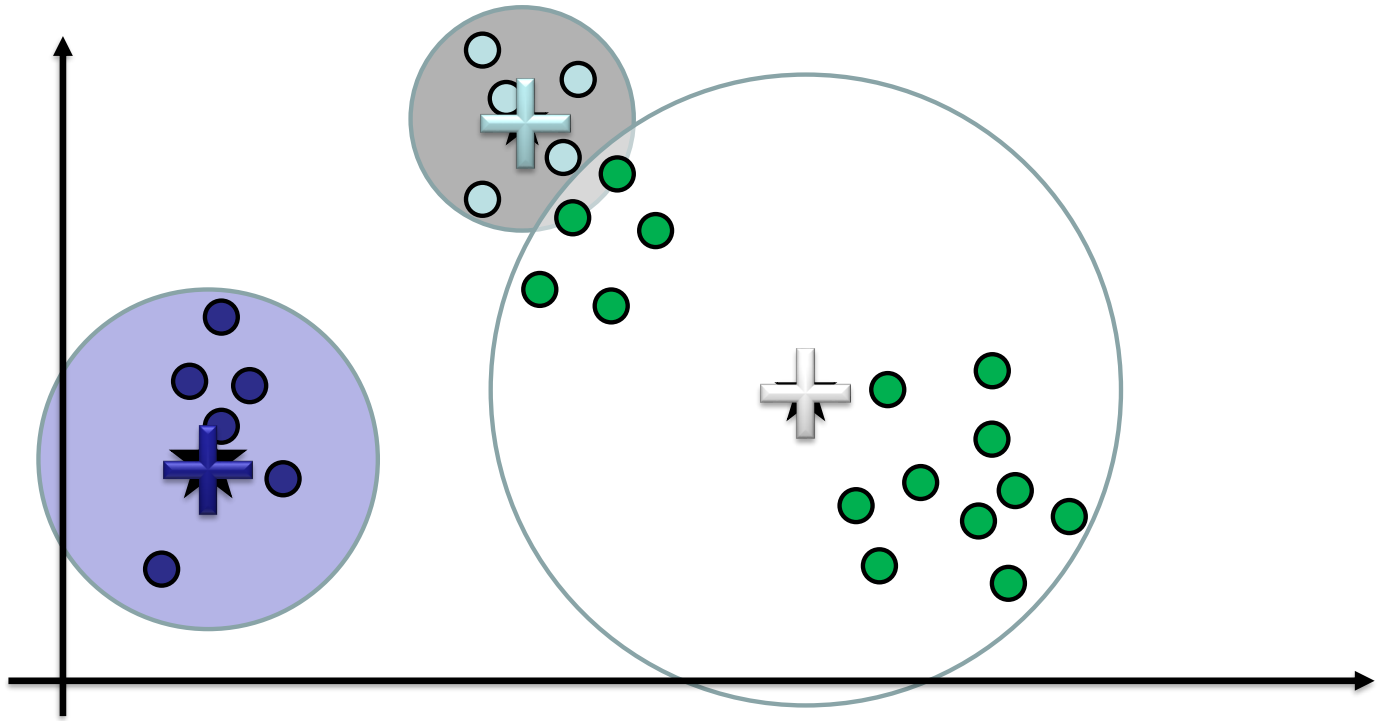
K-Means Clustering: *Intuition*

- Compute mean of points in each "cluster"



