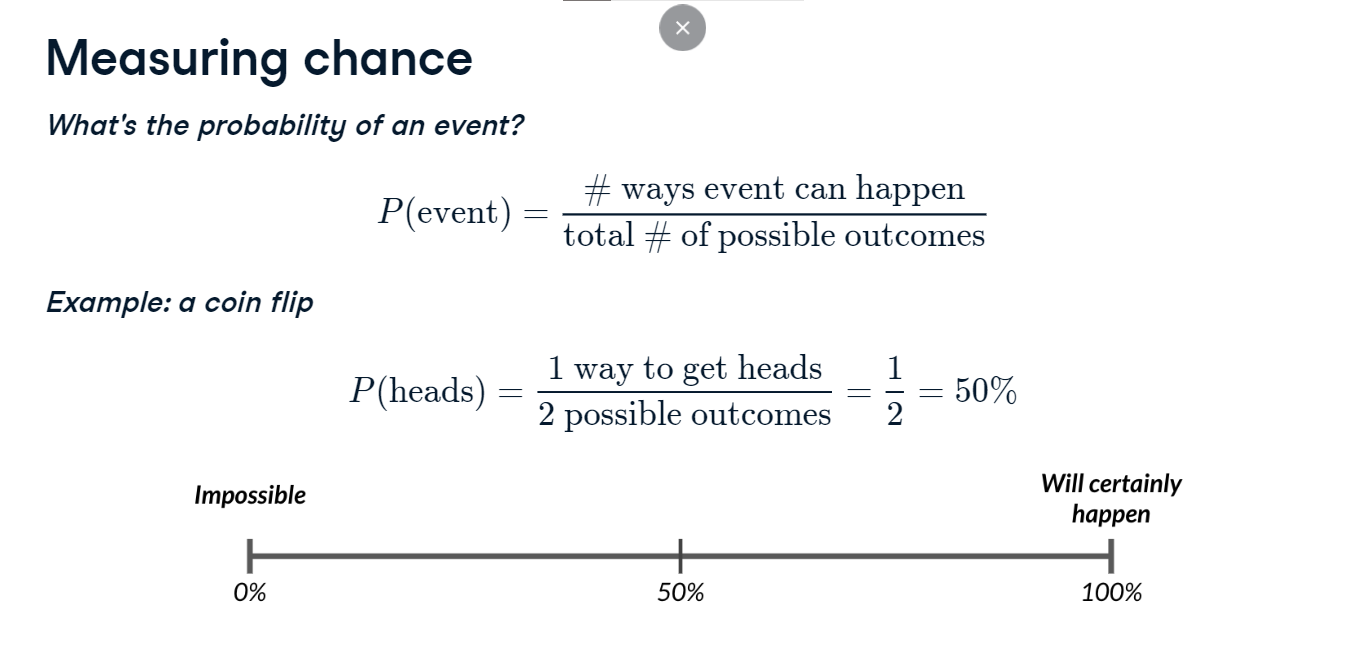
**What are the chances?**

People talk about chance pretty frequently, like what are the chances of closing a sale, of rain tomorrow, or of winning a game? But how exactly do we measure chance?

**Measuring chance**

We can measure the chances of an event using probability. We can calculate the probability of some event by taking the number of ways the event can happen and dividing it by the total number of possible outcomes. For example, if we flip a coin, it can land on either heads or tails. To get the probability of the coin landing on heads, we divide the 1 way to get heads by the two possible outcomes, heads and tails. This gives us one half, or a fifty percent chance of getting heads. Probability is always between zero and 100 percent. If the probability of something is zero, it's impossible, and if the probability of something is 100%, it will certainly happen.



**Assigning salespeople**

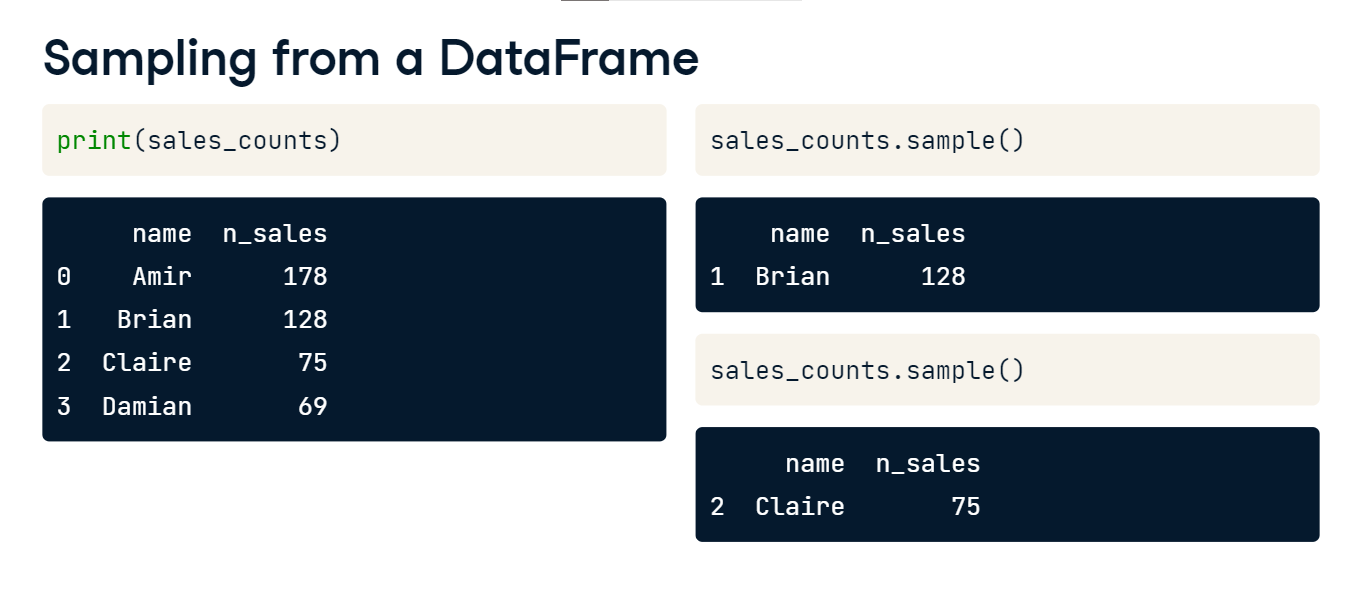
Let's look at a more complex scenario. There's a meeting coming up with a potential client, and we want to send someone from the sales team to the meeting. We'll put each person's name on a ticket in a box and pull one out randomly to decide who goes to the meeting.

