So far, supervised learning : $h: x \to \mathbb{R}$ ← regression
$h: x \to \{0, 1, \ldots, k\}$ ← classification

Unsupervised learning

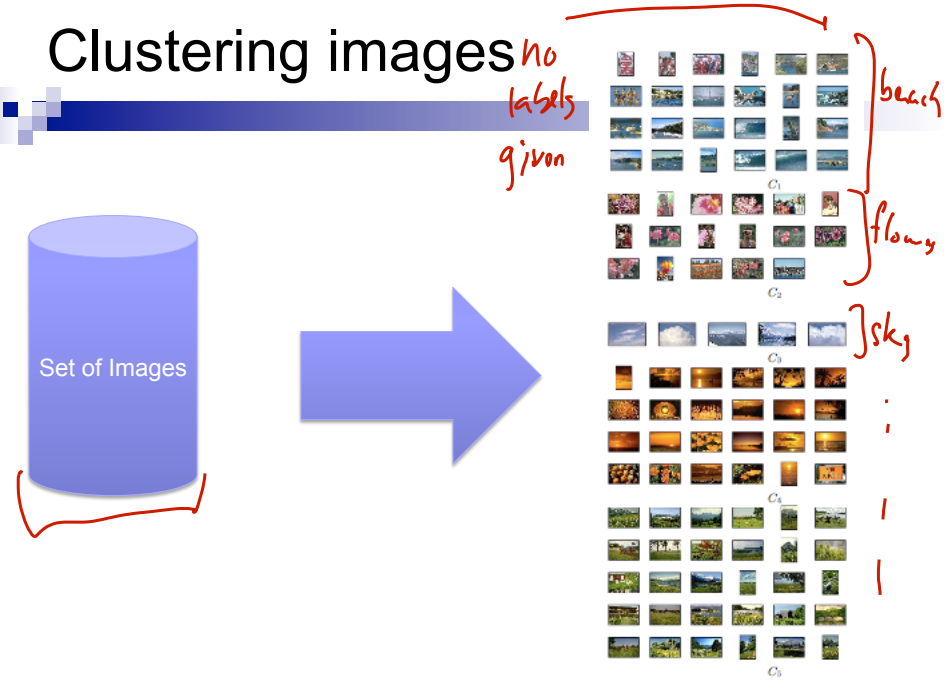
Clustering
K-means Continued

Machine Learning – CSE446
Carlos Guestrin
University of Washington

May 15, 2013

