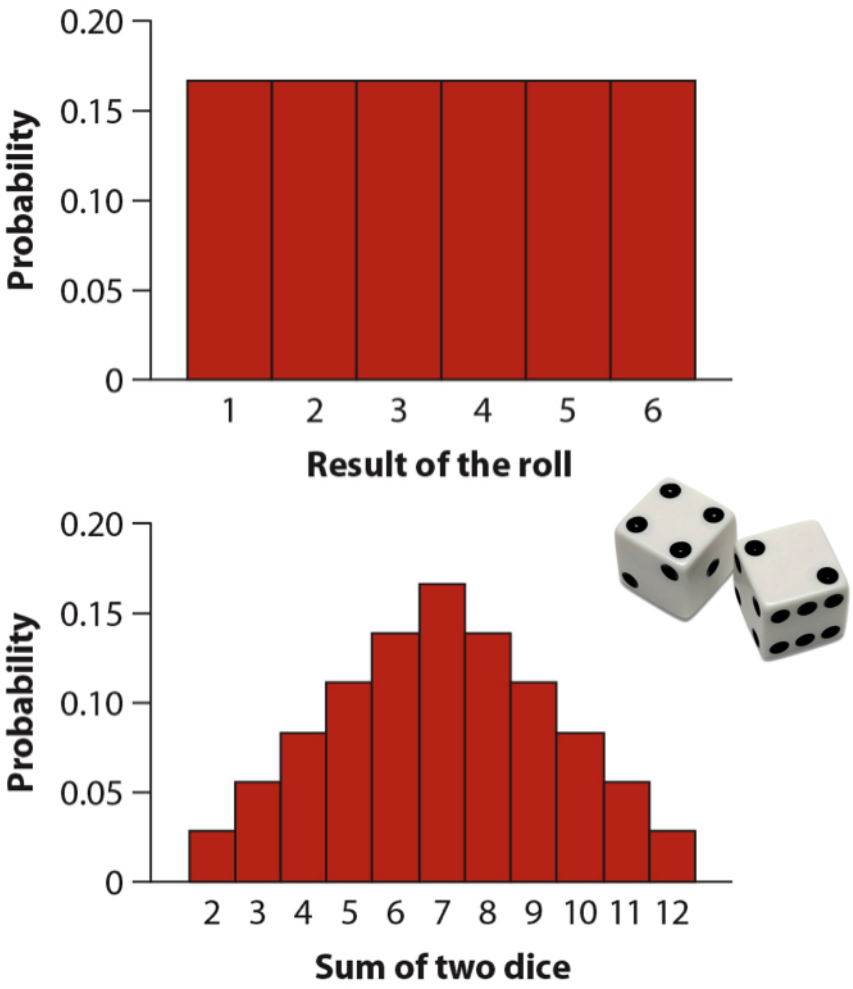
R Lab #5d - Dice probability

Contents

Probability distributions for different outcomes for rolling one or two die

The following picture from the book (Fig 5.4-2) shows the probability distributions for mutually exclusive outcomes when rolling one or two die.



Let's try to mimic this in R...

Rolling a die

If we wanted to roll one die just one time then that would be equivalent to asking for a random integer between 1 and 6. This can be done with sample(). Here are three simulated rolls:

```
sample(1:6, 1)
## [1] 2
sample(1:6, 1)
## [1] 2
sample(1:6, 1)
## [1] 3
```

But to estimate the probability distribution we need to do many trials and calculate the probability of each outcome. So instead of one roll let's do 100:

```
sample(1:6, 100)
## Error in sample.int(length(x), size, replace, prob): cannot take a sample larger than the population
```

Notice that we get an error. This is because by default sample() is run without replacement, so after six rolls we'd exhaust the possible outcomes. Sampling with replacement can be turned on with the argument replace=T. Replacement means that after something is sampled it is "replaced" back into the set

of numbers/objects and can be drawn in a future trail.

```
sample(1:6, 100, replace = T)
##   [1] 4 3 1 3 4 1 2 3 6 6 1 1 2 3 3 2 1 4 5 2 2 6 6 6 1 5 5 5 5 6 3 4 4 4 5 3 5
##  [38] 6 4 2 3 4 4 6 4 2 2 4 2 2 5 5 2 3 2 5 6 3 3 6 5 1 3 4 5 5 5 3 3 2 4 2 2 4
##  [75] 5 5 1 6 3 6 4 2 3 1 4 1 3 3 5 1 1 6 2 5 3 3 3 5 2 5
```

