BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

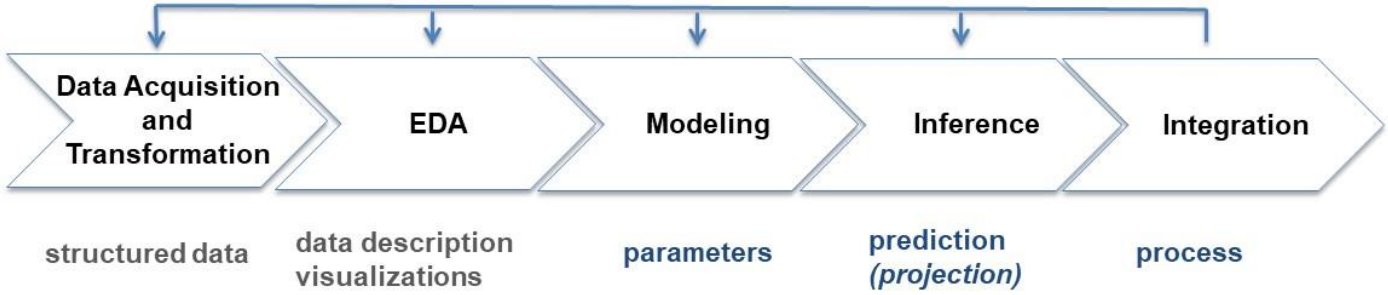
Analytics in Assurance

PCAOB Public Company Accounting Oversight Board

AU Section 329

Substantive Analytical Procedures

"Analytical procedures are an important part of the audit process and consist of evaluations of financial information made by a study of plausible relationships among both financial and nonfinancial data. *Analytical procedures range from simple comparisons to the use of complex models involving many relationships and elements of data"*

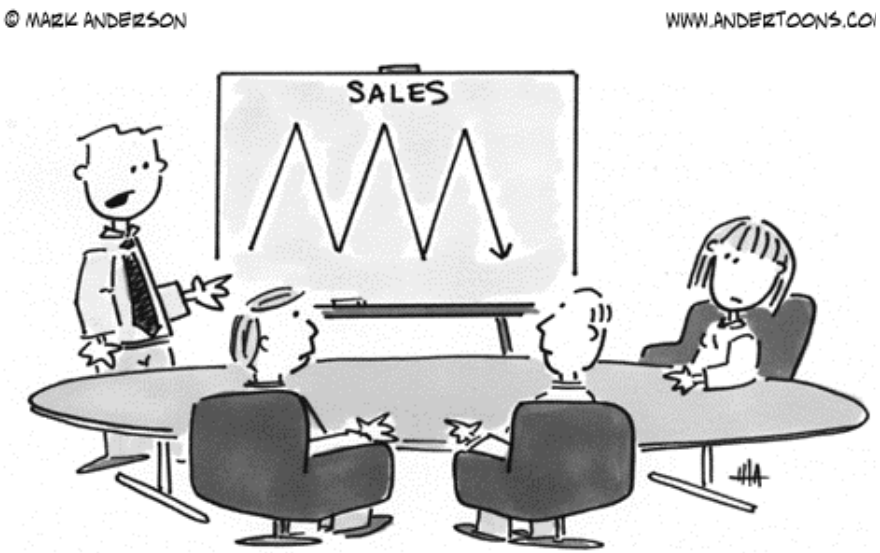


…the auditor develops such expectations by identifying and using plausible ***relationships*** that are ***reasonably expected*** to exist based on the auditor's ***understanding*** of the client and of the industry in which the client operates. The decision about which procedure or procedures to use to achieve a particular audit objective is based on the auditor's ***judgment*** on the expected effectiveness and efficiency of the available procedures.

The PCAOB states in AS2315 §26: "the auditor should ***project*** the misstatement results of the sample to the items from which the sample is selected.

projection (prediction) requires parameters

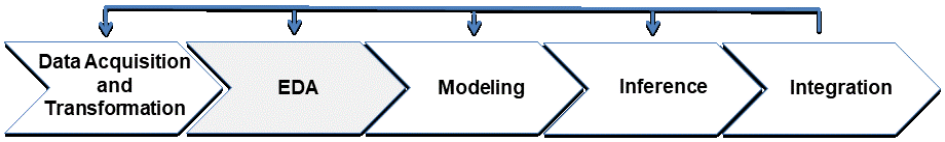
opinions without parameters are… opinions



"..and then another drop this month. But, I have a really good feeling about next month."

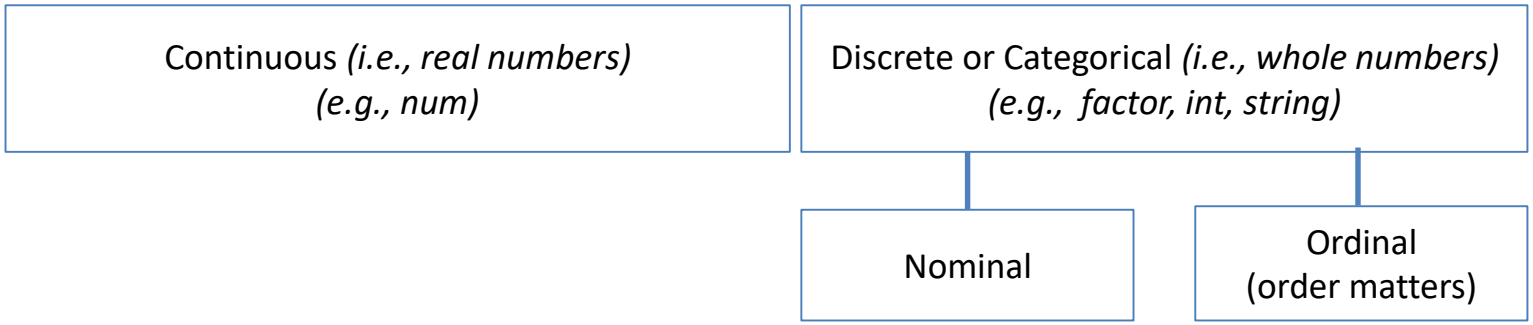
"The dip in sales seems to coincide with the decision to eliminate the sales staff."

Exploratory Data Analysis *(EDA)*



EDA is really the ***description*** *(understanding)* of data. Sometimes it's the end of the analytics process *(e.g., BI dashboards)*, and sometimes it creates the input into modeling. Data description typically begins with data types, distributions and relationships.

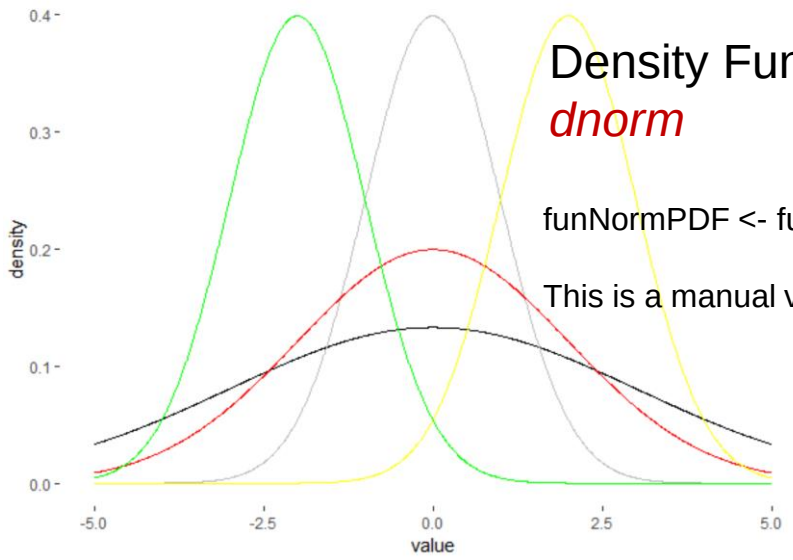
Data types can be *roughly* outlined as follows:



All of these can be combined into vectors (one dimension) or matrices (multiple dimensions) which must be one data type, or data frames (multiple dimensions) which can be multiple data types

Data Distributions are can also be defined as models:

Recall Normal Distribution *(statistics in heaven)* $N(\mu, \sigma^2)$



Density Function *(PDF)*

dnorm

$$\frac{1}{\sqrt{2\pi} * \sigma} \; e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

funNormPDF <- function(x, mean, sd) {1/(sqrt(2*pi)*sd)*exp((-(xmean)^2)/(2*(sd^2)))}

This is a manual version of dnorm in r.

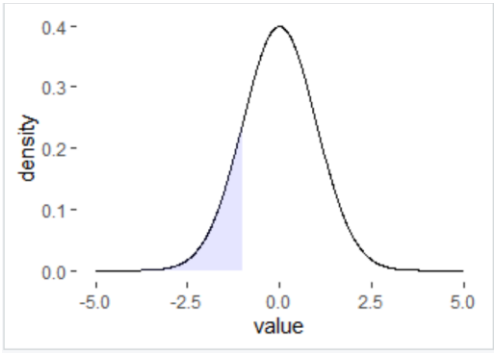
```
> dfDenNorm <- data.frame(x = seq(-5, 5, by = .01))
> p1 <- ggplot(dfDenNorm) +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = .5))) +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = 1)), color = "gray") +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = 2)), color = "red") +
+   geom_line(aes(x,y= dnorm(x, mean = 2, sd = 1)), color = "yellow") +
+   geom_line(aes(x,y= dnorm(x, mean = -2, sd = 1)), color = "green") +
+   theme(panel.background = element_rect(fill = "white"))
> p1
```

moments

mean(x)
sd(x)
skewness(x)
kurtosis(x)

Normal distributions assume a skewness of 0 and a kurtosis of 3, so a truly normal distribution has only 2 parameters $N(\mu, \sigma^2)$

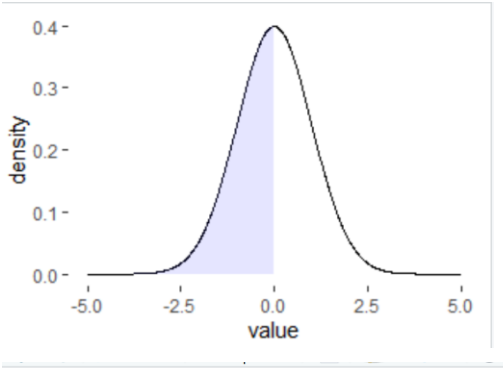
Density and Probability



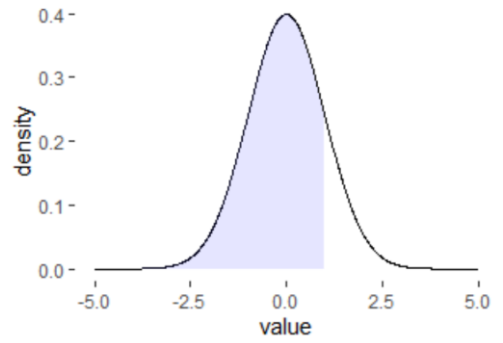
Remember, density is just density – it's a number used to calculate probability *(the area under the density function)*.

(probability of –inf => -1 with mean = 0 and sd = 1)

```
> pnorm(-1, mean = 0, sd = 1)
[1] 0.1586553
```




```
> pnorm(0, mean = 0, sd = 1)
[1] 0.5
```



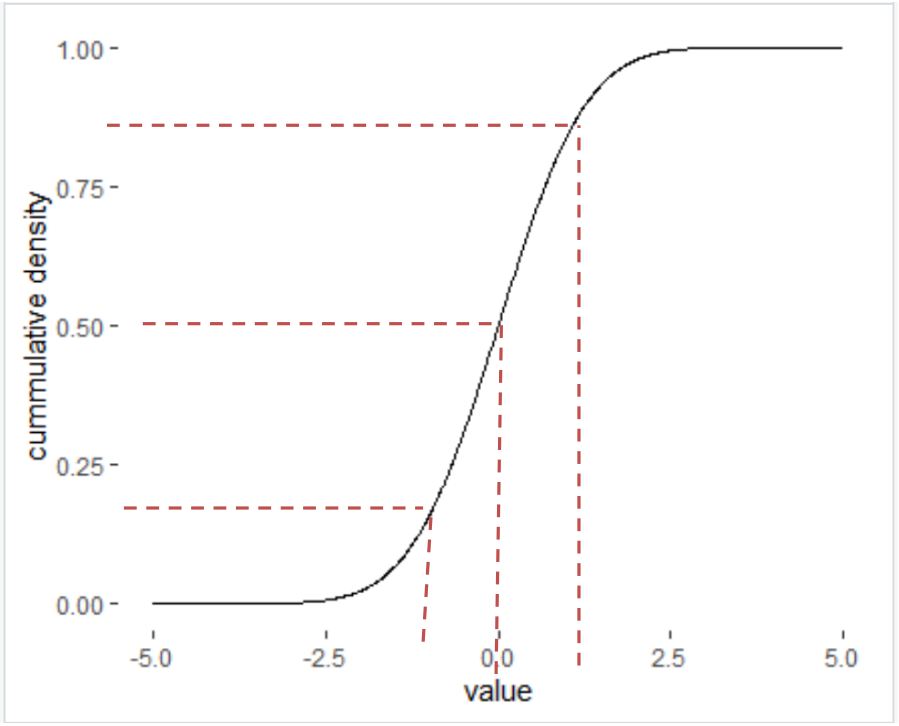

```
> pnorm(1, mean = 0, sd = 1)
[1] 0.8413447
```

Cumulative Distribution Function

```
> dfData <- data.frame(x = seq(-5, 5, by = .01))
> # x axis becomes x values - y is cummulative prob
> p1 <- ggplot(dfData) +
+   geom_line(aes(x,y= pnorm(x, mean = 0, sd = 1))) +
+   xlab("value") +
+   ylab("cummulative density") +
+   theme(panel.background = element_rect(fill = "white"))
> p1
```

We can get this probability with a cumulative distribution function

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt$$

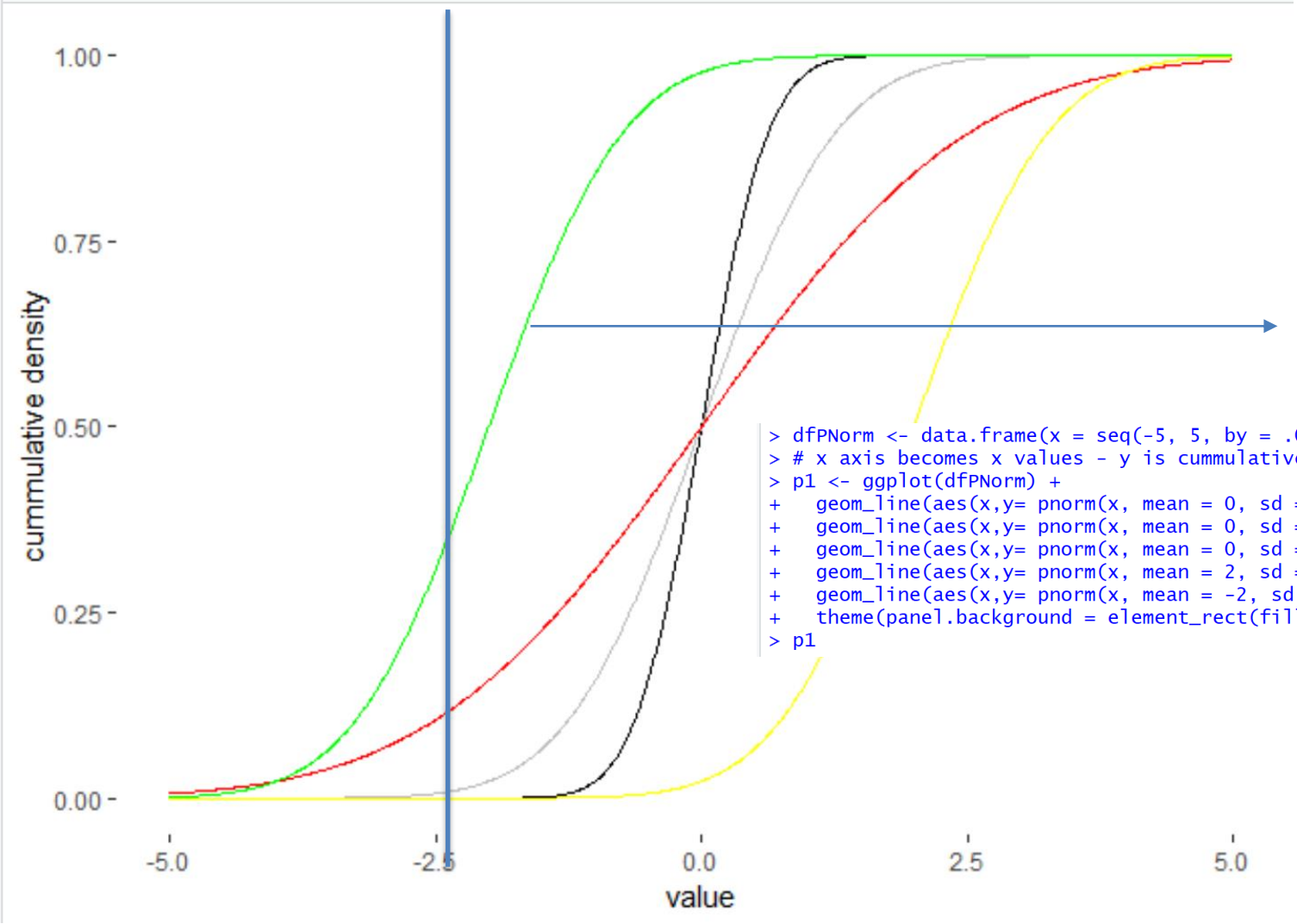



```
> pnorm(1, mean = 0, sd = 1)
[1] 0.8413447
```

```
> pnorm(0, mean = 0, sd = 1)
[1] 0.5
```

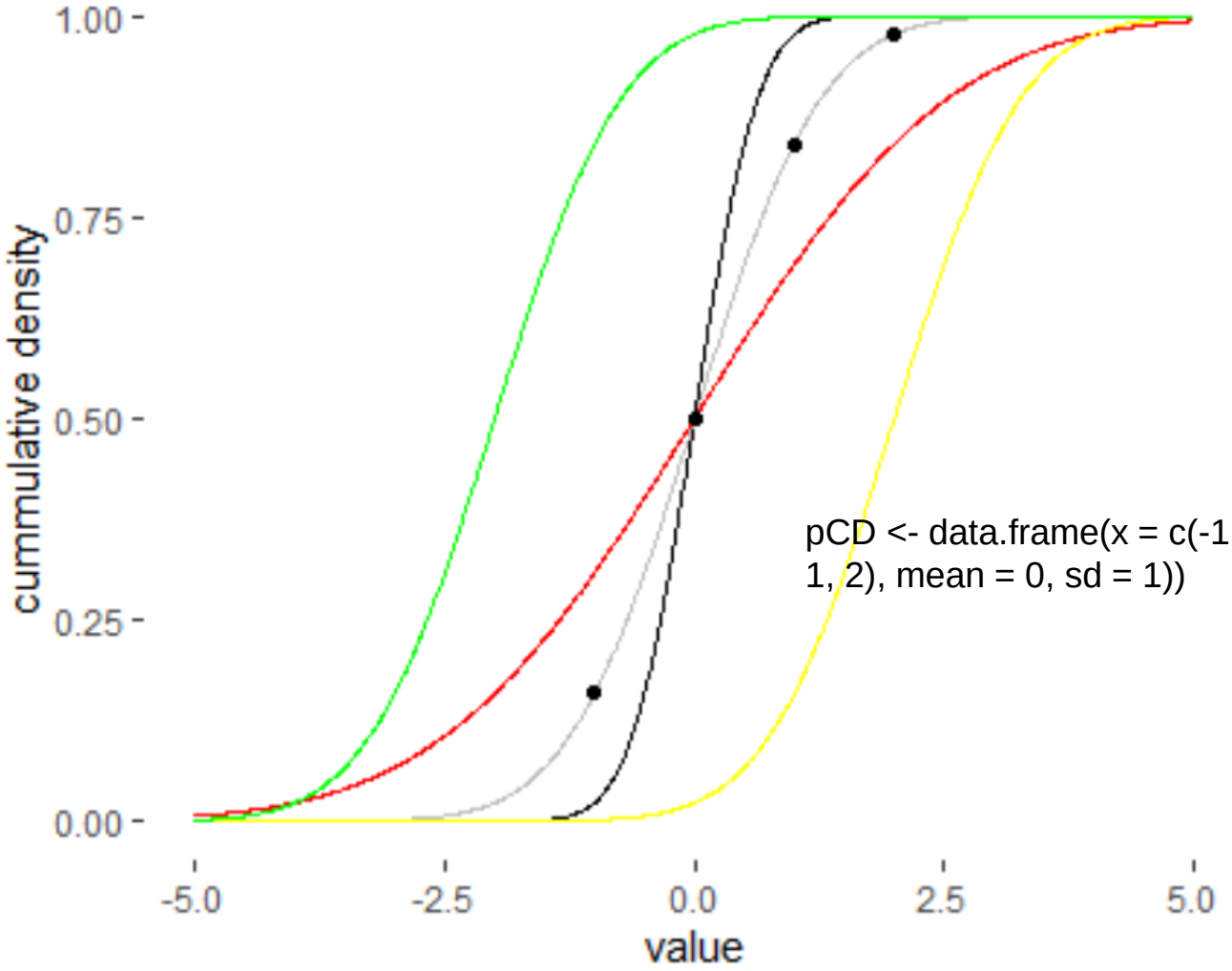
```
> pnorm(-1, mean = 0, sd = 1)
[1] 0.1586553
```

Getting the probability of a range is easy – just subtract the lower limit pnorm from the higher limit pnorm




```
> dfPNorm <- data.frame(x = seq(-5, 5, by = .01))
> # x axis becomes x values - y is cummulative prob
> p1 <- ggplot(dfPNorm) +
+   geom_line(aes(x,y= pnorm(x, mean = 0, sd = .5))) +
+   geom_line(aes(x,y= pnorm(x, mean = 0, sd = 1)), color = "gray") +
+   geom_line(aes(x,y= pnorm(x, mean = 0, sd = 2)), color = "red") +
+   geom_line(aes(x,y= pnorm(x, mean = 2, sd = 1)), color = "yellow") +
+   geom_line(aes(x,y= pnorm(x, mean = -2, sd = 1)), color = "green") +
+   theme(panel.background = element_rect(fill = "white"))
> p1
```

Notice how the red curve develops more slowly (sd = 2 vs. sd = 1)




```
pCD <- data.frame(x = c(-1, 0, 1, 2), y = pnorm(c(-1, 0,
1, 2), mean = 0, sd = 1))
```

Quantiles

You're probably familiar with quartiles *(which is a specific quantile – default with quantile function)*

```
> dfData <- data.frame(x = seq(0, 10, by = .1))
> # since we're using a sequence, it's already ordered
> quantile(dfData$x) # these are quartiles (it orders it for you)
  0%  25%  50%  75% 100%
 0.0  2.5  5.0  7.5 10.0
```

```
> dfData <- data.frame(x = rnorm(n = 1000, mean = 0, sd = 1))
> quantile(dfData$x) # so this orders it for you
          0%          25%          50%          75%         100%
-3.143070342 -0.670856655  0.005698326  0.662786794  3.274741901
> # it doesn't have to be quartiles:
> quantile(dfData$x, probs = c(0.05, 0.95))
       5%       95%
-1.594185  1.684531
```

…but it doesn't have to be quartiles

You might notice that quantile is the opposite (or more properly, inverse) of the CDF

```
> pnorm(0, mean = 0, sd = 1)
[1] 0.5
```

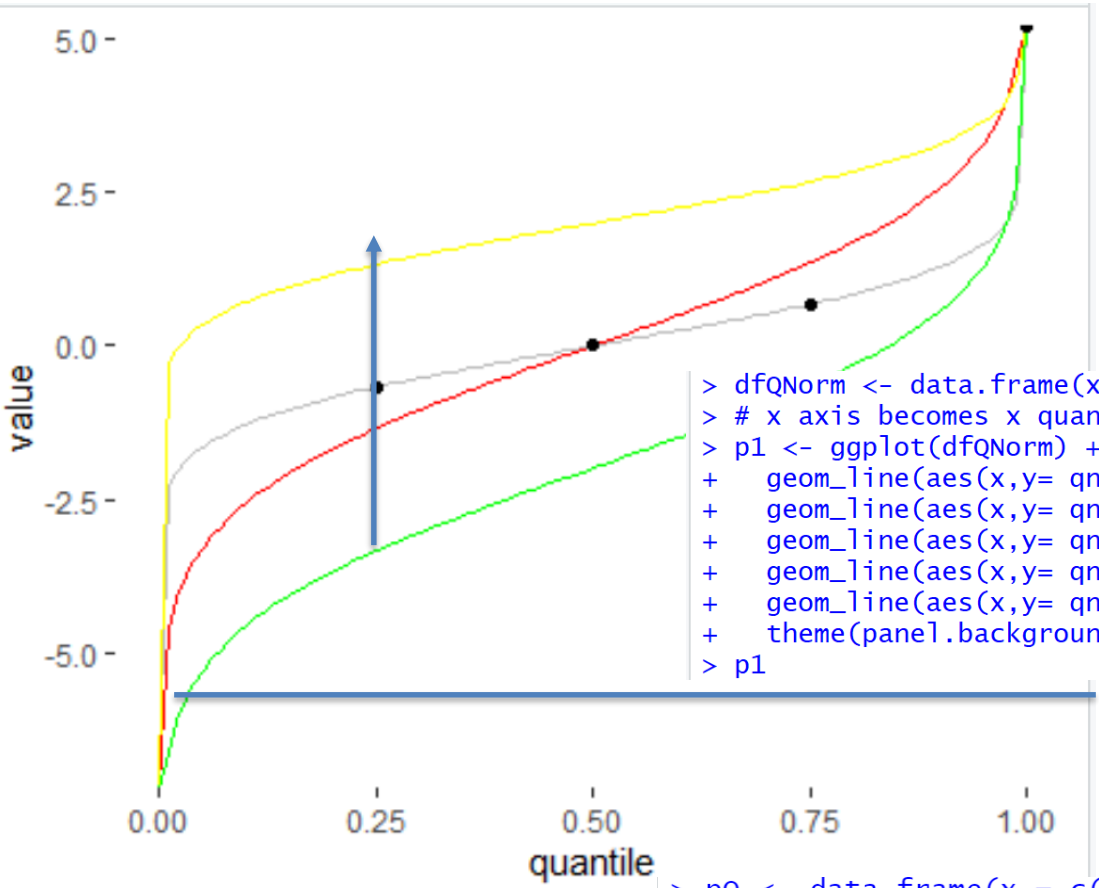
```
> qnorm(.5, mean = 0, sd = 1)
[1] 0
```

$\Phi^{-1}(p)$

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Quantile *(a quantile is just an ordering of your data)*

qnorm



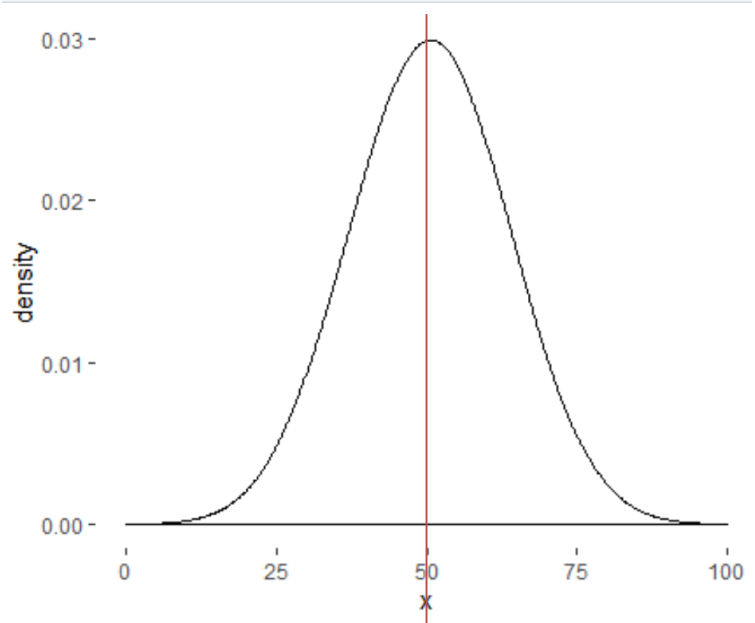
Again, compare the red and black with different spread

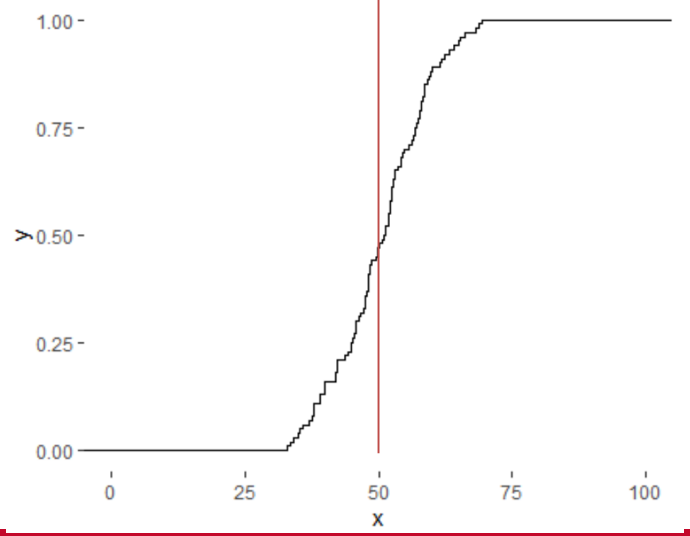
```
> dfQNorm <- data.frame(x = seq(0, 1, by = .01))
> # x axis becomes x quantiles
> p1 <- ggplot(dfQNorm) +
+   geom_line(aes(x,y= qnorm(x, mean = 0, sd = .5))) +
+   geom_line(aes(x,y= qnorm(x, mean = 0, sd = 1)), color = "gray") +
+   geom_line(aes(x,y= qnorm(x, mean = 0, sd = 2)), color = "red") +
+   geom_line(aes(x,y= qnorm(x, mean = 2, sd = 1)), color = "yellow") +
+   geom_line(aes(x,y= qnorm(x, mean = -2, sd = 1)), color = "green") +
+   theme(panel.background = element_rect(fill = "white"))
> p1
```

```
> pQ <- data.frame(x = c(.25, .5, .75, 1),
+                  y = qnorm(c(.25, .5, .75, 1), mean = 0, sd = 1))
> p1 <- p1 + geom_point(data  = pQ, aes(x, y))
> p1
```

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

combining distribution analysis functions

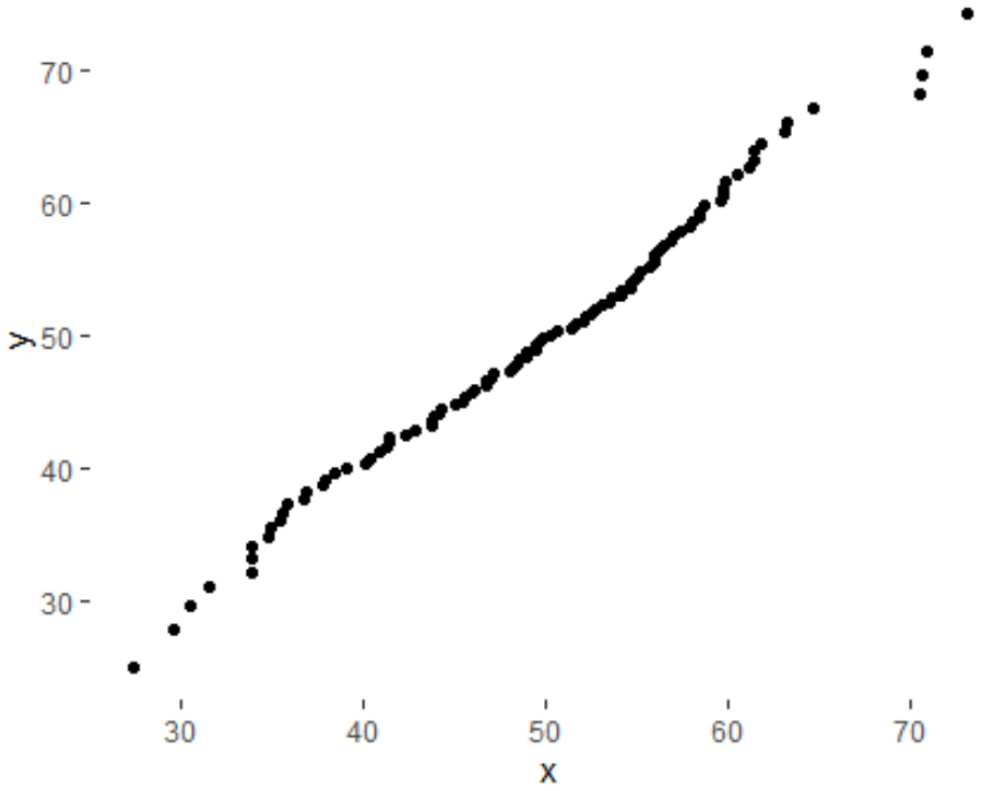





```
> dfData <- data.frame(x = rnorm(n = 100, mean = 50, sd = 10))
> p1b <- ggplot(dfData, aes(x)) + geom_density(bw = 10) +
+   theme(panel.background = element_rect(fill = "white")) +
+   xlim(c(0, 100))
> p1b
> p1b <- ggplot(dfData, aes(x)) + stat_ecdf() +
+   theme(panel.background = element_rect(fill = "white")) +
+   xlim(c(0, 100))
> p1b
```

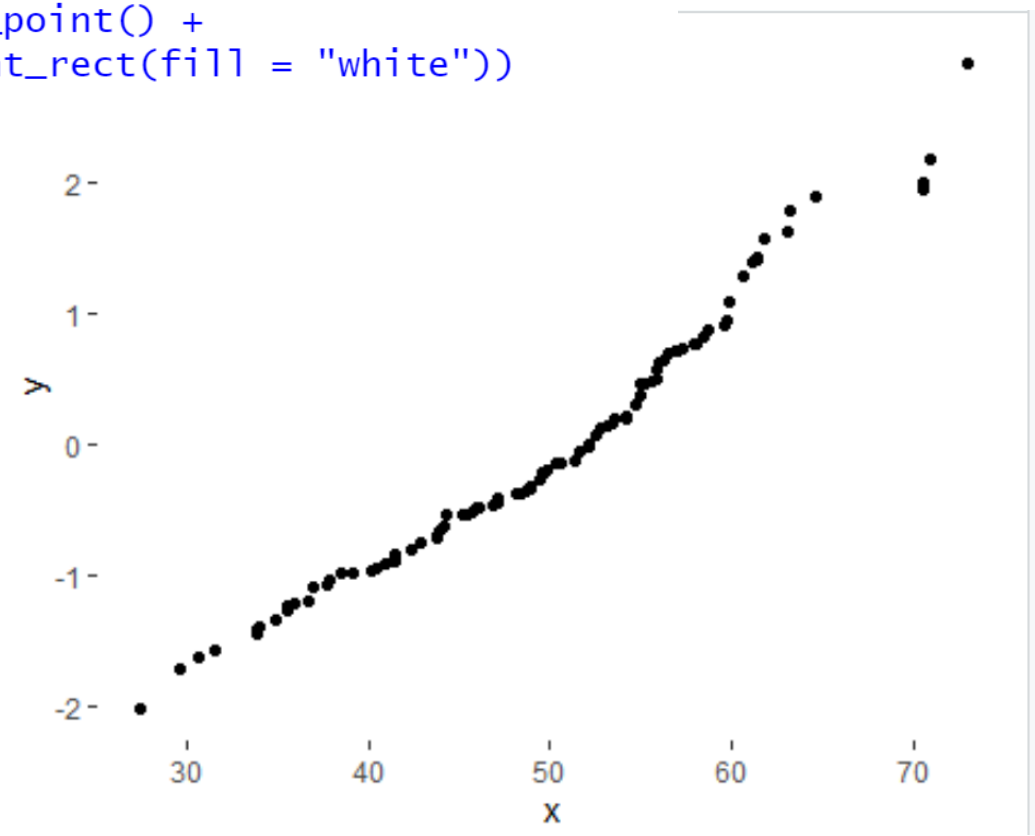
```
> # quantiles are just sorted data
> dfData$x <- sort(dfData$x)
> # what's the 50th percentile?
> dfData %>% slice(as.integer(round(nrow(dfData)*.5,0)))
         x
1 49.67403
> # remember, it's random
>
> mX <- mean(dfData$x)
> sdX <- sd(dfData$x)
> # create a theoretical normal distribution of quantiles
> dfData$y <- sort(qnorm(seq(0, 1, length.out = 100), mean = mX, sd = sdX))  #k and this is a
 theoretical norm distribution
> # remove infs (like nas)
> dfData <- filter(dfData, !is.infinite(y))
> # and compare the two
> ggplot(dfData, aes(x, y)) + geom_point() +
+     theme(panel.background = element_rect(fill = "white'
```

Now we'll take dfData$x *(calling this the "actual" random distribution we just analyzed)*, sort it, and then compare the values with a theoretical normal distribution with the same mean and sd. This "lines up" the values in order.



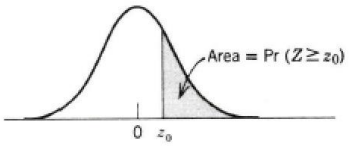
If these values all "line up" then the distributions are very similar

```
> # but here's the thing. The distribution we compare this with
> # doesn't need to have the same mean, sd OR range
> # this comparison still works
>
> dfData$y <- sort(rnorm(98, 0, 1))
> ggplot(dfData, aes(x, y)) + geom_point() +
+   theme(panel.background = element_rect(fill = "white"))
```



BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Looking back: Remember the z tables?



Standard Normal, Cumulative Probability in Right-Hand Tail
(For Negative Values of z, Areas are Found by Symmetry)

	NEXT DECIMAL PLACE OF z_0									
z_0	0	1	2	3	4	5	6	7	8	9
0.0	.500	.496	.492	.488	.484	.480	.476	.472	.468	.464
0.1	.460	.456	.452	.448	.444	.440	.436	.433	.429	.425
0.2	.421	.417	.413	.409	.405	.401	.397	.394	.390	.386
0.3	.382	.378	.374	.371	.367	.363	.359	.356	.352	.348
0.4	.345	.341	.337	.334	.330	.326	.323	.319	.316	.312
0.5	.309	.305	.302	.298	.295	.291	.288	.284	.281	.278
0.6	.274	.271	.268	.264	.261	.258	.255	.251	.248	.245
0.7	.242	.239	.236	.233	.230	.227	.224	.221	.218	.215
0.8	.212	.209	.206	.203	.200	.198	.195	.192	.189	.187
0.9	.184	.181	.179	.176	.174	.171	.169	.166	.164	.161
1.0	.159	.156	.154	.152	.149	.147	.145	.142	.140	.138
1.1	.136	.133	.131	.129	.127	.125	.123	.121	.119	.117
1.2	.115	.113	.111	.109	.107	.106	.104	.102	.100	.099
1.3	.097	.095	.093	.092	.090	.089	.087	.085	.084	.082
1.4	.081	.079	.078	.076	.075	.074	.072	.071	.069	.068
1.5	.067	.066	.064	.063	.062	.061	.059	.058	.057	.056
1.6	.055	.054	.053	.052	.051	.049	.048	.047	.046	.046
1.7	.045	.044	.043	.042	.041	.040	.039	.038	.038	.037
1.8	.036	.035	.034	.034	.033	.032	.031	.031	.030	.029
1.9	.029	.028	.027	.027	.026	.026	.025	.024	.024	.023
2.0	.023	.022	.022	.021	.021	.020	.020	.019	.019	.018
2.1	.018	.017	.017	.017	.016	.016	.015	.015	.015	.014
2.2	.014	.014	.013	.013	.013	.012	.012	.012	.011	.011
2.3	.011	.010	.010	.010	.010	.009	.009	.009	.009	.008
2.4	.008	.008	.008	.008	.007	.007	.007	.007	.007	.006
2.5	.006	.006	.006	.006	.006	.005	.005	.005	.005	.005
2.6	.005	.005	.004	.004	.004	.004	.004	.004	.004	.004
2.7	.003	.003	.003	.003	.003	.003	.003	.003	.003	.003
2.8	.003	.002	.002	.002	.002	.002	.002	.002	.002	.002
2.9	.002	.002	.002	.002	.002	.002	.002	.001	.001	.001

The inside cover of your statistics book usually has a standard normal distribution table where each z value is the number of standard deviations away from the mean.

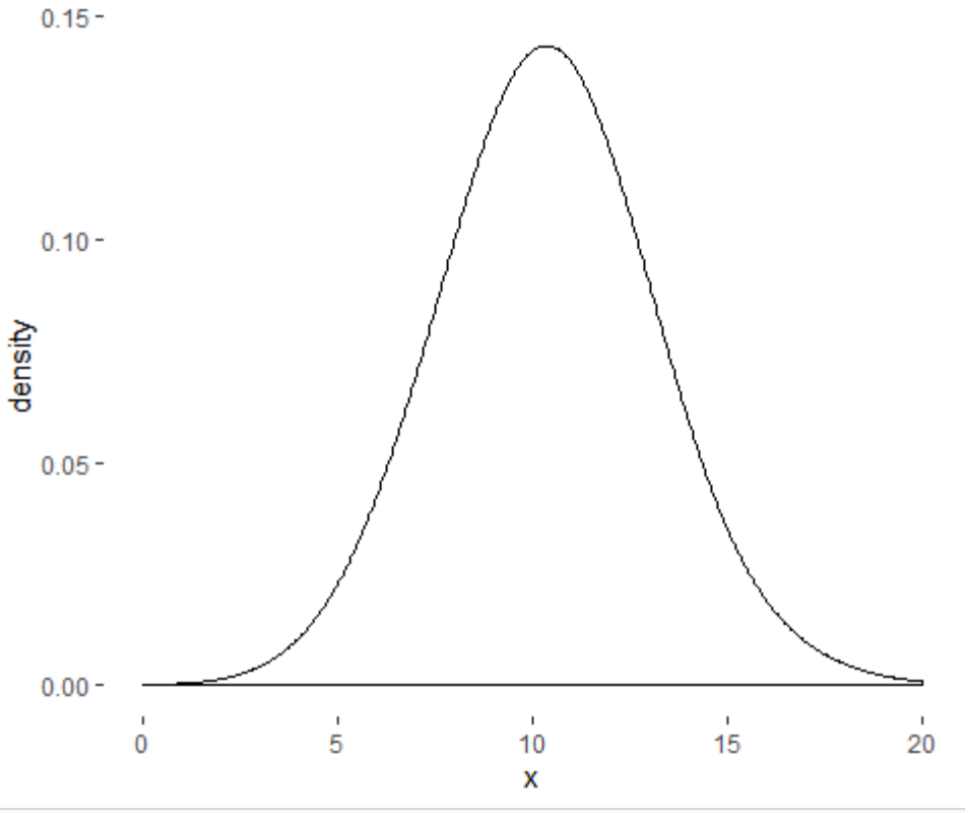
For general normal distributions, we converted the values to z values by standardizing: $Z = \frac{X - \mu}{\sigma}$

To get the ***probability a random values exceeding a standardized z value*** *(standard deviation) you can look it up in the table, or just type pnorm.*

```
> round(1 - pnorm(1.5, mean = 0, sd = 1),3)
[1] 0.067
```

```
> set.seed(35)
>
> # starting with your statistics texbook
> round(1 - pnorm(1.5, mean = 0, sd = 1),3)
[1] 0.067
>
>
> # let's set the sd @ 1.96
> # what does this mean?
> round(1 - pnorm(1.96, mean = 0, sd = 1),3)
[1] 0.025
>
> # lets create a distribution
>
> x <- data.frame(x = rnorm(100, mean = 10, sd = 2))
> ggplot(x, aes(x = x)) + geom_density(bw = 2) +
+   theme(panel.background = element_rect(fill = "white")) +
+   scale_x_continuous(limits = c(0, 20))
>
> # now let's get the mean and variance the hard way
> # whats the probability of x being 1 or less
>
> mean1 <- round(sum(x$x)/length(x$x),0)
> mean1
[1] 10
> sd1 <- round(sqrt(sum((x$x-mean1)^2)/(nrow(x)-1)),0)
> sd1
[1] 2
>
>
> # let's say x = 15
> # compute standardized value for z
> z1 <- abs(15-mean1)/sd1
> z1
[1] 2.5
> # look up in table (close to .006 probability)
>
> # compare to pnorm - neg side of mean now
> round(1 - pnorm(15, mean = mean1, sd = sd1),3)
[1] 0.006
```

$$Z = \frac{X - \mu}{\sigma}$$

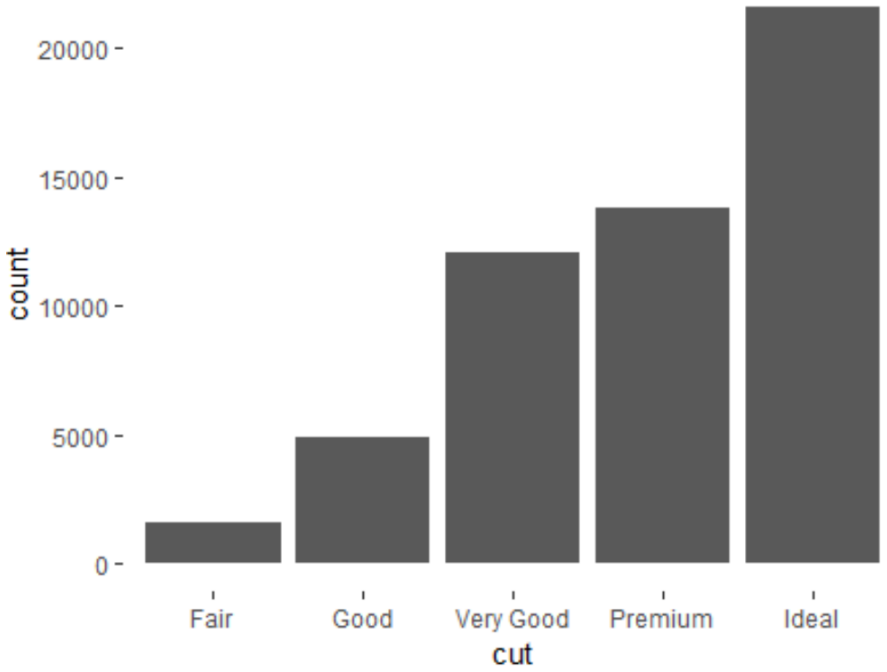


Histograms

That's all great, but we usually don't know the distribution, and assuming it's normal will usually be a mistake. So we have to figure that out.

