Data Science for Economists

Lecture 15: Unsupervised Learning

Drew Van Kuiken
University of North Carolina | ECON 370

Table of contents

Soft Clustering: The EM Algorithm

Soft Clustering

Notice that $\mathbf{K}$-means performs poorly when there's lots of overlap in the data.

Idea: What if we model the clustering and assign probabilities to the group?

Downsides: Must specify a model for the data to be generated from.

Upsides: Can assign probabilities.

The rest of the lecture, we will be using the following example to motivate the method:

Differences in height (and weight) by sex

Height Differences by Sex: NHIS Data

From the CDC's website, you can download data from the National Health Interview Survey.

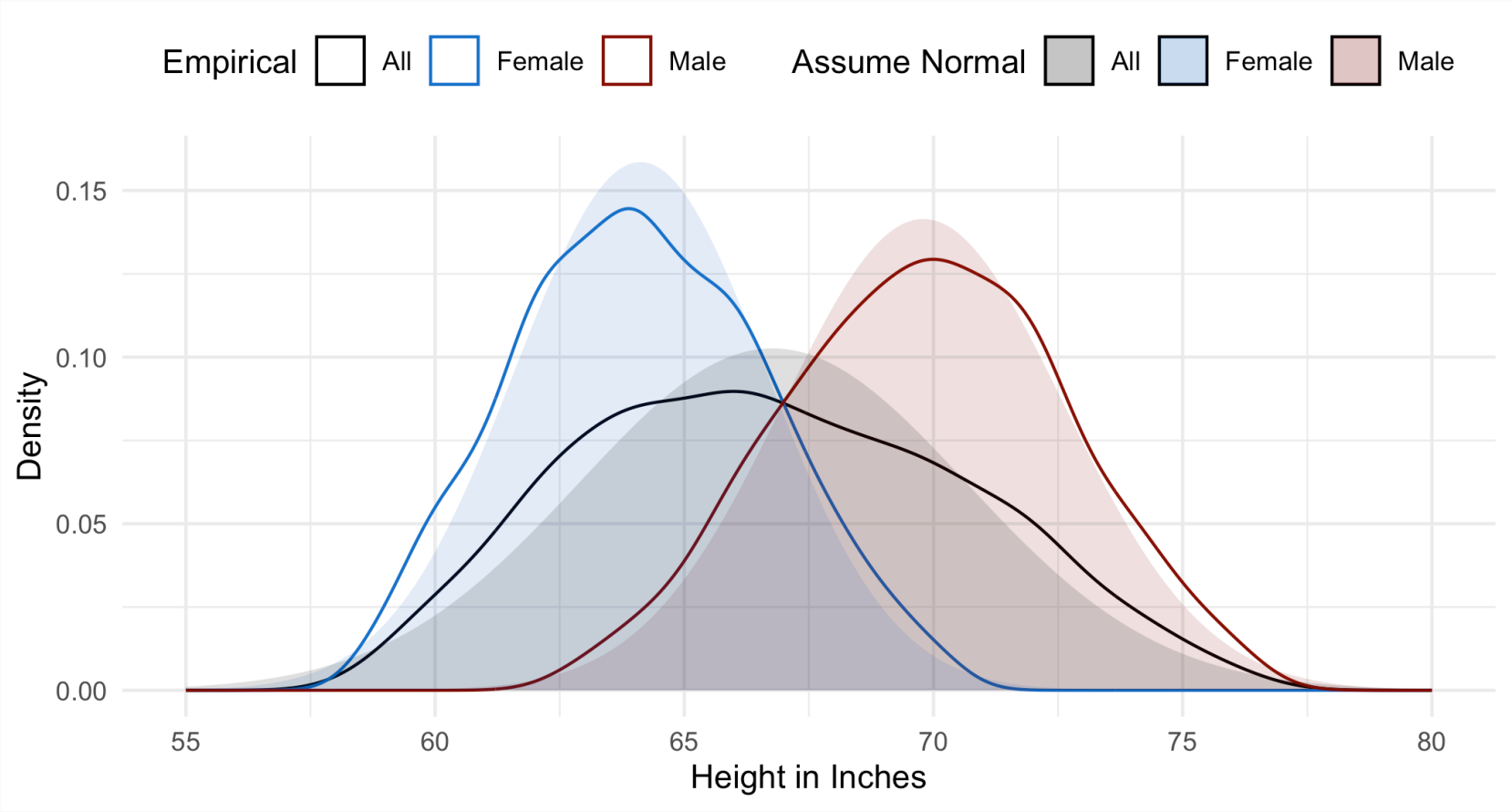
We will be using the adult 2019 data.

