Lisa Yan and Jerry Cain
CS 109

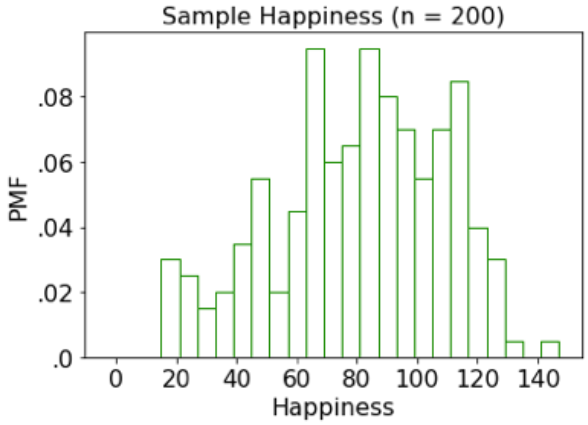
Lecture Notes #19
October 26, 2020

Samples and the Bootstrap

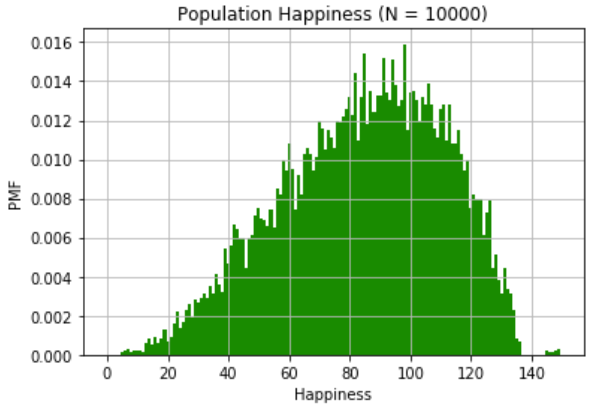
Based on a chapter by Chris Piech

Pre-recorded lecture: Up to Section 3.1
In-lecture: Section 3.2, 3.3

Let's say you are the king of Bhutan and you want to know the average happiness of the people in your country. You can't ask every single person, but you could ask a random subsample. In this next section we will consider principled claims that you can make based on a subsample. Assume we randomly sample 200 Bhutanese and ask them about their happiness. Our data looks like this: $72, 85, \ldots, 71$. You can also think of it as a collection of $n = 200$ IID (independent, identically distributed) random variables $X_1, X_2, \ldots, X_n$.



(a) The sample you collected



(b) The true population, but you definitely don't know this.

1 Estimating Mean and Variance from Samples

We assume that the data we look at are IID from the same underlying distribution (F) with a true mean (μ) and a true variance (σ^2). Since we can't talk to everyone in Bhutan, we have to rely on our sample to estimate the mean and variance. From our sample we can calculate a sample mean ($\bar{X}$) and a sample variance (S^2). These are the best guesses that we can make about the true mean and true variance.

$$\bar{X} = \sum_{i=1}^{n} \frac{X_i}{n} \qquad\qquad S^2 = \sum_{i=1}^{n} \frac{(X_i - \bar{X})^2}{n-1}$$

The first question to ask is: are those unbiased estimates? Yes. Unbiased, means that if we were to repeat this sampling process many times, the expected value of our estimates should be equal to

the true values we are trying to estimate. We will prove that that is the case for $\bar{X}$. The proof for S^2 is in the lecture slides.

$$E[\bar{X}] = E[\sum_{i=1}^{n} \frac{X_i}{n}] = \frac{1}{n} E\left[\sum_{i=1}^{n} X_i\right]$$
$$= \frac{1}{n} \sum_{i=1}^{n} E[X_i] = \frac{1}{n} \sum_{i=1}^{n} \mu = \frac{1}{n} n\mu = \mu$$

The equation for sample mean seems like a reasonable way to calculate the expectation of the underlying distribution. The same could be said about sample variance except for the surprising $(n-1)$ in the denominator of the equation. Why $(n-1)$? That denominator is necessary to make sure that the $E[S^2] = \sigma^2$.

The intuition behind the proof is that sample variance calculates the distance of each sample to the sample mean, *not* the true mean. The sample mean itself varies, and we can show that its variance is also related to the true variance.

2 Standard Error

Ok, you convinced me that our estimates for mean and variance are not biased. But now I want to know how much my sample mean might vary relative to the true mean.

$$\text{Var}(\bar{X}) = \text{Var}(\sum_{i=1}^{n} \frac{X_i}{n}) = \left(\frac{1}{n}\right)^2 \text{Var}\left(\sum_{i=1}^{n} X_i\right)$$
$$= \left(\frac{1}{n}\right)^2 \sum_{i=1}^{n} \text{Var}(X_i) = \left(\frac{1}{n}\right)^2 \sum_{i=1}^{n} \sigma^2 = \left(\frac{1}{n}\right)^2 n\sigma^2 = \frac{\sigma^2}{n}$$
$$\approx \frac{S^2}{n} \qquad \text{Since S is an unbiased estimate}$$
$$\text{SD}(\bar{X}) \approx \sqrt{\frac{S^2}{n}} = SE \qquad \text{Since Std is the square root of Var}$$

This SE term has a special name. It is called the **standard error** and is an estimate of $SD(\bar{X})$—it's how you report uncertainty of estimates of means in scientific papers (and how you get error bars). Great! Now we can compute all these wonderful statistics for the Bhutanese people.

