Inference for one proportion

Put your name here

Put the date here

Introduction

Our earlier work with simulations showed us that when the number of successes and failures is large enough, we can use a normal model as our sampling distribution model.

We revisit hypothesis tests for a single proportion, but now, instead of running a simulation to compute a P-value, we take the shortcut of computing the P-value directly from a normal model.

There are no new concepts here. All we are doing is revisiting the rubric for inference and making the necessary changes.

Instructions

Presumably, you have already created a new project and downloaded this file into it. Please knit the document and work back and forth between this R Markdown file and the PDF output as you work through this module.

When you are finished with the assignment, knit to PDF one last time, proofread the PDF file **carefully**, export the PDF file to your computer, and then submit your assignment.

Sometimes you will be asked to add your own R code. That will appear in this document as a code chunk with a request for you to add your own code, like so:

```
## Add code here to [do some task]...
```

Be sure to remove the line `## Add code here to [do some task]...` when you have added your own code.

Sometimes you will be asked to type up your thoughts. That will appear in the document as follows:

Please write up your answer here.

Again, please be sure to remove the line "Please write up your answer here" when you have written up your answer. In these areas of the assignment, please use contextually meaningful full sentences/paragraphs (unless otherwise indicated) and proper spelling, grammar, punctuation, etc. This is not R code, but rather a free response section where you talk about your analysis and conclusions. If you need to use some R code as well, you can use inline R code inside the block between `\begin{answer}` and `\end{answer}`, or if you need an R code chunk, please go outside the `answer` block and start a new code chunk.

Load Packages

We load the standard `mosaic` package as well as the `broom` package for tidy output and the `openintro` package to access data on heart transplant candidates.

```
library(openintro)
library(broom)
library(mosaic)
```

Revisiting the rubric for inference

Instead of running a simulation, we are going to assume that the sampling distribution can be modeled with a normal model as long as the conditions for using a normal model are met.

Although the rubric has not changed, the use of a normal model changes quite a bit about the way we go through the other steps. For example, we won't have simulated values to give us a histogram. Instead, we'll go straight to graphing a normal model. We won't compute the percent of our samples that are at least as extreme as our test statistic to get the P-value. The P-value from a normal model is found directly from the model itself using the `pdist` command.

What follows is a fully-worked example of inference for one proportion. After the hypothesis test (sometimes called a one-proportion z-test for reasons that will become clear), we also follow up by computing a confidence interval. **From now on, we will consider inference to consist of a hypothesis test and a confidence interval.** Whenever you're asked a question that requires statistical inference, you should follow both the rubric steps for a hypothesis test and for a confidence interval.

The example below will pause frequently for commentary on the steps, especially where their execution will be different from what you've seen before when you used simulation. When it's your turn to work through another example on your own, you should follow the outline of the rubric, but you should **not** copy and paste the commentary that accompanies it.

Research question

Data from the Stanford University Heart Transplant Study is located in the `openintro` package in a data frame called `heartTr`. From the help file we learn, "Each patient entering the program was designated officially a heart transplant candidate, meaning that he was gravely ill and would most likely benefit from a new heart." Survival rates are not good for this population, although they are better for those who receive a heart transplant. Do heart transplant recipients still have less than a 50% chance of survival?

Inference for one proportion

Exploratory data analysis

Use data documentaton (help files, code books, Google, etc.), the str command, and other summary functions to understand the data.

