Finite Mixture Models

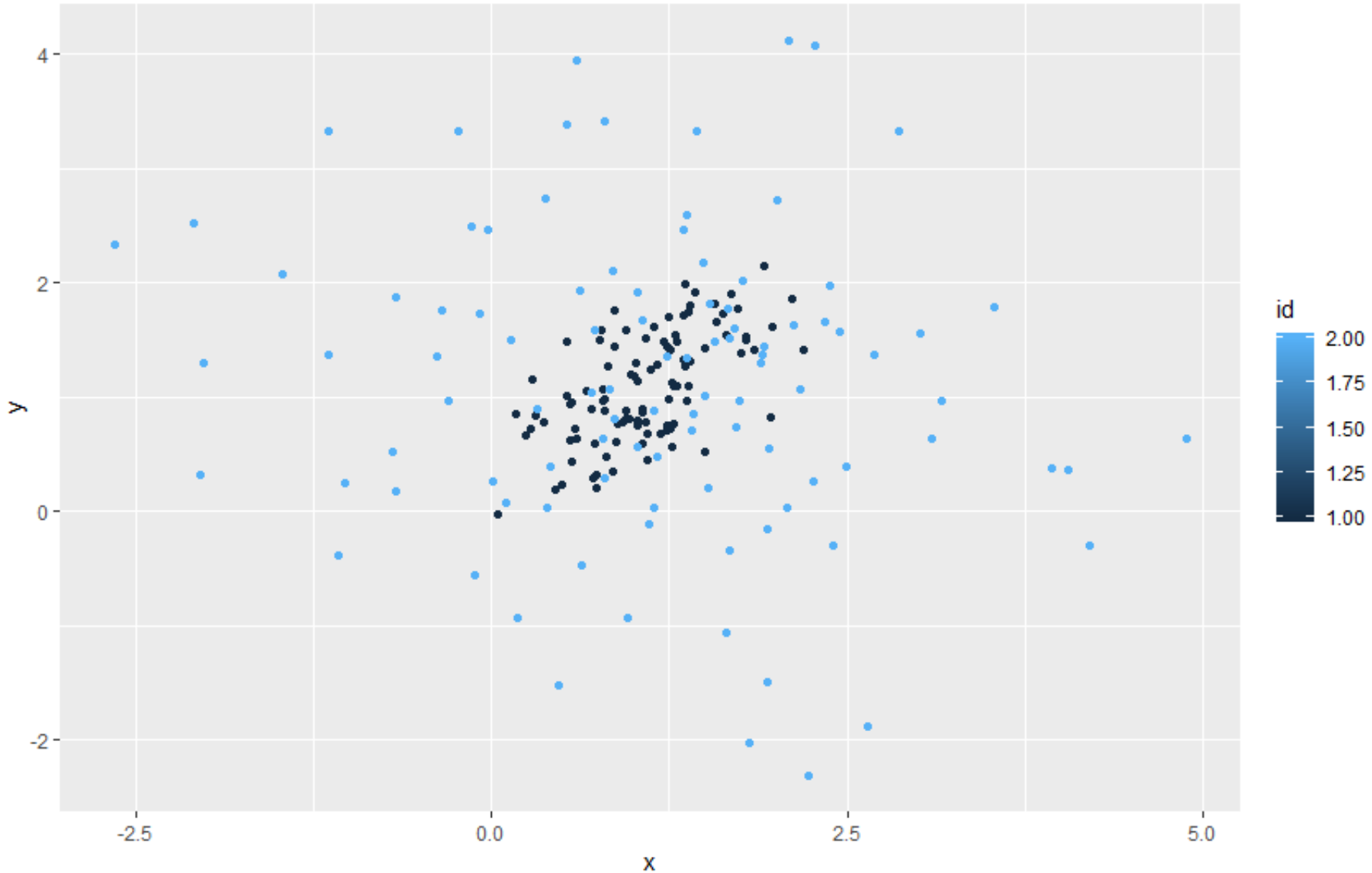
(Probability Based Clustering)