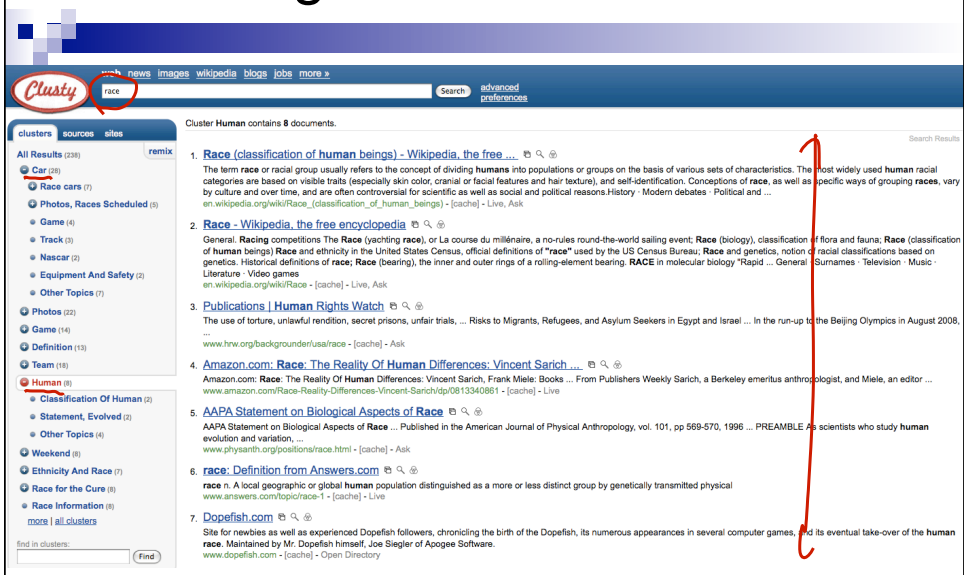
Clustering images

no labels given

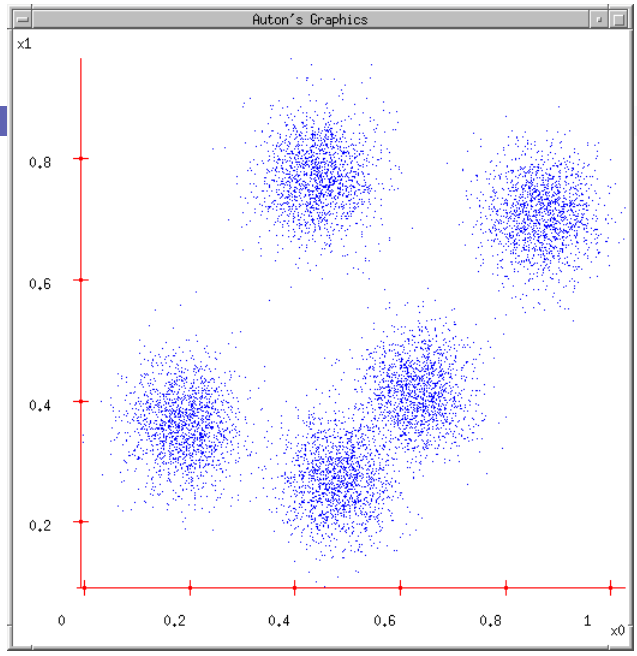


[Goldberger et al.]

Clustering web search results

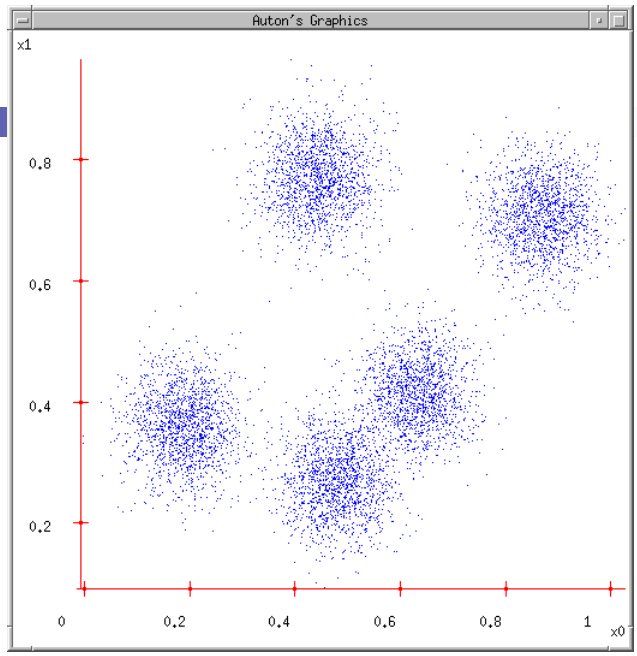


Some Data



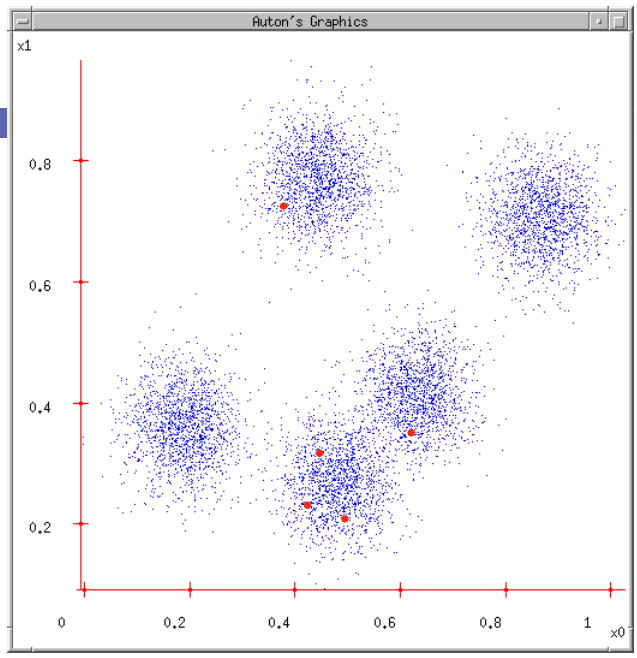
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*



