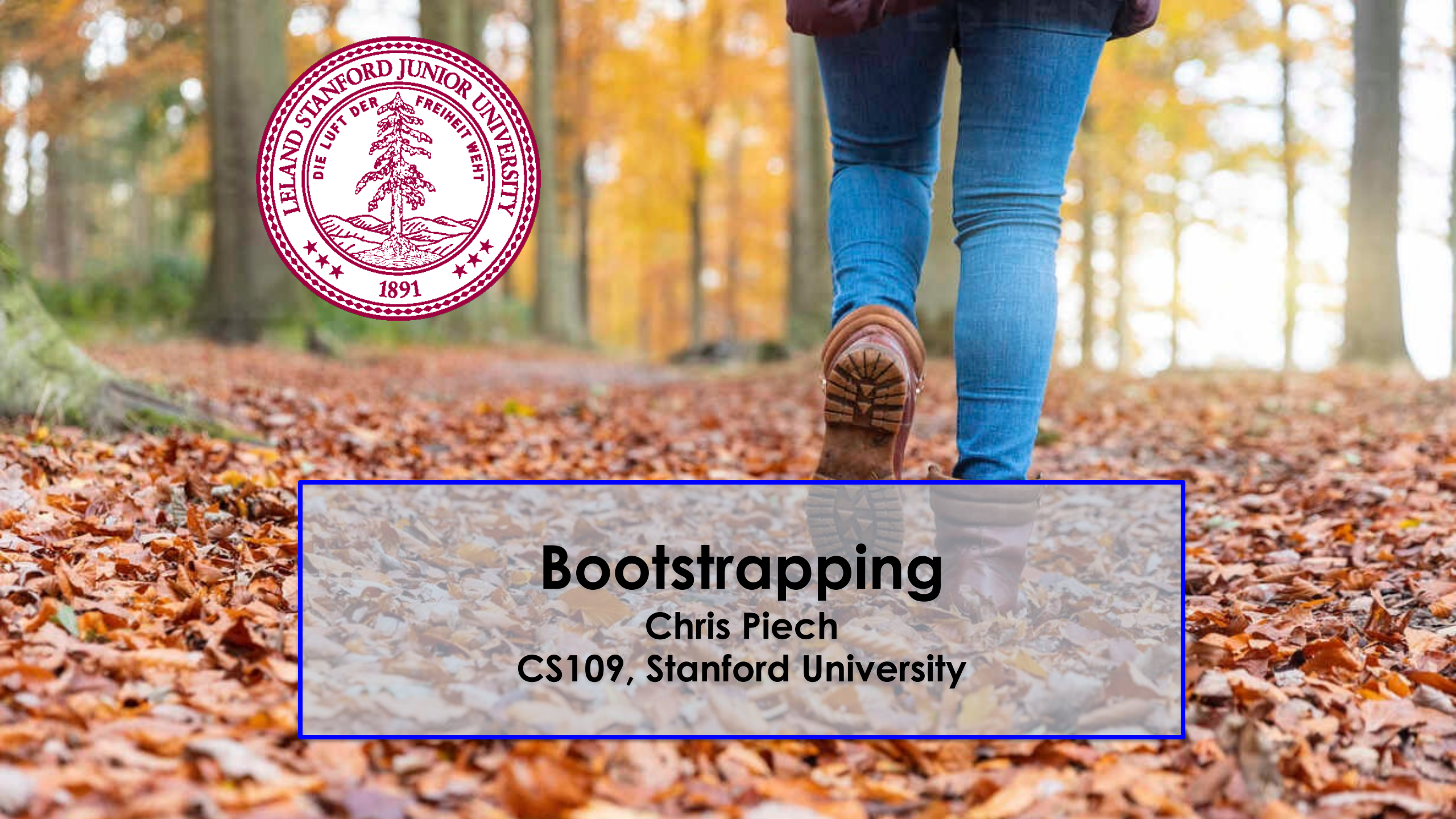


Bootstrapping

Chris Piech
CS109, Stanford University

A real difference?

Learning in Context A	Learning in Context B
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83

18 students

23 students

$\mu_1 = 3.1$

$\mu_2 = 2.4$

Claim: Group 1 and Group 2 are samples from different distributions with a 0.7 difference of means.

How confident are you in this claim?

The Classic Science Test

Group 1	Group 2
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83
$\mu_1 = 3.1$	$\mu_2 = 2.4$

Claim: Group 1 and Group 2 are samples from different distributions with a 0.7 difference of means.

How confident are you in this claim?

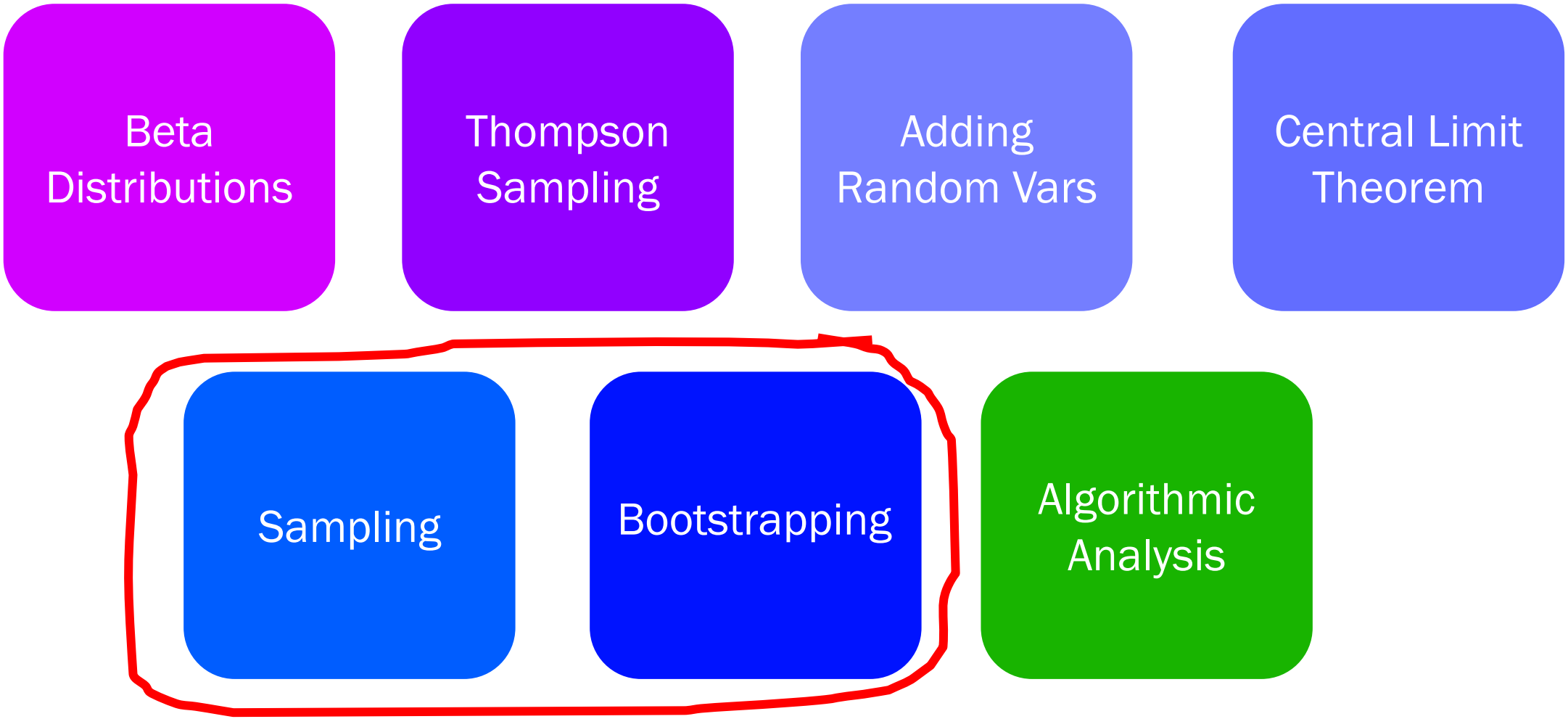
Announcements

TLDR: No class or office hours Monday.

<review>

Where are we in CS109?

Uncertainty Theory



Central Limit Theorem (Summation)

Consider n independent and identically distributed **(i.i.d)** variables $X_1, X_2, \ldots, X_n$ with $E[X_i] = \mu$ and $\text{Var}(X_i) = \sigma^2$.

As $n \to \infty$

$$\sum_{i=1}^{n} X_i \sim \mathcal{N}(n\mu, n\sigma^2)$$

The **sum** of the variables is normally distributed

Central Limit Theorem (Average)

Consider n independent and identically distributed (i.i.d) variables $X_1, X_2, \ldots, X_n$ with $E[X_i] = \mu$ and $\text{Var}(X_i) = \sigma^2$.

$$\text{As } n \to \infty \quad \frac{1}{n}\sum_{i=1}^{n} X_i \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right)$$

The average of the variables is normally distributed

Sampling Statistics

Motivating example

You want to know the true mean and variance of happiness in Bhutan.

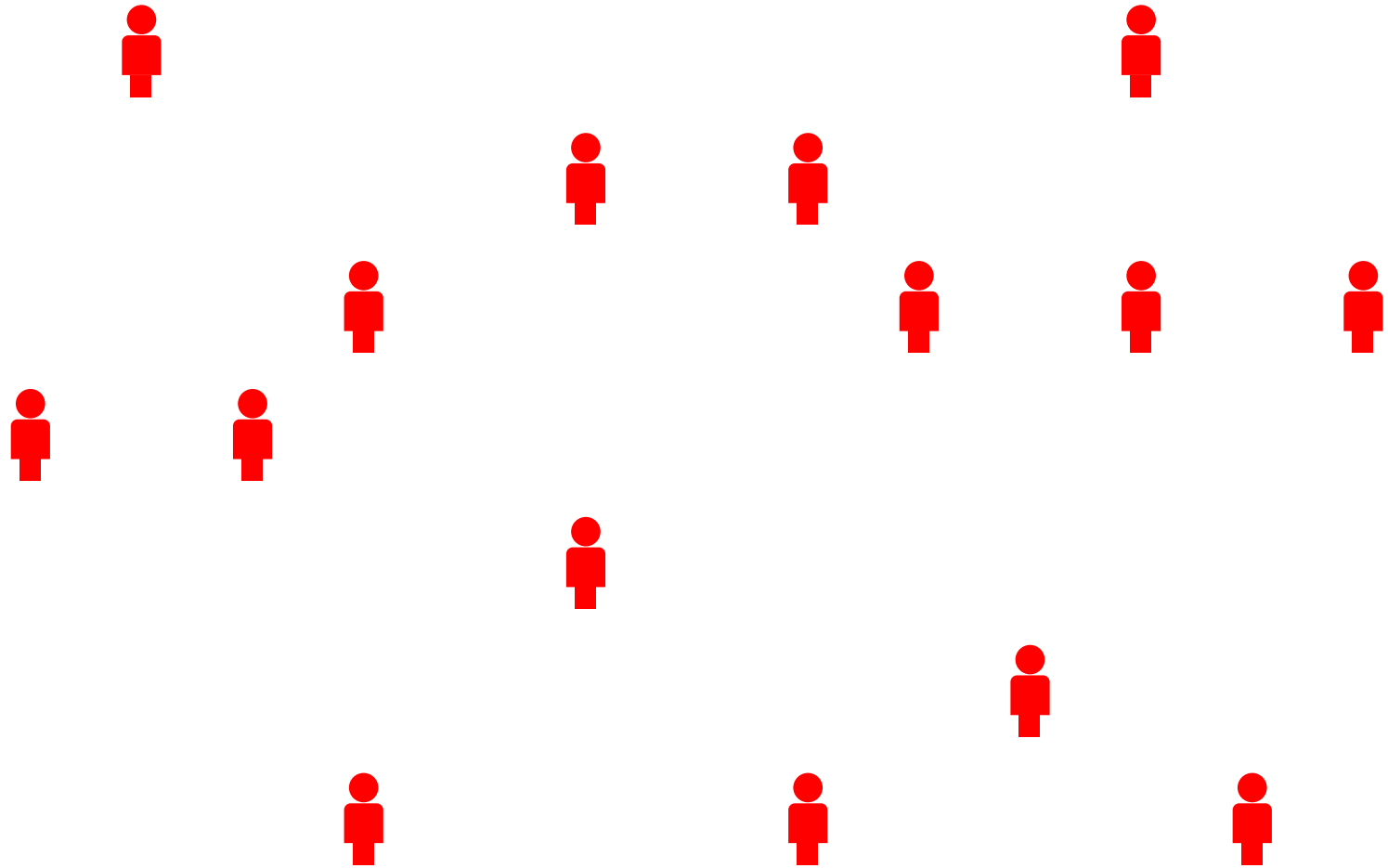
- But you can't ask everyone.
- You poll 200 random people.
- Your data looks like this:

Happiness = {72, 85, 79, 91, 68, ..., 71}

Population

Sample

Sample



Collect one (or more) numbers from each person

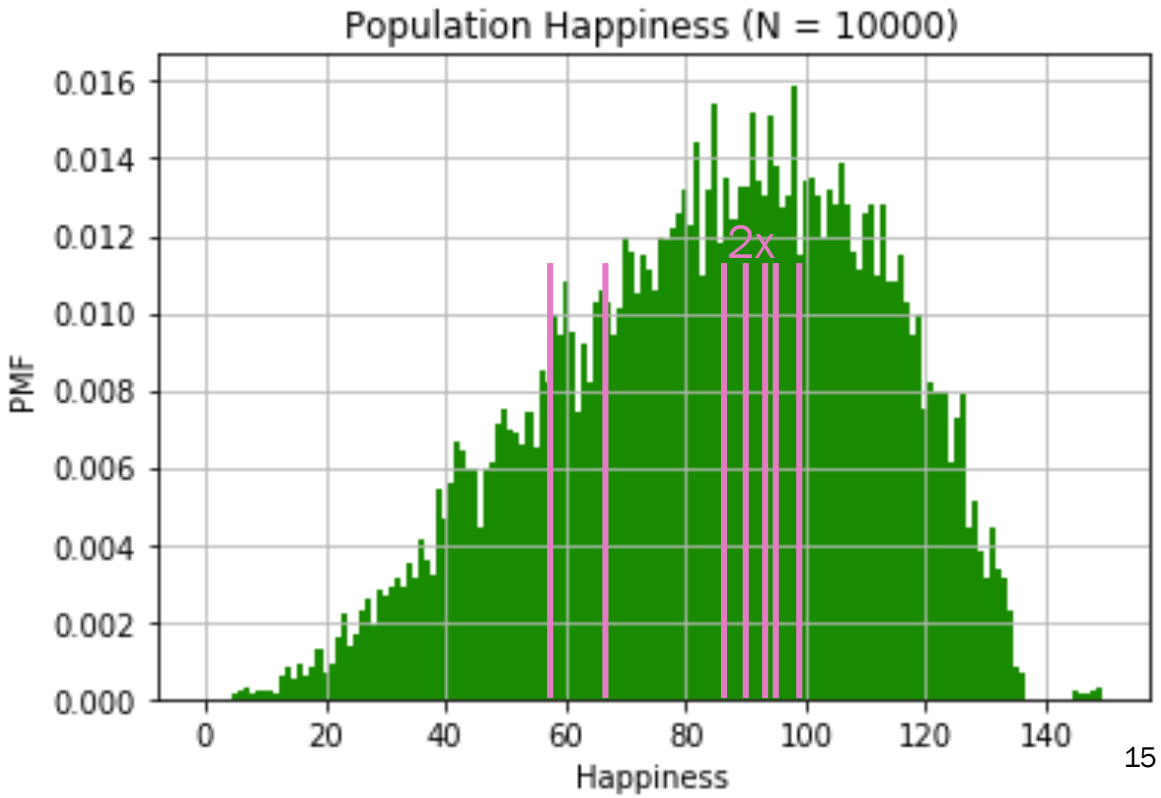
A sample, mathematically

A sample of **sample size** 8:

$$(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)$$

A **realization** of a sample of size 8:

$$(59, 87, 94, 99, 87, 78, 69, 91)$$



Say hello to our new friend: Sample mean

$\bar{X}$

You know I am mean because I have a really *scary* looking hat…

Celebrity look alike:

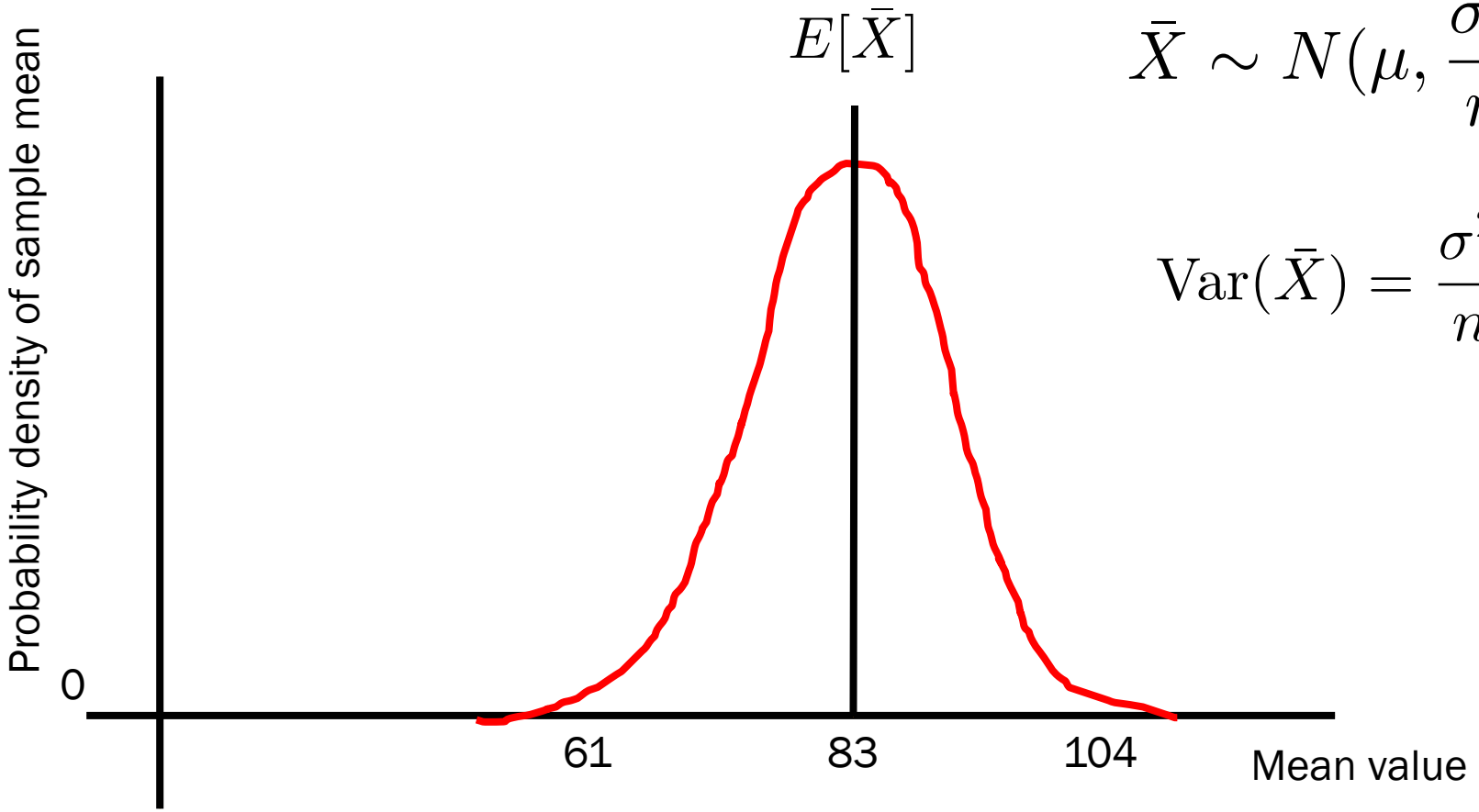
Al Capone

Insight: Sample Mean is an RV with known Var

By central limit theorem:

$$\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$$

$$\text{Var}(\bar{X}) = \frac{\sigma^2}{n}$$



Our Report to Bhutan Government

$$\bar{X} = \frac{1}{n}\sum_i X_i$$

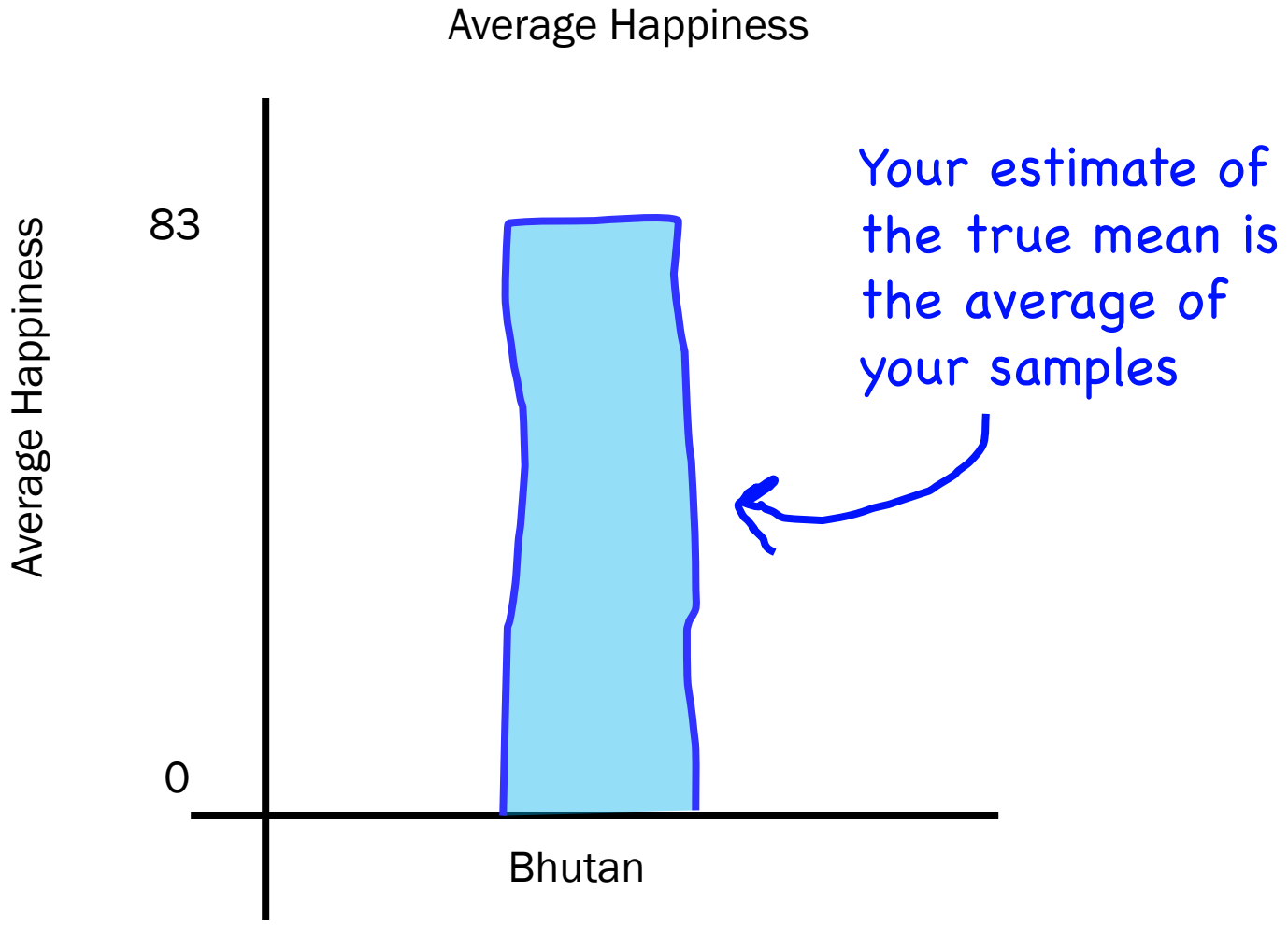
By central limit theorem:

$$\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$$

True variance

True mean

Unbiased estimate: The expected value of the sample mean is the true mean.



