ECE 20875

Python for Data Science

Chris Brinton, Qiang Qiu, and Mahsa Ghasemi

(Adapted from material developed by Profs. Milind Kulkarni, Stanley Chan, Chris Brinton, David Inouye, and Qiang Qiu)

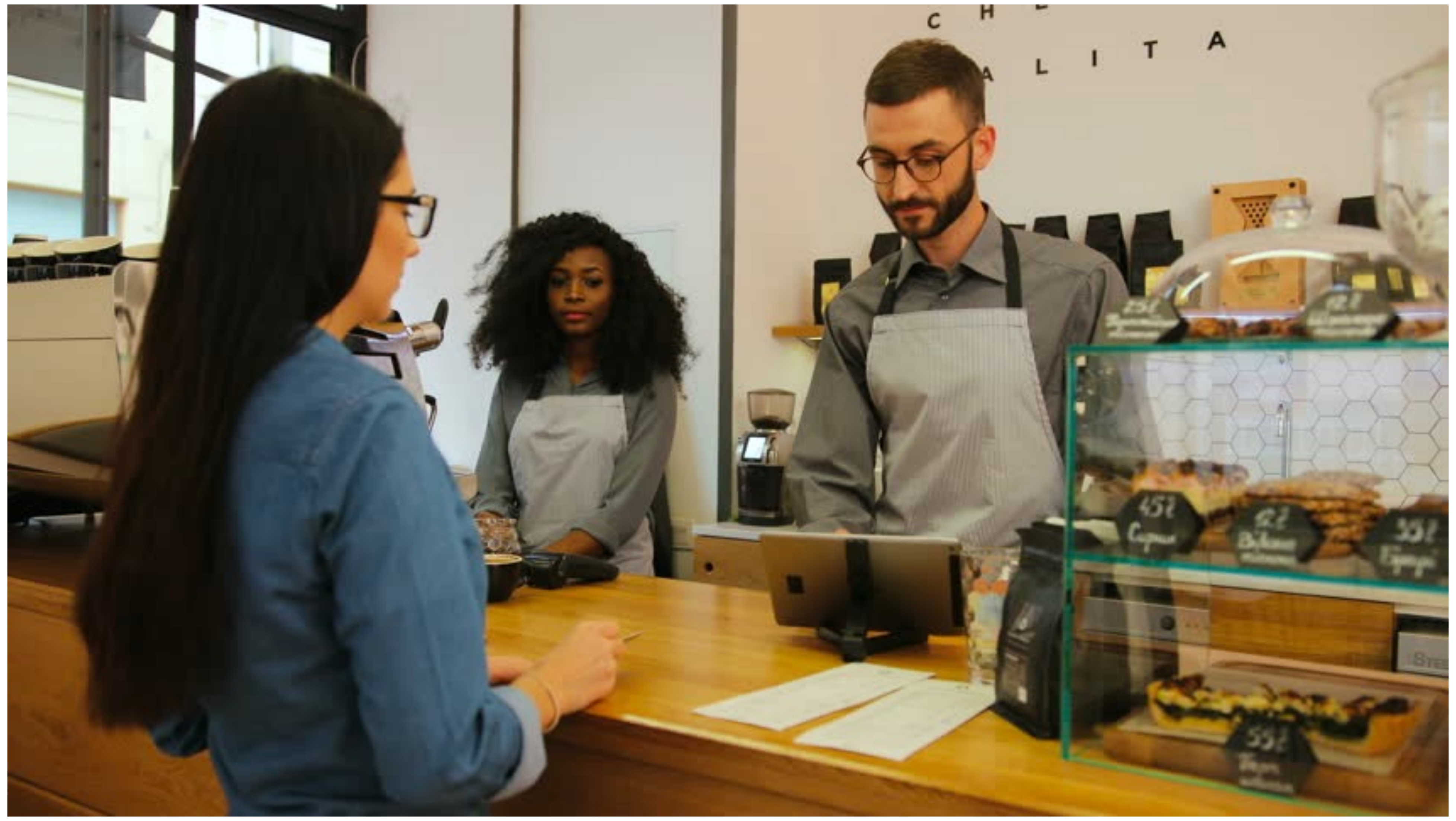
Histograms

a problem

- You’re managing a coffee shop
- Assuming you want to maximize profit, how much coffee should you buy for each day?
    - Too much → Surplus, waste money :(
    - Too little → Unsatisfied demand, under-caffeinated customers :(
- What should you do?

collect data

- Count how many people get coffee in a day
  - Day 1: 37 people
  - Likely different each day of the week, and the type of coffee (cold brew, late, etc.) also has an impact
  - Assume such factors do not matter (problem is still interesting!)
- Should we just get enough coffee for 37 people?



(keep) collect(ing) data

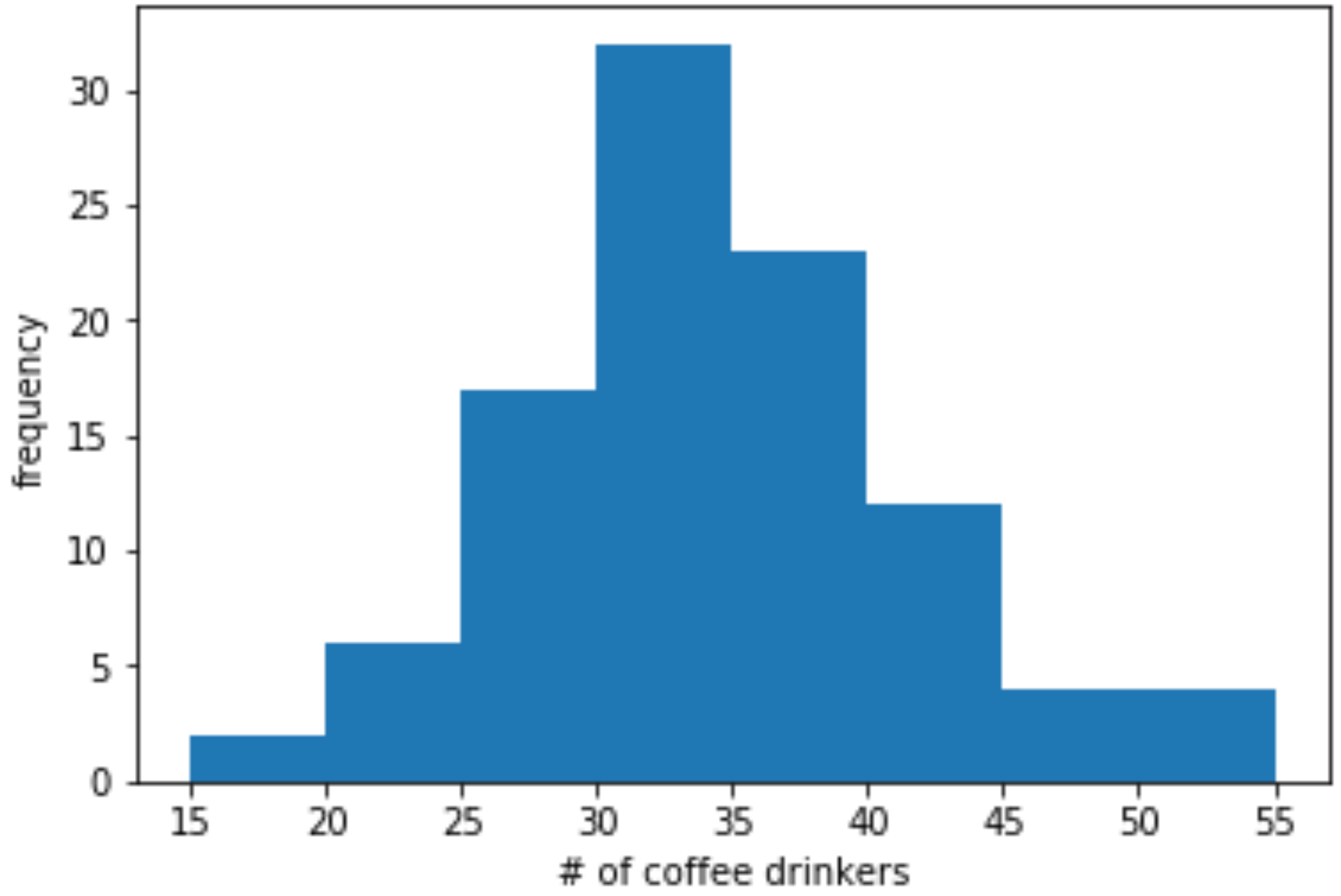
- Day 2: 43
- Day 3: 48
- Day 4: 41
- Day 5: 46
- Day 6: 19 (!)
- Day 7: 38
- …

100 days later ...

```
[37, 43, 48, 41, 46, 19, 28, 35, 34, 38,
31, 32, 32, 23, 23, 33, 35, 39, 34, 28,
39, 28, 29, 38, 28, 30, 25, 35, 39, 35,
31, 28, 25, 26, 15, 31, 28, 32, 40, 21,
34, 38, 30, 47, 34, 31, 51, 30, 41, 36,
33, 51, 22, 25, 29, 50, 32, 39, 25, 37,
54, 33, 36, 25, 30, 22, 41, 35, 31, 40,
30, 33, 27, 36, 27, 34, 24, 41, 37, 29,
48, 40, 31, 32, 33, 32, 40, 31, 32, 40,
31, 33, 32, 38, 37, 41, 37, 39, 38, 42]
```

visualize the data

- Staring at a list of numbers is not very illuminating
- Visualizing the data in a useful way can help reveal patterns
  - **Data visualization** is an important subset of data science
- Since the data consists of a single, numeric variable, we can try a **histogram**



building a histogram

- A histogram visualizes observations of a random variable d
- Each bar in a histogram is a **bin**

