Maximum Likelihood Estimation

MLE

- tool for parameter estimation
- good approach for cases when OLS (ordinary least squares) assumptions are violated
- e.g. for non-linear models with non-normal data
- in MLE, we estimate the parameters of a model that maximize the likelihood of your data

Probability Density Function

- assume an observed **data** vector
  y = (y1, y2, ..., ym)
- goal of MLE is to identify the population (the model) that is **most likely** to have generated the data

Probability Density Function

- Here we assume population (model) is associated with a corresponding probability distribution
- Each probability distribution is characterized by a unique value of the model's **parameter(s)**

Probability Density Function

- As model parameters change, different probability distributions are generated
- Model = the family of probability distributions indexed by the model's parameter(s)

Probability Density Function

- f(y|w) is the probability density function (PDF) specifying the probability of observing **data y**, given **model parameter(s) w**
- note: w may be a parameter vector w = (w1, w2, ..., wk)
- e.g. for a *normal* PDF: w = (mu, sigma)

Probability Density Function

- If observations yi are statistically independent, then by probability theory, the PDF for the data as a whole, y = (y1, ..., ym) given the parameter vector w, can be expressed as the multiplication of PDFs for individual observations:

$$f(y = (y_1, y_2, \ldots, y_n)|w) = f_1(y_1|w) f_2(y_2|w) \ldots f_n(y_n|w)$$

Probability Density Function

- e.g. let’s say our data vector Y is made up of 3 observations
  y1=80, y2=110, y3=130
- and we want to compute the PDF for a normal distribution

$$p(y_i|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_i-\mu)^2}{2\sigma^2}}$$

Probability Density Function

$$p(y_i|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_i-\mu)^2}{2\sigma^2}}$$

$$p(y = (y_1, y_2, y_3)|\mu,\sigma) = p(y_1|\mu,\sigma)p(y_2|\mu,\sigma)p(y_3|\mu,\sigma)$$

- assume our mu=100 and sigma=15

$$p(80|\mu = 100, \sigma = 15) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(80-\mu)^2}{2\sigma^2}} = 0.010934$$

$$p(110|\mu = 100, \sigma = 15) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(80-\mu)^2}{2\sigma^2}} = 0.021297$$

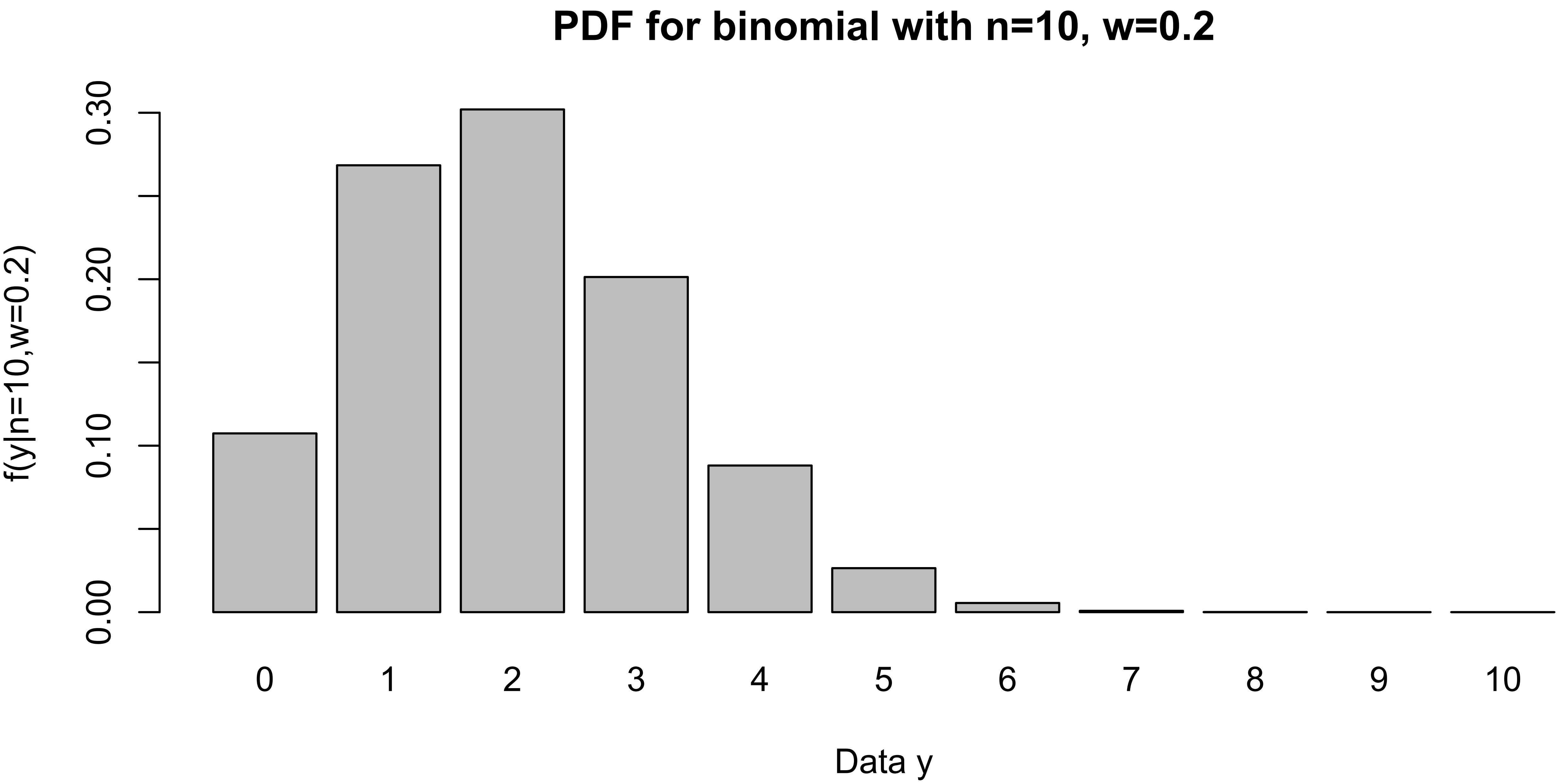
$$p(130|\mu = 100, \sigma = 15) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(80-\mu)^2}{2\sigma^2}} = 0.003599$$

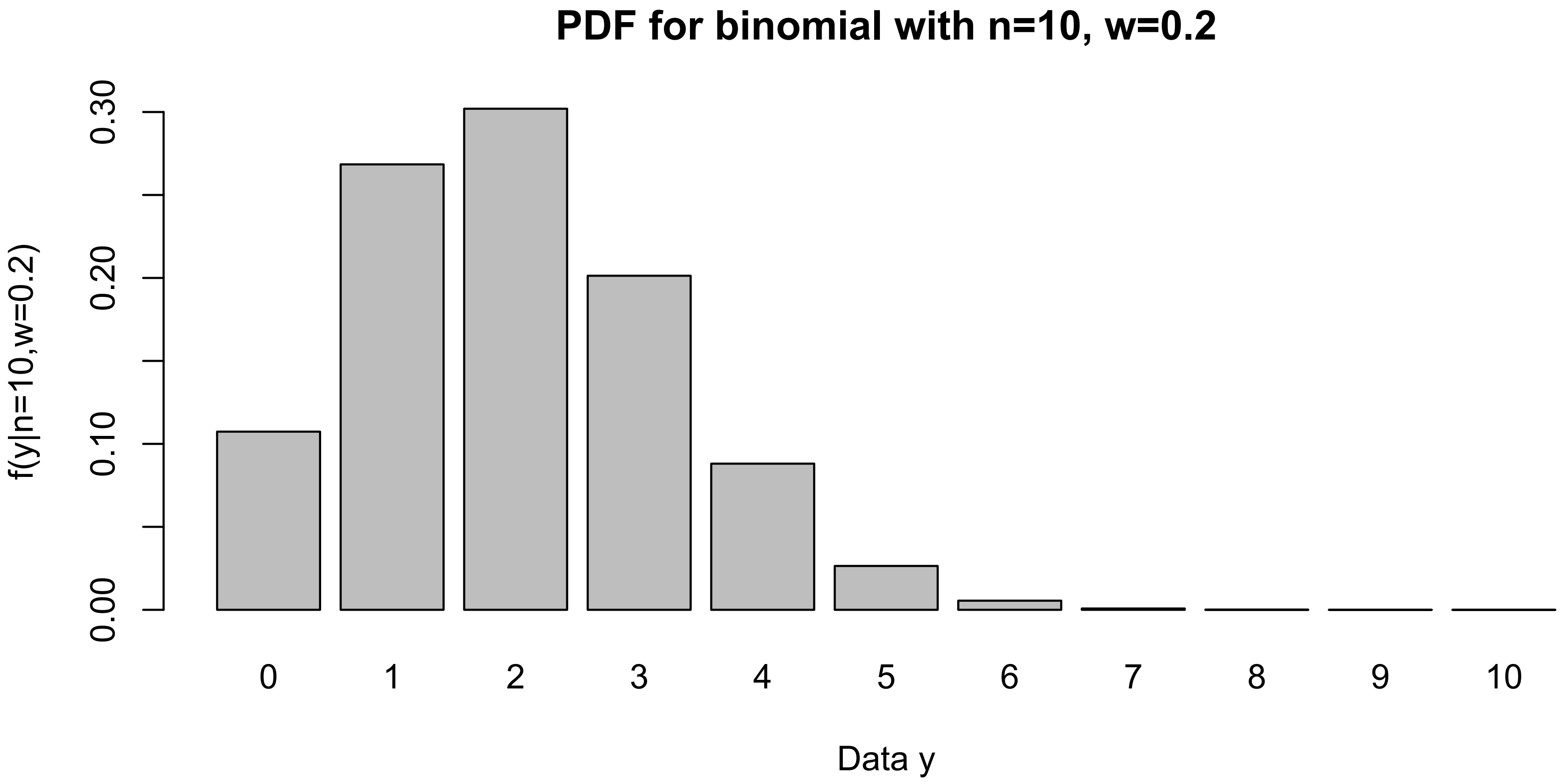
$$p(y = (y_1, y_2, y_3)|\mu,\sigma) = (.010934)(.021297)(.003599) = .000000838$$

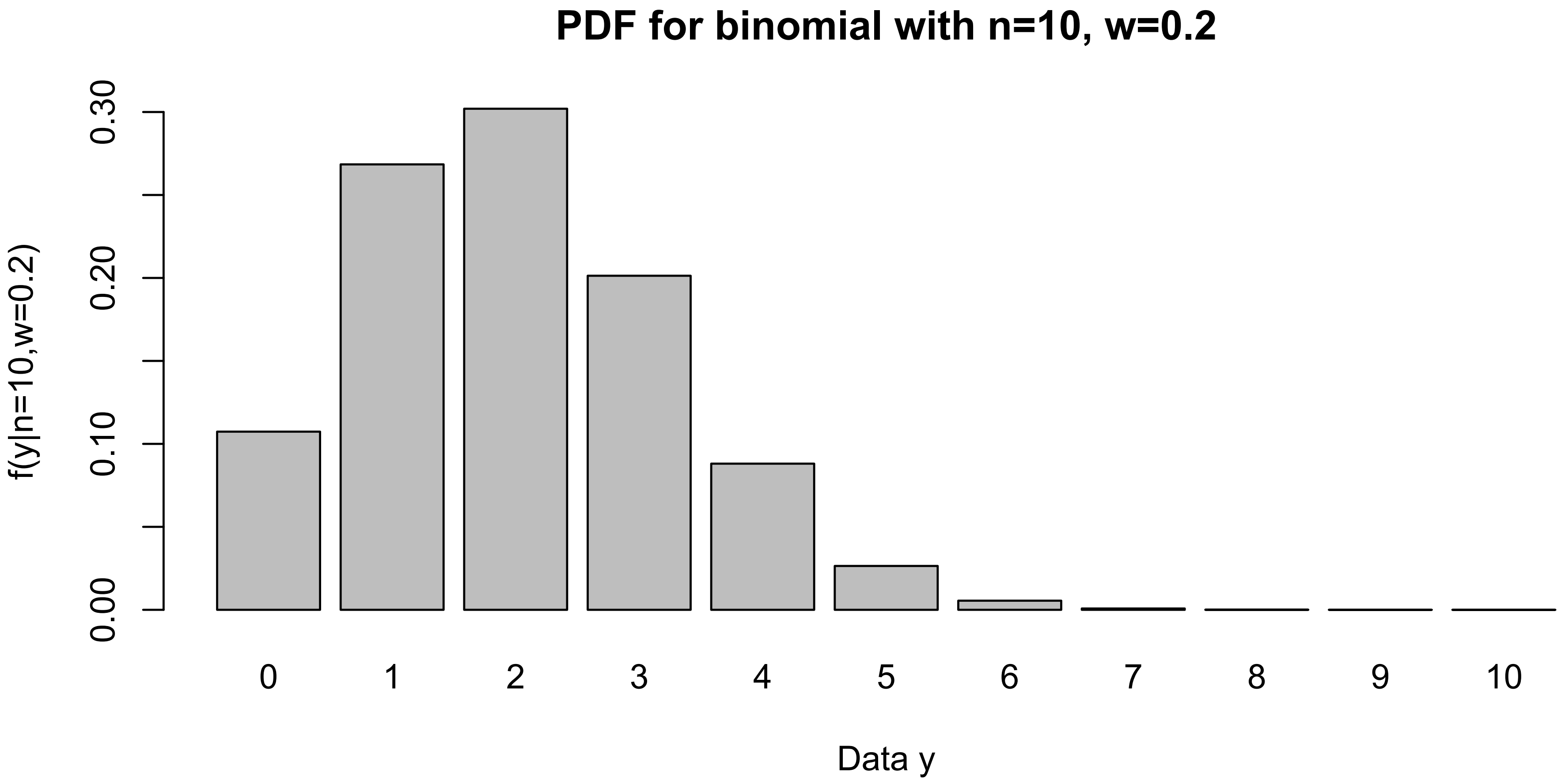
PDF: an example

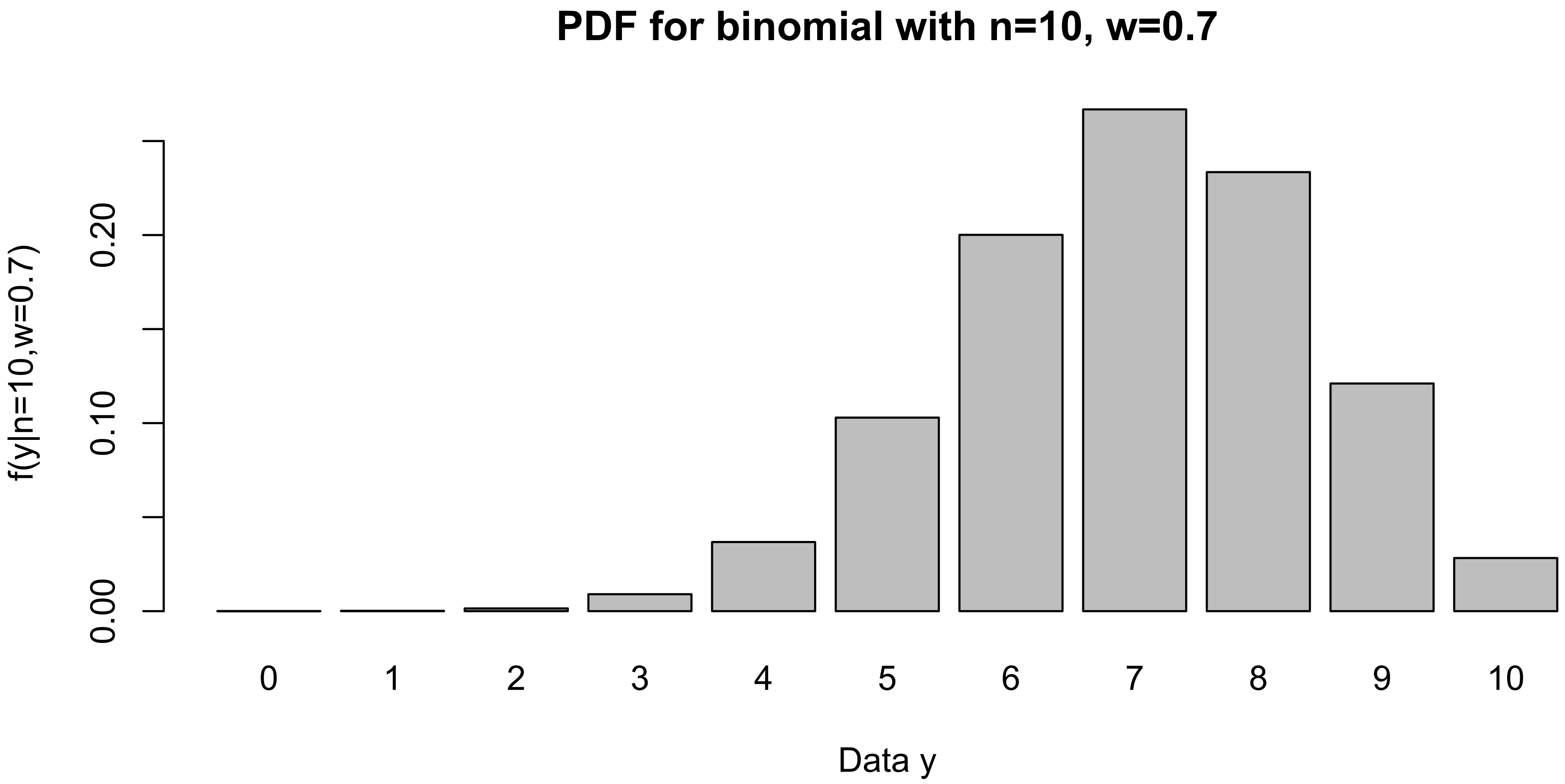
- y is # of successes in a sequence of 10 Bernoulli trials* (e.g. tossing a coin 10 x)
- assume probability of a success on any one trial is 0.2 (a biased coin)
- parameter vector w is n=10, w=0.2
- PDF is: $f(y|n=10, w=0.2) = \frac{10!}{y!(10-y)!}(0.2)^y(0.8)^{10-y} \quad (y = 0, 1, \ldots, 10)$
- this is binomial distribution with n=10, w=0.2

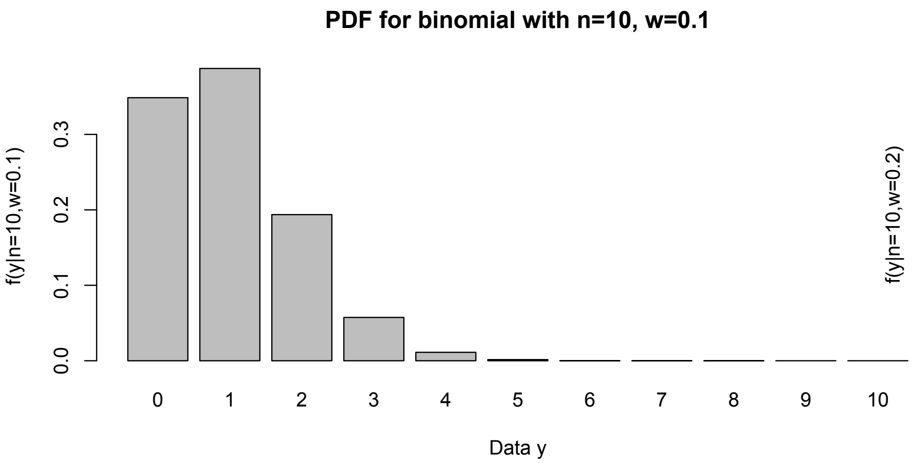
** a **Bernoulli trial** is an experiment whose outcome is random and can be either of two possible outcomes, "success" and "failure".*