</review>

Estimating the population variance

2. Estimate is σ^2, the variance of happiness of Bhutanese people

If we knew the entire population $(x_1, x_2, \ldots, x_N)$:

population mean

population variance $\sigma^2 = E[(X - \mu)^2] = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$

If we only have a sample, $(X_1, X_2, \ldots, X_n)$:

sample mean

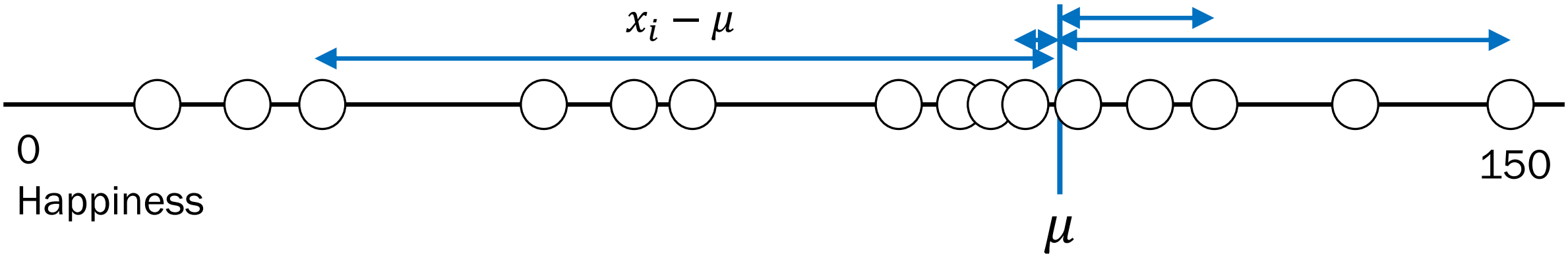
sample variance $E[S^2] = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - E[\bar{X}])^2$

Intuition about the sample variance, S^2

Actual, σ^2

population mean

population variance $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$



Population size, N

Calculating population statistics **exactly** requires us knowing all N datapoints.

Intuition about the sample variance, S^2

Actual, σ^2

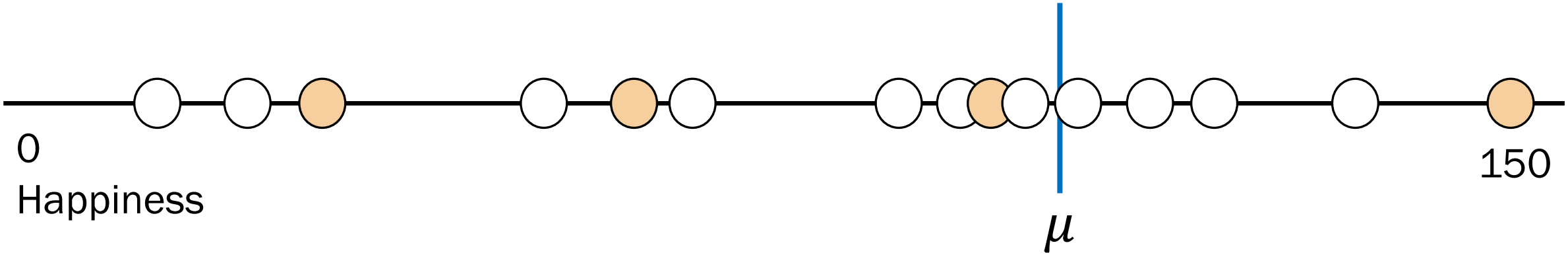
population mean

population variance $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$

Estimate, S^2

sample mean

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



Population size, N

Intuition about the sample variance, S^2

Actual, σ^2

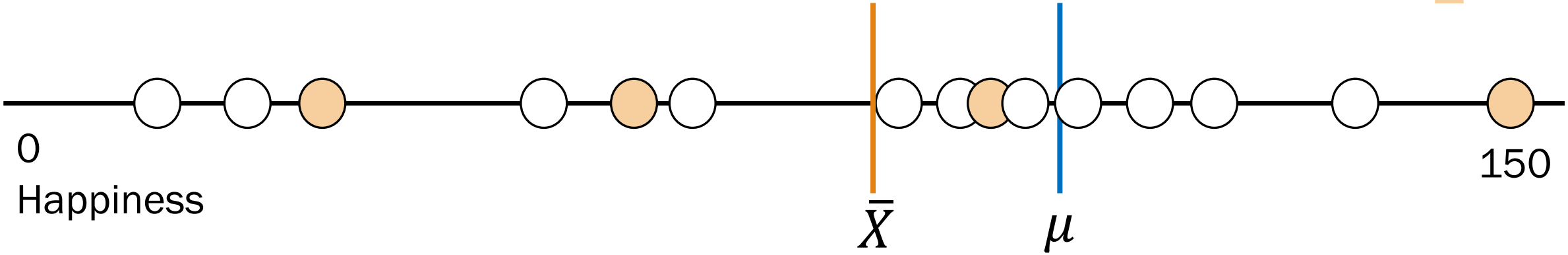
population mean

population variance $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$

Estimate, S^2

sample mean

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



Population size, N

Intuition about the sample variance, S^2

Actual, σ^2

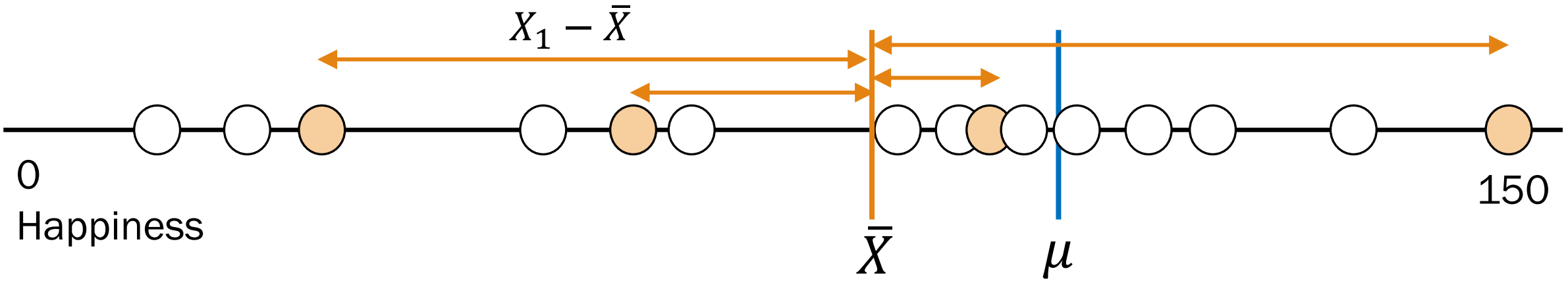
population mean

population variance $\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2$

Estimate, S^2

sample mean

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



Population size, N

Sample variance is an estimate using an estimate, so it needs additional scaling.

Proof that S^2 is unbiased (just for reference)

$$E[S^2] = \sigma^2$$

$$E[S^2] = E\left[\frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2\right] \quad \Rightarrow \quad (n-1)E[S^2] = E\left[\sum_{i=1}^{n}(X_i - \bar{X})^2\right]$$

$$(n-1)E[S^2] = E\left[\sum_{i=1}^{n}\big((X_i - \mu) + (\mu - \bar{X})\big)^2\right] \qquad \text{(introduce } \mu - \mu\text{)}$$

$$= E\left[\sum_{i=1}^{n}(X_i - \mu)^2 + \sum_{i=1}^{n}(\mu - \bar{X})^2 + 2\sum_{i=1}^{n}(X_i - \mu)(\mu - \bar{X})\right]$$

$$= E\left[\sum_{i=1}^{n}(X_i - \mu)^2 + n(\mu - \bar{X})^2 - 2n(\mu - \bar{X})^2\right]$$

$$= E\left[\sum_{i=1}^{n}(X_i - \mu)^2 - n(\mu - \bar{X})^2\right] = \sum_{i=1}^{n} E[(X_i - \mu)]^2 - nE[(\bar{X} - \mu)^2]$$

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

$$2(\mu - \bar{X})\left(\sum_{i=1}^{n} X_i - n\mu\right)$$

$$2(\mu - \bar{X})n(\bar{X} - \mu)$$

$$-2n(\mu - \bar{X})^2$$

$$= n\sigma^2 - n\text{Var}(\bar{X}) = n\sigma^2 - n\frac{\sigma^2}{n} = n\sigma^2 - \sigma^2 = (n-1)\sigma^2$$

Therefore $E[S^2] = \sigma^2$

Estimating the population variance

2. What is σ^2, the variance of happiness of Bhutanese people?

If we only have a sample, $(X_1, X_2, \ldots, X_n)$:

The best estimate of σ^2 is the expectation of **sample variance**:

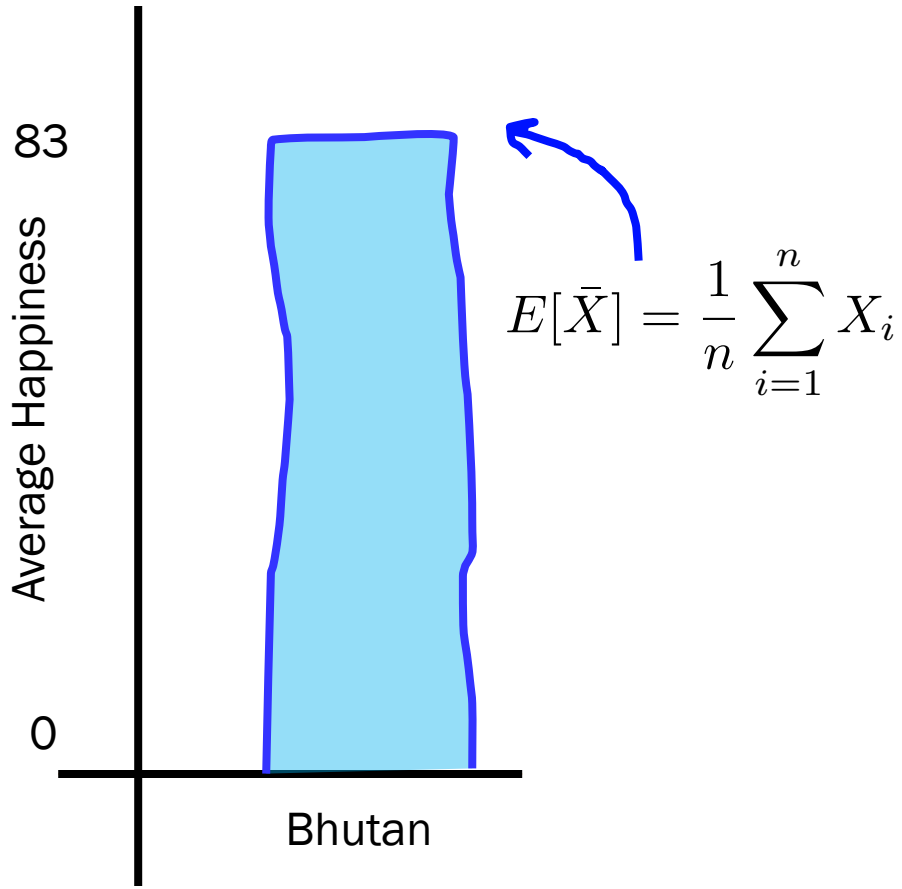
$$E[S^2] = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - E[\bar{X}])^2$$

S^2 is an **unbiased estimator** of the population variance, σ^2. $E[S^2] = \sigma^2$

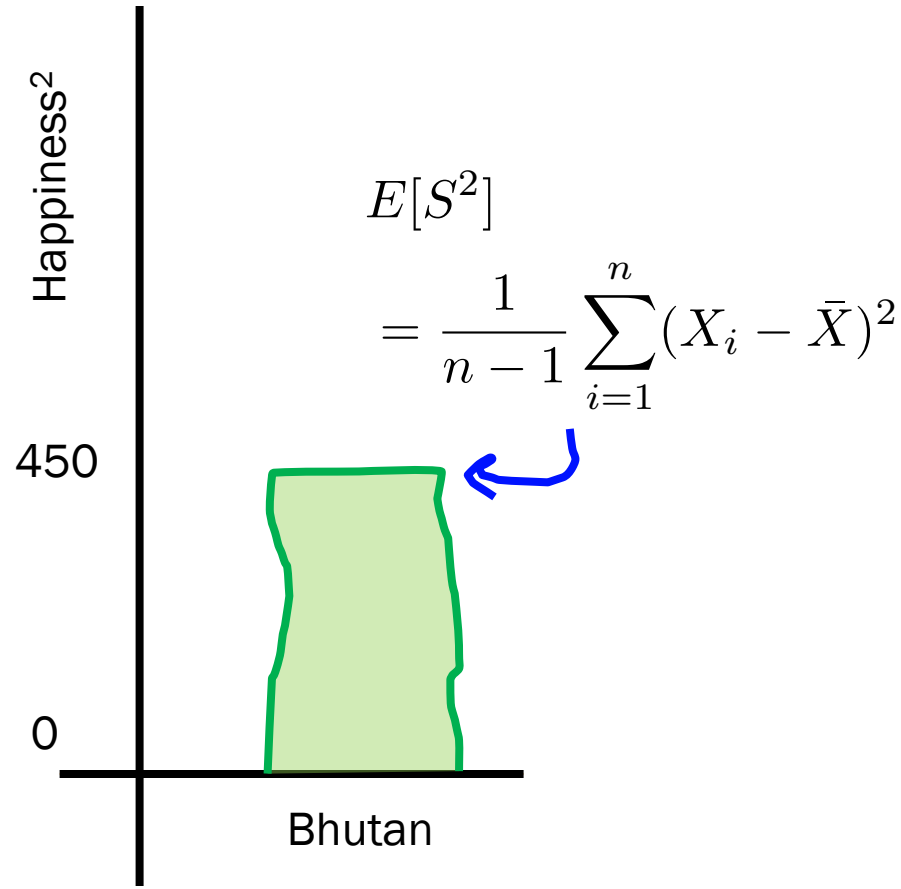
Our Report to Bhutan Government

By CLT: $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$

Average Happiness



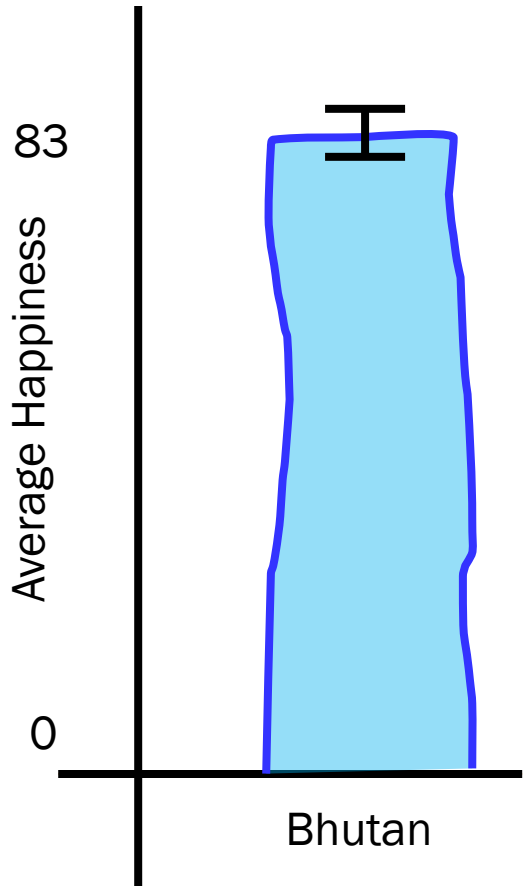
Variance of Happiness



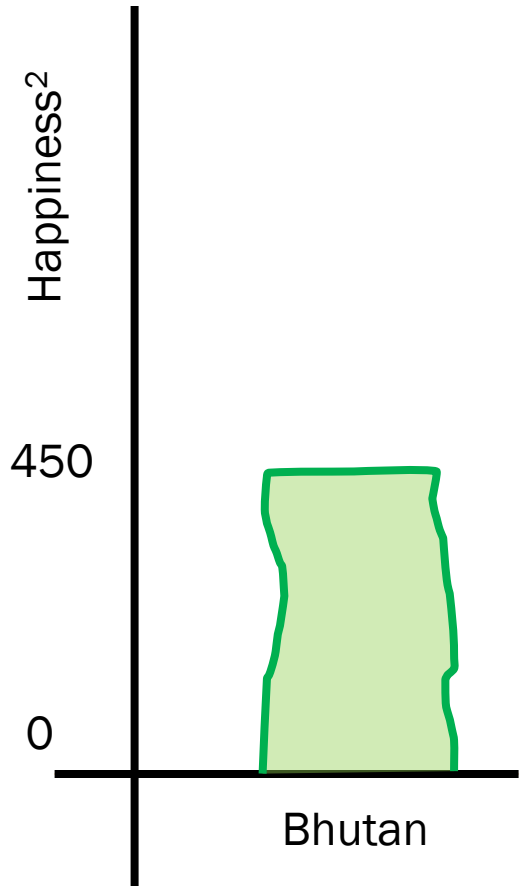
But what about **error bars**???

By CLT: $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$

Average Happiness



Variance of Happiness

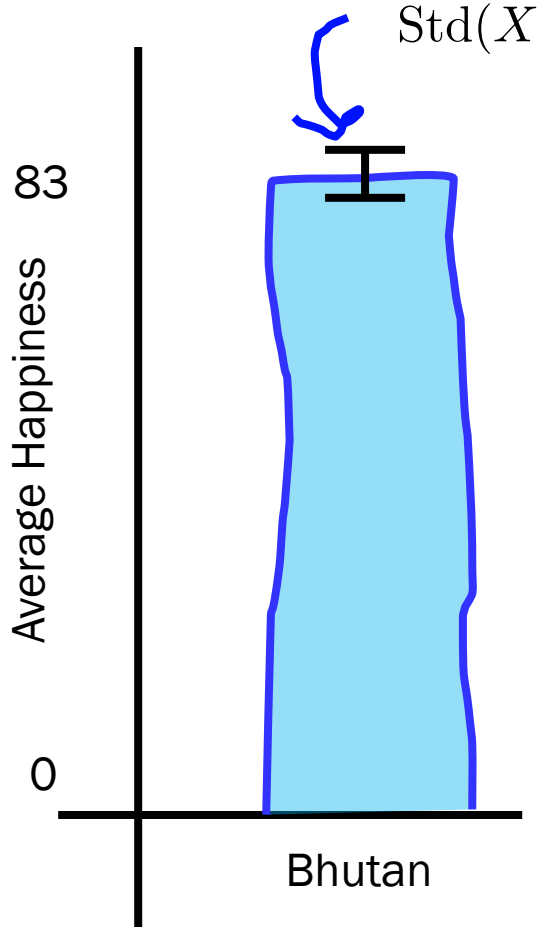


But what about **error bars**???

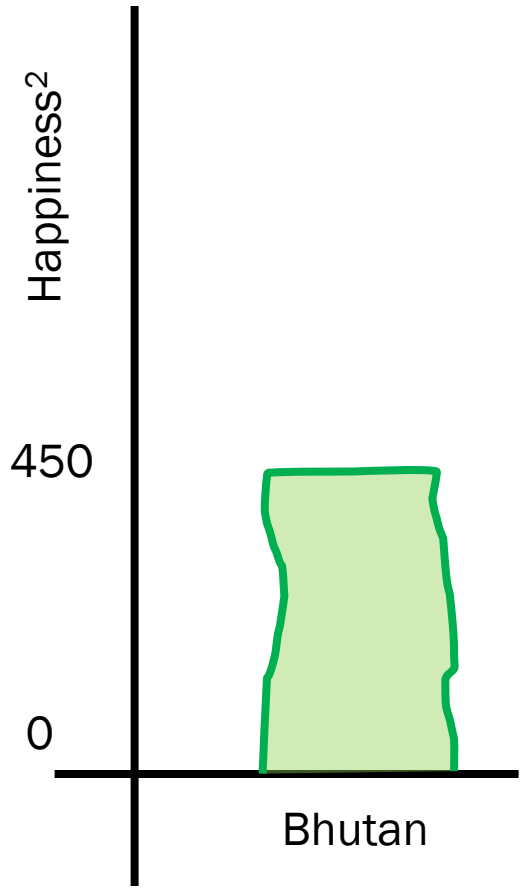
By CLT: $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$

