**Book conventions**

**A white text with black text

Description automatically generated**

**A group of animals with text

Description automatically generated with medium confidence**

**Vinculum (repeating bar)**

A vinculum can indicate the repetend of a repeating decimal value:



 = 3.3333333333333…

In other words it indicates that the digits or fractions are repeating infinitely.

**Order of operations**

**PEMDAS**: parentheses, exponents, multiplication, division, addition, and subtraction

**Expressing dependent and independent variables in a function**

You can use *y = 2x + 1* or *f(x) = 2x + 1* to express the dependent variable *y* as a function of *x*.

If you have *two independent variables* you write the function like *f(x, y) = 2x + 3y*. In this function two independent variables *x* and *y* and one dependent variable the output of *f(x, y)*. The graph of this function will be plotted on three dimensions to produce a plane of values rather than a line.

**Cartesian plane, x-y plane, coordinate plane**

When we plot on a two-dimensional plane with two number lines (one for each variable) it is known as a *Cartesian plane*, *x-y plane*, or *coordinate plane*. По-русски: *прямоугольная система координат* или *Декартова система*. We trace a given x-value and then look up the corresponding y-value, and plot the intersections

as a line.

Example of a Cartesian plane:

A graph of a line in a graph

Description automatically generated

Notice that due to the nature of real numbers (or decimals, if you prefer), there are an infinite number of x values. This is why when we plot the function f(x) we get a continuous line with no breaks in it. There are an infinite number of points on that line, or any part of that line.

**Straight-line function / straight-line graph**

This is a straight-line *f(x) = 2x + 1*:

A graph of a function

Description automatically generated

**Parabola function / parabola graph**

This is a parabola *f(x) = x2 + 1*:

A graph of a function

Description automatically generated

It is continuous but not linear. It does not produce values in a straight line. When a function is continuous but curvy, rather than linear and straight, we call it a *curvilinear function*.

**Summation symbol sigma Σ**

If I want to iterate the numbers 1 through 5, multiply each by 2, and sum them, here is how I would express that using a summation

A number with numbers and a plus two

Description automatically generated with medium confidence





A close-up of a white background

Description automatically generated

**Exponents**

23 2 is the base and 3 is the exponent (number of times 2 should be multiplied by 2: 2 \* 2 \* 2)

A white paper with black text

Description automatically generated

When we multiply exponents together with the same base, we simply add the exponents, which is known as the *product rule*:



Division:

A white background with black text

Description automatically generated

A math equations with numbers and symbols

Description automatically generated with medium confidence

A white paper with black text

Description automatically generated

A black and white image of a mathematical equation

Description automatically generated

**Roots**

A  asks “What number multiplied by itself will give me 4?” which of course is 2. Note here that 41/2 is the same as 4:

41/2 =  = 2

Cubed roots are similar to square roots, but they seek a number multiplied by itself three times to give a result. A cubed root of 8 is expressed as  and asks “What number multiplied by itself three times gives me 8?” This number would be 2 because 2 \* 2 \* 2 = 8.

A math equation with numbers and equations

Description automatically generated with medium confidencepower rule

A number and equal sign

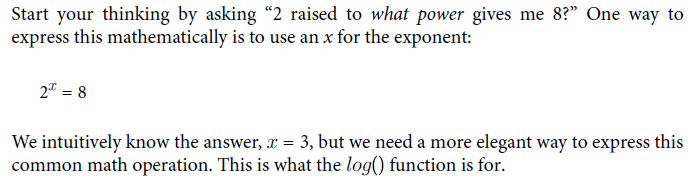
Description automatically generated with medium confidence

A black and white image of a mathematical equation

Description automatically generated

**Logarithms *log()***

A logarithm is a math function that finds a power for a specific number and base.





A math equations on a white background

Description automatically generated

Algebraically speaking, this is a way of isolating the *x*, which is important to solve for *x*.

**Properties for exponents and logarithms**

A math equations on a white background

Description automatically generated

**Natural logarithms *log() ln()***

When we use *e* as our base for a logarithm, we call it a *natural logarithm*.