We often start exploring the distribution using a histogram, but there are some caveats:

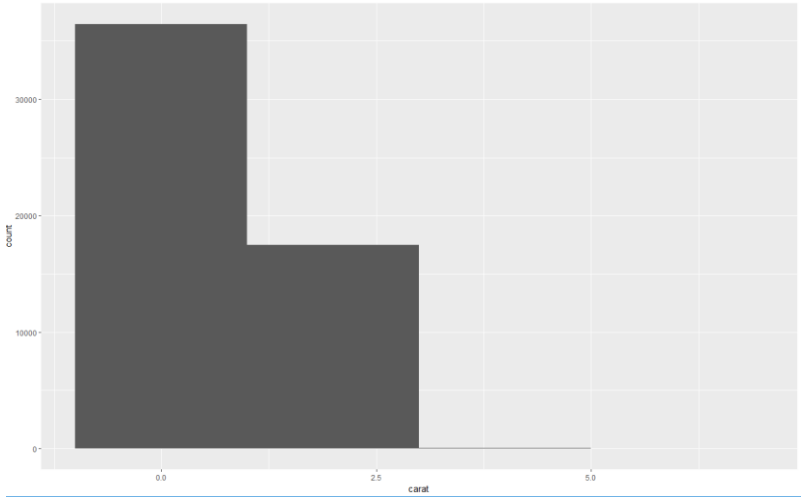
- The visual impression of the data conveyed by a histogram can depend upon the arbitrary origin of the **bin** system
- Binning *factors* is not usually problematic.
- But with continuous variables, the bin system must dissect the range of the variable into class intervals. So, if we use bins that are narrow enough to capture detail where data are plentiful — usually near the center of the distribution — then they may be too narrow to avoid 'noise' where data are sparse — usually in the tails of the distribution. We'll look at all of these issues in the following section.




```
> dfDiamonds %>% dplyr::count(cut)
# A tibble: 5 x 2
  cut           n
  <ord>     <int>
1 Fair       1610
2 Good       4906
3 Very Good 12082
4 Premium   13791
5 Ideal     21551
```

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Lets start with some simple explorations



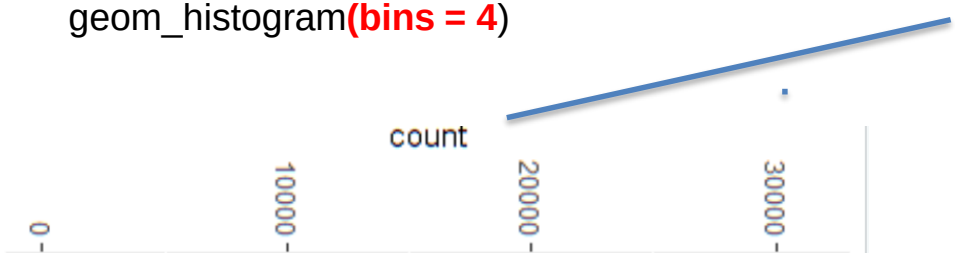

```
ggplot( data = diamonds, aes(x = carat)) +
geom_histogram(binwidth = 2)
diamonds %>% dplyr::count(cut_width(carat, 2))
```

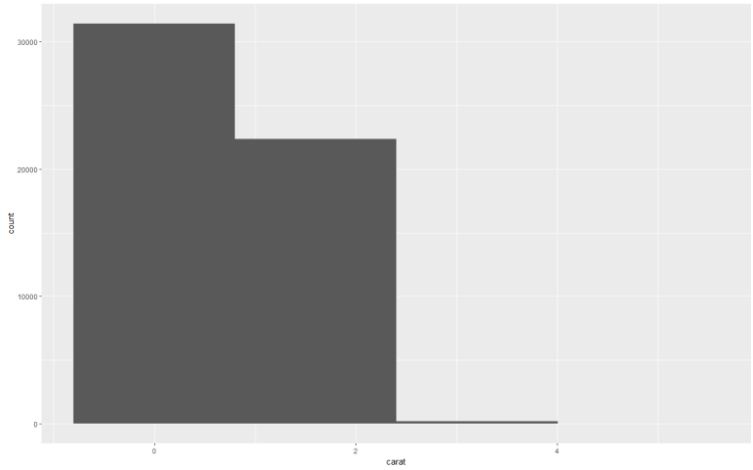
```
# A tibble: 4 x 2
  `cut_width(carat, 2)`     n
                 <fctr> <int>
1                [-1,1] 36438
2                 (1,3] 17470
3                 (3,5]    31
4                 (5,7]     1
```

you can also set boundaries (boundary = 0) or set breaks.

We can also use number of bins instead of binwidth. We can view the distribution count:

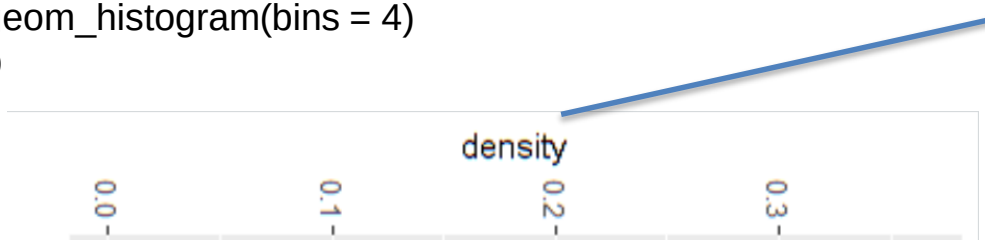
```
p <- ggplot( data = diamonds, aes(x = carat)) +
        geom_histogram(bins = 4)
p
```

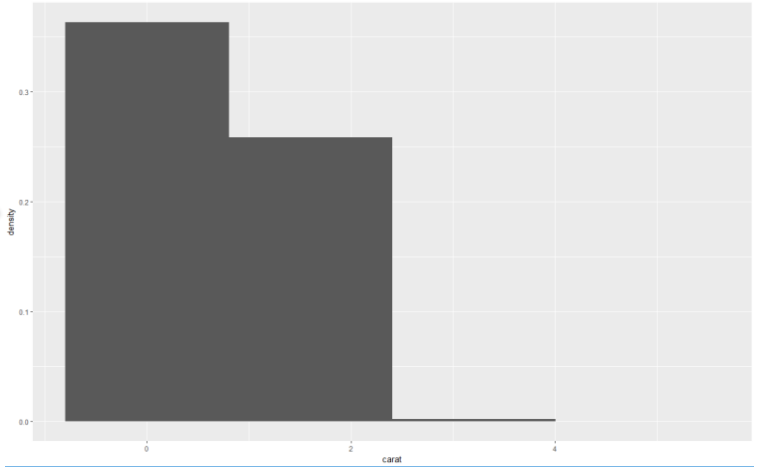




Or the distribution density

```
p <- ggplot( data = diamonds, aes(x = carat, y= ..density..)) +
geom_histogram(bins = 4)
p
```

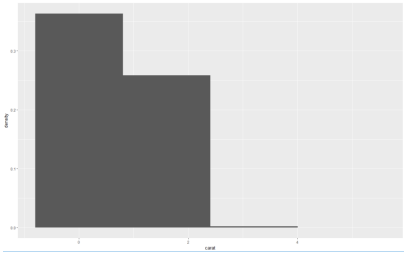




BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Getting the data out of ggplot stat functions.

```
# we can get the data from ggplot this way
pg <- ggplot_build(p)
# this creates an R list, which is a little different data structure:
(http://www.r-tutor.com/r-introduction/list )
# basically a way to store a bunch of different objects
pgData <- pg$data[[1]]
pgData %>%  dplyr::select(x, density)
```



In physics, Density is defined as the volumetric mass of a substance, or mass per unit volume.

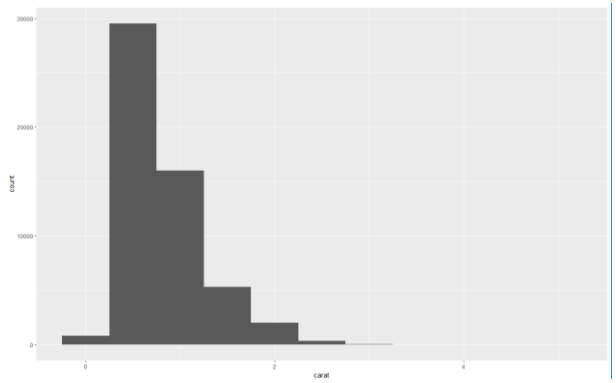
Here, it's the number of observations within a range *(or area)*. The density function returns values to calculate probability using area.

Note that it's easy when the data is binned to find the probability of a range – you're just calculating the area of a rectangle *(h*w)*.

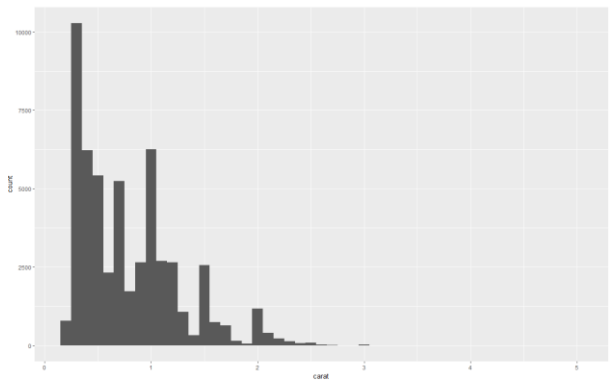
```
> pgData %>%  dplyr::select(x, density)
         x       density
0.000000 0.3632009694
1.603333 0.2583258367
3.206667 0.0021160032
4.810000 0.0000578143

> # what's the probabilty of a sample falling between 0 and 1.603333?
> pgData <- pgData %>% mutate(prob = (xmax- xmin)*density)
> totProb <- sum(pgData$prob, na.rm = T)
> totProb
[1] 1

> # what's the probabilty of a sample falling between 0 and 1.603333?
> dplyr::select(pgData, xmin, xmax, count, density, prob)
        xmin      xmax count       density         prob
1 -0.8016667 0.8016667 31411 0.3632009694 5.823322e-01
2  0.8016667 2.4050000 22341 0.2583258367 4.141824e-01
3  2.4050000 4.0083333   183 0.0021160032 3.392659e-03
4  4.0083333 5.6116667     5 0.0000578143 9.269559e-05
```

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Same data – different binwidths



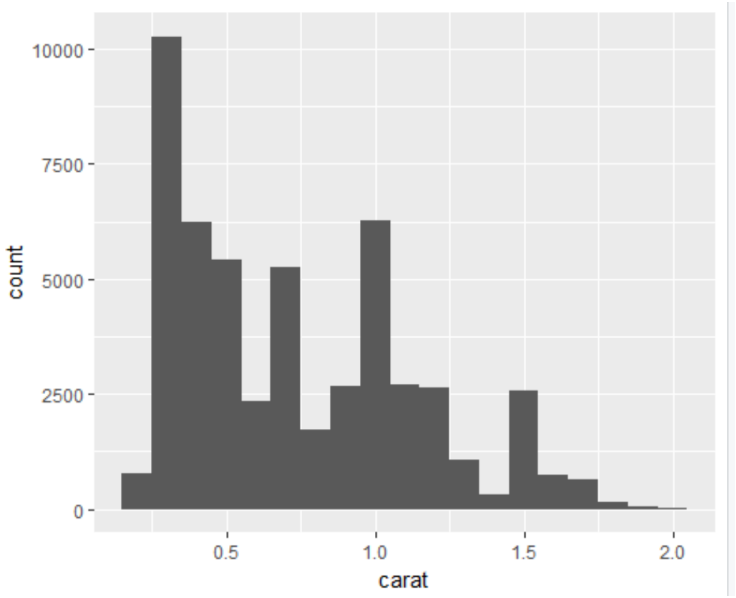

```
> ggplot( data = diamonds, aes(x = carat)) + geom_histogram(binwidth = 0.5)
> diamonds %>% dplyr::count(cut_width(carat, 0.5))
# A tibble: 11 x 2
   `cut_width(carat, 0.5)`     n
                    <fctr> <int>
 1            [-0.25,0.25]   785
 2             (0.25,0.75] 29498
 3             (0.75,1.25] 15977
 4             (1.25,1.75]  5313
 5             (1.75,2.25]  2002
 6             (2.25,2.75]   322
 7             (2.75,3.25]    32
 8             (3.25,3.75]     5
 9             (3.75,4.25]     4
10             (4.25,4.75]     1
11             (4.75,5.25]     1
```




```
> diamonds %>% dplyr::count(cut_width(carat, 0.1))
# A tibble: 38 x 2
   `cut_width(carat, 0.1)`     n
                    <fctr> <int>
 1            [0.15,0.25]   785
 2             (0.25,0.35] 10273
 3             (0.35,0.45]  6231
 4             (0.45,0.55]  5417
 5             (0.55,0.65]  2328
 6             (0.65,0.75]  5249
 7             (0.75,0.85]  1725
 8             (0.85,0.95]  2656
 9             (0.95,1.05]  6258
10             (1.05,1.15]  2687
# ... with 28 more rows
```

ggplot will show ALL the data, so you'll often end up with a plot concentrated to one end, which often means you have some outliers. One technique for analysis is to focus on specific ranges:

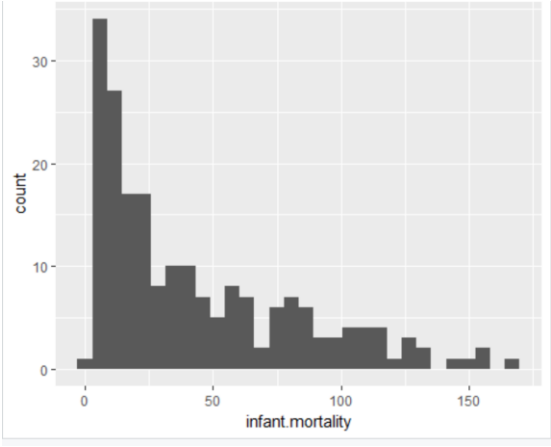
```
# or you can focus in a section of the data
focus <- diamonds %>% filter( carat < 2)
ggplot( data = focus, aes(x = carat)) + geom_histogram(binwidth
= 0.1)
```



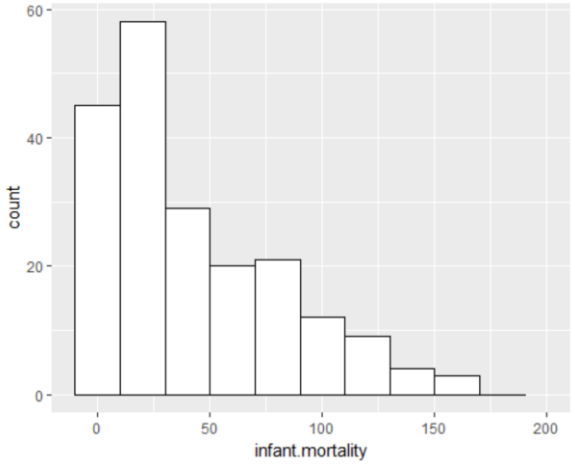
And you can control histogram aesthetics:

```
UN<- UN %>% filter(!is.na(infant.mortality))
ggplot( data = UN, mapping = aes( x = infant.mortality)) +
geom_histogram()
```

NOTE: infant.mortality has been replaced with infantMortality in the UN dataset. Use find/replace on your code



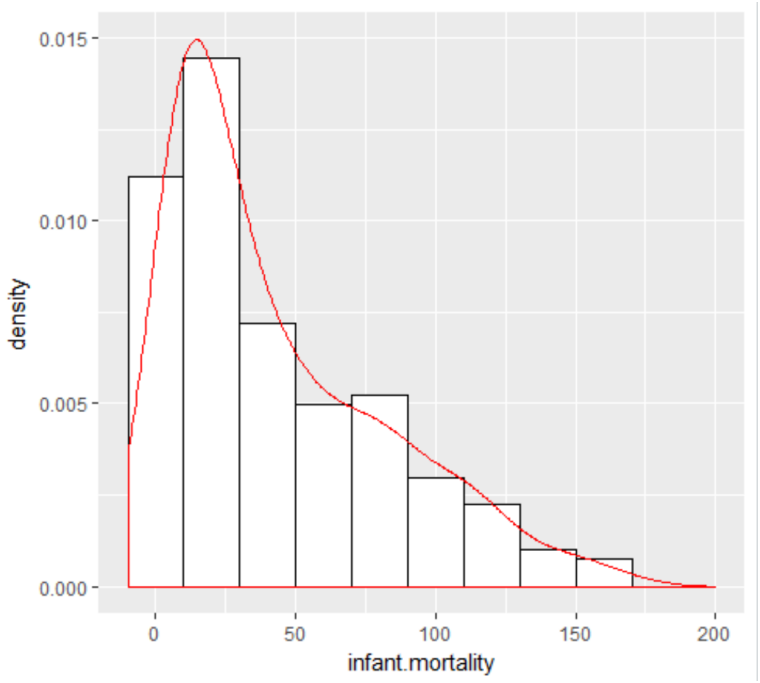

```
ggplot( data = UN, mapping = aes( x = infant.mortality)) +
    geom_histogram(binwidth = 20, color = 'black', fill = 'white')+
    xlim(c(-10, 200))
```



```
# you can use formulas in the bins

ggplot( data = UN, mapping = aes( x = infant.mortality)) +
  geom_histogram(bins = round(max(UN$infant.mortality)/20,0))
```

Back to density, this time with a continuous density curve shown

```
p <- ggplot(UN, aes( x = infant.mortality, y= ..density.. ))
+
      geom_histogram(binwidth = 20, color = 'black', fill =
'white')+
      xlim(c(-10, 200))
p

# plot the density (this is now recalcuated without bins -
pure continuous function)
p <- p + geom_density(color = 'red')
p
```



And again, showing the probability calculations:

```
> pgData <- pgData %>% mutate(prob = (xmax- xmin)*density)
> totProb <- sum(pgData$prob, na.rm = T)
> totProb
[1] 1
> dplyr::select(pgData, xmin, xmax, count, density, prob)
   xmin xmax count      density       prob
1   -10   10    45 0.0111940299 0.22388060
2    10   30    58 0.0144278607 0.28855721
3    30   50    29 0.0072139303 0.14427861
4    50   70    20 0.0049751244 0.09950249
5    70   90    21 0.0052238806 0.10447761
6    90  110    12 0.0029850746 0.05970149
7   110  130     9 0.0022388060 0.04477612
8   130  150     4 0.0009950249 0.01990050
9   150  170     3 0.0007462687 0.01492537
10  170  190     0 0.0000000000 0.00000000
11  190   NA     0 0.0000000000         NA
>
> # so what's the probablity of a value between -10 and 10?
> # it's 22%
```

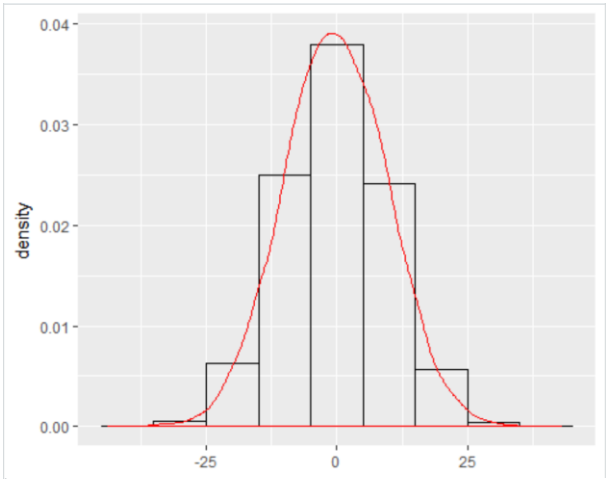
Now we want to use the pdf find the probability that a random variable greater than specific value.

First, let's create a reusable function so we don't have to type it over and over *(see chpt 19 of book for functions)*:

```
> getProb <- function(p, dVal)
+   {
+     pg <- ggplot_build(p)
+     pgData <- pg$data[[1]]
+     pgData <- pgData %>% mutate(prob = (xmax- xmin)*density) %>% filter(x >=  dVal)
+     totProb <- sum(pgData$prob, na.rm = T)
+     return(totProb)
+ }
```

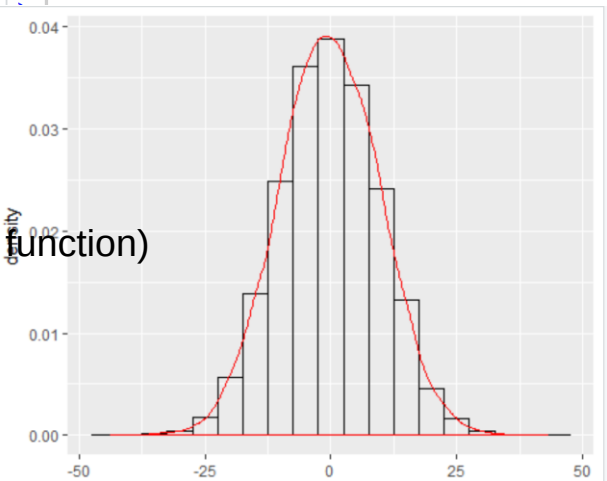
Functions	
getProb	function (p, dVal)

```
ndist <- data.frame(x = rnorm(10000,0,10)) # note how r provides all these statistical functions that make it easy

p <- ggplot(ndist, aes(x = x, y= ..density..)) +
  geom_histogram(binwidth = 10, color = 'black', fill = NA)
p
# plot the density (this is now recalcuated without bins - pure continuous function)
p <- p + geom_density(color = 'red')
p
# now get the estimated probablity over the mean
estProb <- getProb(p, 0)
estProb
```




```
> estProb
[1] 0.6813
> #hmmm not too good!
> #k what if we decrease bin size?
```

```
p <- ggplot(ndist, aes(x = x, y= ..density..)) +
  geom_histogram(binwidth = 5, color = 'black', fill = NA)
p
# plot the density (this is now recalcuated without bins - pure continuous function)
p <- p + geom_density(color = 'red')
p

estProb <- getProb(p, 0)
estProb
```

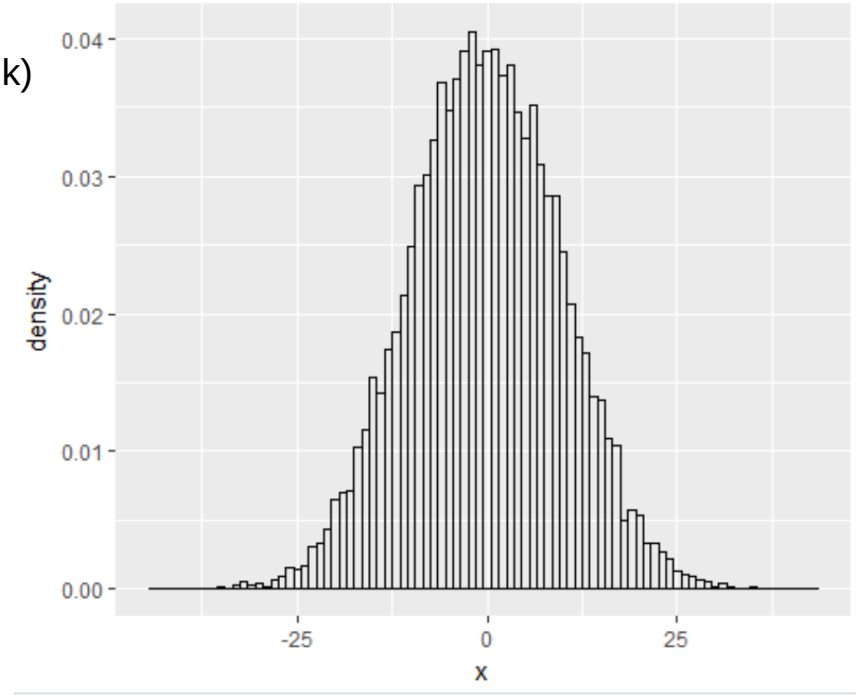



```
> estProb <- getProb(p, 0)
> estProb
[1] 0.5862
```

Now we'll create a loop *(see chpt 21 of book for loops)**, and use that to sequentially decrease the binwidth 10:1***

```
# a little better
#Let's decreate binwidth by 1 using a loop (chpt 21 of book)
for (i in 10:1)
{
  p <- ggplot(ndist, aes(x = x, y= ..density..)) +
geom_histogram(binwidth = i, color = 'black', fill = NA)
  estProb <- getProb(p, 0)
  print(estProb)
  }
p
```

```
+    }
[1] 0.6813
[1] 0.6624
[1] 0.6438
[1] 0.6253
[1] 0.6051
[1] 0.5862
[1] 0.5666
[1] 0.5457
[1] 0.5273
[1] 0.5076
```



Note how the estimated probability gets closer as binwidth gets smaller.

That's the principle underlying integration

So let's just make life easier and just use integration.

At least 3 ways to do that in R

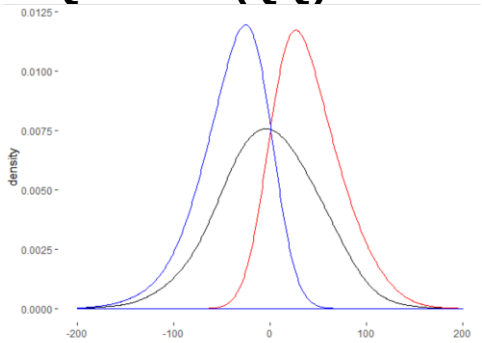
```
> integrate(approxfun(density(ndist$x)), lower = 0, upper = 38, subdivisions=2000)$value
[1] 0.4882888
>
> # breaking this down and looking at different ways to do it:
> df <- approxfun(density(ndist$x))
> DenDF <- data.frame(x = ndist$x, y = df(ndist$x))
> df2 <- dplyr::filter(DenDF, x >= mean(x))
> integrate.xy(df2$x, df2$y)
[1] 0.4984309
> # OR
> auc(df2$x, df2$y)
[1] 0.4985377
>
> # the inteegrate(approxfun) uses linear interpolation, the AUC uses a trapazoid method
```

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

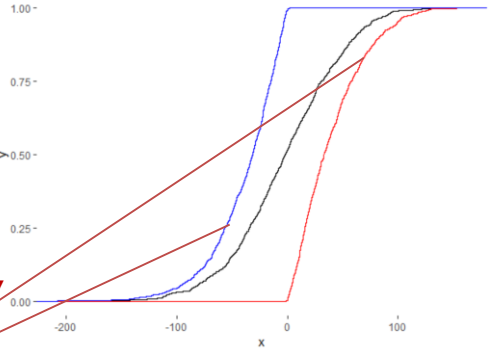
Introducing the Cumulative Distribution Function (CDF) and Quantile-Quantile (QQ) Plots

```
library(sn)
x <- rnorm(1000, mean = 0, sd = 50)
sr <- rsn(1000, xi=0, omega=50, alpha= 50)
sl <- rsn(1000, xi=0, omega=50, alpha=-50)
mP <- data.frame(x, sr, sl)

ggplot(mP, aes(x)) +
  geom_density(bw = 20) +
  geom_density(aes(x= sr), bw = 20, color = "red") +
  geom_density(aes(x= sl), bw = 20, color = "blue") +
  theme(panel.background = element_rect(fill = "white")) +
  scale_x_continuous(limits = c(-200, 200))
 ggplot(mP, aes(x  = x)) +
   stat_ecdf() +
   stat_ecdf(aes(x = sr), color = "red") +
   stat_ecdf(aes(x = sl), color = "blue") +
   theme(panel.background = element_rect(fill = "white"))

 ggplot(mP, aes(sample  = x)) +
   stat_qq() +
   stat_qq(aes(sample = sr), color = "red") +
   stat_qq(aes(sample = sl), color = "blue") +
   theme(panel.background = element_rect(fill = "white"))
```

density

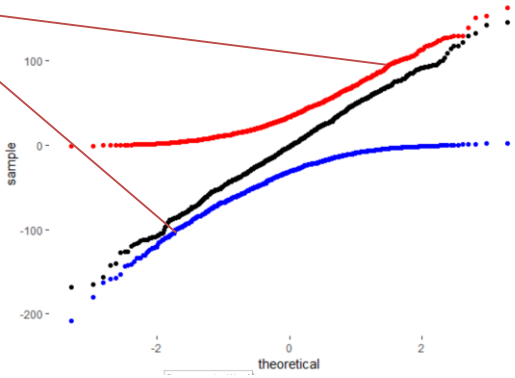


cumulative density



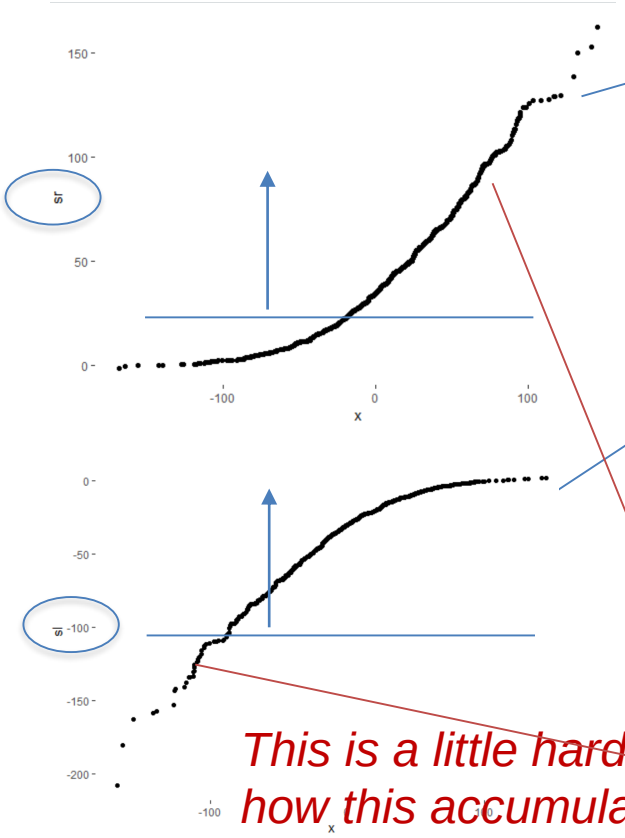
This is a little hard to see – but notice how this accumulates slower here

quantile / quantile (QQ)



This is different from the plot of quantiles we did before. This is a QQ (quantile – quantile), which compares the quantiles of 2 different distributions at n=n. Here, we're comparing our samples to a normal distribution. So the normal ⇔ comparison yields a straight line x=y. Notice how the quantiles of the skew left line up with normal for a while, but then quickly complete. Or, how the quantiles of the skew right quickly catch up to normal and then line up the rest of the way

```
qq <- data.frame(x = sort(mP$x), sr = sort(mP$sr), sl = sort(mP$sl))

ggplot(qq, aes(x, sr))+ geom_point() +
  theme(panel.background = element_rect(fill = "white"))

ggplot(qq, aes(x, sl))+ geom_point() +
  theme(panel.background = element_rect(fill = "white"))
```



Skew right

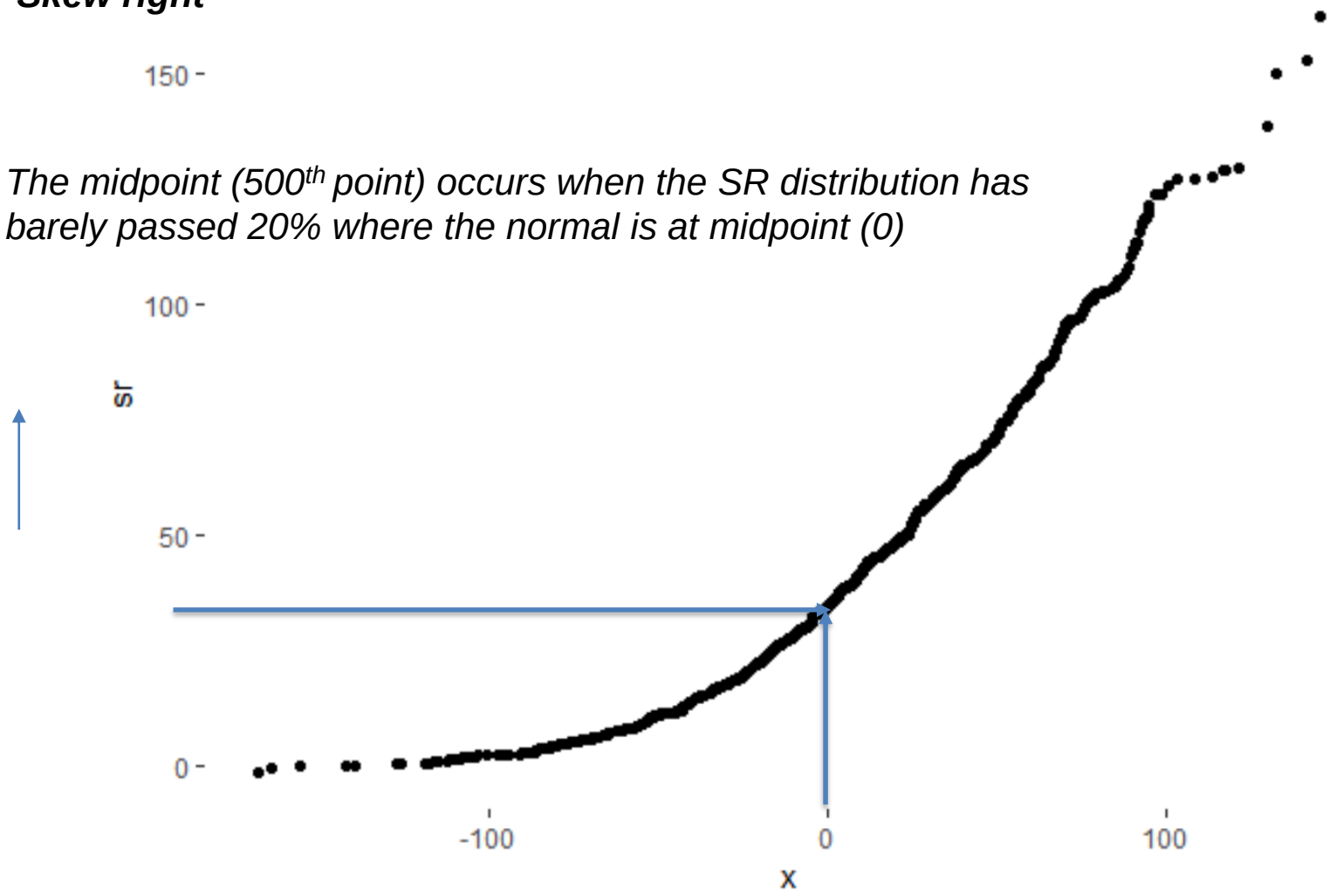
Skew left

	x	sr	sl
1	-168.45534	-1.54021858	-208.07372
2	-164.79091	-0.90163729	-180.80140
3	-156.34185	-0.17653554	-163.20310
4	-142.47699	-0.01483926	-159.10952
5	-139.92888	0.02222768	-157.56213
6	-127.47015	0.02865878	-153.41004
7	-126.73672	0.11494850	-143.47277
8	-126.01772	0.19757623	-142.34012
9	-119.43105	0.31654321	-141.15581
10	-117.75634	0.47908046	-138.26232
11	-116.29072	0.52529150	-134.45171
12	-115.12296	0.55694352	-134.17150
13	-112.91230	0.67152050	-133.99152
14	-112.75408	0.69198448	-130.46394
15	-112.37793	0.74360095	-130.28033
16	-112.32231	1.03287529	-127.50510
17	-111.82166	1.16658115	-125.95093

This is a little hard to see – but notice how this accumulates slower here

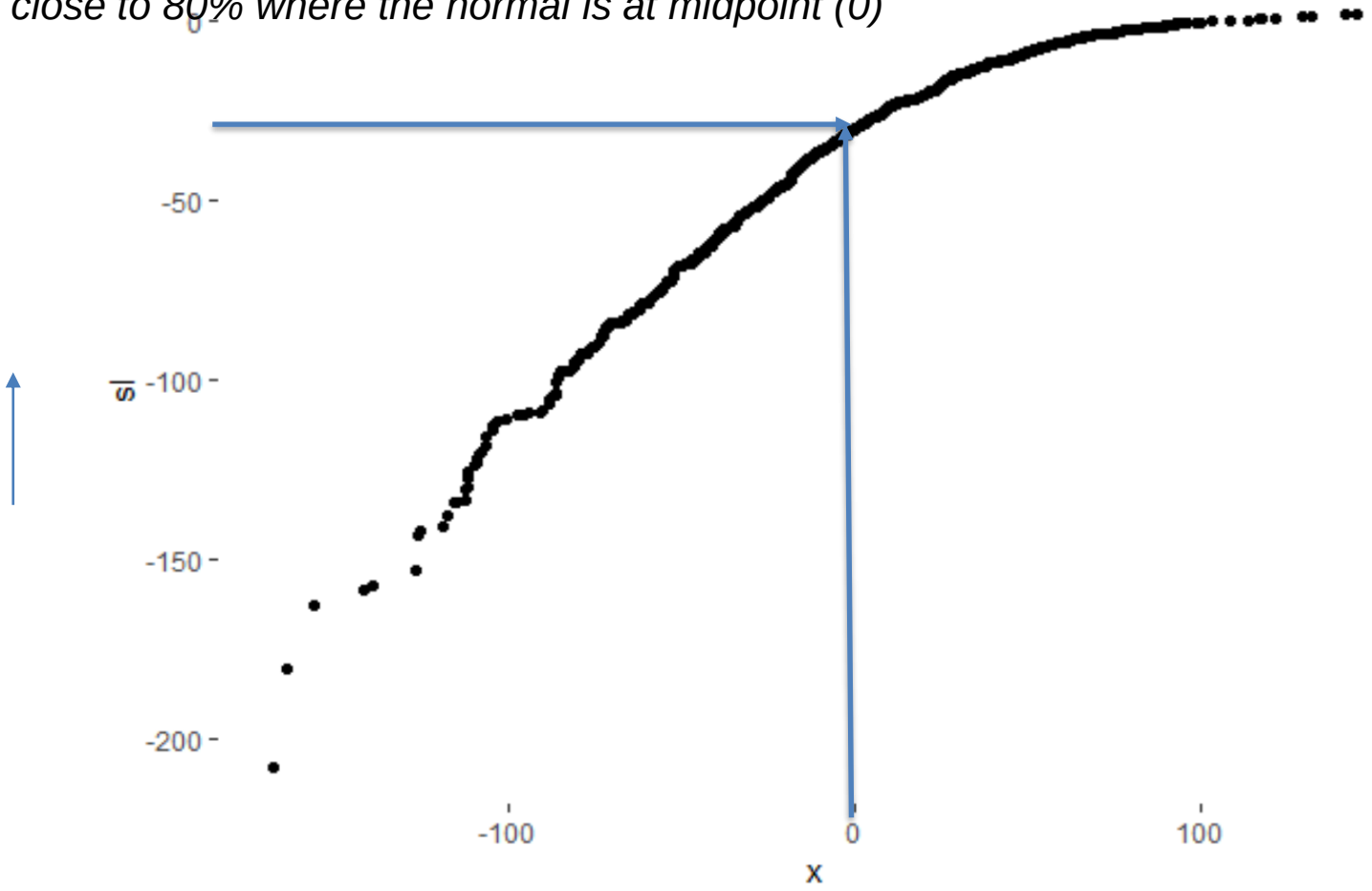
Skew right

The midpoint (500th point) occurs when the SR distribution has barely passed 20% where the normal is at midpoint (0)



Skew left

Here, the 500th point occurs when the SL distribution is getting close to 80% where the normal is at midpoint (0)

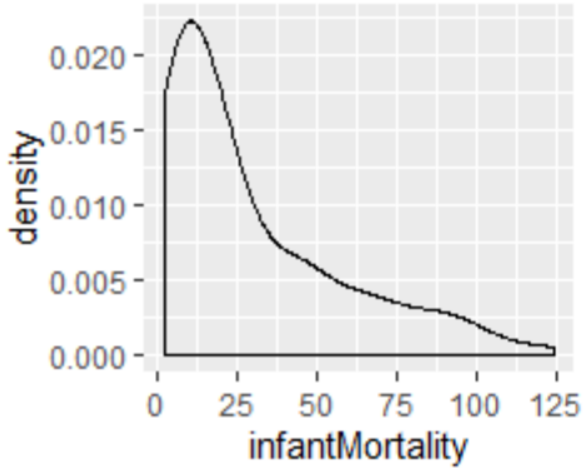


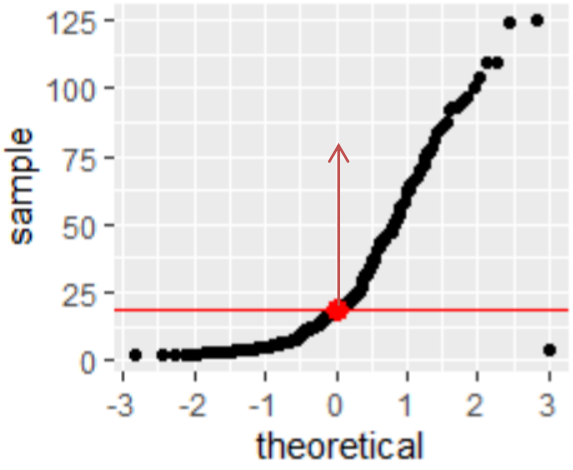
Applying CDF and QQ to the UN data

```
p1 <- ggplot(UN, aes(x = infant.mortality, y = ..density..)) + geom_density()
p2 <- ggplot( data = UN, mapping = aes( x = infant.mortality)) + stat_ecdf()
q1 <- ggplot( data = UN, mapping = aes( sample = infant.mortality)) + geom_qq()
multiplot(p1, p2, q1, cols=2)

pg <- ggplot_build(p2)
pgData1 <- pg$data[[1]]
pgData1 <- pgData1 %>% dplyr::select(x, y) %>% arrange(x)
pgData1Mid <- pgData1[round((nrow(pgData1)/2),0),]
pgData1Mid

pg <- ggplot_build(q1)
qData1 <- pg$data[[1]]
qData1 <- qData1 %>% dplyr::select(x, y) %>% arrange(x)
qData1Mid <- filter(qData1, x == 0) # don't need to find the mid point because the qq is centered @ 0
qData1Mid

pgData1Mid2 <- filter(pgData1, x == qData1Mid$y)
```