Average Happiness

$$\text{Std}(\bar{X}) = \sqrt{\frac{E[S^2]}{n}}$$



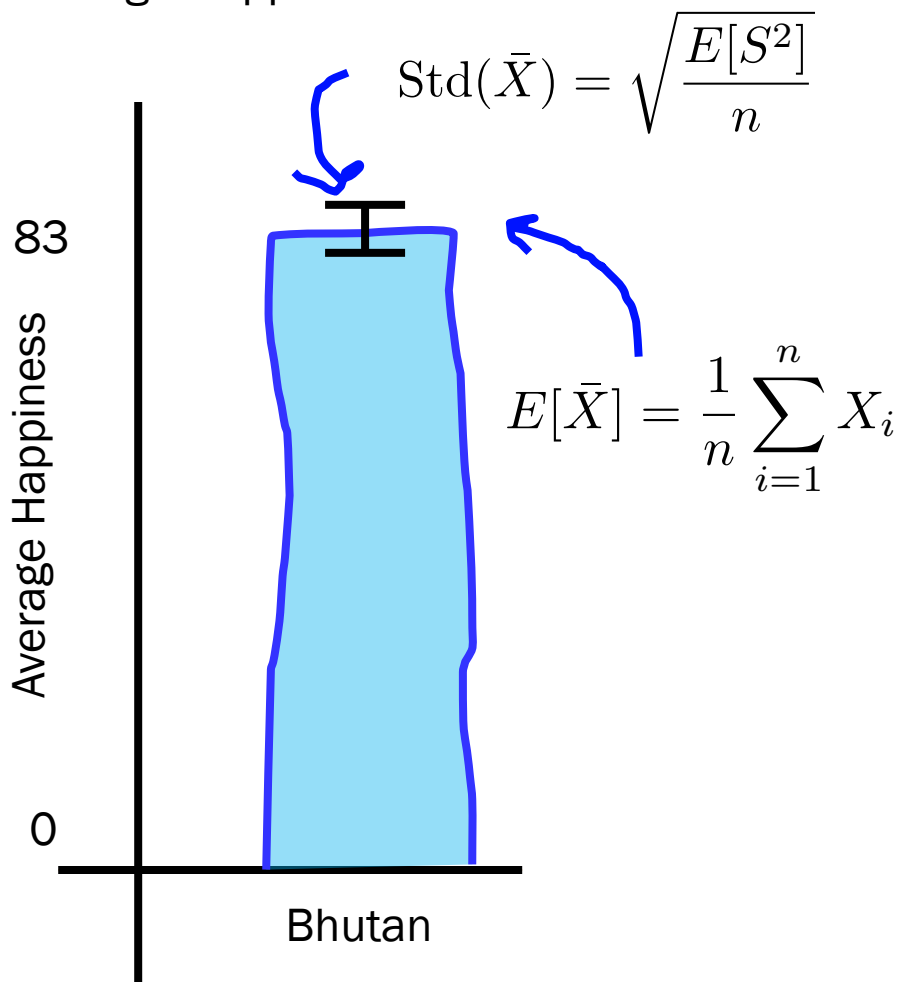
Variance of Happiness



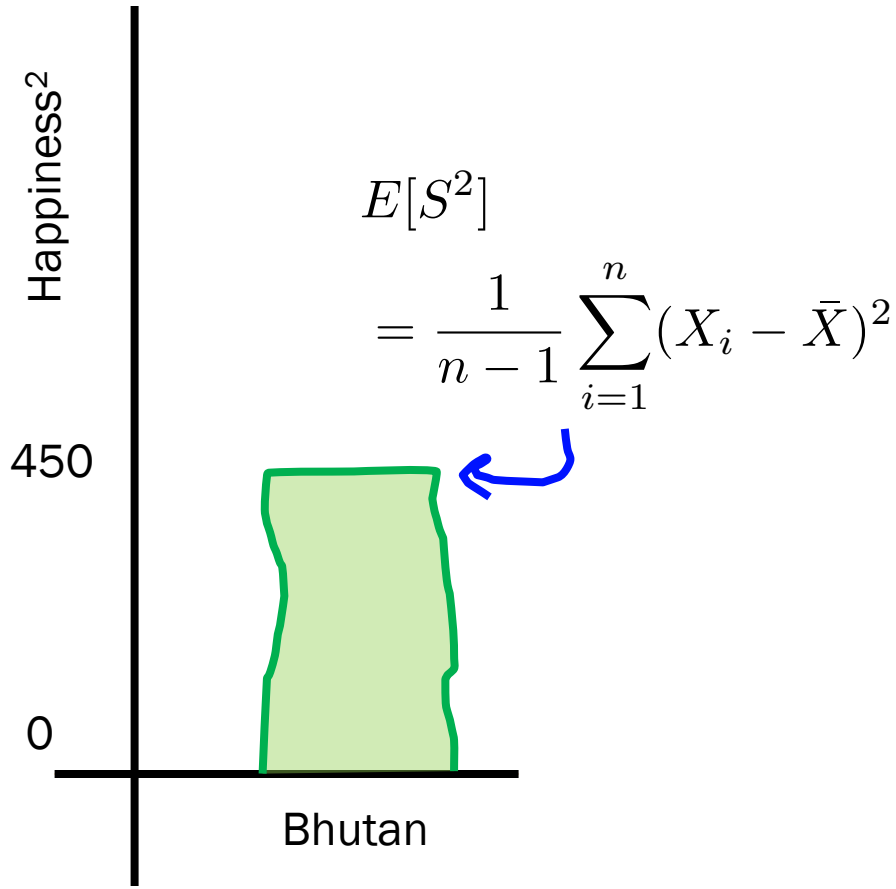
Our Report to Bhutan Government

By CLT: $\bar{X} \sim N(\mu, \frac{\sigma^2}{n})$

Average Happiness



Variance of Happiness



Equations we used to get those values

sample mean estimate

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

Our best guess at the true mean

sample variance estimate

sample mean estimate

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

Our best guess at the true variance

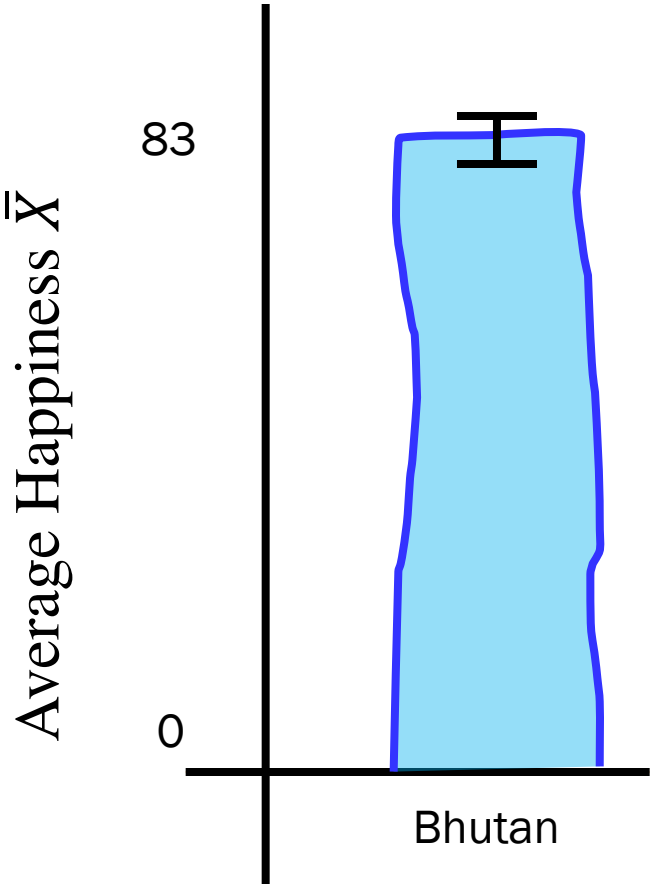
Std error of the mean estimate

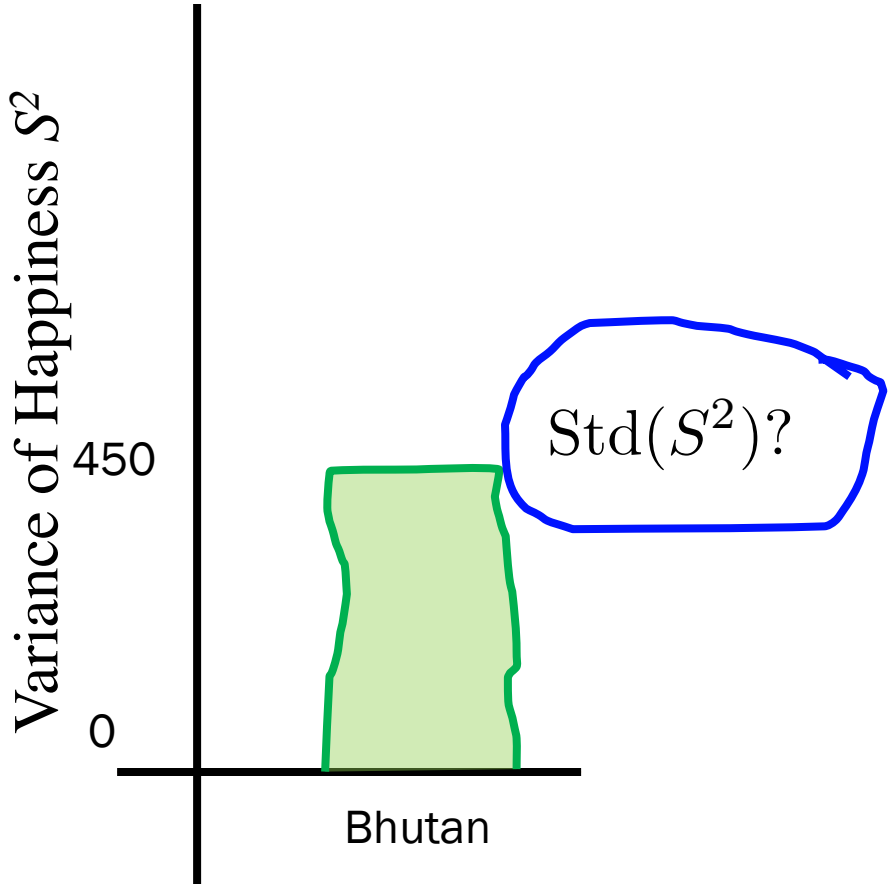
$$\text{Std}(\bar{X}) = \sqrt{\frac{E[S^2]}{n}}$$

sample variance

How wrong do we think our mean estimate is?

Our Report to Bhutan Government





Claim: The average happiness of Bhutan is 83 ± 2

How long does it take CS109 students to complete Psets?

PSet Timings Raw


```
{"pset1-countingcards": [1550.1940480000003, 212.4435679999999, 1722.7378399999996, 531.5406079999998,
2674.1259839999993, 2401.314592, 4181.617599999996, 2420.824192, 276.2297759999999, 1140.
0784479999998, 1206.8157760000001, 560.5036640000001, 931.507056, 1156.8344, 798.4586879999999, 384.
28702399999986, 1050.34776, 1878.2590080000002, 1292.9448640000003, 431.07478400000014, 378.
8209120000001, 818.1577920000002, 2555.404016, 882.420272, 640.076624, 1529.9575360000006, 1232.
9577920000004, 1113.0007039999998, 1846.7954240000001, 1800.8698879999997, 2143.1468569999993, 3184.
684112000002, 1734.1895839999993, 1246.261760000001, 1660.198096000001, 741.8999519999999, 1603.
1233440000017, 710.5813119999996, 5379.367696, 988.8105760000003, 1478.5483519999993, 722.
7267519999994, 578.3234239999999, 518.659968, 1975.309824, 2861.4804320000003, 1358.5260480000002,
1463.927296, 1591.5097599999995, 2572.171295999997, 1580.9689440000004, 2030.1685279999992, 12214.
882255999995, 412.23537600000003, 335.2142080000001, 602.1413119999997, 1871.3165920000006, 1182.
3381600000002, 876.0278399999994, 2358.0574400000014, 627.9507039999999, 1937.6730879999996, 1463.
6703679999996, 478.5180959999999, 1347.6753760000001, 772.8854079999996, 1270.234272, 2764.
7578559999965, 524.3688640000001, 1045.8361919999998, 2220.536159999998, 2045.8206400000013, 2413.
5207999999984, 984.7854719999997, 1006.5262400000004, 869.2664479999995, 745.0748, 0, 385.
2566720000001, 2070.540032, 397.9075039999999, 1666.6341280000004, 751.9160320000001, 2015.
9539999999997, 599.8172480000001, 859.2658560000001, 231.6490880000001, 569.3155360000002, 3276.
267408000002, 3839.6596800000016, 741.7451360000001, 1213.9471680000001, 1824.1588639999995, 586.
0078240000003, 1384.141408, 2077.4697120000005, 1078.9446719999994, 362.7789919999999, 1451.473024,
1229.9791519999997, 388.71222399999994, 261.2879679999998, 870.9573119999999, 1696.6648480000013, 802.
0010240000005, 646.2847200000003, 598.3204320000007, 606.9591680000001, 2392.2958079999985, 508.
41139200000015, 981.5174560000006, 837.928704, 597.6606559999999, 2044.2889280000004, 2387.
9005119999997, 2634.773776000002, 211.579952, 2220.8885120000027, 522.773296, 817.3810079999996, 1136.
7684479999998, 1544.7102240000008, 1365.8311839999997, 883.575952, 448.29513599999984, 3772.
1695199999995, 800.5720319999994, 603.3570560000002, 4195.264224000001, 96.61473600000001, 393.
8498559999999, 533.6176800000002, 325.44255999999984, 706.129216, 2977.463391999999, 3156.
3286400000006, 1732.352831999999, 1258.0552000000002, 214.45435200000009, 862.0855199999994, 2040.
5987840000003, 992.4715680000003, 1434.3783199999996, 1408.3342879999984, 1447.6861600000002, 1884.
4831039999988, 2010.586384000001, 798.7439360000008, 491.58987200000024, 268.505664, 1778.
2884160000015, 1060.6238560000006, 259.94641600000006, 2381.6392799999985, 500.35900800000013, 405.
9288, 1949.8160640000012, 1476.9774559999992, 1552.3747040000007, 504.5615200000002, 1510.
8626239999999, 1034.3021600000004, 3131.245696000002, 1577.3348639999995, 1226.4924480000002, 911.
3914399999999, 234.32780800000006, 1005.4795199999999, 1360.6999999999994, 989.1745439999997, 1390.
004112, 739.4614079999991, 454.34560000000016, 2438.610176000001, 1758.4230880000005, 858.
0467199999998, 462.3268800000001, 227.134608, 569.5653920000002, 859.4128159999997, 663.267392, 778.
0543839999998, 1381.3501920000003, 10107.814288000001, 1320.0240319999998, 1885.6292959999998, 1382.
9131199999993, 430.007232, 457.6967359999998, 1322.0688800000012, 594.5272960000001, 1302.
5745280000006, 427.92484799999994, 762.2496479999998, 280.673488, 2715.640207999996, 1038.
```


For each pset question:
I have a list of how many seconds it took each person

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

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

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

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

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

$$\text{Std}(\bar{X}) = \sqrt{\frac{E[S^2]}{n}}$$

PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

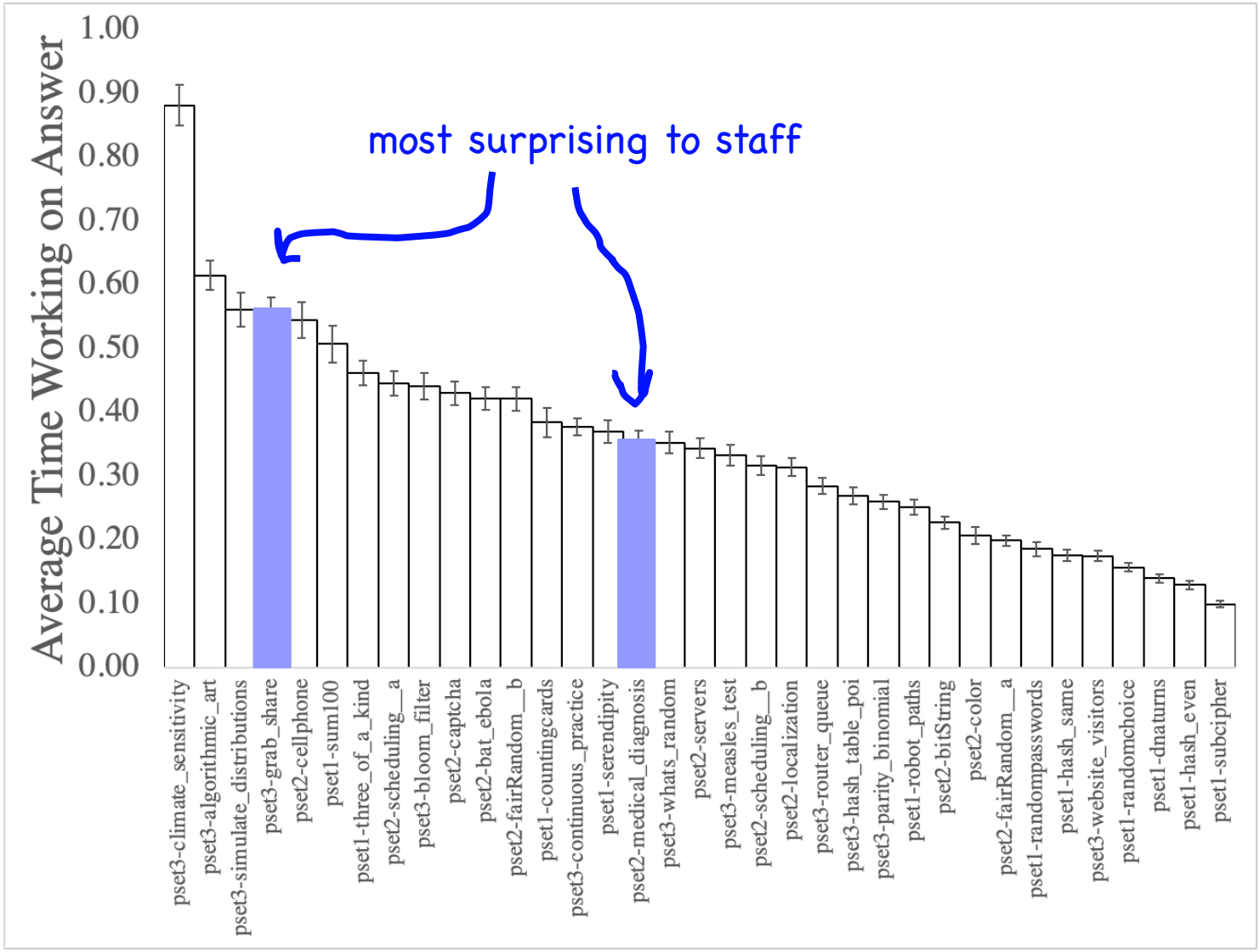
PSet Timings Stats Code

```
def analyse(data):
    for question_key, timings_list in data.items():
        timings_list = remove_zeros(timings_list)
        # calculate n
        n = len(timings_list)
        # estimate the mean
        sample_mean = np.mean(timings_list)
        # estimate the variance (ddof=1 says its a sample... i know...)
        sample_var = np.var(timings_list,ddof=1)
        # calculate the standard error of the mean
        standard_err = math.sqrt(sample_var / n)
        # sample std
        sample_std = math.sqrt(sample_var)
        # print them out
        display_name = question_key[:12]
        print(f'{display_name},\tmean: {sample_mean:.1f} ± {standard_err:.1f},\tstd:
              {sample_std:.1f}')
```

PSet Timings Stats Output

```
pset1-counti,   mean: 1385.0 ± 83.8,    std: 1288.0
pset1-dnatur,   mean: 504.2 ± 25.4,     std: 390.3
pset1-hash_e,   mean: 467.3 ± 22.0,     std: 336.4
pset1-hash_s,   mean: 634.0 ± 30.7,     std: 471.4
pset1-random,   mean: 567.5 ± 23.3,     std: 357.8
pset1-random,   mean: 671.4 ± 39.6,     std: 606.3
pset1-robot_,   mean: 906.0 ± 42.2,     std: 648.6
pset1-serend,   mean: 1335.3 ± 63.9,    std: 978.0
pset1-subcip,   mean: 359.8 ± 19.9,     std: 304.8
pset1-sum100,   mean: 1827.2 ± 106.2,   std: 1627.3
pset1-three_,   mean: 1663.9 ± 70.2,    std: 1078.7
pset2-bat_eb,   mean: 1520.1 ± 62.3,    std: 942.3
pset2-bitStr,   mean: 819.2 ± 34.0,     std: 523.6
pset2-captch,   mean: 1551.0 ± 67.4,    std: 1030.6
pset2-cellph,   mean: 1962.0 ± 102.7,   std: 1567.3
pset2-color,    mean: 745.1 ± 48.3,     std: 742.9
pset2-fairRa,   mean: 717.5 ± 27.2,     std: 418.8
pset2-fairRa,   mean: 1518.2 ± 67.6,    std: 1035.6
pset2-locali,   mean: 1132.4 ± 49.1,    std: 750.6
pset2-medica,   mean: 1275.3 ± 64.2,    std: 981.5
pset2-schedu,   mean: 1606.8 ± 70.1,    std: 1079.0
```

Sampling statistics on your Psets



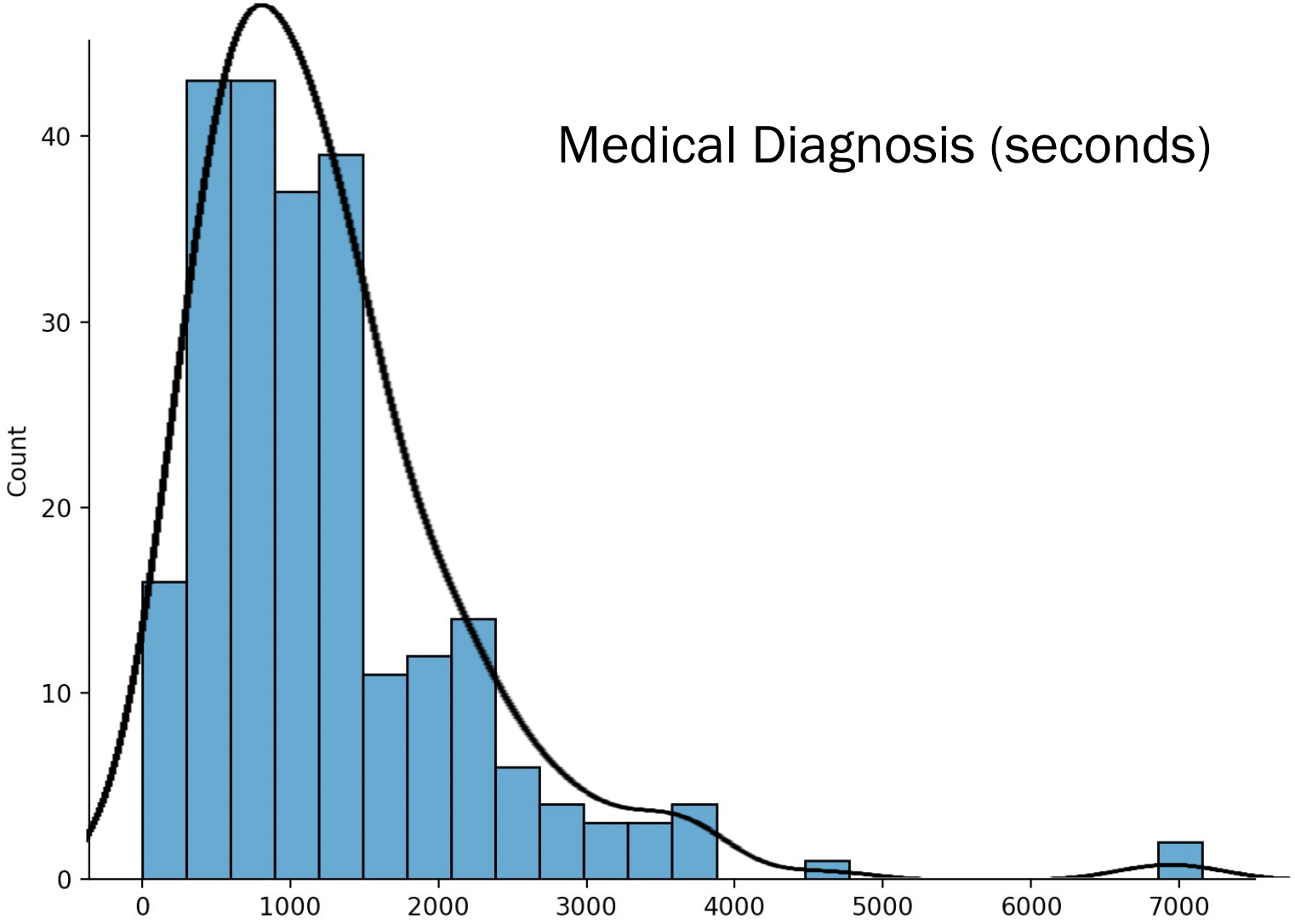
Error bars are standard error of the mean

Expectation of the sum of problems is sum of expectations:

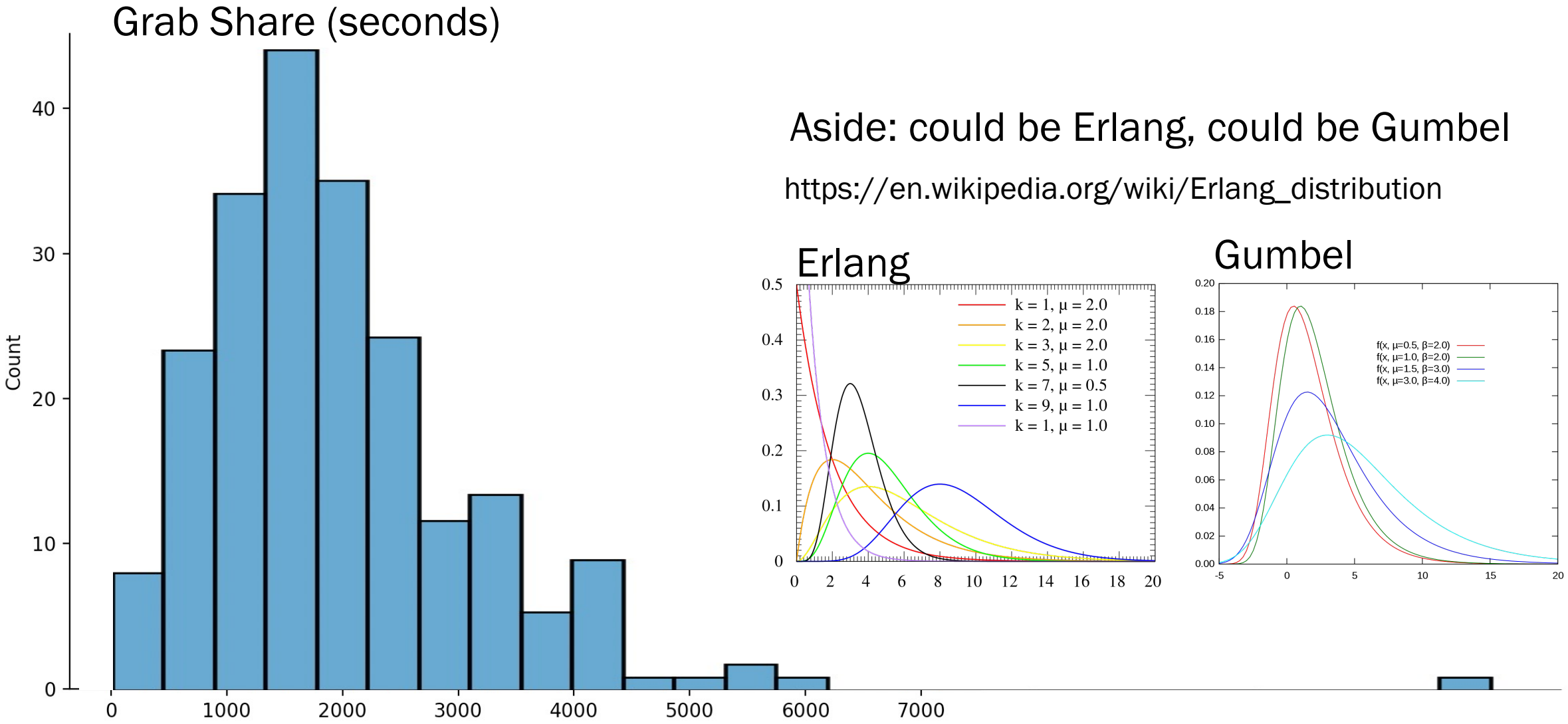
pset1: 2.87 hours on answers
pset2: 4.23 hours on answers
pset3: 5.11 hours on answers

Total: 12.1 hours on answers
Budget: 50 hours for psets

Distribution of PSet Completion Times: Medical Diagnosis



Distribution of PSet Completion Times: Grab Share



Aside: could be Erlang, could be Gumbel

https://en.wikipedia.org/wiki/Erlang_distribution

Erlang

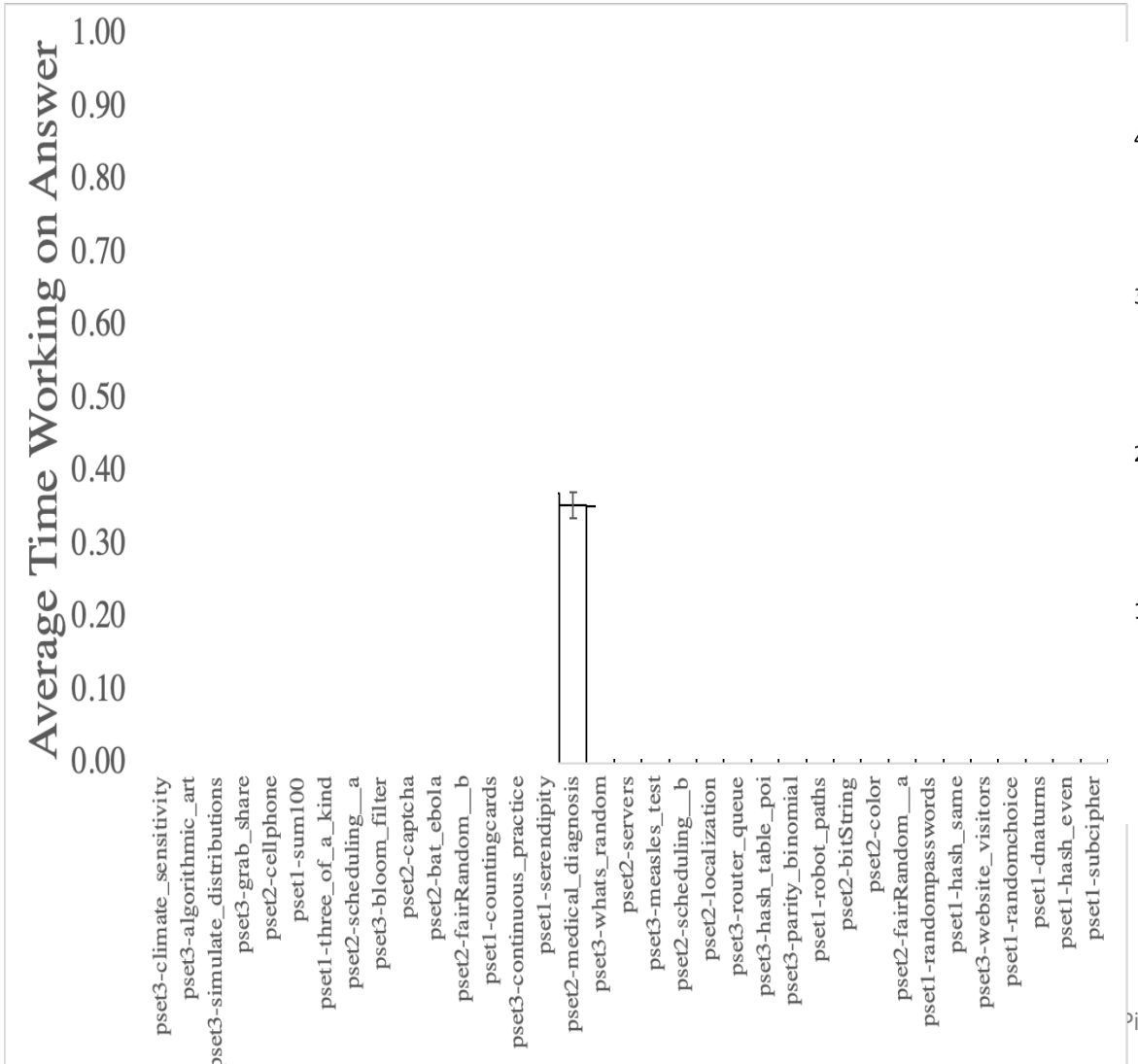
Gumbel

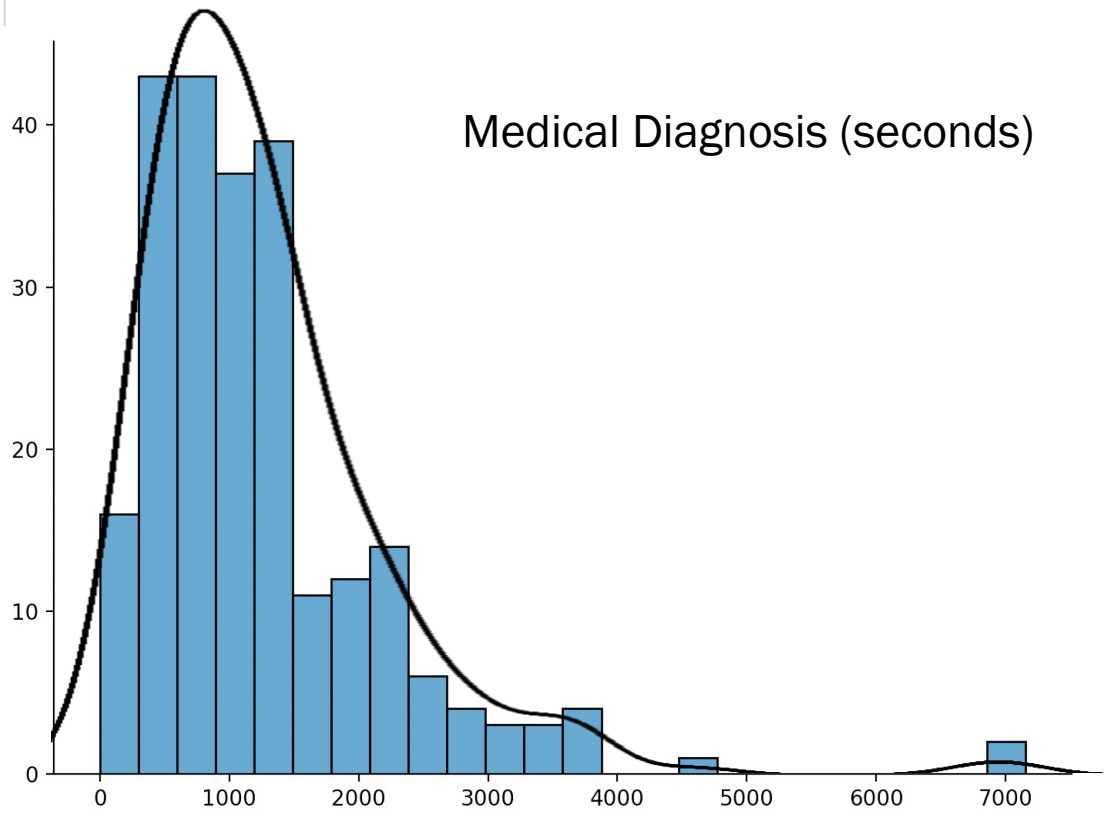
Statistics Vs Distribution

Sampling statistics

vs

Sampling distribution





Questions???

Bootstraping

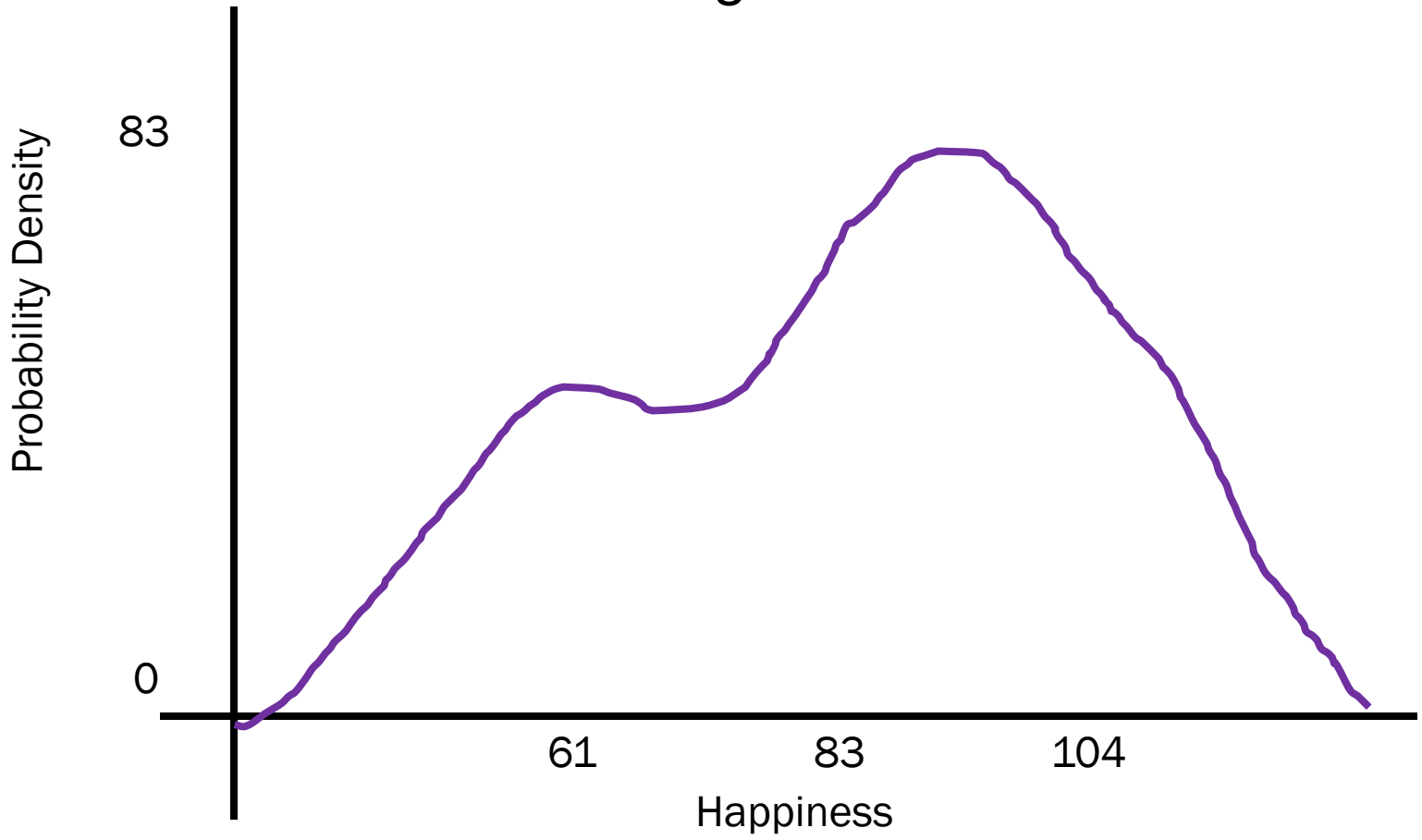
Bootstrap:
Probability for Computer Scientists

Bootstraping allows you to:

- Know the **distribution of *statistics***
- Calculate **p values**

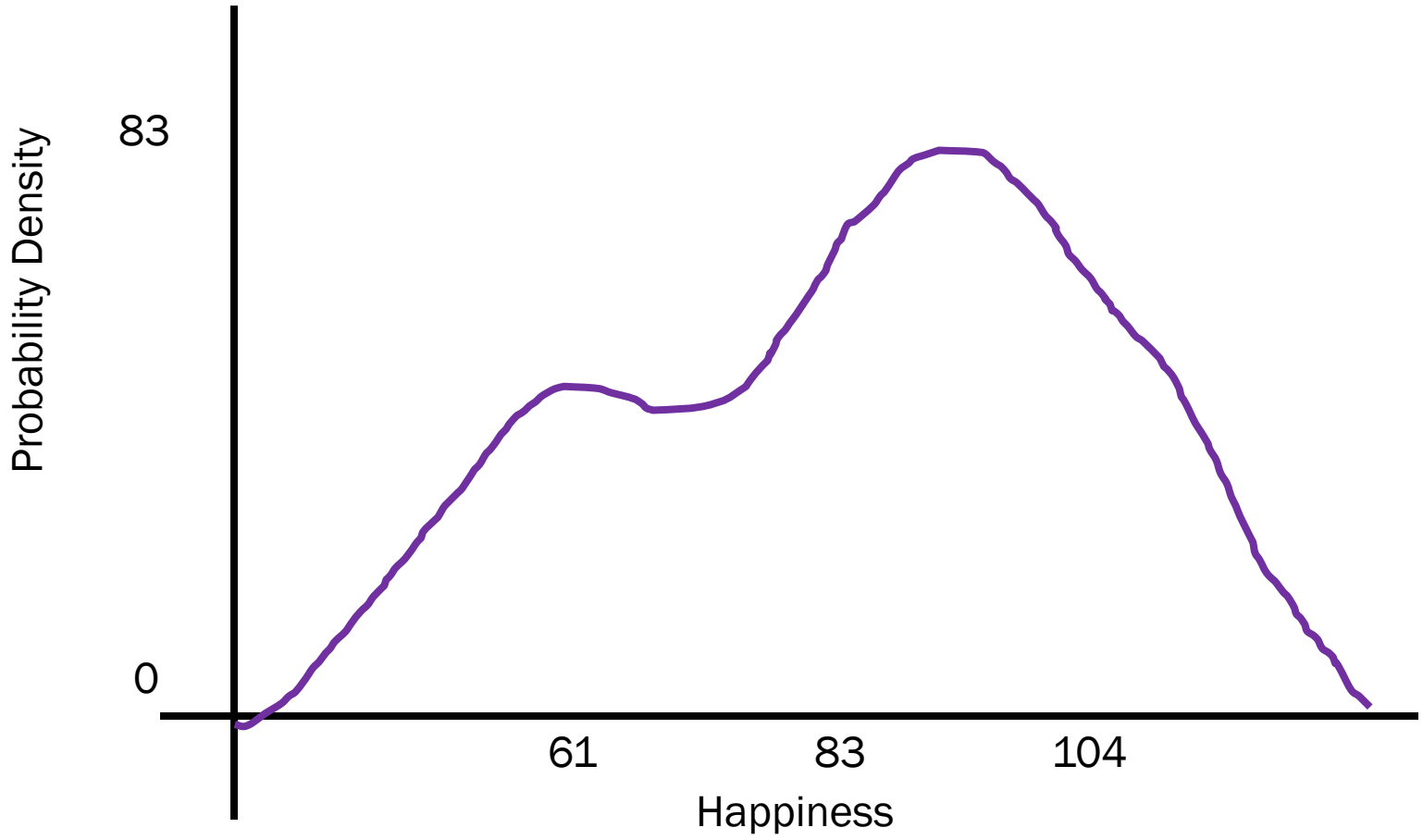
Hypothetical

What is the probability that the **mean** of a sample of 200 people is within the range 81 to 85?



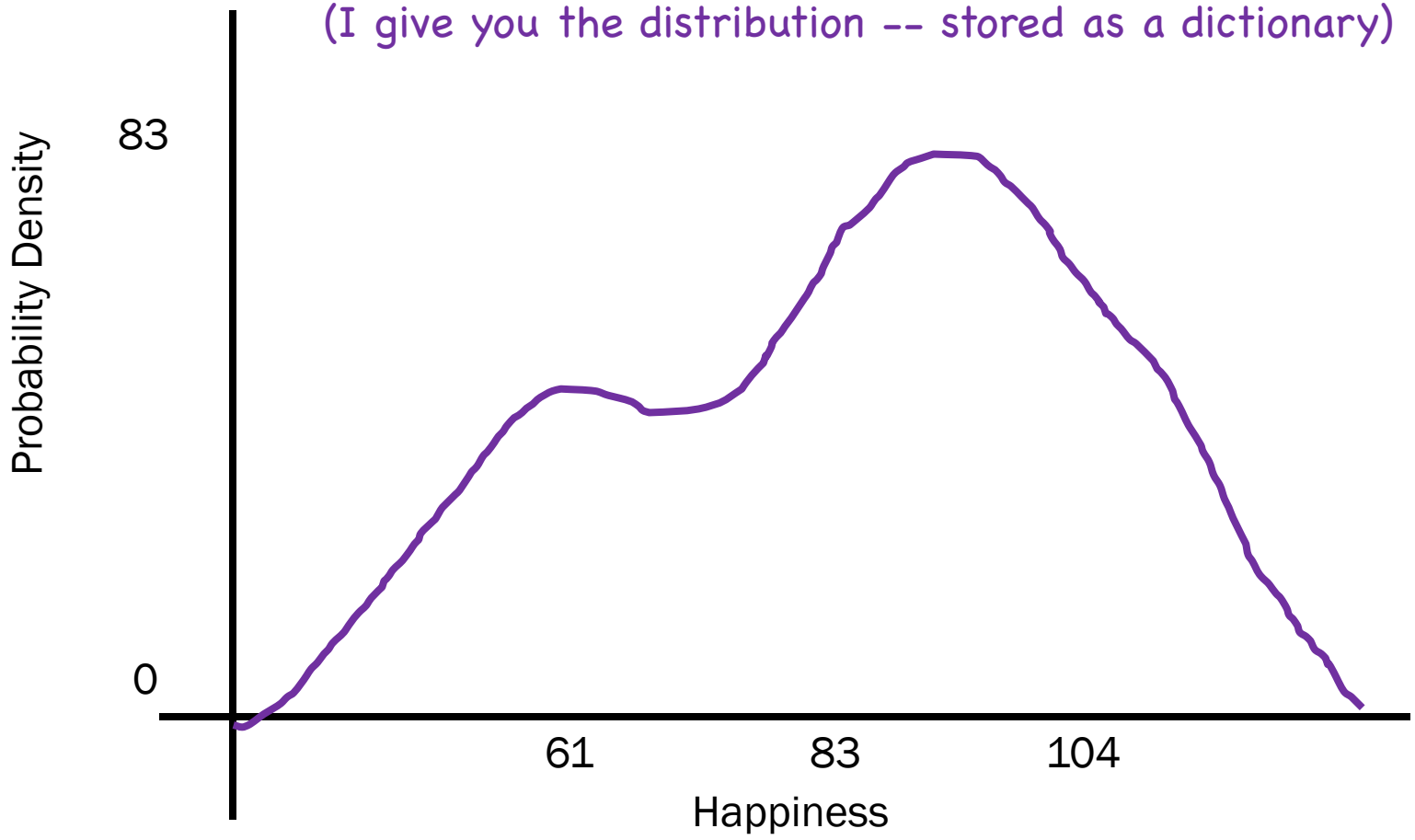
Hypothetical

What is the **std** of the **sample variance**, calculated from 200 people?



