**Jesse's**Course Notes and Screenshots

HarvardX

**Module No.4**

**Inference and Modeling**

**with Prof. Rafael Irizarry**

**Inference and Modeling**

**with Prof. Rafael Irizarry**

**Table of Contents**

[1. Parameters and Estimates 3](#_Toc24885350)

[1.1. Parameters and Estimates 3](#_Toc24885351)

[1.1.1. Sampling model parameters and estimates 3](#_Toc24885352)

[1.1.2. The Sample Average 5](#_Toc24885353)

[1.1.3. Polling versus Forecasting 5](#_Toc24885354)

[1.1.4. Properties of Our Estimate 6](#_Toc24885355)

[2. The Central Limit Theorem in Practice 7](#_Toc24885356)

[2.1. Central Limit Theorem in Practice 7](#_Toc24885357)

[2.1.1. The Central Limit Theorem in Practice 7](#_Toc24885358)

[2.1.2. Margin of Error 8](#_Toc24885359)

[2.1.3. A Monte Carlo Simulation for the CLT 8](#_Toc24885360)

[2.1.4. The Spread 10](#_Toc24885361)

[2.1.5. Bias: Why Not Run a Very Large Poll? 10](#_Toc24885362)

[3. Confidence Intervals and p-Values 12](#_Toc24885363)

[3.1.1. Confidence Intervals 12](#_Toc24885364)

[3.1.2. A Monte Carlo Simulation for Confidence Intervals 15](#_Toc24885365)

[3.1.3. The Correct Language 16](#_Toc24885366)

[3.1.4. Power 16](#_Toc24885367)

[3.1.5. p-Values 16](#_Toc24885368)

[4. Statistical Models 18](#_Toc24885369)

[4.1.1. Poll Aggregators 18](#_Toc24885370)

[4.1.2. Pollsters and Multilevel Models 21](#_Toc24885371)

[4.1.3. Poll Data and Pollster Bias 21](#_Toc24885372)

[4.1.4. Data-Driven Models 25](#_Toc24885373)

[5. Bayesian Statistics 27](#_Toc24885374)

[5.1.1. Bayesian Statistics 27](#_Toc24885375)

[5.1.2. Bayes' Theorem 27](#_Toc24885376)

[5.1.3. Bayes in Practice 29](#_Toc24885377)

[5.1.4. The Hierarchical Model 30](#_Toc24885378)

[6. Election Forecasting 33](#_Toc24885379)

[6.1.1. Election Forecasting 33](#_Toc24885380)

[6.1.2. Mathematical Representations of Models 34](#_Toc24885381)

[6.1.3. Predicting the Electoral College 39](#_Toc24885382)

[6.1.4. Forecasting 45](#_Toc24885383)

[6.1.5. The t-Distribution 48](#_Toc24885384)

[7. Association Tests 51](#_Toc24885385)

[7.1. Assessment 7.1: Association and Chi-Squared Tests 51](#_Toc24885386)

[7.1.1. Association Tests 51](#_Toc24885387)

[7.1.2. Chi-Squared Tests 54](#_Toc24885388)

**List of Figures**

[Figure 1. Sample of 25 beads from the urn 3](#_Toc19439856)

[Figure 2. Four different sample of the same poll 4](#_Toc19439857)

[Figure 3. Histogram and qq plot of X\_hat distribution 9](#_Toc19439858)

[Figure 4. Largest margin of error 10](#_Toc19439859)

[Figure 5. Average Yearly temperatures in New Haven 12](#_Toc19439860)

[Figure 6. First Few interval that were generated in our Monte Carlo simulation 15](#_Toc19439861)

[Figure 7. Visualization of what the intervals of these pollsters would have reported for the difference between Obama and Romne 19](#_Toc19439862)

[Figure 8. Histogram of reported spreads 22](#_Toc19439863)

[Figure 9. Plot of spread by pollster that have done more than 6 polls in the week before the election 24](#_Toc19439864)

[Figure 10. Histogram of the last reported result before the election for 15 pollsters 25](#_Toc19439865)

[Figure 11. Distribution of batting averages for all players with more than 500 at bats during the season 2010,2011 and 2012 30](#_Toc19439866)

[Figure 12. Simulated data of 6 polls taken by 5 pollsters under the model 35](#_Toc19439867)

[Figure 13. Simulated data of 6 polls taken by 5 pollsters under the model 36](#_Toc19439868)

[Figure 14. Posterior mean vs average group by number of polls taken 42](#_Toc19439869)

[Figure 15. Histogram of outcomes of Monte Carlo simulation of 1000 elections 43](#_Toc19439870)

[Figure 16. Histogram of outcomes of Monte Carlo simulation of 1000 elections with genreal bias included 44](#_Toc19439871)

[Figure 17. Histogram of Ipsos spread distributions 45](#_Toc19439872)

[Figure 18. Distribution of spreads across time and pollsters 46](#_Toc19439873)

[Figure 19. Estimates for the difference of actual percentages for the 2 main candidates of the 2016 U.S. election 47](#_Toc19439874)

[Figure 20. T distribution with degree of freedom 3, 5 and 15 48](#_Toc19439875)

[Figure 21. Q-Q plot of sample data versus the normal distribution 49](#_Toc19439876)

1. Parameters and Estimates
   1. Parameters and Estimates
      1. Sampling model parameters and estimates

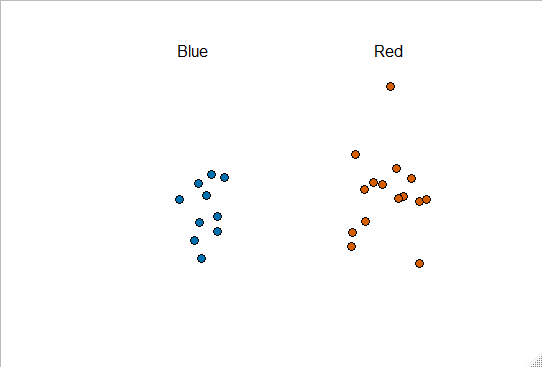
In this course, we going to see the link between all the statistical theory, we saw in the previous module , and poll data. To help us understand the connection we are going to construct a scenario that we can work though, and that is similar to the one that pollsters face. We will use an urn instead of voters, and because pollsters are competing with other pollsters for media attention, we will imitate that by having our competition with a $25 prize. The challenge is to guess the spread between the proportion of blue and red balls in this urn. Before making a prediction, you can take a sample, with replacement from the urn. To mimic the fact that running polls is expensive it will cost you %0.10 per bead you sample. So if you sample size is 250 and you win, you’ll break even, as you’ll have to pay $25 to collect your $25. The entry into the competition can be interval. If the interval you submit contains the true proportion, you get half what you paid and pass to the second phase of the competition. In the second phase, the entry with the smallest interval is selected as the winner.

The dslabs package include a function that shows a random draw from a urn we just described, here is the code:

*ds\_theme\_set()*

*take\_poll(25)*

1. Sample of 25 beads from the urn



What we just described is a simple sampling model for opinion polls, the beads inside the urn represent the individuals that will vote on the election day. Red for Republicans and blue for Democrats. We want to predict the proportion of blue beads in the urn. Let`s call this quantity p, which tell us, in turn, that the proportion of reds beads are 1-p and the spread ,p\*(1-p), which simplifies to 2p-1.

In statistics the beads in the urn are called the **population**, p is called a **parameter**. The 25 beads that we see above is called **a sample**. The task of statistical inference is to predict the parameter, p, using the observed data in the sample. The sample of 25 observation is informative to answer this question, for example, given that we see 13 red and 12 blue, it is unlikely that p is bigger than 0.9 or smaller than 0.1. But are we ready to predict with certainty that there are more red beads than blue?

What we want to do is construct an estimate of p using only the observe information. An estimate can be thought of as a summary of the observed data that we think is informative about the parameter of interest. It seems intuitive to think that the proportion of blue beads in the sample, which in this case is 0.48, must be at least related to the actual proportion p. Note that the sample proportion is a random variable. If we run the command:

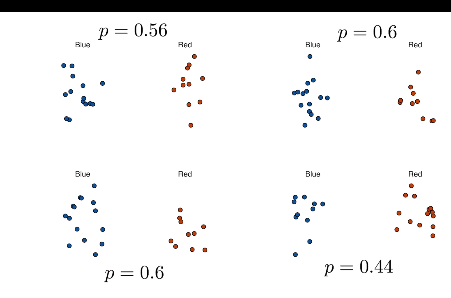
*take\_poll(25)*

*take\_poll(25)*

*take\_poll(25)*

*take\_poll(25)*

1. Four different sample of the same poll



The sample proportion ranges from 0.44 to 0.6. By describing the distribution of this random variable, we’ll be able to gain insights into how good this estimate is and how we can make it better.

* + 1. The Sample Average

Taking an opinion poll is modeled as taking a random sample from an urn. We are proposing the use of the proportion of blue beads in our sample as an estimate of the parameter p. Once we have this estimate we can easily report an estimate of the spread, 2p-1.

For simplicity, we will illustrate the concept of statistical inference for estimating p. We will use our knowledge of probability to defend our use of the sample proportion, and quantify how close we think it is form the population proportion p.

First we define a random variable X. X is going to be 1 if we pick a blue bead at random and 0 if it’s red. This implies that we are assuming that the list population, the beads in the urn, are a list of 0s and 1s. If we sample N beads, then the average of the draws X\_1 through X\_n is equivalent to the proportion of blue beads in our sample:

Were is the average. The theory we just learned about the sum of draws becomes useful because we know the distribution of the sum

For simplicity, let’s assume that the draws are independent. After we see each sample bead, we return it to the urn. In this case we know that the expected value of the sum of draws is :

We know that the average of the 0s and 1s in the urn must be the proportion p, the value we want to estimate. Here, we encounter an important difference with what we did in the probability module. We don’t know what is in the urn.

Just like we use variables to define unknowns in systems of equation, in statistical inference, we define parameters to define unknown parts of our models.

* + 1. Polling versus Forecasting

Before we continue, let’s clarify related problems to the practical problem of forecasting the election. If a poll is conducted 4 months before the election, it is estimating the p for that moment, nor for election day. The p for election night might be different since people’s opinion fluctuate through time. The polls provided the night before the election tend to be the most accurate since opinions don’t change that much in a couple of days. However, forecasters try to build tools that model how opinions vary across time and try to predict the election day result, taking into consideration the fact that opinions fluctuate.

* + 1. Properties of Our Estimate

To understand how good our estimate is, we’ll describe the statistical properties of the random variable we just define, the sample proportion. Using what we learned in the previous module :

We also can use what we learned to figure out the standard error. We know that:

This result shows the power of the polls, the expected value of the sample proportion is the parameter of interest, p and we can make the standard error as small as we want by increasing the sample size, N. The law of large numbers tells us that, with a large enough poll, our estimate converges to p.

If we take a large enough poll to make our standard error, say, about 0.01, we’ll be quite certain about who will win. But how large does a poll have to be for the standard error to be this small? One problem is that we do not know p, so we can’t compute the standard error

1. The Central Limit Theorem in Practice
   1. Central Limit Theorem in Practice
      1. The Central Limit Theorem in Practice

The CLT tells us that the distribution function for a sum of draws is approximately normal. We also learned that when dividing a normally distributed random variable by a nonrandom constant the resulting random variable is also normally distributed.