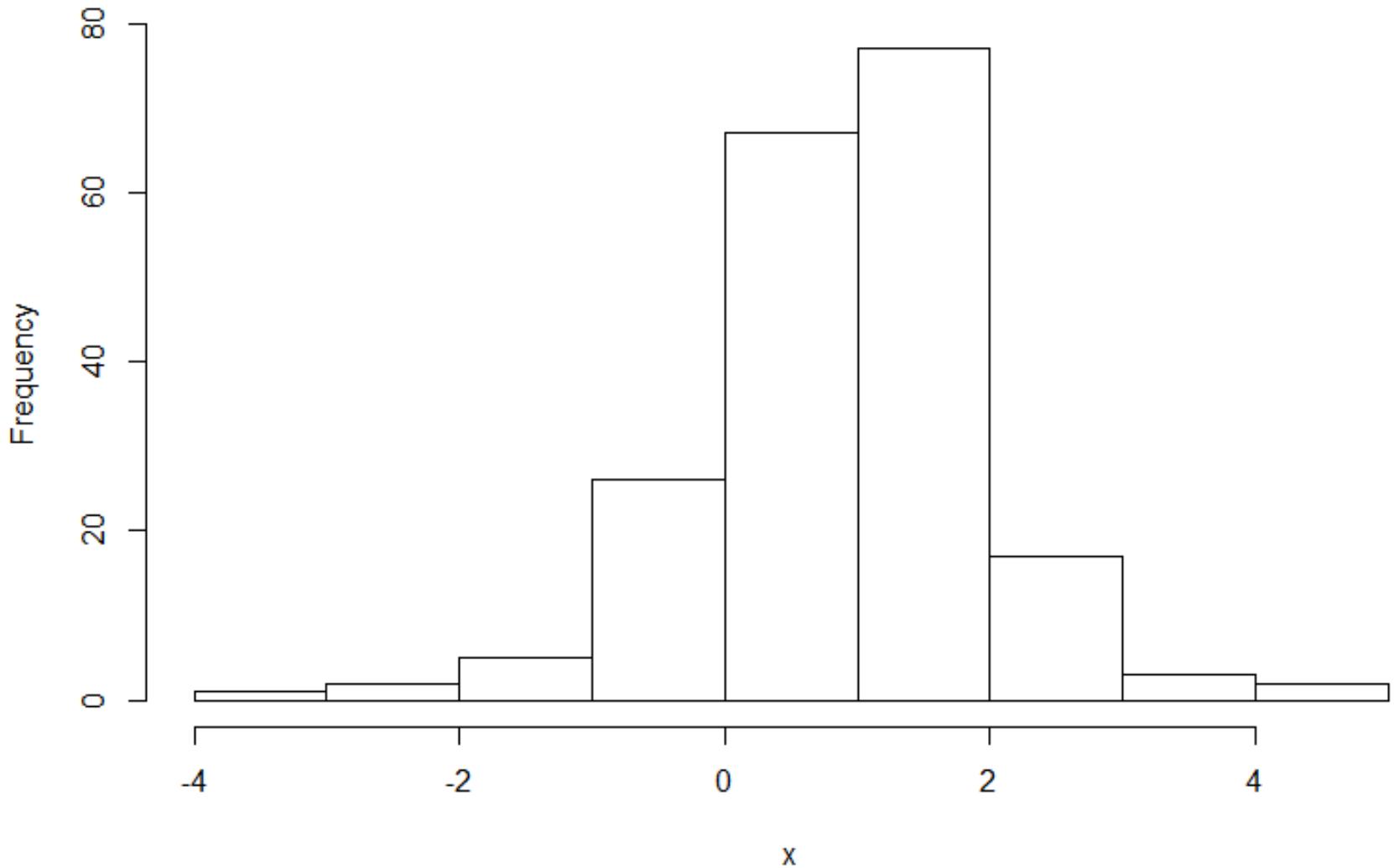
Motivation

Up until this point, we have dealt with hard clustering.

