Hypothesis Testing - Introduction

Introduction

In this section, you'll learn about experimental design and hypothesis testing. All scientific research that comes out of universities uses hypothesis testing to determine if the results of an experiment are significant or not. As a data scientist, you might be tasked with designing, performing, and analyzing the results of an experiment. Finally, you'll also learn about resampling methods, which are modern statistical techniques that involve taking repeated subsamples from a sample and help better estimate the precision of your sample statistics.

Hypothesis Testing

In this section, you'll be looking at experimental design, effect size, T-tests, Type 1 and Type 2 errors, and resampling techniques like the jackknife, bootstrap, and permutation tests.

Experimental Design

Without good experimental design, it's very easy to draw the wrong conclusions from your experiments. Because of that, you'll kick this section off by looking at the scientific method and the key elements of good experimental design - forming **alternative** and **null hypotheses**, conducting an experiment, analyzing the results for statistical significance and drawing conclusions.

Effect Size

We then look at how to calculate and interpret the size of the difference between control and test groups. We'll see how the "Effect Size" can be used to communicate the practical significance of experimental results, to perform meta-analyses of multiple studies, and to perform power analysis to determine the number of participants that a study would require to achieve a certain probability of finding a true effect.

One and Two Sample T-tests

Next, you'll also look at t-tests and how they can be used to compare two averages to see how significant the differences are between one or two samples once we have defined the experimental design.

Type 1 and Type 2 Errors

From there, you'll learn about **type 1 (false positive)** and **type 2 (false negative) errors** and the inherent tradeoff between them.

Jackknife, Bootstrap, and Permutation Tests

We'll look at techniques for taking repeated subsamples from a sample using bootstrapping, jackknife and permutation tests to better estimate the precision of your sample statistics or validate models by using random subsets.

Summary

Without a good understanding of experimental design, it's easy to end up confusing spurious correlations for meaningful results or placing too much (or too little) weight on the results of any given test. In this section, we cover a range of tools and techniques to ensure that you design your experiments rigorously and interpret them thoughtfully.

Introduction to Experimental Design

Introduction

In this lesson, you'll learn about the importance of sound experimental design, and how it underpins every decision you will make as a Data Scientist!

Objectives

You will be able to:

- List the steps of the scientific method
- Explain the purpose of control/experimental groups
- List four assumptions for appropriate sampling techniques and sample size
- Compare and explain the importance of different kinds of randomized control trials
- Set up null and alternative hypotheses

The Scientific Method

You probably remember at least a little bit about the ***Scientific Method*** from your time in school. This lesson will focus on the thing that makes it work--sound experimental design! The scientific method has been responsible for all the great progress humanity has seen in everything from medicine to physics to electronics, all because scientists working on problems knew how to design experiments in a way that helped them answer important questions with as little ambiguity as possible. If the scientific method was a car, then experimental design would be the engine that allows that car to move. This is especially important to Data Scientists because it allows them to examine any problem through the lens of the ***Null Hypothesis***!

The general structure of an experiment is as follows:

1. Make an Observation

The first step of the scientific method is to observe something that you want to test. During this step, you must observe phenomena to help refine the question that you want to answer. This might be anything from "does this drug have an effect on headaches?" to "does the color of this button affect the number of sales a website makes in a day?". Before testing these ideas, you need to observe that there might be some phenomena occurring and then come up with a specific question to answer.

2. Examine the Research

Good data scientists work smart before they work hard. In the case of the scientific method, this means seeing what research already exists that may help you answer your question, directly or indirectly. It could be that someone else has already done an experiment that answers your question--if that's the case, you should be aware of that experiment before starting your own, as it could inform your approach to structuring your experiment, or maybe even answer your question outright!

3. Form a Hypothesis

This is the stage that most people remember from learning the scientific method in grade school. In school, you learned that a hypothesis is just an educated guess that you will try to prove by conducting an experiment. In reality, it's a bit more complicated than that. During this stage, you'll formulate 2 hypotheses to test--your educated guess about the outcome is called the ***Alternative Hypothesis***, while the opposite of it is called the ***Null Hypothesis***. This is where the language behind experimental design (and the idea of what you can actually ***prove*** using an experiment) gets a bit complicated--more on this below.

4. Conduct an Experiment

This step is the part of the scientific method that will be the focus of this section. You can only test a hypothesis by gathering data from a well-structured experiment. A well-structured experiment is one that accounts for all of the mistakes and randomness that could give you false signals relating to the effect of an intervention. Just because you're running an experiment doesn't prove that A causes B, or that there's even a relationship between A and B! A poorly designed experiment will lead to false conclusions that you haven't considered or controlled for. A well-designed experiment leaves you no choice but to acknowledge that the effects seen in a dependent variable are related to an independent variable. The world is messy and random. You have to account for this messiness and randomness in experiments so that you can filter it out and be left only with the things you're actively trying to measure.

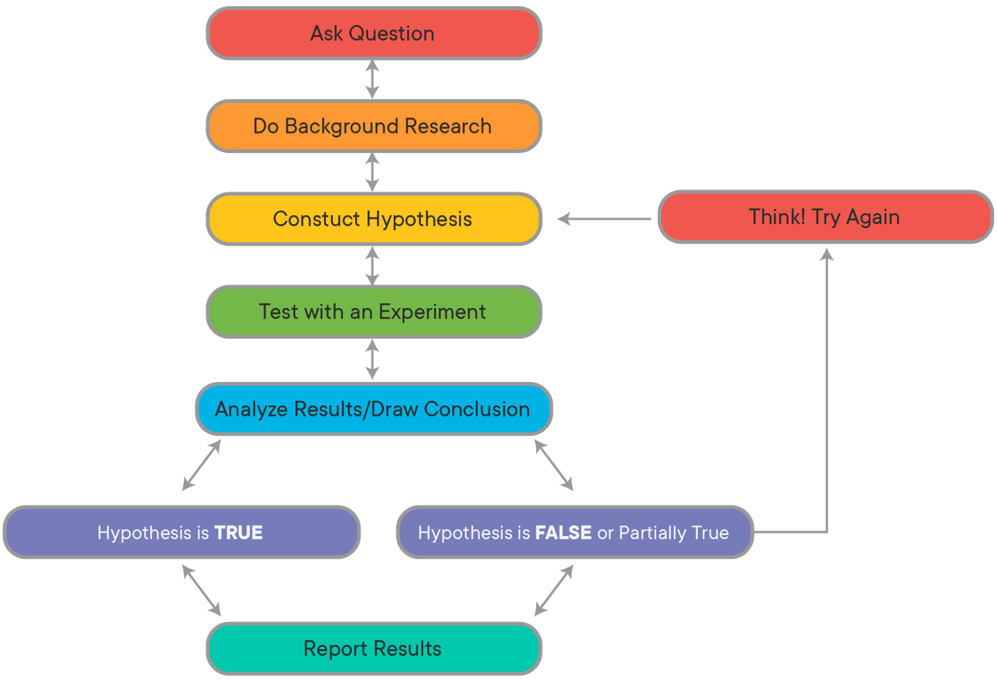
5. Analyze Experimental Results

5. Analyze Experimental Results

Whether you realize it or not, you've already gotten pretty good at this step! All the work you've done with statistics is usually in service of this goal--looking at the data and understanding what happened. During this step, you will tease out relationships, filter out noise, and try to determine if something that happened is ***statistically significant*** or not.

6. Draw Conclusions

This step is the logical endpoint for an experiment. You've asked a question, looked at experimental results from others that could be related to your question, made an educated guess, designed an experiment, collected data, and analyzed the results. All that is left is to use the results of the analysis step to evaluate whether you believe the hypothesis was correct or not! While the public generally oversimplifies this step for determining causal relationships (e.g. "my experiment showed that {x} causes {y}"), true scientists rarely make claims so bold. The reality of this step is that you use your analysis of the data to do one of two things: either ***reject the null hypothesis or fail to reject the null hypothesis***. This is a tricky concept, so you'll explore it in much more detail in a future lesson.



The Scientific Method

The Foundations of a Sound Experiment

All experiments are not created equal--simply following the steps outlined above does not guarantee that the results of any experiment will be meaningful. For instance, there's nothing stopping a person from testing the hypothesis that "wearing a green shirt will make it rain tomorrow!", seeing rain the next day, and rejecting the null hypothesis, thereby incorrectly "proving" that their choice of wardrobe affected the weather. Good experiments demonstrate that independent variables {X} have an effect on the dependent variables {Y} because you control for all the other things that could be affecting {Y}, until you are forced to conclude that the only thing that explains what happened to {Y} is {X}!

Although there are many different kinds of experiments, there are some fundamental aspects of experimental design that all experiments have:

1. A Control Group/Random Controlled Trials

One of the most important aspects of a sound experiment is the use of a ***Control Group***. A Control Group is a cohort that receives no treatment or intervention--for them, it's just business as usual. In a medical test, this might be a ***placebo***, such as a sugar pill. In the example of testing the color of a button on a website, this would be customers that are shown a version of the website with the button color unchanged. Using a control group allows researchers to compare the results of doing nothing (the control) with the effects of doing something (the ***intervention***). Without a control group, you have no way of knowing how much of the results you see can be attributed to the intervention, and how much would have happened anyways.

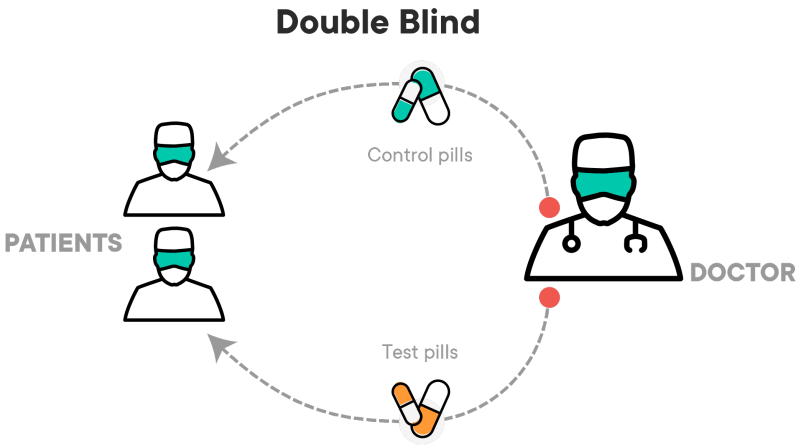
To make this more obvious, consider what you can actually know with confidence after an experiment that doesn't use a control. Assume that a pharmaceutical company decides to test a new drug that is supposed to reduce the amount of time someone has the flu. The company gives the drug to all participants in the study. After analyzing the data, you find that the average length of time a person had the flu was 12 days. Was the drug effective, or not? Without a control, you don't know how long this flu would have lasted if these people were never given a drug. It could be that your drug reduced the time of infection down to 12 days. Then again, it could be that these people would have gotten better on their own after 12 days, and the drug didn't really do anything, or again, it could be that these people would have gotten better on their own after 12 days, and the drug didn't really do anything--or maybe they would have gotten better in 10 days, and the drug made it worse! By using a control group that gets no drugs and recovers naturally, you can compare the results of the treatment (people that received the experimental flu drug) to your control group (people that recovered naturally).