Now we have 100 rolls, but instead let's store them as variable "x":

```
x <- sample(1:6, 100, replace = T)
```

Graphing the die rolls

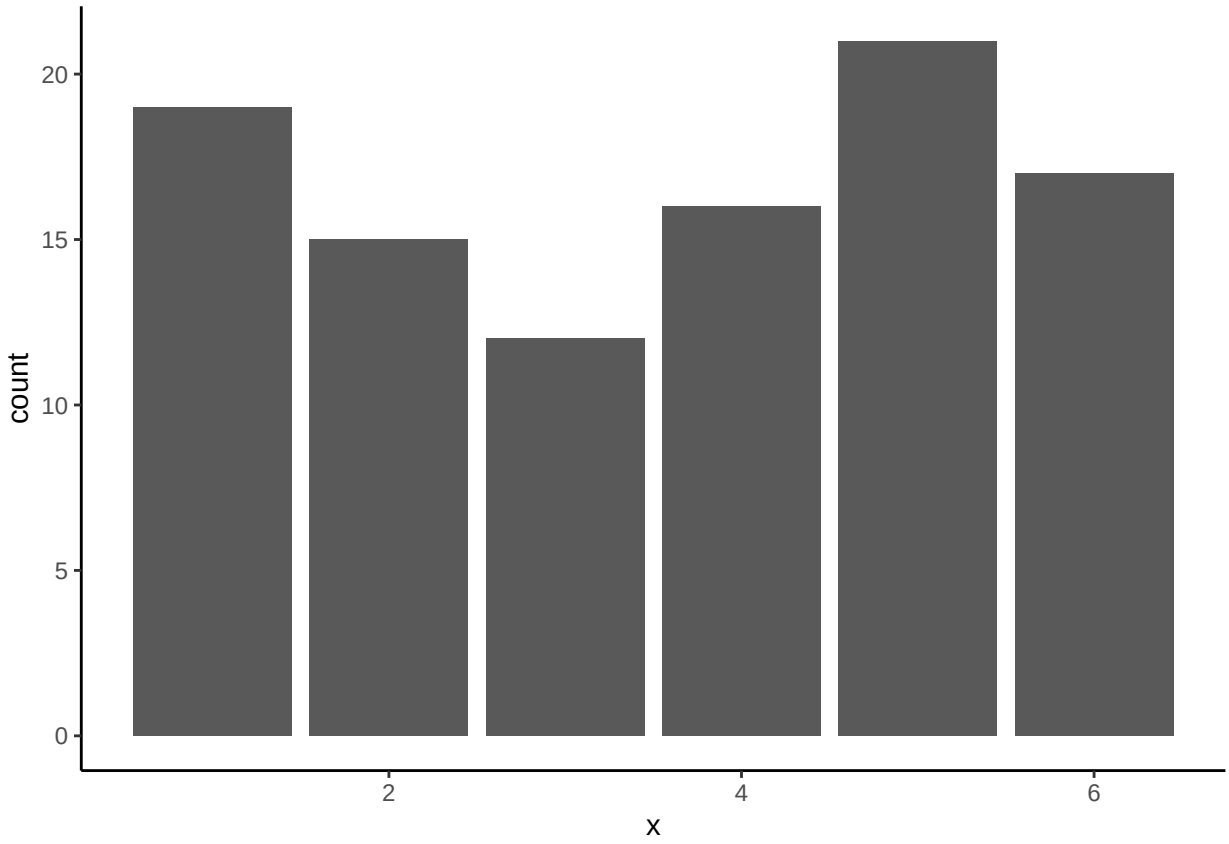
First, note that your sample of 100 rolls will be different then shown here because sample() produces a random sample.