Something is either in or not in a given cluster.

> This methodology works well, but does not necessarily always make sense -> What do we do when we have one tight cluster and one dispersed cluster?

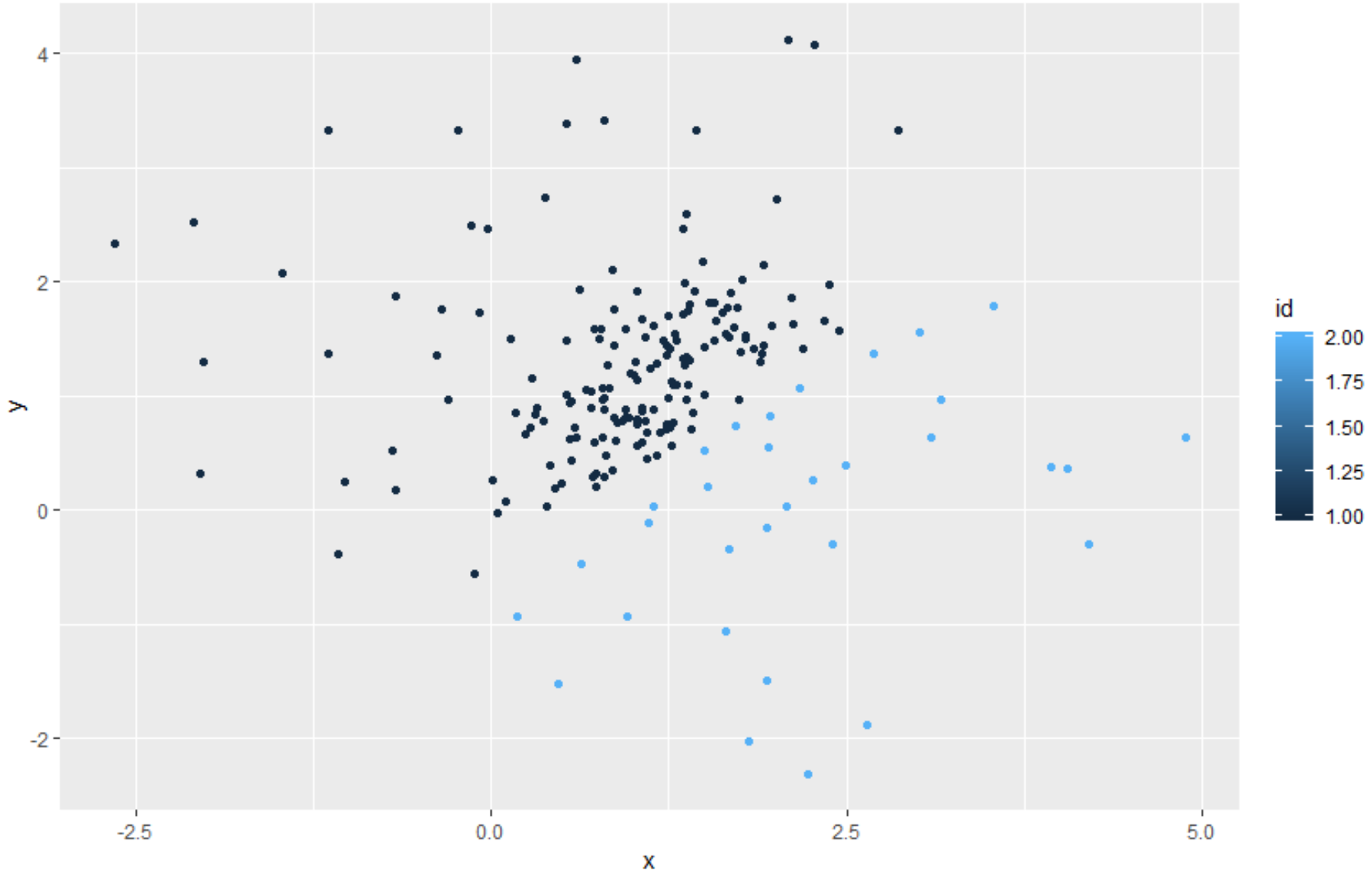

4
2
0
-2
y
-2.5
0.0
2.5
5.0
x
id
2.00
1.75
1.50
1.25
1.00


Frequency
0
20
40
60
80
-4
-2
0
2
4
x

Two clusters one embedded within another…

And kmeans won’t do anything to help!!