This implies that the distribution of is approximately normal. So in summary, we have that has an approximately normal distribution, with an expected value of p and a standard error of (as seen in the previous chapter).

Suppose we want to know what is the probability that we are within one percentage point from p i.e. that we made a very good estimate? We are basically asking:

We can use what we’ve learned to see that this is the same as asking:

By subtracting by the expected value and dividing by the standard error on both sides of the inequation we get a normalized random variable, called Z and the inequation becomes:

We can’t compute this probability yet, we don’t know p. So we can’t actually compute the standard error of using just the data. But the CLT still works if we use an estimate of the standard error that, instead of p, uses in its place. We call this a **plug-in estimate**. Our estimate of the standard error is therefore:

The is used to denote estimates. This is an estimate of the standard error, not the actual value. Note that this estimate can be constructed using the observed data. Now let`s continue our calculations, but now instead of dividing by the standard error we`re going to divide by this estimate of the standard error. Let`s take the example in which we had 12 blue beads and 13 red beads. In that case was 0.48, so to compute the standard error, we simply write this code:

*X\_hat<-0.48*

*se<-sqrt(X\_hat\*(1-X\_hat)/25)*

*se*

*[1] 0.09991997*

Now, we can compute the probability of being as close to p as we wanted. We wanted to be 1 percentage point away. The answer is simply pnorm of 0.01:

*pnorm(0.01/se)-pnorm(-0.01/se)*

*[1] 0.07971926*

* + 1. Margin of Error

So, a poll of only 25 people is not very useful, at least for a close election. Earlier we mentioned the margin of error. Now we can define it because it is simply 2 times the standard error, which we can now estimate. In our case it was:

*2\*se*

*[1] 0.1998399*

We multiply by 2 because there is a 95% chance that will be within 2 standard errors. That’s the margin of error in our case, to p.

* + 1. A Monte Carlo Simulation for the CLT

Suppose we want a Monte Carlo simulation to corroborate that the tools that we’ve been using work. To create the simulation, we would need to write code like this:

*B<-10000*

*> N<-1000*

*> x\_hat<-replicate(B,{*

*+ X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*+ mean(X)*

*+ })*

The problem is, of course, that we don’t know p. However, we could construct an urn and run an analog simulation. It would take a long time because you would be picking beads and counting them, but you could take 10000 samples, count the beads each time, and keep track of the proportion that you see. We can use the function take poll with n of 1000 instead of actually drawing from an urn, but it would still take time because you would have to count beads and enter the results into R. So, one thing we can do to confirm theoretical results is to pick a value of p or several value of p and then run simulations using those. As an example, let’s set p=0.45. We can simulate one poll of 1000 beads or people using this simple code:

*p<-0.45*

*> N<-1000*

*> X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*> X\_hat<-mean(X)*

Now we can take that into a Monte Carlo simulation. Do it 10000 times, each time return the proportion of blue bleads that we get in our sample:

*B<-10000*

*x\_hat<-replicate(B,{*

*+ X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*+ mean(X)*

*+ })*

If we take the means and standard deviation of X\_hat we just created we confirm the results of the theory:

*mean(x\_hat)*

*[1] 0.4499778*

*> sd(x\_hat)*

*[1] 0.015778*

A histogram and a qq plot of this X-hat data confirms that the normal approximation is accurate as well:

*library(gridExtra)*

*p1<-data.frame(X\_hat=x\_hat)%>% ggplot(aes(X\_hat))+geom\_histogram(binwidth = 0.005,color="black")*

*>p2<-data.frame(X\_hat=x\_hat)%>%ggplot(aes(sample=X\_hat))*

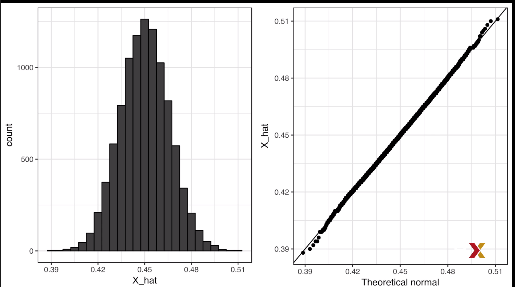
*+stat\_qq(dparams=list(mean=mean(X\_hat),sd=sd(X\_hat)))+geom\_abline()*

*+ylab("X\_hat")*

*+xlab("Theoretical normal")*

*> grid.arrange(p1,p2,nrow=1)*

1. Histogram and qq plot of X\_hat distribution



* + 1. The Spread

The competition is to predict the spread, not the proportion p. However, because we assume there are only two parties, we know that the spread is just:

So everything we have done can easily be adapted to estimate to p minus 1. Once we have our estimate. X\_bar, and our estimate of our standard error of X\_bar, we estimate the spread by:

And, since we’re multiplying a random variable by 2, we know that the standard error goes up by 2. So the standard error of this new random variable is:

In our first example, with just 25 beads, our estimate of p was 0.48 with a margin of error of 0.2. This means that our estimate of the spread is 0.04 (4 percentage points), with a margin of error of 40%, 0.4 (2 times the margin of error of

* + 1. Bias: Why Not Run a Very Large Poll?

Note that for realistic values of p, say between 0.35 and 0.65 for the popular vote, if we run a very large poll with say 100000 people, theory would tell us that we would predict the lection almost perfectly since the largest possible margin of error is about 0.3%, here are the calculations that were used to determine that:

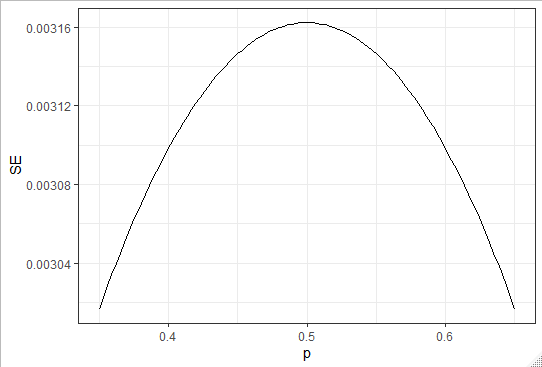
N<-100000

p<-seq(0.35,0.65,length=100)

SE<-sapply(p,function(x)2\*sqrt(x\*(1-x)/N))

data.frame(p=p,SE=SE)%>%ggplot(aes(p,SE))+geom\_line()

1. Largest margin of error



So why are there no pollsters that are conducting polls this large? One reason is that running polls with a sample size of 100000 is very expensive. But perhaps a more important reason is that theory has its limitations. Polling is much more complicated that picking beads from an urn. For example, while the beads are either red or blue, and you can see it with your eyes, people, when you ask them might lie to you. Also, because you’re conducting these polls usually by phone you might miss people that don’t have phones.

The most different way an actual poll is form our urn model is that we actually don’t know for sure who is in our population and who is not .So even if our margin of error is very small, it may not be exactly right that our expected value is p. We call this bias. Historically, we observe that polls are, indeed, biased, although not by that much. The typical bias appears to be between 1% and 2%. This makes election forecasting a bit more interesting.

1. Confidence Intervals and p-Values
   * 1. Confidence Intervals

Confidence intervals are a very useful concept that are widely used by data scientists. A version of these that are very commonly seen com from the ggplot geometry **geom\_smooth()** here’s an example using weather data:

*data("nhtemp")*

*data.frame(year=as.numeric(time(nhtemp)),temperature=as.numeric(nhtemp))%>%*

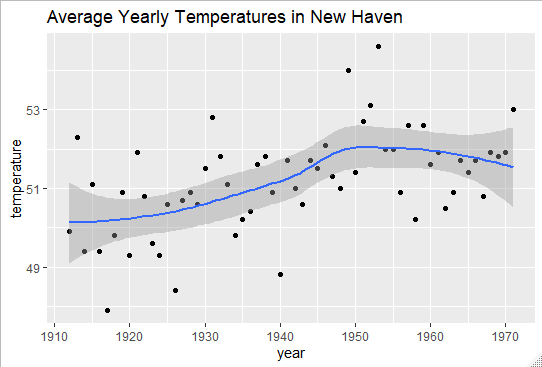
*ggplot(aes(year,temperature))*

*+geom\_point()*

*+geom\_smooth()*

*+ggtitle("Average Yearly Temperatures in New Haven")*

1. Average Yearly temperatures in New Haven



Note the shaded area around the curve. This shaded area is created using the concept of confidence intervals. In our competition, we were asked to give an interval. If the interval you submit includes the actual proportion p, you get half the money you spent on your poll and pass to the next stage of the competition. One way to pass to the second round of the competition to report a very large interval, for example [0,1] this is guaranteed to include p.

However, with an interval this big, we have no chance of winning the competition. Similarly if you are an election forecaster and predict the spread will -100 and 100 you’ll be ridiculed for stating the obvious. Even a smaller interval such as saying that the spread will be between -10% and 10% will not be considered serious. On the other hand, the smaller the interval we report, the smaller our chance of passing to the second round.

Similarly, a bold pollster that reports very small intervals and misses the mark most of the time, will not be considered a good pollster.

We can use the statistical theory we have learned to compute, for any given interval, the probability that it includes p. Similarly if we are asked to create an interval with, say, a 95% chance of including p, we can do that as well. These are called 95% confidence intervals.

Note that, when pollsters report an estimate and a margin of error, they are, in a way, reporting 95% confidence interval. Let’s show how this works mathematically:

We want to know:

First note that the start and end of this interval are random variables. Every time that we take a sample, they change. To illustrate this, we are going to run a Monte Carlo simulation, we’re going to do it just 2 first:

*p<-0.45*

*> N<-1000*

*> X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*> X\_hat<-mean(X)*

*> SE\_hat<-sqrt(X\_hat\*(1-X\_hat)/N)*

*> c(X\_hat-2\*SE\_hat,X\_hat+2\*SE\_hat)*

*[1] 0.4294736 0.4925264*

Note that if we run that same code again we get a different interval:

*p<-0.45*

*> N<-1000*

*> X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*> X\_hat<-mean(X)*

*> SE\_hat<-sqrt(X\_hat\*(1-X\_hat)/N)*

*> c(X\_hat-2\*SE\_hat,X\_hat+2\*SE\_hat)*

*[1] 0.4474051 0.5105949*

To compute the probability of the equation above is equivalent to this:

The term in the middle is an approximately normal random variable with expected value 0 and standard error 1, which we have been denoting with Z, so the equation becomes:

So what is the probability of a standard normal variable being between -2 and 2? It’s about 95%. Note that if we want to have a larger probability say 99%, we need to multiply z by whatever satisfies the following equation:

Note that by using the quantity that we get by typing this code:

*z<-qnorm(0.995)*

*> z*

*[1] 2.575829*

Will do it, because the pnorm of what we get when we type:

*pnorm(qnorm(0.995))*

*[1] 0.995*

And by symmetry:

*pnorm(qnorm(0.995))*

*[1] 0.995*

*> pnorm(1-qnorm(0.995))*

*[1] 0.05753257*

*> 1-pnorm(qnorm(0.995))*

*[1] 0.005*

So if we compute:

*pnorm(z)-pnorm(-z)*

*[1] 0.99*

We can use this approach for any percentile q. We use:

Also note that to get exactly 0.95, we use a slightly smaller number than 2. How do we know? We type:

*qnorm(0.975)*

*[1] 1.959964*

* + 1. A Monte Carlo Simulation for Confidence Intervals

We can run a Monte Carlo simulation to confirm that, in fact, a 95% confidence interval includes p 95% of the time. We write the simulation like this:

*B<-10000*

*> inside<-replicate(B,{*

*+ X<-sample(c(0,1),size=N,replace=TRUE,prob=c(1-p,p))*

*+ X\_hat<-mean(X)*

*+ SE\_hat<-sqrt(X\_hat\*(1-X\_hat)/N)*

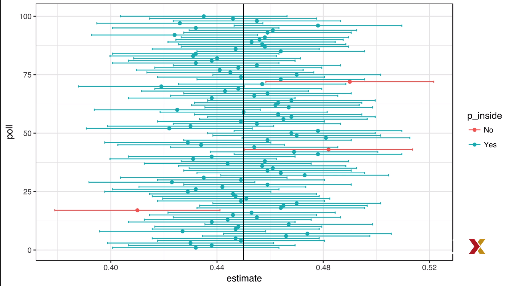
*+ between(p,X\_hat-2\*SE\_hat,X\_hat+2\*SE\_hat)*

*+ })*

*> mean(inside)*

*[1] 0.9538*

1. First Few interval that were generated in our Monte Carlo simulation



In this case we know what p is and it’s represented by the black vertical line in the middle of the plot. Notice that you can the confidence intervals varying. Each time, they fall in slightly different places. This is because they’re random variables. We also know that most of the times, p is included inside the confidence interval. We also notice, that every once in a while we miss p, these confidence intervals are shown in red. We should only see about 5% of the intervals in red because they’re 95% confidence intervals.

