Data Science for Agricultural Professionals

Marin L. Harbur

2021-05-11

Contents

Preface **5**

1 Introduction **7**

1.1 Welcome 7

1.2 R-language 8

1.3 Populations 8

1.4 Case Study: Yield Map 10

1.5 Distributions 12

2 Distributions and Probability" **25**

2.1 Review 25

2.2 Case Study 25

2.3 The Normal Distribution Model 27

2.4 The Z-Distribution 32

Preface

Chapter 1

Introduction

Some text.

1.1 Welcome

Welcome to Data Science for Agricultural Professionals. I have written these course materials for a few reasons. First, it is my job. More inspirationally, however, I wanted to write a guide that satistfied the following criteria: - covers basic statistics used in reporting results from hybrid trials and other controlled experiments - also addresses data science tools used to group environments and make predictions - introduced students to R, an open-source statistical language that you can use after your studies at Iowa State University and can use without installing on your laptop, or using a VPN connection, which your work laptop may not allow.

I also wanted to develop a text that presented statistics around the situations in which you are most likely to encounter data: - yield maps used by farmers and agronomists - side-by-side or split-field trials used often at the retail level - smaller-plot controlled experiments used in seed breeding and other product development - rate recommendation trials for fertilizers and crop protection products - fertilizer prediction maps - decision support tools

I began my career as an a university researcher and professor, but in 2010 entered the private sector, working first in retail as a technical agronomist for a regional cooperative and then as a data scientist for a major distributor, developing product data and insights for a team of researchers and agronomists. In seeing how data were used at the retail and industry levels, I gained an appreciation for what areas of statistics were more often used than others.

Finally, I wanted to develop a reference that appreciated not all of us are mathematical progedies, nor do we have flawless memories when it comes to formula.

At the risk of redudancy, I will re-explain formulas and concepts – or at least provide links – so you aren't forever flipping back and forth trying to remember sums of squares, standard errors, etc.

I hope you will find this guide practical and as painless as a statistics text can be.

1.2 R-language

One of the most immediate this you will learn in this course is how to begin using R to summarise data, calculate statistics, and view trends and relationships in data. R is open-source (free) and incredibly versatile. I have used multiple languages, including SAS and SPSS, in my career; in my opinion, R is not only the cheapest, but the best, and I now use it exclusively and almost daily in my work.

R also has personal connections to Iowa State: Hadley Wickam, Carson Seivert, and other data scientists were once students here at Iowa State !

We will be using an application called R-Studio to work with R language. R Studio allows you to write and save scripts (code) to generate analyses. It also allows you to intersperse Markdown (html code) in between your statistical output so that you can explain and discuss your results. The beauty of Markdown is your report and your statistics are together in a single file; no cutting and pasting is needed between documents.

R is all about having options, and so with R-Studio you have the option of installing it on your laptop or, in case you are using a work laptop without administrative priviledges, you can also use a cloud site, R-Studio Cloud, to work with R and save your work.

R itself is not only open-source, but is supported by many great books which may be accessed for free online, or purchased online for very reasonable prices.

1.3 Populations

Almost every statistics text begins with the concept of a population. A population is the complete set of individuals to which you want to predict values. Let's dwell on this concept, as it is something that did not hit home for me right away in my career. Again, the population is all of the individuals for which you are interested in making a prediction. What do we mean by individuals? Not just people – individuals can plants, insects, disease, livestock or, indeed, farmers.

Just as important as what individuals are in a population is its extent. What do you want the individuals to represent? If you are a farmer, do you want to apply the data from these individuals directly to themselves, or will you use

them to make management decisions for the entire field, or all the fields in your farm? Will you use the results this season or to make predictions for future seasons? If you are an animal nutritionist, will you use rations data to support dairy Herefords, or beef Angus?

If you are a sales agronomist, will you use the data to support sales on one farm, a group of farms in one area, or across your entire sales territory? If you are in Extension, will the individuals you measure be applicable to your entire county, group of counties, or state? If you are in industry like me, will your results be applicable to several states?

This is a very, very critical question, as you design experiments – or as you read research conducted by others. To what do or will the results apply? Obviously, an Iowa farmer should not follow the optimum planting date determined for a grower in North Carolina, nor should an Ohio farmer assume our pale clays will be as forgiving as the dark, mellow loam of southern Minnesota.

Drilling down, you might further consider whether research was conducted in areas that have similar temperature or rainfall, how different the soil texture might be to the areas to which you want to apply results. At the farm level, you might ask how similar the crop rotation, tillage, or planting methods were to that operation. At the field level, you might wonder about planting date or the hybrid that was sown.

When you use the results from set of individuals to make predictions about other individuals, you are making inferences – you are using those data to make predictions, whether it be for that same field next month or next year, or for other locations (areas in a field, fields in a farm, counties, or states). When we speak of an inference space, then, that is the total group of individuals to which you will apply your results. Or, in another word, your population.

In summary, one of the most critical skills you can apply with data science has no formula and, indeed, little to do with math (at least in our application). It is to ask yourself, will my experiment represent the entire population in which I am interested? Indeed, one field trial likely will not address the entire population in which you are interested – it is up to you to determine the population to which you are comfortable applying those results.

In fact, statistics or data science done without this "domain" knowledge whether a dataset is useful or experimental results are reasonable can be disasterous. Just because a model fits one dataset very well or treatments are significantly different does not mean they should be used to make decisions. Your competency as an agronomic data scientist depends on everything you learn in your program of study. Soil science, crop physiology, and integrated pest management, to name just a few subjects, are as much a prerequisite as any math course you have taken.

In some cases all of the individuals in a population can be measured – in such case, we will use the basic statistics described in this unit. The yield map we will analyze in this unit is a loose example of a case where we can measure

In most cases, however, it is not physically or financially feasible to measure all individuals in a population. In that case, subsets of the population, called samples, are used to estimate the range of individuals in a population.

1.4 Case Study: Yield Map

For our first case study, we will use a situation where every individual in our population can be measured: a single soybean field in central Iowa. In this case, yield data were gathered using a combine monitor. In case you don't live and breathe field crops like me, combines (the machines that harvest grain) are usually equipped with a scale that repeatedly weighs grain as the combine moves across the field. The moisture of the grain is also recorded. These data are combined with measures of the combine speed and knowledge of the number of rows harvested at once to calculate the yield per area of grain, adjusted to the market standard for grain moisture.

```
library(sf)
library(tidyverse)
library(rcompanion)


yield = st_read("data-unit-1/merriweather_yield_map/merriweather_yield_map.shp")
```

```
## Reading layer `merriweather_yield_map' from data source `C:\Users\559069\OneDrive -
## Simple feature collection with 6061 features and 12 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -93.15474 ymin: 41.66619 xmax: -93.15026 ymax: 41.66945
## geographic CRS: WGS 84
```

Let's start tearing this dataset apart, using our very first chunk of code. We can examine the first 6 rows of the dataset using R. Our data are in a shapefile named "yield". To view the top six rows of any dataset in R, we use the command "head".