However, in Python, a natural logarithm is specified by the log() function. As discussed earlier, the default base for the log() function is *e*. Just leave the second argument for the base empty and it will default to using *e* as the base

**Function that approaches 0 but never reaches 0 *lim***

A graph of a function

Description automatically generated

A white text with black text

Description automatically generated

**Derivatives**



A math equations with numbers

Description automatically generated with medium confidence

**Partial derivatives**

A math equations with numbers

Description automatically generated with medium confidence

**The chain rule**

Finding the derivative of *z* with respect to *x*:

A math equations with numbers and symbols

Description automatically generated A math equation with numbers and symbols

Description automatically generated A number and a plus one

Description automatically generated with medium confidence A mathematical equation with numbers

Description automatically generated with medium confidence

This is the chain rule, which says that for a given function *y* (with input variable *x*) composed into another function *z* (with input variable *y*), we can find the derivative of *z* with respect to *x* by multiplying the two respective derivatives together:

A mathematical equation with black text

Description automatically generated

The chain rule is a key part of training a neural network with the proper weights and biases. Rather than untangle the derivative of each node in a nested onion fashion, we can multiply the derivatives across each node instead, which is mathematically a lot easier.

**Integrals**

The opposite of a derivative is an integral, which finds the area under the curve for a given range.

Packing rectangles under a curve to approximate area:

A graph on a grid

Description automatically generated

**Riemann sums**

Khan Academy has a great [article](https://www.khanacademy.org/math/ap-calculus-ab/ab-integration-new/ab-6-3/a/definite-integral-as-the-limit-of-a-riemann-sum) explaining how to use limits for Reimann Sums.

**Probability**

Probability is how strongly we believe an event will happen, often expressed as a percentage.

A black text on a white background

Description automatically generated

**Probability** is about quantifying predictions of events yet to happen, whereas **likelihood** is measuring the frequency of events that already occurred. In statistics and machine learning, we often use **likelihood** (the past) in the form of data to predict **probability** (the future).

A close-up of a person's face

Description automatically generated

A white background with black text

Description automatically generated

Convert probability into odds:

A math equations with numbers

Description automatically generated with medium confidence

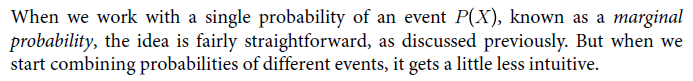
A close up of a text

Description automatically generated

Probability is purely theoretical of how likely an event is to happen and does not require data. Statistics, on the other hand, cannot exist without data and uses it to discover probability and provides tools to describe data.

Think of predicting the outcome of rolling a 4 on a die (that’s the singular of dice). Approaching the problem with a pure probability mindset, one simply says there are six sides on a die. We assume each side is equally likely, so the probability of getting a 4 is 1/6, or 16.666%. However, a zealous statistician might say, “No! We need to roll the die to get data. If we can get 30 rolls or more, and the more rolls we do the better, only then will we have data to determine the probability of getting a 4.” This approach may seem silly if we assume the die is fair, but what if it’s not? If that’s the case, collecting data is the only way to discover the probability of rolling a 4.