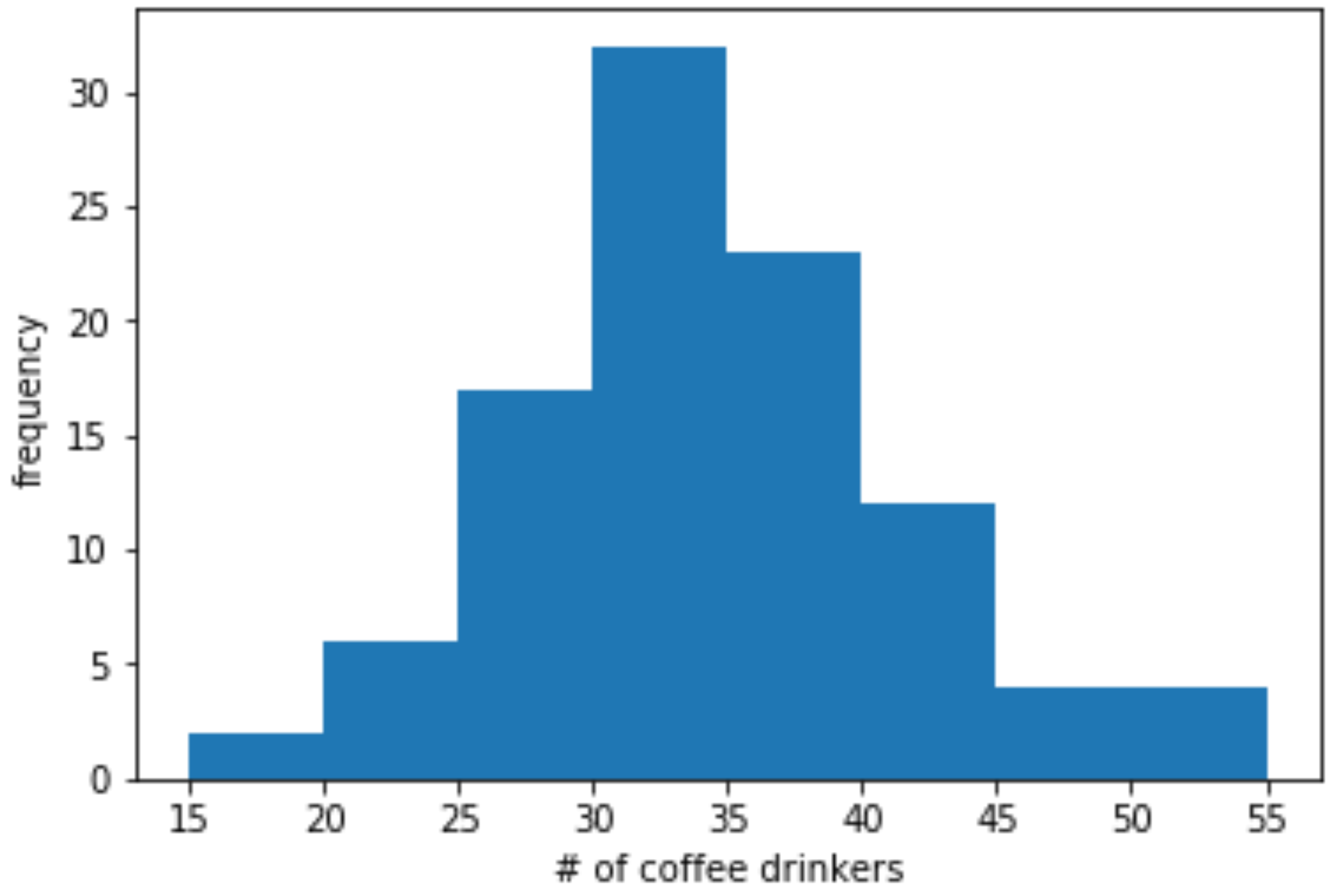
$$x_1, x_2, \ldots$$

- Each observation is placed into one bin

$$x_1 : 15 \leq d < 20, x_2 : 20 \leq d$$
$$< 25, \ldots$$

- The **count** (size/height) of each bin is the number of observations in that bin

$$x_1 : 2, x_2 : 6, \ldots$$




```
import matplotlib.pyplot as plt
_ = plt.hist(data, bins=8, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel(‘frequency')
plt.show()
```

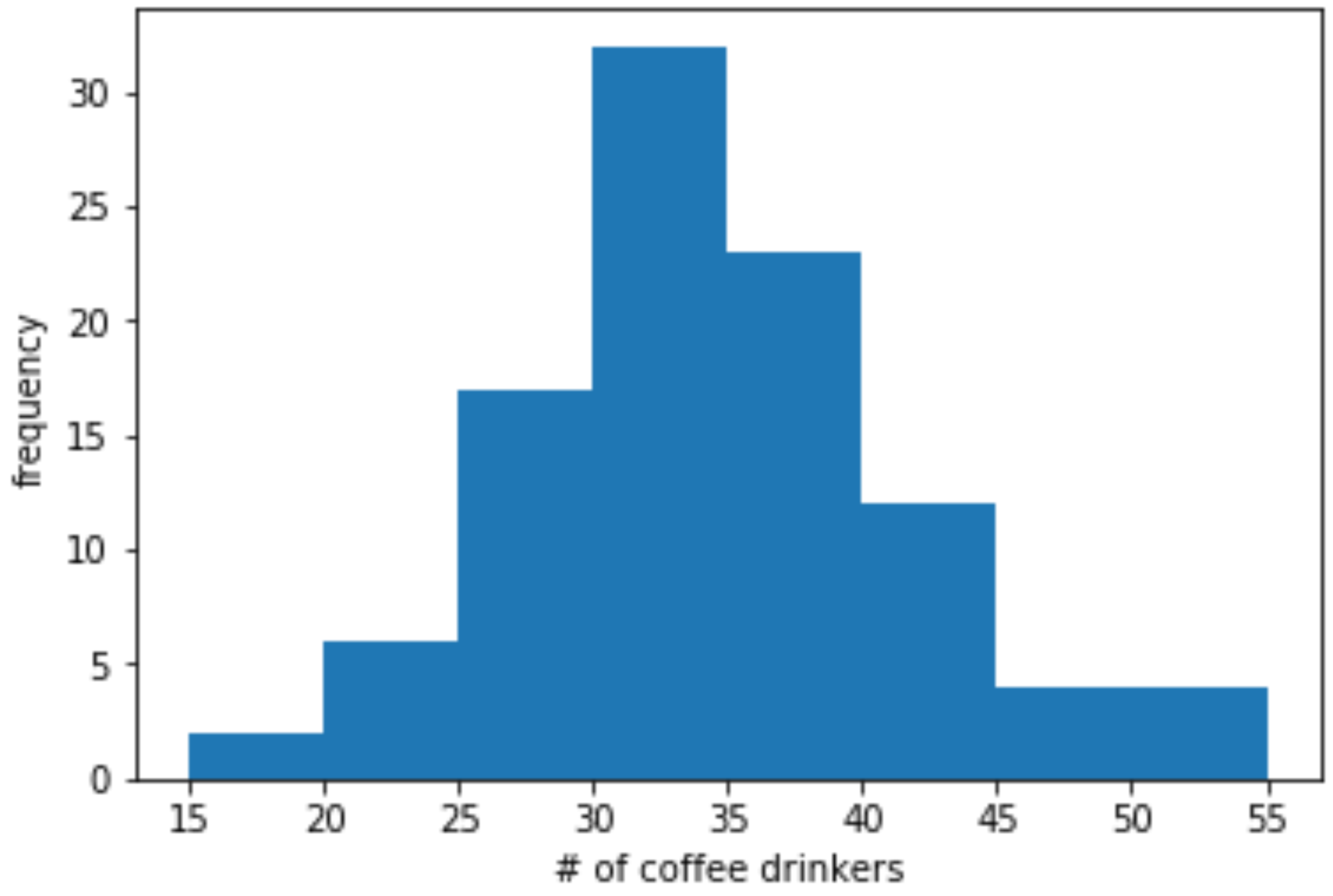
building a histogram

- The empirical (measured) **frequency** of each bin is the fraction of data in that bin

$$\hat{p}_1 = x_1 \,/\, \sum_k x_k = 0.02, \hat{p}_2 = x_2 \,/\, \sum_k x_k = 0.06, \ldots$$

Note that $\sum_k \hat{p}_k = 1$

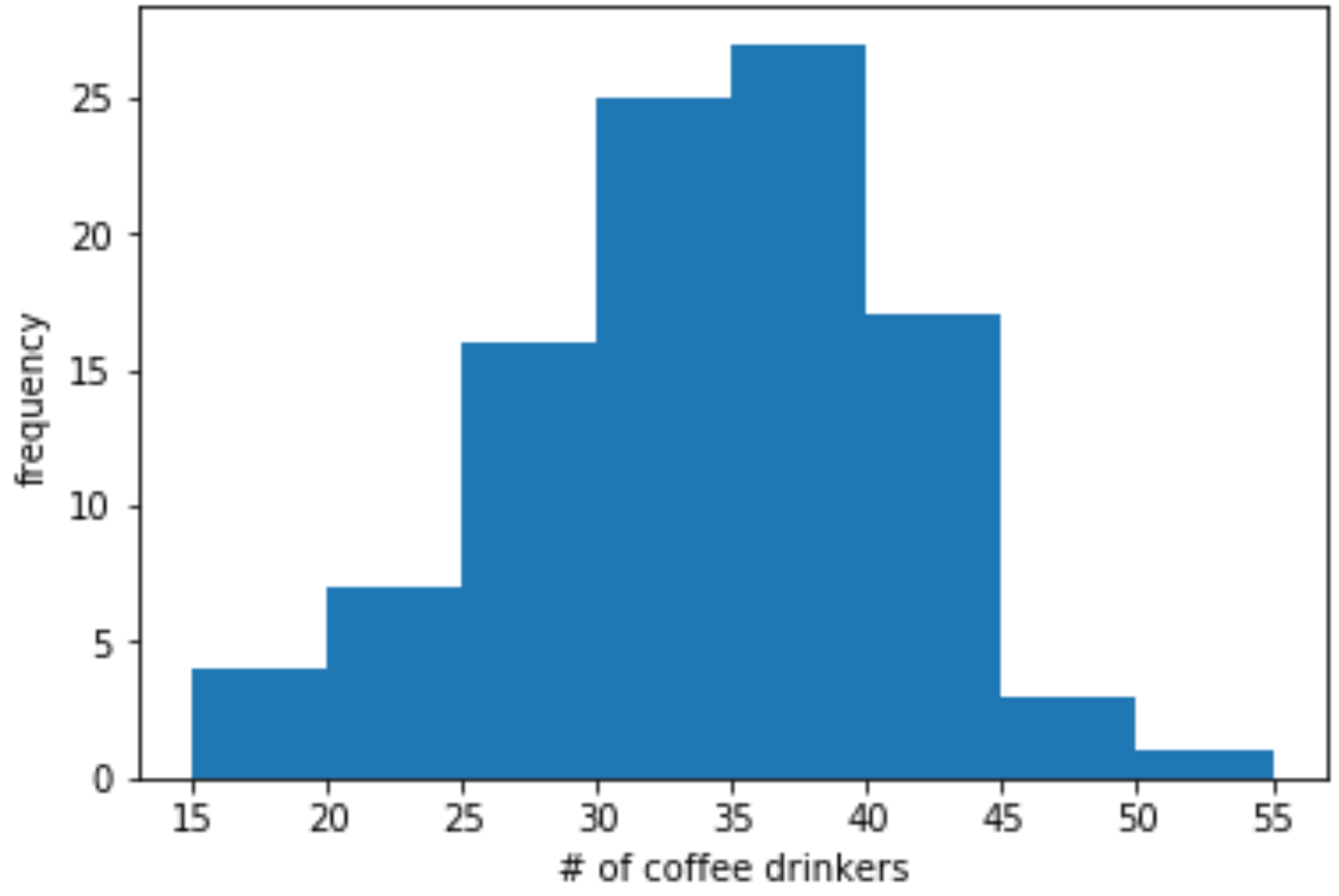
- Often, count is also referred to as frequency
  - The y-axis numbers telling us what exactly is plotted
  - (More details on later slides)




```
_ = plt.hist(data, bins=8, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel('frequency')
```