Just click on the button "Run Code" in the upper right corner of the window below.

```
head(yield)
```

```
## Simple feature collection with 6 features and 12 fields
## geometry type:  POINT
## dimension:      XY
```

```
## bbox:           xmin: -93.15033 ymin: 41.66641 xmax: -93.15026 ymax: 41.66644
## geographic CRS: WGS 84
##    DISTANCE SWATHWIDTH VRYIELDVOL Crop  WetMass Moisture                 Time
## 1 0.9202733          5   57.38461  174 3443.652     0.00 9/19/2016 4:45:46 PM
## 2 2.6919269          5   55.88097  174 3353.411     0.00 9/19/2016 4:45:48 PM
## 3 2.6263101          5   80.83788  174 4851.075     0.00 9/19/2016 4:45:49 PM
## 4 2.7575437          5   71.76773  174 4306.777     6.22 9/19/2016 4:45:51 PM
## 5 2.3966513          5   91.03274  174 5462.851    12.22 9/19/2016 4:45:54 PM
## 6 3.1840529          5   65.59037  174 3951.056    13.33 9/19/2016 4:45:55 PM
##    Heading VARIETY Elevation                  IsoTime yield_bu
## 1 300.1584   23A42  786.8470 2016-09-19T16:45:46.001Z 65.97034
## 2 303.6084   23A42  786.6140 2016-09-19T16:45:48.004Z 64.24158
## 3 304.3084   23A42  786.1416 2016-09-19T16:45:49.007Z 92.93246
## 4 306.2084   23A42  785.7381 2016-09-19T16:45:51.002Z 77.37348
## 5 309.2284   23A42  785.5937 2016-09-19T16:45:54.002Z 91.86380
## 6 309.7584   23A42  785.7512 2016-09-19T16:45:55.005Z 65.60115
##                     geometry
## 1 POINT (-93.15026 41.66641)
## 2 POINT (-93.15028 41.66641)
## 3 POINT (-93.15028 41.66642)
## 4  POINT (-93.1503 41.66642)
## 5 POINT (-93.15032 41.66644)
## 6 POINT (-93.15033 41.66644)
```

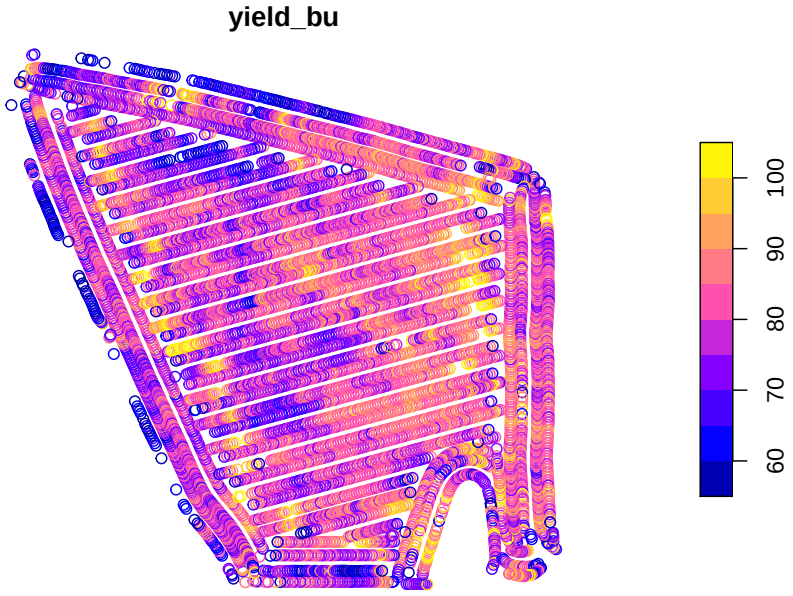
That's it! You just ran your first R command! Seriously, R includes a free software package that allows R to run inside a document like this. How cool is that! You could type any R command in the window above and it would run.

This dataset has several columns, but the two most important to us in this lesson are "yield_bu" and "geometry". Harvest files may not come with a yield column – in this case, I calculated the yield for you. I will not get into the formula for that – after all, this is just the first lesson of the course.

That this dataset has a column named geometry indicates it is a "shapefile" – a dataset in which the measures are georeferenced. That is, we know where these measurements were taken. The geometry column in this case identifies a point with each observation.

We can see this by plotting the data below. This command has three parts. Plot tells R to draw a map. "yield" tells R to use the yield dataset. "yield_bu" is the column we want to plot – we place it in quotes and brackets ([]) to tell R to only plot this column. Otherwise R will plot each of the columns, which will require a longer runtime.

```
plot(yield["yield_bu"])
```



What do you observe if you replace "yield_bu" with "Moisture" in the above code? Does one area of the field appear a little drier than the others?

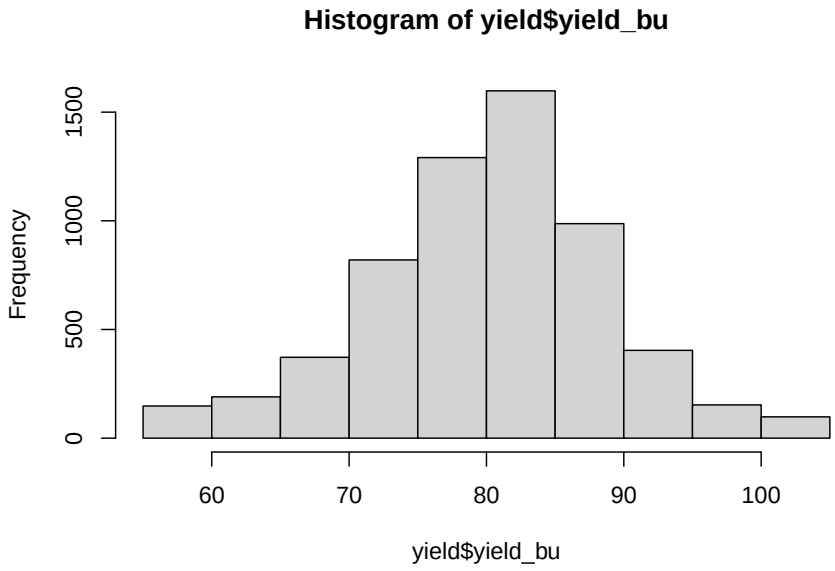
1.5 Distributions

At this point, we have two options any time we want to know about soybean yield in this field. We can pull out this map or the complete dataset (which has over 6,500 observations) and look at try to intuitively understand the data. Or we can use statistics which, in a sense, provide us a formula for regenerating our dataset with just a few numbers.

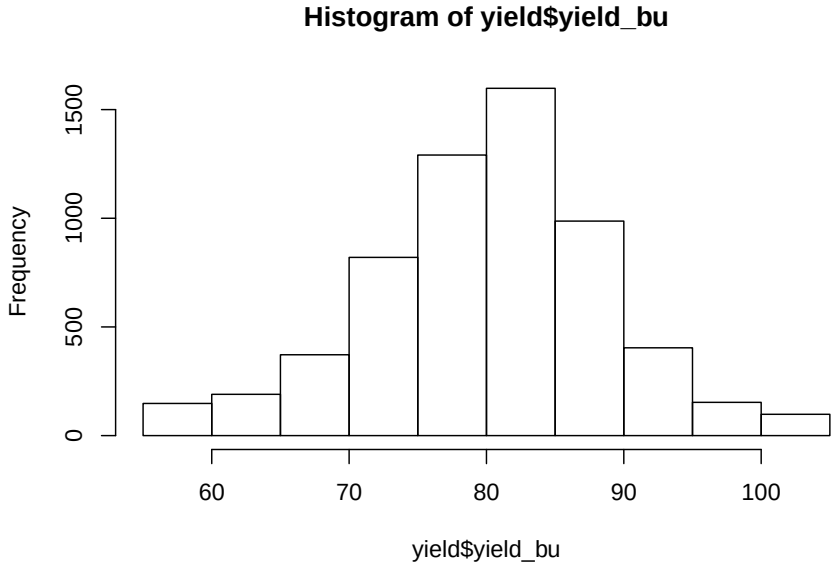
1.5.0.1 Histograms

Before we get into the math required to generate these statistics, however, we should look at the "shape" of our data. One of the easiest and most informative things for us to do is to create a particular bar chart known as a histogram.

```
histogram = hist(yield$yield_bu, breaks = 12)
```




```
plot(histogram)
```



A histogram gives us a quick visualization of our data. In the histogram above, each bar represents range of values. This range is often referred to as a *bin*.