**Marginal probability**



**Joint probability**

A text on a white background

Description automatically generated

A math problem with numbers and equations

Description automatically generated

A close-up of text

Description automatically generated

A math problem with numbers and equations

Description automatically generated with medium confidence

**Union probability**

A white paper with black text

Description automatically generated

A close up of text

Description automatically generated

A math problem with numbers and equations

Description automatically generated with medium confidence

A math problem with numbers and equations

Description automatically generated

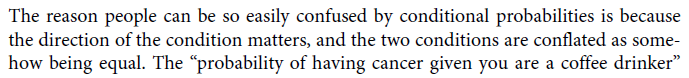
A close up of text

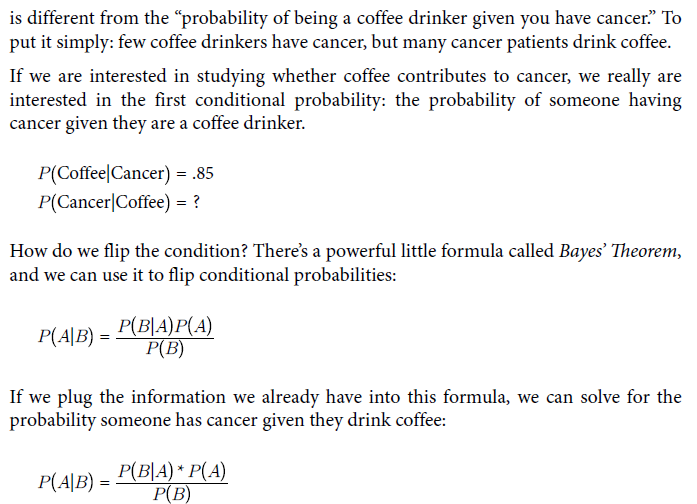
Description automatically generated

**Conditional Probability and Bayes’ Theorem**

A white text with black text

Description automatically generated





A mathematical equation with numbers and lines

Description automatically generated

**Joint and union conditional probabilities**

A math problem with numbers

Description automatically generated with medium confidence

A screenshot of a paper

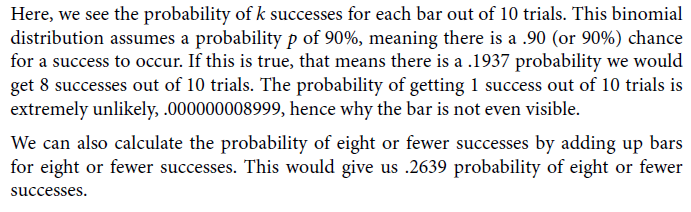
Description automatically generated

**Binomial Distribution**

Binomial distribution measures how likely *k* successes can happen out of *n* trials given *p* probability. Visually it looks like:

A graph of a number of individuals

Description automatically generated



**Beta T-Distribution**

A close up of text

Description automatically generated

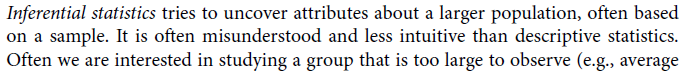
**Descriptive and Inferential Statistics**

**Descriptive statistics: we have data and we summarize it.**

A close up of text

Description automatically generated

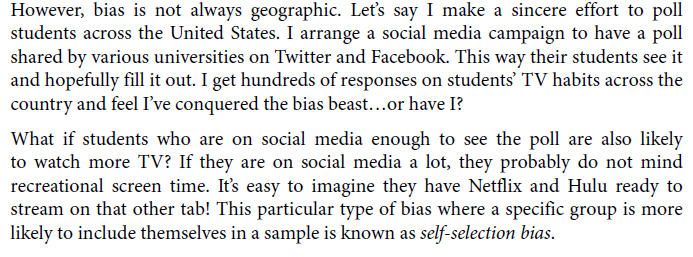
**Inferential statistics: based on a sample look for attributes of a larger population.**