Note: the data from UN also changed, so these functions will yield different results

```
> # ---------------------------- CDF and QQ Applied to UN data ----------------------------------#
>
>
> p1 <- ggplot(UN, aes(x = infantMortality, y = ..density..)) + geom_density()
> p2 <- ggplot( data = UN, mapping = aes( x = infantMortality)) + stat_ecdf()
> q1 <- ggplot( data = UN, mapping = aes( sample = infantMortality)) + geom_qq()
> q1 <- q1 + geom_point(aes(3,4))
> multiplot(p1, p2, q1, cols=2)
>
> # let's look at the cumulative density
> pg <- ggplot_build(p2)
> pgData1 <- pg$data[[1]]
> MidP2 <- pgData1 %>% dplyr::select(x, y) %>% arrange(x)  %>% slice(as.integer(round(nrow(qData1)*.5,
0)))
> p2 <- p2 + geom_point(aes(MidP2$x, MidP2$y), color = "red", size = 3)
> p2 <- p2 + geom_vline(xintercept = MidP2$x, color  = "red")
>
> pg <- ggplot_build(q1)
> qData1 <- pg$data[[1]]
> MidQ1 <- qData1 %>% dplyr::select(x, y) %>% arrange(x) %>% slice(as.integer(round(nrow(qData1)*.5,
0)))
> q1 <- q1 + geom_point(aes(MidQ1$x,MidQ1$y), color = "red", size = 3)
> q1 <- q1 + geom_hline(yintercept = MidQ1$y, color  = "red")
>
>
> multiplot(p1, p2, q1, cols=2)
>
```



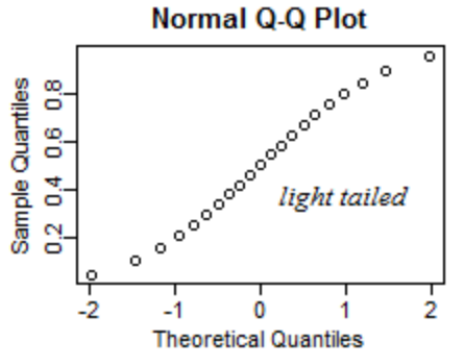


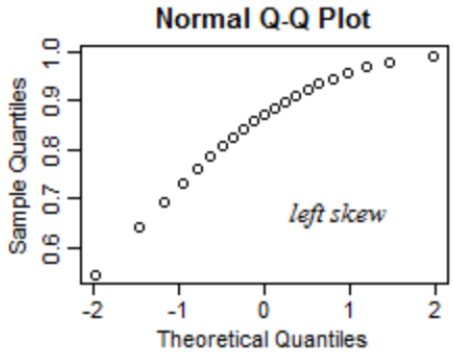
Notice how the sample QQ reaches midpoint early

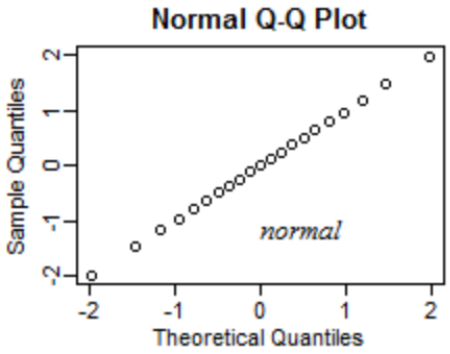
And the cdf also reaches midpoint early

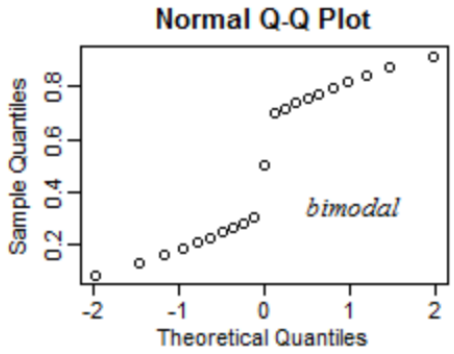
BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

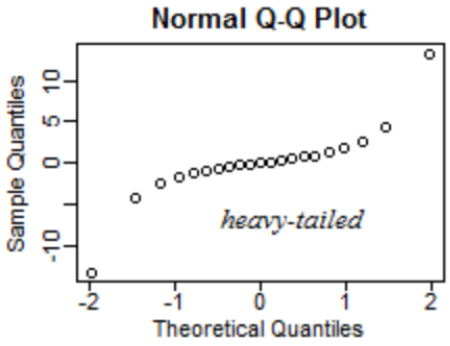
QQ Plots

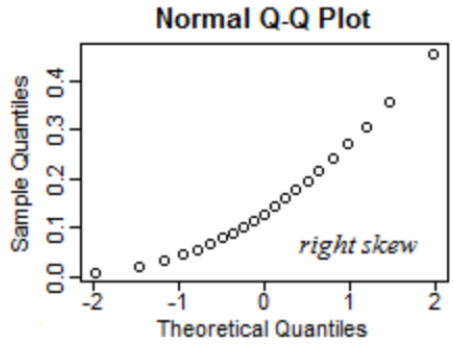








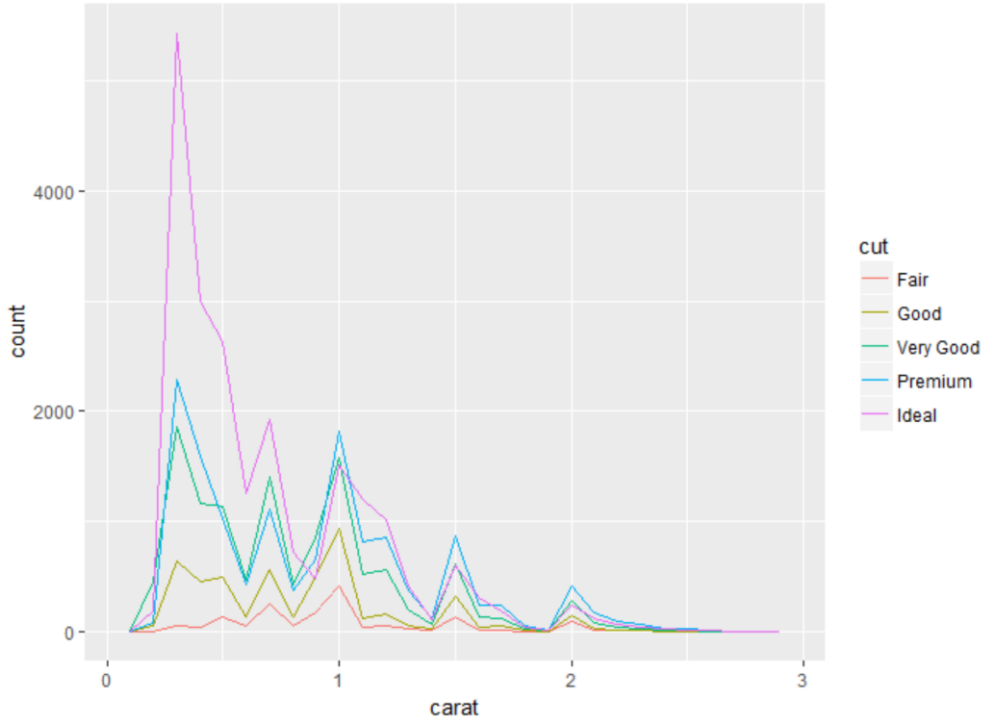




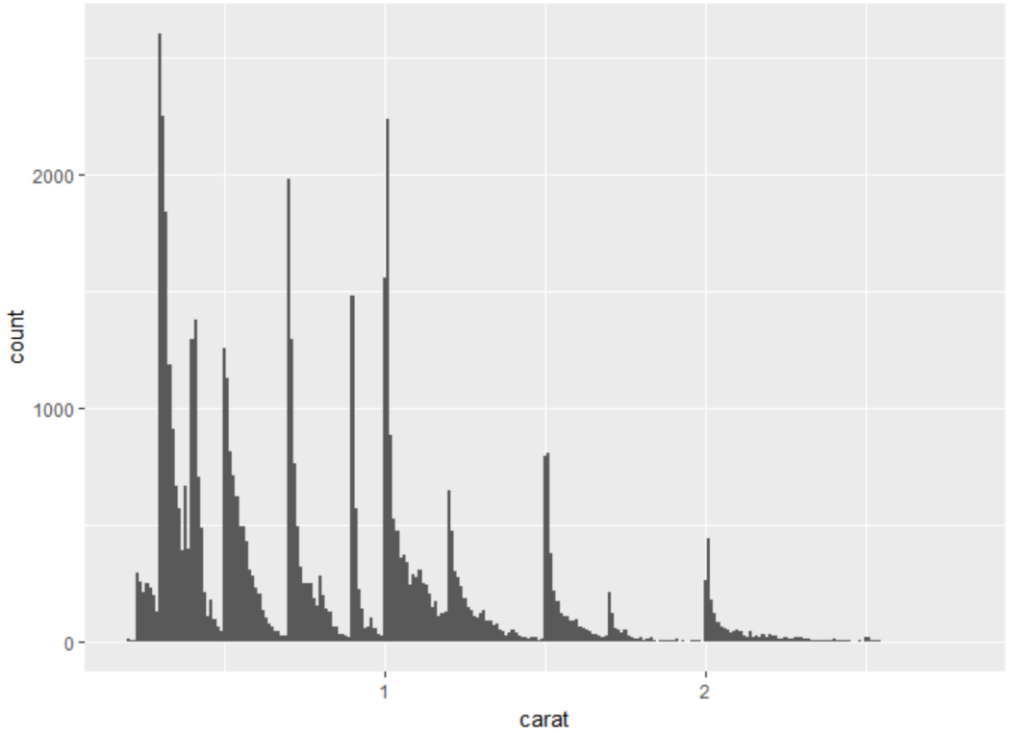
Additional ggplot functions

To overlay multiple histograms in the same plot, use geom_freqpoly() instead of geom_histogram(). geom_freqpoly() performs the same calculation as geom_histogram(), but instead of displaying the counts with bars, uses lines instead. It's much easier to understand overlapping lines than bars:

```
ggplot( data = smaller, mapping = aes( x = carat,
color = cut)) + geom_freqpoly( binwidth = 0.1)
```



```
ggplot( data = smaller, mapping = aes( x = carat)) + geom_histogram( binwidth = 0.01)
```



In general, clusters of similar values suggest that subgroups exist in your data. To understand the subgroups, ask:

- How are the observations within each cluster similar to each other?
- How are the observations in separate clusters different from each other?
- How can you explain or describe the clusters?

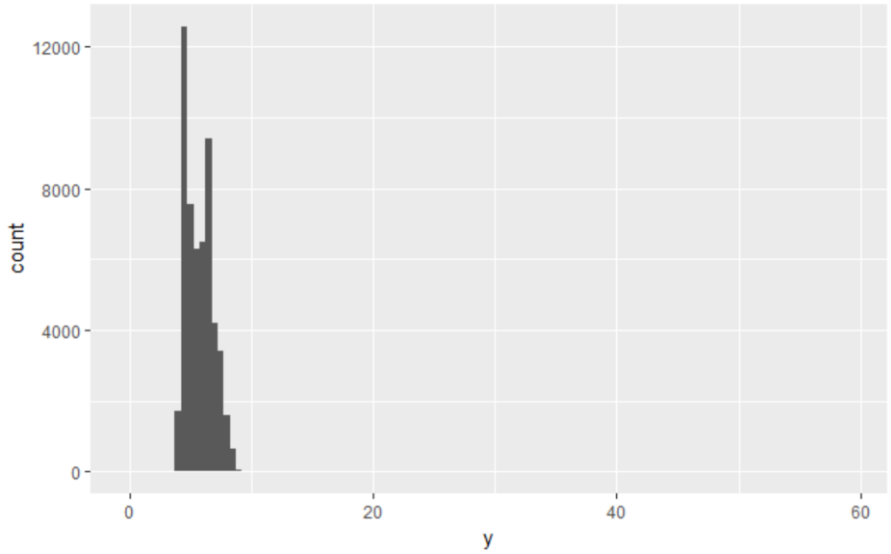
Dealing with Outliers

Outliers are observations that are unusual; data points that don't seem to fit the pattern. Sometimes outliers are data entry errors; other times outliers suggest important abberations.

When you have a lot of data, outliers are sometimes difficult to see in a histogram. For example, take the distribution of the y variable from the diamonds dataset.

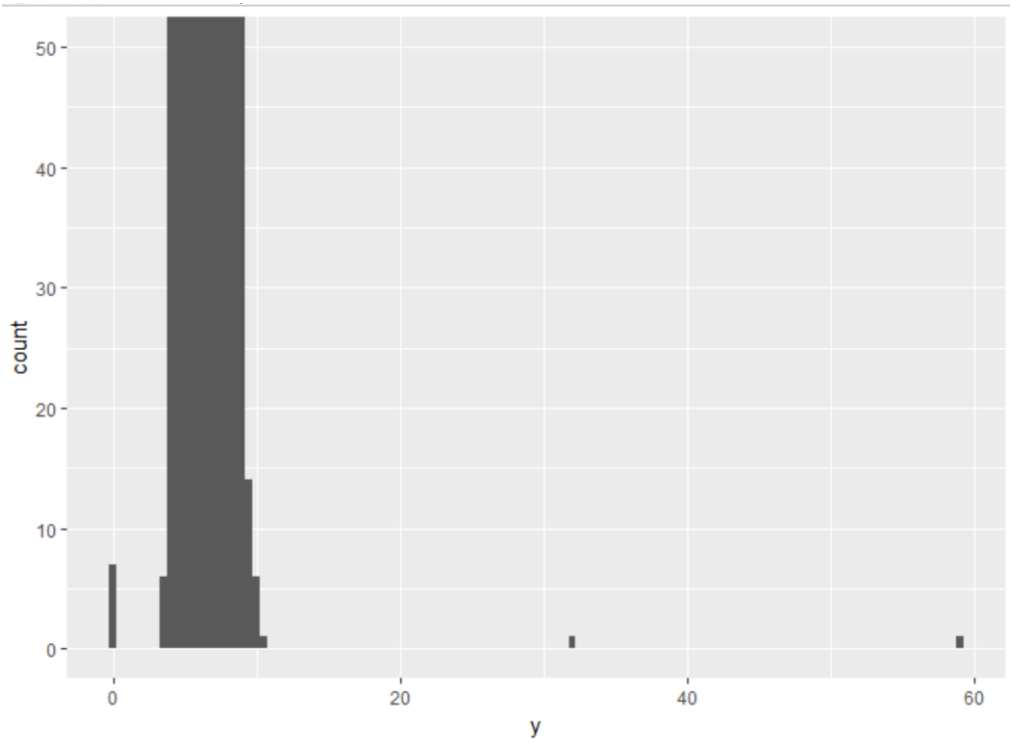
Evidence of outliers is the often wide limits on the y-axis:

```
ggplot( diamonds) + geom_histogram( mapping = aes( x = y), binwidth = 0.5)
```



To make it easy to see the unusual values, we need to zoom in to small values of the y-axis with coord_cartesian():

```
ggplot( diamonds) + geom_histogram( mapping = aes( x = y), binwidth = 0.5) +
coord_cartesian( ylim = c( 0, 50))
```



Missing Values

If you've encountered unusual values in your dataset, and simply want to move on to the rest of your analysis, you have two options: Drop the entire row with the strange values:

```
Complete Case: diamonds2 <- diamonds %>% filter( between( y, 3, 20))
```

Instead, I recommend replacing the unusual values with missing values. The easiest way to do this is to use mutate() to replace the variable with a modified copy. You can use the ifelse() function to replace unusual values with NA:

```
Available Case: diamonds2 <- diamonds %>% mutate( y = ifelse( y < 3 | y > 20, NA, y))
```

Four common ways of dealing with missing data:

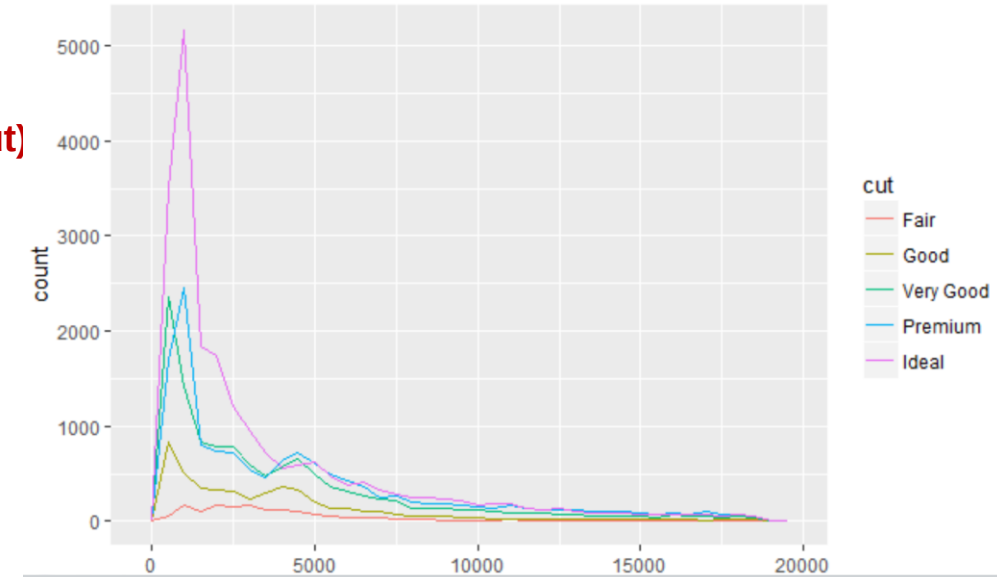
- Complete-Case Analysis. Omitting any case with missing value
- Available Case Analysis. Use available cases and estimate quantities of interest
- Imputation Replace – (e.g, imputing an observed variable mean, or default)
- Multiple imputationUsing different methods for imputing specific, small cases (e.g., Monte Carlo, or business rule)

Categorical and Continuous Variable

It's common to want to explore the distribution of a continuous variable broken down by a categorical variable. The default appearance of geom_freqpoly() is not that useful for that sort of comparison because the height is given by the count. That means if one of the groups is much smaller than the others, it's hard to see the differences in shape.

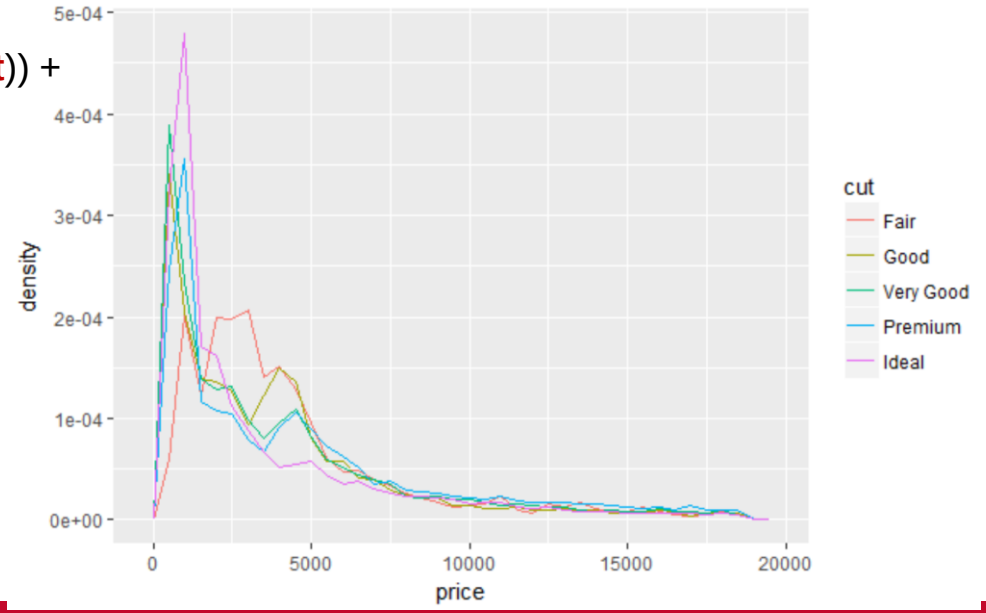
For example, let's explore how the price of a diamond varies with its quality:

```
ggplot(diamonds, mapping = aes(price, color = cut))
+ geom_freqpoly(binwidth = 500)
```



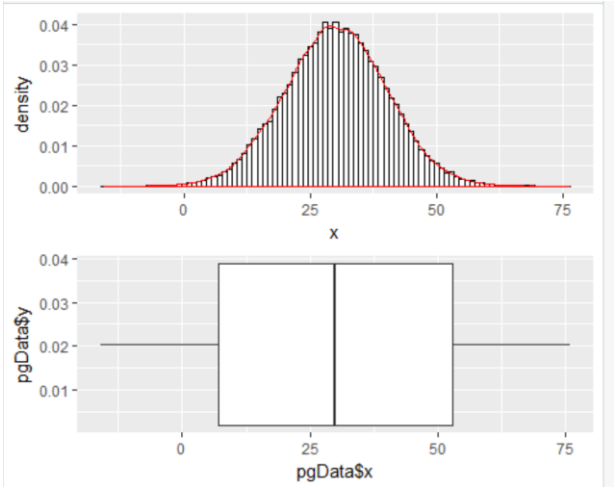

```
ggplot(diamonds, aes(price, ..density.., color = cut)) +
geom_freqpoly(binwidth = 500)
```

Note the distribution of Fair



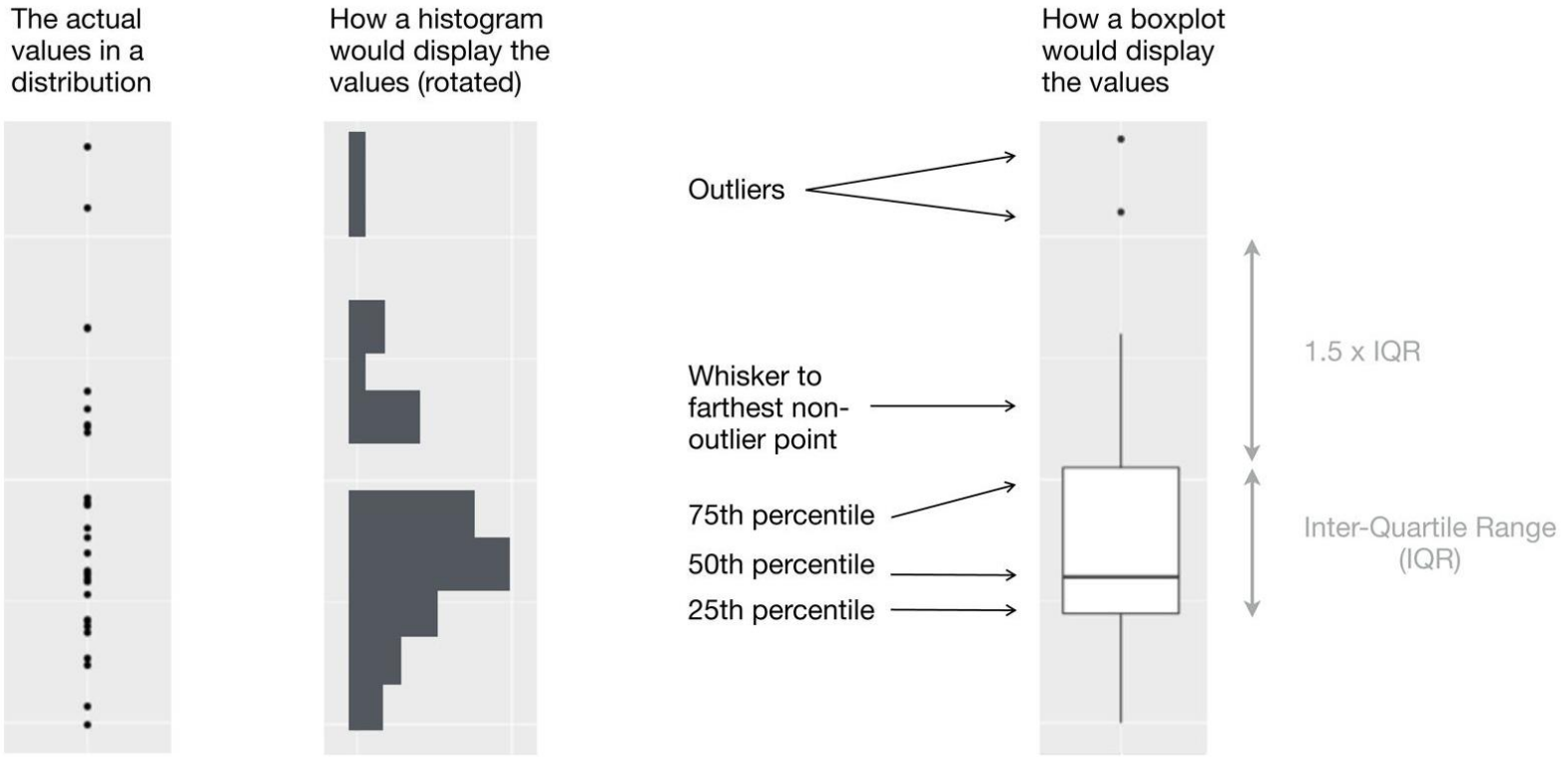
boxplots

```
p <- ggplot(ndist, aes(x = x, y = ..density..)) +
  geom_histogram(binwidth = 1, color = 'black', fill = 'white')
p <- p + geom_density(color = 'red')
p

# get the data from the density

pg <- ggplot_build(p)
pgData <- pg$data[[1]]
integrate(approxfun(pgData$x, pgData$y), 0, 60)
integrate.xy(pgData$x, pgData$y)
auc(pgData$x, pgData$y)

# introduing a boxplot

p2 <- ggplot(pgData, aes(y = pgData$x, x = pgData$y)) +
geom_boxplot() + coord_flip()
p2

library(Rmisc)
multiplot(p, p2, cols=1)
```



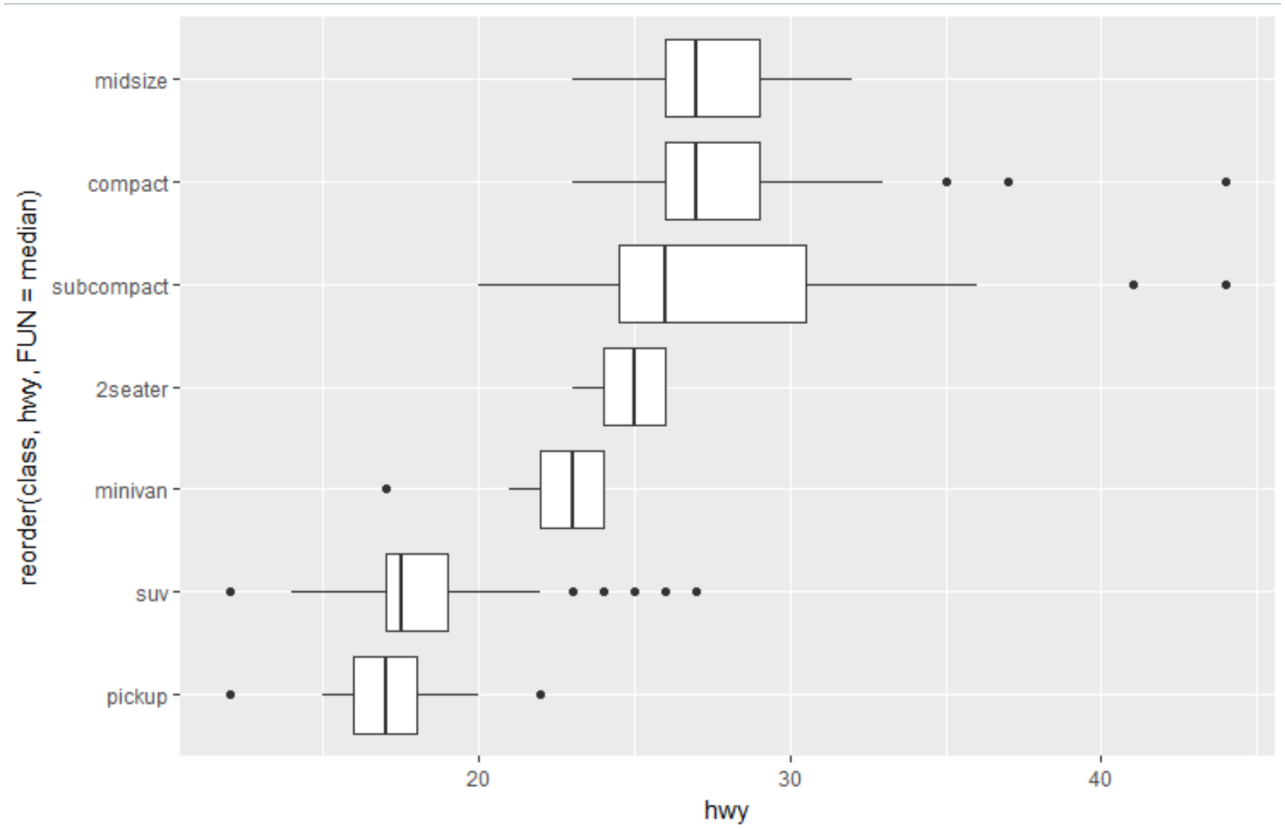
Boxplots are a good way to compare distributions and outliers across groups.

Scatter vs. Histogram vs. Boxplots



boxplots

```
ggplot(data = mpg) +
  geom_boxplot( mapping = aes( x = reorder(class, hwy, FUN = median), y = hwy) ) +
  coord_flip()
```



BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Review of Probability and Distributions

Hypothesis Testing

Probability

- Sample Spaces and Events
- Probability
- Conditional Probability
- Independent Events *(iid)*

Discrete Distributions

- Discrete Uniform Distribution
- Binomial Distribution
- Other Discrete Distributions

Continuous Distributions

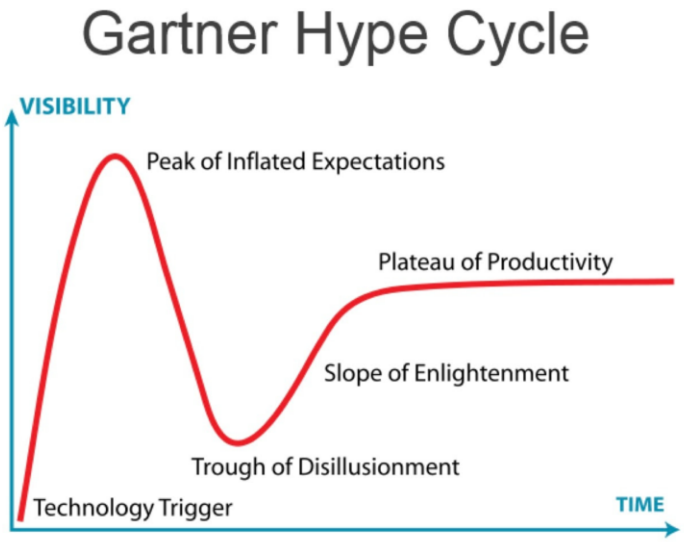
- Normal Distribution

Moments and Distribution Fitting

Bayes Theorem

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Common Misconceptions in the Accounting Industry



Misconception 1. ***"Big Data"*** *(i.e., scalable data platforms)* **will eliminate sampling**. The analytics process *(acquisition, EDA , modeling, inference)* requires sampling at every step *(remember, transactions flow at a minimum of 5k/sec).* We will always need sampling to reduce analysis processing time to efficient levels and reduce analysis complexity to comprehensible, yet comprehensive levels.*Statistics is not hard – just learn it and do your job.* That said – Big data is great for sampling…

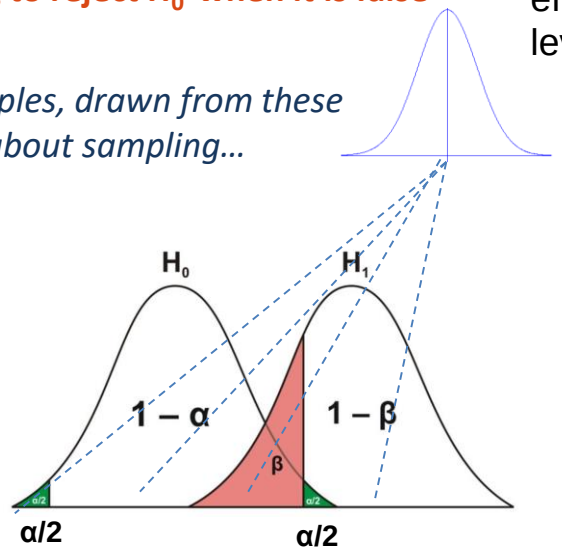
Misconception 2. ***Blockchain will eliminate substantive examination of supporting evidence***. Actually, it will increase complexity - blockchains will bring additional transaction channels, that will still be integrated with the ERP *(bc's will <u>not</u> replace ERP – you have no idea how many business rules are in the ERP)* and it will make substantive testing *(and sampling)* more difficult.

Quick Review NHST – Null Hypothesis Significance Testing

- **α = the conditional probability of incorrectly rejecting H_0 when it is true – Type 1 error.** $\mathbb{P}$*(reject H_0 | H_0 is true) the probability of observing data > α given that H_0 is true.* ***α*** is an experiment level metric - ***p-value*** is the sample level metric.
- **β = the conditional probability of failing to reject H_0 when it is false – Type II error** $\mathbb{P}$*(accept H_0 | H_1 is true)*

Keep in mind, we're concerned with samples, drawn from these populations. Frequentist statistics is all about sampling…

	H_0 is True	**H_0 is False**
p-value > α *(accept H_0)*	True Positive	**Type II Error**
p-value < α *(reject H_0)*	**Type I Error**	True Negative



Power = the compliment of β: the probability of correctly rejecting H_0 when H_1 true $\mathbb{P}$*(reject H_0 | H_1 is true). Go to* https://rpsychologist.com/d3/NHST/ and prototype scenarios to see the effects of sample size, significance level and power

Caveats!!!: *Rejecting H_0 with α = 0.05 this does not mean that we are 95% sure that the alternative hypothesis is true. A p-value does not tell us that our findings are relevant, clinically significant, or of any scientific value whatsoever. Failing to reject the null hypothesis is not evidence of it being true.*

Generally speaking, the null hypothesis is: “no change - nothing to see here”.

- **Significance level:** In a hypothesis test, the significance level, alpha, is the probability of making the wrong decision when the null hypothesis is true.

    - Risk of incorrect acceptance. *Audit handout: the financials are NOT materially misstated when they are.* ***Audit effectiveness -*** type II error *(the serious one ☺)*.
    - Risk of incorrect rejection. *Audit handout: Risk that the sample supports the conclusion that the amount is materially misstated when it is not.* ***Audit efficiency*** *- Type I error.*

    Determining audit significance level (and sample size) is a subject for your audit class, but it relates to materiality, risk and cost – Risk and Assurance is an entire field of study.

    We don’t get into sampling in “Analytics” until level 2 (resampling is key to algorithm development)

- **Confidence level:** The probability that, if samples were repeated over and over again *(the core of “frequentist”, i.e., classic, statistics)*, the results obtained would be the same. A confidence level = 1 – alpha *(the compliment of significance level)*
- **Confidence interval:** A range of results from a poll, experiment, or survey that would be expected to contain the population parameter of interest. For example, an average response. Confidence intervals are constructed using significance levels / confidence levels
  (and they’re inversely related – i.e., the greater the confidence level, the smaller the interval)

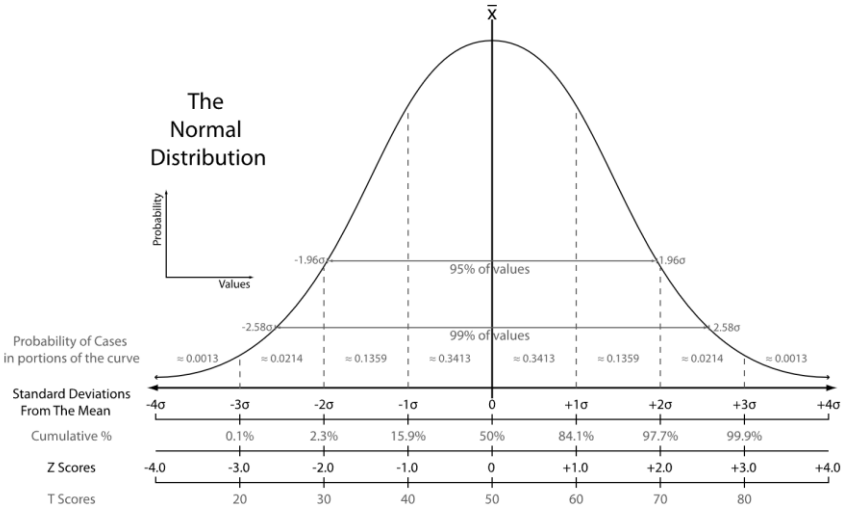
BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Recall from basic stats:

Z	0.00	0.01	0.02	0.03	0.04	0.05
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422

The standard normal distribution table provides the probability that a normally distributed random variable Z, with mean equal to 0 and variance equal to 1, is less than or equal to z where $z = \frac{x - \mu}{\frac{\sigma}{\sqrt{n}}}$.

adjusting for sampling



In this case we want to know the probability of the difference in the means of 2 populations:

$$z = \frac{\mu_0 - \mu_1}{\frac{\sigma}{\sqrt{n}}}$$

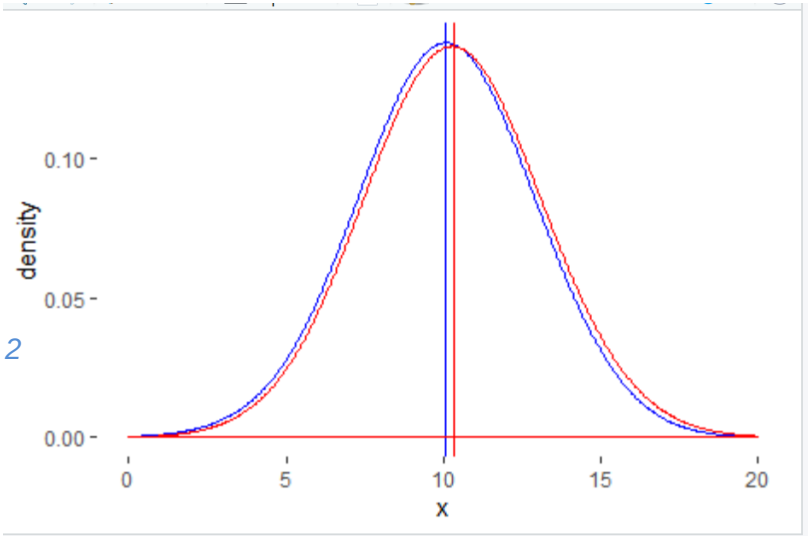
```
z <- (sampMean2 - sampMean)/(sampSD2/sqrt(n))
z
```

$$p = 2 * P(z)$$

```
p <- 2*pnorm(-abs(z))
P
```

*Note: pnorm computes P(–inf, z), so we take the absolute value – set it to the left tail and then multiply * 2*

Two gaussian *(normal)* distributions: one with mean = 10 and the other with mean of 10.25 *(we usually don't know this)*. One random sample of each. Did they come from the same population?




```
> z <- (sampMean2 - sampMean)/(sampSD2/sqrt(n))
> z
[1] 3.634766
> p <- 2*pnorm(-abs(z))
> p
[1] 0.0002782329
```

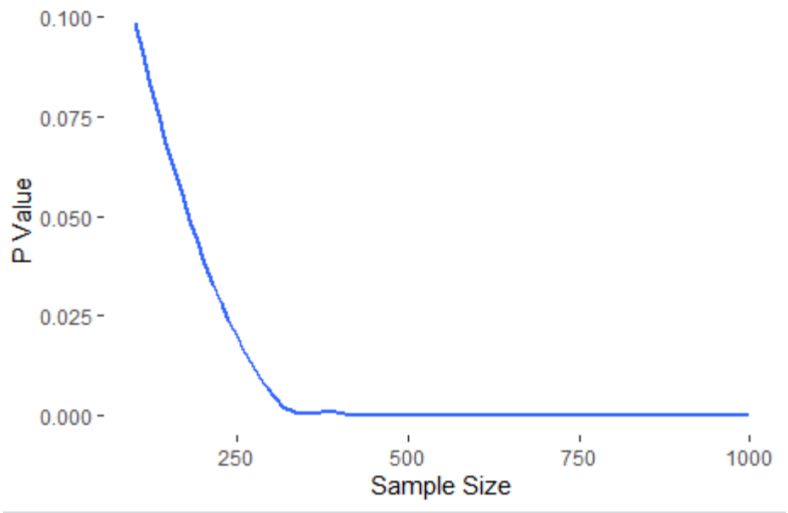
Note that we're really creating a new distribution here – the difference of the base 2

Unlikely.
But is this unlikely enough?
That's why we have an α

Yes if p < α @ .05. But that was one sample – another could yield a different result based on the sampleMean and SD and sample size.