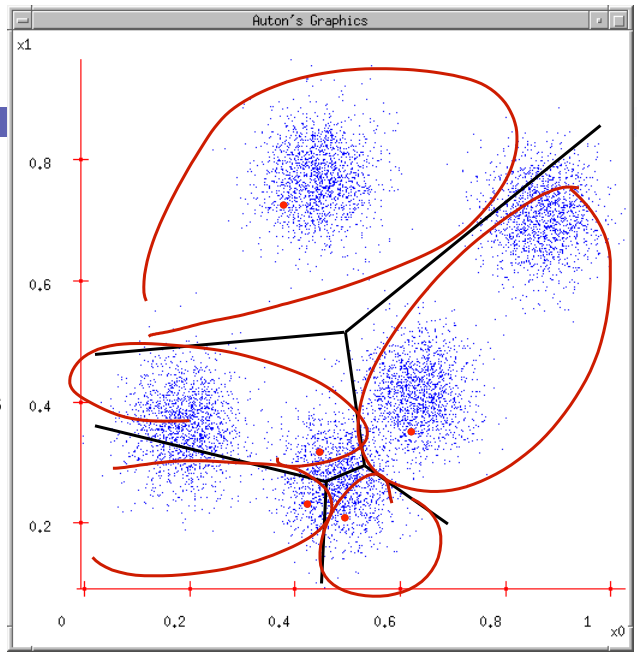
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations



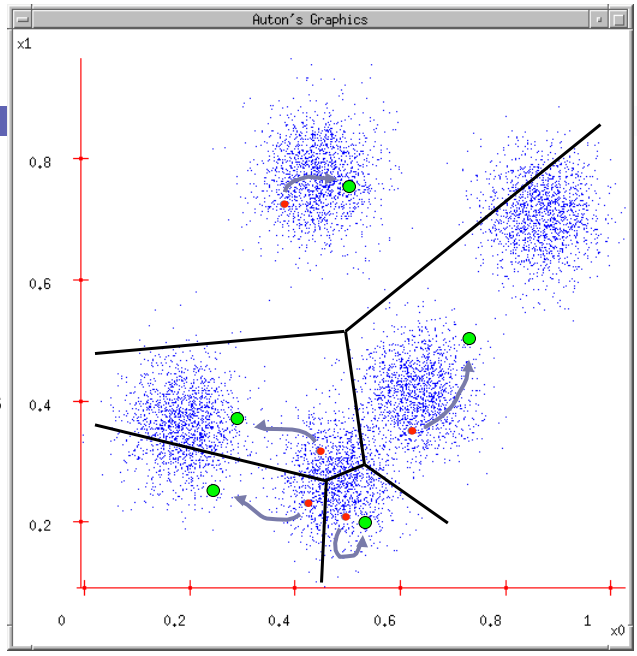
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to. (Thus each Center "owns" a set of datapoints)



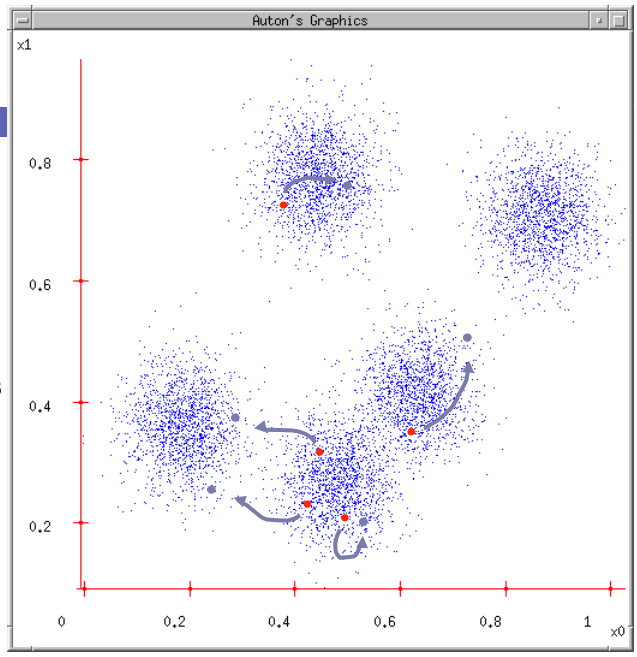
K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns



K-means

1. Ask user how many clusters they'd like. *(e.g. k=5)*
2. Randomly guess k cluster Center locations
3. Each datapoint finds out which Center it's closest to.
4. Each Center finds the centroid of the points it owns…
5. …and jumps there
6. …Repeat until terminated!