[You should type `library(openintro)`, then `?heartTr` at the Console to read the help file.]

```r
str(heartTr)
```

```
## 'data.frame':    103 obs. of  8 variables:
##  $ id        : int  15 43 61 75 6 42 54 38 85 2 ...
##  $ acceptyear: int  68 70 71 72 68 70 71 70 73 68 ...
##  $ age       : int  53 43 52 52 54 36 47 41 47 51 ...
##  $ survived  : Factor w/ 2 levels "alive","dead": 2 2 2 2 2 2 2 2 2 2 ...
##  $ survtime  : int  1 2 2 2 3 3 3 5 5 6 ...
##  $ prior     : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ transplant: Factor w/ 2 levels "control","treatment": 1 1 1 1 1 1 1 2 1 1 ...
##  $ wait      : int  NA NA NA NA NA NA NA 5 NA NA ...
```

Commentary: The variable of interest is `survived`, which is coded as a factor variable with two categories, "alive" and "dead". Keep in mind that because we are interested in survival rates, the "alive" condition will be considered the "success" condition.[1]

There are 103 patients, but we are not considering all these patients. Our sample should consist of only those patients who actually received the transplant. The following table shows that only 69 patients were in the "treatment" group (meaning that they received a heart transplant.)

```r
table(heartTr$transplant)
```

```
##
##   control treatment
##        34        69
```

Prepare the data for analysis.

```r
heartTr_2 <- filter(heartTr, transplant == "treatment")
```

Commentary: Since we only want to look at those who received the transplant, we cannot use the whole data set `heartTr`. The `filter` command is useful for grabbing only certain rows from our data. We will filter the data for those who were in the `treatment` group and assign the resulting data frame to a new variable called `heartTr_2`.

The only unusual thing you might have spotted is the double equal sign. This is used whenever you want to check that two things are equal. Just to illustrate, go to the Console and type `x = 5`. Then type `x == 5`. The first command assigns the number 5 to the variable x (as you should be able to see in your Global Environment). The second command checks if x is equal to 5. (And since you just assigned x to be 5, it is `TRUE`.)

Make tables or plots to explore the data visually.

```r
table(heartTr_2$survived)
```

```
##
## alive  dead
##    24    45
```

```r
prop.table(table(heartTr_2$survived))
```

```
##
##     alive      dead
## 0.3478261 0.6521739
```

Hypotheses

Identify the sample (or samples) and a reasonable population (or populations) of interest.

The sample consists of 69 heart transplant recipients in a study at Stanford University. The population of interest is presumably all heart transplants recipients.

[1] Finally, a "success" condition that actually sounds successful!

Express the null and alternative hypotheses as contextually meaningful full sentences.

H_0: Heart transplant recipients have a 50% chance of survival.

H_A: Heart transplant recipients have less than a 50% chance of survival.

Commentary: it is slightly unusual that we are conducting a one-sided test. The standard default is typically a two-sided test. However, it is not for us to choose: the proposed research question is unequivocal in hypothesizing "less than 50%" survival.

Express the null and alternative hypotheses in symbols.

$H_0 : p = 0.5$

$H_A : p < 0.5$

Model

Identify the sampling distribution model.

We will use a normal model.

Check the relevant conditions to ensure that model assumptions are met.

- Random
  - Since the 69 patients are from a study at Stanford, we do not have a random sample of all heart transplant recipients. We hope that the patients recruited to this study were physiologically similar to other heart patients so that they are a representative sample. Without more information, we have no real way of knowing.
- 10%
  - 69 patients are definitely less than 10% of all heart transplant recipients.
- Success/failure

$$np = 69(0.5) = 34.5 \geq 10$$

$$n(1-p) = 69(0.5) = 34.5 \geq 10$$

Commentary: Notice something interesting here. Why did we not use the 24 patients who survived and the 45 who died as the successes and failures? In other words, why did we use np and $n(1-p)$ instead of $n\hat{p}$ and $n(1-\hat{p})$?

Remember the logic of inference and the philosophy of the null hypothesis. To convince the skeptics, we must assume the null hypothesis throughout the process. It's only after we present sufficient evidence that can we reject the null and fall back on the alternative hypothesis that encapsulates our research question.

Therefore, under the assumption of the null, the sampling distribution is the *null distribution*, meaning that it's centered at 0.5. All work we do with the normal model, including checking conditions, must use the null model with $p = 0.5$.