**Assigning salespeople**

Brian's name gets pulled out. The probability of Brian being selected is one out of four, or 25%.

**. Sampling from a DataFrame**

We can recreate this scenario in Python using the sample() method. By default, it randomly samples one row from the DataFrame. However, if we run the same thing again, we may get a different row since the sample method chooses randomly. If we want to show the team how we picked Brian, this won't work well.



**Setting a random seed**

To ensure we get the same results when we run the script in front of the team, we'll set the random seed using np-dot-random-dot-seed. The seed is a number that Python's random number generator uses as a starting point, so if we orient it with a seed number, it will generate the same random value each time. The number itself doesn't matter. We could use 5, 139, or 3 million. The only thing that matters is that we use the same seed the next time we run the script. Now, we, or one of the sales-team members, can run this code over and over and get Brian every time.



**A second meeting**

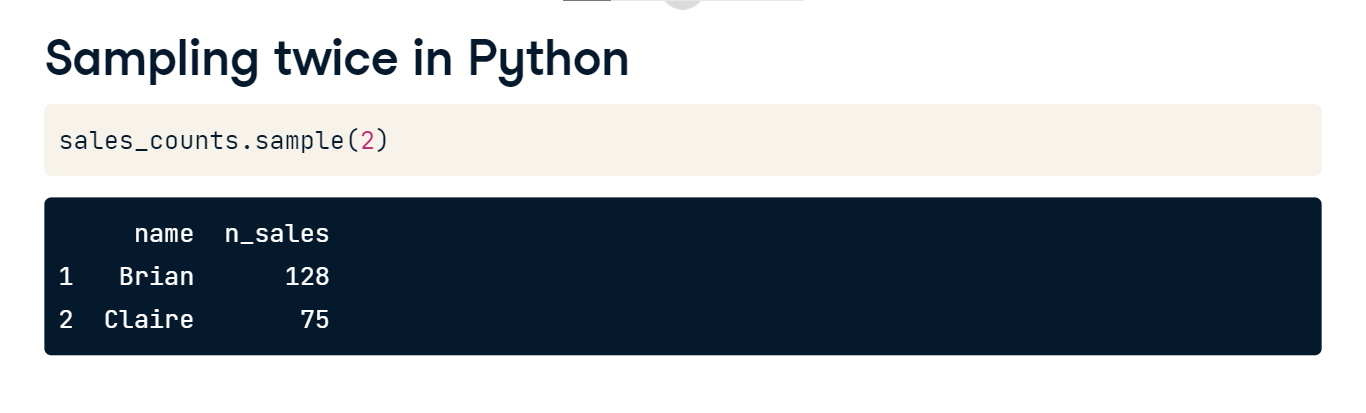
Now there's another potential client who wants to meet at the same time, so we need to pick another salesperson. Brian haas already been picked and he can't be in two meetings at once, so we'll pick between the remaining three. This is called sampling without replacement, since we aren't replacing the name we already pulled out.

**A second meeting**

This time, Claire is picked, and the probability of this is one out of three, or about 33%.

**Sampling twice in Python**

To recreate this in Python, we can pass 2 into the sample method, which will give us 2 rows of the DataFrame.



**Sampling with replacement**

Now let's say the two meetings are happening on different days, so the same person could attend both. In this scenario, we need to return Brian's name to the box after picking it. This is called sampling with replacement.

**Sampling with replacement**

Claire gets picked for the second meeting, but this time, the probability of picking her is 25%.

**Sampling with/without replacement in Python**

To sample with replacement, set the replace argument to True, so names can appear more than once. If there were 5 meetings, all at different times, it's possible to pick some rows multiple times since we're replacing them each time.