Note that a control group is only a control group if they are sampled from the same population as the treatment groups! If they aren't the same, then there's no way of knowing how much the difference in recovery time should be attributed to the flu drug, and how much should be attributed to the way(s) in which the control group is different. For instance, the experiment would not be very effective if the average age of one group was much higher or lower than another. If that was the case, how would you know the age difference isn't actually causing the difference in results (or lack thereof) between the control and treatment groups, instead of the drug intervention?

The main way scientists deal with this is through ***Random Controlled Trials***. In a Random Controlled Trial, there is a control group and an intervention (also called treatment) group, where subjects are ***randomly assigned to each***. You may have heard the term ***Single-Blind*** and ***Double-Blind*** studies--these refer to people knowing which groups they are in. In a sound experiment, people should not know if they are in the treatment group or the control group, as that could potentially affect the outcome of the trial!

A ***Single-Blind*** or ***Blind Trial*** is one where the participant does not know if they are receiving the treatment or a placebo.

A ***Double-Blind Trial*** is one where the participant does not know if they are receiving the treatment or a placebo, and neither does the person administering the experiment (because their bias could affect the outcomes, too!). Instead, knowing whether someone received the treatment or a placebo is kept hidden from everyone until after the experiment is over (obviously, *someone* has to know for recordkeeping purposes, but that person stays away from the actual experiment to avoid contaminating it with bias from that knowledge).



2. Appropriate Sampling Techniques and Sample Size

When data scientists are performing experiments, they rarely have the opportunity to work with an entire population of data. Rather, they must obtain a sample that is representative of the population. In order to get a high quality sample, you should follow these four assumptions related to sampling techniques and sample size.

- **Sample is independent**

Independence means the value of one observation does not influence or affect the value of other observations. Independent data items are not connected with one another in any way (unless you account for it in your model). This includes the observations in both the “between” and “within” groups of your sample. Non-independent observations introduce bias and can make your statistical test give too many false positives.

- **Sample is collected randomly**

A sample is random when each data point in your population has an equal chance of being included in the sample; therefore, the selection of any individual observation happens by chance, rather than by choice. This reduces the chance that differences in materials or conditions strongly bias results. Random samples are more likely to be representative of the population; therefore, you can be more confident with your statistical inferences with a random sample.

- **The sample is approximately normally distributed**

The normal distribution assumption is that the sampling distribution of the mean is normal. That is, if you took a sample, calculated its mean, and then you took another (independent) sample (from the same population) and got its mean (and repeated this an infinite number of times), then the distribution of the values that you wrote down would always be a perfect bell curve. This is the principle behind the Central Limit Theorem, and it is this idea that allows us to perform hypothesis tests. While maybe surprising, this assumption turns out to be relatively uncontroversial, at least when each of the samples is large, such as $N \geq 30$.

- **Appropriate Sample Size** *

Randomness is a big problem in experiments. It can lead you to false conclusions by making you think that something doesn't matter when it does, or vice versa. Small sample sizes make experiments susceptible to the problem of randomness; whereas, large sample sizes protect experiments from it. The following scenario illustrates this point:

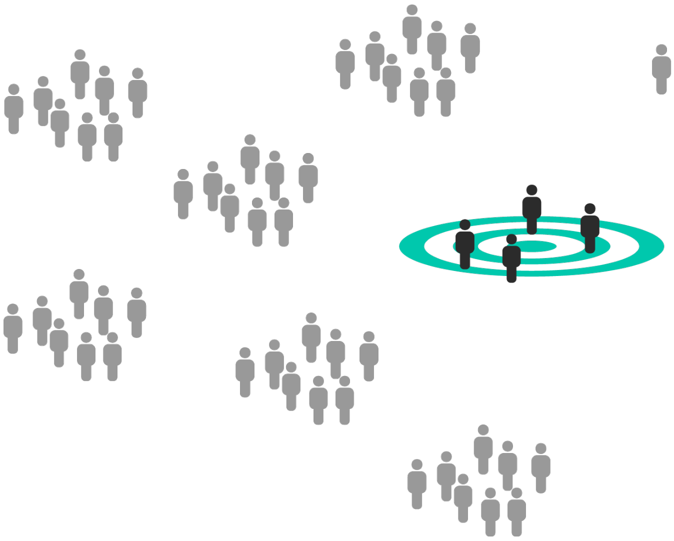
A person tells you that they can predict the outcome of a fair coin flip. You flip a coin, they call "tails", and they are correct. Is this enough evidence to accept or reject this person's statement? What if they got it right 2 times in a row? 5 times in a row? 55 times out of 100?

This situation illustrates two things that are important for us to understand and acknowledge:

1. No matter how large your sample size, there's always a chance that your results can be attributed to randomness or luck.
2. At some point, you would cross a threshold where random chance is small enough that you'd say "this probably isn't random", and are okay with accepting the results as the result of something other than randomness or luck.

With the situation above, you probably wouldn't assume that this person can predict coin flips after only seeing them get 1 correct. However, if this person got 970 out of 1000 correct, you would probably believe very strongly that this person *can* predict coin flips because the odds of guessing randomly and getting 970/1000 correct are very, very small--but not 0!

Large sample sizes protect us from randomness and variance. A more realistic example would be testing a treatment for HIV. Less than 1% of the global population carries a protective mutation that makes them resistant to HIV infection. If you took a randomly selected sample of 1 person from the population, there is a ~1% chance that you may mistakenly attribute successful prevention to the drug you're testing, when the results really happened because you randomly selected a person with this mutation. However, if your sample size was 100 people per sample, your odds of randomly selecting 100 people with that mutation are 0.01^{100}. The larger your sample size, the more unlikely it is that you randomly draw people that happen to affect your study in a way that is not reflected by the general population.



3. Reproducibility

This one is a big one, and it represents a bit of a crisis in some parts of the scientific community right now. Good scientific experiments have ***Reproducible Results***! This means that if someone else follows the steps you outline for your experiment and performs it themselves, they should get pretty much the same results as you did (allowing for natural variance and randomness). If many different people try reproducing your experiment and don't get the same results, this might suggest that your results are due to randomness, or to a ***lurking variable*** that was present in your samples that wasn't present in others. Either way, a lack of reproducibility often casts serious doubts on the results of a study or experiment.

This is less of a problem for data scientists, since reproducibility usually just means providing the dataset you worked with and the corresponding Jupyter notebook. However, this isn't always the case! Luckily, you can use code to easily run your experiments multiple times and show reproducibility. When planning experiments, consider running them multiple times to ensure to really help show that your results are sound, and not due to randomness!

Summary

Great, you now know about experimental design and the fundamental aspects of experiments!

P-Values and the Null Hypothesis

Introduction

In this lesson, you'll learn about the relationship between p-values and the Null Hypothesis, and their role in designing an experiment.

Objectives

You will be able to:

- Describe what it means to "reject the null hypothesis" and how it is related to p-value

Understanding The Null Hypothesis

As stated previously, scientific experiments actually have 2 hypotheses:

Null Hypothesis: There is no relationship between A and B
Example: "There is no relationship between this flu medication and a reduced recovery time from the flu".

The *Null Hypothesis* is usually denoted as H_0

Alternative Hypothesis: The hypothesis traditionally thought of when creating a hypothesis for an experiment
Example: "This flu medication reduces recovery time for the flu."

The *Alternative Hypothesis* is usually denoted as H_1

An easy way to differentiate between the Null Hypothesis and the Alternative Hypothesis is that the Null Hypothesis is the more conservative choice. It always assumes that there is no difference between two different population means, and when it is represented mathematically, it should always contain an equals sign.

The Alternative Hypothesis is whatever claim you are trying to prove with an experiment.

P-Values and Alpha Values

No matter what you're experimenting on, good experiments come down to one question: Is your p-value less than your alpha value? Let's dive into what each of these values represents, and why they're so important to experimental design.

p-value: The probability of observing a test statistic at least as large as the one observed, by random chance, assuming that the null hypothesis is true.

If you calculate a p-value and it comes out to 0.03, you can interpret this as saying "There is a 3% chance of obtaining the results I'm seeing when the null hypothesis is true."

α ***(alpha value)***: The marginal threshold at which you're okay with rejecting the null hypothesis.

An alpha value can be any value set between 0 and 1. However, the most common alpha value in science is 0.05 (although this is somewhat of a controversial topic in the scientific community, currently).

If you set an alpha value of $\alpha = 0.05$, you're essentially saying "I'm okay with accepting my alternative hypothesis as true if there is less than a 5% chance that the results that I'm seeing are actually due to randomness."

When you conduct an experiment, your goal is to calculate a p-value and compare it to the alpha value. If $p < \alpha$, then you ***reject the null hypothesis*** and accept that there is not "no relationship" between the dependent and independent variables. Note that any good scientist will admit that this doesn't prove that there is a *direct relationship* between the dependent and independent variables--just that they now have enough evidence to the contrary to show that they can no longer believe that there is no relationship between them.

In simple terms:

$p < \alpha$: Reject the *Null Hypothesis* and accept the *Alternative Hypothesis*

$p >= \alpha$: Fail to reject the *Null Hypothesis*.

There are many different ways that you can structure a hypothesis statement, but they always come down to this comparison in the end. In normally distributed data, you calculate p-values from t-statistics or (z-scores if the population parameters are known). This is done a bit differently with discrete data. You may also have ***One-Tail*** and ***Two-Tail*** tests.

A ***One-Tail Test*** is when you want to know if a parameter from the treatment group is greater than (or less than) a corresponding parameter from the control group.

Example One-Tail Hypothesis

$H_1 : \mu_1 < \mu_2$ The treatment group given this weight loss drug will lose more weight on average than the control group that was given a competitor's weight loss drug

$H_0 : \mu 1 >= \mu_2$ The treatment group given this weight loss drug will not lose more weight on average than the control group that was given a competitor's weight loss drug".

A ***Two-Tail Test*** is for when you want to test if a parameter falls between (or outside of) a range of two given values.

Example Two-Tail Hypothesis

$H_1 : \mu_1 \neq \mu_2$ "People in the experimental group that are administered this drug will not lose the same amount of weight as the people in the control group. They will be heavier or lighter".

$H_0 : \mu_1 = \mu_2$ "People in the experimental group that are administered this drug will lose the same amount of weight as the people in the control group."

What Does an Experiment Really Prove?

You may be wondering why you need a ***Null Hypothesis*** at all. This is a good question. It has to do with being honest about what an experiment actually proves.

Scientists use the ***Null Hypothesis*** so that they can be very specific in their findings. This is because a successful experiment doesn't actually *prove a relationship* between a dependent and independent variable. Instead, it just proves that there is not enough evidence to convincingly believe there is *no relationship* between the dependent and the independent variable. There can always be a lurking variable behind the scenes that is actually responsible for the relationship between two variables--it's almost impossible to cover every possible angle. However, a successful experiment where a p-value is less than an alpha value (typically, $p < 0.05$) does give enough information to confidently allow someone to say that it's statistically unlikely that there is *no relationship* between the two, which is what would have to be true in order for the null hypothesis to be correct!

The Null Hypothesis Loves You and Wants You To Be Happy

You've covered a lot about the null hypothesis and how it's used in experiments in this lesson, but there's a lot more to learn about it!

Read the following article, The Null Hypothesis Loves You and Wants You To Be Happy. This does an excellent job of explaining why the concept of the *Null Hypothesis* is crucial to good science.

Summary

In this lesson, you learned about the relationship between p-values and the Null Hypothesis. Now you'll see how effect sizes affect your tests!