K-means

- Randomly initialize *k* centers or smartly
  - $\mu^{(0)} = \mu_1^{(0)},\ldots, \mu_k^{(0)}$

  converge when nothing moves (no point changes its cluster)
- **Classify**: Assign each point $j \in \{1,\ldots n\}$ to nearest center:

  ith cluster center

  data point
  - $C^{(t)}(j) \leftarrow \arg\min_i ||\mu_i - x_j||^2$
- **Recenter**: μ_i becomes centroid of its point:
  - $\mu_i^{(t+1)} \leftarrow \arg\min_\mu \sum_{j:C(j)=i} ||\mu - x_j||^2$ ← $\mu_i = \frac{\sum_{j:C(j)=i} x_j}{\text{num of points assigned to cluster } i}$
  - Equivalent to μ_i ← average of its points!

What is K-means optimizing?

- Potential function F(μ,C) of centers μ and point allocations C:
  - $F(\mu, C) = \sum_{j=1}^{N} ||\mu_{C(j)} - x_j||^2$
- Optimal K-means:
  - $\min_{\mu} \min_{C} F(\mu, C)$

Does K-means converge??? Part 1

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix μ, optimize C

Does K-means converge??? Part 2

- Optimize potential function:

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Fix C, optimize μ

Coordinate descent algorithms

$$\min_{\mu} \min_{C} F(\mu, C) = \min_{\mu} \min_{C} \sum_{i=1}^{k} \sum_{j:C(j)=i} ||\mu_i - x_j||^2$$

- Want: $\min_a \min_b F(a,b)$
- Coordinate descent:
  - fix a, minimize b
  - fix b, minimize a
  - repeat
- Converges!!!
  - if F is bounded
  - to a (often good) local optimum
    - as we saw in applet (play with it!)
      - (For LASSO it converged to the optimum)

- K-means is a coordinate descent algorithm!

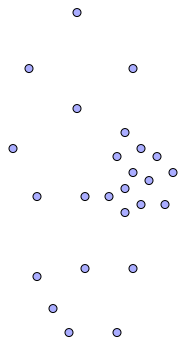
Mixtures of Gaussians

Machine Learning – CSE446
Carlos Guestrin
University of Washington

May 15, 2013

(One) bad case for k-means

- Clusters may overlap
- Some clusters may be "wider" than others



Gaussians in *m* Dimensions

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

Suppose You Have a Gaussian For Each Class

$$P(\mathbf{x} \mid y = i) \propto \frac{1}{(2\pi)^{m/2} \, \| \Sigma_i \|^{1/2}} \exp\left[-\frac{1}{2} \left(\mathbf{x} - \mu_i \right)^T \Sigma_i^{-1} \left(\mathbf{x} - \mu_i \right) \right]$$

Gaussian Bayes Classifier

- You have a Gaussian over **x** for each class y=i:

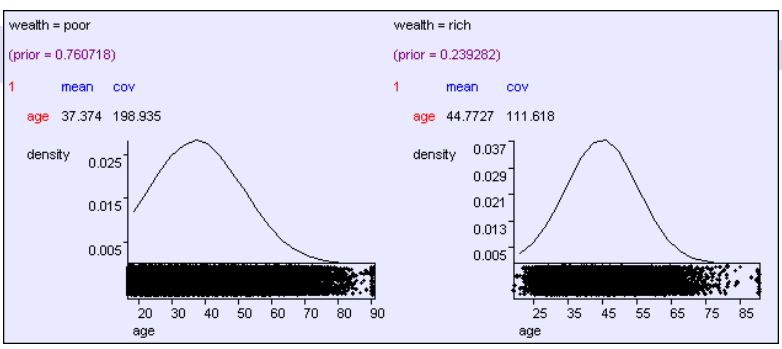
$$P(\mathbf{x} \mid y=i) \propto \frac{1}{(2\pi)^{m/2} \, \|\Sigma_i\|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1} (\mathbf{x}-\mu_i)\right]$$

- But you need probability of class y=i given **x**:

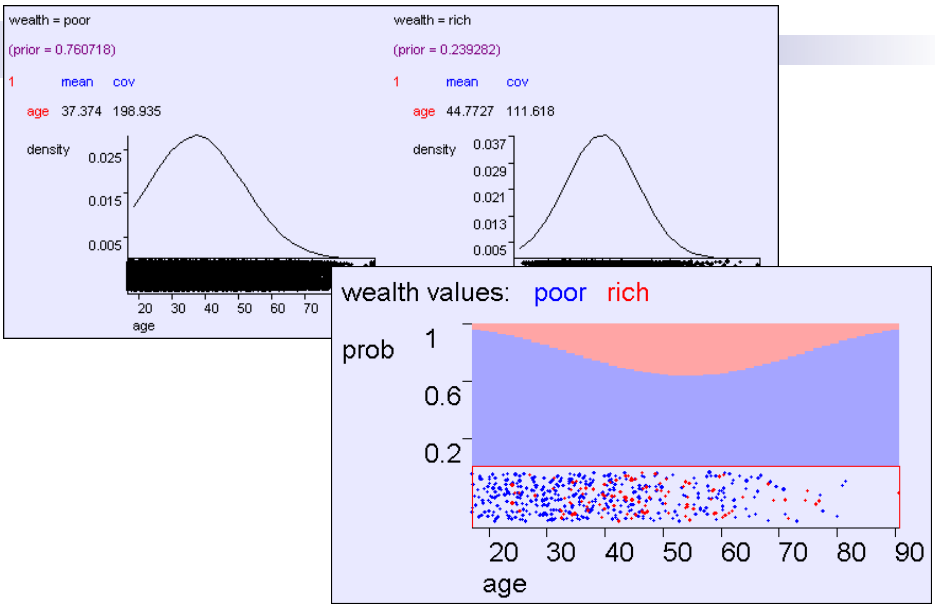
- Thank you Bayes Rule!!

$$P(y=i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid y=i) P(y=i)}{p(\mathbf{x})}$$

Predicting wealth from age

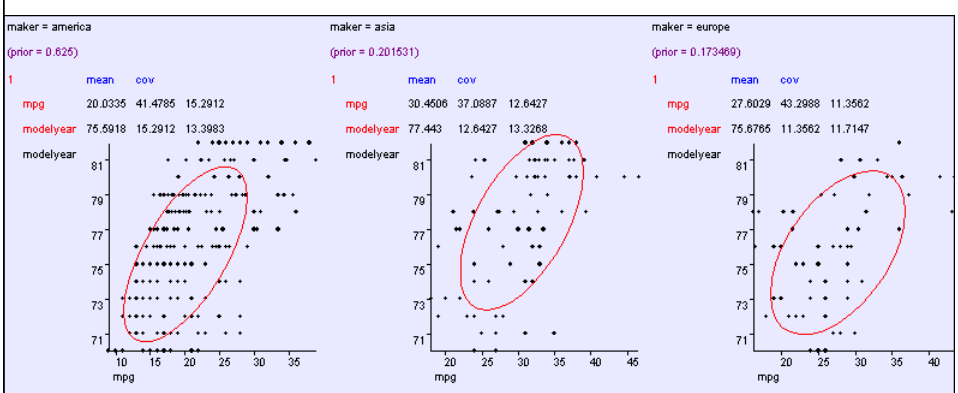


Predicting wealth from age



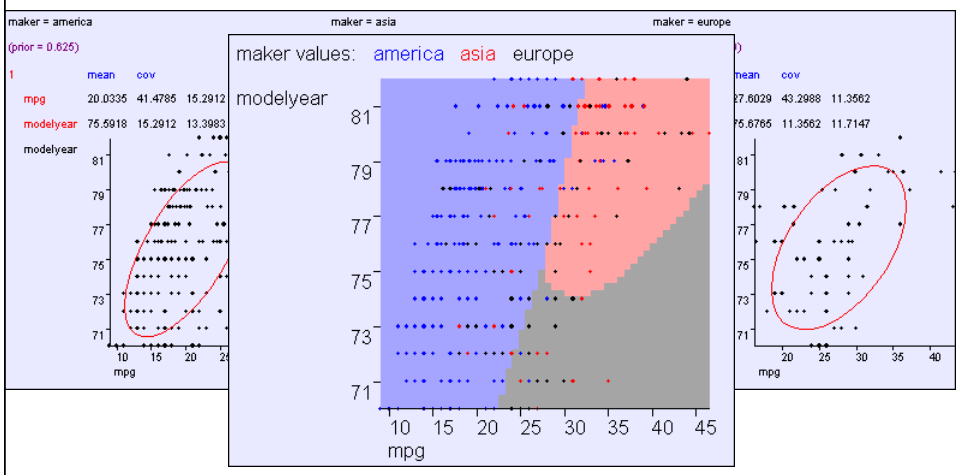
Learning modelyear , mpg ---> maker

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{12} & \sigma_2^2 & \cdots & \sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1m} & \sigma_{2m} & \cdots & \sigma_m^2 \end{pmatrix}$$