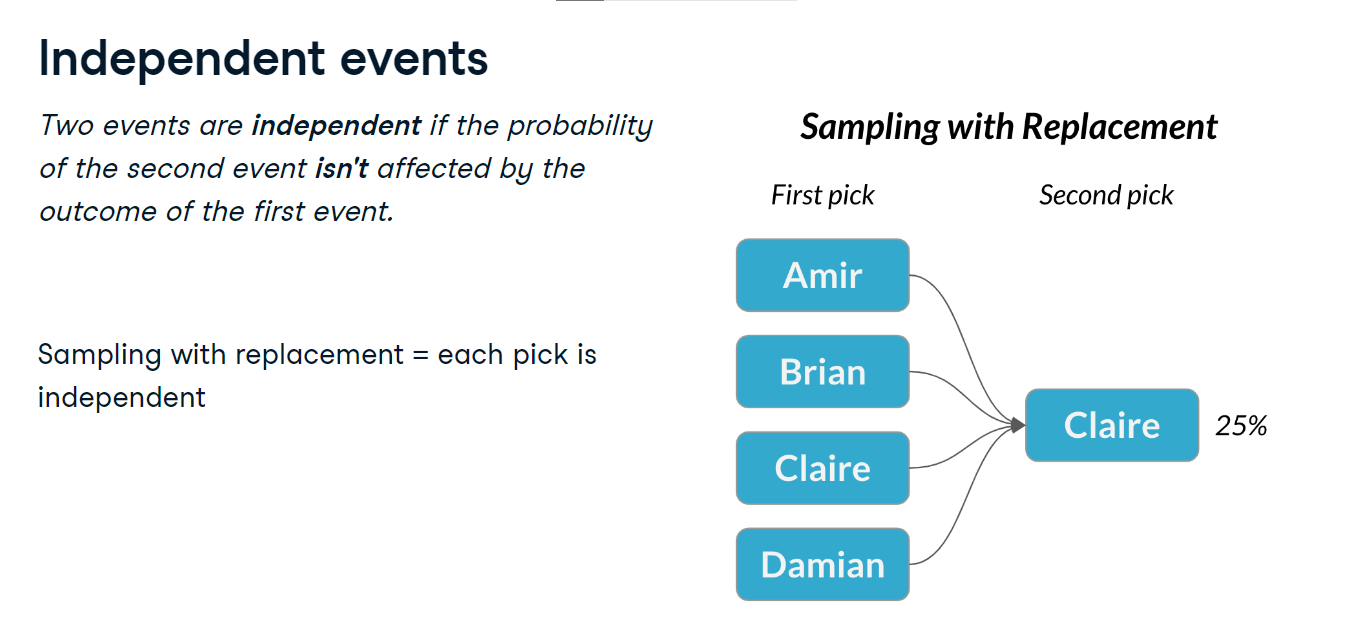


**Independent events**

Let's quickly talk about independence. Two events are independent if the probability of the second event isn't affected by the outcome of the first event. For example, if we're sampling with replacement, the probability

**Independent events**

that Claire is picked second is 25%, no matter who gets picked first. In general, when sampling with replacement, each pick is independent.



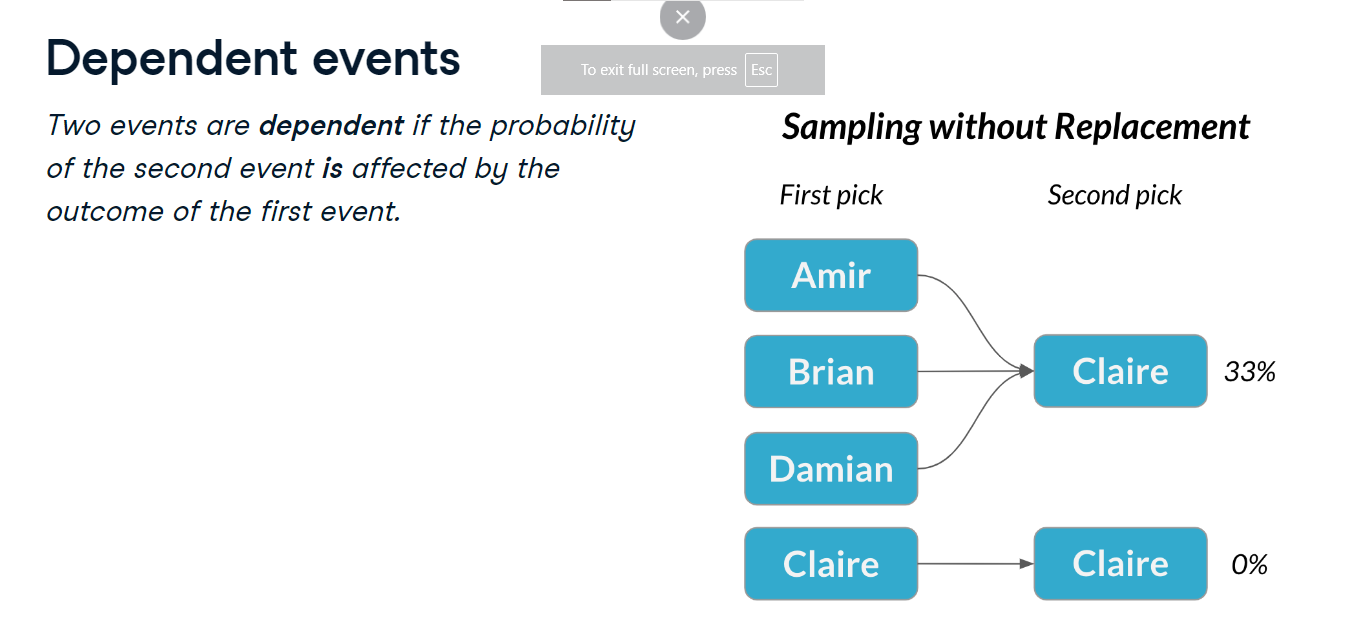
**Dependent events**

Similarly, events are considered dependent when the outcome of the first changes the probability of the second. If we sample without replacement, the probability that Claire is picked second depends on who gets picked first.

**Dependent events**

If Claire is picked first, there's 0% probability that Claire will be picked second.

If someone else is picked first, there's a 33% probability Claire will be picked second. In general, when sampling without replacement, each pick is dependent.



**Discrete distributions**

In this lesson, we'll take a deeper dive into probability and begin looking at probability distributions.

**Rolling the dice**

Let's consider rolling a standard, six-sided die.

**Rolling the dice**

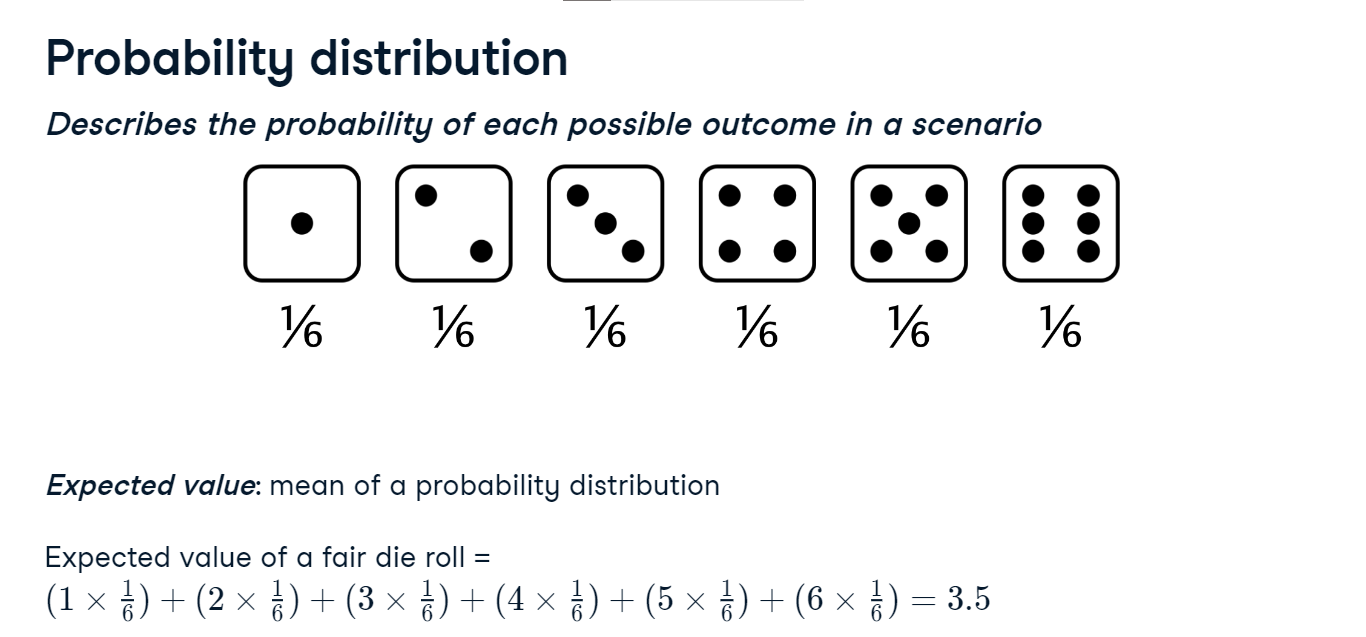
There are six numbers, or six possible outcomes, and every number has one sixth, or about a 17 percent chance of being rolled. This is an example of a probability distribution.