A close up of black text

Description automatically generated

**Self-selection bias:**



A close up of text

Description automatically generated

**Confirmation bias:**

A close up of text

Description automatically generated

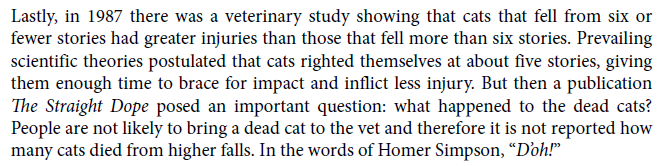
**Survival bias:**

A text on a page

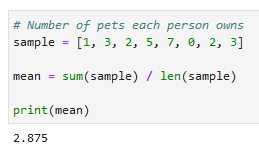
Description automatically generated

A cartoon of a stick figure standing on a table with money bags

Description automatically generated



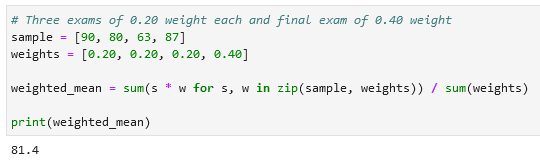
**Mean**

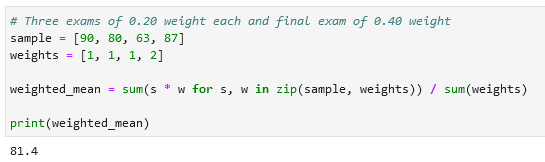


A math equations and numbers

Description automatically generated with medium confidence

**Weighted mean**

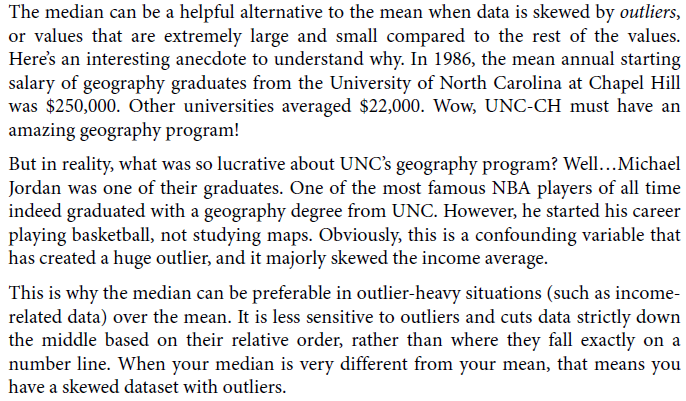




**Median**

A black text on a white background

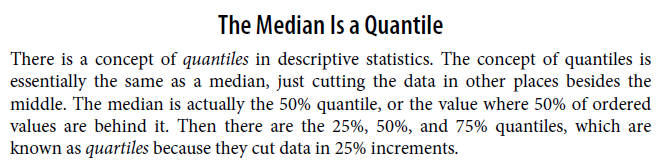
Description automatically generated



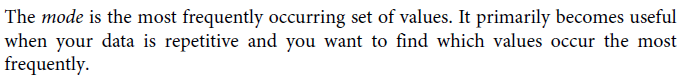
A screenshot of a computer program

Description automatically generated

**Quantiles**



**Mode**





Calculate mode (use functions like from collections import defaultdict(lambda: 0):

A screenshot of a computer code

Description automatically generated

A defaultdict is a dictionary-like object that allows you to set a default value for keys that don't exist in the dictionary. This can be especially useful when you're counting occurrences of elements, like in your mode function.

A screenshot of a computer code

Description automatically generated

**Population variance and standard deviation**

**Population variance**

Variance is a measure of how spread out our data is.

A screenshot of a number of pets

Description automatically generated

A graph of a graph with green dots and black text

Description automatically generated

**A math equations on a white background

Description automatically generated**

**Calculate variance:**

**A screenshot of a computer code

Description automatically generated**

**A close up of text

Description automatically generated**

**Population standard deviation**

**A black text on a white background

Description automatically generated**

**Calculate** *population* **standard deviation:**

**A screenshot of a computer program

Description automatically generated**

**Sample variance and standard deviation**

A white background with black text

Description automatically generated

A black text on a white background

Description automatically generated

**Calculate sample variance and sample standard deviation:**

A screenshot of a computer code

Description automatically generated

Both variance and standard deviation have increased when we treated them as a population and not a sample. This is correct as a sample could be biased and imperfect representing the population. Therefore, we increase the variance (and thus the standard deviation) to increase our estimate of how spread out the values are. A larger variance/standard deviation shows less confidence with a larger range.