repeating the experiment

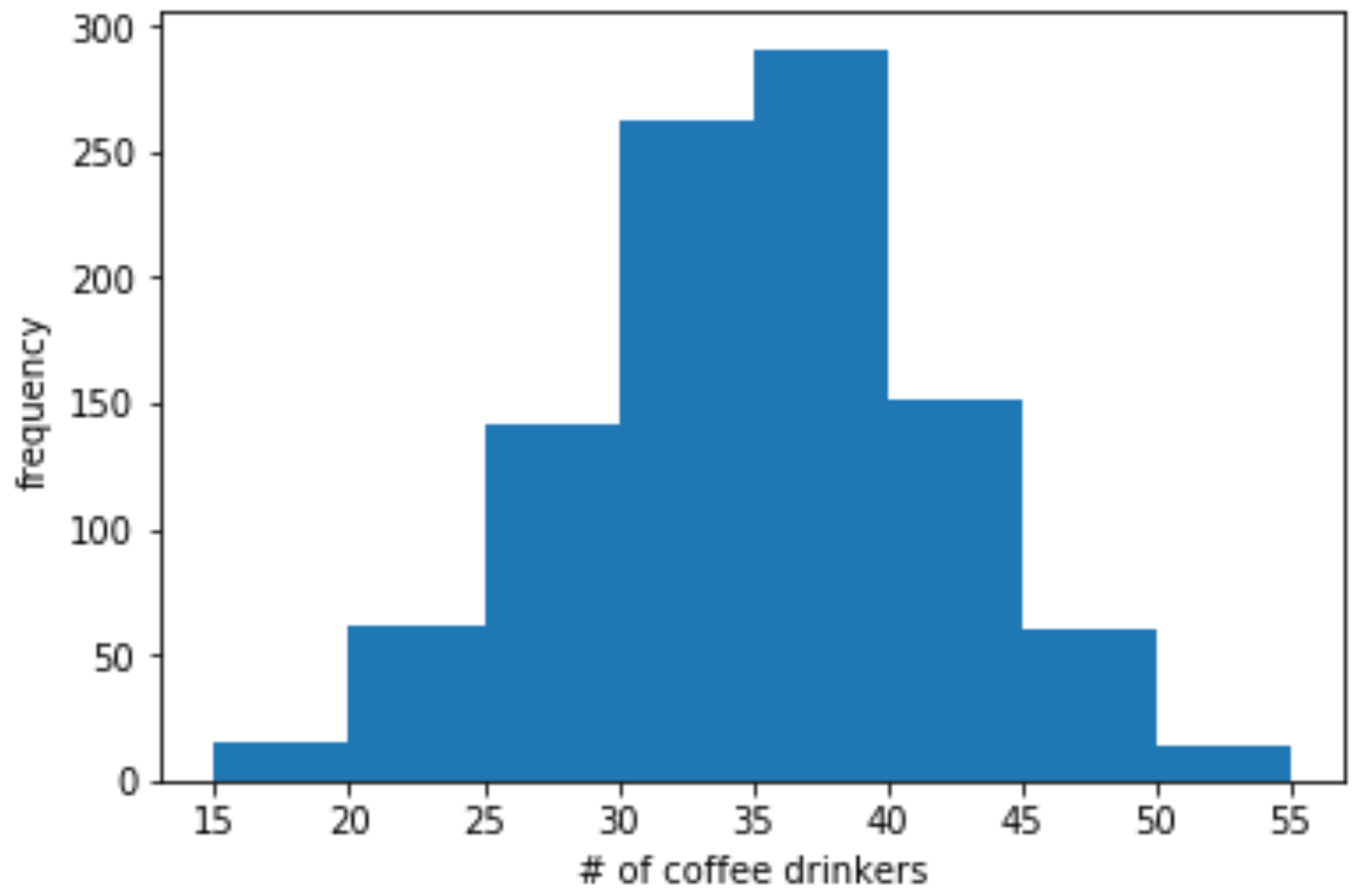
- Remember: This histogram comes from observed data
- If we repeat the experiment, we might not get the same histogram!
  - In fact, there will almost surely be some difference at this sample size
- This is because what we have is a **sample** of the true distribution




```
_ = plt.hist(data, bins=8, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel('frequency')
```

collecting a larger sample

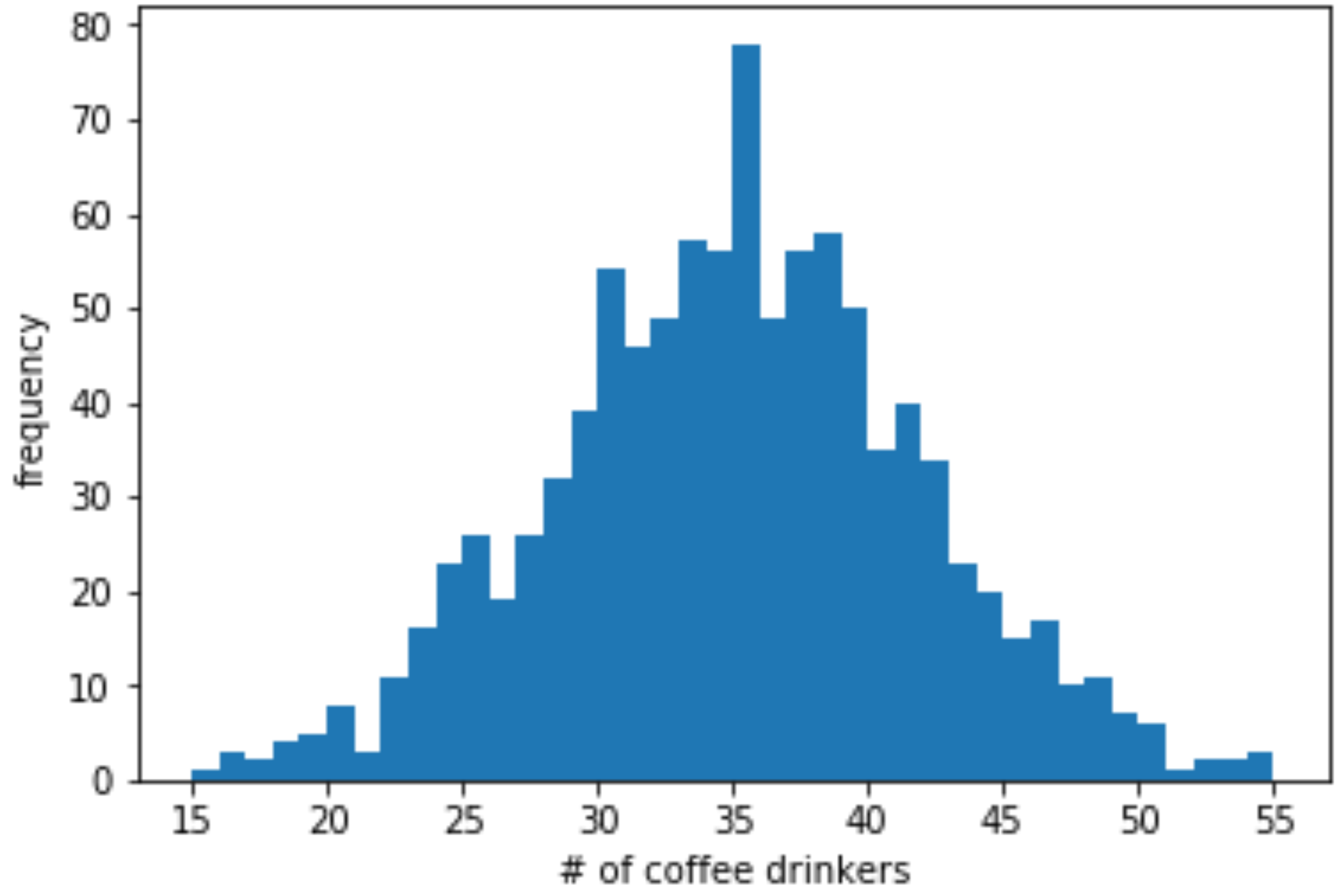
- Suppose we collect 1000 observations instead of 100
- The result on the right looks basically the same!
- Using the same number of bins
  - Each bin has more observations in it
  - But the relative frequencies are not changing much
- But now that we have a larger sample, we can add more bins to see a *finer granularity* of the distribution




```
_ = plt.hist(data, bins=8, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel('frequency')
```

adding more bins

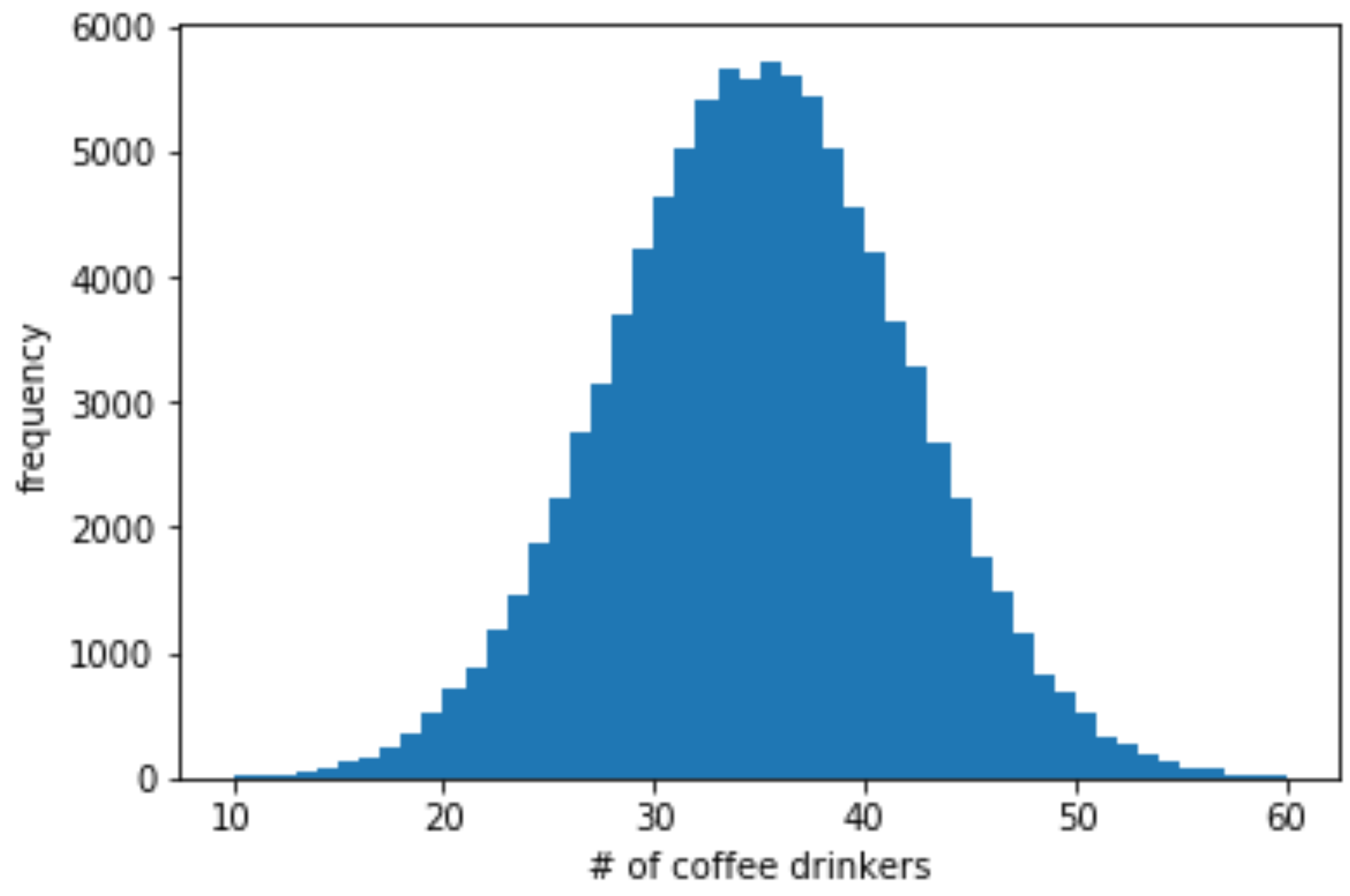
- This looks better!
- Gives us a good sense of what the data looks like, and what the underlying distribution is
- What would happen if we used more than 40 bins here?




```
_ = plt.hist(data, bins=40, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel('frequency')
```