**Choosing salespeople**

This is similar to the scenario from earlier, except we had names instead of numbers. Just like rolling a die, each outcome, or name, had an equal chance of being chosen.

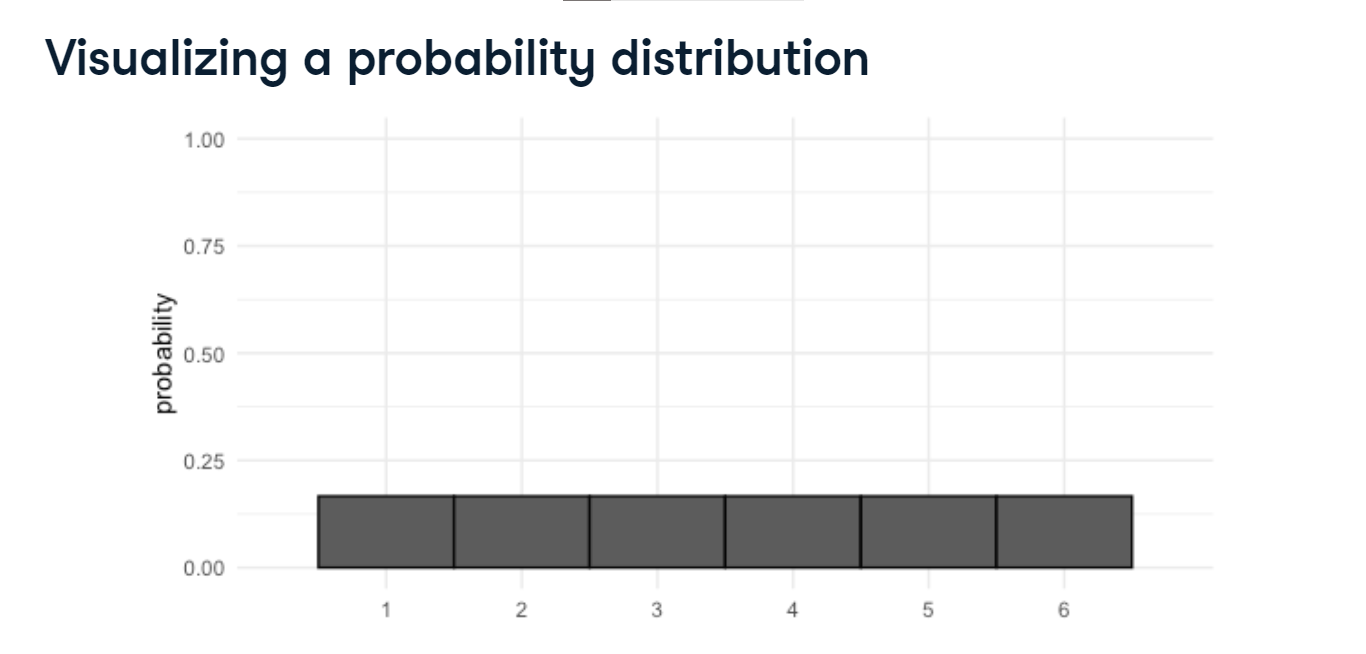
**Probability distribution**

A probability distribution describes the probability of each possible outcome in a scenario. We can also talk about the expected value of a distribution, which is the mean of a distribution. We can calculate this by multiplying each value by its probability (one sixth in this case) and summing, so the expected value of rolling a fair die is 3-point-5.



**Visualizing a probability distribution**

We can visualize this using a barplot, where each bar represents an outcome, and each bar's height represents the probability of that outcome.