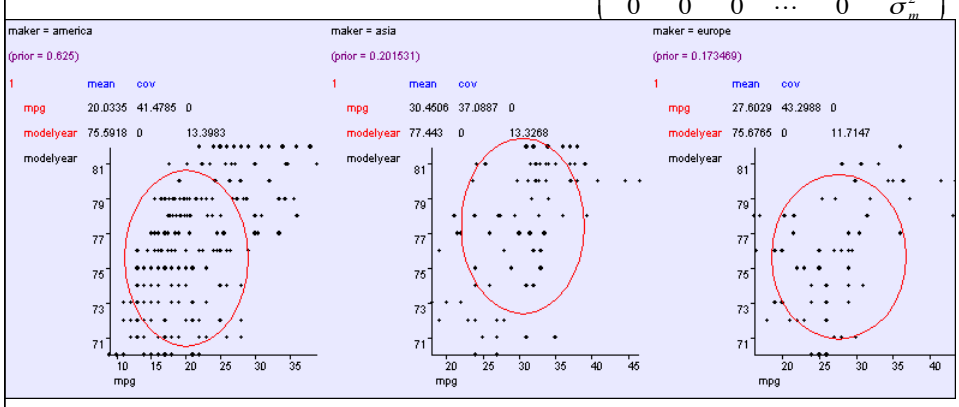
General: $O(m^2)$ parameters

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1m} \\ \sigma_{12} & \sigma_2^2 & \cdots & \sigma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{1m} & \sigma_{2m} & \cdots & \sigma_m^2 \end{pmatrix}$$



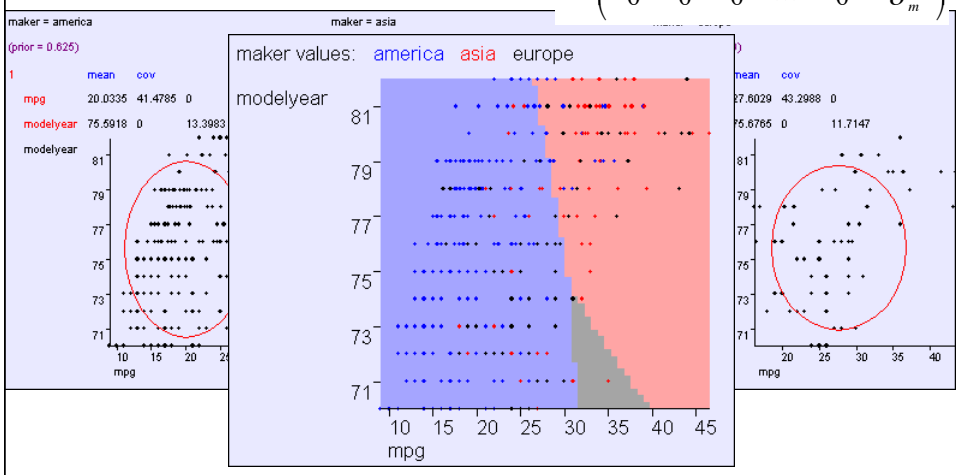
Aligned: $O(m)$ parameters

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2^2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma_3^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_{m-1}^2 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma_m^2 \end{pmatrix}$$



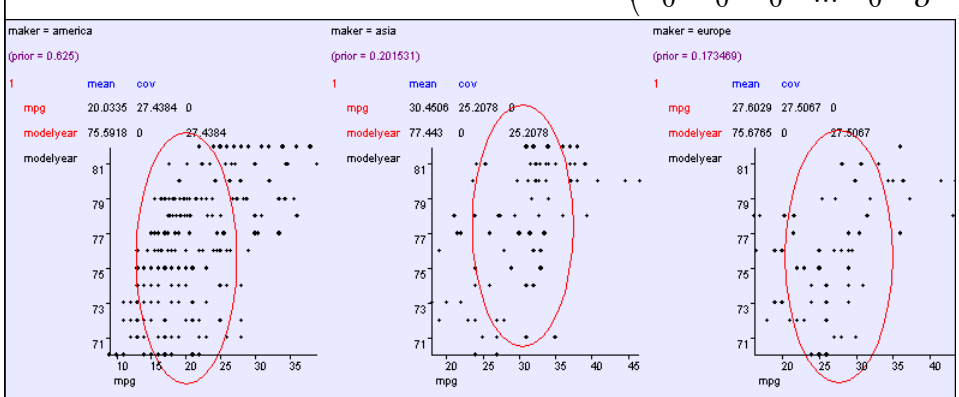
Aligned: *O(m)* parameters

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2^2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma_3^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_{m-1}^2 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma_m^2 \end{pmatrix}$$