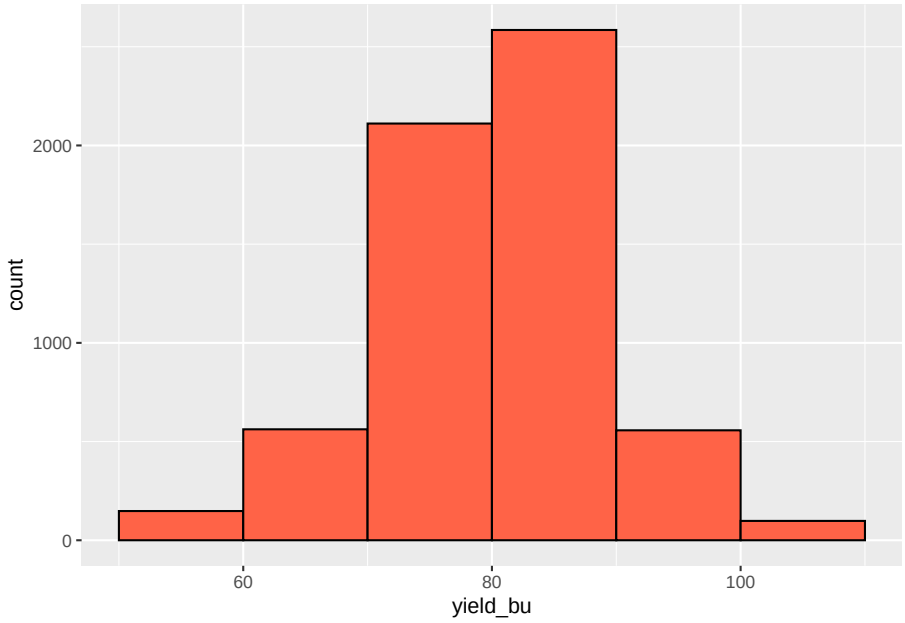
The lowest bin includes values from 50 to 59.0000. The next bin includes values from 60 to 69.9999. And so on. The height of each bar represents the number of individuals in that population that have values within that range.

Bins can also be defined by their midpoints. The midpoint is the middle value in each range. For the bin that includes values from 50 to 50.9999, the midpoint is 55. For the bin that includes values from 60 to 60.9999, the midpoint is 65.

In the plot above, the midpoint for each bar is indicated by the orange bar beneath it.

We can also draw a histogram using the power ggplot package from Hadley Wickham, which allows for much greater customization of plots. We will use this package extensively throughout the course. Here is just a taste:

```
ggplot(data=yield, aes(x=yield_bu)) +
  geom_histogram(breaks=seq(50, 110, 10), fill="tomato", color="black")
```



Varying the bin width provides us with different perspectives on our distribution. Please click on the link below to open an application where you can vary the bin width and see how it changes your perspective.

1.5.0.2 Percentiles

We can also use percentiles to help us describe the data even more numerically. To identify percentiles, the data are numerically ordered (ranked) from lowest

to highest. Each percentile is associated with a number; the percentile is the percentage of all data equal to or less than that number. We can quicly generate the 0th, 25th, 50th, and 75th, and 100th percentile in R using the summary command.

```
summary(yield$yield_bu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   55.12   74.96   80.62   80.09   85.44  104.95
```

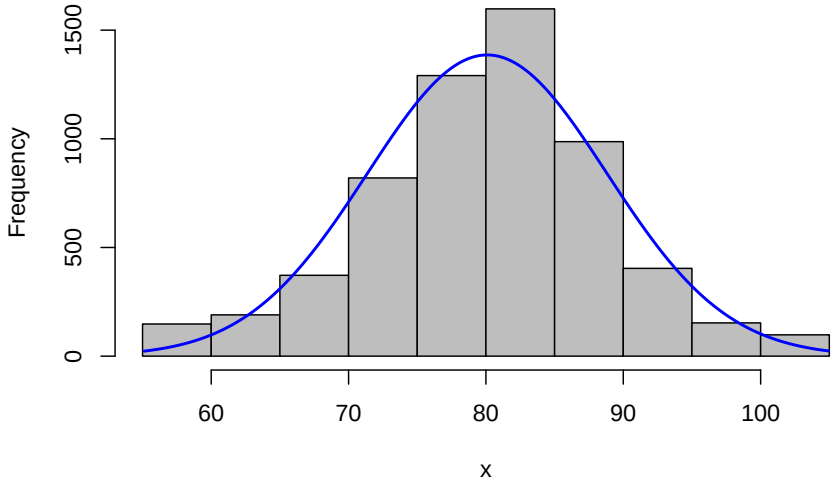
This returns six numbers. The 0th percentile (alternatively referred to as the minimum) is 0.00 – this is the lowest yield measured in the field. The 25th percentile (also called the 1st Quartile) is 87.16. This means that 25% of all observations were equal to 86.55 bushels or less. The 50th percentile (also known as the median) was 97.91, meaning half of all observations were equal to 97.91 bushels or less. 75% of observations were less than 111.13, the 75th percentile (or 3rd quartile). Finally, the 100th percentile (or maximum yield) recorded for this field was 199.60.

We are now gaining a better sense of the range of observations that were most common. But we can do even better and describe this distribution with even fewer numbers.

1.5.0.3 Normal Distribution Model

Let's overlay a curve, representing the normal distribution model, on our histogram with the following function, plotNormalHistogram.

```
plotNormalHistogram(yield$yield_bu, breaks = 10)
```



In a perfect situation, our curve would pass through the midpoint of each bar. This rarely happens with real-world data, and especially in agriculture. The data may be slightly skewed, meaning there are more individuals that measure above the mean than below, or vice versa.

In this example, our data do not appear skewed. Our curve seems a little too short and wide to exactly fit the data. This is a condition called kurtosis. No, kurtosis doesn't mean that our data stink. It just means it is a little more spread out or compressed than in a "perfect" situation.

No problem. We can – and should – conclude it is appropriate to fit these data with a normal distribution. If we had even more data, the curve would likely fit them even better.

Many populations can be handily summarized with the normal distribution curve, but we need to know a couple of statistics about the data. First, we need to know where the *center* of the curve should be. Second, we need to know the width or *dispersion* of the curve.

1.5.0.4 Measures of Center

To better describe our distribution, we will use the arithematic mean and median. The mean is the sum of all observations divided by the number of observations.

$$\mu = \frac{\sum x_i}{n}$$

The μ symbol (a u with a tail) signifies the true mean of a population. The $\sum$ symbol (the character next to x_i which looks like an angry insect) means "sum". Thus, anytime you see the $\sum$ symbol, we are summing the variable(s) to its right. x_i is the value x of the ith individual in the population. Finally, n is the number of individuals in the population.