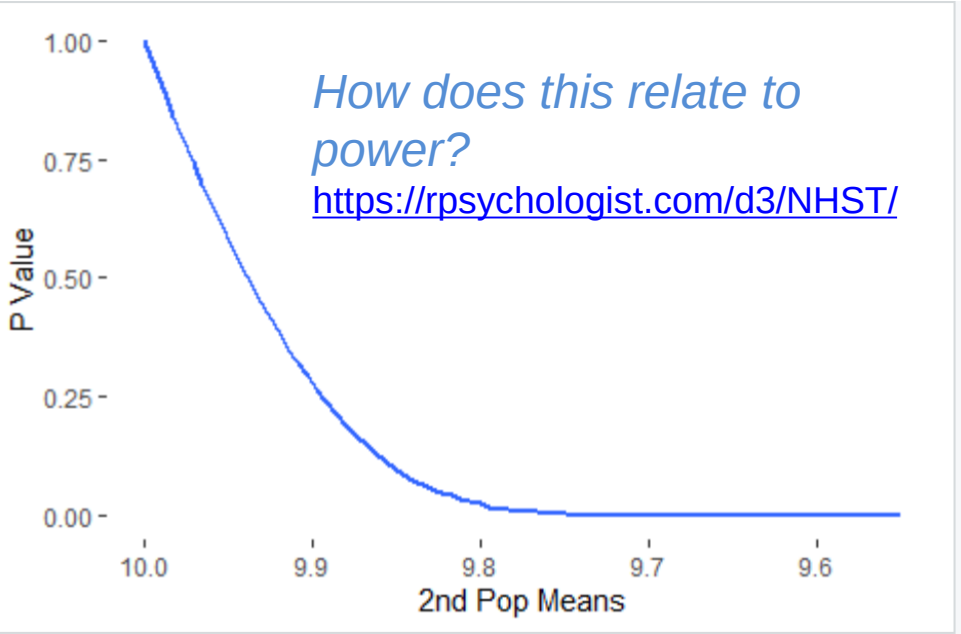
Impact of Sample Size on p-value

```
n <- 100
plotData <- matrix(nrow = 10, ncol = 2)
for(i in 1:10)
{
  n <- i * 100
  samplen <- sample_n(x2, n)
  sampMeann <- mean(samplen$x)
  sampSDn <- sd(sample2$x)
  z <- (sampMean - sampMean2)/(sampSD2/sqrt(n))
  p <- 2*pnorm(-abs(z))
  plotData[i,] <- c(n, p)
}

dfPlotData <- data.frame(x = plotData[,1], y = plotData[,2])
p2 <- ggplot(data = dfPlotData, aes(x=x, y=y)) + geom_smooth(se = F) +
labs(y = 'P Value', x = 'Sample Size')
p2
```



Impact of distance of means on p-value

```
> # so if I need a p less than .05, what size sample do I need?
> # a little more deceptive, how about difference in means
> # other side of the mean just for kicks
>
> plotData <- matrix(nrow = 10, ncol = 2)
> j <- 1
> for(i in seq(9.55,10, by = .05))
+ {
+   n <- 500
+   sampSDn <- 2
+   sampMean <- i
+   p <- 2*pnorm(-abs(10 - sampMean)/(sampSDn/sqrt(n)))
+   plotData[j,] <- c(i, p)
+   j <- j + 1
+ }
>
> dfPlotData <- data.frame(x = plotData[,1], y = plotData[,2])
>
> p2 <- ggplot(data = dfPlotData, aes(x=x, y=y)) +
+   geom_smooth(se = F) +
+   theme(panel.background = element_rect(fill = "white")) +
+   scale_x_continuous(trans='reverse') +
+   labs(y = 'P Value', x = '2nd Pop Means')
> p2
```

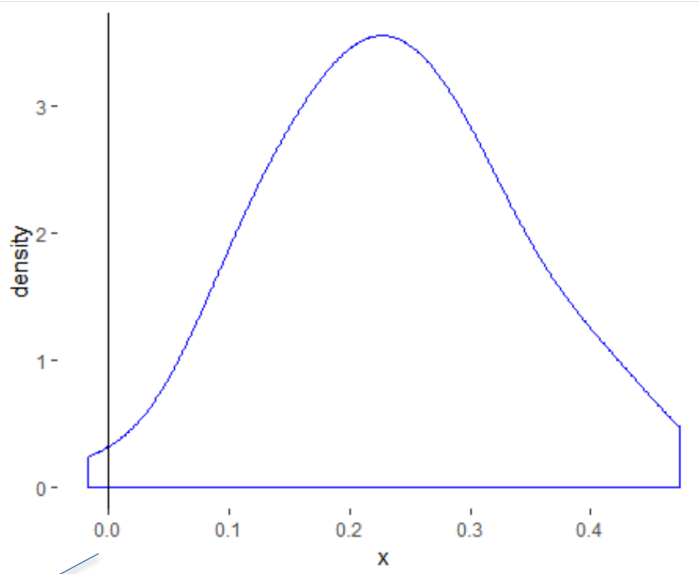


How does this relate to power?

https://rpsychologist.com/d3/NHST/

Looking at this another way:

```
> sdDiff <- sqrt(4/1000 + 4/500)
> meanDiff <- sampMean2 - sampMean
> xDiff <- data.frame(x=rnorm(100, meanDiff, sdDiff))
>
> p1a <- ggplot(data = xDiff, aes(x=x)) +
+   geom_density(color = 'blue', bw =.05) +
+   theme(panel.background = element_rect(fill = "white"))
> p1a
>
> pnorm(0, mean = meanDiff, sd = sdDiff)
[1] 0.01201168
> p1a <- p1a + geom_vline(xintercept = 0)
> p1a
```



The mean of the difference is: $\mu_0 - \mu_1 = .5$

The sd of the difference is: $\sigma_{\text{h1-h0}} = \sqrt{\frac{4}{1000} + \frac{4}{500}} = .1$ (when combining distributions, you can add the 2 variances / n, then take sqrt). Just FYI, More generally, in Bayesian statistics, Posterior $\sigma = \sqrt{\sigma_1^{-2} + \sigma_2^{-2}}$ *(you can add and multiply distributions together). Also, generally speaking, when you combine distributions, variance increases – when you sample parameters from distributions, variance decreases)*

So here, the probability of 0 is .01

In other words, it's unlikely we will draw two samples with no difference from this population
It's also unlikely we'll draw two samples with a large difference (> .5) from this population

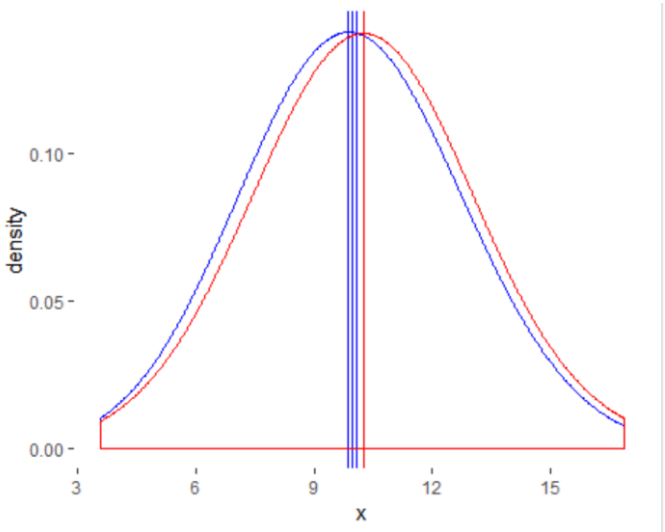
Another way *(better way in my opinion)* is to look at this on the original scale, and calculate the confidence interval: The standard error (SE) of a statistic (usually an estimate of a parameter) is the standard deviation of its sampling distribution[1] or an estimate of that standard deviation. If the parameter or the statistic is the mean, it is called the standard error of the mean (SEM) $\mathrm{SEM} = \frac{\sigma}{\sqrt{n}}$, or $\sqrt{\frac{\sigma^2}{n}}$

Think about this, error increases with variance, and decreases with sample size

Convert to 95% by multiplying by the z score

```
stderr <- function(x) sqrt(var(x,na.rm=TRUE)/length(na.omit(x)))
er <- stderr(sample$x)
error <- qnorm(0.975)*er        95 % confidence
left <- sampMean - error
right <- sampMean + error
```

Is the population H1 mean outside the confidence interval? Then, it's a "significant" difference. "significance" is in quotes because we determined the alpha α value.



Looking ahead, p-values will be used for modeling, but it takes much less "significance" *(pun intended)*

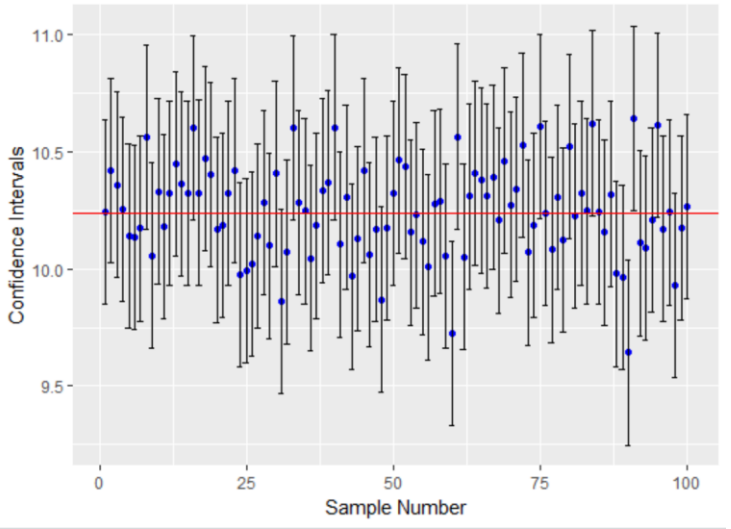
Note: when computing the confidence interval of a simple regression model, CI = est. parameter +- 2*SE

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Confidence Intervals

Taking 100 samples from population with mean = 10.25. The 95% confidence intervals were computed for each sample.

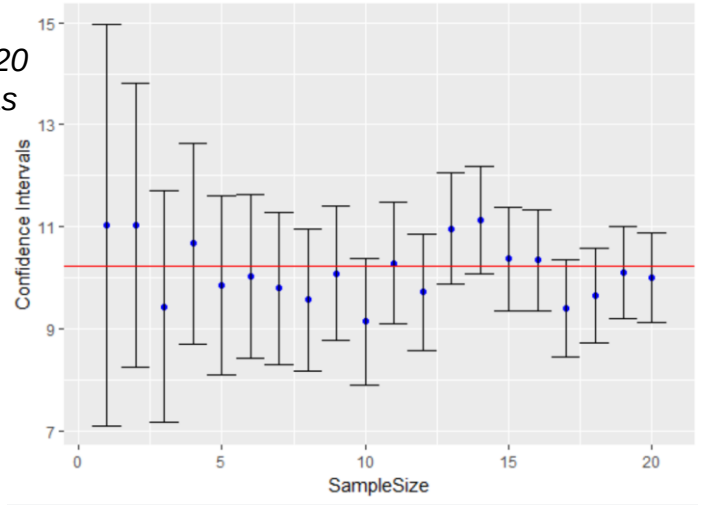
```
> n <- 100
> plotData <- matrix(nrow = n, ncol = 4)
> for(i in 1:100)
+ {
+   samplen <- sample_n(x2, n)
+   sampMeann <- mean(samplen$x)
+   sampSDn <- sd(sample2$x)
+   error1 <- qnorm(0.975)*sampSDn/sqrt(n)
+   left1 <- sampMeann-error1
+   right1 <- sampMeann+error1
+   plotData[i,] <- c(i, sampMeann, left1, right1 )
+ }
>
> dfPlotData2 <- data.frame(plotData)
>
> ggplot(dfPlotData2, aes(x=X1)) + geom_point(aes(y = X2), color = 'blue')+
+   geom_errorbar(aes(ymin=X3, ymax=X4), width=1) +
+     geom_hline(yintercept = mean(x2$x), color = 'red') +
+     labs(y = 'Confidence Intervals', x = 'Sample Number')
>
> tstCnt <- dfPlotData2 %>% filter(X3 < mean(x2$x) & X4 > mean(x2$x)) %>% summarise(c
 = n())
> tstCnt/n
   cnt
1 0.95
```



Note that 95% of the samples include the population mean

Creating 20 samples, increasing sample size from 1 to 20 with 95% confidence level. Notice how the CI narrows as sample size increases.

```
> for(i in 1:20)
+ {
+   n <- i
+   samplen <- sample_n(x2, n)
+   sampMeann <- mean(samplen$x)
+   sampSDn <- sd(sample2$x)
+   error1 <- qnorm(0.975)*sampSDn/sqrt(n)
+   left1 <- sampMeann-error1
+   right1 <- sampMeann+error1
+   plotData[i,] <- c(i, sampMeann, left1, right1 )
+ }
>
> dfPlotData2 <- data.frame(plotData)
>
> ggplot(dfPlotData2, aes(x=X1)) + geom_point(aes(y = X2), color = 'blue')+
+   geom_errorbar(aes(ymin=X3, ymax=X4), width=1) +
+   geom_hline(yintercept = mean(x2$x), color = 'red') +
+   labs(y = 'Confidence Intervals', x = 'SampleSize')
```



P-Values in Regression

We'll see later that we use p-values in regression, but it's a different perspective on significance: the null hypothesis: nothing going on – the independent variable has no effect on the dependent variable

H_0: $\beta_1 = 0$

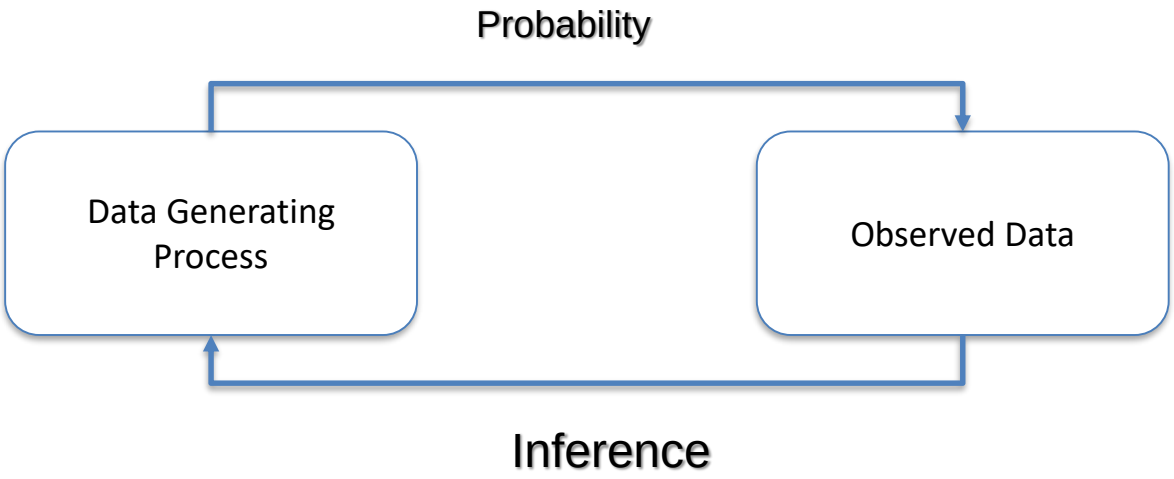
t statistic = $\frac{\beta_1}{SE\ (\beta_1)}$ *, so a large value relative to std error (1 std dev) will reject* H_0

Note, p-values in hypothesis testing for normal distributions follow a uniform distribution. p-values in regression testing follow a f distribution (a fine point but might become a issue some day)

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Review of Probability and Distributions

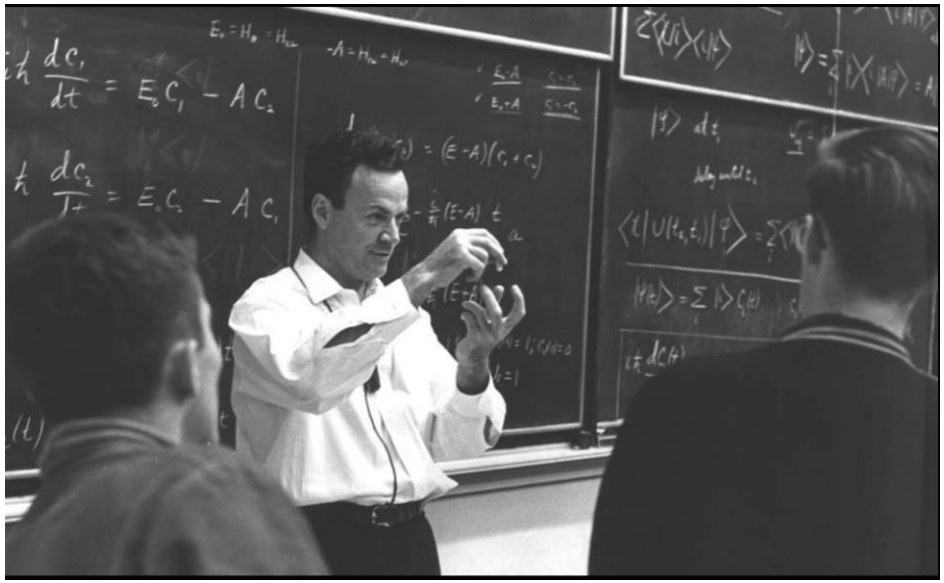
Probability *is the mathematical language for quantifying uncertainty. It is concerned with the properties of outcomes (from a process like sampling or observation). The sample space Ω is the set of all possible outcomes*



Inference *is the process of using data to infer the distribution that generated the data – given the outcomes, What can we say about the population?*

Probability is the basis of modeling, so a good reference on probability is essential. Good books:

- *Statistics – an Introduction using R – Michael Crawley (very accessible and leverages R)*
- *All of Statistics – Larry Wasserman (very comprehensive – seriously)*



"The statements of science are not of what is true and what is not true, but statements of what is known with different degrees of certainty"

Richard Feynman May 11, 1918 – February 15, 1988

Sample Spaces and Events

Sample Spaces and Events

The sample space Ω is the set of all possible outcomes of an experiment. Points ω in Ω are called outcomes. Subsets of Ω are called events.

$$A \cup B = \{\omega \in \Omega: \omega \in A \text{ or } \omega \in B \text{ or } \omega \in \text{both}\}$$

Probability

The function $\mathbb{P}$ that assigns a real number $\mathbb{P}(A)$ to each event A is an probability distribution or a probability measure if it satisfies the following:

$\mathbb{P}(A) \geq 0$ for every A

$\mathbb{P}(\Omega) = 1$

There are many interpretations of $\mathbb{P}(A)$. The two most common are the frequentist *($\mathbb{P}(A)$ is the **long run proportion of times that A is true** in repetitions*, and the Bayesian *($\mathbb{P}(A)$ measures the **observers strength of belief that A is true***.

Probability of Finite Sample Spaces

If Ω is finite and each outcome is equally likely, then $\mathbb{P}(A) = \frac{|A|}{|\Omega|}$, which is called the uniform probability distribution. For example, the probability that the sum of 2 die rolls = 11 is:

```
> outcomes <- rolldie(2)
> n <- sum(outcomes$X1+outcomes$X2==11, na.rm=TRUE)/nrow(outcomes)
> n
[1] 0.05555556
```

Properties of Probabilities

See Probabilities Cheatsheet

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Independent and Identically Distributed Events *(i.i.d.)*

Two events A and B are independent if: $\mathbb{P}(A \cap B) = \mathbb{P}(A) * \mathbb{P}(B)$, for example:

```
s1 <- 5
s2 <- 4
n <- 10
A <- sample(1:n,s1,rep=F)
B <- sample(1:n,s2,rep=F)
PA <- s1/n
PB <- s2/n
PA*PB
```

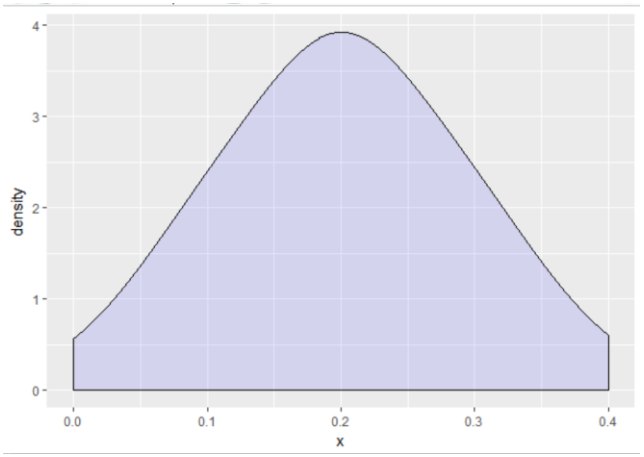
Take 2 random samples, A (n=5) and B (n=4) [they don't have to be the same size], of a sequence (1:10)

Get $\mathbb{P}(A)* \mathbb{P}(A) =$

```
> PA <- s1/n
> PB <- s2/n
> PA*PB
[1] 0.2
```

```
n2 <- 500
PC <- matrix(NA, nrow = n2, ncol = 1)
for(i in 1:n2)
{
A <- sample(1:n,s1,rep=F)
B <- sample(1:n,s2,rep=F)
length(A[A %in% B])
PC[i,1]<- length(A[A %in% B])/n
}
dfPC <- data.frame(PC)
colnames(dfPC)[1] <- 'x'
mean(dfPC$x)
ggplot(data = dfPC) +
  geom_density(aes(x = x), bw=.06, fill = 'blue', alpha = .1)
```

Then take 500 samples from the same population (seq 1:10) and compare $\mathbb{P}(A \cap B)$

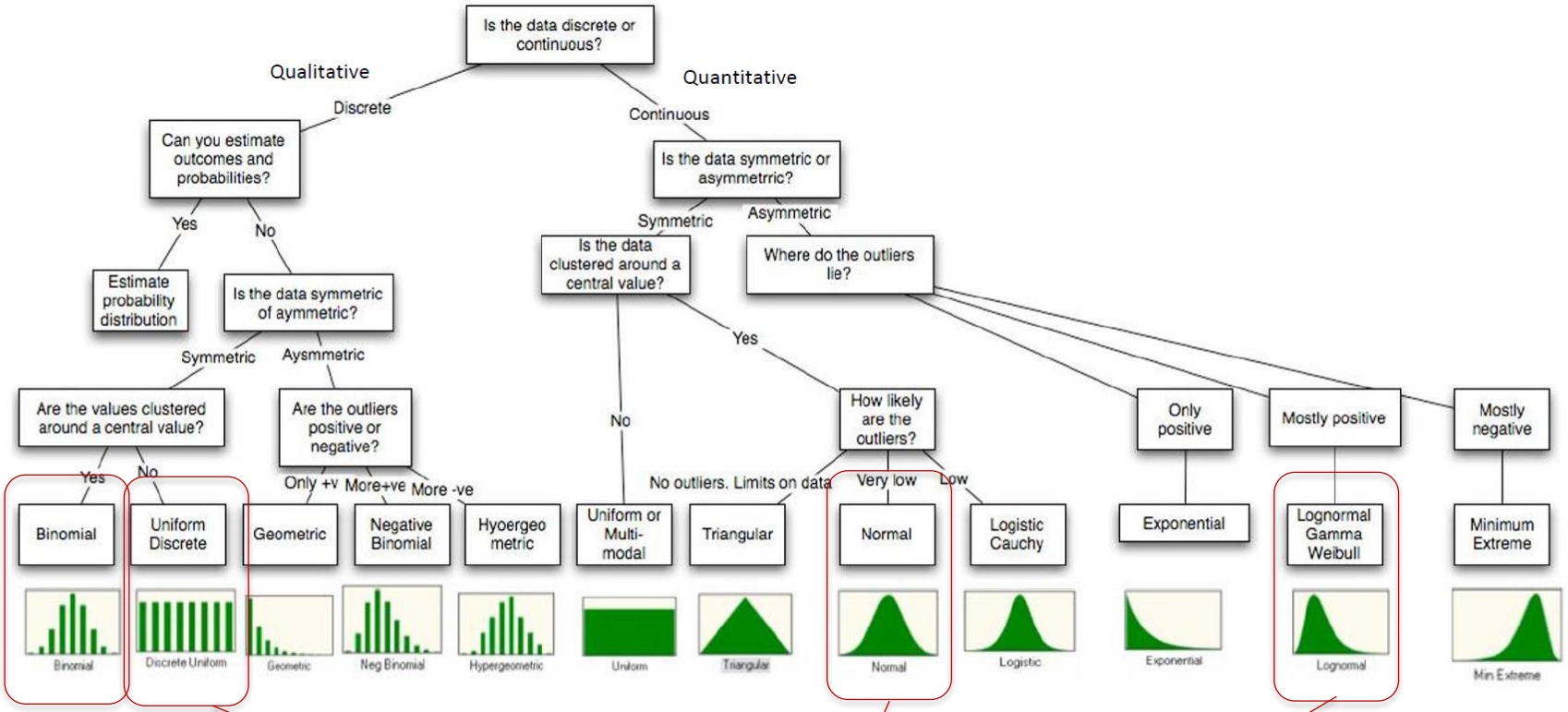


Independent means that the outcome of A has no bearing on the probability of B

Identically distributed means that the samples come from the same population - or very close in distribution parameters (μ,σ)

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Distributions



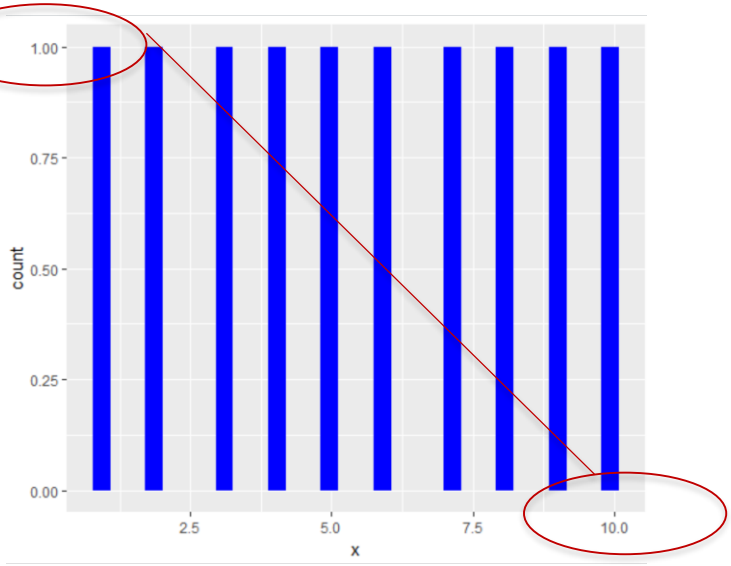
Distributions selected for our study

 # Uniform Discrete

```
x <- seq(1, 10, 1)
y <- sample(1:10,length(x),rep=F)
myPlot <- data.frame(x = x, y = y)
ggplot(myPlot,aes(x))+geom_histogram(fill="blue")
```

1.0

*What's 1 * 10?*



Cumulative Density Function *(CDF)*

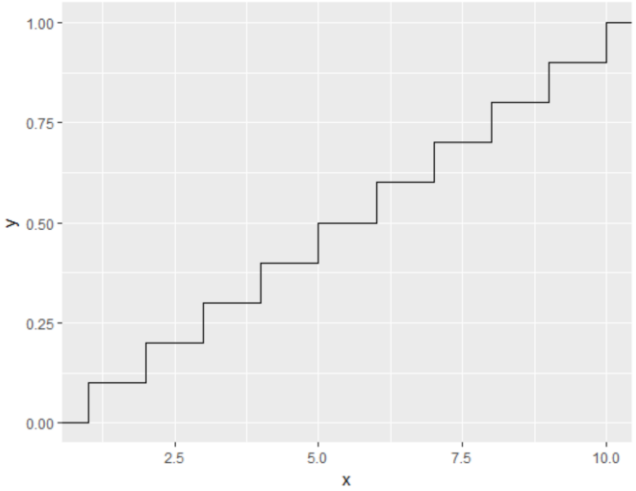
```
> den <- cbind(x, data.frame(y = dunif(x, 0, 10)))
> ggplot(den, aes(x=x, y=y))+ geom_area(fill="blue")
> totDen <- sum(den$y) # density function values will always sum to one.
> totDen
[1] 1
> # density of a dunif function is always a sqaure, so
> denCk <- max(den$y)*max(den$x)
> denCk
[1] 1
```

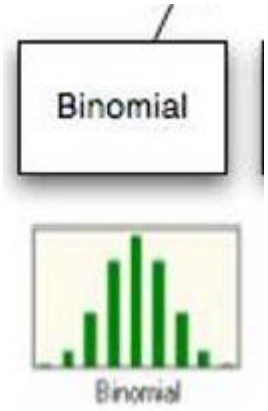
ggplot(myPlot, aes(x)) + stat_ecdf()

*The **empirical CDF** is an estimate of the true CDF it is found by making no assumptions about the underlying distribution.*

A function is ***monotonic*** if it is either entirely non-increasing, or entirely non-decreasing

If continuous, the function is also ***asymptotic*** because it approaches 1 as *x* goes to infinity.





Binomial Discrete Distribution

```
n <- 10
# can't make this too big or you'll have to scale the p's
# just to play with

x <- seq(1,n,by=1)
y <- sample(0:1,length(x),rep=T)
bdata <- data.frame(x=x, y=y)
```

	x	y
1	1	0
2	2	1
3	3	1
4	4	0
5	5	1
6	6	0
7	7	0
8	8	1
9	9	0
10	10	1

$$\frac{n!}{\mathrm{h}!(\mathrm{n}-\mathrm{h})!}p^h(1-p)^{\mathrm{n}-\mathrm{h}}$$

```
> p <- .5
> h <- 5 # sum(y)
> h
[1] 5
> x <- factorial(n)/(factorial(h)*factorial(n-h))
> x*(p)^h*(1-p)^(n-h)
[1] 0.2460938
```

Keep this in your head – we're coming back. Understand what it means, how it works…

$$\frac{10!}{5!(10-5)!}.55(1-.5)^{10-5} =$$

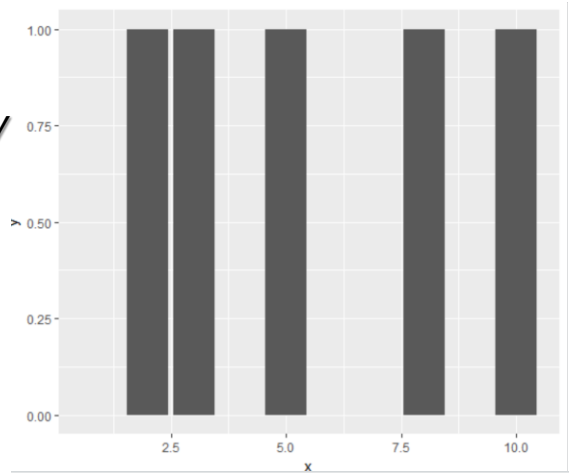
.24

So this means that the probability of observing 5 1's in a set of 10 events is .24… we will plot this probability density…

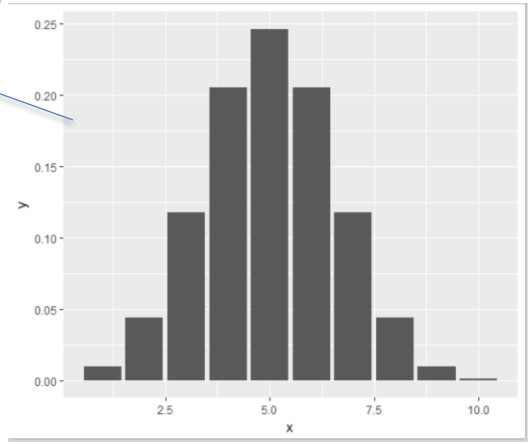
The dbinom function is the binomial pmf. So let's use that here, and create a density plot for different values of h (successes). In this 'roll', we had 5 *(the sample will produce random # successes)*

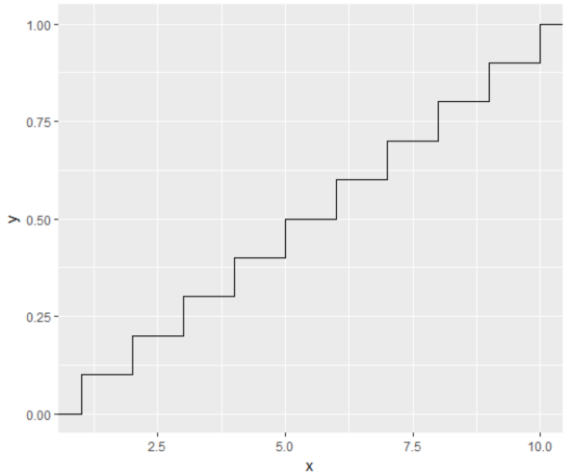
```
# now let's use dbinom (instead of the formula) # and plot the pmf for observing h successes

h <- seq(1,n,by=1)
y <- sample(0:1,length(h),rep=T)
bdata <- data.frame(x=h, y=y)
py <- dbinom(h,n,.5)

# computes the PMF of h vector for n trials with .5 probability of success on each  Trial
pb <- data.frame(x=h, y = py)
ggplot(data = bdata)+ geom_histogram(aes(x = x, y=y), stat = 'identity')
ggplot(data = pb)+ geom_histogram(aes(x = x, y=y), stat = 'identity')
ggplot(pb, aes(x)) + stat_ecdf()
```

frequency



density





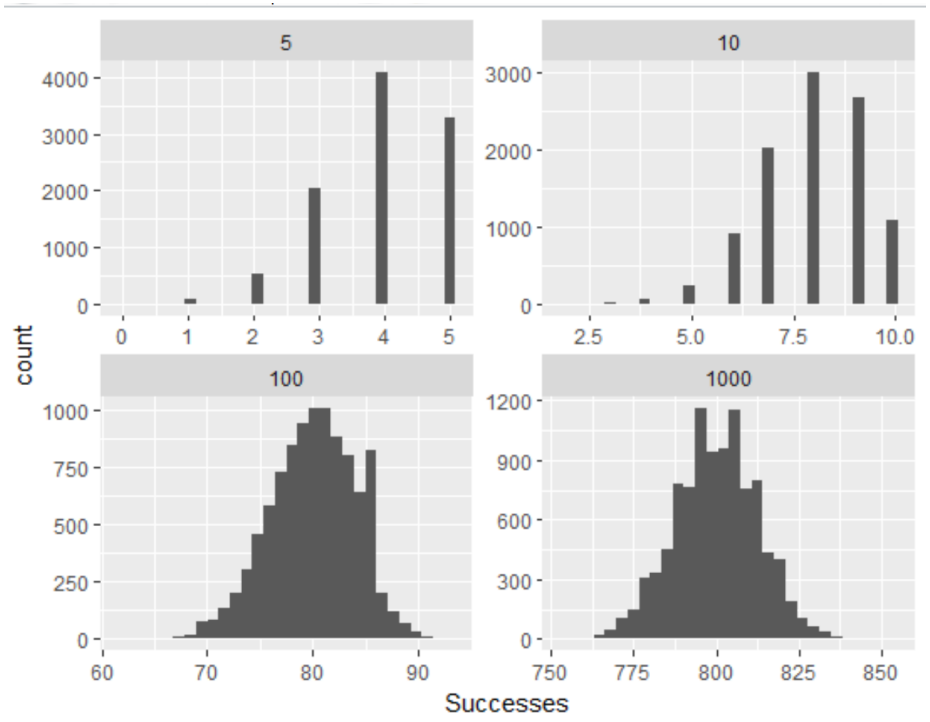
What is the mean?
Remember this…

And this is how dbinom behaves as we increase sample size

Discrete Binomial with increasing Sample Size

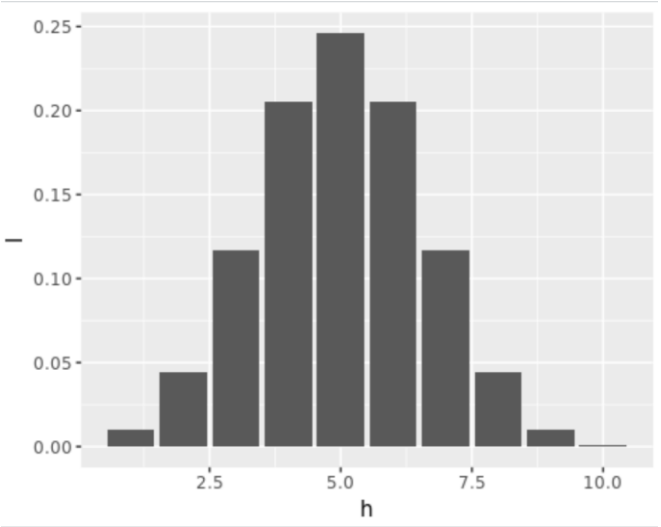
frequency

```
binom5 <- data.frame(Successes=rbinom(n = 10000, size
= 5, prob =.8), Size = 5)
binom10 <- data.frame( Successes = rbinom( n = 10000,
size = 10, prob = 0.8), Size = 10)
binom100 <-data.frame( Successes = rbinom(n = 10000,
size = 100, prob =0.8), Size = 100)
binom1000 <- data.frame( Successes = rbinom( n = 10000,
size = 1000, prob = 0.8), Size = 1000)
binomAll <- rbind( binom5, binom10, binom100, binom1000)
ggplot(binomAll, aes( x = Successes)) +
geom_histogram(bins = 30) + facet_wrap(~Size, scales =
"free")
```



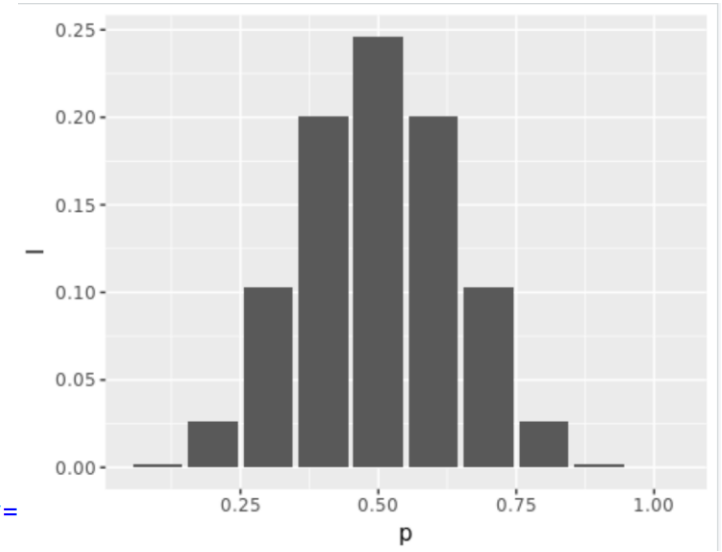
so now let's go back to our pmf and plot it out for a range of h's (# successes). This is what dbinom is doing for n in the background, but here you can see the matrix that's used to plot the pmf

```
> n <- 10
> p <- .5
> h <- seq(1, 10, 1)
> h
 [1]  1  2  3  4  5  6  7  8  9 10
> x <- factorial(n)/(factorial(h)*factorial(n-h))
> #p <- seq(.1, 1, .1)
> l <- x*(p)^h*(1-p)^(n-h)
> ProbMatrix <- data.frame(h, l)
> ProbMatrix
    h           l
1   1 0.0097656250
2   2 0.0439453125
3   3 0.1171875000
4   4 0.2050781250
5   5 0.2460937500
6   6 0.2050781250
7   7 0.1171875000
8   8 0.0439453125
9   9 0.0097656250
10 10 0.0009765625
> ggplot(data = ProbMatrix)+ geom_histogram(aes(x = h, y=l), stat = 'identity')
```



So let's change things a bit; ***instead of varying h – let's vary p*** and plot a pmf for different values of p

```
> n <- 10
> h <- 5 # sum(success)
> x <- factorial(n)/(factorial(h)*factorial(n-h))
> p <- seq(.1, 1, .1)
> l <- x*(p)^h*(1-p)^(n-h)
> ProbMatrix <- data.frame(p, l)
> ProbMatrix
     p           l
1  0.1 0.001488035
2  0.2 0.026424115
3  0.3 0.102919345
4  0.4 0.200658125
5  0.5 0.246093750
6  0.6 0.200658125
7  0.7 0.102919345
8  0.8 0.026424115
9  0.9 0.001488035
10 1.0 0.000000000
> ggplot(data = ProbMatrix)+ geom_histogram(aes(x = p, y=
```



Note that probability is now on the x axis. Let's also rethink this a bit: could p also be the most likely value for the mean of the distribution? If that's so, then we would be mapping the likelihood of the distribution parameters. And also keep in mind, that a distribution is a model.

One last thing about the binomial for now *(we'll come back to this later)*.

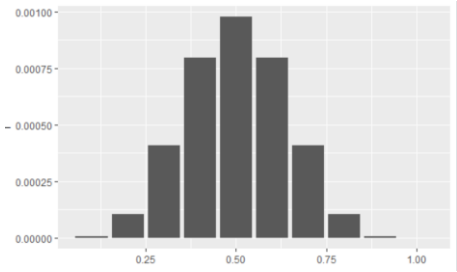
In working with the binomial pdf

$$\frac{n!}{\mathrm{h!(n-h)!}}p^h(1-p)^{\mathrm{n-h}}$$

we can drop the ***constant*** $\frac{n!}{\mathrm{h!(n-h)!}}$ *(wrt p)* and just work with $p^h(1-p)^{\mathrm{n-h}}$ and still get to the same place. This won't work when comparing across different distributions *(or conjugate priors in Bayesian)*, but it's always nice to work with $\propto$ simple equation, as it gets complex later on!