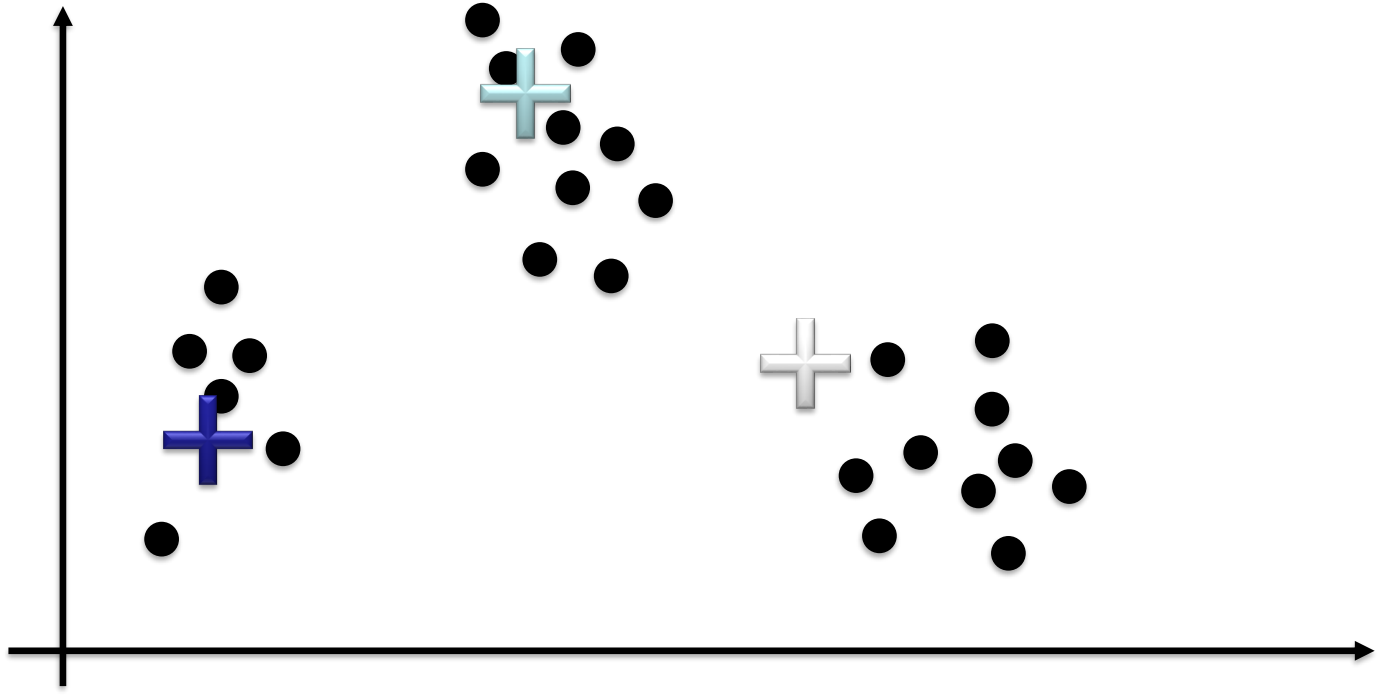
K-Means Clustering: *Intuition*

- Adjust cluster centers to be the mean of the cluster



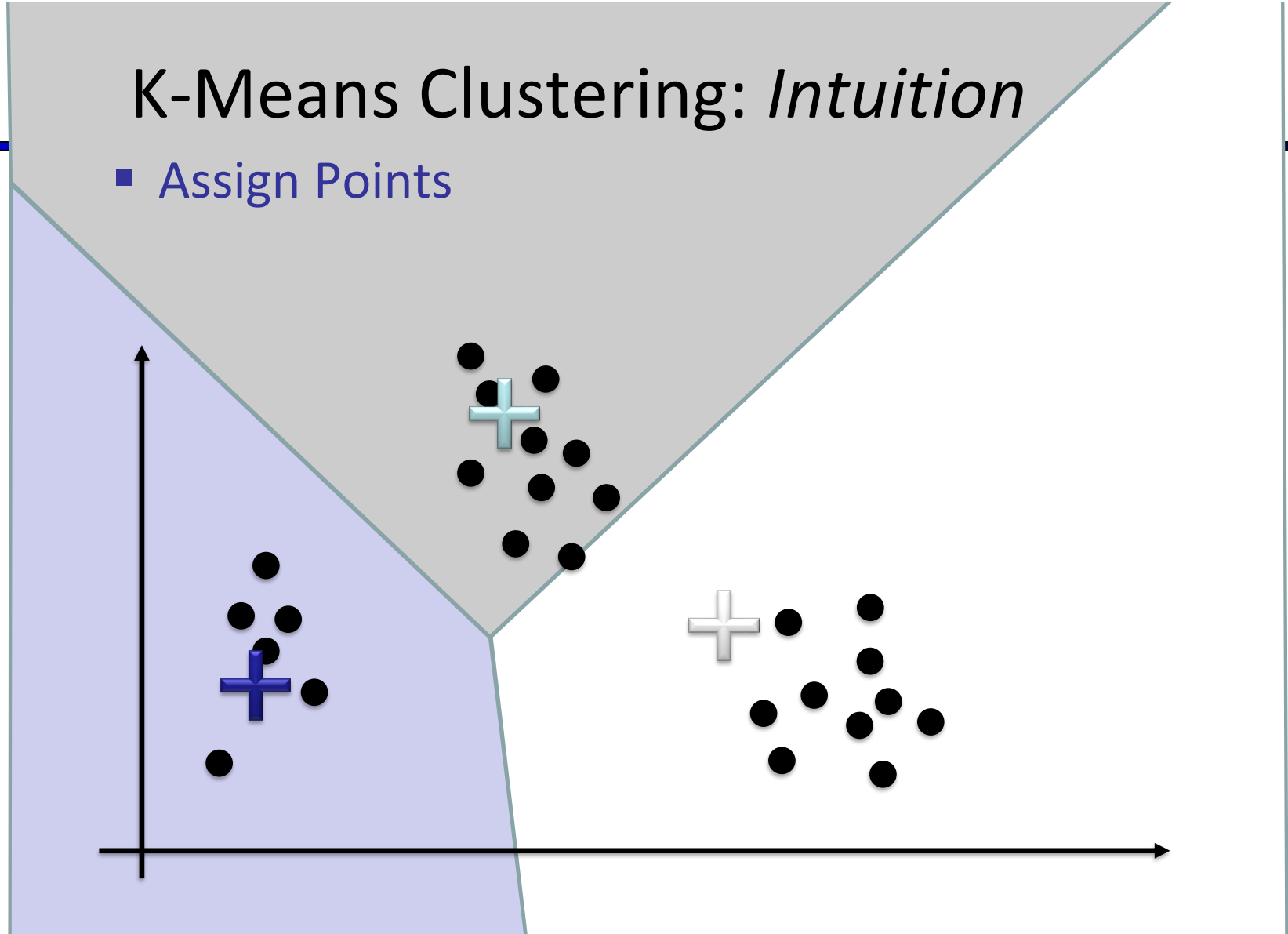
K-Means Clustering: *Intuition*

- Improved?
- Repeat



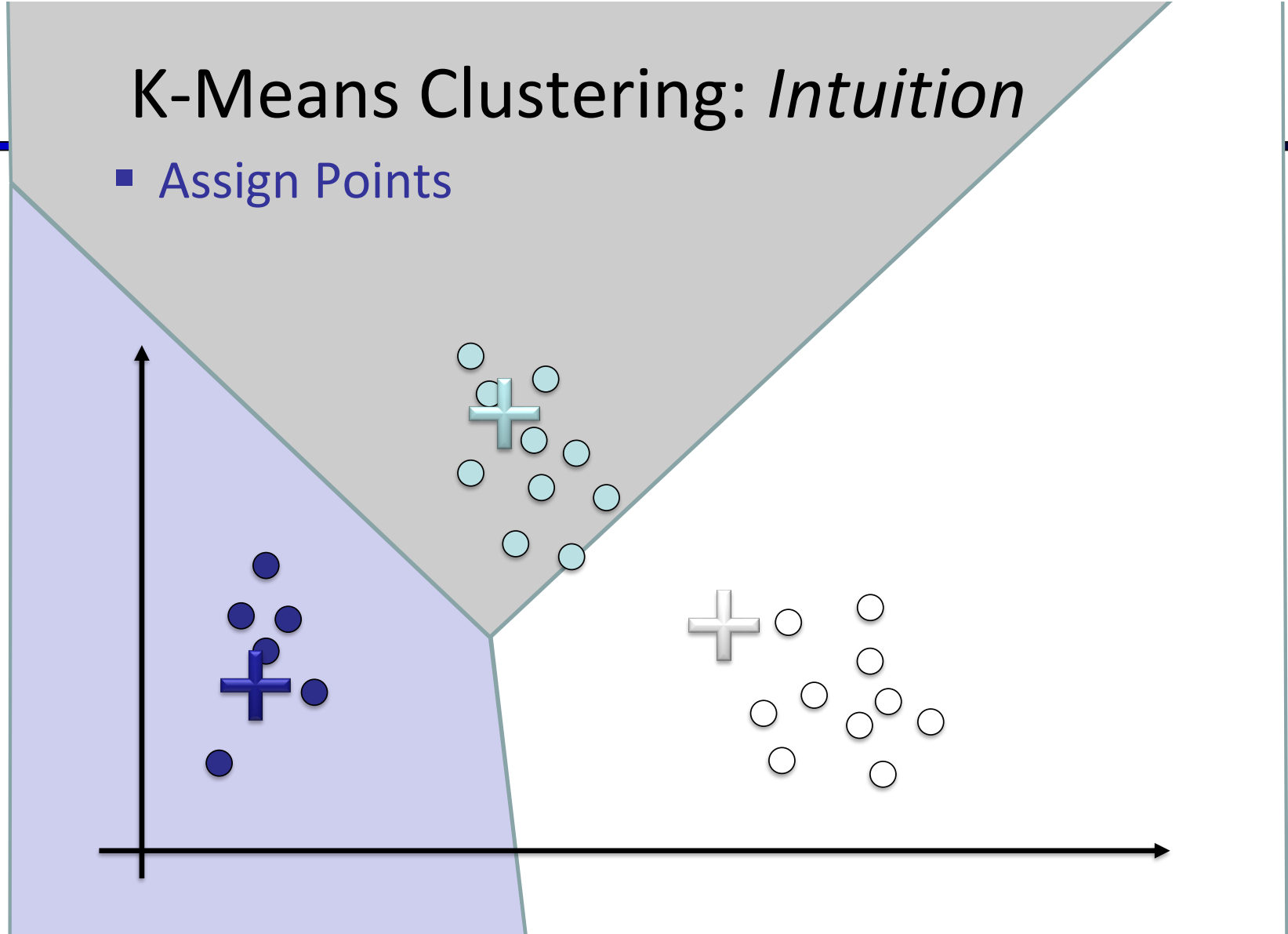
K-Means Clustering: *Intuition*

- Assign Points



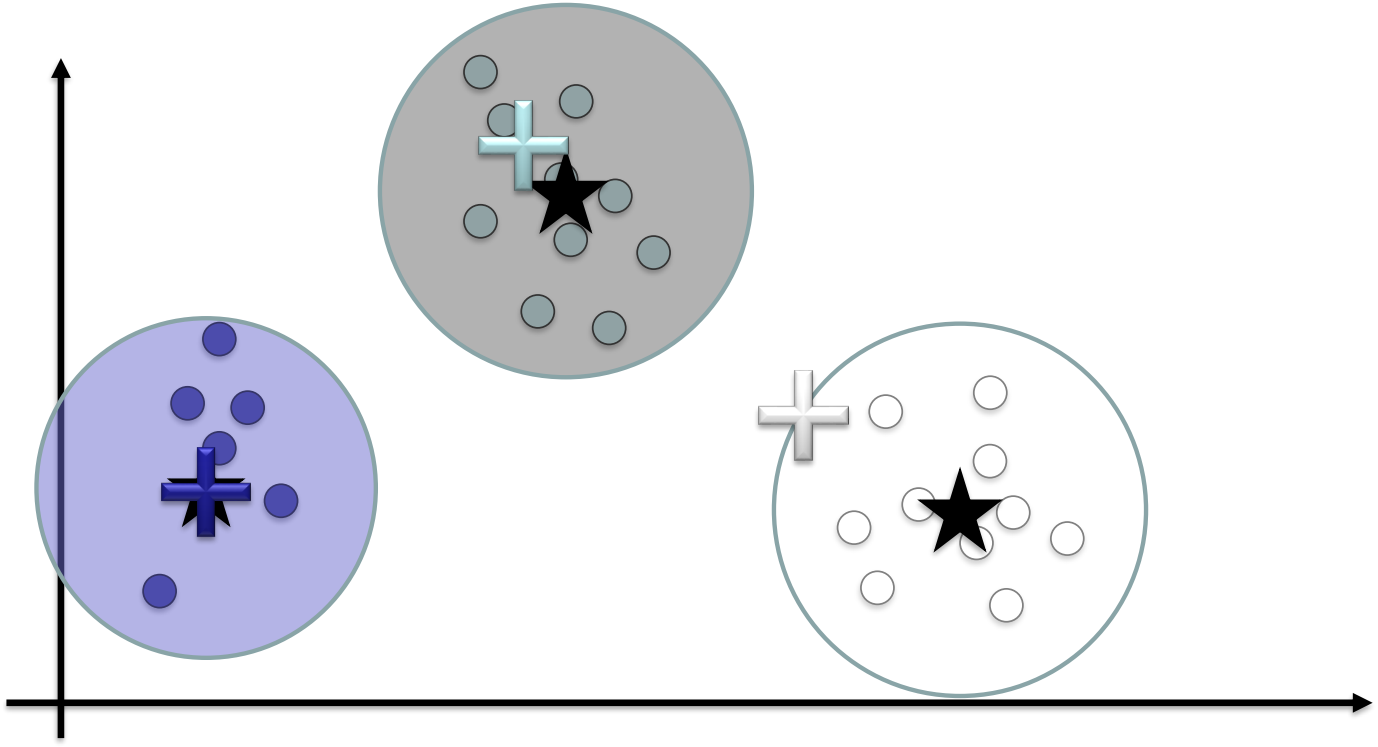
K-Means Clustering: *Intuition*

- Assign Points



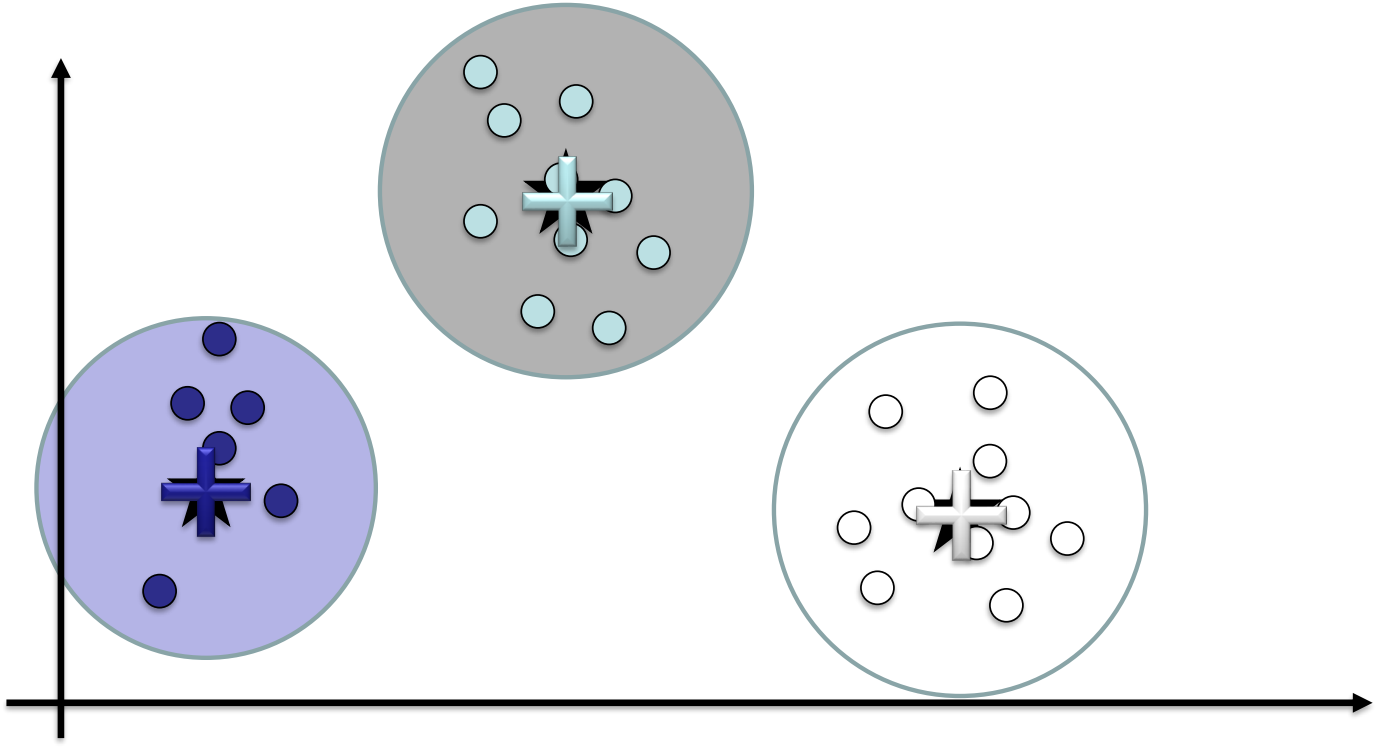
K-Means Clustering: *Intuition*

- Compute cluster means



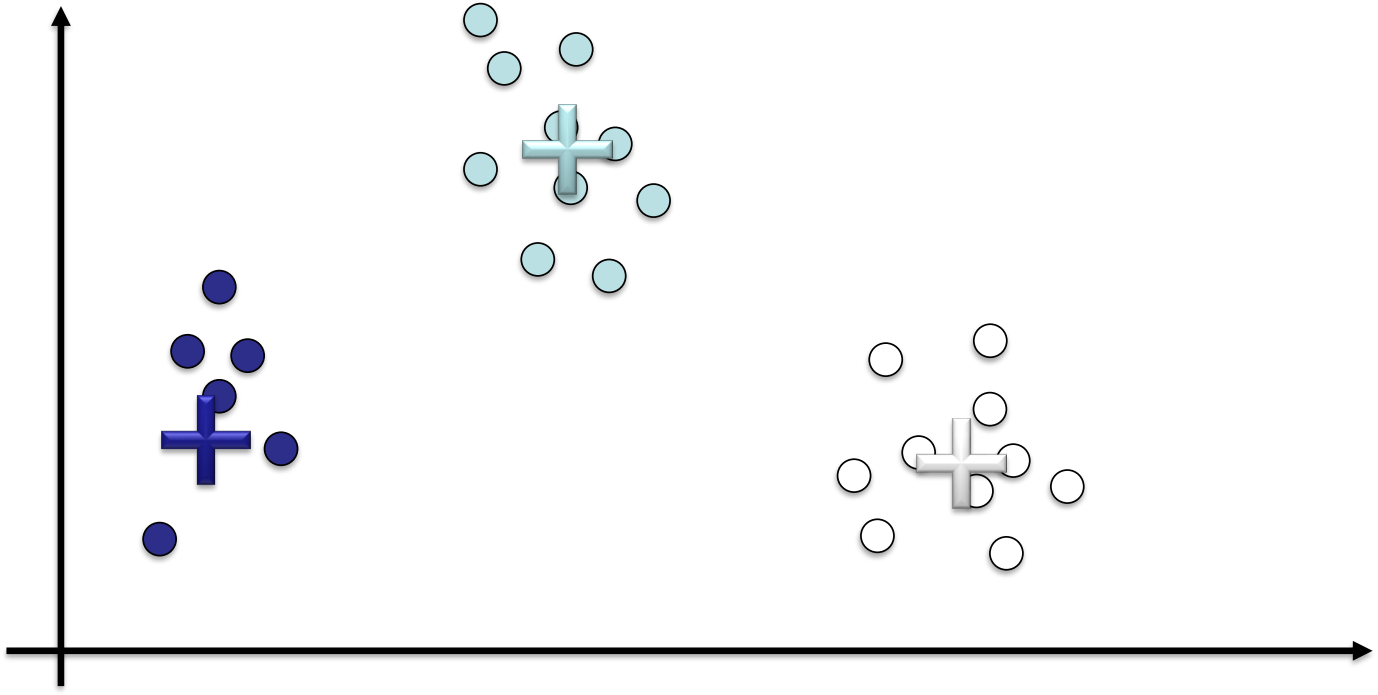
K-Means Clustering: *Intuition*

- Update cluster centers



K-Means Clustering: *Intuition*

- Repeat?
  - Yes to check that nothing changes → Converged!



K-Means as Optimization

- Consider the total distance to the means:

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \text{dist}(x_i, c_{a_i})$$

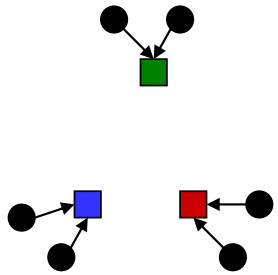
聚类中心

points: $\{x_i\}$ assignments: $\{a_i\}$ means: $\{c_k\}$

squared Euclidean distance

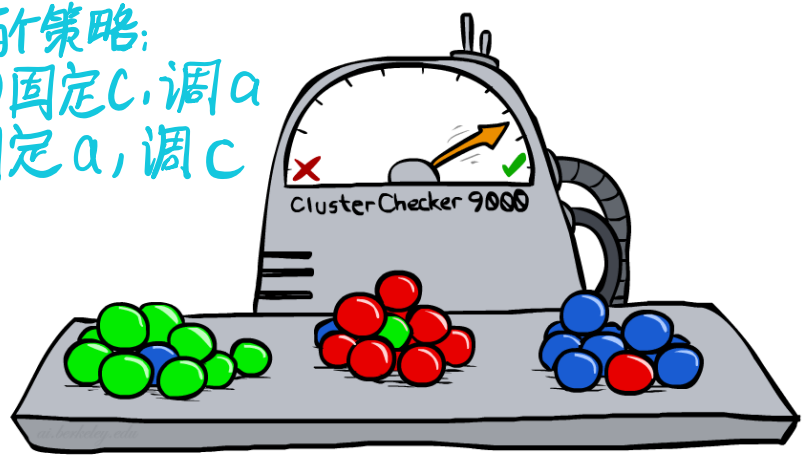
x_i被分配到了a_i这个群中，c_{a_i}为

st ↑最小



- Two stages each iteration: a_i群的mean.
  - Update assignments: fix means c, change assignments a
  - Update means: fix assignments a, change means c
- Each step cannot increase phi