For example, if we have a set of number from 1:5, their mean can be calculated as:

```
(1+2+3+4+5)/5
```

```
## [1] 3
```

R, of course, has a function for this:

```
x = c(1,2,3,4,5)
mean(x)
```

```
## [1] 3
```

In the example above, we created a vector (a dataset with one column) of numbers by listing within c(…). We also assigned this vector a name, x. We won't create too many vectors in this course, but this is how to do it. Try switching the numbers in between the parentheses above and calculating a mean for that dataset.

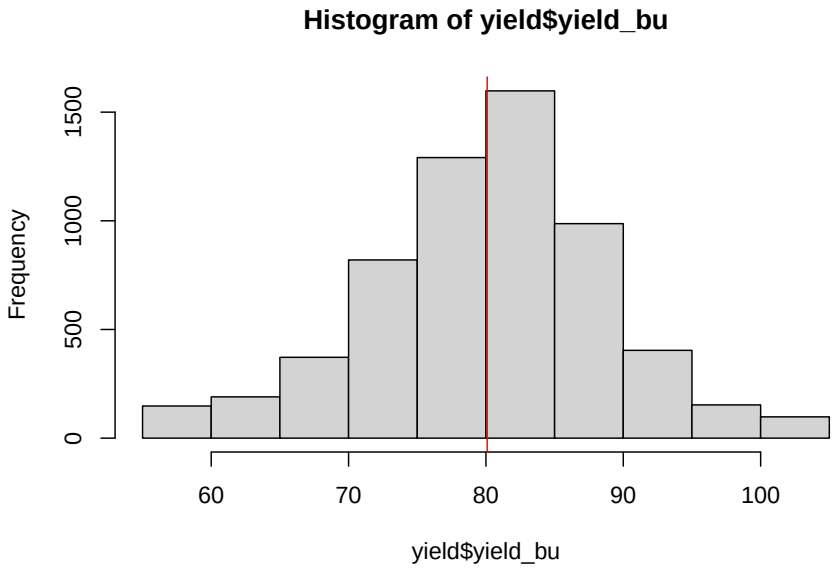
Let's calculate the mean yield for our field.

```
mean(yield$yield_bu)
```

```
## [1] 80.09084
```

The mean is about 95 bushels per acre. Where does this fall on our histogram? After drawing the histogram, we can use additional code to add a reference line:

```
hist(yield$yield_bu, breaks=10)
abline(v = mean(yield$yield_bu), col="red")
```



We use "abline" to tell R to draw a line on our plot. "v" tells it to draw a vertical line, "mean(yield$yield_bu) tells it the X position for this vertical line is equal to the mean value of yield_bu, and"col", you guessed it, specifies the line color. Try changing it to"darkblue".

Earlier, you were introduced to the median. As discussed, the median is a number such that half of the individuals in the population are greater and half are less. If we have an odd number of individuals, the median is the "middle" number as the individuals are ranked from greatest to least.

```
median(c(1,2,3,4,5))
```

```
## [1] 3
```

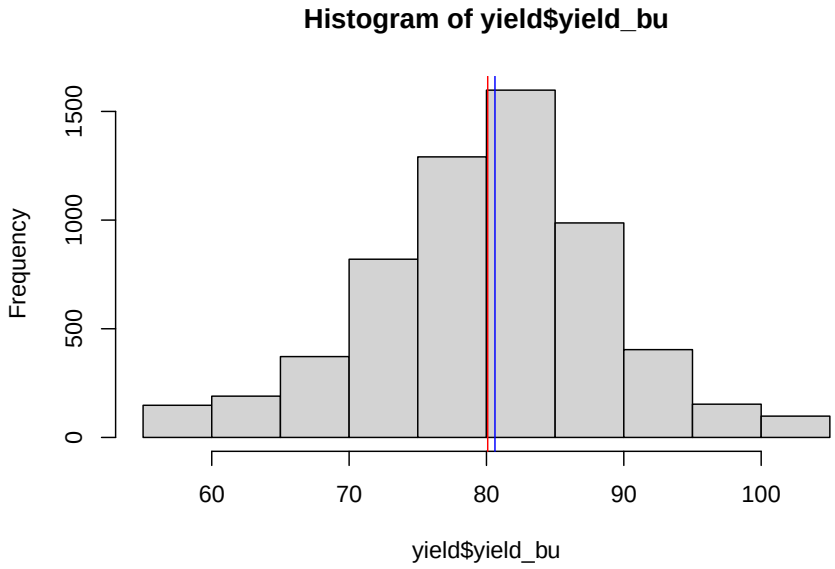
If we have an even number of measures, the median is the average of the middle two individuals in the ranking:

```
median(c(1,2,3,4,5,6))
```

```
## [1] 3.5
```

Let's add a second line representing the median.

```
hist(yield$yield_bu, breaks=10)
abline(v = mean(yield$yield_bu), col="red")
abline(v = median(yield$yield_bu), col="blue")
```



As you can see, they are practically identical. When the mean and median are similar, the number of individuals measuring greater and less than the mean are roughly equivalent. In this case, our data can be represented using a model called the normal distribution.

We also need a statistic that tells us how wide to draw the curve. That statistic is called a measure of dispersion, and we will learn about it next.

1.5.0.5 Measures of Dispersion

To describe the spread of a population, we use three related statistics, which describe the summed difference of all from the mean (sum of squares), the average difference from the mean (variance) and, finally, the difference from the mean in our original units of measurement (standard deviation). Although there is a little math involved in these three statistics, please make yourself comfortable with their concepts because they are *very* important in this course. Almost every statistical test we will learn during the next 14 weeks is rooted in these measures of population width.

1.5.0.5.1 Sum of Squares The first measure of population width is the Sum of Squares. This is, simply, the sum of the squared differences between each observation and the mean. The sum of squares of a measurement x is:

$$S_{xx} = (x_i - \mu)^2$$

Where again x_i is the value x of the ith individual in the population and μ is the true mean of a population.

Why do we square the differences between the observations and means? Simply, if we were to add the unsquared differences they would add to exactly zero. Let's prove this to ourselves. Let's again use the set of numbers (1, 2, 3, 4, 5). We can measure the distance of each individual from the mean by subtracting the mean from it. This difference is called the residual.

```
individuals = c(1,2,3,4,5)
residuals = individuals - mean(individuals)
residuals
```

```
## [1] -2 -1  0  1  2
```

In the first line of code, we defined our individuals as having values of 1, 2, 3, 4, and 5. In the second line of code, we subtracted their mean from each of us. We can see the output – residuals – consist of -2, -1, 0, -1, and -2.

And if we sum these residuals we get zero.

```
sum(residuals)
```

```
## [1] 0
```

Let's now do this with our field data. The number of residuals (almost 6800) is too many to visualize at once, so we can use the sample function to pick 20 at random.

```
yield_residuals = yield$yield_bu - mean(yield$yield_bu)
sample(yield_residuals, 20)
```

```
##  [1]  -3.0179770  13.5059423   2.2998291 -11.2649522   5.6249362  -2.4727784
##  [7]  -1.2592057 -24.6028142   0.3710444  -6.0747732  -9.4139840  -8.5517928
## [13]   2.5625910   2.6459248   4.6420210  -2.3274396  19.9763975  -6.5450104
## [19] -12.0833942   3.1894613
```

If we sum up all the yield residuals, what number do we get? Any guesses before you click "Run Code"?

```
sum(yield_residuals)
```

```
## [1] 1.139e-11
```

Not exactly zero, but a very, very small number – a 4 with 10 zeros in front of it. The difference from zero is the result of rounding errors during the calculation.