Now we can graph x. It is tempting to try a histogram, but that is for a continuous numerical variable. A die is actually an ordinal categorical variable, so a bar plot is correct graph to make. Let's try a similar syntax that we've been using:

```
library(ggplot2)
ggplot(x) + geom_bar() + theme_classic()
## Error in `fortify()`:
## ! `data` must be a <data.frame>, or an object coercible by `fortify()`,
##   not an integer vector.
```

See that this gives an error because the first argument of the ggplot() command is supposed to be a data frame. Here we don't have a data frame because "x" is a vector of values. Instead we need a slightly different syntax where we skip the first argument and skip directly to the "mapping" argument that contains aes:

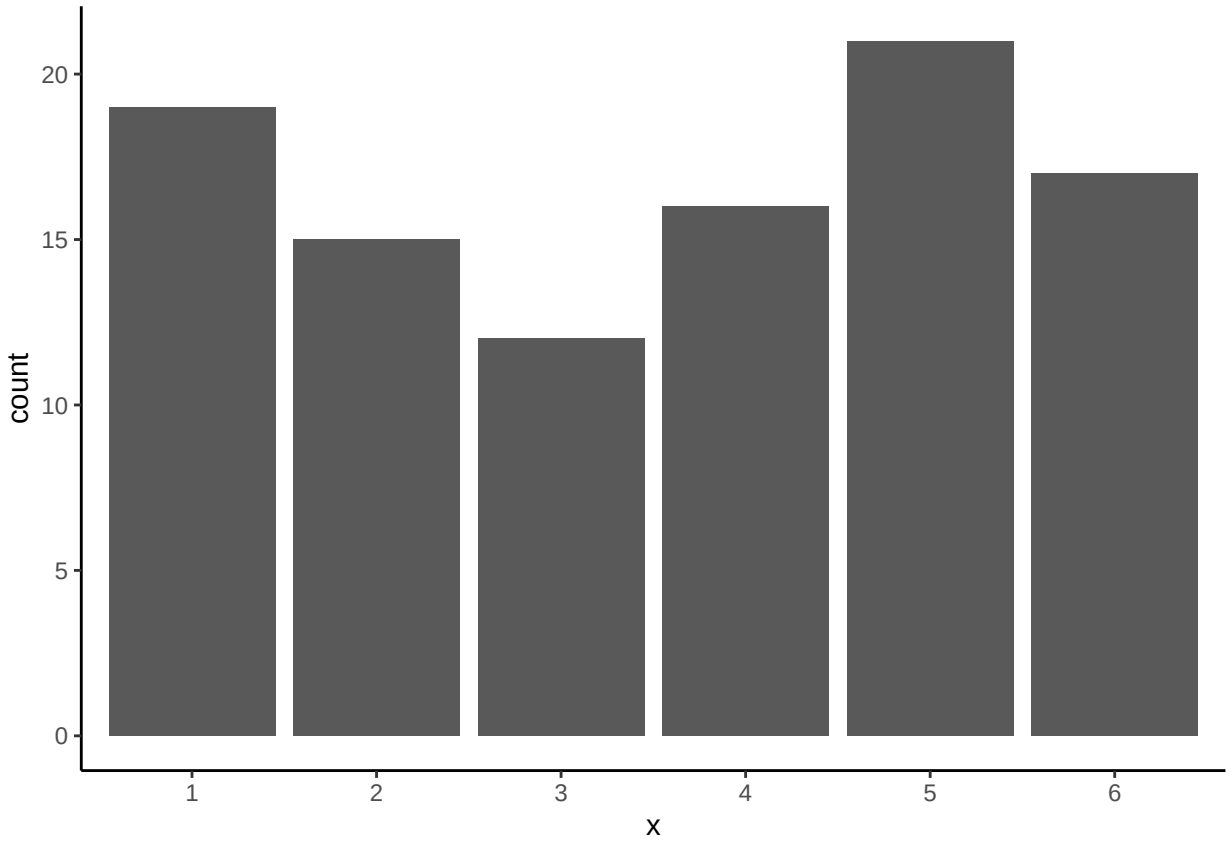
```
ggplot(mapping = aes(x)) + geom_bar() + theme_classic()
```



This should produce a bar graph with a count of each of the six sides in the 100 rolls. Note that you're graph will look different from mine because we did different random samples.