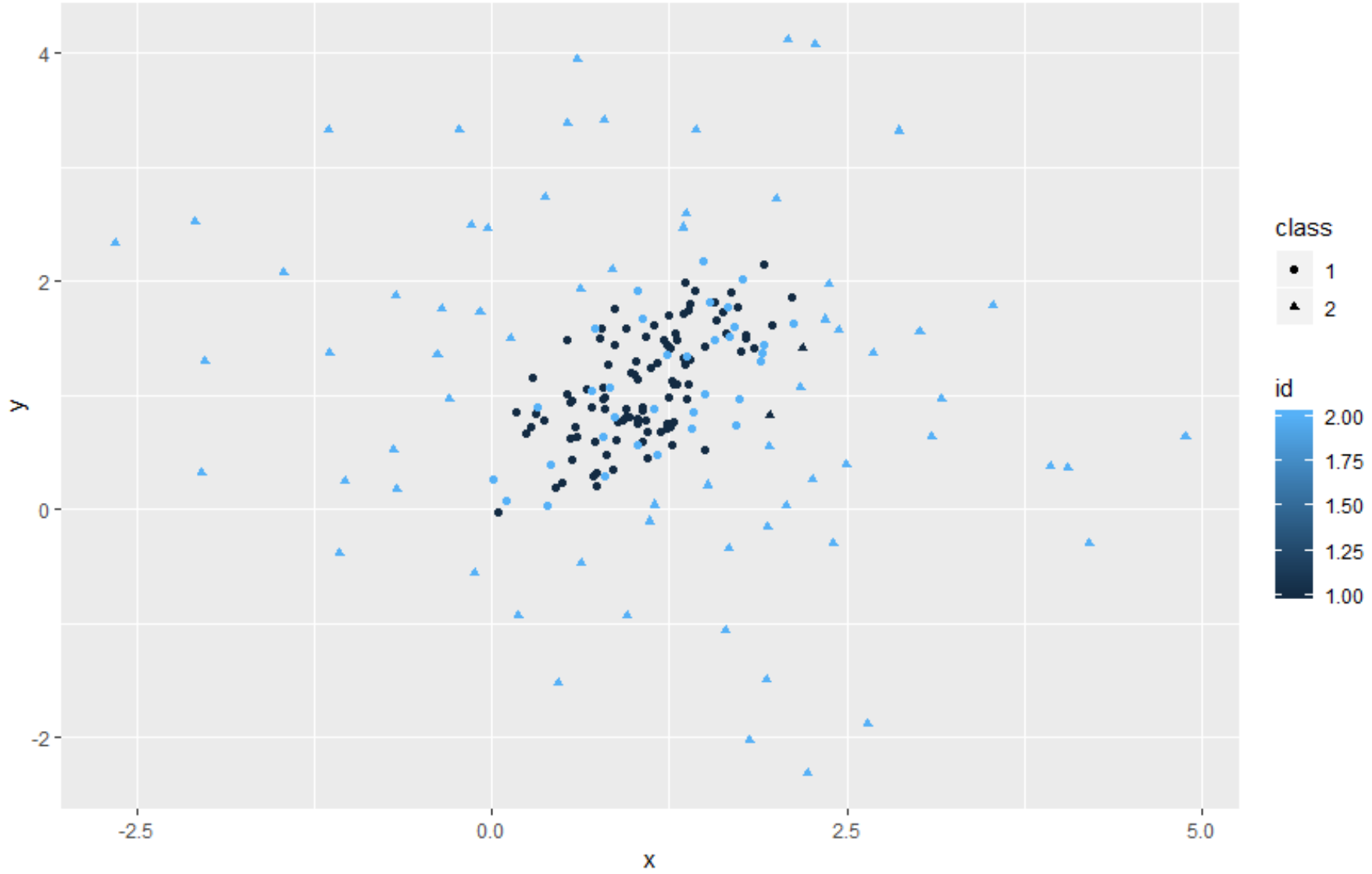
With kmeans



Idea: A mixture model assumes data comes from a probability distribution that has a bunch of components:

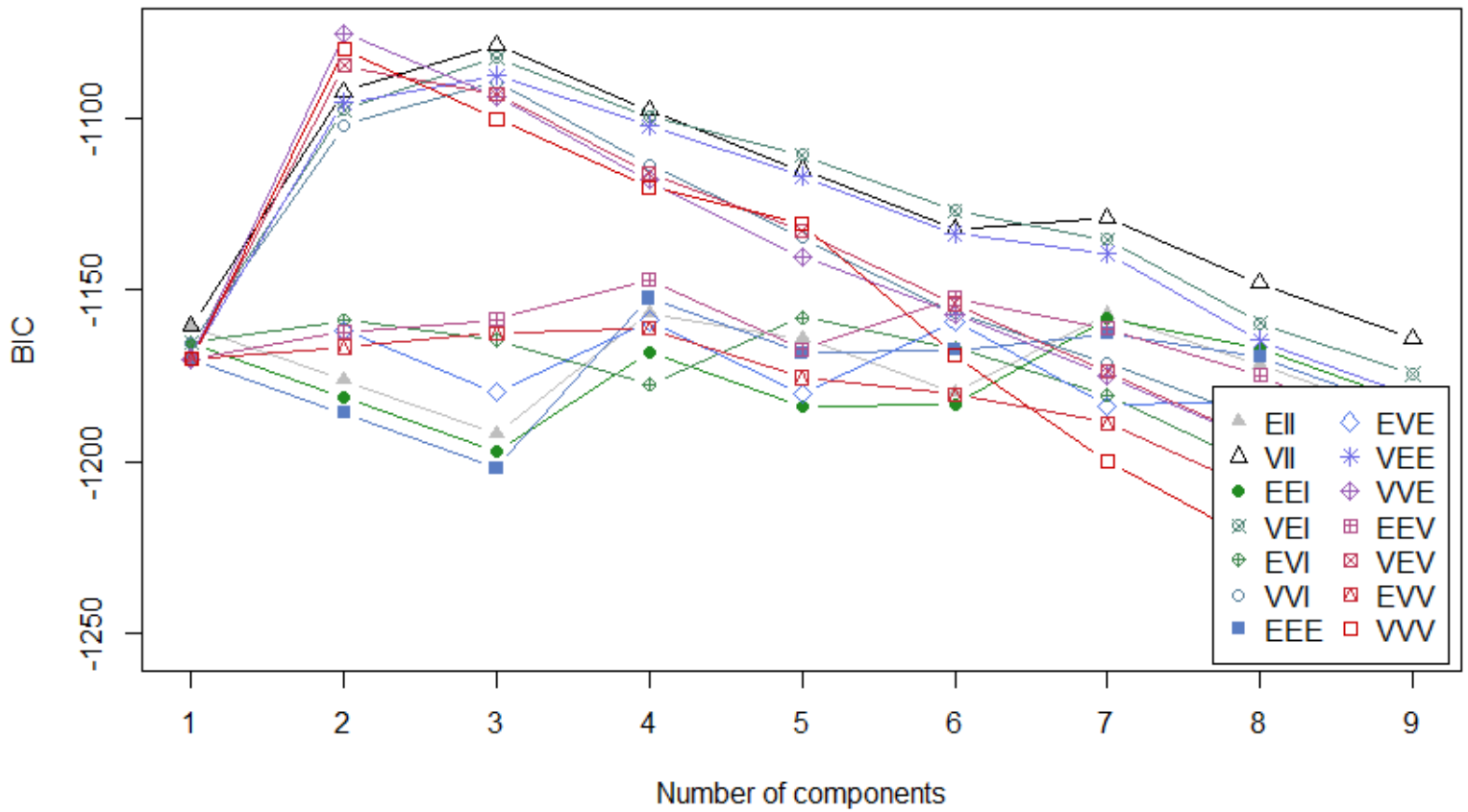
- Each component represents a ‘different’ population.
- Within the population there is process that generates observations.
- Observations are drawn from this process.

