Generative Models: Expectation-Maximization

Alexander G. Ororbia II

Introduction to Machine Learning

CSCI-335

4/3/2026

Let Us Finish Building the Explicit-Density, Mixture-of-Gaussians (MoG)

White board time! Crafting a mixture of Gaussians model

Next time! We will build the optimization pillar for the GMM!

Expectation Maximization (EM)

- Training of GMMs with latent variables accomplished via Expectation Maximization
  - **Step 1:** Expectation (**E**-step)
    - Evaluate the “responsibilities” of each cluster with the current parameters
  - **Step 2:** Maximization (**M**-step)
    - Re-estimate parameters using the existing “responsibilities”

- Similar to k-means training

General Form of EM

- Given a joint distribution over observed and latent variables: $p(X, Z|\theta)$
- Want to maximize: $p(X|\theta)$

1. Initialize parameters θ^{old}
2. E Step: Evaluate:

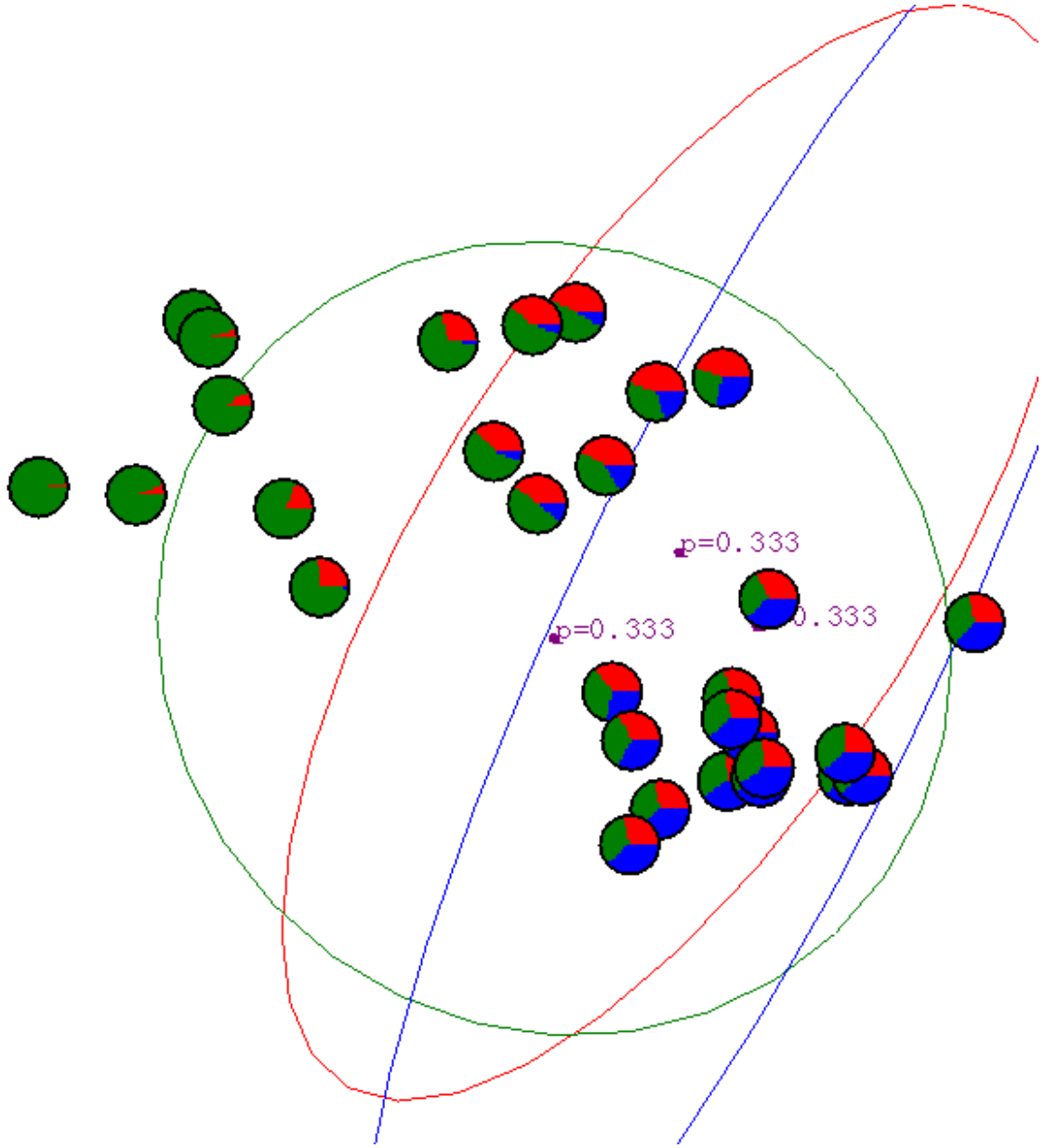
$$p(Z|X, \theta^{old})$$

3. M-Step: Re-estimate parameters (based on expectation of complete-data log likelihood)

$$\theta^{new} = \text{argmax}_\theta \sum_Z p(Z|X, \theta^{old}) \ln p(X, Z|\theta)$$

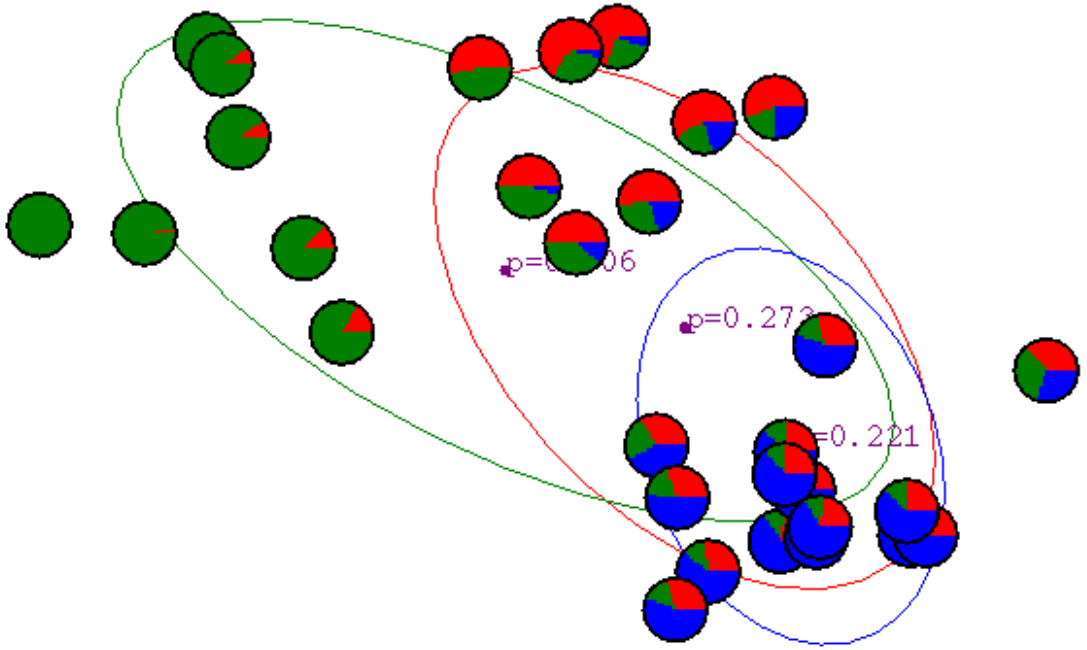
4. Check for convergence of parameters or likelihood