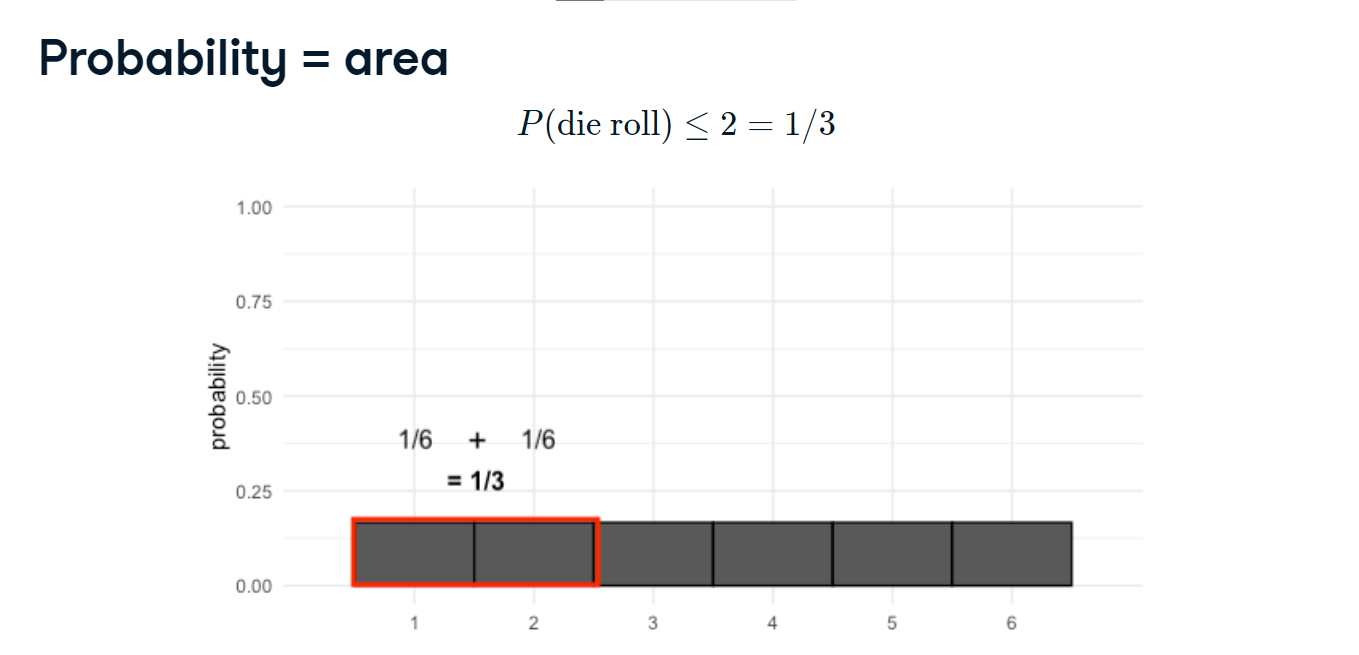


**Probability = area**

We can calculate probabilities of different outcomes by taking areas of the probability distribution. For example, what's the probability that our die roll is less than or equal to 2? To figure this out, we'll take the area of each bar representing an outcome of 2 or less.

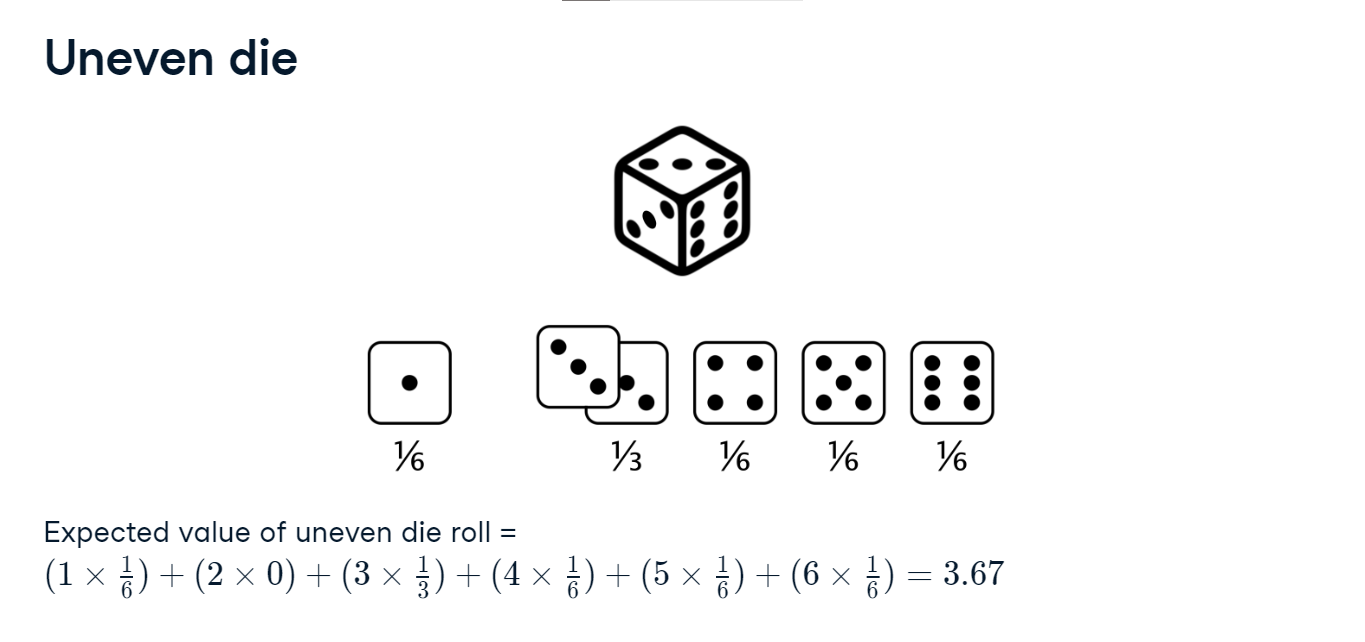
**Probability = area**

Each bar has a width of 1 and a height of one sixth, so the area of each bar is one sixth. We'll sum the areas for 1 and 2, to get a total probability of one third.