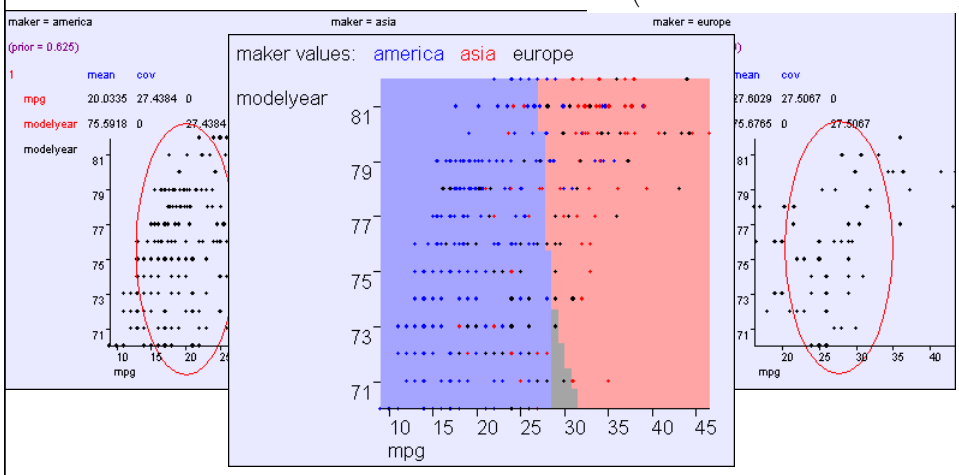
Spherical: *O(1)* cov parameters

$$\Sigma = \begin{pmatrix} \sigma^2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma^2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma^2 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma^2 \end{pmatrix}$$



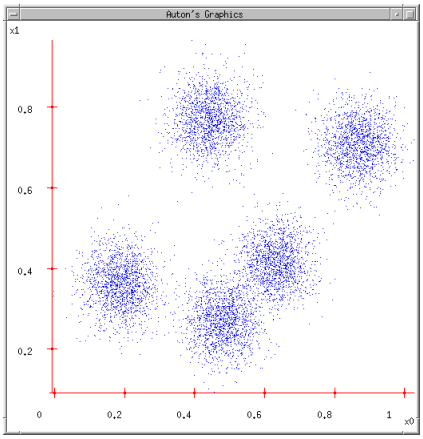
Spherical: *O(1)* cov parameters

$$\Sigma = \begin{pmatrix} \sigma^2 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma^2 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \sigma^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma^2 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \sigma^2 \end{pmatrix}$$

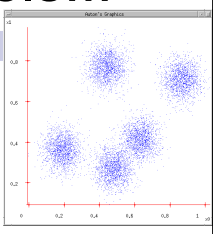


Next… back to Density Estimation

What if we want to do density estimation with multimodal or clumpy data?



But we don't see class labels!!!



- MLE:
  - argmax $\prod_j P(y^j,x^j)$
- But we don't know y^j!!!
- Maximize marginal likelihood:
  - argmax $\prod_j P(x^j)$ = argmax $\prod_j \sum_{i=1}^k P(y^j=i,x^j)$

Special case: spherical Gaussians and hard assignments

$$P(y=i \mid \mathbf{x}^j) \propto \frac{1}{(2\pi)^{m/2} \|\Sigma_i\|^{1/2}} \exp\left[-\frac{1}{2}\left(\mathbf{x}^j-\mu_i\right)^T \Sigma_i^{-1}\left(\mathbf{x}^j-\mu_i\right)\right] P(y=i)$$

- If P(X|Y=i) is spherical, with same σ for all classes:

$$P(\mathbf{x}^j \mid y=i) \propto \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}^j-\mu_i\right\|^2\right]$$

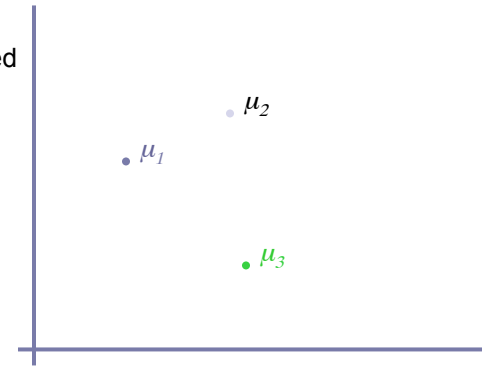
- If each x_j belongs to one class C(j) (hard assignment), marginal likelihood:

$$\prod_{j=1}^m \sum_{i=1}^k P(\mathbf{x}^j, y=i) \propto \prod_{j=1}^m \exp\left[-\frac{1}{2\sigma^2}\left\|\mathbf{x}^j-\mu_{C(j)}\right\|^2\right]$$

- Same as K-means!!!