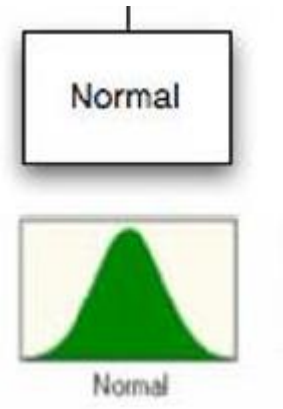
```
> p <- seq(.1, 1, .1)
> l <- (p)^h*(1-p)^(n-h)
> ProbMatrix <- data.frame(p, l)
> ggplot(data = ProbMatrix)+ geom_histogram(aes(x = p, y=l), stat = 'identity')
```

```
> ProbMatrix
    p           l
1  0.1 0.0000059049
2  0.2 0.0001048576
3  0.3 0.0004084101
4  0.4 0.0007962624
5  0.5 0.0009765625
6  0.6 0.0007962624
7  0.7 0.0004084101
8  0.8 0.0001048576
9  0.9 0.0000059049
10 1.0 0.0000000000
```

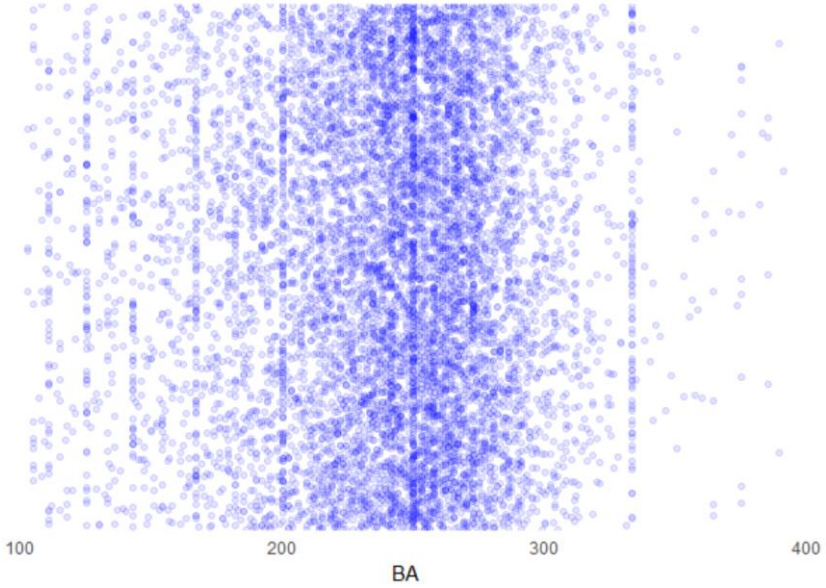



```
> # and if I just want the maximum, it's:
> MaxProb <- ProbMatrix[which(ProbMatrix$l==max(ProbMatrix$l)),]
> MaxProb
   p           l
5 0.5 0.0009765625
>
> # if you want the probability, you can also multiply times the constant
> x*MaxProb
   p         l
5 126 0.2460938
```

OK that's it for the binomial for now, but we're coming back soon!




```
# -------------------- Continuous Distributions ----------------
library(tidyverse)
library(stringr)
library(Lahman)

career <- Batting %>%
  dplyr::filter(AB > 0) %>%
  anti_join(Pitching, by = "playerID") %>%
  group_by(playerID) %>%
  summarize(H = sum(H), AB = sum(AB)) %>%
  mutate(average = H / AB)

career <- Master %>%
  tbl_df() %>%
  dplyr::select(playerID, nameFirst, nameLast) %>%
  unite(name, nameFirst, nameLast, sep = " ") %>%
  inner_join(career, by = "playerID")

career <- mutate(career, BA = round(average*1000, 0))

pBA <- dplyr::filter(career, BA > 100 & BA < 400 )

binwd <- 5

p <- ggplot( data = pBA) + geom_point( aes(x = BA, y = playerID),
color = 'blue', alpha = 0.1)
p <- p + theme(axis.text.y=element_blank(),axis.ticks=element_blank(),
        axis.title.y=element_blank())
p
```

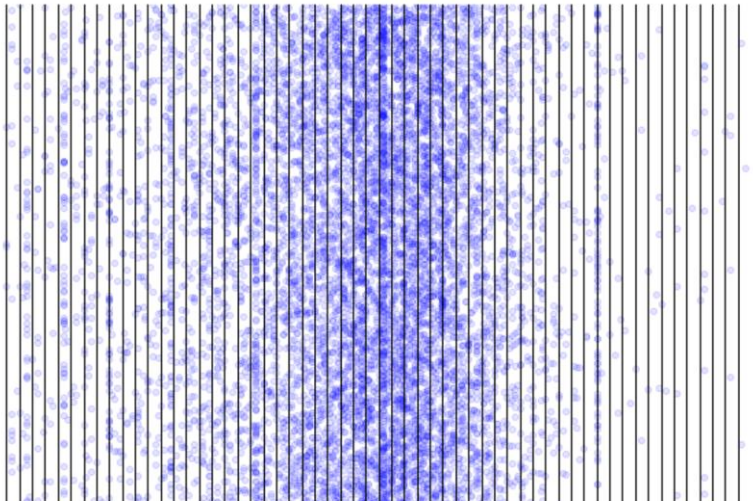


 # NOTE replace summarize with summarise (package conflics)!!!!

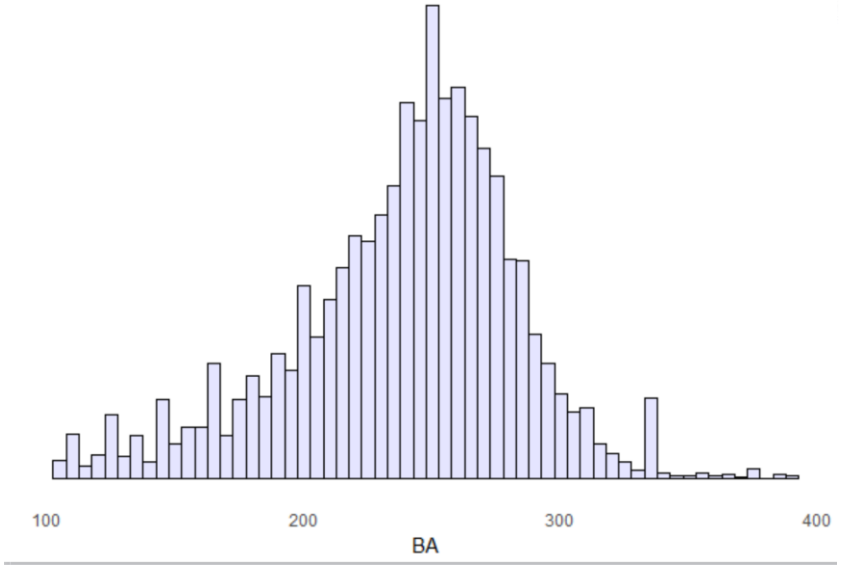
BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Can we convert a continuous to a discrete distribution?

```
# showing bins with random data
xVec <- seq(min(pBA$BA), max(pBA$BA), by = binwd)
p <- p + geom_vline(xintercept = (xVec))
p
```



```
p <- ggplot( data = pBA) + geom_histogram( mapping =
aes( x = BA), binwidth = binwd, fill = 'blue', alpha = 0.1,
col = 'black')
p <- p +
theme(axis.text.y=element_blank(),axis.ticks=element_bl
ank(),
            panel.background = element_rect(fill = "white"),
             axis.title.y=element_blank())
p
```

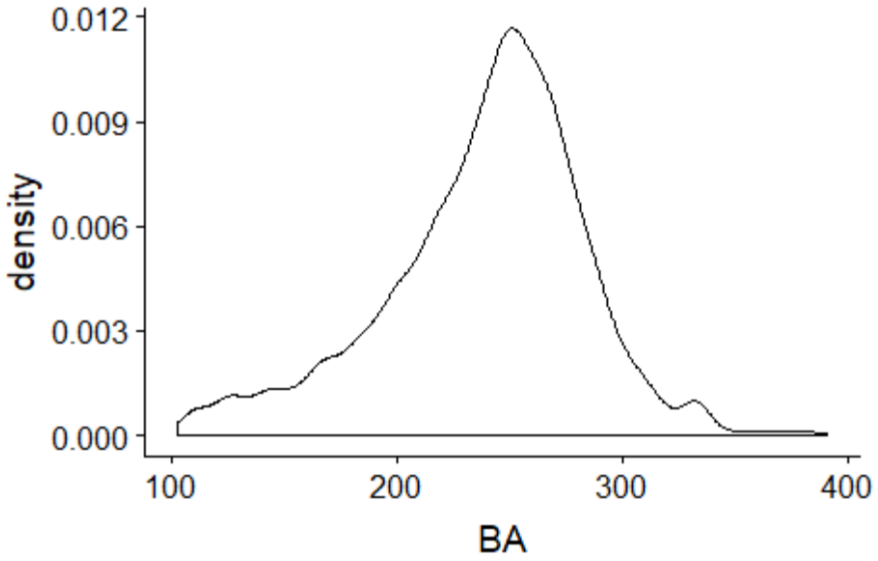


This is an important technique that we will return to again and again

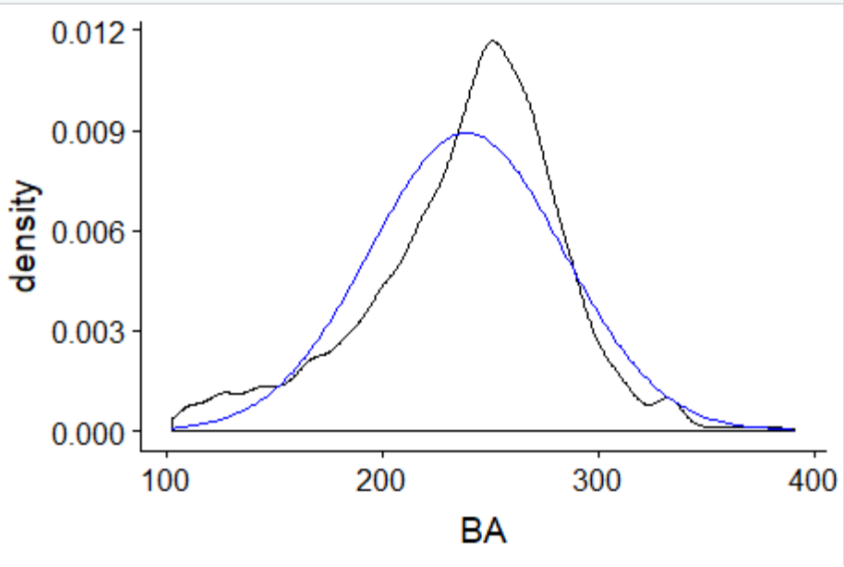
BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Back to Continuous Distribution Perspective

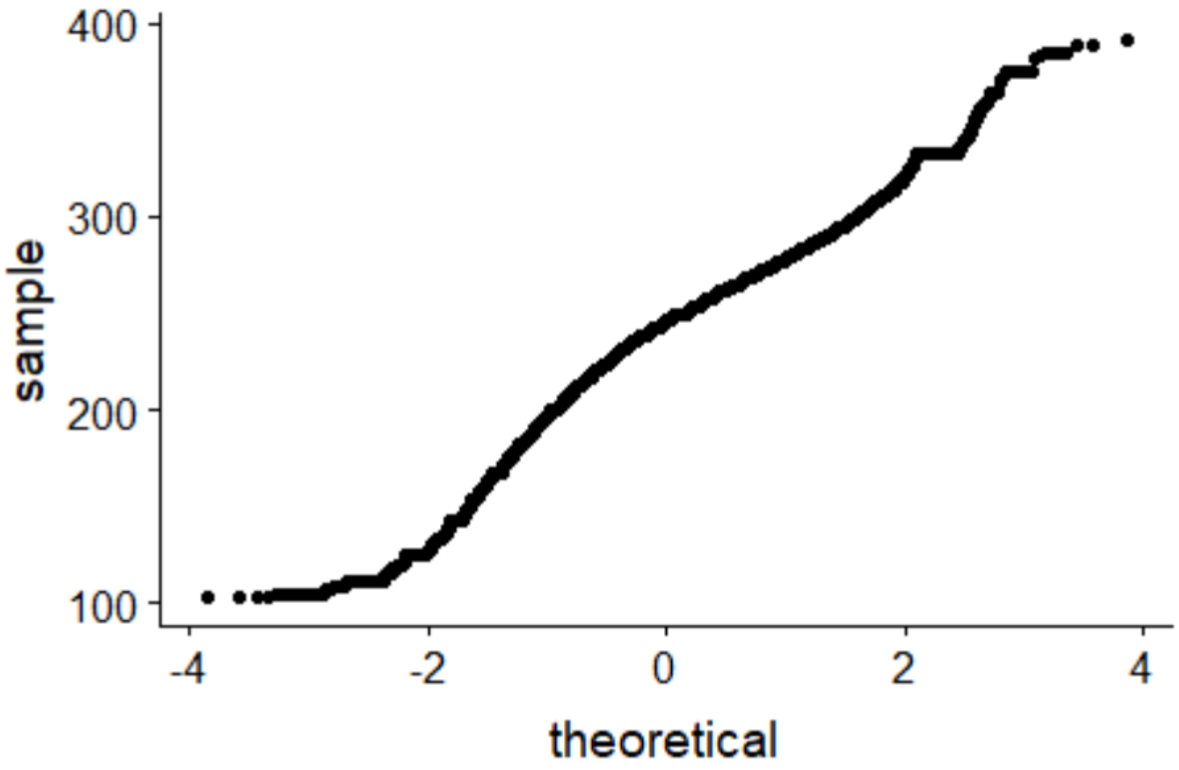
```
df2 <- density(pBA$BA)
x2 <- data.frame(x = df2$x, y = df2$y)
ggplot(x2, aes(x,y)) + geom_line()
```




```
> p
> p <- p + geom_line(data = pBA, aes(x = BA, y = dnorm(BA,
+   mean = mean(pBA$BA), sd = sd(pBA$BA))), color = "blue" )
> p
```



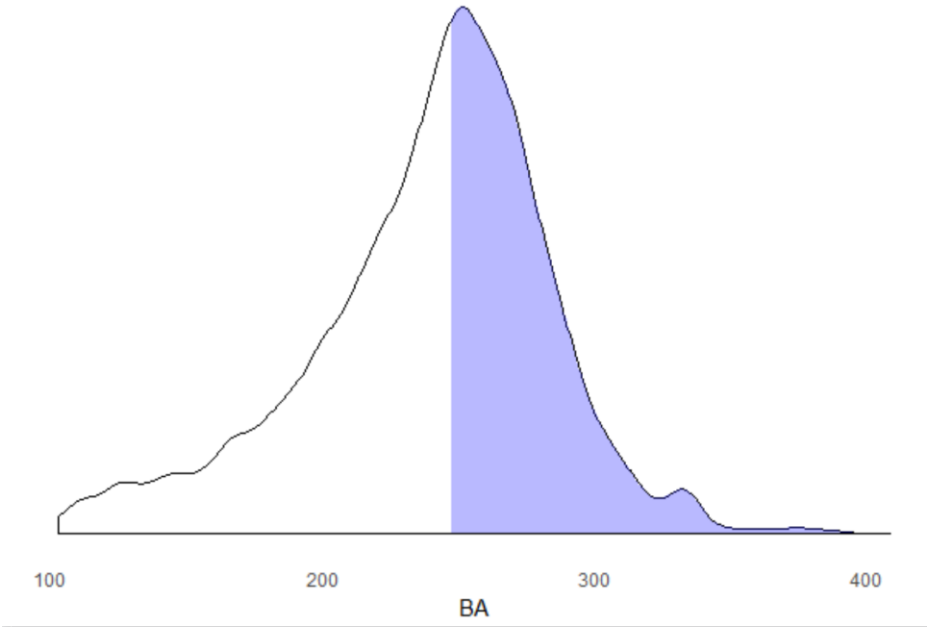
```
> ggplot(pBA, aes(sample = BA)) +
+   stat_qq()
```



BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

…Continuous Distribution *(Analysis)*

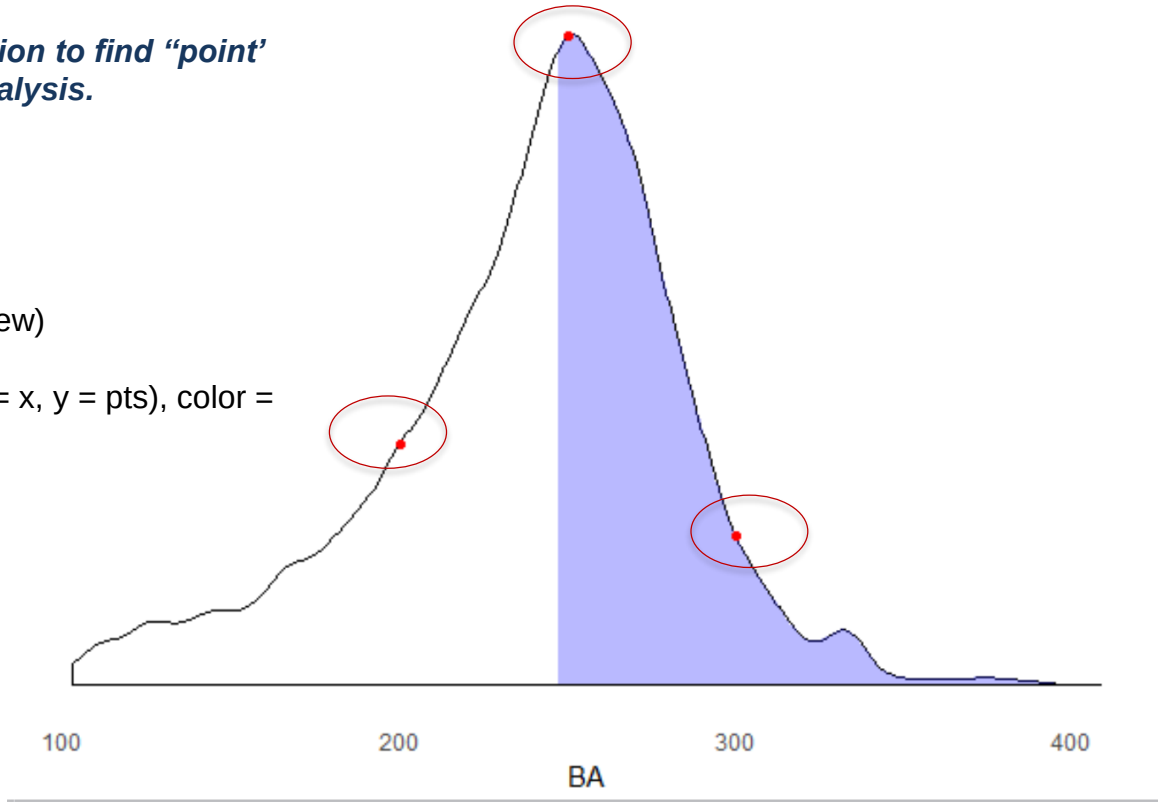
```
# interpolate density function (i.e., get the estimated function)

df <- approxfun(density(pBA$BA))
df2 <- density(pBA$BA)
x2 <- data.frame(x = df2$x, y = df2$y)
ggplot(x2, aes(x,y)) + geom_line()

# interpolate density function and integrate area for all batters and top
batters (over the mean)

allbat <- integrate(approxfun(x2$x,x2$y), lower = 100, upper = 400)
topbat <- integrate(approxfun(x2$x,x2$y), lower = mean(x2$x), upper
= 400)

# check that it's ~ 50%
topbat$value

# set up data for mapping and show probability
topBatters <- dplyr::filter(x2, x > mean(x)) %>% arrange(x)
topBatters2 <- rbind( c( min( topBatters $ x), 0),
                + topBatters,
                + c( max( topBatters $ x), 0))
p <- p + geom_polygon( data = topBatters2, aes( x = x, y = y), fill =
'blue', alpha = 0.1)
p <- p + theme(axis.text.y=element_blank(),axis.ticks=element_blank(),
          panel.background = element_rect(fill = "white"),
          axis.title.y=element_blank())
p
```



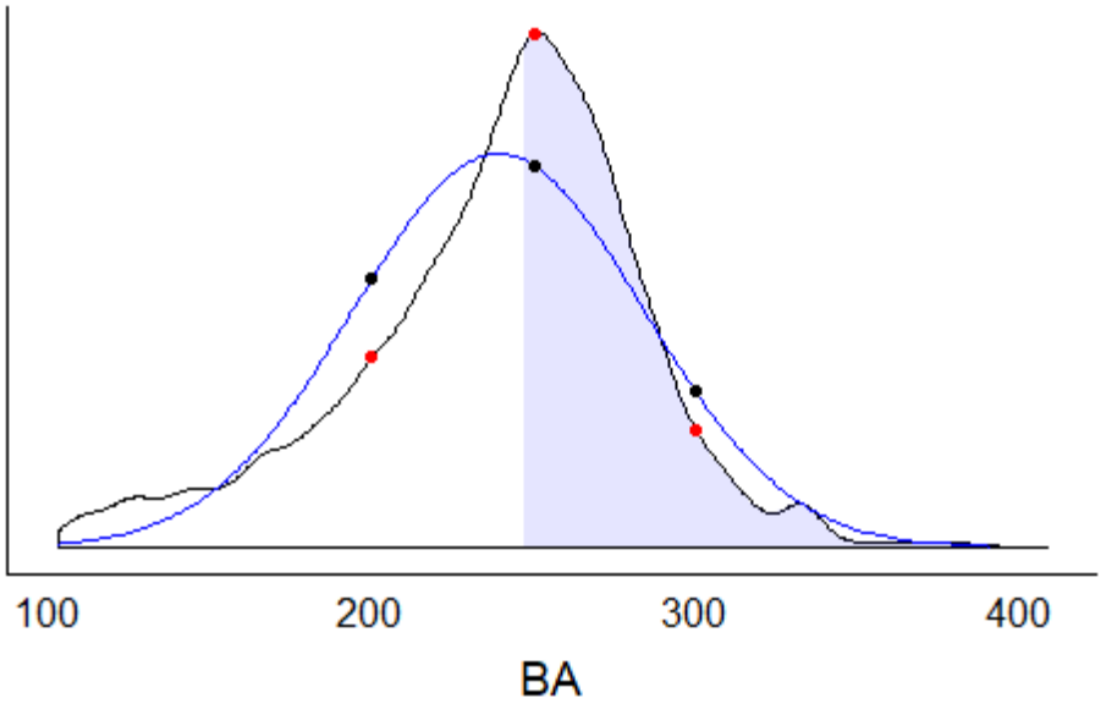
approxfun uses spline interpolation - we will study this in regression.

You can also use the interpolated function to find "point' values which is handy for predictive analysis.

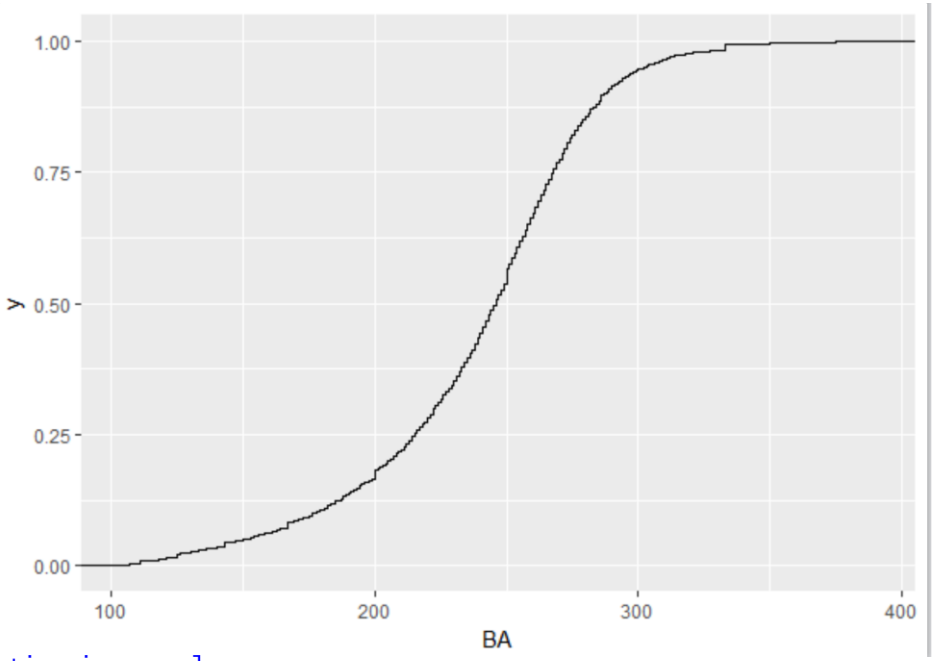
```
df <- approxfun(x2$x,x2$y)
xnew <- c(200,250,300)
dfPts <- data.frame(pts = df(xnew), x = xnew)

p <- p + geom_point( data = dfPts, aes( x = x, y = pts), color =
'red')
p
```



```
> dfPts <- data.frame(pts = df(xnew), x = xnew)
>
> p <- p + geom_point( data = dfPts, aes( x = x, y = pts), color = 'red')
> p
> dfPts <- dfPts %>% mutate(dNormPts = dnorm(x, mean  = mean(pBA$BA), sd = sd(pBA$BA)))
>
>
> p <- p + geom_point(data = dfPts, aes(x = dfPts$x, y = dNormPts))
> p
>
```



and looking at the CDF - notice how it crosses .5 @ 250



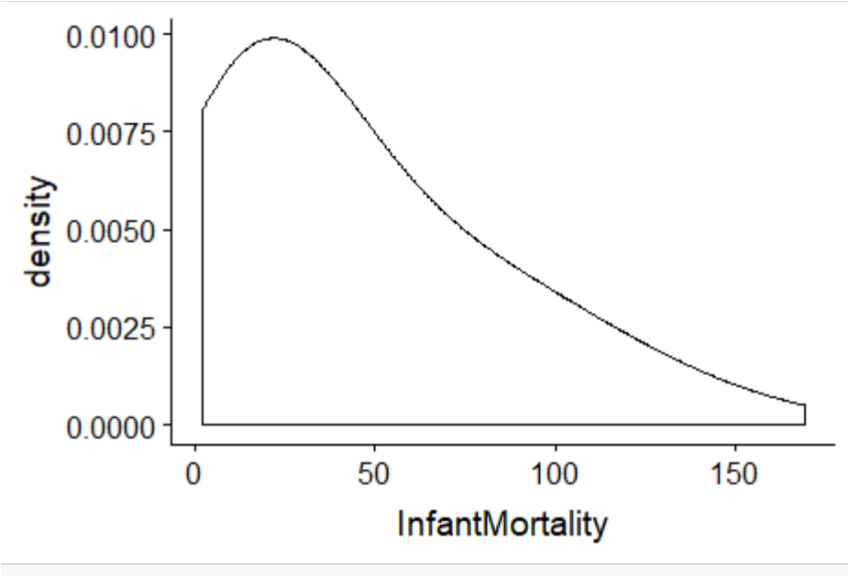

```
ggplot(pBA, aes(BA)) + stat_ecdf(geom = "step")
P = ecdf(pBA$BA)
P(250)
# cdf again
ggplot(pBA, aes(BA)) + stat_ecdf()
```

```
> set.seed(1)
> shapiro.test(rnorm(5000)) # cannot reject null that distribution is normal

        Shapiro-Wilk normality test

data:  rnorm(5000)
W = 0.99957, p-value = 0.3352

>
> shapiro.test(sample(pBA$BA, 50))# cannot reject null that distribution is normal

        Shapiro-Wilk normality test

data:  sample(pBA$BA, 50)
W = 0.97874, p-value = 0.5001
```

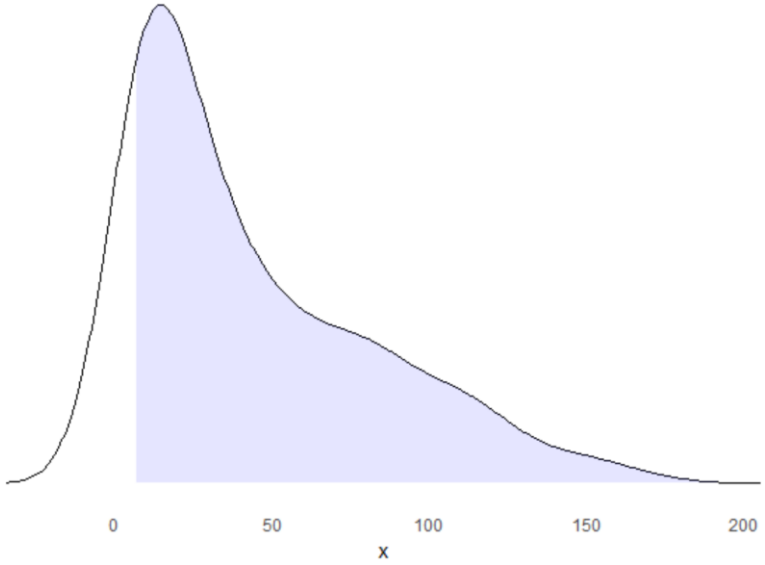
BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Skewed

```
> UN <- read_csv("UN.csv")
Parsed with column specification:
cols(
  infant.mortality = col_double(),
  gdp = col_double(),
  country = col_character()
)
> dfIM <- UN %>% filter(!is.na(infant.mortality))
> dfIM <- dfIM %>% rename("InfantMortality" = "infant.mortality")
>
> ggplot(dfIM, aes(InfantMortality, ..density..)) + geom_density(bw = 25)
> ggplot(dfIM, aes(sample = InfantMortality)) + stat_qq()
> ggplot(dfIM, aes(InfantMortality)) + stat_ecdf()
>
```



```
> USim <- as.numeric( dplyr::filter(dfIM, country == 'United.States') %>% dplyr::select(InfantMortality))
> countriesExceeding <- dplyr::filter(dfIM, InfantMortality > USim)
> nrow(countriesExceeding)/ nrow(dfIM)
[1] 0.8507463
>
>
>
> df4 <- density(dfIM$InfantMortality) # empirical density
> x2 <- data.frame(x = df4$x, y = df4$y)
> p <- ggplot(x2, aes(x,y)) + geom_line()
> p
```



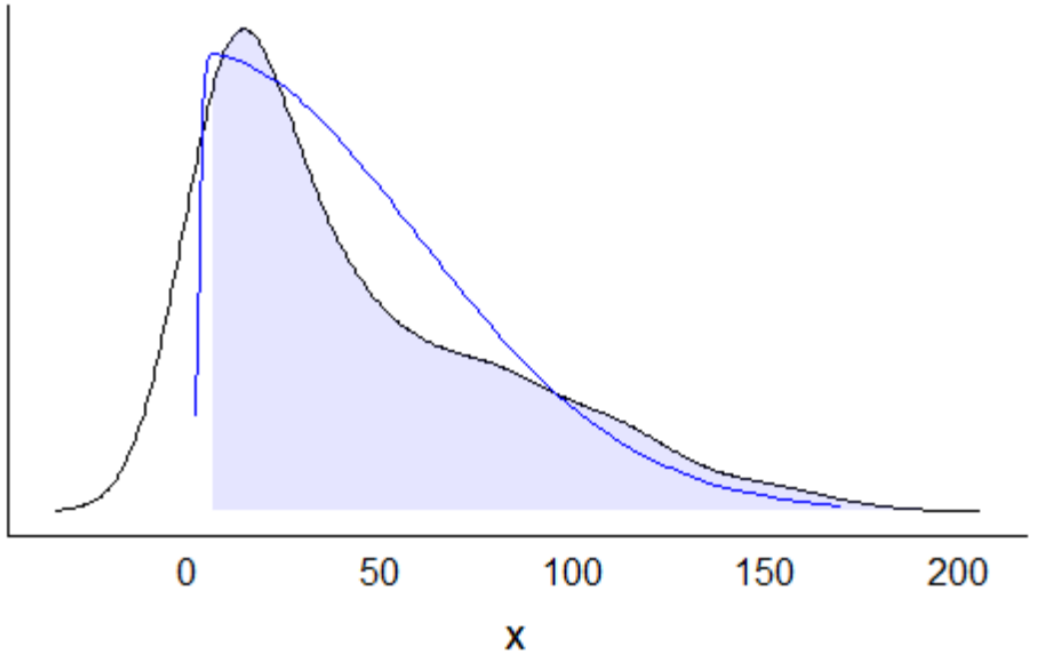
SN package – estimation and simulation

```
> library(sn)
>
> UNestMod <- sn.mple(y = dfIM$InfantMortality, opt.method = "nlminb")$cp
> UNestParam <- cp2dp(UNestMod, family = "SN")
>
> exi <- UNestParam[1]
> eomega <- UNestParam[2]
> ealpha <- UNestParam[3]
>
> dfIM <- dfIM %>% mutate(dSN =dsn(InfantMortality, xi = exi, omega = eomega, alpha = ealpha) )
>
> p <-p + geom_line(data = dfIM, aes(x = InfantMortality, y  = dSN), color = "blue")
> p
```

Estimate skew normal parameters

Use parameters to generate density values

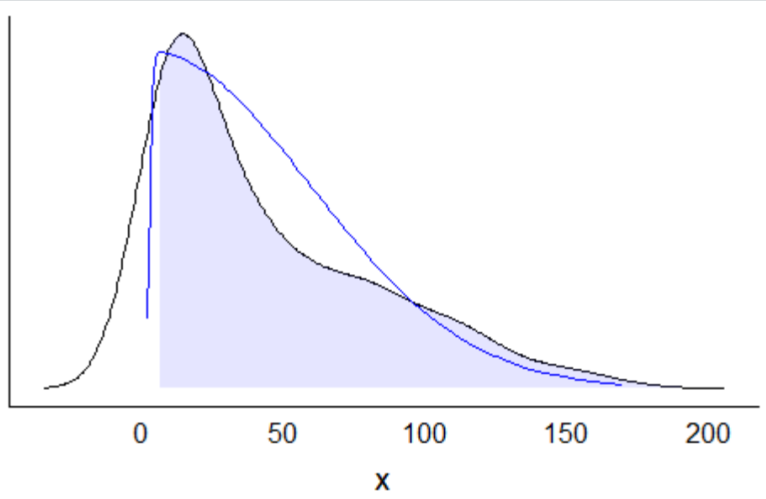
dsn is like dnorm, but with skew normals



```
> 1 -psn(USim, xi= exi, omega = eomega, alpha = ealpha )
[1] 0.9408829
> count(filter(simUN, InfantMortality >= 7))/nrow(simUN) # include countries TIED
         n
1 0.920398
>
```

And psn is like pnorm. Estimate mortality over US (7)

Sn deals with this problem better and handles ties
(like the mode problem earlier)



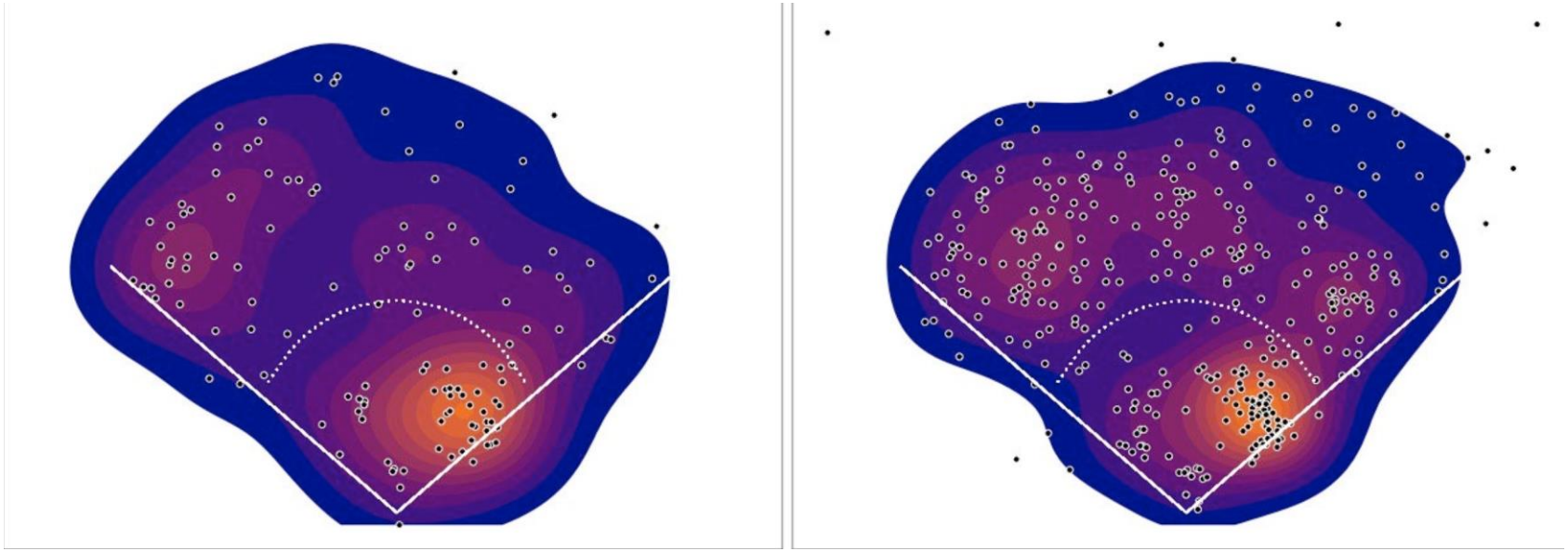
BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Probability and Distributions Cont'd

New resources available free on internet:

New Advanced R from Hadley (WIP) https://adv-r.hadley.nz/

Fundamentals of Data Visualization: http://serialmentor.com/dataviz/



Cool new R tips blog: https://nathaneastwood.github.io/awesome-rtips/

Methods for estimating population parameters

1. Method of Moments
2. Maximum Likelihood

Method of Moments

Checking whether the model's simulation output looks like the data naturally suggests the idea of adjusting the model until it does. This becomes a way of estimating the model. Forms of this involve adjusting parameters of the model until the simulations do look like the data. The most straightforward form of simulation-based inference is the method of moments.

We have a model with a parameter vector θ, and pick a vector m of moments to calculate. The moments, like the expectation of any variables, are functions of the parameters,

- $m = g(\theta)$ for some function g.

If that g is invertible, then we can **recover the parameters from the moments *[$\theta = g^{-1}(m)$]***

- The method of moments estimator takes the observed, sample moments m, and plugs them into the inverted function $\theta_{MM} = g{-}1(m)$. If it's hard to explicitly solve for parameters from moments, we can use minimization:

$\theta_{MM} = \text{argmin}_\theta \, ||g(\theta) - m||^2$

Often, but not always, the first moment m we solve for is the sample mean *(followed by the variance, and then higher order moments like skewness…)*.

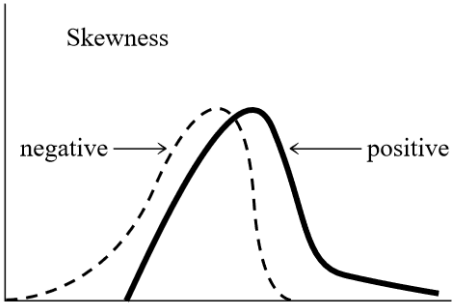
In it's simplest form, a Method of Moments procedure will follow these steps:

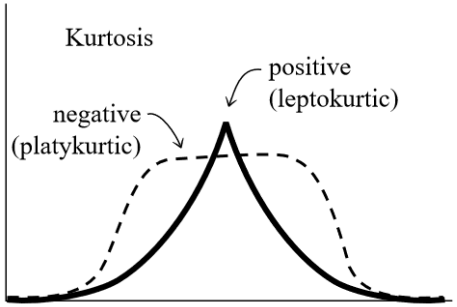
Generate *multiple samples (e.g., resampling)* from a population:
You've seen how multiple samples will generate variance, so now we have to estimate:

- Equate moment 1 $M_1 = \frac{1}{n}\sum_{i=1}^{n} X_i$ to the first *(sample)* theoretical moment
- Equate moment 2 $M_2 = \frac{1}{n}\sum_{i=1}^{n} X_i$ to the second *(sample)* theoretical moment
- *Continue until you have as many equations as parameters*
- *Solve simultaneous equations (most MoM algortihms minimize a cost function – shown earlier)*

Other simulation based methods are employed also. Most MoM processes today integrate Maximum Likelihood and extend to Generalized Method of Moments (GMM). *(good for semiparametric models, where the parameters are unknown, and therefore maximum likelihood estimation won't work – these use link functions like GLM)*

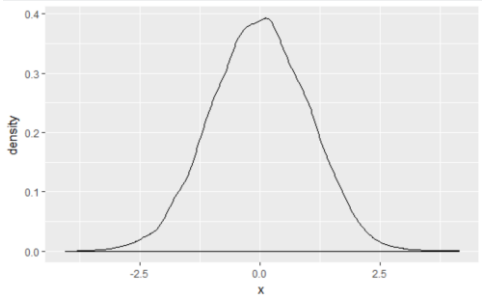
For now, just understand that MoM is a method for estimating distribution parameters that works through recursive solve algorithms, sampling and simulation.





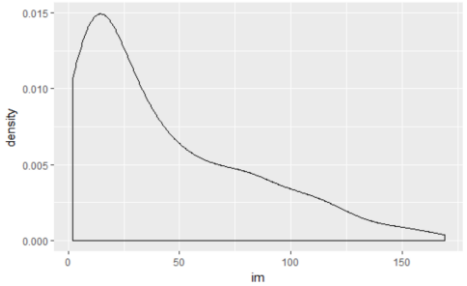

```
library(moments)
tstDist <- data.frame(x = rnorm(10000))
p <- ggplot( data = tstDist) + geom_density(mapping = aes( x = x))
P
all.moments(tstDist,order.max=4)
```

```
[2,] -0.004283687   Mean
[3,]  1.007743692   Std Deviation
[4,] -0.044426706   Skewness (N~ 0)
[5,]  3.042965642   Kurtosis 3σ⁴ , so N ~ 3
>
```




```
all.moments(im,order.max=4)
```

```
[2,] 4.347761e+01   Mean
[3,] 3.384861e+03   Std Deviation
[4,] 3.389880e+05   Skewness (N~ 0)
[5,] 3.862141e+07   Kurtosis 3σ⁴
>
```



These are shape parameters:

Skewness is a measure of symmetry. A normal distribution has a skewness of 0. skewness > |1| is “significant”.

$$a_3 = \sum \frac{(X_i - \bar{X})^3}{ns^3}$$

…the formula for skewness uses sample size as the denominator *(CLT)*.

Kurtosis is a measure of the tail extremity.

$$Kurtosis = \left\{\frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum\frac{(X_i-\bar{X})^4}{s^4}\right\} - \frac{3(n-1)^2}{(n-2)(n-3)}$$

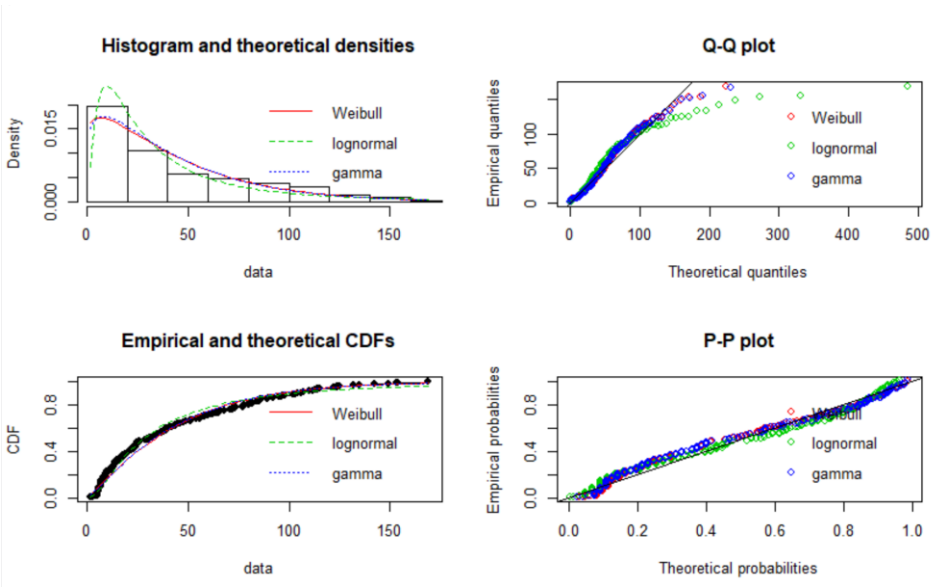
Many software packages *(incl. Excel)* will subtract 3 from kurtosis so that a 0 kurtosis is considered normal instead of 3 *(the correct quantity –statistics for dummies)*

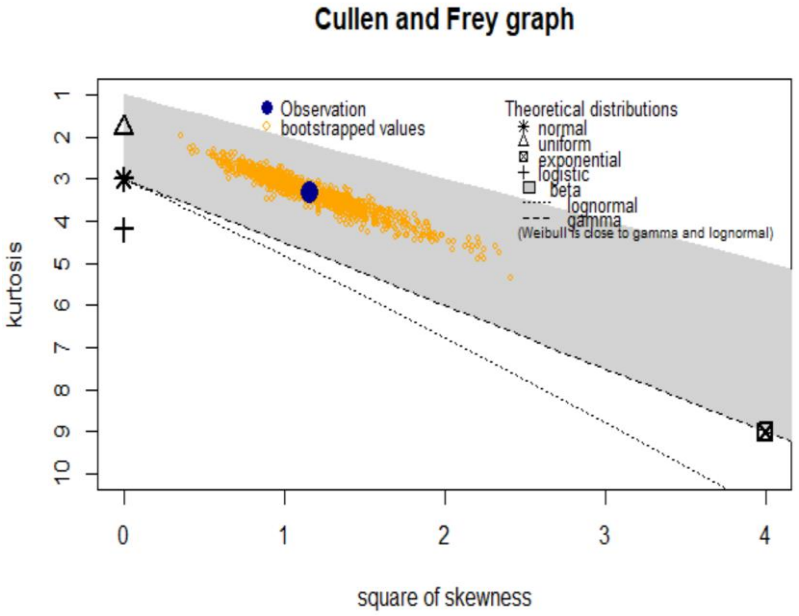
In practice, these moments are rarely used expect for supporting the decision to use a Gaussian methodology *(although there a other “proof points” that work better).*

Distribution Fitting

Fitdistrplus has a range of tools for fitting data to distributions. We're only going to use limited functions, but you should be aware of it's existence. *(see fitdistrplus.pdf on blackboard: Under the iid sample assumption, distribution parameters are by default estimated by maximizing the likelihood function)*

descdist(im$im, boot = 1000)

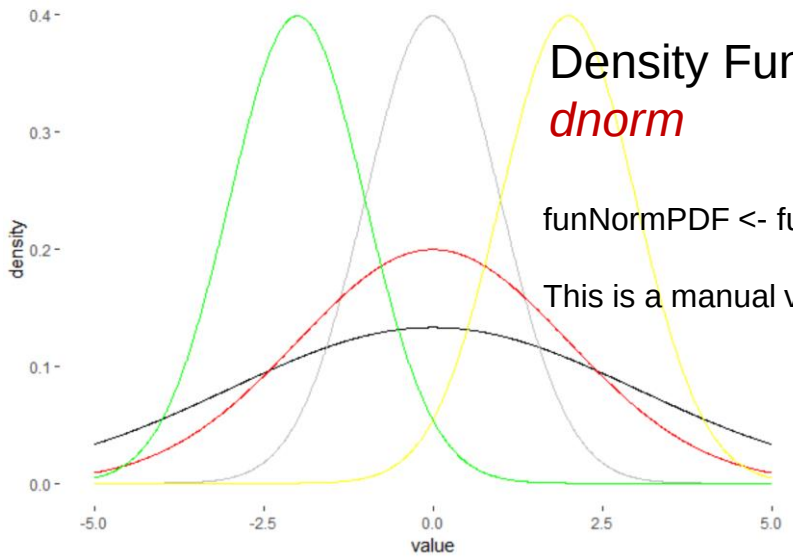




These are many tools for distribution fitting: there's a range of tests: Kolmogorov-Smirnov, Lilliefors, Shapiro-Wilk, Chi-Square, Hellinger distance, Kullback-Leibler divergence, Entropy… Information Theory gives us a range of metrics for balancing goodness of fit vs. simplicity (Occam's razor): AIC and BIC – all DA2 stuff - for now, just use the tools to fit data to distributions.

Reviewing Normal Distribution

$N(\mu, \sigma^2)$



Density Function *(PDF)*

dnorm

$$\frac{1}{\sqrt{2\pi} * \sigma} \; e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

funNormPDF <- function(x, mean, sd) {1/(sqrt(2*pi)*sd)*exp((-(xmean)^2)/(2*(sd^2)))}

This is a manual version of dnorm in r.

```
> dfDenNorm <- data.frame(x = seq(-5, 5, by = .01))
> p1 <- ggplot(dfDenNorm) +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = .5))) +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = 1)), color = "gray") +
+   geom_line(aes(x,y= dnorm(x, mean = 0, sd = 2)), color = "red") +
+   geom_line(aes(x,y= dnorm(x, mean = 2, sd = 1)), color = "yellow") +
+   geom_line(aes(x,y= dnorm(x, mean = -2, sd = 1)), color = "green") +
+   theme(panel.background = element_rect(fill = "white"))
> p1
```

moments

mean(x)
sd(x)
skewness(x)
kurtosis(x)

Normal distributions assume a skewness of 0 and a kurtosis of 3, so a truly normal distribution has only 2 parameters $N(\mu, \sigma^2)$

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Skew Normal

$SN(\lambda, \delta, \alpha)$

Recall normal PDF $\frac{1}{\sqrt{2\pi} * \sigma} - \frac{(x-\mu)^2}{e\ 2\sigma^2}$, and standardized z score $Z = \frac{X - \mu}{\sigma}$

So, we can restate the standardized normal $N(0,1)$ pdf as: $\boldsymbol{\phi}(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$, or $exp\left[\frac{\frac{-x^2}{2}}{\sqrt{2\pi}}\right]$

Note: phi $\boldsymbol{\phi}$ is used to denote a DF

We'll say that the cumulative distribution function *(CDF)* can be denoted as:

$\boldsymbol{\Phi(z) = \int_{-\infty}^{\alpha x} \boldsymbol{\phi}(x)\ dx}$,

and the random variable x is said to have a skew normal distribution if its pdf is given by:

$f_\lambda(x) = 2\ \boldsymbol{\phi}(x)\ \boldsymbol{\Phi}(\alpha x)$

where α is a shape parameter. As α parameter increases in value from 0 *(normal distribution)*, skewness increases (with a $+\alpha$ corresponding to a positive skew, and $-\alpha$ corresponding to a negative skew.