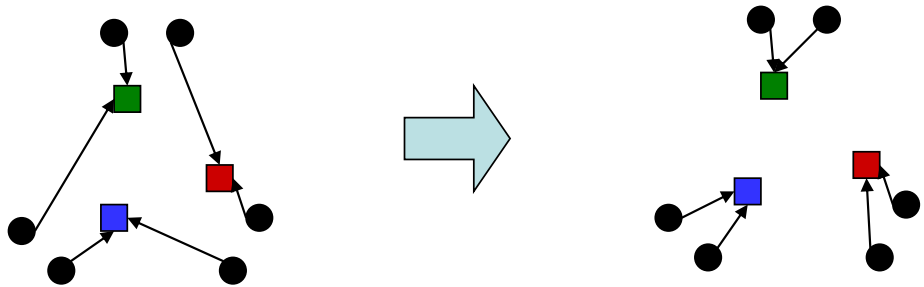
两个策略:
①固定c，调a
②固定a，调c



Phase I: Update Assignments

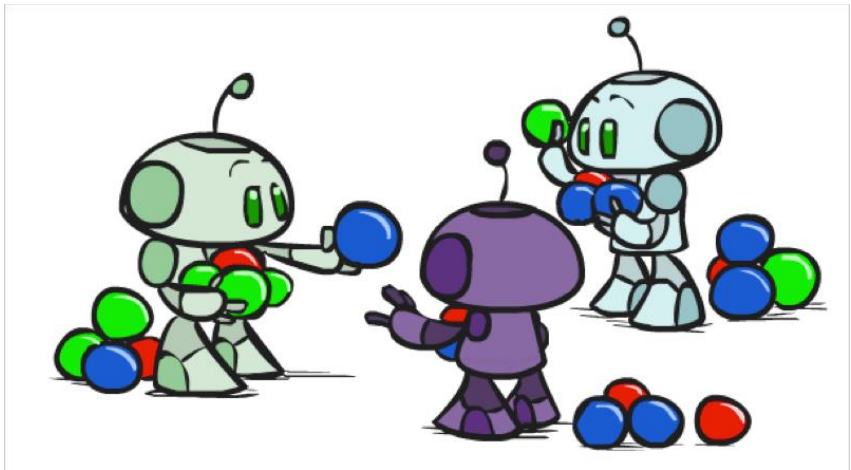
- For each point, re-assign to closest mean:

$$a_i = \underset{k}{\text{argmin}}\, \text{dist}(x_i, c_k)$$



- Cannot increase total distance phi!

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \text{dist}(x_i, c_{a_i})$$

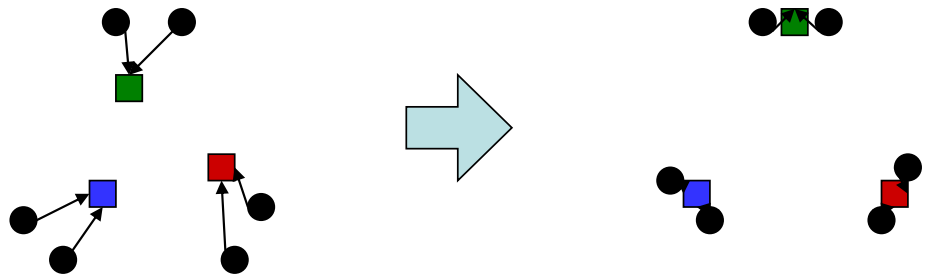


Phase II: Update Means

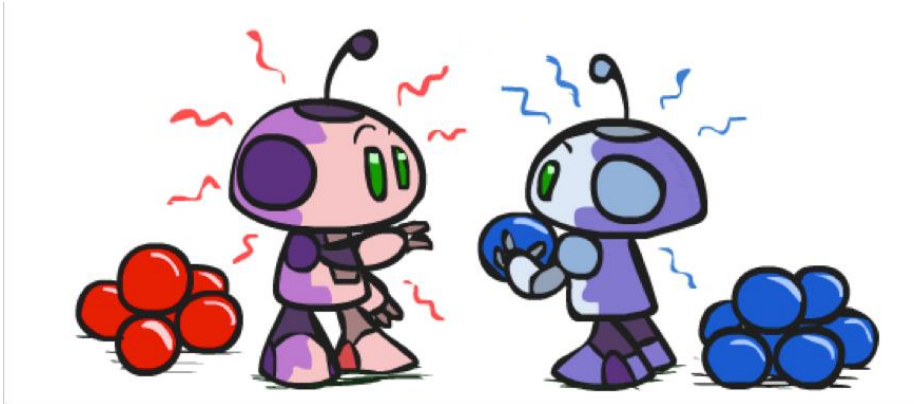
K means 一定会收敛　　调整 Means

- Move each mean to the average of its assigned points:

$$c_k = \frac{1}{|\{i : a_i = k\}|} \sum_{i:a_i=k} x_i$$

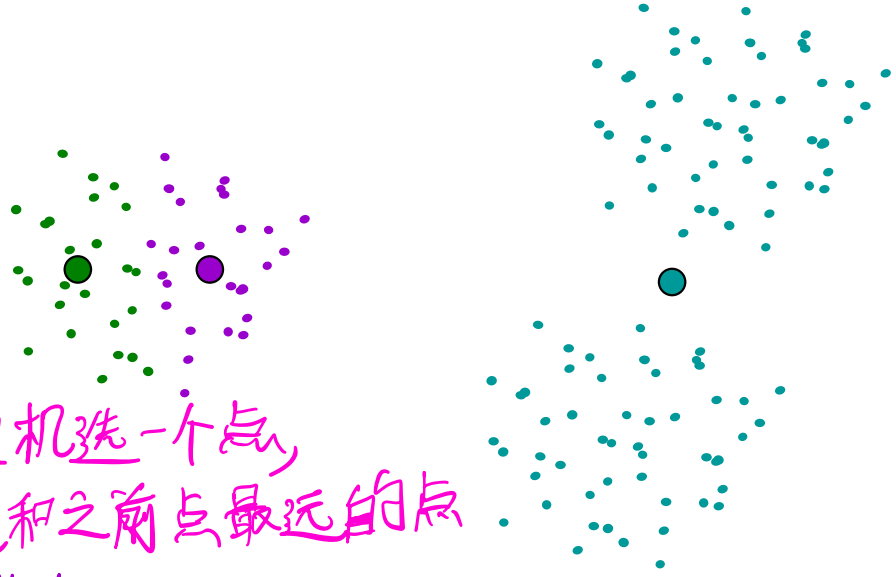


- Also cannot increase total distance
  - Fun fact: the point y with minimum squared Euclidean distance to a set of points {x} is their mean

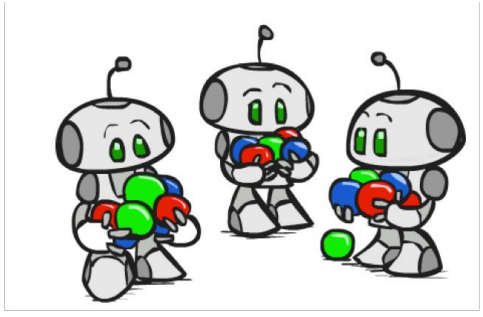


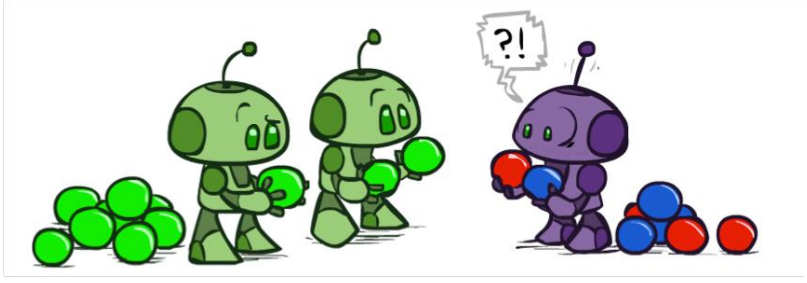
Initialization

- K-means is non-deterministic
  - Requires initial means
  - It does matter what you pick!
  - What can go wrong?
    - Local optima



