**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ AP Stats – Lab 6 – Instructions**

**Sampling Distribution of a Sample Proportion – Polling Voters**

In this lab, we simulate a population of voters, a certain proportion of whom will vote in favor of a particular proposition. We investigate the question of how accurately a random sample of voters can predict the outcome of an election.

Suppose the city of Edwardsville had a rent control initiative, Proposition A, on the ballot. The local newspaper is going to conduct a poll three weeks before the election to gauge public sentiment. Staff members need to know how big a sample to poll. **Our job is to set up a simulation they can use to determine, for any given sample size, the accuracy they should expect.** **We want to give them a visual tool that demonstrates how sample size affects the sampling distribution of the sample proportion.**

1. Open a new, empty Fathom document.
2. (a) Click File > Show Page Breaks. (b) Change the page orientation to landscape under File > Print Setup. As you work, attempt to keep your work on one page for easy printing. (c) Drag in a text object and put your name in the upper right-hand corner.

I. Modeling the Population of Voters

We start by creating a model for the population, the people who will vote in the election for or against the proposition. The model will consist of a single number—the *proportion of voters who will vote yes*.

1. Drag in a new slider object and name it *probYes*. Double-click the slider’s axis. This brings up the slider’s inspector, which lets us control the slider’s appearance and behavior. Edit the values for the *Lower* and *Upper* properties (the lower and upper bounds of the axis) to be 0 and 1.

The slider models the population. We’ll use a collection to model the simulated samples. We’ll start with samples of 100.

1. Drag in a new collection and name it *Sample of Voters* (double click the name of the collection to rename). Add 100 cases to the collection by right-clicking and selecting “New Cases.” Enter 100 when prompted.
2. Double-click the collection. The inspector now shows properties of the collection. (There is only one inspector window for any collections you may eventually have. You change what it inspects by double-clicking the desired object.)
3. In the inspector, create an attribute called *vote* by clicking <new> and typing the attribute name. We want the values for this attribute to be “yes” and “no.” The values will be drawn from an infinite population, whose proportion of yeses is set by the slider.

|  |  |
| --- | --- |
| Double-click the Formula field for the *vote* attribute to show the formula editor.Enter the formula shown at right by typing if(random( )<probYes Tab “yes” Tab “no” | clip0070 |

Close the formula editor.

For each case, Fathom will generate a random number between 0 and 1 and evaluate it. If the number is less than the slider’s value, Fathom will give the case the value “yes”; otherwise, Fathom will give the case the value “no.” (The function *random( )* has a minimum of 0 and a maximum of 1, unless you specify otherwise.)

1. It’s always good to check your simulations. Graph the *vote* attribute (drag in a graph object and then drag the *vote* attribute into the *x*-axis). You should now have the three objects shown below (and the inspector).



Collect a new sample by re-randomizing your collection (either right-click the collection and choose “rerandomize,” use the Collection menu, or use shortcut “Ctrl+Y”). Do this a few times and observe how the bars change, reflecting the results of a new sample.

1. Drag the slider’s thumb to change its value to somewhere around 0.80. The vote is no longer close; it is a slam dunk for the proposition.
2. Move the slider’s thumb back to somewhere near 0.50 to model a close election.

II. Simulating Repeated Surveys

We now have a population whose “true vote” is controlled by a slider, and we want to see how well a sample of 100 people accurately predicts election results (compared with other sample sizes we’ll do later). We need to run the simulation many times to see how well the sampling does in the long run.

We could simply rerandomize many times, each time recording the proportion of yeses for each run (our sample statistic of interest), or we can have Fathom do this grunt work for us. In Fathom, this is called *collecting measures*. First, we need to define measures to collect.