This is the basic concept of a skew normal, but we add location λ and scale δ parameters to define values. (we will typically call λ xi)

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

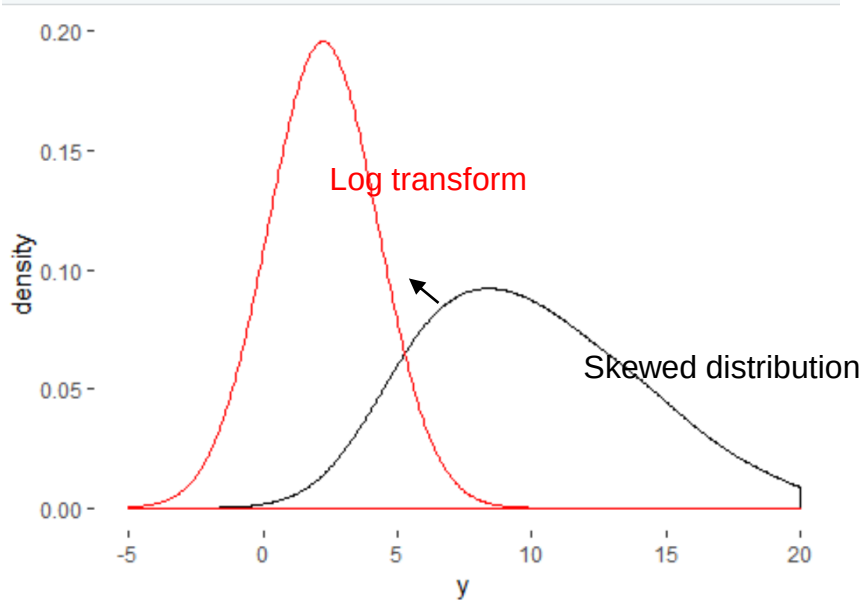
Brief review of log functions

```
> b <- 2
> n <- 3
> a <- b^n
> a
[1] 8
> c <- log(a)
> c
[1] 2.079442
> d <- exp(c)
> d
[1] 8
>
> # also, we use logs to remove exponents
> # (making our job easier with equations)
>
> c<- log(b^n)
> c
[1] 2.079442
> c2 <- n*log(b)
> c2
[1] 2.079442
> exp(c2)
[1] 8
```

Logarithms are one of the most important mathematical tools in the toolkit of statistical modeling, so you need to be very familiar with their properties and uses.

A logarithm function is defined with respect to a "base", which is a positive number: **if b denotes the base number, then the base-b logarithm of X is, by definition, the number Y such that $b^Y = X$**. For example, the base-2 logarithm of 8 is equal to 3, because $2^3 = 8$, and the base-10 logarithm of 100 is 2, because $10^2 = 100$. There are three kinds of logarithms in standard use: the base-2 logarithm (predominantly used in computer science and music theory), the base-10 logarithm (predominantly used in engineering), and the *natural* logarithm (predominantly used in mathematics and physics *and in economics and business*). In the natural log function, the base number is the transcendental number "***e***" *(like π)* whose expansion is 2.718282…, so the natural log function and the exponential function (e^x) are inverses of each other.

```
> library(sn)
> set.seed(4)
> x = seq(from = 5, to = 20, length.out = 100)
> dfSN <- data.frame(x = x, y = rsn(x, 5, 7, 9))
> dfSN$z <- log(dfSN$y)
> p1 <- ggplot(data = dfSN) +
+   geom_density(aes(y), bw = 2)  +
+   geom_density(aes(z), bw = 2, color = "red")+
+   xlim(-5, 20)+
+   theme(panel.background = element_rect(fill = "white"))
> p1
```



It's really not complicated. A log is a TRANSFORMATION, just like we use a scaling to transform values *(e.g., z values, standard deviations, standard errors, mean scaling, max-min scaling….)* .

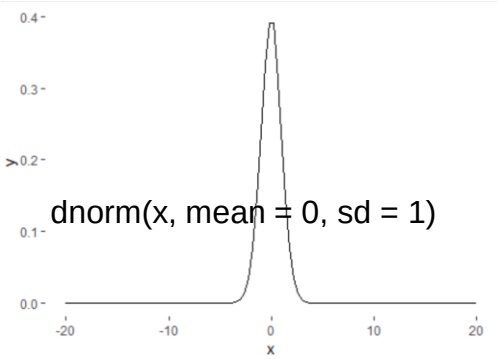
We transform data <u>often</u> in analytics. We transform entire equation outputs using a logit to yield classification log $(\frac{P(x)}{1-P(x)})$, and we transform data to infinite dimensions to apply linear algorithms to non-linear data *(all in DA2)*.We also use log transforms to visualize data that is has very large scale differences.

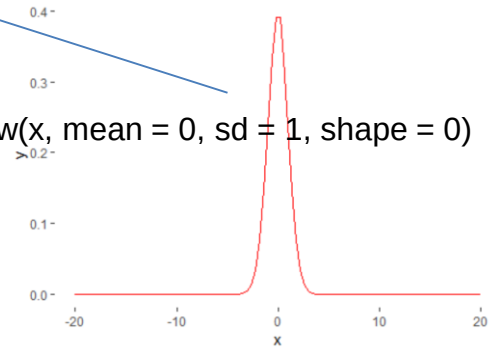
And sometimes we use logs to transform a skew distribution to a normal so we can apply theories and tools that rely on normality – e.g., simple linear regression *(there are many issues with this, but it works sometimes)*.

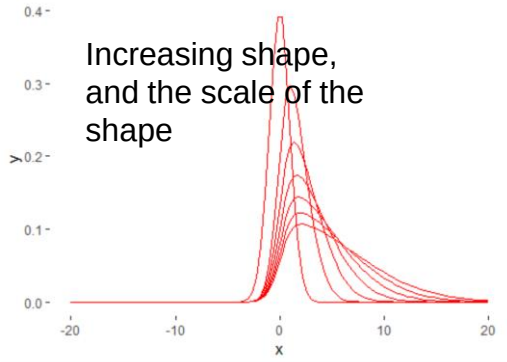
```
> skew <- function(x,e,w,a){
+   t <- (x-e)/w
+   2/w * dnorm(t) * pnorm(a*t)
+ }
>
> # e location
> # w scale
> # a shape
>
> set.seed(4)
> x = seq(from = -20, to = 20, length.out = 100)
> dfSN <- data.frame(x = x, y=dnorm(x, mean = 0, sd = 1))
> p1 <- ggplot(data = dfSN) +
+   geom_line(aes(x, y))+
+   xlim(-20, 20)+
+   theme(panel.background = element_rect(fill = "white"))
> p1
>
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 1, 0)), color = "red"); p1
>
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 2, 2)), color = "red"); p1
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 3, 3)), color = "red"); p1
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 4, 4)), color = "red"); p1
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 5, 5)), color = "red"); p1
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 6, 6)), color = "red"); p1
> p1 <- p1 + geom_line(aes(x, y = skew(x, 0, 7, 7)), color = "red"); p1
```

There is no closed form solution to sn parameters – parameter values are functions of functions that work recursively.

Notice how the scale and shape parameters work together to change the distribution. And if the shape parameter = 0, then there is no skew.





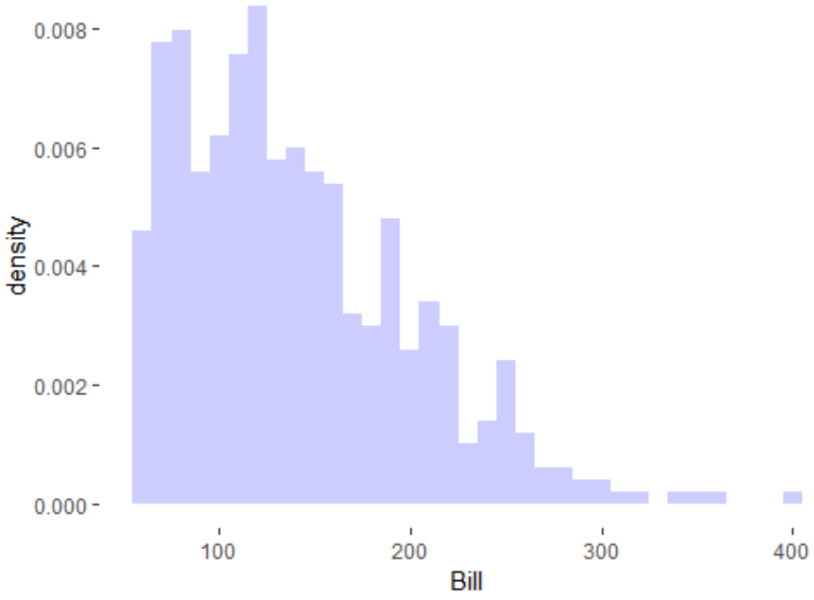


BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Exercise – Audit

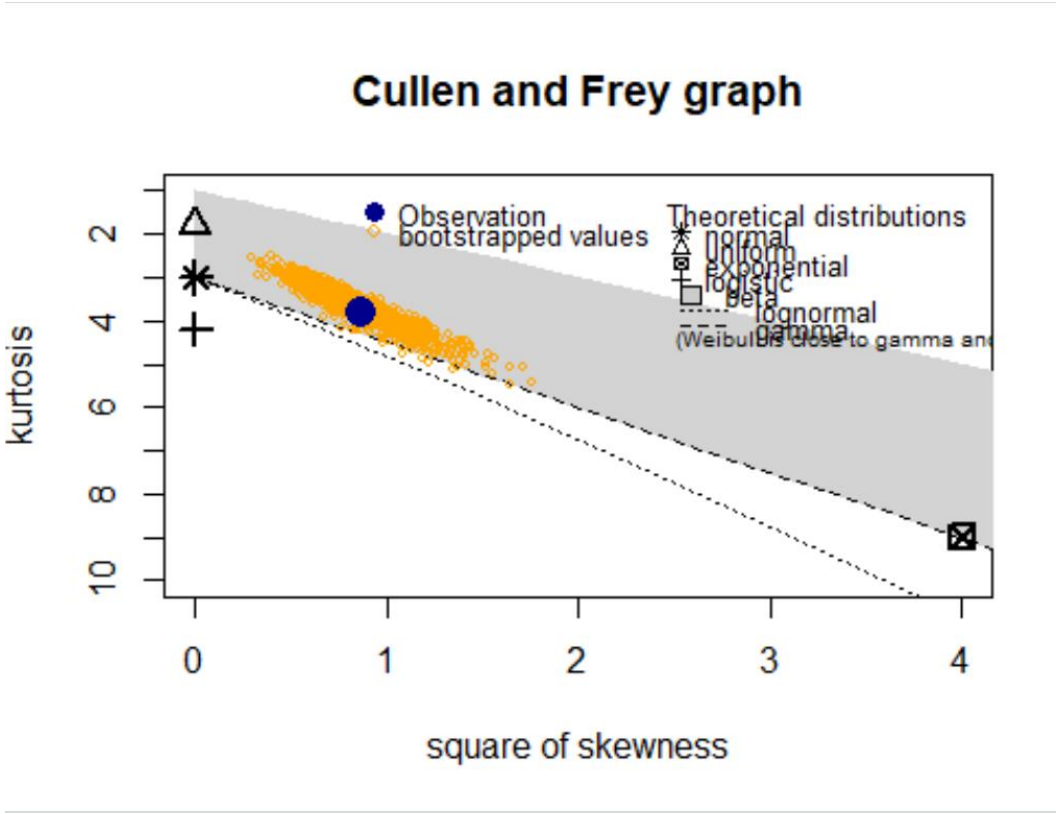
Let's walk through an example. Read in the billing data for 2012 and 2013, and plot a histogram of 2012

```
# read in 2 years of billings
Bill13 <- read_csv("sumBill13.csv")
Bill12 <- read_csv("sumBill12.csv")

# plot 2012 transactions
p2 <- ggplot(data =Bill12, aes(Bill, ..density..))  +
  geom_histogram(binwidth = 200, alpha = .2, fill = 'blue')
p2
```

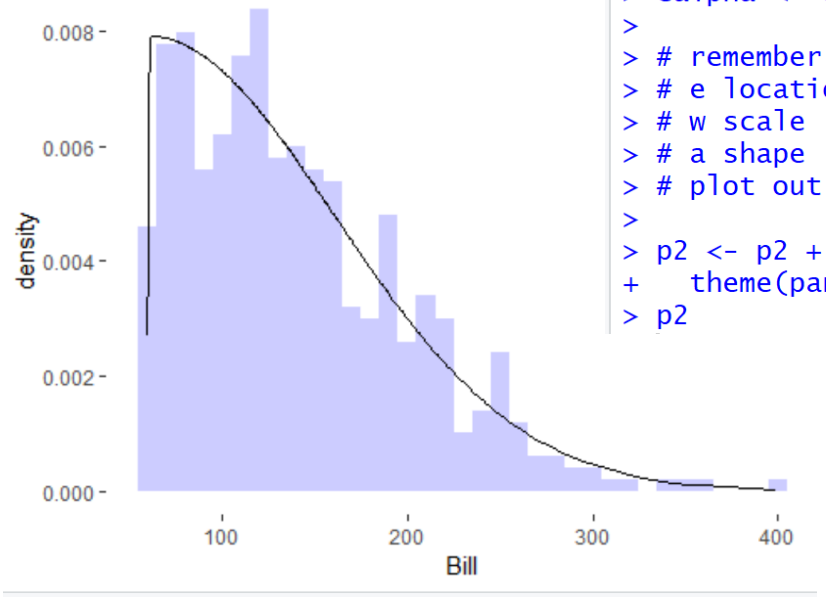


We'll run a quick CF analysis. Obviously the distribution is skewed.

Gamma or Weibull don't fit this data *(values have to be positive, and these are really designed to model product fatigue and such)*. So, we'll go with a skew normal.



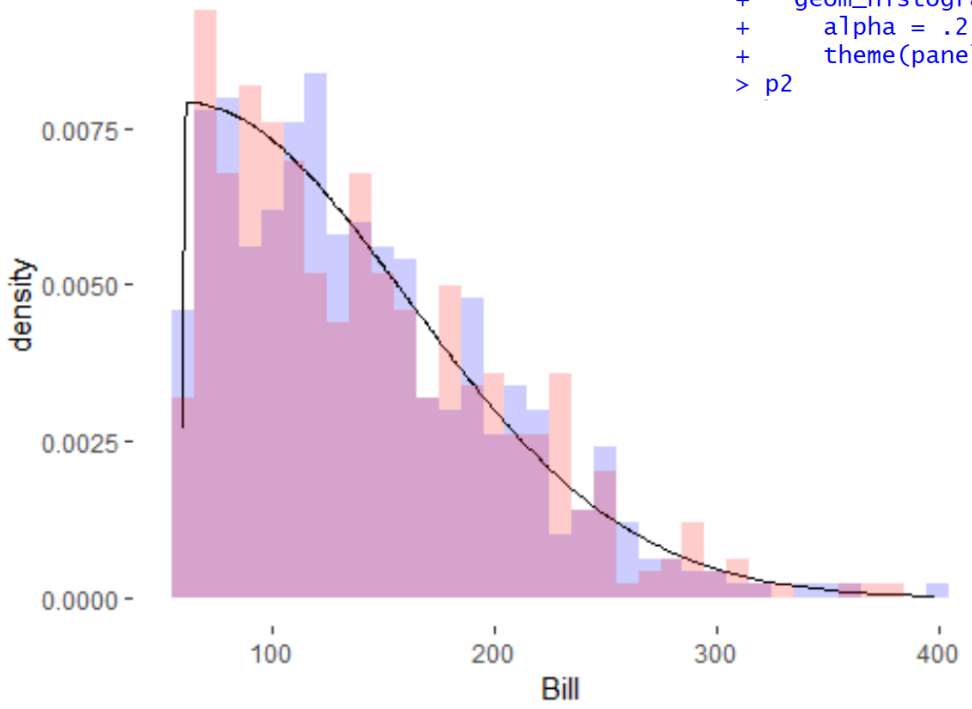
The sn package has a great maximum likelihood parameter estimate. Here, we use sn.mple to estimate the distribution, and then use cp2dp to get parameters. We then pass the parameters to our simple skew function and then we plot of the pdf values…




```
> cp.est <- sn.mple(y = Bill12$Bill, opt.method = "nlminb")$cp
> # get values
> dp.est <- cp2dp(cp.est, family = "SN")
> # put parameters in variables
>
> exi <- dp.est[1]
> eomega <- dp.est[2]
> ealpha <- dp.est[3]
>
> # remember parameter meanings!
> # e location
> # w scale
> # a shape
> # plot out the density using the skew fuction (just an estimate)
>
> p2 <- p2 + geom_line(aes(Bill, skew(Bill, exi, eomega, ealpha))) +
+   theme(panel.background = element_rect(fill = "white"))
> p2
```

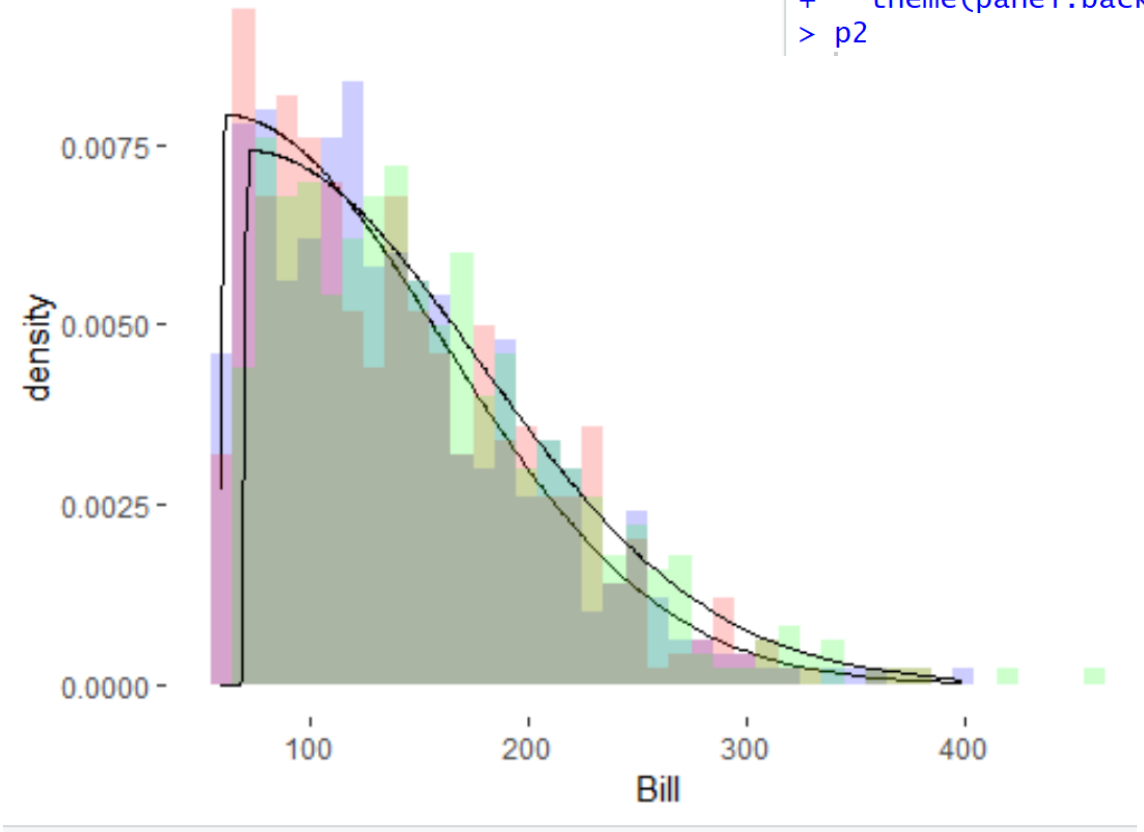
Now we use the same parameters we estimated for 2012, and we simulate 2013 using the same number of 2013 transactions with the 2012 parameters. We plot that out in red *(notice how well it fits the density outline – we would expect that in simple simulation)*

```
> Bill13$y2 <- rsn(nrow(Bill13), xi=exi , omega=eomega , alpha=ealpha )
>
> # plot and compare
> p2 <- p2 +
+   geom_histogram(data = Bill13, aes(y2, ..density..),
+     alpha = .2, binwidth = 200, fill = 'red') +
+     theme(panel.background = element_rect(fill = "white"))
> p2
```

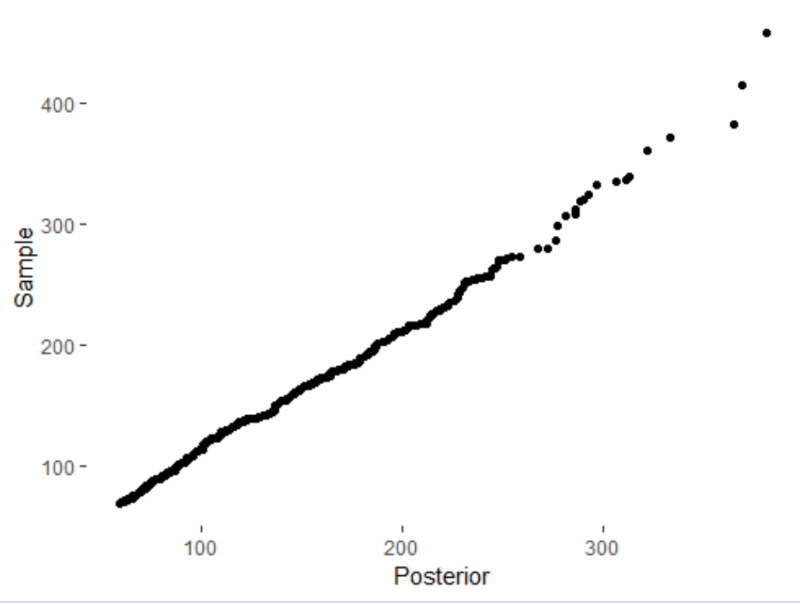


Now we plot out the actual 2013 data and compare that with the simulated. Still a pretty good fit. Let's check for outliers.

```
> p2 <- p2 +
+   geom_histogram(data = Bill13, aes(Bill, ..density..),
+     alpha = .2, binwidth = 200, fill = 'green') +
+   theme(panel.background = element_rect(fill = "white"))
> p2
```



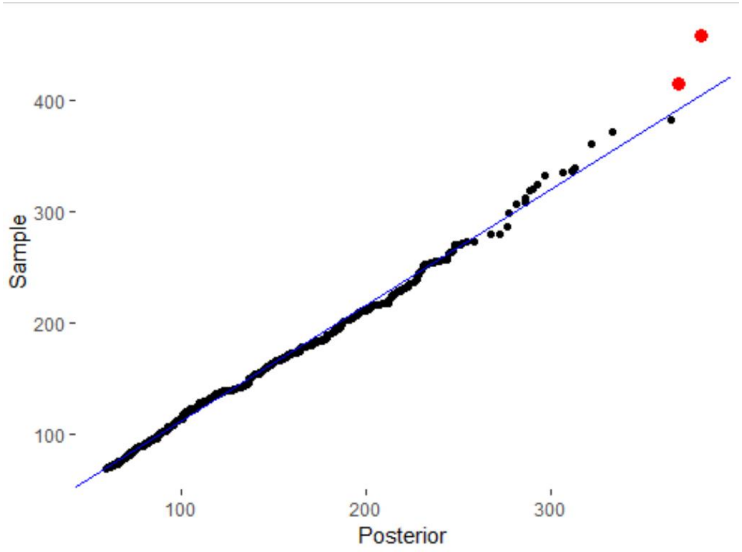
Remember the QQ? It's not just for normal distributions. We can compare any two distributions that are conjugate. In this case, we're getting a fairly good tracking between the two distributions' quantiles. Maybe a few outliners…

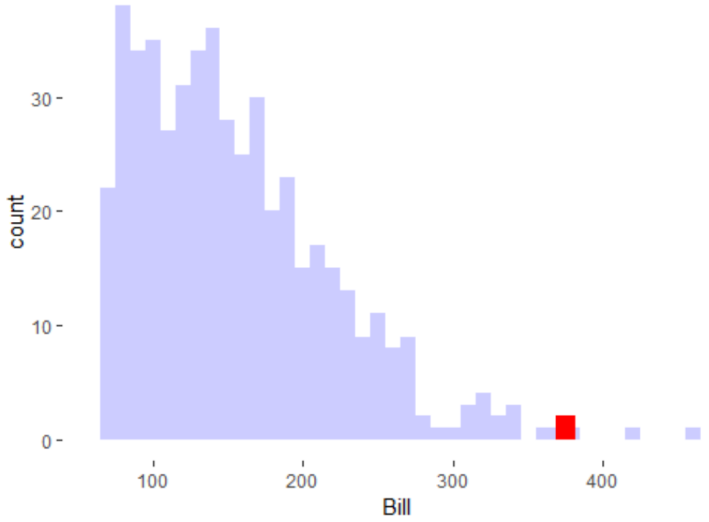



```
> tst <- data.frame(q5 = sort(Bill13$Bill), q6 = sort(Bill13$y2))
> tst <- rownames_to_column(tst, "SampleID")
>
> # then plot using a qq
> p3 <- ggplot(tst) +
+   geom_point(aes(x=q5, y=q6)) +
+   labs(x="Posterior",y="Sample") +
+   theme(panel.background = element_rect(fill = "white"))
> p3
```

BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Here, we create a liner model of the qq values, we plot it out (see the blue line). And then we apply a Bonferonni p-value test on the residuals to identify outliers. We then replot the outliers in red.





BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Data Description

- Correlation
- Principal Component Analysis
- Clustering

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Correlation

- ***Measure of association***.
- Reflects strength, not significance (*though significance can be computed for correlation coefficients)*
- Varies from 0 *(random relationship)* to 1 *(perfect positive linear relationship)* or from 0 to -1 *(perfect negative linear relationship)*.
- The Pearson correlation is usually reported in terms of its square (r^2), which is interpreted as percent of variance explained in one variable by the other. For instance, if Pearson is r^2 .25, then one variable is said to explain 25% of the variance in the other variable.
- It is a ***symmetric measure*** not itself implying causality. Therefore, for two correlated variables, the “percent explained” in one is the same as that in the other.

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Correlations *(Parametric Methods for Testing Association)*

Methods	Purpose	Assumption	Hypothesis
One sample t-test	Whether the mean of a variable is less than, greater than, or equal to a specific value. Usually, the known value is a population mean.	The dependent variable is normally distributed	Null: There is no significant difference between the sample mean and the population mean. Alternate: There is a significant difference between the sample mean and the population mean.
Paired sample t-test	Determine whether two population means are equal. It tests whether the difference between the two variables is significantly different from zero or not.	Both variables should be normally distributed.	Null: There is no significant difference between the means of the two variables. Alternate: There is a significant difference between the means of the two variables.
One-Way ANOVA	Compares the mean of k groups based on one independent variable.	1. The dependent variable is normally distributed. 2. The two groups have approximately equal variance on the dependent variable.	Null: There are no significant differences between the groups' mean scores. Alternate: There is a significant difference between the groups' mean scores.
Two-way ANOVA	Determines how a response is affected by two factors.	1. The standard deviations (SD) of the populations for all groups are equal - this is sometimes referred to as an assumption of the homogeneity of variance. 2. The samples are randomly selected from the population	The null hypothesis is that there is no interaction between columns (data sets) and rows. More precisely, the null hypothesis states that any systematic differences between columns are the same for each row and that any systematic differences between rows are the same for each column.
Pearson Regression	Test magnitude and two way direction of the ***linear association*** between two variables that are on an interval or ratio scale.	Both variables are ***normally distributed.***	Null: There is no association between the two variables. Alternate: There is an association between the two variables.
Partial Regression	Describe the linear relationship between two variables while controlling for the effects of one or more additional variables.	The Partial Correlations procedure assumes that each pair of variables is bivariate normal.	Null: There is no association between the two variables. Alternate: There is an association between the two variables.
Simple Linear Regression	Amount of variance accounted for by one variable in predicting another variable	1. The data are linear 2. The dependent variable is normally distributed.	Null: The slope equals zero (there is no slope) Alternate: The slope is not equal to zero

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Nonparametric Methods for Testing Association

Chi Square Goodness of Fit	Determines if the observed frequencies are different from what we would expect to find.	1.None of the expected values may be less than 1 2.No more than 20% of the expected values may be less than 5	Null: There are approximately equal numbers of cases in each group Alternate: There are not equal numbers of cases in each group
Chi Square Test of Independence	Determine the association between ***2 categorical variables***.	1. None of the expected values may be less than 1 2. No more than 20% of the expected values may be less than 5	Null: There is no association between the two variables. Alternate: There is an association between the two variables.
Two independent- samples test	Compares difference of two independent groups of cases on one variable.	1. Random samples from populations 2. Independence within samples and mutual independence between samples 3. Measurement scale is at least ordinal	Null: The shapes of the two groups are not significantly different. Alternate: The shapes of the two groups are significantly different.
Two dependent- samples test	Compares the distributions of two dependent variables.	The population distribution of the paired differences is assumed to be symmetric.	Null: There is no significant difference between the shapes of the two variables. Alternate: There is a significant difference between the shapes of the two variables.
Kendall's Rank Correlation	Test association for ***ordinal or ranked variables*** that take ties into account. The sign of the coefficient indicates the direction of the relationship, and its absolute value indicates the strength, with larger absolute values indicating stronger relationships.	**Both variables are NOT normally distributed.**	Null: There is no association between the two variables. Alternate: There is an association between the two variables.
Spearman Rho correlation	Test magnitude and direction of the association between two **nominal or ordinal variables** that are on an interval or ratio scale	**Both variables are NOT normally distributed.**	Null: There is no association between the two variables. Alternate: There is an association between the two variables.

Correlation - Pearson

Degree of the relationship between ***linear related variables***. ***(But note that even non-linear associations will frequently have a linear component)***

For example, in the stock market, if we want to measure how two commodities are related to each other, Pearson r correlation is used to measure the degree of relationship between the two commodities.

$$r = \frac{\sum(X - \overline{X})(Y - \overline{Y})}{\sqrt{\sum(X - \overline{X})^2}\sqrt{\sum(Y - \overline{Y})^2}}$$

think: covariance / SD
i.e., covariance standardized

There are several ***common pitfalls*** when using Pearson correlation:

- Correlation is symmetrical, not providing evidence of which way causation flows.
- If one variable is considered the independent variable and another is considered the dependent variable, then if unmeasured variables also cause the dependent variable, then any covariance they share with the hypothesized independent variable may be falsely attributed to that variable.
- To the extent that there is a nonlinear relationship between the two variables being correlated, correlation will understate the relationship.

Correlation - Spearman

Spearman rank correlation is a ***non-parametric test*** that is used to measure the degree of association between two variables. Spearman does not assume distributions data and is **a good correlation analysis when the variables are measured on a scale that is nominal or ordinal.**

The following formula is used to calculate the Spearman rank correlation:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

$$\rho = 1 - \frac{6 \times 194}{10(10^2-1)} = -.17$$

Where:
P= Spearman rank correlation
di= the *difference between the ranks* of corresponding values Xi and Yi
n= number of value in each data set

IQ, X_i	Hours of TV per week, Y_i	rank x_i	rank y_i	d_i	d_i^2
86	0	1	1	0	0
97	20	2	6	−4	16
99	28	3	8	−5	25
100	27	4	7	−3	9
101	50	5	10	−5	25
103	29	6	9	−3	9
106	7	7	3	4	16
110	17	8	5	3	9
112	6	9	2	7	49
113	12	10	4	6	36

Correlation - Chi Square

Chi-square test for independence. The test is applied when you have two **categorical variables** from a single population. It is used to determine whether there is a significant association between the two variables.

For example, in an election survey, voters might be classified by gender (male or female) and voting preference (Democrat, Republican, or Independent). We could use a chi-square test for independence to determine whether gender is related to voting preference.

When to Use Chi-Square Test for Independence

If sample data are displayed in a ***contingency table*** and the expected frequency count for each cell of the table is at least 5 *(usually more)*

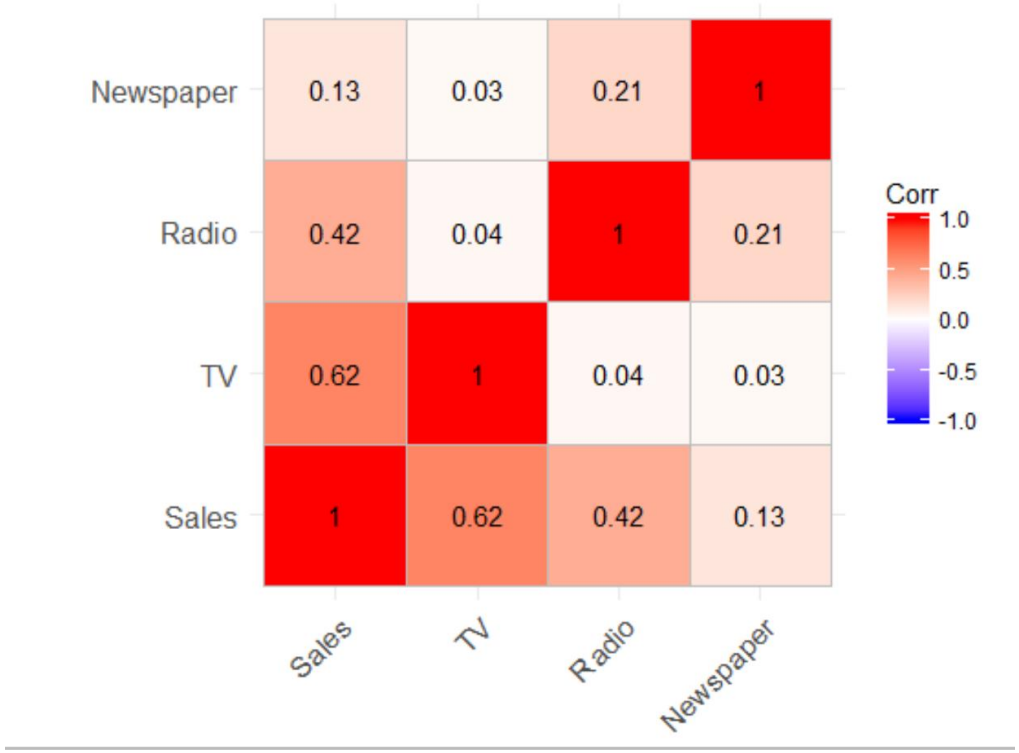
The null hypothesis is there is no relationship between the variables, so a small p value will reject H_0

CS will pick up ***non-linear relationships***

Correlation Plots

```
Advertising <- dbGetQuery(con2,"
SELECT
 [TV]
,[Radio]
,[Newspaper]
,[Sales]
FROM [dbo].[Advertising]
")
```

```
Ad <- dplyr::select(Advertising, Sales, TV,
Radio, Newspaper)

AdCorP <- cor(Ad, method="pearson")

ggcorrplot(AdCorP)
```



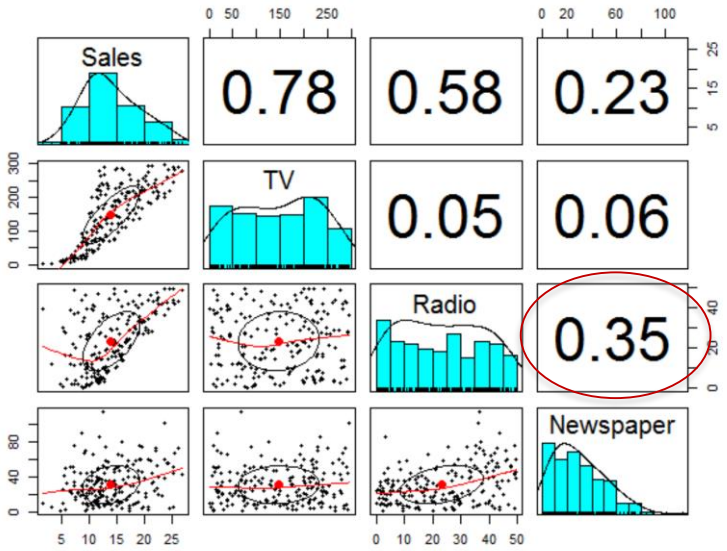
Correlation and SPLOMS

pairs.panels(Ad)

Scatter Plot Matrices (SPLOMS) are very useful for describing the data.

The **pairs.panels** function, produces:

- xy scatter plots of each pair of variables and shows the ***lowess*** locally t regression line as well. An ellipse around the mean with the axis length one standard deviation of the x and y variables is also drawn. The x axis in each scatter plot represents the column variable, the y axis the row variable.
- ***histogram*** with density of each variable on the diagonal,
- Correlation (compare to correlation matrix – previous slide.

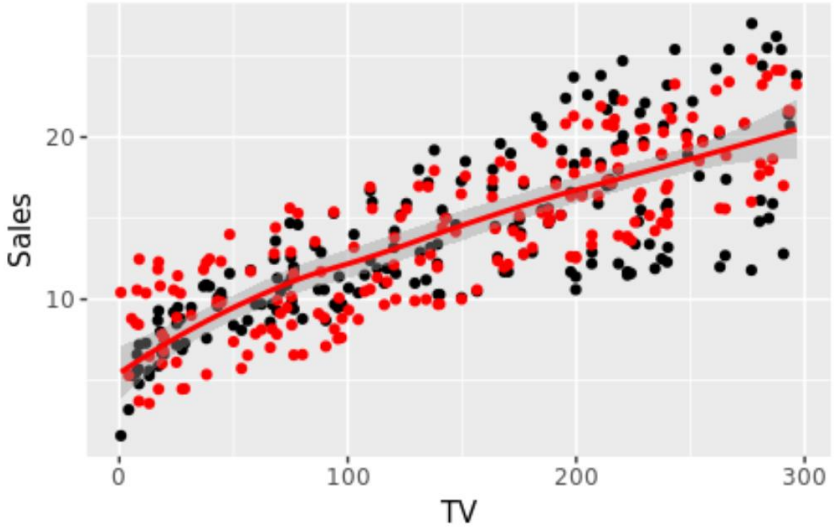


The correlation section of your presentation should say something like :

- Overall, the correlation between the explanatory and response variables appears to be significant,
- Correlation between the explanatory variable appears weak, which indicates independence. Distributions are somewhat normal with some variables,
- Some degree of homoscedasticity is present.

Conclusion: a good candidate for regression.

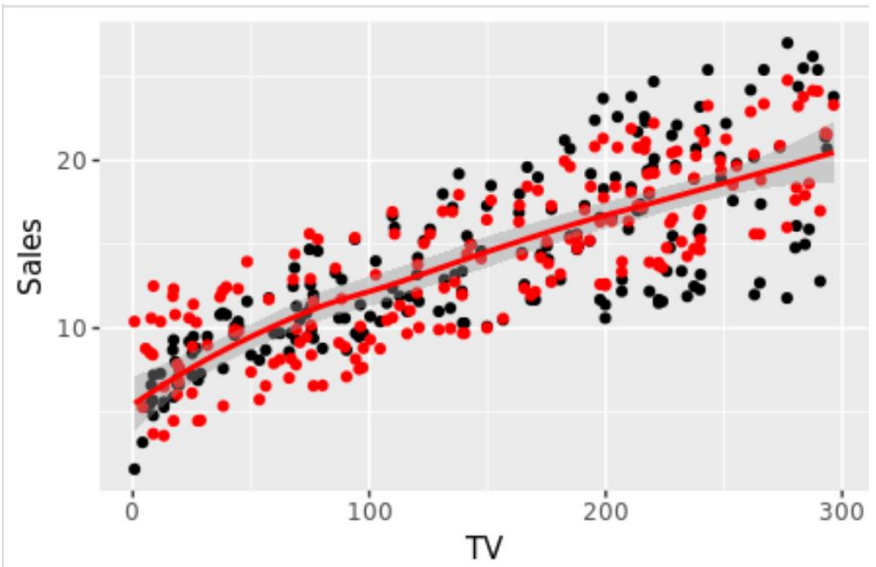
```
> lMod <- lm(Sales ~ TV+Radio+Newspaper, data = Ad)
> summary(lMod)

Call:
lm(formula = Sales ~ TV + Radio + Newspaper, data = Ad)

Residuals:
    Min      1Q  Median      3Q     Max
-8.8277 -0.8908  0.2418  1.1893  2.8292

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
Radio        0.188530   0.008611  21.893   <2e-16 ***
Newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.686 on 196 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8956
F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```




```
> cor(Ad, method="pearson", use="pairwise")
               Sales         TV      Radio  Newspaper predSales
Sales      1.0000000 0.78222442 0.57622257 0.22829903 0.9472120
TV         0.7822244 1.00000000 0.05480866 0.05664787 0.8258177
Radio      0.5762226 0.05480866 1.00000000 0.35410375 0.6083354
Newspaper 0.2282990 0.05664787 0.35410375 1.00000000 0.2410221
predSales 0.9472120 0.82581766 0.60833536 0.24102209 1.0000000
```

Taking out Newspaper

```
> lMod <- lm(Sales ~ TV+Radio, data = Ad)
> summary(lMod)

Call:
lm(formula = Sales ~ TV + Radio, data = Ad)

Residuals:
    Min      1Q  Median      3Q     Max
-8.7977 -0.8752  0.2422  1.1708  2.8328

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.92110    0.29449   9.919   <2e-16 ***
TV           0.04575    0.00139  32.909   <2e-16 ***
Radio        0.18799    0.00804  23.382   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1

Residual standard error: 1.681 on 197 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8962
F-statistic: 859.6 on 2 and 197 DF,  p-value: < 2.2e-16
```




```
> cor(Ad, method="pearson", use="pairwise")
              Sales         TV      Radio predSales
Sales     1.0000000 0.78222442 0.57622257 0.9472034
TV        0.7822244 1.00000000 0.05480866 0.8258252
Radio     0.5762226 0.05480866 1.00000000 0.6083409
predSales 0.9472034 0.82582520 0.60834091 1.0000000
```

Notice that we took out Newspaper, yet our prediction was just as accurate

Correlation - Chi Square

```
#
# Chi Square - for categorical data
#
# get data from survey of students
Popular <- read.csv(file="StudentSurvey.csv", header=TRUE, sep=",")
PopTbl <- table(Popular$District, Popular$Priority)
PopTbl
chisq.test(PopTbl)
# is this significant at the .05 level?
```

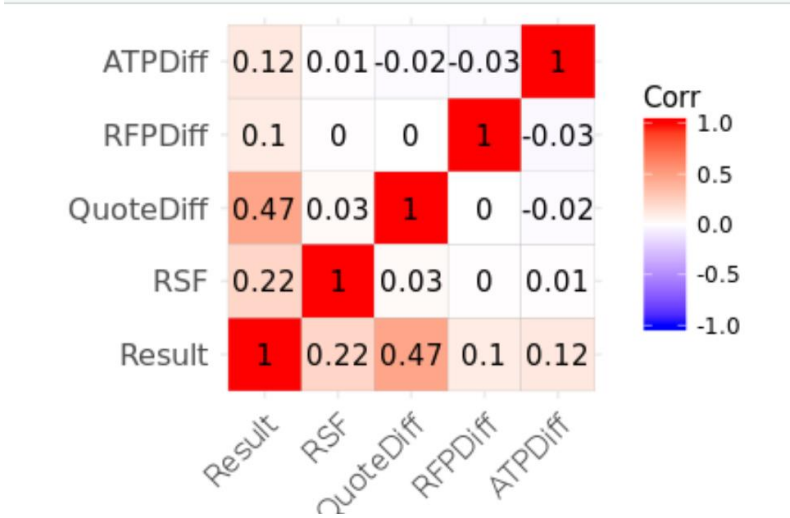
```
          Grades Popular Sports
  Rural       57      50     42
  Suburban    87      42     22
  Urban       24       6      5
> chisq.test(PopTbl)

        Pearson's Chi-squared test

data:  PopTbl
X-squared = 18.564, df = 4, p-value = 0.000957

> # is this significant at the .05 level?
>
```

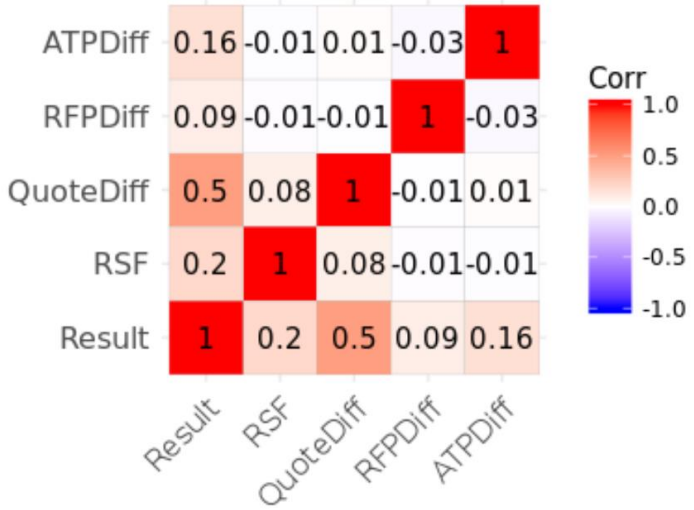
Note: For the test of independence, also known as the test of homogeneity, a chi-squared probability of ***less than or equal to 0.05*** (or the chi-squared statistic being at or larger than the 0.05 critical point) is commonly interpreted as ***justification for rejecting the null hypothesis*** (independence between the response and explanatory variables – i.e., there's nothing going on here). So a low score means there's a significant relationship.

One more Example - Correlation

See R file for connections and SQL Statements

```
QuoteCor <- dplyr::select(SampleData, RSF, QuoteDiff,RFPDiff, ATPDiff, Result)
mQuoteCor <- data.matrix(QuoteCor)
QuoteCorP <- cor(mQuoteCor, method="pearson")
QuoteCorS <- cor(mQuoteCor, method="spearman")
QuoteCorK <- cor(mQuoteCor, method="kendall")

ggcorrplot(QuoteCorP)
ggcorrplot(QuoteCorS)
ggcorrplot(QuoteCorK, lab = TRUE)
```





BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

```
# ok, now chi square on this data


Quote <- dplyr::select(SampleData, Quote_ID, RSF, Result)

Quote$RSF <- as.character(Quote$RSF)
Quote$Result <- as.character(Quote$Result)

Quote %>% distinct(Result)
Quote$Result[Quote$Result=="l"] <- "L" # fix a data error
# Just for visual:
Quote$RSF[Quote$RSF=="1"] <- "1 - None"
Quote$RSF[Quote$RSF=="2"] <- "2 - Developing"
Quote$RSF[Quote$RSF=="3"] <- "3 - Established"
Quote$RSF[Quote$RSF=="4"] <- "4 - Strong"

tblQuote <- table(Quote$RSF, Quote$Result)
tblQuote

#tblQuote <- tblQuote[,2:3]
#tblQuote
chisq.test(tblQuote)
```

So, we suspect that there are stronger relationships here than the linear correlation tests revealed.