最远点采样：随机选一个点，
第二点选和之前点最远的点
（问题：噪声的影响）

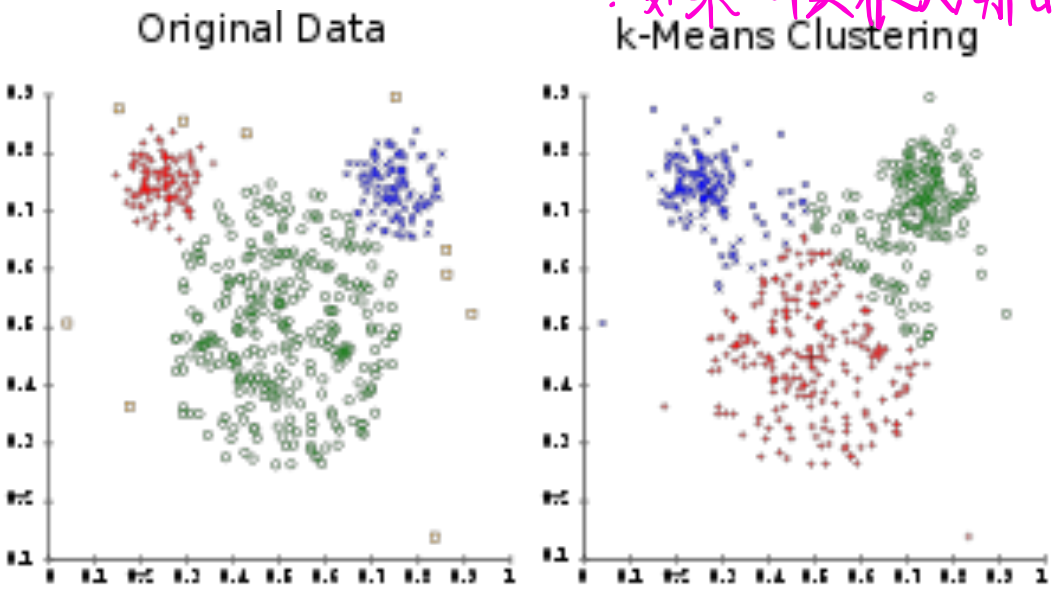




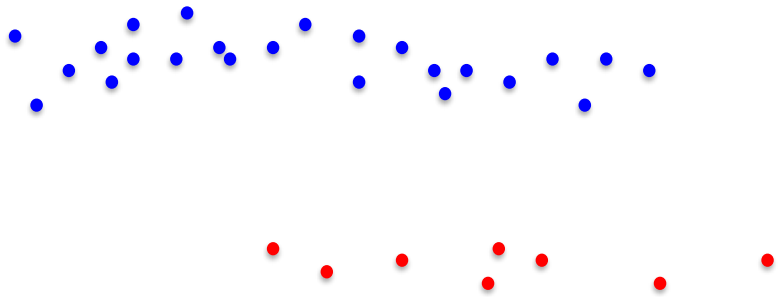
Inductive Bias

偏向偏置：(1)预测不同类簇的大不相似 (2)偏向于预测圆形的类簇

∵如果一个类很大，那distance就大

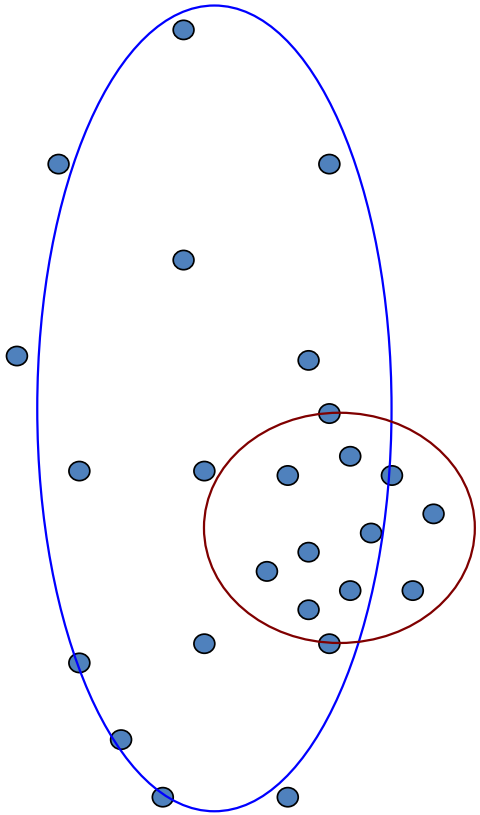


Equally Sized Clusters



Circular Clusters

Problems with k-means



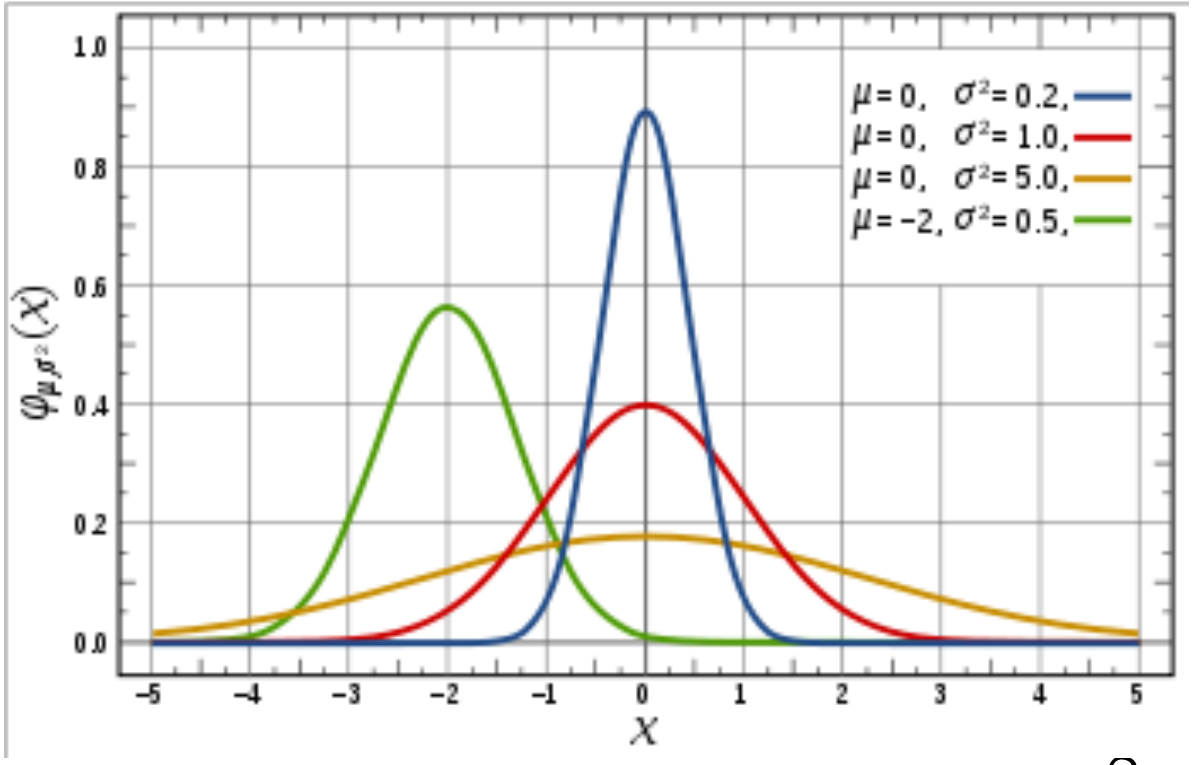
- Assigning data to closest centers
  - But some clusters may be "wider" than others
  - Distances can be deceiving!
- Hard Assignments
  - But clusters may overlap

Probabilistic Clustering

- Try a probabilistic model!
  - allows overlaps, clusters of different sizes/shapes, etc.

- Gaussian mixture model (GMM) 高斯混合模型.
  - also called Mixture of Gaussians

Review: Gaussians



$$P(x \mid \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

Learning Gaussian Parameters

Given fully-observable data: 这里是求导求出来的

$$\hat{\mu}_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\hat{\sigma}^2_{MLE} = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu})^2$$

用结果估计高斯分布的M值和6.
※拿到任何一个i.i.d的样本可以得到高斯分布

推导 $P(x|\mu,\Sigma)=\frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$

$$
\begin{aligned}
l(\mu,\Sigma) &= ln\prod_{i=1}^{n}p(x_n|\mu,\Sigma) \\
&= \sum_{i=1}^{n}ln(p(x_n|\mu,\Sigma)) \\
&= \sum_{i=1}^{n}(-\frac{d}{2}ln(2\pi)-\frac{1}{2}ln|\Sigma|-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))
\end{aligned}
$$

接下来就是 $max(l(\mu,\Sigma))$ 对 μ 和 Σ 分别求导，这一部分，一般课本上都是直接给出了结果，在这我给写出详细的推导过程，有矩阵论基础的应该可以看懂，不懂也没关系，就像我最开始的学习一样就是记住结果，仔细观察一下形式还是蛮容易记的。

以单样本形式对 μ 求微分：

$$
\begin{aligned}
d(tr(\hat{l}\,(\mu,\Sigma)) &= d(tr(-\frac{1}{2}(x^T\Sigma^{-1}x-x^T\Sigma^{-1}\mu-\mu^T\Sigma^{-1}x+\mu^T\Sigma^{-1}\mu)) \\
&= tr(-\frac{1}{2}(-x^T\Sigma^{-1}d\mu-d\mu^T\Sigma^{-1}x+d\mu^T\Sigma^{-1}\mu+\mu^T\Sigma^{-1}d\mu)) \\
&= tr(-\frac{1}{2}(-x^T\Sigma^{-1}-x^T\Sigma^{-1}+\mu^T\Sigma^{-1}+\mu^T\Sigma^{-1})d\mu) \\
&= tr((x^T\Sigma^{-1}-\mu^T\Sigma^{-1})d\mu)
\end{aligned}
$$

$\frac{d\hat{l}\,(\mu,\Sigma)}{d\mu}=\frac{dtr(\hat{l}\,(\mu,\Sigma))}{d\mu}=\Sigma^{-1}x-\Sigma^{-1}\mu$

由 $\sum_{i=1}^{n}\Sigma^{-1}(x_i-\hat{\mu})=0$ 可以推出 $\hat{\mu}=\frac{1}{n}\sum_{i=1}^{n}x_i$

同理以单样本形式对 Σ 求微分：

$$
\begin{aligned}
d(tr(\hat{l}\,(\mu,\Sigma)) &= -\frac{1}{2}(tr(\frac{1}{|\Sigma|}|\Sigma|\,\Sigma^{-1}d\Sigma)+dtr((x-\mu)(x-\mu)^T\Sigma^{-1}) \\
&= -\frac{1}{2}tr(\Sigma^{-1}d\Sigma-\Sigma^{-1}(x-\mu)(x-\mu)^T\Sigma^{-1}d\Sigma)
\end{aligned}
$$

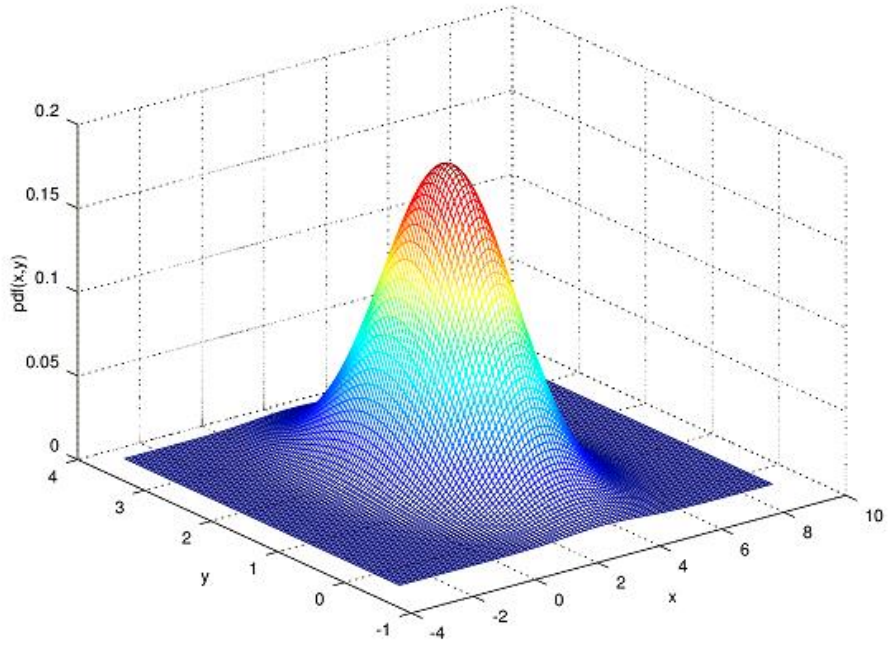
$$
\frac{dtr(\hat{l}\,(\mu,\Sigma))}{d\Sigma}=-\frac{1}{2}(\Sigma^{-1}-\Sigma^{-1}(x-\mu)(x-\mu)^T\Sigma^{-1})
$$

求导为0，两边同时左乘右乘一个 Σ 可以化简为

$\sum_{i=1}^{n}(\hat{\Sigma}-(x_i-\hat{\mu})(x_i-\hat{\mu})^T)=0$ 可以推出 $\hat{\Sigma}=\frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{\mu})(x_i-\hat{\mu})^T$

到此高斯分布的 μ,Σ 都根据数据集合 D 可以估计出来了，下面进入正题。

Multivariate Gaussians



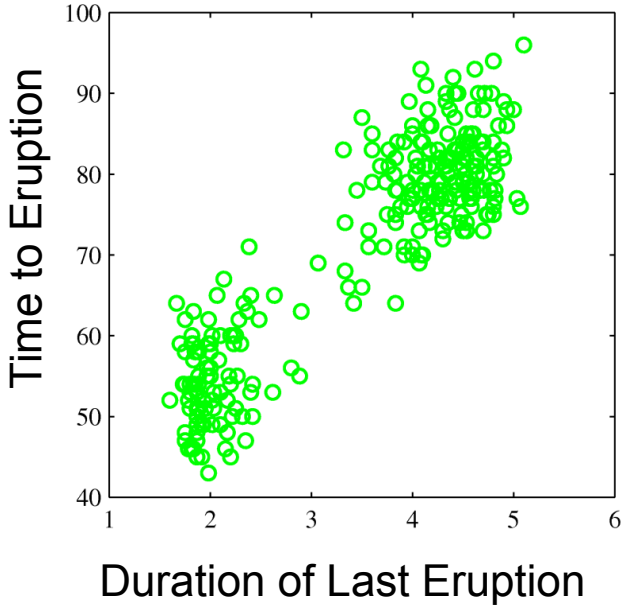
Covariance matrix Σ: degree to which x_i vary together

$$P(X = \mathbf{x}) = \frac{1}{(2\pi)^{m/2}\|\Sigma\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right]$$

$\Sigma = \sigma_1^2\sigma_2^2 \cdots \sigma_n^2$

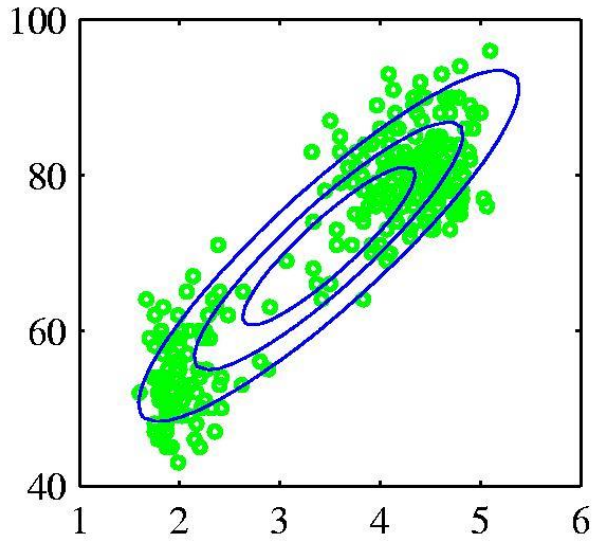
Mixtures of Gaussians

- Old Faithful Data Set



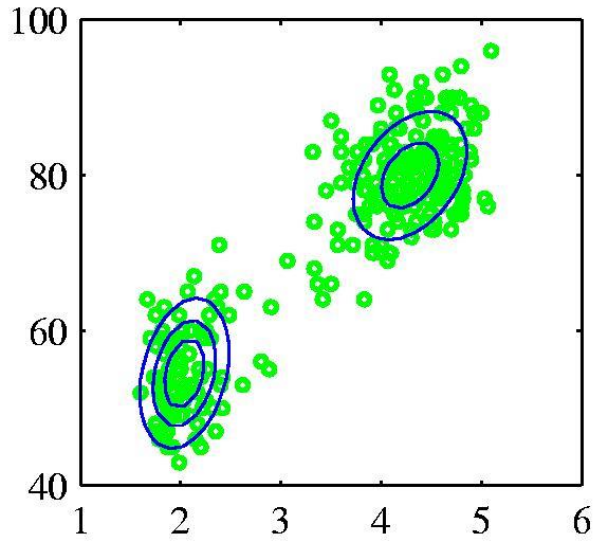
Mixtures of Gaussians

- Old Faithful Data Set



Single Gaussian

(同一个Gaussian,独立同分布采样)



Mixture of two Gaussians

两个Gaussian分布,混合的结果.

Mixtures of Gaussians

由K个Gaussian成分线性组合

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

π_k：先验，属于第k个Gaussian分布的概率，先验概率之和为0

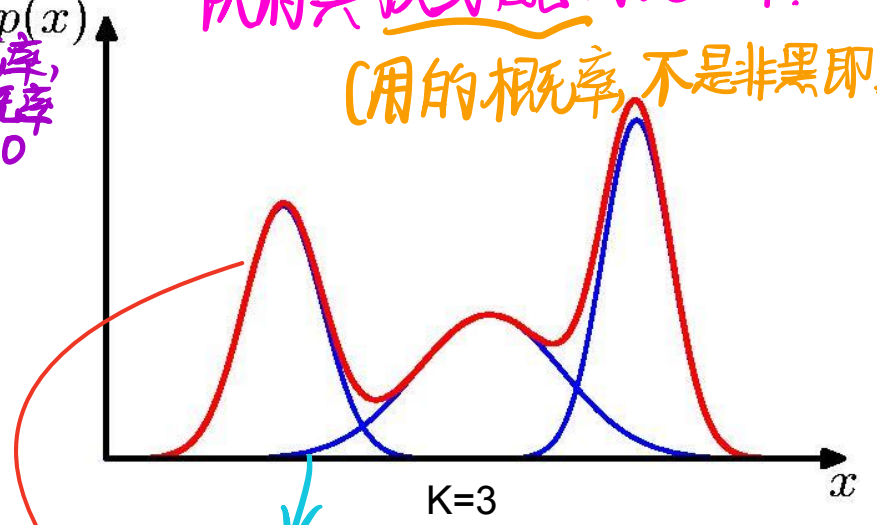
$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$：Component

π_k：Mixing coefficient

$$\forall k : \pi_k \geqslant 0 \qquad \sum_{k=1}^{K} \pi_k = 1$$

对于任何一个样本，我们可以将其软分配到类簇中.