PDF for binomial with n=10, w=0.2
f(y|n=10,w=0.2)
0.00
0.10
0.20
0.30
0
1
2
3
4
5
6
7
8
9
10
Data y

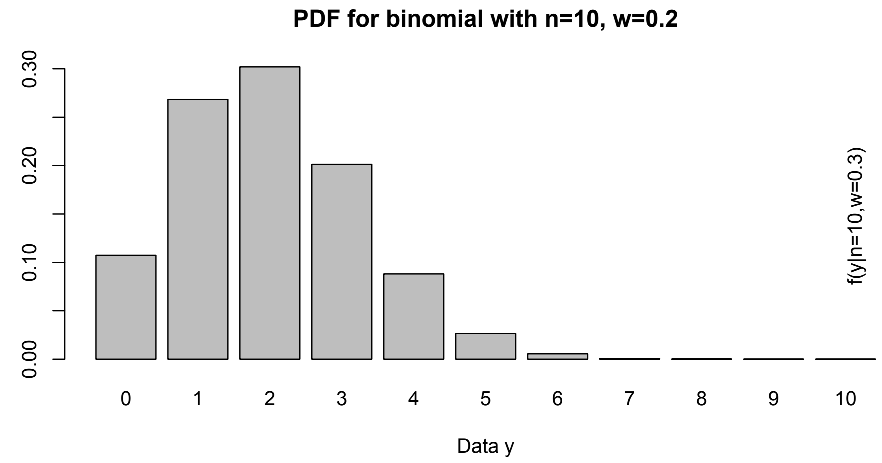

PDF for binomial with n=10, w=0.2
f(y|n=10,w=0.2)
0.00
0.10
0.20
0.30
0
1
2
3
4
5
6
7
8
9
10
Data y


PDF for binomial with n=10, w=0.2
f(y|n=10,w=0.2)
0.00
0.10
0.20
0.30
0
1
2
3
4
5
6
7
8
9
10
Data y

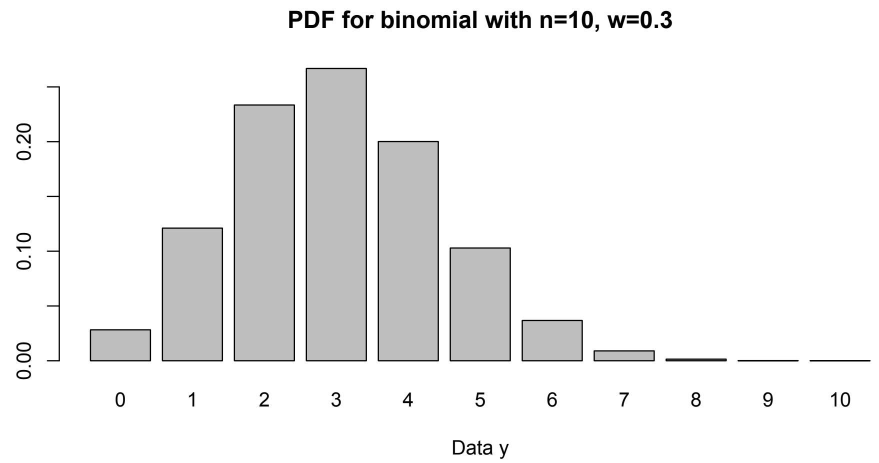

PDF for binomial with n=10, w=0.7
f(y|n=10,w=0.7)
0.00
0.10
0.20
0
1
2
3
4
5
6
7
8
9
10
Data y



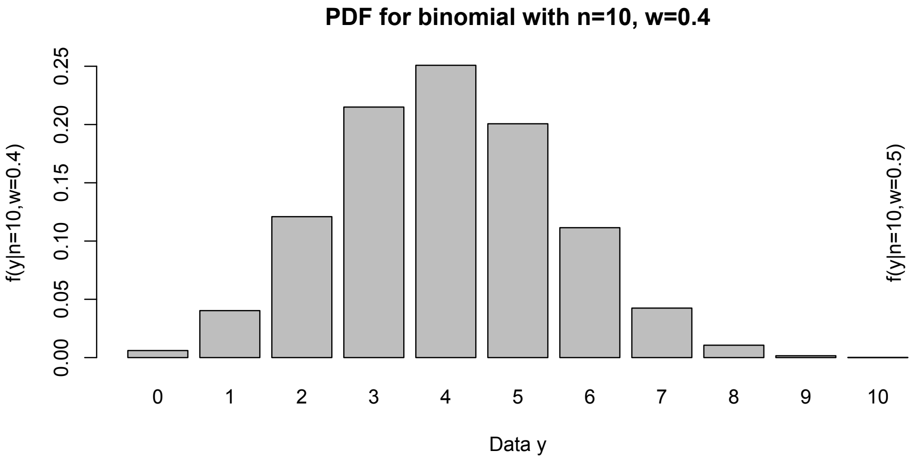
PDF for binomial with n=10, w=0.1



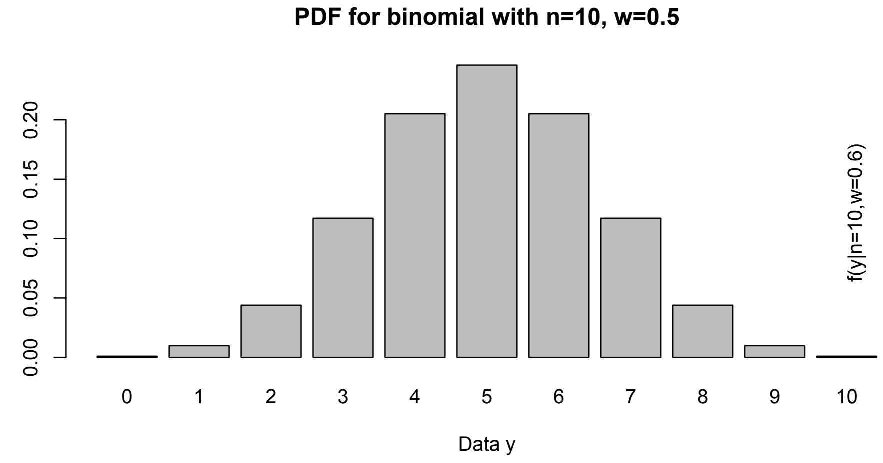
PDF for binomial with n=10, w=0.2



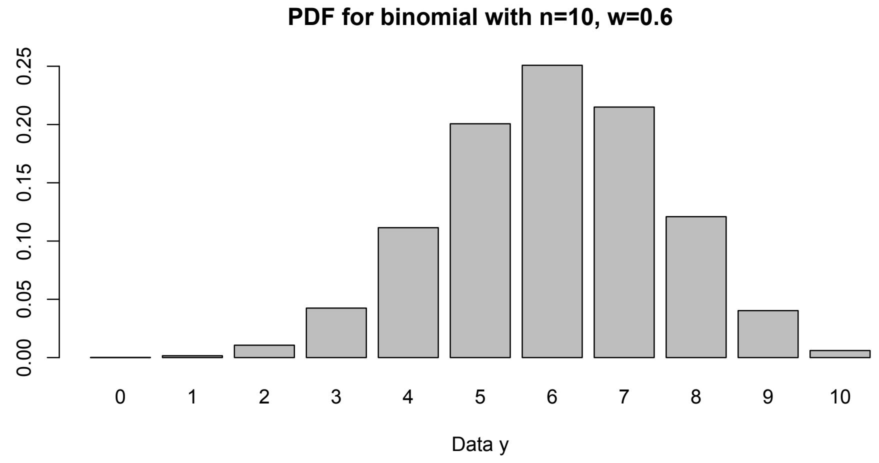
PDF for binomial with n=10, w=0.3



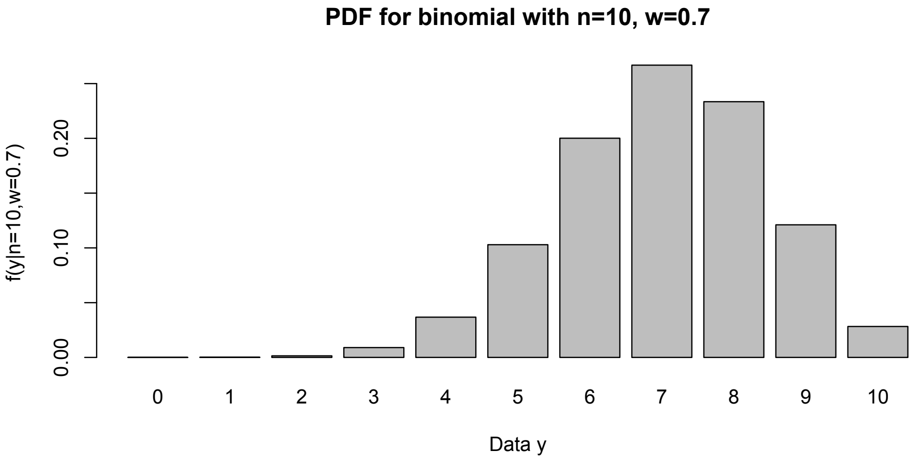
PDF for binomial with n=10, w=0.4



PDF for binomial with n=10, w=0.5

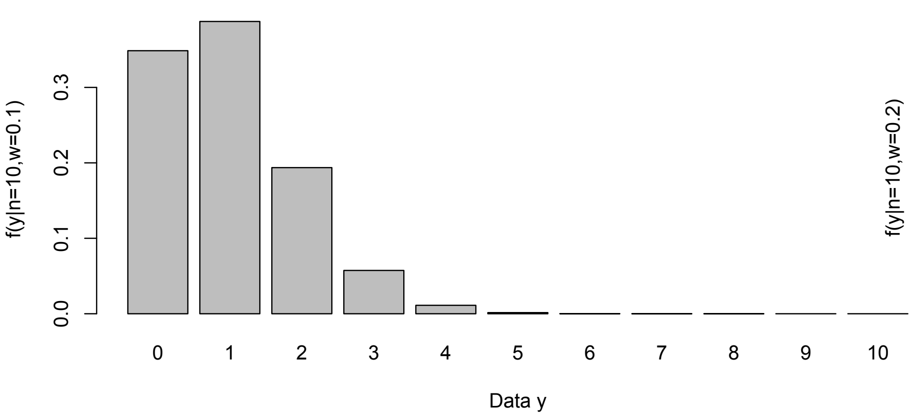


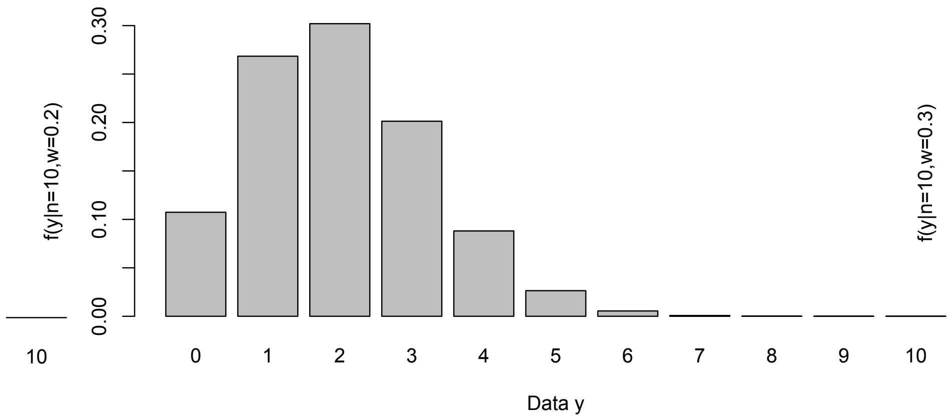
PDF for binomial with n=10, w=0.6

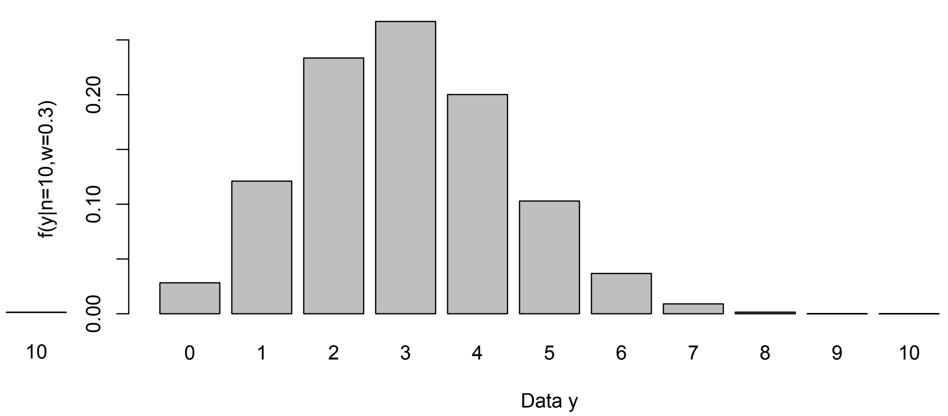


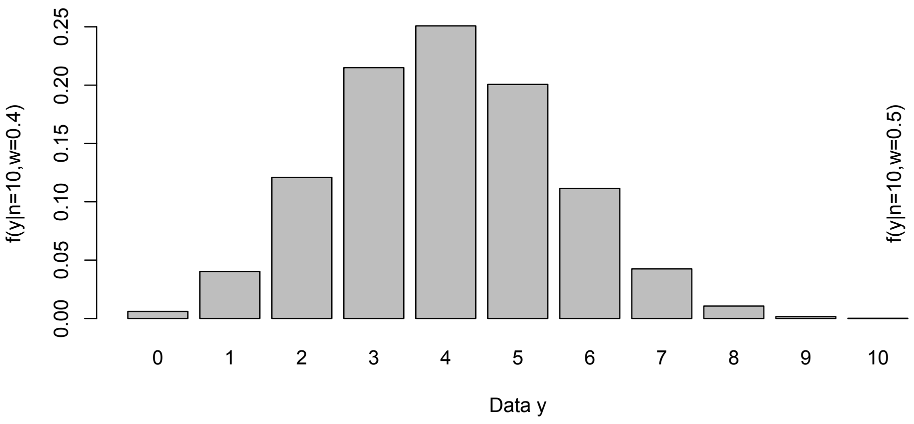
PDF for binomial with n=10, w=0.7

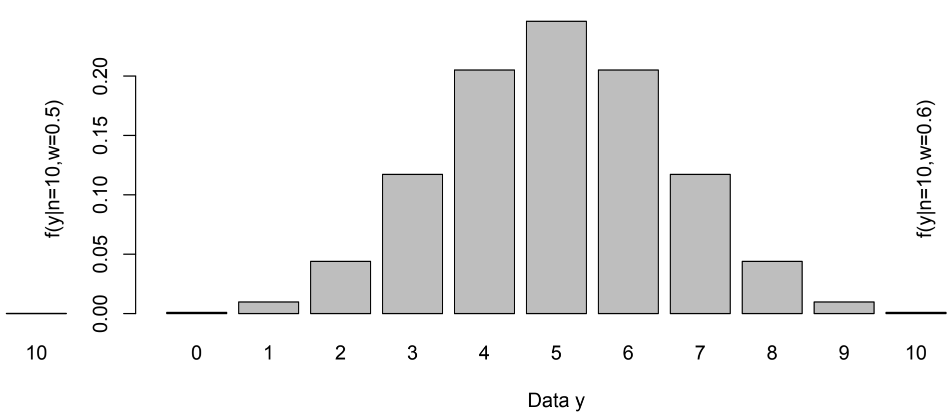
and so on ...

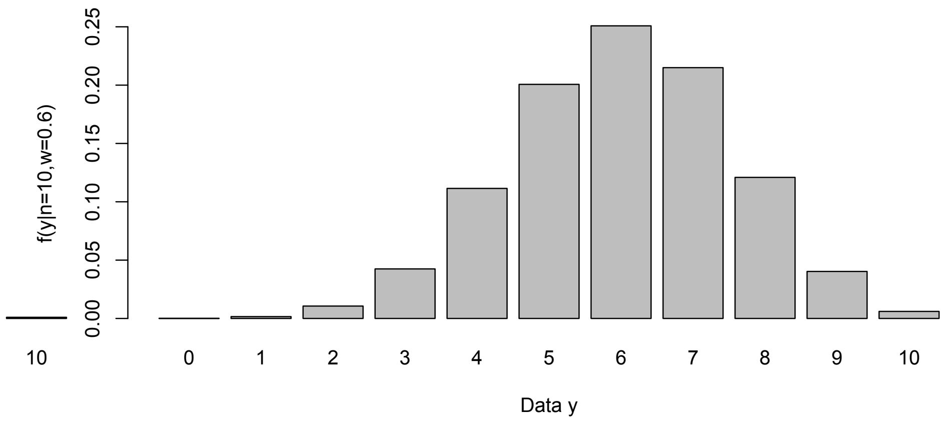

PDF for binomial with n=10, w=0.1

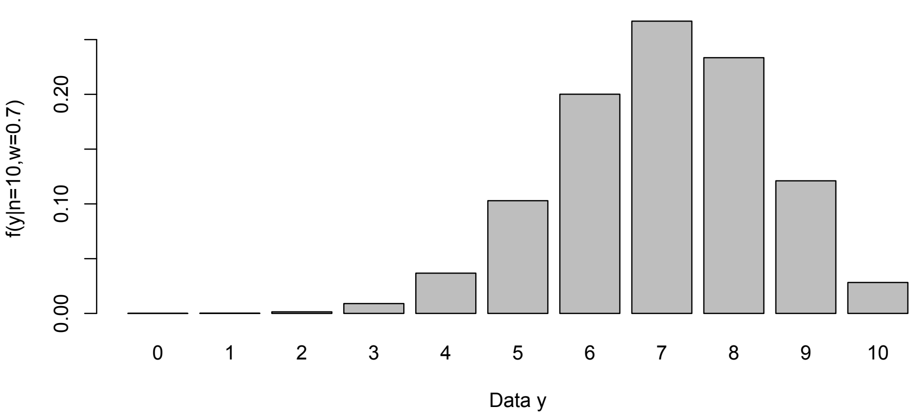

PDF for binomial with n=10, w=0.2


PDF for binomial with n=10, w=0.3


PDF for binomial with n=10, w=0.4


PDF for binomial with n=10, w=0.5


PDF for binomial with n=10, w=0.6


PDF for binomial with n=10, w=0.7

and so on ...