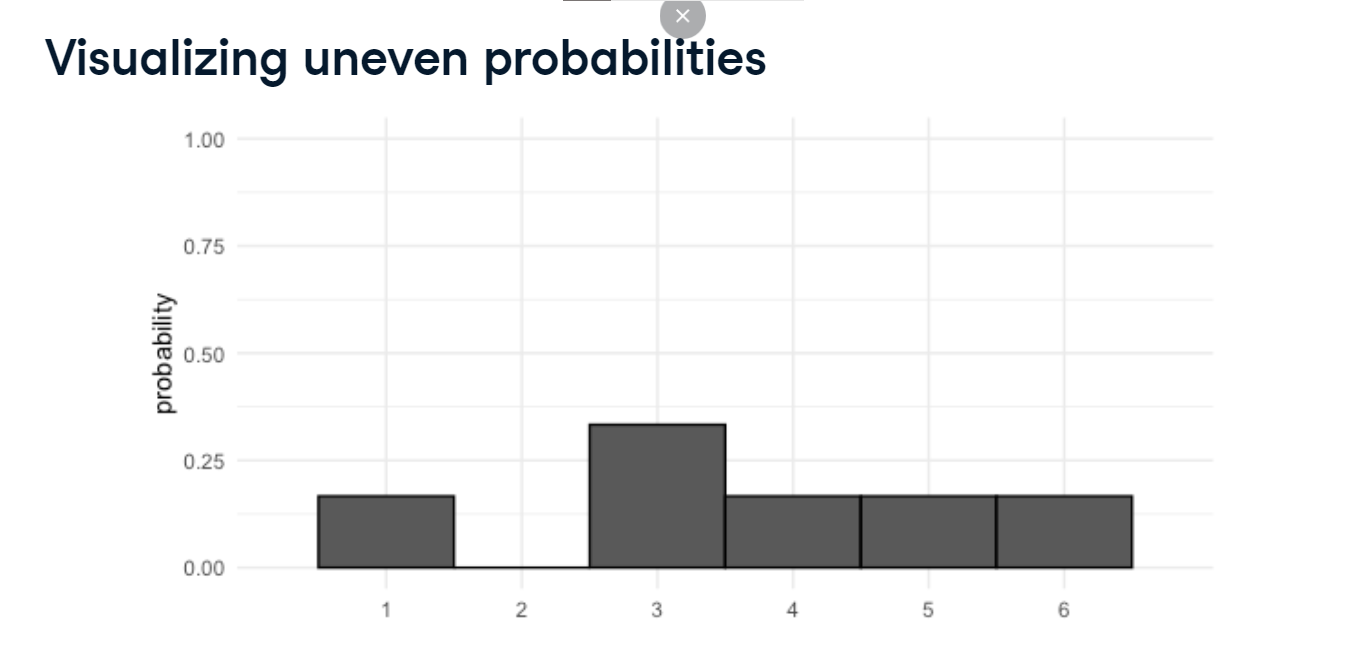
**Uneven die**

Now let's say we have a die where the two got turned into a three. This means that we now have a 0% chance of getting a 2, and a 33% chance of getting a 3. To calculate the expected value of this die, we now multiply 2 by 0, since it's impossible to get a 2, and 3 by its new probability, one third. This gives us an expected value that's slightly higher than the fair die.



**Visualizing uneven probabilities**

When we visualize these new probabilities, the bars are no longer even.

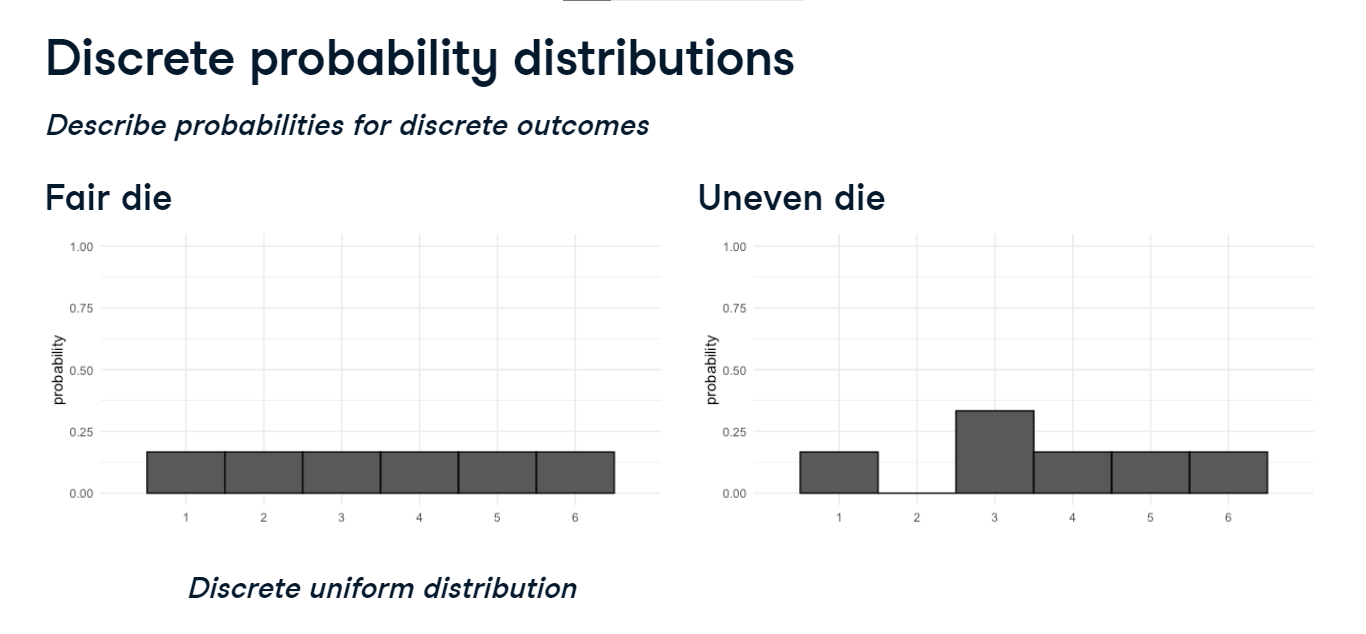


**Adding areas**

With this die, what's the probability of getting something less than or equal to 2? There's a one sixth probability of getting 1, and zero probability of getting 2, which sums to one sixth.