Gaussian Mixture Example: Start

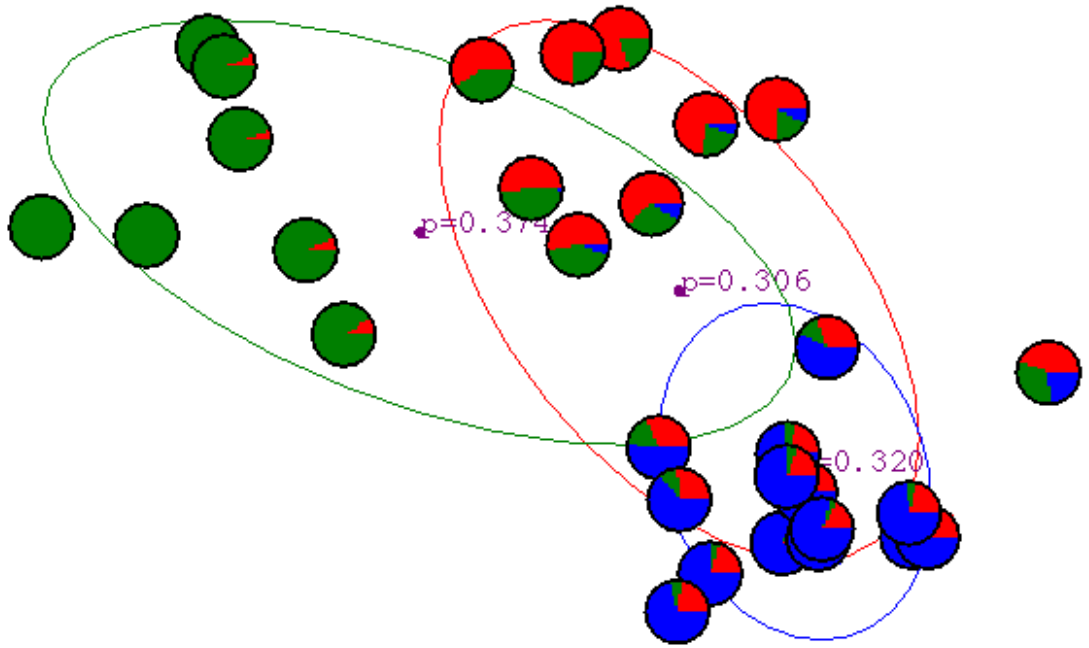


Thanks: Andrew W. Moore

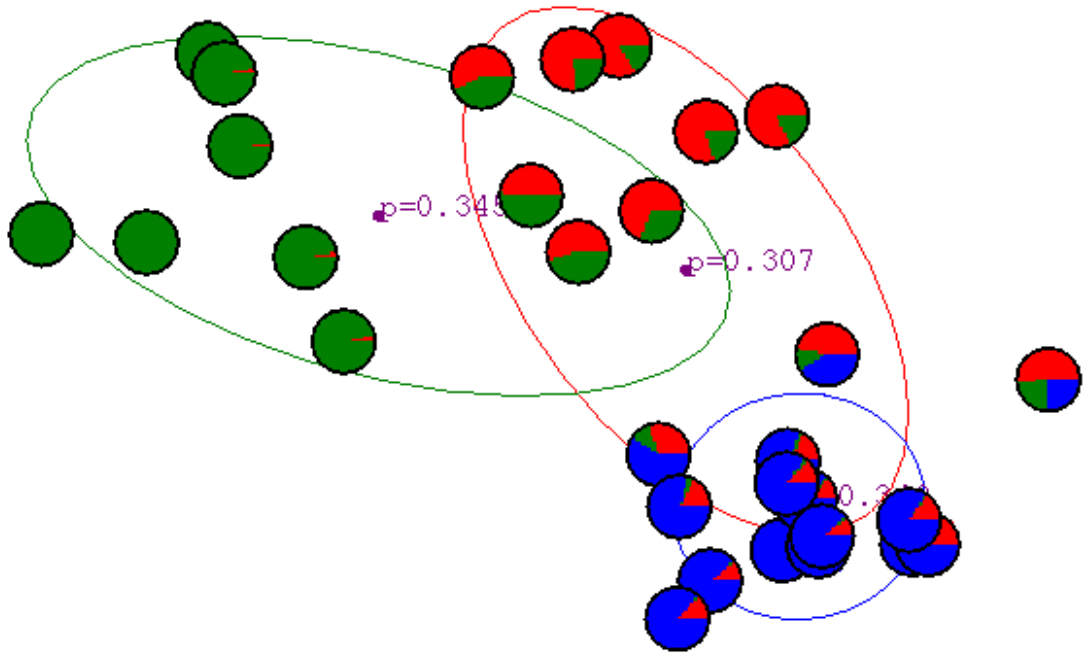
After first iteration



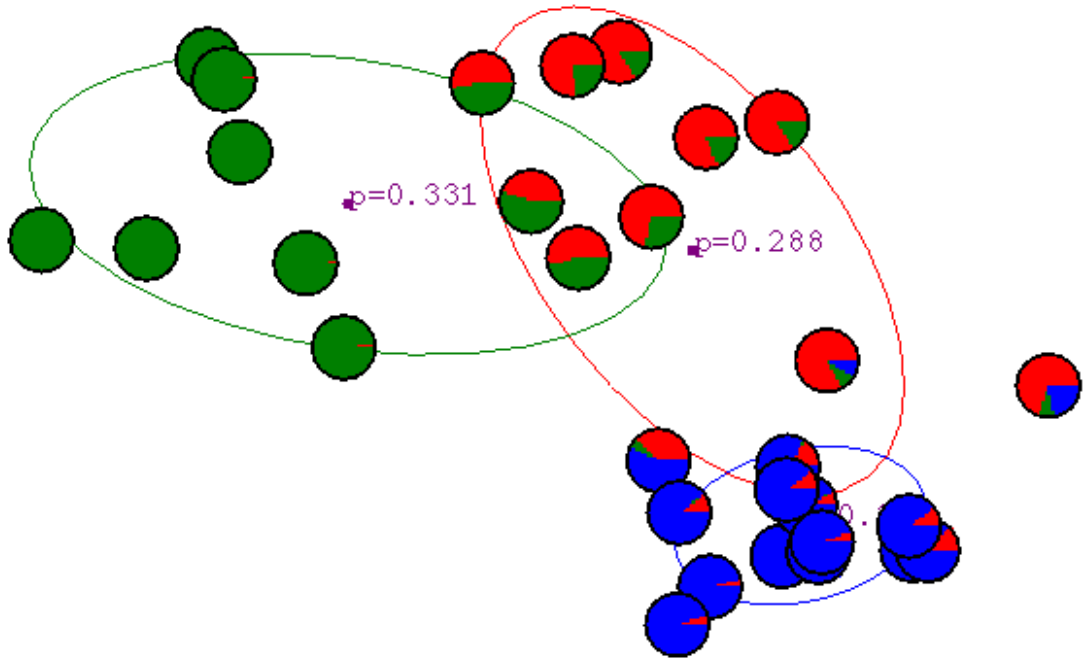
After 2nd iteration



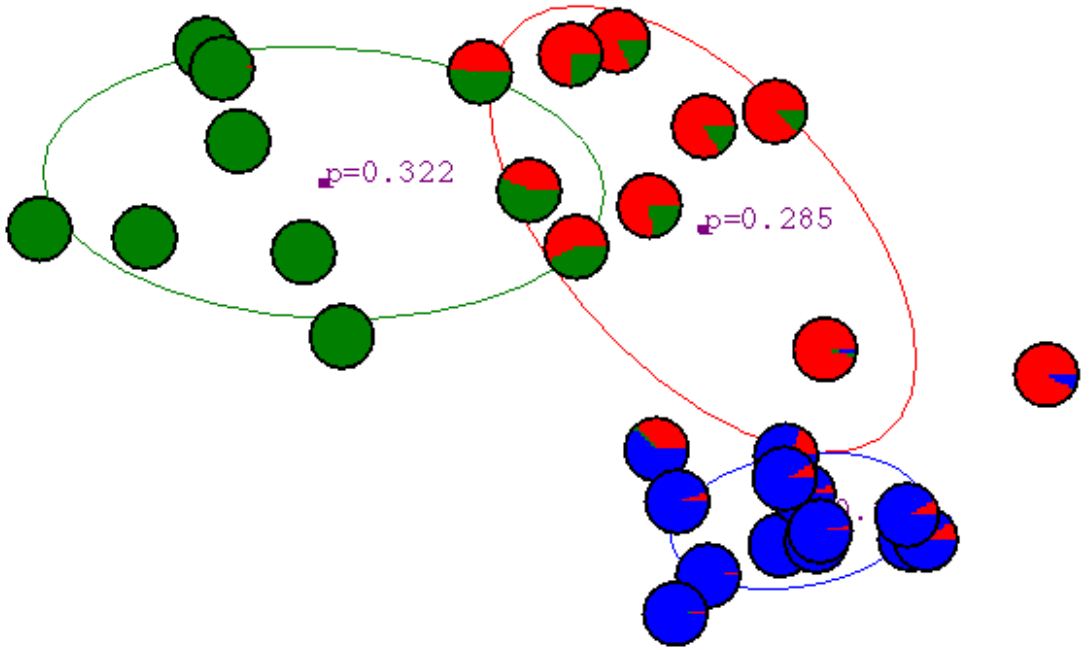