- The collection of all such PDFs generated by varying the parameter across its range defines a **model**

Likelihood function

- Given a set of parameter values, the corresponding PDF will show that some data are more probable than other data
- In fact we have already observed the data

Likelihood function

- We are faced with the inverse problem
- Given the observed data, and a model of the process by which the data was generated,

  find the **one PDF**, among all the probability densities that the model prescribes, that is **most likely to have produced the data**

Likelihood function

- we define the likelihood function by reversing the roles of the data vector y and the parameter vector w in f(y|w):

$$L(w|y) = f(y|w)$$

Likelihood function

$$L(w|y) = f(y|w)$$

- L(w|y) represents the likelihood of the parameter w given the observed data y
- For our one-dimensional binomial example the likelihood function for **y=7** and n=10 is

$$\begin{aligned} L(w|n=10, y=7) &= f(y=7|n=10, w) \\ &= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1) \end{aligned}$$

Likelihood function

$$L(w|y) = f(y|w)$$

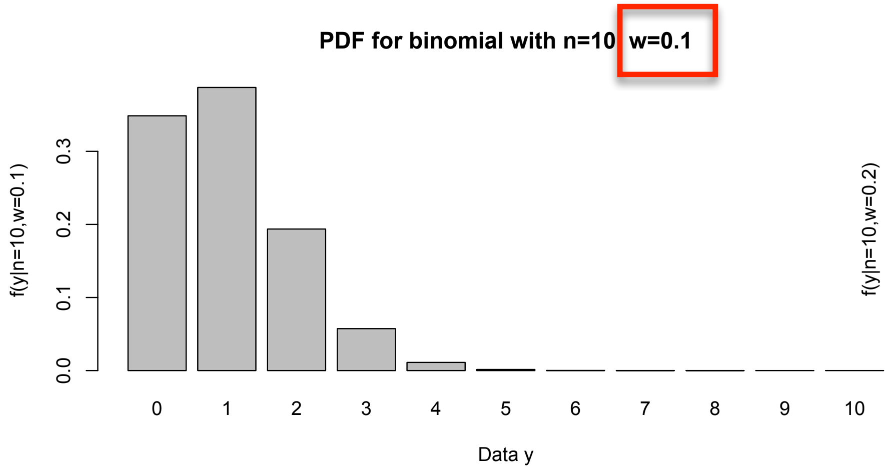
- L(w|y) represents the likelihood of the parameter w given the observed data y
- For our one-dimensional binomial example the likelihood function for **y=7** and n=10 is

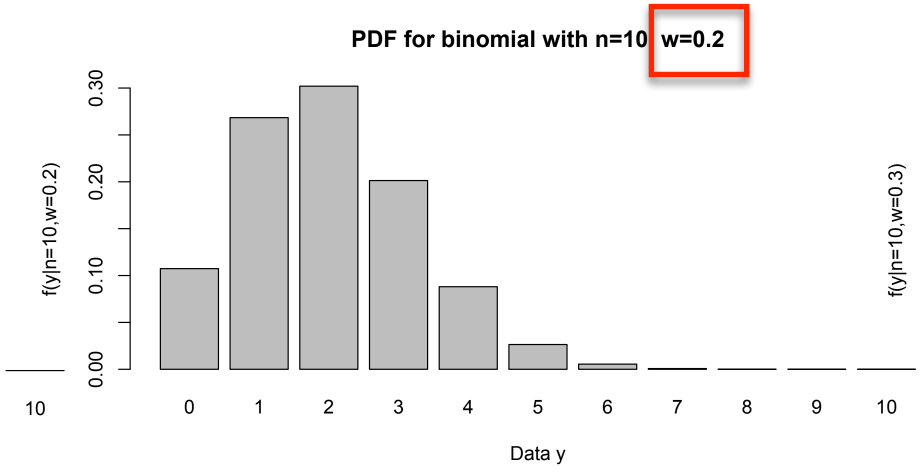
$$\begin{aligned} L(w|n=10, y=7) &= f(y=7|n=10, w) \\ &= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1) \end{aligned}$$

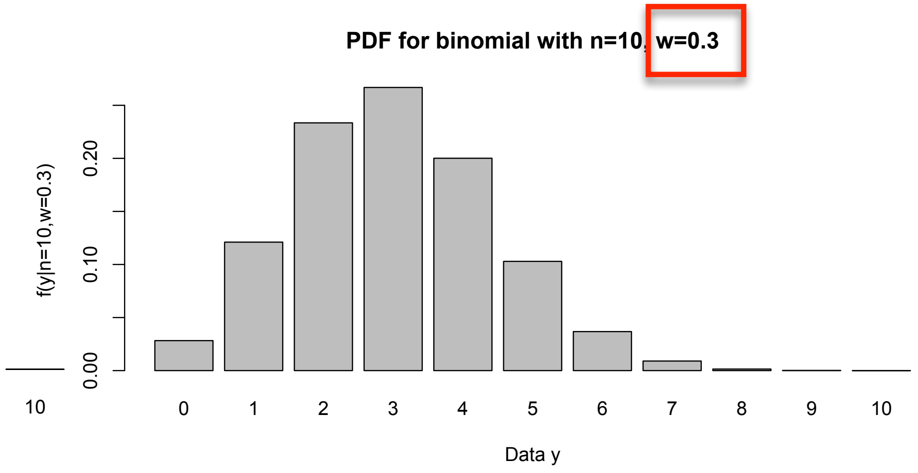
but what value of *w*?

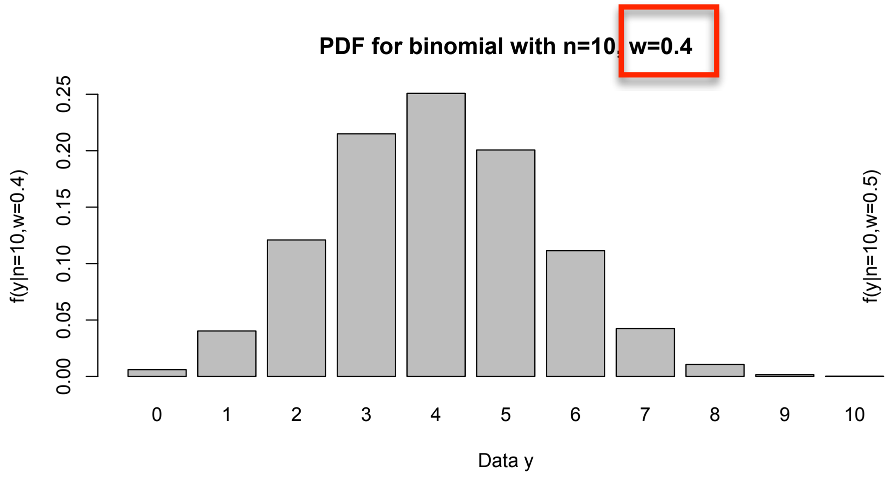
let’s try all values of w between 0.0 and 1.0

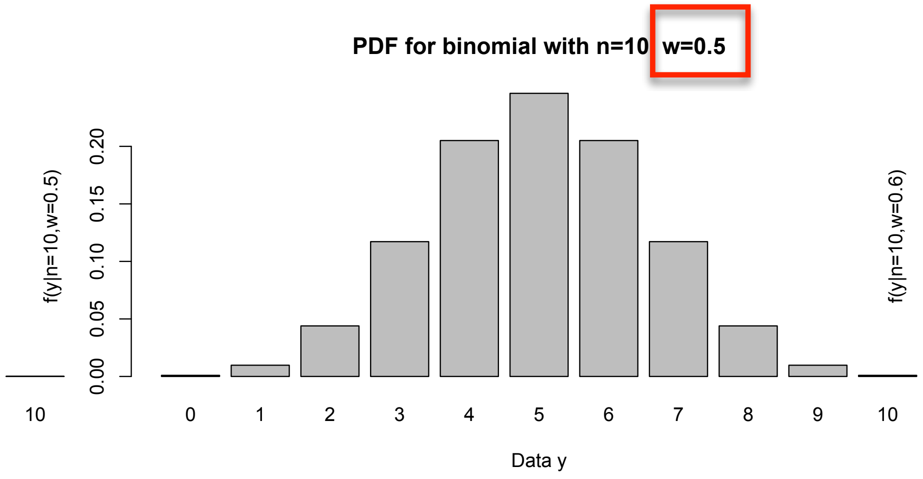
y=7

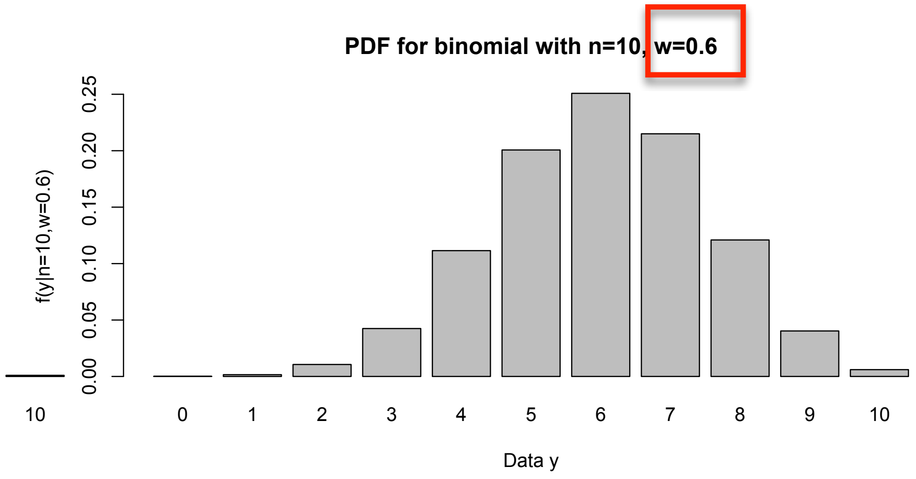


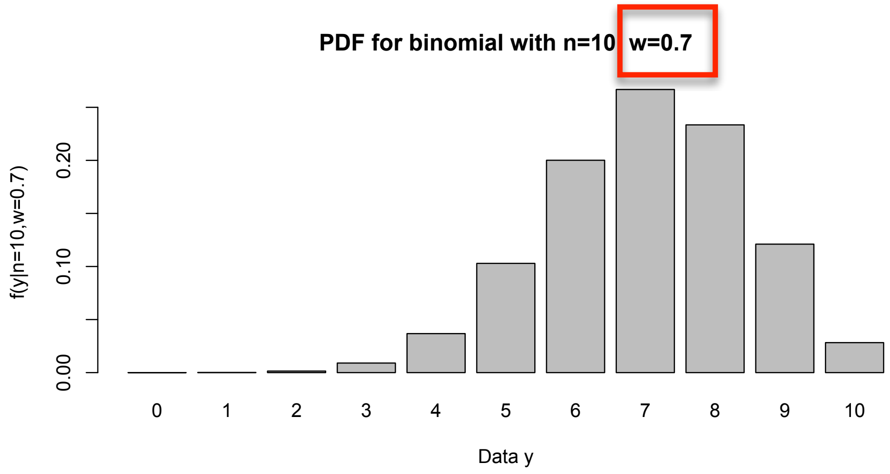








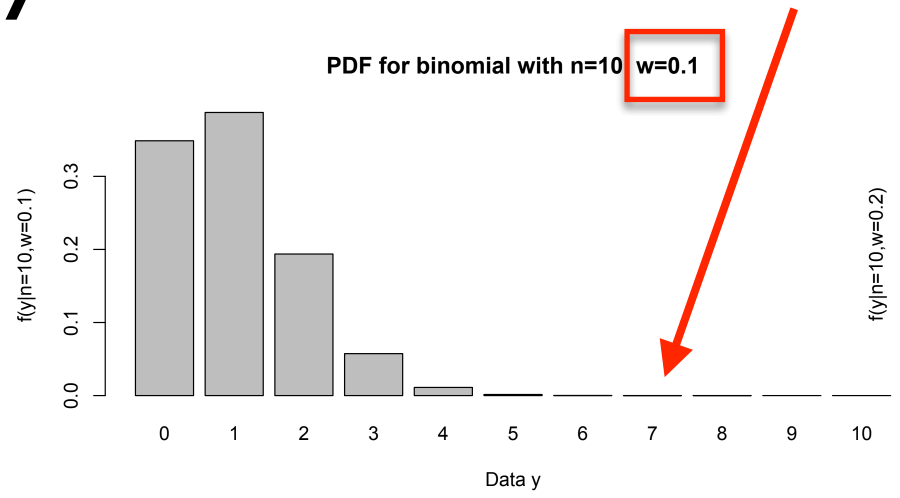


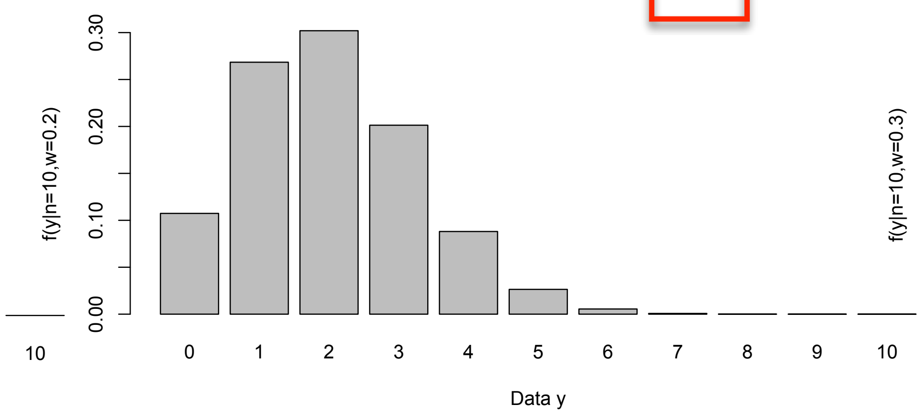


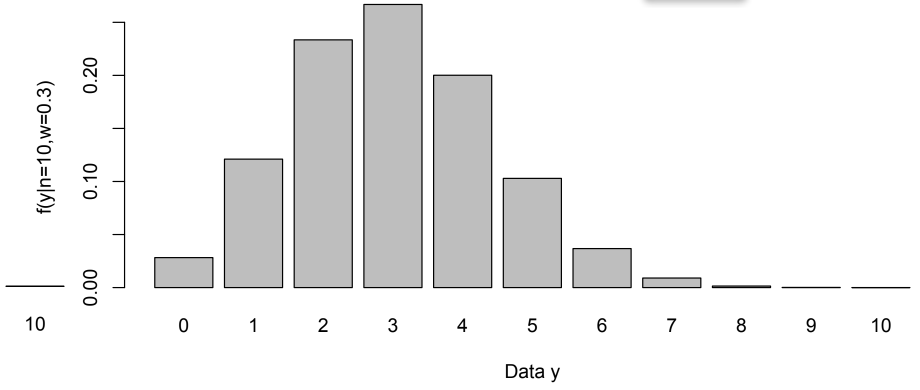
… and so on

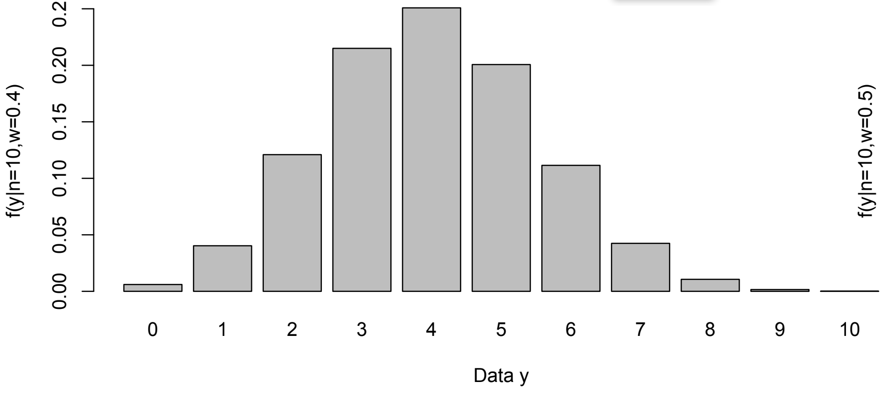
let's try all values of w between 0.0 and 1.0