adding even more data

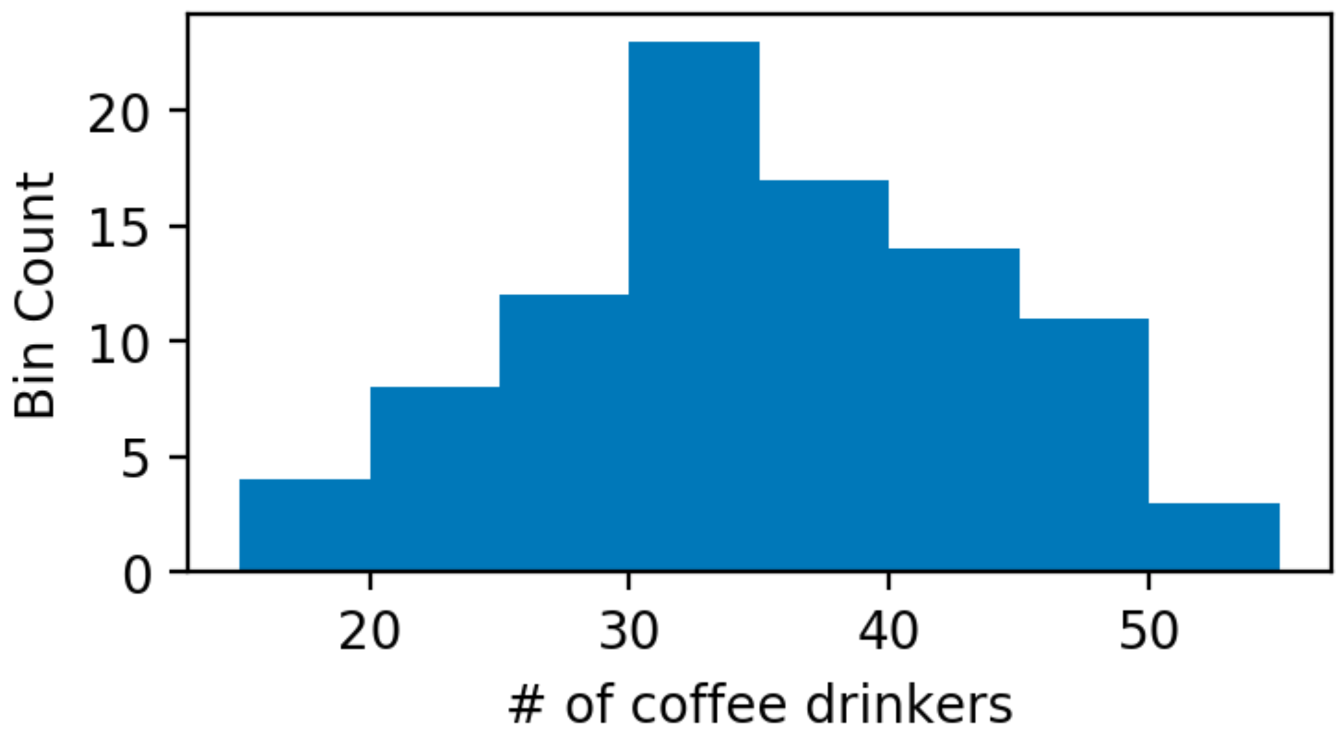
- This looks even better!
- As we add more data points, our histogram looks more and more like the "true" shape of the underlying distribution
  - We'll get in to what this means when we talk about distributions and sampling

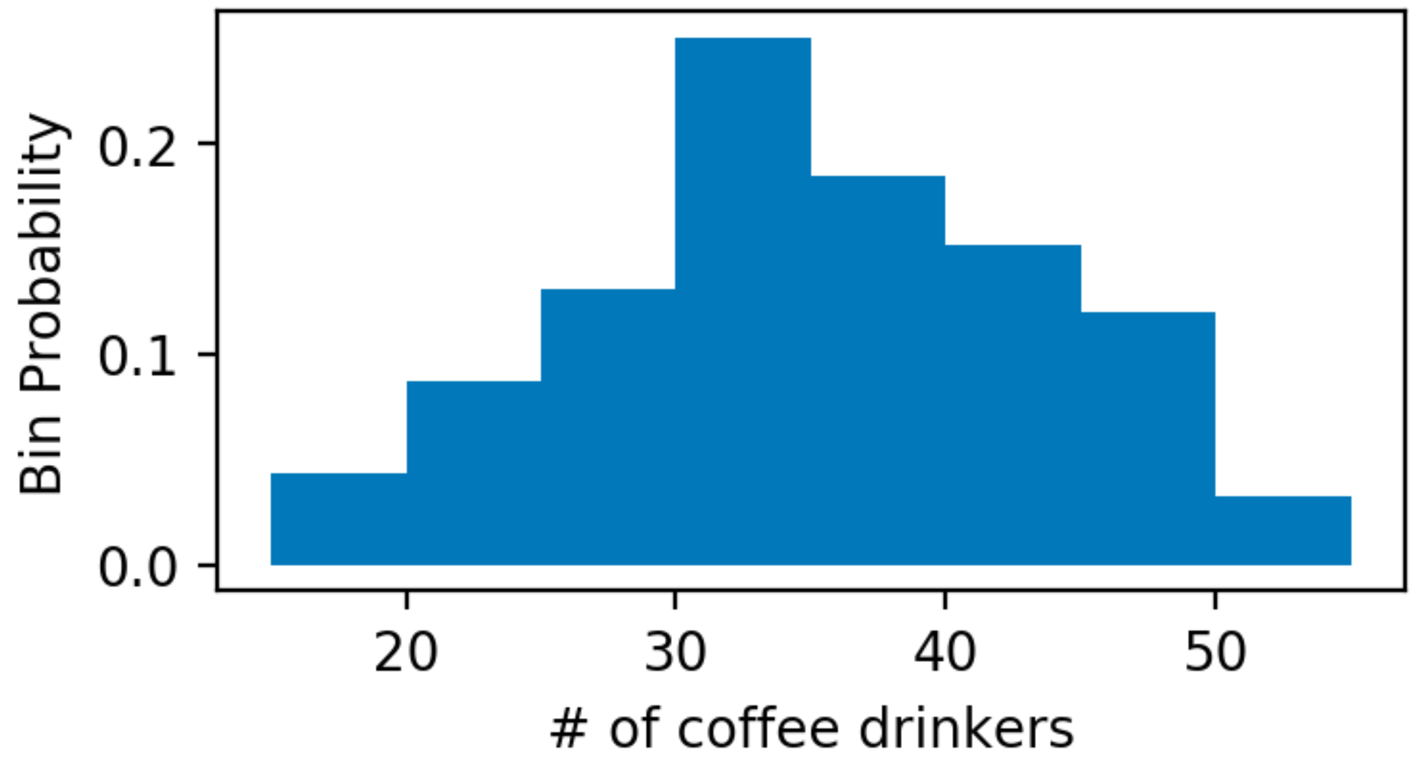


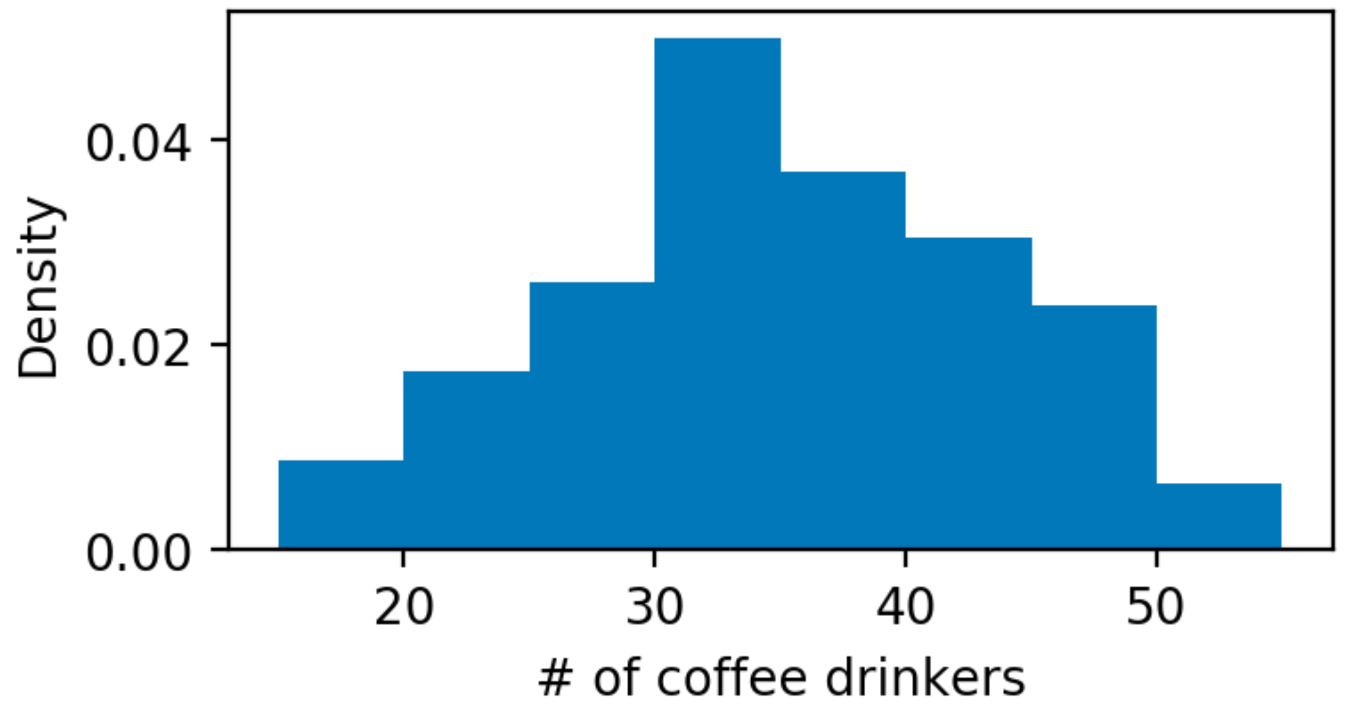

```
_ = plt.hist(data, bins=40, range=(15,55))
plt.xlabel('# of coffee drinkers')
plt.ylabel('frequency')
```

histogram bin normalization

- **Count** - y-axis is the count in each bin, denoted x_k
  - $\sum_{k=1}^{n} x_k = m$, sum of all bins is total number of samples m
- **Probability** - y-axis is probability for each bin, denoted $\hat{p}_k = \frac{x_k}{\sum_l x_l}$
  - $\sum_k \hat{p}_k = 1$, sum of all bin probabilities is 1
- **Density** - y-axis is normalized by both probability and **bin width**, $\hat{d}_k = \frac{\hat{p}_k}{w}$
  - So $\sum_k w \cdot \hat{d}_k = 1$, i.e., the **area under the curve** is 1
- “Frequency” can be used for both “count” and “probability” above