After 3rd iteration



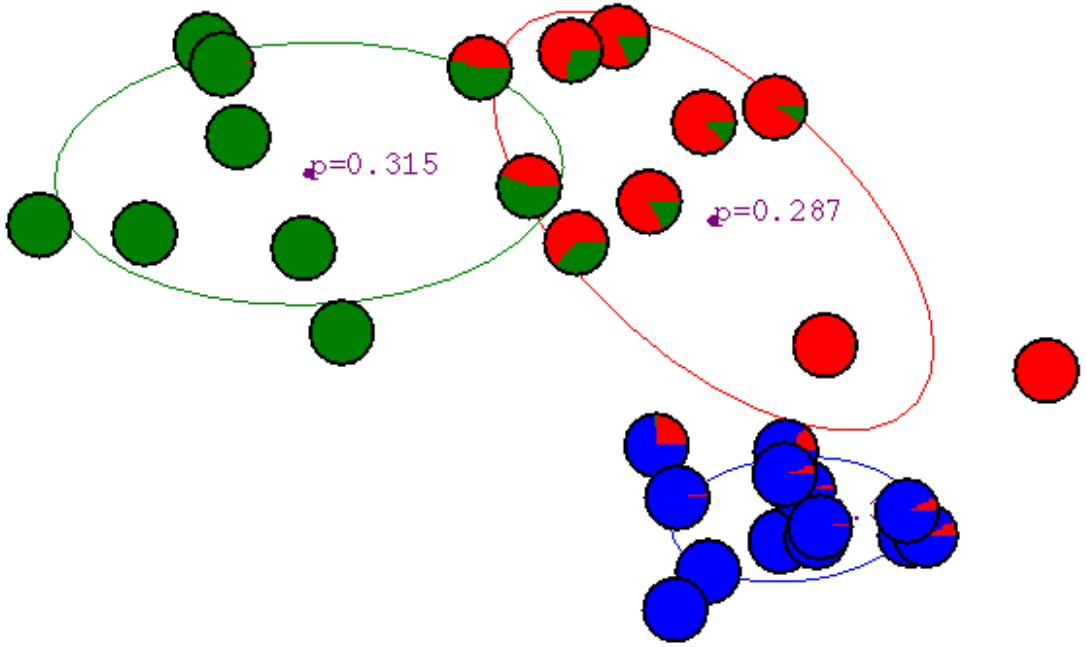
After 4th iteration



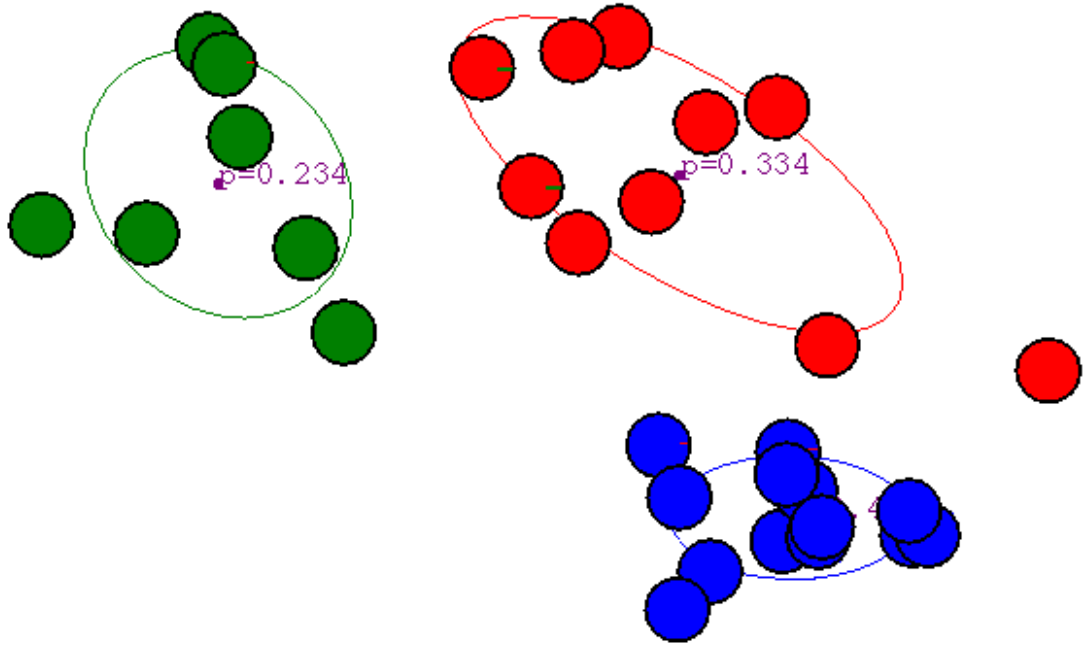
After 5th iteration



After 6th iteration



After 20th iteration



Mixture Distributional Models

- You can construct any mixture model so long as you:
  - Can write down the PMF/PDF of your distribution
  - Can derive/produce MLE estimates of distributional parameters
- Example: Exponential Mixture Model (*EMM*)

Likelihood function:

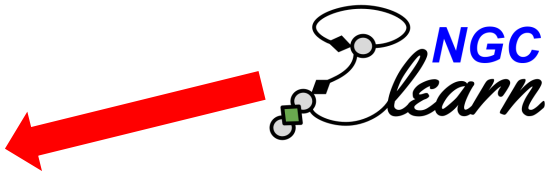
$$\mathcal{L}(X;\theta) = \Pi_{i=1}^{N} \lambda \exp(-\lambda x_i) = \lambda^N \exp\left(-\lambda N \frac{1}{N} \sum_i x_i\right) = \lambda^N \exp\left(-\lambda \sum_i x_i\right)$$

MLE Estimator:

$$\hat{\lambda} = \frac{1}{(1/N)\sum_i \mathbf{x}_i} = \frac{N}{\sum_i \mathbf{x}_i}$$