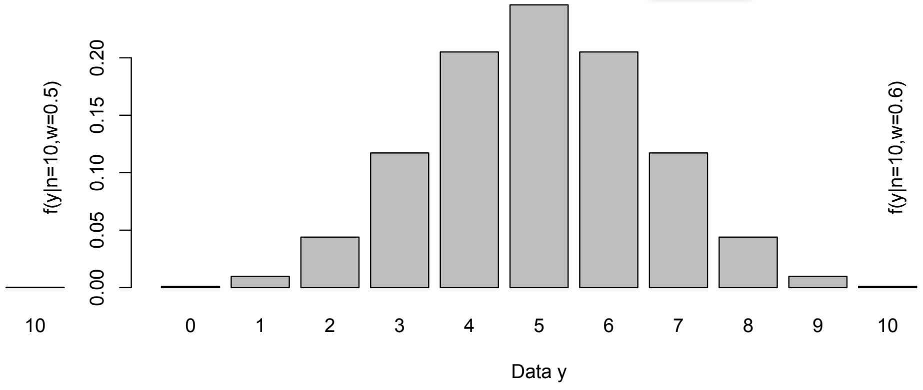
y=7

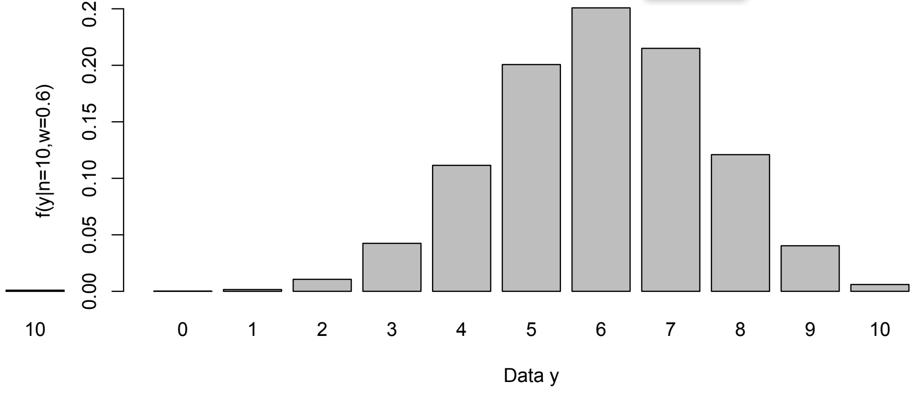

PDF for binomial with n=10, w=0.1

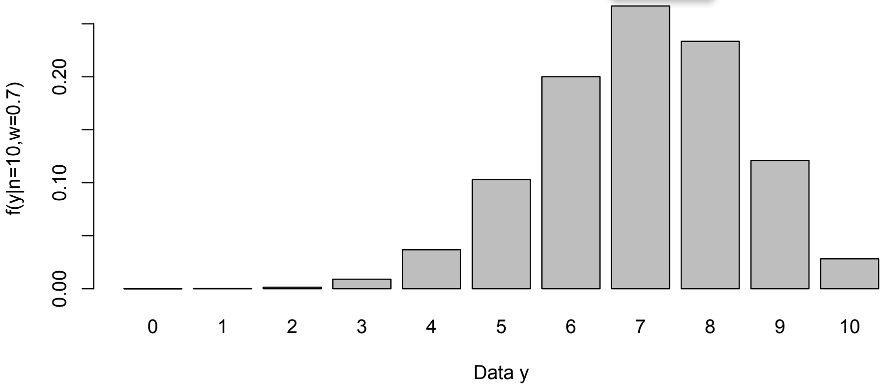

PDF for binomial with n=10, w=0.2


PDF for binomial with n=10, w=0.3


PDF for binomial with n=10, w=0.4


PDF for binomial with n=10, w=0.5

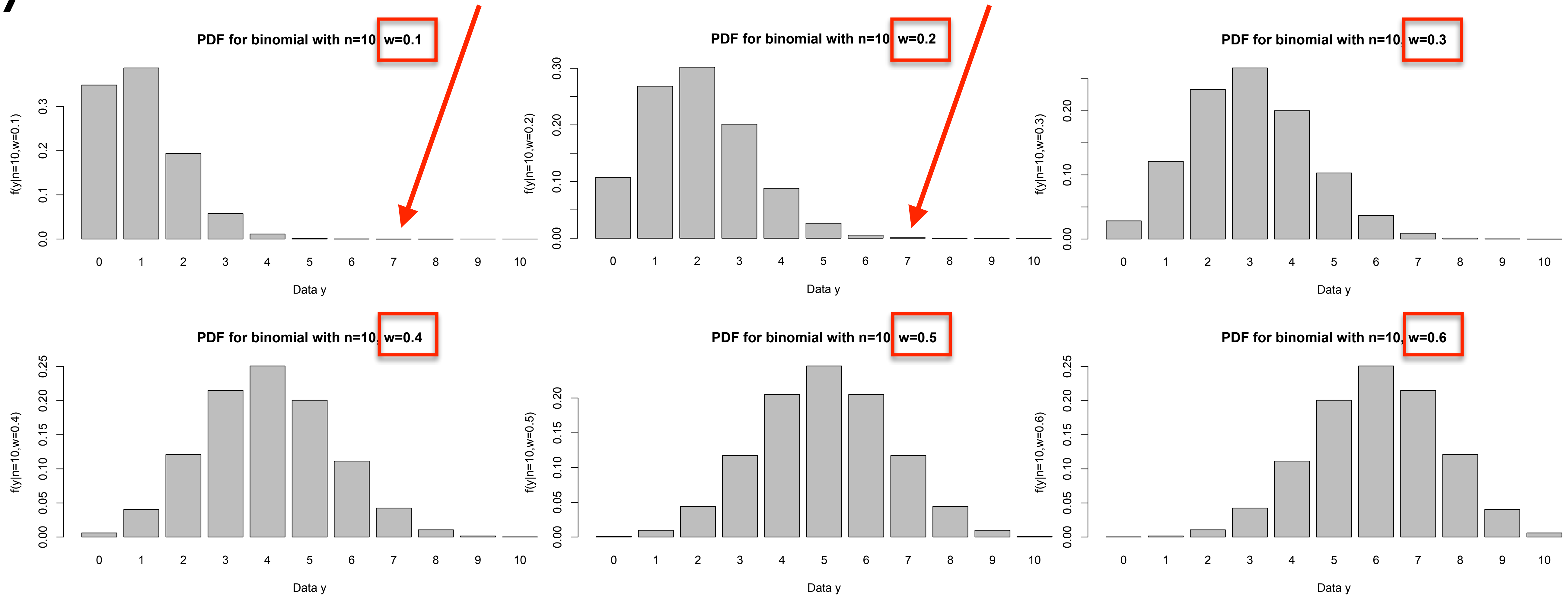

PDF for binomial with n=10, w=0.6

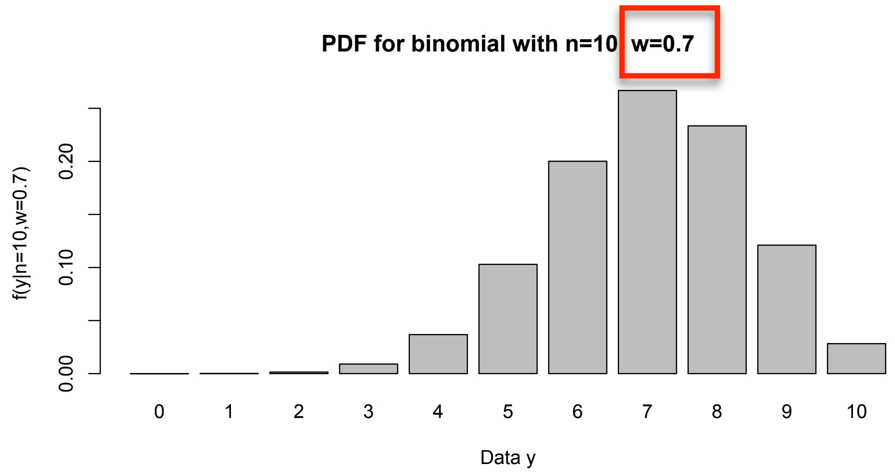

PDF for binomial with n=10, w=0.7

… and so on

let's try all values of w between 0.0 and 1.0

y=7

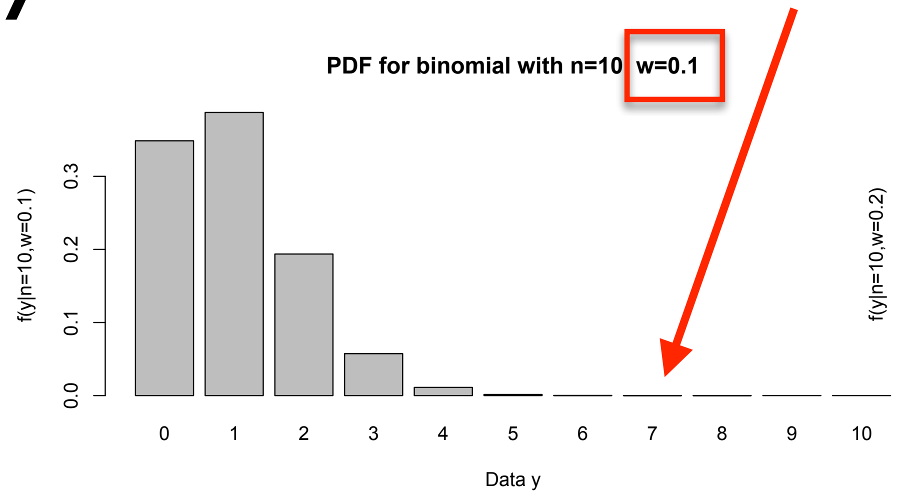


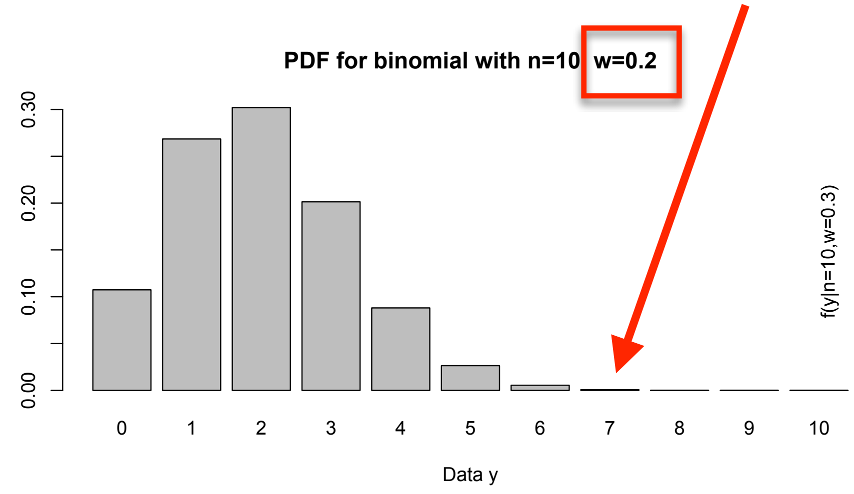


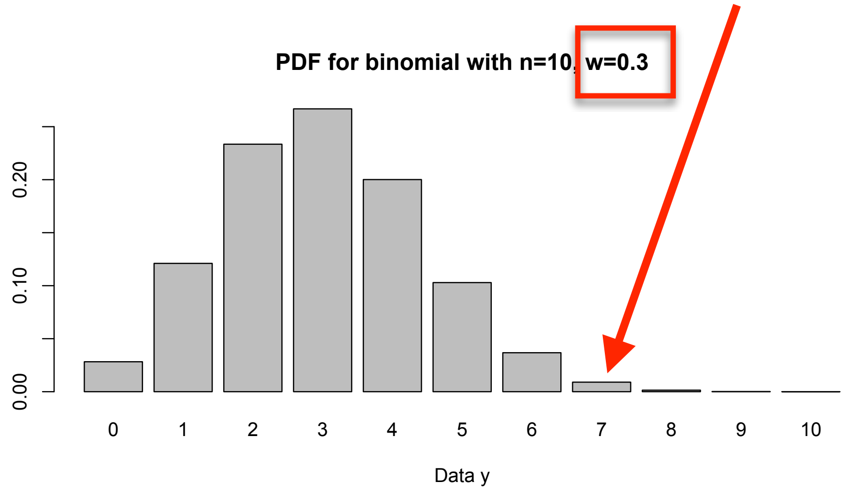
… and so on

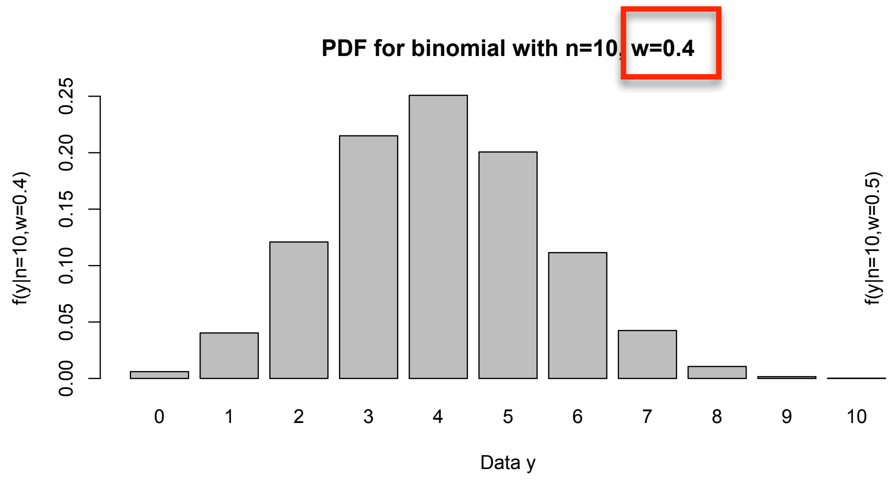
let’s try all values of w between 0.0 and 1.0

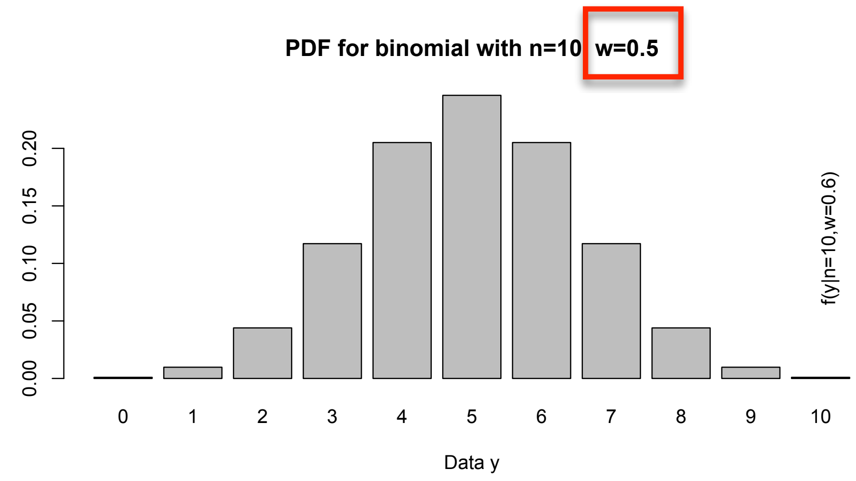
y=7

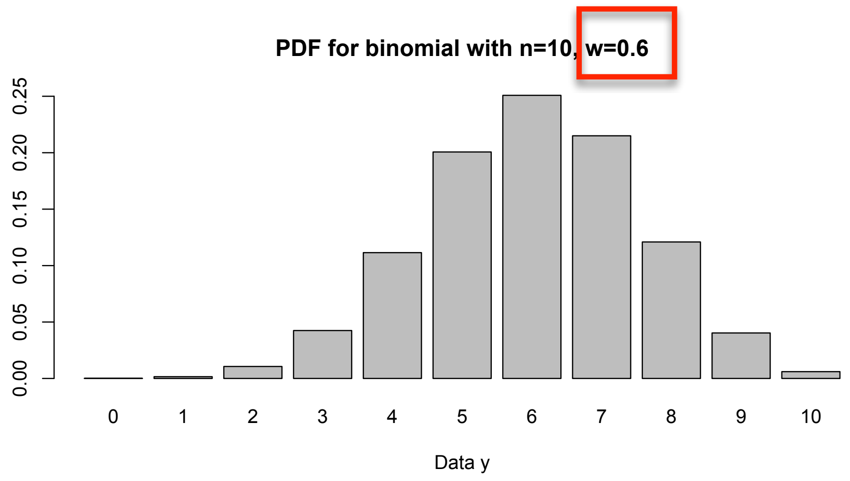


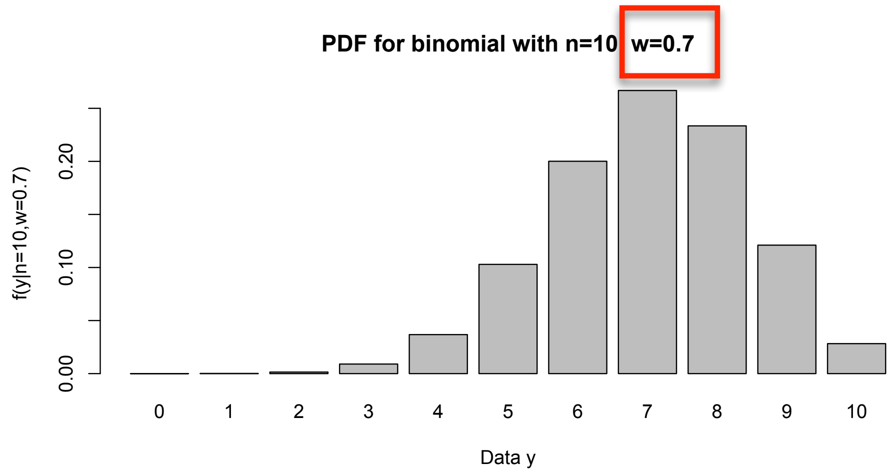








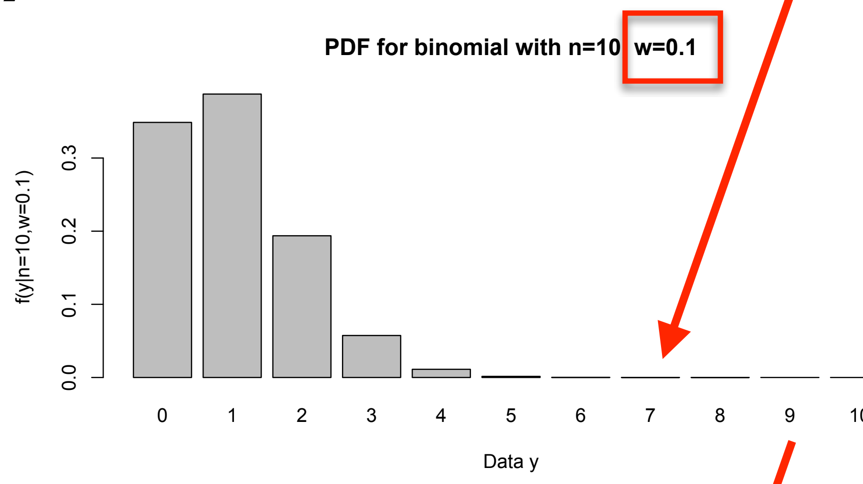


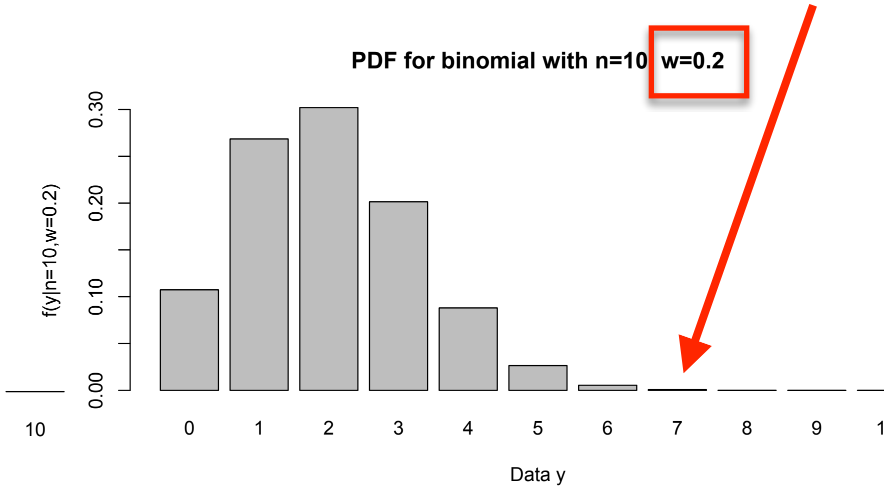


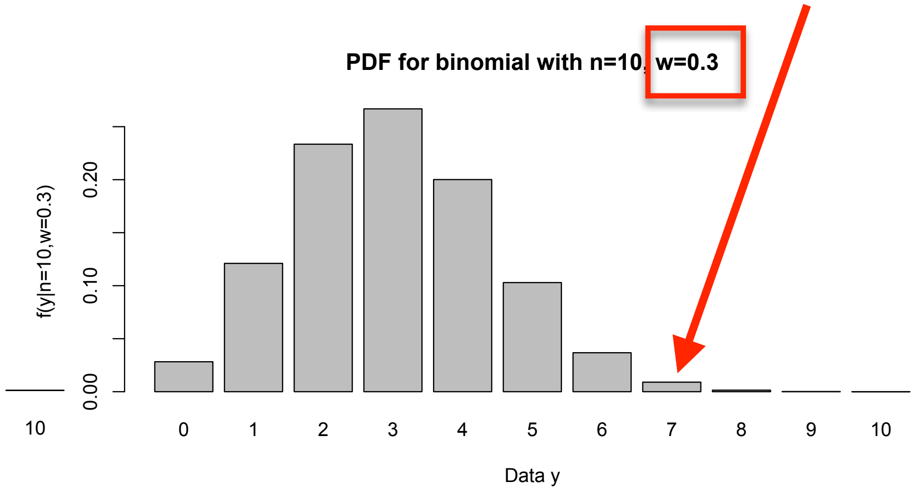
… and so on

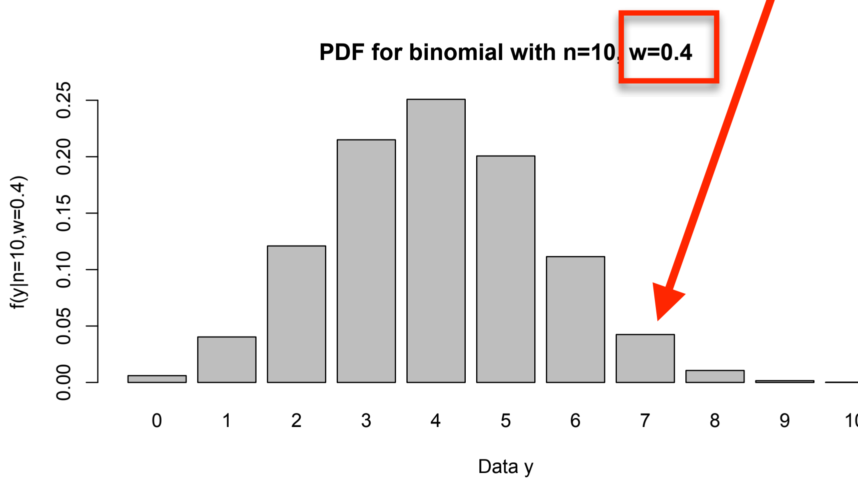
let’s try all values of w between 0.0 and 1.0