The sum of squares is calculated by squaring each residual and then summing the residuals. For our example using the set (1, 2, 3, 4, 5):

```
individuals = c(1,2,3,4,5)
residuals = individuals - mean(individuals)
squared_residuals = residuals^2
sum(squared_residuals)
```

```
## [1] 10
```

And for our yield data:

```
yield_residuals = yield$yield_bu - mean(yield$yield_bu)
squared_yield_residuals = yield_residuals^2
yield_sum_of_squared_residuals = sum(squared_yield_residuals)
yield_sum_of_squared_residuals
```

```
## [1] 461211.3
```

1.5.0.5.2 Variance The sum of squares helps quantify spread: the larger the sum of squares, the greater the spread of observations around the population mean. There is one issue with the sum of squares, though: since the sum of square is derived from the differences between each observation and the mean, it is also related to the number of individuals overall in our population. In our example above, the sum of squares was 10:

```
individuals = c(1,2,3,4,5)
residuals = individuals - mean(individuals)
squared_residuals = residuals^2
sum(squared_residuals)
```

```
## [1] 10
```

Change the first line of code above so it reads "individuals = c(1,2,3,4,5,1,2,3,4,5). You will notice the sum of squares increases to 20. The spread of the data did not change, only the number of observations. Given any distribution, the sum of squares will always increase with the number of observations.

If we want to compare the spread of two different populations with different numbers of individuals, we need to adjust our interpretation to allow for the number of observations. We do this by dividing the sum of squares, S_{xx} by its associated degrees of freedom, $n-1$. In essense, we are calculating an "average" of the sum of squares. This value is the variance, σ^2.

$$\sigma^2 = \frac{S_{xx}}{n}$$

We can calculate the variance as follows.

```
individuals = c(1,2,3,4,5)
residuals = individuals - mean(individuals)
squared_residuals = residuals^2
sum_squared_residuals = sum(squared_residuals)
variance = sum_squared_residuals / length(residuals)
variance
```

```
## [1] 2
```

We introduced a new function above, "length". Length gives us the number of individuals in any dataset, in this case 5.

By dividing the sum of squares by the number of individuals, we "average" our sum of squares. The variance of this dataset is 2.

Change the first line of code above so it reads "individuals = c(1,2,3,4,5,1,2,3,4,5) and run the code again. The variance will now divide the sum of squares by 10. The variance will again be 2.

Our yield variance is:

```
yield_variance = yield_sum_of_squared_residuals/length(yield_residuals)
```

1.5.0.5.3 Standard Deviation Our challenge in using the variance to describe population spread is it's units are not intuitive. When we square the measure we also square the units of measure. For example the variance of our yield is measured in units of bushels^2. Wrap your head around that one. Our solution is to report our final estimate of population spread in the original units of measure. To do this, we calculate the square root of the variance. This statistic is called the standard deviation, σ.

$$\sigma = \sqrt{(\sigma^2)}$$

```
sqrt(variance)
```

```
## [1] 1.414214
```

The standard deviation of our yield data is:

```
sqrt(yield_variance)
```

```
## [1] 8.72324
```

That is enough theory for this first week of class. The remainder of this lesson will focus on introducing you to RStudioCloud.

Chapter 2

Distributions and Probability"

2.1 Review

In this unit we will continue with normal distribution model introduced in the previous unit. As you will recall, the normal distribution is a symmetrical curve that represents the frequency with which individuals with different particular measured values occur in a population.

The peak of the curve is located at the population mean, μ. The width of the curve reflects how spread out other individuals are from the mean. We learned three ways to measure this spread: the sum of squares, the variance, and the standard deviation. These statistics can roughly be thought of as representing the sums of the squared differences, the average of the squared distances, and the distances presented in the original units of measure. These four statistics – mean, sum of squares, variance, and standard deviation – are among the most important you will learn in this course.

2.2 Case Study

This week, we will continue to work with the Iowa soybean yield dataset introduced to us in Unit 1.

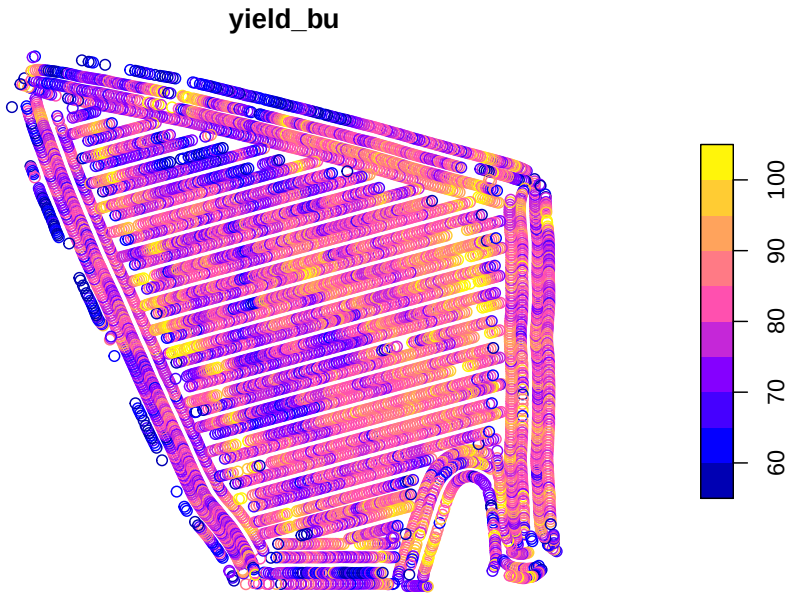
Let's review the structure of this dataset:

```
head(yield)
```

```
## Simple feature collection with 6 features and 12 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: -93.15033 ymin: 41.66641 xmax: -93.15026 ymax: 41.66644
## geographic CRS: WGS 84
##    DISTANCE SWATHWIDTH VRYIELDVOL Crop  WetMass Moisture                Time
## 1 0.9202733          5   57.38461  174 3443.652     0.00 9/19/2016 4:45:46 PM
## 2 2.6919269          5   55.88097  174 3353.411     0.00 9/19/2016 4:45:48 PM
## 3 2.6263101          5   80.83788  174 4851.075     0.00 9/19/2016 4:45:49 PM
## 4 2.7575437          5   71.76773  174 4306.777     6.22 9/19/2016 4:45:51 PM
## 5 2.3966513          5   91.03274  174 5462.851    12.22 9/19/2016 4:45:54 PM
## 6 3.1840529          5   65.59037  174 3951.056    13.33 9/19/2016 4:45:55 PM
##    Heading VARIETY Elevation                 IsoTime yield_bu
## 1 300.1584   23A42  786.8470 2016-09-19T16:45:46.001Z 65.97034
## 2 303.6084   23A42  786.6140 2016-09-19T16:45:48.004Z 64.24158
## 3 304.3084   23A42  786.1416 2016-09-19T16:45:49.007Z 92.93246
## 4 306.2084   23A42  785.7381 2016-09-19T16:45:51.002Z 77.37348
## 5 309.2284   23A42  785.5937 2016-09-19T16:45:54.002Z 91.86380
## 6 309.7584   23A42  785.7512 2016-09-19T16:45:55.005Z 65.60115
##                     geometry
## 1 POINT (-93.15026 41.66641)
## 2 POINT (-93.15028 41.66641)
## 3 POINT (-93.15028 41.66642)
## 4  POINT (-93.1503 41.66642)
## 5 POINT (-93.15032 41.66644)
## 6 POINT (-93.15033 41.66644)
```

And map the field:

```
library(sf)
plot(yield["yield_bu"])
```



2.3 The Normal Distribution Model

Mean, sum of squares, variance, and standard deviation are so important because they allow us to reconstruct the normal distribution model. Before we go further, what is a model? Is it frightening we are using that term in the second week of class?