Mixture Distributional Models

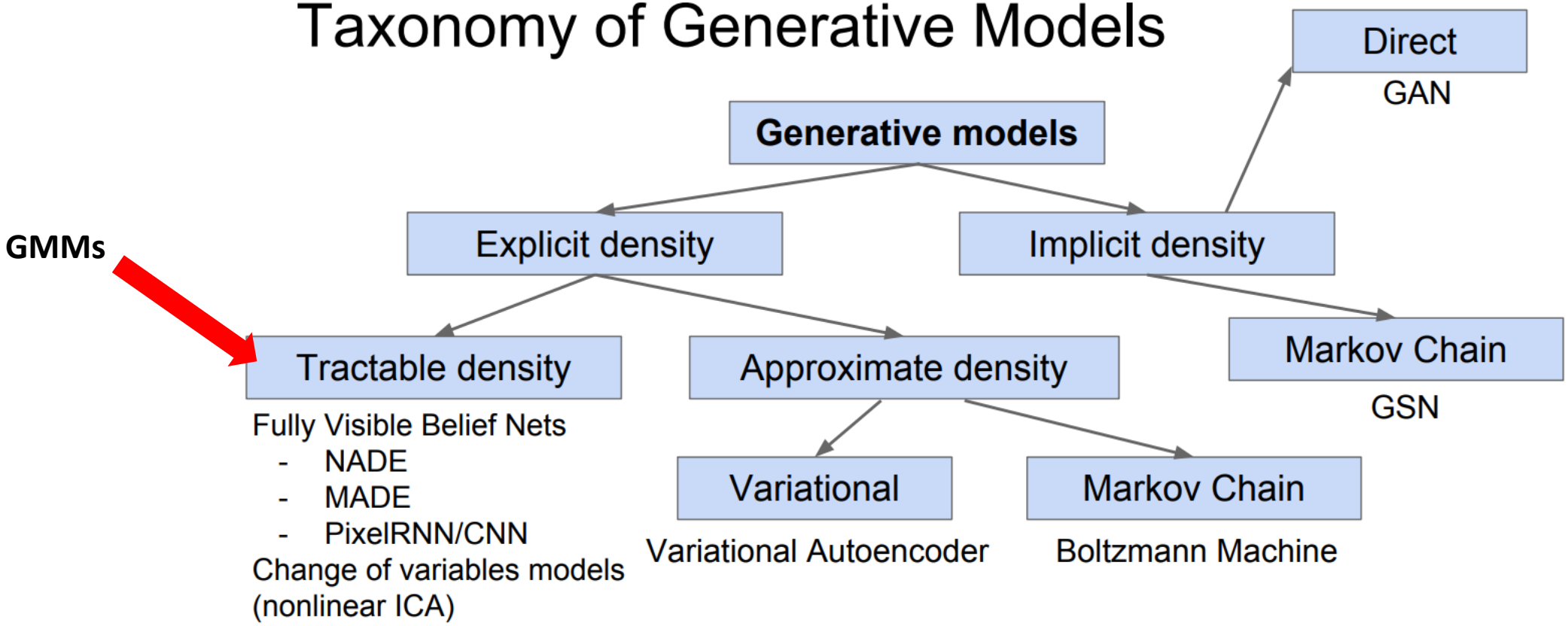
- Other mixture models
  - Bernoulli mixture model (*BMM*)
  - Categorical mixture model (*CMM*) – discrete variables w/ dictionary size *V*
  - Etc., etc.
- There are also Bayesian forms of any mixture model (!)
  - Impose priors over distributional parameters and mixture weights (priors should be conjugate –> prior & posterior in same family, e.g., exponential family)



GMM Applications

- Feature extraction from speech data → speech recognition systems
- Used extensively in object tracking of multiple objects
  - Where # of mixture components & their means predict object locations at each frame in video sequence
  - EM algorithm used to update component means over time as video frames update -> allows object tracking
- And much more!
  - Applications similar to clustering too!