Effect Size

Introduction

When comparing results between groups, and results prove to be different, it is important to understand what the size of the difference is. You'll learn about that here!

Objectives

- Compare and contrast p-value and effect size for identifying the significance of results
- Interpret the results of a simple effect size and identify shortcomings of this approach
- Calculate and interpret standardized and unstandardized effect sizes
- Create a visualization to demonstrate different effect sizes between distributions of data

Introduction to Effect Size

Effect size is used to quantify the *size of the difference* between two groups under observation. Effect sizes are easy to calculate, understand and apply to any measured outcome and are applicable to a multitude of study domains. It is highly valuable towards quantifying the *effectiveness of a particular intervention, relative to some comparison*. Measuring effect size allows scientists to go beyond the obvious and simplistic *'Does it work or not?'* to the far more sophisticated, *'How well does it work in a range of contexts?'*.

More on effect size

P-value vs. Effect Size

Effect size measurement places its emphasis on the effect size only, unlike statistical significance which combines effect size and sample size, thus promoting a more scientific approach towards knowledge accumulation. Effect size is therefore routinely used in **Meta-Analysis** i.e. for combining and comparing estimates from different studies conducted on different samples.

By increasing sample size, you can show there is a statistically significant difference between two means. However, **statistically significant does not necessarily imply "significant."**.

> **P value** = probability sample means are the same.
>
> (1 – P) or **Confidence Level** = probability sample means are different.
>
> **Effect Size** = how different sample means are

In light of this, it is possible to achieve highly significant p-values for effect sizes that have no practical significance. In contrast, study designs with low power can produce non-significant p-values for effect sizes of great practical importance.

Further details on p-value vs. effect size calculation

Why do data scientists need to know about 'Effect Size'?

Consider the experiment conducted by Dowson (2000) to investigate time of day effects on children learning: do children learn better in the morning or afternoon? A group of 38 children was included in the experiment. Half were randomly allocated to listen to a story and answer questions about it at 9 am, the other half heard exactly the same story and had to answer the same questions at 3 pm. Their comprehension was measured by the number of questions answered correctly out of 20.

The average score was 15.2 for the morning group and 17.9 for the afternoon group, giving a difference of 2.7. **How big of a difference is this?**

If the results were measured on a standard scale, such as a 4 point GPA scale, interpreting the difference would not be a problem. If the average difference was, say, half a grade or a full grade, most people would have a fair idea of the educational significance of the effect of reading a story at different times of the day. However, in many experiments, there is no familiar scale available on which to record the outcomes i.e. student comprehension in this case. The experimenter often has to invent a scale or use (or adapt) an already existing one - but generally, most people would be unfamiliar with the interpretation of this scale.

In a data analytics domain, effect size calculation serves three primary goals:

- Communicate the **practical significance** of results. An effect might be statistically significant, but does it matter in practical scenarios?

scenarios?

- Effect size calculation and interpretation allows you to draw **Meta-Analytical** conclusions. This allows you to group together a number of existing studies, calculate the meta-analytic effect size and get the best estimate of the effect size of the population.
- Perform **Power Analysis**, which helps determine the number of participants (sample size) that a study requires to achieve a certain probability of finding a true effect - if there is one.

Calculating effect size in Python

Using SciPy for measuring effect size

SciPy (pronounced “Sigh Pie”) is open-source software for mathematics, science, and engineering. The SciPy package contains various toolboxes dedicated to common issues in scientific computing. Its different submodules correspond to different applications, such as interpolation, integration, optimization, image processing, statistics, special functions, etc. For an experiment, you can use `scipy.stats` package which contains statistical tools and probabilistic descriptions of random processes. Detailed documentation of SciPy is available here.

In [1]:

```
# Import necessary modules
from __future__ import print_function, division
import numpy as np

# Import SciPy stats and matplotlib for calculating and visualising effect size
import scipy.stats
import matplotlib.pyplot as plt

%matplotlib inline

# seed the random number generator so you get the same results
np.random.seed(10)
```

Example:

To explore statistics that quantify effect size, let's first look at the difference in height between men and women in the USA, based on the mean and standard deviation for male and female heights as given in (BRFSS) Behavioral Risk Factor Surveillance System.

> **Males Height** (Mean = 178 , Standard Deviation = 7.7)
>
> **Female Height** (Mean = 163 , Standard Deviation = 7.3)

You can use `scipy.stats.norm()` to represent the height distributions by passing mean and standard deviation values as arguments for creating normal distribution.

In [2]:

```
#Mean height and sd for males
male_mean = 178
male_sd = 7.7

# Generate a normal distribution for male heights
male_height = scipy.stats.norm(male_mean, male_sd)
```

The result `male_height` is a SciPy `rv` object which represents a **normal continuous random variable**.

In [3]:

```
male_height
```

Out[3]:

```
<scipy.stats._distn_infrastructure.rv_frozen at 0x2b086d9fc88>
```

Use the mean and standard deviation for female height and repeat calculations shown above to calculate `female_height` as an `rv` object.

In [4]:

```
female_mean = 163
female_sd = 7.3
female_height = scipy.stats.norm(female_mean, female_sd)
```

Evaluate Probability Density Function (PDF)

A continuous random variable, as calculated above, takes on an uncountably infinite number of possible values.

For a **discrete** random variable, X, that takes on a finite or infinite number of possible values, we determine P(X = x) for all of the possible values of X and call it the probability mass function (PMF).

For **continuous** random variables, as in the case of heights, the probability that X takes on any particular value x is 0. That is, finding P(X = x) for a continuous random variable X is not going to work. Instead, you'll need to find the probability that X falls in some interval (a, b) i.e. you'll need to find **P(a < X < b)** using a **probability density function(PDF)**.

The following function evaluates the normal (Gaussian) probability density function within 4 standard deviations of the mean. The function takes an rv object and returns a pair of NumPy arrays.

In [5]:

```
def evaluate_PDF(rv, x=4):
    '''Input: a random variable object, standard deviation
    output : x and y values for the normal distribution
    '''

    # Identify the mean and standard deviation of random variable
    mean = rv.mean()
    std = rv.std()

    # Use numpy to calculate evenly spaced numbers over the specified interval (4 sd) and generate 100
samples.
    xs = np.linspace(mean - x*std, mean + x*std, 100)

    # Calculate the peak of normal distribution i.e. probability density.
    ys = rv.pdf(xs)

    return xs, ys # Return calculated values
```

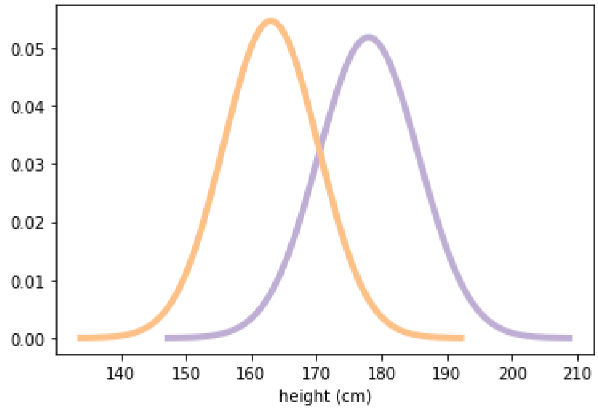
Let's use the function above to calculate `xs` and `ys` for male and female heights (pass the `rv` object as an argument) and plot the resulting `xs` and `ys` for both distributions to visualize the effect size.**

In [6]:

```
# Male height
xs, ys = evaluate_PDF(male_height)
plt.plot(xs, ys, label='male', linewidth=4, color='#beaed4')

#Female height
xs, ys = evaluate_PDF(female_height)
plt.plot(xs, ys, label='female', linewidth=4, color='#fdc086')

plt.xlabel('height (cm)')
```

Out[6]:

```
Text(0.5, 0, 'height (cm)')
```



Let's assume for the sake of simplicity that these are the true distributions for the population. As you studied earlier, in real life one would never observe the true population distribution. You generally have to work with a random sample from the population. Let's try to work out how different these two groups are with respect to height by calculating un-standardized and standardized effect sizes.

Un-standardized or Simple Effect Size Calculation

An unstandardized effect size simply tries to find the difference between two groups by calculating the difference between distribution means. Here is how you can do it in Python.

You can use the `rvs` method from `scipy.stats` to generate a random sample of size 1000 from the population distributions. Note that these are totally random and representative samples, with no measurement error.

Visit this link for more details on `sciPy.stats`.

In [7]:

```
male_sample = male_height.rvs(1000)
```

The resulting samples are NumPy arrays, so we can now easily calculate the mean and standard deviation of random samples.

In [8]:

```
mean1, std1 = male_sample.mean(), male_sample.std()
mean1, std1
# (177.88791390576085, 7.222274730410271)
```

Out[8]:

```
(177.88791390576085, 7.222274730410271)
```

The sample mean is close to the population mean, but not exactly the same, as expected.

Now, perform above calculation for female heights to calculate mean and sd of random samples from `female_height` `rv` object**

In [9]:

```
female_sample = female_height.rvs(1000)
mean2, std2 = female_sample.mean(), female_sample.std()
mean2, std2
# (162.91903182040372, 7.261850929417819)
```

Out[9]:

```
(162.91903182040372, 7.261850929417819)
```

And the results are similar for the female sample.

Now, there are many ways to describe the magnitude of the difference between these distributions. An obvious one is the difference in the means.

Now, calculate the difference in means of both distributions identified above.**

In [10]:

```
difference_in_means = male_sample.mean() - female_sample.mean()
difference_in_means # in cm
# 14.968882085357137
```

Out[10]:

```
14.968882085357137
```

This shows that, on average, men are around 15 centimeters taller. For some applications, that would be a good way to describe the difference, but there are caveats:

- Without knowing more about the distributions (like the standard deviations or *spread* of each distribution), it's hard to interpret whether a difference like 15 cm is a **big difference** or not.
- The magnitude of the difference depends on the units of measure, making it hard to compare across different studies that may be conducted with different units of measurement.

There are a number of ways to quantify the difference between distributions. A simple option is to express the difference as a percentage of the mean.

Let's figure out the relative difference in the means of two populations, scaled by the mean of male heights and expressed as a percentage.

In [11]:

```
relative_difference = difference_in_means / male_sample.mean()
relative_difference * 100   # percent

#  8.414783082614122
```

Out[11]:

```
8.414783082614122
```

But a problem with relative differences is that you have to choose which mean to express them relative to.

In [12]:

```
relative_difference = difference_in_means / female_sample.mean()
relative_difference * 100    # percent

# 9.18792722869745
```

Out[12]:

```
9.18792722869745
```

Overlap threshold

As you can see above, there is still a difference in results when you express the relative difference, depending on whether we choose to represent the ratio relative to male height or female height. Perhaps you can look for the amount of overlap between the two distributions. To define overlap, you choose a threshold between the two means. The simple threshold is the midpoint between the means:

In [13]:

```
simple_thresh = (mean1 + mean2) / 2
simple_thresh
```

Out[13]:

```
170.4034728630823
```

A better, but slightly more complicated threshold is the place where the PDFs cross.