A model is a simplified representation of reality. No, that doesn't mean that models are "fake" (SMH). It means that they summarize aspects of data, both measured and predicted. The normal distribution model describes the relationship between the values of individuals and how frequently they appear in the population. It's value is it can be reconstructed by knowing just two things about the original dataset – its mean and its standard deviation.

2.3.1 The Bell Curve

The normal distribution is also referred to as the "bell curve", since it is taller in the middle and flared on either side. This shape reflects the tendency of measures within many populations to occur more frequently near the population mean than far from it. Why does this occur and how do we know this?

As agronomists, we can reflect on what it takes to produce a very good – or very bad crop. For a very good crop, many factors need to coincide: temperature, precipitation, soil texture, nitrogen mineralization, proper seed singulation

(spacing), pest control, and hybrid or variety selection, to name a few. We may, in a typical season or within a field, optimize one or two of these conditions, but the possibility of optimizing every one is exceedingly rare. Thus, if we are measuring yield, measures near the mean yield will occur more frequently. Extremely high yields will occur less frequently.

Conversely, very low yields require we manage a crop very badly or that catastrophic weather conditions occur: a hailstorm, flood, or tornado. A frost at exactly the wrong time during seed germination or a drought. A planter box running out of seed or a fertilizer nozzle jamming. These things do occur, but less frequently.

The distribution of individuals around the mean is also a the result of measurement inaccuracies. Carl Friedrich Gauss, who introduced the normal distribution model, showed that it explained the variation in his repeated measurements of the position of stars in the sky. All measurements of continuous data (those that can be measured with a ruler, a scale, a gradated cylinder, or machine) have variation – we use the term "accuracy" to explain their variation around the population mean.

2.3.2 Distribution and Probability

Some areas of mathematics like geometry and algebra produce proven concepts (ie theorems), the product of statistics are almost always probabilities. To be more specific, most of the statistical tests we will learn can be reduced to the probability that a particular value is observed in a population. These probabilities include: - the probability an individual measurement or population mean is observed (normal distribution) - the difference between two treatments is not zero (t-test and least-significant difference) - the probability of one measure given another (regression) - the probability that the spread of individual measures in a population is better predicted by treatment differences than random variation (F-test and analysis of variance)

These probabilities are all calculated from distributions – the frequency with which individuals appear in a population. Another way of stating this is that probability is the proportion of individuals in a population that are have measured values within a given range. Examples could include: - the proportion of individual soybean yield measurements, within one field, that are less than 65 bushels per acre - the proportion of individual corn fields that have an an average less than 160 bushels per acre - the proportion of trials in which the difference between two treatments was greater than zero - the proportion of observations in which the actual crop yield is greater than that predicted from a regression model

2.3.3 Probability and the Normal Distribution Curve

Probability can be calculated as the proportion of the area underneath the normal distribution that corresponds to a particular range of values. We can visualize this, but first we need to construct the normal distribution curve for our soybean field.

We need just two data to construct our curve: the mean and standard deviation of our yield. These are both easily calculated in R.

```
library(muStat)
yield_mean = mean(yield$yield_bu)
yield_sd = stdev(yield$yield_bu, unbiased = FALSE)

# to see the value of yield_mean and yield_sd, we just type their names and run the code.
yield_mean
```

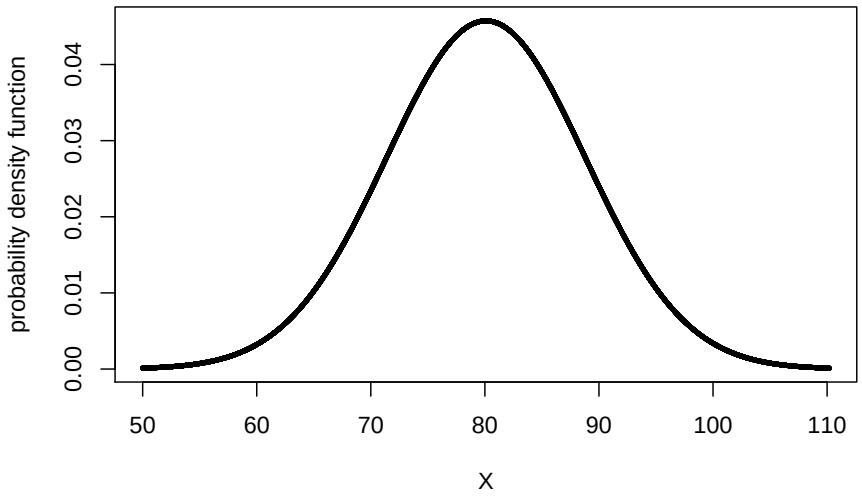
```
## [1] 80.09084
```

```
yield_sd
```

```
## [1] 8.72252
```

Now we can build our curve with a single line of code. The plotDist function in R will reconstruct the distribution curve for any population, given three arguments, in the following order: what kind of curve to draw, the population mean, and population standard deviation.
- *dnorm* tells plotDist to draw a normal distribution curve. This needs to enclosed in quotes - *yield_mean* tells plotDist to use the mean we calculated in the previous chunk as the population mean - *yield_sd* tells plotDist to use the standard deviation we calculated in the previous chunk as the standard deviation

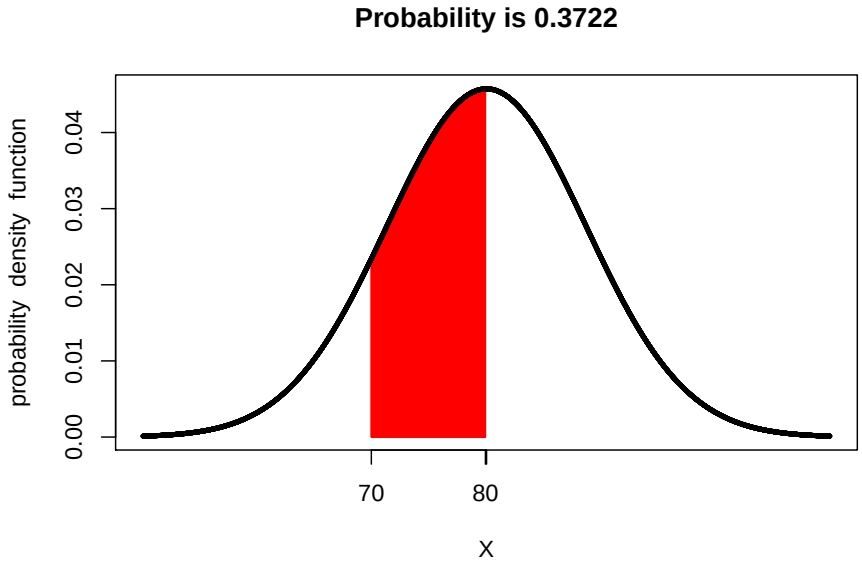
```
library(fastGraph)
plotDist("dnorm", yield_mean, yield_sd)
```



Let's now shade the area underneath the normal curve corresponding to X values from 70 - 80. This area will represent the proportion of the population Where individuals were measured to have values between 70 and 80 bushels.