This is real data gathered from the US population regarding health.

```
data_path = "/Users/drewvankuiken/Dropbox/ECON370/local/lec15/"
NHIS_data = fread(paste0(data_path,"adult19.csv"))
NHIS_data = NHIS_data[SEX_A<7]    #remove responses not male or female
setnames(NHIS_data,c("HEIGHTTC_A","WEIGHTLBTC_A"),c("height","weight"))
NHIS_data = NHIS_data[height<96]  #remove non-numerical responses
NHIS_data = NHIS_data[weight<900] #remove non-numerical responses
NHIS_data[,sex:="Male"]; NHIS_data[SEX_A==2,sex:="Female"] # create sex var
NHIS_plot_data = copy(NHIS_data); NHIS_plot_data[,sex:="All"]
NHIS_plot_data = rbind(NHIS_plot_data[,.(sex,height,weight)],
                       NHIS_data[,.(sex,height,weight)])
hght_moms = NHIS_plot_data[,list("mu"=mean(height),"var"=var(height)),by=sex]
hght_moms[order(sex,decreasing = T),]
```

```
##       sex       mu       var
##    <char>    <num>     <num>
## 1:   Male 69.79632  7.953308
## 2: Female 64.11915  6.332597
```

Distribution of Height by Sex



Each empirical distribution (i.e. the distribution of the data) by sex looks normal!

- The "total" distribution is a *mixture* of males and females.

Model for Sex Differences in Height

It seems reasonable to assume that height is normally distributed conditional on sex where males and females have different distributions.

- If sex is labeled in the data, then can simply group on sex.

What if we don't observe someone's sex? How can we model this?

- To be clear, this is a thought experiment. There is a variable for sex in the data.
- This is to show you what to do if we didn't.

Simple Statistical Model for Height

- Person $\mathbf{i}$ is assigned $\mathbf{M}$ with probability α and $\mathbf{F}$ with probability $1 - \alpha$
- $\mathbf{s_i} \in \{\mathbf{M}, \mathbf{F}\}$ and $\mathbf{s_i}$ is person $\mathbf{i}$'s sex.
- Assume height $\mathbf{h_i}$ is drawn from $\mathbf{N}(\mu_{s_i}, \sigma^2_{s_i})$
  - Each person's height is drawn from a distribution that depends upon their sex
- This is called a Normal (or Gaussian) Mixture Model
  - The entire population's height distribution is a mixture of two normals where each sex has its own distribution.

Given a sample of $(\mathbf{h_i})_{i=1}^{N}$, how can we estimate $\mu_M, \mu_F, \sigma^2_M, \sigma^2_F$, and α?

Expectation-Maximization (EM)

An algorithm called the EM algorithm can be used to estimate the parameters.

- Today we discuss it in the context of a Gaussian Mixture Model, but it can be used for any model where one variable is a mixture of different groups and the group labels are not observed.

There are two steps:

1. **The Expectation Step**: calculate probability each observation belongs a group.
2. **The Maximization Step**: re-estimate the parameters given the new probabilities.

This requires something called Bayes' Rule.

Bayes' Rule and EM

Denote $\mathrm{P}(\mathrm{M}|\mathrm{h_i}) = \mathrm{P}(\mathrm{sex_i} = \mathrm{M}|\mathrm{height} = \mathrm{h_i})$ i.e. the prob i is male *given* i's height $\mathrm{h_i}$.

Bayes's rule states:

$$\mathrm{P}(\mathrm{M}|\mathrm{h_i}) = \frac{\mathrm{P}(\mathrm{h_i}|\mathrm{M})\mathrm{P}(\mathrm{M})}{\mathrm{P}(\mathrm{h_i}|\mathrm{M})\mathrm{P}(\mathrm{M}) + \mathrm{P}(\mathrm{h_i}|\mathrm{F})\mathrm{P}(\mathrm{F}))}$$

- $\mathrm{P}(\mathrm{M}) = \alpha$ and $\mathrm{P}(\mathrm{F}) = 1 - \alpha$ are the baseline probs of being M/F.
- $\mathrm{P}(\mathrm{h_i}|\mathrm{M})$ is the "probability" of height $\mathrm{h_i}$ if we **knew** the individual was male. Since our model assumes $\mathrm{h_i} \sim \mathrm{N}(\mu_\mathrm{M}, \sigma^2_\mathrm{M})$ if male, we know $\mathrm{P}(\mathrm{h_i}|\mathrm{M}) = \varphi(\mathrm{h_i}; \mu_\mathrm{M}, \sigma^2_\mathrm{M})$
  - That is φ is the pdf of the normal distribution; the `dnorm` function in `R`.

$$\mathrm{P}(\mathrm{M}|\mathrm{h_i}) = \frac{\varphi(\mathrm{h_i}; \mu_\mathrm{M}, \sigma^2_\mathrm{M})\alpha}{\varphi(\mathrm{h_i}; \mu_\mathrm{M}, \sigma^2_\mathrm{M})\alpha + \varphi(\mathrm{h_i}; \mu_\mathrm{F}, \sigma^2_\mathrm{F})(1 - \alpha)}$$

If we knew $(\mu_\mathrm{M}, \mu_\mathrm{F}, \sigma^2_\mathrm{M}, \sigma^2_\mathrm{F}, \alpha)$, we could compute $\mathrm{P}(\mathrm{M}|\mathrm{h_i})$.

However, we need $\mathrm{P}(\mathrm{M}|\mathrm{h_i})$ to estimate $(\mu_\mathrm{M}, \mu_\mathrm{F}, \sigma^2_\mathrm{M}, \sigma^2_\mathrm{F}, \alpha)$.

The EM algorithm allows us to do both by alternating between the E and M steps.