Mixture Modeling



But wait there’s more…

We can choose the number of components mathematically… through model selection



And we can automatically compute the centroids and variance …

We compute these as a form of a mixture of distributions. In this case, we compute them as a mixture of multivariate normal distributions.

So what is the Multivariate Normal distribution?

On the Multivariate normal distribution

It is a generalization of the normal distribution; here, each individual variable is a normal variable but it accounts for the covariance between different variables….

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

A bunch of INDEPENDENT NORMALS X, Y

$$g(x)g(y) = \frac{1}{\sigma_x \sigma_y 2\pi} \exp\left[-\frac{(x-\mu_x)^2}{2\sigma_x^2} - \frac{\left(y-\mu_y\right)^2}{2\sigma_y^2} \right]$$

Or in DREADED matrix form…

$$g(x)g(y) = \frac{1}{\sigma_x \sigma_y 2\pi} \exp\left[-0.5\left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}\right)' \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}^{-1} \left(\begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}\right)\right]$$

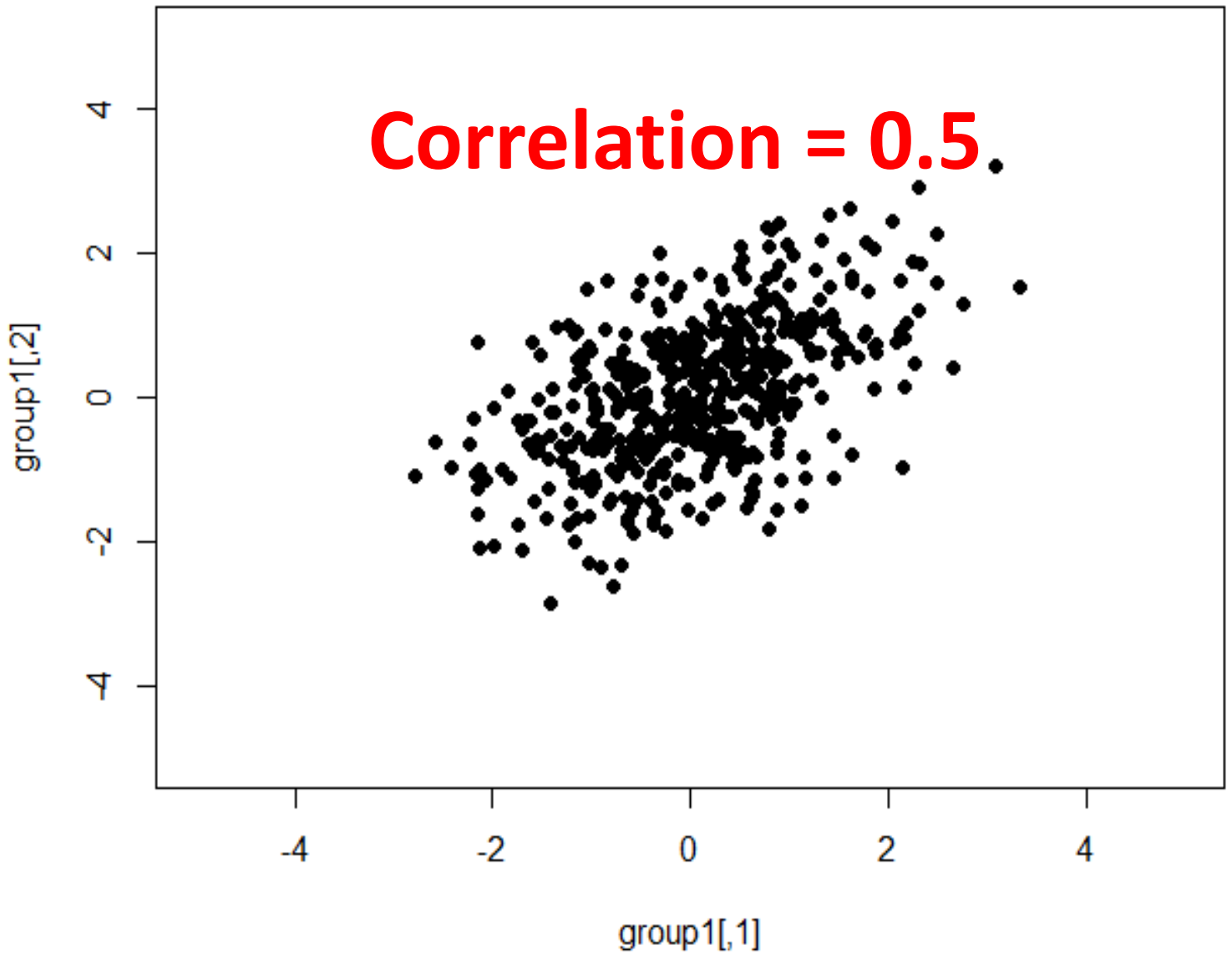
How is this related to Euclidean distance?