That's also why the numbers don't have to be whole numbers. If the null states that of the 69 patients, 50% are expected to survive, then we expect 50% of 69, or 34.5, to survive. Of course, you can't have half of a survivor. But these are not *actual* survivors. Rather, they are the expected number of survivors in a group of 69 patients *on average* under the assumption of the null.

Mechanics

Compute the test statistic.

```
surv_test <- tidy(prop.test(heartTr_2$survived, success = "alive"))
z <- (surv_test$estimate -  0.5)/sqrt(0.5 * (1 - 0.5) / 69)
```

The test statistic has a z-score of -2.5281029.

Commentary: We run the `prop.test` command to give us almost everything we need. As seen before, the `tidy` command from the `broom` package makes everything neat and gives us easy access to the various pieces of the output. What this command does not give us is a z-score. So we calculate it directly using the `estimate` ($\hat{p}$) and the mean and standard error of the null distribution.

Rememeber that are working under the assumption of the null hypothesis. This means that we use $p = 0.5$ everywhere in the formula for the standard error. In other words,

$$z = \frac{(0.348 - 0.5)}{\sqrt{\frac{0.5(1-0.5)}{69}}} = -2.53.$$

Either $\hat{p}$ or z could be considered the test statistic. If we use $\hat{p}$ as the test statistic, then we're considering the null model to be

$$N\left(0.5, \sqrt{\frac{0.5(1-0.5)}{69}}\right).$$

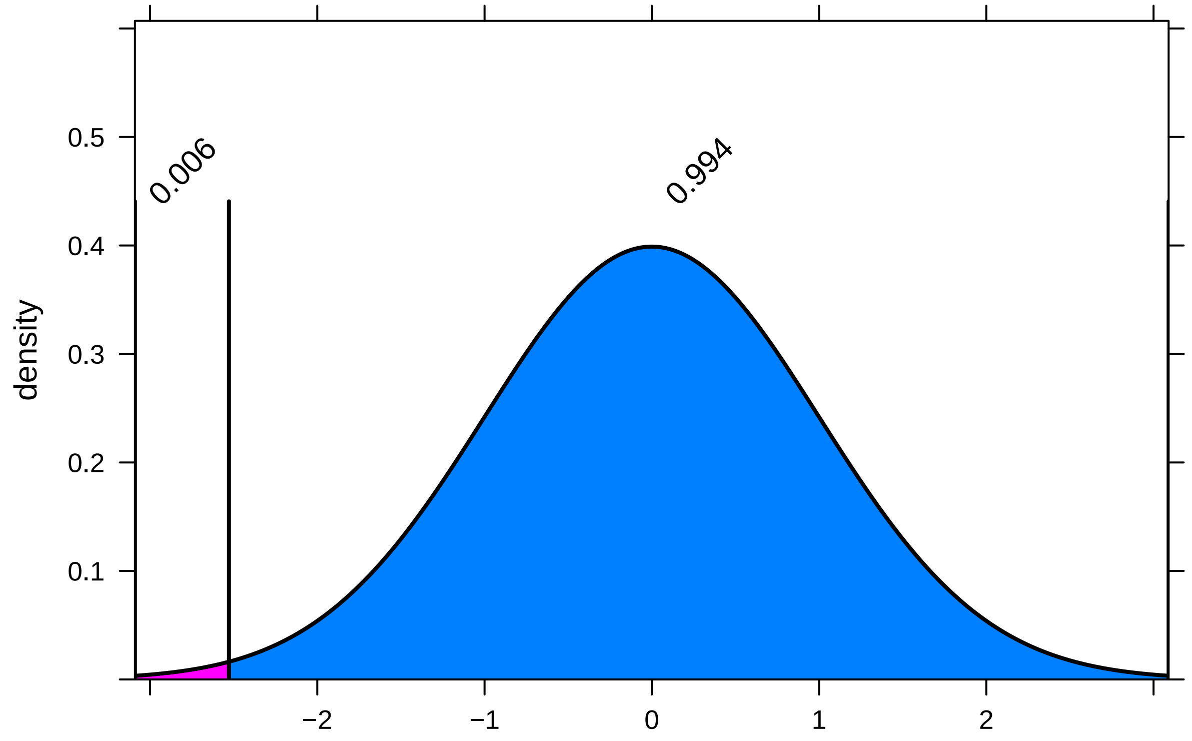
If we use z as the test statistic, then we're considering the null model to be the *standard normal model*