```
> chisq.test(tblQuote)

        Pearson's Chi-squared test

data:  tblQuote
X-squared = 38.358, df = 3, p-value = 2.374e-08
```

Test for RFP

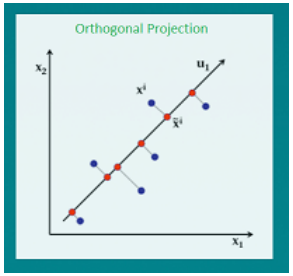
```
                  L   W
  1 - On Time   266 409
  2 - Late       50  24
  3 - Very Late  28  15
> chisq.test(tblQuote)

        Pearson's Chi-squared test

data:  tblQuote
X-squared = 30.225, df = 2, p-value = 2.734e-07
```

Test for ATP

```
                  L   W
  1 - On Time   241 390
  2 - Late       44  40
  3 - Very Late  59  18
> chisq.test(tblQuote)

        Pearson's Chi-squared test

data:  tblQuote
X-squared = 44.313, df = 2, p-value = 2.385e-10
```

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Primary Components Analysis



Principal component analysis (PCA) is a statistical procedure that uses an ***orthogonal transformation*** to convert a set of observations of possibly correlated variables into a set of values ***of linearly uncorrelated variables called principal components***. The number of principal components is less than or equal to the number of original variables.

This transformation is defined in such a way that the ***first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components***. The resulting vectors are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables.

PCA is ***mostly used as a tool in exploratory data analysis*** and for making predictive models. PCA can be done by ***eigenvalue decomposition*** of a data covariance (or correlation) matrix or ***singular value decomposition*** of a data matrix, ***usually after mean centering (and normalizing or using Z-scores)*** the data matrix for each attribute.

The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score).[5]

PCA is the simplest of the true eigenvector-based multivariate analyses. Often, its operation can be thought of as revealing the internal structure of the data in a way that best explains the variance in the data. If a multivariate dataset is visualized as a set of coordinates in a high-dimensional data space (1 axis per variable), PCA can supply the user with a lower-dimensional picture, a projection or "shadow" of this object when viewed from its (in some sense; see below) most informative viewpoint. This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced.

PCA is closely related to factor analysis. Factor analysis typically incorporates more domain specific assumptions about the underlying structure and solves eigenvectors of a slightly different matrix.

PCA is also related to canonical correlation analysis (CCA). CCA defines coordinate systems that optimally describe the cross-covariance between two datasets while PCA defines a new orthogonal coordinate system that optimally describes variance in a single dataset.

dimension reduction technique

Eigenvectors & Eigenvalues *(repeat from last section)*

Originally utilized to study principal axes of the rotational motion of rigid bodies, eigenvalues and eigenvectors have a wide range of applications, for example in stability analysis, vibration analysis, atomic orbitals, facial recognition, and matrix diagonalization. In essence, an eigenvector v of a linear transformation T is a non-zero vector that, when T is applied to it, does not change direction. Applying T to the eigenvector only scales the eigenvector by the scalar value λ, called an eigenvalue. This condition can be written as the equation:

$Av = \lambda v$, or $(A- \lambda I) = 0$
if det($(A- \lambda I) = 0$

λ is a **scalar** ***eigenvalue*** associated with an ***eigenvector*** v that can be used for ***transformation*** of a matrix.
Eigenvalues are:

- Non-Zero
- nxn *(square)* matrix only (matrix is diagonalizable)
- From a geometrical perspective, does not change the direction of a vector A (next slide)
- There always exists at least one eigenvalue / eigenvector

When eigenvectors are applied to linear transformation, the matrix just gets scaled, and the transformation still tells us what we need to know about the original matrix

It breaks down the linear transformation into simple operations.
To solve for eigenvalues:

1. ***Compute the determinant of*** $A - \lambda I$. With λ subtracted along the diagonal, this determinant starts with λ^n or $-\lambda^n$. It is a polynomial in λ of degree n.
2. ***Find the roots of this polynomial,*** by solving $\det(A - \lambda I) = 0$. The n roots are the n eigenvalues of A. They make $A - \lambda I$ singular.
3. For each eigenvalue λ, ***solve*** $(A - \lambda I)x = 0$ ***to find an eigenvector*** x.

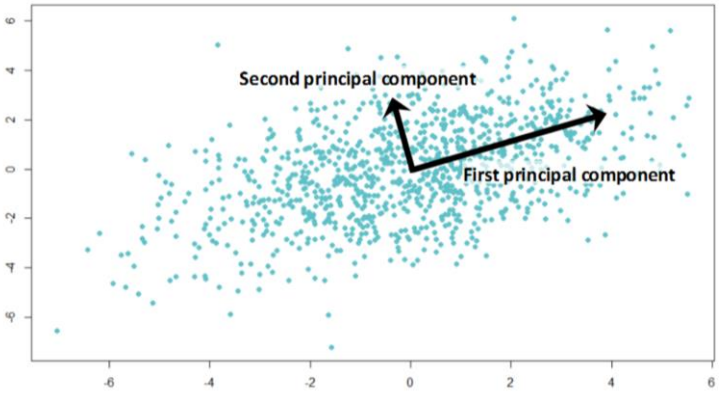
BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

PCA

Second principal component (Z^2) is also a linear combination of original predictors which captures the remaining variance in the data set and is uncorrelated with Z^1. In other words, the correlation between first and second component should is zero. It can be represented as:

$$Z^2 = \Phi^{12}X^1 + \Phi^{22}X^2 + \Phi^{32}X^3 + ... + \Phi^{p2}X^p$$

If the two components are uncorrelated, their directions should be orthogonal (image below). This image is based on a simulated data with 2 predictors. Notice the direction of the components, as expected they are orthogonal. This suggests the correlation b/w these components in zero



BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Scaling (covariance vs. correlation)

Scaling in PCA using the Correlation matrix to scale (divides by SD)

The argument against automatically using correlation matrices is that it is quite a brutal way of standardizing your data. The problem with automatically using the covariance matrix, is that the variables with the highest variance will dominate the first principal component (the variance maximizing property).

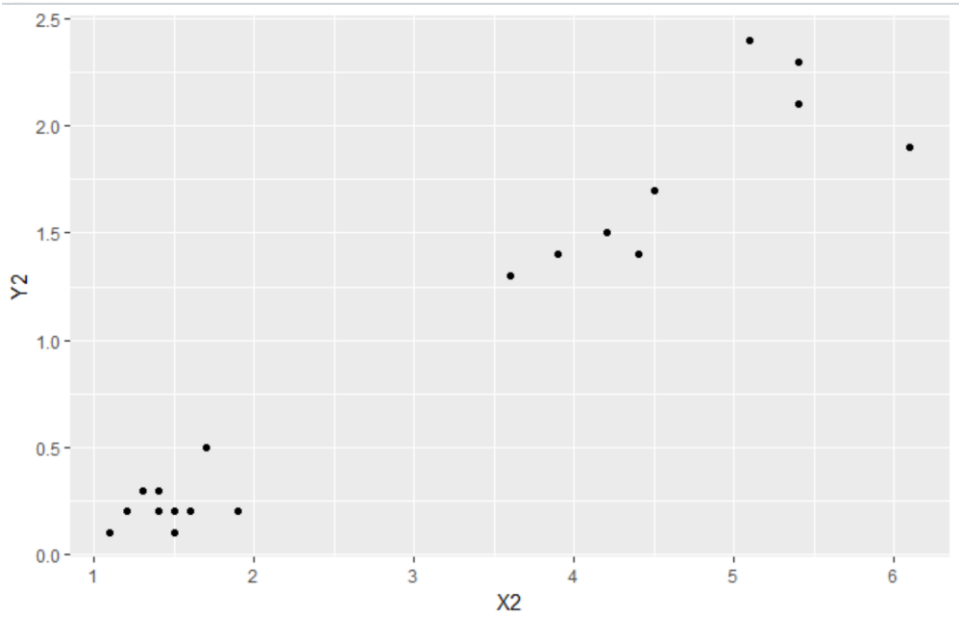
An example might be applying PCA to term structure analysis on bond yields in finance. Variances of yields on varied maturities vary, but since they are all yields, the varying scales are normally not unacceptably wide. Indeed, more/less volatility of certain maturity yield itself provides rich information. PCA is used extensively in derivative trading algorithms.

An example where scaling is necessary is in the sales data coming up – you have ordinal relationship data (1 to 4) and quote differences (-20,000 to 20,000+).

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

PCA – how it works

```
rawdata <- read.csv(file="PCAData.csv", header=TRUE,
sep=",")
#testInd <- sample(1:150,10)
testInd <- sample(1:150,20)
pcadata <- rawdata[testInd, 3:4]

#get the data and subtract the means
XMean <- sum(pcadata$X)/nrow(pcadata)
YMean <- sum(pcadata$Y)/nrow(pcadata)

pcadata$VarX <- round((pcadata$X - XMean),2)
pcadata$VarY <- round((pcadata$Y - YMean),2)

p <- ggplot(pcadata, aes(x=X2,
y=Y2))+geom_point(color="black")
p
```



BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

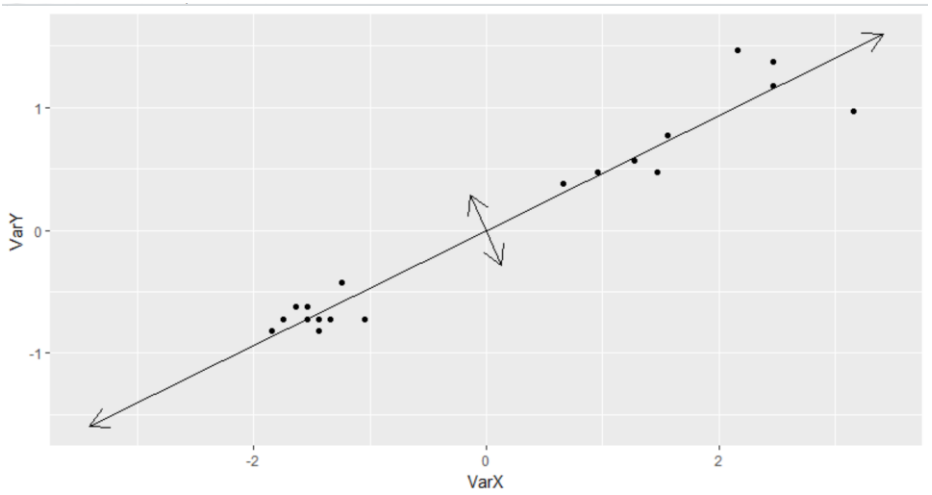
PCA – how it works

```
# We can build covariances specifically
pcadata$covXY <- round(pcadata$VarX * pcadata$VarY,2)
totXY <- sum(pcadata$covXY)
covXY <- mean(pcadata$covXY)
pcaCovXY <- cov(pcadata$X, pcadata$Y)

#or just call a cov function
pcaCov <- data.frame(cov(pcadata[1:2]))

# note that both are positive, so we expect X and Y to increase together

pcaEigen <- eigen(pcaCov)
dfPcaEigen <- data.frame(values=pcaEigen$values,pcaEigen$vectors)

VarMatrix <- pcadata[,3:4]
write.csv(VarMatrix, file = "VarMatrix.csv")
EigenMatrix <- t(dfPcaEigen[1,2:3])
write.csv(EigenMatrix, file = "EigenMatrix.csv")

newMatrix <- VarMatrix*EigenMatrix #no-not yet

# write the covariance matrix for further analysis
write.csv(pcaCov, file = "PCA Ex 1 Output.csv")

#now let's get the eigens of te covariance matrix
pcaEigen <- eigen(pcaCov[1:2, 1:2])
pcaEigen <- data.frame(values=pcaEigen$values,pcaEigen$vectors)


write.csv(pcaEigen, file = "PCAEigen.csv")
```

PCA – how it works

```
# get the 2 eigenvectors
V1<-data.frame(X=c(0, pcaEigen[1,2]),Y=c(0, pcaEigen[2,2]))
V1<- rbind(V1, c(V1[2,1]*pcaEigen[1,1],V1[2,2]*pcaEigen[1,1]))
V2<-data.frame(X=c(0, pcaEigen[1,3]),Y=c(0, pcaEigen[2,3]))
V2<- rbind(V2, c(V2[2,1]*pcaEigen[2,1],V2[2,2]*pcaEigen[2,1]))

#plot them out
pcaDataT <- t(pcadata)
write.csv(pcaDataT, file = "PCADataT.csv")

p <- ggplot(pcadata, aes(x=VarX,
y=VarY))+geom_point(color="black")
p <- p + geom_segment(aes(x = V1[1,1], y = V1[1,2], xend =
V1[3,1], yend =V1[3,2] ), arrow = arrow(length = unit(0.5, "cm")))
p <- p + geom_segment(aes(x = V1[1,1], y = V1[1,2], xend = -
1*V1[3,1], yend = -1*V1[3,2] ), arrow = arrow(length = unit(0.5,
"cm")))
p <- p + geom_segment(aes(x = V2[1,1], y = V2[1,2], xend =
10*V2[3,1], yend = 10*V2[3,2] ), arrow = arrow(length = unit(0.5,
"cm")))
p <- p + geom_segment(aes(x = V2[1,1], y = V2[1,2], xend = -
10*V2[3,1], yend = -10*V2[3,2] ), arrow = arrow(length =
unit(0.5, "cm")))
p
```

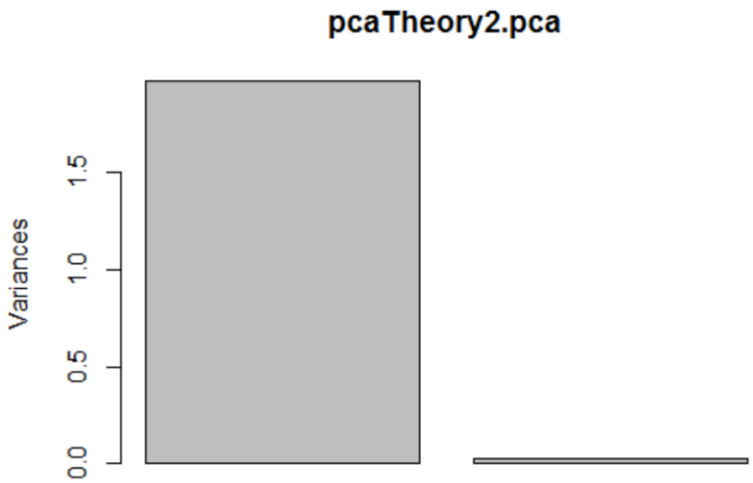
Note - added coefficients and opposite direction eignvectors to improve visibility



- PCA uses linear combination of the covariance matrix
- The First PC defines the most variance
- The Second PC defines the next… on and on
- N dimension Matrix – N PCs

The easy way:

```
pcaTheory2  <- rawdata[testInd, 3:4]
pcaTheory2.pca <- prcomp(pcaTheory2, retx=TRUE, center=TRUE,
scale.=TRUE)
pcaTheory2.pca
plot(pcaTheory2.pca)
summary(pcaTheory2.pca)
```




```
Rotation:
           PC1        PC2
X2 -0.7071068  0.7071068
Y2 -0.7071068 -0.7071068
> plot(pcaTheory2.pca)
> summary(pcaTheory2.pca)
Importance of components:
                          PC1     PC2
Standard deviation     1.4045 0.16586
Proportion of Variance 0.9862 0.01376
Cumulative Proportion  0.9862 1.00000
>
```

PCA Dimension Analysis

```
Quote1.pca <- prcomp(Quote1[-5], retx=TRUE, center=TRUE,
scale.=TRUE)
Quote1.pca
plot(Quote1.pca)

# really simple plot of PCs
summary(Quote1.pca)
```

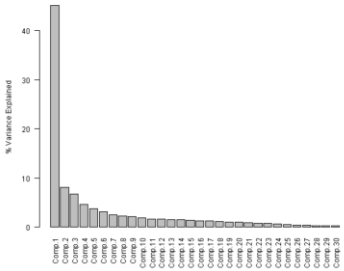
Note: prcomp works a little differently than the theory example:

1. **It scales the data**
2. **It rotates and 'zeros' the axis**

Generally, you should *consider* keeping Eigenvalues > 1 *(Kaiser rule)*

This tells you that there's not going to be a good opportunity for dimension reduction – i.e., PC's all contribute to variance.

A graph that shows opportunity to reduce dimensions looks more like this:

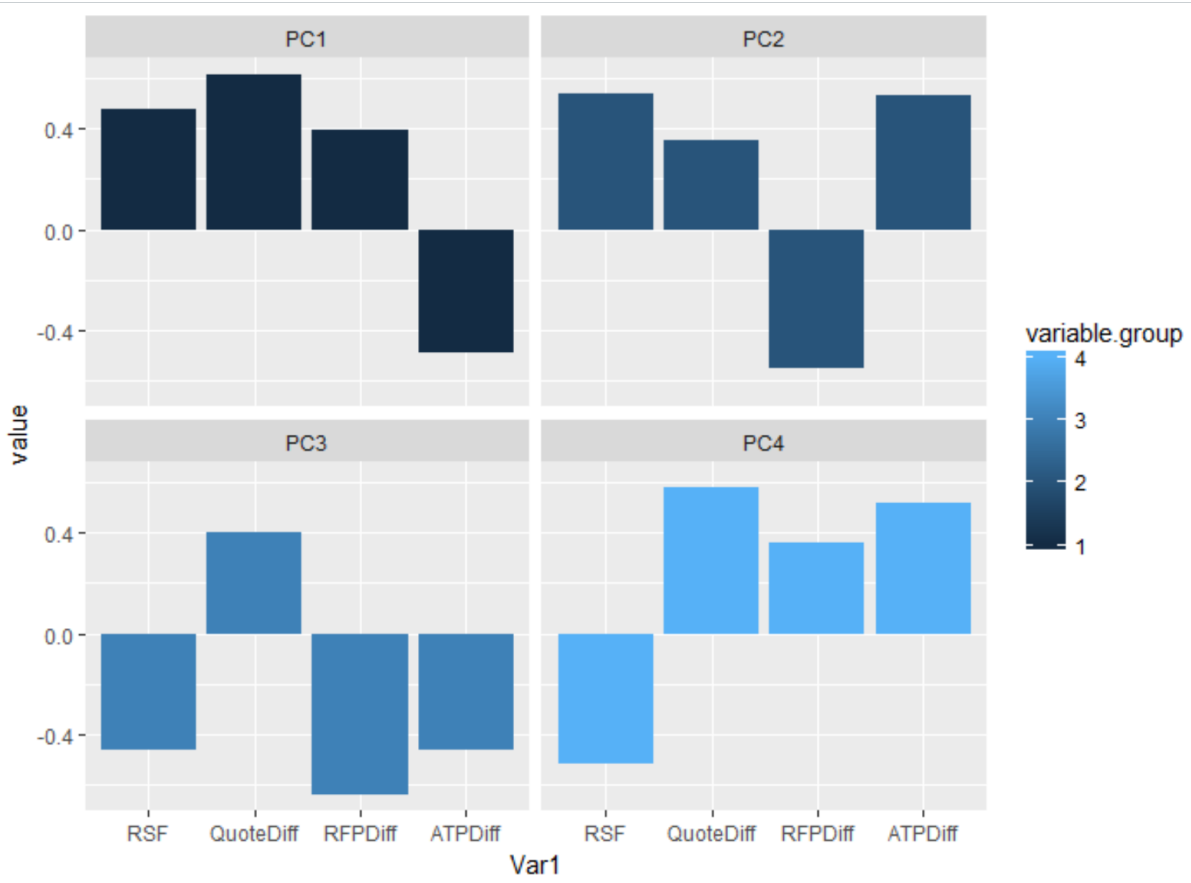


Generally, you should *consider* keeping variables with a factor loading > |.25|

```
Rotation:
                PC1        PC2        PC3        PC4
RSF        0.4768005  0.5405178 -0.4639906 -0.5149898
QuoteDiff  0.6153318  0.3534156  0.4009154  0.5794231
RFPDiff    0.3938005 -0.5522994 -0.6397793  0.3613433
ATPDiff   -0.4888244  0.5271653 -0.4633164  0.5181558
> plot(Quote1.pca)
> plot(Quote1.pca)
>
> # really simple plot of PCs
> summary(Quote1.pca)
Importance of components:
                          PC1    PC2    PC3    PC4
Standard deviation     1.0191 1.0111 0.9913 0.9780
Proportion of Variance 0.2597 0.2556 0.2457 0.2391
Cumulative Proportion  0.2597 0.5152 0.7609 1.0000
>
```

another view

```
variable.group <- c(rep(1, 4), rep(2, 4), rep(3, 4),
                    rep(4, 4))

melted <- cbind(variable.group,
melt(Quote1.pca$rotation[,1:4]))
Quote1.pca$rotation

barplot <- ggplot(data=melted) +
  geom_bar(aes(x=Var1, y=value, fill=variable.group),
stat="identity") +
  facet_wrap(~Var2)
barplot
```

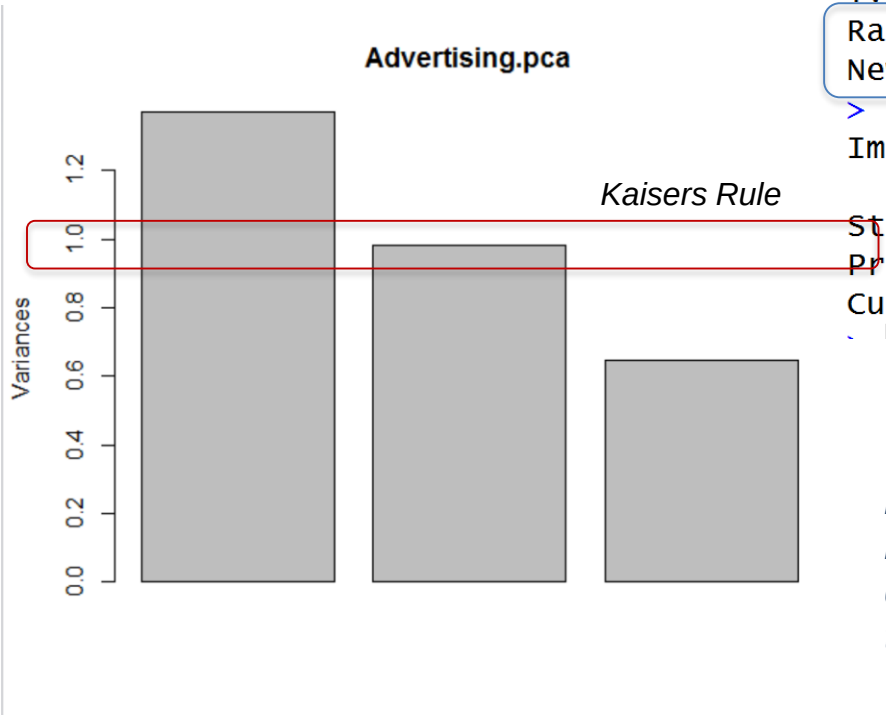


BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Another Example – PCA

```
# Advertising Data

myServer <- "tcp:analytics1.database.windows.net,1433"
myUser <- "Student"
myPassword <- "Acct7397"
myDatabase <- "Accounting"
myDriver <- "ODBC Driver 13 for SQL Server" # Must correspond to an entry in the Drivers tab of "ODBC Data Sources"

connectionString <- str_c(
  "Driver=", myDriver,
  ";Server=", myServer,
  ";Database=", myDatabase,
  ";Uid=", myUser,
  ";Pwd=", myPassword)

sq <- function (myQuery){
  conn <- odbcDriverConnect(connectionString)
  tQuery <- (sqlQuery(conn, myQuery))
  close(conn)
  return (tQuery)
}




myQuery <- "
SELECT
[Obs]
,[TV]
,[Radio]
,[Newspaper]
,[Sales]
FROM [dbo].[Advertising]
"
Advertising <- sq(myQuery)
```

```
Advertising.pca <- prcomp(Advertising[2:4], retx=TRUE, center=TRUE,
scale.=TRUE)
Advertising.pca
plot(Advertising.pca)
summary(Advertising.pca)
```

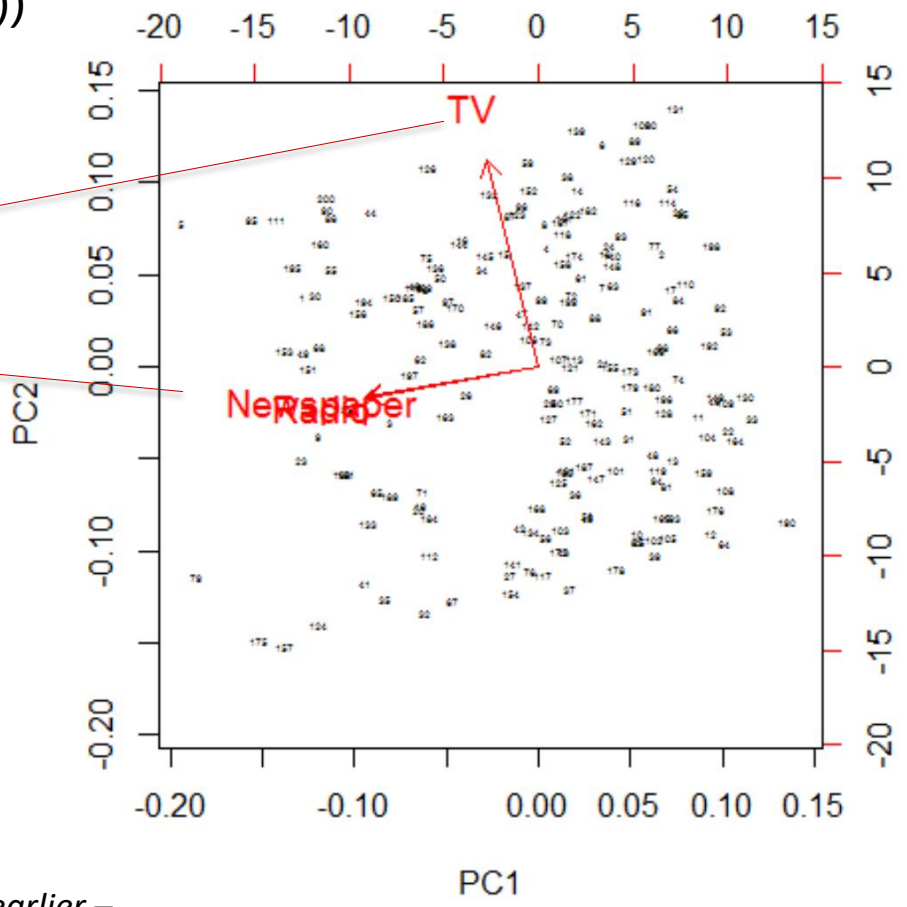



```
Rotation:
                PC1        PC2         PC3
TV        -0.2078739  0.9781484  0.003765898
Radio     -0.6913967 -0.1496553  0.706805372
Newspaper -0.6919241 -0.1443227 -0.707398038
> summary(Advertising.pca)
Importance of components:
                          PC1    PC2    PC3
Standard deviation     1.171 0.9916 0.8037
Proportion of Variance 0.457 0.3277 0.2153
Cumulative Proportion  0.457 0.7847 1.0000
```

The eigen values here are pointing to significant interdependence which can cause problems in linear regression. Also, you typically are looking to reduce dimensions and this indicates an opportunity to do that.

biplot(Advertising.pca, cex=c(.3,1.2))

```
Rotation:
                PC1        PC2         PC3
TV        -0.2078739  0.9781484  0.003765898
Radio     -0.6913967 -0.1496553  0.706805372
Newspaper -0.6919241 -0.1443227 -0.707398038
> summary(Advertising.pca)
Importance of components:
                          PC1    PC2    PC3
Standard deviation     1.171 0.9916 0.8037
Proportion of Variance 0.457 0.3277 0.2153
Cumulative Proportion  0.457 0.7847 1.0000
```



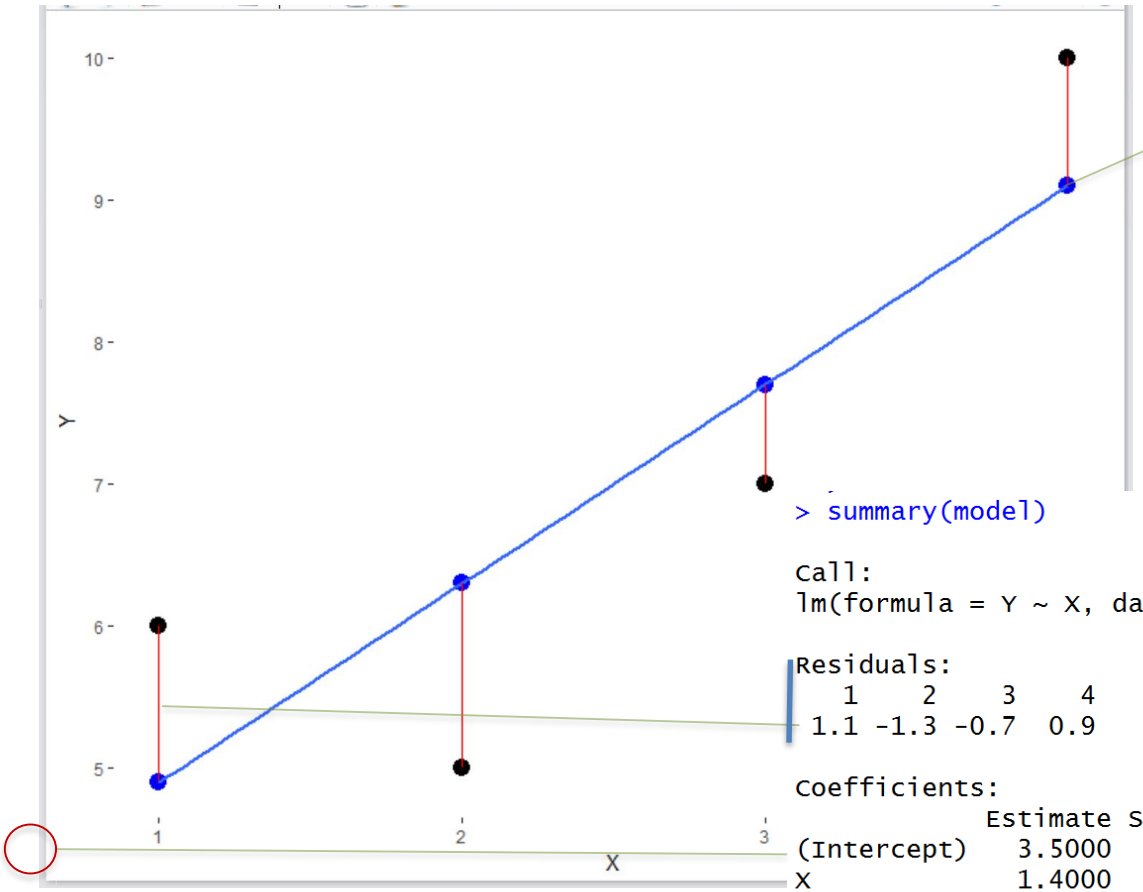
So this brings us to the same conclusion we arrived out earlier – the Newspaper and Radio dimensions are surrogates

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Regression

- Regression Diagnostics
- Confidence Intervals
- Regression Topics
  - Multiple (or multivariate) Regression
  - Categorical Variables
  - Polynomial
  - Interaction

Regression Diagnostics




```
> fitted(model)
  1   2   3   4
4.9 6.3 7.7 9.1
```

```
> summary(model)

Call:
lm(formula = Y ~ X, data = mydata)

Residuals:
   1    2    3    4
 1.1 -1.3 -0.7  0.9

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.5000     1.7748   1.972    0.187
X             1.4000     0.6481   2.160    0.163

Residual standard error: 1.449 on 2 degrees of freedom
Multiple R-squared:    0.7,     Adjusted R-squared:   0.55
F-statistic: 4.667 on 1 and 2 DF,  p-value: 0.1633
```

This is really quartiles, it just worked out this way because we had 4 data points.

R lm summary

```
> summary(model)

Call:
lm(formula = Y ~ X, data = mydata)

Residuals:
   1    2    3    4
 1.1 -1.3 -0.7  0.9

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.5000     1.7748   1.972    0.187
X             1.4000     0.6481   2.160    0.163

Residual standard error: 1.449 on 2 degrees of freedom
Multiple R-squared:    0.7,     Adjusted R-squared:   0.55
F-statistic: 4.667 on 1 and 2 DF,  p-value: 0.1633
```

1 2 3 4 5 6 8 9 10

…legend on next slide

BAUER COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

#	Name	Description
1	Residuals	y – ŷ. If our residuals are normally distributed, this indicates the mean of the difference between our predictions and the actual values is close to 0 (good).
2	Significance Stars	The stars are shorthand for significance levels (p-value) computed. *** for high significance and * for low significance.
3	Estimated Coefficient	The estimated coefficient is the value of slope calculated by the regression (think of the Intercept as a slope that is always multiplied by 1). Always good to spot check this number to make sure it seems reasonable.
4	Standard Error of the Coefficient	Measure of the variability in the estimate for the coefficient. Lower means better but this number is relative to the value of the coefficient. As a rule of thumb, you'd like this value to be at least an order of magnitude less than the coefficient estimate.
5	t-value of the Coefficient Estimate	Score that measures whether or not the coefficient for this variable is meaningful for the model. Used to calculate the p-value and the significance levels.
6	Variable p-value	Probability the variable is NOT relevant. You want this number to be as small as possible. If the number is really small, R will display it in scientific notation. In or example 2e-16 means that the odds that parent is meaningless is about $1/5000000000000000$
7	Significance Legend	The more punctuation there is next to your variables, the better. Blank=bad, Dots=pretty good, Stars=good, More Stars=very good
8	Residual Std Error / Degrees of Freedom	Standard deviation of your residuals. You'd like this number to be proportional to the quantiles of the residuals. For a normal distribution, the 1st and 3rd quantiles should be 1.5 +/- the std error. The Degrees of Freedom is the difference between the number of observations included in your training sample and the number of variables used in your model (intercept counts as a variable). Degree of freedom is the number of non-zero coefficient estimates
9	R-squared	Metric for evaluating the goodness of fit of your model. Higher is better with 1 being the best. Corresponds with the amount of variability in what you're predicting that is explained by the model. In this instance.

```
> summary(model)

Call:
lm(formula = Y ~ X, data = mydata)

Residuals:
   1    2    3    4
 1.1 -1.3 -0.7  0.9

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.5000     1.7748   1.972    0.187
X             1.4000     0.6481   2.160    0.163

Residual standard error: 1.449 on 2 degrees of freedom
Multiple R-squared:    0.7,     Adjusted R-squared:   0.55
F-statistic: 4.667 on 1 and 2 DF,  p-value: 0.1633
```

Dimension Level

SSR			
Y Hat	Mean Y	Var	Sum Sq
4.9	7	-2.10	4.41
6.3	7	-0.70	0.49
7.7	7	0.70	0.49
9.1	7	2.10	4.41
Total		**0.00**	**9.80**

SSE			
Y	Y Hat	Var	Sum Sq
6	4.9	1.10	1.21
5	6.3	-1.30	1.69
7	7.7	-0.70	0.49
10	9.1	0.90	0.81
Total		**0.00**	**4.20**

Degrees of freedom: df = n-k-1
n = number of observations
k = number of estimated parameters

Model Level

SST (SSR + SSE)	14.00
R^2 (SSR/SST)	0.70
MST (SST / SST df))	4.67
MSE (SSE / (SSE df))	2.10
Adj R^2 (1-MSE/MST)	0.55
RSE (SQRT(SSE/df))	1.45

SSR = $\sum (\hat{y}_i - \bar{y})^2$ = sum of squares due to the regression

SSE = $\sum (y_i - \hat{y})^2$ = sum of squares due to error – also called the residual sum of squares

SST = SSR + SSE

$$\text{SE } \beta_1 = \sqrt{\frac{MSE}{\sum(x - \bar{x})^2}} = \sqrt{\frac{2.1}{5}} = 0.6481$$

This value with a β_1 of 1.4 does not inspire comfort!

Note how MSE gets smaller as n gets larger, and $\sum(x - \bar{x})^2$ gets larger as n gets larger, so SE gets smaller.

BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Model and Coefficient Errors

The model level SE represents the average distance that the observed values fall from the regression line. $\sqrt{\frac{SSE}{df}}$

The previous example had a poor SE – the sample size was very small *(note that df is in the denominator)* Which resulted in a p-value of .16

The coefficient Standard Error measures the average amount that the coefficient estimates vary from the actual average value of our response variable. The coefficient SE's can be used to compute confidence intervals and to statistically test the hypothesis of the existence of relationships. Computing the coefficient SE is a little more complex because ***with multiple regression***, the ***coefficient estimates the average effect on the response variable of that variable while holding other variables constant (see pg. 74 of ISL)***. So this can be computed using the covariance matrix. As you can see, the confidence intervals in this model are quite high. We will extend this briefly.