EM Algorithm

Denote $\theta_n = (\mu_{M,n}, \mu_{F,n}, \sigma^2_{M,n}, \sigma^2_{F,n}, \alpha_n)$ as the parameter estimates after n iterations.

Step 0: Initialize θ_0 somehow.

- The easiest way to do this is to run K-means.
- Use the resulting groups from K-means to calculate group means, variances, and proportions.
- Assume males are the group with the larger mean height.
- This gives us initial guesses of $\theta_0 = (\mu_{M,0}, \mu_{F,0}, \sigma^2_{M,0}, \sigma^2_{F,0}, \alpha_0)$.

Step 1:

- **E-Step**: Calculate $P(M|h_i; \theta_0) = \pi_1^M(h_i)$ using Bayes' Rule.
  - Note $\pi_1^F(h_i) = 1 - \pi_1^M(h_i)$
- **M-Step**: Calculate θ_1 given $\pi_1^M(h_i)$ and the data. Formulas on next slide.
  - These are basically just weighted versions of the sample mean and sample variance where the weights are $\pi_1^M(h_i)$ and $\pi_1^F(h_i)$ for males and females respectively.

Step n: Continue alternating between the E-Step and M-Step until $||\theta_n - \theta_{n-1}||_p < \varepsilon$

Normal Mixture Formulas

$$N_M = \sum_{i=1}^{N} \pi_n^M(h_i)$$

$$N_F = N - N_M$$

$$\mu_{M,n} = \frac{1}{N_M} \sum_{i=1}^{N} \pi_n^M(h_i) h_i$$

$$\mu_{F,n} = \frac{1}{N_F} \sum_{i=1}^{N} \pi_n^F(h_i) h_i$$

$$\sigma_{M,n}^2 = \frac{1}{N_M} \sum_{i=1}^{N} \pi_n^M(h_i)(h_i - \mu_{M,n})^2$$

$$\sigma_{F,n}^2 = \frac{1}{N_F} \sum_{i=1}^{N} \pi_n^F(h_i)(h_i - \mu_{F,n})^2$$

$$\alpha_n = \frac{N_M}{N}$$

Functions for Parameter Estimation

The formulas on the last slides are essentially weighted means and variances

- The weights are the probability of belonging to the group.

```
## --- Function for weighted mean
mean_wgt = function(x,ws) sum(ws*x)/sum(ws)
## --- Function for weighted variance
var_wgt = function(x,ws) sum(ws*(x-mean_wgt(x,ws))^2)/sum(ws)
## --- Function for weight sample proportion
prop_wgt = function(ws) sum(ws)/length(ws)
```

EM Algorithm: Set-Up

Use K-means with $\mathbf{K = 2}$ to classify observations into two initial groups.

- Use these groups to get initial calculations of $(\mu_M, \mu_F, \sigma^2_M, \sigma^2_F, \alpha)$.

```
## ---- Initial clusters using k-means
his   = NHIS_data[,height]           # store heights
h_kms = kmeans(NHIS_data[,height],2) # run k-means with k=2

## ---- Create table of parameters based on the k-means clusters
M_id = which.max(h_kms$centers)                    # get male id based on max height
tdat = data.table(height=his,id=h_kms$cluster)     # create table of heights and ids
tdat[,sex:=ifelse(id==M_id,"Male","Female")]       # add sex labels to ids
tdat = tdat[,list(mu=mean(height),sig2=var(height),# calc mu's, sig2's, and probs
                  prob=.N/nrow(tdat)),by=sex]      # by sex
tdat = tdat[order(sex,decreasing=T)]               # make sure Male is first row
theta0 = c(tdat[,mu],tdat[,sig2],tdat[1,prob])    # form theta0
names(theta0)=c("muM","muF","sig2M","sig2F","alpha") # give names to elements of theta
theta0
```

```
##        muM        muF      sig2M      sig2F      alpha
## 69.9959749 63.5208104  5.5077275  3.7491362  0.5025026
```

EM Algorithm: Initial Step

```
#E-Step
mu0s  = theta0[c("muM","muF")]                 # store mus
sig0s = sqrt(theta0[c("sig2M","sig2F")])       # store sigmas
alpha = theta0["alpha"]                        # store alpha
pMh   = dnorm(his,mu0s[1],sig0s[1])*alpha      # prob data and male
pFh   = dnorm(his,mu0s[2],sig0s[2])*(1-alpha) # prob data and female
piMh  = pMh/(pMh+pFh)                          # prob male given data

#M-Step
theta1["muM"]   = mean_wgt(his,piMh)      # male mean
theta1["muF"]   = mean_wgt(his,1-piMh)    # female mean
theta1["sig2M"] = var_wgt(his,piMh)       # male variance
theta1["sig2F"] = var_wgt(his,1-piMh)     # female variance
theta1["alpha"] = prop_wgt(piMh)          # prob male

cbind(theta0,theta1)
```

```
##           theta0     theta1
## muM   69.9959749 69.8373759
## muF   63.5208104 63.6533026
## sig2M  5.5077275  6.5504499
## sig2F  3.7491362  4.5392340
## alpha  0.5025026  0.5047312
```