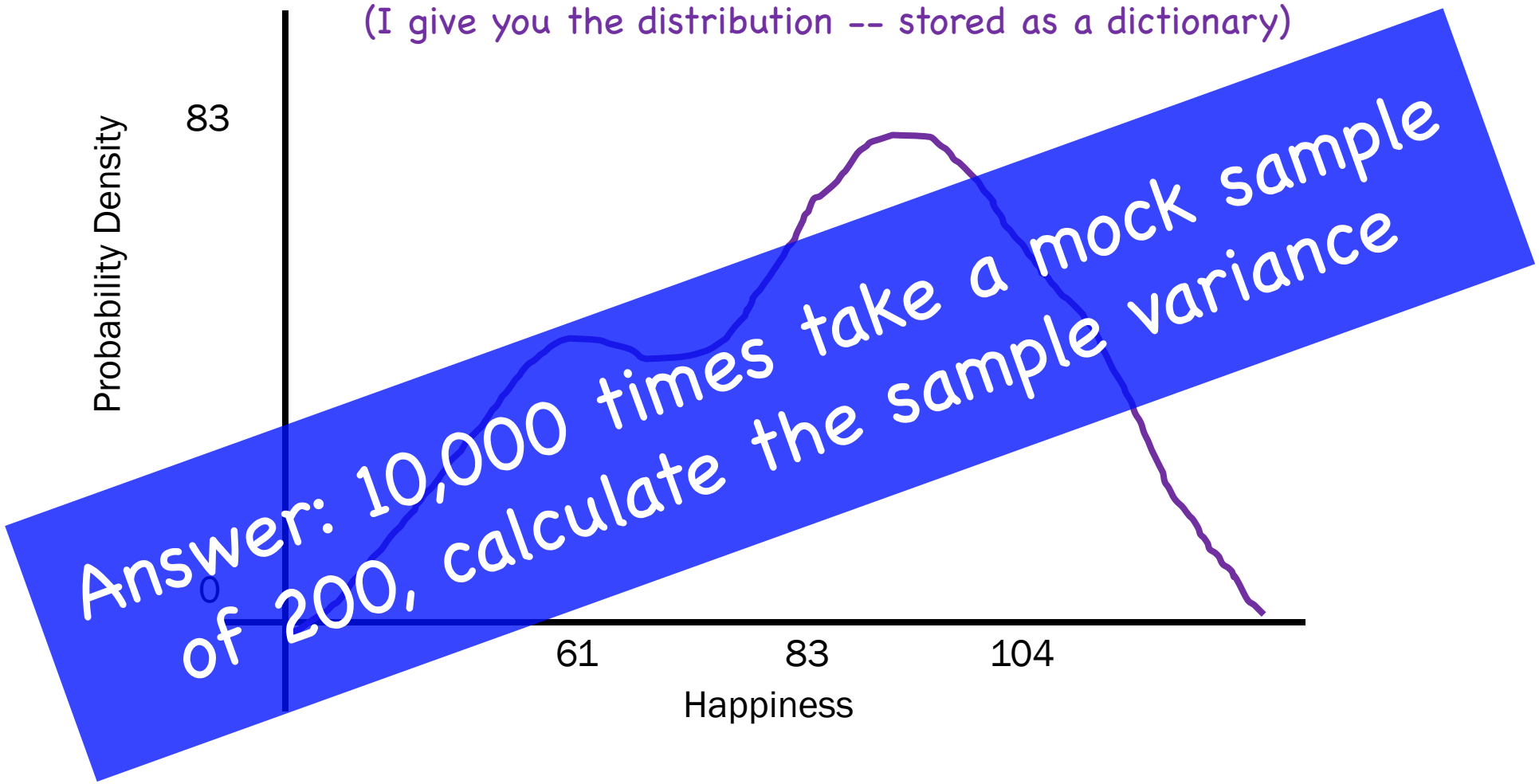
If I Gave You the True Distribution, what would you do?

What is the **std** of the **sample variance**, calculated from 200 people?



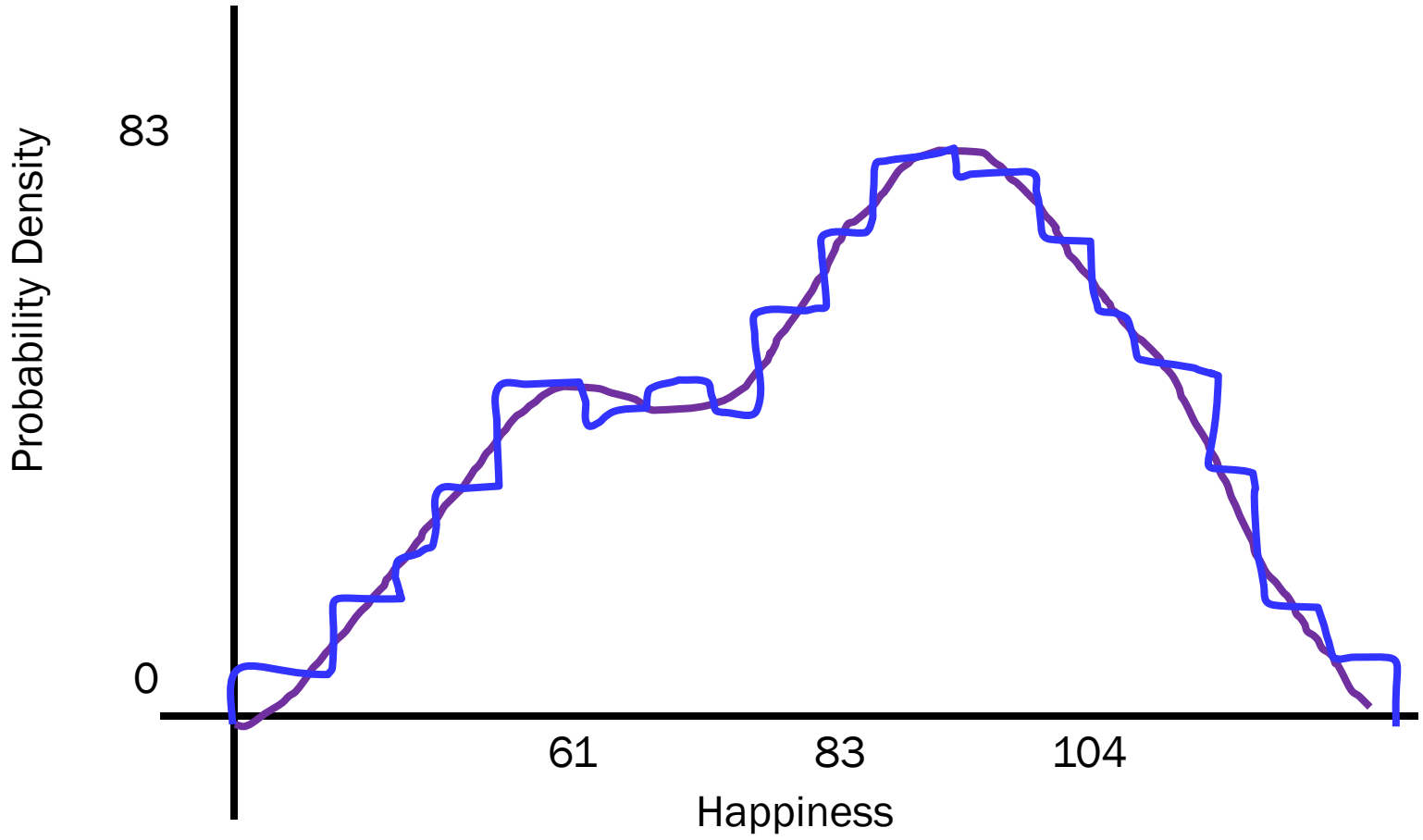
If I Gave You the True Distribution, what would you do?

What is the **std** of the **sample variance**, calculated from 200 people?



But Wait – What If You Actually Have a Good Estimate?

You can estimate the PMF of the underlying distribution, using your sample.*



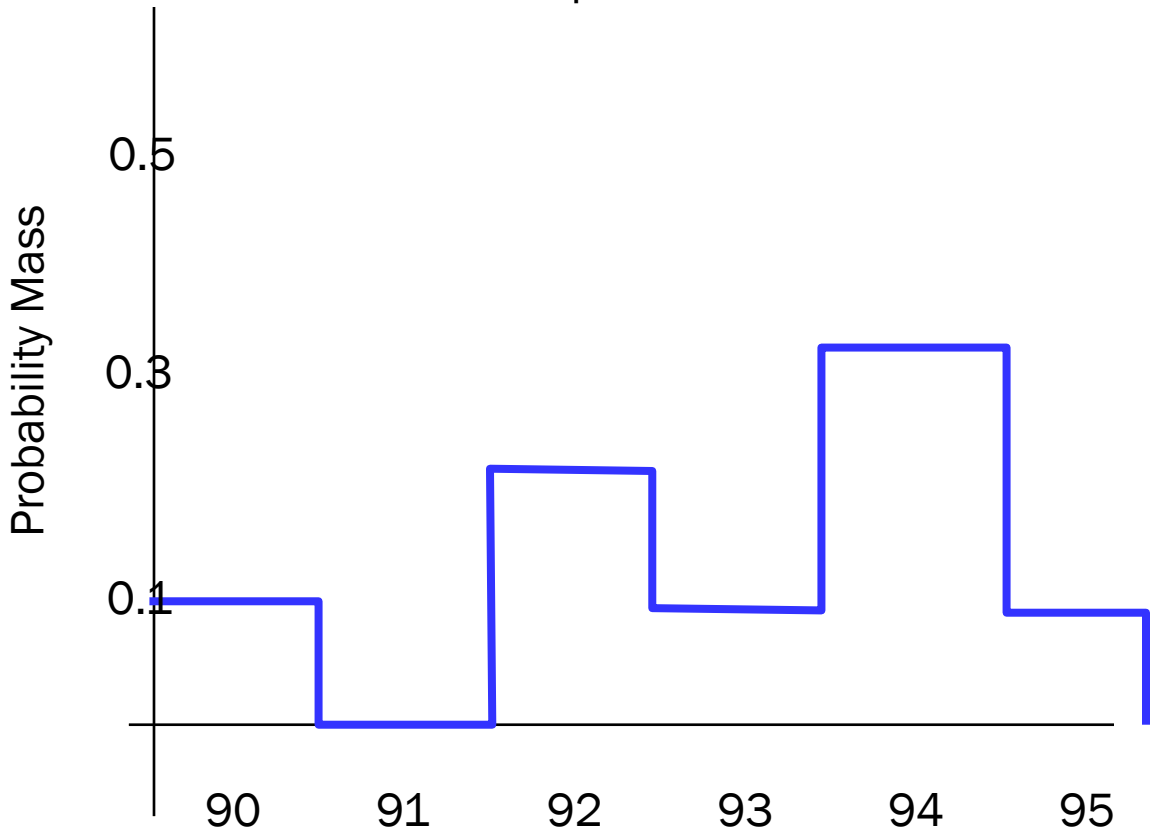
* This is just a histogram of your data!!

Key Insight

IID Samples

90,
92,
92,
93,
94,
94,
94,
95,

Sample Distribution



Bootstrapping Assumption

$$F \approx \hat{F}$$

The underlying distribution

The sample distribution

(aka the histogram of your data)

Algorithm

Bootstrap Algorithm (sample):

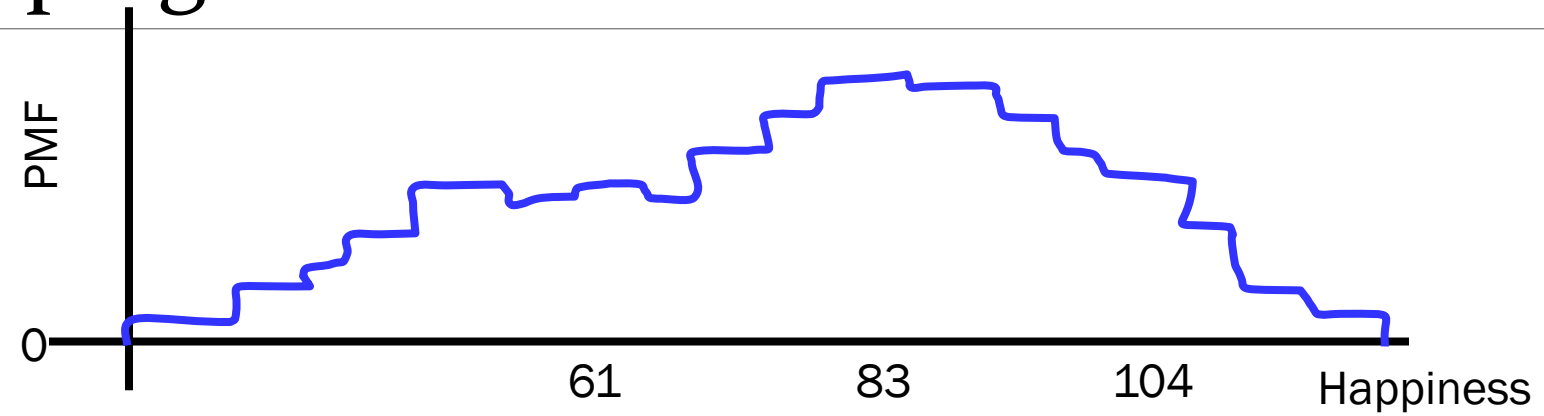
1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Resample **len(sample)** from PMF
   b. Recalculate the stat on the resample
3. You now have a **distribution of your stat**

Bootstrapping of Means (we could do this with CLT)

Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

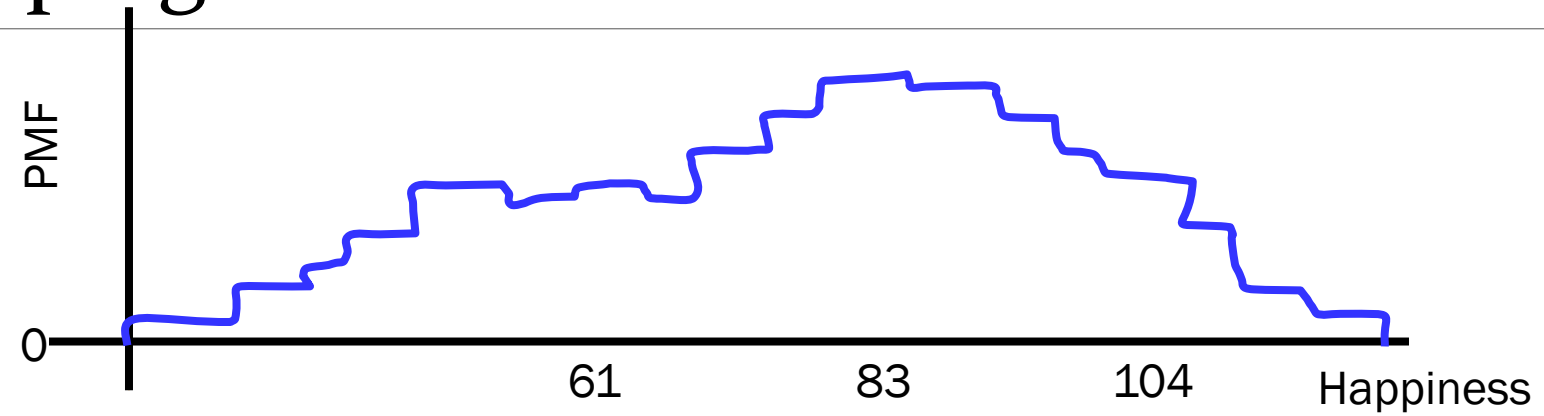
Bootstrapping of Means



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

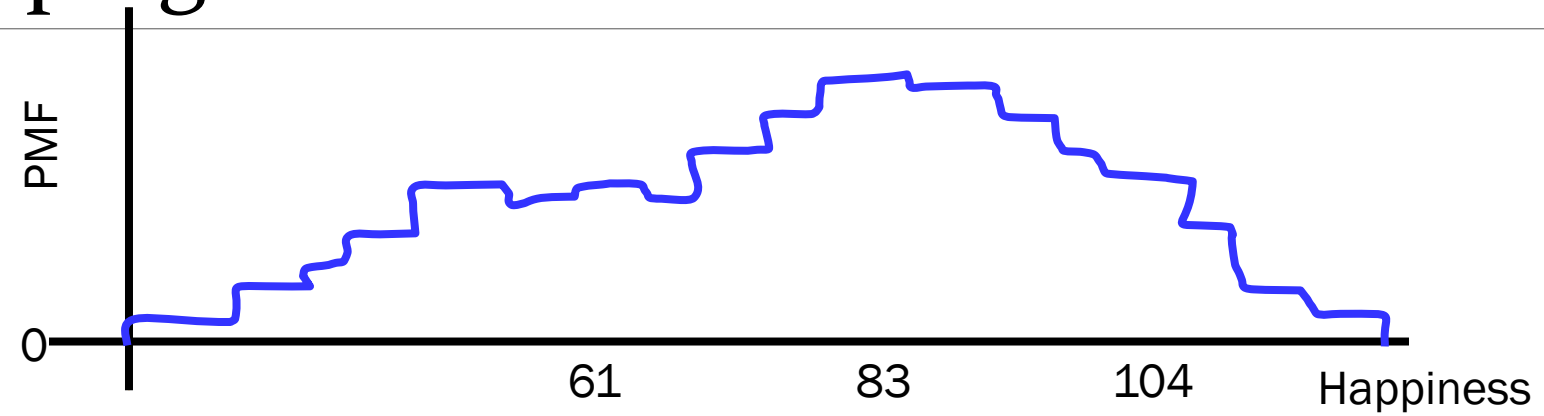
Bootstrapping of Means



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

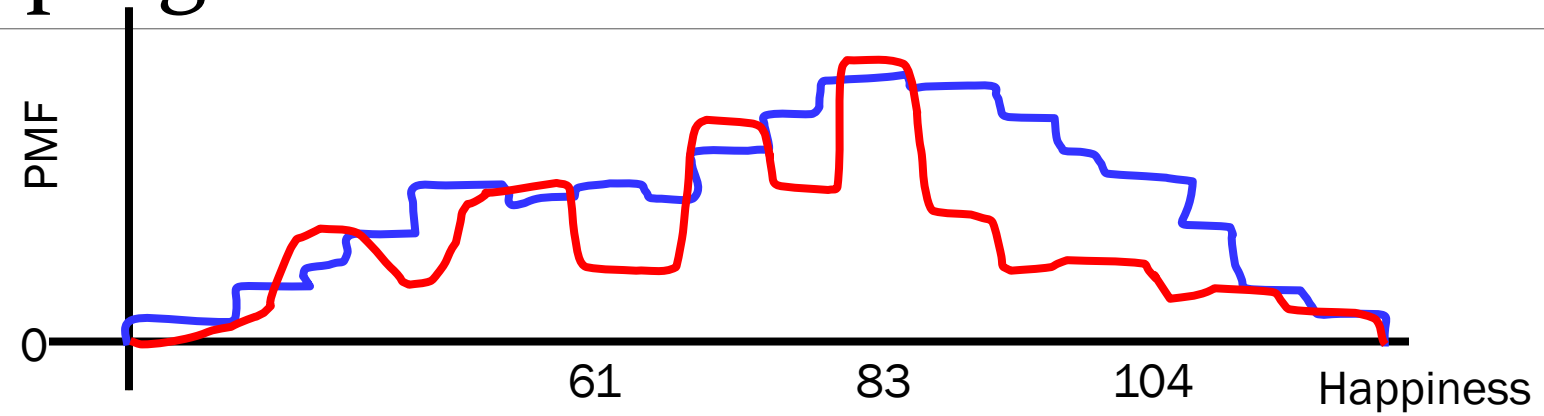
Bootstrapping of Means



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

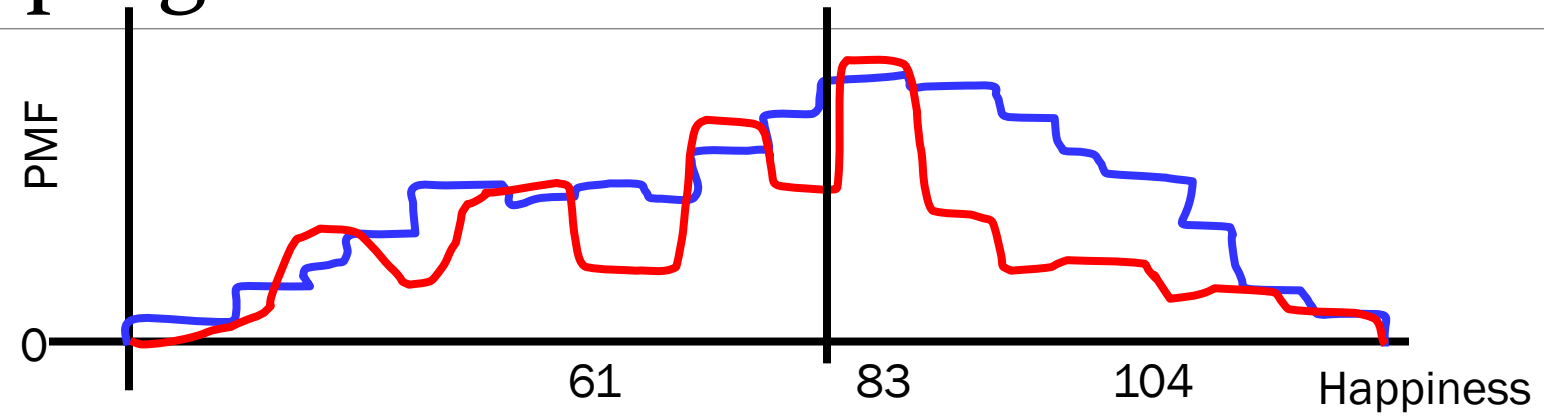
Bootstrapping of Means



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

Bootstrapping of Means

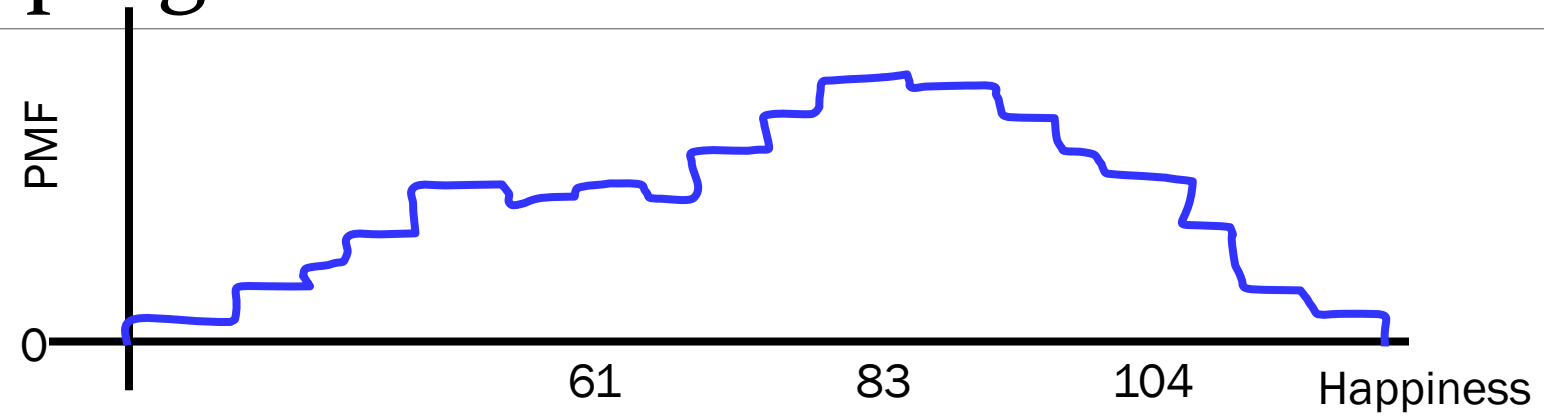


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

Means = [82.7]