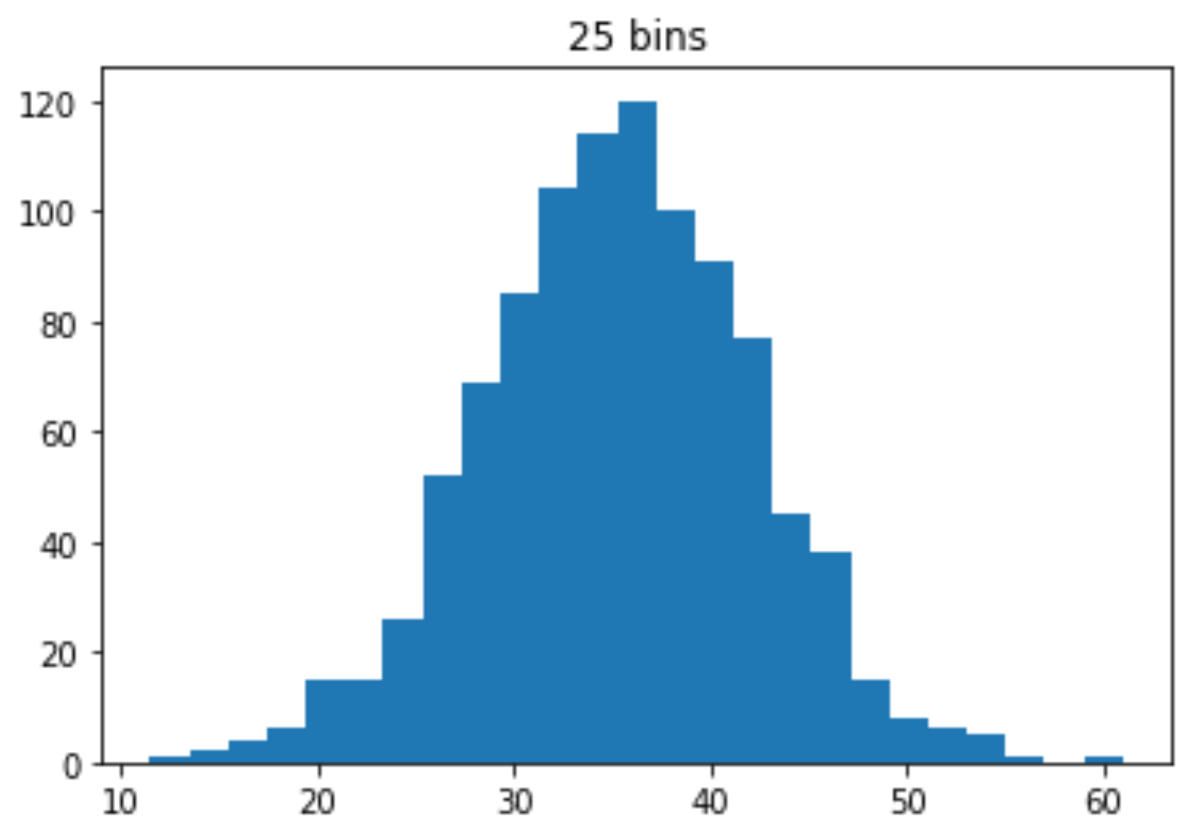


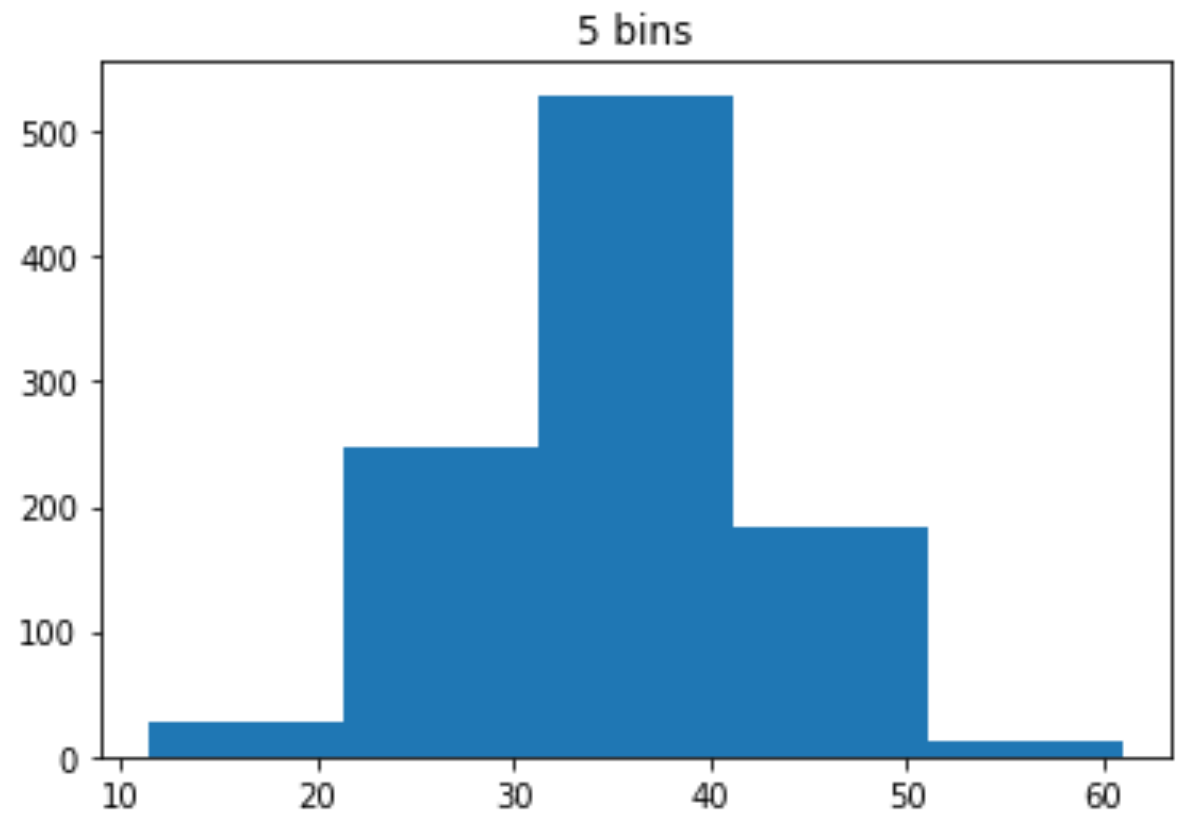


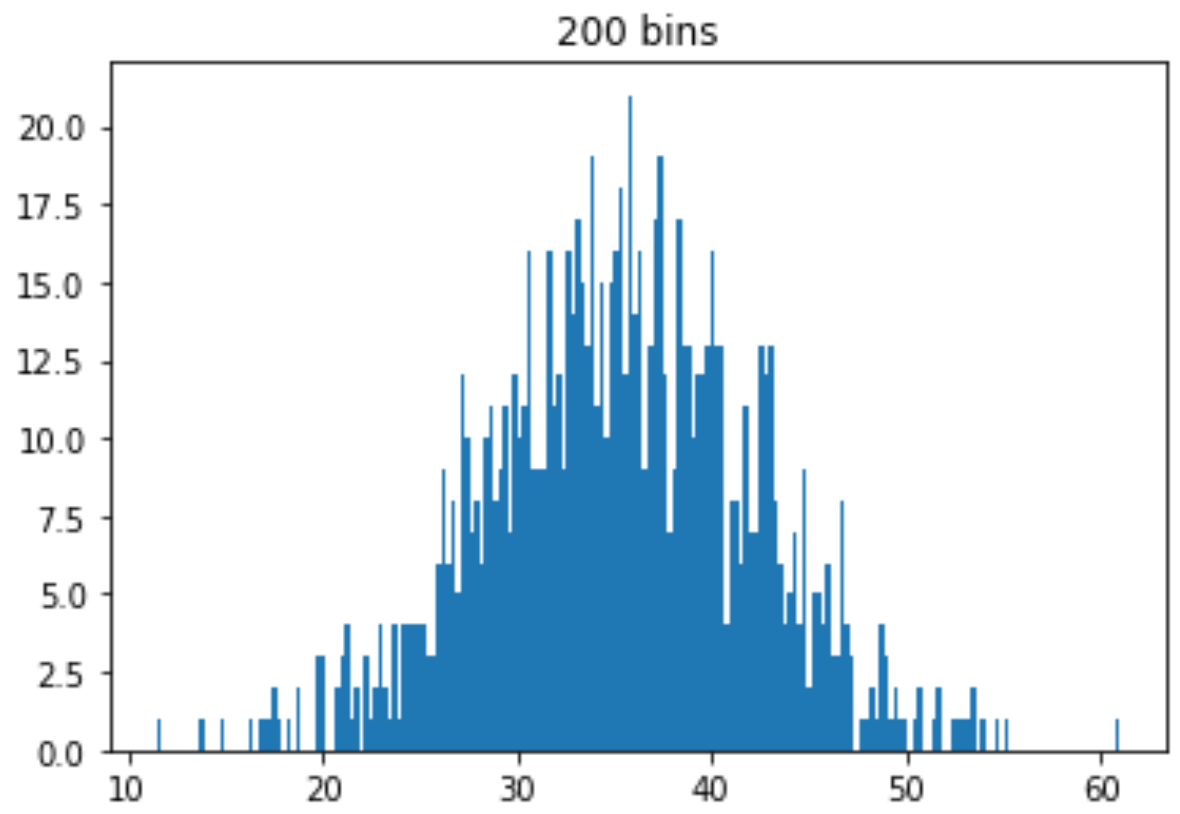

```
_ = plt.hist(data, bins=8,
range=(15,55), density='True')
```

choice of bins

- The histogram has a few parameters
  - Number of bins n, width of bins w, and even number of samples m can be viewed as one
  - Bins don't even have to be homogeneous
- Several formulas have been proposed for choosing n and w based on the sample
  - Square root: $n = \lceil\sqrt{m}\rceil$
  - Sturges' formula: $n = \lceil\log_2 m\rceil + 1$
  - Rice rule: $n = \lceil 2m^{1/3}\rceil$
  - Scott's normal reference rule: $w = 3.5\hat{\sigma}/m^{1/3}$
- How do we reason about the "optimal" choice?