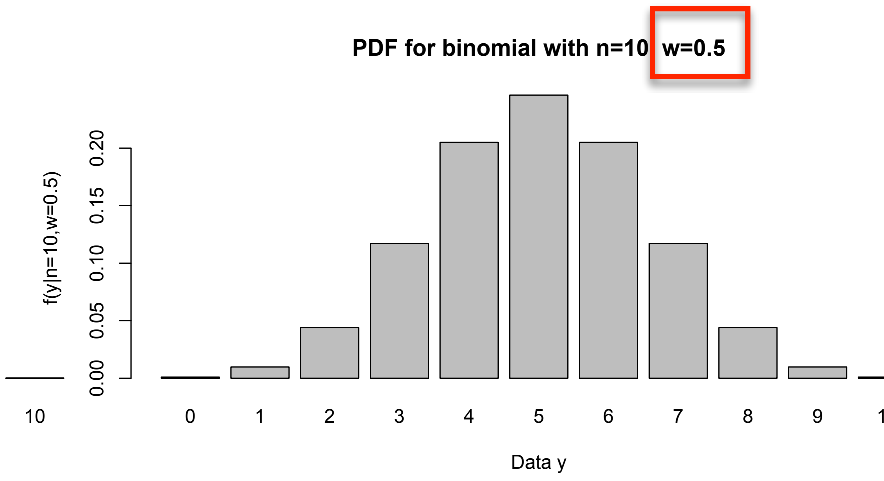
y=7

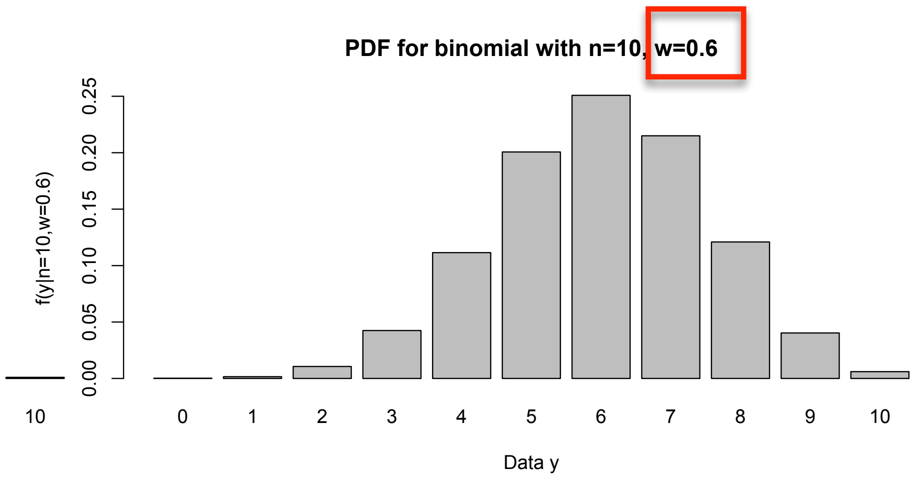


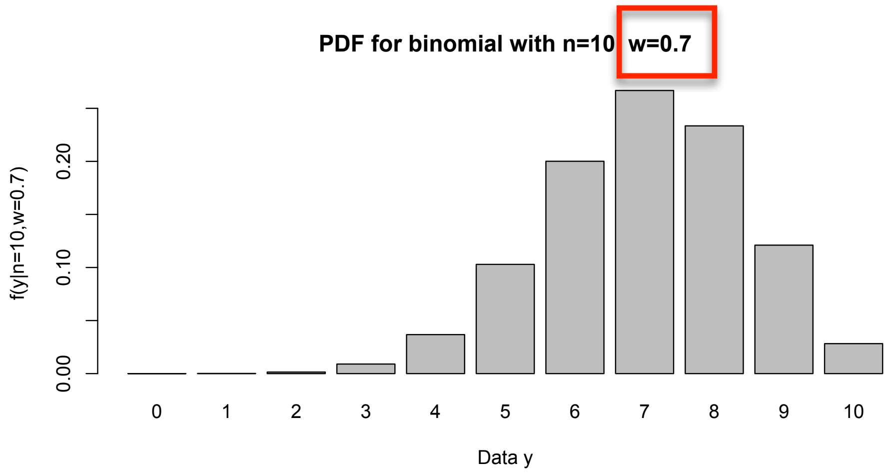








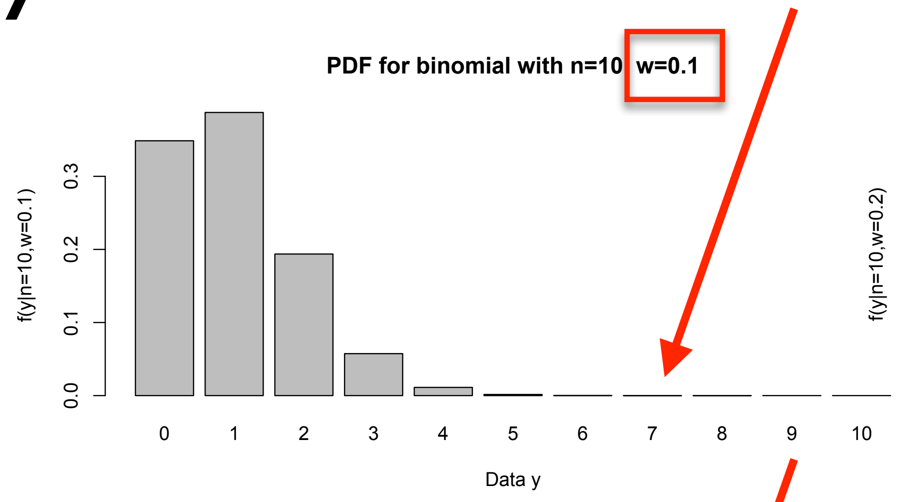


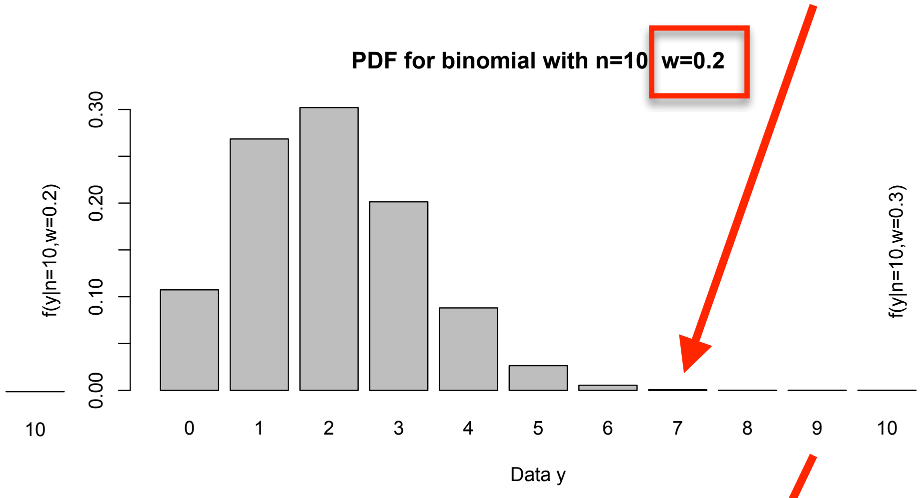


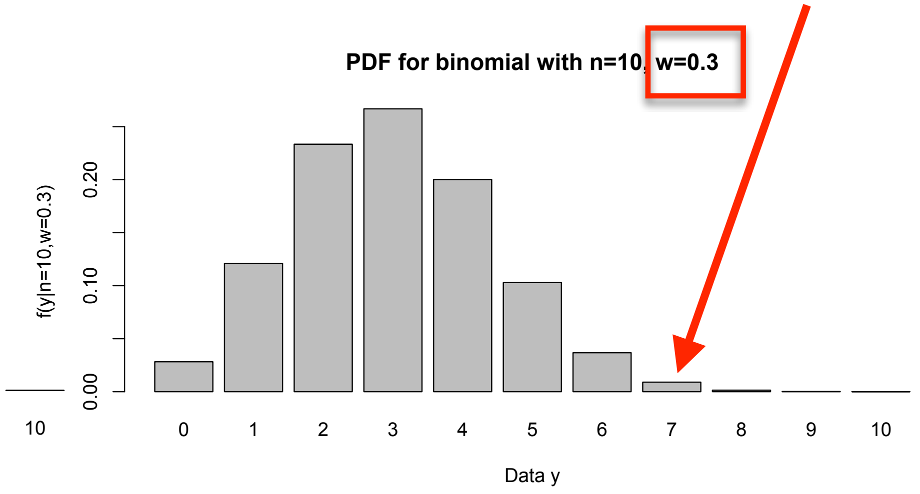
… and so on

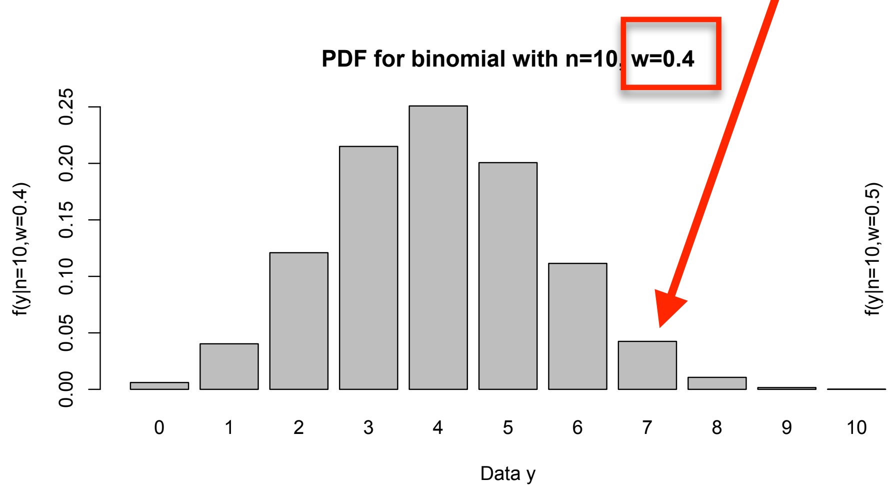
let’s try all values of w between 0.0 and 1.0