EM Algorithm: While Loop

```
## ---- EM Loop: Run Until Convergence
while (Norm(theta0 - theta1, p=Inf) > 1e-12) {
  ## ---- E-Step
  theta0 = theta1                                  # update theta0
  mu0s   = theta0[c("muM","muF")]                  # store mus
  sig0s  = sqrt(theta0[c("sig2M","sig2F")])        # store sigmas
  alpha  = theta0["alpha"]                         # store alpha
  pMh    = dnorm(his,mu0s[1],sig0s[1])*alpha       # prob data and male
  pFh    = dnorm(his,mu0s[2],sig0s[2])*(1-alpha)   # prob data and female
  piMh   = pMh/(pMh+pFh)                           # prob male given data

  #M-Step
  theta1["muM"]   = mean_wgt(his,piMh)     # male mean
  theta1["muF"]   = mean_wgt(his,1-piMh)   # female mean
  theta1["sig2M"] = var_wgt(his,piMh)      # male variance
  theta1["sig2F"] = var_wgt(his,1-piMh)    # female variance
  theta1["alpha"] = prop_wgt(piMh)         # prob male
}
theta1
```

```
##        muM        muF      sig2M      sig2F      alpha
## 69.5209013 63.7703840  8.2256854  5.3738272  0.5224249
```

EM Algorithm: Results

```
EM_moms   = data.table(sex  = c("Male","Female"),
                       mu   = theta1[c("muM","muF")],
                       sig2 = theta1[c("sig2M","sig2F")],
                       prob = c(theta1["alpha"],1-theta1["alpha"]))
hght_moms = cbind(hght_moms,data.table(prob=c(mean(NHIS_data$sex=="Male"),
                                              mean(NHIS_data$sex=="Female"),1)))
```

Mean and variance by sex estimated with EM algorithm

```
##        sex       mu     sig2      prob
##     <char>    <num>    <num>     <num>
## 1:    Male 69.52090 8.225685 0.5224249
## 2: Female 63.77038 5.373827 0.4775751
```

Mean and variance by sex grouping by the sex variable in the data

```
##        sex       mu      var      prob
##     <char>    <num>    <num>     <num>
## 1:    Male 69.79632 7.953308 0.4677408
## 2: Female 64.11915 6.332597 0.5322592
```

They are pretty similar, though the baseline probabilities α are "switched."

- This is a known problem with the EM algorithm

mclust Package

The `mclust` package can be used to implement Gaussian Mixture Models.

```
library(mclust)
GMM         = Mclust(his,G=2,control=emControl(tol=c(1e-15,1e-15)))
GMM_mu      = GMM$parameters$mean
GMM_sig2    = GMM$parameters$variance$sigmasq
GMM_probs   = GMM$parameters$pro
mclust_moms = data.table(sex  = c("Female","Male"),
                         mu   = GMM_mu,
                         sig2 = GMM_sig2,
                         prob = GMM_probs)
mclust_moms = mclust_moms[order(sex,decreasing = T)]
```

mcluster Package Comparison

```
mclust_moms # results from mclust package
```

```
##       sex       mu     sig2      prob
##    <char>    <num>    <num>     <num>
## 1:   Male 69.52105 8.225321 0.5224018
## 2: Female 63.77049 5.374034 0.4775982
```

```
EM_moms # results from my EM code
```

```
##       sex       mu     sig2      prob
##    <char>    <num>    <num>     <num>
## 1:   Male 69.52090 8.225685 0.5224249
## 2: Female 63.77038 5.373827 0.4775751
```

```
hght_moms[sex≠"All",] # means and vars by sex from data
```

```
##       sex       mu      var      prob
##    <char>    <num>    <num>     <num>
## 1:   Male 69.79632 7.953308 0.4677408
## 2: Female 64.11915 6.332597 0.5322592
```

The results from my code are basically the same as the package.

Assigning Probabilities In Data

```
pMh  = dnorm(his,theta1["muM"],sqrt(theta1["sig2M"]))*theta1["alpha"]
pFh  = dnorm(his,theta1["muF"],sqrt(theta1["sig2F"]))*(1-theta1["alpha"])
piMh = pMh/(pMh+pFh)

NHIS_data[,prob_M := piMh]
NHIS_data[,.(height,prob_M,sex)]
```

```
##         height     prob_M    sex
##          <int>      <num> <char>
##      1:     71 0.99011749   Male
##      2:     62 0.03662435 Female
##      3:     74 0.99977382   Male
##      4:     72 0.99699611   Male
##      5:     72 0.99699611   Male
##     ---
## 29166:     70 0.96993279   Male
## 29167:     71 0.99011749   Male
## 29168:     71 0.99011749   Male
## 29169:     61 0.02140810 Female
## 29170:     64 0.12228689 Female
```

Extreme heights (really small or really large) makes it easy to distinguish a man vs a woman based solely on the height.