Bootstrapping of Means

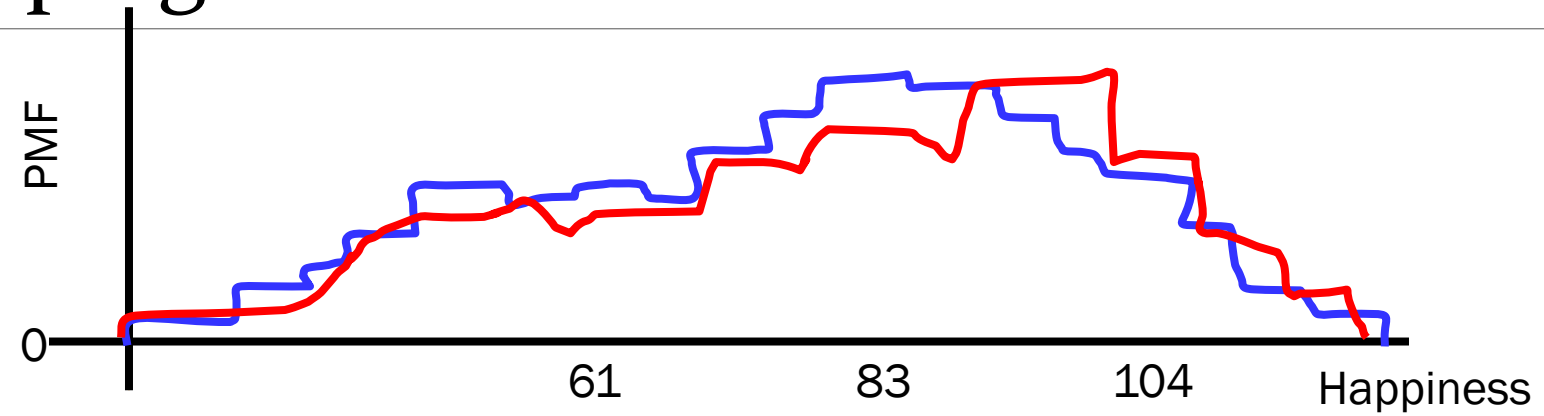


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean** on the resample
3. You now have a **distribution of your means**

Means = [82.7]

Bootstrapping of Means

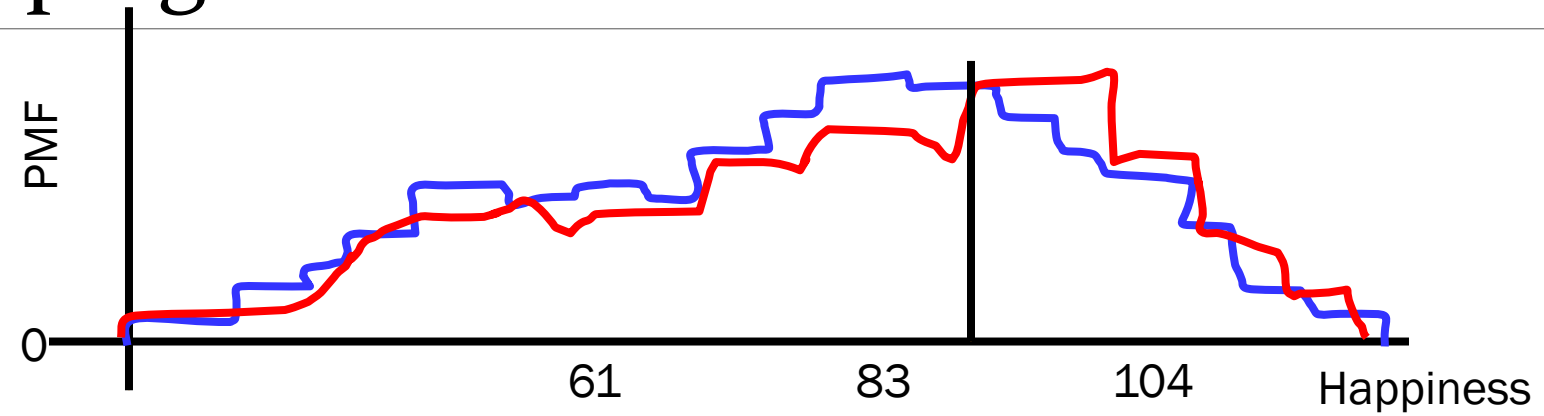


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

Means = [82.7]

Bootstrapping of Means

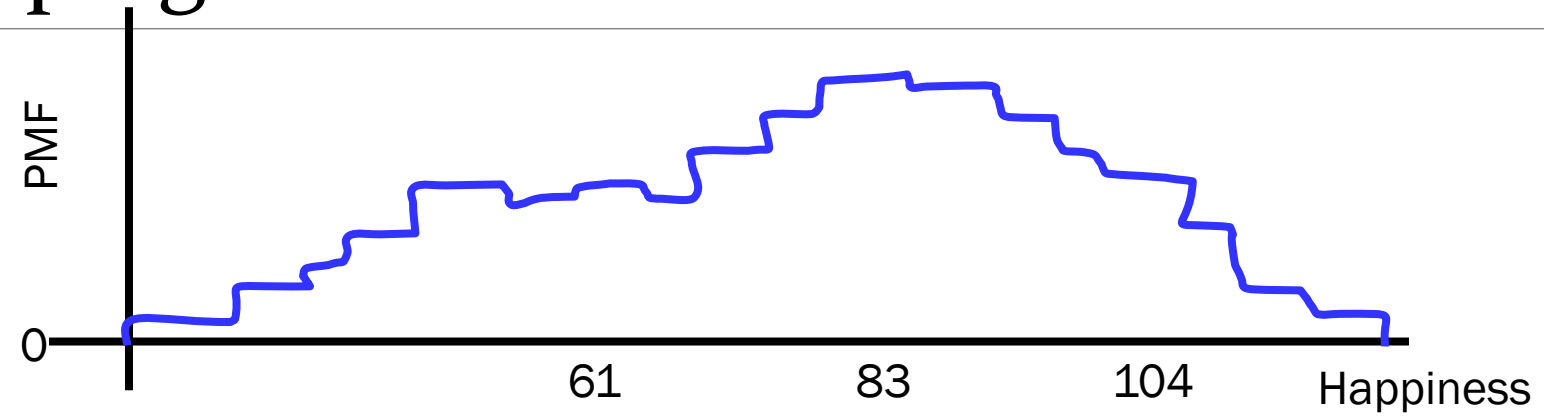


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean on the resample**
3. You now have a **distribution of your means**

Means = [82.7, 83.4]

Bootstrapping of Means

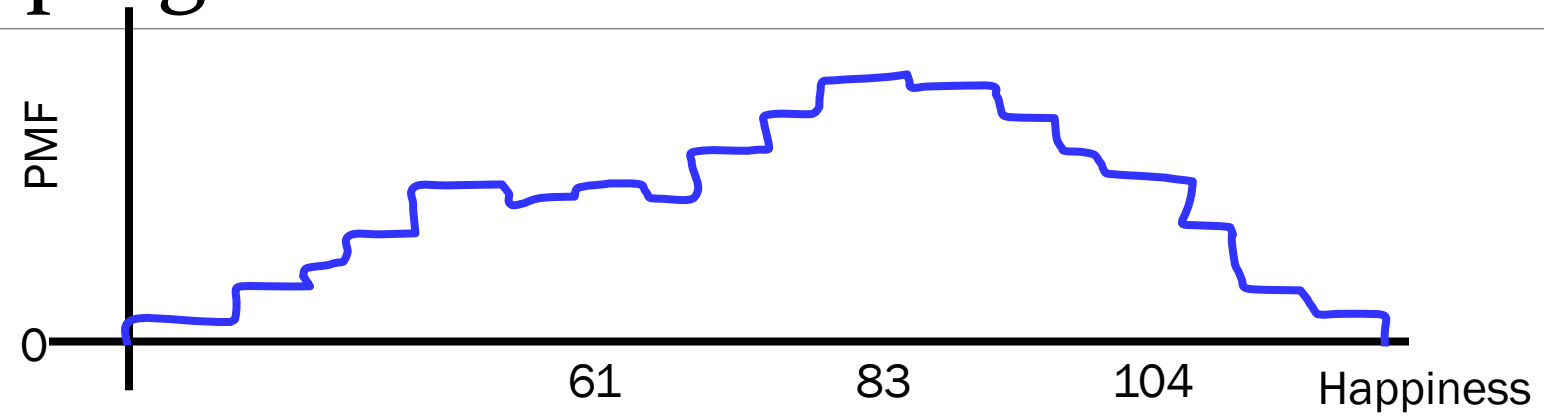


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean** on the resample
3. You now have a **distribution of your means**

Means = [82.7, 83.4]

Bootstrapping of Means



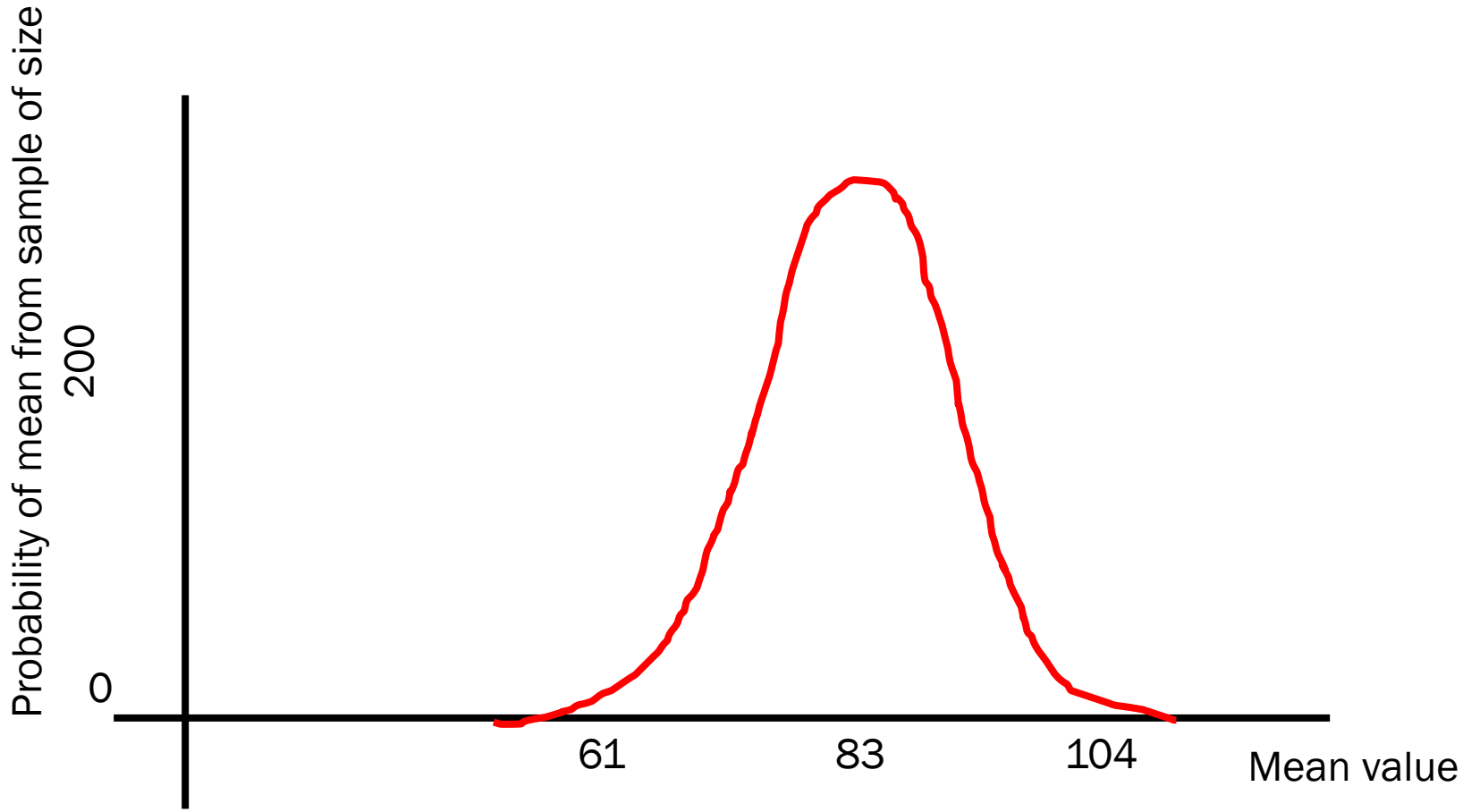
Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the mean** on the resample
3. You now have a **distribution of your means**

Means = [82.7, 83.4, 82.9, 91.4, 79.3, 82.1, ..., 81.7]

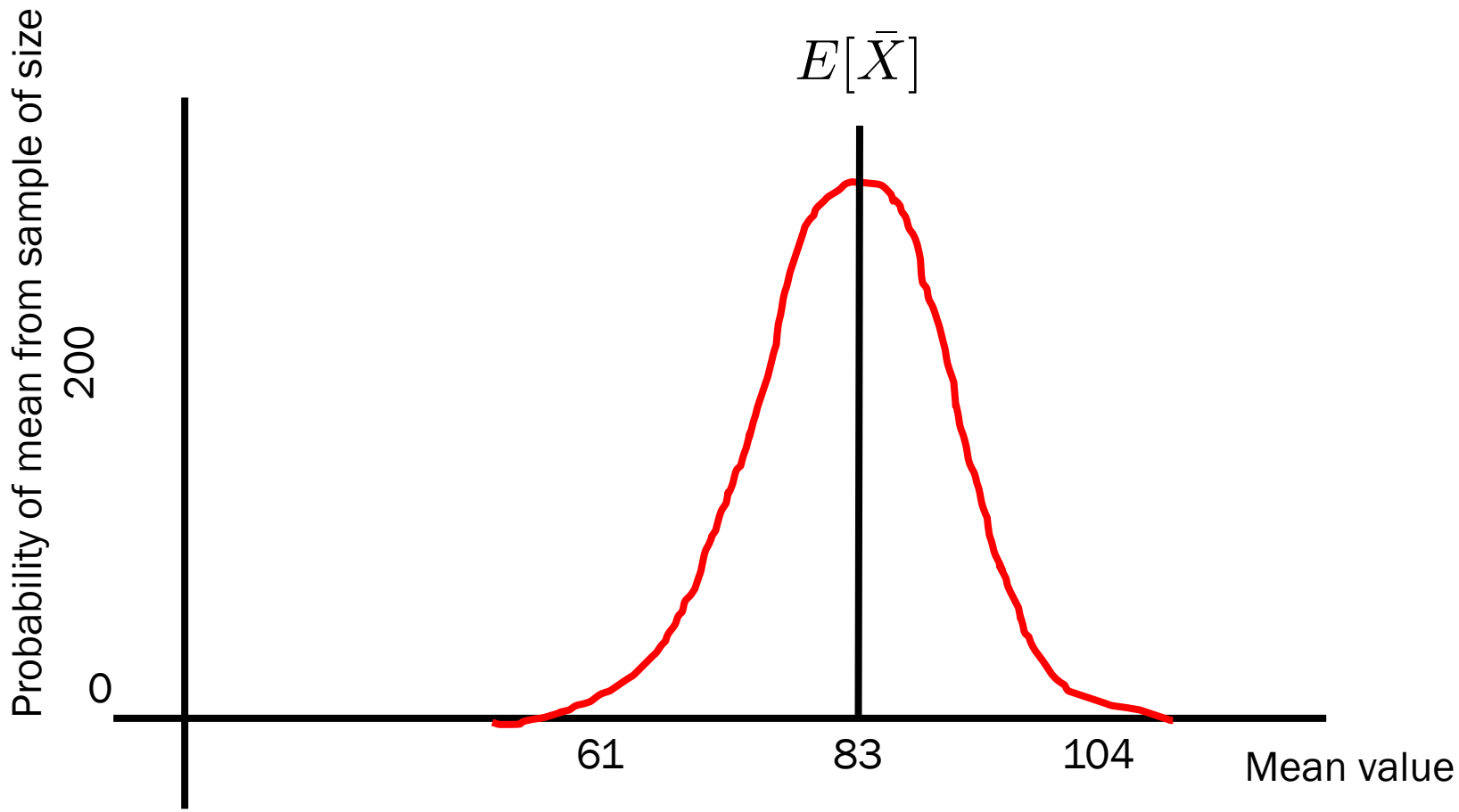
Bootstrapping of Means

Means = [82.7, 83.4, 82.9, 91.4, 79.3, 82.1, ..., 81.7]



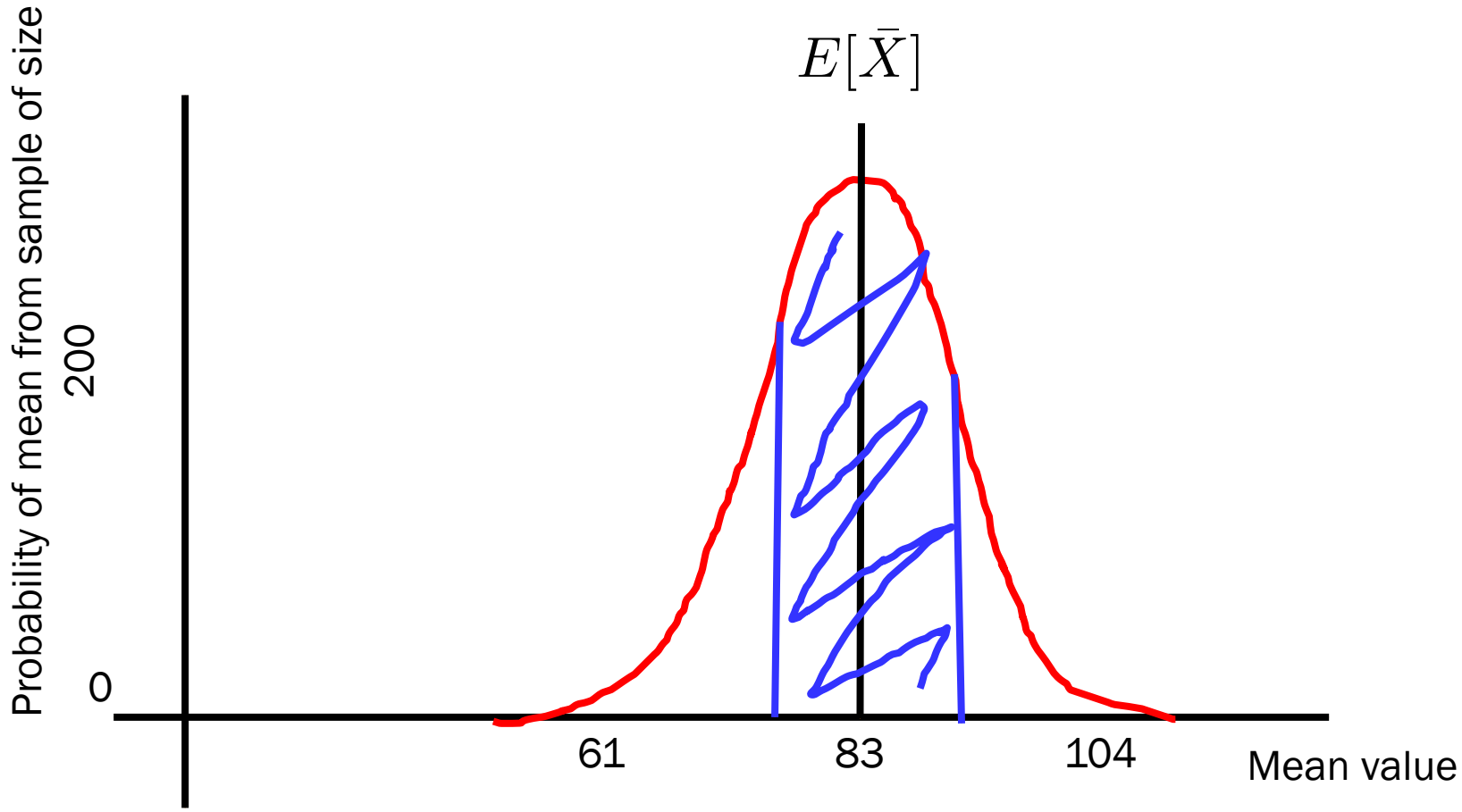
Bootstrapping of Means

Means = [82.7, 83.4, 82.9, 91.4, 79.3, 82.1, ..., 81.7]

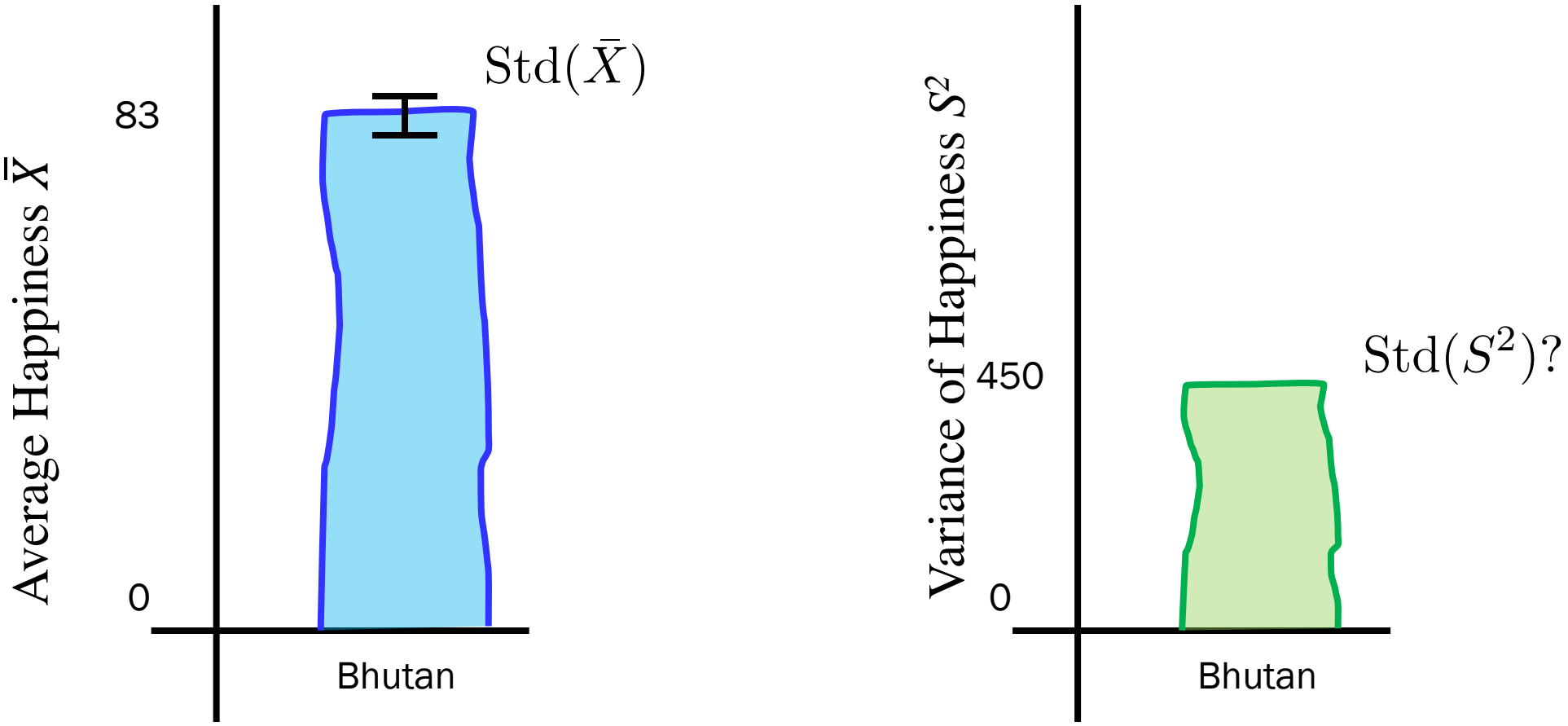


Bootstrapping of Means

What is the probability that the mean is in the range 81 to 85?