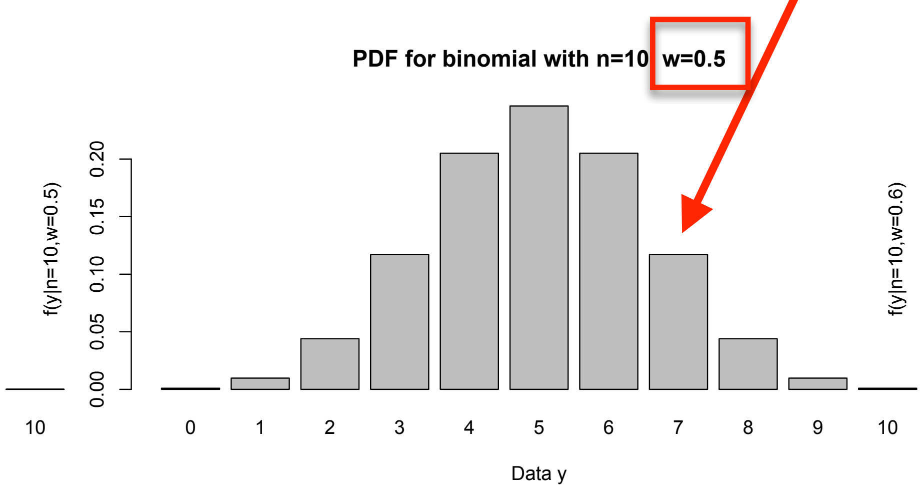
y=7

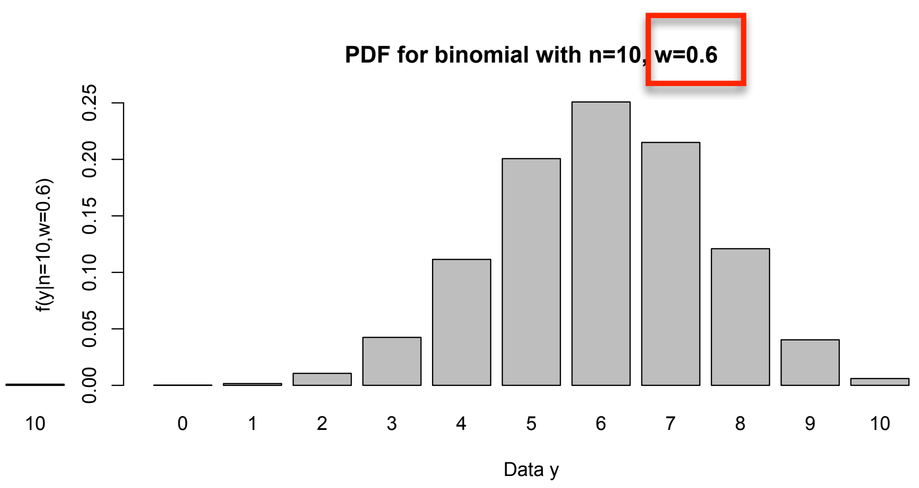


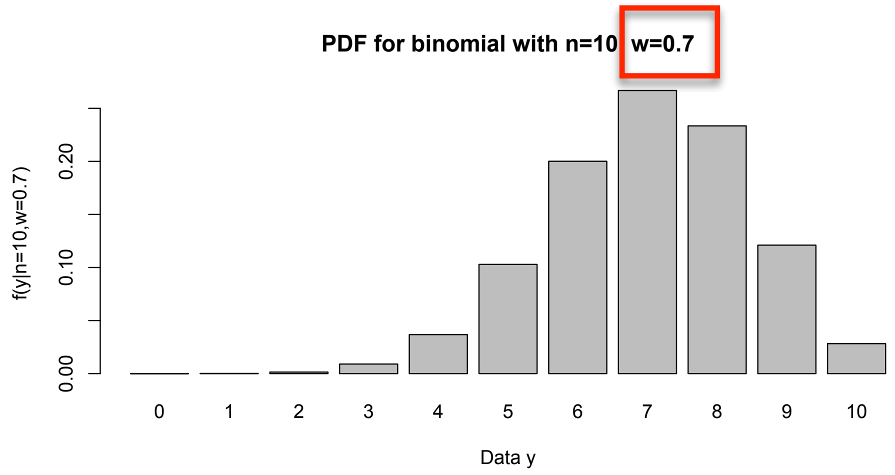








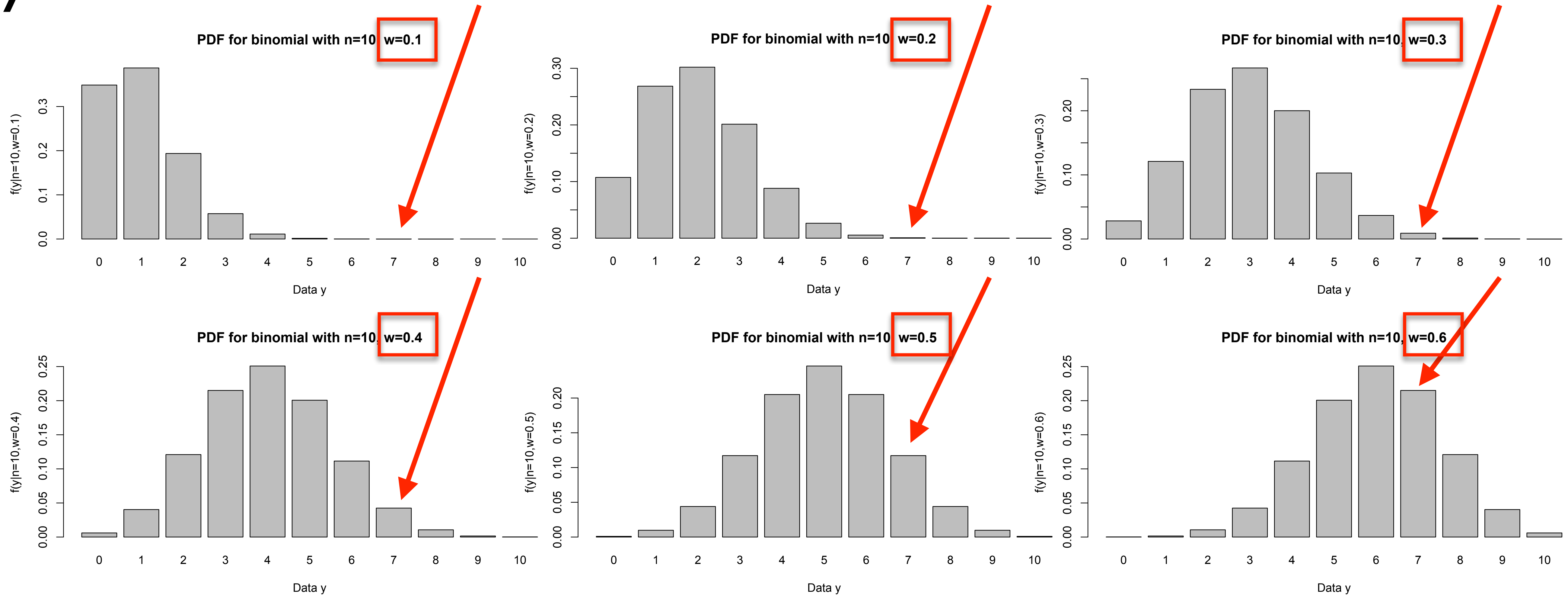


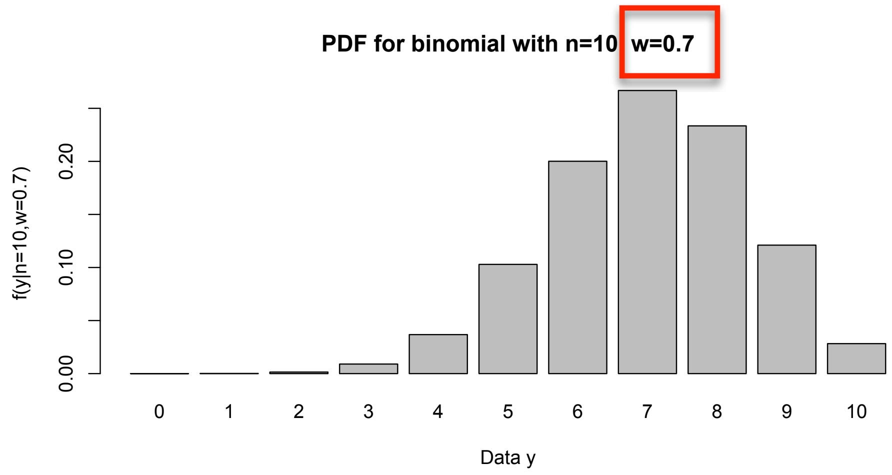


… and so on

let’s try all values of w between 0.0 and 1.0

y=7

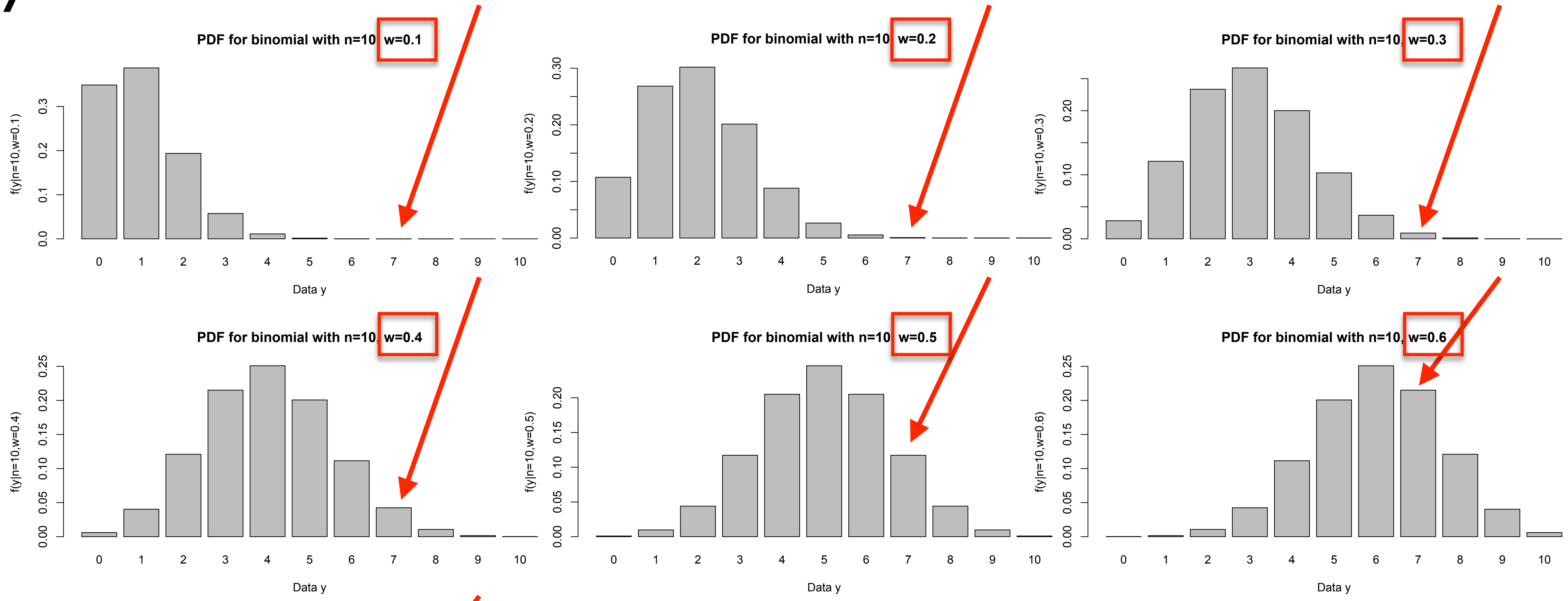


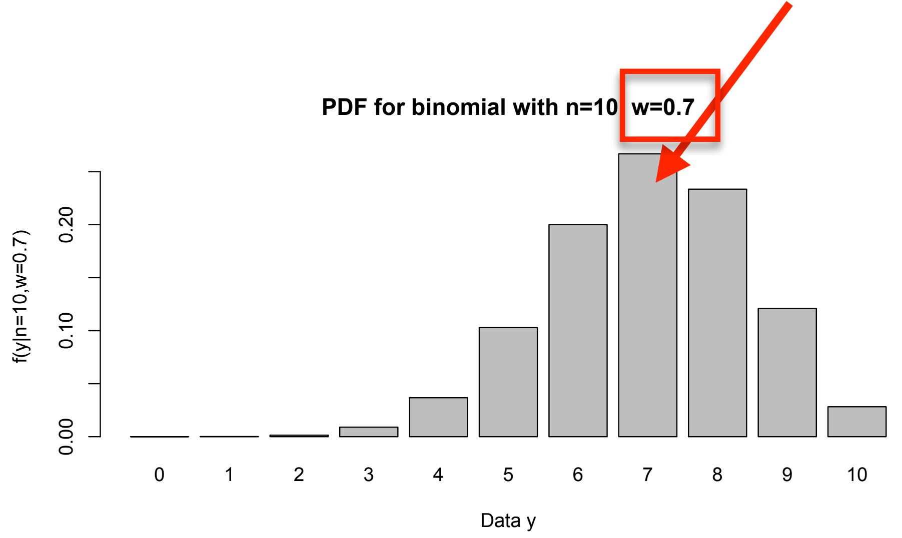


… and so on

let’s try all values of w between 0.0 and 1.0

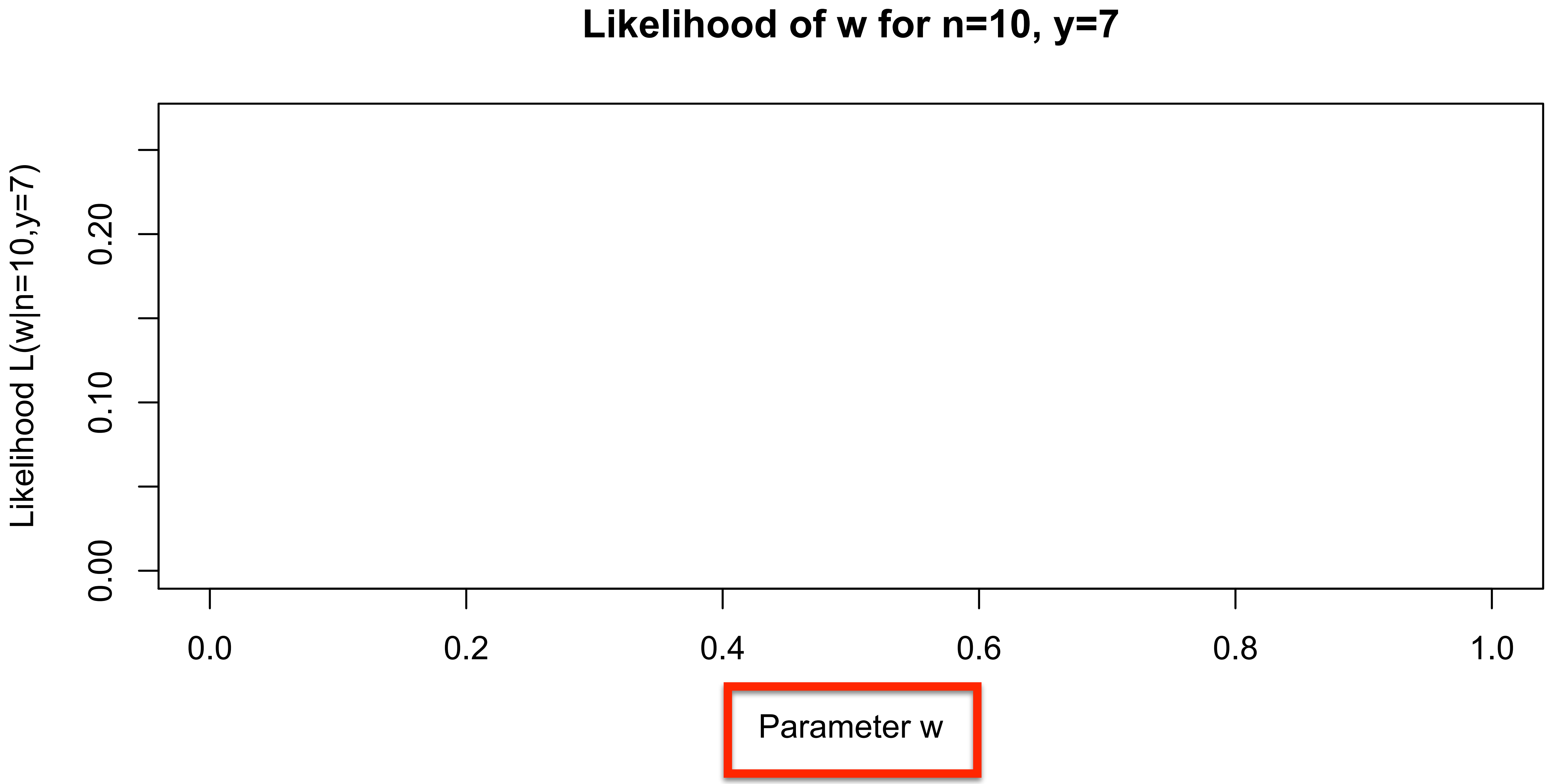
y=7





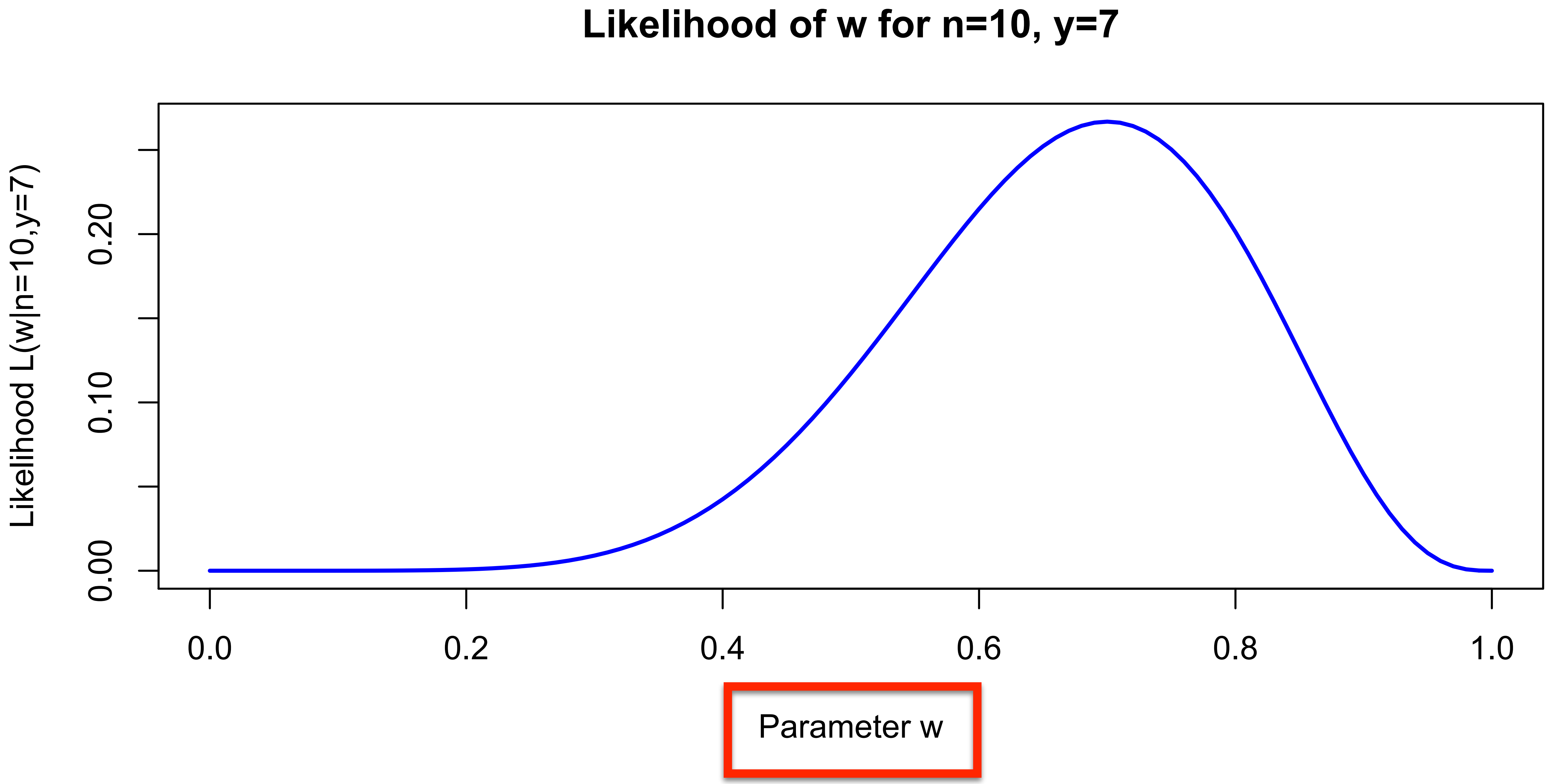
… and so on

let’s try all values of w between 0.0 and 1.0



$$
\begin{aligned}
L(w|n=10, y=7) &= f(y=7|n=10, w) \\
&= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1)
\end{aligned}
$$

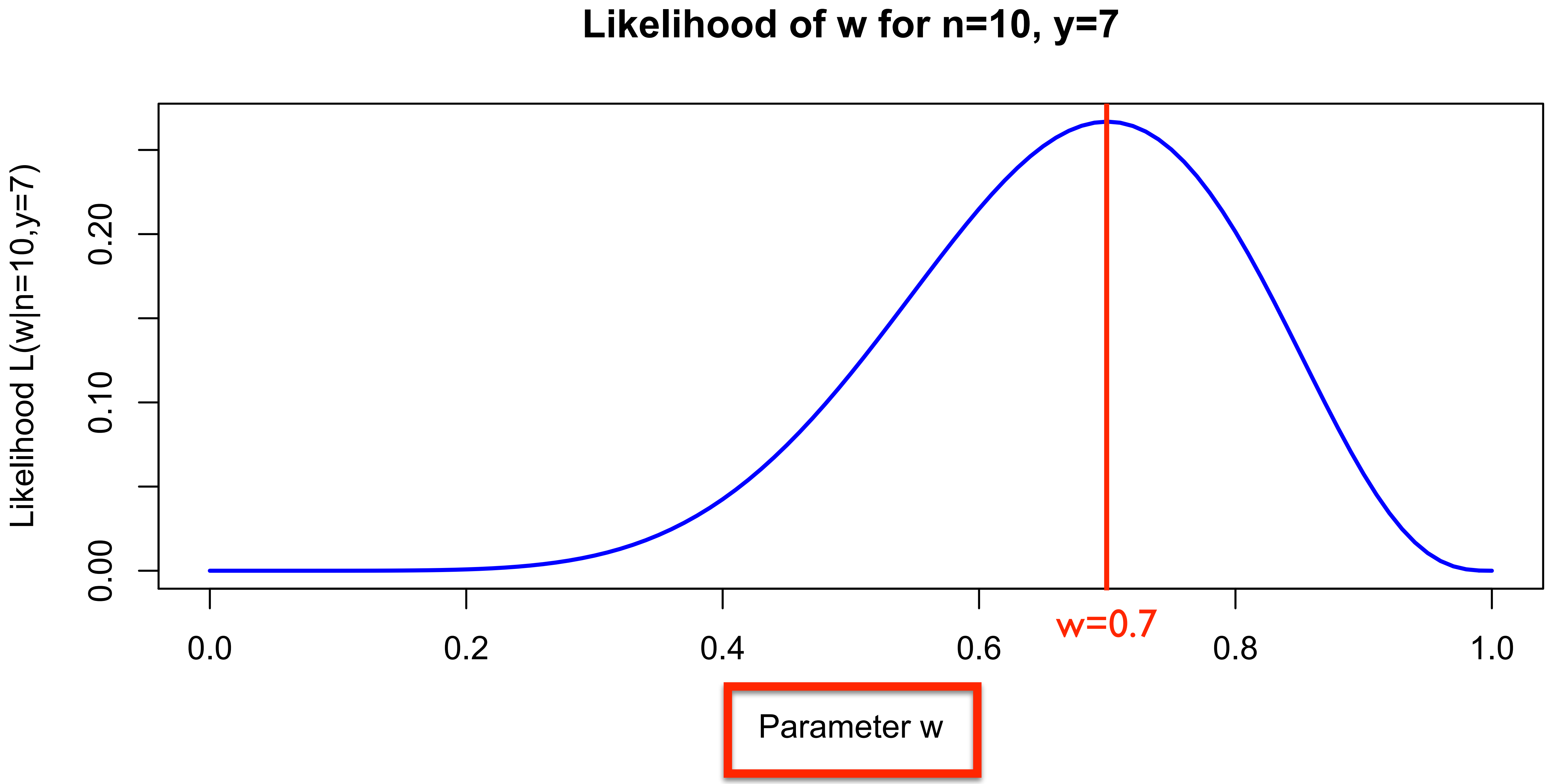
let's try all values of w between 0.0 and 1.0



$$L(w|n=10, y=7) = f(y=7|n=10, w)$$

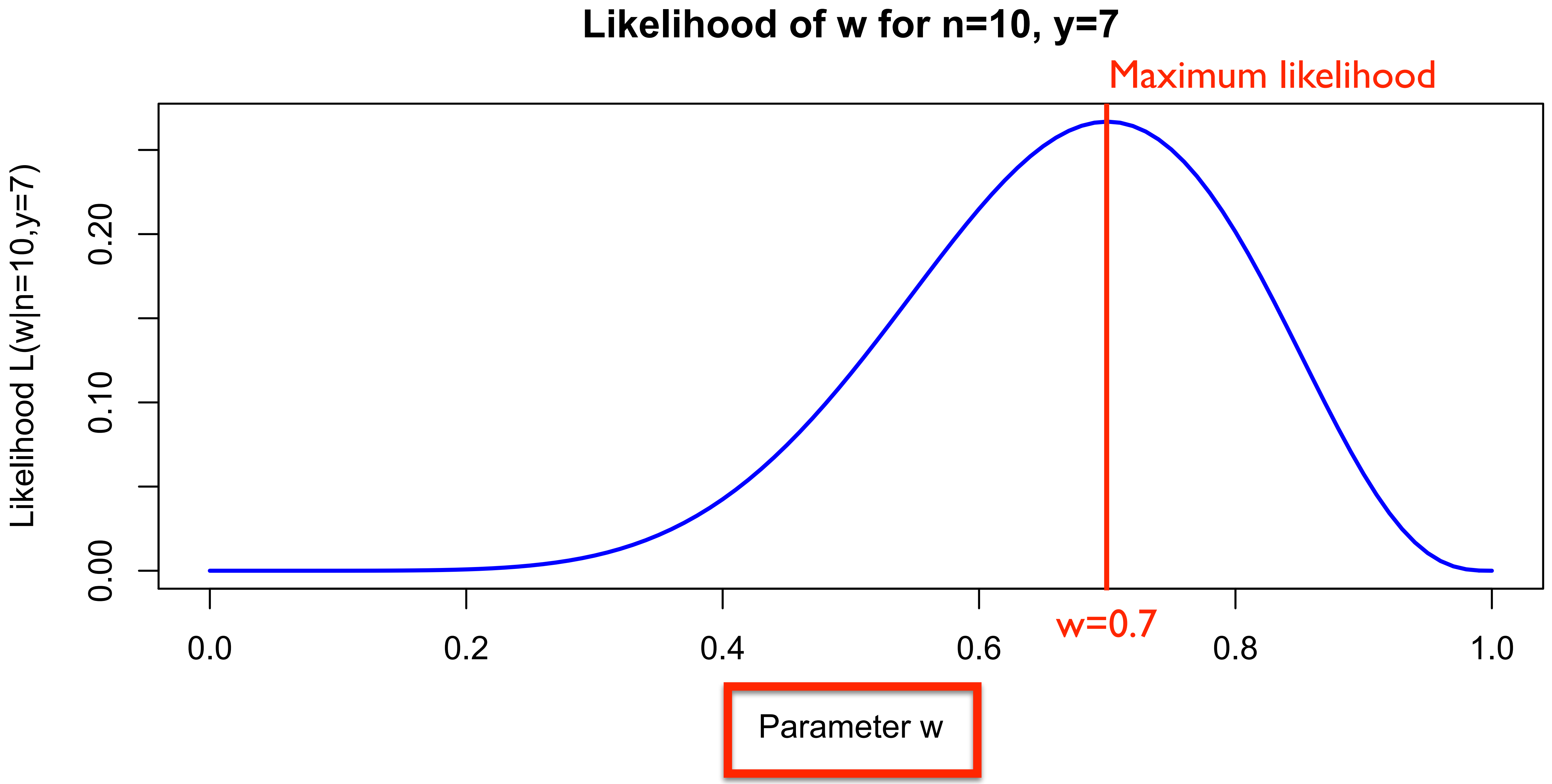
$$= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1)$$