$$N(0,1).$$

The standard normal model is more intuitive and easier to work with, both conceptually, and in R. Therefore, we will henceforth consider z as the test statistic so that we can consider our null model to be the standard normal model. For example, knowing that our test statistic is more than four standard deviations to the left of the null value already tells us a lot. We can anticipate a small P-value leading to rejection of the null.

Plot the null distribution.

```
pdist("norm", q = z)
```

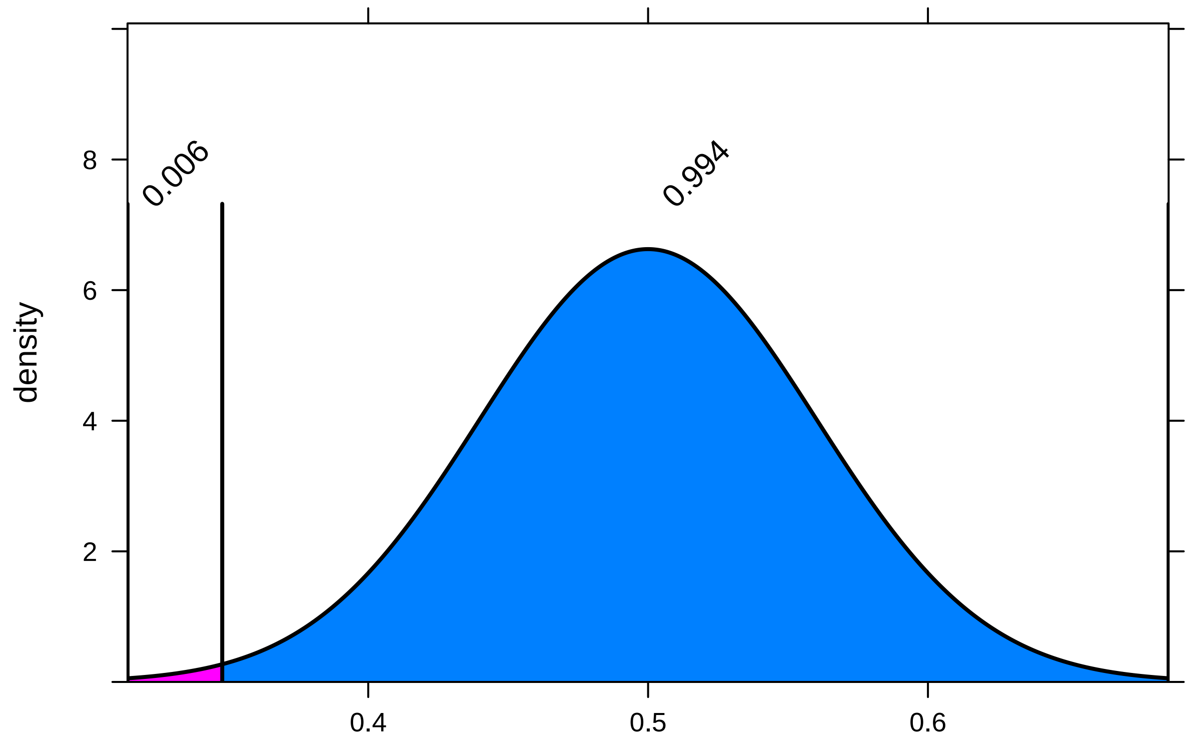


```
## [1] 0.005734037
```

Commentary: Notice how simple this `pdist` command is. One reason is that we're using z, which means that we are working with the standard normal model. As this is the default for the `pdist` command, there is no need to specify the `mean` and `sd` in the command.