Our Report to Bhutan Government



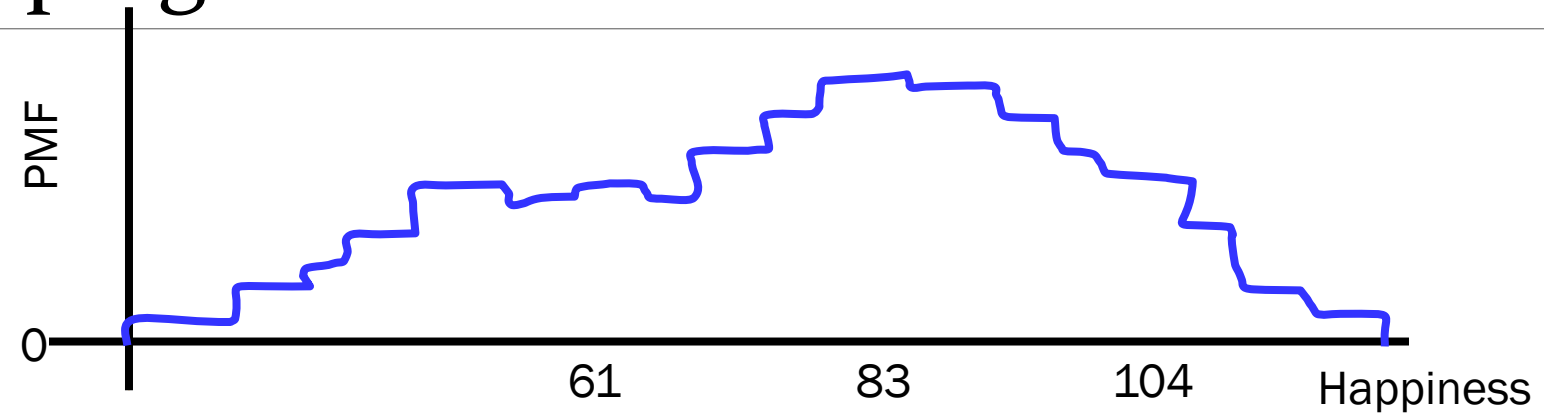
Claim: The average happiness of Bhutan is 83 ± 2

Bootstrapping of Variance

Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the variance on the resample**
3. You have a **distribution of your variances**

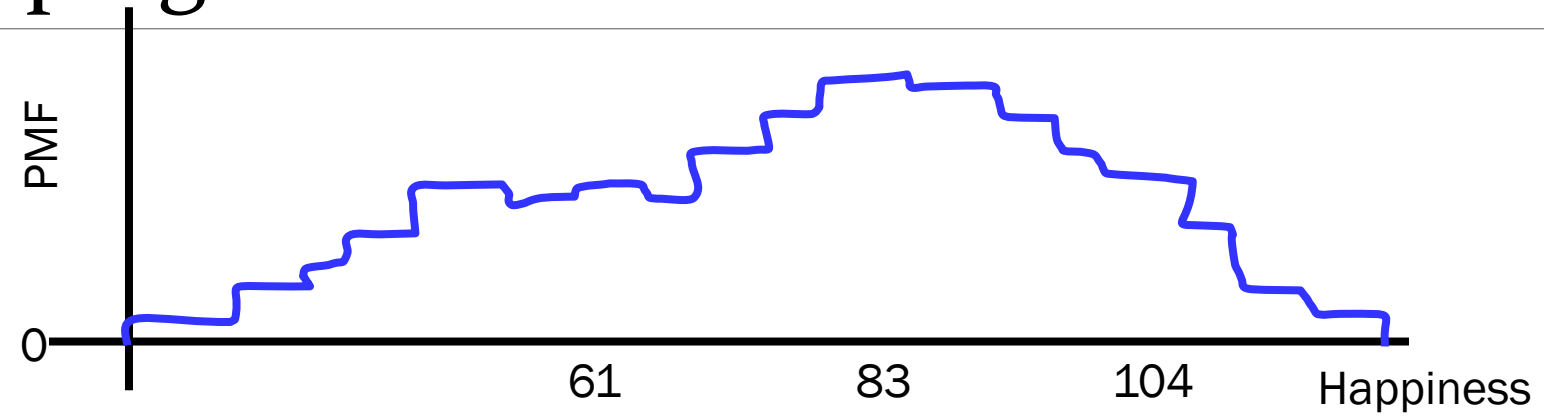
Bootstrapping of Variance



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

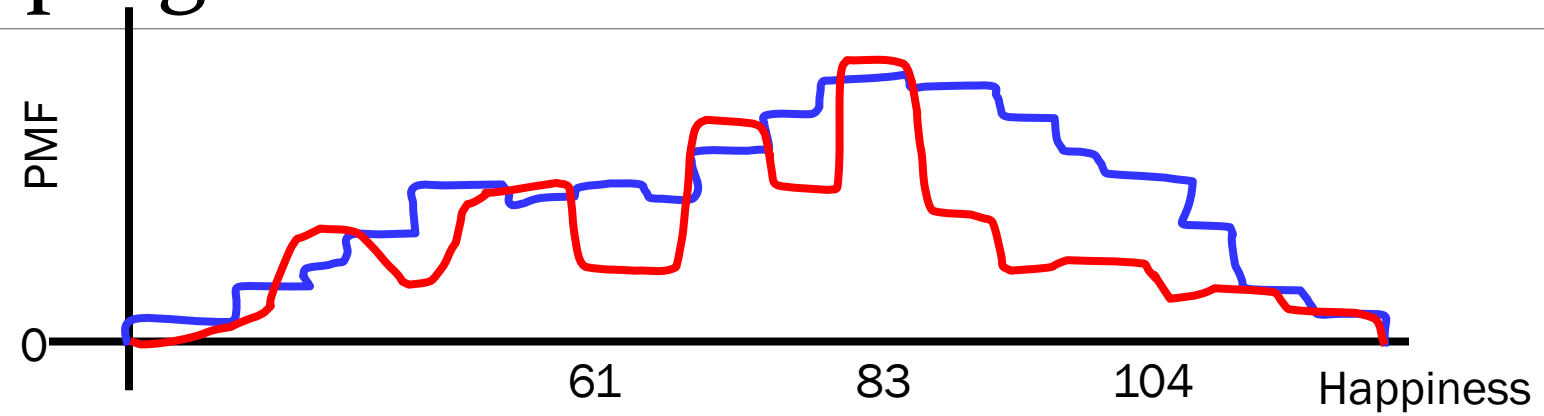
Bootstrapping of Variance



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

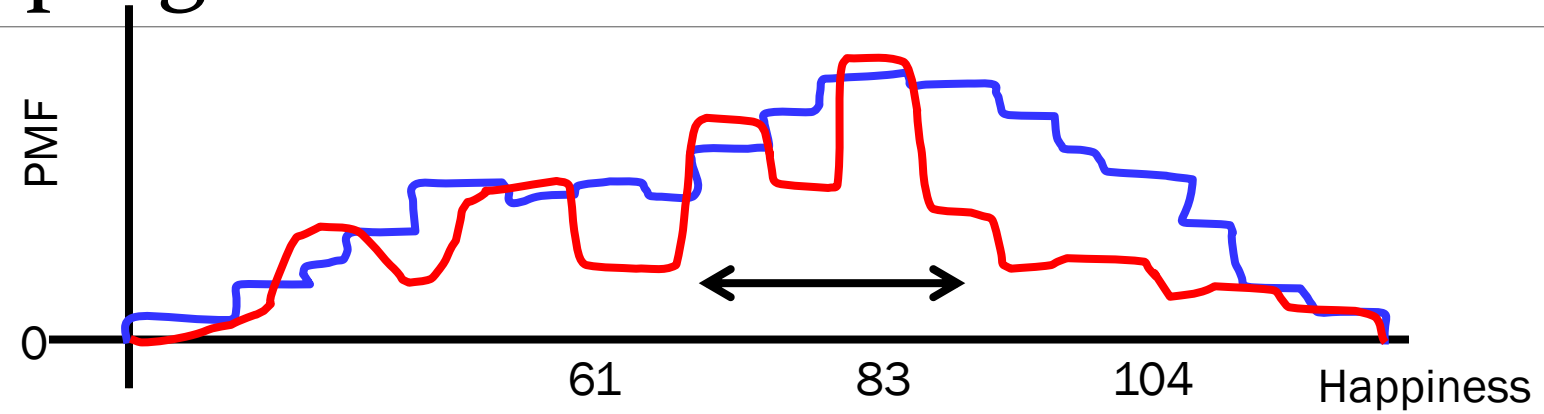
Bootstrapping of Variance



Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var** on the resample
3. You now have a **distribution of your vars**

Bootstrapping of Variance

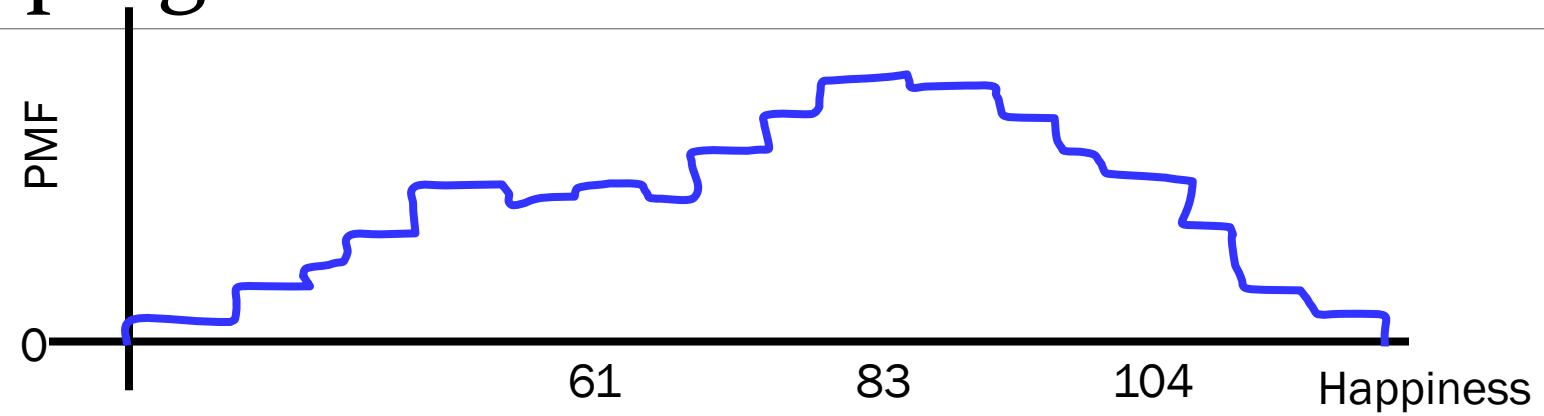


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the vars on the resample**
3. You now have a **distribution of your vars**

Vars = [472.7]

Bootstrapping of Variance

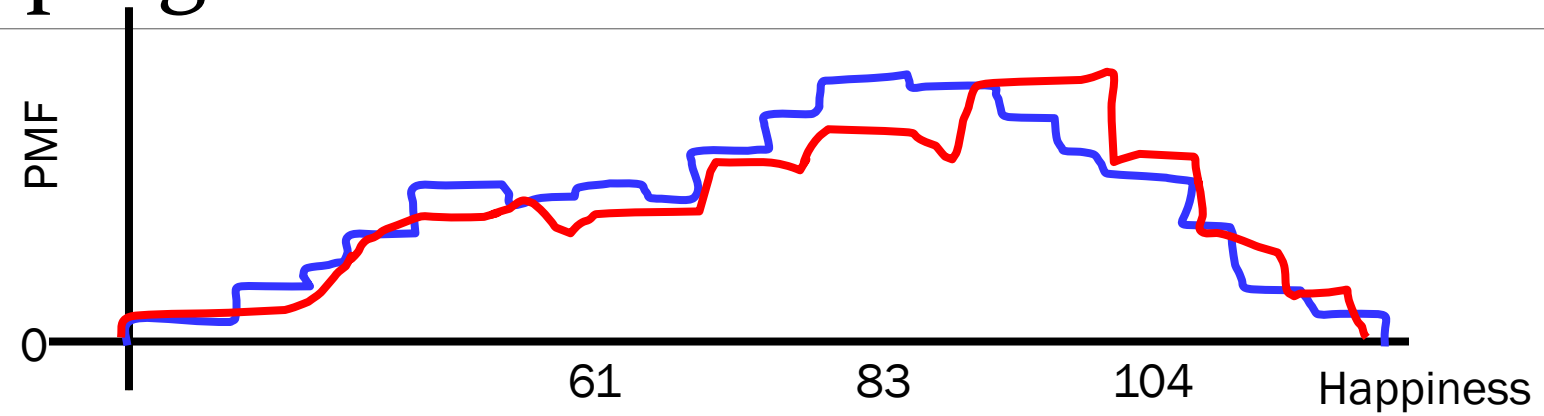


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var** on the resample
3. You now have a **distribution of your vars**

Vars = [472.7]

Bootstrapping of Variance

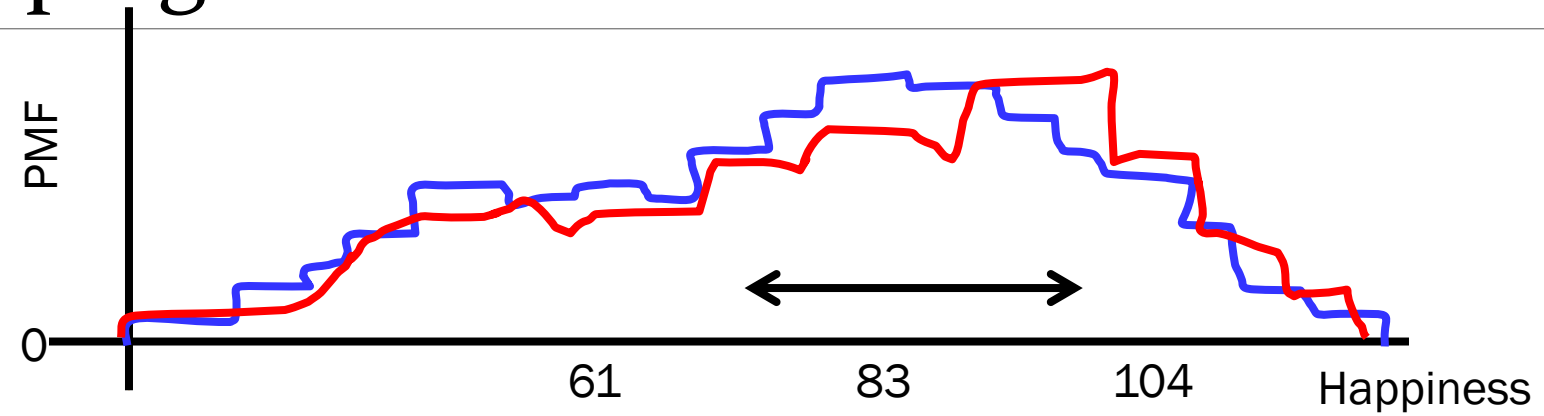


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

Vars = [472.7]

Bootstrapping of Variance

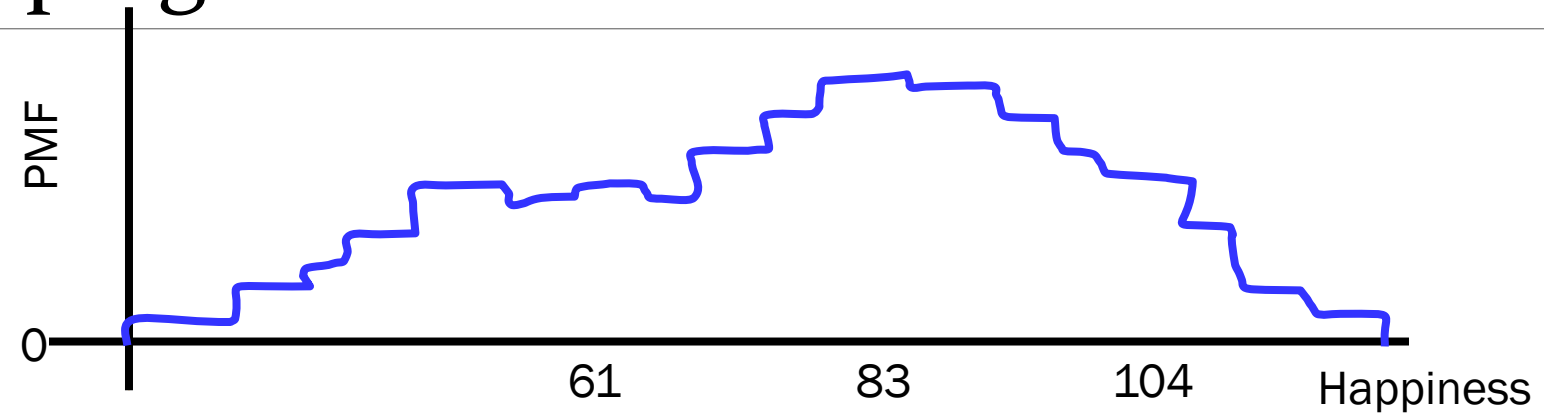


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

Vars = [472.7, 478.4]

Bootstrapping of Variance

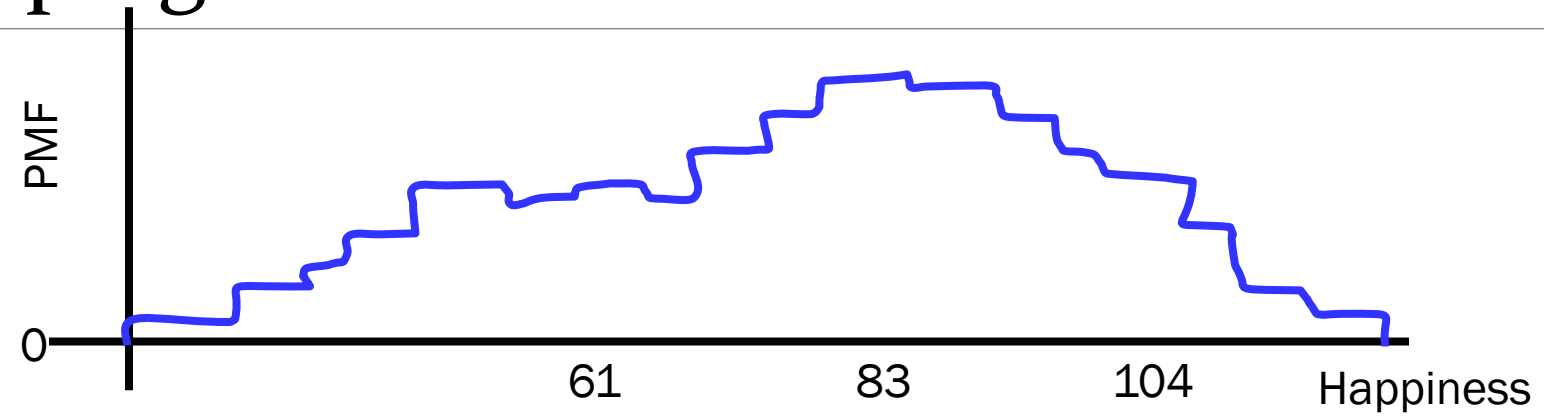


Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

Vars = [472.7, 478.4]

Bootstrapping of Variance



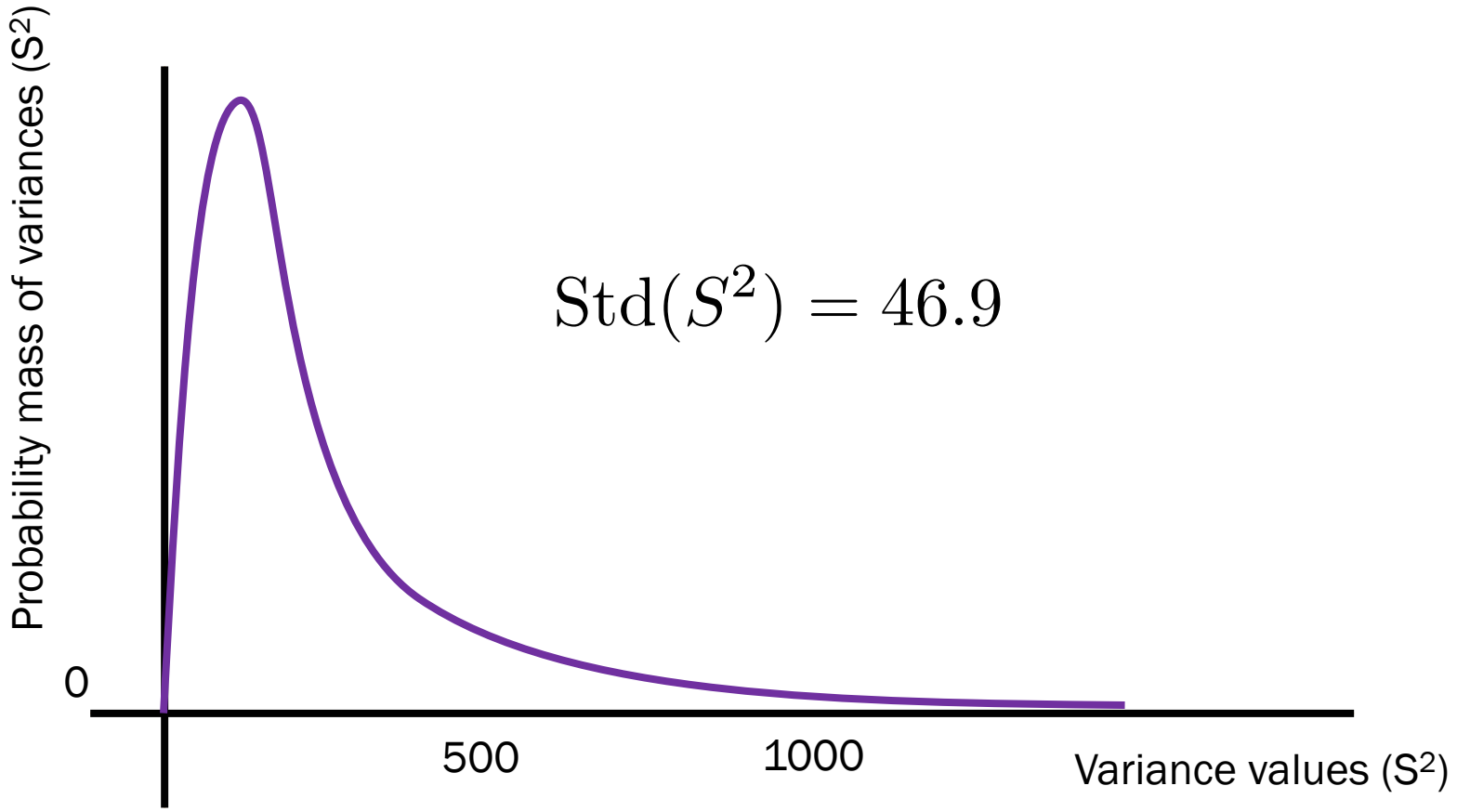
Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw **len(sample)** new samples from PMF
   b. **Recalculate the var on the resample**
3. You now have a **distribution of your vars**

Vars = [472.7, 478.4, 469.2, ..., 476.2]