**The Normal Distribution**

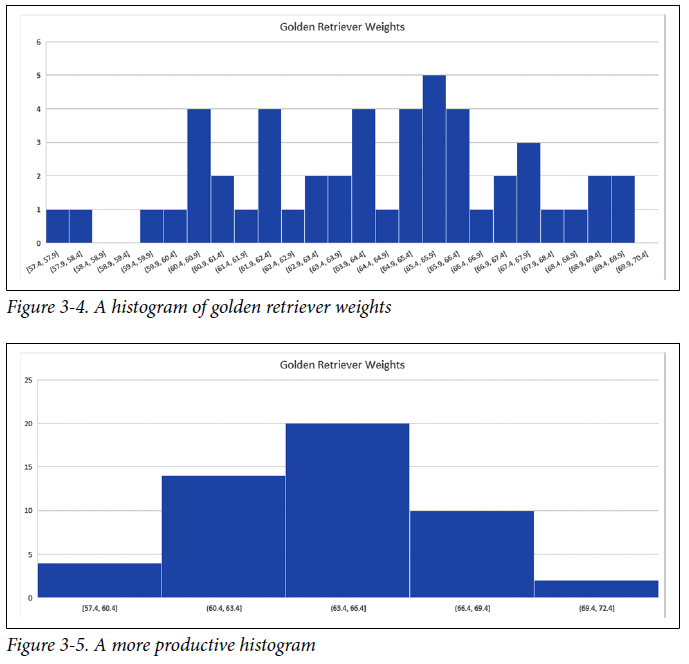
We touched on probability distributions in the last chapter, particularly the binomial distribution and beta distribution. However the most famous distribution of all is the normal distribution. The normal distribution, also known as the Gaussian distribution, is a symmetrical bell-shaped distribution that has most mass around the mean, and its spread is defined as a standard deviation. The “tails” on either side become thinner as you move away from the mean.

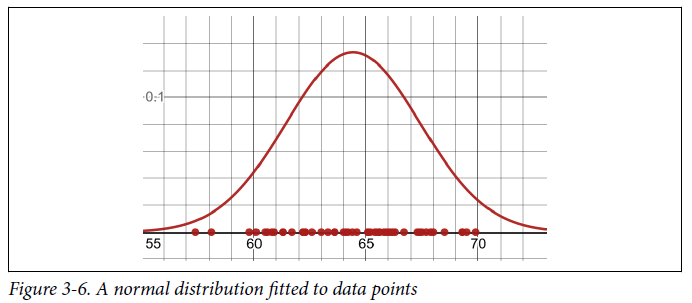
A graph with a red line

Description automatically generated

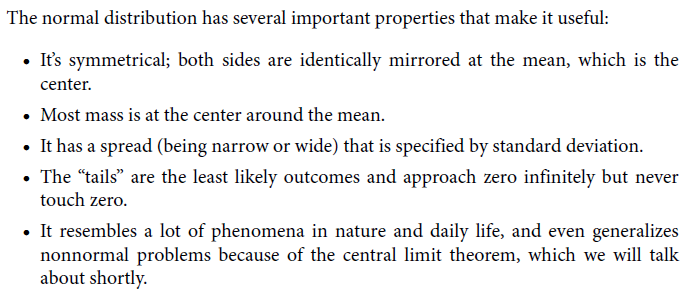
A screenshot of a computer

Description automatically generated

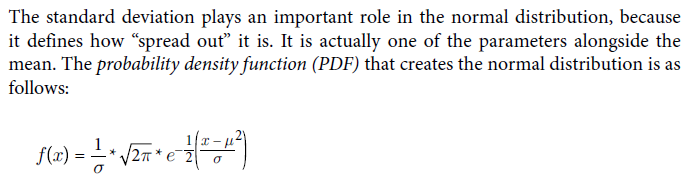




**Properties of a normal distribution**



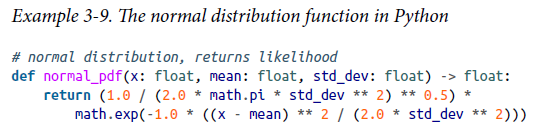
**The Probability Density Function (PDF)**



The correct formula is ([from](https://en.wikipedia.org/wiki/Probability_density_function)):