Taxonomy of Generative Models



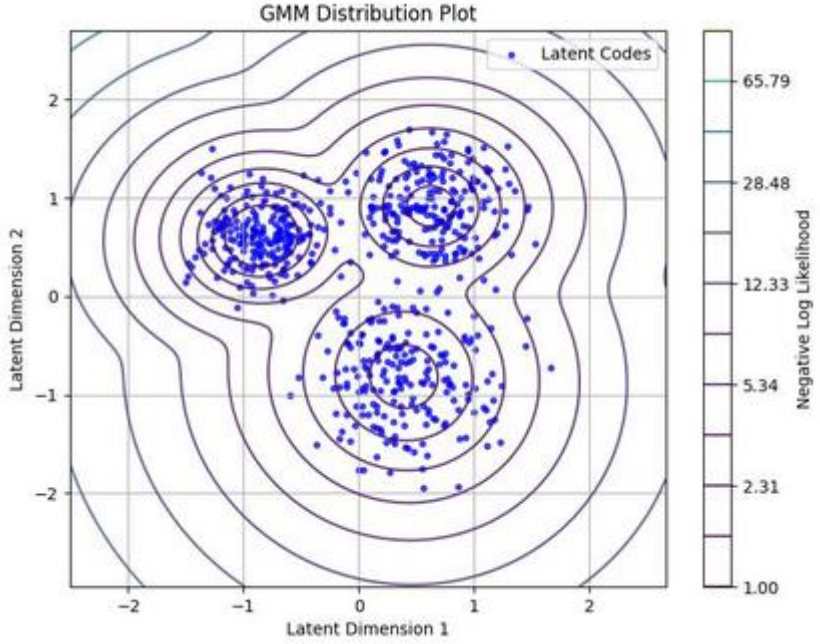
Building a Mixture Models with NGC-Learn

https://github.com/NACLab/ngc-learn
(A computational neuroscience library)

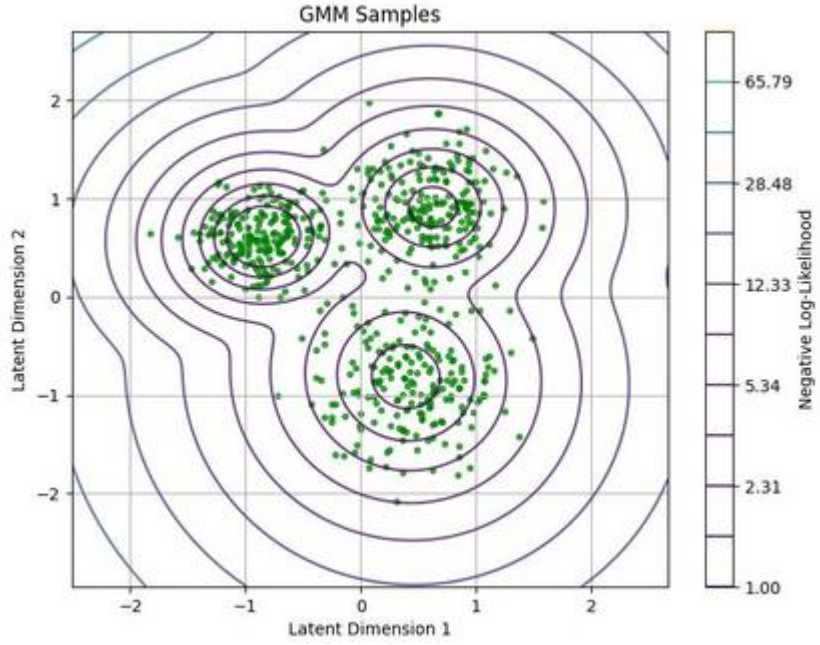
```
from ngclearn.utils.density.gaussianMixture import GaussianMixture as GMM ## pull out density estimator
```

```
n_iter = 100 ## maximum number of iterations to fit GMM to data
n_components = 3 ## number of mixture components w/in GMM
model = GMM(K=n_components, max_iter=n_iter, key=skey[0])
model.init(X) ## initailize the GMM to dataset X
```

```
## estimate GMM parameters over dataset via E-M
model.fit(X, tol=1e-3, verbose=True) ## set verbose to `False` to silence the fitting process
```




```
## Examine GMM samples
Xs = model.sample(n_samples=200 * 3) ## draw 600 samples from fitted GMM
```




```
# Calculate the GMM log likelihood
_, logPX = model.calc_log_likelihood(X) ## 1st output is log-likelihood per data pattern
print(f"log[p(X)] = {logPX} nats")
```

```
log[p(X)] = -1060.006591796875 nats
```

QUESTIONS?