(用的概率，不是非黑即白).



三个高斯成分，$\pi_1 = \pi_2 = \pi_3 = \frac{1}{3}$

红色为描绘的分布

BTW，我们先要从样本来估计 P(X)模型.

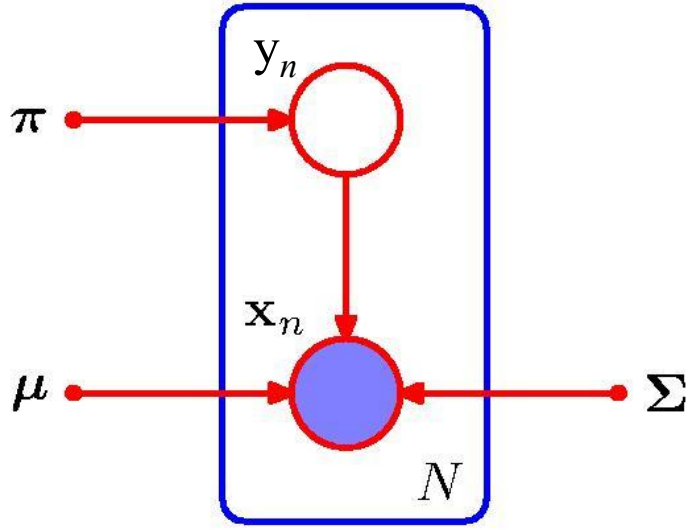
Gaussian mixture model

- P(Y): Distribution over *k* components (clusters) 就是π_k啦，选到第[Σ,μ]的概率
- P(X|Y): Each component generates data from a **multivariate Gaussian** with mean μ_i and covariance matrix Σ_i

$P(X=x \mid [\Sigma, \mu])$

Each data point is sampled from a ***generative process*:**

1. Choose component *i* with probability π_i
2. Generate data point from $N(\mathbf{x} \mid \mu_i, \Sigma_i)$



我们有一组数据 x = （x1, x2, ...），和GMM模型的数学公式：

$$P(x) = \sum_{k=1}^{K} \lambda_k N(x|\theta_k)$$

建立似然函数Likelyhood：

代入 $(x_1, x_2 \cdots x_n)$

$$L(x|\theta) = \prod_{i=1}^{N} P(x_i) = \prod_{i=1}^{N} [\sum_{k=1}^{K} \lambda_k N(x_i|\theta_k)]$$

使用log运算，建立log-likelyhood：

$$\begin{aligned} log[L(x|\theta)] &= log[\prod_{i=1}^{N} P(x_i)] \\ &= \sum_{i=1}^{N} logP(x_i) \\ &= \sum_{i=1}^{N} log[\sum_{k=1}^{K} \lambda_k N(x_i|\theta_k)] \end{aligned}$$

此时，对log-liklyhood 函数求参数的偏导，令偏导等于 0 的手段，不会管用。因为log函数内部有加法，没有得到更加简洁的形式，解不出来想要的解。

Supervised learning for GMM

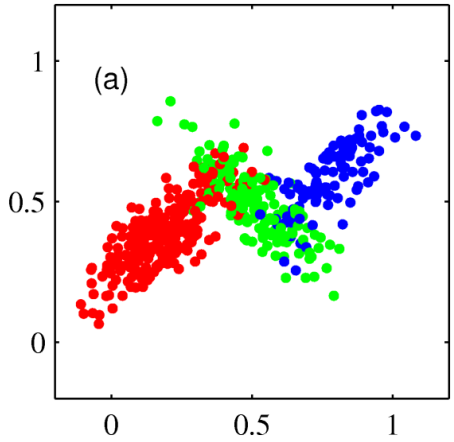
- We observe both the data points and their labels (generated from which Gaussian components) 采样得到了GMM. (有point data和其label)
- How do we estimate parameters of GMM?

有监督学习

我们现在有每一个 data point 的 $x(x_1 \cdots x_n)$

和它对应的 label，现在要估计

GMM中 每一个 $[M_i, \Sigma_i]$ 以及 π_k



Supervised learning for GMM

- We observe both the data points and their labels (generated from which Gaussian components)
- How do we estimate parameters of GMM?
- Objective: maximize the likelihood 使用最大似然估计

$$\prod_j P(y_j = i, \mathbf{x}_j) = \prod_j \pi_i N(\mathbf{x}_j | \mu_i, \Sigma_i)$$

这里$y_j=i$证明选择了第i个的高斯模型

$P(y_j=i, x_j)$

$= \frac{\pi_i}{D} \cdot N(x_j | \mu_i, \Sigma_i)$

选择i Gauss

- Closed form solution: 一共有m个数据点，对于j，j的标签为i
  - *m* data points. For component *i*, suppose we have *n* data points with label *i*.

$$\mu_i = \frac{1}{n}\sum_{j=1}^{n} \mathbf{x}_j \qquad \Sigma_i = \frac{1}{n}\sum_{j=1}^{n} (\mathbf{x}_j - \mu_i)(\mathbf{x}_j - \mu_i)^T \qquad \pi_i = \frac{n}{m}$$

$\frac{n}{m}$ ⇒ label k的样本数 / 总的采样数

Pf. in page 26 ∵因为已知样本属于哪一个高斯分布，所以可以独立算单个gaussian

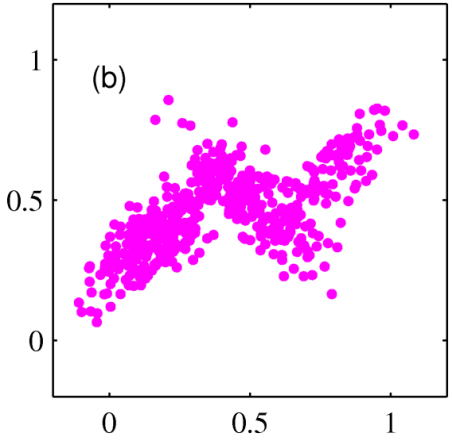
Unsupervised learning for GMM

- In clustering, we don't know the labels Y! (但实际，无法提前已知 label)
- Maximize marginal likelihood: 我们可以规避掉 z_i，而使用边际概率

$$\prod_j P(\mathbf{x}_j) = \prod_j \sum_i P(y_j = i, \mathbf{x}_j) = \prod_j \sum_i \pi_i N(\mathbf{x}_j | \mu_i, \Sigma_i)$$

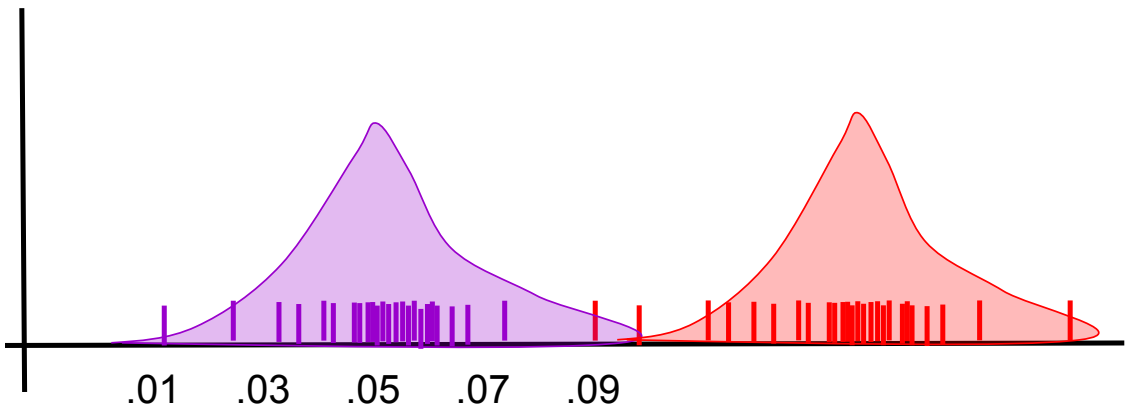
- How do we optimize it?
  - No closed form solution (没有封闭解形式)