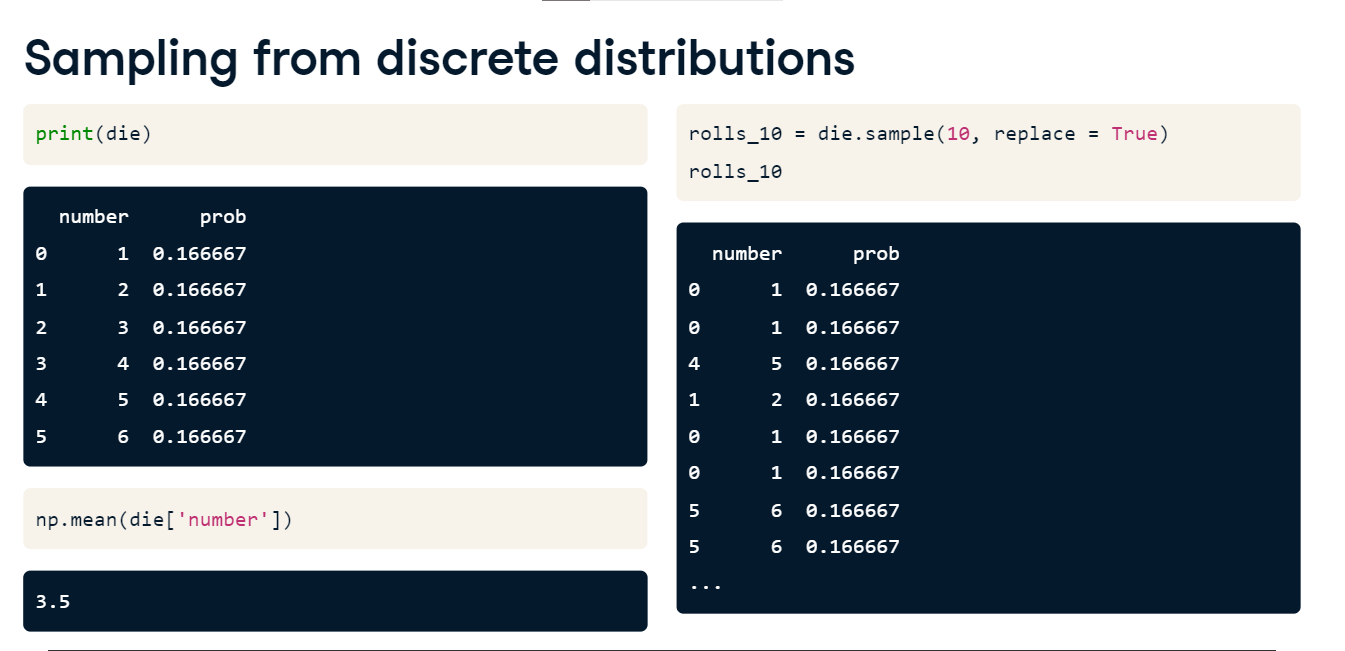
**Discrete probability distributions**

The probability distributions you've seen so far are both discrete probability distributions, since they represent situations with discrete outcomes. Recall from chapter 1 that discrete variables can be thought of as counted variables. In the case of a die, we're counting dots, so we can't roll a 1-point-5 or 4-point-3. When all outcomes have the same probability, like a fair die, this is a special distribution called a discrete uniform distribution.



**Sampling from discrete distributions**

Just like we sampled names from a box, we can do the same thing with probability distributions like the ones we've seen. Here's a DataFrame called die that represents a fair die, and its expected value is 3-point-5. We'll sample from it 10 times to simulate 10 rolls. Notice that we sample with replacement so that we're sampling from the same distribution every time.

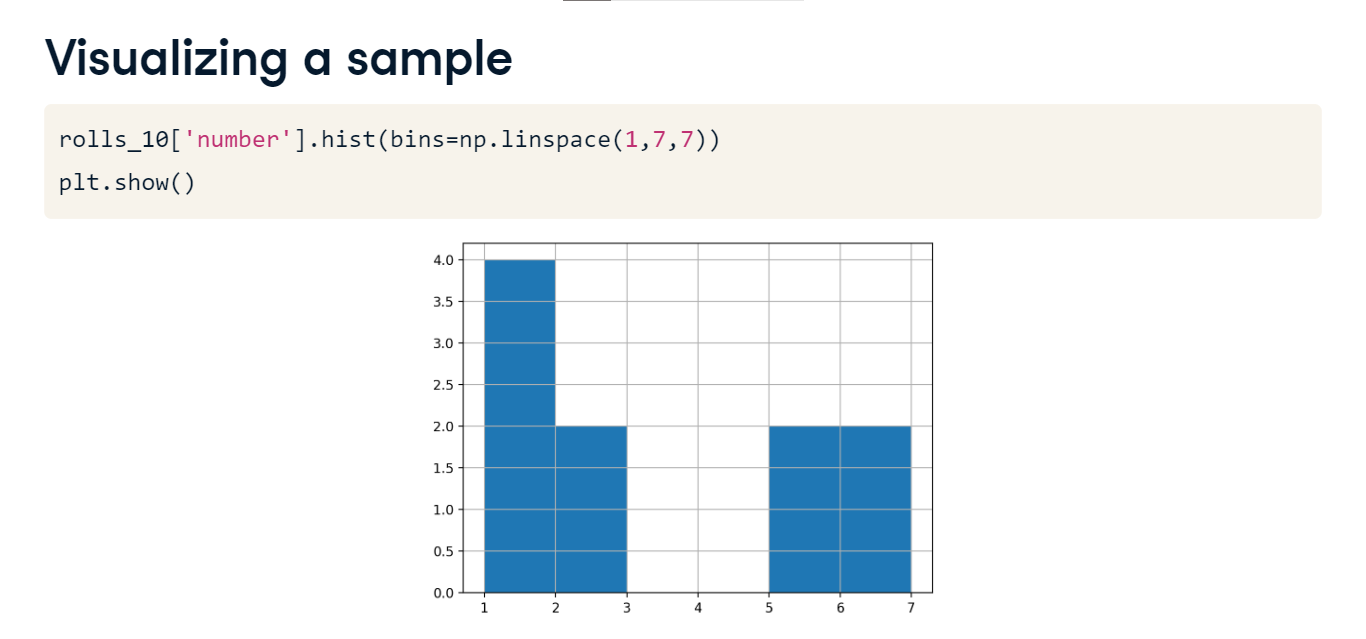


**Visualizing a sample**

We can visualize the outcomes of the ten rolls using a histogram, defining the bins we want using np-dot-linspace.