let’s try all values of w between 0.0 and 1.0



$$L(w|n=10, y=7) = f(y=7|n=10, w)$$
$$= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1)$$

let's try all values of w between 0.0 and 1.0

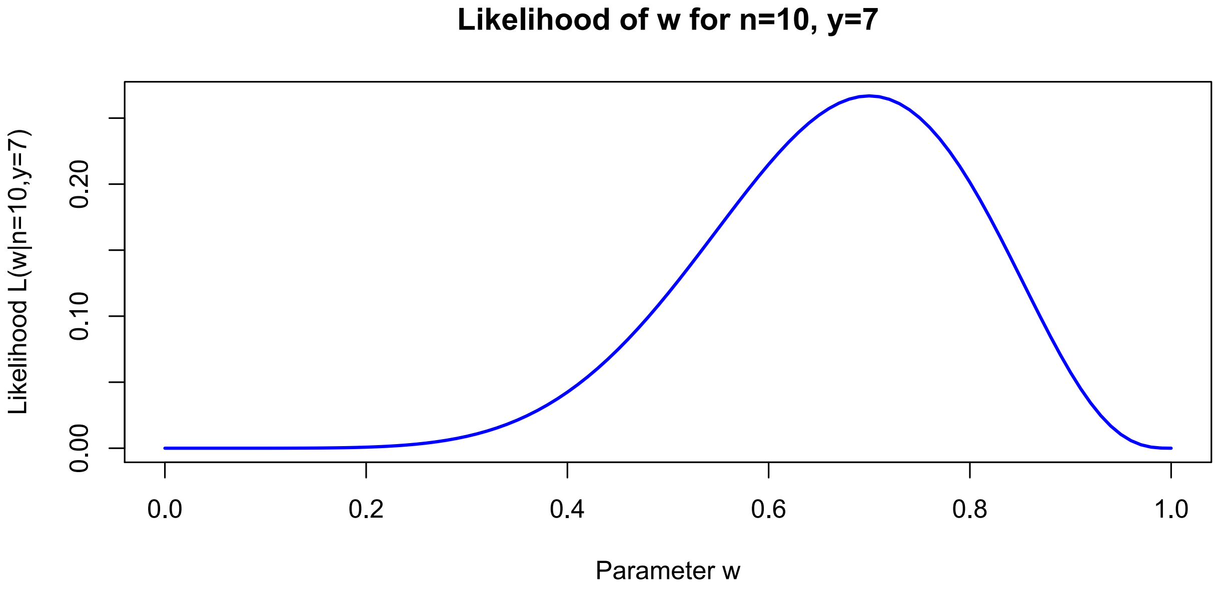


$$
\begin{aligned}
L(w|n=10, y=7) &= f(y=7|n=10, w) \\
&= \frac{10!}{7!3!} w^7 (1-w)^3 \quad (0 \leq w \leq 1)
\end{aligned}
$$

Maximum Likelihood Estimation

- find the probability distribution (the model) that makes the observed data most likely
- seek the value of the **parameter vector w** that maximizes the likelihood function L(w|y)
- the resulting parameter vector w is known as the MLE estimate

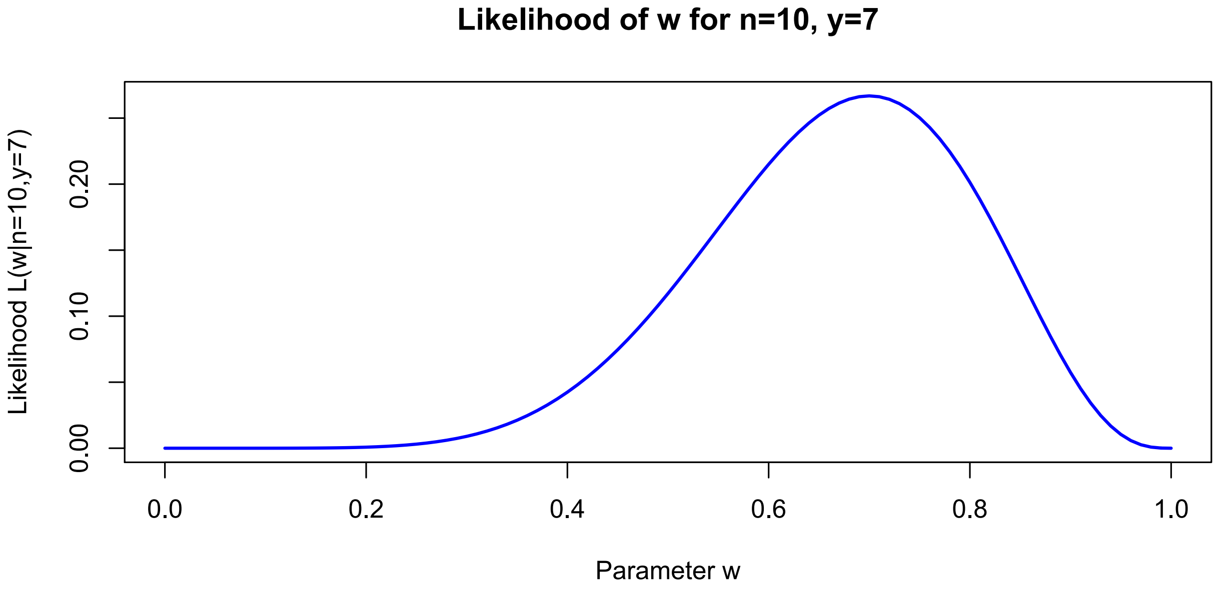
Maximum Likelihood Estimation



- three ways of finding the MLE
- 1. **analytically**: use calculus to solve for the parameter value(s) w that result in a peak
- zero derivative and a negative second derivative

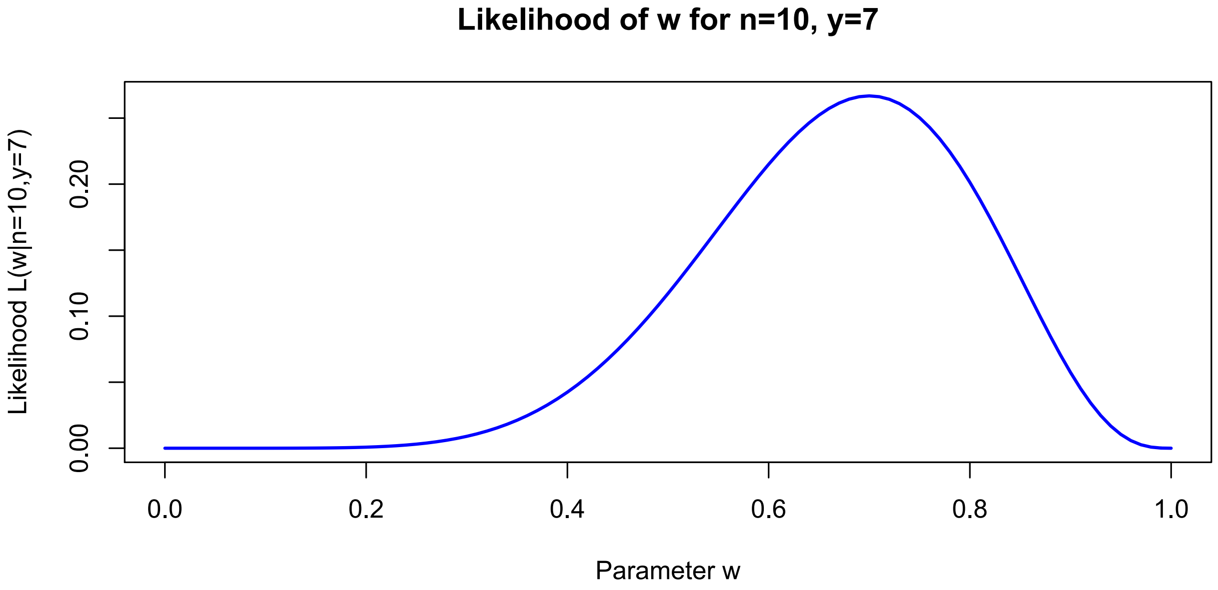
$$\frac{\partial L}{\partial w} = 0 \qquad \frac{\partial^2 L}{\partial^2 w} < 0$$

Maximum Likelihood Estimation



- three ways of finding the MLE
- 2. **grid search**: exhaustive search through parameter space
- (inefficient, could take long time for high dimensional parameter vector)

Maximum Likelihood Estimation



- three ways of finding the MLE
- 3. **numerically**: use non-linear optimization (e.g. gradient descent) to iteratively find the peak

Numerical Considerations

- we saw before that the PDF for observed data, y = (y1, ..., ym) given a parameter vector w, can be expressed as the **product (multiply) of PDFs for individual observations**

$$L(w|y = (y_1, y_2, \ldots, y_n)) = L_1(w|y_1)L_2(w|y_2)\ldots L_n(w|y_n)$$

Numerical Considerations

$$f(y = (y_1, y_2, \ldots, y_n)|w) = f_1(y_1|w) f_2(y_2|w) \ldots f_n(y_n|w)$$

$$p(y = (y_1, y_2, y_3)|\mu, \sigma) = (.010934)(.021297)(.003599) = .000000838$$

- multiplying together a lot of values that lie between 0 and 1, (as many as there are data points) will result in a **very small number**
- in fact the more data, the smaller the resulting product will be
- computers are not good at representing very small numbers

Numerical Considerations

- solution: take the logarithm
- this reformulates the series of products, as a series of **sums**
- the more data, the higher the resulting sum

$$\ln\left[L_1(w|y_1)L_2(w|y_2)\dots L_n(w|y_n)\right] = \ln\left[L_1(w|y_1)\right] + \ln\left[L_2(w|y_2)\right] + \cdots + \ln\left[L_n(w|y_n)\right]$$

Numerical Considerations

- another problem: most optimization algorithms are formulated in terms of **minimizing** an objective function, not maximizing
- solution: rather than maximizing the log-likelihood, we will **minimize the negative log-likelihood**

$$\text{find } w \text{ that minimizes} : -\ln\left[L(w|y)\right]$$

$$\text{find } w \text{ that minimizes} : -\ln\left[L_1(w|y_1)\right] - \ln\left[L_2(w|y_2)\right] - \cdots - \ln\left[L_n(w|y_n)\right]$$

An Example

- Let’s say I claim I can correctly identify espresso brewed with Illy beans (as opposed to Lavazza beans)
- My lab designs an experiment to test me
- They give me 20 cups of coffee in random order and I have to say “Illy” or “Lavazza”
- Observed data: I get 16 correct, 4 incorrect

An Example

- Observed data: I get 16 correct, 4 incorrect
- This experiment can be modelled as 20 Bernoulli trials (outcome of each trial is random and can be either of two possible outcomes, "success" and "failure")
- we know PDF is binomial, which has 2 parameters: **n** (# trials) and **w** (prob of a success on a given trial)

An Example

- we know PDF is binomial, which has 2 parameters: **n** (# trials) and **w** (prob of a success on a given trial)
- what model explains the observed data?
- equivalent to asking, **what is the value of the parameter w**?
- high w (e.g. near 1.0) means I have a good ability to discriminate
- w near 0.5 means I am flipping a coin

Likelihood function

- binomial distribution: gives probability of observing y successes in n trials, given probability w of success on any single trial

$$prob(y|n,w) = \frac{n!}{y!(n-y)!} w^y (1-w)^{n-y}$$

Likelihood function

- in our experiment, n=20, y=16 and w is unknown
- our likelihood function needs to provide likelihood of a particular value of parameter w, given n=20 and y=16

$$L(w|n=20, y=16) = \frac{20!}{16!4!} w^{16}(1-w)^4$$

Likelihood function

- now let’s take the logarithm:

$$L(w|n=20, y=16) = \frac{20!}{16!4!} w^{16} (1-w)^{4}$$

$$\ln\left[L(w|n=20, y=16)\right] = \ln\left[\frac{20!}{16!4!}\right] + 16\ln\left[w\right] + 4\ln\left[(1-w)\right]$$

Find MLE w

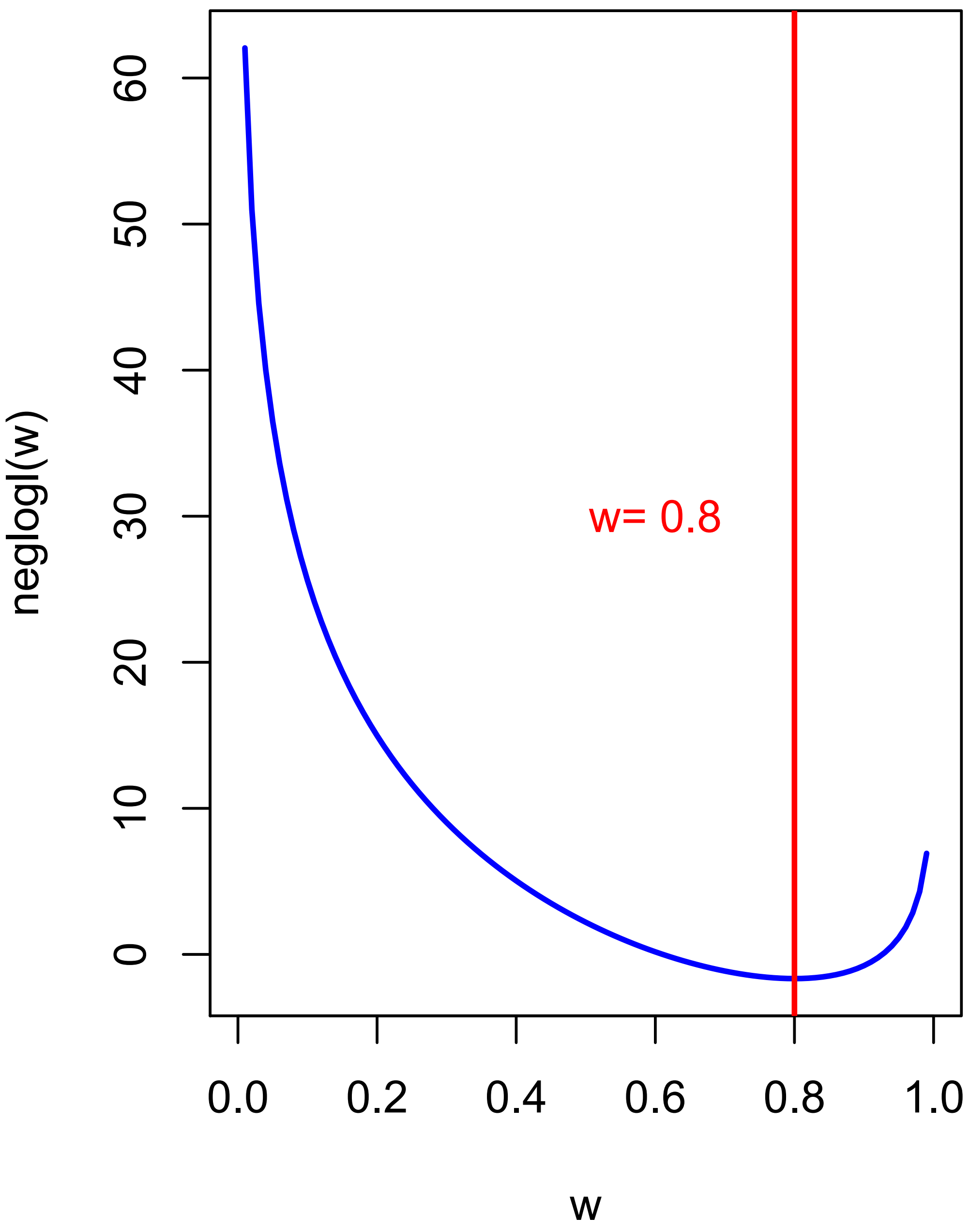
$$\ln\left[L(w|n=20,y=16)\right]=\ln\left[\frac{20!}{16!4!}\right]+16\ln\left[w\right]+4\ln\left[(1-w)\right]$$

- we have our log-likelihood function
- now we need to find w that minimizes the negative log-likelihood

Find MLE for w: brute force

$$\ln\left[L(w|n=20, y=16)\right] = \ln\left[\frac{20!}{16!4!}\right] + 16\ln\left[w\right] + 4\ln\left[(1-w)\right]$$

```
> neglogl <- function(w) {
    loglik <- log(116280) + 16*log(w) + 4*log(1-w)
    return(-1*loglik)
}
> w <- seq(0,1,.01)
> plot(w, neglogl(w), type="l", col="blue", lwd=2)
> imin <- which(neglogl(w)==min(neglogl(w)))
> abline(v=w[imin], col="red", lwd=2)
> text(.6, 30, paste("w=",w[imin]),col="red")
```