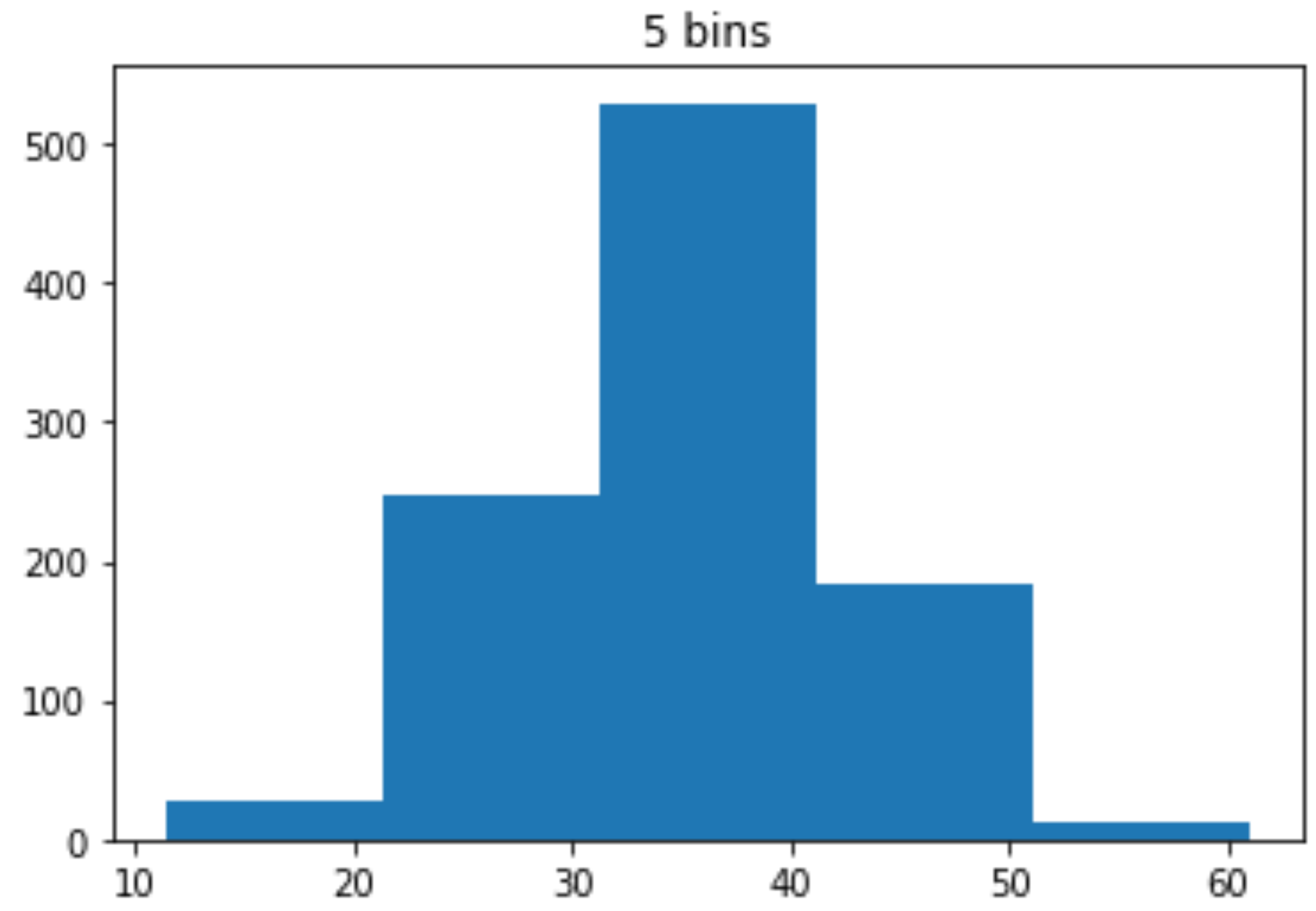


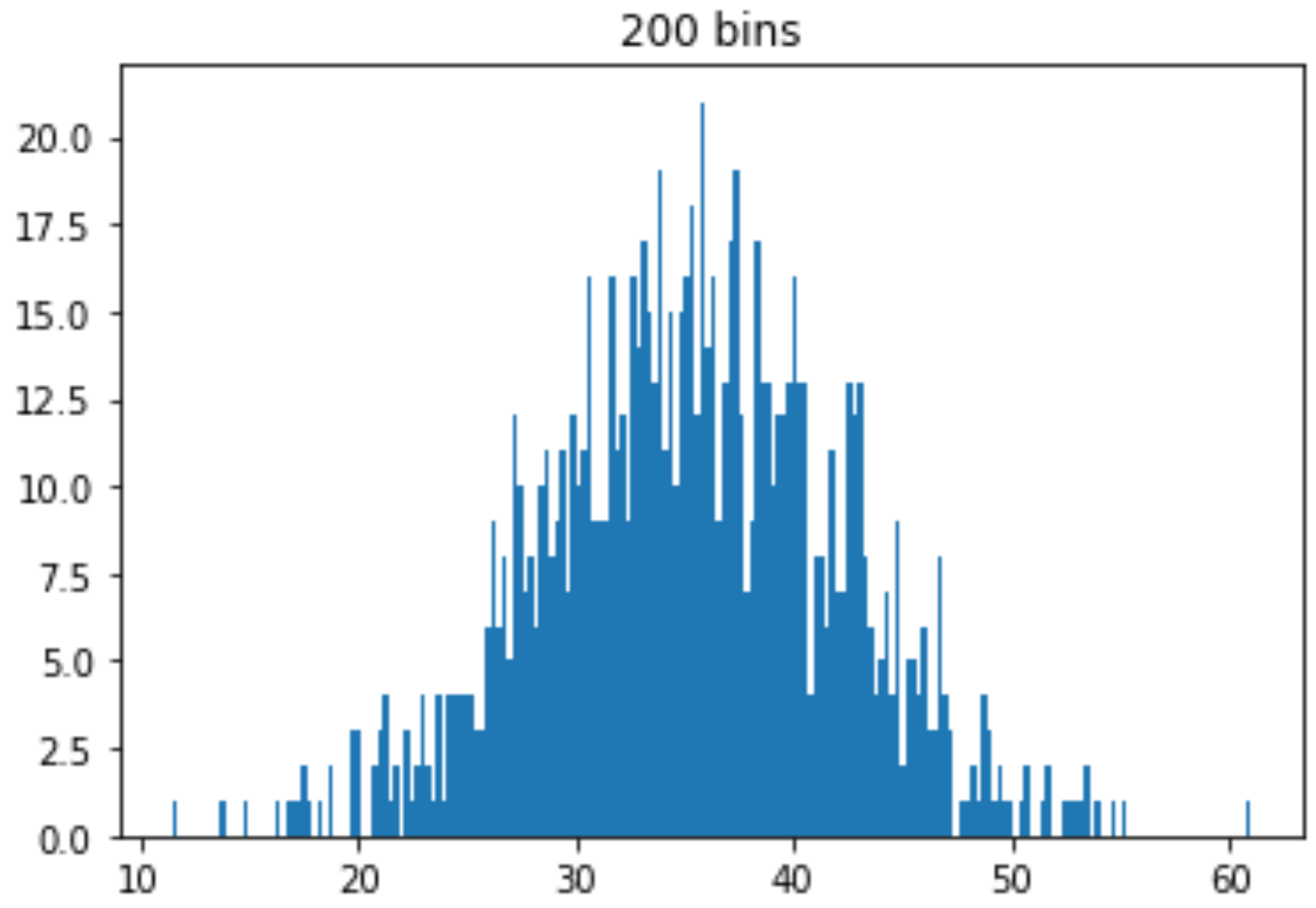




bin width intuition

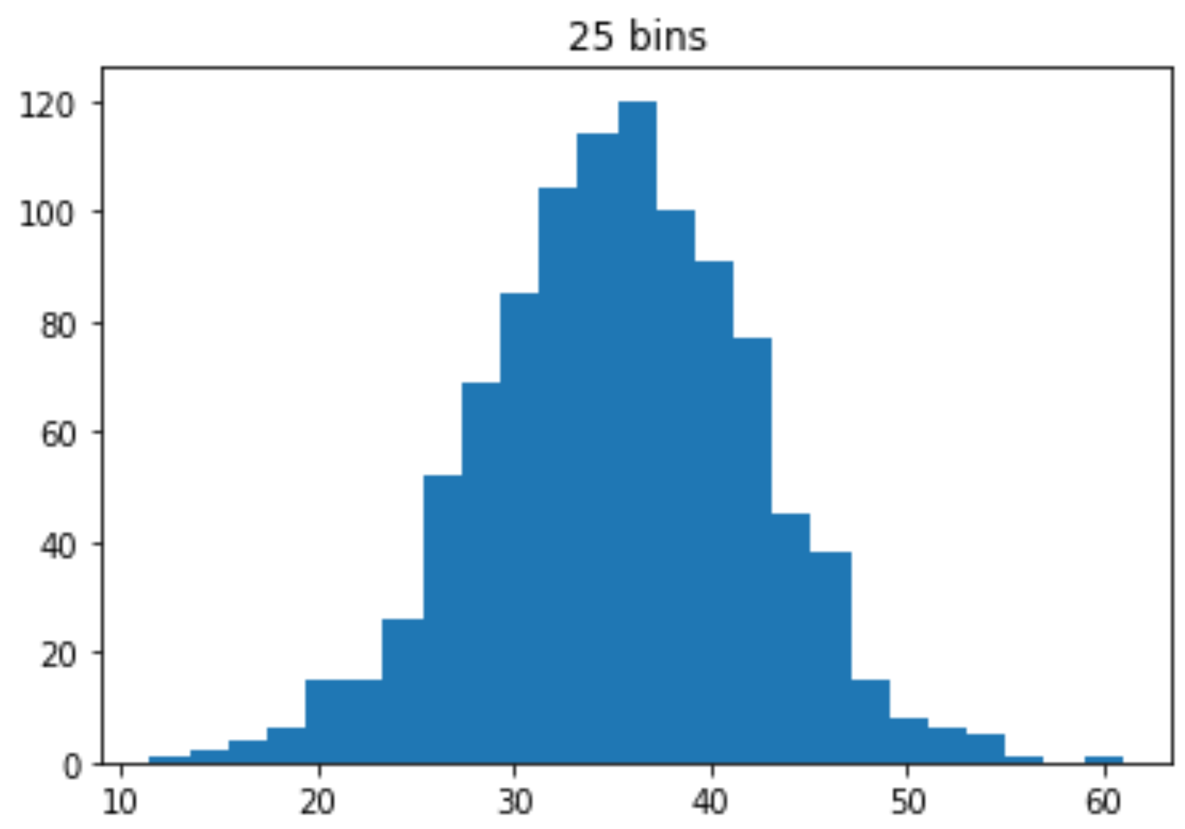
- Choosing large bin size w
  - Broad range of points (some rare, some common) put into the same bin and given the same estimate
- Choosing small bin size w
  - Each bin is based on fewer samples, so harder to estimate how likely the bin is
  - In the limit: Buckets of size 0 (is it practical?)
- So how do we choose the bin size in general?