The shade.norm function makes effort painless. There are five arguments in this function: - *xshade* is the range of individuals in our population for which we want to calculate their area under the distribution curve. We pass this range as a vector using c(minimum, maximum), in this case c(70,80) - *ddist* tells the function what kind of distribution curve to draw. "dnorm" tells it to construct a normal distribution curve - the third value we include (and it must be in this order), is the population mean. In this case, we pass the variable yield_mean whose value we calculated in the previously - the fourth value is the population standard deviation, which we have already calculated and assigned to the variable yield_sd - *lower.tail=FALSE* tells shade.norm to shade the area under the curve between 70 and 80 bushels and to calculate its area.

```
shadeDist(xshade=c(70,80), ddist = "dnorm", yield_mean, yield_sd, lower.tail = FALSE)
```



Pretty cool, huh? The red area is the proportion of the soybean yield population that was between 70 and 80 bushels. shadeDist has also presented us with the proportion of the curve represented by that area, which it has labelled "Probability". The probabiliity in this case is 0.3722. What does that number mean?

The total area of the curve is 1. The proportion of the area under the curve that corresponds with yields from 70 to 80 bushels, then, is 3.967 percent of the area. This means that 37.22 percent of the individuals in our yield population had values from 70 and 80 bushels

But wait a second – why is R using the term "Probability"? Think of it this way. Imagine you sampled 1000 individuals from our population. If 37.22 percent of our individuals have values from 70 to 80 bushels, then about 37% of the individuals in your sample should have values from 70 to 80 bushels. In other words, there is a 37% probability that any individual in the population will have a value from 70 to 80 bushels.

Let's test this. Let's randomly sample 1000 individuals from our sample. Then lets count the number of individuals that have yields between 70 and 80 bushels. We will run three lines of code here: - we will use the "sample" function to randomly sample our population and return a vector of those numbers - we will use the "subset" function to subset our data into those that meet logical conditions and return a dataframe or vector. We provide two arguments: first, the name of the dataset or vector, and second the conditions. - we will use the "length" function to count the number of observations

```
yield_sample = sample(yield$yield_bu, 1000)

# "yield_sample >=70 & yield_sample <=80 tells it to only include measures from 70 to
yield_subset = subset(yield_sample, yield_sample >=70 & yield_sample <=80)

length(yield_subset)
```

```
## [1] 344
```

Is the proportion predicted by the normal distribution curve exactly that of the actual population? Probably not. The normal distribution curve is, after all, a model – it is an approximation of the actual population.

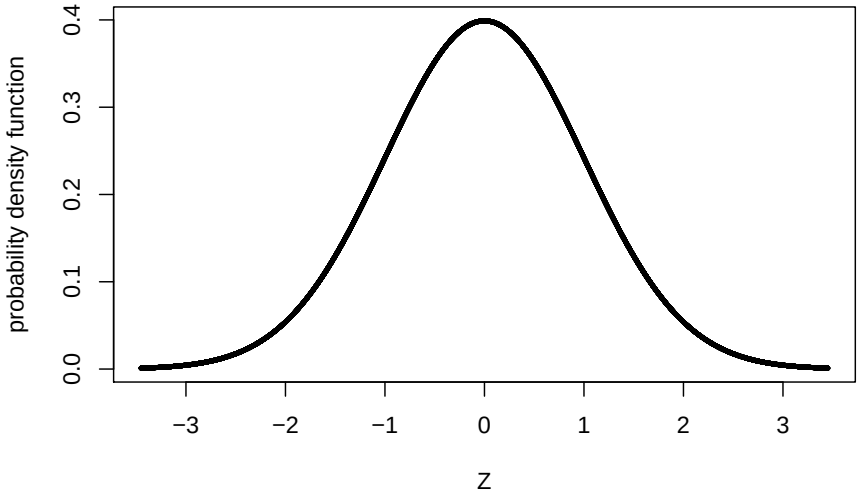
Run the code above multiple times and observe how the percentages change. The proportions will vary slightly but generall stay in a range from about 0.32 to 0.37.

We will talk more about sampling in the next unit.

2.4 The Z-Distribution

The relationship between probability and the normal distribution curve is based on the concept of the Z-distribution. In essence, Z-distribution describes a normal distribution curve with a population mean of 0 and a standard deviation of 1.

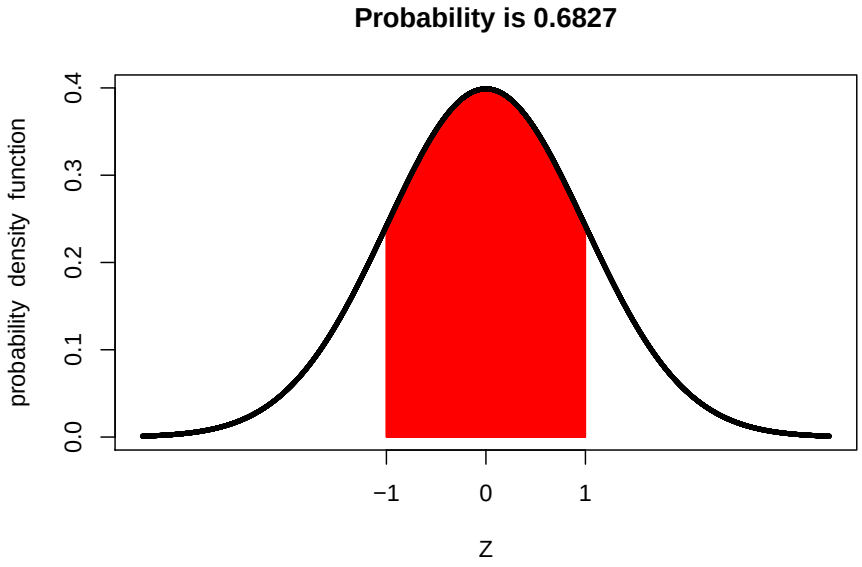
```
plotDist("dnorm", 0, 1)
```



The Z-distribution helps us understand how probability relates to standard deviation, regardless of the nature of a study or its measurement units.

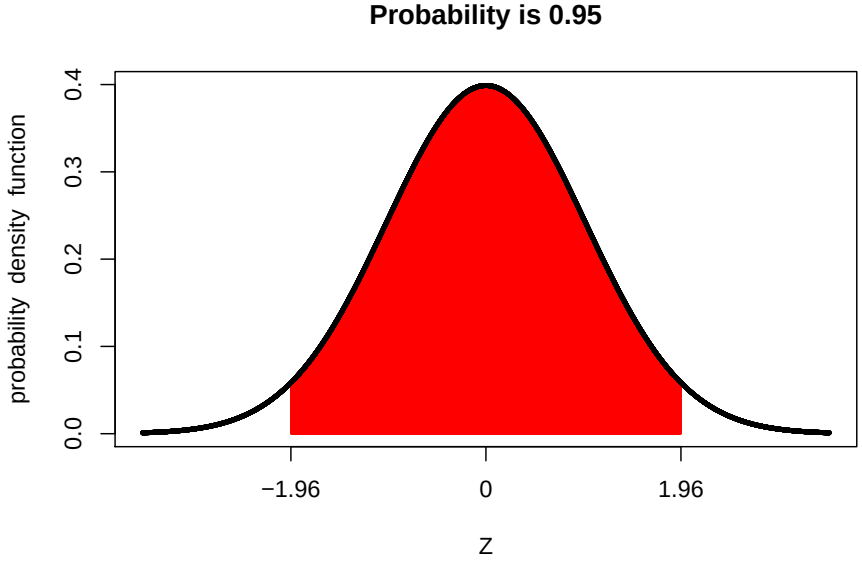
For example, the proportion of a population within one standard deviation of the mean is about 68 percent:

```
shadeDist(xshade=c(-1, 1), ddist = "dnorm", 0, 1, lower.tail = FALSE)
```