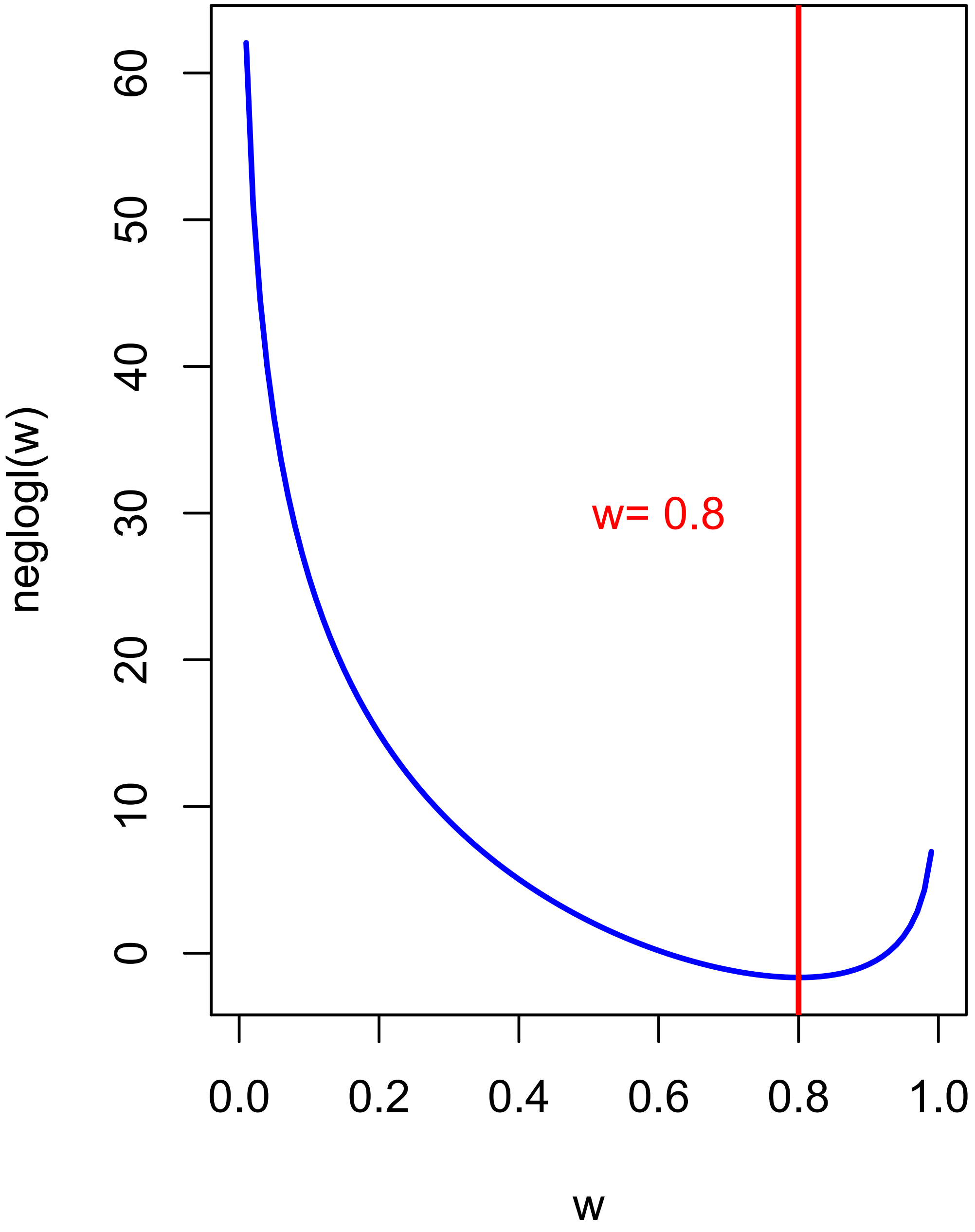


the MLE for w given the data y=16 (and n=20) is w=0.80

Find MLE for w: optimize

$$\ln\left[L(w|n=20,y=16)\right] = \ln\left[\frac{20!}{16!4!}\right] + 16\ln\left[w\right] + 4\ln\left[(1-w)\right]$$

```
> neglogl <- function(w) {
      loglik <- log(116280) + 16*log(w) + 4*log(1-w)
      return(-1*loglik)
}
> nlm(f=neglogl, p=0.5)
$minimum
[1] -1.655708

$estimate
[1] 0.7999995

$gradient
[1] -8.881784e-10

$code
[1] 1

$iterations
[1] 7
```

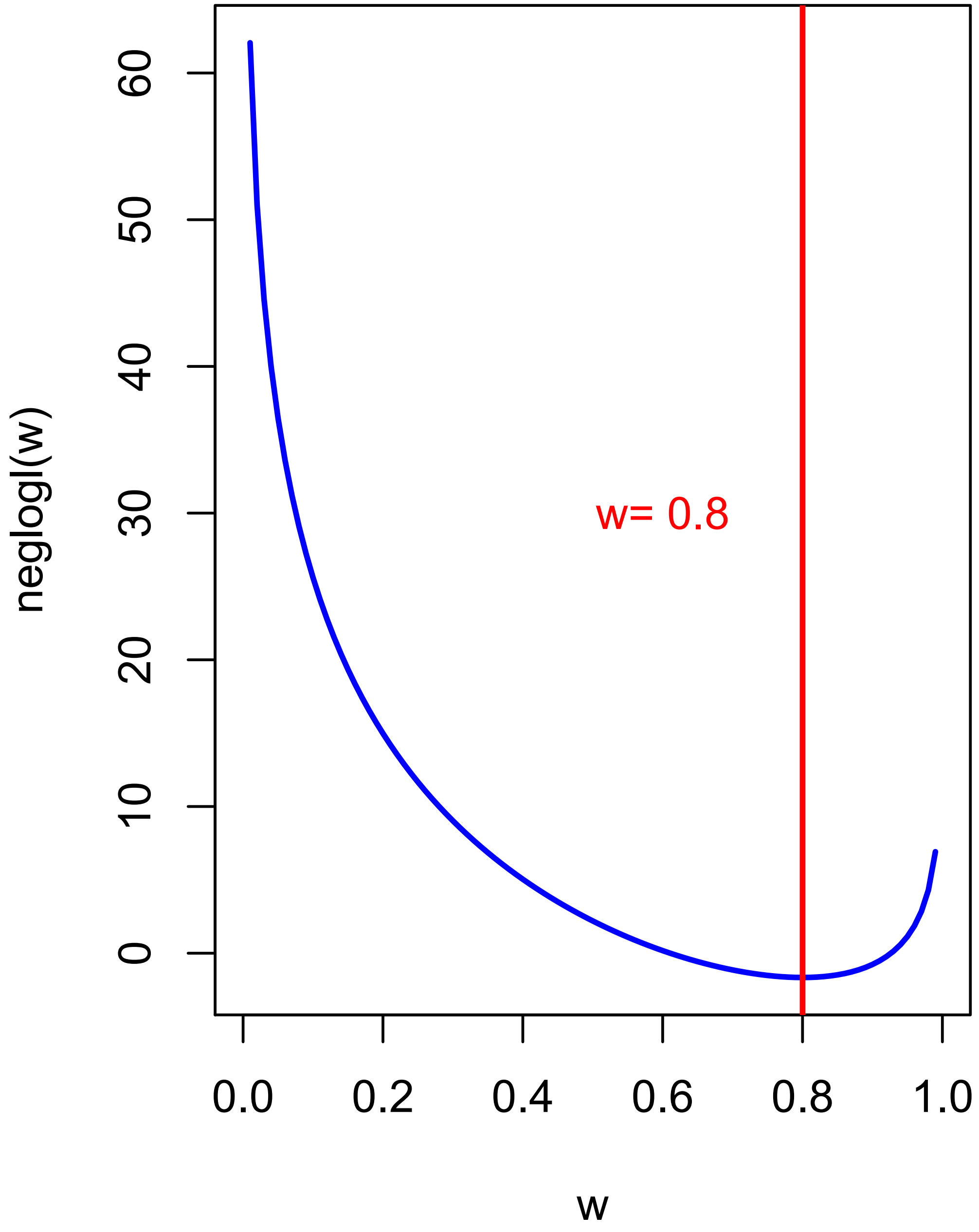


Find MLE for w: optimize

$$\ln\left[L(w|n=20, y=16)\right] = \ln\left[\frac{20!}{16!4!}\right] + 16\ln\left[w\right] + 4\ln\left[(1-w)\right]$$

```
> neglogl <- function(w) {
      loglik <- log(116280) + 16*log(w) + 4*log(1-w)
      return(-1*loglik)
}
> nlm(f=neglogl, p=0.5)
$minimum
[1] -1.655708

$estimate
[1] 0.7999995

$gradient
[1] -8.881784e-10

$code
[1] 1

$iterations
[1] 7
```

a gradient descent optimizer in R



MLE for binomial

- in fact it is known for binomial that MLE for w is equal to y/n
- 16/20
- = 0.80

MLE for binomial

- if we approximate the binomial distribution with a normal distribution (OK for large #s of observations)
- confidence interval is $\hat{w} \pm z_{1-\frac{\alpha}{2}} \sqrt{\frac{\hat{w}(1-\hat{w})}{n}}$
- so 95% confidence interval for Illy is

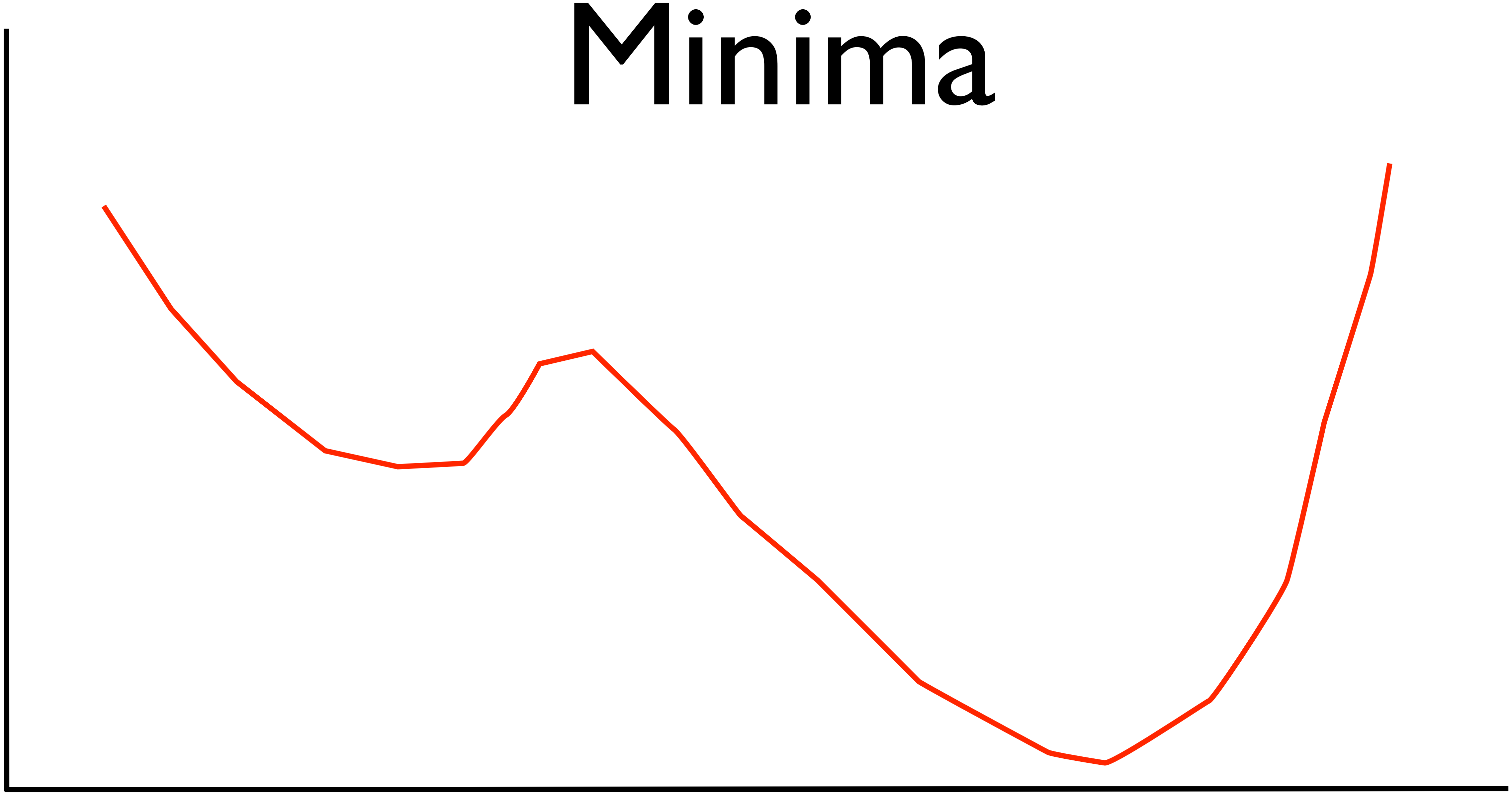
$$0.8 \pm 1.96\sqrt{\frac{0.8(1-0.8)}{20}} = 0.8 \pm 0.175$$

- = 0.625 - 0.975

MLE in general

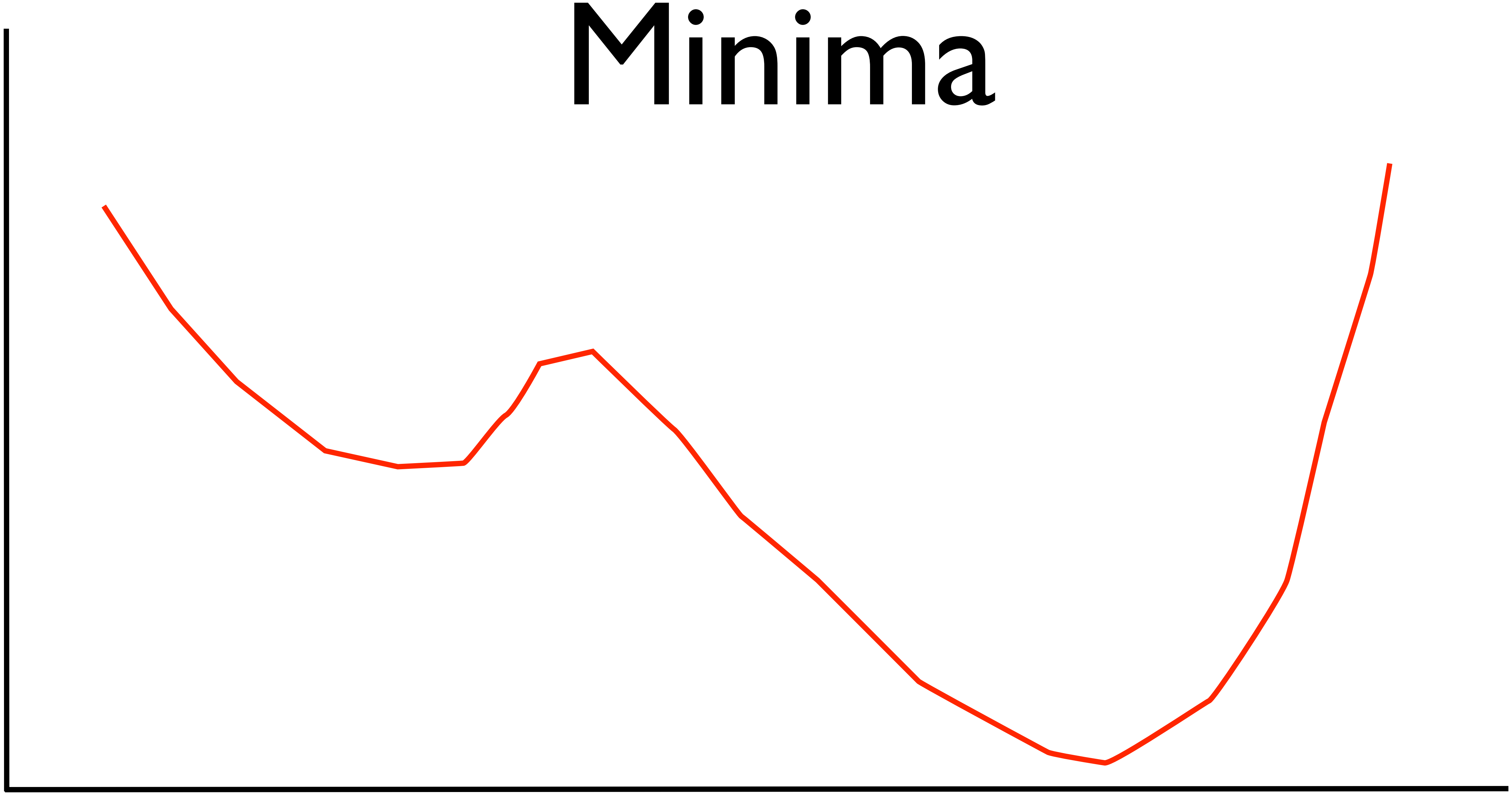
- MLE for many distributions are known (look it up)
- MLE for more complex models can sometimes be determined analytically
- Often however not possible/feasible
- iterative optimization is a common method in these cases

Optimization: Local Minima



- repeat optimization starting from different initial guesses

Optimization: Local Minima



- use stochastic optimization algorithms like simulated annealing

The Bottom Line

- If you can write an equation for the Likelihood function
- i.e. probability of obtaining your observed data, given a model with parameter(s) w
- then you can find the MLE for w
- i.e. you can find the model that is most likely to generate your data

Analytic Solutions: Bernoulli Distribution

$$\text{find } \ w \ \text{ for } \quad \frac{\partial\left(L(w|n,y)\right)}{\partial w} = 0$$

$$\text{gives } \ w = \frac{\sum y_i}{n}$$

- http://mathworld.wolfram.com/MaximumLikelihood.html

Normal Distribution

$$f(x_1,\ldots,x_n|\mu,\sigma) = \prod \frac{1}{\sigma\sqrt{2\pi}} e^{-(x_i-\mu)^2/(2\sigma^2)}$$

$$= \frac{(2\pi)^{-n/2}}{\sigma^n} \exp\left[-\frac{\sum(x_i-\mu)^2}{2\sigma^2}\right]$$

$$\text{so} \quad \ln f = -\frac{1}{2} n \ln(2\pi) - n\ln\sigma - \frac{\sum(x_i-\mu)^2}{2\sigma^2}$$

$$\text{and} \quad \frac{\partial(\ln f)}{\partial\mu} = \frac{\sum(x_i-\mu)}{\sigma^2} = 0$$

$$\text{giving} \quad \hat{\mu} = \frac{\sum x_i}{n}$$

- http://mathworld.wolfram.com/MaximumLikelihood.html

Normal Distribution

$$\text{Similarly,}\quad \frac{\partial(\ln f)}{\partial\sigma} = -\frac{n}{\sigma} + \frac{\sum(x_i-\mu)^2}{\sigma^3} = 0$$

$$\text{gives}\quad \hat{\sigma} = \sqrt{\frac{\sum(x_i-\hat{\mu})^2}{n}}$$

- http://mathworld.wolfram.com/MaximumLikelihood.html

Hypothesis Testing

- We can use the Likelihood Ratio Test to compare two models
- e.g. Illy vs Lavazza example:
- 16 correct out of 20 trials
- our MLE for p was 0.80
- let’s test this against a null hypothesis that p=0.50

Likelihood Ratio test

- test statistic D is a ratio:
- D = -2 log ((likelihood for null model) /
  (likelihood for alternative model))
- D = -2 (log (likelihood null) - log (likelihood alt))
- D = -2 (LL_null - LL_alt)

Likelihood Ratio Test

- the probability distribution of test statistic D is approximately a chi-squared distribution with df = df2-df1
- df2 and df1 are number of free parameters of models 1 (null) and 2 (alternative)

Likelihood Ratio Test

- Illy vs Lavazza:
- null model is L(p=0.5|data)
- alternative model is p for max(L(p|data)) (p=0.8)
- df for null = 0 (no parameters are free to vary)
- df for alt = 1 (p is free to vary)

Likelihood Ratio Test

$$L(p|y,n) = \frac{n!}{y!(n-y)!} p^y (1-p)^{n-y}$$

- D = -2 (LL_null + LL_alt)
- our data: 16 correct and 4 incorrect
- LL_null = log(L(p=0.5 | y=16, n=20)) = -5.38
- MLE of p is p=0.8, so
- LL_alt = log (L(p=0.8 | y=16, n=20)) = -1.52
- D = -2 (-5.38 - -1.52) = -2(-5.38 + 1.52) = 7.72

Likelihood Ratio Test

- D = 7.72
- now compute a p-value using chi-square distribution with df = 1-0 = 1

```
pval <- pchisq(q=7.72, df=1, lower.tail=FALSE)

0.005461
```

Likelihood Ratio Test

- p-value = 0.00546
- we can reject the null with a Type-I error rate of 0.00546 (5 in 10,000)