那么可以假设一种分布，再进行更新。

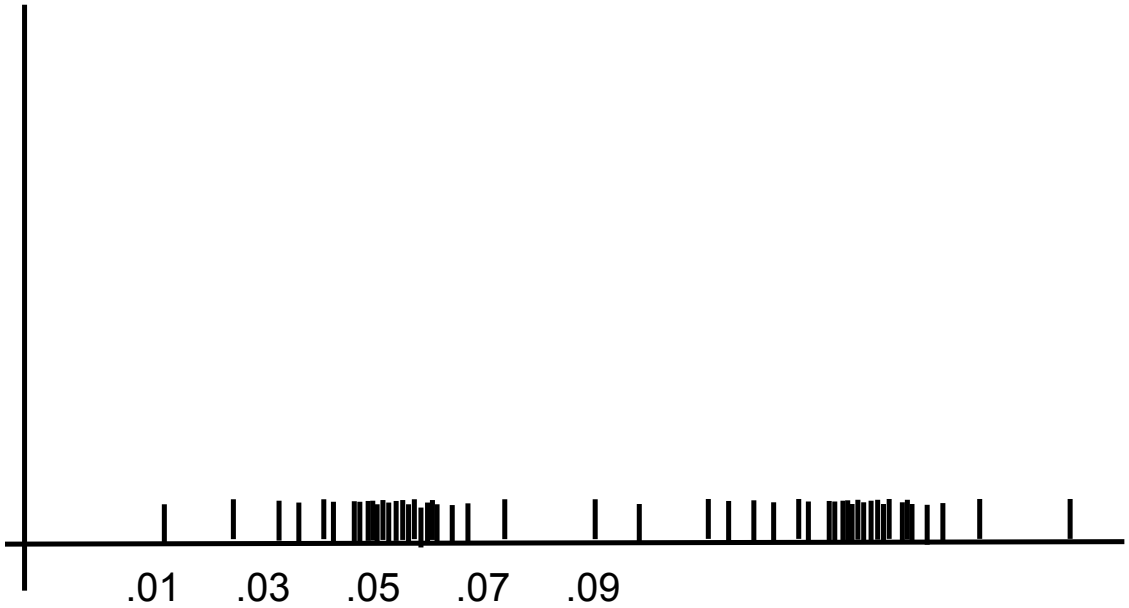


Simplest Example

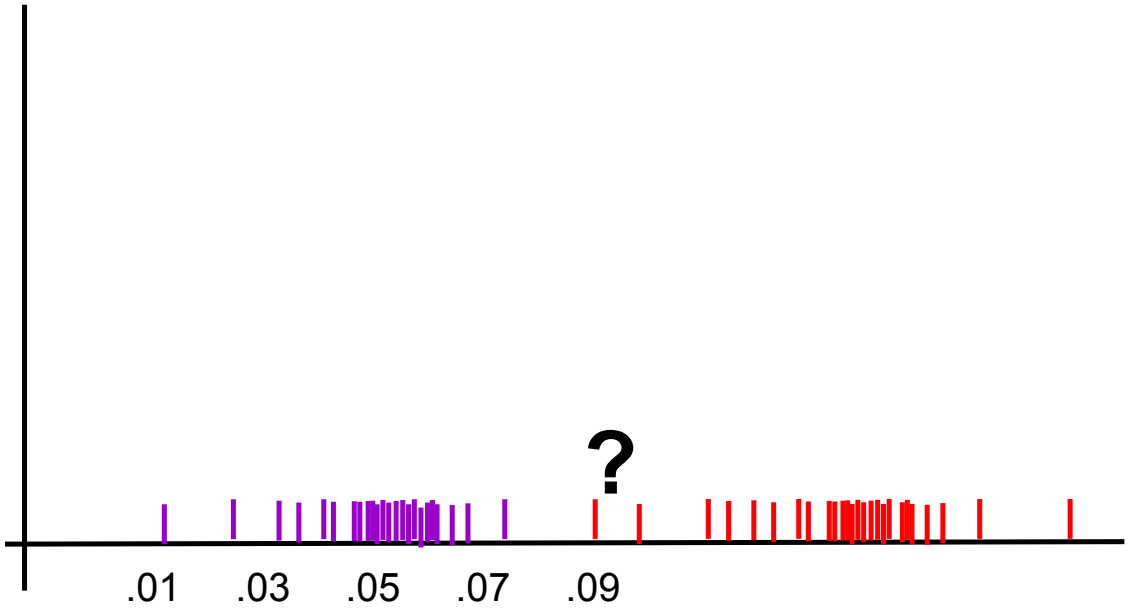
Mixture of two distributions



Input Looks Like

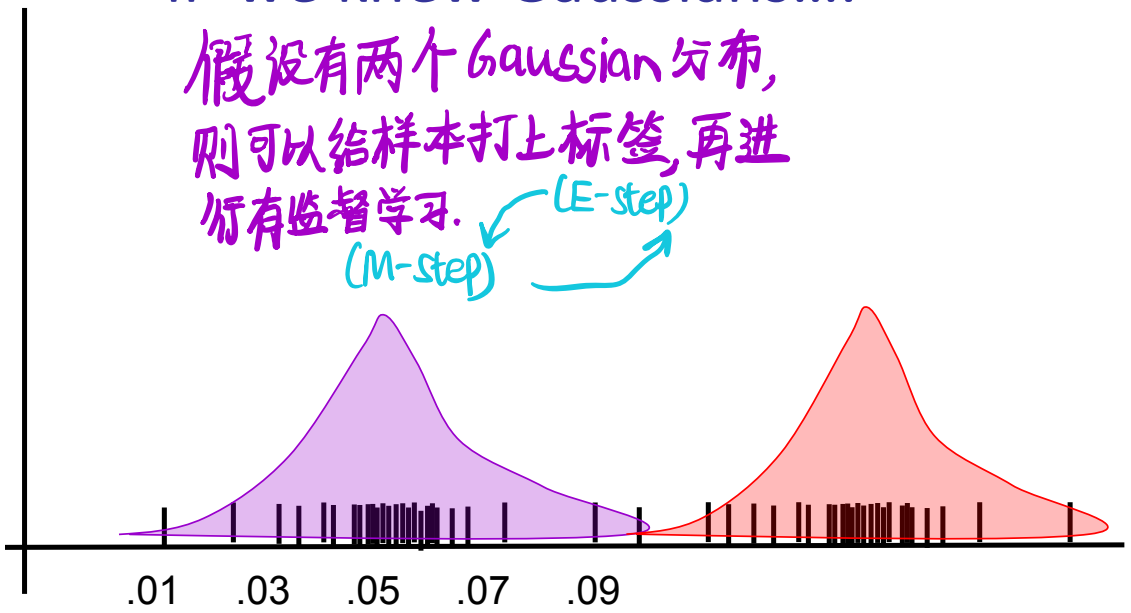


We Want to Predict



Chicken & Egg

Note that coloring instances would be easy if we knew Gaussians….

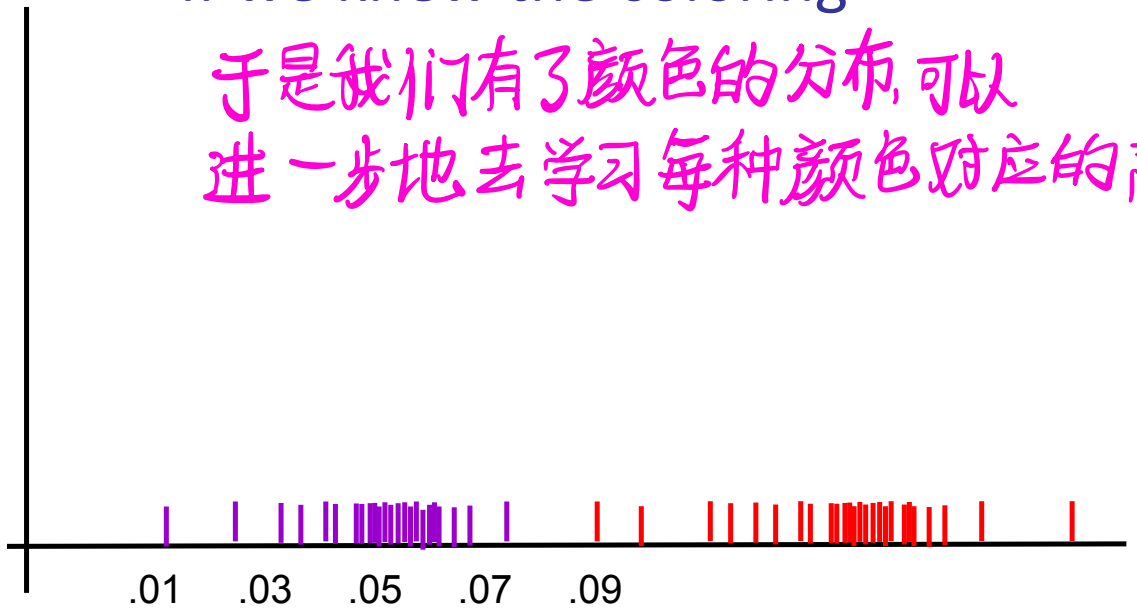


Chicken & Egg

And finding the Gaussians would be easy

If we knew the coloring

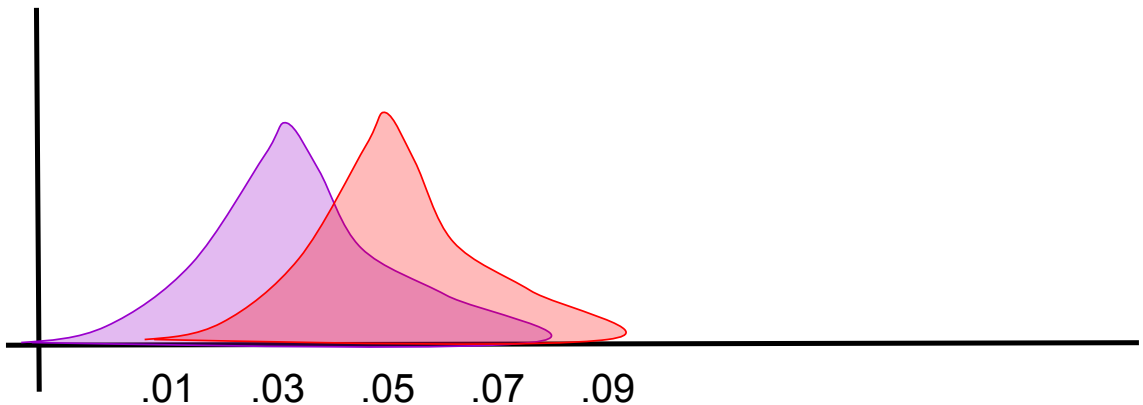
于是我们有了颜色的分布，可以
进一步地去学习每种颜色对应的高斯分布



Expectation Maximization (EM)

- Pretend we do know the parameters
  - Initialize randomly

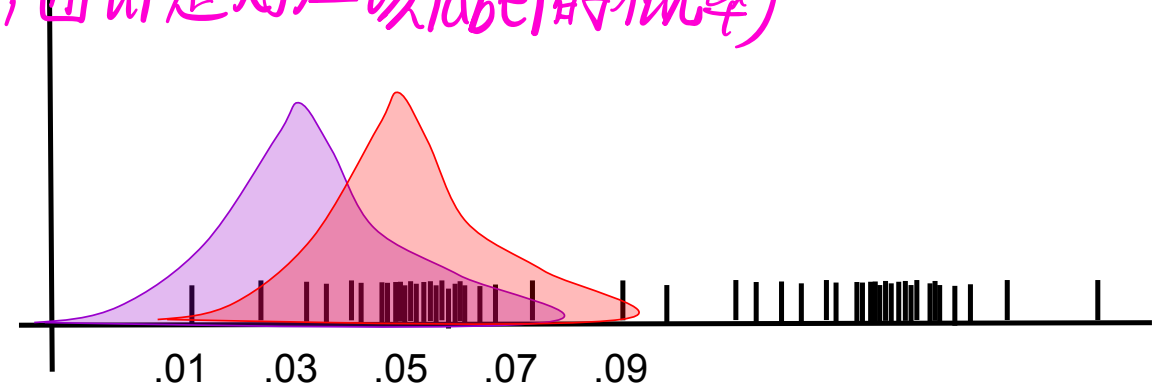
①假定我们有高斯分布且已知参数



Expectation Maximization (EM)

- [E step] Compute probability of each instance having each possible label

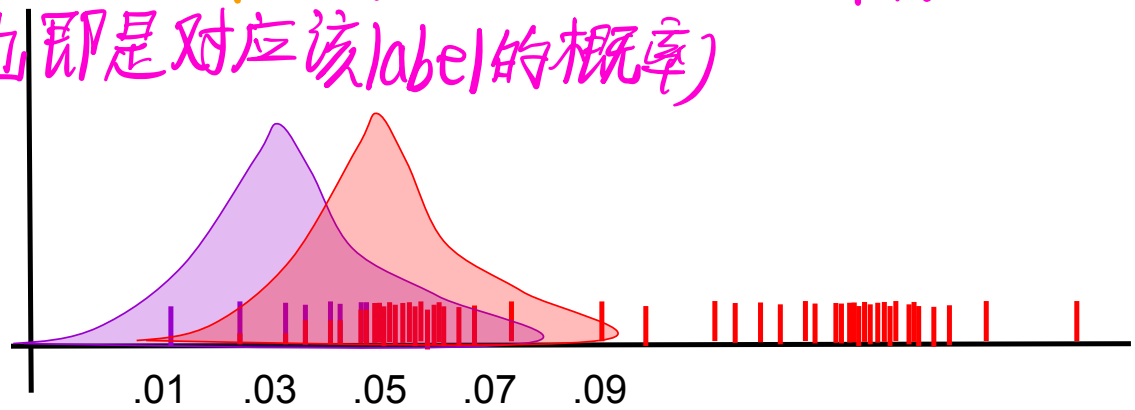
②根据已知的高斯分布计算每个数据点在该高斯分布下的概率(因于每一个Gauss分布代表一个label，也即是对应该label的概率)



Expectation Maximization (EM)

- [E step] Compute probability of each instance having each possible label

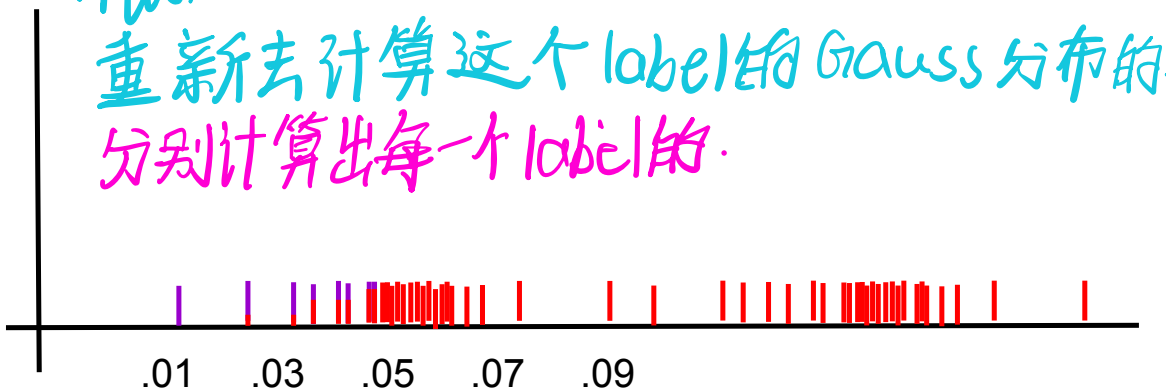
②根据已知的高斯分布计算每个数据点在该高斯分布下的概率(由于每一个Gauss分布代表一个label，也即是对应该label的概率)



Expectation Maximization (EM)

- [E step] Compute probability of each instance having each possible label
- [M step] Treating each instance as fractionally having both labels, compute the new parameter values

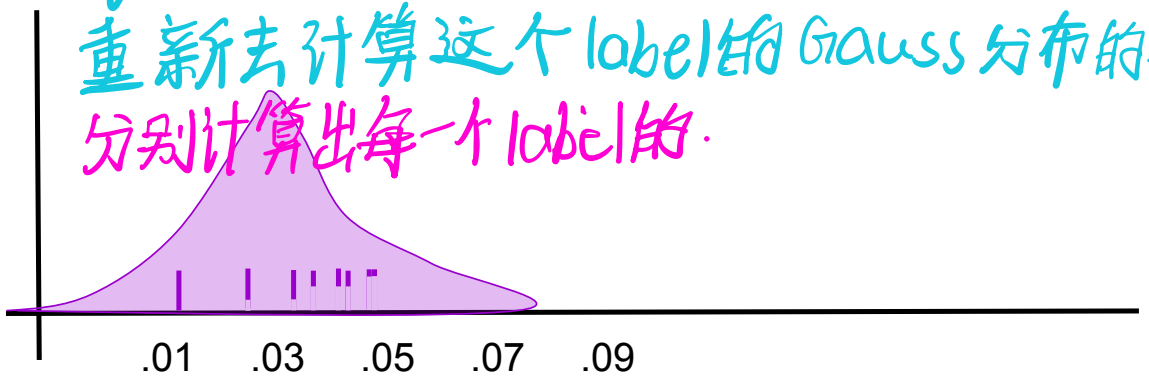
M: 根据每个数据点的上一步得出的对应某label的概率，重新去计算这个label的Gauss分布的参数.
分别计算出每一个label的.