The GMM assumption

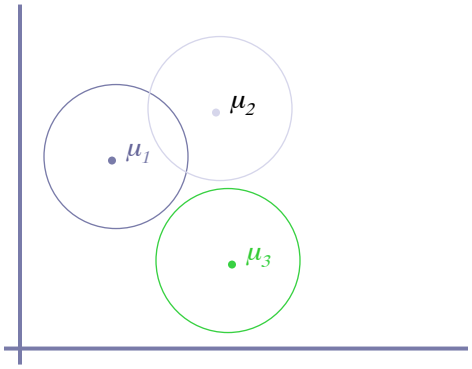
- There are k components
- Component *i* has an associated mean vector μ_i



The GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 I$

Each data point is generated according to the following recipe:

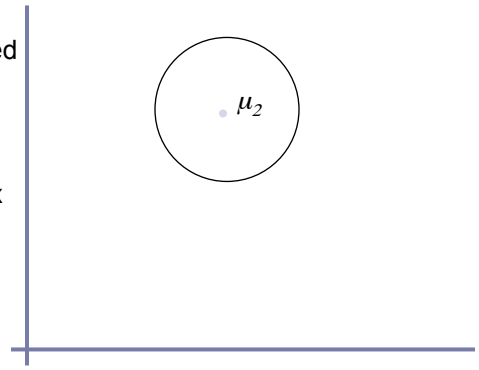


The GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 \boldsymbol{I}$

Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*

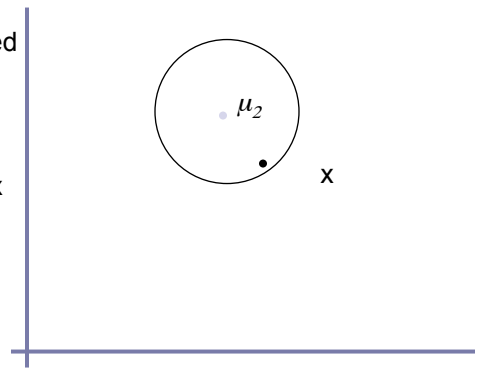


The GMM assumption

- There are k components
- Component *i* has an associated mean vector μ_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix $\sigma^2 \boldsymbol{I}$

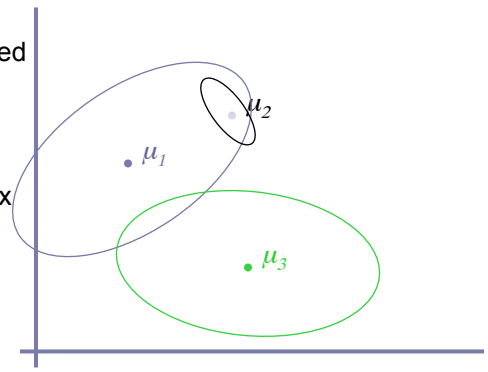
Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*
2. Datapoint $\sim \mathrm{N}(\mu_i, \sigma^2 \boldsymbol{I})$



The General GMM assumption

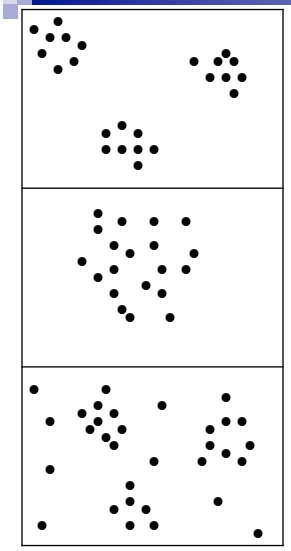
- There are k components
- Component *i* has an associated mean vector m_i
- Each component generates data from a Gaussian with mean m_i and covariance matrix Σ_i



Each data point is generated according to the following recipe:

1. Pick a component at random: Choose component i with probability *P(y=i)*
2. Datapoint ~ N(m_i, Σ_i)

Unsupervised Learning: not as hard as it looks



Sometimes easy

Sometimes impossible

and sometimes in between

*IN CASE YOU'RE WONDERING WHAT THESE DIAGRAMS ARE, THEY SHOW 2-d UNLABELED DATA (**X** VECTORS) DISTRIBUTED IN 2-d SPACE. THE TOP ONE HAS THREE VERY CLEAR GAUSSIAN CENTERS*