```
> vcov(model)
             (Intercept)     X
(Intercept)        3.15 -1.05
X                 -1.05  0.42
> se <- sqrt(diag(vcov(model))) #SE This is wh
nts
> se
(Intercept)           X
  1.7748239   0.6480741
> model$coefficients[1] + (2* se[1]) #UCI
(Intercept)
   7.049648
> model$coefficients[1] - (2* se[1]) #LCI
(Intercept)
-0.04964787
```

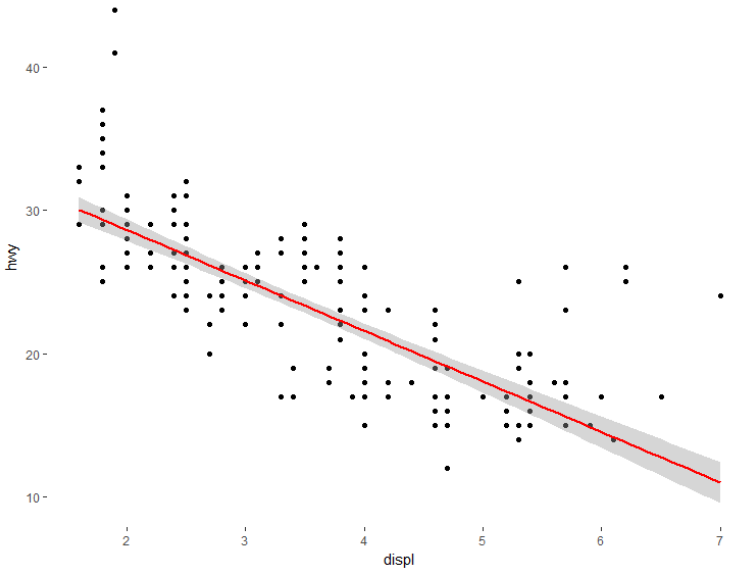
BAUER
COLLEGE OF BUSINESS
UNIVERSITY of HOUSTON

Confidence Intervals

Assumptions of Linear Regression

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad i = 1, \ldots, n.$$

- Linearity *(explanatory and response variables are linearly related)*
- Independence (*little or no correlation in the explanatory variables)*
- Constant Variance *(homoscedasticity – variance is evenly distributed)*
- Normality *(residuals are normally distributed)*

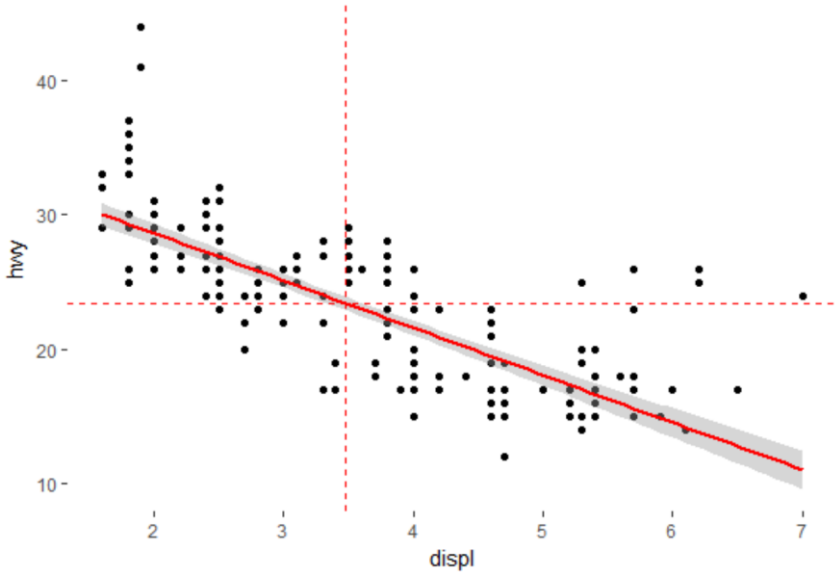



```
library(tidyverse)
dfMPG <- mpg

# chart the mpg data

p <- ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  geom_smooth(method = lm, mapping = aes(x = displ, y = hwy),
col = "red")+
  theme(
    panel.background = element_rect(fill = "white") # I do this so
it's easier to see in  class
  )
p
```

For normal distributions, 95% means (μ) will lie between ± 2SE

Confidence Interval : prediction ± 2SE

```
# add the means to the chart

ym <- mean(mpg$hwy)
xm <- mean(mpg$displ)
p <- p+ geom_hline(yintercept=ym, linetype="dashed", color =
"red")
p <- p+ geom_vline(xintercept=xm, linetype="dashed", color =
"red")
p
```



(note calculating the SE is more complex with multiple coefficients – see that section for calculation)

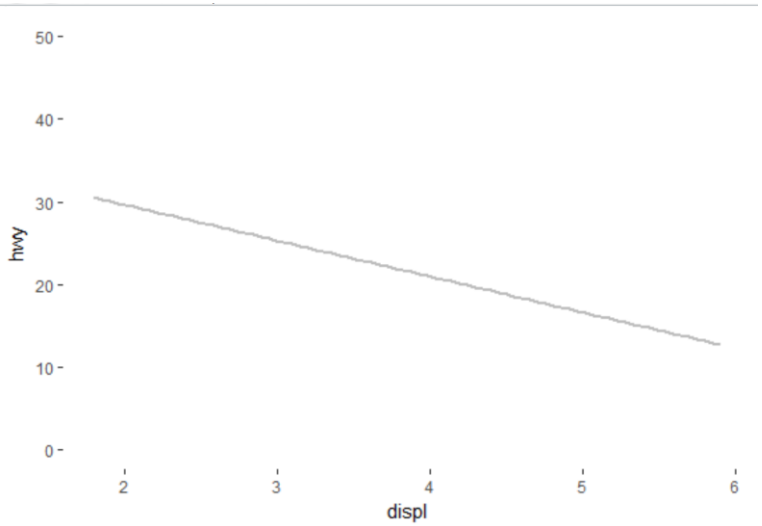
NOT ON EXAM

```
# this illustrates how sample size and CI relate
# adjust sample size and resampling to show effect on CI on
slope

masterMod <- lm(data = mpg, hwy~displ) # get the full population
summary(masterMod) # display std errors
SEI <- summary(masterMod)$coefficients[1,2] # get the SE for
the Intercept
MLI <- masterMod$coefficients[1] - (2*SEI) # set a point for lower
CI
MUI <- masterMod$coefficients[1] + (2*SEI) # set a point for
upper CI
MLI
MUI
n <- 20
# number of resampling iterations
resample <- 100

library(tidyverse)
indexes = sample(1:nrow(mpg), n)
mpg2 <- mpg[indexes,]

p <- ggplot(data=mpg2, aes(x=displ, y=hwy)) +
  geom_smooth(method=lm, formula = y ~ x, color = "gray",
se=FALSE) +
  theme(
    panel.background = element_rect(fill = "white")
  )+
  scale_y_continuous(limits = c(0, 50))
p
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  35.6977     0.7204   49.55   <2e-16 ***
displ        -3.5306     0.1945  -18.15   <2e-16 ***
```

```
> MLI
(Intercept)
   34.25692
> MUI
(Intercept)
   37.13839
```

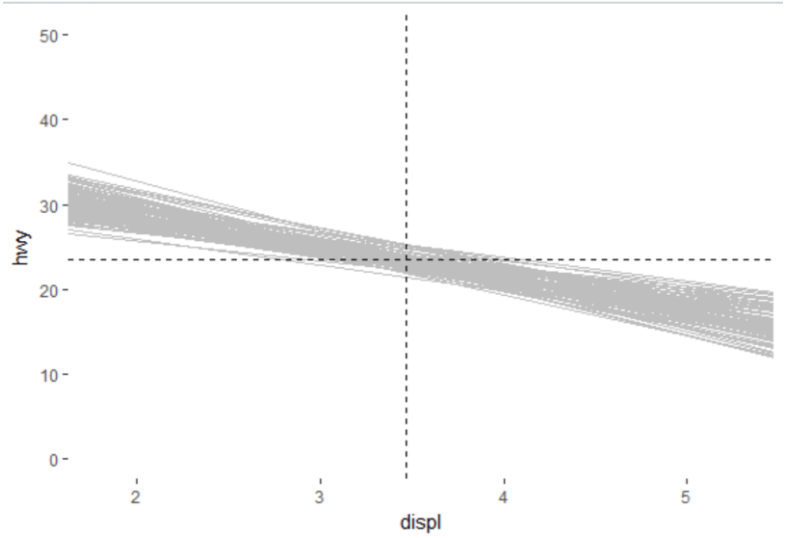
We'll gradually increase the sample size



```
Modcoef <- matrix(NA, nrow = resample, ncol = 3)
icnt <- 1
while (icnt <= resample)
{
  indexes = sample(1:nrow(mpg), n)
  mpg2 <- mpg[indexes,]
  mod <- lm(data = mpg2, hwy~displ)
  Modcoef[icnt, 1] <- mod$coefficients[1]
  Modcoef[icnt, 2] <- mod$coefficients[2]
  Modcoef[icnt, 3] <- summary(mod)$coefficients[1,2]
  p <- p +
       geom_abline(intercept = Modcoef[icnt, 1], slope =
Modcoef[icnt, 2], color = "gray")
     icnt <- icnt+1;
}
p

ym <- mean(mpg$hwy)
xm <- mean(mpg$displ)
p <- p+ geom_hline(yintercept=ym, linetype="dashed")
p <- p+ geom_vline(xintercept=xm, linetype="dashed")
p

dfModcoef <- data.frame (Modcoef)

CITest <- nrow(dfModcoef %>% filter(X1 < MUI & X1 > MLI))
```

CITest/resample

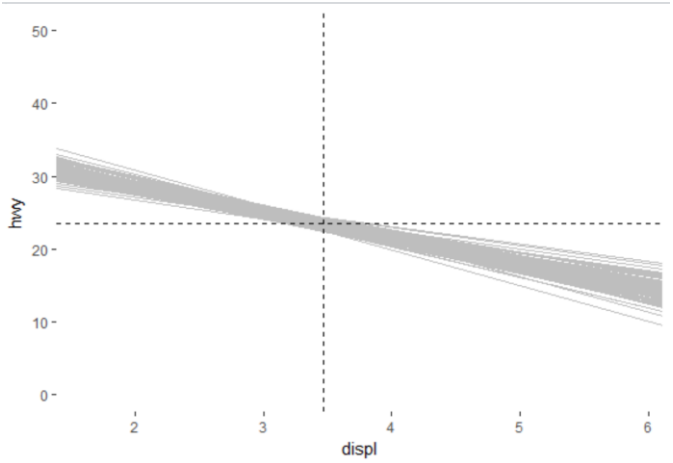
n <- 20




```
> CITest/resample
[1] 0.43
```

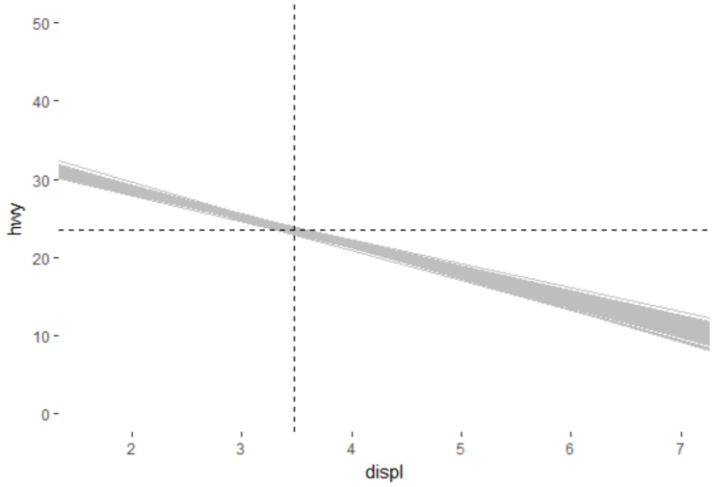
n <- 50

```
> CITest/resample
[1] 0.62
```



n <- 150

```
> CITest/resample
[1] 0.99
```



BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Regression Application

```
 Advertising <-  dbGetQuery(con2,"
SELECT
 [TV]
,[Radio]
,[Newspaper]
,[Sales]
FROM [dbo].[Advertising]
")


rfit <- lm(Sales ~ Radio, data = Advertising)
nfit <- lm(Sales ~ Newspaper, data = Advertising)

summary(rfit)
summary(nfit)

mFit <- lm(Sales ~ TV + Radio + Newspaper, data = Advertising)
summary(mFit)

# correlation matrix
cor(Advertising, method = 'pearson', use = 'pairwise')
```

Simple and multiple regression coefficients can be quite different. This difference stems from the fact that in the simple regression case, the slope term represents the average effect of a $1,000 increase in newspaper advertising, ignoring other predictors such as TV and radio. In contrast, in the multiple regression setting, the coefficient for newspaper represents the average effect of increasing newspaper spending by $1,000 while holding TV and radio fixed.

Compare Linear and Multivariate Models v2.R

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.31164    0.56290  16.542   <2e-16 ***
Radio        0.20250    0.02041   9.921   <2e-16 ***
---
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 12.35141    0.62142   19.88  < 2e-16 ***
Newspaper    0.05469    0.01658    3.30  0.00115 **
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
Radio        0.188530   0.008611  21.893   <2e-16 ***
Newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
> cor(Advertising, method = 'pearson', use = 'pairwise')
                TV      Radio  Newspaper     Sales
TV        1.00000000 0.05480866 0.05664787 0.7822244
Radio     0.05480866 1.00000000 0.35410375 0.5762226
Newspaper 0.05664787 0.35410375 1.00000000 0.2282990
Sales     0.78222442 0.57622257 0.22829903 1.0000000
```

Comparing

Tendency to spend more on newspaper in markets where more is spent on radio.

i.e., collinearity

surrogate variables *(see ISL pg 74)*

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
Radio        0.188530   0.008611  21.893   <2e-16 ***
Newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.686 on 196 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8956
F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

The t value is estimated by dividing the $\hat{\beta}$ value of the parameter by its standard error $\frac{\beta_1}{\sqrt{\frac{MSE}{\Sigma(x_1 - \bar{x}_1)^2}}}$. This statistic is a measure of the likelihood that the actual value of the parameter is not zero. The larger the absolute value of t, the less likely that the actual value of the parameter could be zero. The p-value is the probability of obtaining the estimated value of the parameter if the actual parameter value is zero (H_0).

Note: the $\frac{\hat{\beta} - \beta}{SE\hat{\beta}} \sim t_{n-k}$ *the Student t distribution (Gosset) is often used to compare 2 populations (e.g., t-tests comparing* $u_1 = u_2$*).*

The F statistic tests the overall significance of the model. Specifically, it tests the null hypothesis that *all* of the regression coefficients are equal to zero H_0:$\beta_1 \ldots \beta_p = 0$

Note: The F distribution *(named after Fisher)* is used to compare multiple populations $u_1 = u_3 = u_4 = u_5 \ldots$ *(vs. 2 with the t distribution)*. A simple interpretation is that it compares the variance ***between*** the populations to the variance ***within*** the populations: F-Statistic = $\frac{SSBetween/df}{SSWithin/df}$ so a large value says that the difference between populations is larger than the difference within, so we can reject H_0. And the p-value here is the probability of falsely rejecting H_0 *(type 1 error)*.

F statistic

Instead of judging coefficients of individual variables on their own for significance using t test, the F statistic judges on ***multiple coefficients taken together at the same time***.

H_0 : The fit of intercept only model and the current model is same. i.e. Additional variables do not provide value taken together.

H_a : The fit of intercept only model is significantly less compared to our current model. i.e. Additional variables do make the model significantly better.

$$F = \frac{explained\ variation/(k-1)}{unexplained\ variation/(n-k)} = \frac{R^2/(k-1)}{(1-R^2)(n-k)}$$

$$= \frac{.8972/(3)}{(1-.8972)(196)} = 570$$

re: http://analyticspro.org/2016/03/15/r-tutorial-how-to-interpret-f-statistic-in-regression-models/

If there is no relationship between the response and the predictors, then F will be near 1. In this case it's large and a strong indicator that relationships exist

```
mFit <- lm(Sales ~ TV + Radio + Newspaper, data = Advertising)
summary(mFit)

# Manually Calculate SE's

vY <- as.matrix(dplyr::select(Advertising, Sales)) # set up y values in matrix
mX <- as.matrix(cbind(1, dplyr::select(Advertising, TV, Radio, Newspaper))) # set up x values in matrix
vBeta <- solve(t(mX)%*%mX, t(mX)%*%vY) # solve using normal equations
dSigmaSq <- sum((vY - mX%*%vBeta)^2)/(nrow(mX)-ncol(mX)) # estimate the variance
mVarCovar <- dSigmaSq*chol2inv(chol(t(mX)%*%mX))
vStdErr <- sqrt(diag(mVarCovar))
print(cbind(vBeta, vStdErr))
summary(mFit)
```

```
> print(cbind(vBeta, vStdErr))
                Sales     vStdErr
1          2.938889369 0.311908236
TV         0.045764645 0.001394897
Radio      0.188530017 0.008611234
Newspaper -0.001037493 0.005871010
```

```
Residuals:
    Min      1Q  Median      3Q     Max
-8.8277 -0.8908  0.2418  1.1893  2.8292

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.938889   0.311908   9.422   <2e-16 ***
TV           0.045765   0.001395  32.809   <2e-16 ***
Radio        0.188530   0.008611  21.893   <2e-16 ***
Newspaper   -0.001037   0.005871  -0.177     0.86
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.686 on 196 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8956
F-statistic: 570.3 on 3 and 196 DF,  p-value: < 2.2e-16
```

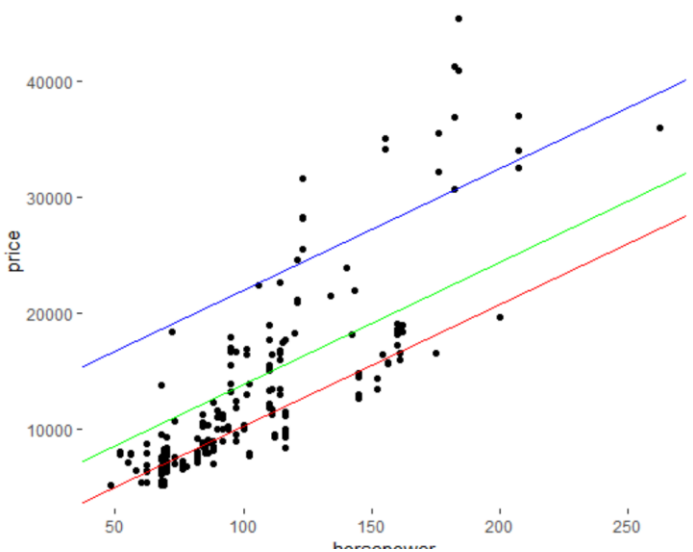
Coefficient f and p values report the partial effect of adding that variable to the model (notice Newspaper)

Standard errors are affected by collinearity too, and will grow artificially if it exists (notice the contrast with coefficient vs std error for TV vs Newspaper)

Categorical Variables

```
model2 <- lm(price ~ horsepower + make, Autos)
summary(model2)

Intercept <- coef(model2)["(Intercept)"] # alpha romeo, btw
BMW <- coef(model2)["makebmw"]
Honda <- coef(model2)["makehonda"]
Slope <- coef(model2)["horsepower"]

# BMW - you take the overall coefficients and add the
coefficients associated with the dummy varibles
p <- ggplot(Autos, aes(x=horsepower, y=price)) + geom_point() +
  geom_abline(intercept = (Intercept + BMW) , slope = Slope,
color = 'blue') +
  theme(panel.background = element_rect(fill = "white"))
p

# Honda

p <- p + geom_abline(intercept = (Intercept + Honda) , slope =
Slope, color = 'red')
p

coefMat <- summary(model2)$coefficients

AllMakes # just for reference
p <- p + geom_abline(intercept = mean(coefMat[1,1] +
coefMat[3:22,1]) , slope = Slope, color = 'green')
p
```

Regression Categorical Intro.R



Most datasets will include categorical data. You must get comfortable with using this as predictors.

lm automatically creates ***dummy variables (also called indicators - AML)*** for text and categorical data (translates to numerical values). If you want to control that you will use factors and set levels. Or just create an new column and set using some transform.

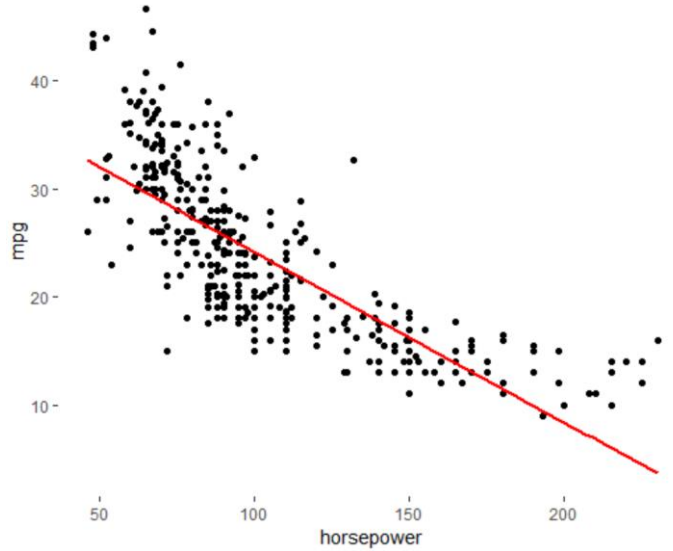
In this case the data are nominal and letting lm set dummy values works well.

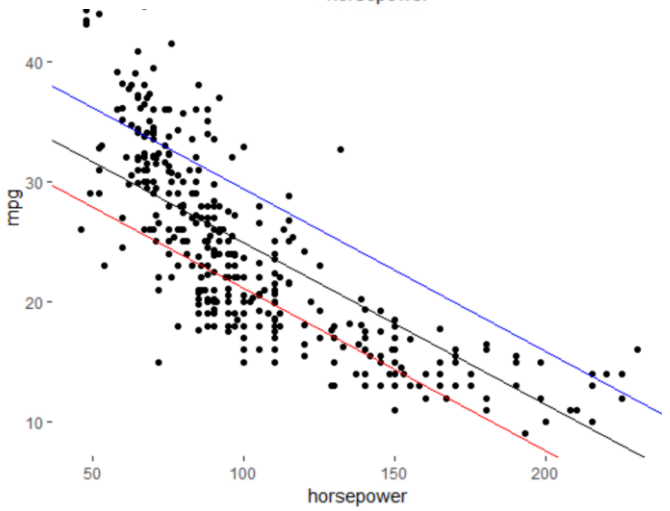
Categorical Variables + Polynomial

extended example from 3.3.2

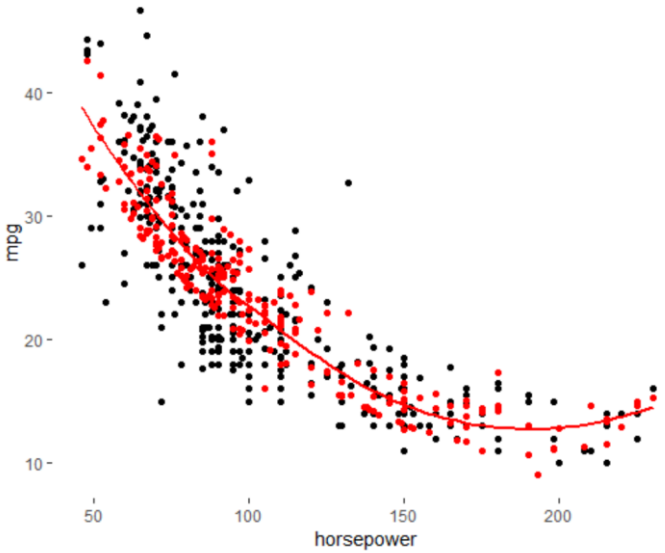
Regression Categorical Variables 2 v2.R

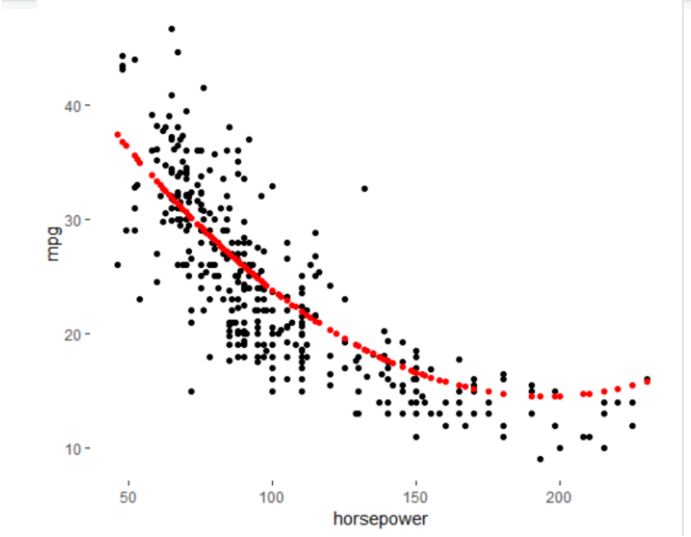
```
mod1 <- lm(mpg ~ make + horsepower, data = dfMPG)
summary(mod1)
rmse(mod1$residuals)


Intercept <- coef(mod1)["(Intercept)"]
Audi <- coef(mod1)["makeaudi"]
Honda <- coef(mod1)["makehonda"]
Ford <- coef(mod1)["makeford"]
Horsepower <- coef(mod1)["horsepower"]


p <- ggplot(data = dfMPG) +
  geom_point(mapping = aes(x = horsepower, y = mpg)) +
  geom_abline(intercept = Intercept+Audi, slope = Horsepower) +
  geom_abline(intercept = Intercept+Honda, slope = Horsepower,
color = "blue") +
  geom_abline(intercept = Intercept+Ford, slope = Horsepower,
color = "red") +
  theme(
    panel.background = element_rect(fill = "white") # I do this so
it's easier to see in  class
  )
p
```





```
# note how the shape of the data is non-linear and there is consistent shape
# so we're going to test a polynomial model
mod2 <- lm(mpg ~ make + horsepower + I(horsepower^2), data = dfMPG)
rmse(mod2$residuals)
summary(mod2)
dfMPG$predict <- predict(mod2, dfMPG)
p <- ggplot(data = dfMPG) +
  geom_point(mapping = aes(x = horsepower, y = mpg)) +
  geom_point(mapping = aes(x = horsepower, y = predict), color = "red") +
  geom_smooth(mapping = aes(x = horsepower, y = predict), color = "red", se = F) +
  theme(
    panel.background = element_rect(fill = "white") # I do this so it's easier to see in  class
  )
p
# ok let's pull the equation out and work with that
p <- ggplot(data = dfMPG) +
  geom_point(mapping = aes(x = horsepower, y = mpg)) +
  theme(
    panel.background = element_rect(fill = "white") # I do this so it's easier to see in  class
  )
p
Intercept <- coef(mod2)["(Intercept)"]
Audi <- coef(mod2)["makeaudi"]
Horsepower <- coef(mod2)["horsepower"]
Phorsepower <- coef(mod2)["I(horsepower^2)"]

Intercept + Audi
# here's the equation using Audi
dfMPG$pPredict <- (Intercept + Audi) + (Horsepower*dfMPG$horsepower) + (Phorsepower *
(dfMPG$horsepower^2))
p <- p + geom_point(data = dfMPG, aes(x = horsepower, y = pPredict), color = "red")
p
# what's the value fo y @ x = 100?
x <- 100
y <- (53) + (-.4*x) + (.001 * (x^2))
```





BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

Interaction Effects

See ISL 3.3.2

The interaction effect captures the "synergy" of multiple channels in advertising – relaxing the additive effect

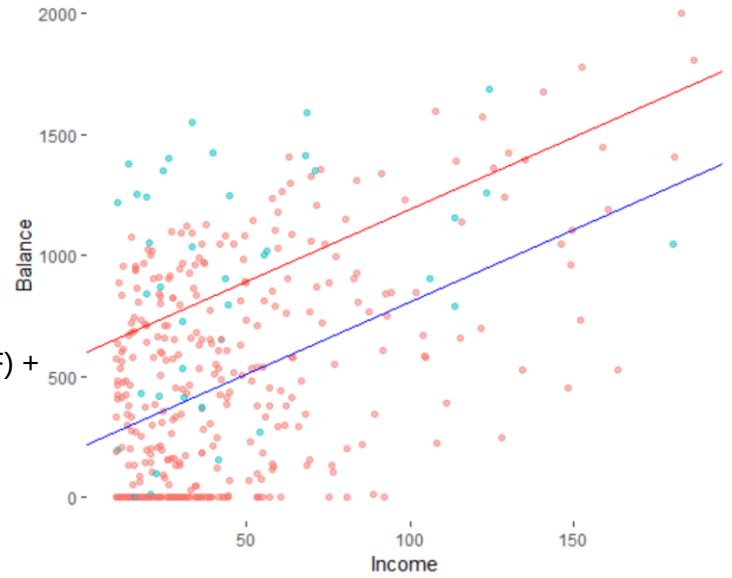
```
> Advertising <-  dbGetQuery(con2,"
+                            SELECT
+                            [TV]
+                            ,[Radio]
+                            ,[Newspaper]
+                            ,[Sales]
+                            FROM [dbo].[Advertising]
+                            ")
>
>
> mFit1 <- lm(Sales ~ TV + Radio, data = Advertising)
> summary(mFit1)

Call:
lm(formula = Sales ~ TV + Radio, data = Advertising)

Residuals:
    Min      1Q  Median      3Q     Max
-8.7977 -0.8752  0.2422  1.1708  2.8328

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.92110    0.29449   9.919   <2e-16 ***
TV           0.04575    0.00139  32.909   <2e-16 ***
Radio        0.18799    0.00804  23.382   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.681 on 197 degrees of freedom
Multiple R-squared:  0.8972,    Adjusted R-squared:  0.8962
F-statistic: 859.6 on 2 and 197 DF,  p-value: < 2.2e-16
```

```
> rmse(mFit1$residuals) # using this here will let us compare insamp
e
[1] 1.668703
>
> mFit2 <- lm(Sales ~ TV + Radio + (TV*Radio), data = Advertising)
> summary(mFit2)

Call:
lm(formula = Sales ~ TV + Radio + (TV * Radio), data = Advertising)

Residuals:
     Min       1Q   Median       3Q      Max
-6.3366 -0.4028  0.1831  0.5948  1.5246

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 6.750e+00  2.479e-01  27.233   <2e-16 ***
TV          1.910e-02  1.504e-03  12.699   <2e-16 ***
Radio       2.886e-02  8.905e-03   3.241   0.0014 **
TV:Radio    1.086e-03  5.242e-05  20.727   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9435 on 196 degrees of freedom
Multiple R-squared:  0.9678,    Adjusted R-squared:  0.9673
F-statistic:  1963 on 3 and 196 DF,  p-value: < 2.2e-16

> rmse(mFit2$residuals) # using this here will let us compare insamp
e
[1] 0.9340326
```

DA1 Interaction Effects.R

Another example using different data. ***(read chapter 3!)***

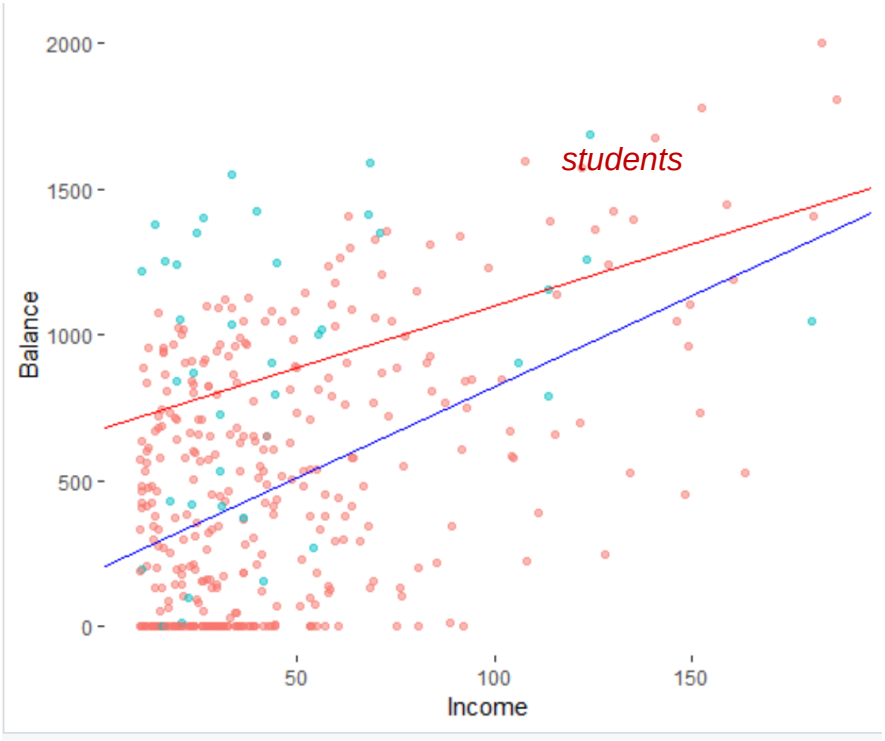
```
dfCredit <- Credit

mFit1 <- lm(Balance ~ Income + Student, data = dfCredit)

summary(mFit1)
mFit1$coefficients
rmse(mFit1$residuals) # using this here will let us compare insample to
outofsample rmse

ggplot(dfCredit, aes(x = Income, y = Balance)) +
  geom_point(alpha = .5, aes(color = Student)) +
  geom_smooth(method = "lm", aes(group = Student, color = Student), se = F) +
  theme(legend.position="none")  +
  theme(panel.background = element_rect(fill = "white"))

ggplot(dfCredit, aes(x = Income, y = Balance)) +
  geom_point(alpha = .5, aes(color = Student)) +
  geom_abline(data = dfCredit, aes(intercept = mFit1$coefficients[1], slope =
mFit1$coefficients[2]), color = "blue") +
  geom_abline(data = dfCredit, aes(intercept =
mFit1$coefficients[1]+mFit1$coefficients[3], slope = mFit1$coefficients[2]), color
= "red") +
  theme(legend.position="none")  +
  theme(panel.background = element_rect(fill = "white"))
```



Now with interaction effect

The lower slope suggests that changes in income have less increase in credit card balance with students

```
mFit2 <- lm(Balance ~ Income + Student + (Income * Student),
data = dfCredit)

summary(mFit2)
mFit2$coefficients
rmse(mFit2$residuals) # using this here will let us compare
insample to outofsample rmse


ggplot(dfCredit, aes(x = Income, y = Balance)) +
  geom_point(alpha = .5, aes(color = Student)) +
  geom_abline(data = dfCredit, aes(intercept =
mFit2$coefficients[1], slope = mFit2$coefficients[2]), color =
"blue") +
  geom_abline(data = dfCredit, aes(intercept =
mFit2$coefficients[1]+mFit2$coefficients[3], slope =
mFit2$coefficients[2]+mFit2$coefficients[4]), color = "red") +
  theme(legend.position="none")  +
  theme(panel.background = element_rect(fill = "white"))
```



$$\text{Balance} = \beta_0 + \beta_1 * \text{Income} + \begin{cases} \beta_2 + \beta_3 * \text{income} & \textit{Student} = 1 \\ 0 & \textit{Student} = 0 \end{cases}$$

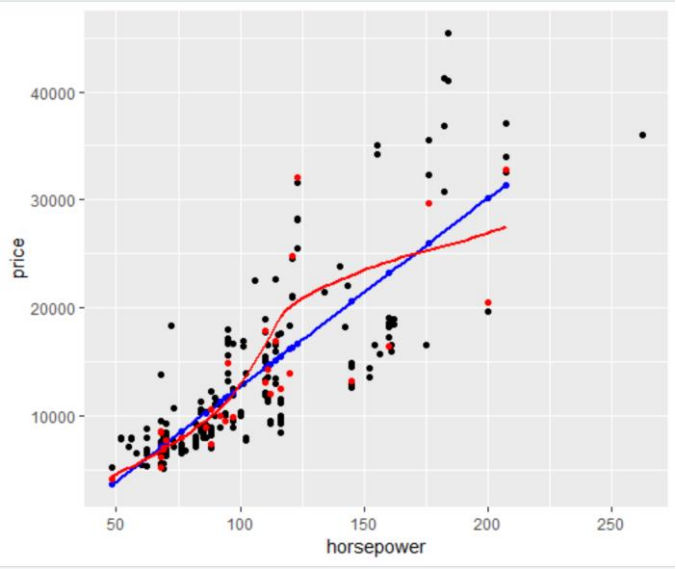
```
> contrasts(dfCredit$Student)
    Yes
No    0
Yes   1
```

BAUER COLLEGE OF BUSINESS UNIVERSITY of HOUSTON

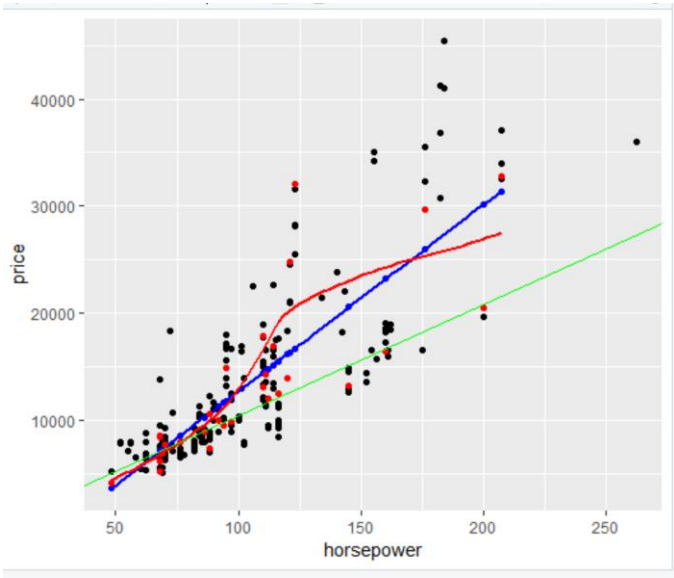
Exercise 1

```
# this eliminates makes with less than 2 obs - more on stratified sampling in DA2
Autos <- rowid_to_column(Autos, var="SampleID") # this creates a primary key for sampling
by_MakeStyle <- Autos %>% group_by(make) %>% dplyr::mutate(cnt = n()) %>% filter(cnt > 2)
xTrain <- sample_frac(by_MakeStyle, .8)
xTest <- anti_join(by_MakeStyle, xTrain, by = "SampleID")
filter(xTrain %>% distinct(make)) %>% anti_join(filter(xTest %>% distinct(make)), by = "make") %>% nrow()

# chart out the data in terms of horsepower

p <- ggplot(Autos, aes(x=horsepower, y=price))+geom_point()
P
```

```
model2 <- lm( price ~ make + horsepower, xTrain)
xTest$pred2 <- predict(model2, xTest)

# chart the MV model too, and compare statistics

p <- p + geom_point(data=xTest, aes(horsepower, pred2), color = "red") +
  geom_smooth(data=xTest, aes(horsepower, pred2), se=FALSE, color = "red")
p
```

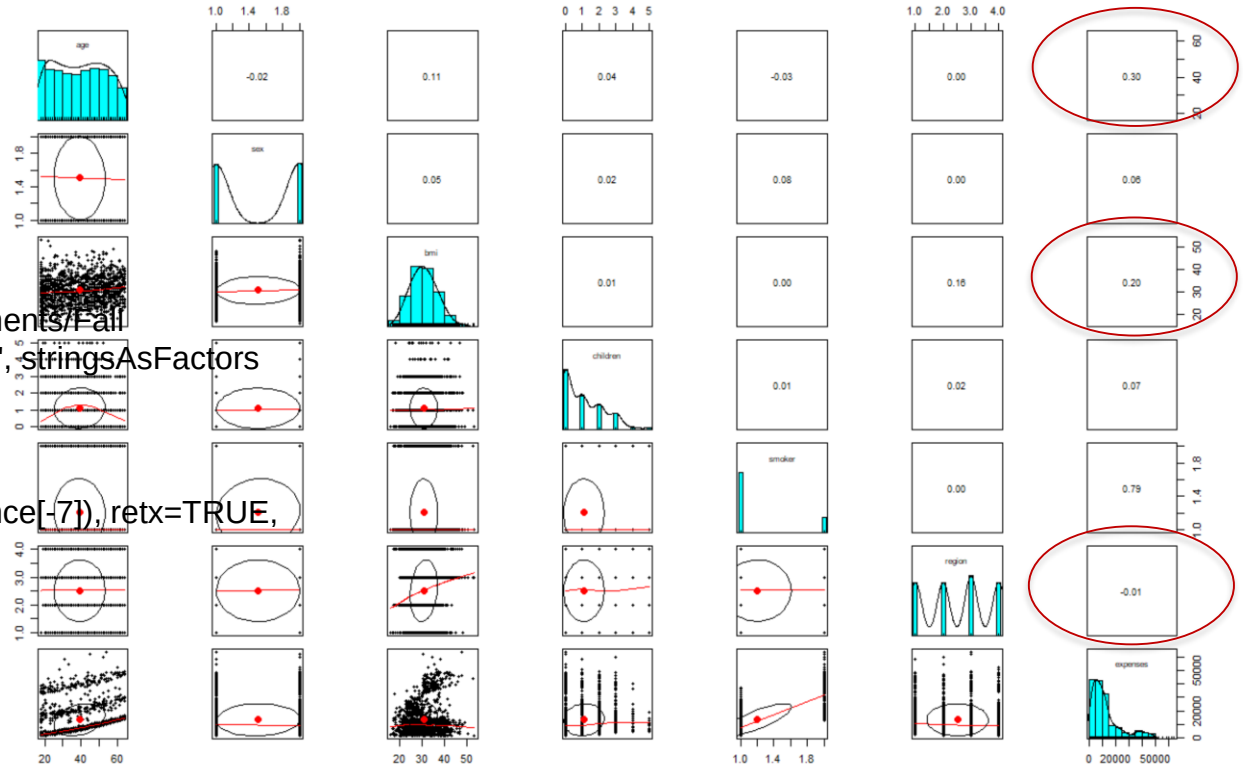


```
> modRMSE2
[1] 2400.455
>
> Intercept <- coef(model2)["(Intercept)"]
> Horsepower <- coef(model2)["horsepower"]
> Honda <- coef(model2)["makehonda"]
>
> p <- p + geom_abline(intercept = Intercept+Honda, slope = Horsepower, color = "green")
> p
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
`geom_smooth()` using method = 'loess' and formula 'y ~ x'
>
> # model for honda:
>
> x <- 76
> y <- (2742.593 - 2743.667) + 103.8295 * x
> y
[1] 7889.968
> # confirm
> test3 <- filter(xTest, make == "honda" & horsepower == 76)
> predict(model2, test3)
       1        2
7889.967 7889.967
>
> # how much horsepower can I buy for 20,000 with a honda
>
> y <- 20000
> x <- (y - (2742.593 - 2743.667))/103.8295
> x
[1] 192.6338
```

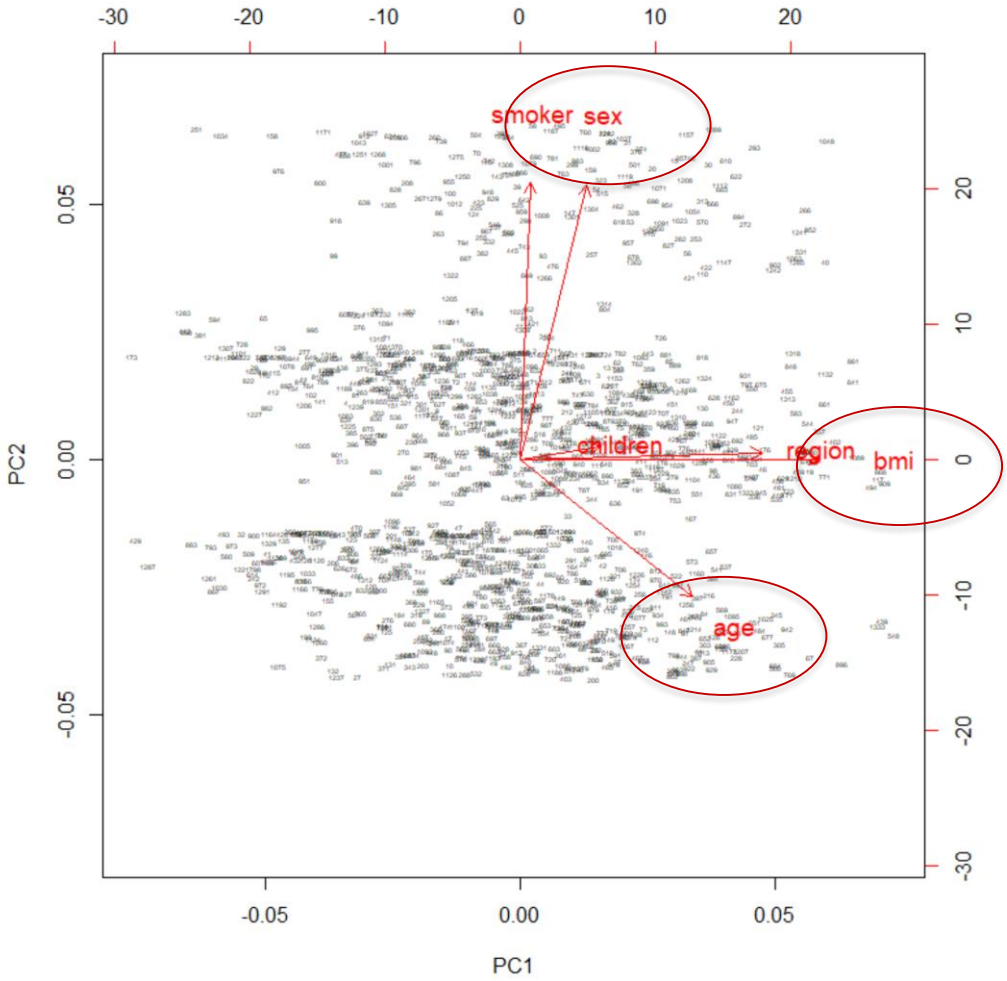


Exercise 2

```
library(tidyverse)
library(psych)

rmse <- function(error)
{
  sqrt(mean(error^2))
}

insurance <- read.csv("C:/Users/ellen/Documents/Fall
2018/DA1/Section2/New Data/insurance.csv", stringsAsFactors
= TRUE)

pairs.panels(data.matrix(insurance))
Insurance.pca <- prcomp(data.matrix(insurance[-7]), retx=TRUE,
center=TRUE, scale.=TRUE)
biplot(Insurance.pca, cex=c(.3,1.2))
```



```
Insurance.pca <-
prcomp(data.matrix(insurance[-7]), retx=TRUE,
center=TRUE, scale.=TRUE)
biplot(Insurance.pca, cex=c(.3,1.2))
```



```
> insurance <- rowid_to_column(insurance, var="SampleID") # this creat
r sampling
> xTrain <- sample_frac(insurance, .6)
> xTest <- anti_join(insurance, xTrain, by = "SampleID")
>
> mod1 <- lm(expenses ~ age + children + bmi + sex + smoker + region,
+                  data = xTrain)
> summary(mod1)

> xTest$Pred <- predict(mod1, xTest)
> # compare the correlation
> cor(xTest$expenses, xTest$Pred)
[1] 0.8430417
> rmse(mod1$residuals)
[1] 5794.378
> rmse(xTest$expenses - xTest$Pred)
[1] 6417.452
> # keep in mind that the model residuals are just an estimate!!

> # now let's try different models and see how they fare!
>
> mod2 <- lm(expenses ~ age + bmi + sex + smoker, data = xTrain)
> rmse(mod2$residuals)
[1] 5824.079
> mod3 <- lm(expenses ~ age + bmi + sex + smoker + bmi*smoker, data = xTrain)
> rmse(mod3$residuals)
[1] 4631.609
> mod4 <- lm(expenses ~ age + bmi + sex + smoker + bmi*smoker + I(age^2), data = xTrain)
> rmse(mod4$residuals)
[1] 4620.584
```

Interaction effect captures relationship between bmi and smoker (i.e., an overweight smoker is more likely to have health issues than an overweight, or smoker alone)

The polynomial on age captures a non-linear effect of aging (as someone gets older, there is a marginal increase in likelihood of health issues)