Let's say we calculate the our sample of happiness has $n = 200$ people. The sample mean is $\bar{X} = 83$ (what is the unit here? happiness score?) and the sample variance is $S^2 = 793$. We can now calculate the standard error of our estimate of the mean to be 1.992. When we report our results we will say that the average happiness score in Bhutan is 83 ± 1.992 with variance 793.

Small technical sidenote: SE is actually the standard error *of the mean* (most people will know what you're talking about if you just say "standard error"). The standard error of a statistic in general is an estimate of the standard deviation of that statistic; for example, the standard error of the variance is an estimate of $SD(S^2)$. But wait! You never told me how to estimate $SD(S^2)$. True, that is outside the scope of CS109. You can find it on Wikipedia if you want.

3 The Bootstrap

The bootstrap is a newly invented statistical technique for both understanding distributions of statistics and for calculating p-values (a p-value is a the probability that a scientific claim is incorrect). It was invented here at Stanford in 1979 when mathematicians were just starting to understand how computers, and computer simulations, could be used to better understand probabilities.

The first key insight is that: if we had access to the underlying distribution (F) then answering almost any question we might have as to how accurate our statistics are becomes straightforward. For example, in the previous section we gave a formula for how you could calculate the sample variance from a sample of size n. We know that in expectation our sample variance is equal to the true variance. But what if we want to know the probability that the true variance is within a certain range of the number we calculated? That question might sound dry, but it is critical to evaluating scientific claims! If you knew the underlying distribution, F, you could simply repeat the experiment of drawing a sample of size n from F, calculate the sample variance from our new sample and test what portion fell within a certain range.

The next insight behind bootstrapping is that the best estimate that we can get for F is from our sample itself! The simplest way to estimate F (and the one we will use in this class) is to assume that $P(X = k)$ is simply the fraction of times that k showed up in the sample. Note that this defines the probability mass function of our estimate $\hat{F}$ of F.

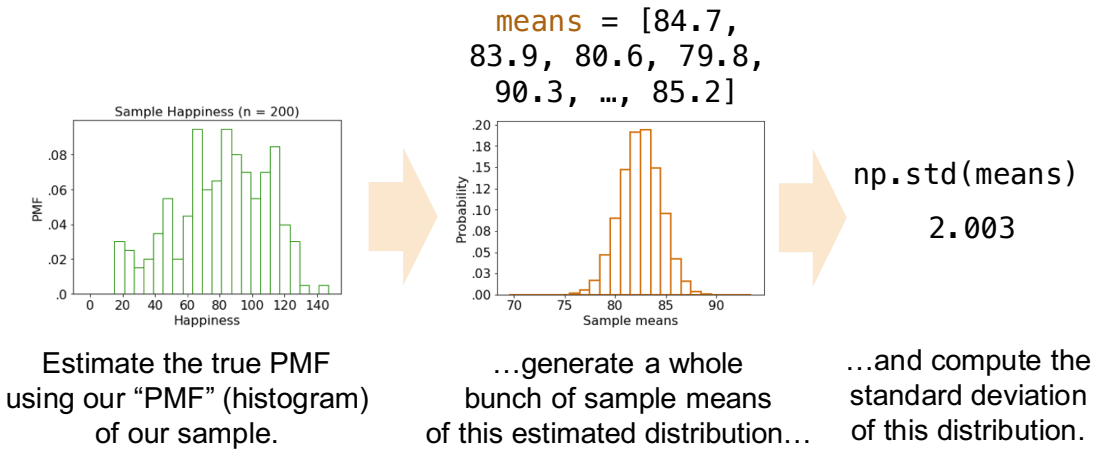
```
def bootstrap(sample):
   n = number of elements in sample
   pmf = estimate the underlying pmf from the sample
   stats = []
   repeat 10,000 times:
      resample = draw n new samples from the pmf
      stat = calculate your stat on the resample
      stats.append(stat)
   stats can now be used to estimate the distribution of the stat
```

The bootstrap has strong theoretic guarantees, and is accepted by the scientific community, when calculating any statistic. It breaks down when the underlying distribution has a “long tail" or if the samples are not IID.