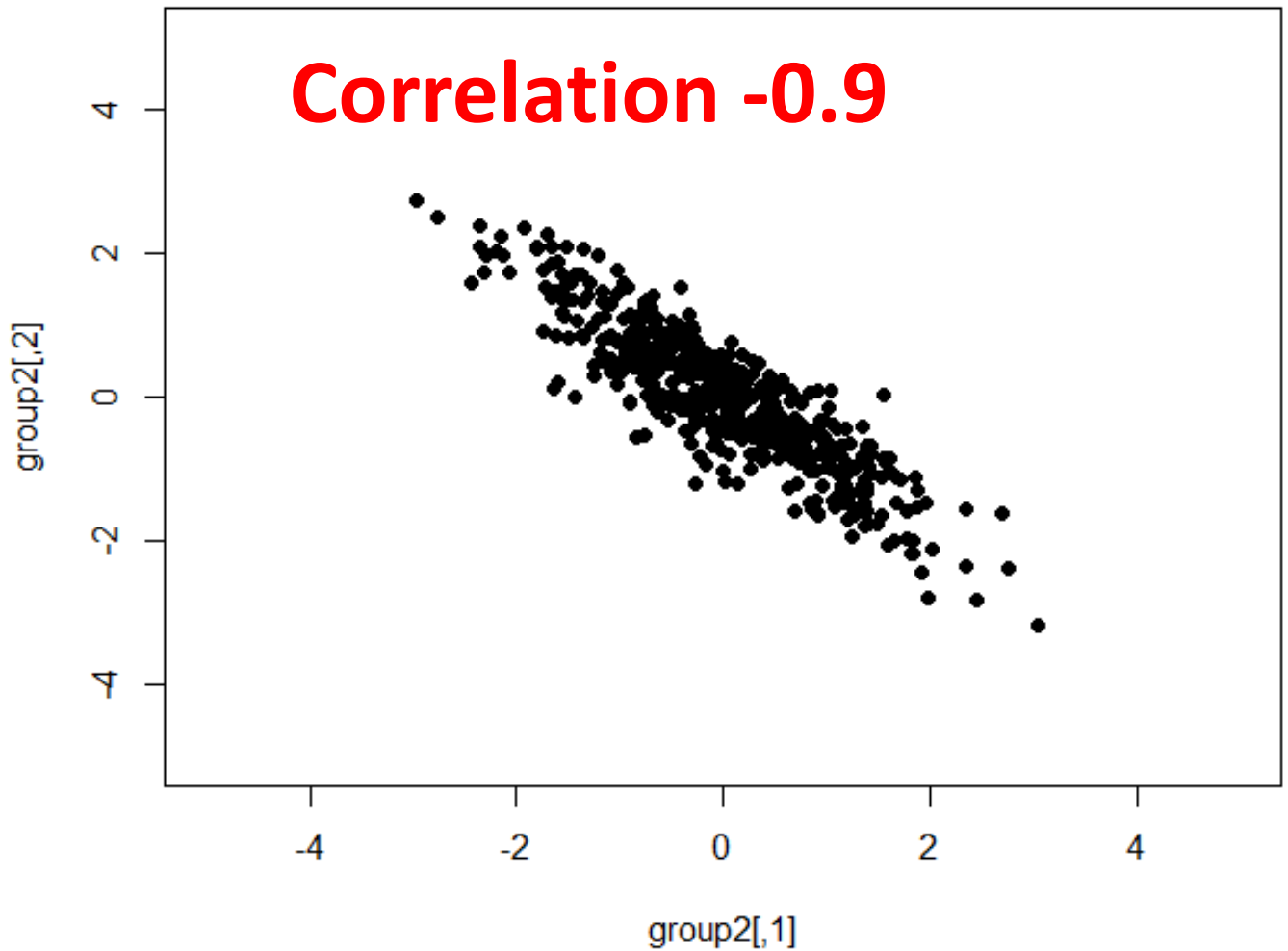
The inverse of the covariance matrix Σ

So really we just have a “normal distribution” that accounts for the covariance between variables.

Think about the function cov(x) in R.

This matrix determines the cigar shape of the distribution


Correlation = 0.5
group1[,2]
4
2
0
-2
-4
-4
-2
0
2
4
group1[,1]


Correlation -0.9
group2[,2]
4
2
0
-2
-4
-4
-2
0
2
4
group2[,1]

So the IDEA is the data we observe come from a bunch of these "multivariate normal" distributions/populations, and we can fit a bunch of these distributions to the observed data. Key: Each population has a probability of membership. We label these the mixing probabilities $(\pi_1, \pi_2 \ldots \pi_n)$, such that $\sum_{i=1} \pi_i = 1$.

Think of it this way.

I can pull out a random person from the population.

There is about a 50% chance that person is Female

There is a 50% chance that person is Male

If the person is Female, there is a distribution of heights and weights

If the person is Male, there is a DIFFERENT distribution of heights and weights.

Based solely on the (height and weight), I can guess if that person is male or female.

This is a mixture distribution

$$g(X) = \pi_1 g_{male}(X|\mu_{male}, \Sigma_{male}) + \pi_2 g_{female}(X|\mu_{female}, \Sigma_{female})$$

Each individual distribution DEFINES our clusters. Now this is soft because there is NEVER 100% probability an observation falls into one cluster or another. Instead, we define the probability an observation falls into a cluster as:

$$pr(X \in C_j) = \frac{\pi_{c_j} g_{c_j}(X)}{\sum_{i=1} \pi_{c_i} g_{c_i}(X)}$$

We then cluster based upon some THRESHOLD probability of membership.

This is totally DIFFERENT than our previous APPROACHES!

Even though ‘DISTANCE’ is defined similarly.

What is the difference in our version of EUCLIDEAN distance now and that of simple kmeans???

Ok let’s slow down even further…

This is the foundation of much of modern machine learning, including unsupervised learning … Latent Dirichlet Allocation (LDA)... Is just a super fancy (in)finite mixture model

Chalk Board and computer time…

Based upon what I write and questions you ask, I will update the notes accordingly.