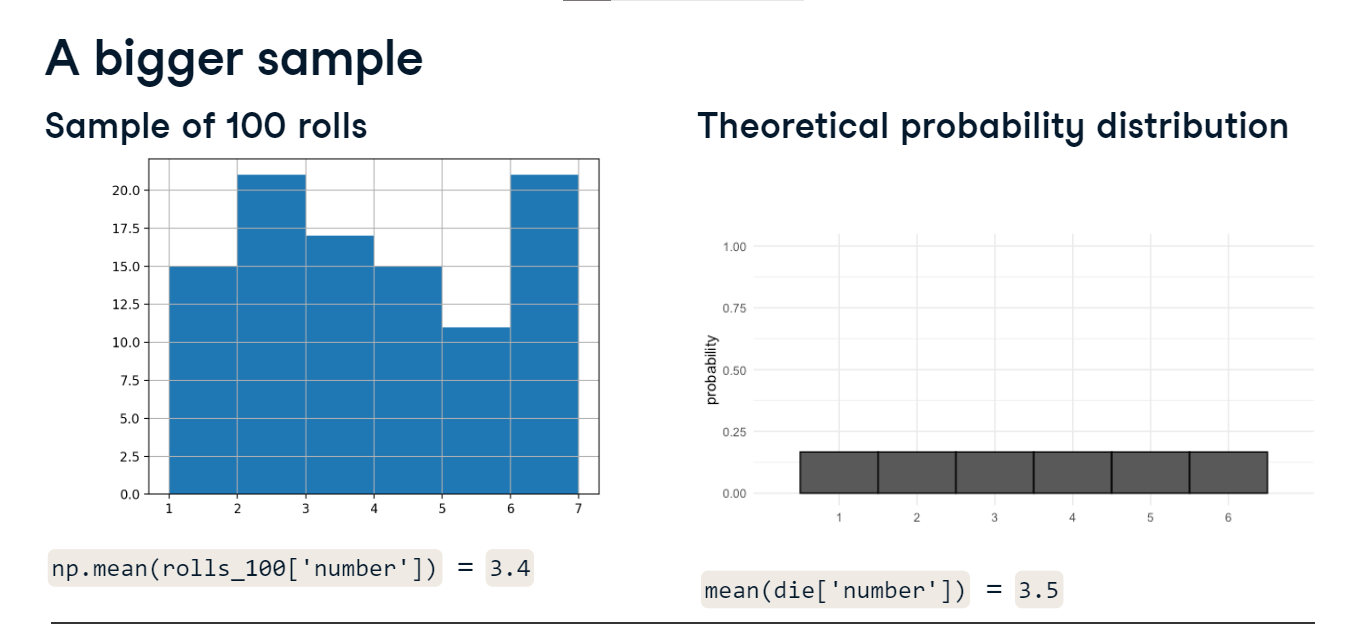
**Sample distribution vs. theoretical distribution**

Notice that we have different numbers of 1's, 2's, 3's, and so on since the sample was random, even though on each roll we had the same probability of rolling each number. The mean of our sample is 3-point-0, which isn't super close to the 3-point-5 we were expecting.



**A bigger sample**

If we roll the die 100 times, the distribution of the rolls looks a bit more even, and the mean is closer to 3-point-5.



**An even bigger sample**

If we roll 1000 times, it looks even more like the theoretical probability distribution and the mean closely matches 3-point-5.

**Law of large numbers**

This is called the law of large numbers, which is the idea that as the size of your sample increases, the sample mean will approach the theoretical mean.

**Continuous distributions**

We can use discrete distributions to model situations that involve discrete or countable variables, but how can we model continuous variables?

**Waiting for the bus**

Let's start with an example. The city bus arrives once every twelve minutes, so if you show up at a random time, you could wait anywhere from 0 minutes if you just arrive as the bus pulls in, up to 12 minutes if you arrive just as the bus leaves.

**Continuous uniform distribution**

Let's model this scenario with a probability distribution. There are an infinite number of minutes we could wait since we could wait 1 minute, 1-point-5 minutes, 1-point-53 minutes, and so on, so we can't create individual blocks like we could with a discrete variable.

**Continuous uniform distribution**

Instead, we'll use a continuous line to represent probability. The line is flat since there's the same probability of waiting any time from 0 to 12 minutes. This is called the continuous uniform distribution.

**Probability still = area**

Now that we have our distribution, let's figure out what the probability is that we'll wait between 4 and 7 minutes. Just like with discrete distributions, we can take the area from 4 to 7 to calculate probability.

**Probability still = area**

The width of this rectangle is 7 minus 4 which is 3. The height is one twelfth.

Multiplying those together to get area, we get 3/12 or 25%.

**Uniform distribution in Python**

Let's use the uniform distribution in Python to calculate the probability of waiting 7 minutes or less. We need to import uniform from scipy-dot-stats. We can call uniform-dot-cdf and pass it 7, followed by the lower and upper limits, which in our case is 0 and 12. The probability of waiting less than 7 minutes is about 58%.

**"Greater than" probabilities**

If we want the probability of waiting more than 7 minutes, we need to take 1 minus the probability of waiting less than 7 minutes.

**Combining multiple uniform.cdf() calls**

How do we calculate the probability of waiting 4 to 7 minutes using Python?

**Combining multiple uniform.cdf() calls**

We can start with the probability of waiting less than 7 minutes, then subtract the probability of waiting less than 4 minutes. This gives us 25%.

**Total area = 1**

To calculate the probability of waiting between 0 and 12 minutes, we multiply 12 by 1/12, which is 1, or 100%. This makes sense since we're certain we'll wait anywhere from 0 to 12 minutes.

**Generating random numbers according to uniform distribution**

To generate random numbers according to the uniform distribution, we can use uniform-dot-rvs, which takes in the minimum value, maximum value, followed by the number of random values we want to generate. Here, we generate 10 random values between 0 and 5.

**Other continuous distributions**

Continuous distributions can take forms other than uniform where some values have a higher probability than others.

No matter the shape of the distribution, the area beneath it must always equal 1.

**Other special types of distributions**

This will also be true of other distributions you'll learn about later on in the course, like the normal distribution or exponential distribution, which can be used to model many real-life situations.