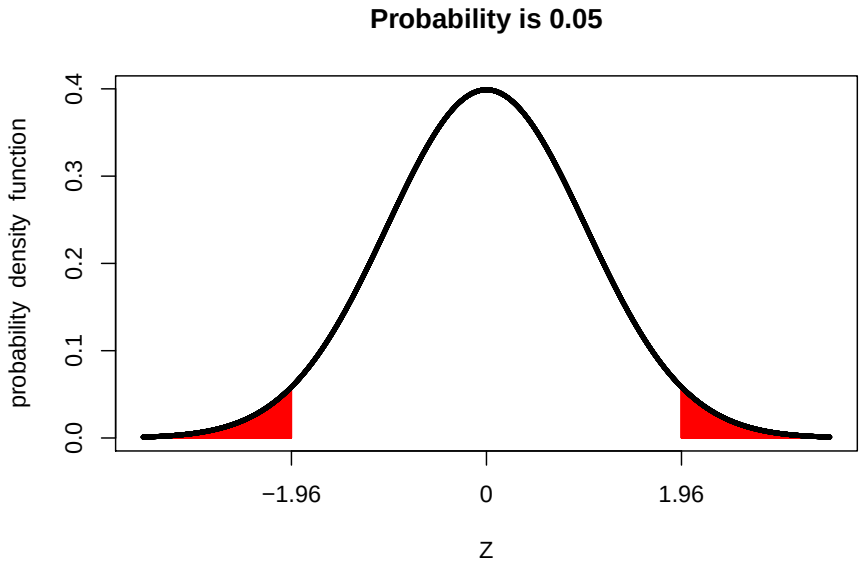
Similarly, the proportion of a population within 1.96 standard deviations of the mean is about 95 percent:

```
shadeDist(xshade=c(-1.96, 1.96), ddist = "dnorm", 0, 1, lower.tail = FALSE)
```



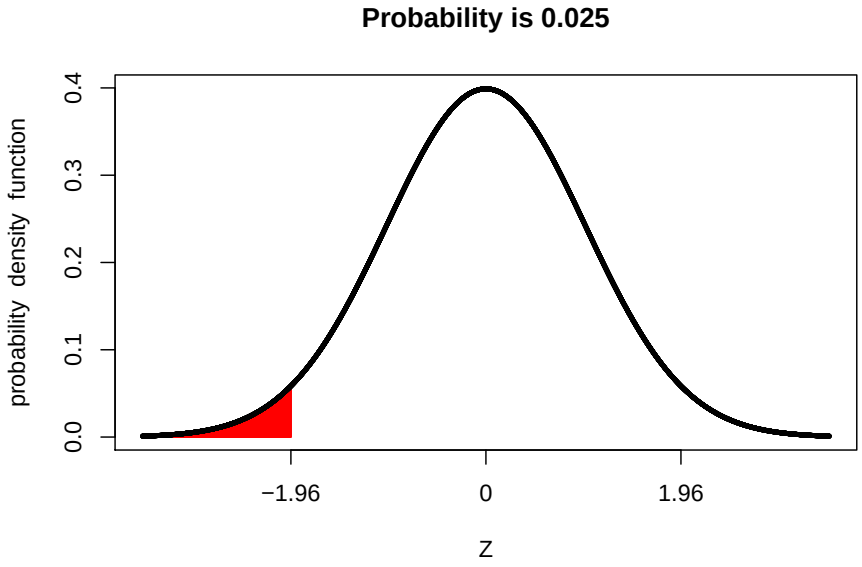
Conversely, the proportion of a population beyond 1.96 standard deviations from the mean is about 5 percent. We can visualize this by changing the last argument of our code to "lower.tail = TRUE".

```
shadeDist(xshade=c(-1.96, 1.96), ddist = "dnorm", 0, 1, lower.tail = TRUE)
```



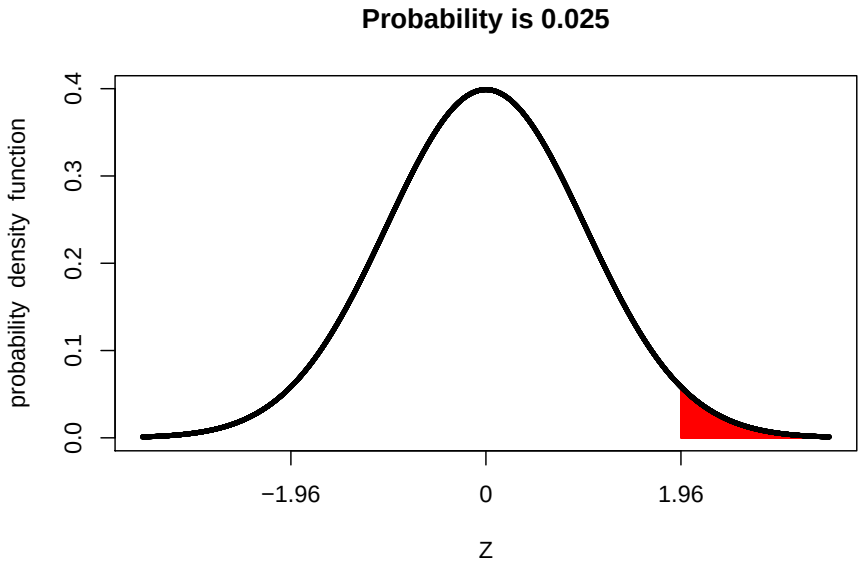
We refer to the upper and lower ends of the distribution as tails. In a normal distribution we would expect about 2.5% of observations to less than -1.96 standard deviations of the mean. We can measure the proportion in one tail by changing our argument in the shadeDist command to "xshade = -1.96".

```
shadeDist(xshade=-1.96, ddist = "dnorm", 0, 1, lower.tail = TRUE)
```



And 2.5% of the population to be more than +1.96 above the mean:

```
shadeDist(xshade=1.96, ddist = "dnorm", 0, 1, lower.tail = FALSE)
```



Notice we changed the last argument back to "lower.tail = TRUE". The causes

R to shade the area above 1.96 standard deviations of the mean.

2.4.1 Important Numbers: 95% and 5%

Above we learned that 95% of a normal distribution is between 1.96 standard deviations of the mean, and that 5% of a normal distribution is outside this range. Perhaps these numbers sound familiar to you. Have you ever seen results presented with a 95% confidence interval? Have you ever read that two treatments were significantly different at the P=0.05 level?

For population statistics, the normal distribution is the origin of those numbers. As we get further into this course, we will learn about additional distributions – binomial, chi-square, and F – and the unique statistical tests they allow. But the concept will stay the same: identifying whether observed statistical values are more likely to occur (i.e., within the central 95% of values expected in a distribution), or whether the values are unusual (occurring in the remaining 5%).

[insert shiny app where they can sample multiple times and see results]

```
set.seed(051520)

output = matrix(ncol=1, nrow=100)

for(i in c(1:100)){
  yield_sample = sample(yield$yield_bu, 1000)
  N = length(yield_sample[yield_sample >=80 & yield_sample <=100])
  output[i] = N
}
```