In [14]:

```
thresh = (std1 * mean2 + std2 * mean1) / (std1 + std2)
thresh
```

Out[14]:

```
170.42392323303363
```

In this example, there's not much difference between the two thresholds. Now you can count how many men are below the threshold:

In [15]:

```
male_below_thresh = sum(male_sample < thresh)
male_below_thresh
```

Out[15]:

```
154
```

Similarly, you can calculate how many women are above the calculated threshold

In [16]:

```
female_above_thresh = sum(female_sample > thresh)
female_above_thresh
```

Out[16]:

```
152
```

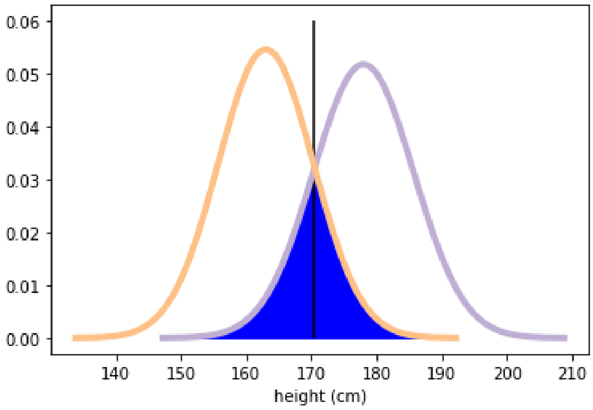
Now, take a look at what these thresholds look like when laid over the Probability Density Functions of both samples' distributions.

In [17]:

```
# Male height
m_xs, male_ys = evaluate_PDF(male_height)
plt.plot(m_xs, male_ys, label='male', linewidth=4, color='#beaed4')

#Female height
f_xs, female_ys = evaluate_PDF(female_height)
plt.plot(f_xs, female_ys, label='female', linewidth=4, color='#fdc086')
plt.vlines(thresh,ymin=0,ymax=0.06)
plt.fill_betweenx(male_ys,x1 = m_xs,x2=thresh, where = m_xs < thresh,color='b')
plt.fill_betweenx(female_ys,x1=f_xs,x2=thresh, where = f_xs > thresh,color='b')
plt.xlabel('height (cm)')
```

Out[17]:

```
Text(0.5, 0, 'height (cm)')
```



The "overlap" (shaded region above) is the total **AUC (Area Under the Curves)**. You can use this to identify the samples that end up on the wrong side of the threshold. You can calculate the amount of overlap as shown below.

In [18]:

```
# Calculate the overlap
overlap = male_below_thresh / len(male_sample) + female_above_thresh / len(female_sample)
overlap
```

Out[18]:

```
0.306
```

Or in more practical terms, you might report the fraction of people who would be misclassified if you tried to use height to guess sex:

In [19]:

```
misclassification_rate = overlap / 2
misclassification_rate
```

Out[19]:

```
0.153
```

Probability of superiority (Non-parametric)

Another "non-parametric" way to quantify the difference between distributions is what's called **"probability of superiority"**, which is the probability that *"a randomly-chosen man is taller than a randomly-chosen woman"*, which makes perfect sense.

> Question: If you chose a male and a female sample at random, what is the probability that males are taller than females?

In [20]:

```
# Python zip() The zip() function take iterables (can be zero or more),
# makes iterator that aggregates elements based on the iterables passed,
# and returns an iterator of tuples.

sum(x > y for x, y in zip(male_sample, female_sample)) / len(male_sample)
```

Out[20]:

```
0.94
```

> Question: If you chose a female and a male sample at random, what is the probability that females are smaller than males in height? Is it different/same as above?

In [21]:

```
sum(x < y for x, y in zip(female_sample, male_sample)) / len(female_sample)
```

Out[21]:

```
0.94
```

Overlap (or misclassification rate) as shown above and "probability of superiority" have two good properties:

- As probabilities, they don't depend on units of measure, so they are comparable between studies.
- They are expressed in operational terms, so a reader has a sense of what practical effect the difference makes.

There is one other common way to express the difference between distributions (i.e. the difference in means) standardizing by dividing by the standard deviation.

Here's a function that encapsulates the code you have already seen for computing overlap and probability of superiority.

In [22]:

```
def overlap_superiority(group1, group2, n=1000):
    """Estimates overlap and superiority based on a sample.

    group1: scipy.stats rv object
    group2: scipy.stats rv object
    n: sample size
    """

    # Get a sample of size n from both groups
    group1_sample = group1.rvs(n)
    group2_sample = group2.rvs(n)

    # Identify the threshold between samples
    thresh = (group1.mean() + group2.mean()) / 2
    print(thresh)

    # Calculate no. of values above and below for group 1 and group 2 respectively
    above = sum(group1_sample < thresh)
    below = sum(group2_sample > thresh)

    # Calculate the overlap
    overlap = (above + below) / n

    # Calculate probability of superiority
    superiority = sum(x > y for x, y in zip(group1_sample, group2_sample)) / n

    return overlap, superiority
```

In [23]:

```
overlap_superiority(male_height, female_height, n=1000)
```

```
170.5
```

Out[23]:

```
(0.336, 0.94)
```

Standardized effect size

When analysts generally talk about effect sizes, they refer to some method of calculating a *standardized* effect size. The standardized effect size statistic would divide effect size by some standardizer i.e. standard deviation:

> **Effect Size / Standardiser**

When interpreting, this statistic would be in terms of standard deviations e.g. The mean height of males in USA is 1.4 standard deviations higher than mean female heights etc. The effect size measure you will be learning about in this lesson is Cohen’s d. This measure expresses the size of an effect in terms of the number of standard deviations, similar to a z-score in statistics.

In [24]:

```
## not covered yet
"Cohen’s d is similar to the unpaired t test t value. It relies on Standard Deviations instead of Stand
ard Errors"
```

Out[24]:

```
'Cohen’s d is similar to the unpaired t test t value. It relies on Standard Deviations instead of Stand
ard Errors'
```

Cohen's d

Cohen’s d is one of the most common ways to measure effect size. As an effect size, Cohen's d is typically used to represent the magnitude of differences between two (or more) groups on a given variable, with larger values representing a greater differentiation between the two groups on that variable.

The basic formula to calculate Cohen’s d is:

> **d = effect size (difference of means) / pooled standard deviation**

The denominator is the **standardiser**, and it is important to select the most appropriate one for a given dataset. The pooled standard deviation is the average spread of all data points around their group mean (not the overall mean).

In [25]:

```
def Cohen_d(group1, group2):

    # Compute Cohen's d.

    # group1: Series or NumPy array
    # group2: Series or NumPy array

    # returns a floating point number

    diff = group1.mean() - group2.mean()

    n1, n2 = len(group1), len(group2)
    var1 = group1.var()
    var2 = group2.var()

    # Calculate the pooled threshold as shown earlier
    pooled_var = (n1 * var1 + n2 * var2) / (n1 + n2)

    # Calculate Cohen's d statistic
    d = diff / np.sqrt(pooled_var)

    return d
```

Computing the denominator is a little complicated; in fact, people have proposed several ways to do it. Here is a brief description of using standardizers while calculating Cohen's d for standardized effect sizes.

This implementation uses the "pooled standard deviation," which is a weighted average of the standard deviations of the two groups.

And here's the result for the difference in height between men and women.

In [26]:

```
Cohen_d(male_sample, female_sample)
```

Out[26]:

```
2.0660985209851077
```

```
2.0669285200851877
```

Interpreting d

Most people don't have a good sense of how big d = 2.0 is. If you are having trouble visualizing what the result of Cohen’s D means, use these general “rule of thumb” guidelines (which Cohen said should be used cautiously):

> **Small effect = 0.2**
>
> **Medium Effect = 0.5**
>
> **Large Effect = 0.8**

Here is an excellent online visualization tool developed by Kristoffer Magnusson to help interpret the results of cohen's d statistic.

The following function that takes Cohen's d, plots normal distributions with the given effect size, and prints their overlap and superiority.

In [27]:

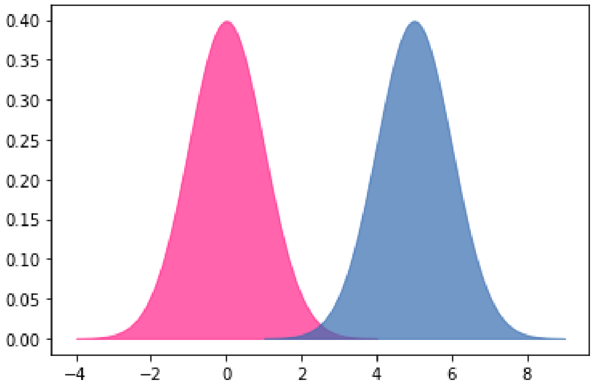
```
def plot_pdfs(cohen_d=2):
    """Plot PDFs for distributions that differ by some number of stds.

    cohen_d: number of standard deviations between the means
    """
    group1 = scipy.stats.norm(0, 1)
    group2 = scipy.stats.norm(cohen_d, 1)
    xs, ys = evaluate_PDF(group1)
    plt.fill_between(xs, ys, label='Group1', color='#ff2289', alpha=0.7)

    xs, ys = evaluate_PDF(group2)
    plt.fill_between(xs, ys, label='Group2', color='#376cb0', alpha=0.7)

    o, s = overlap_superiority(group1, group2)
    print('overlap', o)
    print('superiority', s)
```

Here's an example that demonstrates the function:

In [28]:

```
plot_pdfs(5)
# Try changing the d value and observe the effect on the outcome below
```

```
2.5
overlap 1.992
superiority 0.0
```



Cohen's d has a few nice properties:

- Because mean and standard deviation have the same units, their ratio is dimensionless, so you can compare d across different studies.
- In fields that commonly use d, people are calibrated to know what values should be considered big, surprising, or important.
- Given d (and the assumption that the distributions are normal), you can compute overlap, superiority, and related statistics.

Summary

Summary

In this lesson, you highlighted the importance of calculating and interpreting effect size in Python as a measure of observing real world differences between two groups. You learned about simple (unstandardized) effect size calculation as the difference of means, as well as the standardization of this calculation with standard deviation as a standardizer. You also learned what Cohen's d statistic is and how to use it for practical purposes. The best way to report effect size often depends on the audience, goals, and subjects of study. There is often a tradeoff between summary statistics that have good technical properties and statistics that are meaningful to a general audience.

Conducting T-Tests

Introduction

Just as you previously used the t distribution to provide confidence intervals for estimating the population mean, you can also use similar methods to test whether two populations are different, statistically speaking. To do this, you can use a t-test.

Objectives

You will be able to:

- Compare when you would use one sample vs. two sample t-tests
- Perform a one sample t-test and make conclusions about an experiment based on the results

Hypothesis testing using the T-distribution

In frequentist hypothesis testing, you construct a test statistic from the measured data and use the value of that statistic to decide whether to accept or reject the null hypothesis. The test statistic is a lower-dimensional summary of the data but still maintains the discriminatory power necessary to make the decision whether or not to reject the null hypothesis.

t-test

t-tests (also called Student's t-test) are very practical hypothesis tests that can be employed to compare two averages (means) to assess if they are different from each other. You should run a t-test when you either:

- Don't know the population standard deviation
- You have a small sample size

Like a z-test, the t-test also tells you how significant the differences are i.e. it lets you know if those differences could have happened by chance. In this lesson, you will get an introduction to t-tests, in particular, the 1-sample t-test. There are additional kinds of t-tests including the 2-sample t-test and paired t-test. This lesson will show you the mathematical calculations behind a 1-sample t-test as well as how to perform a t-test in Python using NumPy and SciPy.