However, the x-axis looks a little "off" in that only the ticks at 2, 4, and 6 are labeled. We can fix this by adding the command scale_x_discrete() along with the argument limits to the line. Note that I put the side number in quotes because I want R to treat them as text rather than numbers.

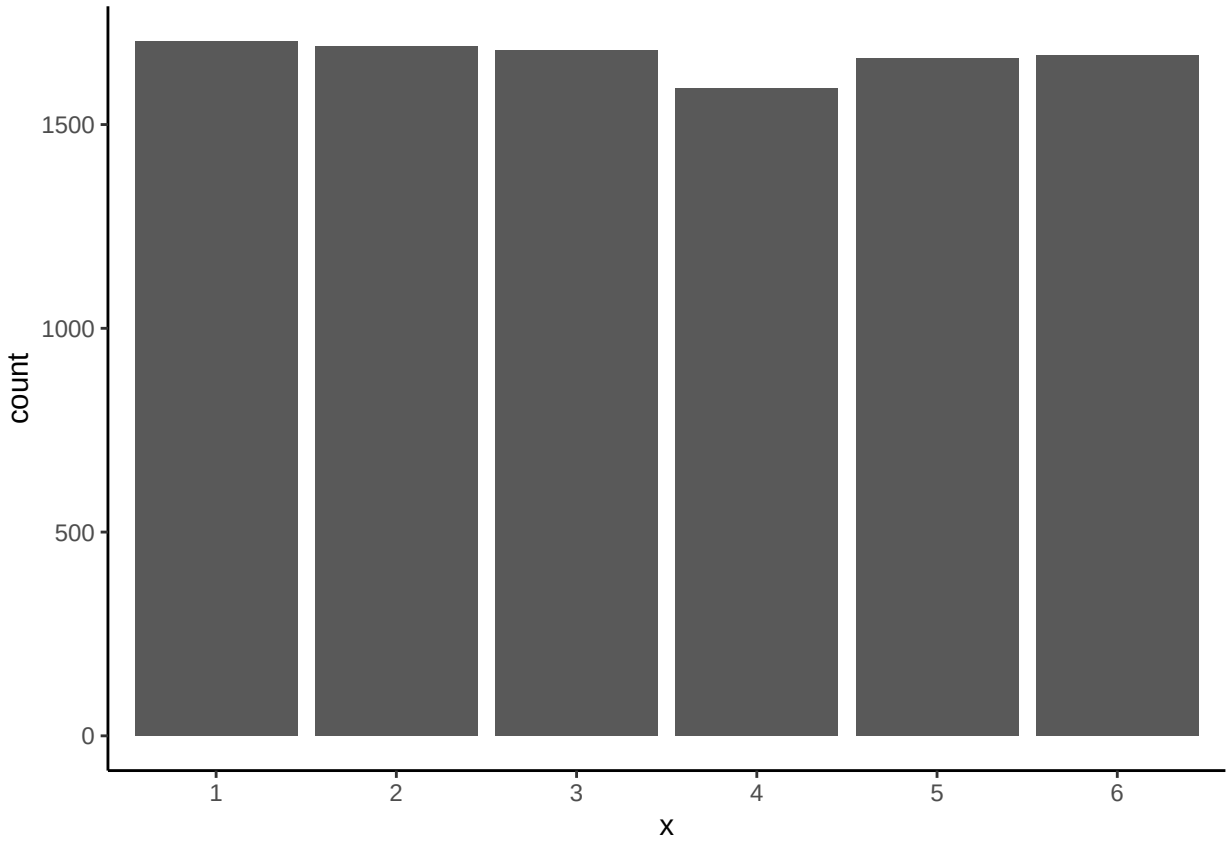
```
ggplot(mapping = aes(x)) + geom_bar() + theme_classic() + scale_x_discrete(limits = c("1",
    "2", "3", "4", "5", "6"))
```