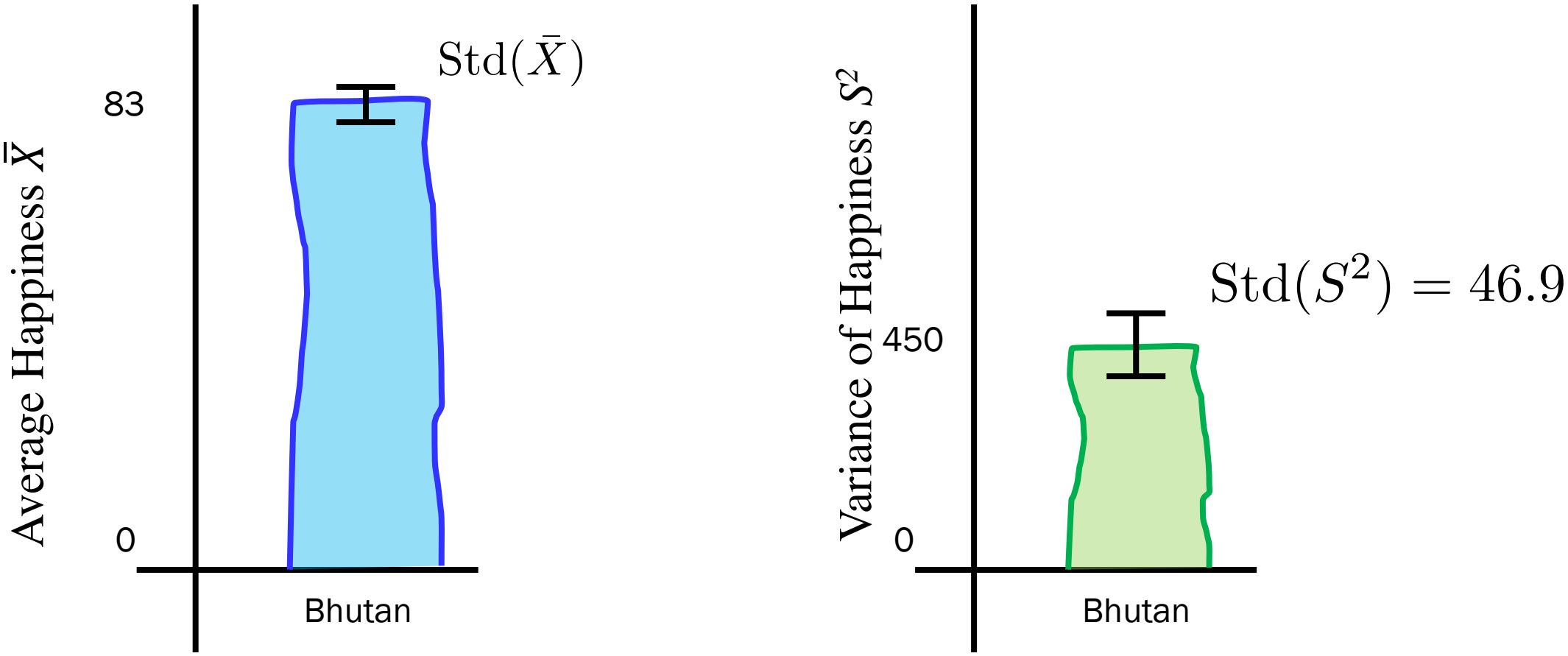
Bootstrapping of Variance

Sample Vars = [472.7, 478.4, 469.2, ..., 476.2]



$$\text{Std}(S^2) = 46.9$$

Our Report to Bhutan Government



Claim: The average happiness of Bhutan is 83 ± 2

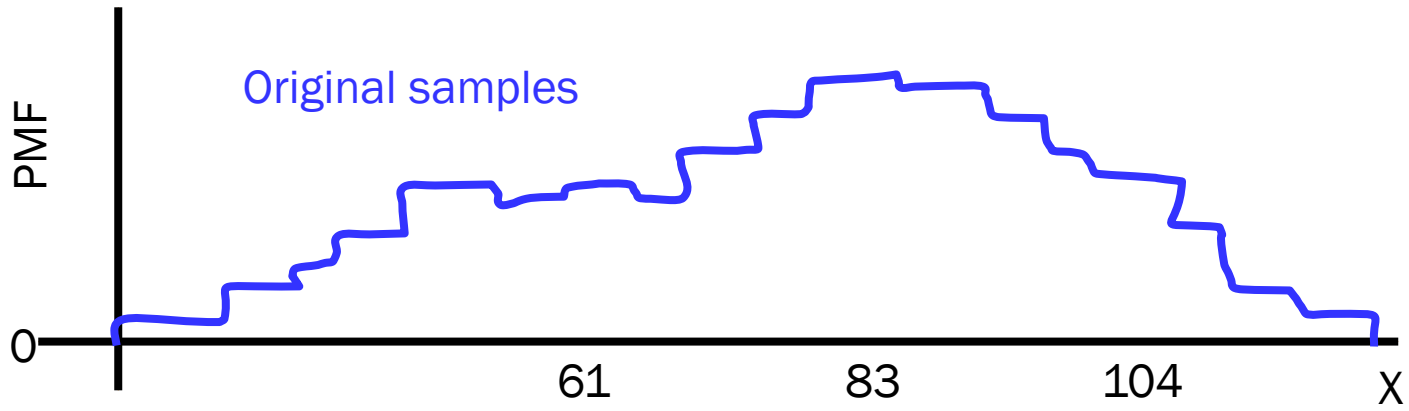
Pedagogical pause

Bootstrap Algorithm for E[S^2] (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Draw len(**sample**) new samples from PMF
   b. Recalculate the var on the resample
3. You now have a **distribution of your vars**

Bootstrapping in Practice

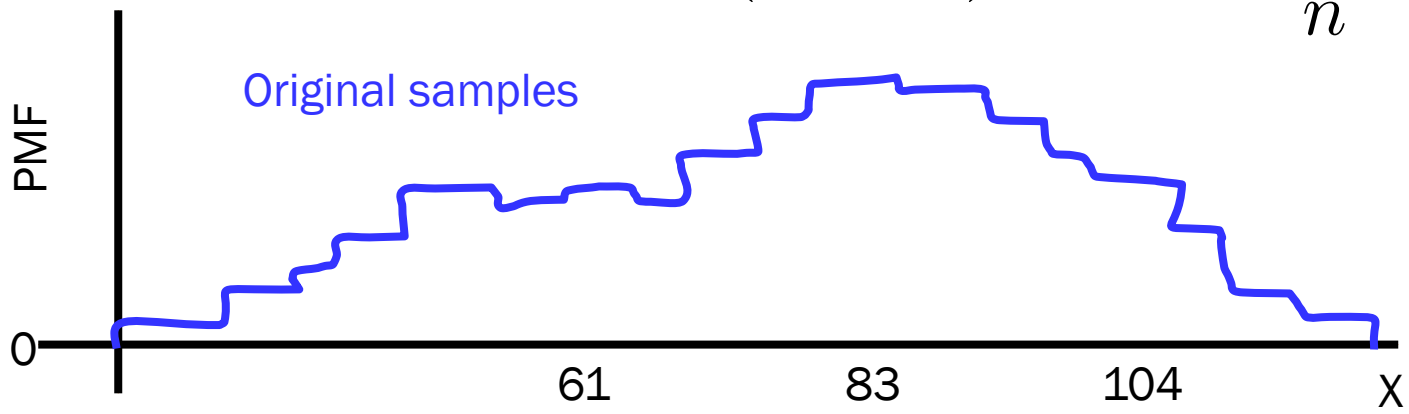
```
def resample(samples, K):
    # Estimate the PMF using the samples
    # Draw K new samples from the PMF
```



Bootstrapping in Practice

```
def resample(samples, K):
    # Estimate the PMF using the samples
    # Draw K new samples from the PMF
    return np.random.choice(samples, K,
                            replace = True)
```

$$P(X = k) = \frac{\text{count}(X = k)}{n}$$



OG Bootstrapping

Bootstrap Algorithm (sample):

1. Estimate the **PMF** using the sample
2. Repeat **10,000** times:
   a. Resample **len(sample)** from PMF
   b. Recalculate the stat on the resample
3. You now have a **distribution of your stat**

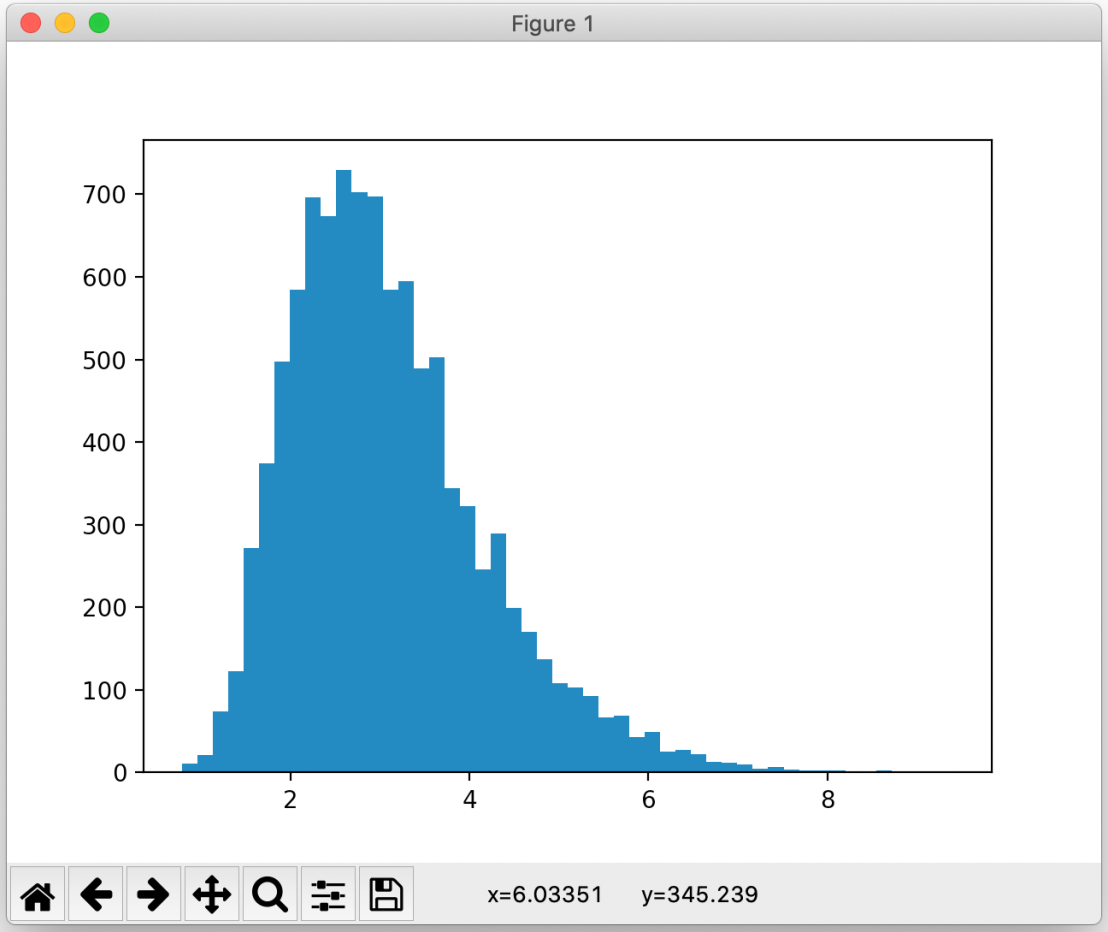
Bootstrapping in Practice

Bootstrap Algorithm (sample):

1. Repeat **10,000** times:
   a. **Choose len(sample) elems from sample, with replacement**
   b. Recalculate the stat on the resample
2. You now have a **distribution of your stat**

To the code!

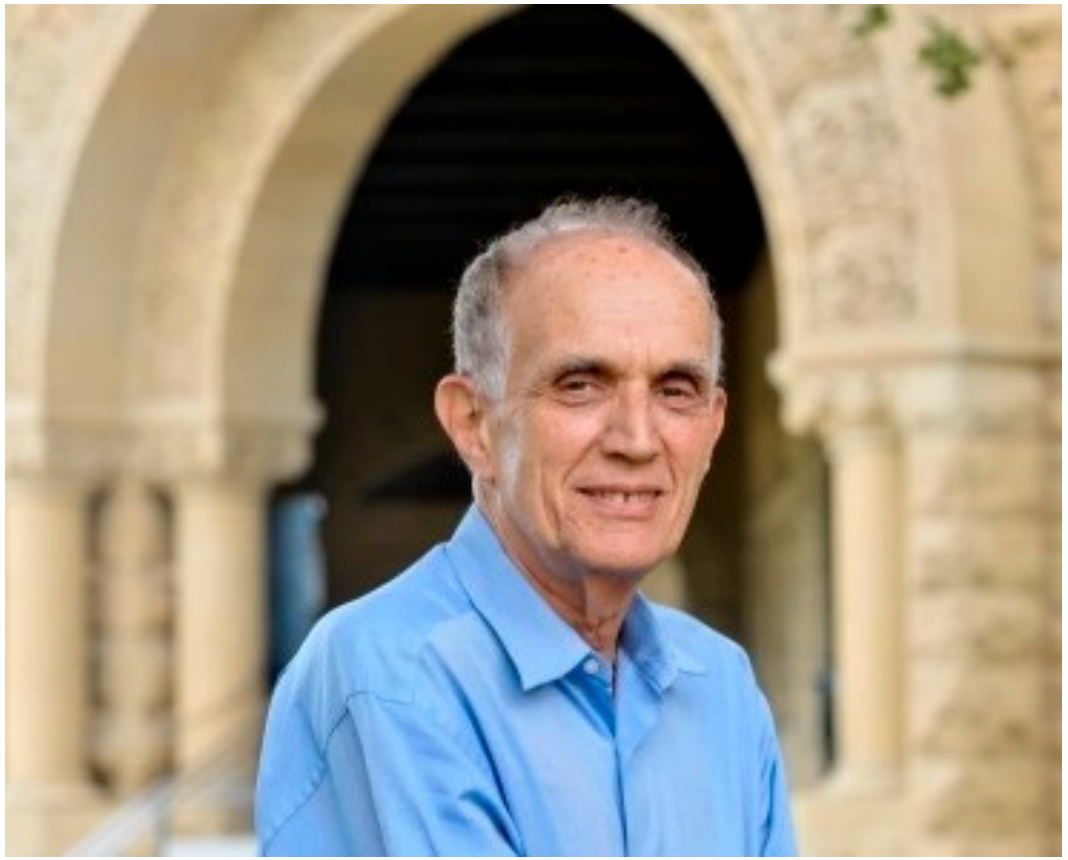
The Distribution of the Sampling Variance



Bootstrap provides a way to calculate probabilities of statistics using code.

Bootstrap

Bradley Efron



Invented bootstrapping in 1979

Still a professor at Stanford

Won a National Science Medal

According to starbyface.com:
Dolph Lundgren

Works for any statistic*

*as long as your samples are IID and the underlying distribution doesn’t have a long tail

The Classic Science Test

Group 1	Group 2
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83
$\mu_1 = 3.1$	$\mu_2 = 2.4$

Claim: Group 1 and Group 2 are samples from different distributions with a 0.7 difference of means.

How confident are you in this claim?

A real difference?

Learning in Context A	Learning in Context B
4.44	2.15
3.36	3.01
5.87	2.02
2.31	1.43
...	...
3.70	1.83

18 students

23 students

$\mu_1 = 3.1$

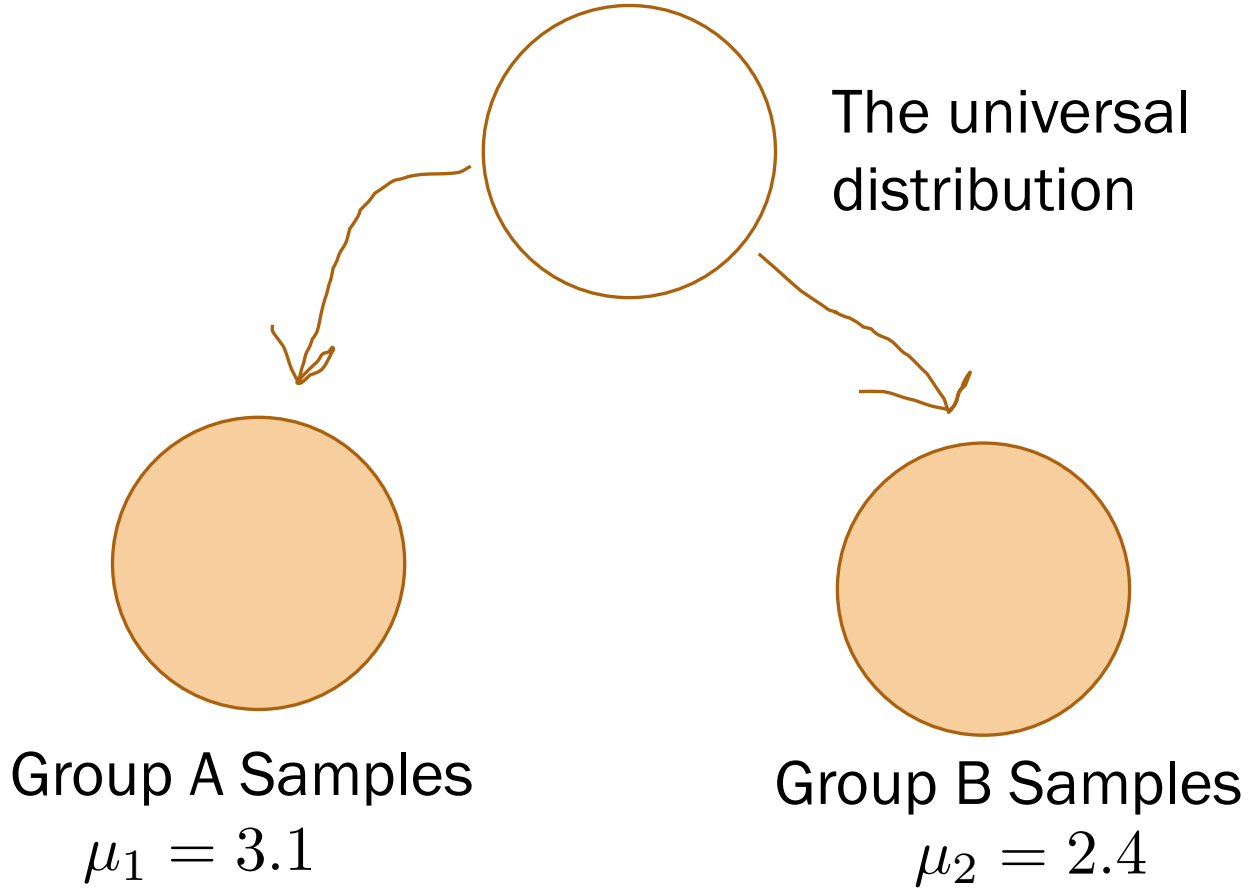
$\mu_2 = 2.4$

Claim: Group 1 and Group 2 are samples from different distributions with a 0.7 difference of means.

How confident are you in this claim?

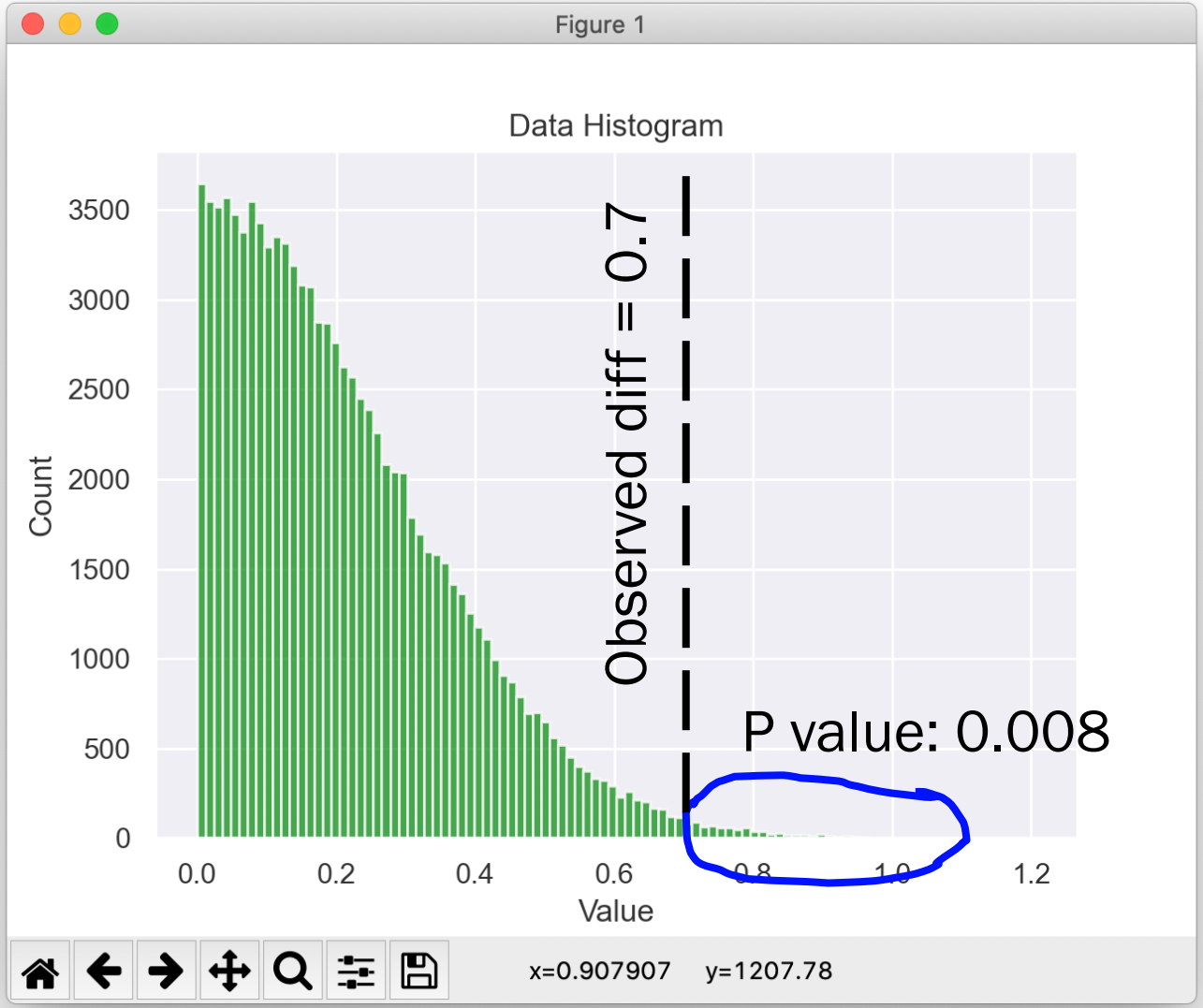
The Null Hypothesis

There is no difference between the two groups, so everyone is drawn from the same distribution. Any difference you observe is due to sampling error.



To the code!

Distribution of Mean Diffs under Null Hypothesis



Food For Thought

Two Opinions on Distributions

Results of flipping a coin 20 times. Give your belief distribution of p:

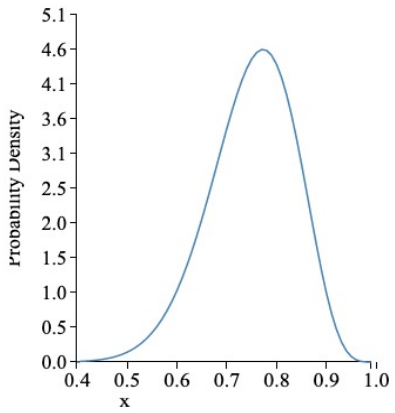
H, H, H, T, H, T, H, H, H, H, H, T, H, H, H, H, H, H, T, H

4 tails, 16 heads

Bayesian:

Let's use Laplace prior

X ~ Beta(a = 18, b = 6)



Frequentist:

Let's bootstrap