Detailed descriptions of hypothesis testing with t-tests can be found here and here

One Sample t-test

The 1-sample t-test is a statistical procedure used to determine whether a sample of observations could have been generated by a process with a specific mean. The one sample t-test compares the mean of your sample data to a known value. For example, you might want to know how your sample mean compares to the population mean. Here is a quick example of a scenario where a 1-sample t-test could be applied.

Suppose you are interested in determining whether a bakery production line produces cakes with a weight of exactly 2 pounds. To test this hypothesis, you could collect a sample of cakes from the production line, measure their weights, and compare the sample with a value of 2 using a one-sample t-test.

Two Sample t-tests

The two-sample t-test is used to determine if two population means are equal. The main types of two-sampled t-tests are paired and independent tests. Paired tests are useful for determining how different a sample is affected by a certain treatment. In other words, the individual items/people in the sample will **remain the same** and researchers are comparing how they change after treatment. Here is an example of a scenario where a two-sample paired t-test could be applied:

The US Olympic weightlifting team is trying out a new workout techniques to in an attempt to improve everyone's powerlifting abilities. Did the program have an effect at a 95% significance level?

Because we are looking at how specific individuals were affected by a treatment, we would use the paired t-test.

Independent two-sample t-tests are for when we are comparing two different, unrelated samples to one another. Unlike paired t-tests, we are not taking paired differences because there is no way to pair two unrelated samples! Here is an example of a scenario where a two-sample independent t-test could be applied:

Agricultural scientists are trying to compare the difference in soybean yields in two different counties of Mississippi.

You will learn more about the specifics of two sample t-tests in future lessons, but this lesson will focus on executing a one sample t-test.

Assumptions for performing t-tests

When performing various kinds of t-tests, you assume that the sample observations have numeric and continuous values. You also assume that the sample observations are independent from each other (that is, that you have a simple random sample) and that the samples have been drawn from normal distributions. You can visually inspect the distribution of your sample using a histogram, for example.

In the case of unpaired two-sample t-tests, you also assume that the populations the samples have been drawn from have the same variance. For paired two-sample t-tests, you assume that the *difference* between the two sets of samples are normally distributed.

Regardless of the type of t-test you are performing, there are 5 main steps to executing them:

1) Set up null and alternative hypotheses

2) Choose a significance level

3) Calculate the test statistic

4) Determine the critical or p-value (find the rejection region)

5) Compare t-value with critical t-value to accept or reject the Null hypothesis.

Now, you're going to go through these 5 steps in more detail to complete a t-test.

Let's begin with a sample experiment:

Sample question:

> *"Acme Ltd. wants to improve sales performance. Past sales data indicate that the average sale was 100 dollars per transaction. After training the sales force, recent sales data (from a random sample of 25 salesmen) is shown below:"*

```
[122.09, 100.64, 125.77, 120.32, 118.25,
  96.47, 111.4 ,  80.66, 110.77, 111.14,
  102.9, 114.54,  88.09,  98.59,  87.07,
 110.43, 101.9 , 123.89,  97.03, 116.23,
  108.3, 112.82, 119.57, 131.38, 128.39]
```

Did the training work?

Before completing the hypothesis test, let's calculate some summary statistics to see if the mean of the sample differed a substantial amount from the population. After, you can check to ensure that the data is relatively normal.

- **The population mean (μ):** Given as 100 (from past data).
- **The sample mean ($\bar{x}$):** Calculate from the sample data
- **The sample standard deviation (s):** Calculate from sample data
- **Number of observations(n):** 25 as given in the question. This can also be calculated from the sample data.
- **Degrees of Freedom(df):** Calculate from the sample as df = total no. of observations - 1

In [1]:

```python
## Import the packages
import numpy as np
from scipy import stats
import math

# For visualizing distributions - optional
import seaborn as sns
import matplotlib.pyplot as plt
sample = np.array([122.09, 100.64, 125.77, 120.32, 118.25,  96.47, 111.4 , 80.66,
       110.77, 111.14, 102.9 , 114.54,  88.09,  98.59,  87.07, 110.43,
       101.9 , 123.89,  97.03, 116.23, 108.3 , 112.82, 119.57, 131.38,
       128.39])
```

```
# Population mean (μ)
mu = 100

# Sample mean (x̄) using NumPy mean()
x_bar = sample.mean()

# Sample Stadrad Deviation (sigma) using Numpy
sigma = np.std(sample,ddof=1)

# Sample size (n)
n = len(sample)

# Degrees of Freedom
df = n-1

# Difference in sample mean
diff = x_bar - mu


# Print the findings
print ('The sample contains', n, 'observations, having a mean of', x_bar, "and a standard deviation (si
gma) = ", sigma,
       ", with", df, 'degrees of freedom. The difference between sample and population means is:', diff
)

# The sample contains 25 observations, having a mean of 109.5456
# and a standard deviation (sigma) =  13.069276668584225 ,
# with 24 degrees of freedom.
# The difference between sample and population means is: 9.54
```

```
The sample contains 25 observations, having a mean of 109.5456 and a standard deviation (sigma) =  13.3
38774643871902 , with 24 degrees of freedom. The difference between sample and population means is: 9.5
45599999999993
```

The sample mean is $9.54 per sale higher than the population mean, which indicates that, at least superficially, the training program appears to have improved performance.

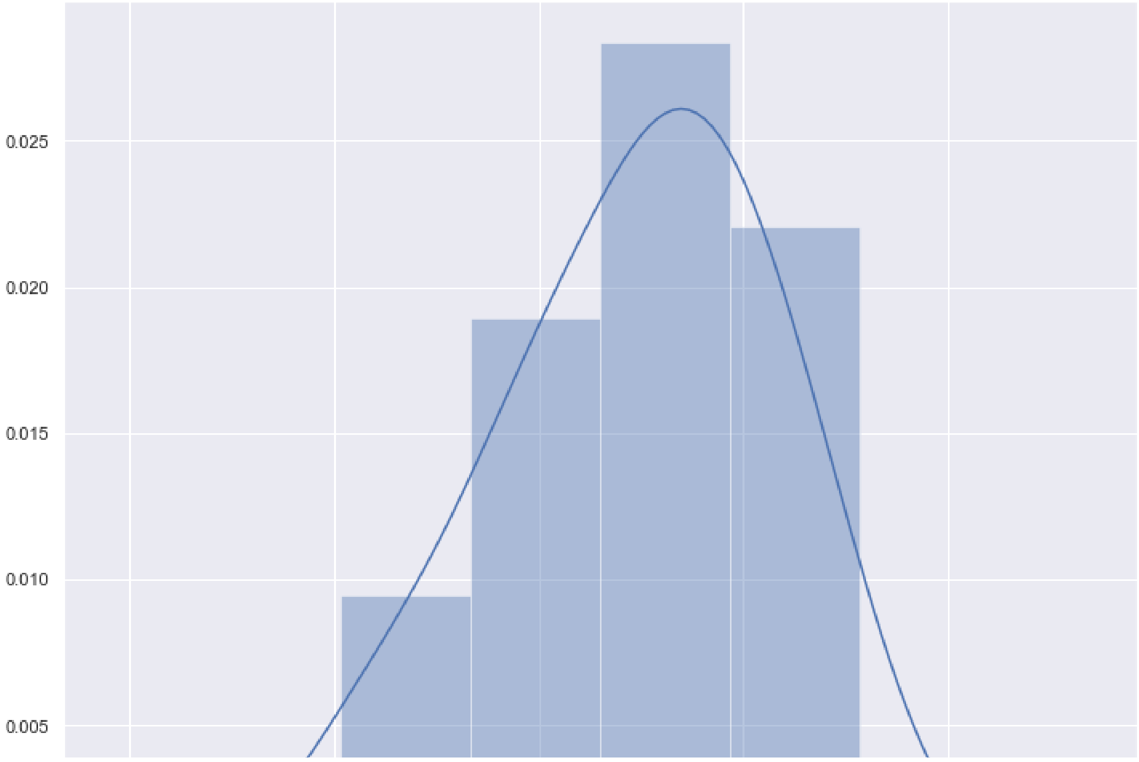
Is the sample roughly normally distributed?

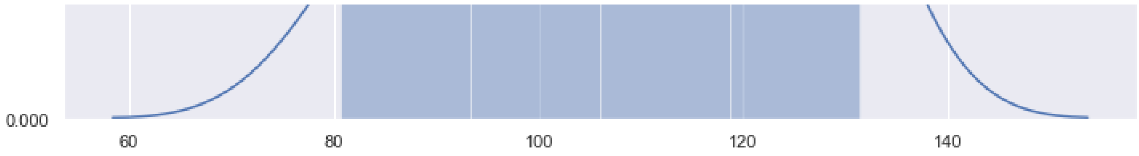
In [2]:

```
sns.set(color_codes=True)
sns.set(rc={'figure.figsize':(12,10)})
sns.distplot(sample)
```