Expectation Maximization (EM)

- [E step] Compute probability of each instance having each possible label
- [M step] Treating each instance as fractionally having both labels, compute the new parameter values

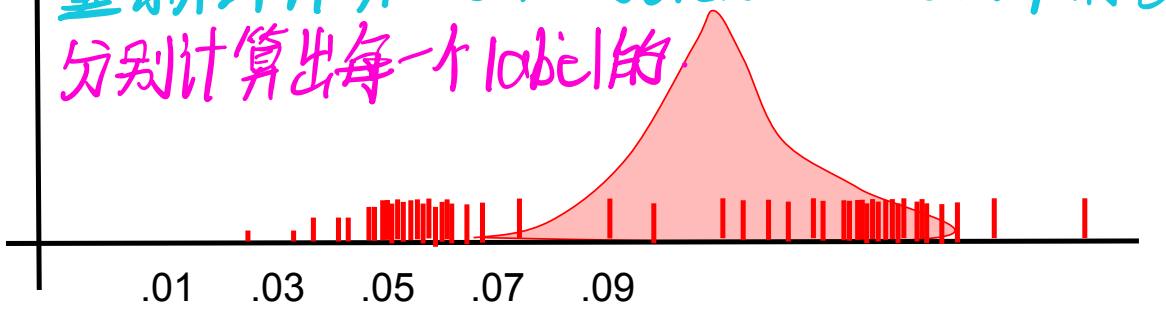
M: 根据每个数据点的上一步得出的对应某label的概率，重新去计算这个label的Gauss分布的参数.
分别计算出每一个label的.



Expectation Maximization (EM)

- [E step] Compute probability of each instance having each possible label
- [M step] Treating each instance as fractionally having both labels, compute the new parameter values

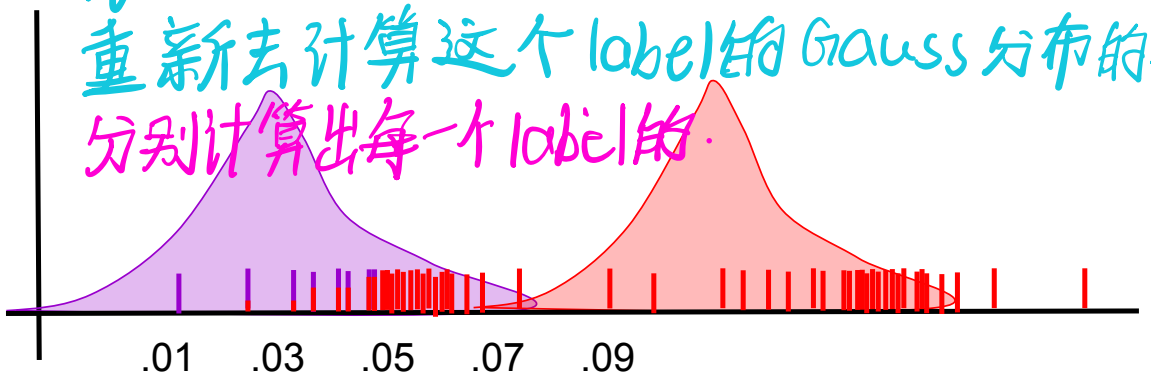
M: 根据每个数据点的上一步得出的对应某label的概率，重新去计算这个label的Gauss分布的参数.
分别计算出每一个label的.



Expectation Maximization (EM)

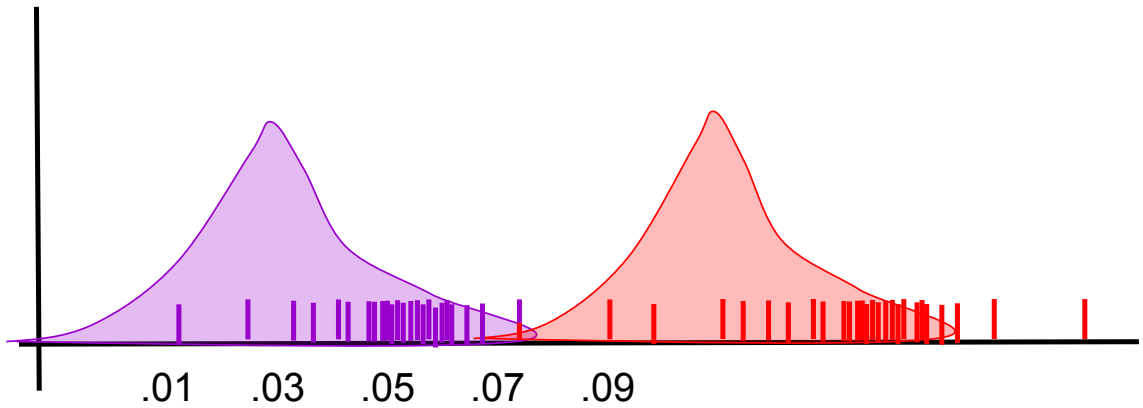
- [E step] Compute probability of each instance having each possible label
- [M step] Treating each instance as fractionally having both labels, compute the new parameter values

M: 根据每个数据点的上一步得出的对应某label的概率，重新去计算这个label的Gauss分布的参数.
分别计算出每一个label的.



Expectation Maximization (EM)

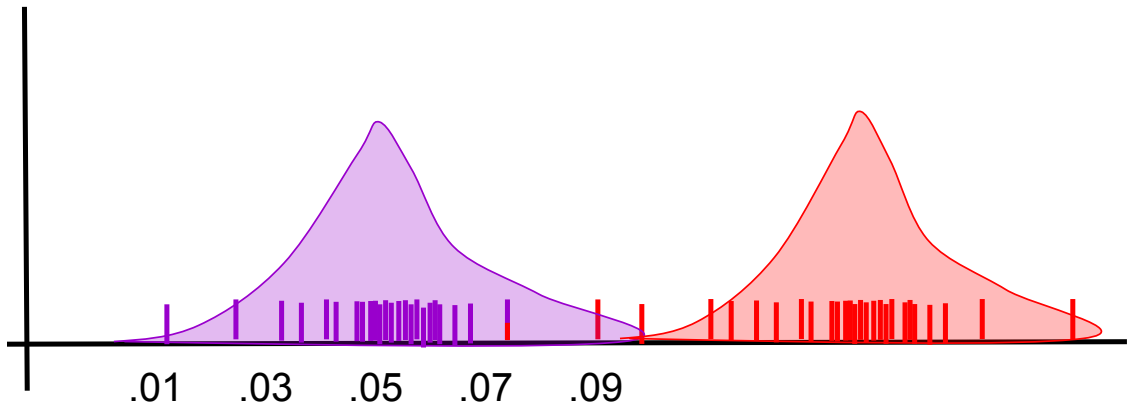
- Repeat E-step 重复E步



Expectation Maximization (EM)

- Repeat E-step
- Repeat M-step 重复M步

… until convergence (直到收敛) ⇐ π_k, μ, σ 不变



Expectation Maximization (EM)

- Pick K random cluster models (Gaussians)
  - 人为指定的☆？(how to precise)
- Alternate:
  - EM并不一定要求是Gaussian模型
  - 先将数据点按比例地分给不同的 model (label)
  - Assign data instances proportionately to different models
  - Revise each cluster model based on its (proportionately) assigned points
- Stop when no changes
  - 再根据在该model (label)内的数据点的概率值，去调整 model的参数.

EM: Two Easy Steps

EM:目标，找到使 $\prod_j \sum_{i=1}^{k} P(y_j=i, x_j|\theta)$ 最大的 θ

Objective: $\text{argmax}_\theta \prod_j \sum_{i=1}^{k} P(y_j=i,x_j|\theta) = \sum_j \log \sum_{i=1}^{k} P(y_j=i,x_j|\theta)$

Data: $\{x_j \mid j=1 .. n\}$

Notation a bit inconsistent
Parameters = $\theta=\lambda$

- **E-step**: Compute expectations to "fill in" missing y values according to current parameters, θ
  - For all examples j and values i for y, compute: $P(y_j=i \mid x_j, \theta)$

先将数据点按比例地分给不同的 model(label)

- **M-step**: Re-estimate the parameters with "weighted" MLE estimates
  - Set $\theta = \text{argmax}_\theta \sum_j \sum_{i=1}^{k} P(y_j=i \mid x_j, \theta) \log P(y_j=i,x_j|\theta)$

再根据在该model(label)内的数据点的概率值，去调整 model的参数.

Especially useful when the E and M steps have closed form solutions!!!

EM for GMM

Iterate: On the *t*'th iteration let our estimates be

$$\theta^{(t)} = \{ \mu_1^{(t)}, \mu_2^{(t)} \dots \mu_k^{(t)}, \Sigma_1^{(t)}, \Sigma_2^{(t)} \dots \Sigma_k^{(t)}, \pi_1^{(t)}, \pi_2^{(t)} \dots \pi_k^{(t)} \}$$

$\mu_i = \frac{1}{n} \sum_{j=1}^{n} x_j \qquad \pi_i = \frac{n}{m}$

$\Sigma_i = \frac{1}{n} \sum_{j=1}^{n} (x_j - \mu_i)(x_j - \mu_i)^T$

E-step

Compute label distribution of each data point

$$P\left(y_j = i \,\middle|\, x_j, \theta^{(t)}\right) \propto \pi_i^{(t)} N\left(x_j \,\middle|\, \mu_i^{(t)}, \Sigma_i^{(t)}\right)$$

Just evaluate a Gaussian at x_j

在t时刻假定θ，计算P(yi=i)

M-step

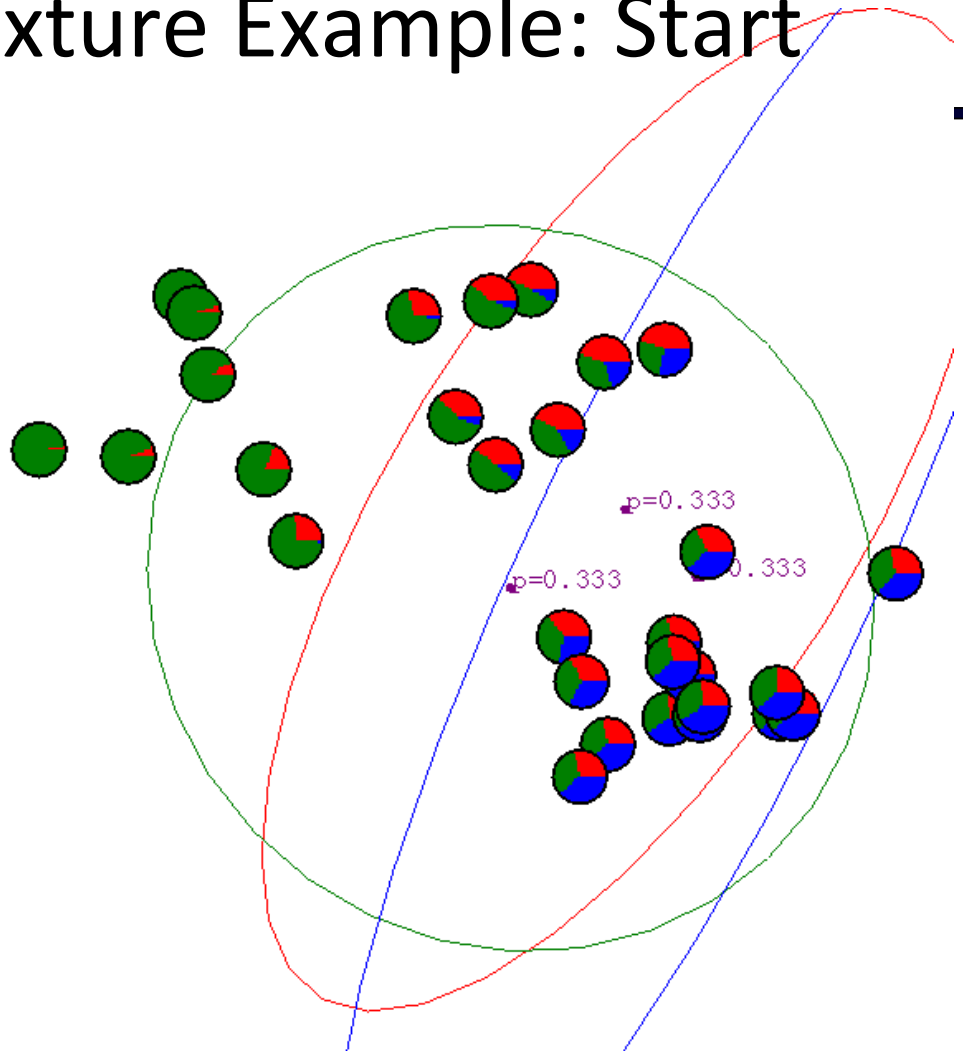
Compute weighted MLE of parameters given label distributions

$$\mu_i^{(t+1)} = \frac{\sum_j P\left(y_j = i \,\middle|\, x_j, \theta^{(t)}\right) x_j}{\sum_{j'} P\left(y_{j'} = i \,\middle|\, x_{j'}, \theta^{(t)}\right)} \qquad \Sigma_i^{(t+1)} = \frac{\sum_j P\left(y_j = i \,\middle|\, x_j, \theta^{(t)}\right) \left[x_j - \mu_i^{(t+1)}\right]\left[x_j - \mu_i^{(t+1)}\right]^T}{\sum_{j'} P\left(y_{j'} = i \,\middle|\, x_{j'}, \theta^{(t)}\right)}$$