A black and white math equation

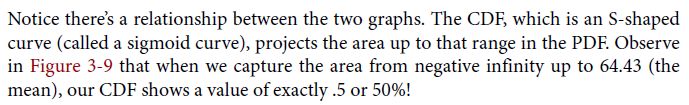
Description automatically generated

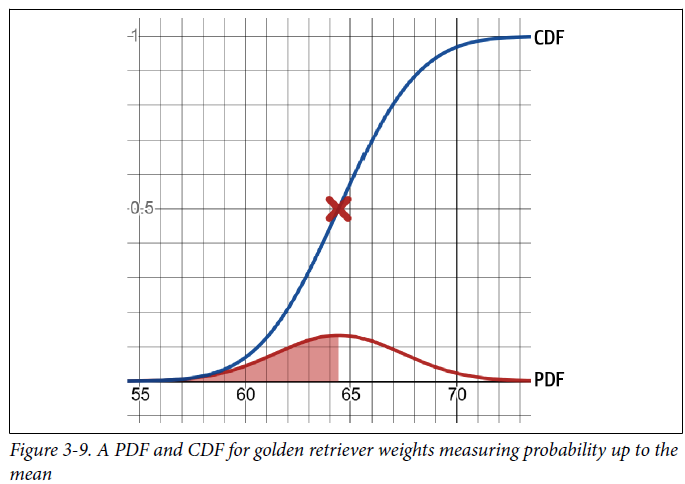


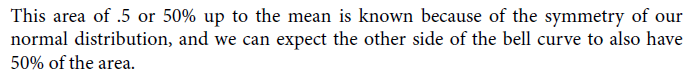
**The Cumulative Distribution Function (CDF)**

A graph with a red line

Description automatically generated



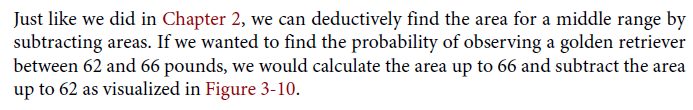




**Calculate the normal distribution CDF in Python using scipy.stats and norm.cdf:**

A screenshot of a computer program

Description automatically generated



A diagram of a function

Description automatically generated

A computer screen shot of a program

Description automatically generated



**The Inverse CDF**

You can use it to find *x*. For example if you want to *find the weight that 95% of golden retrievers fall under*.

A graph with a line

Description automatically generated

Calculate the inverse CDF (or norm.ppf):

A screenshot of a computer

Description automatically generated

You can use inverse CDF to generate random numbers that follow the normal distribution. For example, if you want to generate 10 realistic cat weights you can use this code:

A screenshot of a computer

Description automatically generated

Of course NumPy and other libraries can generate random values off a distribution for you, but this highlights one use case where the inverse CDF is handy.

**Z-Scores**

It is common to rescale a normal distribution so that the mean is 0 and the standard deviation is 1, which is known as the standard normal distribution. This makes it easy to compare the spread of one normal distribution to another normal distribution, even if they have different means and variances.

A black text on a white background

Description automatically generated

A white paper with black text

Description automatically generated

A math problem with numbers and a few words

Description automatically generated with medium confidence

**Calculate the Z-Score in Python:**

A screenshot of a computer

Description automatically generated

**Coefficient of Variation**

A screenshot of a math test

Description automatically generated

**Inferential statistics**

**Central Limit Theorem**

A graph with a blue line

Description automatically generated with medium confidence

A close-up of a text

Description automatically generated

A white text with black text

Description automatically generated

uniform distribution

A graph of a number of blue and white lines

Description automatically generated with medium confidence

A close up of text

Description automatically generated

**How much sample is enough?**

A close-up of a paper

Description automatically generated

**Asymmetrical or multimodal distribution**

Distribution is asymmetrical or multimodal means that it has several peaks rather than one at the mean.

[**The Central Limit Theorem: What’s Large Enough**](https://www.dummies.com/article/academics-the-arts/math/statistics/the-central-limit-theorem-whats-large-enough-142651/)

In a nutshell, the Central Limit Theorem says you can use the normal distribution to describe the behavior of a sample mean even if the individual values that make up the sample mean are not normal themselves. But this is only possible if the sample size is “large enough.” Many statistics textbooks would tell you that n would have to be at least 30.

But why is n = 30 the benchmark? Many variables in nature, finance, and other applications have a distribution that’s very close to the normal curve. For example, by looking at the t-table, you see that the various values of t start to get really close to the values of z by the time you hit about 30 degrees of freedom. One reason for this is that the t-distributions and the normal distribution share two important characteristics: They are symmetric, and they are unimodal (having one peak).

If the distribution of your individual data values is far off from either of these qualities, you might need more than a sample size of 30 to use the Central Limit Theorem. The further away the data is from being symmetric and unimodal, the more data you’ll need.