Out[2]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2667e6ef080>
```





Step 1: Write your null and alternative hypothesis statements

As you are trying to monitor a change in the sales performance after the training, the null-hypothesis addresses the fact that there is no change and sales performance before and after the training is exactly the same.

$\boldsymbol{H_0}$: *The null hypothesis is that there is no difference in sales, so:*

> $H_0 : \mu$ = $100.

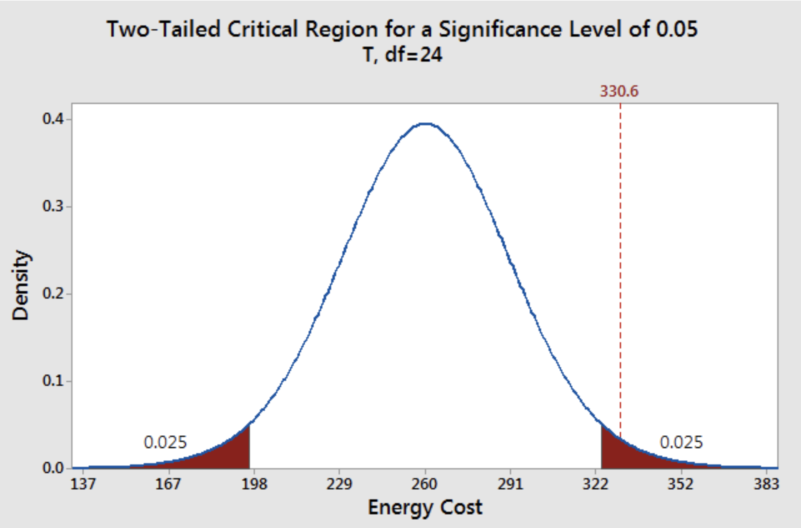
This is the one that you are testing. Our alternate hypothesis should address the expected change in the sales performance i.e. the sales performance has increased and the mean of sales post-training is greater than 100.

$\boldsymbol{H_1}$: *The alternative hypothesis is that there is a change i.e. the mean sales increased.*

> $H_1 : \mu$ > $100.

Step 2: Choose a Significance Level (Alpha)

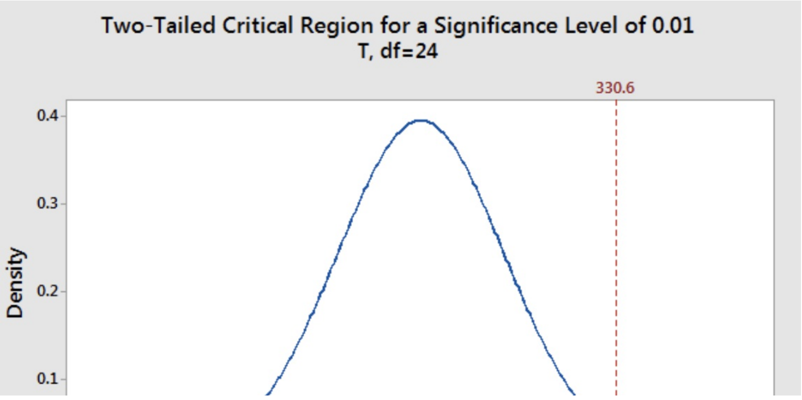
The significance level, also denoted as alpha or α, is the probability of rejecting the null hypothesis when it is true. For example, a significance level of 0.05 indicates a 5% risk of concluding that a difference exists when there is no actual difference. Look at the following graphs for a better understanding:

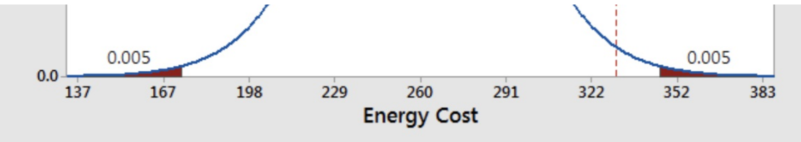


In the graph above, the two shaded areas are equidistant from the null hypothesis value and each area has a probability of 0.025, for a total of 0.05. In statistics, you call these shaded areas the critical regions for a two-tailed test. If the population mean is 260, you'd expect to obtain a sample mean that falls in the critical region 5% of the time. The critical region defines how far away our sample statistic must be from the null hypothesis value before you can say it is unusual enough to reject the null hypothesis.

Our sample mean (330.6) falls within the critical region, which indicates it is statistically significant at the 0.05 level.

You can also see if it is statistically significant using the other common significance level of 0.01.





The two shaded areas each have a probability of 0.005, the two of which add up to a total probability of 0.01. This time the sample mean does not fall within the critical region, and you fail to reject the null hypothesis. This comparison shows why you need to choose your significance level before you begin your study. It protects you from choosing a significance level because it conveniently gives you significant results!

Using the graph, data scientists are able to determine that their results are statistically significant at the 0.05 level without using a p-value. However, when you use the numeric output produced by statistical software, you'll need to compare the p-value to your significance level to make this determination.

For Acme's experiment, you can assume an α of 0.05.

Step 3: Calculate the t-statistic

The sample looks like a nicely shaped normal distribution. After fulfilling the three requirements for a t-test mentioned above i.e. normality, independence, and randomness, we are ready to calculate our t statistic using the formula for one-sample t-test given as:

$$t = \frac{\bar{x} - \mu}{\frac{s}{\sqrt{n}}}$$

> Using the formula given above, calculate the t-value in Python:

In [3]:

```
# Calculate Sigma
t = (x_bar -  mu)/(sigma/np.sqrt(n))
t
# 3.578139767278185
```

Out[3]:

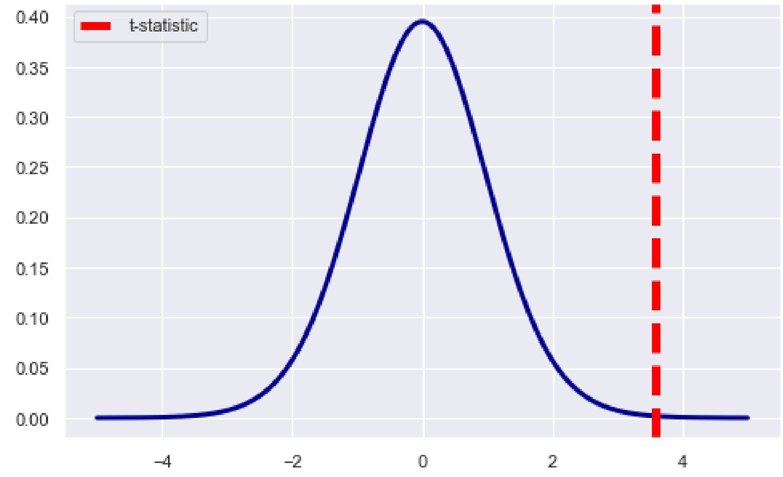
```
3.578139767278185
```

The sample generated a t-statistic of around 3.58. Where in the t-distribution is this located? Let's try visualizing the calculated t-statistic on top of a PDF.

In [4]:

```
# generate points on the x axis between -5 and 5:
xs = np.linspace(-5, 5, 200)

# use stats.t.pdf to get values on the probability density function for the t-distribution
# the second argument is the degrees of freedom
ys = stats.t.pdf(xs, df, 0, 1)

# initialize a matplotlib "figure"
fig = plt.figure(figsize=(8,5))

# get the current "axis" out of the figure
ax = fig.gca()

# plot the lines using matplotlib's plot function:
ax.plot(xs, ys, linewidth=3, color='darkblue')

# plot a vertical line for our measured difference in rates t-statistic
```

```
ax.axvline(t, color='red', linestyle='--', lw=5,label='t-statistic')
ax.legend()
plt.show()
```



Step 4: Calculate critical value (find rejection region)

Note that a positive t-value indicates that the sample mean is greater than the population mean and vice versa. This means that the sample's average sales performance post-training is greater than average population sales performance.

This sounds like good news, **BUT** is the increase high enough to reject the null hypothesis and accept that there is a significant increase in the mean of post-training sales performance, or is it just by chance. It's possible to calculate a critical t-value with a t-table and also by using Python `scipy.stats` module.

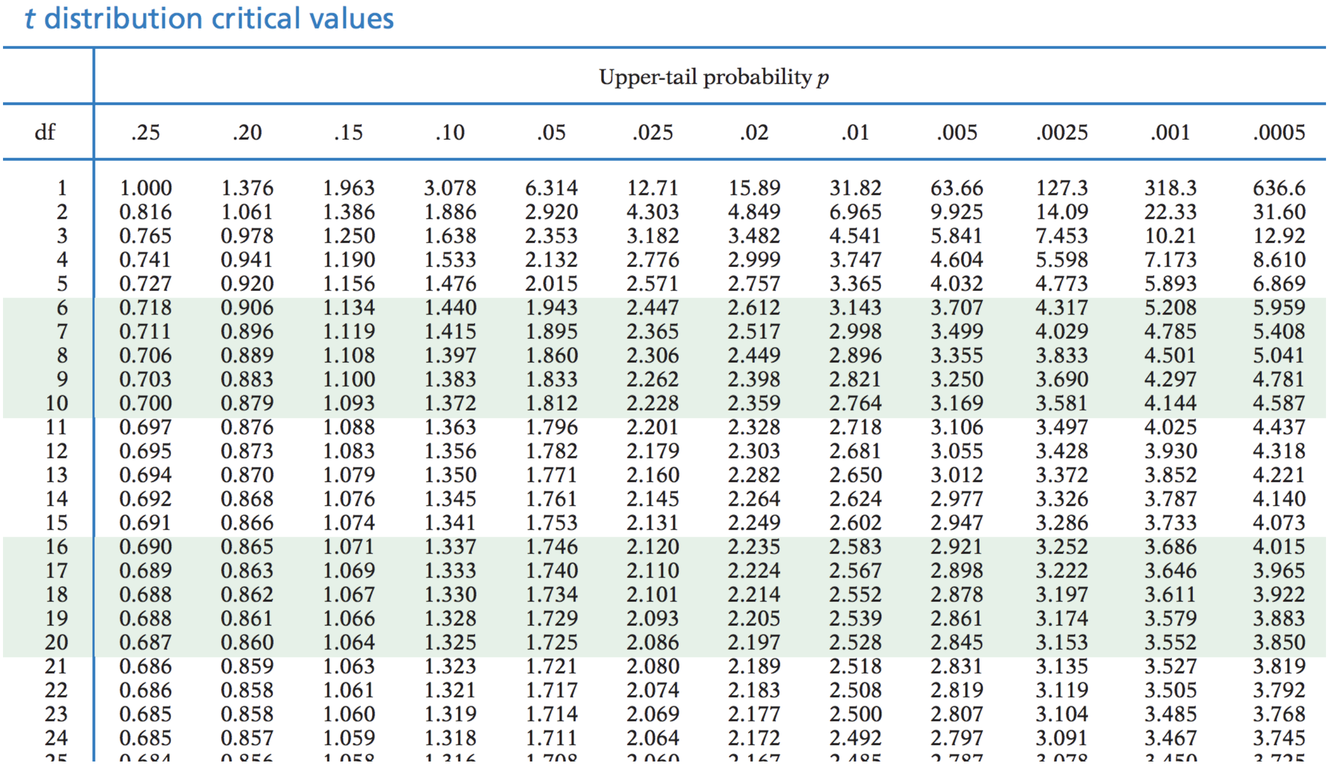
The critical value approach involves determining "likely" or "unlikely", by determining whether or not the observed test statistic is more extreme than would be expected if the null hypothesis were true. This involves comparing the observed test statistic to some cutoff value, called the **"critical value"**.

> If the test statistic is more extreme than the critical value, then the null hypothesis is rejected in favor of the alternative hypothesis. If the test statistic is not as extreme as the critical value, then the null hypothesis is not rejected.

You need two values to find this:

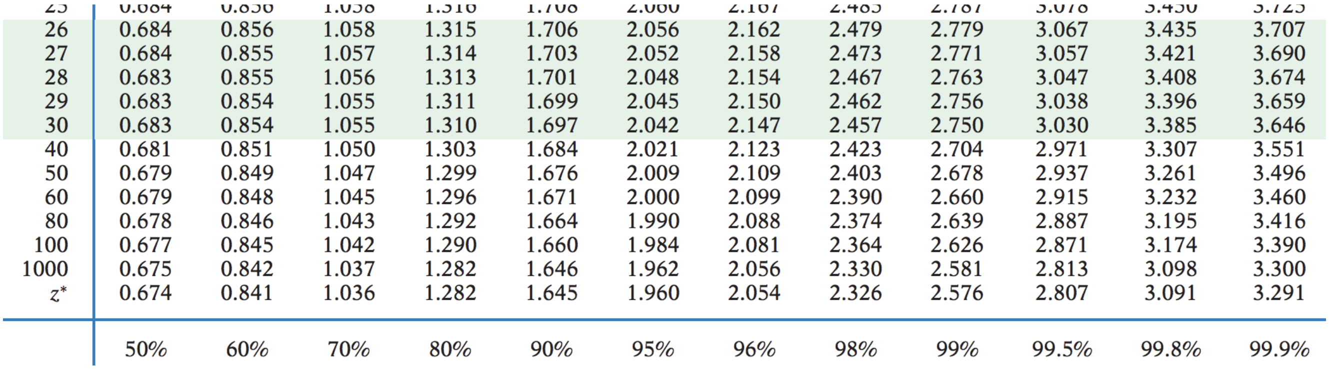
The **alpha level**: given as 5% in the question.

Degrees of freedom, which is the number of items in the sample (n) minus 1: 25 – 1 = 24.



t distribution critical values

df	Upper-tail probability *p*											
	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	0.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	0.765	0.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	0.741	0.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	0.727	0.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	0.718	0.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	0.711	0.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	0.706	0.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	0.703	0.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	0.700	0.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	0.697	0.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	0.695	0.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	0.694	0.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	0.692	0.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	0.691	0.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	0.690	0.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	0.689	0.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	0.688	0.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	0.688	0.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	0.687	0.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	0.686	0.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	0.686	0.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	0.685	0.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	0.685	0.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	0.684	0.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725



25	0.684	0.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	0.684	0.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	0.684	0.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	0.683	0.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	0.683	0.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	0.683	0.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	0.681	0.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	0.679	0.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	0.679	0.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	0.678	0.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	0.677	0.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	0.675	0.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
z^*	0.674	0.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%

You use a one-tailed t-test towards the positive (right side of the t-distribution) to identify an increase in the sales performance.

Look up 24 degrees of freedom in the left column and a p-value of 0.05 (from 5% alpha level - 95% confidence level) in the top row. The intersection is `1.711`. This is our one-sample critical t-value.

For the Null hypothesis to be true, what this critical value means is that you would expect most values to fall under 1.711. If your calculated t-value (from Step 4) falls within this range, the null hypothesis is likely true and you would fail to reject the null hypothesis.

This value can also be calculated in Python using `scipy.stats` module using `ppf()` (Percent Point Function) as `scipy.stats.t.ppf(1-alpha, df)`.

Let's calculate the critical t using this formula and confirm our earlier findings.

In [5]:

```
# Calculate critical t-value
t_crit = np.round(stats.t.ppf(1 - 0.05, df=24),3)
t_crit
# 1.711
```

Out[5]:

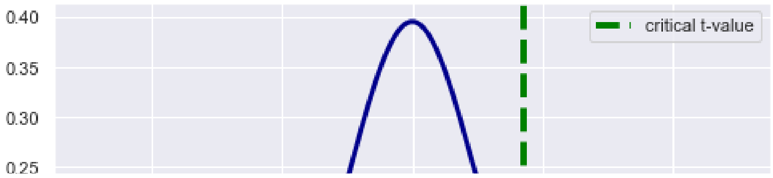
```
1.711
```

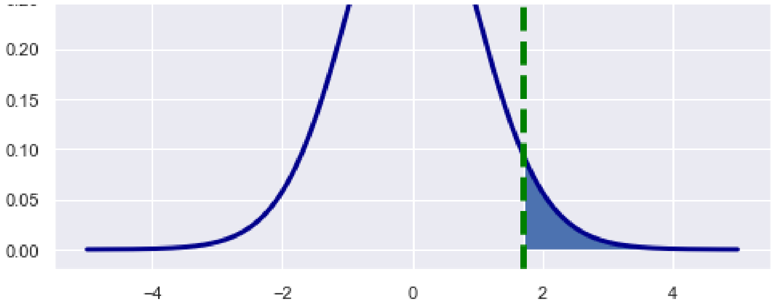
As you can see, the critical value returned from the function (rounded off 2 to two decimal places) is the same as the one you should have found in the t-distribution table i.e. 1.711.

Using matplotlib, you can graph the rejection region as such. Any t-statistic that falls within the shaded region to the right will cause the hypothesis to be rejected.

In [6]:

```
# generate points on the x axis between -5 and 5:
xs = np.linspace(-5, 5, 200)

# use stats.t.pdf to get values on the probability density function for the t-distribution
# the second argument is the degrees of freedom
ys = stats.t.pdf(xs, df, 0, 1)

# initialize a matplotlib "figure"
fig = plt.figure(figsize=(8,5))

# get the current "axis" out of the figure
ax = fig.gca()

# plot the lines using matplotlib's plot function:
ax.plot(xs, ys, linewidth=3, color='darkblue')


ax.axvline(t_crit,color='green',linestyle='--',lw=4,label='critical t-value')
ax.legend()
ax.fill_betweenx(ys,xs,t_crit,where= xs > t_crit)
plt.show()
```





Step 5: Compare t-value with critical t-value to accept or reject the Null Hypothesis

Any t-value which is greater than 1.711 will fall into the shaded region in the above figure. t-values greater than 1.711 would reflect an "extreme" result and can be used to reject the null hypothesis.

Your calculated t-value, known as the t-statistic is 3.65, which is greater than 1.711 and hence the results can be called "statistically significant" and will allow researchers to reject the null hypothesis and with 95% confidence state that:

We are 95% sure that the mean sales performance post training is higher than the population mean prior to training.

NOTE: This calculation can also be performed using the `ttest_1samp` function in `SciPy.stats` indicated here:

> **scipy.stats.ttest_1samp(a, popmean, axis=0, nan_policy='propagate')**

Where a is the sample mean ($\bar{x}$) and popmean (μ) is the population mean. This function returns the t-value and p-value for the sample. Here, you are using a one-tailed t-test as you are looking for an increase in sales performance.

In [7]:

```
results = stats.ttest_1samp(a= sample, popmean= mu)
print ("The t-value for sample is", round(results[0], 2), "and the p-value is", np.round((results[1]),
4))
#  Print results
# The t-value for sample is 3.58 and the p-value is 0.0015
```

```
The t-value for sample is 3.58 and the p-value is 0.0015
```

You can use our null and alternate hypotheses defined earlier to state the results from our findings.

In [8]:

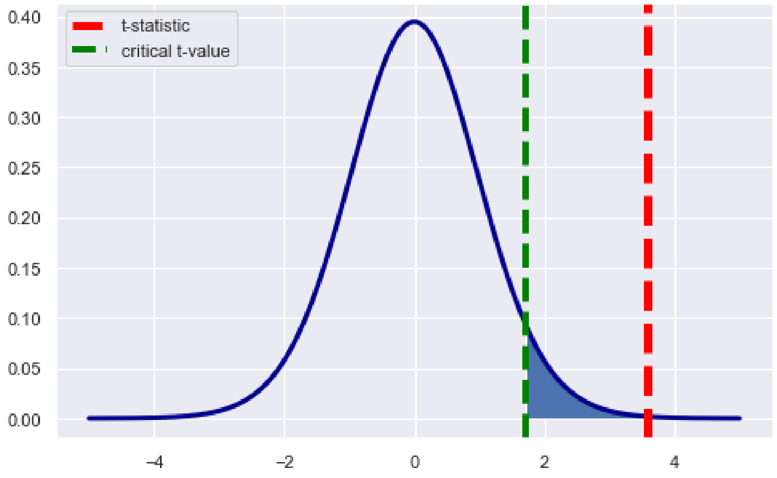
```
if (results[0]>t_crit) and (results[1]<0.05):
    print ("Null hypothesis rejected. Results are statistically significant with t-value =",
           round(results[0], 2), "and p-value =", np.round((results[1]), 4))
else:
    print ("Null hypothesis is Accepted")
```

```
Null hypothesis rejected. Results are statistically significant with t-value = 3.58 and p-value = 0.001
5
```

It's also possible to visualize where the calculated t-statistic is compared to the critical t-value:

In [9]:

```
# generate points on the x axis between -5 and 5:
xs = np.linspace(-5, 5, 200)

# use stats.t.pdf to get values on the probability density function for the t-distribution
# the second argument is the degrees of freedom
ys = stats.t.pdf(xs, df, 0, 1)

# initialize a matplotlib "figure"
fig = plt.figure(figsize=(8,5))

# get the current "axis" out of the figure
ax = fig.gca()

# plot the lines using matplotlib's plot function:
ax.plot(xs, ys, linewidth=3, color='darkblue')

# plot a vertical line for our measured difference in rates t-statistic
# plot a vertical line for our measured difference in rates t-statistic
ax.axvline(t, color='red', linestyle='--', lw=5,label='t-statistic')

ax.axvline(t_crit,color='green',linestyle='--',lw=4,label='critical t-value')
ax.fill_betweenx(ys,xs,t_crit,where= xs > t_crit)

ax.legend()
plt.show()
```



Summary

In this lesson, you saw a quick introduction to hypothesis testing using frequentists methods with t-values and p-values. You saw how a one sample t-test can be applied to contexts where the population mean is unknown and you have a limited amount of sample data. You looked at all the stages required for such hypothesis testing with a description of steps and also, how to perform these steps in Python.

Type I and Type II errors

Introduction

In hypothesis testing, you are performing statistical tests to determine whether you believe a statement to be true or false. This initial statement you are testing is called the **null hypothesis**. One common example of this is whether you believe two populations to be statistically different from one another. For example, you might be interested in testing if a new website layout is more effective at getting customers to make a purchase. In order to determine if the new layout was indeed effective, you would compare statistics, such as the average number of purchases in a given day, before and after the change.

There are times, however, when researchers reject the null hypothesis when they should have not rejected it. The opposite might happen as well, where you might fail to reject the null hypothesis when it should have been rejected. Data Scientists refer to these errors as type I and type II errors, respectively. You will soon dive into each one in more detail.

Objectives

You will be able to:

- Define Type I and Type II errors
- Describe the relationship between alpha and Type I errors
- Differentiate how Type I and Type II errors relate to the p and z-value

Alpha and Type I Errors

When conducting hypothesis testing, there will almost always be the chance of accidentally rejecting a null hypothesis when it should not have been rejected. Data scientists have the ability to choose a confidence level, alpha (α) that they will use as the threshold for accepting or rejecting the null hypothesis. This confidence level is also the probability that you reject the null hypothesis when it is actually true. This scenario is a type I error, more commonly known as a **False Positive**.

Here is a scenario that will better explain how a type I error might occur:

Say that you flipped a coin 30 times and get a total of 23 heads. The first thought in your head is, is this a fair coin? With that you can create the following null hypothesis:

Null Hypothesis: This coin is fair.

Alternative Hypothesis: This coin is not fair.

Or expressed mathematically:

$H_0 : \mu = 0.5$

$H_1 : \mu \neq 0.5$

The null hypothesis is assumed to be true unless there is overwhelming evidence to the contrary. To quantify this, you must determine what level of confidence for which you will reject the null hypothesis. If a researcher was to set **alpha (α) = .05**, this indicates that there is a 5% chance that you will reject the null hypothesis when it is actually true. Another way to think about this is that if you repeated this experiment 20 times, you would expect to see the hypothesis rejected, purely by chance, one time. The threshold for alpha varies significantly depending on the scientific discipline. Physics, for example, often require that findings are significant to the an alpha level of 0.0000003 or, in other words, one would expect results to occur by chance at most one out of 3.5 million trials! For most other disciplines, an α level of 0.05 is enough to prove some results are statistically significant.

Beta and Type II Errors

Another type of error is beta (β), which is the probability that you fail to reject the null hypothesis when it is actually false. Type II errors are also referred to as **False Negatives**. Beta is related to something called *Power*, which is the probability of rejecting the null hypothesis given that it actually is false. Mathematically, *Power* = 1 - β. When designing an experiment, scientists will frequently choose a power level they want for an experiment and from that obtain their type II error rate.

Balancing Type I and Type II Errors Examples

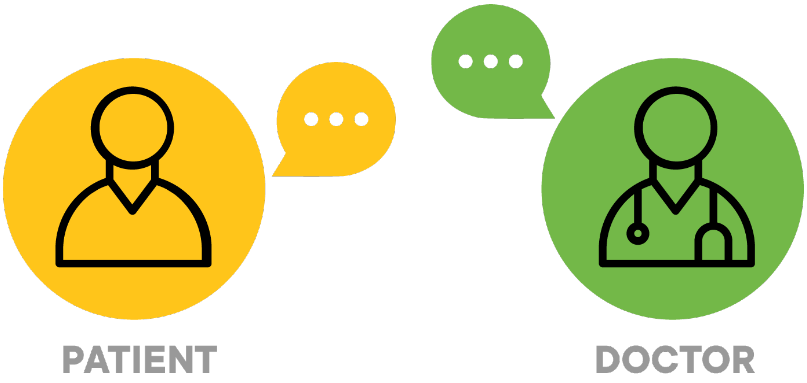
Different scenarios call for scientists to minimize one type of error over another. The two error types are inversely related to one other; reducing type I errors will increase type II errors and vice versa. Let's go through some different real-life scenarios to determine when it would be better to have a type I or type II error. Imagine you are on a jury and need to decide if someone will be sent to prison for life for a crime. Given that you don't know the truth as to whether or not this individual committed a crime, which would be worse, a type I or a type II error here?

- $H_0 : defendant = innocent$
- $H_1 : defendant \neq innocent$

Hopefully, you said a type I error would be worse! A type I error would mean that you would send someone to jail when they were truly not guilty! In other words, the jury has rejected the null hypothesis that the defendant is innocent, even though he has not committed any crime. Of course, you would also not want to have a type II error because this would mean that someone actually has committed a crime, and the jury is letting them get away with it.

Let's take a look at an example of a medical scenario. A patient with symptoms of a consistent headache goes to a doctor's office and gets an MRI scan of their head because the doctor suspects the patient might have a brain tumor. Would it be worse to have a type I or type II error in this scenario?

- $H_0 : patient = healthy$
- $H_1 : patient \neq healthy$



Hopefully, you said a type II error would be worse! A type II error would mean that the patient actually has a brain tumor, but the doctor claims there is nothing wrong with them. In other words, the null hypothesis is that the person has no brain tumor and this hypothesis fails to be rejected, meaning the person is diagnosed as healthy when in actuality, they are far from it.

When scientists are designing experiments, they need to weigh the risks of type I and type II errors and make decisions about choosing alpha level and power, which you will cover in more detail soon, to optimize for whichever type of error they want to minimize.

Testing an Unfair Coin

In [1]:

```
import numpy as np
import scipy
```

Here you'll simulate an unfair coin with 75% chance of heads and 25% chance of tails. You'll then *flip* this coin 20 times and perform a test to determine whether you believe it to be fair or unfair.

In [2]:

```
n = 20 #Number of flips
p = .75 #Simulating an unfair coin
coin1 = np.random.binomial(n, p)
coin1
```

Out[2]:

```
13
```

In this case, you know the theoretical mean and standard deviation of a fair coin; it can be modeled by a binomial distribution with p = 0.5. In future cases, you'll often use a t-test (as you've already previewed) in order to compare samples, but don't know the overall population statistics.

The standard deviation of a binomial distribution is given by:

$$\sigma = \sqrt{n \cdot p \cdot (1 - p)}$$

So you would expect that for a sample of 20 elements, the standard deviation from the expected number of heads (10) for a fair coin should be:

In [3]:

```
sigma = np.sqrt(n*.5*(1-.5))
sigma
```

Out[3]:

```
2.23606797749979
```

And with that you can now calculate a p-value using a traditional z-test:

$$z = \frac{\bar{x}-\mu}{\sigma/\sqrt{n}}$$

Here, $\bar{x}$ is the number of heads, μ (mu) is the expected number of heads (10), σ (sigma) is the standard deviation (calculated above) and n is the number of observations (20).

In [4]:

```
z = (coin1 - 10) / (sigma / np.sqrt(n))
z
```

Out[4]:

```
6.0
```

Finally, you can take your z-score and apply standard lookup tables based on your knowledge of the normal distribution to determine the probability

In [5]:

```
import scipy.stats as st
```

In [6]:

```
st.norm.sf(np.abs(z))
```

Out[6]:

```
9.865876450376946e-10
```

This is an absolutely tiny p-value, meaning that you can reject the null hypothesis *this coin is fair* and conclude that the coin is unfair!

Here is a demonstration of how the average p-values change as the size of the sample increases.

In [7]:

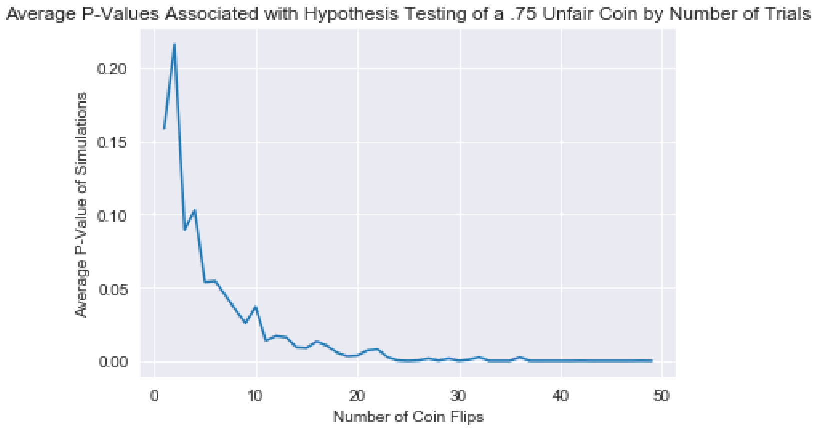
```
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('darkgrid')
%matplotlib inline
```

In [8]:

```
#How many times would you have to flip a 75% heads coin to determine it was false?
p_vals = []
#Iterate through various numbers of trials
for n in range(1,50):
    #Do multiple runs for that number of samples to compare
    p_val = []
    for i in range(200):
        p = .75 # Simulating an unfair coin
        n_heads = np.random.binomial(n, p)
        mu = n / 2
        sigma = np.sqrt(n*.5*(1-.5))
        z  = (n_heads - mu) / (sigma / np.sqrt(n))
        p_val.append(st.norm.sf(np.abs(z)))
    p_vals.append(np.mean(p_val))
plt.plot(list(range(1,50)), p_vals)
plt.title('Average P-Values Associated with Hypothesis Testing of a .75 Unfair Coin by Number of Trials
')
plt.ylabel('Average P-Value of Simulations')
plt.xlabel('Number of Coin Flips')
```

Out[8]:

```
Text(0.5, 0, 'Number of Coin Flips')
```



Summary

Great! You now know what type I and type II errors are. Let's go and practice your knowledge!

Resampling Methods

Introduction

Resampling techniques are modern statistical techniques that involve taking repeated subsamples from a sample. These procedures tend to be computationally intensive, since they involve computing statistics of a subsample, creating new subsamples and repeating the process thousands or perhaps millions of times. This can allow for additional analysis of the subsamples leading to increased confidence and knowledge of the larger population. The three main techniques we will discuss here are bootstrapping, jackknife, and permutation tests.

Objectives

You will be able to:

- Identify when resampling is used
- Describe the process of bootstrapping
- Describe permutation testing

Jackknife and Bootstrapping

Let's start by defining the sampling methodology for these techniques. The bootstrap method works by taking random samples with replacement from the original sample of size n. In contrast, the jackknife, the older of the two methods, works by taking samples by removing one, or more, observations at a time. Each one of these (n-1) sized sub-samples is aggregated to create the new jackknife sample. The purpose of these resampling methods is to be able to increase the size of our samples without having to actually go out and obtain more samples. Resampling methods attempt to estimate the variability of point estimators derived from the original samples.

The motivating principle behind both is that by analyzing the variance of parameter estimates from these synthetic samples, we can also gauge the variance of our point estimate for the population itself. For example, we might take an original sample from our population and then use the jackknife or bootstrapping method to generate additional synthetic samples. By calculating the point estimate of interest for these synthetic samples, we can better gauge the confidence interval and variability of our original point estimator.

Permutation Tests

Another related methodology is permutation tests. Permutation tests can be used in lieu of assumed parameter distributions for any statistical test. For example, we discussed the central limit theorem: that when taking the mean of a repeated sample from a population, the means of these samples will form a normal distribution. From this, we were then able to extrapolate confidence intervals surrounding our estimate for the mean of the entire population by assuming that our sample mean was from a normal distribution. This allowed us to define our confidence bands associated with various levels of type I errors which we set with α. In a hypothesis test, we used the same procedure to calculate the probability of a given sample, and based on alpha, rejected or confirmed the null hypothesis. In a permutation test, rather then assume the distribution itself and calculate p-values, we would calculate all permutations of our relabeling our data and compute the parameter statistic in question for these permutations.

For example, let's say we had two samples, one with 37 observations and the other with 45 observations. We calculate the mean of both samples and wish to perform a hypothesis test with a 5% confidence interval for whether the two samples belong to the same overall population. In our previous work, we would use a t-test to perform this comparison. The permutation test alternative would be to compare the difference in these sample means to the difference in sample means of all possible permutations of 37-45 splits between our 82 data points. In other words, we compare the difference between our actual sample means to the difference in sample means between all variations of all those 82 points in order to calculate our p-values and determine whether we accept or reject the null-hypothesis.

> **Note**: While it's called a permutation test, calculating all of the possible combinations of the observations into two groups is a more pragmatic approach. After all, you are comparing the sample means of the groups and as such the order of group members is irrelevant. When you implement permutation tests in the upcoming lab, you'll use combinations to make the problem computationally feasible. Even so, as you will see, the size of possible variations can quickly explode leading to other estimations of the permutation test, which you'll investigate towards the end of the section.

Additional Resources

- http://hydrodictyon.eeb.uconn.edu/eebedia/images/9/9d/FelsensteinChap20.pdf
- https://www.scss.tcd.ie/Rozenn.Dahyot/453Bootstrap/05_Permutation.pdf

Summary

In this lesson, we continued discussing non-parametric statistics and investigated resampling techniques. This included bootstrapping, jackknife, and permutation tests. In the upcoming lab, you'll define functions that implement these techniques and then use them to conduct statistical simulations and tests.

Hypothesis Testing - Recap

Introduction

You just learned how to create an experiment and interpret the results! Let's review some of the specific things you have learned.

Key Takeaways

Some of the key takeaways from this section include:

- It's important to have a sound approach to experimental design to be able to determine the significance of your findings
- Start by examining any existing research to see if it can shed light on the problem you're studying
- Start with a clear alternative and null hypothesis for your experiment to "prove"
- It's important to have a thoughtfully selected control group from the same population for your trial to distinguish effect from variations based on population, time or other factors
- Your sample size needs to be selected carefully to ensure your results have a good chance of being statistically significant
- Your results should be reproducible by other people and using different samples from the population
- The p-value for an outcome determines how likely it is that the outcome could occur under the null hypothesis
- α is the marginal threshold at which we're comfortable rejecting the null hypothesis
- An α value of 0.05 is a common choice for many experiments
- Effect size measures just the size of the difference between two groups under observation, whereas statistical significance combines effect size with sample size
- A one sample t-test is used to determine whether a sample comes from a population with a specific mean
- A two sample t-test is used to determine if two population means are equal
- Type 1 errors (false positives) are when we accept an alternative hypothesis which is actually false
- The α that we pick is the likelihood that we will get a type 1 error due to random chance
- Type 2 errors (false negatives) are when we reject an alternative hypothesis which is actually true
- Resampling methods allow for improved precision in estimating sample statistics and validating models by using random subsets
- Common resampling techniques include bootstrapping, jackknifing and permutation tests