3.1 Example: Bootstrapping sample mean

If we didn’t have that magical formula for **standard error** $SE = \sqrt{S^2/n} \approx \text{Var}(\bar{X})$, we would still be able to compute a *bootstrapped* standard error. We could resample 10,000 times and compute each resample’s mean as $\bar{X}_i$. This gives us a bootstrapped distribution on $\bar{X}$. We could then report the standard deviation of these 10,000 values.

When we tried this out with our Bhutanese dataset, we got a bootstrapped error of 2.003. This was pretty close to our standard error computation, where $SE = 1.992$.



3.2 *Example: Bootstrapping sample variance*

To estimate $\text{Var}(S^2)$, we could calculate S_i^2 for each resample i and after 10,000 iterations, we could calculate the variance of the S_i^2s.

The Python code below computes a *bootstrapped standard error of variance*—i.e., an estimate for $\sqrt{\text{Var}(S^2)}$–via bootstrapping with 10,000 iterations. Notice that we are sampling *with replacement* by using `np.random.choice(sample, size=len(sample), replace=True)`, where n = `len(sample)`.

```
def compute_sample_variance(data):
    tot_dev = 0
    sample_mean = np.mean(data)
    n = len(data)
    for s in data:
        tot_dev += (s - sample_mean) ** 2
    return tot_dev/(n - 1)

def se_sample_variance(sample, niters=10000):
    sample_variance_boots = []
    for i in range(niter):
        resample = np.random.choice(sample, len(sample), replace=True)
        resample_s_2 = compute_sample_variance(resample)
        sample_variance_boots.append(resample_s_2)
    return np.std(sample_variance_boots)
```

3.3 *Example: Bootstrapping for p-values*

The bootstrap can also be used to perform **statistical hypothesis testing**.

Suppose that we had two samples of happiness from two different countries—Bhutan and Nepal—and we wanted to determine if the difference in average happiness between the two groups was

statistically significant. In other words, is it possible that the difference in means between these two groups could just have arisen due to chance? Hypothesis testing generally follows a few steps:

1. **State the null hypothesis.** Assume that the samples of the two groups were drawn from the *same* distribution.
2. **Compute the p-value.** Compute the probability that two groups drawn from the same distribution would have a difference in means greater than or equal to the observed difference.
3. **Determine statistical significance.** If the p-value is less than 0.05, this means that it was extremely unlikely that the observed difference occurred under the null hypothesis. The observed result is therefore statistically significant, and we reject the null hypothesis.

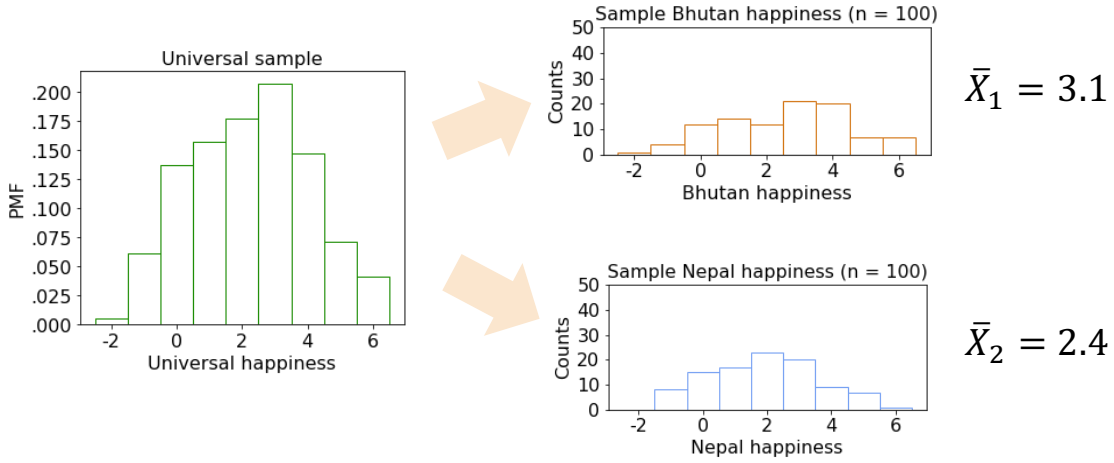


Figure: The null hypothesis assumes that the two observed samples were drawn from some universal distribution. In bootstrapping, we assume the universal distribution is the *universal sample*.

```
def compute_pval(sample1, sample2, niters=10000):
    # make universal sample
    universal_sample = sample1 + sample2

    count = 0
    observed_diff = abs(np.mean(sample1) - np.mean(sample2))
    for i in range(niters):
        # 1. resample
        resample1 = np.random.choice(universal_sample,
                                     len(sample1), replace=True)
        resample2 = np.random.choice(universal_sample,
                                     len(sample2), replace=True)

        # 2. recalculate the statistic
        resample_diff = np.abs(np.mean(resample1) - np.mean(resample2))
```

```
        # c. count how many times the statistic is more extreme
        if resample_diff >= observed_diff:
            count += 1

    return count/niters
```