* + 1. The Correct Language

When using the theory, we just described, it is important to remember that it is the intervals that are at random, not p. We showed a plot where we could see the random interval that. So the 95% relates to the probability that the random interval falls on top of p. Saying that p has a 95% chance of being between this and that is technically an incorrect statement again, because p is not random.

* + 1. Power

Pollsters do not become successful for providing confidence intervals, bur rather for predicting who will win. When we took a sample of size 25, the confidence interval for the spread was:

*N<-25*

*> X\_hat<-0.48*

*> (2\*X\_hat-1)+c(-2,2)\*2\*sqrt(X\_hat\*(1-X\_hat)/sqrt(N))*

*[1] -0.9337114 0.8537114*

This includes 0. If we were pollsters and we were forced to make a declaration about the election, we would have no choice but to say it’s a tossup. A problem with our poll results is that given the sample size and the value of p, we would have to sacrifice on the probability of an incorrect call to create an interval that does not include 0, an interval that makes a call of who’s going to win. The fact that our interval includes 0, does not mean that this election is close. It only means that we have a small sample size. In statistics, this is called **a lack of power**. In the context of polls, power can be thought of as the probability of detecting a spread different from 0. By increasing our sample size, we lower our standard error and therefore have a much better chance of detecting the direction of the spread.

* + 1. p-Values

Let’s consider the blue and red bead example again. Suppose that rather than wanting to estimate the spread or the proportion of blue, I’m interested only in the question, are there more blue beads than red beads? Or in another way:

Suppose we take a random sample of, say, 100 beads and we observe 52 blue beads. This give us a spread of 4%. Indicating that it seems there is more blue beads than red beads, because 4% is larger than 0. 52%>48%. However as data scientist we need to be skeptical. We know there is chance involved is this process, and we can get a 52 even when the actual spread is 0. The null hypothesis is the skeptic’s hypothesis. In this case, it would be the spread is 0. We have observed a random variable:

**The p-value** is the answer to the question, how likely is it to see a value this large when the null hypothesis is true? Or put mathematically:

Which is equivalent to asking what is the chance that the spread is 4 or more? The null hypothesis is that the spread is 0 or that p is half. Under the null hypothesis, we know that this quantity:

Is a standard normal. We’ve taken a random variable and divided it by it’s standard error after subtracting its expected value. So we can compute the probability, which is a **p-value**, using this equation, which reduce this equation, where w is a standard normal:

And now we can use this code to compute this:

*N<-100*

*> z<-sqrt(N)\*0.02/0.5*

*> 1-(pnorm(z)-pnorm(-z))*

*[1] 0.6891565*

We compute that the probability, which is equal to 69% in this case. This is the p-value. In our case, there’s a actually a large chance of seeing 52 blue beads or more under the null hypothesis that there is the same amount of blue beads and red beads. So, the 52 blue beads are not very strong evidence, if we want to make the case that there are more blue beads than red beads.

Note that there’s a close connection between p-values and confidence intervals. If a 95% confidence interval of the spread does not include 0, we can do a little bit of math to see that this implies that the p-value must be smaller than 1-0.95=0.05.

In general, we prefer reporting confidence intervals over p-value, since it give us and idea of the size of the estimate.

1. Statistical Models
   * 1. Poll Aggregators

In the 2012 presidential, Barack Obama won the electoral college and the popular vote by a margin of 3.9%. If we go back to a week before knew the outcome. Nate Silver was giving Obama a 90% chance of winning. Yet, none of the individual polls were nearly that sure. In fact, political commentator Joe Scarborough said during his show, “Anybody that thinks that this race is anything but a tossup right now is such an ideologue—they’re jokes” to which Nate Silver responded, “if you think it’s a tossup, let’s bet. If Obama wins, you donate $1000 to the American Red Cross. If Romney wins, I do Deal?” How is Mr. Silver so confident? We’ll illustrate what Nate Silver saw that Joe Scarborough and other pundits did not.

We’re going to use a Monte Carlo simulation. We’re going to generate results for 12 polls taken the week before the election. We’re going to mimic the sample sizes from actual polls. We’re going to construct and report 95% confidence intervals for each of those polls, here’s the code we’re going to use:

*d<-0.039*

*> Ns<-c(1298,533,1342,897,774,254,812,324,1291,1056,2172,516)*

*> p<-(d+1)/2*

*> confidence\_intervals<-sapply(Ns,function(N){*

*+ X<-sample(c(0,1),size = N,replace=TRUE,prob=c(1-p,p))*

*+ X\_hat<-mean(X)*

*+ SE\_hat<-sqrt(X\_hat\*(1-X\_hat)/N)*

*+ 2\*c(X\_hat,X\_hat-2\*SE\_hat,X\_hat+2\*SE\_hat)-1*

*+ })*

Once we do this, we generate a data frame that has all the results:

*names(polls)<-c("poll","estimate","low","high","sample\_size")*

*> polls*

*poll estimate low high*

*1 1 1.993837 1.938325 2.049348*

*2 2 1.956848 1.870299 2.043397*

*3 3 1.995529 1.940934 2.050124*

*4 4 2.074693 2.008102 2.141285*

*5 5 2.031008 1.959154 2.102862*

*6 6 2.047244 1.921893 2.172595*

*7 7 1.990148 1.919965 2.060331*

*8 8 1.969136 1.858078 2.080194*

*9 9 2.048799 1.993203 2.104396*

*10 10 2.005682 1.944137 2.067227*

*11 11 2.042357 1.999482 2.085233*

*12 12 2.058140 1.970243 2.146036*

*sample\_size*

*1 1298*

*2 533*

*3 1342*

*4 897*

*5 774*

*6 254*

*7 812*

*8 324*

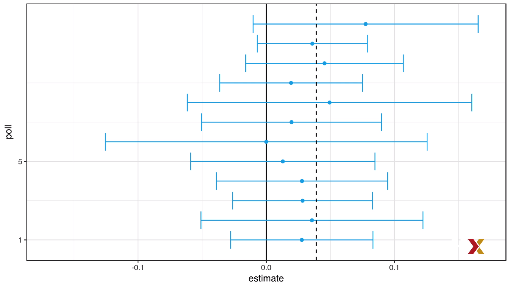
*9 1291*

*10 1056*

*11 2172*

*12 516*

1. Visualization of what the intervals of these pollsters would have reported for the difference between Obama and Romne



All 12 polls report confidence intervals that include the election nigh result, which is shown with the dashed line, this is because they are 95% confidence interval, however all 12 polls intervals includes 0 (which is shown in black line). Therefore, individually if we asked for a prediction from the pollsters, from each individual pollster they would have to agree with Scarborough, it’s a tossup. Poll aggregator, as Nate Silver, realize that by combining the results of different polls we could greatly improve precision. By doing this, effectively we’re conducting a poll with a huge sample size. As a result, we can report a smaller confidence interval, and therefore a more precise prediction. Although as aggregators we do not have access to the raw poll data, we can use mathematics to reconstruct what we would have obtained had we made one large poll with, in this case, 11269 people. Basically, we construct an estimate of the spread, let’s called it d with a weighted average int the following way:

*d\_hat<-polls%>%summarize(avg=sum(estimate\*sample\_size)/sum(sample\_size))%>%.$avg*

We basically multiply each individual spread by the sample size. That’s going to give us a total spread, and then we’re going to divide by the total number of participants in our aggregated poll. This give us d\_hat, which is an estimate of d. Once we have d\_hat we can construct an estimate for the proportion voting for Obama, which we can then use to estimate the standard error:

*d\_hat<-polls%>%*

*+ summarize(avg=sum(estimate\*sample\_size)/sum(sample\_size))%>%*

*+ .$avg*

*p\_hat<-(d\_hat+1)/2*

*> moe<-2\*1.96\*sqrt(p\_hat\*(1-p\_hat)/sum(polls$sample\_size))*

*> moe*

*[1] 0.01844203*

We can see that the margin of error of the aggregated poll is 0.018. Thus, using the weighted average, we can predict that the spread will be 4.8% plus or minus 1.8$ which not only includes the actual result but is quite far from including 0:

round(d\_hat\*100,1)

[1] 4.8

> round(moe\*100, 1)

[1] 1.8

Once we combine the 12 polls, we become quite certain that Obama will win the popular vote. In this figure, you can see, in red, the interval that was created using the combine polls.

* + 1. Pollsters and Multilevel Models

Now we’re going to get ready to explain how pollsters fit multilevel models to public poll data and use this to forecast election results. In the 2008 and 2012 US presidential elections, Nate Silver used this approach to make an almost perfect prediction and silenced the pundits. Since the 2008 election other organizations have started their own election forecasting groups that, like Nate Silver, aggregate polling data and use statistical models to make predictions. In 2016, forecasters greatly underestimated Trump’s chances of winning the election. For example, The Princeton Election Consortium gave Trump less than 1% of winning, while the Huffington Post gave him 2%. In contrast, FiveThirtyEight had Trump’s chances of winning at 29%. Although they didn’t correctly predict him to have a higher probability, note that 29% is a higher probability than of the probability of tossing 2 coins and getting 2 heads. It’s also much bigger than what the other pollster had predicted. By understanding statistical models and how these forecasters use them, we will start to understand how this happened. Although nor nearly as interesting as predicting the electoral college, the actual outcome election, for illustrative purposes we will start by looking at the predictions for the popular vote.