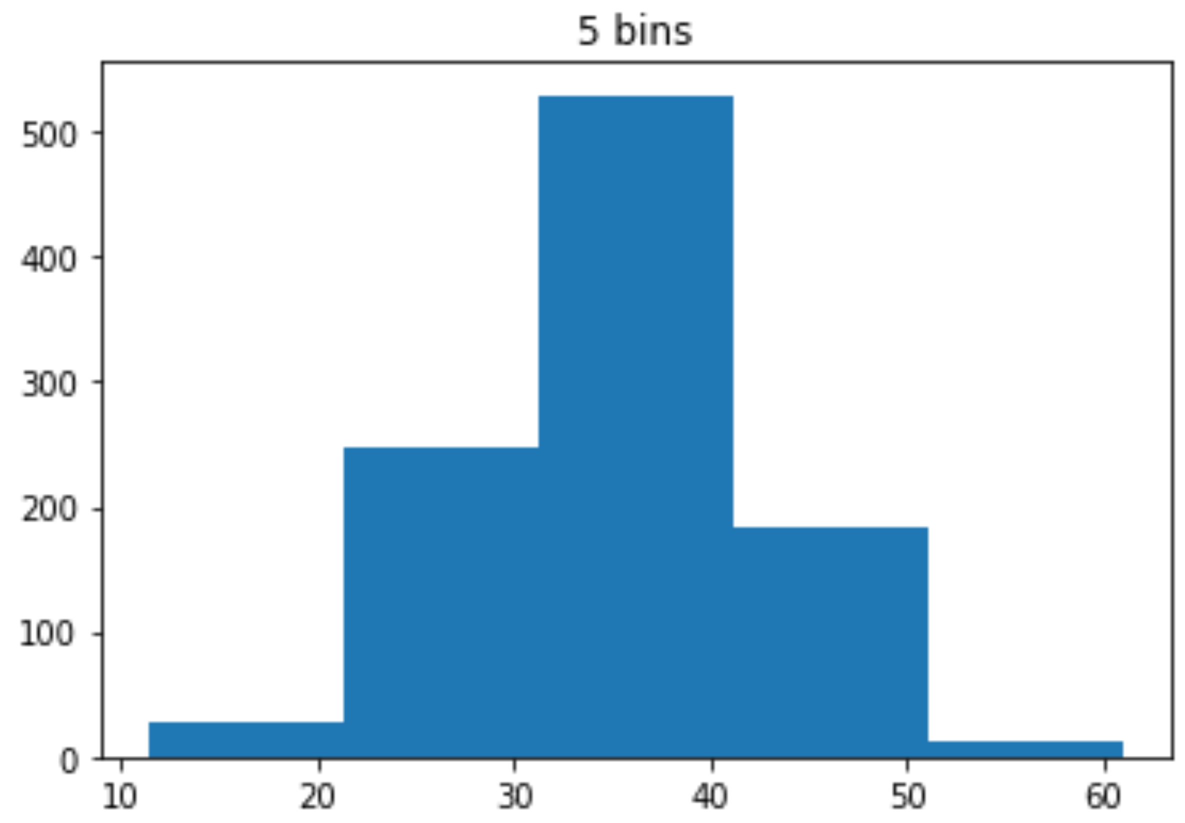


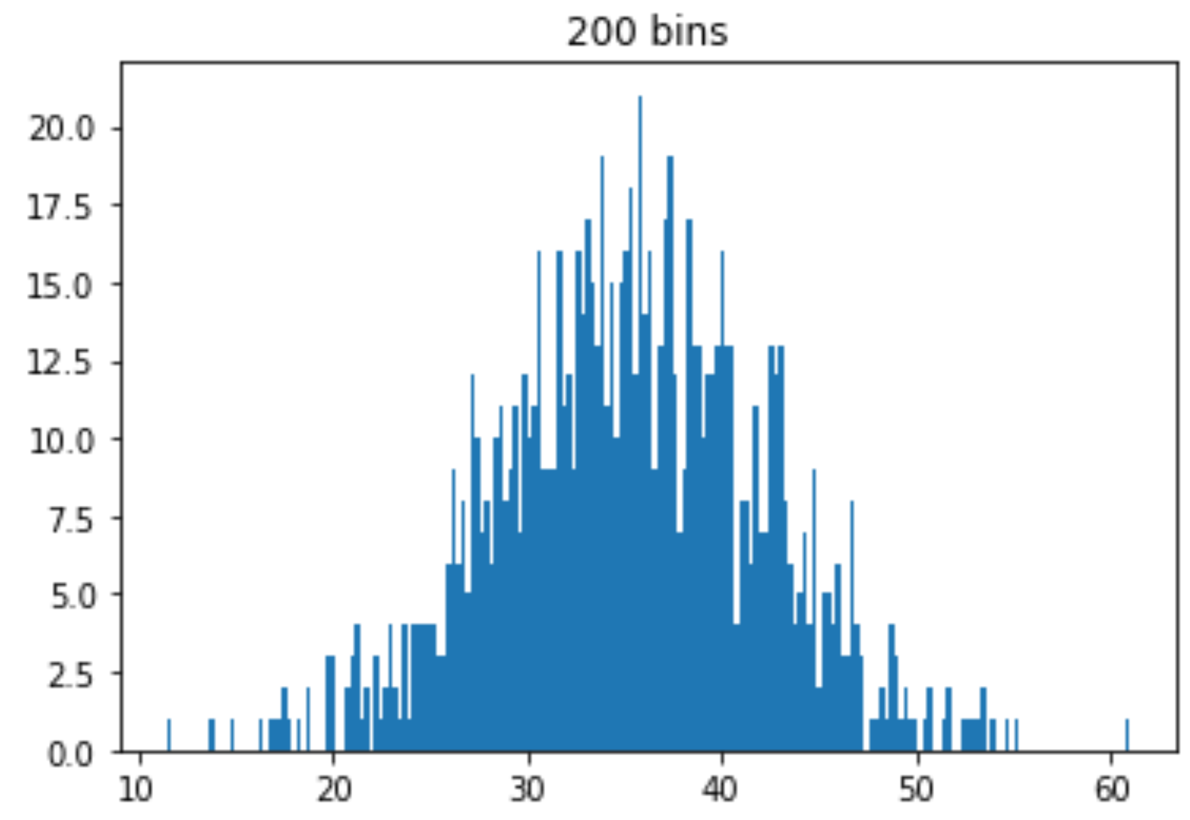


evaluation of histograms

- We can choose many different bin widths w (or equivalently the number of bins n)
- How do we **evaluate** which bin width w is better?
  - *Visual appeal* - Which is most visually appealing to humans?
  - *Usefulness* - Which helps the owner know how much coffee to make?
  - *Mathematical metrics* - Which satisfies some mathematical notion of goodness? (Ideally this is tied to *usefulness*)
- We will focus on **mathematical metrics**