Assigning Probabilities In Data

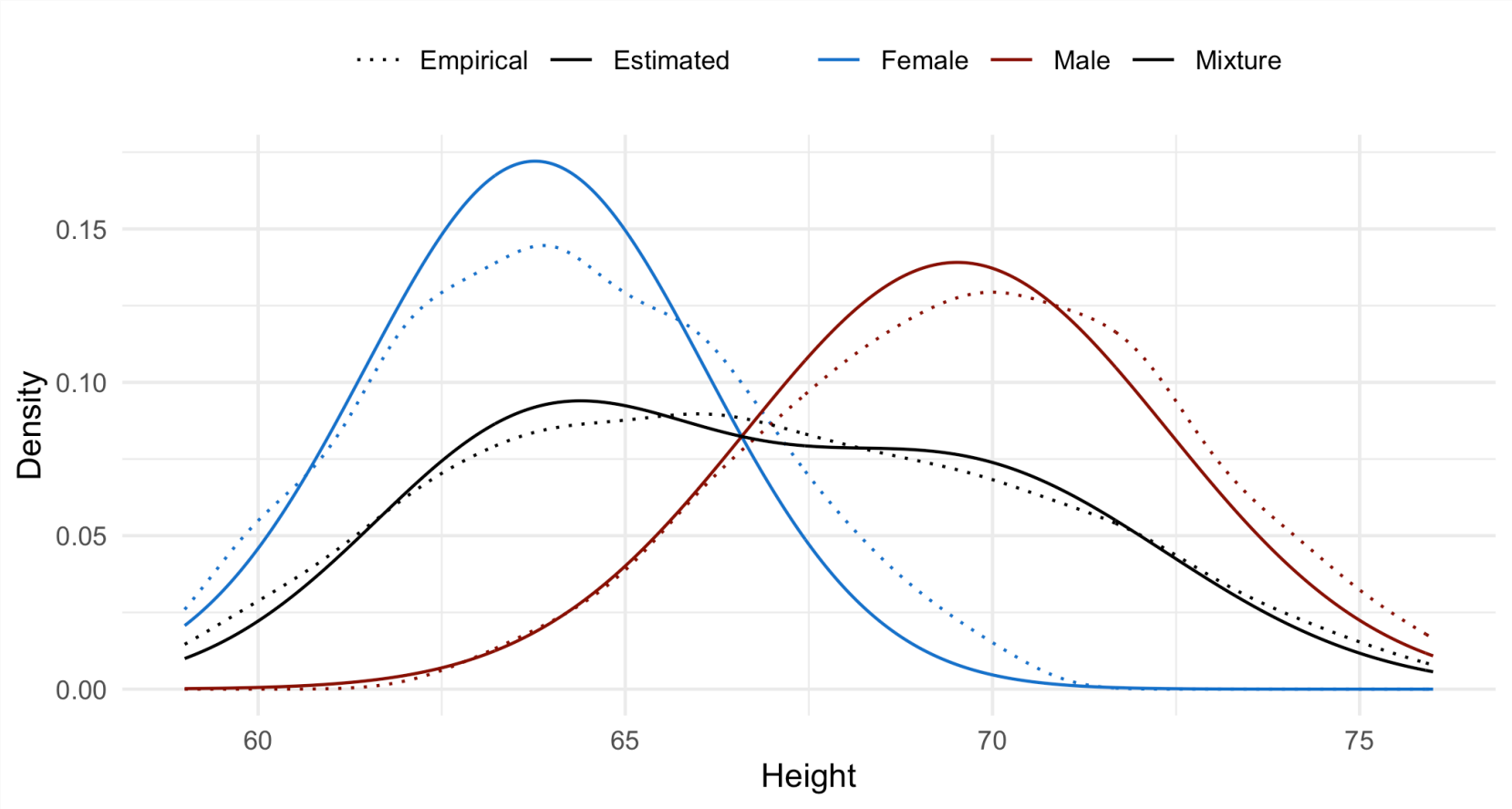
```
NHIS_data[abs(prob_M-0.5)<=0.15,.(height,prob_M,sex)]
```

```
##        height    prob_M    sex
##         <int>     <num> <char>
##     1:     67 0.6132741   Male
##     2:     66 0.3979287 Female
##     3:     66 0.3979287 Female
##     4:     67 0.6132741 Female
##     5:     66 0.3979287 Female
##    ---
##  5321:     67 0.6132741 Female
##  5322:     66 0.3979287 Female
##  5323:     67 0.6132741   Male
##  5324:     67 0.6132741   Male
##  5325:     67 0.6132741   Male
```

The heights that are around 0.5 probability are 5' 6" and 5' 7".

- Makes sense since these are the heights that are likely to have the most overlap between men and women.

Estimated vs Empirical Distributions



EM does a pretty good job of estimating the distribution of the data.

Modeling Height and Weight

Could do the same thing with height and weight.

- Will now also estimate the covariance between height and weight.