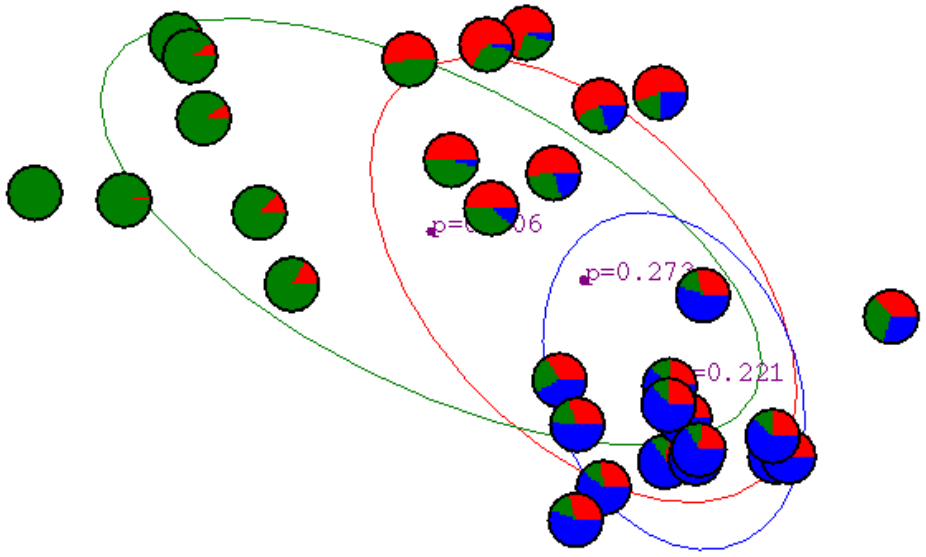
$$\pi_i^{(t+1)} = \frac{\sum_j P\left(y_j = i \,\middle|\, x_j, \theta^{(t)}\right)}{m}$$

m = #training examples

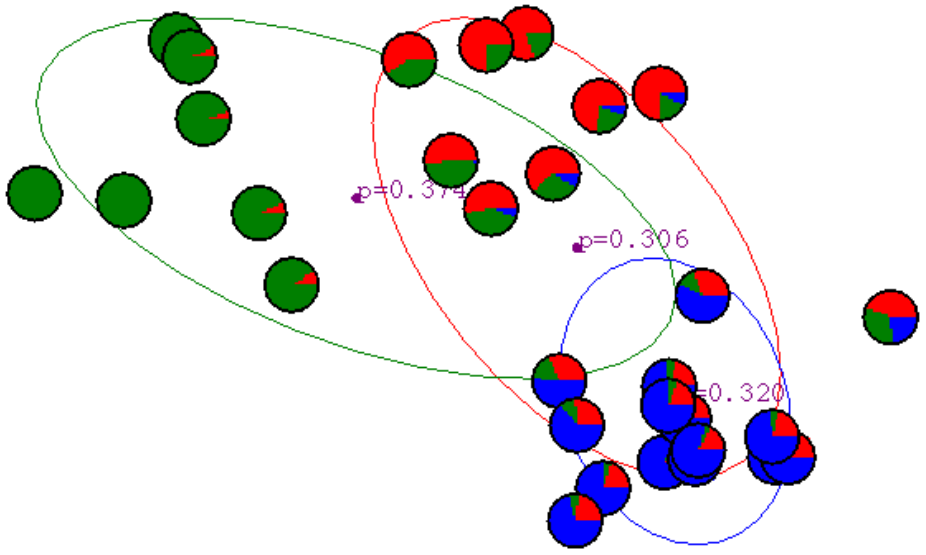
Gaussian Mixture Example: Start



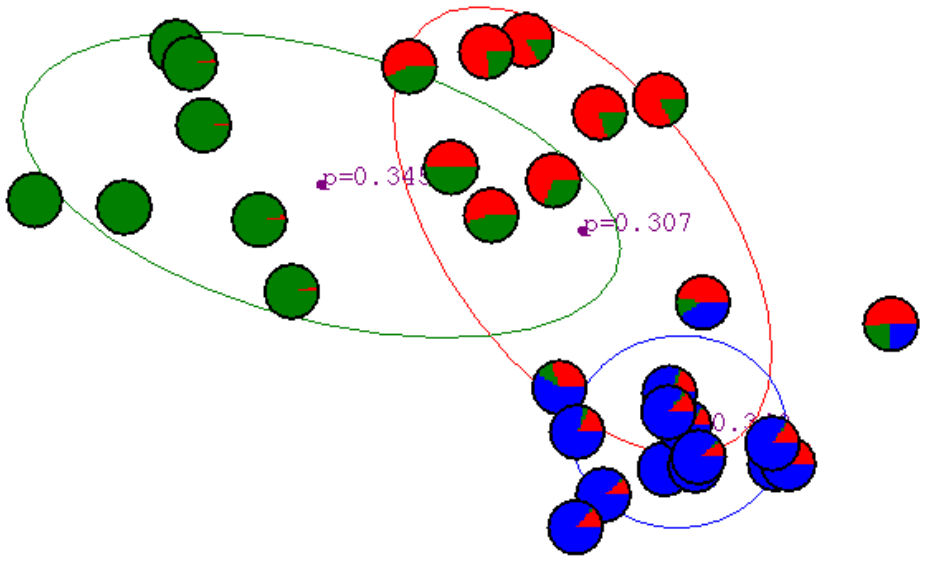
After first iteration



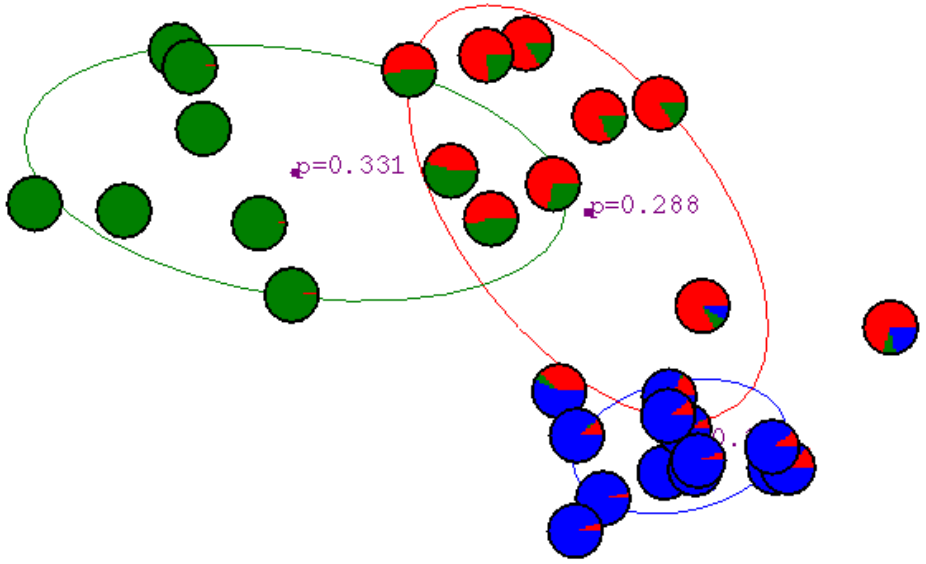
After 2nd iteration



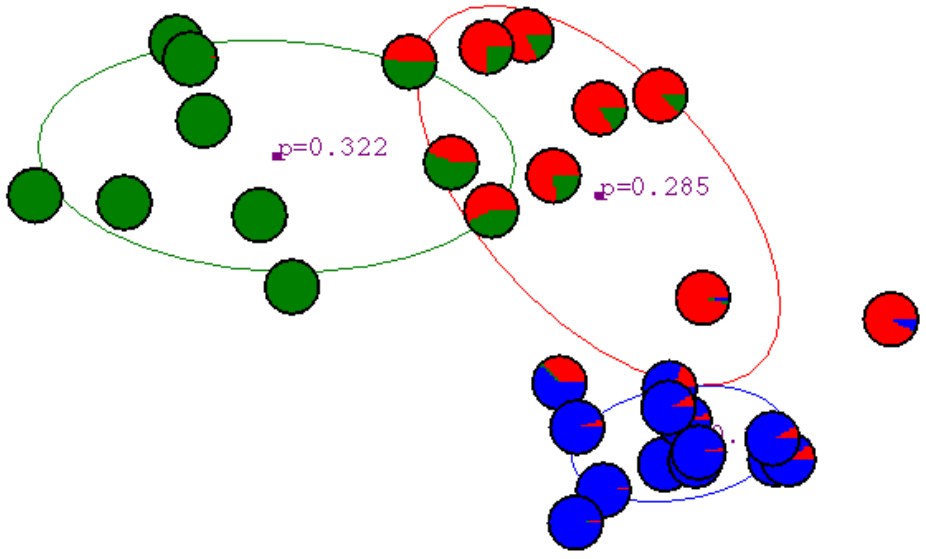
After 3rd iteration



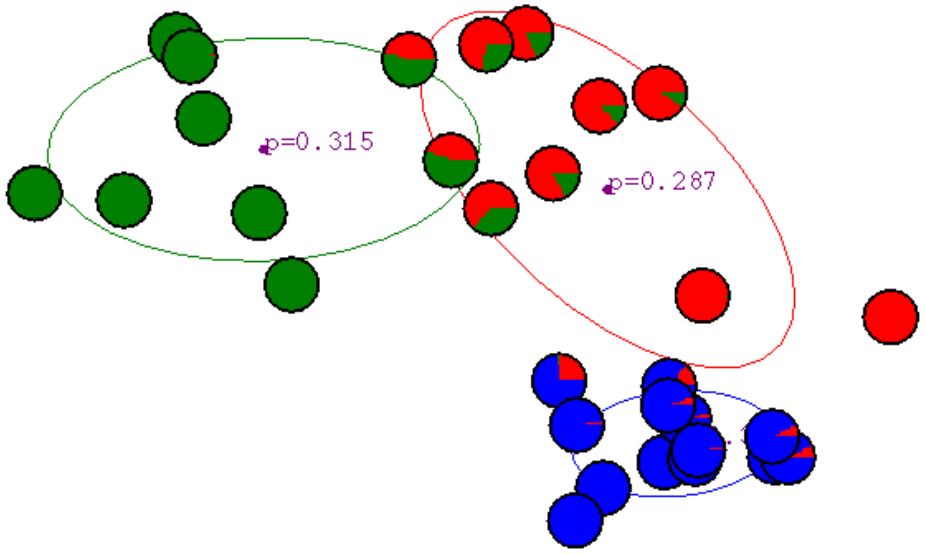
After 4th iteration



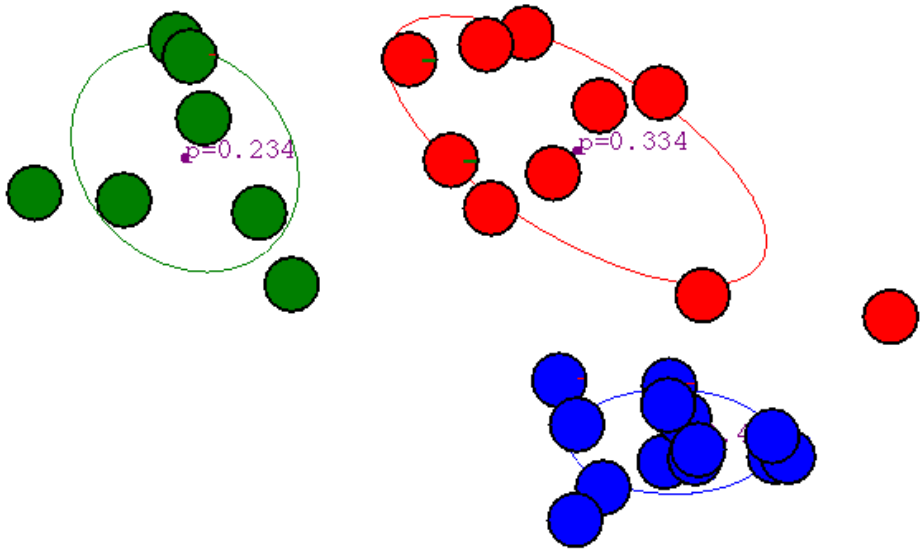
After 5th iteration



After 6th iteration



After 20th iteration



EM and K-means

也是迭代了两步

- EM degrades to k-means if we assume
  - All the Gaussians are spherical and have identical weights and covariances
    - i.e., the only parameters are the means
  - The label distributions computed at E-step are point-estimations
    - i.e., hard-assignments of data points to Gaussians
    - Alternatively, assume the variances are close to zero

球形的

(硬分配)

(不硬分配中,每个高斯成分的方差为零)

EM in General

- Can be used to learn any model with hidden variables (missing data)
- Alternate:
  - Compute distributions over hidden variables based on current parameter values
  - Compute new parameter values to maximize expected log likelihood based on distributions over hidden variables
- Stop when no changes

Summary

- Clustering 聚类算法
  - Group together similar instances
- K-means
  - Assign data instances to closest mean
  - Assign each mean to the average of its assigned points
- EM
  - Assign data instances proportionately to different Gaussian models
  - Revise each model based on its (proportionately) assigned points