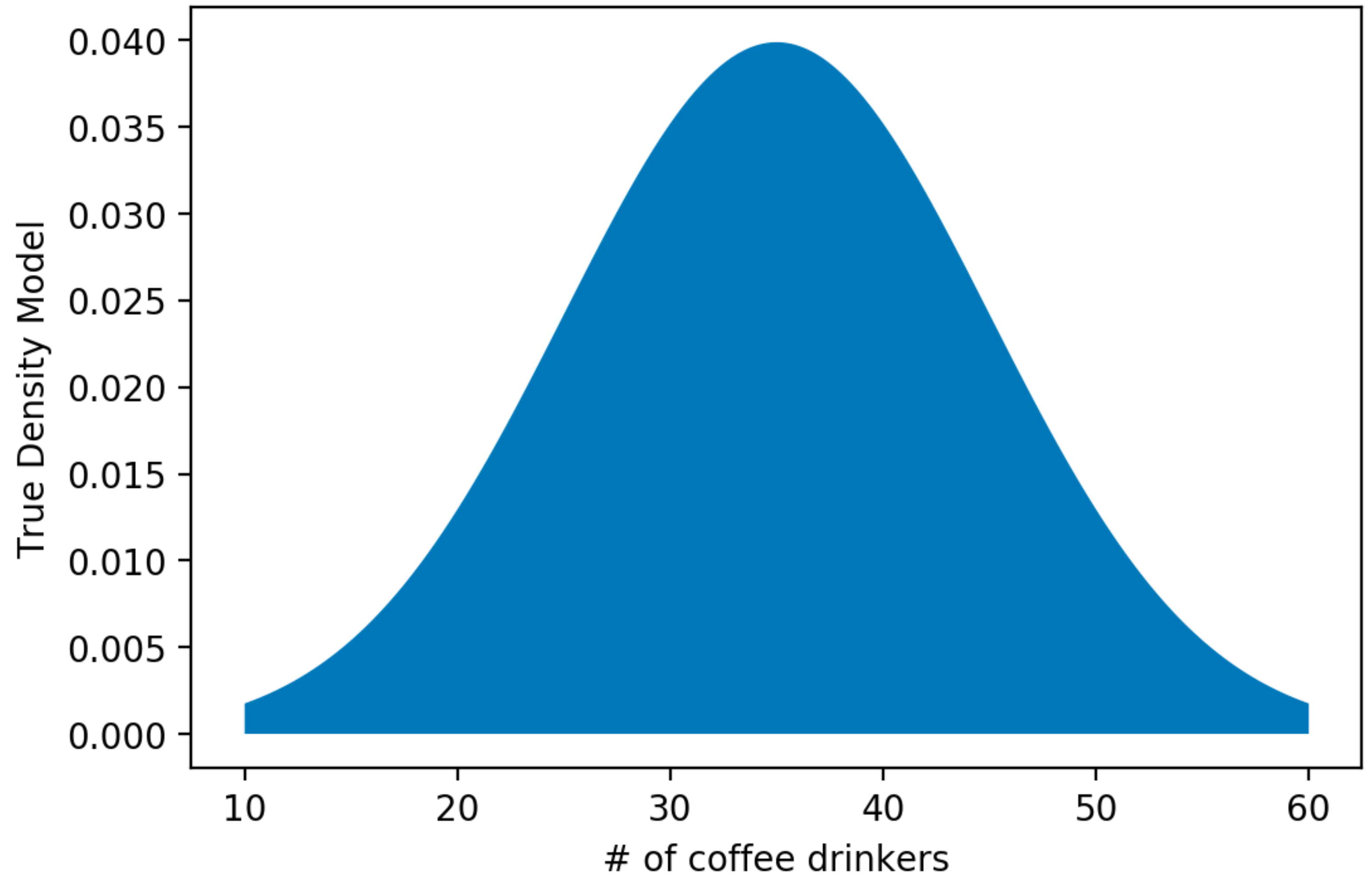


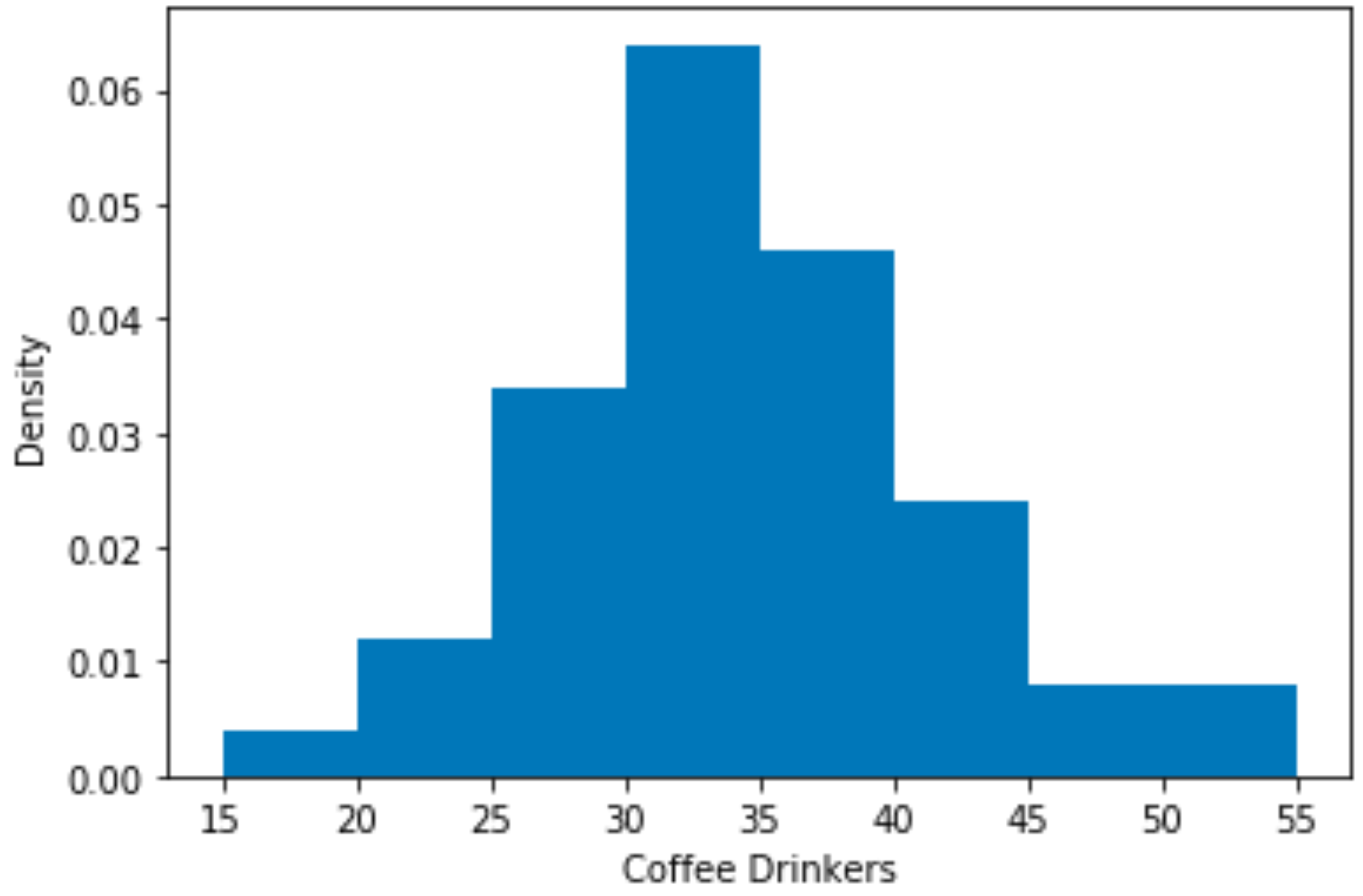




estimated vs. "true" model

- First, we *assume* there is some **"true" underlying model** (often denoted by $f(x)$) for the phenomena of interest
    - Importantly, this "true" model is **unknown** (or **hidden**)
    - For example, we don't know before collecting data the distribution of coffee purchases
    - Even after collecting data, we can only **estimate** the distribution
- Histograms are an **estimate** (or **approximation**, often denoted by $\hat{f}(x)$) of the true distribution



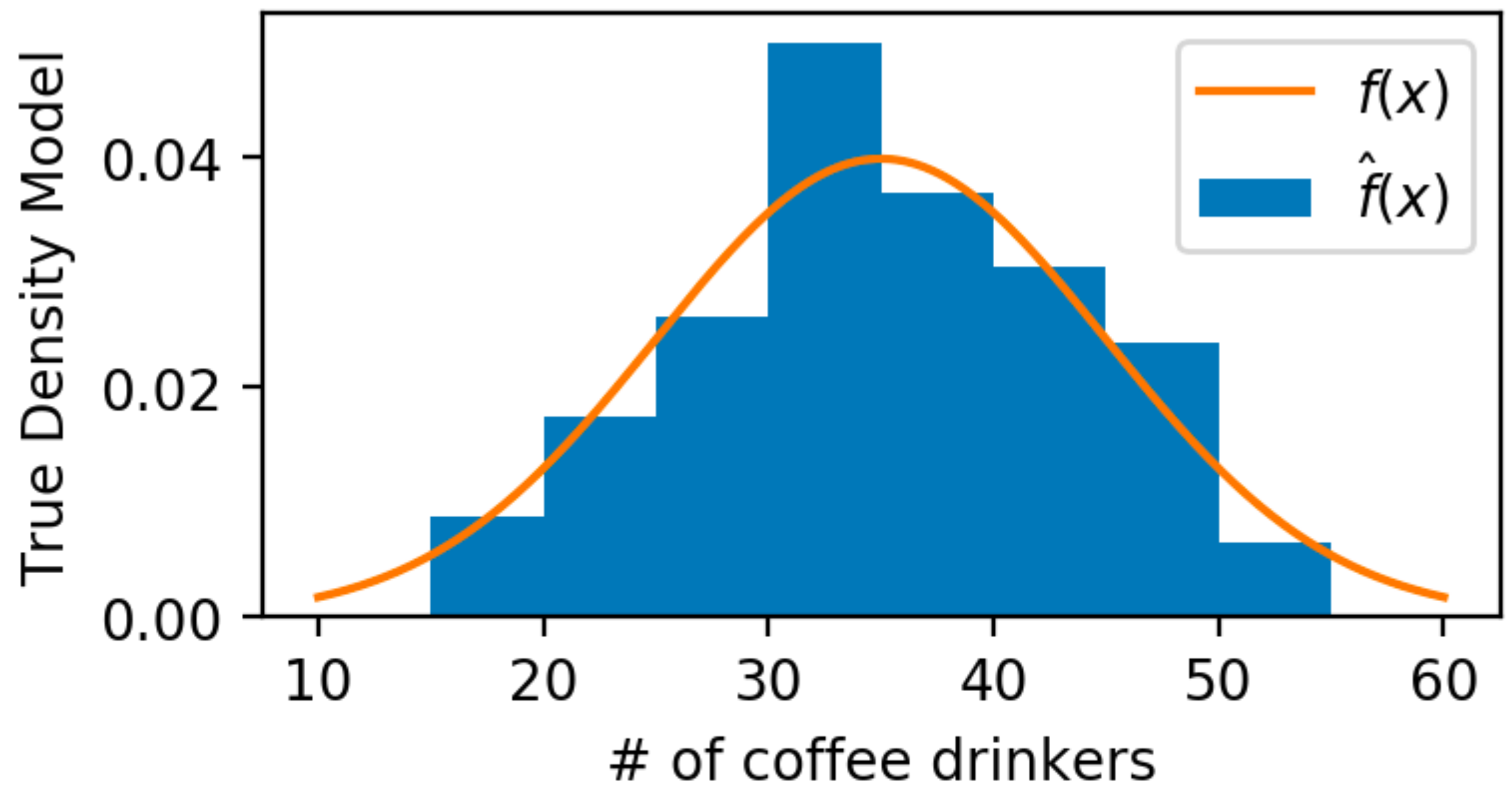


minimizing the estimation error

- We can pick the bin size w that minimizes the error of estimating a point
- The **Integrated Square Error** (**ISE**) of a histogram can be written as a function of the bin width (i.e., the smoothing parameter)

$$L(w) = \int \left(\hat{f}_m(x) - f(x)\right)^2 dx$$

  - Here, $\hat{f}_m(x)$ is the density estimate of the histogram with m samples
  - However, $f(x)$ is the "true" but *unknown* model, so how do we compute $L(w)$?



estimating the error with samples

- The Integrated Square Error (ISE):

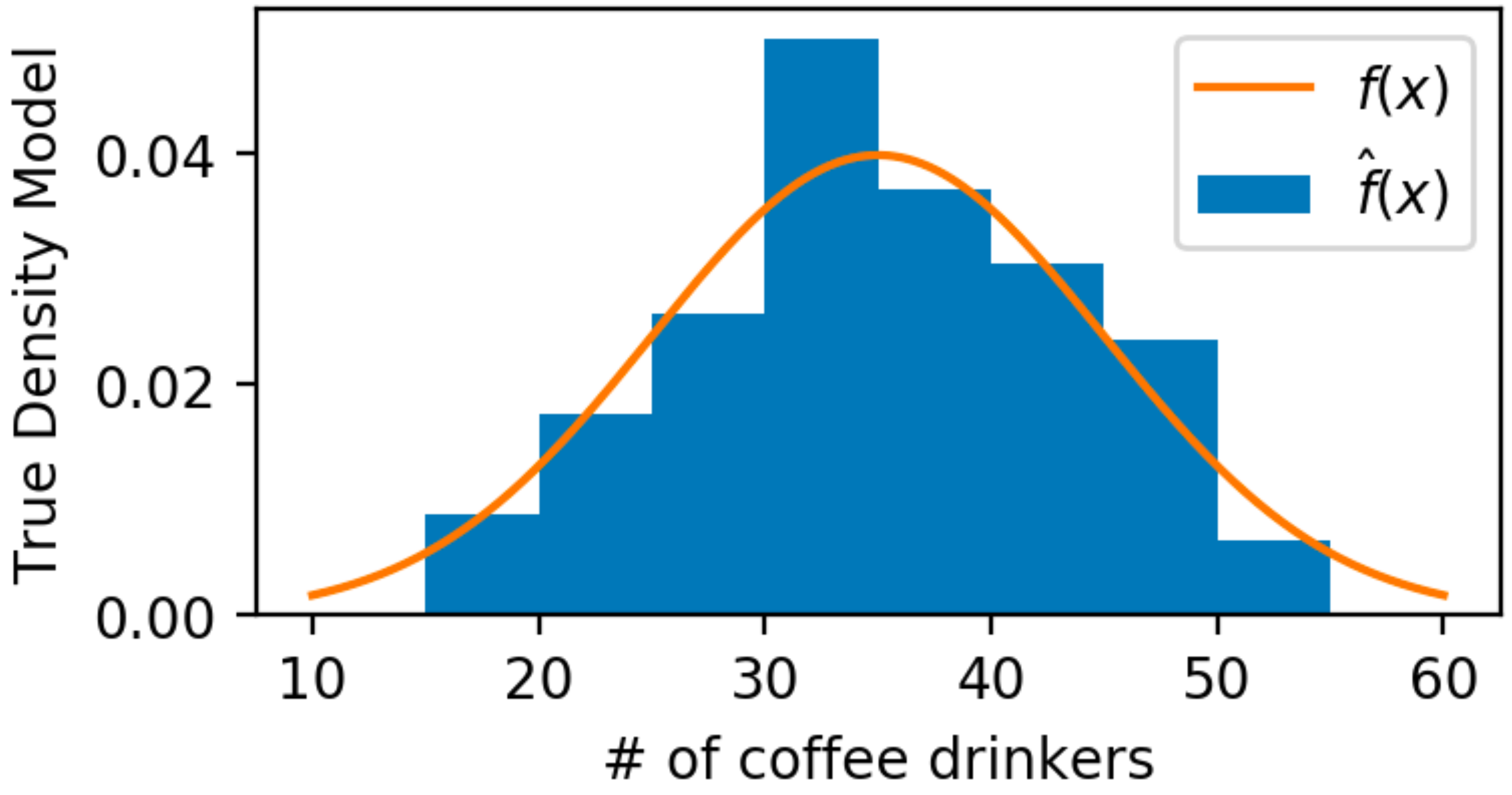
$$L(w) = \int \left(\hat{f}_m(x) - f(x)\right)^2 dx$$

- We can approximate with data samples by $L(w) \approx J(w) + constant$, where

$$J(w) = \frac{2}{(m-1)w} - \frac{m+1}{(m-1)w}(\hat{p}_1^2 + \hat{p}_2^2 + \cdots + \hat{p}_n^2)$$

  - w is bin width, m is the number of samples and $\hat{p}_k, k = 1, \ldots, n$ are the bin probabilities

- We can choose the "optimal" bin width by minimizing $J(w)$, which approximates $L(w)$!



minimizing $J(w)$

- The brute-force way is to try as many values of w as possible and choose the best
- Better to work with n here in this case, since there is a finite number of possibilities
- For each $n = 1, \ldots, m$:
  - calculate w
  - use this to calculate J

  Plot the results, choose the best one
- To narrow down the number of values we need to try, **grid search** procedures are also possible

Testing all numbers of bins