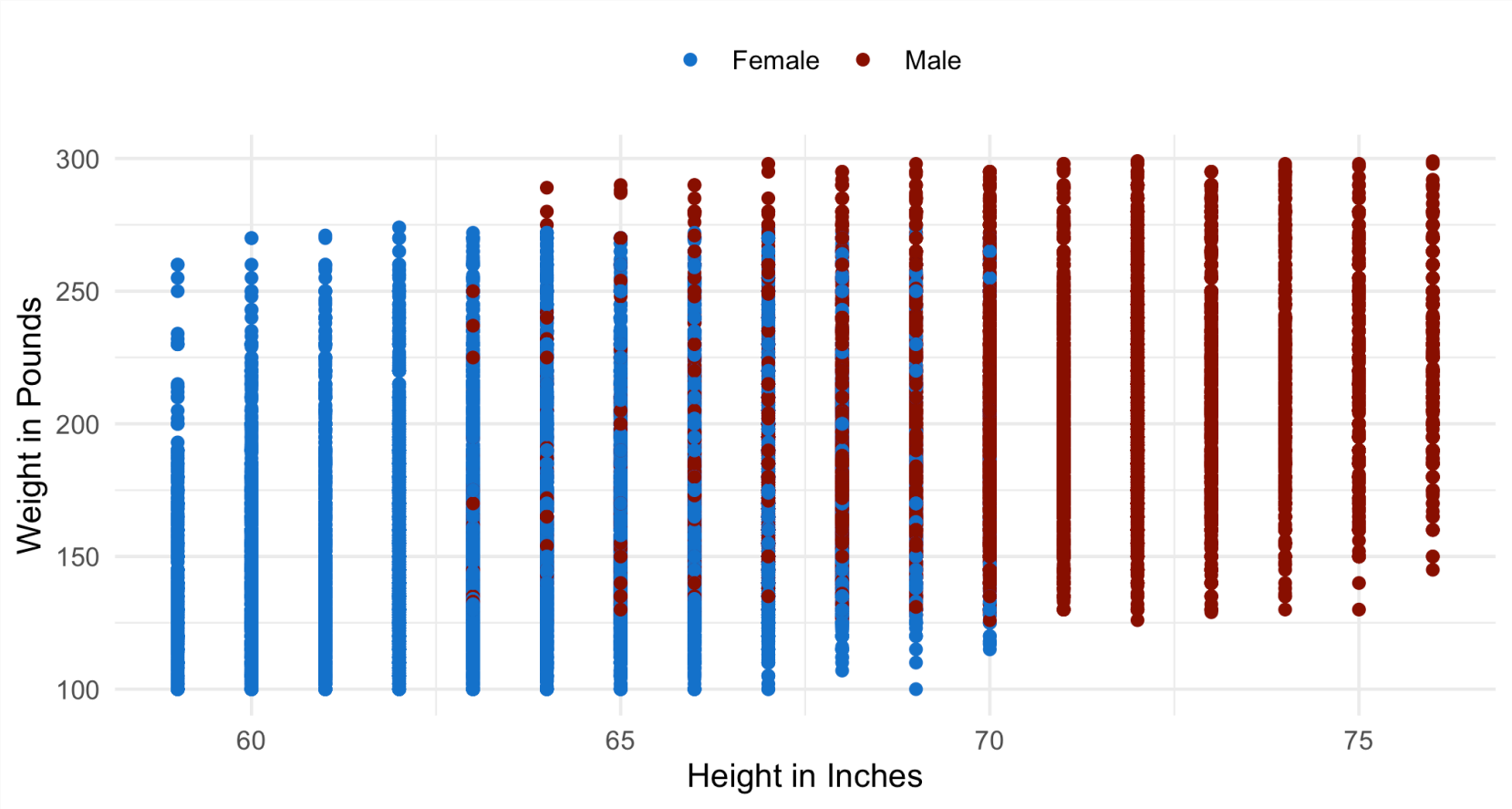
```
data_moms = NHIS_plot_data[,list(
  "mu_h"=mean(height),"var_h"=var(height),
  "mu_w"=mean(weight),"var_w"=var(weight),
  "cov_hw"=cov(height,weight),"alpha"=.N/nrow(NHIS_data)),by=sex]
data_moms[order(sex,decreasing = T)]
```

```
##       sex     mu_h     var_h     mu_w    var_w   cov_hw     alpha
##    <char>    <num>     <num>    <num>    <num>    <num>     <num>
## 1:   Male 69.79632  7.953308 194.4378 1260.671 37.32035 0.4677408
## 2: Female 64.11915  6.332597 161.6400 1294.121 22.56490 0.5322592
## 3:    All 66.77460 15.114710 176.9808 1546.246 75.82307 1.0000000
```

"mu_h" is the mean of height, "var_w" is the variance of weight, etc.