Increasing the number of rolls

Notice that we do not get a nice flat uniform distribution like in the textbook. This is because we only have 100 rolls and there is sampling error. Increasing to 10,000 rolls will flatten out the probability distribution:

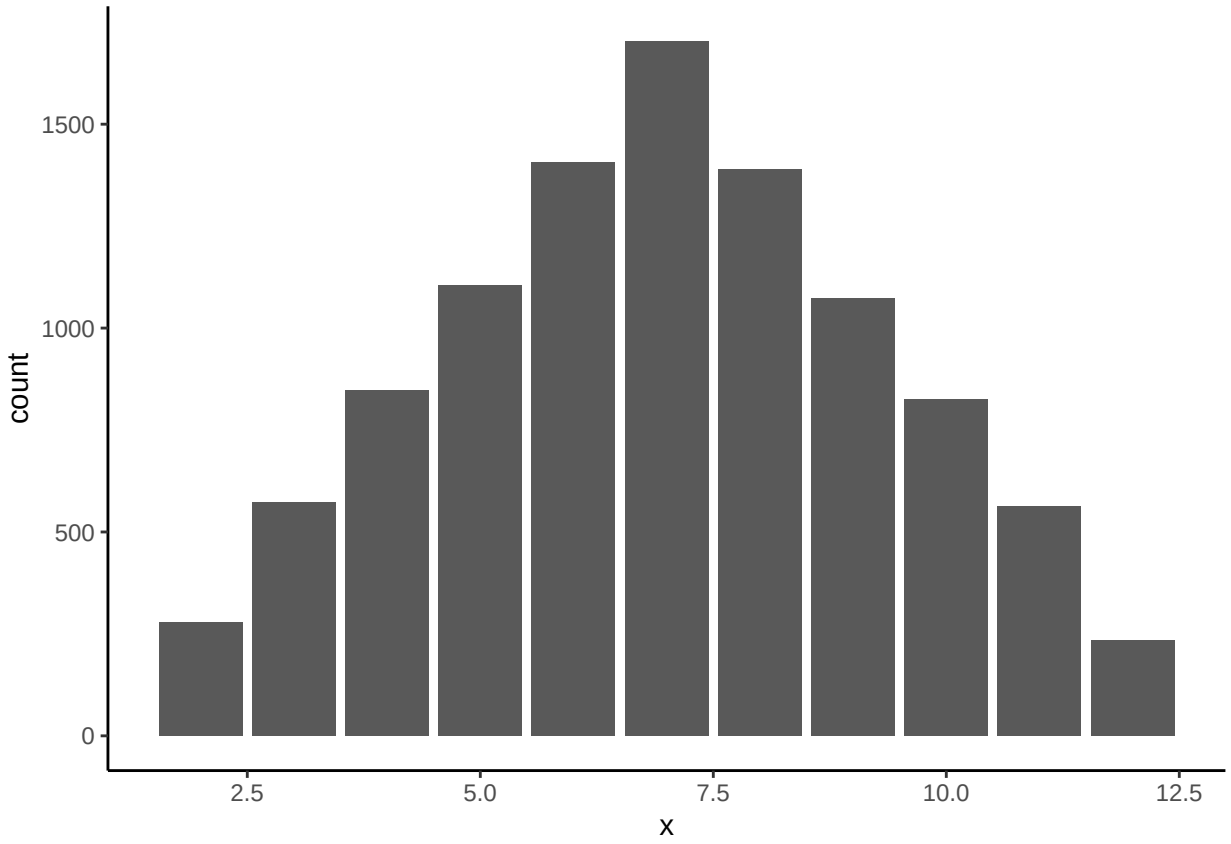
```
x <- sample(1:6, 10000, replace = T)
ggplot(mapping = aes(x)) + geom_bar() + theme_classic() + scale_x_discrete(limits = c("1",
    "2", "3", "4", "5", "6"))
```