1. In the collection’s inspector, go to the Measures panel by clicking its tab. This looks much like the Cases panel, in that there’s a prompt for creating/naming measures and a Formula column for defining how each measure is computed. The interface for working with measures is similar to that of working with attributes, but measures themselves are different. Whereas attributes have distinct values for each case, a measure has one value for the collection as a whole (for us, this will be the sample proportion).

|  |  |
| --- | --- |
| 1. Create a measure called *propYesVote*. Give it the formula

 proportion(vote = “yes”)Check the value of the measure against the graph, and rerandomize a few times to check that your setup is working as expected. | clip0072 |

Now that we have defined a measure, we want to collect a lot of them in a new collection (to sample our population repeatedly). Let’s use the drag-and-drop method.

1. Drag in a new, empty collection, putting it to the right of the existing objects. Drop the name of the measure “propYesVote” onto this collection. You’ve told Fathom to “collect this.” Fathom rerandomizes the source collection of voters, calculates the measure, and stores it in the measures collection, which is now called *Measures from Sample of Voters*. Little green balls fly from the source to the measures collection to help show what’s happening. Note that the inspector for your new collection will appear; we’ll come back to that in a bit.
2. Make a case table for the measures collection by selecting your new “Measures” collection and then dragging in a new table object. What was a measure in the source collection is now an attribute in the measures collection; each case represents the results of one survey of 100 people. We got five cases by default (you can see this value set to five in the inspector).
3. Now graph these data by dragging in a new graph object, then dragging the attribute “propYesVote” onto the x-axis.
4. Now we need to do more surveys. The controls for the measures collection are in its inspector.

|  |  |
| --- | --- |
| -Double-click the measures collection to show its inspector.        -Go to the last panel, the Collect Measures panel.-Change the number of measures to collect from 5 to 20.-Click Collect More Measures.The sample rerandomizes in the graph of the sample of voters, giving a new collection of votes. The proportion of yeses then appears in the graph of the measures collection. Play with adding more measures a few times to see how the graph displays each sample proportion that is generated. | clip0075 |

III. Changing Sample Size

Now let’s investigate the effects of sample size. The trouble is, at the moment we aren’t collecting the sample size. To do that, we need another measure, which we need to define for the Sample of Voters collection.

1. Double-click the *Sample of Voters*. In the Measures panel, define a second measure, *SampleSize*, giving it the formula *count()*. (In Fathom, the count function, without any arguments in it, simply gives the number of cases in that collection.) We need to start the simulation over, because we’re now collecting the sample sizes.
2. Double-click the *Measures* collection, and check *Replace existing cases* in the *Collect Measures* panel. Also, change the number of measures to be collected from 10 to 50 measures. Finally, choose “Collect More Measures.” Notice that the case table first empties, then fills, with two attributes this time, one for each of the measures defined.
3. Uncheck Replace existing cases. You want to keep these measures after you change the sample size, not throw them away.
4. Select the *Sample of Voters* collection, and add 300 new cases (right click or choose Collection | New Cases). This changes your sample size to 400 for each survey.
5. Open the *Measures* collection a little bit by dragging its bottom-right corner down and out until you can see the *Collect More Measures* button. Click this button to collect 50 measures of samples of 400 each.
6. We need to add this information to our measures graph. What we want is a split dot plot. Drag *SampleSize* from the *Measures* collection’s case table and drop it on the vertical axis of the measures graph. You get a scatter plot, which isn’t what you want. You need *SampleSize* to be treated as a categorical attribute, not a numeric one…
7. Therefore, undo the last step. Drop the attribute on the vertical axis again, but this time hold down the Shift key when you drop. This forces Fathom to treat the numeric attribute as categorical, and you get a split dot plot, showing the distributions of proportions of yeses for the two sample sizes. We now have a tool to use with the newspaper staff to show them the effects of sample size on polling results: Smaller samples have more spread than larger, for example.



Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Class period: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Answer the following, then turn in **this page and your Fathom lab printout**:

1. For your particular results (we could have all set “probYes” to a different value on the slider), how would you report these results to the city if they were consulting you? In other words, to what extent would they get more accurate results using a sample size of 400 instead of 100?
2. Bonus: Repeat steps 19 and 20 to increase the sample size to something much larger, such as 1000. You should automatically see the results in your graph. Describe how much of a difference this made in the distribution of sample proportions generated.