## **Symmetry**

If you know or suspect that your parent distribution is not symmetric about the mean, then you may need a sample size that’s significantly larger than 30 to get the possible sample means to look normal (and thus use the Central Limit Theorem).

Consider the following right-skewed histogram, which records the number of pets per household.

A graph of a number of household pets

Description automatically generated

Now, suppose it represents the entire population of households. You repeatedly sample *n* = 30 households from that population. Here is what distribution of possible sample means looks like.

A graph of a distribution of a sample

Description automatically generated

You can see that this distribution is not normal because the right tail still stretches out farther from the central peak than the left tail does. It’s not symmetric. For this population, you need to take a sample of around *n* = 100 to get the sample means to settle into a symmetric curve.

A graph of a number of bars

Description automatically generated

## **Unimodal**

If you know or suspect that your parent distribution is not unimodal and has more than one peak, then you might need more than 30 in your sample to feel good about using the Central Limit Theorem.

Consider the following multimodal population histogram with three distinct peaks.

A graph of multiple columns

Description automatically generated

If you only sample *n* = 30 from that population, you do get a unimodal distribution, but it’s still not quite symmetric.

A graph of a distribution of a sample

Description automatically generated

For this population, you need to take a sample of at least *n* = 50 to feel comfortable that your sample mean distribution is roughly normal.

A graph of a distribution of a sample

Description automatically generated

**Confidence Intervals**

A confidence interval is a range calculation showing how confidently we believe a sample mean (or other parameter) falls in a range for the population mean.

**For example:**

* *Based on a* ***sample of 31*** *golden retrievers*
* *with a* ***sample mean of 64.408***
* *and a* ***sample standard deviation*** *of* ***2.05****,*
* *I am* ***95% confident***
* *that the* ***population mean*** *lies* ***between 63.686 and 65.1296****.*

A white text with black text

Description automatically generatedlevel of confidence (LOC)

A graph of a normal distribution

Description automatically generated

A graph of a function

Description automatically generated

**Retrieve a critical z-value in Python:**

A screenshot of a computer program

Description automatically generated

A white paper with black text

Description automatically generatedmargin of error (E), confidence interval

**Calculate confidence intervals in Python (example 1):**

A screenshot of a computer program

Description automatically generated

**Calculate confidence intervals in Python (example 2):**

A screenshot of a computer program

Description automatically generated

A close up of a number

Description automatically generated

A close up of text

Description automatically generated

**P-values**

When we say something is *statistically significant*, what do we mean by that? Technically, it has to do with something called the *p-value*.

A close up of text

Description automatically generated

A text on a page

Description automatically generated

**Hypothesis Testing**

A graph with a red line

Description automatically generated

Calculate the probability of recovering between 15 and 21 days:

A screenshot of a computer program

Description automatically generated

A close-up of text

Description automatically generated

A graph of a drug

Description automatically generated

**One-Tailed Test**



A white background with black text

Description automatically generated

A graph with a red line

Description automatically generated

Calculate the x-value with 5% of area behind it in Python using inverse CDF or norm.ppf:

A screenshot of a computer code

Description automatically generated

A close-up of a white paper

Description automatically generated

A graph of a drug test

Description automatically generated



Calculate the p-value in Python using norm.cdf:

A screenshot of a computer code

Description automatically generated



**Two-Tailed Test**

A text on a white background

Description automatically generated

A graph on a grid

Description automatically generated

Two-tailed test in Python:

A screenshot of a computer code

Description automatically generated

A graph with a pill and a pill on it

Description automatically generated

A close up of text

Description automatically generated

A graph with a pill and a red line

Description automatically generated

Calculating the two-tailed p-value in Python:

A screenshot of a computer

Description automatically generated

A close-up of a white background

Description automatically generated

A screenshot of a test

Description automatically generated

**p-hacking**

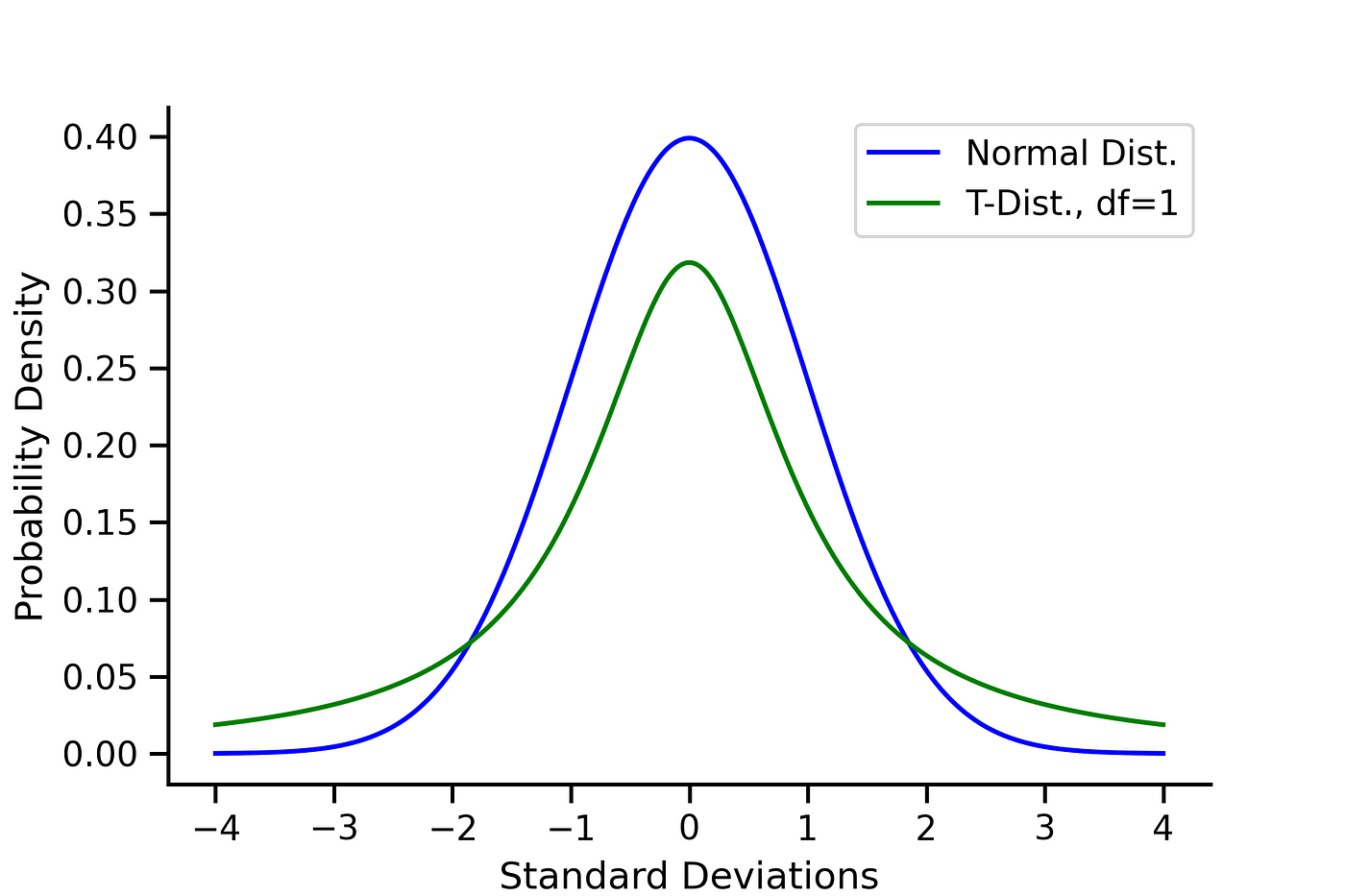
A close up of text

Description automatically generated

**T-Distribution (dealing with small samples)**

A graph of a normal distribution

Description automatically generated



A close up of text

Description automatically generated

A close-up of a text

Description automatically generated

Get a critical value range with a T-distribution in Python using t.ppf:

A screenshot of a computer program

Description automatically generated



A close up of a text

Description automatically generated

**Big Data Considerations**

A close up of text

Description automatically generated