Contrast this to the command we would need if using $\hat{p}$ as our test statistic:

```
pdist("norm", q = surv_test$estimate,
      mean = 0.5, sd = sqrt(0.5 * (1 - 0.5) / 69))
```

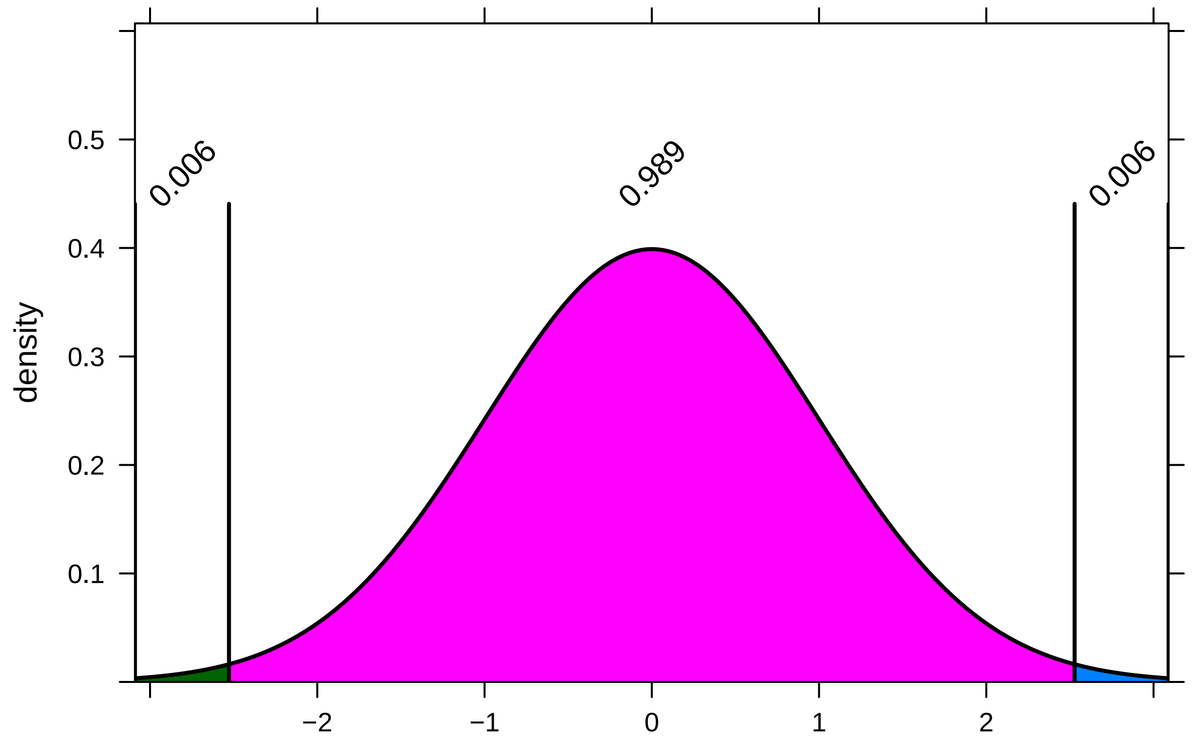


```
## [1] 0.005734037
```

This works and gives us the same answer, but it's a lot more complicated.

The other thing that makes this relatively simple is that we're running a one-sided test corresponding to the alternative hypothesis $p < 0.5$. Keep in mind that if we were running a two-sided test, we would need this instead:

```
pdist("norm", q = c(-z, z))
```



```
## [1] 0.994265963 0.005734037
```

Calculate the P-value.

```
pdist("norm", q = z, plot = FALSE)
```

```
## [1] 0.005734037
```

Commentary: Since we are running a one-sided test using the left tail (recall the alternative hypothesis was $p < 0.5$), the result of the `pdist` command is already the correct P-value. We don't need the plot again, so we can put `plot = FALSE` into the command.