FirstThirtyEight predicted a 3.6% advantage for Clinton. Their interval, their prediction interval, included the actual result of 2.1%. They were much more confident about Clinton winning this, the popular vote, giving her a 81.4% chance of winning

* + 1. Poll Data and Pollster Bias

We are going to use the data organized by FiveThirtyEight for the 2016 president election. The data is included as of the dslabs package. You can get the data like this:

*data("polls\_us\_election\_2016")*

*names(polls\_us\_election\_2016)*

*[1] "state" "startdate"*

*[3] "enddate" "pollster"*

*[5] "grade" "samplesize"*

*[7] "population" "rawpoll\_clinton"*

*[9] "rawpoll\_trump" "rawpoll\_johnson"*

*[11] "rawpoll\_mcmullin" "adjpoll\_clinton"*

*[13] "adjpoll\_trump" "adjpoll\_johnson"*

*[15] "adjpoll\_mcmullin"*

The table include results for national polls, as well as state polls, taken in the year before the election. For this first illustrative example, we will filter the data to include national polls that happened during the week before the election. We also remove polls that FiveThirtyEight has determined not to be reliable and they have graded them with a B or less. Some polls have not been graded, we’re going to leave these in. here’s the code:

*polls<-polls\_us\_election\_2016%>%*

*filter(state=="U.S."& enddate>="2016-10-31"&(grade %in% c("A+","A","A-","B+")|is.na(grade)))*

We also add a spread estimate (the thing we want to estimate):

*polls<-polls%>%mutate(spread=rawpoll\_clinton/100-rawpoll\_trump/100)*

For illustrative purposes, we will assume that there are only 2 parties, and call p the proportion voting for Clinton, and 1-p the proportion for voting for Trump. We’re interested in the spread, which we’ve shown is 2p-1. Let’s call this spread d. Note that we have several estimates of this spread coming from the different polls. The theory we learned tell us that these estimates are a random variable with probability distribution that is approximately normal. The expected value is the election night spread, d. and the standard error is:

Assuming the urn model we described earlier are useful models, we can use this information to construct a confidence interval based on the aggregated data. The estimate spread is now computed like this because now the sample size is the sum of all the sample sizes:

*d\_hat<-polls%>%summarize(d\_hat-sum(spread\*samplesize)/sum(samplesize))%>%.$d\_hat*

and if we use this, we get a standard error:

*moe<-1.96\*2\*sqrt(p\_hat\*(1-p\_hat)/sum(polls$samplesize))*

*> moe*

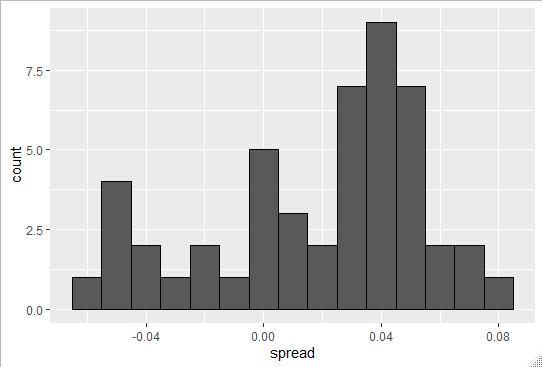
*[1] 0.006616158*

So, if we were going to use this data, we would report a spread of 1.43% with a margin of error of 0.66%. On election night, we find out that the actual percentage, is 2.1%, which is outside of the 95% confidence interval. So, what happened?

A histogram of the reported spreads shows another problem:

polls%>% ggplot(aes(spread))+geom\_histogram(color="black",binwidth = 0.01)

1. Histogram of reported spreads



The data does not appear to be normally distributed, and the standard error appears to be larger than 0.0066. The theory is not quite working here.

To see why, notice that various pollsters are involved and some are taking several polls a week. Here’s a table showing you how many polls each pollster took that last week:

*polls%>%group\_by(pollster)%>%summarize(n())*

*# A tibble: 15 x 2*

*pollster `n()`*

*<fct> <int>*

*1 ABC News/Washington Post 7*

*2 Angus Reid Global 1*

*3 CBS News/New York Times 2*

*4 Fox News/Anderson Robbins Researc~ 2*

*5 IBD/TIPP 8*

*6 Insights West 1*

*7 Ipsos 6*

*8 Marist College 1*

*9 Monmouth University 1*

*10 Morning Consult 1*

*11 NBC News/Wall Street Journal 1*

*12 RKM Research and Communications, ~ 1*

*13 Selzer & Company 1*

*14 The Times-Picayune/Lucid 8*

*15 USC Dornsife/LA Times 8*

Let’s visualize the data for pollsters that are regularly polling. We write this piece of code that first filters for only pollsters that polled more than 6 times. And then we simply plot the spreads estimated by each pollster:

*polls%>%group\_by(pollster)%>%*

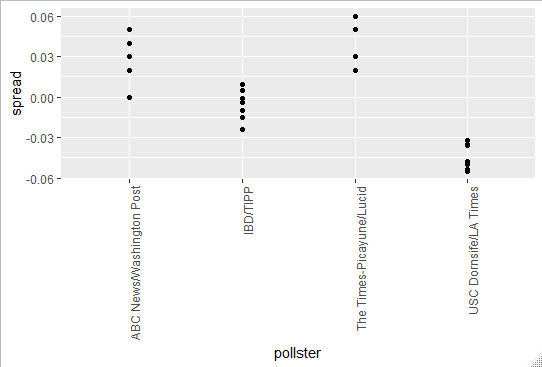
*filter(n()>6)%>%*

*ggplot(aes(pollster,spread))*

*+geom\_point()*

*+theme(axis.text.x=element\_text(angle=90,hjust=1))*

1. Plot of spread by pollster that have done more than 6 polls in the week before the election



This plot reveals first that the standard error, predicted by theory for each poll, give us values between 0.018 and 0.033. However there appears to be differences across the polls. This is not explained by the theory. Note for example, how the USC Dornsife/LA Times pollster is predicting a 4$ win for Trump while Ipsos is predicting a win larger than 5% for Clinton. The theory of learned says nothing about different pollsters producing polls with different expected values. All the polls should have the same expected values. The actual spread, the spread we will see on election night. We call this **pollster bias**. Rather than use the theory, we’re instead going to develop a data-driven model to produce a better estimate and a better confidence interval.

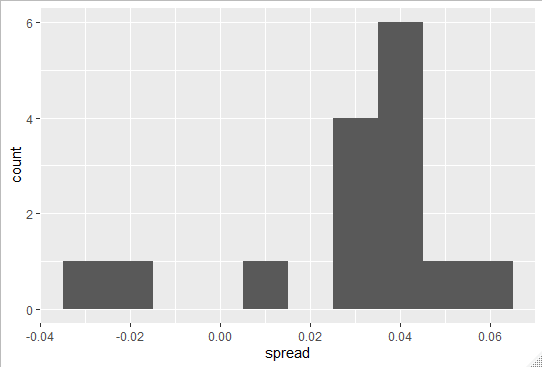
* + 1. Data-Driven Models

For each pollsters, let’s collect their last-reported result before the election using the simple piece of code:

*one\_poll\_per\_pollster<-polls%>%group\_by(pollster)%>%filter(enddate==max(enddate))%>%ungroup()*

*one\_poll\_per\_pollster%>%ggplot(aes(spread))+geom\_histogram(binwidth = 0.01)*

1. Histogram of the last reported result before the election for 15 pollsters



In the last chapter, we saw that using the urn model theory to combine these results might not be the appropriate due to the pollster effect. Instead we will model this spread data by the data directly. The new model can also be thought of as an urn model, although the connection to the urn idea is not as direct. Rather than having beads with 0s and 1s inside, the urn contains poll results from all possible pollsters. We assume that the expected value of our urn is the actual spread, which we have been calling d, which is equal to:

Now, because rather than 0s and 1s our urn contains continuous number between -1 and 1, the standard deviation of the urn is no longer sqrt(p\*(1-p)). Rather than just the sampling variability we get from taking different samples of 0s and 1s the standard error for our average now includes the pollster to pollster variability.

This standard deviation is now an unknown parameter σ. Now in summary, we have 2 unkown parameters the expected value d, and the standard deviation σ. Our task is to estimate d and provide inference for it. Because we model the observed values, let’s call them X1 though XN as a random sample from the urn, the central limit theorem still works for the average of these values because it’s an average of independent random variables. For a large enough sample size N, the probability distribution of the sample average, which we’ll call X\_Bar, is approximately normal with expected value d and standard deviation σ divided by the square root of N.

If we are willing to consider N equals to 15 large enough, we can use this to construct a confidence interval. A problem, though, is that we don’t know σ. But the theory tell us that we can estimate the urn model σ, with the sample standard deviation, which is defined by:

We can compute this calculation with the function **sd()** in R, like this:

*sd(one\_poll\_per\_pollster$spread)*

*[1] 0.02419369*

We are now ready to form a confidence interval based on our new data-driven model. We simply use the central limit theorem and create a confidence interval, like this:

*results<-one\_poll\_per\_pollster%>%*

*summarize(avg=mean(spread),se=sd(spread)/sqrt(length(spread)))%>%*

*mutate(start=avg-1.96\*se,end=avg+1.96\*se)*

*> round(results\*100,1)*

*avg se start end*

*1 2.9 0.6 1.7 4.1*

We get an average, a standard error, and then a start of 1.7% and an end of 4.1%. That’s our 95% confidence interval, using our data-driven model. Our new confidence interval is wider, and it now incorporates the pollster variability. It does include the election night result of 2.1% and, it’s small enough not to include 0. Now are we ready to declare a probability of Clinton winning as the pollsters do? Not yet, in our model, d is a fixed parameter, so we can’t talk about probabilities.

1. Bayesian Statistics
   * 1. Bayesian Statistics

What does it mean when an election forecaster tells us that a given candidate has a 90% chance of winning? In the context of the urn model this would be equivalent to stating that the probability that the proportion p of people voting for this candidate being bigger than 50% (0.5) is 90%. But as discussed, in the urn model, p is a fixed parameter, and it does not make sense to talk about probability of p being this or that. With Bayesian statistic, we assume it is in fact random, and then it make sense to talk about probability. Forecasters also use models to describe variability at different levels. For example: sampling variability, pollster to pollster variability. One of the most successful approaches used to describe these different levels of variability are called **hierarchical models**.

* + 1. Bayes' Theorem

Let’s start by an example, with cystic-fibrosis test. Suppose a test for cystic fibrosis has an accuracy of 99%. We will use the following notation to represent this:

With D representing the patient having the disease and + or – the result of the test. Suppose we select a random person and thest test positive, what is the probability that they have the disease? We write this as:

The cystic fibrosis rate is 1 in 3900, which implies that:

To answer that question we’ll use the **Bayes’s theorem** which in tells that :

Using the multiplication rule it becomes:

In the case of our cystic fibrosis example (as often) we know the probability Pr(A|B) and not the probability Pr(B|A). If we apply the theorem to our example we get:

We know Pr(+|D=1), Pr(D=0). So know using Bayes’s formula, we write out the equation:

This says that despite the test having a 99% accuracy, the probability of having the disease given positive test is only 2%. This may appear counterintuitive to some. But we’re going to see how it makes sense. The reason this is the case is because we must factor in the very rare possibility that a person chosen at random has the disease. This is the Bayesian way of thinking. To illustrate this, we can use a Monte Carlo simulation. The following simulation is meant to help visualize Bayes’ theorem. We start by randomly selecting 100000 people from a population in which the disease in question has a 1 in 3900 prevalence so we get:

*prev<-0.00025*

*> N<-100000*

*> outcome<-sample(c("Disease","Healthy"),N,replace=TRUE,prob=c(prev,1-prev))*

Note that because the prevalence is so low, once we take the sample the number of people with the disease is low:

*N\_D<-sum(outcome=="Disease")*

*N\_D*

*[1] 21*

And of course, there’s a lot of healthy people:

*N\_H<-sum(outcome=="Healthy")*

*N\_H*

*[1] 99979*

This makes the probability that we see some false positives quite high. There are so many people without the disease that are getting test that, although it’s rare, we are going to get a few people getting a positive test despite them being healthy. Here’s the code to show it:

*accuracy<-0.99*

*test<-vector("character",N)*

*test[outcome=="Disease"]<-sample(c("+","-"),N\_D,replace=TRUE,prob=c(accuracy,1-accuracy))*

*test[outcome=="Healthy"]<-sample(c("-","+"),N\_H,replace=TRUE,prob=c(accuracy,1-accuracy))*

Each person has a 99% chance of getting the test giving them the right answer. We can make a table that show the number of people in each one of these four combinations like this:

*table(outcome,test)*

*test*

*outcome - +*

*Disease 0 21*

*Healthy 98971 1008*

You can see that there a lot of people that are healthy and got a positive outcome. That’s because there are so many more healthy people. From this table, we can also see that the proportion of positive tests that have the disease is 21 and this out of a total of 21+1008=1029, if you divide 21 by 1029 you get 0.02 or 2% which is exactly what Bayes’ theorem told us it should be

* + 1. Bayes in Practice

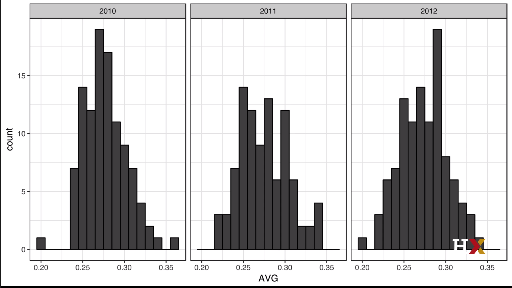
To demonstrate the usefulness of hierarchical models, Bayesian models, in practice, we’re going to show you a baseball example. In sports, we use Bayesian thinking all the time, even if we don’t realize it. Let’s go to the example. Jose Iglesias is a professional baseball player. In April 2013, when he was performing rather well. He had been to bat 20 times and he had 9 hits, which is an average of 0.450. Which means Jose had been successful 45% of the times he had batted, which is rather high historically speaking. Note, for example, that no one has finished a season with an average of 0.400 or more since Ted Williams did it in 1941. To illustrate the way hierarchical models are powerful, we will try to predict Jose’s batting average at the end of the season.

In a typical season, players have about 500 at bats. With the techniques we have learned up to now, referred to as **frequentist statistics**, the best we can do is provide a confidence interval. We can think of outcomes for hitting as a binomial with a success rate of p. So, if the success rate is indeed 0.45, the standard error of just 20 at bats can be computed like this:

We can use this to construct a 95% interval, which will be from 0.228 to 0.672. The prediction has 2 problems. First, it’s very large, so it’s not very useful. Second, it’s centered at 0.450, which implies that our best guess is that this relatively unknown player will break Ted Williams’ longstanding record. This seems wrong if you follow baseball, and this because you’re implicitly using the hierarchical model that factors in information from years of following baseball.

Here we show how we can quantify this intuition. First, let’s explore the distribution of batting averages for all players with more than 500 at bats during the season 2010,2011 and 2012:

1. Distribution of batting averages for all players with more than 500 at bats during the season 2010,2011 and 2012



We note that the average player had an average of 0.275 and the standard deviation of the population of all these players was 0.027. So, we can see already that 0.450 would be quite an anomaly.

* + 1. The Hierarchical Model

The hierarchical model provides a mathematical description of how we can come to see the observation of 0.450. First, we pick a player at random with an intrinsic ability summarized by, for example, p, the proportion of times they will be successful. Then we see 20 random outcomes with success probability p. We use a model to represent 2 levels of variability in our data. First, each player is assigned a natural ability to hit at birth. Based on the plots, we assume that p has a normal distribution. If we just pick a player at random, the random variable p will have a normal distribution. We also know that the expected value is 0.27 and the standard error of 0.027. Now, the second level of variability has to do with luck. Regardless of how good or bad a player is, sometimes you have bad luck and sometimes you don’t, at each at bat, this player has a probability of success p. If we add up these successes and failures as 0s and 1s, then the CLT tell us that the observed average, let’s call it Y, has a normal distribution with expected value p and standard error sqrt(p\*(1-p)/N), where N is the number at bats. Statistical textbooks will write the model like this:

Where p is a random variable, with expected value µ and standard error , describes the randomness in picking a player. Now we describe the distribution of the observed batting average Y, given that this player has a talent, p, is also normally distributed with expected value p and a standard error σ:

That describe the randomness in the performance of this particular player. In our example, mu=0.270, tau=0.027 and

Because there are 2 levels, we call these hierarchical models. The first one is the player to player variability. The second is the variability due to luck when batting. In Bayesian framework, the first level is called **prior distribution** and the second the **sampling distribution**.

Now, let’s use this model for Jose’s data. Suppose we want to predict his innate ability in the form of his true batting average, p. This would be the hierarchical model for our data:

We now are ready to compute what is called a **posterior distribution** to summarize our prediction of p. What Bayesian statistics let us do is compute the probability distribution of p given that we have observed data. This is called a **posterior distribution**. There is a continuous version of Bayes’ rule that let us compute the posterior distribution in cases like this, where the distribution is continuous. The normal distribution is a continuous distribution. We can use this continuous version of Bayes’ rule to derive a posterior distribution function for p assuming that we have observed Y equals, for example y. In our case, we can show that his posterior distribution follows a normal distribution with expected value:

So if B were to be 1, this would mean that we’re just saying Jose is just an average player, so we’re going to predict mu. If B is 0, we would be saying forget the past, we’re going to predict that Jose is what he is, what we’ve observed. His average is 0.450. Now look how B is constructed, B is the standard error sigma squared divided by the sum of the standard error sigma squared, plus the standard error tau squared. So B, the weight, is going to be closer to 1 when sigma is large, i.e when the standard error of our observed data is large. When we don’t trust our observed data too much sigma is large. So we make B=1. In this case, we would predict that Jose Iglesias is an average player. We would predict mu. On the other hand, if the signa is very small, this means that we really do trust our data Y, and we’re going to say, no, we trust our data, and we are going to actually ignore the past and predict Y.

Of course, B is somewhere in the middle. This weighted average is sometimes referred to as **shrinking**, because it shrinks the observed Y towards a prior mean, which in this case is mu. We shrink the observed data towards what the average is, mu. In the case of Jose Iglesias, we can fill in those numbers:

The expected value is a number between the 0.450 that we saw and the 0.270 that we have seen historically for the average player. The standard error can also be computed:

So we started with a frequentist 95% confidence interval that ignored data from other players from the past and simply summarized Jose’s data as 0.450 plus or minus 0.220. Then we used a Bayesian approach that incorporated data from the past, from other players and obtained a posterior probability. Note that this is referred as empirical Bayesian approach. In a traditional Bayesian approach, we simply state the prior. Using the posterior distribution, we can report what is called a 95% credible interval. This is a region centered at the expected value with a 95% chance of occurring. Remember that p is now random, so we can talk about the chances of p happening. Falling here or falling there. In our case, we can construct this by adding 2 the standard eror to the expected value of the posterior distribution:

Note that the Bayesian approach is giving us a prediction that is much lower than the 0.450. it’s also giving us a much more precise interval.

1. Election Forecasting
   * 1. Election Forecasting

Pollsters tend to make probabilistic statements about the result of the election. For example, the chance that Obama wins the electoral college is 91%. That is a probabilistic statement about the parameter d. We show that for the 2016 election, FiveThirtyEight gave Clinton 81.4% chance of winning the popular vote, and that happened. To do this, they used the Bayesian approach we describe here. We assume a hierarchical model similar to the one we used to predict the performance of a baseball player. In this case, we write it like this:

d, the spread, is going to be assumed to come from a normal distribution with expected value mu and standard error tau. This describes our best guess before we see any polling data. Then, once we collect data for a given spread and compute and average, we have that this is going to be normally distributed with expected value d and standard error sigma. This probability distribution describes randomness due to sampling and the pollster effect. Four our best guess, we note that before any poll is available we can use data sources other than polling data.

A popular approach is to use what are called fundamentals, which are based on properties about the current economy and other factors that historically appear to have an effect in favor or against an incumbent party. We don`t use those here. Instead we’ll simply set mu to 0 which is interpreted as a model that simply does not provide any information on who will win. For the standard deviation, we will use recent historical data that shows the winner of the popular vote has an average spread of about 3.5%, so we set tau to 0.035.

Now we can use the formulas for the posterior distribution for the parameter d that we previously learned to report the probability of d being bigger than 0 given the observed poll data. With this code:

*mu<-0*

*tau<-0.035*

*sigma<-results$se*

*Y<-results$avg*

*B<-sigma^2/(sigma^2+tau^2)*

*posterior\_mean<-B\*mu+(1-B)\*Y*

*posterior\_se<-sqrt(1/(1/sigma^2+1/tau^2))*

*posterior\_mean*

*[1] 0.02808534*

*posterior\_se*

*[1] 0.006149604*

To make a probability statement, we use the fact that the posterior distribution is also normal. So one thing we can do is report what is called a credible interval.

The posterior mean plus minus 1.96 time the posterior standard error to the posterior mean plus 1.96 times the posterior standard error give us interval that has a 95% chance of occurring. The interval is now random:

*posterior\_mean+c(-1.96,1.96)\*posterior\_se*

*[1] 0.01603212 0.04013857*

We can also report the probability that d is bigger than 0 that we can compute using pnorm:

*1-pnorm(0,posterior\_mean,posterior\_se)*

*[1] 0.9999975*

That’s the probability that we’re going to report for Clinton winning the popular vote. We’re saying it’s almost 100%. This seems to be a little bit too overconfident. It is not in agreement with the 81.4% of FiveThirtyEight. To understand the difference, we can look at what happens after the elections. After the election, one can look at the difference between the pollsters’ prediction and the actual result. An important observation that our model does not account for is that it is common to see what is called a general bias that affect many pollsters in the same way. There is no good explanation for this. We observe in historical data. One election the average of polls favors Democrats by 2%. The next election they favor Republicans with 1%. Then next there’s no bias. Then the next Republicans are favored with 3% and so on

* + 1. Mathematical Representations of Models

Suppose we are collecting data from one pollster, and we assume there’s no general bias. The pollster collects several polls with a sample size of N. So we observe several measurements of the spread let’s call it:

The theory tell us that these random variables have expected value d and standard error:

We can represent this model mathematically like this:

We use the Greek letter ε for the error and we use the index j to represent different polls. is define as the random variable that explains the poll to poll variability introduced by the sampling error. To do this we assume we have expected value of 0 and standard error :

If d is 2.1 and the sample size for these polls was, say, 2000 we could simulate six data points from a model using this simple code:

*J<-6*

*N<-2000*

*d<-0.21*

*p<-(d+1)/2*

*X<-d+rnorm(J,0,2\*sqrt(p\*(1-p)/N))*

Now, suppose we have six data points from 5 different pollsters. To represent this, we now need 2 indices; one for pollster and one for the polls each pollster takes. We are going to use:

With i representing the pollster and j representing the jth poll from a given pollster. If we apply the same model , we would then write:

To simulate data, we now have to use a loop. We’re going to use the sapply function like this:

*I<-5*

*J<-6*

*N<-2000*

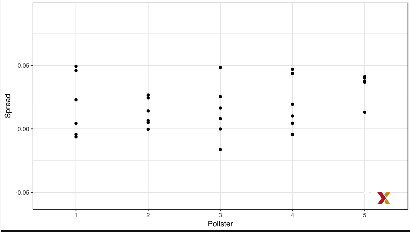
*X<-sapply(1:I,function(i){*

*+ d+rnorm(J,0,2\*sqrt(p\*(1-p)/N))*

*+ })*

This creates data for five different pollsters. Here is the simulated data

1. Simulated data of 6 polls taken by 5 pollsters under the model



The simulated data does not really seem to capture an important feature of the actual data, which we can see here. The model does not account for pollster to pollster variability. To fix this, we add a new term for the pollster effect. We’re going to use:

To represent the house effect for the ith pollster. So now the model looks like this:

To simulate the data for a specific pollster, we now need to draw an for each pollster then add the epsilons. We can do this using this simulation:

*I<-5*

*J<-6*

*N<-2000*

*d<-0.021*

*p<-(d+1)/2*

*h<-rnorm(I,0,0.025)*

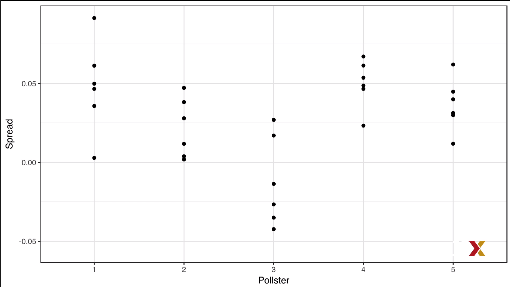
*X<-sapply(1:I,function(i){*

*+ d+h[i]+rnorm(J,0,2\*sqrt(p\*(1-p)/N))*

*+ })*

The simulated data now looks like this:

1. Simulated data of 6 polls taken by 5 pollsters under the model



Note that is common to all observed spreads from a specific pollster. Different pollster have different , which explains why we can see the groups of points shift up and down from pollster to pollster.

Now in the model, we assume the average house effect is 0. We think that for every pollster that’s biased in favor of one party, there’s another that is favored in favor of the other, so it all averages out. But historically, we see that every election has a general bias affecting all polls, as we said earlier. We can’t observe this with just the 2016 data. But if we were to collect historical data, we will see that the average of polls misses by more than models like the one we showed would predict. To see this we would take for each election year the average of polls and compare it to the actual value. If we did this, we would see differences with standard deviations of between 2% and 3%. To incorporate this into the model, we can add yet another term that accounts for this general bias variability.

So now the model looks like this:

The b is modeled to have expected value 0 and we assume. Based on historical data, that the standard error is about 0.025. Note that the variability of b is not observed in one year, because every single poll we observe that year has this general bias. So we don’t see that variability. Every single poll has the same value. An implication of adding this term to the model, though, is that the standard deviation of is actually higher than what we earlier called sigma which combines the pollster variability and the sample n variability. We have to add the general bias variability. Since we add this, now we note that sample average which is :

Implies that the standard deviation of includes this term :

Because the same b is in every measurement, the average does not reduce its variance. This is an important point. It does not matter how many polls you take. The bias does not get reduced by taking averages. If we redo the Bayesian calculation taking this variability into account, we get a result much closer to FiveThirtyEight. We write the code again:

*mu<-0*

*> tau<-0.035*

*> sigma<-sqrt(results$se^2+0.025^2)*

*> Y<-results$avg*

*> B<-sigma^2/(sigma^2+tau^2)*

*> posterior\_mean<-B\*mu+(1-B)\*Y*

*> posterior\_se<-sqrt(1/(1/sigma^2+1/tau^2))*

*> 1-pnorm(0,posterior\_mean,posterior\_se)*

*[1] 0.8174373*

We get a probability of Clinton winning the popular vote of 81.7% much lower than the 99.999, again because we’re including the general bias variability.

* + 1. Predicting the Electoral College

Up to now we have focused on the popular vote. But in the U.S. elections are not decided by popular vote, but rather by what is called the electoral college. Each state and DC get a number of electoral votes that depend in a somewhat complex way on the population size of the state. Here are the top five states ranked by electoral votes:

*results\_us\_election\_2016%>%arrange(desc(electoral\_votes))%>%top\_n(5,electoral\_votes)*

*state electoral\_votes clinton trump others*

*1 California 55 61.7 31.6 6.7*

*2 Texas 38 43.2 52.2 4.5*

*3 Florida 29 47.8 49.0 3.2*

*4 New York 29 59.0 36.5 4.5*

*5 Illinois 20 55.8 38.8 5.4*

*6 Pennsylvania 20 47.9 48.6 3.6*

We can see California has 55 votes, Texas has 38 votes etc.

With some minor exceptions that we don’t discuss, the electoral votes are won all or nothing. So, for example, if you win California by just one vote, you still get all of its 55 electoral votes. This means that by winning a few big states by a large margin, but losing many small states by a small margin, you can win the popular vote and lose the electoral college. This happened in 1876, 1888, 2000 and 2016. The idea behind this is to avoid a few large states having too much power and dominate the presidential election. Although many in the US consider the electoral college unfair and would like to see it changed, this is how the elections are decided. We are now ready to predict the electoral college result for 2016. We start by aggregating results from polls taken during the last week before the election. We write this code:

results<-polls\_us\_election\_2016%>%

+ filter(state!="U.S."&

+ !grepl("CD",state) &

+ enddate>="2016-10-31" &

+ (grade %in%c("A+","A","A-","B+")|is.na(grade)))%>%

+ mutate(spread=rawpoll\_clinton/100-rawpoll\_trump/100)%>%

+ group\_by(state)%>%

+ summarize(avg=mean(spread),sd=sd(spread),n=n())%>%

+ mutate(state=as.character(state))

Here are the 10 closest races according to the polls already summarized:

*results%>%arrange(abs(avg))*

*# A tibble: 47 x 4*

*state avg sd n*

*<chr> <dbl> <dbl> <int>*

*1 Florida 0.00356 0.0163 7*

*2 North Carolina -0.00730 0.0306 9*

*3 Ohio -0.0104 0.0252 6*

*4 Nevada 0.0169 0.0441 7*

*5 Iowa -0.0197 0.0437 3*

These are called “battleground states”. We now introduce a command called **left\_join()** and it will let us easily add the number of electoral votes for each state. Note that some states have no polls. This is because a winner is pretty much known. No polls were conducted in DC, Rhode Island, Alaska and Wyoming, because the first two are sure to be Democrats and the last two Republicans. This code assigns a standard deviation to the states that had just one poll by substituting the missing value by the median of the standard deviation of all the other states:

*results<-left\_join(results,results\_us\_election\_2016,by="state")*

*results\_us\_election\_2016%>%filter(!state %in% results$state)*

*state electoral\_votes clinton trump others*

*1 Rhode Island 4 54.4 38.9 6.7*

*2 Alaska 3 36.6 51.3 12.2*

*3 Wyoming 3 21.9 68.2 10.0*

*4 District of Columbia 3 90.9 4.1 5.0*

*results<-results%>%mutate(sd=ifelse(is.na(sd),median(results$sd,na.rm=TRUE),sd))*

We’re going to use a Monte Carlo simulation to generate outcomes from simulated elections. Then we’re going to use this to make probability statements. For each state, we apply the Bayesian approach we learned to generate Election Day d for each state. We could construct priors for each state based on recent history. However, to keep it simple, we assign the same prior to each state. This prior is going to assume that we know nothing about what will happen. So, the expected value will be 0. Because from election year to election year the results from specific state don’t change that much, we will assign a standard deviation of 2%. So tau is going to be now 0.02.

The Bayesian calculation looks like this:

mu<-0

> tau<-0.02

> results%>%mutate(sigma=sd/sqrt(n),

+ B=sigma^2/(sd^2+tau^2),

+ posterior\_mean=B\*mu+(1-B)\*avg,

+ posterior\_se=sqrt(1/(1/sigma^2+1/tau^2)))%>%

+ arrange(abs(posterior\_mean))

# A tibble: 47 x 12

state avg sd n electoral\_votes clinton trump others sigma B

<chr> <dbl> <dbl> <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl>

1 Flor~ 0.00356 0.0163 7 29 47.8 49 3.2 0.00618 0.0572

2 Nort~ -0.00730 0.0306 9 15 46.2 49.8 4 0.0102 0.0779

3 Ohio -0.0104 0.0252 6 18 43.5 51.7 4.8 0.0103 0.102

4 Iowa -0.0197 0.0437 3 6 41.7 51.1 7.1 0.0252 0.276

5 Neva~ 0.0169 0.0441 7 6 47.9 45.5 6.6 0.0167 0.118

6 Mich~ 0.0209 0.0203 6 16 47.3 47.5 5.2 0.00827 0.0844

7 Ariz~ -0.0326 0.0270 9 11 45.1 48.7 6.2 0.00898 0.0717

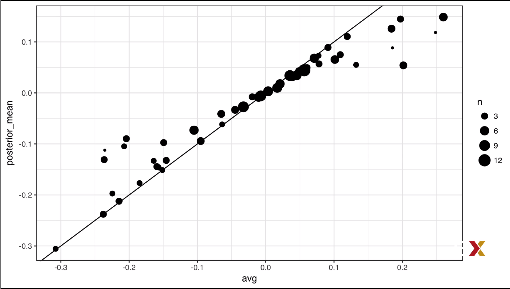
8 Penn~ 0.0353 0.0116 9 20 47.9 48.6 3.6 0.00387 0.0280

9 New ~ 0.0389 0.0226 6 5 48.3 40 11.7 0.00921 0.0933

10 Geor~ -0.0448 0.0238 4 16 45.9 51 3.1 0.0119 0.147

Note that estimates based on posteriors move the estimates towards 0, although the states with many polls are influenced less, you can see it with this plot:

1. Posterior mean vs average group by number of polls taken



This is expected, as the more poll data we collect, the more we trust those results. Now we repeat this 10000 times and generate an outcome from the posterior:

*clinton\_EV<-replicate(1000,{*

*+ results %>% mutate(sigma= sd/sqrt(n),*

*+ B= sigma^2/ (sigma^2+tau^2),*

*+ posterior\_mean=B\*mu+(1-B)\*avg,*

*+ posterior\_se=sqrt(1/(1/sigma^2+1/tau^2)),*

*+ simulated\_result=rnorm(length(posterior\_mean),posterior\_mean,posterior\_se),*

*+ clinton=ifelse(simulated\_result>0,electoral\_votes,0))%>%*

*+ summarize(clinton=sum(clinton))%>%*

*+ .$clinton+7##7 for Rhode Island and D.C.*Forecasting

*mean(clinton\_EV>269)*

*[1] 0.996*

This model gives Clinton over 99% chance of winning, as we can see by writing this code.