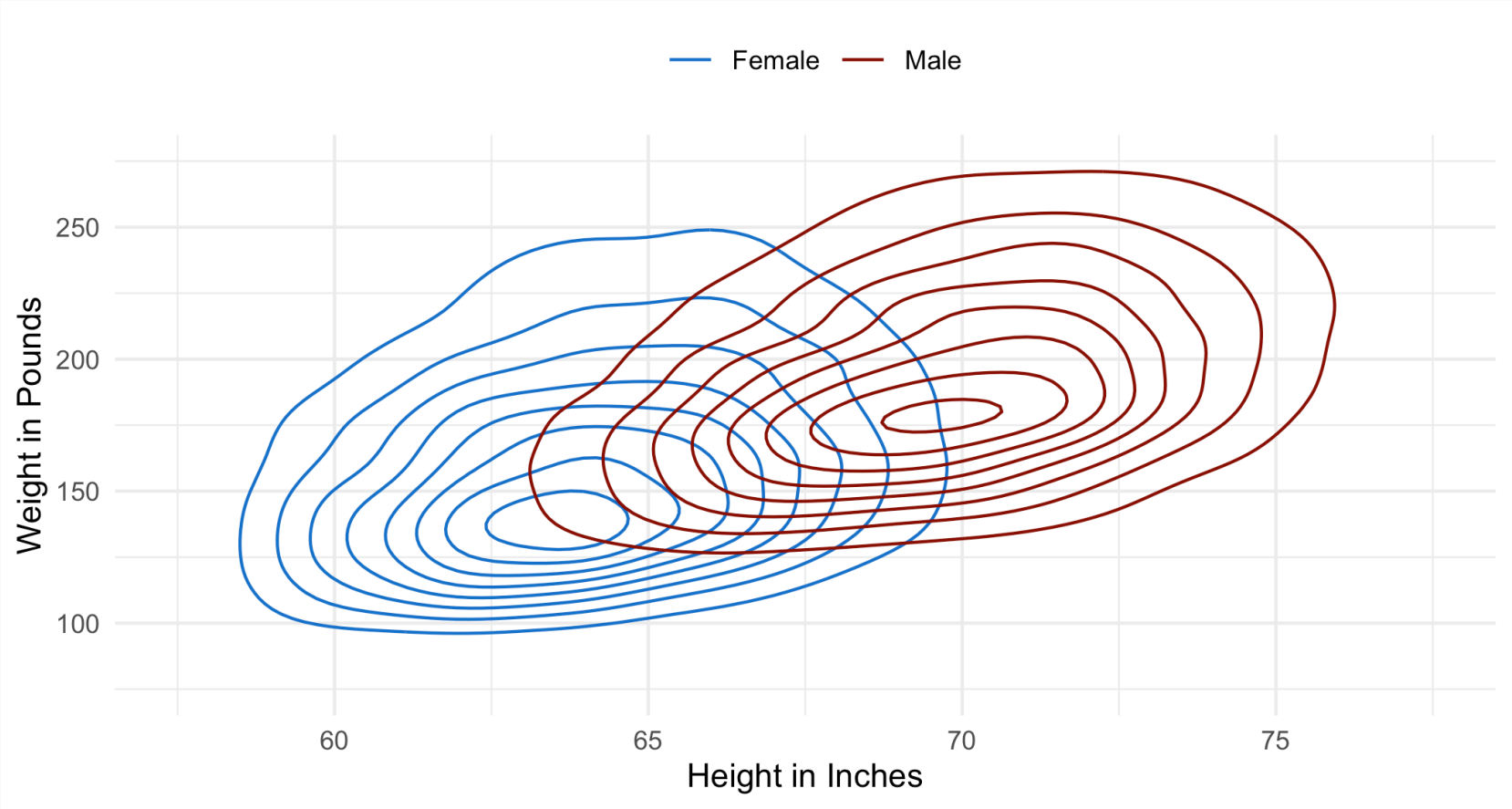
- "cov_hw" is the covariance of height and weight.
- Now 11 parameters to estimate.

Height vs Weight Scatter Plot by Sex



Men and women are clearly seperated in two groups that overlap a bit.

Joint Distribution by Sex



Assuming that height and weight are distributed jointly normal might be reasonable

- Weight seems to be skewed upwards, so it isn't exactly normal.

Estimation with mclust package

```r
# Fit model 100 times and take the result with lowest BIC
# EVV is a model about the covariance matrix:
# Basically, we are only assuming the "volume" of M/F vcov mat are the same
# The same volume means the determinants of the two matrices are the same
bic = Inf
for(i in 1:100){
  temp=Mclust(NHIS_data[,.(height,weight)],G=2,modelNames = "EVV",
              control=emControl(tol=c(1e-15,1e-15),itmax=c(1000,1000)))
  if(temp$bic<bic) GMM = temp; bic = GMM$bic
}
## --- Store estimated parameters
mus_est    = GMM$parameters$mean
vcovs_est  = list(GMM$parameters$variance$sigma[,,1],
                  GMM$parameters$variance$sigma[,,2])
alphas_est = GMM$parameters$pro
Mid        = which.max(mus_est["height",])
ids        = c(Mid,setdiff(1:2,Mid))

## --- Give Male/Female labels to objects
colnames(mus_est)[ids] = c("Male","Female")
names(vcovs_est)[ids]  = c("Male","Female")
names(alphas_est)[ids] = c("Male","Female")
```

Estimation Results

```
mus_est # estimated means
```

```
##           Female      Male
## height  64.44962  70.41078
## weight 162.76720 199.21046
```

```
vcovs_est # estimated covariance matrices
```

```
## $Female
##           height    weight
## height  6.921702   24.0905
## weight 24.090498 1180.5560
##
## $Male
##          height     weight
## height  6.26121   24.20829
## weight 24.20829 1306.00093
```

```
alphas_est # estimated proportions/probabilities
```

```
##   Female     Male
## 0.609979 0.390021
```

Compare Results

```
GMM_results_dt
```

```
##       sex     mu_h    var_h     mu_w    var_w   cov_hw    alpha
##    <char>    <num>    <num>    <num>    <num>    <num>    <num>
## 1:   Male 70.41078 6.261210 199.2105 1306.001 24.20829 0.390021
## 2: Female 64.44962 6.921702 162.7672 1180.556 24.09050 0.609979
```