The P-value can also be computed as part of the `prop.test` function:

```
surv_test$p.value
```

```
## [1] 0.01605262
```

However, under the hood, the `prop.test` command is not quite using the same method we're using. Therefore, the P-value is different. Also, the `prop.test` function assumes by default that you are performing a two-sided test. Therefore, if we want a one-sided p-value, we have to divide this answer by 2:

```
surv_test$p.value / 2
```

```
## [1] 0.008026308
```

Even this value isn't quite the same as the one generated from our normal model. Nevetheless, the two methods are close, so you should come to the same conclusion (i.e., reject the null or fail to reject the null).

If you look more deeply into the `prop.test` function, you might discover that there is a way to get it to run a one-sided test. However, doing so messes up the confidence interval that we'll need later in the rubric. Most of the time, it's a moot point because we will often be running two-sided tests.

Conclusion

State the statistical conclusion.

We reject the null hypothesis.

State (but do not overstate) a contextually meaningful conclusion.

We have sufficient evidence that heart transplant recipients have less than a 50% chance of survival.

Identify the possibility of either a Type I or Type II error and state what making such an error means in the context of the hypotheses.

As we rejected the null, we run the risk of making a Type I error. It is possible that the null is true and that there is a 50% chance of survival for these patients, but we got an unusual sample that appears to have a much smaller chance of survival.

Confidence interval

Conditions

- Random
  - Same as above.
- 10%
  - Same as above.
- Success/failure
  - There were 24 patients who survived and 45 who died in our sample. Both are larger than 10.

Commentary: In the "Confidence Interval" section of the rubric, there is no need to recheck conditions that have already been checked. The sample has not changed; if it met the "Random" and "10%" conditions before, it will meet them now.

So why recheck the success/failure condition?

Keep in mind that in a hypothesis test, we temporarily assume the null is true. The null states that $p = 0.5$ and the resulting null distribution is, therefore, centered at $p = 0.5$. The success/failure condition is a condition that applies to the normal model we're using, and for a hypothesis test, that's the null model.

By contrast, a confidence interval is making no assumption about the "true" value of p. The inferential goal of a confidence interval is to try to capture the true value of p, so we certainly cannot make any assumptions about it. Therefore, we go back to the original way we learned about the success/failure condition. That is, we check the actual number of successes and failures.

Calculation

```
surv_test$conf.low
```

```
## [1] 0.2398327
```

```
surv_test$conf.high
```

```
## [1] 0.4728747
```

Commentary: there's not much to do here as the confidence interval was already computed as a byproduct of the `prop.test` command.

Conclusion

We are 95% confident that the true proportion of heart transplant recipients who survive is captured in the interval (23.9832666%, 47.2874695%).

Commentary: Note that when we state our contextually meaningful conclusion, we also covert the decimal proportions to percentages. Humans like percentages a lot better.

Your turn

Follow the rubric to answer the following research question:

Some heart transplant candidates have already had a prior surgery. Use the variable `prior` in the `heartTr` data set to determine if less than 50% of patients have had a prior surgery. (To be clear, you are being asked to perform a one-sided test again.) Be sure to use the full `heartTr` data, not the modified `heartTr_2` from the previous example.

This time, you will not be given the rubric outline. Please copy and paste the steps of the rubric below. You may refer to the worked example above and modify it accordingly. Copying and pasting R code is fine; copying and pasting text is not. Please state everything in your own words. (Also, remember to strip out all the commentary. That is just exposition for your benefit in understanding the steps, but is not meant to form part of the formal inference process.)

Another word of warning: the copy/paste process is not a substitute for your brain. You will often need to modify more than just the names of the data frames and variables to adapt the worked examples to your own work. For example, if you run a two-sided test instead of a one-sided test, there are three or four steps that all have to be adjusted accordingly. Understanding the sampling disribution model and the computation of the P-value goes a long way toward understanding these changes that must be made. Do not blindly copy and paste code without understanding what it does. And you should **never** copy and paste text. All the sentences and paragraphs you write are expressions of your own analysis. They must reflect your own understanding of the inferential process.