Here’s a histogram of the outcomes:

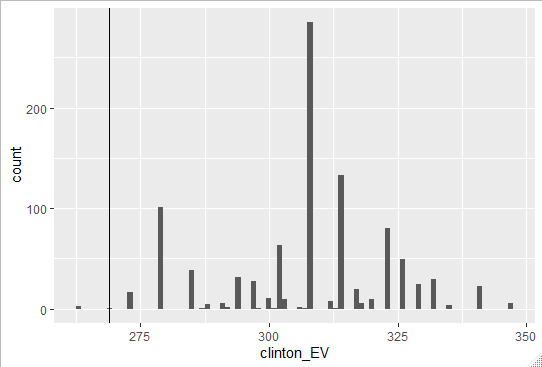
*data.frame(clinton\_EV)%>%*

*+ ggplot(aes(clinton\_EV))+*

*+ geom\_histogram(binwidth = 1)+*

*+ geom\_vline(xintercept = 269)*

1. Histogram of outcomes of Monte Carlo simulation of 1000 elections



Note that a similar prediction was made by the Princeton Election Consortium. We know that they were quite off, so what happened? Note that the model that we just showed ignores the general bias. The general bias in 2016 was not that big compared to other years, but it was between 1% and 2%. But because the election was close in several big states and the large number of polls made the estimates of the standard error small, by ignoring the variability introduced by the general bias, pollsters were overconfident of the polling data. FiveThirtyEight on the other hand, models the general bias in a rather sophisticated way and reported closer results. We can simulate the results now using this bias term. For the state level, we’re going to assume the general bias is larger.

So we’re going to set sigma b=0.03, this code recompute the Monte Carlo simulation but accounting for the general bias:

*clinton\_EV2<-replicate(1000,{*

*+ results%>%mutate(sigma=sqrt(sd^2+bias\_sd^2),*

*+ B=sigma^2/(sigma^2+tau^2),*

*+ posterior\_mean=B\*mu+(1-B)\*avg,*

*+ posterior\_se=sqrt(1/(1/sigma^2+1/tau^2)),*

*+ simulated\_results<-rnorm(length(posterior\_mean),posterior\_mean,posterior\_se),*

*+ clinton=ifelse(simulated\_results>0,electoral\_votes,0))%>%*

*+ summarize(clinton=sum(clinton)+7)%>%.$clinton## 7 for Rhodes Island and D.C.*

*+*

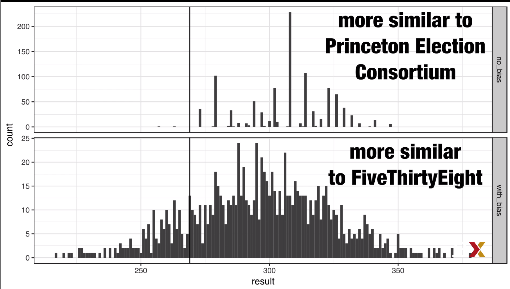
*+ })*

*mean(clinton\_EV2>269)*

*[1] 0.753*

When we do this, the probability of Clinton winning goes way down to 75%. Looking at the outcomes of the simulation for these two approaches, we see how bias term adds variability to the final results you can see this:

1. Histogram of outcomes of Monte Carlo simulation of 1000 elections with genreal bias included



FiveThirtyEight includes many other features we did not include here. One is that they model variability with distributions that have higher probabilities for extreme events compared to what the normal distribution give us. They ended up predicting a probability of 71%.

* + 1. Forecasting

Forecasters like to make predictions as well before the election. The predictions are adapted as new polls come out. However, an important question forecaster must ask is, how informative are polls taken several weeks before the election? Here we study the variability of poll results across time. In our example, to make sure the variability we observe is not due to pollster effects, we’re going to stick to just one pollster, using this code:

*one\_pollster<-polls\_us\_election\_2016%>%*

*+ filter(pollster=="Ipsos" & state=="U.S.")%>%*

*+ mutate(spread=rawpoll\_clinton/100-rawpoll\_trump/100)*

Since there is no pollster effect, perhaps the theoretical standard error will match the data derived standard deviation. We can compute both like this:

*se<-one\_pollster %>%*

*+ summarize(empircal=sd(spread),*

*+ theoretical=2\*sqrt(mean(spread)\*(1-mean(spread))/min(samplesize)))*

*> se*

*empircal theoretical*

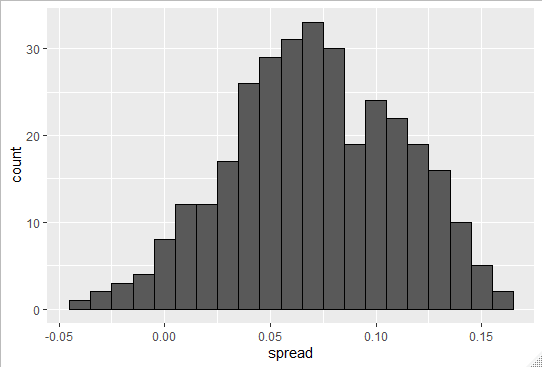
*1 0.0403 0.0326*

And we see that the empirical standard deviation is a little bit higher than the theoretical one. Furthermore, the distribution of the data does not look normal as the theory would predict, as we can see in this figure:

*one\_pollster%>%ggplot(aes(spread))+*

*+ geom\_histogram(binwidth = 0.01,color="black")*

1. Histogram of Ipsos spread distributions



Where is this extra variability coming from? This plot makes a strong case that the extra variability comes from time variation not accounted for by the theory that assumes that p is fixed across time. Some of the peaks and valleys we see coincide with events such as the party conventions, which tend to give candidates a boost. We can see them consistently across several pollsters, not just one. With this code:

*polls\_us\_election\_2016%>%filter(state=="U.S." & enddate>="2016-07-01")%>%group\_by(pollster)%>%*

*filter(n()>10)%>%*

*ungroup()%>%*

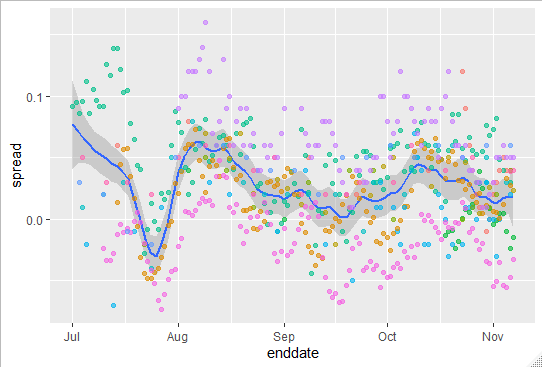
*mutate(spread=rawpoll\_clinton/100-rawpoll\_trump/100)%>%*

*ggplot(aes(enddate,spread))*

*+geom\_smooth(method="loess",span=0.1)*

*+geom\_point(aes(color=pollster),show.legend = FALSE,alpha=0.6)*

1. Distribution of spreads across time and pollsters



This implies that if we’re going to forecast, our model must include a term to model the time effect. We could write a model that includes a bias term for time like this:

The standard deviation of would depend on time, since the close we get to the election day, the smaller this variability should become. Pollsters also try to estimate trends. Let’s call them f(t). They try to estimate them from the data and incorporate them into the predictions, the model would then look like this:

In many pollsters’ websites, we see the estimated f(t), not for the difference but for the actual percentages for the two main candidates. Like this:

*polls\_us\_election\_2016%>%*

*filter(state=="U.S." & enddate>="2016-07-01")%>%*

*select(enddate,pollster,rawpoll\_clinton,rawpoll\_trump)%>%*

*rename(Clinton=rawpoll\_clinton,Trump=rawpoll\_trump)%>%*

*gather(candidate,percentage,-enddate,-pollster)%>%*

*mutate(candidate=factor(candidate,level=c("Trump","Clinton")))%>%*

*group\_by(pollster)%>%*

*filter(n()>=10)%>%*

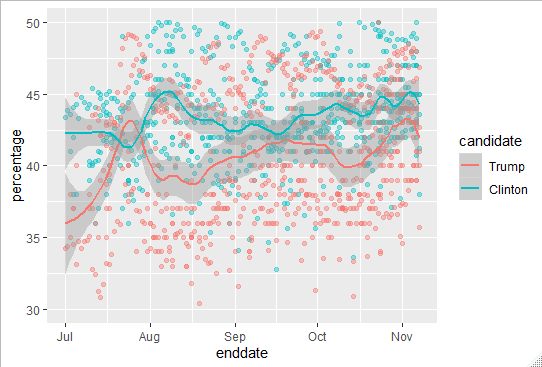
*ungroup()%>%*

*ggplot(aes(enddate,percentage,color=candidate))*

*+geom\_point(show.legend = FALSE,alpha=0.4)*

*+geom\_smooth(method="loess",span=0.15)+scale\_y\_continuous(limits = c(30,50))*

1. Estimates for the difference of actual percentages for the 2 main candidates of the 2016 U.S. election



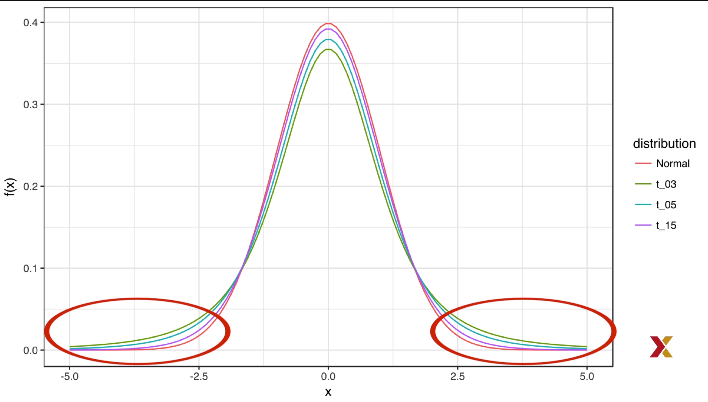
* + 1. The t-Distribution

Previously, we made use of the Central Limit Theorem with sample sizes as small as 15. Because we’re also estimating a second parameter, sigma, further variability is introduced into our confidence interval, and this results in a confidence interval that is overconfident because it doesn’t account for that variability. For very large sample sizes, this extra variability is negligible. But in general, for values smaller than 30, we need to be cautious about using the Central Limit Theorem. However, if the data in the urn is known to follow a normal distribution, in other words, if the population data is known to follow a normal distribution, then we actually have a mathematical theory that tell us how much bigger we need to make the intervals to account for the estimation of sigma. Using this theory, we can construct confidence intervals for any urn, but again only if the data in the urn is known to follow a normal distribution. So, for the 0, 1 data of previous urn models, this theory does definitely not apply. The statistic on which confidence intervals for are based is this one:

We call it Z. The CLT tell us that Z is approximately normal distributed with expected value 0 and standard error 1. But in practice, we don’t know sigma, so we use s instead:

By doing this we introduce some variability. The s, as variability, is estimated from data. This theory that we mentioned tell sus that Z follows what is called a t-distribution with what is called N minus 1 degrees of freedom. The degrees of freedom is a parameter that controls the variability via what are called fatter tails.

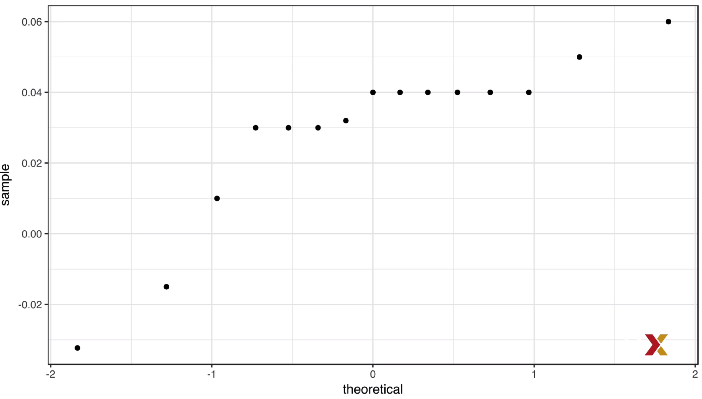
1. T distribution with degree of freedom 3, 5 and 15



You can see how the tails, the ends, go higher and higher, meaning that large values have larger probabilities for smaller values of the degrees of freedom. In our case of pollster data, if we are willing to assume that the pollster effect data is normally distributed, then we can use this theory.