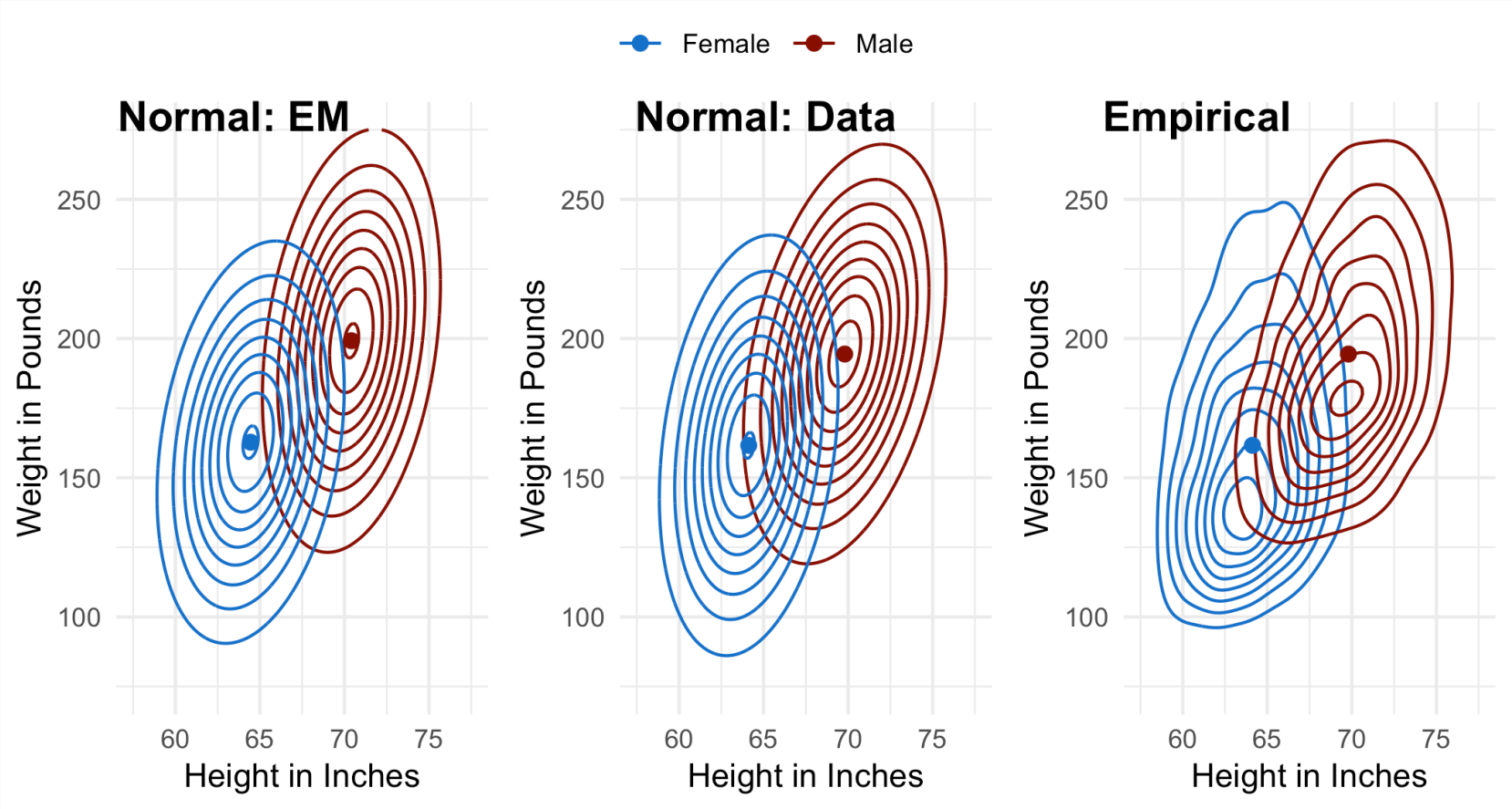
```
data_moms[1:2,]
```

```
##       sex     mu_h    var_h     mu_w    var_w   cov_hw     alpha
##    <char>    <num>    <num>    <num>    <num>    <num>     <num>
## 1:   Male 69.79632 7.953308 194.4378 1260.671 37.32035 0.4677408
## 2: Female 64.11915 6.332597 161.6400 1294.121 22.56490 0.5322592
```

Results are pretty good, though some of the variance/covariance terms are off.

- These are generally much harder to pin down.

Comparing EM Results to Data



Height and weight might not be distributed jointly normal.

- EM still recovers the parameters if we assume that they are.
- Remember: Empirical means assuming nothing about the distribution.

Readings and Other Resources

Readings

- Chapter 12 of *An Introduction to Statistical Learning*
- Chapter 9 of *Pattern Recognition and Machine Learning* by Christopher Bishop
- Economics paper that uses EM:
  - "A Study of Cartel Stability: The Joint Executive Committee, 1880-1886" by Robert Porter (1983 *The Bell Journal of Economics*) (now *The RAND Journal of Economics*)

Other Resources

- `mixtools` package for general, finite mixture models
  - Can fit more than just Normal (Gaussian) Mixture Models
  - Paper about `mixtools` package
  - `mixtools` documentation

Next lecture: Supervised Learning