Rolling two die

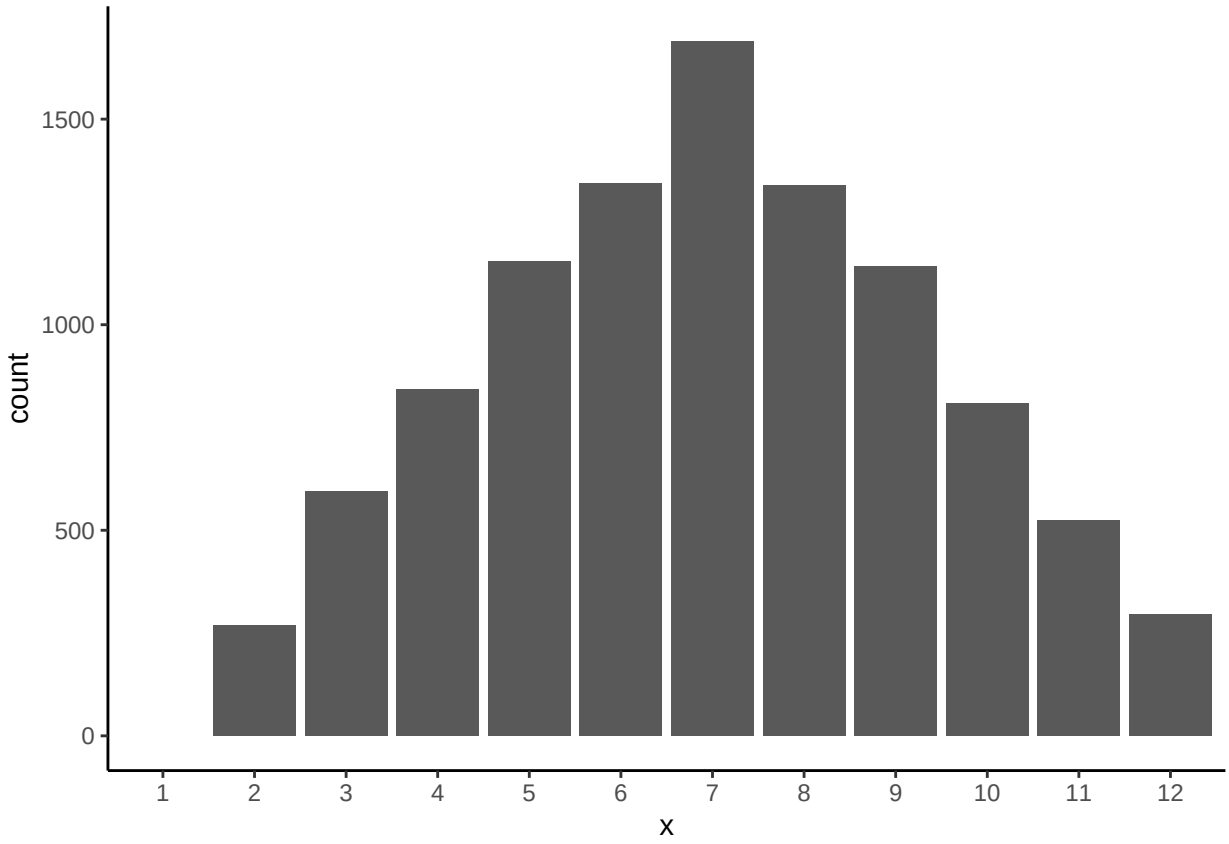
The second graph from the textbook shows the probability for the different possible sums from rolling two die. All we need to do is sum two rolls when creating the sample:

```
x <- sample(1:6, 10000, replace = T) + sample(1:6, 10000, replace = T)
ggplot(mapping = aes(x)) + geom_bar() + theme_classic()
```



Here the axis looks a bit weird because the tick labels are at 2.5, 5.0, 7.5, etc., so R is treating it like a continuous variable. This can again be fixed with the command scale_x_discrete():

```
x <- sample(1:6, 10000, replace = T) + sample(1:6, 10000, replace = T)
ggplot(mapping = aes(x)) + geom_bar() + theme_classic() + scale_x_discrete(limits = c("1",
    "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12"))
```


1500
1000
500
0
count
1
2
3
4
5
6
7
8
9
10
11
12
x