Based on the sample, we can corroborate if, in fact, the data is normally distributed. Here’s a q-q plot showing us a normal distribution versus our sample data

1. Q-Q plot of sample data versus the normal distribution



It’s not a perfect match but is relatively close, and this particular theory is quite robust to deviations from normality. So once we male that decision, then, perhaps a better confidence interval for is constructed using the t distribution instead of the normal distribution:

*z<-qt(0.975,nrow(one\_poll\_per\_pollster)-1)*

*one\_poll\_per\_pollster%>%*

*summarize(avg=mean(spread),moe=z\*sd(spread)/sqrt(length(spread)))%>%*

*mutate(start=avg-moe,end=avg+moe)*

*# A tibble: 1 x 4*

*avg moe start end*

*<dbl> <dbl> <dbl> <dbl>*

*1 0.0290 0.0134 0.0156 0.0424*

The new confidence interval goes from 1.5% to 4.2%. So, it is a little bit bigger than the one we made using the normal distribution.

This is, of course, expected because the quantile from the t-distribution is larger than the quantile from the normal distribution, as we can see here:

*qt(0.975,14)*

*[1] 2.14*

*> qnorm(0.975)*

*[1] 1.96*

FiveThirtryEight uses the t-distribution to generate errors that better model the deviation we see in election data. Again, because they have fatter tails. So, for example, the deviation we saw in Wisconsin between the polls and the actual result ( Trump won by 0.7%) is more in line with t-distributed data,

1. Association Tests
   1. Assessment 7.1: Association and Chi-Squared Tests
      1. Association Tests

The statistical tests we have covered up to now leave out a substantial portion of data types, specifically, we have not discussed inference for binary, categorical ordinal data. To give a very specific example, consider the following case study. A 2014 PNAS paper analyzed success rates from funding agencies in the Netherlands and concluded that their “results reveal gender bias favoring male applicants in the prioritization of their quality of research”. The main evidence for this conclusion comes down to a comparison of the percentages. The first table in the supplement of the paper include the information we need. It is included in the dslabs package.

*data("research\_funding\_rates")*

*> research\_funding\_rates*

*discipline applications\_total applications\_men applications\_women awards\_total*

*1 Chemical sciences 122 83 39 32*

*2 Physical sciences 174 135 39 35*

*3 Physics 76 67 9 20*

*4 Humanities 396 230 166 65*

*5 Technical sciences 251 189 62 43*

*6 Interdisciplinary 183 105 78 29*

*7 Earth/life sciences 282 156 126 56*

*8 Social sciences 834 425 409 112*

*9 Medical sciences 505 245 260 75*

*awards\_men awards\_women success\_rates\_total success\_rates\_men success\_rates\_women*

*1 22 10 26.2 26.5 25.6*

*2 26 9 20.1 19.3 23.1*

*3 18 2 26.3 26.9 22.2*

*4 33 32 16.4 14.3 19.3*

*5 30 13 17.1 15.9 21.0*

*6 12 17 15.8 11.4 21.8*

*7 38 18 19.9 24.4 14.3*

*8 65 47 13.4 15.3 11.5*

*9 46 29 14.9 18.8 11.2*

We can compute the differences in percentages for men and women:

*totals<-research\_funding\_rates%>%*

*select(-discipline)%>%*

*summarize\_all(funs(sum))%>%*

*summarize(yes\_men=awards\_men,*

*no\_men=applications\_men-awards\_men,*

*yes\_woman=awards\_women,*

*no\_women=applications\_women-awards\_women)*

*totals%>%*

*summarize(percent\_man=yes\_men/(yes\_men+no\_men),*

*percent\_women=yes\_woman/(yes\_woman+no\_women))*

*percent\_man percent\_women*

*1 0.177 0.149*

We see a larger percent for men received awards than women. But could this be due just to random variability? Here we learn how to perform inference for this type of data.

R.A. Fisher was one of the first to formalize hypothesis testing. The Lady Tasting Tea is one of the most famous examples. The story goes like this :

Muriel Bristol, a colleague of Fisher’s, claimed that she could tell if milk was added before or after tea was poured. Fisher was skeptical. He designed an experiment to test this claim. He gave her 4 pairs of cups of tea, 1 with milk poured first, the other after. The order of the two was randomized. The null hypothesis here is that she was just guessing. Fisher derived the distribution of the number of correct picks on the assumption that the choices were random and independent. As an example, suppose she picked 3 out of 4 correctly. Do we believe she has a special ability based on this? The basic question we ask is, if the tester is actually guessing, what are the chances that she gets 3 or more correct? Just as we have done before, we can compute a probability under the null hypothesis that she’s guessing four of each.

Under this null hypothesis, we can think of this particular example as picking 4 beads out of an urn where 4 are blue. Those are the correct answers. And 4 are red. Those are the incorrect answers. Remember that she knows that there are 4 before tea and 4 after. Under the null hypothesis that she’s simply guessing, each bead has the same chance of being picked. We can then use the combinatorics to figure out each probability, the probability of picking 3 can be derived by:

The probability of picking 4 correct is given by this formula:

Thus, the chance of observing 3 correct answers or more under the null hypothesis is approximately 0.24. This is the p-value. The procedure that produces p-value is called Fisher’s exact test, and it uses the hypergeometric distribution to compute the probabilities.

The data from this type of experiment is usually summarized by a table like this:

*tab<-matrix(c(3,1,1,3),2,2)*

*> rownames(tab)<-c("Poured Before","Poured After")*

*> colnames(tab)<-c("Guessed Before","Guessed After")*

*> tab*

*Guessed Before Guessed After*

*Poured Before 3 1*

*Poured After 1 3*

The function fisher.test performs the inference calculations and can be applied to the 2 by 2 table using this simple piece of code:

*fisher.test(tab,alternative="greater")*

*Fisher's Exact Test for Count Data*

*data: tab*

*p-value = 0.2*

*alternative hypothesis: true odds ratio is greater than 1*

*95 percent confidence interval:*

*0.314 Inf*

*sample estimates:*

*odds ratio*

*6.41*

* + 1. Chi-Squared Tests

Note that, in a way, our funding rates case study is similar to the lady tasting tea example. However, in the tasting tea example, the number of blue and red beads is experimentally fixed. The number of answers given for each category is also fixed. This is because Fischer made sure there were 4 before tea and 4 after tea, the lady knew this so the answer is would also have 4 and 4. If this is the case, the sums of the rows and the sum of the columns of the 2x2 table are fixed. This defines a constraint on the possible ways we can fill the 2x2 table and also permits us to use the hypergeometric distribution. In general, this is not the case. Nonetheless, there’s another very similar approach. The **chi-squared test;** imagine we have 2823 individuals. Some are men and some are women, some get funded; others don’t. There you have 2 binary variables. We saw that the success rate for men were about 18% and 15% for the women. We can compute the overall funding rate:

*funding\_rate<-totals%>%*

*summarize(percent\_total=*

*(yes\_men+yes\_woman)/*

*(yes\_men+no\_men+yes\_woman+no\_women))%>%*

*.$percent\_total*

*funding\_rate*

*[1] 0.1654*

So now the question is will we see a difference between men and women as big as the one we see if funding was assigned at random using this rate? The chi-squared test answers this question. The first step is to create a 2x2 table just like before. In our case, we can use this code and construct the table:

*two\_by\_two<-tibble(awarded=c("no","yes"),*

*men=c(totals$no\_men,totals$yes\_men),*

*women=c(totals$no\_women,totals$yes\_woman))*

*two\_by\_two*

*# A tibble: 2 x 3*

*awarded men women*

*<chr> <dbl> <dbl>*

*1 no 1345 1011*

*2 yes 290 177*

The general idea of a chi-squared test is to compare this 2x2 table, the observed 2x2 table to what you expect to see at the overall funding rate, which we can compute using this code:

*tibble(awared=c("no","yes"),*

*men=(totals$no\_men+totals$yes\_men)\*c(1-funding\_rate,funding\_rate),*

*women=(totals$no\_women+totals$yes\_woman)\*c(1-funding\_rate,funding\_rate))*

*# A tibble: 2 x 3*

*awared men women*

*<chr> <dbl> <dbl>*

*1 no 1365. 991.*

*2 yes 270. 197.*

We can see that more men than expected and less women than expected received funding. However, under the null hypothesis, this observation is a random variable. The chi-squared test tells us how likely it is to see a deviation like this, or larger, by chance. This test uses an asymptotic result, similar to the central limit theorem, related to the sums of independent binary outcomes in a context like this. The R function **chisq.test()** takes a 2x2 table and returns the results from this test:

*two\_by\_two%>%*

*select(-awarded)%>%*

*chisq.test()*

*Pearson's Chi-squared test with*

*Yates' continuity correction*

*data: .*

*X-squared = 3.8, df = 1, p-value =0.05*

We that the p-value is 0.05, this means that the probability of seeing a deviation like the one we see or bigger under the null that funding is assigned at random is 0.05.

So, we described how to obtain p-values. But now let’s talk about summary statistics. An informative summary statistic associated with 2x2 tables in the odds ratio. Define the two variables X=1 if you are male or 0 otherwise and Y=1 if you’re funded and 0 otherwise.

The odds of getting funded if you’re a man is defined as follows:

and can be computed using the simple code:

*odds\_men<-(two\_by\_two$men[2]/sum(two\_by\_two$men))/(two\_by\_two$men[1]/sum(two\_by\_two$men))*

The odds of being funded if you’re a woman is given by this simple formula:

And can be computed like this:

*odds\_women<-(two\_by\_two$women[2]/sum(two\_by\_two$women))/(two\_by\_two$women[1]/sum(two\_by\_two$women))*

The odds ratio is the ratio of these two odds. How many times larger are the odds for men than for women? In this case, we get 1.23. A quick note of caution regarding p-values from 2x2 tables. As mentioned earlier, reporting only p-values is not an appropriate way to report the results of data analysis. In scientific journals, for example, some studies seem to overemphasize p-values. Some of these studies have large sample size and report impressively small p-values. Yet when one looks closely at the results, we realize that the odds ratios are quite modest, barely bigger than 1. In this case, the difference may not be practically significant or scientifically significant.

Note that the relationship between odds ratios and p-values is not one to one. It depends on the sample size. So a very small p-value does not necessarily mean a very large odds ratio. Look at what happens to the p-value if we multiply our 2x2 table by 10:

*two\_by\_two%>%select(-awarded)%>%mutate(men=men\*10,women=women\*10)%>%chisq.test()*

*Pearson's Chi-squared test with*

*Yates' continuity correction*

*data: .*

*X-squared = 40, df = 1, p-value =*

*3e-10*

The odds ratio remains the same, but the p-value becomes very small. Earlier we mentioned that instead of p-values, it’s more appropriate to report confidence intervals. However, computing confidence intervals for odds ratio is not mathematically straightforward. Unlike other statistics for which you can derive useful approximations for the distributions, the odds ratio is not only a ratio, but a ratio of ratios. Therefore, There’s no simple way to using, for example, the central limit theorem. One approach is to use the theory of generalized lineal models, which we will not see in this course. But you can learn more about in this book:

**“General Linear Models second edition”**,by P.McCullagh and J.A. Nelder